Exponential Separation Between Powers of Regular and General Resolution Over Parities

Sreejata Kishor Bhattacharya*   Arkadev Chattopadhyay†   Pavel Dvořák‡

Abstract

Proving super-polynomial lower bounds on the size of proofs of unsatisfiability of Boolean formulas using resolution over parities is an outstanding problem that has received a lot of attention after its introduction by Raz and Tzamaret [13]. Very recently, Efremenko, Garlík and Itsykson [6] proved the first exponential lower bounds on the size of ResLin proofs that were additionally restricted to be *bottom-regular*. We show that there are formulas for which such regular ResLin proofs of unsatisfiability continue to have exponential size even though there exist short proofs of their unsatisfiability in ordinary, non-regular resolution. This is the first super-polynomial separation between the power of general ResLin and and that of regular ResLin for any natural notion of regularity.

Our argument, while building upon the work of Efremenko et al., uses additional ideas from the literature on *lifting theorems*.

1 Introduction

One of the most basic and well studied proof systems in propositional proof complexity is resolution. Its weakness by now is reasonably well understood after years of research. Yet, this understanding is quite fragile as natural and simple strengthenings of resolution quickly pose challenges that remain outstanding. One such system is resolution over parities, introduced by Raz and Tzameret [13], which has been abbreviated as ResLin. More precisely, a *linear* clause is a disjunction of affine equations, generalizing the notion of ordinary clauses. If A and B are two such linear disjunctions and ℓ is a linear form, then the inference rule of ResLin derives the linear clause $A \vee B$ from clauses $A \vee (\ell = 1)$ and $B \vee (\ell = 0)$. To appreciate the power of this system, let us recall that a linear clause, unlike an ordinary clause, can be expressed using many different bases. Indeed, no super-polynomial lower bounds on the size of general proofs in this system is currently known for any explicit unsatisfiable Boolean formula.

Progress was first made in the work of Itsykson and Sokolov [10] when they proved exponential lower bounds on the size of tree-like ResLin proofs for central tautologies including the Pigeonhole Principle and Tseitin formulas over expanding graphs. Proving such lower bounds was systematized, only recently, in the independent works of Chattopadhyay, Mande, Sanyal and Sherif [4] and that of Beame and Koroth [2]. These works could be used to conclude that tree-like ResLin proofs are exponentially weaker than general ResLin proofs.

In the world of ordinary resolution, it is known that there is an intermediate proof system, known as *regular* resolution, whose power strictly lies in between tree-like and general proofs. In the graph of a regular resolution proof, no derivation path from an axiom clause to the final empty clause resolves a variable of the formula more than once. In the dual view of searching for a falsified clause, this corresponds precisely to read-once branching programs, where no source to sink path queries a variable more than once. Taking cue from this, Gryaznov, Pudlák and Talebanfard [9] introduced models of read-once linear branching programs (ROLBP), to capture notions of regularity in ResLin. They identified two such notions that extend the notion of regularity in ordinary resolution, or the read-once property of branching programs. Consider a node v of an ROLBP. Let $Pre(v)$ denote the vector space spanned by all the linear queries that appear in some path from the source node to v. Similarly, let $Post(v)$ denote the space spanned by all linear queries that lie in some path from v to a sink node. In the most restrictive notion, called strongly regular proofs or strongly read-once linear branching programs, $Pre(v)$

---

and $Post(v)$ have no non-trivial intersection for every v. In a more relaxed notion, called weak regularity or weakly ROLBP, the linear query made at node v is not contained in $Pre(v)$. Gryaznov et al. [9] were able to prove an exponential lower bound on the size of strongly ROLBP for computing a *function*, using the notion of directional affine dispersers. However, their argument is not known to work for search problems for even strongly ROLBP.

There is another natural notion of weakly read-once linear branching programs (dually, weakly regular ResLin) that complements the notion defined in [9]. In this notion, we forbid the linear query made at a node v of the branching program to lie in an affine space $Post(u)$, for each u that is a child of v. We will call this notion bottom-read-once (bottom-regular proofs) and the former notion of Gryaznov et al as top-read-once (top-regular proofs). Both are generalizations of strongly read-once linear branching programs (strongly regular ResLin proofs). Very recently, Efremenko, Garlik and Itsykson [6] proved the first exponential lower bounds on the size of bottom-regular ResLin proofs. The tautology they used was the Binary Pigeonhole Principle (BPHP). It is plausible that the BPHP remains hard for general ResLin proofs. One way to prove such a bound would be to show that every general ResLin proof could be converted to a bottom-regular ResLin proof with a (quasi-)polynomial blow up.

Our main result is a strong refutation of that possibility. We show that bottom-regular ResLin proofs require exponential size blow-up to simulate non-regular proofs even in the ordinary resolution system. The formulas we use are twists of certain formulas used by Alekhnovich, Johannsen, Pitassi and Urquhart [1] for providing separation between regular and general resolution. Alekhnovich et al. provided two formulas for proving such a separation. In the second one, the starting point is a pebbling formula on pyramid graphs. It turns out that they are easy for regular resolution. To get around that, they consider *stone formula* for pyramid graphs, that they prove is hard for regular resolution while remaining easy for general resolution. We further obfuscate such stone formulas using another idea of Alekhnovich et al. that appears in the construction of their first formula. Finally, we *lift* these formulas by logarithmic size Inner-product (IP) gadgets.

This lifted formula presents fresh difficulties to be overcome to carry out the implicit technique of Efremenko et al. We overcome them by exploitng two properties of the Inner-product. First, we exploit the property that IP has low discrepancy, invoking a result of Chattopadhyay, Filmus, Koroth, Meir and Pitassi [3]. Second, we use the fact that IP has the stifling property, inspired by the recent work of Chattopadhyay, Mande, Sanyal and Sherif [4].

Our unsatisfiable formula that yields a separation of resolution and bottom-regular ResLin is a stone formula for a pyramid graph G_n of n levels lifted by an inner-product gadget $\mathsf{IP} : \{0,1\}^b \to \{0,1\}$. We denote this formula as $\mathsf{SP}_n \circ \mathsf{IP}$ and it is defined over $M := 2N^2 \cdot b$ variables, where $b = \Theta(\log n)$ and $N = n(n+1)/2$ is the number of vertices of the pyramid graph G_n. For exact definition of $\mathsf{SP}_n \circ \mathsf{IP}$, see Section 2.

Theorem 1. *The formula* $\mathsf{SP}_n \circ \mathsf{IP}$ *admits a resolution refutation of length that is polynomial in M but any bottom-regular ResLin refutation of it must have length at least* $2^{\Omega(M^{1/12}/\log^{1+\varepsilon} M)}$ *for* $\varepsilon = 1/12$.

Comparision with Efremenko et al. [6]: Our work builds upon the very recent work of Efremenko, Garlík and Itsykson, who proved an exponential lower bound for the regular linear resolution complexity of the formula Binary-PHP$_n^{n+1}$. A crucial property of this formula that they use is that if we sample an assignment to the variables from the uniform distribution, with high probability one needs to make at least $n^{\Omega(1)}$ bit-queries to locate a falsified clause. Later, they use the following simple property of the uniform distribution: let A be an affine subspace of co-dimension r. Then, the probability mass of A under uniform distribution is very small (inverse-exponential in r). Call this property (*).

Our goal is to show an exponential lower bound on the regular linear resolution complexity of a formula that has a small resolution refutation. A candidate formula would be a CNF which exhibits exponential separation between resolution and regular resolution. Some such formulas are $\mathrm{MGT}_{n,\rho}$ and *stone formulas* with auxiliary variables to keep width of clauses short (both defined in Alekhnovich et al. [1]). However, all such formulas have constant width – and therefore, a uniformly random assignment falsifies a constant fraction of clauses. It follows that for both these formulas there is a query algorithm making only constantly many queries which finds a falsified clause with high probability under the uniform distribution. Thus, directly adapting the argument of Efremenko et al. would not work for these formulas.

Our main observation is that property (*) continues to hold for a much larger class of distributions than the uniform distribution when the base formula is *lifted* with an appropriate gadget. More precisely, if we take any distribution μ on the assignments of the base formula and let μ' be its *uniform lift*, property

(*) holds for μ'. We are now free to choose *any* distribution on the assignments of the base formula for which locating a falsifying axiom requires many queries on average (this is just a sketch; we actually need something slightly stronger). This gives us enough freedom to construct appropriate distributions. Some more ingredients are required to make this idea work; we explain them in the subsequent sections.

Some Other Related Work: Following up on the work by Alekhnovich et al [1], Urquhart [17] proved a stronger separation between the length of regular and general resolution proofs. Much more recently, Vinyals, Elffer, Johannsen and Nordström [18], designed a different formula for showing an even stronger separation between regular and general resolutions. The constructions of Urquhart [17] as well as that of Vinyals et al [18] are somewhat related to the hard formulas that we construct in this paper. We talk about them more at the end of Section 2 after describing our construction in detail. In another direction, the model of read-once linear branching programs, introduced by Gryaznov, Pudlák and Talebanfard [9], spawned research in directional affine extractors and pseudo-randomness first by Chattopadhyay and Liao [5], and then further by Li and Zhong [11]. These work on extractors, while independently interesting, are not known to have consequences for ResLin.

Organization of the Paper: We present our hard formula in the next section and briefly compare it with constructions done in earlier work. In Section 3, we define ResLin refutation system and its regular and tree-like restrictions. Further, we present the connection between resolution proof systems and branching programs. In Section 4, we present some results from linear algebra which we use in our proofs. In Section 5, we sketch the outline and main ideas of the proof of our main result, Theorem 1. Then in Section 6, we prove the upper bound part of Theorem 1, i.e., our hard formula has short resolution refutation. We finish our proof in Section 7 where we prove the lower bound part of Theorem 1, i.e., any bottom-regular ResLin refutation of our hard formula must have exponential length. Finally, in Section 8, we conclude with some of the many open problems that our work raises.

2 A Formula Hard For Just Regular ResLin

Let us first recall the stone formula that was used by Alekhnovich et al. [1] for separating the powers of regular and general resolution. The formula that we shall use is a *lift* of this formula by an appropriate gadget. Let $G = (V, E)$ be a directed acyclic graph such that it has exactly one root (vertex with indegree 0), r, and every vertex of G has outdegree either 0 or 2. Call the vertices with outdegree 0 the sinks of G. Let $|V| = N$. We describe the *stone formula on G twisted with ρ*, $\text{Stone}(G, \rho)$ below. In words, the contradiction we are about to describe states the following:

- There are $|V|$ stones. Each stone has a color: red or blue.

- At least one stone must be placed on each vertex.

- All stones placed on sinks must be red.

- All stones placed on the root must be blue.

- Let v be a node with out-neighbors u, w. If a red stone j is placed on u and a red stone k is placed on w, then all stones placed on v must be red.

We shall twist this formula with an obfuscation map ρ to make it hard for regular resolution. We formally define the formula below. We introduce the following set of variables.

Vertex variables: For all $v \in V, 1 \leq j \leq N : P_{v,j}$.
    Semantic interpretation: $P_{v,j}$ is set to 1 iff stone j is placed on vertex v.

Stone variables: For all $1 \leq j \leq N : R_j$.
    Semantic interpretation: R_j is set to 1 if stone j is colored red, otherwise it is set to 0.

Auxiliary variables: For all $v \in V, 1 \leq j \leq N - 1 : Z_{v,j}$.
    Semantic interpretation: These are auxiliary variables used to encode the fact that at least one stone is placed on vertex v, with a bunch of constant-width clauses.

Let $\mathcal{V}$ denote the set of all variables mentioned above and $\rho : [N]^3 \to \mathcal{V}$ be an arbitrary mapping that we call an *obfuscation map*. Let S be the set of sinks of G. We define $\text{Stone}(G, \rho)$ to be the formula comprising the following set of clauses:

Root clauses: For all $1 \leq j \leq N$: $\neg P_{r,j} \vee \neg R_j$

Semantic interpretation: All stones placed on the root r of G must be coloured blue.

Sink clauses: For all $1 \leq j \leq N, s \in S$: $\neg P_{s,j} \vee R_j$

Semantic interpretation: Each stone placed on a sink of G must be coloured red.

Induction clauses: For all $v \in V(G)$ with out-neighbors u, w and for each $i, j, k \in [N]$:

$$\neg P_{u,i} \vee \neg R_i \vee \neg P_{w,j} \vee \neg R_j \vee \neg P_{v,k} \vee R_k \vee \rho(i,j,k)$$

$$\neg P_{u,i} \vee \neg R_i \vee \neg P_{w,j} \vee \neg R_j \vee \neg P_{v,k} \vee R_k \vee \neg\rho(i,j,k)$$

Semantic interpretation: after resolving out the variable $\rho(i,j,k)$, the clause says that if the stones placed on u and w are colored red, the stone placed at v must also be colored red, i.e., the implication

$$(P_{u,i} \wedge R_i \wedge P_{w,j} \wedge R_j \wedge P_{v,k}) \implies R_k.$$

Stone-placement clauses: For all $v \in V(G)$:

$$P_{v,1} \vee \neg Z_{v,1}$$
$$Z_{v,1} \vee P_{v,2} \vee \neg Z_{v,2}$$
$$\cdots$$
$$Z_{v,N-2} \vee P_{v,N-1} \vee \neg Z_{v,N-1}$$
$$Z_{v,N-1} \vee P_{v,N}$$

Semantic interpretation: Together, the clauses are equivalent to

$$P_{v,1} \vee P_{v,2} \vee \cdots \vee P_{v,N}$$

i.e., at least one stone is placed on the vertex v.

Let G_n be the pyramid graph on n levels. The vertex set is $V = \{(i,j) | 1 \leq i \leq n, 1 \leq j \leq i\}$. The *level* of a vertex (i,j) is defined to be its first coordinate i. The edge set is $E = \{(i,j) \to (i+1,j) | 1 \leq i < n, 1 \leq j \leq i\} \cup \{(i,j) \to (i+1,j+1) | 1 \leq i < n, 1 \leq j \leq i\}$. See Figure 1, for an example of the pyramid graph. The sinks of G_n are the vertices at layer n, i.e., (n,i) for $1 \leq i \leq n$. The root is $(1,1)$. We have $|V| = N = \frac{1}{2}n(n+1)$.
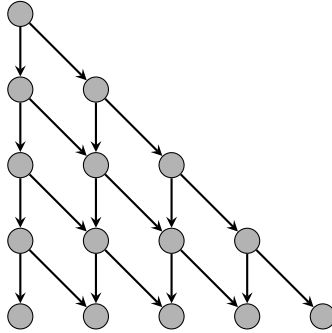

Figure 1: An example of the pyramid graph with $n = 5$ levels.

We instantiate the stone formula with $G = G_n$. Let $\mathsf{SP}_{n,\rho} = \mathrm{Stone}(G_n, \rho)$. We denote the number of variables of $\mathsf{SP}_{n,\rho}$ by m, i.e. $|\mathcal{V}| = m = N^2 + N + N(N-1) = 2N^2$. In order to prove our lower bound against regular ResLin, it turns out to be convenient working with a formula that is obtained by *lifting* $\mathsf{SP}_{n,\rho}$. Let $g : \{0,1\}^b \to \{0,1\}$ be a Boolean function, called gadget. For $a \in \{0,1\}$, we denote the set of all pre-images of a by $g^{-1}(a)$, i.e., $g^{-1}(a) := \{x \in \{0,1\}^b \mid g(x) = a\}$.

