Improved Pseudorandom Generators for AC^0 Circuits

Xin Lyu[*]

February 17, 2022

Abstract

We give PRG for depth-d, size-m AC^0 circuits with seed length $O(\log^{d-1}(m)\log(m/\varepsilon)\log\log(m))$. Our PRG improves on previous work [TX13, ST19, Kel21] from various aspects. It has optimal dependence on $\frac{1}{\varepsilon}$ and is only one "$\log\log(m)$" away from the lower bound barrier. For the case of $d = 2$, the seed length tightly matches the best-known PRG for CNFs [DETT10, Tal17].

There are two technical ingredients behind our new result; both of them might be of independent interest. First, we develop a "partitioning-based" approach to construct PRGs based on restriction lemmas for AC^0. Previous works [TX13, ST19, Kel21] usually built PRGs on the Ajtai-Wigderson framework [AW89]. Compared with them, our new approach avoids the extra "$\log(n)$" factor that usually arises from the Ajtai-Wigderson framework, allowing us to get the almost-tight seed length. Our partitioning-based approach is quite general, and we believe it can help design PRGs for classes beyond constant-depth circuits.

Second, improving and extending [TX13, ST19, Kel21], we prove a full derandomization of the powerful multi-switching lemma [Hås14]. We show that one can use a short random seed to sample a restriction, such that a family of DNFs simultaneously simplifies under the restriction with high probability. This answers an open question in [Kel21]. Previous derandomizations were either partial (that is, they pseudorandomly choose variables to restrict, and then fix those variables to truly-random bits) or had sub-optimal seed length. In our application, having a fully-derandomized switching lemma is crucial, and the randomness-efficiency of our derandomization allows us to get an almost-tight seed length.

1 Introduction

Let $\mathcal{F}$ be a class of functions. A pseudorandom generator (PRG) for $\mathcal{F}$ is an algorithm $G : \{0,1\}^s \to \{0,1\}^n$ that maps a short random seed x into a longer string $\mathcal{G}(x)$ that appears random to every distinguisher in $\mathcal{F}$. More specifically, we say that $\mathcal{G}$ ε-fools $\mathcal{F}$, if for every $f \in \mathcal{F}$, it holds

$$|\mathbb{E}_{x\sim\mathcal{U}_s}[f(G(x))] - \mathbb{E}_{x\sim\mathcal{U}_n}[f(x)]| \le \varepsilon.$$

In this work, we consider the class of bounded-depth Boolean circuits (aka. AC^0 circuits), a circuit class that has been studied extensively over the past few decades. Constructing PRGs for AC^0 circuits is a central problem that has been studied extensively [AW89, Has89, NW94, Bra10, Tal17, HS19, TX13, ST19, Kel21].

Quantitatively, let $\mathsf{AC}^0_d[m(n)]$ be the class of depth-d, size-m circuits. By the probabilistic method, one can show that there exists an ε-PRG for $\mathsf{AC}^0_d[m(n)]$ with seed length $O(\log(m/\varepsilon))$. However, finding an explicit PRG with the same seed length seems beyond current technique. In particular, any PRG for $\mathsf{AC}^0_d[m(n)]$ with seed length $\log^{o(d)}(m)$ would give a non-trivial PRG for

---

[*]Department of EECS, University of California at Berkeley. Email: `xinlyu@berkeley.edu`

NC^1 and imply that $\mathsf{NP} \not\subseteq \mathsf{NC}^1$ (see, e.g., [GW13, Appendix A]). Also, it was observed in [TX13, ST19, Kel21] that, if there is an explicit pseudorandom generator with seed length $o(\log^d(m/\varepsilon))$ for $\mathsf{AC}^0_d[m(n)]$, then there is an explicit function that requires AC^0_d-circuits of size $2^{\omega(n^{1/(d-1)})}$ to compute, improving Håstad's lower bound [Has89] that has resisted attack for more than 30 years!

There is an extensive line of work [AW89, Has89, NW94, Bra10, Tal17, HS19, TX13, ST19, Kel21] aiming to construct better and better PRGs for AC^0. The seminal paper by Ajtai and Wigderson [AW89] gave the first non-trivial pseudorandom generator for AC^0. Their PRG has seed length $n^{o(1)}$ for polynomial-size AC^0 circuits. Later, Nisan constructed PRG for AC^0 circuits by applying Håstad's correlation bound [Has89] to the Nisan-Wigderson "hardness-to-randomness" framework [NW94]. Nisan's PRG has seed length $\log^{2d+O(1)}(m/\varepsilon)$ when ε-fooling $\mathsf{AC}^0_d[m(n)]$ circuits. A breakthrough result by Braverman [Bra10] showed that any $\log^{O(d^2)}(m/\varepsilon)$-wise independent distribution ε-fools $\mathsf{AC}^0_d[m(n)]$. Combined with the standard construction of k-wise independent distribution, this gave a PRG with seed length $\log^{O(d^2)}(m/\varepsilon)$. Braverman's analysis was further sharpened by Tal [Tal17] and by Harsha and Srinivasan [HS19], bringing the seed length down to $\log^{3d+O(1)}(m)\log(1/\varepsilon)$. The Ajtai-Wigderson technique was revisited by Trevisan and Xue [TX13], who constructed a PRG with seed length $\log^{d+O(1)}(m/\varepsilon)$. Recently there were two incomparable improvements over the Trevisan-Xue result, one by Servedio and Tan [ST19] with seed length $\log^{d+O(1)}(m) \cdot \log(1/\varepsilon)$ (i.e., it had optimal dependence on $\frac{1}{\varepsilon}$), and the other by Kelley [Kel21], who got seed length $\widetilde{O}(\log^d(m/\varepsilon)\log n)$.

1.1 Our Result

The main result of this work is a new PRG for $\mathsf{AC}^0_d[m]$ with improved seed length $O(\log^{d-1}(m) \cdot \log(m/\varepsilon) \cdot \log\log m)$.

