

Treewidth Inapproximability and Exponential ETH Lower Bound

Édouard Bonnet   

Univ Lyon, CNRS, ENS de Lyon, Université Claude Bernard Lyon 1, LIP UMR5668, France

Daniel Neuen  

Max Planck Institute for Informatics, Saarland Informatics Campus, Saarbrücken, Germany

Marek Sokołowski   

Max Planck Institute for Informatics, Saarland Informatics Campus, Saarbrücken, Germany

Abstract

Treewidth is a central parameter to algorithmic graph theory. The current state-of-the-art in computing and approximating treewidth consists of a $2^{O(k^2)}n$ -time exact algorithm and a polynomial-time $O(\text{OPT} \log^{3/2} \text{OPT})$ -approximation algorithm, where the former algorithm returns an *elimination forest* of height k (witnessing that treewidth is at most k) for the n -vertex input graph G , or correctly reports that G has treewidth larger than k , and OPT is the actual value of the treewidth. On the complexity side, exactly computing treewidth is NP-complete, but the known reductions do not rule out a polynomial-time approximation scheme (PTAS), and under the Exponential Time Hypothesis (ETH) only exclude a running time of $2^{o(\sqrt{n})}$ for exact algorithms.

We show that 1.0003-approximating TREewidth is NP-hard, and that exactly computing the treewidth of an n -vertex graph requires time $2^{\Omega(n)}$, unless the ETH fails. We further derive that there exist absolute constants $\delta, c > 0$ such that any $(1 + \delta)$ -approximation algorithm requires time $2^{\Omega(n / \log^c n)}$. We do so via a simple direct reduction from SATISFIABILITY to TREewidth, inspired by a reduction recently designed for TREewidth [STOC '25].

2012 ACM Subject Classification Theory of computation $\rightarrow$ Problems, reductions and completeness; Theory of computation $\rightarrow$ Parameterized complexity and exact algorithms

Keywords and phrases treewidth, lower bounds, approximation

Digital Object Identifier 10.4230/LIPIcs.IPEC.2025.17

Related Version Full Version: <https://arxiv.org/abs/2507.13818>

Funding Édouard Bonnet: Supported by the French National Research Agency through the project TWIN-WIDTH with reference number ANR-21-CE48-0014.

1 Introduction

The treewidth $\text{td}(G)$ of a graph G is the least integer k such that there is a rooted forest F of height k with same vertex set as G such that every edge of G is between two nodes in ancestor–descendant relationship in F . For every n -vertex graph G , its treewidth and treewidth, denoted by $\text{tw}(G)$, are related by the inequalities $\text{tw}(G) + 1 \leq \text{td}(G) \leq \text{tw}(G) \cdot (1 + \log n)$. An n -vertex path has treewidth (even pathwidth) 1, but treewidth $\Theta(\log n)$.

Treewidth comes into play in various contexts. Notably, in the sparsity theory initiated by Nešetřil and Ossona de Mendez [26], treewidth provides a characterization of classes of bounded expansion (roughly speaking, classes excluding, as subgraphs, short subdivisions of graphs of large average degree). Graph classes with bounded expansion are exactly those with so-called *low treewidth covers* (some form of cover by graphs of bounded treewidth) [21].

Graphs of bounded treewidth famously lend themselves to fixed-parameter tractable (FPT) algorithms for various NP-hard problems, with parameter the width of the computed (or given) tree-decompositions, by performing dynamic programming over these decompositions



© Édouard Bonnet, Daniel Neuen, and Marek Sokołowski;
licensed under Creative Commons License CC-BY 4.0

20th International Symposium on Parameterized and Exact Computation (IPEC 2025).

Editors: Akanksha Agrawal and Erik Jan van Leeuwen; Article No. 17; pp. 17:1–17:10

Leibniz International Proceedings in Informatics



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

(see, e.g., [7, Chapter 7]). However, this method consumes essentially as much space as it takes time; in particular, most of these algorithms for NP-hard problems take exponential space in the treewidth. Bounded treedepth, in contrast, often allows for parameterized algorithms with comparable running time but using only polynomial space; see for instance [12, 20, 22, 23]. Other uses of treedepth can be found in formula complexity [18], distributed model checking [10], product structure theory [9], relation to polynomial minors [16, 14], graph circumference [6] etc. We now survey the current state of the art on computing and approximating the treedepth of an input graph – the topic of the current paper. Note that TREEDEPTH was the selected problem for the 2020 edition of the PACE challenge [17].

The decision version of TREEDEPTH is NP-complete [24, 3]. There is an easy algorithm that computes the treedepth of an n -vertex graph in time $O(2^n \cdot n)$, and a slightly faster exponential algorithm computing the decomposition in time $O(1.9602^n)$ [11]. An FPT (exact) algorithm to decide whether an n -vertex graph has treedepth at most k with running time $2^{O(k^2)}n$ has been established, first needing exponential space [25], and later improved to only use polynomial space [19]. More precisely, these algorithms run in time $2^{O(\text{td}(G)\text{tw}(G))}n$ on an n -vertex graph G . An outstanding open question is whether running time $2^{O(k)}n^{O(1)}$ can be obtained. Notably, TREEWIDTH admits constant-approximation algorithms in this running time. In contrast to treewidth, such a result remains elusive for TREEDEPTH. On the front of polynomial-time approximation algorithms, the best factor that currently can be achieved is $O(\text{tw}(G) \log^{3/2} \text{tw}(G))$ [8], hence $O(\text{td}(G) \log^{3/2} \text{td}(G))$.