Let $c_1, c_2, \cdots, c_k \in \{0,1\}$. Let $C = [X_1 = c_1] \vee \cdots \vee [X_k = c_k]$ be a clause over variables $X_1, \ldots, X_k$. Here $[X_i = c_i]$ denotes a literal, i.e. X_i if $c_i = 1$, and $\neg X_i$ otherwise. To lift Clause C we introduce b variables $Y_1^i, \ldots, Y_b^i$ for each variable X_i of C. The lift of C, $C \circ g$, is a set of clauses which, in conjunction, are semantically equivalent to $[g(Y_1^1, \cdots, Y_b^1) = c_1] \vee \cdots \vee [g(Y_1^k, \cdots, Y_b^k) = c_k]$, i.e., the following:

$$C \circ g := \left\{ \bigvee_{1 \leq i \leq k, 1 \leq j \leq b} [Y_j^i = 1 - d_j^i] \;\middle|\; d^1 \in g^{-1}(1 - c_1), \ldots, d^k \in g^{-1}(1 - c_k) \right\},$$

where each d^i is a b-bit string and d_j^i is its j-th bit.

Observation 1. *An assignment $(Y_j^i)_{1 \leq i \leq k, 1 \leq j \leq b}$ satisfies every clause in $C \circ g$ if and only the lifted assignment $(x_1, x_2, \cdots, x_k)$ given by*

$$x_1 = g(Y_1^1, \cdots, Y_b^1), \cdots, x_k = g(Y_1^k, \cdots, Y_b^k)$$

satisfies the clause C.

For a technical reason, we shall also need the following simple observation:

Observation 2. *Let ψ be any clause of $C \circ g$. Suppose, one of the variables Y_j^i appears in ψ. Then, for all $k \in [b]$, the variable Y_k^i also appears in ψ.*

For a set of clauses Φ (a CNF formula), we define its lift as $\Phi \circ g := \cup_{C \in \Phi}(C \circ g)$. We have the following corollary of Observation 1.

Corollary 1. *The set of clauses Φ is unsatisfiable if and only if the set of clauses $\Phi \circ g$ is unsatisfiable.*

We remark that if the base set Φ contains only clauses of width at most k, then $\Phi \circ g$ contains clauses of width at most kb and $|\Phi \circ g| \leq 2^{bk} \cdot |\Phi|$. In particular, a constant-width, poly-size unsatisfiable formula, when lifted by an $O(\log n)$ size gadget, yields an $O(\log n)$-width, poly-size unsatisfiable formula. Our hard formula will be the stone formula lifted by an inner product gadget $\mathsf{SP}_{n,\rho} \circ \mathsf{IP}$, where $\mathsf{IP} : \{0,1\}^b \to \{0,1\}$ is the inner-product function, i.e. for each $x, y \in \{0,1\}^{b/2}$, set $\mathsf{IP}(x,y) = x_1 y_1 + \cdots + x_{b/2} y_{b/2} \pmod 2$ where $b = O(\log n)$ is an even integer.

Comparison with related formulas from previous work: We briefly mention some formulas that were used in the past, that are related to $\mathrm{Stone}(G_n, \rho)$. First, the formula sans obfuscation $\mathrm{Stone}(G_n)$ was one of the formulas used by Alekhnovich et al. [1] to yield the first exponential separation between regular and general resolution. This separation was further strengthened by Urquhart [17] in a follow-up work, using the following more involved formula. Fix a bijective placement of stones to the vertices of G_n. This reduces $\mathrm{Stone}(G_n)$ to a plain pebbling formula on pyramid graph, denoted by $\mathrm{Peb}(G_n)$. This reduces the number of clauses to linear in N. Urquhart considers the 2-bit XOR lift of such a formula, i.e. $\mathrm{Peb}(G_n) \circ \oplus$. This formula blows up the number of clauses, but still keeps the number of variables to $O(N)$. Finally, he shows that for a suitable ρ, the obfuscation of $\mathrm{Peb}(G_n) \circ \oplus$ by ρ, just as we obfuscate $\mathrm{Stone}(G_n)$ by $\mathrm{Stone}(G_n, \rho)$, yields a separation between regular and general resolution that is stronger than the one by Alekhnovich et al. [1]. More recently, Vinyals et al. [18] worked with a different *sparsification* of $\mathrm{Stone}(G_n)$. This comes about naturally by considering $\mathrm{Stone}(G_n)$ as a densification of $\mathrm{Peb}(G_n)$ by using a complete bi-partite graph with N vertices on each side. Roughly speaking, Vinyals et al. used a constant degree bi-partite expander gadget with $\mathrm{Peb}(G_n)$, inspired by the earlier work of Razborov [15]. This results in more modular and optimal arguments. However, the lift by a stifled gadget of a base stone formula, like we do as in $\mathrm{Stone}(G_n, \rho) \circ \mathsf{IP}$, seems not to have been considered earlier.

3   Resolution Proof Systems and Branching Programs

A proof in a propositional proof system starts from a set of clauses Φ, called axioms, that is purportedly unsatisfiable. It generates a proof by deriving the empty clause from the axioms, using inference rules. The main inference rule in the standard resolution, called the resolution rule, derives a clause $A \vee B$ from clauses $A \vee x$ and $B \vee \neg x$ (i.e., we resolve the variable x). If we can derive the empty clause from the original set Φ then it proves the set Φ is unsatisfiable. We will need the following basic and well known fact that states resolution is complete without being too inefficient.

Lemma 1. *Let C be any clause, and Φ be any CNF formula over n Boolean variables and of polynomial size, that semantically implies C. Then, C can be derived from Φ by a resolution proof of size at most $2^{O(n)}$.*

Linear resolution, aka ResLin and introduced by Raz and Tzamaret [13], is a generalization of standard resolution, using linear clauses (disjunction of linear equations over $\mathbb{F}_2$) to express lines of a proof. It consists of two rules:

Resolution Rule: From linear clauses $A \vee (\ell = 0)$ and $B \vee (\ell = 1)$ derive a linear clause $A \vee B$.

Weakening Rule: From a linear clause A derive a linear clause B that is semantically implied by A (i.e., any assignment satisfying A also satisfies B).

The length of a resolution (or ResLin) refutation of a formula Φ is the number of applications of the rules above in order to refute the formula Φ. The width of a resolution (or ResLin) refutation is the maximum width of any (linear) clause that is used in the resolution proof.

It is known that a resolution proof and a linear resolution proof, for an unsatisfiable set of clauses Φ, correspond to a branching program and a linear branching program, respectively, for a search problem $Search(\Phi)$ (see for example Garg et al [7], who credit it to earlier work of Razborov [14] that was simplified by Pudlák [12] and Sokolov[16]) that is defined as follows. For a given assignment α of the n variables of Φ, one needs to find a clause in Φ that is unsatisfied by α (at least one exists as the set Φ is unsatisfiable). A linear branching program computing a search problem $\mathsf{P} \subseteq \mathbb{F}_2^n \times O$ is defined as follows.

- There is a directed acyclic graph $\mathcal{P}$ of one source and some sinks. Each non-sink node has out-degree at most two. For an inner node v the two out-neighbors u and w (i.e., there are edges (v, u) and (v, w) in $\mathcal{P}$) are called children of v.

- Each node v of $\mathcal{P}$ is labeled by an affine space $A_v \subseteq \mathbb{F}_2^n$.

- The source is labeled by $\mathbb{F}_2^n$.

- Let v be a node of out-degree 2 and u and w be children of v. Then, $A_u = A_v^0$ and $A_w = A_v^1$, where $A_v^c = \{x \in A_v \mid \langle f_v, x \rangle = c\}$ for a linear query $f_v = \mathbb{F}_2^n$ and $c \in \{0, 1\}$. We call such v a *query node*.

- Let v be a node of out-degree 1 and u be the child of v. Then, $A_v \subseteq A_u$. We call such v a *forget node*.

- Each sink v of $\mathcal{P}$ has an assigned output $o_v \in O$ such that A_v is o_v-monochromatic according to P, i.e., $\alpha \in A_v \implies (\alpha, o_v) \in \mathsf{P}$.

A standard/ordinary branching program is defined analogously but its nodes are labeled by cubes instead of affine spaces. Consequently, variables instead of arbitrary linear functions are queried at its query nodes.

The correspondence between a branching program computing $Search(\Phi)$ and a (linear) resolution proof refuting Φ is roughly the following. We can represent the resolution proof as a directed acyclic graph where nodes are labeled by (linear) clauses. The sources are labeled by clauses of Φ and there is exactly one sink that is labeled by an empty clause. Each node that is not a source has at most two parents and it corresponds to an application of the (linear) resolution rule (if the node has 2 parents), or the weakening rule (if the nodes has 1 parent). To get a (linear) branching program for $Search(\Phi)$ we just flip the direction of the edges in the resolution graph and negate the clauses that are used for node labeling. Thus, each node is labeled by a cube or an affine space, the query nodes correspond to applications of the resolution rule, and the forget nodes correspond to applications of the weakening rule. It is clear the size of a branching program $\mathcal{P}$ (number of nodes of $\mathcal{P}$) is exactly the same as length of the corresponding resolution refutation.

Regular resolution is a subsystem of the resolution system, such that in any path of the resolution proof graph each variable can be resolved at most once. A read-once branching program corresponds to a regular resolution proof, i.e., on each directed path from the source to a sink each variable is queried at most once. There is interest in two generalizations of regular resolution to linear regular resolution – top-regular linear resolution [9] and bottom-regular linear resolution [6] (in both papers called as regular linear resolution). We will define both of them by their corresponding linear branching programs.

Definition 1 ([9]). *Let v be a node of a linear branching program $\mathcal{P}$. Let $Pre(v)$ be the space spanned by all linear functions queried on any path from the source of $\mathcal{P}$ to v. Let $Post(v)$ be the space spanned by all linear functions queried on any path from v to any sink of $\mathcal{P}$.*

A linear branching program is *top-read-once*[1] [9] if for each query node v, we have $f_v \notin Pre(v)$. A linear branching program is *bottom-read-once* [6] if for each edge (v, u) such that v is a query node holds that $f_v \notin Post(u)$. A linear resolution proof is *top-regular*, or *bottom-regular* if the corresponding branching program is top-read-once, or bottom-read-once, respectively. We use both notion of branching program and resolution to state and prove our result, whichever is more suitable for the presentation at hand. Our separation is only for bottom-regular ResLin, i.e., for bottom-read-once linear branching program, which we abbreviate to BROLBP.

Lemma 2 (Lemma 2.6 [6] stated for branching programs). *Let $\mathcal{P}$ be a BROLBP computing a search problem $\mathsf{P} \subseteq \{0,1\}^n \times O$. Let v be a node of $\mathcal{P}$ such that there is a path of length t from the source of $\mathcal{P}$ to v. Then, $\dim(Post(v)) \leq n - t$.*

Lemma 3 (Lemma 2.3 [6] stated for branching programs). *Let $\mathcal{P}$ be a linear branching program computing a search problem $\mathsf{P} \subseteq \{0,1\}^n \times O$. Let u and v be nodes of $\mathcal{P}$ such that there is a directed path p from u to v. Let $(M|c)$ be the system of linear equations given by queries along the path p and A be the affine space of solution of $(M|c)$, i.e., $A = \{\alpha \in \{0,1\}^n \mid M\alpha = c\}$. Then, for A_u and A_v the affine spaces associated with u, and v, respectively, holds that $A_u \cap A \subseteq A_v$.*

Even more restrictive subsystems, are tree-like resolution, and tree-like ResLin. These subsystems correspond to decision trees and parity decision trees. A parity decision tree (PDT) is a linear branching program such that its underlying graph is a tree, and a decision tree (DT) is a restriction where we query only bits of the input, instead of linear functions. It is clear that tree-like resolution is a subsystem of regular resolution. Analogously, tree-like ResLin is a subsystem of both bottom-regular and top-regular ResLin.

It is easy to see that strongly read-once linear branching programs are both top-read-once and bottom-read-once. Chattopadhyay and Liao [5] showed that strongly read-once linear branching programs can simulate parity decision trees. A super-polynomial separation between tree-like ResLin and bottom-regular ResLin follows from the lifting theorems in Chattopadhay et al [3] as well as in Beame and Koroth [2]. We have the following containments:
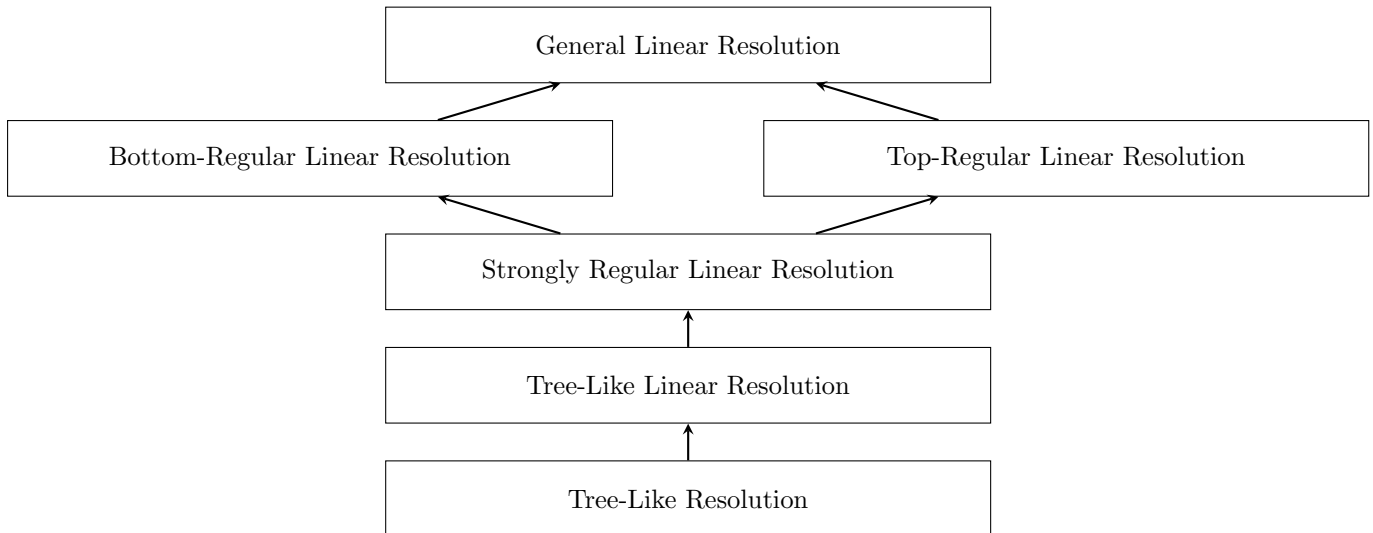
In this paper we show the existence of a CNF formula with a polynomial sized resolution refutation for which any bottom-regular linear resolution refutation requires exponential size. Thus, we show that the containment *bottom-regular linear resolution $\subseteq$ general ResLin* is strict.

3.1 ROLBP Computing Boolean Function

In the context of computing Boolean functions, branching programs are defined, usually, in a more relaxed fashion in a certain sense. For instance, ordinary branching programs are defined without placing the restriction that the set of inputs reaching a node can be contained in a non-trivial sub-cube. This is something we insist when we define BPs here as our focus is on capturing the limitations of those BPs that

---

[1]Gryaznov et al. [9] used just the name weakly read once for such programs.

are derived from a resolution proof DAG by reversing the direction of its edges. It, possibly, would have been more meaningful to call these latter objects affine DAGs, but we chose to call it linear branching programs for the sake of continuity wrt the earlier works by Efremenko et al. [6] and Gryaznov et al. [9]. In this section, we take the liberty of indeed calling them DAGs to compare them with BPs. Affine DAGs severely restrict the power of computing Boolean functions. This is because the set of sink nodes of such a DAG of small size computing a Boolean function f simply provides an efficient affine cover of $f^{-1}(0)$ as well as $f^{-1}(1)$. Thus, immediately, one concludes that any affine DAG, without any restriction on the number of reads of a variable, computing the Inner-product on n bits requires $2^{\Omega(n)}$ size[2] as IP has large affine cover number. On the other hand, IP can be easily seen to be computed by a linear-size read-once and bit-querying branching program.