Theorem 1. *For every $d \in \mathbb{N}$ the following is true. For every $m, n \in \mathbb{N}$ such that $m \geq n$ and every $\varepsilon > 0$, there is an ε-PRG for $\mathsf{AC}^0_d[m]$ circuits with seed length $O(\log^{d-1}(m) \cdot \log(m/\varepsilon) \cdot \log\log(m))$.*

Our PRG construction improves two incomparable results by Servedio, Tan [ST19] and Kelley [Kel21]. Its seed length has optimal dependence on $\frac{1}{\varepsilon}$, and is only one "$\log\log m$" away from the barrier of Håstad's lower bounds [Has89, Hås14]. For the case of $d = 2$, the seed length becomes $O(\log(m)\log(m/\varepsilon)\log\log m)$, tightly matching the best-known PRG for CNFs [DETT10, Tal17]. Furthermore, if the $\log\log(m)$ term in the PRG for CNF can be shaved, then our construction directly implies PRG for depth-d circuits with seed length $O(\log^{d-1}(m)\log(m/\varepsilon))$, tightly matching current hardness bounds for AC^0 circuits. Interpreted from the "hardness-to-randomness" perspective [NW94], our result has converted *almost* all the "hardness" against AC^0 into pseudorandomness for AC^0.

2 Techniques

Our PRG crucially depends on two new technical ingredients. Both of them might be of independent interest. First, we show a framework to construct PRGs based on switching lemmas[1]. Our framework shares some similarities with the seminal Ajtai-Wigderson framework [AW89] but achieves shorter seed length. Second, improving and extending results from [TX13, ST19, Kel21], we show a fully-derandomized multi-switching lemma for small-width DNFs. That is to say, we give an algorithm that samples a pseudorandom restriction from a short random seed, such that a

---

[1]More generally, just like the Ajtai-Wigderson framework, our framework can apply to any kind of "simplify-under-restriction" lemmas (e.g., the shrinkage lemma for De-Morgan formulae).

family of DNFs *simultaneously* simplifies under the restriction with high probability. Applying our framework with the new derandomization gives PRGs for AC^0 circuits with the claimed seed length.

Notation. We define some useful pieces of notation first. Let $f\colon \{0,1\}^n \to \{0,1\}$ be a function. Let $\Lambda \subseteq [n]$ be a set and $x \in \{0,1\}^n$ be a string. A set-string pair (Λ, x) gives a restriction to f. The restricted function is denoted by $f|_{\Lambda[x,\star]}$ and is defined as $f|_{\Lambda[x,\star]}(y) = f(\Lambda[x,y])$, where

$$\Lambda[x,y]_i = \begin{cases} x_i & i \notin \Lambda \\ y_i & i \in \Lambda \end{cases}.$$

Intuitively, this means the all but the Λ part of the input is fixed to the corresponding bits in x, and $f|_{\Lambda[x,\star]}$ is now only a function of those Λ bits.

Let $\boldsymbol{\Lambda} \subseteq [n]$ be a random variable. We say that $\boldsymbol{\Lambda}$ has marginal p, if for each $i \in [n]$ it holds that $\Pr[i \in \boldsymbol{\Lambda}] = p$. We say that $\boldsymbol{\Lambda}[\mathbf{U}, \star]$ is a (truly) p-random restriction, if each $i \in [n]$ is included in $\boldsymbol{\Lambda}$ independently with probability p, and $\mathbf{U}$ is a uniformly random n-bit string. We always use $\mathbf{U}$ to denote the uniform distribution over $\{0,1\}^n$.

2.1 A Partitioning-Based PRG

Our almost-tight (with respect to the lower bounds barrier) seed length crucially depends on a new framework to construct PRGs, called "partitioning-based" approach. Previous best PRGs for AC^0 [ST19, Kel21] were using the iterative restriction framework, developed by Ajtai and Wigderson in their seminal work [AW89]. Using our approach, we get simpler proof for the correctness of our PRG with even improved seed length. We believe the partitioning-based approach could also apply to function classes beyond bounded-depth circuits.

The PRG Template. We briefly describe our construction. Suppose we want to design a PRG for a circuit class $\mathcal{C}_{goal}$. What we have is a pseudorandom distribution $\mathbf{X} \in \{0,1\}^n$ for another related circuit class $\mathcal{C}_{simple}$. We further assume a derandomized "simplify-under-restriction" lemma: there is an integer $k \geq 1$, a real $p > 0$ and a pseudorandom distribution $\mathbf{Y} \in \{0,1\}^n$ satisfying the following:

- Let $\boldsymbol{\Lambda} \subseteq [n]$ be a k-wise independent set with marginal p. Then for every $\mathcal{C}_{goal}$ circuit C, with high probability over $\boldsymbol{\Lambda}, \mathbf{Y}$, the restricted circuit

$$C|_{\boldsymbol{\Lambda}[\mathbf{Y},\star]}(x) := C(\boldsymbol{\Lambda}[\mathbf{Y}, x])$$

  is in $\mathcal{C}_{simple}$.

Then, we choose $w = \frac{1}{p}$ and let $\mathbf{H} : [n] \to [w]$ be a k-wise independent hash function. Consider the following distribution

$$\mathbf{B} = \mathbf{Y} \oplus (\mathbf{X}^{(1)} \wedge \mathbf{H}_1) \oplus (\mathbf{X}^{(2)} \wedge \mathbf{H}_2) \oplus \cdots \oplus (\mathbf{X}^{(w)} \wedge \mathbf{H}_w).$$

Here, $\oplus, \wedge$ denote bitwise $\mathsf{XOR}, \mathsf{AND}$ respectively. $\mathbf{X}^{(1)}, \ldots, \mathbf{X}^{(w)}$ are w independent copies of $\mathbf{X}$.