The previously known hardness results [24, 3] are rather unsatisfactory: First of all, they did not rule out a polynomial-time approximation scheme (PTAS) for TREEDEPTH. Furthermore, by following the reductions presented in both papers, we can only infer a $2^{\Omega(\sqrt{n})}$ lower bound for the exact variant of the problem under the Exponential Time Hypothesis (ETH).¹ Naturally, this excludes any $2^{o(\sqrt{k})}n^{O(1)}$ -time parameterized exact algorithm for the problem under ETH.

Our Results. In this work, we design a simple linear reduction from SATISFIABILITY to TREEDEPTH. It draws inspiration from a recent similar result for TREEWIDTH [4], and also relies on the hardness of VERTEX COVER (i.e., the task of finding a smallest vertex subset hitting every edge) on tripartite graphs. Our reduction yields the following inapproximability results and computational lower bounds for TREEDEPTH.

► **Theorem 1.** *It is NP-hard to 1.0003-approximate TREEDEPTH.*

In particular, the theorem rules out a polynomial-time approximation scheme (PTAS) for TREEDEPTH (assuming $P \neq NP$).

► **Theorem 2.** *Assuming ETH, there is some $\varepsilon > 0$ such that the treedepth of an n -vertex graph cannot be computed in time $O(2^{\varepsilon n})$.*

This also excludes $2^{o(k)}n^{O(1)}$ -time parameterized exact algorithms for the problem under ETH.

In fact, we obtain that even approximating treedepth to a small constant factor requires almost exponential time.

► **Theorem 3.** *Assuming ETH, there exist absolute constants $\delta, \varepsilon, c > 0$ such that $(1 + \delta)$ -approximating the treedepth of an n -vertex graph cannot be done in time $O(2^{\varepsilon n / \log^c n})$.*

¹ The assumption that there is some $\lambda > 1$ such that n -variable 3-SAT requires time $\Omega(\lambda^n)$.

All three results follow the same reduction chain. First, we transform a SATISFIABILITY instance into an instance of VERTEX COVER on tripartite graphs. Then, we reduce the VERTEX COVER problem on tripartite graphs to TREEDPTH. We first describe the second step of the reduction chain in Section 3. In Section 4, we then describe the first step, and deduce Theorems 1–3.

2 Preliminaries

In this work, we consider only simple undirected graphs with no self-loops. For a graph G , we define $V(G)$ to be the set of vertices, $E(G)$ to be the set of edges, and $cc(G)$ to be the set of connected components of G . Also, we write $N_G(v)$ for the set of neighbors of a vertex $v \in V(G)$ in the graph G . A forest F is a graph without cycles; it is rooted if each connected component of F (a *tree*) has a *root*. The depth of a forest is then the number of vertices on the longest root-to-leaf path in F . A vertex u is an *ancestor* of v in F if u lies on the path between v and the root of the tree of F containing v ; we equivalently then say that v is a *descendant* of u . In particular, every vertex is its own ancestor and descendant.

A set $S \subseteq V(G)$ is a *vertex cover* if each edge of G is incident to at least one vertex of S . The *vertex cover number* of G , denoted by $vc(G)$, is the minimum cardinality of a vertex cover of G .

The treedepth of G , denoted $td(G)$, is defined recursively as follows:

$$td(G) = \begin{cases} 0 & \text{if } G \text{ has no vertices,} \\ \max_{H \in cc(G)} td(H) & \text{if } G \text{ is disconnected,} \\ \min_{v \in V(G)} 1 + td(G - v) & \text{if } G \text{ is connected.} \end{cases}$$

We also use an equivalent definition of treedepth involving *elimination forests*. Here, we say that a rooted forest F is an elimination forest of G if $V(F) = V(G)$ and for every edge uv of G , vertices u and v are in the ancestor–descendant relationship in F . Then, the treedepth of G is the minimum possible depth of an elimination forest of G .

We use the following straightforward facts in our arguments:

► **Observation 4.** *If K is a clique in G , then in every elimination forest of G , all vertices of K are contained in a single root-to-leaf path.*

► **Observation 5.** *If for some nonempty $S \subseteq V(G)$, the induced subgraph $G[S]$ is connected, then in every elimination forest of G some vertex of S is an ancestor of all elements of S .*

We say that a set $X \subseteq V(G)$ of vertices of G is a *set of (false) twins* if $N_G(v) = N_G(w)$ for all $v, w \in X$. Observe that X is an independent set in this case. Every inclusion-wise minimal vertex cover of G either contains X in its entirety or is disjoint from X .

In this paper, we work with formulas with Boolean variables, say $x_1, \dots, x_n$. A *literal* is a formula of the form x_i or $\neg x_i$. For an integer k , a Boolean formula is said to be in *k -CNF form* (or: is a *k -CNF formula*) if it is a conjunction of *clauses*: subformulas of the form $\ell_1 \vee \dots \vee \ell_k$ for literals $\ell_1, \dots, \ell_k$, each containing a different variable of the formula.