On the other hand, the situation for problems of searching a falsified clause, is quite different. As Lemma 2.4 in the work of Efremenko et al. [6] proves, for any unsatisfiable CNF Ψ, any top-read-once linear branching program solving $Search(\Psi)$ gives rise to a top-read-once affine DAG for $Search(\Psi)$ with hardly any blow-up in its size. The proof can be easily verified to additionally yield that a strongly read-once linear BP for $Search(\Psi)$, completely analogously, yields a strongly read-once affine DAG for $Search(\Psi)$ with no essential blow-up to its size. Thus, our main result, Theorem 1 holds equally for strongly read-once linear branching programs that are not restricted by definition to be affine DAGs.

4 Linear Algebraic Facts

In this section, we will describe the notions of linear algebra that we will need in our arguments. Let us introduce some notation first. Let $M \in \mathbb{F}_2^{t \times m}$ be a matrix. We denote the row space of M by $\mathcal{R}(M)$. For a vector $c \in \{0,1\}^t$, $\mathcal{S}(M,c)$ is the affine space of solutions to the linear system $(M|c)$, i.e., $\mathcal{S}(M,c) = \{\alpha \in \{0,1\}^m \mid M \cdot \alpha = c\}$.

The entries of vectors of $\mathbb{F}_2^{mb}$ are naturally divided into m blocks, each having b co-ordinates/bits, i.e., for $j \in [m]$, the j-th block contains the coordinates $(j-1)b+1, \ldots, jb$. For $j \in [m]$, $\mathrm{BLOCK}(j) = \{(j-1)b+1, \cdots, jb\}$. Also for $T \subseteq [m]$ define $\mathrm{BLOCK}(T) = \cup_{t \in T} \mathrm{BLOCK}(t)$. For a vector $u \in \mathbb{F}_2^{mb}$ and a block $j \in [m]$, $u^j \in \mathbb{F}_2^b$ is the vector corresponding to the block j of u, i.e., $u^j = (u_{(j-1)b+1}, \ldots, u_{jb})$. We say a vector $u \in \mathbb{F}_2^{mb}$ touches a block $j \in [m]$ if the vector u^j is non-zero. A set of vectors $R \subseteq \mathbb{F}_2^{mb}$ touches a block j if at least one of the vectors in R touches j. Let U be a subspace of $\mathbb{F}_2^{mb}$ and $T \subseteq [m]$ be a set of blocks. The subspace U_T of U is the linear space of all vectors u that do not touch any block outside T, i.e., $U_T = \{u \in U \mid \forall j \notin T : u^j = (0, \ldots, 0)\}$. For $S = \mathrm{BLOCK}(T)$, the subspace $U_{\downarrow T}$ of $\mathbb{F}_2^S$ is the projection of U onto T, i.e., $U_{\downarrow T} = \{x \in \mathbb{F}_2^S \mid \exists y \in \mathbb{F}_2^{[mb] \setminus S} : (x,y) \in U\}$. We call a tuple of vectors $R = (u_1, \ldots, u_t), u_i \in \mathbb{F}_2^{mb}$ to be safe if the following condition holds:

- The vectors $(u_1, \ldots, u_t)$ form a matrix $M \in \mathbb{F}^{t \times mb}$ in echelon form, i.e., there are t distinct coordinates $a_1, \ldots, a_t \in [mb]$ such that for all $i, j \in [t]$:

$$(u_i)_{a_j} = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{otherwise} \end{cases}$$

  In other words, the matrix M restricted to the columns $a_1, \ldots, a_t$ is the identity matrix $I_t \in \mathbb{F}_2^{t \times t}$.

- Moreover, each pivot a_i lies in a distinct block.

The a_i's are called the pivot variables of R. There might be multiple possible choices for the tuple of pivot variables $(a_1, \ldots, a_t)$. In that case we pick any valid choice, say the lexicographically smallest valid choice, and call it the set of pivots.

A subspace U of $\mathbb{F}_2^{mb}$ is spread if any set of k linearly independent vectors of U touches at least k blocks, for each $1 \leq k \leq m$. We say a set of blocks $T \subseteq [m]$ is an obstruction of a space U if $U_{\downarrow \bar{T}}$ is spread, where $\bar{T}$ is complement of T, i.e., $\bar{T} = [m] \setminus T$. An obstruction $T \subseteq [m]$ of a space U is minimal if any proper subset $T' \subset T$ is not an obstruction of U, i.e., $U_{\downarrow \bar{T}'}$ is not spread. Efremenko et al. [6] showed the following result.

Theorem 2 (Theorem 3.1, Efremenko et al. [6]). *Let U be a spread subspace of $\mathbb{F}_2^{mb}$ with $dim(U) \leq m$. Then, U has a safe basis.*

---

[2]In fact, the case for cube DAGs is known to be more dramatic. If a Boolean function f can be computed by a cube DAG of size s, then it can be also computed by a decision tree of size $s^{O(\log(s) \cdot \log n)}$.

The following is a basic fact.

Observation 3. *Let U be any subspace of $\mathbb{F}_2^{mb}$ and $T \subseteq [m]$ be a set of blocks. Then,*

$$dim(U) = dim(U_T) + dim\big(U_{\downarrow\bar{T}}\big).$$

Proof. Let U' be a suitable subspace of U such that $U = U_T \oplus U'$. It is simple to see that $U_{\downarrow\bar{T}} = (U')_{\downarrow\bar{T}}$. This is because every vector $u \in U$ can be written as $x + u'$, with $x \in U_T$ and $u' \in U'$. But, as $x_{\downarrow\bar{T}} = 0$, we have $u_{\downarrow\bar{T}} = u'_{\downarrow\bar{T}}$. Hence, we conclude that $\dim\big(U_{\downarrow\bar{T}}\big) = \dim\big((U')_{\downarrow\bar{T}}\big) \leq \dim\big(U'\big)$. To establish our result, we will simply show that $\dim(U') \leq \dim\big((U')_{\downarrow\bar{T}}\big)$. This follows if we show that whenever $u'_1, \ldots, u'_r \in U'$ are linearly independent vectors, so are $\big(u'_1\big)_{\downarrow\bar{T}}, \ldots, \big(u'_r\big)_{\downarrow\bar{T}}$. If that is not the case then there exists a vector $x \in U_T$ such that $x, u'_1, \ldots, u'_r$ are not linearly independent, contradicting our assumption. $\square$

Efremenko et al. [6] showed the following properties of minimal obstructions.

Lemma 4. *Let U be a subspace of $\mathbb{F}_2^{mb}$. Then, a minimal obstruction $T \subseteq [m]$ of U is unique and $|T| \leq \dim(U)$.*

Definition 2. *For an affine space $A = \mathcal{S}(M, c) \subseteq \mathbb{F}_2^{mb}$ we define its closure $Cl(A) \subseteq [m]$ to be its unique minimal obstruction. Also, define $VarCl(A) \subseteq [mb]$ to be the set of variables that appear in the blocks of $Cl(A)$, i.e.,*

$$VarCl(A) = BLOCK(Cl(A)).$$

Efremenko et al. [6] proved the following relationship between the closures of two affine spaces when one contains the other.

Lemma 5. *Let $A \subseteq A'$ be two affine spaces of $\mathbb{F}_2^{mb}$. Then, $Cl(A') \subseteq Cl(A)$.*

A *partial assignment* α' of m variables is a string in $\{0, 1, *\}^m$. A variable $X \in [m]$ is *assigned* if $\alpha_X \in \{0, 1\}$. For a total assignment $\alpha \in \{0, 1\}^m$ and $T \subseteq [m]$ we define the restriction $\alpha_{|T}$ of α to T to be the partial assignment arising from α by unassigning the variables that are not in T, i.e., for each $i \in [m]$

$$\big(\alpha_{|T}\big)_i = \begin{cases} \alpha_i & \text{if } i \in T, \\ * & \text{otherwise.} \end{cases}$$

We describe the notion of *stifling* introduced by Chattopadhay et al. [4].

Definition 3. *A Boolean function $g : \{0, 1\}^b \to \{0, 1\}$ is* stifled[3] *if the following holds*

$$\forall i \in [b] \text{ and } a \in \{0, 1\} \ \exists \delta \in \{0, 1\}^b$$
$$\text{such that for all } \gamma \in \{0, 1\}^b \text{ with } \gamma_{|[b]\setminus\{i\}} = \delta_{|[b]\setminus\{i\}} \text{ holds that } g(\gamma) = a.$$

We call δ from the previous definition a *stifling assignment* for i and a. The utility of stifling is the following. An adversary can pick any variable $i \in [b]$ of g. For any $a \in \{0, 1\}$, we can pick a partial assignment $\delta_a \in \{0, 1, *\}^m$ that assigns a value to all variables except the i-th variable. Now, no matter how the adversary chooses the value for the i-th variable to get a total assignment $\gamma_a \in \{0, 1\}^b$ from δ_a, the value $g(\gamma_a)$ will be always a.

Definition 4. *A partial assignment $\beta \in \{0, 1, *\}^{mb}$ is called* block-respecting *if for each block $j \in [m]$, either all variables or no variables are assigned, i.e.,*

$$(\beta^j)_i \in \{0, 1\} \text{ for all } i \in [b] \text{ or } (\beta^j)_i = * \text{ for all } i \in [b].$$

A block-respecting assignment $\beta \in \{0, 1, *\}^{mb}$ naturally gives a partial assignment $\overrightarrow{g}(\beta) \in \{0, 1, *\}^m$ by applying the gadget g to the assigned blocks. Formally, for each $j \in [m]$ we have

$$\overrightarrow{g}(\beta)_j = \begin{cases} g(\beta_1^j, \ldots, \beta_b^j) & \text{if for all } i \in [b] : \beta_i^j \text{ are assigned,} \\ * & \text{otherwise.} \end{cases}$$

---

[3]1-stifled called by Chattopadhyay et al. [4]

Definition 5. *Let $A \subseteq \mathbb{F}^{mb}$ be an affine space and $\beta \in A$. The* closure-assignment *of β, $\beta|_{VarCl(A)}$ is the partial assignment which fixes all coordinates in blocks of $Cl(A)$ according to β and keeps other coordinates free. In other words,*

$$(\beta|_{VarCl(A)})^j = \begin{cases} \beta^j & \text{if } j \in Cl(A), \\ (*, \ldots, *) & \text{otherwise.} \end{cases}$$

Lemma 6. *Let $A = \mathcal{S}(M, c) \subseteq \mathbb{F}^{mb}$ be an affine space and let $g : \{0,1\}^b \to \{0,1\}$ be a stifled gadget. Let $\beta \in A$ be a vector and $\beta' \in \{0,1,*\}^{mb}$ be its closure assignment. Let $\alpha' := \overrightarrow{g}(\beta') \in \{0,1,*\}^m$. Then, for any extension of α' to a total assignment $\alpha \in \{0,1\}^m$, there exists $\gamma \in A$ such that $\overrightarrow{g}(\gamma) = \alpha$.*

Proof. WLOG assume the rows of M are linearly independent. Let $U = \mathcal{R}(M)$ be the row-space of M and let $T \subseteq [m]$ be the closure of A. First, we construct a matrix M' which has the same row-space as M.

Construction of M'

1. Let $(u_1, \ldots, u_d)$ be an arbitrary basis of U_T and let $M_1 \in \mathbb{F}^{d \times mb}$ be the matrix whose rows are the vectors $u_1, \ldots, u_d$:

$$M_1 = \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_d \end{bmatrix}$$

2. Let $(w_1, \ldots, w_{d'})$ be a safe basis of $U_{\downarrow \bar{T}}$. Such a basis exists by the definition of closure and Theorem 2. Let $a_1, \ldots, a_{d'}$ be pivots of $w_1, \ldots, w_{d'}$. Each of these pivots lie in a distinct block. Moreover, none of these pivots are in the blocks of T.

   Let $L : U \to U_{\downarrow \bar{T}}$ be the projection of U to $U_{\downarrow \bar{T}}$. Let w'_i be an arbitrary pre-image of w_i according to L, i.e., $L(w'_i) = w_i$. Since $(w_1, \ldots, w_{d'})$ are linearly independent, the vectors $(w'_1, \ldots, w'_{d'})$ are linearly independent as well. Let $M_2 \in \mathbb{F}^{d' \times mb}$ be the matrix with the vectors $w'_1, \ldots, w'_{d'}$ as its rows.

$$M_2 = \begin{bmatrix} w'_1 \\ w'_2 \\ \vdots \\ w'_{d'} \end{bmatrix}$$

3. Take $M' \in \mathbb{F}^{(d+d') \times mb}$ to be the matrix obtained by stacking M_1 on top of M_2:

$$M' = \begin{bmatrix} M_1 \\ M_2 \end{bmatrix}$$

Claim 1. *The matrices M and M' have the same row-space.*

Proof. By Observation 3, $\dim(U) = \dim(U_T) + \dim(U_{\downarrow \bar{T}}) = d + d'$. No row of M_2 can be generated by rows of M_1 as the pivots $a_1, \ldots, a_{d'}$ of the matrix M_2 lie in columns where the matrix M_1 has only 0 entries. Thus, $\mathrm{rank}(M') = \mathrm{rank}(M_1) + \mathrm{rank}(M_2) = d + d' = \dim(U) = \mathrm{rank}(M)$. Moreover, any row of M' lies in $U = \mathcal{R}(M)$. It follows that $\mathcal{R}(M) = \mathcal{R}(M')$. $\square$

Thus, there is a vector $c' \in \mathbb{F}^{mb}$ such that $A = \mathcal{S}(M', c')$. Now, we prove the lemma. We are given a vector $\beta \in \mathcal{S}(M', c')$ and a target assignment $\alpha \in \{0,1\}^m$ such that $g(\beta^j) = \alpha_j$ for all $j \in Cl(A)$. Our goal is to show the existence of a $\gamma \in \mathbb{F}^{mb}$ such that $\overrightarrow{g}(\gamma) = \alpha$ and $M'\gamma = c'$. Recall that the tuple of rows of M_2, $(w'_1, \ldots, w'_{d'})$ is a safe tuple with set of pivots $a_1, \ldots, a_{d'}$. Suppose $a_j \in \mathrm{BLOCK}(b_j)$. The blocks $b_1, \ldots, b_{d'}$ are distinct and all of them lie in $[m] \setminus T$. Let $\mathrm{PIVOTS} = \{b_1, b_2, \cdots, b_{d'}\}$ and $\mathrm{FREE} = [m] \setminus (T \cup \mathrm{PIVOTS})$. We construct γ in two steps. In the first step we construct a $\tilde{\beta} \in \mathbb{F}^{mb}$ such that $\overrightarrow{g}(\tilde{\beta}) = \alpha$, but it is not necessarily the case that $M'\tilde{\beta} = c'$. In the second step we modify $\tilde{\beta}$ in the coordinates $a_1, \ldots, a_{d'}$ to get an assignment $\gamma \in \mathcal{S}(M', c')$.

Constructing $\tilde{\beta}$

- For each $i \in T = \mathrm{Cl}(A)$, $\tilde{\beta}$ agrees with β on $\mathrm{BLOCK}(i)$, i.e., $(\tilde{\beta})^i = \beta^i$.

- For each $i \in \mathrm{FREE}$, choose an arbitrary preimage $u_i \in g^{-1}(\alpha_i)$ and set $(\tilde{\beta})^i = u_i$.

- For each $i = b_j \in \mathrm{PIVOT}$: Suppose the pivot a_j is the ℓ-th coordinate of $\mathrm{BLOCK}(j)$. Pick $u_i \in g^{-1}(\alpha_i)$ to be a stifling assignment for the ℓ-th coordinate, i.e., $g(u_i) = g(u_i^{(l)}) = \alpha_i$ (where $s^{(l)}$ denotes s with l-th coordinate flipped). Set $(\tilde{\beta})^i = u_i$.