The idea is, for every $i \in [w]$, if we zoom in and check the set $\mathbf{H}^{-1}(i)$, we see that $\mathbf{H}^{-1}(i)$ is a k-wise independent set with marginal $\frac{1}{w}$. Let $C \in \mathcal{C}_{goal}$ be a circuit that we wish to fool. Imagine that we sample all but the $\mathbf{X}^{(i)}$ part of $\mathbf{B}$ first. We then calculate $\mathbf{Z}_i := \mathbf{Y} \oplus \bigoplus_{j \neq i}(\mathbf{X}^{(j)} \wedge \mathbf{H}_j)$ and consider the restricted function $C|_{\boldsymbol{\Lambda}[\mathbf{Z}_i,\star]}$. We hope that with high probability over $\mathbf{H}$ and $\mathbf{Z}_i$, the

restricted function is in $\mathcal{C}_{simple}$, which can be fooled by $\mathbf{X}^{(i)}$. As we will show, this is indeed the case with one minor technicality[2]. Therefore, we conclude that C cannot distinguish $\mathbf{B}$ from another distribution where we replace the $\mathbf{X}^{(i)}$ part in $\mathbf{B}$ with a uniform random string $\mathbf{U}^{(i)}$. Applying a hybrid argument allows us to show that C fails to distinguish between $\mathbf{B}$ and

$$\mathbf{B}' = \mathbf{Y} \oplus (\mathbf{U}^{(1)} \wedge \mathbf{H}_1) \oplus (\mathbf{U}^{(2)} \wedge \mathbf{H}_2) \oplus \cdots \oplus (\mathbf{U}^{(w)} \wedge \mathbf{H}_w) \equiv \mathbf{U},$$

implying that $\mathbf{B}$ fools $\mathcal{C}_{goal}$ circuits.

Assume the seed length to sample $\mathbf{Y}, \mathbf{H}$ is short enough so that it would not be the bottleneck to sample $\mathbf{B}$. Then, the seed length for sampling $\mathbf{B}$ is larger than that for $\mathbf{X}$ by a factor of $w = O(1/p)$.

Application to AC^0 circuits. In the next section, we will show a derandomized simplify-under-restriction lemma (i.e., the derandomized Håstad's multi-switching lemma) for AC^0 circuits. For now let us assume the lemma, which works with parameter $\frac{1}{p} = O(\log m)$ and simplifies depth-d circuits to depth-$(d-1)$ circuits with high probability. Plugging the lemma in our framework gives a PRG for AC^0_d circuits with seed length longer than AC^0_{d-1}-PRG by $O(\log m)$. Currently, the best PRG for depth-2 circuits (i.e., CNF/DNFs) has seed length $O(\log(m) \log(m/\varepsilon) \log \log m)$ ([DETT10, Tal17]). Using it as our starting point, for every $d \geq 3$ we construct a PRG for depth-d circuits with seed length $O(\log^{d-1}(m) \log(m/\varepsilon) \log \log(m))$, as claimed.

Comparison with the Ajtai-Wigderson framework. Ajtai and Wigderson were the first to apply restriction lemmas to construct PRGs [AW89]. They developed the so-called "iterative restrictions" framework and gave the first non-trivial PRG for AC^0 circuits. Compared with the Nisan-Wigderson "hardness-to-randomness" framework [NW94], the Ajtai-Wigderson framework can "open up" the black box of lower bounds proof, which enables us to construct short PRGs for some delicate circuit classes (e.g., read-once AC^0 formulae [GMR+12]). For these reasons, the Ajtai-Wigderson framework has been increasingly popular in recent years, and its applications went far beyond AC^0 [GMR+12, TX13, HLV18, LV20, FK18, MRT19, DMR+21].

It would be instructive to compare our new approach with their framework. In the following, we briefly review their framework first. Let $t = \Theta(\log n/p)$ be a parameter. Let $\mathbf{X}^{(1)}, \ldots, \mathbf{X}^{(t)}$ be t independent copies of $\mathbf{X}$ (the pseudorandom distribution for $\mathcal{C}_{simple}$). We sample a list of random sets $\mathbf{\Lambda}^1, \ldots, \mathbf{\Lambda}^t$ as follows.

- First, sample $\mathbf{\Lambda}^1 \subseteq [n]$ being a k-wise independent p-marginal subset of $[n]$.

- Having observed $\mathbf{\Lambda}^1$, we sample $\mathbf{\Lambda}^2 \subseteq [n] \setminus \mathbf{\Lambda}^1$ in a k-wise independent and p-marginal way.

- For every $i \geq 3$. We first observe $\mathbf{\Lambda}^1 \cup \cdots \cup \mathbf{\Lambda}^{i-1}$ and then sample $\mathbf{\Lambda}^i \subseteq [n] \setminus (\bigcup_{j=1}^{i-1} \mathbf{\Lambda}^j)$, also in a k-wise independent and p-marginal way.

In the real implementation, we can first sample $\mathbf{\Lambda}^t \subseteq [n]$ and then subtract $\bigcup_{j=1}^{i-1} \mathbf{\Lambda}^j$ from it. Given these primitives, the Ajtai-Wigderson PRG outputs

$$\mathbf{D} = (\mathbf{X}^{(1)} \wedge \mathbf{\Lambda}^1) \oplus (\mathbf{X}^{(2)} \wedge \mathbf{\Lambda}^2) \oplus \cdots \oplus (\mathbf{X}^{(t)} \wedge \mathbf{\Lambda}^t).$$

We observe that with high probability, $\mathbf{\Lambda}^1 \sqcup \cdots \sqcup \mathbf{\Lambda}^t$ forms a partition of $[n]$. The proof of correctness is also by a hybrid argument. We observe two major differences between the Ajtai-Wigderson framework and ours.

---

[2]Specifically, note that there might be a correlation between $\mathbf{H}$ and $\mathbf{Z}_i$, because $\mathbf{H}$ determines which part of each $\mathbf{X}^{(j)}$ gets added to $\mathbf{Z}_i$. We show how to handle this issue in Section 2.2. See also Section 4 for the formal proof of the construction.

1. As an advantage, the Ajtai-Wigderson framework does not need to sample $\mathbf{Y}$, and a partial derandomization of the "simplify-under-restriction" lemma suffices for applying Ajtai-Wigderson. That is, it only requires that the circuit $C|_{\mathbf{\Lambda}[\mathbf{U},\star]}$ simplifies with high probability over a partially-pseudorandom restriction $(\mathbf{\Lambda}, \mathbf{U})$, where the restriction set $\mathbf{\Lambda}$ is pseudorandom and the string $\mathbf{U}$ is truly random.