3 Treedepth and Vertex Cover of Tripartite Graphs

In this section, we describe a reduction from VERTEX COVER on tripartite graphs to TREEDPTH.

We say that a graph $G = (V, E)$ is tripartite if there exists a tripartition $V = A \cup B \cup C$ such that the subgraphs of G induced by A , B , and C , respectively, are edgeless. We argue that it is possible to extend each tripartite graph G to a supergraph H by adding suitable clique gadgets so that the treedepth of H is tightly controlled by the vertex cover number of G .

► **Lemma 6.** *Let $G = (V, E)$ be a tripartite graph with tripartition $V = A \cup B \cup C$ such that A, B, C are nonempty and let ℓ be a positive integer such that $\ell \geq \text{vc}(G)$. Consider a supergraph $H = (V', E')$ of G created by adding to G three ℓ -vertex cliques K_A, K_B, K_C and three additional vertices z_A, z_B, z_C , and adding for each $X \in \{A, B, C\}$ all edges between K_X and $X \cup \{z_X\}$. That is,*

$$\begin{aligned} V' &= V \cup K_A \cup K_B \cup K_C \cup \{z_A, z_B, z_C\}, \\ E' &= E \cup (A \cup \{z_A\}) \times K_A \cup (B \cup \{z_B\}) \times K_B \cup (C \cup \{z_C\}) \times K_C. \end{aligned}$$

Then $\text{td}(H) = \text{vc}(G) + \ell + 1$.

Proof. Let S be the minimum vertex cover of G . Then $H - S$ has three connected components, namely, for each $X \in \{A, B, C\}$, the subgraph of H induced by $K_X \cup \{z_X\} \cup (X \setminus S)$. Noting that $|K_X| = \ell$ and $\{z_X\} \cup (X \setminus S)$ is an independent set in H , we have $\text{td}(H[K_X \cup \{z_X\} \cup (X \setminus S)]) \leq |K_X| + 1 = \ell + 1$, and therefore

$$\text{td}(H) \leq |S| + \max_{X \in \{A, B, C\}} \text{td}(H[K_X \cup \{z_X\} \cup (X \setminus S)]) \leq \text{vc}(G) + \ell + 1.$$

We now move to the lower bound on $\text{td}(H)$. Aiming for contradiction, suppose that $\text{td}(H) \leq \text{vc}(G) + \ell$; then there exists an elimination forest F of H of depth at most $\text{vc}(G) + \ell$.

By Observation 4, for each $X \in \{A, B, C\}$, all vertices of K_X belong to a single root-to-leaf path (possibly different for different choices of X). We can thus choose three vertices $\kappa_A \in K_A, \kappa_B \in K_B, \kappa_C \in K_C$ as the deepest vertices of the respective cliques in F .

► **Claim 7.** Let $X \in \{A, B, C\}$. Suppose that $R \subseteq V' \setminus (K_X \cup \{z_X\})$ is so that all vertices of R are ancestors of κ_X in F . Then the depth of F is at least $|R| + \ell + 1$.

Proof. All vertices of K_X are ancestors of κ_X , and z_X is either an ancestor or a descendant of κ_X . In either case, vertices of $R \cup K_X \cup \{z_X\}$ lie on a single root-to-leaf path in F , implying that the depth of F is at least $|R| + \ell + 1$. ◁

► **Claim 8.** No pair of vertices in $\{\kappa_A, \kappa_B, \kappa_C\}$ is in ancestor–descendant relationship in F .

Proof. Suppose without loss of generality that κ_A is an ancestor of κ_B . Then all vertices of K_A are ancestors of κ_B , so from Claim 7 we infer that the depth of F is at least $|K_A| + \ell + 1 = 2\ell + 1 > \text{vc}(G) + \ell$; a contradiction. ◁

► **Claim 9.** Let $X, Y \in \{A, B, C\}$ with $X \neq Y$ and pick $v_X \in X, v_Y \in Y$ connected by an edge. Then (at least) one of the vertices v_X, v_Y is an ancestor of both κ_X and κ_Y .

Proof. Let $R = \{v_X, v_Y, \kappa_X, \kappa_Y\}$. Noting that $\kappa_X v_X v_Y \kappa_Y$ is a path in H , we have by Observation 5 that some vertex of R is an ancestor of all vertices in R . Since κ_X and κ_Y are not in the ancestor–descendant relationship (Claim 8), the claim follows. ◁

Let us now resolve a degenerate case where at least one of the sides of G (say, C) is not incident to any edge in G . Define Q to be the set of vertices of $A \cup B$ that are ancestors of both κ_A and κ_B ; by Claim 9, Q is a vertex cover of G . Hence by Claim 7 for $X = A$, the depth of F is at least $|Q| + \ell + 1 > \text{vc}(G) + \ell$; a contradiction.