Constructing γ: We modify $\tilde{\beta}$ in the coordinates $a_1, \ldots, a_{d'}$ to get an assignment γ in $\mathcal{S}(M', c')$ as follows. For $1 \leq j \leq d$, let $f_j = \langle w'_j, \tilde{\beta} \rangle + (c')_j$. Let $\gamma \in \mathbb{F}^{mb}$ be the following assignment:

$$\gamma_i = \begin{cases} (\tilde{\beta})_i & \text{if } i \notin \{a_1, \ldots, a_{d'}\}, \\ (\tilde{\beta})_i + f_j & \text{if } i = a_j. \end{cases}$$

Claim 2. $\overrightarrow{g}(\gamma) = \alpha$ and $\gamma \in \mathcal{S}(M', c')$.

Proof. We show both points separately.

Showing $\overrightarrow{g}(\gamma) = \alpha$: We argue that $g(\gamma^i) = \alpha_i$ for all $i \in [m]$.

Case 1, $i \in T$: We have set $(\tilde{\beta})^i = \beta_i$. Note that γ differs from $\tilde{\beta}$ only in coordinates $a_1, a_2, \cdots, a_{d'}$. All these coordinates lie outside $\mathrm{BLOCK}(T)$. Thus, $g(\gamma^i) = g(\beta^i) = \alpha_i$.

Case 2, $i \in \mathrm{FREE}$: We have set $(\tilde{\beta})^i = u_i$ where $u_i \in g^{-1}(\alpha_i)$. Again, note that γ differs from $\tilde{\beta}$ only in the coordinates $a_1, a_2, \cdots, a_{d'}$, all of which lie outside $\mathrm{BLOCK}(i)$. It follows that $g(\gamma^i) = \alpha_i$.

Case 3, $i \in \mathrm{PIVOTS}$: Let $i = b_j$ and let $a_j \in \mathrm{BLOCK}(b_j)$ be the corresponding pivot variable. Recall that each pivot variable lies in a distinct block. Let a_j be the ℓ-th coordinate of $\mathrm{BLOCK}(b_j)$. We have set $(\tilde{\beta}^i) = u_i$ where $u_i \in g^{-1}(\alpha_i)$ is a stifling assignment for ℓ and α_i. This means that $g((u_i)^{(\ell)}) = g(u_i) = \alpha_i$ ($s^{(\ell)}$ denotes s with ℓ-th coordinate flipped). Notice that γ and $\tilde{\beta}$ agree everywhere on $\mathrm{BLOCK}(b_j)$ except possibly a_j. This implies $g(\gamma^i) = \alpha_i$.

Showing $\gamma \in \mathcal{S}(M', c')$: Note that all equations corresponding to rows in M_1 are satisfied by γ since they are satisfied by β and hence by $\tilde{\beta}$ too. That the equations corresponding to M_2 are satisfied by γ follows from the row echelon structure of M_2, i.e., the fact that after an appropriate permutation of the columns, M' looks as follows:

$$\text{M'} = \begin{array}{c|c|c} B_1 & 0 & 0 \\ B_2 & I_{d'} & B_3 \\ \hline \text{Closure } T & \text{Pivots of } M_2 & \end{array} \begin{array}{l} = M_1 \\ = M_2 \end{array}$$

$\square$

Since $\mathcal{S}(M', c') = \mathcal{S}(M, c) = A$, Lemma 6 follows immediately from Claim 2. $\square$

5 Proof Outline

In this section, we provide an outline of the proof of our main result, Theorem 1. The proof consists of two parts. The first part shows that the formula $\mathrm{Stone}(G, \rho) \circ g$ has a polynomial length resolution proof for any directed acyclic graph G on N vertices and out-degree 2, any obfuscation map $\rho : [N]^3 \rightarrow \mathcal{V}$, and any gadget $g : \{0,1\}^b \rightarrow \{0,1\}$, where b is logarithmic in N (recall that the number of variables m of the formula $\mathrm{Stone}(G, \rho)$ is $2N^2$). This part of the proof is an adaptation of an analogous proof for the stone formula given by Alekhnovich et al. [1].

The second part establishes that there is a graph G and an obfuscation map $\rho : [N]^3 \rightarrow \mathcal{V}$ such that any bottom-regular ResLin proof of $\mathrm{Stone}(G, \rho) \circ \mathsf{IP}$ has exponential length in m, where IP is the inner product function on $b = \Theta(\log m)$ bits. The proof of this part is involved and non-trivial. We outline the main steps in the figure below, immediately followed by a high-level description of each step depicted.
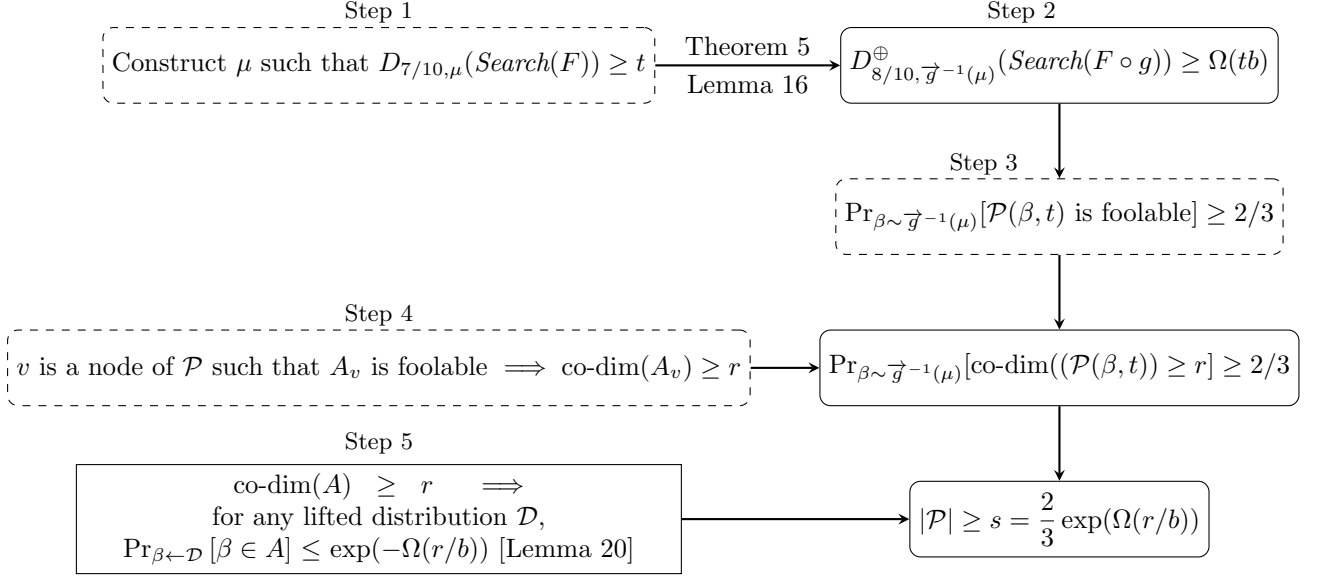
Figure 2: Outline of the proof. The solid boxes refer to parts that are quite general and not specific to a formula, while the dashed boxes contain modules that are more specific to $\mathsf{SP}_n \circ \mathsf{IP}$ and similar formulas.

Outline of the Lower Bound Proof. Our argument is an adaptation of the method presented in Efremenko et al [6] with addition of some new ingredients. See Figure 2, for depicting the method. Let $\mathcal{P}(\beta, t)$ be the node that $\mathcal{P}$ arrives at after making t linear queries on β.

Some Details: Let $\mathcal{P}$ be a bottom-read-once branching program computing $Search(\text{Stone}(G, \rho) \circ \mathsf{IP})$ corresponding to a bottom-regular ResLin proof of $\text{Stone}(G, \rho) \circ \mathsf{IP}$ where G is a pyramid graph of n levels and ρ is a carefully chosen obfuscation map. The proof consists of several steps.

1. We design a distribution μ over the assignment of variables of the base formula F over m variables, typically supported over *critical assignments*, i.e. those which result in the falsification of exactly one clause. This module requires one to show that $Search(F)$ is average-case hard for deterministic decision trees of small height wrt μ. In particular, our Lemma 15 proves that the problem $Search(\text{Stone}(G, \rho))$ is average-case hard for deterministic decision trees of height at most $O(n^{1/3})$. As the μ exhibited is formula specific, the box corresponding to this module is dashed.

2. In this step, we prove that the search problem associated with the lifted formula $F \circ g$ remains average case hard for *parity decision trees* wrt a *lifted distribution* as long as the gadget g has small rectangular discrepancy. More precisely, let $\overrightarrow{g}^{-1}(\mu)$ denote the lifted distribution generated by the following sampling: sample an input $z \in \{0,1\}^m$ according to μ. Then, sample at random an input $\beta \in \{0,1\}^{mb}$, conditioned on $\overrightarrow{g}(\beta) = z$. Using Theorem 5, implicit in the proof of the main result of Chattopadhyay, Filmus, Koroth, Meir and Pitassi [3], we conclude that $Search(F \circ g)$ is average-case hard for deterministic parity decision trees of small height, under the lifted distribution $\overrightarrow{g}^{-1}(\mu)$. This step is generic and works for any gadget of size $c \cdot \log(m)$, that has sufficiently small rectangular discrepancy under the uniform distribution over $\{0,1\}^b$. The gadget we use here is IP.

3. We then want to define a notion of progress the branching program $\mathcal{P}$ has made on arriving at a node v. To do so, consider the affine space A_v that labels the node. A_v may have nearly fixed/exposed the values of some of the blocks of input. These dangerous blocks are precisely $\text{Cl}(A_v)$ as defined in Section 4. They form the minimum obstruction set. Intuitively, the danger is $\mathcal{P}$ may have nearly found out a falsified clause of $F \circ g$ on reaching v if that clause was made up entirely of variables from blocks in $\text{Cl}(A_v)$. However, in this step we observe that the average-case hardness of the Search problem for PDTs proved in the previous step precludes this from happening with appreciable probability, when the input is sampled according to the lifted distribution $\overrightarrow{g}^{-1}(\mu)$. To formalize this idea, we need to concretely say when A_v is (not) dangerous. So far, we have not been able to lay out a general notion of danger, but notions specific to individual formulas have been defined. For $\text{Stone}(G, \rho)$, this notion is captured by Definition 8 of *foolable* spaces, provided

in Section 7.3. Theorem 6 shows that w.h.p., $\mathcal{P}$ reaches a foolable space on walking for $n^{1/3}$ steps, querying an input sampled according to $\overrightarrow{g}^{-1}(\mu)$.

4. In this step, we show that when the affine space A_v is not dangerous, i.e. it is foolable or consistent, the appropriate notion depending on the formula at hand, A_v has large co-dimension. All steps until now held for general branching programs (or equivalently proof DAGs). This step is the only one where the bottom-read-once property is exploited. For $\text{Stone}(G, \rho)$, this is achieved in Section 7.4, at the end, by Lemma 19.

5. In this step, we prove a general result about lifted distributions. For any affine space A of co-dim$(A) = r$ and any distribution μ on $\{0,1\}^m$, we prove that β sampled by $\overrightarrow{g}^{-1}(\mu)$ is in A only with probability $2^{-\Omega(r/b)}$, as long as the gadget g is balanced and stifling. In other words, lifted distributions, even though their support is quite sparse in the ambient space, are pseudo-random for the rank measure. This property, though simple to prove, turns out to be extremely useful, especially for formulas like the stone formulas that are barely hard.

At this stage we are ready to put together the above steps in the following way. Let R be a set of nodes w of $\mathcal{P}$ such that there is a path from the root of $\mathcal{P}$ to w of length t, and co-dim$(A_w) \geq t$. Setting $t := n^{1/3}$ we have the following.

$$\frac{7}{10} \leq \Pr_{\beta \sim \overrightarrow{\mathsf{IP}}^{-1}(\mu)} \left[\text{co-dim}(A_v) \geq t \text{ for } v = \mathcal{P}(\beta, t) \right] \qquad \text{(by Step 3 and 4)}$$

$$\leq \sum_{w \in R} \Pr_{\beta \sim \overrightarrow{\mathsf{IP}}^{-1}(\mu)} \left[\mathcal{P}(\beta, t) = w \right] \qquad \text{(by union bound)}$$

$$\leq |R| \cdot 2^{\Omega(-t/b)} \qquad \text{(by Step 5)}$$

By rearranging, we get the lower bound $|R| \geq 2^{\Omega(n^{1/3}/\log n)}$. Recall that the number of variables of $\text{Stone}(G, \rho) \circ \mathsf{IP}$ is $M = \Theta(n^4 \log(n))$. In terms of M, the lower bound is $2^{\Omega(M^{1/12}/\log^{13/12} M)} = 2^{M^{\Omega(1)}}$.

6 Upper Bound

In this section, we show the upper bound part of Theorem 1.

Theorem 3. *Let $G = (V, E)$ be an directed acyclic graph with N vertices such that there is exactly one root r (vertex with indegree 0), and each non-sink vertex has outdegree exactly 2. Let $\rho : [N]^3 \to \mathcal{V}$ be any obfuscation map, and $g : \{0,1\}^b \to \{0,1\}$ be a Boolean function for $b \leq O(\log N)$. Then, the formula $\text{Stone}(G, \rho) \circ g$ admits a resolution refutation of length polynomial in N.*

The proof of Theorem 3 is an adaptation of the proof given by Alekhnevich et al. [1] for lifted formulas. We remark that Alekhnevich et al. [1] presented a resolution refutation for the stone formulas of constant width. This allow us to adapt the refutation for the lifted formula. For the rest of the section, we fix a graph G, an obfuscation map ρ, and a gadget g satisfying the assumptions of Theorem 3. First, we prove several auxiliary lemmas about the formula $\text{Stone}(G, \rho) \circ g$.
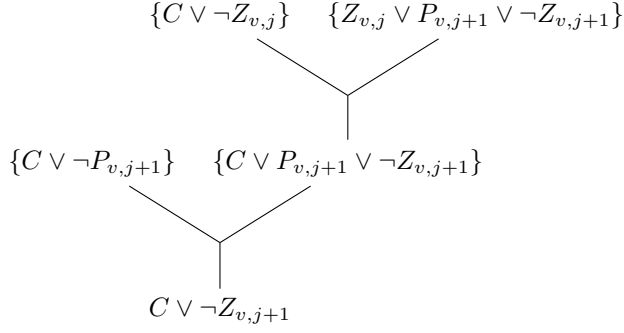
Lemma 7. *Let C be a clause with width w. Suppose we have derived the clauses $C \vee \neg P_{v,j}$ for a fixed $v \in V$ and all $1 \leq j \leq N$. Then, we can derive C in N steps in width $\leq w + 2$.*

Proof. We derive the clause C in N steps. We will subsequently derive $C \vee \neg Z_{v,j+1}$ from $C \vee \neg Z_{v,j}$
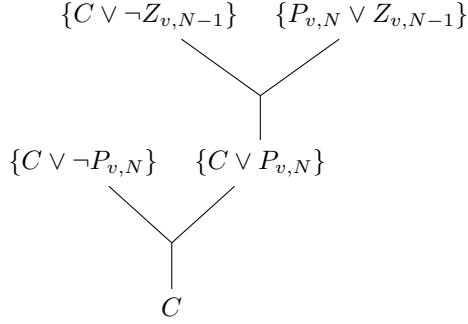
Base step: Deriving $C \vee \neg Z_{v,1}$.

$$\{C \vee \neg P_{v,1}\} \qquad \{P_{v,1} \vee \neg Z_{v,1}\}$$

$$C \vee \neg Z_{v,1}$$

Step j: For $j \in [N-2]$, deriving $C \vee \neg Z_{v,j+1}$ from $C \vee \neg Z_{v,j}$.

$$\{C \vee \neg Z_{v,j}\} \quad \{Z_{v,j} \vee P_{v,j+1} \vee \neg Z_{v,j+1}\}$$

$$\{C \vee \neg P_{v,j+1}\} \quad \{C \vee P_{v,j+1} \vee \neg Z_{v,j+1}\}$$

$$C \vee \neg Z_{v,j+1}$$

Final step: Deriving C.

$$\{C \vee \neg Z_{v,N-1}\} \quad \{P_{v,N} \vee Z_{v,N-1}\}$$

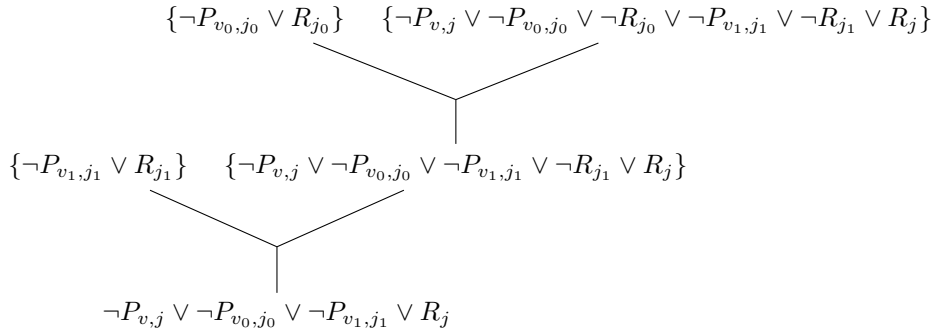$$\{C \vee \neg P_{v,N}\} \quad \{C \vee P_{v,N}\}$$

$$C$$

$\square$

For a vertex v, we define the set of clauses $S(v) = \{\neg P_{v,j} \vee R_j | 1 \leq j \leq N\}$.