   To see why this is true, we look into the analysis of the hybrid argument. For example, consider comparing the hybrid distribution

   $$\mathbf{D}^{(0)} = (\mathbf{U}^{(1)} \wedge \mathbf{\Lambda}^1) \oplus (\mathbf{U}^{(2)} \wedge \mathbf{\Lambda}^2) \oplus \cdots \oplus (\mathbf{U}^{(t)} \wedge \mathbf{\Lambda}^t),$$

   with

   $$\mathbf{D}^{(1)} = (\mathbf{X}^{(1)} \wedge \mathbf{\Lambda}^1) \oplus (\mathbf{U}^{(2)} \wedge \mathbf{\Lambda}^2) \oplus \cdots \oplus (\mathbf{U}^{(t)} \wedge \mathbf{\Lambda}^t).$$

   For simplicity, let us assume that $\mathbf{\Lambda}^1 \sqcup \cdots \sqcup \mathbf{\Lambda}^t$ always covers $[n]$. Then we have $\mathbf{D}^{(0)} = \mathbf{\Lambda}^1[\mathbf{U}, \mathbf{U}^{(1)}]$ and $\mathbf{D}^{(1)} = \mathbf{\Lambda}^1[\mathbf{U}, \mathbf{X}^{(1)}]$, which enable us to apply the partially-derandomized restriction lemma.

2. However, when there is a fully-derandomized restriction lemma, using our approach results in a PRG of shorter seed length. Note that the Ajtai-Wigderson framework partitions the $[n]$ coordinates into $t = \Theta(\log n/p)$ blocks and fills in each block with independent pseudorandom strings. Since the set $\mathbf{\Lambda}^i$ covers (roughly) p-fraction of *currently uncovered* coordinates at each time $i \in [t]$, it is crucial to set $t = \Omega(\log n/p)$ so that $\bigcup_{j=1}^{t} \mathbf{\Lambda}^j$ covers $[n]$ with high probability. Our construction, on the other hand, samples a k-wise independent hash function $\mathbf{H} \colon [n] \to [w]$, which naturally induces a partition $\mathbf{H}^{-1}(1) \sqcup \cdots \sqcup \mathbf{H}^{-1}(w)$. Then we only need $w = O(1/p)$ independent samples of $\mathbf{X}$ to complete the construction. In other words, we save the $\log(n)$ overhead by exploiting the symmetry between blocks in our design.

Concluding remarks. Our partitioning-based framework is quite general: for every scenario that Ajtai-Wigderson applies, if we can prove a fully-derandomized "simplify-under-restriction" lemma, then we may hope to use the new framework to shave the $\log(n)$ overhead in seed length. Two more concrete examples are sparse $\mathbb{F}_2$-polynomials [ST19] and small-size De-Morgan formulae [IMZ12, HHTT21]. However, the $\log(n)$ overhead from the Ajtai-Wigderson framework is minor in those applications. For example, the PRG for S-sparse $\mathbb{F}_2$-polynomials has seed length $2^{O(\sqrt{S})}$ [ST19]. Improving a $\log(n)$ factor here is not as significant as for AC^0 circuits.

In the case of (arbitrary-order) read-once branching programs, Forbes and Kelley [FK18] have shown PRGs with seed length $O(\log^3 n)$ and $\widetilde{O}(\log^2 n)$ for general and constant-width ROBPs, respectively. Their PRGs are based on the Ajtai-Wigderson framework, and both of them have an extra $\log(n)$ overhead in seed length due to this reason. It would be exciting to see if one can use the partitioning-based approach to give improved PRG for these models. If this turns out to be true, then we may hope for a simpler construction of nearly-logarithmic seed PRG for ROBPs of all constant width, which would be a major advance in the derandomization of small-space computation. Currently, we only know a nearly-logarithmic seed PRG for width-3 ROBP [MRT19], whose proof seems hard to generalize to larger widths.

We also note that Meka and Zuckerman [MZ13] have used a similar strategy to construct PRGs for low-degree polynomial threshold functions (PTFs). Their PRG also partitions $[n]$ coordinates into small blocks by a bounded-independent hash function and fills in each block with independent but pseudorandom bits. However, their analysis was completely different from ours, nor did they need to add a noise string "$\mathbf{Y}$" to fool any restriction lemma. As far as we are aware, our framework is novel in constructing PRG based on restriction lemmas.

2.2 Derandomized Multi-Switching Lemma

The second technical ingredient behind our result is a fully-derandomized multi-switching lemma for small-width DNFs.

Håstad's switching lemma. Switching lemmas are perhaps the most powerful and versatile tools in analyzing low-depth Boolean circuits, with applications ranging from proving lower bounds [Has89, Hås14, Vio21], constructing pseudorandom generators [AW89, TX13, ST19, Kel21], learning of AC^0 functions [LMN93], designing circuit-analysis algorithms for AC^0 [BIS12, IMP12], to proving Fourier-analytic properties of AC^0 [LMN93, Tal17], to name a few.

The standard switching lemma, originally proved by Håstad, says that for a width-w DNF[3] F, if we apply a $\frac{1}{20w}$-random restriction $(\mathbf{\Lambda}, \mathbf{x})$, then with probability $1 - \varepsilon$, $F|_{\mathbf{\Lambda}[\mathbf{x},\star]}$ collapse to a decision tree of depth $O(\log(1/\varepsilon))$. We can also prove a switching lemma for small-size DNFs: suppose F is a size-m unbounded-width DNF. Then, applying a $\frac{1}{\log(m/\varepsilon)}$-random restriction collapses F to a depth-$O(\log(m/\varepsilon))$ decision tree with probability $1 - \varepsilon$.