Thus each side of G is incident to an edge of G . Therefore, we can easily verify that H is connected and so F is a single rooted tree. Let then u_{AB} be the lowest common ancestor of κ_A and κ_B in F ; analogously define u_{BC} and u_{CA} . The three newly defined vertices pairwise remain in the ancestor–descendant relationship: for instance, u_{AB} and u_{BC} are both ancestors of κ_B , so one of them is an ancestor of the other. In particular, u_{AB} , u_{BC} and u_{CA} lie on a single root-to-leaf path, and (at least) one of these vertices – say, u_{AB} – is a descendant of all three. Let $Q \subseteq V$ be the set of vertices of G that are ancestors of κ_A . Applying Claim 9 multiple times, we observe that:

- for each $v_A v_B \in E(G)$ with $v_A \in A$, $v_B \in B$, we get $v_A \in Q$ or $v_B \in Q$;
- for each $v_A v_C \in E(G)$ with $v_A \in A$, $v_C \in C$, we get $v_A \in Q$ or $v_C \in Q$;
- for each $v_B v_C \in E(G)$ with $v_B \in B$, $v_C \in C$, we get that v_B or v_C is an ancestor of both κ_B and κ_C in F ; hence it is also an ancestor of u_{BC} , so also an ancestor of u_{AB} and thus also κ_A . Consequently $v_B \in Q$ or $v_C \in Q$.

We conclude that Q is a vertex cover of G ; and so by Claim 7, the depth of F is at least $|Q| + \ell + 1 > \text{vc}(G) + \ell$; a contradiction. ◀

4 Hardness Proofs

In this section, we will prove the announced hardness results (Theorems 1–3). All reductions will start from an instance φ of SATISFIABILITY in k -CNF form, in which every variable occurs a bounded number of times, and produce a graph (or a family of graphs) whose treedepth tightly depends on the maximum number of clauses that can be satisfied in φ . At the heart of our framework lies a construction of a tripartite graph adapted from the work of Bonnet [4], which we formally describe below.

Let φ be a k -CNF formula and p be an integer. We then define a tripartite graph $G(\varphi, p)$ as follows. Let $\gamma = 2^{k-1}p$. Define the vertex set of $G(\varphi, p)$ as $A \cup B_+ \cup B_-$, where:

- For every clause $C_i = \ell_1 \vee \dots \vee \ell_k$ of φ and every possible choice $s_1 \in \{\ell_1, \neg \ell_1\}, \dots, s_k \in \{\ell_k, \neg \ell_k\}$ such that $(s_1, \dots, s_k) \neq (\neg \ell_1, \dots, \neg \ell_k)$, we add to A a vertex $a_i(s_1, \dots, s_k)$. Let $A(C_i)$ be the set of all vertices added to A for clause C_i . Intuitively, $A(C_i)$ contains all valuations of variables represented by the literals $\ell_1, \dots, \ell_k$ that satisfy the clause C_i .
- For every variable x_j of φ and every $t \in [\gamma]$, we add to B_+ a vertex $b_{j,t}$ and to B_- a vertex $c_{j,t}$. Let then $B_+(x_j) = \{b_{j,1}, \dots, b_{j,\gamma}\}$ and $B_-(x_j) = \{c_{j,1}, \dots, c_{j,\gamma}\}$. Also, for convenience, let $B(x_j) = B_+(x_j)$ and $B(\neg x_j) = B_-(x_j)$.

We will now construct the set of edges of $G(\varphi, p)$ to ensure that every valuation of variables of φ corresponds to an inclusion-wise minimal vertex cover S that includes: (i) all vertices of A , except one vertex of $A(C_i)$ for each satisfied clause C_i , corresponding to the valuation of the variables represented by the literals of C_i , and (ii) all vertices of $B_+(x_j)$ if x_j is evaluated positively, or $B_-(x_j)$ if x_j is evaluated negatively. To this end, we construct the set E of edges of $G(\varphi, p)$ via the following process:

- Add every edge between $B(x_j)$ and $B(\neg x_j)$ for every variable x_j ;
- Connect each vertex $a_i(s_1, \dots, s_k) \in A$ with every vertex of $B(s_1) \cup \dots \cup B(s_k)$.

This concludes the construction. We now show that the vertex cover number of $G(\varphi, p)$ is controlled by the maximum number of clauses satisfiable by φ .

► **Lemma 10.** *Let φ be a k -CNF formula with n variables and m clauses where every variable appears at most $2p + 1$ times. Suppose also that m' is the maximum number of clauses that can be satisfied in φ . Then*

$$vc(G(\varphi, p)) = (2^k - 1)m - m' + 2^{k-1}pn.$$

Proof. First, suppose that F is a valuation of variables of φ that satisfies m' clauses; we think of F as a set of literals that includes exactly one literal from $\{x_j, \neg x_j\}$ for each variable x_j . We define a set S of vertices of $G(\varphi, p)$ by including:

- every vertex of the form $a_i(s_1, \dots, s_k) \in A$ such that $\{s_1, \dots, s_k\} \not\subseteq F$; and
- all vertices of $B(\ell_j)$ for every literal $\ell_j \in F$.

Observe that S is a vertex cover of $G(\varphi, p)$. Indeed, every edge between B_+ and B_- is covered by S . Next, consider the edges between A and $B := B_+ \cup B_-$. Pick a vertex $a_i(s_1, \dots, s_k) \in A \setminus S$. Every neighbor of this vertex belongs to $B(s_1) \cup \dots \cup B(s_k)$. By construction of $S \cap A$, we have $\{s_1, \dots, s_k\} \subseteq F$, and so $B(s_1) \cup \dots \cup B(s_k) \subseteq S$.