Lemma 8. *Let v be a vertex in G with children v_0, v_1. We can derive $S(v)$ from $S(v_0)$, and $S(v_1)$ in constant width and length $O(N^3)$.*

Proof. We derive $S(v)$ in several steps.

1. For every $j, j_0, j_1 \in [N]$, we perform the following sequence of operations:

$$\{\neg P_{v_0,j_0} \vee R_{j_0}\} \quad \{\neg P_{v,j} \vee \neg P_{v_0,j_0} \vee \neg R_{j_0} \vee \neg P_{v_1,j_1} \vee \neg R_{j_1} \vee R_j\}$$

$$\{\neg P_{v_1,j_1} \vee R_{j_1}\} \quad \{\neg P_{v,j} \vee \neg P_{v_0,j_0} \vee \neg P_{v_1,j_1} \vee \neg R_{j_1} \vee R_j\}$$

$$\neg P_{v,j} \vee \neg P_{v_0,j_0} \vee \neg P_{v_1,j_1} \vee R_j$$

2. For each fixed j_0, j, we apply Lemma 7 to the clause $C := \neg P_{v,j} \vee \neg P_{v_0,j_0} \vee R_j$ and we derive $\neg P_{v,j} \vee \neg P_{v_0,j_0} \vee R_j$.

3. For each fixed j, we apply Lemma 7 to the clause $C := \neg P_{v,j} \vee R_j$ and we derive $\neg P_{v,j} \vee R_j$.

$\square$

Lemma 9. *The formula $Stone(G, \rho)$ has a resolution refutation of width $O(1)$ and size polynomial in N.*
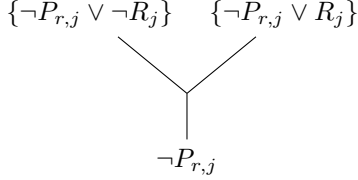
Proof. The refutation proceeds in the following steps.

Elimination of the ρ's : For every induction clause C, we resolve the appended ρ-variable.

$$\{C \vee \rho\} \quad \{C \vee \neg\rho\}$$

$$C$$
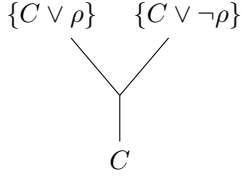
Derivation of $S(r)$: For each sink s of G, the clauses $S(s)$ are present in the axioms of $\mathrm{Stone}(G, \rho)$. By Lemma 8, we subsequently derive the set $S(r)$ for the root r of G.

Empty clause derivation: For each $1 \le j \le N$, we derive $\neg P_{r,j}$.

$$\{\neg P_{r,j} \vee \neg R_j\} \quad \{\neg P_{r,j} \vee R_j\}$$

$$\neg P_{r,j}$$

Now by applying Lemma 7 for C being an empty clause $\bot$, we derive $\bot$, that concludes the proof.

$\square$

Now, from constant-width polynomial-length refutation of $\mathrm{Stone}(G, \rho)$ we derive a polynomial-length refutation of the lifted formula $\mathrm{Stone}(G, \rho) \circ g$.

Lemma 10. *Let $g : \{0,1\}^b \to \{0,1\}$ be a Boolean function and Φ be a CNF unsatisfiable formula over n variables containing only constant width clauses. Suppose Φ has a resolution refutation of length ℓ and constant width. Then, $\Phi \circ g$ contains clauses of width $O(b)$ and admits a resolution refutation of size $\ell \cdot 2^{O(b)}$.*

Proof. By construction, if C is a clause of width k, then $|C \circ g| \le 2^{bk}$. If k is constant, this is $2^{O(b)}$. We show that, for every derivation step $(A \vee x), (B \vee \neg x) \to (A \vee B)$ in a proof for Φ, we can derive all clauses of $(A \vee B) \circ g$ from the clauses of $(A \vee x) \circ g$ and $(B \vee \neg x) \circ g$ in polynomial size, assuming each of A, B has constant width. This follows from the fact that $(A \vee x) \circ g$ and $(B \vee \neg x) \circ g$ semantically imply $(A \vee B) \circ g$: an assignment $(x_{i,1}, \ldots, x_{i,b})_{i \in [M]}$ satisfies formula $C \circ g$ if and only if the assignment $(g(x_{i,1}, \ldots, x_{i,b}))_{i \in [n]}$ satisfies clause C. And since clauses $A \vee x, B \vee \neg x$ semantically imply $A \vee B$, it follows that the formulas $(A \vee x) \circ g$ and $(B \vee \neg x) \circ g$ semantically imply the formula $(A \vee B) \circ g$.

As both A and B are constant-width clauses, each of the formulae $(A \vee x) \circ g$ and $(B \vee \neg x) \circ g$ are defined on at most $O(b)$ variables. By Lemma 1, we can derive each clause in $(A \vee B) \circ g$ from $(A \vee x) \circ g$ and $(B \vee \neg x) \circ g$ in at most $2^{O(b)}$ resolution steps.

Using this fact, we can mimic the resolution refutation of Φ. For each intermediate clause C derived in the resolution refutation for Φ, we can derive all clauses in $C \circ g$. In the end, we derive $\bot \circ g = \{\bot\}$, i.e., the empty clause. Assuming the width of the resolution refutation for Φ is bounded by some constant, the total length of our simulation is at most $\ell \cdot 2^{O(b)}$. $\square$

Now, Theorem 3 is a corollary of Lemma 9, and 10.

7 Lower Bound

In this subsection, we prove the lower bound part of Theorem 1 following the outline given in Section 5.

Theorem 4. *There is an obfuscation map $\rho : [N]^3 \to \mathcal{V}$ such that any bottom-regular ResLin refutation of $\mathsf{SP}_{n,\rho} \circ \mathsf{IP}$ must have length at least $2^{\Omega(n^{1/3}/\log n)}$.*

Recall that number of variables of $\mathsf{SP}_{n,\rho} \circ \mathsf{IP}$ is $\Theta(n^4 \log n)$. Thus, the lower bound given by Theorem 4 yields the lower bound claimed in Theorem 1. For the rest of this section, we fix $G = (V, E)$ to be the pyramid graph of n levels, and $N = n(n+1)/2$ vertices.

7.1 The Stone Formula is Average-Case Hard for Decision Trees

We shall construct a distribution μ on $\{0,1\}^m$ such that for any obfuscation map $\rho : [N]^3 \to \mathcal{V}$, the search problem $Search(\mathsf{SP}_{n,\rho})$ is hard on average w.r.t. μ for deterministic decision trees of sufficiently small height (around $n^{1/3}$).

First, we fix an arbitrary bijection $f : [N] \to V$ between stones and vertices of the pyramid. All assignments in $\mathrm{Supp}(\mu)$ will place the stone i on vertex $f(i)$. The distribution μ samples the assignments as follows.

1. Assign stone i to vertex $f(i)$. Formally for each $v \in V$, $i \in [N]$, and $j \in [N-1]$, we set:

$$P_{v,i} = \begin{cases} 1 & \text{if } f(i) = v \\ 0 & \text{otherwise} \end{cases}$$

$$Z_{v,j} = \begin{cases} 0 & \text{if } j < f^{-1}(v) \\ 1 & \text{otherwise} \end{cases}$$

2. Sample $n-2$ independent uniform bits $B_2, \ldots, B_{n-1} \in \{0,1\}$

3. Let $X_1 = 1$, and for $2 \leq j \leq n-1$, let $X_j = X_{j-1} + B_j$. Color the vertices (j, X_j) blue for $1 \leq j \leq n-1$ and other vertices red, i.e., for each stone $i \in [N]$, we set:

$$R_i = \begin{cases} 0 & \text{if } j \leq n-1 \text{ and } (j, X_j) = f(i) \\ 1 & \text{otherwise} \end{cases}$$

Let $\alpha \in \mathrm{Supp}(\mu)$. The assignment α corresponds to the following stone placement. It places a different stone on each vertex. There is a path P from the root $r = (1,1)$ to a vertex v in the level $n-1$ given by the random variables $X_1, \ldots, X_{n-1}$, i.e. the vertices of the path are $\{(1, X_1), \ldots, (n-1, X_{n-1})\}$. The stones on the vertices of P are colored blue, all other stones are colored red. An example of such a coloring is shown in Figure 3.
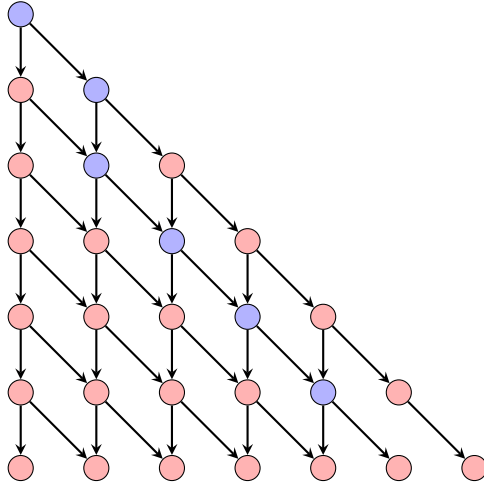

Figure 3: An example of a pyramid graph with coloring giving by an assignment sampled by the hard distribution μ.

We call the path P as the *blue path induced by* α and the vertex v as the *end* of P. Note that the only clause falsified by the assignment α is one of the induction clauses for the vertex v and its children u and w. Formally, for the stones $i = f^{-1}(u), j = f^{-1}(w)$, and $k = f^{-1}(v)$, the assignment α falsifies exactly one of the following two induction clauses (depending how α sets the value of $\rho(i,j,k)$):

$$D_1(v) := \neg P_{u,i} \vee \neg R_i \vee \neg P_{w,j} \vee \neg R_j \vee \neg P_{v,k} \vee R_k \vee \rho(i,j,k)$$

$$D_0(v) := \neg P_{u,i} \vee \neg R_i \vee \neg P_{w,j} \vee \neg R_j \vee \neg P_{v,k} \vee R_k \vee \neg \rho(i,j,k)$$

Consider a random walk $Y_1, \ldots, Y_k$ on a number line starting at $q \in \mathbb{N}$ distributed as follows: $Y_1 = q$, and for $i > 1$

$$Y_i = \begin{cases} Y_{i-1} + 1 & \text{with probability } \frac{1}{2} \\ Y_{i-1} & \text{with probability } \frac{1}{2} \end{cases}$$

Note that the random variables $X_1, \ldots, X_{n-1}$ used in the construction of μ are distributed as $Y_1, \ldots, Y_{n-1}$ for $Y_1 = 1$.

Lemma 11. *There exists a constant $c_1 \geq 0$ such that for any $p \in \{q, \ldots, q + k - 1\}$ and $t \in \{2, \ldots, k\}$, we have $\Pr[Y_t = p] \leq c_1/\sqrt{t}$.*

Proof. Note that $Y_t = q + \sum_{i=2}^{t} B_i$, where each B_i is an independent uniform random bit. Now, $Y_t = p = p' + q$ for $p' \in \{0, \ldots, k - 1\}$ if and only if $\sum_{i=2}^{t} B_i = p'$.

$$\Pr[Y_t = p] = \Pr\left[\sum_{i=2}^{t} B_i = p'\right] = \binom{t-1}{p'} \cdot 2^{-t+1} \leq \binom{t-1}{\lfloor \frac{t-1}{2} \rfloor} \cdot 2^{-t+1} \leq \frac{c_1}{\sqrt{t}}$$

For an appropriate constant $c_1 > 0$, the last inequality is implied by Stirling's formula. $\square$

For a set $S \subseteq [k] \times \mathbb{N}$, we say the random walk W avoids S if for all $(i, j) \in S$ it holds that $Y_i \neq j$.

Lemma 12. *Let $c_2 \geq 1$ be a constant and $S \subseteq [k] \times \mathbb{N}$ be a set of forbidden points with $|S| \leq t$. Suppose there exists an interval $I = [L, R] \subseteq [k]$ with $|I| \geq c \cdot t^2$ for sufficiently large constant c depending on c_2, such that no point of S has the first coordinate in I, i.e., for all $(i, j) \in S : i < L$ or $i > R$. If the random walk $W = Y_1, \ldots, Y_k$ avoids S with non-zero probability, then for any $z \in \{q, \ldots, q + k - 1\}$ it holds that*

$$\Pr[Y_k = z \mid W \text{ avoids } S] \leq \frac{1}{c_2 t}.$$

Proof. We partition S into two subsets S_1 and S_2 of points before and after the interval I, $S_1 = \{(i, j) \in S \mid i < L\}$, and $S_2 = \{(i, j) \in S \mid i > R\}$. Note that by the assumption, we have $S = S_1 \dot\cup S_2$.

We show that for all p such that $\Pr[Y_L = p \mid W \text{ avoids } S] > 0$, we have $\Pr[Y_k = z \mid Y_L = p, W \text{ avoids } S] \leq 1/c_2 t$. We set $c := 4c_1^2 c_2^2$ where c_1 is the constant given by Lemma 11. We have by Lemma 11 that

$$\Pr[Y_k = z \mid Y_L = p, W \text{ avoids } S_1] = \Pr[Y_k = z \mid Y_L = p] \leq \frac{1}{2c_2 t}, \tag{7.1}$$

since $z - L \geq |I| \geq c \cdot t^2$. Again, for all $(i, j) \in S_2$ (i.e., $i > R$) we have by Lemma 11 that

$$\Pr[Y_i = j \mid Y_L = p, W \text{ avoids } S_1] \leq \frac{1}{2c_2 t} \leq \frac{1}{2t}.$$

By union bound,

$$\Pr[\exists (i, j) \in S_2 \text{ such that } Y_i = j \mid Y_L = p, W \text{ avoids } S_1] \leq \frac{1}{2}. \tag{7.2}$$

Therefore,

$$\begin{aligned}
\Pr[Y_k = z \mid Y_L = p, W \text{ avoids } S] &= \frac{\Pr[Y_k = z, W \text{ avoids } S_2 \mid Y_L = p, W \text{ avoids } S_1]}{\Pr[W \text{ avoids } S_2 \mid Y_L = p, W \text{ avoids } S_1]} \\
&\leq 2 \cdot \Pr[Y_k = z \mid Y_L = p, W \text{ avoids } S_1] \quad \text{(by (7.2))} \\
&\leq \frac{1}{c_2 t} \quad \text{(by (7.1))}
\end{aligned}$$

Now, we are ready to finish the proof.

$$\begin{aligned}
\Pr[Y_k = z \mid W \text{ avoids } S] &= \sum_p \Pr[Y_L = p \mid W \text{ avoids } S] \cdot \Pr[Y_k = z \mid Y_L = p, W \text{ avoids } S] \\
&\leq \sum_p \Pr[Y_L = p \mid W \text{ avoids } S] \cdot \frac{1}{c_2 t} = \frac{1}{c_2 t}
\end{aligned}$$

$\square$

Now we show that when inputs are sampled according to μ, any deterministic decision tree for $Search(\mathsf{SP}_{n,\rho})$ with small height makes an error with high probability. Note that for each $v \in V, i \in [N], j \in [N-1]$, the assignment to the variables $P_{v,i}$ and $Z_{v,j}$ are fixed by any assignment in $\mathrm{Supp}(\mu)$ (in other words, for each vertex the stone placed on it is fixed). Thus, we can assume WLOG the decision tree only queries the variables R_j. We say a decision tree queries the color of a vertex v of G if it queries the variable $R_{f^{-1}(v)}$. (Recall that the stone $f^{-1}(v)$ is placed on the vertex v by assignments in $\mathrm{Supp}(\mu)$.)