In Section 5, we show a derandomization for the standard switching lemma. That is, we prove

Lemma 1 (Derandomized Switching Lemma, slightly-simplified). *Let $k, m \geq 1$ be integers, and $\varepsilon, p > 0$ be reals. Let $F = \bigvee_{i=1}^{m} C_i$ be a size-m width-k DNF over inputs $\{0,1\}^n$. For all $t \geq 1$, let $(\mathbf{\Lambda}, \mathbf{x})$ be any joint random variable such that:*

- *$\mathbf{\Lambda}$ is a $(t+k)$-wise p-marginal subset of $[n]$.*

- *Conditioning on $\mathbf{\Lambda}$, $\mathbf{x}$ is a random string that ε-fools CNF of size at most m.*

Consider the random restriction $F|_{\mathbf{\Lambda}[\mathbf{x},\star]}$, we have:

$$\Pr_{\mathbf{\Lambda},\mathbf{x}}[\mathsf{DT}(F|_{\mathbf{\Lambda}[\mathbf{x},\star]}) > t] \leq (10kp)^t + (4m)^{t+k} \cdot \varepsilon.$$

See Section 5 for the stronger statement and the formal proof. Also note that we allow correlation between $\mathbf{\Lambda}$ and $\mathbf{x}$, which is crucial to apply the lemma in our PRG framework.

One can already construct a $\frac{1}{m}$-error PRG for $\mathsf{AC}^0_d[m]$ with seed length $O(\log^d(m) \log\log(m))$ based on Lemma 1. Let $C \in \mathsf{AC}^0_d[m]$ be a size-m depth-d circuit. We assume that each bottom-layer gate of C has fan-in bounded by $O(\log(m))$ for simplicity[4]. We apply Lemma 1 with $p = \frac{1}{c \log m}, t = c \log(m)$, and $\varepsilon = 2^{-c \log^2(m)}$ for some large constant $c > 1$. We also take $\mathbf{x}$ as a pseudorandom string that ε-fools CNF. Then with probability at least $1 - \frac{1}{m^2}$ over the pseudorandom restriction $(\Lambda, \mathbf{x})$, every depth-2 sub-circuit of C simplifies to a depth-t decision tree, which means that we can express $C|_{\mathbf{\Lambda}[\mathbf{x},\star]}$ as a depth-$(d-1)$ circuit of size poly(m). Hence, assuming we can fool depth-$(d-1)$ circuit with seed length $O(\log^{d-1}(m) \log\log(m))$, then we can fool depth-d circuit with seed length $O(\log^d(m) \log\log(m))$ by applying the partitioning-based PRG. Here we have omitted the seed length to sample $\mathbf{Y}$ and $\mathbf{H}$. It turns out they will not be the bottleneck: see Section 4 for the details.

---

[3]The width of a DNF F is defined as the maximum number of variables in any term of F.

[4]This assumption can be met by first applying a $\frac{1}{2}$-(pseudo)random restriction, because with high probability every bottom-layer gate with large fan-in is killed under such a restriction.

Multi-switching lemma. For the case that $\varepsilon < m^{-\omega(1)}$, using Lemma 1 may result in a longer seed length. In fact, to simplify the circuit with probability at least $1 - \varepsilon$, one must take the "t" parameter in Lemma 1 as $\Theta(\log(m/\varepsilon))$. Then, applying Lemma 1 once simplifies C to a depth-$(d-1)$ circuit with bottom fan-in bounded by $t = \Theta(\log(m/\varepsilon))$. To further apply the lemma, one has to set p as $\frac{1}{\Omega(\log(m/\varepsilon))}$ to make the probability bound in Lemma 1 non-trivial. Therefore, the depth-d PRG would have seed length longer than the depth-$(d-1)$ PRG by $\frac{1}{p} = \Omega(\log(m/\varepsilon))$, bringing the total seed length to $\Omega(\log^d(m/\varepsilon)\log\log(m))$.

If we insist on using a $\frac{1}{\log(m)}$-random restriction and want to have the same $(1 - \varepsilon)$ probability guarantee, we can use the multi-switching lemma. We give its statement first.

Lemma 2. *Let $t, w, k, n, m \geq 1$ be integers. Let $p, \delta > 0$ be reals. Let $\mathcal{F} = \{F_1, \ldots, F_m\}$ be a list of size-m width-k DNFs on inputs $\{0,1\}^n$. Let $(\Lambda, \mathbf{x})$ be a joint random variable satisfying the following:*

- *Λ is a $(t + k)$-wise p-marginal subset of $[n]$,*

- *Conditioning on Λ, $\mathbf{x}$ is an n-bit random string that δ-fools size-(m^2) CNF.*

Then with probability at least $1 - \left(4m^{t/w}(24pk)^t + (24m)^{t+k} \cdot \delta\right)$ over $(\Lambda, \mathbf{x})$, there exists a common w-partial depth-t decision tree[5] for $\mathcal{F}|_{\Lambda[\mathbf{x},\star]}$. That is, we can construct a list of decision trees $T_1, \ldots, T_m$ computing $F_1|_{\Lambda[\mathbf{x},\star]}, \ldots, F_m|_{\Lambda[\mathbf{x},\star]}$. Each T_i is of depth at most $(t + w)$, and all of T_i's share the same query strategy in the first t queries.

Let $C \in \mathsf{AC}^0_d[m]$ be a circuit with bottom fan-in bounded by $k = \log(m)$. Let $\mathcal{F}$ denote the family of depth-2 sub-circuits of C. Apply Lemma 2 on $\mathcal{F}$ with $\frac{1}{p} = O(\log(m))$, $t = O(\log(m/\varepsilon))$, $w = O(\log(m))$ and $\delta = \frac{\varepsilon}{m^{O(t)}}$. We know that $\mathcal{F}|_{\Lambda[\mathbf{x},\star]}$ fails to simplify with probability at most

$$\left(4m^{t/w}(24pk)^t + (24m)^{t+k} \cdot \delta\right) \leq \varepsilon.$$

When $\mathcal{F}$ does simplify, we can compute $C|_{\Lambda[\mathbf{x},\star]}$ by a hybrid model: a depth-t decision tree with $\mathsf{AC}^0_{d-1}[m \cdot 2^w]$-circuits on leaves. The decision tree part performs t adaptive queries according to the common partial decision tree of $\mathcal{F}$. After that, functions in $\mathcal{F}$ can be expressed as depth-w decision trees, which means that C can be computed by an $\mathsf{AC}^0_{d-1}[m \cdot 2^w]$ circuit.