Now let us determine the size of S . Since F satisfies m' clauses of φ , there exist exactly m' vertices of the form $a_i(s_1, \dots, s_k) \in A$ such that $s_1, \dots, s_k \in F$. These are precisely the vertices of A that do not belong to S . Therefore $|A \setminus S| = m'$ and so $|S \cap A| = (2^k - 1)m - m'$. Also, $|S \cap B| = \gamma n = 2^{k-1}pn$. Therefore, $|S| = (2^k - 1)m - m' + 2^{k-1}pn$.

On the other hand, suppose that S is a minimum-cardinality vertex cover of $G(\varphi, p)$; and out of those, S contains the fewest number of vertices of B . Assume that $|S| \leq (2^k - 1)m - m'' + 2^{k-1}pn$, aiming to show that at least m'' clauses of φ can be satisfied.

For every variable x_j of φ , each of the sets $B_+(x_j)$ and $B_-(x_j)$ is a set of twins in $G(\varphi, p)$, so S either contains it fully or is disjoint from it; and moreover, S must contain at least one of the sets $B_+(x_j)$ and $B_-(x_j)$ since the vertices of both sets are connected by a complete bipartite graph. Suppose now that S contains both sets $B_+(x_j)$ and $B_-(x_j)$. Since the variable x_j appears at most $2p + 1$ times in φ , we can pick a literal $\ell_j \in \{x_j, \neg x_j\}$ that appears at most p times in φ . Consider now the following set S^* :

$$S^* = (S \setminus B(\ell_j)) \cup \{a_i(s_1, \dots, s_k) \in A \mid \ell_j \in \{s_1, \dots, s_k\}\}.$$

Observing that S^* is formed from S by removing $B(\ell_j)$ and introducing all neighbors of $B(\ell_j)$ in $G(\varphi, p)$, we conclude that S^* is also a vertex cover of $G(\varphi, p)$. Since $|B(\ell_j)| = \gamma$, $B(\ell_j) \subseteq S$ and ℓ_j appears at most p times in φ , the cardinality of S^* is at most $|S^*| \leq |S| - \gamma + 2^{k-1}p = |S|$. This contradicts the minimality of S . Therefore, for every variable x_j , S contains fully one of the sets $B_+(x_j)$, $B_-(x_j)$ and is disjoint from the other. We obtain a valuation F of φ by including in F , for every variable x_j , the literal $\ell_j \in \{x_j, \neg x_j\}$ such that $B(\ell_j) \subseteq S$. We aim to show that F satisfies at least m'' clauses of φ .

Note that $|S \cap B| = \gamma n$ and $|A| = (2^k - 1)m$ and so $|A \setminus S| \geq m''$. Moreover, $|A(C_i) \setminus S| \leq 1$ for each clause C_i of φ ; otherwise, we would have $a_i(s_1, \dots, s_k), a_i(s'_1, \dots, s'_k) \in A(C_i) \setminus S$, where $s'_j = \neg s_j$ for some $j \in [k]$. But then $a_i(s_1, \dots, s_k)$ is connected to the vertices of $B(s_j)$ and $a_i(s'_1, \dots, s'_k)$ is connected to the vertices of $B(s'_j)$, and by the minimality of S , either of the sets $B(s_j)$, $B(s'_j)$ is disjoint from S ; a contradiction with the assumption that S is a vertex cover. Therefore, there exist at least m'' clauses C_i of φ such that $|A(C_i) \setminus S| = 1$. Observe that each such clause C_i is satisfied by F . Indeed, suppose $C_i = \ell_1 \vee \dots \vee \ell_k$ and let $a_i(s_1, \dots, s_k)$ be the only element of $A(C_i) \setminus S$. Then $s_j \in \{\ell_j, \neg \ell_j\}$ for each $j \in [k]$ and $(s_1, \dots, s_k) \neq (\neg \ell_1, \dots, \neg \ell_k)$ by the construction of $G(\varphi, p)$, and $B(s_1) \cup \dots \cup B(s_k) \subseteq S$ by the fact that S is a vertex cover. Thus $s_1, \dots, s_k \in F$ by the construction of F and so F satisfies φ (i.e., there is a literal $s_j \in \{s_1, \dots, s_k\}$ such that $s_j = \ell_j$). ◀

Overall, we obtain the following reduction from SATISFIABILITY to VERTEX COVER that we record here for future reference:

► **Corollary 11.** *For all pairs of integers $k \geq 2$, $p \geq 1$ there exists a polynomial-time algorithm that receives as input a k -CNF formula φ with n variables and m clauses where every variable appears at most $2p + 1$ times, and outputs a graph $G(\varphi, p)$ with $|V(G(\varphi, p))| = O(n)$ and the following property: If m' is the maximum number of clauses that can be satisfied in φ , then*

$$vc(G(\varphi, p)) = (2^k - 1)m - m' + 2^{k-1}pn.$$

As an immediate corollary, we get that:

