Approximate Turing Kernelization and Lower Bounds for Domination Problems

Stefan Kratsch
Algorithm Engineering, Humboldt-Universität zu Berlin, Germany

Pascal Kunz
Algorithm Engineering, Humboldt-Universität zu Berlin, Germany

Abstract

An α-approximate polynomial Turing kernelization is a polynomial-time algorithm that computes an (αc)-approximate solution for a parameterized optimization problem when given access to an oracle that can compute c-approximate solutions to instances with size bounded by a polynomial in the parameter. Hols et al. [ESA 2020] showed that a wide array of graph problems admit a $(1+\varepsilon)$-approximate polynomial Turing kernelization when parameterized by the treewidth of the graph and left open whether Dominating Set also admits such a kernelization.

We show that Dominating Set and several related problems parameterized by treewidth do not admit constant-factor approximate polynomial Turing kernelizations, even with respect to the much larger parameter vertex cover number, under certain reasonable complexity assumptions. On the positive side, we show that all of them do have a $(1+\varepsilon)$-approximate polynomial Turing kernelization for every $\varepsilon > 0$ for the joint parameterization by treewidth and maximum degree, a parameter which generalizes cutwidth, for example.

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

Keywords and phrases Approximate Turing kernelization, approximation lower bounds, exponential-time hypothesis, dominating set, capacitated dominating, connected dominating set, independent dominating set, treewidth, vertex cover number

Digital Object Identifier 10.4230/LIPIcs.IPEC.2023.32

Funding *Pascal Kunz*: Supported by the DFG Research Training Group 2434 "Facets of Complexity".

1 Introduction

The gold standard in kernelization is a polynomial (exact) kernelization, i.e. a compression of input instances to a parameterized problem to a size that is polynomial in the parameter such that an exact solution for the original instance can be recovered from the compressed instance. Several weaker notions of kernelization have been developed for problems that do not admit polynomial kernelizations. Turing kernelization [1, 11] does away with the restriction that the solution must be recovered from a single compressed instance and instead allow several small instances to be created and the solution to be extracted from solutions to all of these instances. Lossy kernelizations [21], in turn, do away with the requirement that the solution that can be recovered from the compressed instance be an optimum solution, allowing the solution to the original instance to be worse than optimal by a constant factor. Hols et al. [12] introduced lossy Turing kernelizations, which allow both multiple compressed instances and approximate solutions, and showed that several graph problems parameterized by treewidth admit $(1+\varepsilon)$-approximate Turing kernelizations for every $\varepsilon > 0$. They left as an open question whether or not the problem Dominating Set parameterized by treewidth also admits a constant-factor approximate Turing kernelization.

Our contribution. We answer this question in the negative and show that a $\mathcal{O}(2^{\log^c \mathrm{vc}})$-approximate polynomial Turing kernelization for Dominating Set[vc][1], where vc refers to the vertex cover number, would contradict the Exponential Time Hypothesis. We prove analogous lower bounds for Capacitated Dominating Set[vc], Connected Dominating Set[vc] for Hitting Set[$|U|$], where U is the universe, and for Node Steiner Tree[$|V \setminus T|$] where $V \setminus T$ is the set of non-terminal vertices. Of course, the lower bounds for the vertex cover number also imply lower bounds for the smaller parameter treewidth. Using a second approach for obtaining lower bounds for approximate Turing kernelizations, essentially a gap-introducing polynomial parameter transformation (PPT), we show that Independent Dominating Set[vc] does not have an α-approximate polynomial Turing kernelization for any constant α, unless every problem in the complexity class MK[2] has a polynomial (exact) Turing kernelization, which would contradict a conjecture by Hermelin et al. [11]. We then show that for the joint parameterization by treewidth and the maximum degree, each of the aforementioned domination problems does have a $(1+\varepsilon)$-approximate polynomial Turing kernelization for every $\varepsilon > 0$. This generalizes, for instance, parameterization for cutwidth or bandwidth.

Related work. For an introduction to kernelization, including brief overviews on lossy and Turing kernelizations, we refer to the standard textbook [7]. Binkele-Raible et al. [1] introduced the first Turing kernelization for a problem that does not admit a polynomial kernelization. Since then numerous Turing kernelizations have been published for such problems. Hermelin et al. [11] introduced a framework, which we will make use of in Section 3.2, for ruling out (exact) polynomial Turing kernelizations. Fellows et al. [6] were the first to combine the fields of kernelization and approximation. A later study by Lokshtanov et al. [21] introduced the framework of lossy kernelization that has become more established. Finally, Hols et al. [12] gave the first approximate Turing kernelizations.

Approximation algorithms and lower bounds for domination problems have received considerable attention. They are closely related to the problems Hitting Set and Set Cover. A classical result by Chvátal [3] implies a polynomial-time $\mathcal{O}(\log n)$-factor approximation for Dominating Set. There are also $\mathcal{O}(\log n)$-factor approximations for Connected Dominating Set [8] and Capacitated Dominating Set [23], but Independent Dominating Set does not have a $\mathcal{O}(n^{1-\varepsilon})$-approximation for any $\varepsilon > 0$, unless P = NP [9]. Chlebík and Chlebíková [2] showed that Dominating Set, Connected Dominating Set, and Capacitated Dominating Set do not have constant-factor approximations even on graphs with maximum degree bounded by a constant Δ and that Independent Dominating Set does not have better than a Δ-factor approximation.

2 Preliminaries

Graphs. We use standard graph terminology and all graphs are undirected, simple, and finite. For a graph $G = (V, E)$ and $X \subseteq V$, we use $N[X] \coloneqq X \cup \{v \in V \mid \exists u \in X \colon \{u, v\} \in E\}$ to denote the *closed neighborhood* of X and, if $v \in V$, then we let $N[v] \coloneqq N[\{v\}]$. We will use Δ and vc to refer to the maximum degree and vertex cover number of a graph, respectively.

Let $G = (V, E)$ be a graph $X \subseteq V$ a vertex set. The set $X \subseteq V$ is a *dominating set* if $N[X] = V$. It is an *independent set* if there is no edge $\{u, v\} \in E$ with $u, v \in X$. It is an *independent dominating set* if it is both an independent and a dominating set. It is a *connected*

[1] We use FOO[X] to refer to the problem FOO parameterized by X.

dominating set if it is a dominating set and the graph $G[X]$ is connected. A *capacitated graph* $G = (V, E, \text{cap})$ consists of a graph (V, E) and a *capacity function* $\text{cap}\colon V \to \mathbb{N}$. A *capacitated dominating set* in a capacitated graph $G = (V, E, \text{cap})$ is a pair (X, f) where $X \subseteq V$ and $f\colon V \setminus X \to X$ such that (i) v and $f(v)$ are adjacent for all $v \in V \setminus X$ and (ii) $|f^{-1}(v)| \leq \text{cap}(v)$ for all $v \in X$. The size of (X, f) is $|X|$.

Let $G = (V, E)$ be a graph. A *tree decomposition* of G is a pair $\mathcal{T} = (T = (W, F), \{X_t\}_{t \in W})$ where T is a tree, $X_t \subseteq V$ for all $t \in W$, $\bigcup_{t \in W} X_t = V$, for each $e \in E$ there is a $t \in W$ such that $e \subseteq X_t$, and for each $v \in V$ the node set $\{t \in W \mid v \in X_t\}$ induces a connected subgraph of T. The *width* of $\mathcal{T}$ is $\max_{t \in W} |X_t| - 1$. The *treewidth* $\text{tw}(G)$ of G is the minimum width of any tree decomposition of G. A *rooted tree decomposition* consists of a tree decomposition $\mathcal{T} = (T = (W, F), \{X_t\}_{t \in W})$ along with a designated root $r \in W$. Given this rooted decomposition and a node $t \in W$, we will use $V_t \subseteq V$ to denote the set of vertices v such that $v \in X_{t'}$ and t' is a descendant (possibly t itself) of t in the rooted tree (T, r). The rooted tree decomposition is *nice* if $X_r = \emptyset$ and $X_t = \emptyset$ for every leaf of T and every other node t of T is of one of three types: (i) a *forget node*, in which case t has a single child t' and there is a vertex $v \in V$ such that $X_{t'} = X_t \cup \{v\}$, (ii) an *introduce node*, in which case t has a single child t' and there is a vertex $v \in V$ such that $X_t = X_{t'} \cup \{v\}$, or (iii) a *join node*, in which case t has exactly two children t_1 and t_2 and $X_t = X_{t_1} = X_{t_2}$.

Turing kernelization. A *parameterized decision problem* is a set $L \subseteq \Sigma^* \times \mathbb{N}$. A *Turing kernelization* of size $f\colon \mathbb{N} \to \mathbb{N}$ for a parameterized decision problem L is a polynomial-time algorithm that receives as input an instance $(x, k) \in \Sigma^* \times \mathbb{N}$ and access to an oracle that, for any instance $(x', k') \in \Sigma^* \times \mathbb{N}$ with $|x'| + k' \leq f(k)$, outputs whether $(x', k') \in L$ in a single step, and decides whether $(x, k) \in L$. It is a *polynomial Turing kernel* if f is polynomially bounded.