If we can fool every depth-$(d-1)$ circuit on the leaves with error $\frac{\varepsilon}{2^t}$, then we can fool this hybrid model with error ε. Assuming a $O(\log^{d-2}(m)\log(m/\varepsilon)\log\log(m))$ PRG for depth-$(d-1)$ circuits, fooling this hybrid model requires seed length

$$O\left(\log^{d-2}(m \cdot 2^w)\log(m2^t/\varepsilon)\log\log(m)\right) = O(\log^{d-2}(m)\log(m/\varepsilon)\log\log(m)).$$

This allows us to construct a PRG for depth-d circuits with seed length $O(\log^{d-1}(m) \cdot \log(m/\varepsilon) \cdot \log\log(m))$, as claimed.

Proof intuition. When the restriction string is truly random, Kelley's technique [Kel21] shows a clear picture about what makes a random restriction $\Lambda[\mathbf{U}, \star]$ bad. Fix an $\mathsf{AC}^0_d[m]$ circuit C. Given a restriction $\Lambda[\mathbf{U}, \star]$, we want to know whether C simplifies under $\Lambda[\mathbf{U}, \star]$. Roughly speaking, Kelley shows that one can first observe the restriction string $\mathbf{U}$ and come up with a list of $t = \log(m)^{\log(m)}$ sets $S_1, \ldots, S_t$, each of size $c\log(m)$ for a large constant $c \geq 1$. Then, we look at the set Λ: C

---

[5]See Section 3.2 for the formal definition of "partial decision tree".

fails to simplify under $\mathbf{\Lambda}[\mathbf{U}, \star]$, only when $\mathbf{\Lambda}$ contains at least one set S_i from the list. Since $\mathbf{\Lambda}$ is $O(\log(m))$-wise $\frac{1}{c\log(m)}$-marginal, this happens with probability at most $2^{-c\log(m)}$ by a simple union bound.

We apply Kelley's technique to derandomize the multi-switching lemma [Hås14]. The proof of the multi-switching lemma involves many tricks and technicalities. Here we try to give some (over-simplified) intuition. At a very high level, the multi-switching lemma is proved by combining the standard switching lemma with a union bound. If $\mathcal{F}|_\rho$ fails a have w-partial depth-t decision tree, then there is a subset of *at most* $\frac{t}{w}$ formulae in $\mathcal{F}_\rho$, such that the summation of their decision tree complexities exceeds t. This is because every formula with decision tree complexity no larger than w can be handled "for free". Therefore, each bad formula contributes at least w to the summation. There are at most $m^{t/w} = 2^{O(t)}$ such subsets. For each of them, we bound the probability that the summation of their DT complexities exceeds t by $O(kp)^t$. This step is rather similar (in spirit) to the case of standard switching lemma, and Kelley's technique applies.

The final piece in our analysis is the full derandomization. By Kelley's technique, we know that the partially-pseudorandom restriction $\mathbf{\Lambda}[\mathbf{U}, \star]$ is as good as a truly random one. We further derandomize the random string by using the techniques by Trevisan, Xue, and by Servedio, Tan [TX13, ST19]. Specifically, they constructed bounded-depth circuits (called "testers") that take a restriction as input and decide whether the restriction is good or not (for simplifying the target circuit). Now, consider sampling a string $\mathbf{x}$ from a distribution that fools bounded-depth circuits, Given the tester, we can show that $\mathbf{\Lambda}[\mathbf{x}, \star]$ is as good as $\mathbf{\Lambda}[\mathbf{U}, \star]$. Since fooling higher-depth circuits requires longer random bits, to control the final seed length of our PRG, we need the tester to be implementable in AC_2^0. For the standard switching lemma, the Trevisan-Xue tester [TX13] does have depth 2. For the multi-switching lemma, things become a bit trickier: the Servedio-Tan tester [ST19] was designed as a depth-3 circuit and did the test faithfully. We (implicitly) implemented a "upper-side approximator" of their tester. Our one-sided tester can be expressed as a CNF, and is equally useful when upper-bounding the probability of picking bad restrictions.

See our derandomized switching lemmas for the standard and multi-switching versions in Section 5 and Section 6, respectively. Instead of playing with decision trees and tracing down query paths, we strive to present the proof based on the "canonical query algorithm". Our proof is more operational and, in our opinion, easier to follow.

Comparison with previous works. For the task of designing PRGs for AC^0 circuits, before our result, there were two incomparable results on the frontier, one by Servedio and Tan [ST19] and the other by Kelley [Kel21]. Both of them were built on derandomization results for switching lemmas.

The Servedio-Tan PRG is based on a derandomization of Håstad's multi-switching lemma [Hås14], and has seed length $\log^{d+O(1)}(m)\log(1/\varepsilon)$ when ε-fooling $\mathsf{AC}_d^0[m(n)]$. Due to the usage of multi-switching lemma, their PRG has optimal dependence on the error parameter ε. However, their derandomization of the switching lemma is weaker in seed length. The two factors together determined the final seed length of their PRG.

Kelley's PRG, on the other hand, is based on a stronger derandomization of the standard switching lemma [Has89]. It has seed length $\widetilde{O}(\log^d(m/\varepsilon)\log(n))$. The exponent on $\log(m)$ matches the lower bound barrier, credit to the fact that their stronger derandomization allows one to sample a restriction using a much shorter seed. However, Kelley only showed a partial derandomization, which is not applicable in our framework. Also, the dependence on $\frac{1}{\varepsilon}$ is inferior due to the somewhat coarse analysis in the standard switching lemma. It was left as an open question in [Kel21] whether one can get the same high-equality derandomization of the multi-switching lemma and optimize the dependency on $\frac{1}{\varepsilon}$.

We answer this question in the affirmative by showing a fully-derandomized multi-switching lemma that improves both works. Combined with the partitioning-based PRG framework, our lemma gives a PRG for AC^0 with an almost tight seed length. We hope our derandomization of the switching lemmas could find applications in other contexts.