► **Corollary 12.** *For all pairs of integers $k \geq 2$, $p \geq 1$ there exists a polynomial-time algorithm that receives as input a k -CNF formula φ with n variables and m clauses where every variable appears at most $2p + 1$ times, and outputs a graph $H(\varphi, p)$ with $|V(H(\varphi, p))| = O(n)$ and the following property: If m' is the maximum number of clauses that can be satisfied in φ , then*

$$td(H(\varphi, p)) = 2(2^k - 1)m - m' + 2^k pn + 1.$$

Proof. Apply Lemma 6 to the graph $G(\varphi, p)$ with vertex set $A \cup B_+ \cup B_-$ and the parameter $\ell = |A \cup B_+| = (2^k - 1)m + 2^{k-1}pn$ (the choice of ℓ comes from the fact that $A \cup B_+$ is a vertex cover of $G(\varphi, p)$). The value of $td(H(\varphi, p))$ follows from Lemmas 6 and 10. ◀

We are now almost ready to show the approximation hardness of TREEDPTH (Theorem 1). We start from the following approximation hardness of the maximization variant of 2-SAT:

► **Theorem 13** ([2, Theorem 12]). *For every $\varepsilon > 0$, within the family of m -clause 2-CNF formulas where each variable appears 3 or 4 times, it is NP-hard to distinguish between the formulas where at least $(1 - \varepsilon)m$ clauses are satisfiable and formulas where at most $(\frac{251}{252} + \varepsilon)m$ clauses are satisfiable.*

Proof of Theorem 1. Consider the family of 2-CNF formulas where each variable appears 3 or 4 times. In this family, every n -variable, m -clause formula satisfies $2m \geq 3n$, or equivalently $n \leq \frac{2}{3}m$. By Corollary 12, every such formula φ can be translated – in polynomial time – to a graph $H(\varphi)$ with the property that if m' is the maximum number of clauses satisfiable in φ , then $td(H(\varphi)) = 6m - m' + 8n + 1$.

Now let $\delta < \frac{1}{2604}$ be fixed. Then there is some $\varepsilon > 0$ such that, for large enough m and $n \leq \frac{2}{3}m$,

$$(6m - (1 - \varepsilon)m + 8n + 1)(1 + \delta) < 6m - \left(\frac{251}{252} + \varepsilon\right)m + 8n + 1.$$

So, letting $k = 6m - (1 - \varepsilon)m + 8n$, distinguishing between formulas φ where at least $(1 - \varepsilon)m$ clauses are satisfiable and formulas where at most $(\frac{251}{252} + \varepsilon)m$ clauses are satisfiable reduces to distinguishing between graphs of treedepth at most k and those of treedepth at least $(1 + \delta)k$. Hence, we obtain hardness by Theorem 13. ◀

We move to the hardness results under the Exponential Time Hypothesis (ETH); both presented proofs invoke the Sparsification Lemma of Impagliazzo, Paturi, and Zane [15]. We use this lemma in the following form:

► **Lemma 14** (Sparsification Lemma [15]). *For every $0 < \varepsilon' < 0.1$ there exists a constant $B > 0$ such that a 3-CNF formula φ with n' variables can be transformed in time $O(2^{\varepsilon' n'})$ into $s \leq 2^{\varepsilon' n'}$ 3-CNF formulas $\varphi_1, \dots, \varphi_s$ with the same set of variables such that*

- (i) *each variable appears at most B times in each formula φ_i , and*
- (ii) *φ is satisfiable if and only if one of the formulas φ_i is satisfiable.*

The Sparsification Lemma, when combined with ETH and Corollary 12, yields a straightforward proof of Theorem 2:

Proof of Theorem 2. Lemma 14 together with ETH implies that, for some absolute constants $\varepsilon', B > 0$, no algorithm solves 3-CNF instances of the satisfiability problem on n' variables, with each variable appearing at most B times, in time $O(2^{\varepsilon' n'})$. Given such an instance φ with n' variables and m' clauses, we use Corollary 12 with $k = 3$, $p = \lfloor \frac{B}{2} \rfloor$ to transform it, in polynomial time, into a graph $H(\varphi)$ with $|V(H(\varphi))| \leq c \cdot n'$ for some fixed constant c (where n' is sufficiently large). Then φ is satisfiable if and only if $\text{td}(H(\varphi)) = (2^{k+1} - 3)m' + 2^k p n' + 1$. Therefore, under ETH, the treedepth of an n -vertex graph cannot be computed in time $O(2^{\varepsilon n})$ where $\varepsilon := \varepsilon'/(2c)$. ◀

The approximation time complexity lower bound (Theorem 3) requires a bit more work:

Proof of Theorem 3. Assuming ETH, there is some $\lambda > 0$ such that n' -variable 3-SAT cannot be solved in time $O(2^{\lambda n'})$. Pick $0 < \varepsilon' < \min\{0.1, \frac{1}{2}\lambda\}$ and invoke the Sparsification Lemma (Lemma 14). Let the 3-CNF formulas φ and $\varphi_1, \dots, \varphi_s$ be as in the statement of the lemma. Combining the Sparsification Lemma with a polynomial-time SATISFIABILITY inapproximability framework of Håstad [13] and a quasi-linear-size construction of polynomially-checkable proofs (PCP) by Bafna, Minzer, Vyas, and Yun [1], we find absolute constants $0 < \alpha_1 < \alpha_2 < 1$ and $B^*, c > 0$ such that we can translate each formula φ_i in polynomial time to a 3-CNF formula φ_i^* with $n^* \leq n' \log^c n'$ variables, each appearing at most B^* times, with the following property: If φ_i^* has m^* clauses, then:

- If φ_i is satisfiable, then at least $\alpha_2 m^*$ clauses of φ_i^* can be satisfied.
- If φ_i is not satisfiable, then at most $\alpha_1 m^*$ clauses of φ_i^* can be satisfied.