Note that there is a simple, even non-adaptive, decision tree of height $O(\sqrt{n})$ that makes few errors. It simply queries the colours of $O(\sqrt{n})$ nodes of the pyramid graph, centered around the $(n-1)/2$-th node at level $n-1$. With very high probability, there is a blue node among the queried ones which uniquely identifies the falsified induction clause. Nevertheless, we will show that all decision trees of height at most $n^{1/3}$, will make errors with large probability to identify a falsified clause.

Consider a decision tree $\mathcal{T}$ for $Search(\mathsf{SP}_{n,\rho})$. We transform $\mathcal{T}$ into a decision tree $\mathcal{T}'$ in a canonical form:

- Initially, $\mathcal{T}'$ always queries the color of the root r of G.

- Suppose $\mathcal{T}$ outputs an induction clause $D_0(v)$ or $D_1(v)$ for a vertex v of G. Then, $\mathcal{T}'$ queries the color of the vertex v first. If the color of v is red (i.e., $R_{f^{-1}(v)} = 1$), then $\mathcal{T}'$ outputs an error symbol. Otherwise it outputs the same induction clause that $\mathcal{T}$ outputs.

- If $\mathcal{T}$ outputs any other clause, then $\mathcal{T}'$ outputs an error symbol.

We remark this modification increases the height of the tree by at most two. Given any assignment in $\mathrm{Supp}(\mu)$, a decision tree $\mathcal{T}'$ in a canonical form can output either an induction clause from $\{D_0(v), D_1(v) \mid v \in V(G)\}$ or an error symbol. The probability of $\mathcal{T}$ making an error is precisely the probability of reaching a leaf node of $\mathcal{T}'$ labeled with an error symbol.

Note that for each cube $C \subseteq \{0,1\}^m$ there is a corresponding partial assignment $\alpha_C \in \{0,1,*\}^m$ such that the cube C is exactly the set of total extensions of α_C, i.e., $C = \{\alpha \in \{0,1\}^m \mid \alpha \text{ extends } \alpha_C\}$. We say a cube $C \subseteq \{0,1\}^m$ fixes a vertex $v \in V(G)$ to red (or blue) if the corresponding partial assignment α_C assigns a value 1 (or 0) to the variable $R_{f^{-1}(v)}$.

Now, fix a decision tree $\mathcal{T}$ for $Search(\mathsf{SP}_{n,\rho})$ in a canonical form and let $h := \gamma \cdot n^{1/3}$ be the height of $\mathcal{T}$, where $\gamma > 0$ is sufficiently small constant.

Definition 6. *We say a cube $C \subseteq \{0,1\}^m$ is* useful *if there exist $1 \leq L_1 \leq L_2 \leq L_3 \leq L_4 \leq N$ such that:*

1. *The cube C fixes the some vertex in level L_1 to blue and some vertex in level L_4 to blue.*

2. *For all $L_2 \leq \ell \leq L_3$, the cube C does not fix the color of any vertex in level ℓ.*

3. $L_3 - L_2 \geq \dfrac{n}{2h}$

A node p of $\mathcal{T}$ is called useful *if the cube C_p associated with it is useful.*

Clearly, the root of $\mathcal{T}$ is not useful.

Lemma 13. *Let $\alpha \in Supp(\mu)$ be an assignment on which $\mathcal{T}$ reaches the leaf p. If p does not output an error symbol, then p is useful.*

Proof. Let v be the endpoint of the blue path induced by α. Then, p outputs one of the induction clauses $D_0(v)$ or $D_1(v)$ for $v \in V(G)$. Since $\mathcal{T}$ is in the canonical form, the cube C_p fixes the vertex v and the root r of G to blue. Recall that the vertex v is in level $n-1$. Let $1 = \ell_1 < \cdots < \ell_d = n-1$ be the levels where C_p fixes some vertices. Since $d \leq h$, there must exist an i such that $\ell_{i+1} - \ell_i \geq \dfrac{n-1}{h+1} > \dfrac{n}{2h}$.. There is no fixed vertex on levels $\ell_i + 1, \ldots, \ell_{i+1} - 1$.

We take largest the ℓ_1 such that $\ell_1 \leq \ell_i$ and C_p fixes a vertex on ℓ_1 to blue. Similarly, we take the smallest ℓ_2 such that $\ell_2 \geq \ell_{i+1}$ and C_p fixes a vertex on ℓ_2 to blue. The cube C_p satisfies the conditions of being a useful node in Definition 6 by taking $(L_1, L_2, L_3, L_4) = (\ell_1, \ell_i + 1, \ell_{i+1} - 1, \ell_2)$. $\qquad\square$

Lemma 14. *For any $\varepsilon > 0$ there exists $\gamma > 0$ such that*

$$\Pr_{\alpha \sim \mu}[\text{The computation path of } \mathcal{T} \text{ on } \alpha \text{ reaches a useful node}] \leq \varepsilon.$$

Proof. Let $\mathcal{T}(\alpha, k)$ denote the node of $\mathcal{T}$ reached by α after k queries. For each $1 \leq k \leq h$, we upper bound the probability that the computation path of α reaches a useful node for the first time at step k. Then, we shall use union bound on k. Formally, we bound the probability as follows.

$$\Pr_{\alpha \sim \mu}[\text{computation path of } \alpha \text{ reaches a useful node}]$$

$$= \Pr_{\alpha \sim \mu}\left[\exists k \in [h] : \mathcal{T}(\alpha, k) \text{ is useful and } \mathcal{T}(\alpha, k-1) \text{ is not useful}\right]$$

$$\leq \sum_{k=1}^{h} \Pr_{\alpha \sim \mu}\left[\mathcal{T}(\alpha, k) \text{ is useful and } \mathcal{T}(\alpha, k-1) \text{ is not useful}\right]$$

$$\leq h \cdot \max_{k \in [h]} \Pr_{\alpha \sim \mu}\left[\mathcal{T}(\alpha, k) \text{ is useful} \mid \mathcal{T}(\alpha, k-1) \text{ is not useful}\right]$$

We bound the last probability for any $k \in [h]$. Let $p = \mathcal{T}(\alpha, k-1)$. We assume the node p is not useful. Let $(i, j) \in V(G)$ be the lowest vertex that is fixed by C_p to blue. Suppose that in the the next step $\mathcal{T}$ queries a color of the vertex (i', j'). If the next node has to be useful, the response to the query has to be blue. Moreover, there have to be $n/2h$ consecutive layers $i < \ell', \ldots, \ell' + \frac{n}{2h} - 1 < i'$ such that C_p does not fix any vertex on those layers. The probability that the response to the query is blue is

$$\Pr_{\alpha \sim \mu | C_p}[\text{The blue path induced by } \alpha \text{ visits } (i', j')].$$

Consider the random walk $X_1, \ldots, X_{n-1}$ that determines the blue path P induced by $\alpha \sim \mu$. Recall that the vertices of P are $\{(1, X_1), \ldots, (n-1, X_{n-1})\}$. The cube C_p fixes colors of some vertices of G. Let B and R be the set of vertices whose colors are fixed by C_p to blue and red respectively.

Conditioning on the cube C_p restricts the random walk $X_1, \ldots, X_{n-1}$ that it must visit the points in B and must not visit the points in R. Formally, for any $(q, y) \in B$ it holds that $X_q = y$ and for any $(q', y') \in R$ it holds that $X_{q'} \neq y'$. We know there is at least one walk that avoids R and visits B (the walk corresponding to P). Moreover, we have $|R| \leq h$ and there is a large "gap" in R, i.e., for each $(i_1, j_1) \in R$ it holds that $i_1 < \ell'$ or $i_1 > \ell' + \frac{n}{2h} - 1$. Recall that we set $h = \gamma n^{1/3}$. Set γ to a sufficiently small constant so that $\frac{n}{2h} \geq ch^2$, where c is a sufficiently large constant for which we can apply Lemma 12 with $c_2 = \varepsilon^{-1}$. By applying an appropriate time shift, we have by Lemma 12 that

$$\Pr_{\alpha \sim \mu | C_p}[\text{The blue path induced by } \alpha \text{ visits } (i', j')] = \Pr_{\mu | C_p}[X_{i'} = j'] \leq \frac{1}{c_2 h} \leq \frac{\varepsilon}{h}.$$

Thus, we conclude that for all $k \in [h]$,

$$\Pr_{\mu \sim \alpha}\left[\mathcal{T}(\alpha, k) \text{ is useful} \mid \mathcal{T}(\alpha, k-1) \text{ is not useful}\right] \leq \frac{\varepsilon}{h}.$$

Therefore, the probability that the computation path ever reaches a useful node is at most ε. $\qquad\square$

We end this subsection with by showing that the formula $\mathsf{SP}_{n,\rho}$ is average-case hard for decision trees.

Lemma 15. *For any $\varepsilon > 0$, there exists $\gamma > 0$ such that every deterministic decision tree of height at most $\gamma \cdot n^{1/3}$ for $\text{Search}(\mathsf{SP}_{n,\rho})$ makes error with probability $\geq 1 - \varepsilon$ w.r.t. the distribution μ.*

Proof. If the decision tree answers correctly, by Lemma 13 it must reach a useful node at some point. By Lemma 14, the probability of this ever happening is at most ε if γ is small enough. $\qquad\square$

7.2   Lifting the Average-Case Hardness to Parity Decision Trees

We lift the distribution μ to a distribution μ' of variables of $\mathsf{SP}_{n,\rho} \circ \mathsf{IP}$ as follows:

1. Sample an assignment α according to μ.

2. Sample a uniformly random assignment from $\overrightarrow{\mathsf{IP}}^{-1}(\alpha)$.

We remark that an assignment β sampled by μ' falsifies exactly one clause of $\mathsf{SP}_{n,\rho} \circ \mathsf{IP}$, in particular one clause that arises by a lifting clause C of $\mathsf{SP}_{n,\rho}$ where C is the unique clause falsified by the assignment $\overrightarrow{\mathsf{IP}}(\beta)$.

In this section, we prove $Search(\mathsf{SP}_{n,\rho} \circ \mathsf{IP})$ is average-case hard for parity decision trees of small height under the lifted distribution. To do so, we shall use a result of Chattopadhyay et al. [3], that built upon the earlier work of Göös, Pitassi and Watson [8].

We will need to consider randomized decision trees that output Boolean strings in $\{0,1\}^t$, rather than 0/1. For a given deterministic 2-party communication protocol Π, let $\Pi(x,y)$ denote the transcript generated by Π on input (x,y).

Theorem 5 (Implicit in [3]). *Assume $b \geq 50\log(m)$. Let Π be any deterministic 2-party communication protocol of cost c, where Alice and Bob each get inputs from $\{0,1\}^{mb}$. For any $z \in \{0,1\}^m$, let (X_z, Y_z) denote the distribution on pairs obtained by sampling from $\overrightarrow{\mathsf{IP}}^{-1}(z)$ uniformly at random. Then, there exists a randomized decision tree $\mathcal{T}$ of cost $O(c/b)$ such that the following holds for every $z \in \{0,1\}^m$:*

$$d_{TV}\big(\mathcal{T}(z), \Pi(X_z, Y_z)\big) \leq 1/10.$$

The above theorem says that a randomized decision tree is able to simulate by probing only a few bits of its input z, the transcript of a deterministic communication protocol when it is given a random input pair X_z, Y_z. Its relevance for us is due to the following simple observation.

Observation 4. *Every deterministic parity decision tree of height h can be simulated exactly by a deterministic 2-party communication protocol of cost at most $2h$.*

Now we can lift our avarage-case hardness to parity decision trees.

Lemma 16. *There exists a constant $c > 0$ such that for every obfuscation map ρ and every parity decision tree $\mathcal{T}$ of height at most $c \cdot n^{1/3} \log n$ purporting to solve $Search(\mathsf{SP}_{n,\rho} \circ \mathsf{IP})$, the following is true:*

$$\Pr_{\beta \sim \mu'} \left[\mathcal{T}(\beta) \text{ is falsified on } \beta \right] \leq \frac{2}{5}.$$

Proof. Assume $\mathcal{T}$ makes an error with probability $< 3/5$. Then our main idea is that we would be able to construct an ordinary decision tree for $Search(\mathsf{SP}_{n,\rho})$ of depth $O(n^{1/3})$ which makes error with probability $< 7/10$ under distribution μ. This contradicts Lemma 15.

Using Observation 4, we get a deterministic 2-party protocol of cost at most $2 \cdot \text{depth}(\mathcal{T})$ that makes error less than $3/5$ for solving $Search(\mathsf{SP}_{n,\rho} \circ \mathsf{IP})$. Theorem 5 then yields a randomized decision tree $\mathcal{T}'$ with the following properties. On input $\alpha \sim \mu$,

- $\mathcal{T}'$ makes at most $O(\text{cost}(\Pi)/\log n)$ queries to α, i.e., at most $O(n^{1/3})$.

- If $\mathcal{D}_1$ denotes the actual distribution of the transcript of Π when it is run on input sampled uniformly at random from $\overrightarrow{\mathsf{IP}}^{-1}(\alpha)$ and $\mathcal{D}_2$ denotes the distribution of the transcript of Π simulated by $\mathcal{T}'$,

$$||\mathcal{D}_1 - \mathcal{D}_2|| \leq \frac{1}{10}$$

We now modify $\mathcal{T}'$ to output a clause as follows. A transcript of Π leads it to output a clause of $\mathsf{SP}_{n,\rho} \circ \mathsf{IP}$. The modified $\mathcal{T}'$ outputs the unique corresponding un-lifted clause of $\mathsf{SP}_{n,\rho}$. The probability of error is at most $\Pr[\Pi \text{ errs }] + 1/10 < 7/10$. This gives a randomized decision tree; by fixing the coins we can replace it by a deterministic decision tree, contradicting Lemma 15. $\square$

7.3 Foolable Nodes Are Frequent

Let $\mathcal{P}$ be a bottom-read-once linear branching program for $Search(\mathsf{SP}_{n,\rho} \circ \mathsf{IP})$ that corresponds to a bottom-regular ResLin proof of the unsatisfiability of $\mathsf{SP}_{n,\rho} \circ \mathsf{IP}$. Our goal is to show size of $\mathcal{P}$ is large. To do so, we establish that the affine spaces associated with many nodes of $\mathcal{P}$ have a certain property that allows to fool them. In particular, let β be an assignment sampled by μ'. We will prove that with high probability after making $t = O(n^{1/3})$ linear queries, according to β, we will end in a node v of $\mathcal{P}$ such that the associated affine space A_v does not have much information about β. We will show that this implies the affine space A_v contains many useful assignments that allows us to prove the co-dimension A_v is large. Now, we define the sought property formally.