A *polynomial parameter transformation* (PPT) from one parameterized decision problem L to a second such problem L' is a polynomial-time computable function $f\colon \Sigma^* \times \mathbb{N} \to \Sigma^* \times \mathbb{N}$ such that $(x, k) \in L$ if and only $(x', k') \in L'$ and there is a polynomially bounded function p such that $k' \leq p(k)$ for all $(x, k), (x', k') \in \Sigma^* \times \mathbb{N}$ with $f(x, k) = (x', k')$.

A *parameterized minimization problem* is defined by a computable function $\mathcal{P}\colon \Sigma^* \times \mathbb{N} \times \Sigma^* \to \mathbb{R} \cup \{\infty, -\infty\}$. The optimum value for an instance $(I, k) \in \Sigma^* \times \mathbb{N}$ is $\text{OPT}_{\mathcal{P}}(I, k) := \min_{x \in \Sigma^*} \mathcal{P}(I, k, x)$. We will say that a solution $x \in \Sigma^*$ is *α-approximate* if $\mathcal{P}(I, k, x) \leq \alpha \cdot \text{OPT}_{\mathcal{P}}(I, k)$. In order to simplify notation, we will allow ourselves to write $\mathcal{P}(I, x)$ instead of $\mathcal{P}(I, k, x)$ and $\text{OPT}_{\mathcal{P}}(I)$ instead of $\text{OPT}_{\mathcal{P}}(I, k)$ if those values do not depend on k. The problems (Capacitated/Connected/Independent) Dominating Set are defined by $\mathcal{P}(G, X) := |X|$ if X is a (capacitated/connected/independent) dominating set in X and $\mathcal{P}(G, X) := \infty$, otherwise. The problem Node Steiner Tree is defined by $\text{NST}((G, T), X) := |X|$ if $G[X \cup T]$ is connected and $\text{NST}((G, T), X) := \infty$, otherwise. Hitting Set is a problem whose input $(U, \mathcal{S})$ consists of a set U and a family $\mathcal{S} \subseteq 2^U$ of nonempty sets, a solution X is a subset of U, and $\text{HS}(X) := \infty$ if there is an $S \in \mathcal{S}$ such that $X \cap S = \emptyset$ and $\text{HS}(X) := |X|$, otherwise.

Let $\alpha \in \mathbb{R}$ with $\alpha \geq 1$ and let $\mathcal{P}$ be a parameterized minimization problem. An *α-approximate Turing kernelization* of size $f\colon \mathbb{N} \to \mathbb{N}$ for $\mathcal{P}$ is a polynomial-time algorithm that given an instance (I, k) computes a $(c\alpha)$-approximate solution when given access to an oracle for P which outputs a c-approximate solution to any instance (I', k') with $|I'| + k' \leq f(k)$ in a single step. It is an *α-approximate polynomial Turing kernelization* if f is polynomially bounded. Note that the algorithm is not given access to c, the approximation factor of the oracle, and is not allowed to depend on c. In practice, it can also be helpful

for the approximate Turing kernelization algorithm to receive a witness for the parameter value k as input. Similarly to Hols et al. [12], we will assume that our approximate Turing kernelizations for the parameterization treewidth plus maximum degree are given as input a graph G and a nice tree decomposition of width $\mathrm{tw}(G)$. Alternatively, one could also use the polynomial-time algorithm due to Feige et al. [5] to compute a tree decomposition of width $\mathcal{O}(\sqrt{\log \mathrm{tw}(G)} \cdot \mathrm{tw}(G))$ and then use this tree decomposition. We will also assume that the given tree decomposition is nice, which is not really a restriction, because there is a polynomial-time algorithm that converts any tree decomposition into a nice tree decomposition without changing the width [18].

3 Lower bounds

3.1 Exponential-time hypothesis

In the following, we will show that several problems do not have an approximate Turing kernelization assuming the exponential-time hypothesis (ETH). The proof builds on a proof due to Lokshtanov et al. [20, Theorem 12] showing that Hitting Set parameterized by the size of the universe does not admit a lossy (Karp) kernelization unless the ETH fails.

Let 3-CNF-SAT denote the satisfiability problem for Boolean formulas in conjunctive normal form with at most three literals in each clause. The *exponential-time hypothesis* (ETH) [13] states that there is there is a fixed $c > 0$ such that 3-CNF-SAT is not solvable in time $2^{cn} \cdot (n+m)^{\mathcal{O}(1)}$, where n and m are the numbers of variables and clauses, respectively.

Let $\mathcal{P}$ and $\mathcal{P}'$ be parameterized minimization problems and $f\colon \Sigma^* \times \mathbb{N} \to \mathbb{R}_+$ a real-valued function that takes instances of $\mathcal{P}$ as input. An *f-approximation-preserving polynomial parameter transformation* (f-APPT) from $\mathcal{P}$ to $\mathcal{P}'$ consists of two algorithms:

- a polynomial-time algorithm $\mathcal{A}$ (the *reduction algorithm*) that receives as input an instance (I,k) for $\mathcal{P}$ and outputs an instance (I',k') for $\mathcal{P}'$ with $k' \leq p(k)$ for some polynomially bounded function p and
- a polynomial-time algorithm $\mathcal{B}$ (the *lifting algorithm*) that receives as input the instances (I,k), (I',k'), where the latter is the output of $\mathcal{A}$ when given the former, as well as a solution x for (I',k') and outputs a solution y for (I,k) with

$$\mathcal{P}(I,k,y) \leq \frac{f(I,k) \cdot \mathrm{OPT}_{\mathcal{P}}(I,k) \cdot \mathcal{P}'(I',k,x)}{\mathrm{OPT}_{\mathcal{P}'}(I',k)}.$$

We will use the following lemma, which is a weaker version of a result by Nelson [22].

▶ **Lemma 1** ([22]). *If there are a constant $c < 1$ and a polynomial-time algorithm that computes an $\mathcal{O}(2^{\log^c |U|})$-factor approximation for Hitting Set, then the ETH fails.*

▶ **Lemma 2.** *Let $\mathcal{P}$ be a parameterized minimization problem that satisfies the following two conditions:*

(a) *There is an f-APPT from Hitting Set$[|U|]$ to $\mathcal{P}$ with $f(U,\mathcal{S}) \in \mathcal{O}(2^{\log^{c_1} |U|})$ for some constant $c_1 < 1$.*

(b) *There is a constant $c_2 < 1$ and a polynomial-time algorithm that computes a $\mathcal{O}(2^{\log^{c_2} |I|})$-factor approximation for $\mathcal{P}$ where I is an instance for $\mathcal{P}$.*

Then, there is no $\mathcal{O}(2^{\log^{c_3} k})$-approximate polynomial Turing kernelization for $\mathcal{P}$ for any $c_3 < 1$, unless the ETH fails.

Proof. Suppose that $\mathcal{P}$ satisfies conditions (a) and (b) and admits a $\mathcal{O}(2^{\log^{c_3} k})$-approximate polynomial Turing kernelization of size $\mathcal{O}(k^d)$. Furthermore, assume that $p(n) \leq \mathcal{O}(n^{d'})$ where p is the polynomial parameter bound for the reduction algorithm of the f-APPT. Choose any constant c with $\max\{c_1, c_2, c_3\} < c < 1$ and observe that for any constant α and $i \in \{1, 2, 3\}$ we have that $2^{\alpha \log^{c_i} n} \leq \mathcal{O}(2^{\log^c n})$. We will give a $\mathcal{O}(2^{\log^c |U|})$-approximation algorithm for Hitting Set. By Lemma 1, this proves the claim.

The algorithm proceeds as follows. Given an instance $I = (U, \mathcal{S})$ of Hitting Set as input, it first applies the reduction algorithm of the APPT to obtain an instance (I', k) of $\mathcal{P}$. Then, it runs the given approximate Turing kernelization on (I', k). Whenever this Turing kernelization queries the oracle, this query is answered by running the approximation algorithm given by condition (b). Once the Turing kernelization outputs a solution X, the algorithm calls the lifting algorithm of the APPT on X, $(I, |U|)$, and (I', k). The algorithm outputs the solution Y given by the lifting algorithm.