(This reduction is standard; see [5, Lemma 2] for the proof of an analogous result under the assumption of the existence of a linear-size PCP construction.) Chaining this result with Corollary 12, we find absolute constants $0 < \beta_1 < \beta_2 < 1$ and a polynomial-time algorithm that transforms each φ_i^* into a graph H_i with at most dn^* vertices, where d is a fixed constant, such that:

- If φ_i is satisfiable, then $\text{td}(H_i) \leq \beta_1 |V(H_i)|$.
- If φ_i is not satisfiable, then $\text{td}(H_i) > \beta_2 |V(H_i)|$.

Now, let $\varepsilon := \varepsilon'/d$ and $\delta = \beta_2/\beta_1 - 1$. Then, supposing that a $O(2^{\varepsilon n / \log^c n})$ -time $(1 + \delta)$ -approximation algorithm for treedepth existed, the satisfiability of φ could be decided in time

$$O(2^{\varepsilon' n'}) + s \cdot ((n')^{O(1)} + O(2^{\varepsilon d n^* / \log^c n^*})) \leq O(2^{\varepsilon' n'}) + 2^{\varepsilon' n'} \cdot ((n')^{O(1)} + O(2^{\varepsilon' n'})) = O(2^{\lambda n'}),$$

thus refuting the ETH. ◀

References

- 1 Mitali Bafna, Dor Minzer, Nikhil Vyas, and Zhiwei Yun. Quasi-linear size PCPs with small soundness from HDX. In Michal Koucký and Nikhil Bansal, editors, *Proceedings of the 57th Annual ACM Symposium on Theory of Computing, STOC 2025, Prague, Czechia, June 23-27, 2025*, pages 45–53. ACM, 2025. doi:10.1145/3717823.3718197.
- 2 Piotr Berman and Marek Karpinski. Improved approximation lower bounds on small occurrence optimization. *Electron. Colloquium Comput. Complex.*, TR03-008, 2003. URL: <https://eccc.weizmann.ac.il/eccc-reports/2003/TR03-008/index.html>.