Definition 7. *Let $\alpha \in Supp(\mu)$ and P be the blue path induced by α that ends at v. Let u and w be the two children of v. We say a subset $T \subseteq [m]$ is α-foolable if T does not contain any variable mentioning v, u or w, i.e. the variables $P_{x,i}, Z_{x,j}$ for $x \in \{u, v, w\}, i \in [N], j \in [N-1]$.*

Definition 8. *An affine space $A \subseteq \mathbb{F}_2^{mb}$ is α-foolable if $Cl(A)$ is α-foolable with $\alpha \in Supp(\mu)$ and there exists $\beta \in A$ such that $\alpha = \overrightarrow{\mathsf{IP}}(\beta)$.*

We call a node v of $\mathcal{P}$ *α-foolable* if the associated affine space A_v is α-foolable. Recall that $\mathcal{P}(\beta, t)$ is the node that $\mathcal{P}$ arrives at after making t linear queries on β. It turns out the node $\mathcal{P}(\beta, t)$ is α-foolable with high probability if t is sufficiently small. We prove this in the following theorem.

Theorem 6. *Let $\mathcal{P}$ be any bottom-read-once linear branching program corresponding to a bottom-regular ResLin proof of $\mathsf{SP}_{n,\rho} \circ \mathsf{IP}$. There exists a constant $c > 0$ such that if $t < c \cdot n^{1/3}$, then*

$$\Pr_{\alpha \sim \mu, \beta \sim \overrightarrow{\mathsf{IP}}^{-1}(\alpha)} \left[\mathcal{P}(\beta, t) \text{ is } \alpha\text{-foolable} \right] > \frac{3}{5}.$$

Proof. Let v denote the random node $\mathcal{P}(\beta, t)$. Let the blue path induced by α end at a and let the children of a be b and c. Notice that the second condition of being α-foolable (that A_v contains an element of $\overrightarrow{\mathsf{IP}}^{-1}(\alpha)$) is always satisfied by $\mathcal{P}(\beta, t)$ since one such element is β. We just need to lower bound the probability of the first condition of α-foolability being satisfied, i.e., the probability that $Cl(A_v)$ does not contain any variable mentioning a, b or c.

We construct a PDT $\mathcal{T}$ for $Search(\mathsf{SP}_n \circ \mathsf{IP})$ from $\mathcal{P}$ in the following manner: on input β, it will simulate the path traced out in $\mathcal{P}$ for t steps by making precisely those linear queries that would have been issued in $\mathcal{P}$. At the end of it, $\mathcal{T}$ does the following: Let A be the affine space corresponding to the queries issued and responses received so far. For every vertex $k = (i, j)$ in the pyramid graph G_n such that one of its variables ($P_{k,\ell}$ or $Z_{k,\ell}$ for some $\ell \in [N]$) is in $Cl(A)$, query the b coordinates from the blocks of the following set of variables:

$$S = \left\{ R_{f^{-1}(w)} | w \in \{(i-1, j-1), (i-1, j), (i, j-1), (i, j), (i, j+1), (i+1, j), (i+1, j+1)\} \cap V(G) \right\}$$

If one of the induction clauses mentioning only stones in S is falsified (recall that the placement of stones to vertices is the same for all assignments in μ), output the corresponding clause. Otherwise, output an error symbol.

Clearly, the depth of $\mathcal{T}$ is $O(n^{1/3} \log n)$. Thus, by Lemma 16, the probability that it outputs a falsified clause is at most $2/5$. Let A_v denote the affine space at $\mathcal{P}(\beta, t)$. Note that $A \subseteq A_v \implies Cl(A_v) \subseteq Cl(A)$, by Lemma 5. It is straight-forward to verify that if A_v is not α-foolable, then $\mathcal{T}$ successfully outputs a clause falsified by β: if one of the variables belonging to a, b or c is in $Cl(A)$, in the final step the PDT queries the stones placed on a, b, c and detects that an induction clause at a is falsified. The result now follows from Lemma 16. $\qquad\square$

7.4 Foolability Implies Large Rank

In this subsection, we prove there is an obfuscation map $\rho : [N]^3 \to \mathcal{V}$ such that for a foolable node v of a bottom-read-once linear branching program $\mathcal{P}$ computing $Search(\mathsf{SP}_{n,\rho} \circ \mathsf{IP})$, the associated affine space A_v must have large co-dimension.

First, we prove an auxiliary lemma. Let $T \subseteq [m]$ be a subset of the variables of $\mathsf{SP}_{n,\rho}$. We say a stone j is *marked* by T if T contains a variable that mentions the stone j, i.e. $R_j, P_{v,j}$ for any vertex $v \in V, P_{f(j),k}$ for any stone $k \in [N]$, or $Z_{f(j),\ell}$ for any $\ell \in [N-1]$. Let $Q(T) \subseteq [N]$ be the set of stones marked by T.

Lemma 17. *Let ρ be any obfuscation map and $\alpha \in Supp(\mu)$. Let v be the vertex at which the blue path induced by α ends. Let $T \subseteq [m]$ be an α-foolable subset with $|Q(T)| < N/2$. For any $i, j, k \in [N] \setminus Q(T)$, there exists an assignment $\gamma \in \{0, 1\}^m$ extending the restriction $\alpha|_T$ which satisfies all clauses of $\mathsf{SP}_{n,\rho}$ except one of the following two:*

$$C_1 := \neg P_{u,i} \vee \neg R_i \vee \neg P_{w,j} \vee \neg R_j \vee \neg P_{v,k} \vee R_k \vee \rho(i, j, k), \text{ or}$$
$$C_2 := \neg P_{u,i} \vee \neg R_i \vee \neg P_{w,j} \vee \neg R_j \vee \neg P_{v,k} \vee R_k \vee \neg\rho(i, j, k),$$

where u and w are the out-neighbors of v.

Proof. Let $S \subseteq V(G)$ be the set of vertices in the blue path induced by α. We assign stone k to vertex v and stones i, j to u, w respectively. To all other vertices we assign arbitrary stones as long as they are consistent with $\alpha_{|T}$. Formally: pick two stones ℓ_1, ℓ_2 in $[N] \setminus (Q(T) \cup f^{-1}(S) \cup \{i, j, k\})$. Consider the following map $\text{STONE} : V \to [N]$.

$$
\text{STONE}(p) = \begin{cases}
f^{-1}(p) & \text{if } f^{-1}(p) \in Q(T) \\
i & \text{if } p = v \\
j & \text{if } p = u \\
k & \text{if } p = w \\
\ell_1 & \text{if } f^{-1}(p) \notin Q(T) \cup \{i, j, k\}, f^{-1}(p) \in S \\
\ell_2 & \text{if } f^{-1}(p) \notin Q(T) \cup \{i, j, k\} \cup S
\end{cases}
$$

Define a coloring map $\text{COLOR} : [N] \to \{\text{RED}, \text{BLUE}\}$ as follows:

$$
\text{COLOR}(s) = \begin{cases}
\text{RED} & \text{if } s \in Q(T), f(s) \notin S \\
\text{BLUE} & \text{if } s \in Q(T), f(s) \in S \\
\text{BLUE} & \text{if } s = i \\
\text{RED} & \text{if } s = j \\
\text{RED} & \text{if } s = k \\
\text{BLUE} & \text{if } s = \ell_1 \\
\text{RED} & \text{if } s = \ell_2 \\
\text{BLUE} & \text{otherwise}
\end{cases}
$$

We remark the color used in the last case does not matter as these stones are not used in the stone placement given by the map STONE. Let $\gamma \in \{0,1\}^m$ be the assignment which sets the variables according to this placement and coloring map, i.e.,

$$
P_{v,j} = \begin{cases}
1 & \text{if } j = \text{STONE}(v) \\
0 & \text{otherwise}
\end{cases}
$$

$$
Z_{v,j} = \begin{cases}
0 & \text{if } j < \text{STONE}(v) \\
1 & \text{otherwise}
\end{cases}
$$

$$
R_j = \begin{cases}
1 & \text{if } \text{COLOR}(j) = \text{RED} \\
0 & \text{if } \text{COLOR}(j) = \text{BLUE}
\end{cases}
$$

Notice that this assignment is consistent with α on T and it falsifies a single induction clause at v: stone i is placed at v, stones j, k are placed at u, w respectively; stones j, k are red while stone i is blue. We remark that there is no set of clause in $\mathsf{SP}_{n,\rho}$ which forces the placement of stones to vertices to be bijective. Thus, the only clause of $\mathsf{SP}_{n,\rho}$ falsified by γ is one of the following:

$$
C_1 := \neg P_{u,i} \vee \neg R_i \vee \neg P_{w,j} \vee \neg R_j \vee \neg P_{v,k} \vee R_k \vee \rho(i,j,k),
$$
$$
C_2 := \neg P_{u,i} \vee \neg R_i \vee \neg P_{w,j} \vee \neg R_j \vee \neg P_{v,k} \vee R_k \vee \neg\rho(i,j,k),
$$

$\square$

For our hard formula, we use an appropriate obfuscation map ρ given by the following lemma. An analogous lemma was proved by Alekhnovich et al. [1] with different constants as their formula is over a slightly smaller set of constants, but still quadratic in N. For sake of completeness, we state the proof in Appendix.

Lemma 18. *For N sufficiently large, there exists a mapping $\rho : [N]^3 \to \mathcal{V}$ such that for every $Q \subseteq [N]$ with $|Q| \leq N/400$ and every $X \in \mathcal{V}$, there exist $i < j < k \in [N] \setminus Q$ such that $\rho(i,j,k) = X$.*

We remark that the proof of the following lemma is the only place where we are using an obfuscation map with a certain property (given by Lemma 18) and also that the branching program computing $Search(\mathsf{SP}_{n,\rho} \circ \mathsf{IP})$ is bottom-read-once.

Lemma 19. *There exists an obfuscation map $\rho : [N]^3 \to \mathcal{V}$ such that the following holds. Let $\beta \in \mathbb{F}_2^{mb}$, $t > 0$, and $p = \mathcal{P}(\beta, t)$ be a node in a bottom-read-once branching program $\mathcal{P}$ computing $Search(\mathsf{SP}_{n,\rho} \circ \mathsf{IP})$. If p is α-foolable for $\alpha = \overrightarrow{\mathsf{IP}}(\beta)$, then $co\text{-}dim(A_p) \geq \min\{\frac{N}{800}, t\}$.*

Proof. Fix $\rho : [N]^3 \to \mathcal{V}$ to be a map with the property guaranteed by Lemma 18. Let $A_p = \mathcal{S}(M, c)$. Suppose $\mathrm{rank}(M) \leq \frac{N}{800}$. Let S be the set of sinks of $\mathcal{P}$ reachable from p.

Claim 3. *For each variable Y of $\mathsf{SP}_{n,\rho} \circ \mathsf{IP}$ there is a sink in S outputting a clause D such that Y or $\neg Y$ is in D.*

Proof of Claim 3. We shall show that there exists an assignment $\gamma \in A_p$ such that γ falsifies only one clause of $\mathsf{SP}_{n,\rho} \circ \mathsf{IP}$ and that clause contains Y or $\neg Y$. Let $Y \in \mathrm{BLOCK}(Z)$. Let v be the endpoint of the blue path induced by α, and let its children be u and w.

Let $T = \mathrm{Cl}(A_p)$. Note that $|T| < N/800$, thus $Q(T) < N/400$. By Lemma 18, there exist $i < j < k$ in $[N] \setminus Q(T)$ such that $\rho(i, j, k) = Z$. By assumption, T is α-foolable. By Lemma 17, we can extend the restriction $\alpha_{|T}$ to a full assignment $\gamma \in \mathbb{F}_2^m$ such that γ does not falsify any clause of $\mathsf{SP}_{n,\rho}$ other than one of the following two clauses:

$$C_1 := \neg P_{u,i} \vee \neg R_i \vee \neg P_{w,j} \vee \neg R_j \vee \neg P_{v,k} \vee R_k \vee Z$$
$$C_2 := \neg P_{u,i} \vee \neg R_i \vee \neg P_{w,j} \vee \neg R_j \vee \neg P_{v,k} \vee R_k \vee \neg Z.$$

By Lemma 6, there is an assignment $\beta' \in A_p$ such that $\overrightarrow{\mathsf{IP}}(\beta') = \gamma$. Note that β' falsifies only one clause of $\mathsf{SP}_{n,\rho} \circ \mathsf{IP}$, and that clause belongs to either $C_1 \circ \mathsf{IP}$ or $C_2 \circ \mathsf{IP}$. By Observation 2, every clause in $C_1 \circ \mathsf{IP}$ and $C_2 \circ \mathsf{IP}$ contains every variable in the block of $\rho(i, j, k) = Z$ (possibly with negations). Thus, the only clause falsified by β' contains either Y or $\neg Y$. Since $\beta' \in A_p$ and β' falsifies only one clause $\tilde{C}$ of $\mathsf{SP}_{n,\rho} \circ \mathsf{IP}$, one of the sinks in S must be labelled $\tilde{C}$. $\square$

We continue the proof of Lemma 19. Let U be the space spanned by rows of matrices defining the spaces of sinks in S. Formally, let $s \in S$ be labelled by the affine space $A_s = \mathcal{S}(M_s, c_s)$ (this corresponds to a clause of $\mathsf{SP}_{n,\rho} \circ \mathsf{IP}$). Then, $U = \mathrm{Span}(\{\mathcal{R}(M_s) \mid s \in S\})$. By Claim 3 each variable of $\mathsf{SP}_{n,\rho} \circ \mathsf{IP}$ is mentioned at some sink in S, so the space U has full dimension, i.e., $\dim(U) = mb$. Let $W = \mathrm{Span}(\mathcal{R}(M) \cup Post(p))$. By Lemma 2, we have

$$\dim(W) \leq \mathrm{rank}(M) + \dim(Post(p)) \leq \mathrm{rank}(M) + mb - t.$$

On the other hand by Lemma 3, $U \subseteq W$ and thus, $\dim(W) \geq \dim(U) = mb$. Putting both inequalities together, we get $\mathrm{rank}(M) \geq t$.

$\square$

7.5 Lifted Distributions Fool Rank

In the earlier two subsections, we have established the following two facts: (i) in any BROLBP $\mathcal{P}$ corresponding to a bottom-regular ResLin proof of $\mathsf{SP}_{n,\rho} \circ \mathsf{IP}$, when inputs β are sampled according to the IP lift of μ, the node $\mathcal{P}(\beta, t)$ is a foolable node with high probability; (ii) in such a BROLBP, the constraint matrix for the affine space associated with a foolable node has large rank.

To prove that $\mathcal{P}$ has large size, it is sufficient to argue that each large rank constraint system is satisfied with small probability under the IP lift of μ. Of course, such a statement is well known to be true if we sample inputs from the uniform distribution in $\mathbb{F}_2^{bm}$. However, our distribution is not so at all. In particular, it has quite sparse support. Still, it turns out that any lifted distribution is pseudo-random with respect to the rank measure if the gadget satisfies a generalization of the stifling property. Consider a gadget $g : \{0, 1\}^b \to \{0, 1\}$. For any $i \in [b]$, and $o \in \{0, 1\}$ an assignment α to the bits different from i is called *o-stifling for i* if g gets fixed to o by α, i.e., the induced subfunction $g_{|([b] \setminus \{i\}) \leftarrow \alpha}$ gets fixed to the constant function that always evaluates to o, no matter how the i-th bit is set. We say g is *ε-balanced, stifled* if for any $i \in [b]$, and for any $o \in \{0, 1\}$, the following is true: when we sample $x \in \{0, 1\}^b$ uniformly at random from $g^{-1}(o)$, the projection of x on co-ordinates different from i is o-stifling for i with probability at least ε.