It remains to show that $|Y| \leq \mathcal{O}(2^{\log^{c_3} |U|} \cdot \mathrm{OPT}_{\mathrm{HS}}(I))$. First, observe that the $\mathcal{O}(2^{\log^{c_2} |I|})$-factor approximation algorithm is only run on instances (J, ℓ) with $|J| \leq \mathcal{O}(k^d)$, so it always outputs a solution Z with $\mathcal{P}(J, \ell, Z) \leq \mathcal{O}(2^{\log^{c_2} k^d} \cdot \mathrm{OPT}_{\mathcal{P}}(J, \ell))$. Hence, in the algorithm described above the Turing kernelization is given a $\mathcal{O}(2^{d \log^{c_2} k})$-approximate oracle, so it follows that $\mathcal{P}(I', k, X) \leq \mathcal{O}(2^{\log^{c_2} k} \cdot 2^{d \log^{c_3} k} \cdot \mathrm{OPT}_{\mathcal{P}}(I', k))$. Since $k \leq \mathcal{O}(|U|^{d'})$, it follows that $\mathcal{P}(I', k, X) \leq \mathcal{O}(2^{\log^{c_3} |U|^{d'}} \cdot 2^{d \log^{c_2} |U|^{d'}} \cdot \mathrm{OPT}_{\mathcal{P}}(I', k))$. Therefore:

$$
\begin{aligned}
|Y| &\leq \frac{f(I,k) \cdot \mathrm{OPT}_{\mathrm{HS}}(I) \cdot \mathcal{P}(I', k, X)}{\mathrm{OPT}_{\mathcal{P}}(I', k)} \leq \mathcal{O}\left(2^{\log^{c_3} |U|^{d'}} \cdot 2^{d \log^{c_2} |U|^{d'}} \cdot f(I,k) \cdot \mathrm{OPT}_{\mathrm{HS}}(I)\right) \\
&\leq \mathcal{O}\left(2^{\log^{c_3} |U|^{d'}} \cdot 2^{d \log^{c_2} |U|^{d'}} \cdot f(I) \cdot \mathrm{OPT}_{\mathrm{HS}}(I)\right) \\
&\leq \mathcal{O}\left(2^{\log^{c_3} |U|^{d'}} \cdot 2^{d \log^{c_2} |U|^{d'}} \cdot 2^{\log^{c_1} |U|} \cdot \mathrm{OPT}_{\mathrm{HS}}(I)\right) \\
&\leq \mathcal{O}(2^{\log^{c} |U|} \cdot \mathrm{OPT}_{\mathrm{HS}}(I)).
\end{aligned}
$$

◀

With Lemma 2, we can prove approximate Turing kernelization lower bounds for several parameterized minimization problems.

▶ **Theorem 3.** *Unless the ETH fails, there are no $\mathcal{O}(2^{\log^c k})$-approximate polynomial Turing kernels, for any $c < 1$ and where k denotes the respective parameter, for the following parameterized minimization problems:*

- **(i)** *Hitting Set*$[|U|]$*,*
- **(ii)** *Dominating Set*$[\mathrm{vc}]$*,*
- **(iii)** *Capacitated Dominating Set*$[\mathrm{vc}]$*,*
- **(iv)** *Connected Dominating Set*$[\mathrm{vc}]$*, and*
- **(v)** *Node Steiner Tree*$[V \setminus T]$*.*

Proof. For each problem, we will prove conditions (a) and (b) from Lemma 2.

- **(i) (a)** Immediate.
  - **(b)** Chvátal [3] gives a $\mathcal{O}(\log|\mathcal{S}|)$-factor approximation algorithm.
- **(ii) (a)** The following folklore reduction is a 1-APPT. Let $(U, \mathcal{S})$ be an instance of hitting set. The algorithm $\mathcal{A}$ creates a graph G as follows. For every $x \in U$ and for every $S \in \mathcal{S}$, G contains vertices v_x and w_S, respectively, and G also contains an additional vertex u. The vertices $\{v_x \mid x \in U\} \cup \{u\}$ form a clique and there is an edge between v_x and w_S if and only if $x \in S$. Observe that the vertices $\{v_x \mid x \in U\}$ form a vertex cover in G, so clearly $\mathrm{vc}(G) \leq |U|$, and that $\mathrm{OPT}_{\mathrm{HS}}(U, \mathcal{S}) = \mathrm{OPT}_{\mathrm{DS}}(G)$. Let X be a dominating set in G. Let X' be obtained from X by removing z and replacing any w_S by an arbitrary v_x with $x \in S$ (such an element u must

exist, as we assume that all $S \in \mathcal{S}$ are non-empty). The algorithm $\mathcal{B}$ outputs $Y \coloneqq \{x \in U \mid v_x \in X'\}$. This set is a hitting set, because for any $S \in \mathcal{S}$ one of the following cases applies: (i) $w_S \notin X$, meaning that X contains a neighbor v_x of w_S. Then, also $x \in X'$ and, hence $x \in Y$ and $x \in S$. (ii) $w_S \in X$, meaning that w_S is replaced by v_x with $x \in S$ when creating X'. Then $x \in Y$ and $x \in S$. Finally, $|Y| = |X| = \frac{\mathrm{OPT}_{\mathrm{HS}}(U,\mathcal{S}) \cdot |X|}{\mathrm{OPT}_{\mathrm{DS}}(G)}$, since $\mathrm{OPT}_{\mathrm{HS}}(U,\mathcal{S}) = \mathrm{OPT}_{\mathrm{DS}}(G)$.

(b) The $\mathcal{O}(\log n)$-factor approximation for Hitting Set [3] can also be used in a straightforward manner to approximate Dominating Set.

(iii) (a) Any instance of Dominating Set can be transformed into an equivalent instance of Capacitated Dominating Set by setting $\mathrm{cap}(v) \coloneqq \deg(v)$ for all vertices v. The claim then follows by the same argument as for Dominating Set.

(b) Wolsey [23] gives a $\mathcal{O}(\log|\mathcal{S}|)$ factor approximation for Capacitated Hitting Set which can be adapted to approximate Capacitated Dominating Set.

(iv) (a) The APPT given in (ii) for Dominating Set also works for Connected Dominating Set, because, in the graphs produced by $\mathcal{A}$, $\mathrm{OPT}_{\mathrm{CON}}(G) = \mathrm{OPT}_{\mathrm{DS}}(G)$ and the solution output by $\mathcal{B}$ is always a clique and, therefore, connected.

(b) Guha and Khuller [8] give a $\mathcal{O}(\log \Delta) \leq \mathcal{O}(\log n)$-factor approximation for Connected Dominating Set.

(v) (a) The following reduction is essentially the same as the one given by Dom et al. [4] Let $(U,\mathcal{S})$ be an instance of Hitting Set. The algorithm $\mathcal{A}$ creates the graph G as in the reduction for Dominating Set in (ii) and sets $T \coloneqq \{w_s \mid s \in \mathcal{S}\} \cup \{u\}$. Clearly, $|V \setminus T| = |U|$ and it easy to show that $\mathrm{OPT}_{\mathrm{HS}}(U,\mathcal{S}) = \mathrm{OPT}_{\mathrm{NST}}(G,T)$. By a similar argument as in (ii), the algorithm $\mathcal{B}$ can output $\{x \in U \mid v_x \in X\}$ where X is a given solution for the Node Steiner Tree instance (G,T).

(b) Klein and Ravi [17] give a $\mathcal{O}(\log n)$-factor approximation for this problem. ◀

If $\mathcal{C}$ is a hereditary class of graphs, then we may define the Restricted $\mathcal{C}$-Deletion problem as follows: We are given a graph $G = (V,E)$ and $X \subseteq V$ such that $G - X \in \mathcal{C}$ and are asked to find a minimum $Y \subseteq X$ such that $G - Y \in \mathcal{C}$. For Restricted Perfect Deletion$[|X|]$, Restricted Weakly Chordal Deletion$[|X|]$, and Restricted Wheel-Free Deletion$[|X|]$, reductions given by Heggernes et al. [10] and Lokshtanov [19] can be shown to be 1-APPTs. However, it is open whether they have a $\mathcal{O}(2^{\log^c |I|})$-factor approximation with $c < 1$, so we cannot rule out an approximate polynomial Turing kernelization. However, we can observe that, under ETH, they cannot have both an approximation algorithm with the aforementioned guarantee *and* an approximate polynomial Turing kernelization.

More generally, we can deduce from the proof of Lemma 2 the following about any parameterized minimization problem $\mathcal{P}$ that only satisfies the first condition in Lemma 2: If we define an *approximate polynomial Turing compression* of a problem $\mathcal{P}$ into a problem $\mathcal{P}'$ to be essentially an approximate polynomial Turing kernelization for $\mathcal{P}$, except that it is given access to an approximate oracle for $\mathcal{P}'$ rather than $\mathcal{P}$, then we can rule out (under ETH) an approximate polynomial Turing compression of any problem $\mathcal{P}$ that satisfies the first condition into any problem $\mathcal{P}'$ that satisfies the second condition in the same lemma.

3.2 MK[2]-hardness

The approach described in Section 3.1 is unlikely to work for the problem Independent Dominating Set. That approach requires a $\mathcal{O}(2^{\log^c n})$-factor approximation algorithm with $c < 1$ to answer the queries of the Turing kernelization. However, there is no $\mathcal{O}(n^{1-\varepsilon})$-factor approximation for this problem for any $\varepsilon > 0$ unless P = NP [9].

In the following, we will prove that there is no constant-factor approximate polynomial Turing kernelization for Independent Dominating Set[vc], assuming a conjecture by Hermelin et al. [11] stating that parameterized decision problems that are hard for the complexity class MK[2] do not admit polynomial (exact) Turing kernelizations.