- 3 Hans L. Bodlaender, Jitender S. Deogun, Klaus Jansen, Ton Kloks, Dieter Kratsch, Haiko Müller, and Zsolt Tuza. Rankings of graphs. *SIAM J. Discret. Math.*, 11(1):168–181, 1998. doi:10.1137/S0895480195282550.
- 4 Édouard Bonnet. Treewidth inapproximability and tight ETH lower bound. In Michal Koucký and Nikhil Bansal, editors, *Proceedings of the 57th Annual ACM Symposium on Theory of Computing, STOC 2025, Prague, Czechia, June 23–27, 2025*, pages 2130–2135. ACM, 2025. doi:10.1145/3717823.3718117.
- 5 Édouard Bonnet, Bruno Escoffier, Eun Jung Kim, and Vangelis Th. Paschos. On Subexponential and FPT-Time Inapproximability. *Algorithmica*, 71(3):541–565, 2015. doi:10.1007/s00453-014-9889-1.
- 6 Marcin Briański, Gwenaél Joret, Konrad Majewski, Piotr Micek, Michał T. Seweryn, and Roohani Sharma. Treedepth vs circumference. *Comb.*, 43(4):659–664, 2023. doi:10.1007/s00493-023-00028-5.
- 7 Marek Cygan, Fedor V. Fomin, Lukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michał Pilipczuk, and Saket Saurabh. *Parameterized Algorithms*. Springer, 2015. doi:10.1007/978-3-319-21275-3.
- 8 Wojciech Czerwiński, Wojciech Nadara, and Marcin Pilipczuk. Improved bounds for the excluded-minor approximation of treedepth. In Michael A. Bender, Ola Svensson, and Grzegorz Herman, editors, *27th Annual European Symposium on Algorithms, ESA 2019, September 9–11, 2019, Munich/Garching, Germany*, volume 144 of *LIPIcs*, pages 34:1–34:13. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2019. doi:10.4230/LIPIcs.ESA.2019.34.
- 9 Vida Dujmović, Robert Hickingbotham, Gwenaél Joret, Piotr Micek, Pat Morin, and David R. Wood. The excluded tree minor theorem revisited. *Comb. Probab. Comput.*, 33(1):85–90, 2024. doi:10.1017/s0963548323000275.
- 10 Fedor V. Fomin, Pierre Fraigniaud, Pedro Montealegre, Ivan Rapaport, and Ioan Todinca. Distributed model checking on graphs of bounded treedepth. In Dan Alistarh, editor, *38th International Symposium on Distributed Computing, DISC 2024, October 28 to November 1, 2024, Madrid, Spain*, volume 319 of *LIPIcs*, pages 25:1–25:20. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2024. doi:10.4230/LIPIcs.DISC.2024.25.
- 11 Fedor V. Fomin, Archontia C. Giannopoulou, and Michał Pilipczuk. Computing tree-depth faster than 2^n . *Algorithmica*, 73(1):202–216, 2015. doi:10.1007/s00453-014-9914-4.
- 12 Martin Fürer and Huiwen Yu. Space saving by dynamic algebraization based on tree-depth. *Theory Comput. Syst.*, 61(2):283–304, 2017. doi:10.1007/s00224-017-9751-3.
- 13 Johan Håstad. Some optimal inapproximability results. *J. ACM*, 48(4):798–859, 2001. doi:10.1145/502090.502098.
- 14 Meike Hatzel, Gwenaél Joret, Piotr Micek, Marcin Pilipczuk, Torsten Ueckerdt, and Bartosz Walczak. Tight bound on treedepth in terms of pathwidth and longest path. *Comb.*, 44(2):417–427, 2024. doi:10.1007/s00493-023-00077-w.
- 15 Russell Impagliazzo, Ramamohan Paturi, and Francis Zane. Which problems have strongly exponential complexity? *J. Comput. Syst. Sci.*, 63(4):512–530, 2001. doi:10.1006/jcss.2001.1774.
- 16 Ken-ichi Kawarabayashi and Benjamin Rossman. A polynomial excluded-minor approximation of treedepth. *J. Eur. Math. Soc. (JEMS)*, 24(4):1449–1470, 2022. doi:10.4171/JEMS/1133.
- 17 Lukasz Kowalik, Marcin Mucha, Wojciech Nadara, Marcin Pilipczuk, Manuel Sorge, and Piotr Wygocki. The PACE 2020 parameterized algorithms and computational experiments challenge: Treedepth. In Yixin Cao and Marcin Pilipczuk, editors, *15th International Symposium on Parameterized and Exact Computation, IPEC 2020, December 14–18, 2020, Hong Kong, China (Virtual Conference)*, volume 180 of *LIPIcs*, pages 37:1–37:18. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2020. doi:10.4230/LIPIcs.IPEC.2020.37.
- 18 Deepanshu Kush and Benjamin Rossman. Tree-depth and the formula complexity of subgraph isomorphism. *SIAM J. Comput.*, 52(1):273–325, 2023. doi:10.1137/20m1372925.

- 19 Wojciech Nadara, Michał Pilipczuk, and Marcin Smulewicz. Computing treedepth in polynomial space and linear FPT time. In Shiri Chechik, Gonzalo Navarro, Eva Rotenberg, and Grzegorz Herman, editors, *30th Annual European Symposium on Algorithms, ESA 2022, September 5-9, 2022, Berlin/Potsdam, Germany*, volume 244 of *LIPIcs*, pages 79:1–79:14. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022. doi:10.4230/LIPIcs.ESA.2022.79.
- 20 Jesper Nederlof, Michał Pilipczuk, Céline M. F. Swennenhuis, and Karol Węgrzycki. Hamiltonian cycle parameterized by treedepth in single exponential time and polynomial space. *SIAM J. Discret. Math.*, 37(3):1566–1586, 2023. doi:10.1137/22m1518943.
- 21 Jaroslav Nešetřil and Patrice Ossona de Mendez. Grad and classes with bounded expansion i. decompositions. *Eur. J. Comb.*, 29(3):760–776, 2008. doi:10.1016/j.ejc.2006.07.013.
- 22 Michał Pilipczuk and Sebastian Siebertz. Polynomial bounds for centered colorings on proper minor-closed graph classes. *J. Comb. Theory B*, 151:111–147, 2021. doi:10.1016/j.jctb.2021.06.002.
- 23 Michał Pilipczuk and Marcin Wrochna. On space efficiency of algorithms working on structural decompositions of graphs. *ACM Trans. Comput. Theory*, 9(4):18:1–18:36, 2018. doi:10.1145/3154856.
- 24 Alex Pothén. The complexity of optimal elimination trees. *Technical Report, Pennsylvania State University*, 1988. URL: <https://www.cs.purdue.edu/homes/apothén/Papers/shortest-etree1988.pdf>.
- 25 Felix Reidl, Peter Rossmanith, Fernando Sánchez Villaamil, and Somnath Sikdar. A Faster Parameterized Algorithm for Treedepth. In Javier Esparza, Pierre Fraigniaud, Thore Husfeldt, and Elias Koutsoupias, editors, *Automata, Languages, and Programming - 41st International Colloquium, ICALP 2014, Copenhagen, Denmark, July 8-11, 2014, Proceedings, Part I*, volume 8572 of *Lecture Notes in Computer Science*, pages 931–942. Springer, 2014. doi:10.1007/978-3-662-43948-7_77.
- 26 Jaroslav Nešetřil and Patrice Ossona de Mendez. *Sparsity - Graphs, Structures, and Algorithms*, volume 28 of *Algorithms and combinatorics*. Springer, 2012. doi:10.1007/978-3-642-27875-4.