Claim 4. *Inner-product defined on $2b \geq 8$ bits is a 7/18-balanced and stifled gadget.*

Proof. By simple manipulation,

$$\Pr_{(x,y)\sim\{0,1\}^{2b}}\left[\mathsf{IP}(x,y)=0 \wedge x_1=0\right] = \mathbb{E}\left[\left(\frac{1+(-1)^{\sum_{i=1}^{b} x_i y_i}}{2}\right)\left(\frac{1+(-1)^{x_1}}{2}\right)\right].$$

The RHS becomes,

$$\frac{1}{4} + \frac{1}{4}\mathbb{E}\left[(-1)^{x_1}\right] + \frac{1}{2}\mathbb{E}\left[(-1)^{\sum_{i=1}^{b} x_i y_i}\right]$$

The second sum is 0, and the third is at most $\frac{1}{2^{b+1}}$. Overall, this gives that

$$\Pr_{(x,y)\sim\{0,1\}^{2b}}\left[\mathsf{IP}(x,y)=0 \wedge x_1=0\right] \geq \frac{1}{4} - \frac{1}{2^{b+1}}.$$

Further, by a similar method,

$$\Pr_{(x,y)\sim\{0,1\}^{2b}}\left[\mathsf{IP}(x,y)=0\right] \leq \frac{1}{2} + \frac{1}{2^b}.$$

Thus,

$$\Pr_{(x,y)\sim\{0,1\}^{2b}}\left[x_1=0\,|\,\mathsf{IP}(x,y)=0\right] \geq \frac{1/4 - 1/2^{b+1}}{1/2 + 1/2^b}.$$

Note that any assignment that sets $x_1 = 0$ stifles y_1. Hence, y_1 is stifled with probability at least 7/18, if $b \geq 4$, by the projection of a random 0 (and similarly 1) assignment to IP. Completely analogously, any bit is stifled with the same probability. $\qquad\square$

Remark 7.1. *It is worth noting that several gadgets, including Inner-Product, Indexing, Majority, even on sufficiently large but constant number of bits, are stifled and balanced.*

Now we state the utility of balanced, stifling gadgets.

Lemma 20. *Let g be any ε-balanced, stifled gadget and $z \in \mathbb{F}_2^m$ be any fixed vector. Then, for every matrix $M \in \mathbb{F}_2^{r \times bm}$ of full rank r, and sny vector $\gamma \in \mathbb{F}_2^r$ the following holds:*

$$\Pr_{\beta\sim\overrightarrow{g}^{-1}(z)}\left[M\beta = \gamma\right] = 2^{-\Omega_\varepsilon(r/b)}.$$

Proof. After Gaussian elimination, turning M into row-echelon form, there are at least r/b different blocks in which pivots of rows appear. Let us call each such block a pivot block. The distribution $\overrightarrow{g}^{-1}(z)$ samples independently at random from $g^{-1}(z_i)$ for each of the i-th block. It will be convenient to think that we sample, one after the other, independently from blocks in this way, starting from the rightmost. Consider the situation when we arrive at a pivot block having sampled all blocks to its right. Let the bit of z corresponding to this pivot block be $o \in \{0,1\}$. Consider any one row that has a pivot in that block. Let the equation corresponding to this row be denoted by ℓ. By the property of g, with probability at least ε the random assignment from $g^{-1}(o)$ will be stifling for the pivot of ℓ. Conditioned on that event, the stifled bit will be set to each of $0,1$ with probability exactly $1/2$ as each possible setting gives rise to a distinct assignment in $g^{-1}(o)$. Hence, the probability that ℓ is satisfied by the sampled assignment to this block is at most $(1 - \varepsilon/2)$. Thus, continuing this way, the probability that all the equations are satisfied is at most $\left(1 - \varepsilon/2\right)^{r/b}$, yielding the desired result. $\qquad\square$

7.6 Putting Everything Together

Now, we are ready to finish the proof of our lower bound, i.e., Theorem 4.

Proof of Theorem 4. Let $\mathcal{P}$ be the BROLBP derived from a bottom-regular ResLin proof of $\mathsf{SP}_{n,\rho} \circ \mathsf{IP}$. Let $t = \lfloor c \cdot n^{1/3} \rfloor$ for an appropriately chosen small constant $c > 0$. Combining Theorem 6 and Lemma 19, we get

$$\Pr_{\beta\sim\mu'}\left[A_{\mathcal{P}(\beta,t)} \text{ has co-dimension } \geq t\right] \geq \frac{3}{5}. \tag{7.3}$$

On the other hand, for any node v of $\mathcal{P}$ which has co-dim$(A_v) \geq t$, Lemma 20 yields,

$$\Pr_{\beta \sim \mu'}\left[\mathcal{P}(\beta, t) \text{ is } v\right] \leq 2^{-\Omega\left(\frac{t}{\log n}\right)}. \tag{7.4}$$

If s is the total number of nodes of $\mathcal{P}$, combining (7.3) and (7.4), we get immediately

$$s \cdot 2^{-\Omega\left(t/\log(n)\right)} \geq \frac{3}{5}.$$

Substituting the value of t in the above, the result immediately follows.

$\square$

8   Future Directions

We provided the first super-polynomial separation between the powers of bottom-regular and general ResLin proofs. We believe the general proof strategy that we implemented, modifying and generalizing the recent technique of Efremenko, Garlík and Itsykson [6], should yield exponential lower bounds on the length of bottom-regular ResLin proofs for other formulas as well. For instance, formula MGT_n which is the constant-width version of GT_n, that encodes the contradiction that a finite total order has no minimal element, when obfuscated appropriately and then lifted with inner-product can be proved to be hard for bottom-regular ResLin. Indeed, the first part of our proof strategy is quite general and applies to all ResLin proofs without any assumption on regularity. Here, we need just the fact that the search for a falsified clause in the base formula is hard on average for small height decision trees w.r.t some distribution μ. That is sufficient, thanks to lifting theorems, to yield the fact that after the first few (typically $n^{\Omega(1)}$) linear queries, the branching program corresponding to the ResLin proof is still far from discovering a falsified clause in the lifted formula with high probability, when the input is sampled from the lifted distribution $\overrightarrow{g}^{-1}(\mu)$. This is step 2 of our proof outline. In most formulas, one could then define some natural notion of *foolability* and then say the affine space associated with nodes of the branching program are frequently foolable. How do we know this is useful? Unfortunately, the usefulness of these notions seem formula-specific. For the binary pigeonhole principle, Efremenko et al. observed that local consistency was enough to yield high rank of the dual of the affine space. For our formula, we achieved the same exploiting the obfuscation map and stifling nature of our lifting gadget. But this seems not immediately generalizable. An interesting direction here is the following:

Problem 1. *Prove strong lower bounds on the size of bottom-regular proofs for the lift of Tseitin formulas over expander graphs.*

Another direction is to consider the formulas where even implementing the first step of our strategy seems impossible.

Problem 2. *Prove strong lower bounds on size of bottom-regular ResLin proofs for appropriate lifts of random constant-width CNF formulas.*

The above seems challenging as for random formulas of constant-width, for every distribution, there exists a decision tree that finds in $O(1)$ queries a falsified clause with high probability. This, very likely, requires changing our technique substantially. Finally, one of the challenges posed by Gryaznov et al. [9] remains still open.

Problem 3. *Prove super-polynomial lower bounds on the size of top-regular ResLin proofs.*

References

[1] Michael Alekhnovich, Jan Johannsen, Toniann Pitassi, and Alasdair Urquhart. An exponential separation between regular and general resolution. *Theory of Computing*, 3(5):81–102, 2007. Preliminary version in STOC, 2002.

[2] Paul Beame and Sajin Koroth. On disperser/lifting properties of the index and inner-product functions. In Yael Tauman Kalai, editor, *14th Innovations in Theoretical Computer Science Conference, ITCS 2023, January 10-13, 2023, MIT, Cambridge, Massachusetts, USA*, volume 251 of *LIPIcs*, pages 14:1–14:17. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2023.

[3] Arkadev Chattopadhyay, Yuval Filmus, Sajin Koroth, Or Meir, and Toniann Pitassi. Query-to-communication lifting using low-discrepancy gadgets. *SIAM J. Comput.*, 50(1):171–210, 2021. Preliminary version in ICALP, 2019.

[4] Arkadev Chattopadhyay, Nikhil S. Mande, Swagato Sanyal, and Suhail Sherif. Lifting to parity decision trees via stifling. In Yael Tauman Kalai, editor, *14th Innovations in Theoretical Computer Science Conference, ITCS 2023, January 10-13, 2023, MIT, Cambridge, Massachusetts, USA*, volume 251 of *LIPIcs*, pages 33:1–33:20. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2023.

[5] Eshan Chattopadhyay and Jyun-Jie Liao. Hardness against linear branching programs and more. In Amnon Ta-Shma, editor, *38th Computational Complexity Conference, CCC 2023, July 17-20, 2023, Warwick, UK*, volume 264 of *LIPIcs*, pages 9:1–9:27. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2023.

[6] Klim Efremenko, Michal Garlík, and Dmitry Itsykson. Lower bounds for regular resolution over parities. *Electron. Colloquium Comput. Complex.*, TR23-187, 2023.

[7] Ankit Garg, Mika Göös, Pritish Kamath, and Dmitry Sokolov. Monotone circuit lower bounds from resolution. *Theory Comput.*, 16:1–30, 2020. Preliminary version in STOC 2018.

[8] Mika Göös, Toniann Pitassi, and Thomas Watson. Query-to-communication lifting for BPP. *SIAM J. Comput.*, 49(4), 2020. Preliminary version in FOCS 2017.

[9] Svyatoslav Gryaznov, Pavel Pudlák, and Navid Talebanfard. Linear branching programs and directional affine extractors. In Shachar Lovett, editor, *37th Computational Complexity Conference, CCC 2022, July 20-23, 2022, Philadelphia, PA, USA*, volume 234 of *LIPIcs*, pages 4:1–4:16. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022.

[10] Dmitry Itsykson and Dmitry Sokolov. Lower bounds for splittings by linear combinations. In Erzsébet Csuhaj-Varjú, Martin Dietzfelbinger, and Zoltán Ésik, editors, *Mathematical Foundations of Computer Science 2014 - 39th International Symposium, MFCS 2014, Budapest, Hungary, August 25-29, 2014. Proceedings, Part II*, volume 8635 of *Lecture Notes in Computer Science*, pages 372–383. Springer, 2014.

[11] Xin Li and Yan Zhong. Explicit directional affine extractors and improved hardness for linear branching programs. *CoRR*, abs/2304.11495, 2023.

[12] Pavel Pudlák. On extracting computations from propositional proofs (a survey). In Kamal Lodaya and Meena Mahajan, editors, *IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2010, December 15-18, 2010, Chennai, India*, volume 8 of *LIPIcs*, pages 30–41. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2010.

[13] Ran Raz and Iddo Tzameret. Resolution over linear equations and multilinear proofs. *Ann. Pure Appl. Log.*, 155(3):194–224, 2008.

[14] Alexander A. Razborov. Unprovability of lower bounds on circuit size in certain fragments of bounded-arithmetic. *Izvestiya. Math.*, 59(1):205–227, 1995.

[15] Alexander A. Razborov. A new kind of tradeoffs in propositional proof complexity. *J. ACM*, 63(2):16:1–16:14, 2016.

[16] Dmitry Sokolov. Dag-like communication and its applications. In Pascal Weil, editor, *Computer Science - Theory and Applications - 12th International Computer Science Symposium in Russia, CSR 2017, Kazan, Russia, June 8-12, 2017, Proceedings*, volume 10304 of *Lecture Notes in Computer Science*, pages 294–307. Springer, 2017.

[17] Alasdair Urquhart. A near-optimal separation of regular and general resolution. *SIAM J. Comput.*, 40(1):107–121, 2011.

[18] Marc Vinyals, Jan Elffers, Jan Johannsen, and Jakob Nordström. Simplified and improved separations between regular and general resolution by lifting. In Luca Pulina and Martina Seidl, editors, *Theory and Applications of Satisfiability Testing - SAT 2020 - 23rd International Conference, Alghero, Italy, July 3-10, 2020, Proceedings*, volume 12178 of *Lecture Notes in Computer Science*, pages 182–200. Springer, 2020.

Appendix

Proof of Lemma 18. We prove a random mapping has the sought property with non-zero probability. We choose ρ uniformly randomly from all possible mappings from $[N]^3$ to $\mathcal{V}$. Let $X \in \mathcal{V}$ and $Q \subseteq [N]$ with $|Q| \leq \varepsilon N$ for $\varepsilon = \frac{1}{400}$. Recall that the number of variable of $\mathsf{SP}_{n,\rho}$ is $m = 2N^2$. Then, we have

$$\Pr[\text{For all } i < j < k \in [N] \setminus Q : \rho(i,j,k) \neq X] = \left(\frac{m-1}{m}\right)^{\binom{(1-\varepsilon)N}{3}}$$

$$\leq \left(1 - \frac{1}{m}\right)^{(1-\varepsilon)^3 N^3/12} \leq e^{-(1-\varepsilon)^3 N/24}$$

By union bound over all possible $X \in \mathcal{V}$ and $Q \subseteq [N]$, we get the following bound for the probability that there is $X \in V$ and $Q \subseteq [N]$ of size εN such that for all triples $i < j < k \in [N] \setminus Q$ holds that $\rho(i,j,k) \neq X$.

$$2N^2 \cdot \binom{N}{\varepsilon N} \cdot e^{-(1-\varepsilon)^3 N/24} \leq 2N^2 \cdot e^{(H(\varepsilon)-(1-\varepsilon)^3/24)N} \leq 2N^2 \cdot e^{-0.0238N}$$

The first inequality holds because $\binom{N}{\varepsilon N} \leq e^{H(\varepsilon)N}$ for $H(\varepsilon) = -\varepsilon \ln \varepsilon - (1-\varepsilon)\ln(1-\varepsilon)$. The second inequality holds for $\varepsilon = \frac{1}{400}$. The last term is clearly less than 1 for large enough N. $\qquad\square$

Proof of Lemma 1. Suppose the set of clauses Φ semantically implies $C := [x_1 = a_1] \vee \cdots \vee [x_k = a_k]$. Then, the CNF formula $\Phi' := \Phi \wedge [x_1 \neq a_1] \wedge \cdots \wedge [x_k \neq a_k]$ is unsatisfiable. We construct a decision tree $\mathcal{T}$ for $Search(\Phi')$ as follows:

Step 1: First, $\mathcal{T}$ queries the variables $x_1, \ldots, x_k$. If it sees that $x_j = a_j$ for some $j \in [k]$, it outputs the clause $[x_j \neq a_j]$.

Step 2: Then, $\mathcal{T}$ queries the variables of Φ' one-by-one and outputs a falsified clause as soon as it finds one.

By the well-known equivalence between tree-like resolution and decision trees, $\mathcal{T}$ can be converted into tree-like resolution refutation for Φ'. Each node v of $\mathcal{T}$ is labelled with a clause C_v. If a node queries variable x_i and has two children v_0, v_1, then C_v can be derived from C_{v_0}, C_{v_1} by resolving x_i. The root is labelled with the empty clause.

Let v be the node of $\mathcal{T}$ where Step 2 starts. The clause C_v must be $C = [x_1 = a_1] \vee \cdots \vee [x_k = a_k]$, because resolving it with $[x_1 \neq a_1], \ldots, [x_k \neq a_k]$ results in the empty clause. Note that every leaf under v is a clause of Φ. Moreover, the subtree rooted in v has size at most 2^n. Thus, $C = C_v$ can be derived from Φ in at most 2^n steps. $\qquad\square$