Let CNF-SAT denote the satisfiability problem for Boolean formulas in conjunctive normal form. The class MK[2] may be defined as the set of all parameterized problems that can be reduced with a PPT to CNF-SAT[n] where n denotes the number of variables.[2]

We will prove that an α-approximate polynomial Turing kernelization for Independent Dominating Set[vc] implies the existence of a polynomial Turing kernelization for CNF-SAT[n]. For this, we will need the following lemma allowing us to translate queries between oracles for Independent Dominating Set and CNF-SAT using a standard self-reduction:

▶ **Lemma 4.** *There is a polynomial-time algorithm that, given as input a graph G and access to an oracle that decides in a single step instances of CNF-SAT whose size is polynomially bounded in the size of G, outputs a minimum independent dominating set of G.*

Proof. The decision version of Independent Dominating Set, in which one is given a graph H and an integer k and is asked to decide whether H contains an independent dominating set of size at most k, is in NP and CNF-SAT is NP-hard, so there is a polynomial-time many-one reduction from Independent Dominating Set to CNF-SAT. For any graph H and integer k let $R(H,k)$ denote the instance of CNF-SAT obtained by applying this reduction to (H,k).

Let n be the number of vertices in G. We first determine the size of a minimum independent dominating set in G by querying the oracle for CNF-SAT on the instance $R(G,k)$ for each $k \in [n]$. Observe that the size of $R(G,k)$ is polynomially bounded in the size of G. Hence, we may input this instance to the oracle. Let k_0 be the smallest value of k for which this query returns yes.

We must then construct an independent dominating set of size k_0 in G. We initially set $\ell := k_0$, $H := G$, and $S := \emptyset$ and perform the following operation as long as $\ell > 0$. For each vertex u in H, we query the oracle on the instance $R(H - N[u], \ell - 1)$. If this query returns yes, then we add u to S and set $H := H - N[u]$ and $\ell := \ell - 1$. Once $\ell = 0$, we output S.

We claim that this procedure returns an independent dominating set of size k_0 if k_0 is the size of a minimum independent dominating set in G. Let X be a minimum independent dominating set in G and $u \in X$. Then, $X \setminus \{u\}$ is a minimum dominating set of size $k_0 - 1$ in $G - N[u]$ and the claim follows inductively. ◀

▶ **Theorem 5.** *If, for any $\alpha \geq 1$, there is an α-approximate polynomial Turing kernelization for Independent Dominating Set[vc], then there is a polynomial Turing kernelization for CNF-SAT[n].*

Proof. Assume that there is an α-approximate Turing kernelization for Independent Dominating Set[vc] whose size is bounded by the polynomial p. We will give a polynomial Turing kernelization for CNF-SAT[n].

Let the input be a formula F in conjunctive normal form over the variables $x_1, \ldots, x_n$ consisting of the clauses $C_1, \ldots, C_m$. First, we compute a graph G on which we then run the approximate Turing kernelization for Independent Dominating Set. The construction of the graph G in the following is due to Irving [14].

[2] This is not directly the definition given by Hermelin et al. [11], but an equivalent characterization.

Let $s := \lceil \alpha \cdot n \rceil + 1$. The graph $G = (V, E)$ contains vertices $v_1, \ldots, v_n$ and $\overline{v_1}, \ldots, \overline{v_n}$, representing the literals that may occur in F. Additionally, for each $j \in [m]$, there are s vertices $w_j^1, \ldots, w_j^s$ representing the clause C_j. For each $i \in [n]$, there is an edge between v_i and $\overline{v_i}$. There is also an edge between v_i and w_j^ℓ for all $\ell \in [s]$ if $x_i \in C_j$ and an edge between $\overline{v_i}$ and w_j^ℓ for all $\ell \in [s]$ if $\neg x_i \in C_j$. The intuition behind this construction is as follows: In G any independent dominating set may contain at most one of v_i and $\overline{v_i}$ for each $i \in [n]$, so any such set represents a partial truth assignment of the variables $x_1, \ldots, x_n$. If F is satisfiable, then G contains an independent dominating set of size n. Conversely, if F is not satisfiable, then any independent dominating set must contain $w_j^1, \ldots, w_j^s$ for some $j \in [m]$, so it must have size at least $s > \alpha \cdot n$, thus creating a gap of size greater than α between yes and no instances. Moreover, $\{v_i, \overline{v_i} \mid i \in [n]\}$ is a vertex cover in G, so the vertex cover number of G is polynomially bounded in n.

The Turing kernelization for CNF-SAT proceeds in the following manner. Given the formula F, it first computes the graph G and runs the α-approximate Turing kernelization for Independent Dominating Set on G. Whenever the α-approximate Turing kernelization queries the oracle on an instance G', this query is answered using the algorithm given by Lemma 4. Observe that the size of G' is polynomially bounded in the vertex cover number of G, which in turn is polynomially bounded in n, so the oracle queries made by this algorithm are possible. Let X be the independent dominating set for G output by the approximate Turing kernelization. Since this Turing kernelization is given access to a 1-approximate oracle, $|X| \leq \alpha \cdot \mathrm{OPT}_{\mathrm{IND}}(G)$. We claim that F is satisfiable if and only if $|X| \leq \alpha \cdot n$. With this claim, the Turing kernelization can return yes if and only if this condition is met.

If F is satisfiable and $\varphi\colon \{x_1, \ldots, x_n\} \to \{0, 1\}$ is a satisfying truth assignment, then $Y := \{v_i \mid \varphi(x_i) = 1\} \cup \{\overline{v_i} \mid \varphi(x_i) = 0\}$ is an independent dominating set in G. Hence, $|X| \leq \alpha \cdot \mathrm{OPT}_{\mathrm{IND}}(G) \leq \alpha \cdot |Y| = \alpha \cdot n$. Conversely, suppose that F is not satisfiable. For each $i \in [n]$, the set X may contain at most one of the vertices v_i and $\overline{v_i}$. Consider the partial truth assignment with $\varphi(x_i) := 1$ if $v_i \in X$ and $\varphi(x_i) := 0$ if $\overline{v_i} \in X$. Because F is unsatisfiable there is a clause C_j that is not satisfied by φ. Hence, X must contain the vertices $w_j^1, \ldots, w_j^s$. Therefore, $|X| \geq s > \alpha \cdot n$. ◀

4 Turing kernelizations for parameter tw + Δ

In this section, we will prove that the domination problems for which we proved lower bounds when parameterized by the vertex cover number do have $(1 + \varepsilon)$-approximate polynomial Turing kernels when parameterized by treewidth plus maximum degree. The following lemma is a generalization of [12, Lemma 11] and can be proved in the same way.

▶ **Lemma 6.** *Let G be a graph with n vertices, $\mathcal{T}$ be a nice tree decomposition of G, and $s \leq n$. Then, there is a node t of $\mathcal{T}$ such that $s \leq |V_t| \leq 2s$. Moreover, such a node t can be found in polynomial time.*

4.1 Dominating Set

We start with Dominating Set.

▶ **Lemma 7.** *Let $G = (V, E)$ be a graph.*

(i) *If Δ is the maximum degree of G, then $\mathrm{OPT}_{\mathrm{DS}}(G) \geq \frac{|V|}{\Delta+1}$.*

(ii) *If $A, B, C \subseteq V$ with $A \cup C = V$, $A \cap C = B$, and there are no edges between $A \setminus B$ and $C \setminus B$, then $\mathrm{OPT}_{\mathrm{DS}}(G) \geq \mathrm{OPT}_{\mathrm{DS}}(G[A]) + \mathrm{OPT}_{\mathrm{DS}}(G[C]) - 2|B|$.*

Algorithm 1 A $(1+\varepsilon)$-approximate polynomial Turing kernelization for Dominating Set parameterized by $\mathrm{tw}+\Delta$.

input : A graph $G=(V,E)$, nice tree decomposition $\mathcal{T}$ of width tw, $\varepsilon > 0$
1 $s \leftarrow 2 \cdot \frac{1+\varepsilon}{\varepsilon} \cdot (\mathrm{tw}+1) \cdot (\Delta+1)$
2 **if** $|V| \leq s$ **then**
3 Apply the c-approximate oracle to G and output the result.
4 **else**
5 Use Lemma 6 to find a node t in $\mathcal{T}$ such that $s \leq |V_t| \leq 2s$.
6 Apply the c-approximate oracle to $G[V_t]$ and let S_t be the solution output by the oracle.
7 Let $\mathcal{T}'$ be the tree obtained by deleting the subtree rooted at t except for the node t from $\mathcal{T}$.
8 Apply this algorithm to $(G-(V_t \setminus X_t), \mathcal{T}', \varepsilon)$ and let S' be the returned solution.
9 Return $S'_t \cup S'$.
10 **end**

Proof.

(i) Every vertex can only dominate its at most Δ neighbors and itself.

(ii) Let X be a dominating set in G of size $\mathrm{OPT}_{\mathrm{DS}}(G)$. Then $Y := (X \cap A) \cup B$ and $Z := (X \cap C) \cup B$ are dominating sets in $G[A]$ and $G[C]$, respectively. Hence,

$$\begin{aligned}\mathrm{OPT}_{\mathrm{DS}}(G) = |X| &= |X \cap A| + |X \cap C| - |X \cap B| \\ &\geq |Y| - |B \setminus X| + |Z| - |B| \\ &\geq \mathrm{OPT}_{\mathrm{DS}}(G[A]) + \mathrm{OPT}_{\mathrm{DS}}(G[C]) - 2|B|.\end{aligned}$$

◀

▶ **Theorem 8.** *For every $\varepsilon > 0$, there is a $(1+\varepsilon)$-approximate Turing kernelization for Dominating Set with $\mathcal{O}(\frac{1+\varepsilon}{\varepsilon} \cdot \mathrm{tw} \cdot \Delta)$ vertices.*

Proof. Consider Algorithm 1. This algorithm always returns a dominating set of G. If the algorithm terminates in line 3, then this is true because the oracle always outputs a dominating set. If it terminates in line 9, then let $v \in V$ be an arbitrary vertex. If $v \in V_t$, then v is dominated by a vertex in S_t, because S_t is a dominating set in $G[V_t]$. If $v \in V \setminus V_t$, then v is dominated by a vertex in S'.