Finally, we remark that the "decision-tree-followed-by-circuit" type hybrid model also appears in many previous works. The applications include proving correlation bounds and Fourier spectrum bounds [Hås14, Tal17], constructing PRGs [ST19, HHTT21], designing circuit-analysis algorithms [CSS18], etc.

3 Preliminaries

In this section, we set up necessary pieces of notation, and review some well-known and useful facts from the literature of pseudorandmoness and complexity theory.

3.1 Restrictions, Partial-Assignments and Strings

We use the term "restriction" and "partial assignment" interchangeably. Both of them refer to a string of the form $\rho \in \{0, 1, \star\}$. Here, if $\rho_i = 0/1$, it means the i-th bit of ρ is fixed to that value. Otherwise, the i-th bit of ρ is unfixed.

For two partial assignments $\rho, \sigma \in \{0, 1, \star\}$, define their composition $\rho \circ \sigma$ as:

$$(\rho \circ \sigma)_i = \begin{cases} \rho_i & \rho_i \neq \star \\ \sigma_i & o.w. \end{cases}.$$

Note that the left partial assignment always has a higher priority than the right one.

Let $\Lambda \subseteq [n]$ be a set. For two partial assignments $\rho, \sigma \in \{0, 1, \star\}^n$, let $\Lambda[\rho, \sigma]$ be the assignment defined as

$$\Lambda[\rho, \sigma]_i = \begin{cases} \rho_i, & i \notin \Lambda \\ \sigma_i, & i \in \Lambda \end{cases}.$$

Let $f : \{0, 1\}^n \to \{0, 1\}$ be a function and $\rho \in \{0, 1, \star\}^n$ be a partial assignment. We use $f|_\rho \colon \{0, 1\}^n \to \{0, 1\}$ to denote the restriction of f on ρ. That is, $f|_\rho(y) := f(\rho \circ y)$.

3.2 Computational Models

$\mathsf{AC}^0_d[s(n)]$ denotes the family of Boolean circuits of size at most $s(n)$ and depth at most d. Such circuits can have AND, OR, NOT gates. Here, NOT gates do not count in the depth, and AND, OR gates can have unbounded fan-in. We measure the size of a circuit by the total number of wires (including input wires) in it. For the case of $d = 2$, we also use the terms DNF and CNF to refer to OR $\circ$ AND and AND $\circ$ OR circuits respectively. The width of a DNF or CNF is defined as the maximum of its bottom fan-in. We also use k-DNF (resp. k-CNF) to denote DNF (resp. CNF) of width at most k.

A decision tree T is a binary tree. Each inner node of T is labelled with an index $i \in [n]$, and has exactly two children, which are labelled with 0 and 1. Each leaf of T is labelled with a Boolean value $b \in \{0, 1\}$. A decision tree T computes a function in the following manner: on an input $x \in \{0, 1\}^n$, we start from the root of T. In each turn we observe the index i of current node, query x_i and move to the left/right child depending on the bit x_i we received. Once we reach a leaf with label b, we output $T(x) = b$. The depth of a decision tree is the length of the longest path from root to any leaf.

We also consider a special decision tree model for *a list of functions*. See the definition below.

Definition 1. Let $t, w, n, m \geq 1$ be integers. Let $\mathcal{F} = \{F_1, \ldots, F_m\}$ be a list of functions mapping from $\{0,1\}^n$ to $\{0,1\}$. A w-partial depth-t decision tree for $\mathcal{F}$ is a depth-t decision tree T satisfying the following. For every $F_i \in \mathcal{F}$ and every leaf ℓ of T, let $\alpha \in \{0, 1, \star\}^n$ be the partial assignment that corresponds to ℓ (That is, α_i equals $\star$ if T does not query x_i before reaching ℓ, otherwise α_i equals to the value that leads T to move towards ℓ). Then it holds that $\mathsf{DT}(F_i|_\alpha) \leq w$.

3.3 Pseudorandomness

Recall the definition of bounded independence.

Definition 2. Let n, m be two integers. Let $\mathcal{H}$ by a distribution over hash functions mapping $\{0,1\}^n$ into $\{0,1\}^m$. We say that $\mathcal{H}$ is k-wise independent, if for any k input-output pairs $(x_1, y_1), \ldots, (x_k, y_k) \in \{0,1\}^n \times \{0,1\}^m$ where $x_1, \ldots, x_t$ are distinct, it holds that

$$\Pr_{h \sim \mathcal{H}}[\forall i \in [k], h(x_i) = y_i] = 2^{-km}.$$

We have the following standard construction of bounded independence hash functions (check e.g., [Vad12, Chapter 3.5.5]).

Lemma 3. *For every $n, m, k \geq 1$, there is an explicit k-wise independent hash functions $\mathcal{H}$ that maps $\{0,1\}^n$ into $\{0,1\}^m$. One can sample a function in $\mathcal{H}$ using $O(k(n+m))$ random bits.*

We also consider a weaker notion of pseudorandomness called "k-wise p-boundedness", first defined and studied by [Kel21].

Definition 3. Suppose $\boldsymbol{\Lambda}$ is a random subset of $[n]$. We say that $\boldsymbol{\Lambda}$ is k-wise p-bounded if for any set $B \subseteq [n]$ of size at most k, it holds that $\Pr_{\boldsymbol{\Lambda}}[B \subseteq \boldsymbol{\Lambda}] \leq p^{|B|}$.

For intuition, if we sample $\boldsymbol{\Lambda}$ by independently including each i in $\boldsymbol{\Lambda}$ with probability at most p, then $\boldsymbol{\Lambda}$ is n-wise p-bounded.

4 Improved PRG for Constant-Depth Circuits

In this section, we aim to prove the main theorem, re-stated below.

Reminder of Theorem 1. *For every $d \geq 2$ the following is true. For every $m, n \in \mathbb{N}$ such that $m \geq n$ and every $\varepsilon > 0$, there is an ε-PRG for $\mathsf{AC}_d^0[m]$ circuits with seed length $O(\log^{d-1}(m) \cdot \log(m/\varepsilon) \cdot \log\log(m))$.*

We start with the following fact, which is crucial in our construction.