The algorithm runs in polynomial time.

Finally, we must show that the solution output by the algorithm contains at most $c \cdot (1+\varepsilon) \cdot \mathrm{OPT}_{\mathrm{DS}}(G)$ vertices. We prove the claim by induction on the number of recursive calls. If there is no recursive call, the algorithm terminates in line 3 and the solution contains at most $c \cdot \mathrm{OPT}_{\mathrm{DS}}(G)$ vertices. Otherwise, by induction:

$$\begin{aligned}|S'_t \cup S'| &\leq |S'_t| + |S'| \leq c \cdot \mathrm{OPT}_{\mathrm{DS}}(G[V_t]) + |S'| \\ &= c \cdot (1+\varepsilon) \cdot \mathrm{OPT}_{\mathrm{DS}}(G[V_t]) - c \cdot \varepsilon \cdot \mathrm{OPT}_{\mathrm{DS}}(G[V_t]) + |S'| \\ &\overset{\dagger}{\leq} c \cdot (1+\varepsilon) \cdot \mathrm{OPT}_{\mathrm{DS}}(G[V_t]) - \frac{c \cdot \varepsilon \cdot |V_t|}{\Delta+1} + |S'| \\ &\leq c \cdot (1+\varepsilon) \cdot \mathrm{OPT}_{\mathrm{DS}}(G[V_t]) - 2 \cdot c \cdot (1+\varepsilon) \cdot (\mathrm{tw}+1) + |S'| \\ &\leq c \cdot (1+\varepsilon) \cdot \mathrm{OPT}_{\mathrm{DS}}(G[V_t]) - 2 \cdot c \cdot (1+\varepsilon) \cdot |X_t| + |S'|\end{aligned}$$

$$\begin{aligned}
&\leq c \cdot (1+\varepsilon) \cdot \mathrm{OPT}_{\mathrm{DS}}(G[V_t]) - 2 \cdot c \cdot (1+\varepsilon) \cdot |X_t| + c \cdot (1+\varepsilon) \cdot \mathrm{OPT}_{\mathrm{DS}}(G - V_t) \\
&= c \cdot (1+\varepsilon) \cdot (\mathrm{OPT}_{\mathrm{DS}}(G[V_t]) - 2|X_t| + \mathrm{OPT}_{\mathrm{DS}}(G - V_t)) \\
&\overset{\ddagger}{\leq} c \cdot (1+\varepsilon) \cdot \mathrm{OPT}_{\mathrm{DS}}(G).
\end{aligned}$$

Here, the inequality marked † follows from Lemma 7(i) and the one marked ‡ follows from Lemma 7(ii) with $A = (V \setminus V_t) \cup X_t$, $B = X_t$, and $C = V_t \setminus X_t$. ◀

4.2 Capacitated Dominating Set

Next, we consider Capacitated Dominating Set.

▶ **Lemma 9.** *Let $G = (V, E, \mathrm{cap})$ be a capacitated graph with maximum degree Δ and $A, B, C \subseteq V$ such that $A \cup C = V$, $A \cap C = B$, and there are no edges from $A \setminus B$ to $C \setminus B$.*

- **(i)** $\mathrm{OPT}_{\mathrm{CAP}}(G) \geq \mathrm{OPT}_{\mathrm{CAP}}(G[A]) + \mathrm{OPT}_{\mathrm{CAP}}(G[C]) - 2|B|$.
- **(ii)** *Given capacitated dominating sets (X, f) and (Y, g) in $G[A]$ and $G[C]$, respectively, one can in construct in polynomial time a capacitated dominating set for G of size at most $|X| + |Y| + (\Delta + 1) \cdot |B|$.*

Proof.

- **(i)** Let (X, f) with $X \subseteq V$ and $f \colon V \setminus X \to X$ be a capacitated dominating set of size $\mathrm{OPT}_{\mathrm{CAP}}(G)$. Then, (Y, g) with $Y \coloneqq (X \cap A) \cup B$ and $g(v) \coloneqq f(v)$ for all $v \in A \setminus Y$ is a capacitated dominating set in $G[A]$ and (Z, h) with $Z \coloneqq (X \cap C) \cup B$ and $h(v) \coloneqq f(v)$ for all $v \in C \setminus Z$ is a capacitated dominating set in $G[C]$. Hence,

$$\begin{aligned}
\mathrm{OPT}_{\mathrm{CAP}}(G) = |X| &= |X \cap A| + |X \cap C| - |X \cap B| \\
&= |Y| - |B \setminus X| + |Z| - |B| - |X \cap B| \\
&\geq |Y| + |Z| - 2|B| \\
&\geq \mathrm{OPT}_{\mathrm{CAP}}(G[A]) + \mathrm{OPT}_{\mathrm{CAP}}(G[C]) - 2|B|.
\end{aligned}$$

- **(ii)** We construct the capacitated dominating set (Z, h) for G as follows. Let $Z \coloneqq X \cup Y \cup N[B]$. Observe that $|N[B]| \leq (\Delta + 1)|B|$. Define h by setting $h(v) \coloneqq f(v)$ for all $v \in A \setminus Z$ and $h(v) \coloneqq g(v)$ for all $v \in C \setminus Z$. One can easily verify that this is a capacitated dominating set. ◀

▶ **Theorem 10.** *For every $\varepsilon > 0$, there is a $(1+\varepsilon)$-approximate Turing kernelization for Capacitated Dominating Set with $\mathcal{O}(\frac{1+\varepsilon}{\varepsilon} \cdot \mathrm{tw} \cdot \Delta^2)$ vertices.*

Proof. Consider Algorithm 2. This algorithm always returns a capacitated dominating set of G. If the algorithm terminates in line 3, then this is true because the oracle always outputs a capacitated dominating set. If it terminates in line 10, then (S_t, f_t) and (S', f') are capacitated dominating sets for $G[V_t]$ and $G - (V_t \setminus X_t)$, respectively. It follows by Lemma 9(ii), that (S, f) is a capacitated dominating set for G.

The algorithm runs in polynomial time.

Finally, we must show that the solution output by the algorithm contains at most $c \cdot (1+\varepsilon) \cdot \mathrm{OPT}_{\mathrm{CAP}}(G)$ vertices. We prove the claim by induction on the number of recursive calls. If there is no recursive call, the algorithm terminates in line 3 and the solution contains at most $c \cdot \mathrm{OPT}_{\mathrm{CAP}}(G)$ vertices. Otherwise, by induction:

Algorithm 2 A $(1+\varepsilon)$-approximate polynomial Turing kernelization for Capacitated Dominating Set parameterized by $\mathrm{tw}+\Delta$.

input : A graph $G=(V,E)$, nice tree decomposition $\mathcal{T}$ of width tw, $\varepsilon>0$
$s \leftarrow 3\cdot\frac{1+\varepsilon}{\varepsilon}\cdot(\mathrm{tw}+1)\cdot(\Delta+1)^2$
if $|V|\leq s$ **then**
  Apply the c-approximate oracle to G and output the result.
else
  Use Lemma 6 to find a node t in $\mathcal{T}$ such that $s\leq|V_t|\leq 2s$.
  Apply the c-approximate oracle to $G[V_t]$ and let (S_t,f_t) be the solution output by the oracle.
  Let $\mathcal{T}'$ be the tree obtained by deleting the subtree rooted at t except for the node t from $\mathcal{T}$.
  Apply this algorithm to $(G-(V_t\setminus X_t),\mathcal{T}',\varepsilon)$ and let (S',f') be the returned solution.
  Apply Lemma 9(ii) with $(X,f)=(S',f')$, $(Y,g)=(S_t,f_t)$, $A=(V\setminus V_t)\cup X_t$, $B=X_t$, and $C=V_t$. Let (S,f) be the resulting solution for G.
  Return (S,f).
end

$$
\begin{aligned}
|S| &\leq |S'_t|+|S'|+(\Delta+1)\cdot|X_t| \leq c\cdot\mathrm{OPT}_{\mathrm{CAP}}(G[V_t])+|S'|+(\Delta+1)\cdot|X_t| \\
&= c\cdot(1+\varepsilon)\cdot\mathrm{OPT}_{\mathrm{CAP}}(G[V_t])-c\cdot\varepsilon\cdot\mathrm{OPT}_{\mathrm{CAP}}(G[V_t])+|S'|+(\Delta+1)\cdot|X_t| \\
&\overset{\dagger}{\leq} c\cdot(1+\varepsilon)\cdot\mathrm{OPT}_{\mathrm{CAP}}(G[V_t])-\frac{c\cdot\varepsilon\cdot|V_t|}{\Delta+1}+|S'|+(\Delta+1)\cdot|X_t| \\
&\leq c\cdot(1+\varepsilon)\cdot\mathrm{OPT}_{\mathrm{CAP}}(G[V_t])-3\cdot c\cdot(1+\varepsilon)\cdot(\mathrm{tw}+1)\cdot(\Delta+1)+|S'|+(\Delta+1)\cdot|X_t| \\
&\overset{\ddagger}{\leq} c\cdot(1+\varepsilon)\cdot\mathrm{OPT}_{\mathrm{CAP}}(G[V_t])-3\cdot c\cdot(1+\varepsilon)\cdot|X_t|\cdot(\Delta+1)+|S'| \\
&\quad +c\cdot(1+\varepsilon)\cdot(\Delta+1)\cdot|X_t| \\
&\leq c\cdot(1+\varepsilon)\cdot\mathrm{OPT}_{\mathrm{CAP}}(G[V_t])-2\cdot c\cdot(1+\varepsilon)\cdot|X_t|\cdot(\Delta+1)+|S'| \\
&\leq c\cdot(1+\varepsilon)\cdot\mathrm{OPT}_{\mathrm{CAP}}(G[V_t])-2\cdot c\cdot(1+\varepsilon)\cdot|X_t|+c\cdot(1+\varepsilon)\cdot\mathrm{OPT}_{\mathrm{CAP}}(G-V_t) \\
&= c\cdot(1+\varepsilon)\cdot(\mathrm{OPT}_{\mathrm{CAP}}(G[V_t])-2|X_t|+\mathrm{OPT}_{\mathrm{CAP}}(G-V_t)) \\
&\overset{\P}{\leq} c\cdot(1+\varepsilon)\cdot\mathrm{OPT}_{\mathrm{CAP}}(G)
\end{aligned}
$$

The inequality marked † follows from Lemma 7(i) and the fact that $\mathrm{OPT}_{\mathrm{CAP}}(G)\geq\mathrm{OPT}_{\mathrm{DS}}(G)$. ‡ follows from the fact that $c\cdot(1+\varepsilon)\geq 1$ and ¶ from Lemma 9(i). ◀

4.3 Independent Dominating Set

The next problem we consider is Independent Dominating Set.

▶ **Lemma 11.** *Let $G=(V,E)$ be a graph with maximum degree Δ and $A,B,C\subseteq V$ such that $A\cup C=V$, $A\cap C=B$, and there are no edges from $A\setminus B$ to $C\setminus B$.*

- **(i)** *If X is an independent set in G, then there is an independent dominating set X' that contains X and $|X'\setminus X|$ is at most the number of vertices not dominated by X, and such a set X' can be computed in polynomial time.*
- **(ii)** $\mathrm{OPT}_{\mathrm{IND}}(G)\geq\mathrm{OPT}_{\mathrm{IND}}(G[A])+\mathrm{OPT}_{\mathrm{IND}}(G[C])-2|B|$.
- **(iii)** *Given independent dominating sets X and Y in $G[A]$ and $G[C]$, respectively, one can in construct in polynomial time an independent dominating set for G of size at most $|X|+|Y|+(\Delta+1)\cdot|B|$.*

Algorithm 3 A $(1+\varepsilon)$-approximate polynomial Turing kernelization for Independent Dominating Set parameterized by $\mathrm{tw}+\Delta$.

input : A graph $G=(V,E)$, nice tree decomposition $\mathcal{T}$ of width tw, $\varepsilon>0$
1 $s \leftarrow |V| \leq 3\cdot\frac{1+\varepsilon}{\varepsilon}\cdot(\mathrm{tw}+1)\cdot(\Delta+1)^2$
2 **if** $|V|\leq s$ **then**
3 Apply the c-approximate oracle to G and output the result.
4 **else**
5 Use Lemma 6 to find a node t in $\mathcal{T}$ such that $s\leq|V_t|\leq 2s$.
6 Apply the c-approximate oracle to $G[V_t]$ and let S_t be the solution output by the oracle.
7 Let $\mathcal{T}'$ be the tree obtained by deleting the subtree rooted at t except for the node t from $\mathcal{T}$.
8 Apply this algorithm to $(G-(V_t\setminus X_t),\mathcal{T}',\varepsilon)$ and let S' be the returned solution.
9 Apply Lemma 11(iii) with $X=S'$, $Y=S_t$, $A=(V\setminus V_t)\cup X_t$, $B=X_t$, and $C=V_t$. Let S be the resulting solution for G.
10 Return S.
11 **end**

Proof.

(i) If X is a dominating set, then $X' \coloneqq X$. Otherwise, there is a vertex $v\in V\setminus N[X]$. We add v to X and continue. Observe that when v is added to X, the latter remains an independent set.

(ii) Let X be an independent dominating set of size $\mathrm{OPT}_{\mathrm{IND}}(G)$ in G. Let $Y\coloneqq X\cap A$. Since $Y\subseteq X$, it follows that Y is an independent set. Moreover, Y dominates all vertices in $(A\setminus B)\cup(X\cap B)$, leaving at most $B\setminus X$ vertices undominated. We apply (i) to Y and obtain Y', an independent dominating set in $G[A]$ of size at most $|X\cap A|+|B|-|X\cap B|$. We apply the same argument to $G[C]$ to obtain an independent dominating set Z of size at most $|X\cap C|+|B|-|X\cap B|$. It follows that:

$$\begin{aligned}\mathrm{OPT}_{\mathrm{IND}}(G)=|X|&=|X\cap A|+|X\cap C|-|X\cap B|\\&=|Y|-|B\setminus X|+|Z|-|B|-|X\cap B|\\&\geq|Y|+|Z|-2|B|\\&\geq\mathrm{OPT}_{\mathrm{IND}}(G[A])+\mathrm{OPT}_{\mathrm{IND}}(G[C])-2|B|.\end{aligned}$$

(iii) $Z\coloneqq(X\cup Y)\setminus B$ is an independent set in G. Since $X\cup Y$ is a dominating set and at most $(\Delta+1)\cdot|B|$ vertices can be dominated by vertices in B, it follows that Z leaves at most that many vertices in G undominated. Applying (i) to Z yields an independent dominating set of size at most $|X|+|Y|+(\Delta+1)\cdot|B|$. ◀

▶ **Theorem 12.** *For every $\varepsilon>0$, there is a $(1+\varepsilon)$-approximate Turing kernelization for Independent Dominating Set with $\mathcal{O}(\frac{1+\varepsilon}{\varepsilon}\cdot\mathrm{tw}\cdot\Delta^2)$ vertices.*

Proof. Consider Algorithm 3. This algorithm always returns an independent dominating set of G. If the algorithm terminates in line 3, then this is true because the oracle always outputs an independent dominating set. If it terminates in line 10, then S_t and S' are independent dominating sets for $G[V_t]$ and $G-(V_t\setminus X_t)$, respectively. It follows by Lemma 11(iii), that S is an independent dominating set for G.

The algorithm runs in polynomial time.

Finally, we must show that the solution output by the algorithm contains at most $c \cdot (1+\varepsilon) \cdot \mathrm{OPT}_{\mathrm{IND}}(G)$ vertices. We prove the claim by induction on the number of recursive calls. If there is no recursive call, the algorithm terminates in line 3 and the solution contains at most $c \cdot \mathrm{OPT}_{\mathrm{IND}}(G)$ vertices. Otherwise, by induction:

$$\begin{aligned}
|S| &\leq |S'_t| + |S'| + (\Delta+1) \cdot |X_t| \leq c \cdot \mathrm{OPT}_{\mathrm{IND}}(G[V_t]) + |S'| + (\Delta+1) \cdot |X_t| \\
&= c \cdot (1+\varepsilon) \cdot \mathrm{OPT}_{\mathrm{IND}}(G[V_t]) - c \cdot \varepsilon \cdot \mathrm{OPT}_{\mathrm{IND}}(G[V_t]) + |S'| + (\Delta+1) \cdot |X_t| \\
&\overset{\dagger}{\leq} c \cdot (1+\varepsilon) \cdot \mathrm{OPT}_{\mathrm{IND}}(G[V_t]) - \frac{c \cdot \varepsilon \cdot |V_t|}{\Delta+1} + |S'| + (\Delta+1) \cdot |X_t| \\
&\leq c \cdot (1+\varepsilon) \cdot \mathrm{OPT}_{\mathrm{IND}}(G[V_t]) - 3 \cdot c \cdot (1+\varepsilon) \cdot (\mathrm{tw}+1) \cdot (\Delta+1) + |S'| + (\Delta+1) \cdot |X_t| \\
&\overset{\ddagger}{\leq} c \cdot (1+\varepsilon) \cdot \mathrm{OPT}_{\mathrm{IND}}(G[V_t]) - 3 \cdot c \cdot (1+\varepsilon) \cdot |X_t| \cdot (\Delta+1) \\
&\quad + |S'| + c \cdot (1+\varepsilon) \cdot (\Delta+1) \cdot |X_t| \\
&\leq c \cdot (1+\varepsilon) \cdot \mathrm{OPT}_{\mathrm{IND}}(G[V_t]) - 2 \cdot c \cdot (1+\varepsilon) \cdot |X_t| \cdot (\Delta+1) + |S'| \\
&\leq c \cdot (1+\varepsilon) \cdot \mathrm{OPT}_{\mathrm{IND}}(G[V_t]) - 2 \cdot c \cdot (1+\varepsilon) \cdot |X_t| + c \cdot (1+\varepsilon) \cdot \mathrm{OPT}_{\mathrm{IND}}(G - V_t) \\
&= c \cdot (1+\varepsilon) \cdot (\mathrm{OPT}_{\mathrm{IND}}(G[V_t]) - 2|X_t| + \mathrm{OPT}_{\mathrm{IND}}(G - V_t)) \\
&\overset{\P}{\leq} c \cdot (1+\varepsilon) \cdot \mathrm{OPT}_{\mathrm{IND}}(G)
\end{aligned}$$

The inequality marked with † follows from Lemma 7(i) and the fact that $\mathrm{OPT}_{\mathrm{IND}}(G) \geq \mathrm{OPT}_{\mathrm{DS}}(G)$. ‡ follows from the fact that $c \cdot (1+\varepsilon) \geq 1$ and ¶ from Lemma 11(ii). ◀

4.4 Connected Dominating Set

Finally, we consider the problem Connected Dominating Set. If $S \subseteq V$ is a vertex set in a graph $G = (V, E)$, then let $R(G, S)$ denote the graph obtained by deleting S, introducing a new vertex z, and connecting z to any vertex in $V \setminus S$ that has a neighbor in S.

▶ **Lemma 13.** *Let $G = (V, E)$ be a connected graph and $A, B, C \subseteq V$ such that $A \cup C = V$, $A \cap C = B$, there are no edges from $A \setminus B$ to $C \setminus B$, and $A \setminus B$ and $C \setminus B$ are both non-empty.*

- **(i)** $\mathrm{OPT}_{\mathrm{CON}}(G) \geq \mathrm{OPT}_{\mathrm{CON}}(R(G[A], B)) + \mathrm{OPT}_{\mathrm{CON}}(R(G[C], B)) - 2$.
- **(ii)** *Given connected dominating sets X and Y in $R(G[A], B)$ and $R(G[C], B)$, respectively, one can in construct in polynomial time a connected dominating set for G of size at most $|X| + |Y| + 3|B|$.*

Proof.

- **(i)** Let X be a connected dominating set in G of size $\mathrm{OPT}_{\mathrm{CON}}(G)$. We claim that $Y \coloneqq (X \cap (A \setminus B)) \cup \{z\}$ and $Z \coloneqq (X \cap (C \setminus B)) \cup \{z\}$ are connected dominating sets in $R(G[A], B)$ and $R(G[C], B)$, respectively. We only prove this for Y and $R(G[A], B))$, as the case of Z and $R(G[C], B)$ is analogous.

  First, we show that Y is a dominating set. Let v be a vertex in $R(G[A], B))$. If $v \in \{z, z'\}$, then v is dominated by z in Y. Otherwise, $v \in A \setminus B$ and there is a vertex $w \in X$ that dominates v in G. If $w \in B$, then z is adjacent to v in $R(G[A], B))$ and v is dominated by z in that graph. If $w \in A \setminus B$, then $w \in Y$ and v is dominated by w in $R(G[A], B))$.

  We must also show that the subgraph of $R(G[A], B)$ induced by Y is connected. Let $v, v' \in Y$. We must show that the subgraph of $R(G[A], B)$ induced by Y contains a path from v to v'. First we assume that $v, v' \neq z, z'$, implying that $v, v' \in X$. Hence, there is a path P from v to v' in $G[X]$. If $P \subseteq A \setminus B$, then $P \subseteq Y$ and we are done. Otherwise, P must pass through B. Let w be the first vertex in B on P and w' the

final one. Obtain P' by replacing the subpath of P between and including w and w' with z. Then, P' is a path from v to v' in the subgraph of $R(G[A], B)$ induced by Y. Finally, suppose that $v' = z$. If $v = z$, there is nothing to show, so we assume that $v \neq z$ and, therefore, $v \in X \cap (A \setminus B)$. Because $C \setminus B$ is non-empty, X must contain a vertex $w \in B$. Because $G[X]$ is connected, $G[X]$ must also contain a path P from v to w. Let w' be the first vertex in B on the path P (possibly, $w' = w$). We obtain P', a path from v to z in the subgraph of $R(G[A], B)$ induced by Y, by taking the subpath of P from v to w' and replacing w' with z. This proves that Y is a connected dominating set in $R(G[A], B)$. Then,

$$\begin{aligned}\mathrm{OPT}_{\mathrm{CON}}(G) = |X| &\geq |X \cap (A \setminus B)| + |X \cap (C \setminus B)| \\ &\geq |Y| - 1 + |Z| - 1 \\ &\geq \mathrm{OPT}_{\mathrm{CON}}(R(G[A], B)) + \mathrm{OPT}_{\mathrm{CON}}(R(G[C], B)) - 2.\end{aligned}$$

(ii) Let $Z' \coloneqq X \cup Y \cup B$. Every connected component of $G[Z']$ contains a vertex in B, so this graph as at most $|B|$ connected components. We obtain a connected dominating set Z in G as follows. We start with $Z \coloneqq Z'$. Choose two connected components C_1, C_2 in $G[Z]$. Because G is connected, it contains a path P starting in $v_1 \in C_1$ and ending in $v_2 \in C_2$. This path must contain a vertex that is not adjacent to any vertex in C_1, because if every vertex in $P \setminus C_1$ were adjacent to a vertex in C_1, then v_2 is adjacent to a vertex in C_1, implying that C_1 and C_2 are not distinct connected components in $G[Z]$ Let w be the first vertex on P that is not adjacent to a vertex in C_1. Because Z is a dominating set in G, there must be a vertex $x \in Z \setminus C_1$ such that $w \in N[x]$ (note that, possible $w = x$). Adding w and x merges C_1 with the connected component of $G[Z]$ containing x. This process must be repeated at most $|B|$ times to obtain a connected dominating set. In each iteration at most two vertices are added to Z. Since Z initially contains $|X| + |Z| + |B|$ vertices, we obtain a connected dominating set containing at most $|X| + |Z| + 3|B|$ vertices. ◀

▶ **Theorem 14.** *For every $\varepsilon > 0$, there is a $(1+\varepsilon)$-approximate Turing kernelization for Connected Dominating Set with $\mathcal{O}(\frac{1+\varepsilon}{\varepsilon} \cdot \mathrm{tw} \cdot \Delta)$ vertices.*

Proof. Consider Algorithm 4. This algorithm always returns a connected dominating set of G. If the algorithm terminates in line 3, then this is true because the oracle always outputs a connected dominating set. If it terminates in line 10, then S_t and S' are connected dominating sets for $R(G[V_t], X_t)$ and $R(G - (V_t \setminus X_t))$, respectively. It follows by Lemma 13(ii), that S is a connected dominating set for G.

The algorithm runs in polynomial time.

Finally, we must show that the solution output by the algorithm contains at most $c \cdot (1+\varepsilon) \cdot \mathrm{OPT}_{\mathrm{CON}}(G)$ vertices. We prove the claim by induction on the number of recursive calls. If there is no recursive call, the algorithm terminates in line 3 and the solution contains at most $c \cdot \mathrm{OPT}_{\mathrm{CAP}}(G)$ vertices. Otherwise, by induction:

$$\begin{aligned}|S| &\leq |S'_t| + |S'| + 3|X_t| \leq c \cdot \mathrm{OPT}_{\mathrm{CON}}(R(G[V_t], X_t)) + |S'| + 3|X_t| \\ &= c \cdot (1+\varepsilon) \cdot \mathrm{OPT}_{\mathrm{CON}}(R(G[V_t], X_t)) - c \cdot \varepsilon \cdot \mathrm{OPT}_{\mathrm{CON}}(R(G[V_t], X_t)) + |S'| + 3|X_t| \\ &\overset{\dagger}{\leq} c \cdot (1+\varepsilon) \cdot \mathrm{OPT}_{\mathrm{CON}}(R(G[V_t], X_t)) - \frac{c \cdot \varepsilon \cdot (|V_t| - X_t + 1)}{\Delta + 1} + |S'| + 3|X_t| \\ &= c \cdot (1+\varepsilon) \cdot \mathrm{OPT}_{\mathrm{CON}}(R(G[V_t], X_t)) - \frac{c \cdot \varepsilon \cdot |V_t|}{\Delta + 1} + |S'| + \frac{c \cdot \varepsilon \cdot (|X_t| + 1)}{\Delta + 1} + 3|X_t|\end{aligned}$$

Algorithm 4 A $(1+\varepsilon)$-approximate polynomial Turing kernelization for Connected Dominating Set parameterized by $\mathrm{tw}+\Delta$.

input : A graph $G=(V,E)$, nice tree decomposition $\mathcal{T}$ of width tw, $\varepsilon > 0$
$s \leftarrow 4 \cdot \frac{1+\varepsilon}{\varepsilon}(\Delta+1)(\mathrm{tw}+1) + \frac{(2\Delta+2)(1+\varepsilon)}{\varepsilon}$
if $|V| \leq s$ **then**
  Apply the c-approximate oracle to G and output the result.
else
  Use Lemma 6 to find a node t in $\mathcal{T}$ such that $s \leq |V_t| \leq 2s$.
  Apply the c-approximate oracle to $R(G[V_t], X_t)$ and let S_t be the solution output by the oracle..
  Let $\mathcal{T}'$ be the tree obtained by deleting the subtree rooted at t except for the node t from $\mathcal{T}$.
  Apply this algorithm to $(R(G-(V_t \setminus X_t), X_t), \mathcal{T}', \varepsilon)$ and let S' be the returned solution.
  Apply Lemma 13(ii) with $X = S'$, $Y = S_t$, $A = (V \setminus V_t) \cup X_t$, $B = X_t$, and $C = V_t$. Let S be the resulting solution for G.
  Return S.
end

$$
\begin{aligned}
&\leq c\cdot(1+\varepsilon)\cdot \mathrm{OPT}_{\mathrm{CON}}(G[V_t]) - 4\cdot c\cdot(1+\varepsilon)\cdot(\mathrm{tw}+2) - 2c(1+\varepsilon) + |S'| \\
&\quad + \frac{c\cdot\varepsilon\cdot(|X_t|+1)}{\Delta+1} + 3|X_t| \\
&\overset{\ddagger}{\leq} c\cdot(1+\varepsilon)\cdot \mathrm{OPT}_{\mathrm{CON}}(G[V_t]) - 4\cdot c\cdot(1+\varepsilon)\cdot(|X_t|+1) - 2c(1+\varepsilon) + |S'| \\
&\quad + c\cdot(1+\varepsilon)\cdot(4|X_t|+1) \\
&\leq c\cdot(1+\varepsilon)\cdot \mathrm{OPT}_{\mathrm{CON}}(G[V_t]) - 2\cdot c(1+\varepsilon) + |S'| \\
&\leq c\cdot(1+\varepsilon)\cdot \mathrm{OPT}_{\mathrm{CON}}(G[V_t]) - 2c(1+\varepsilon) + c\cdot(1+\varepsilon)\cdot \mathrm{OPT}_{\mathrm{CON}}(G-V_t) \\
&= c\cdot(1+\varepsilon)\cdot(\mathrm{OPT}_{\mathrm{CON}}(G[V_t]) - 2 + \mathrm{OPT}_{\mathrm{CON}}(G-V_t)) \\
&\overset{\P}{\leq} c\cdot(1+\varepsilon)\cdot \mathrm{OPT}_{\mathrm{CON}}(G)
\end{aligned}
$$

The inequality marked with † follows from Lemma 7(ii) and the fact that $\mathrm{OPT}_{\mathrm{CON}}(G) \geq \mathrm{OPT}_{\mathrm{DS}}(G)$. ‡ follows from the fact that $c\cdot(1+\varepsilon) \geq 1$ and ¶ from Lemma 13(i). ◀

5 Conclusion

We conclude by pointing out two open problems concerning approximate Turing kernelization:

- Does Connected Feedback Vertex Set parameterized by treewidth admit an approximate polynomial Turing kernelization? The approach employed by Hols et al. [12] for Connected Vertex Cover and here for Connected Dominating Set cannot be used for Connected Feedback Vertex Set, because the ratio between the size of a minimum connected feedback vertex and the size of a minimum feedback vertex set is unbounded.
- The biggest open question in Turing kernelization is whether or not there are polynomial Turing kernelizations for the problems Longest Path and Longest Cycle parameterized by the solution size [11]. There has been some progress on this problem by considering the restriction to certain graph classes [15, 16]. Developing an *approximate* Turing kernelization may be another way of achieving progress in this regard.

References

1 Daniel Binkele-Raible, Henning Fernau, Fedor V. Fomin, Daniel Lokshtanov, Saket Saurabh, and Yngve Villanger. Kernel(s) for problems with no kernel: On out-trees with many leaves. *ACM Transactions on Algorithms*, 8(4), 2012. doi:10.1145/2344422.2344428.

2 M. Chlebík and J. Chlebíková. Approximation hardness of dominating set problems in bounded degree graphs. *Information and Computation*, 206(11):1264–1275, 2008. doi:10.1016/j.ic.2008.07.003.

3 Vasek Chvátal. A greedy heuristic for the set-covering problem. *Mathematics of Operations Research*, 4(3):233–235, 1979. doi:10.1287/moor.4.3.233.

4 Michael Dom, Daniel Lokshtanov, and Saket Saurabh. Kernelization lower bounds through colors and IDs. *ACM Transactions on Algorithms*, 11(2):1–20, 2014. doi:10.1145/2650261.

5 Uriel Feige, MohammadTaghi Hajiaghayi, and James R. Lee. Improved approximation algorithms for minimum weight vertex separators. *SIAM Journal on Computing*, 38(2):629–657, 2008. doi:10.1137/05064299X.

6 Michael R. Fellows, Ariel Kulik, Frances Rosamond, and Hadas Shachnai. Parameterized approximation via fidelity preserving transformations. *Journal of Computer and System Sciences*, 93:30–40, 2018. doi:10.1016/j.jcss.2017.11.001.

7 Fedor V Fomin, Daniel Lokshtanov, Saket Saurabh, and Meirav Zehavi. *Kernelization: Theory of Parameterized Preprocessing*. Cambridge University Press, 2019. doi:10.1017/9781107415157.

8 Sudipto Guha and Samir Khuller. Approximation algorithms for connected dominating sets. *Algorithmica*, 20:374–387, 1998. doi:10.1007/PL00009201.

9 Magnús M. Halldórsson. Approximating the minimum maximal independence number. *Information Processing Letters*, 46(4):169–172, 1993. doi:10.1016/0020-0190(93)90022-2.

10 Pinar Heggernes, Pim van 't Hof, Bart M.P. Jansen, Stefan Kratsch, and Yngve Villanger. Parameterized complexity of vertex deletion into perfect graph classes. *Theoretical Computer Science*, 511:172–180, 2013. doi:10.1016/j.tcs.2012.03.013.

11 Danny Hermelin, Stefan Kratsch, Karolina Sołtys, Magnus Wahlström, and Xi Wu. A completeness theory for polynomial (Turing) kernelization. *Algorithmica*, 71(3):702–730, 2015. doi:10.1007/s00453-014-9910-8.

12 Eva-Maria C. Hols, Stefan Kratsch, and Astrid Pieterse. Approximate Turing kernelization for problems parameterized by treewidth. In *Proceedings of the 28th Annual European Symposium on Algorithms (ESA)*, pages 60:1–60:23, 2020. doi:10.4230/LIPIcs.ESA.2020.60.

13 Russell Impagliazzo and Ramamohan Paturi. On the complexity of k-SAT. *Journal of Computer and System Sciences*, 62(2):367–375, 2001. doi:10.1006/jcss.2000.1727.

14 Robert W. Irving. On approximating the minimum independent dominating set. *Information Processing Letters*, 37(4):197–200, 1991. doi:10.1016/0020-0190(91)90188-N.

15 Bart M. P. Jansen. Turing kernelization for finding long paths and cycles in restricted graph classes. *Journal of Computer and System Sciences*, 85:18–37, 2017. doi:10.1016/j.jcss.2016.10.008.

16 Bart M. P. Jansen, Marcin Pilipczuk, and Marcin Wrochna. Turing kernelization for finding long paths in graphs excluding a topological minor. In *Proceedings of the 12th International Symposium on Parameterized and Exact Computation (IPEC)*, pages 23:1–23:13, 2018. doi:10.4230/LIPIcs.IPEC.2017.23.

17 Philip Klein and R. Ravi. A nearly best-possible approximation algorithm for node-weighted Steiner trees. *Journal of Algorithms*, 19(1):104–115, 1995. doi:10.1006/jagm.1995.1029.

18 Ton Kloks. *Treewidth: Computations and Approximations*. Springer, 1994. doi:10.1007/BFb0045375.

19 Daniel Lokshtanov. Wheel-free deletion is W[2]-hard. In *Proceedings of the 3rd International Symposium on Parameterized and Exact Computation (IPEC)*, pages 141–147, 2008. doi:10.1007/978-3-540-79723-4_14.

20 Daniel Lokshtanov, Fahad Panolan, M. S. Ramanujan, and Saket Saurabh. Lossy kernelization. *CoRR*, abs/1604.04111, 2016. Full version of [21]. doi:10.48550/arXiv.1604.04111.

21 Daniel Lokshtanov, Fahad Panolan, M. S. Ramanujan, and Saket Saurabh. Lossy kernelization. In *Proceedings of the 49th Annual ACM Symposium on Theory of Computing (STOC 2017)*, pages 224–237, 2017. doi:10.1145/3055399.3055456.

22 Jelani Nelson. A note on set cover inapproximability independent of universe size. *Electronic Colloquium on Computational Complexity*, TR07-105, 2007. URL: https://eccc.weizmann.ac.il/eccc-reports/2007/TR07-105/index.html.

23 Laurence A. Wolsey. An analysis of the greedy algorithm for the submodular set covering problem. *Combinatorica*, 2(4):385–393, 1982. doi:10.1007/BF02579435.