Theorem 2 ([DETT10, Tal17]). *For every $m, n \in \mathbb{N}$ such that $m \geq n$ and every $\varepsilon > 0$, there is an ε-PRG for $\mathsf{AC}_2^0[m]$ circuits (namely, CNF/DNF formulae) with seed length $O(\log(m) \cdot \log(m/\varepsilon) \cdot \log\log(m))$.*

We give the formal statement of the derandomized multi-switching lemma below. This is the full version of Lemma 2 in the introduction.

Lemma 4. *Let $t, w, k, n, m \geq 1$ be integers. Let $p, \varepsilon > 0$ be reals. Let $\mathcal{F} = \{F_1, \ldots, F_m\}$ be a list of size-m k-DNFs on inputs $\{0,1\}^n$. Let $(\boldsymbol{\Lambda}, \mathbf{x})$ be a joint random variable satisfying the following:*

- $\mathbf{\Lambda}$ *is a* $(t+k)$*-wise* p*-bounded subset of* $[n]$,

- *Conditioning on* $\mathbf{\Lambda}$, $\mathbf{x}$ *is an* n*-bit random string that* ε*-fools size-*(m^2) *CNF.*

Then we have

$$\Pr_{T \sim \mathbf{\Lambda}, x \sim \mathbf{x}}[\mathcal{F}|_{\Lambda[x,\star]} \text{ does not have } w\text{-partial depth-}t \text{ } DT] \leq 4m^{t/w}(24pk)^t + (24m)^{t+k} \cdot \varepsilon.$$

We defer the proof of Lemma 4 to Section 6. Assuming Lemma 4, we prove Theorem 1. We prove a slightly stronger form of Theorem 1, stated below.

Theorem 3. *For every* $d \geq 2$ *the following is true. For every* $m, n \in \mathbb{N}$ *such that* $m \geq n$, $k \in \mathbb{N}$ *and every* $\varepsilon > 0$, *there is an* ε*-error,* $O((\log^2(m) + k \cdot \log^{d-2}(m)) \cdot \log(m/\varepsilon) \cdot \log\log(m))$*-seed PRG for* $\mathsf{AC}^0_d[m]$ *circuits with bottom fan-in bounded by* k.

Theorem 3 shows that, when the bottom fan-in of the AC^0_d circuits is smaller than $o(\log(m))$, we can hope for shorter seed length through our construction. Note that we can always interpret a depth-d AC^0 circuit with unbounded bottom fan-in as a depth-$(d+1)$ AC^0 circuit with bottom fan-in being 1. Then, Theorem 1 follows from Theorem 3 easily.

Proof. We use induction on the depth d. The case for $d = 2$ follows from Theorem 2. Assuming this is true for depth $d - 1 \geq 2$, we prove it for the case of d. Let $w = 40k$ and $t = 80\log(m/\varepsilon)$. We prepare the following pseudorandom primitives.

- First, let $\mathbf{H} : [n] \to [w]$ be a $2t$-wise independent hash function, samplable using $O(\log(n)\log(m/\varepsilon))$ bits. In the following, we will use $\mathbf{H}_i$ to denote $\mathbf{H}^{-1}(i)$. We remark that $\mathbf{H}_i$ can be equivalently expressed as an n-bit string. Namely, $(\mathbf{H}_i)_j = 1$ if and only if $j \in \mathbf{H}_i$.

- Second, let $\varepsilon' = \varepsilon/(w \cdot 2^{t+1})$. Sample $\mathbf{X}_1, \ldots, \mathbf{X}_w \in \{0,1\}^n$, each being an independent string that ε'-fools AC^0_{d-1}-circuits of size $2m^2$ and bottom-width $\log m$. When $d - 1 \geq 3$, $\mathbf{X}_i$ is samplable using $O(\log^{d-2}(m) \cdot \log(m/\varepsilon) \cdot \log\log(m))$ bits by the induction hypothesis. For the case $d - 1 = 2$, $\mathbf{X}_i$ is samplable using $O(\log(m)\log(m/\varepsilon)\log\log(m))$ bits by Theorem 2.

- Lastly, let $\mathbf{Y} \in \{0,1\}^n$ be a random string that $\varepsilon/(24m)^{2t}$-fools size-m CNF, samplable using $O(\log^2 m \cdot \log(m/\varepsilon) \cdot \log\log m)$ bits (by Theorem 2).

The seed for our generator is the concatenation of the seeds used to sample all the primitives above. We compute the output of our generator as

$$\mathbf{Y} \oplus (\mathbf{X}_1 \wedge \mathbf{H}_1) \oplus (\mathbf{X}_2 \wedge \mathbf{H}_2) \oplus \cdots \oplus (\mathbf{X}_w \wedge \mathbf{H}_w). \tag{1}$$

Here, $\wedge$ and $\oplus$ denote bit-wise AND and XOR respectively. The seed length is bounded by

$$O\left(\log(n)\log(m/\varepsilon) + w \cdot \log^{d-2}(m)\log(m \cdot 2^t/\varepsilon)\log\log(m) + \log^2 m \log(m/\varepsilon)\log\log m\right)$$
$$\leq \quad O((\log^2(m) + k \cdot \log^{d-2}(m)) \cdot \log(m/\varepsilon) \cdot \log\log(m)).$$

We argue the correctness by a hybrid argument. Fix C to be an AC^0_d-circuit that we wish to fool. Let $\mathbf{U}_1, \ldots, \mathbf{U}_w$ denote w independent uniformly random strings from $\{0,1\}^n$. For every $i \in \{0, 1, \ldots, w\}$, we define the i-th hybrid distribution as

$$\mathcal{D}_i := \mathbf{Y} \oplus (\mathbf{U}_1 \wedge \mathbf{H}_1) \oplus \cdots \oplus (\mathbf{U}_i \wedge \mathbf{H}_i) \oplus (\mathbf{X}_{i+1} \wedge \mathbf{H}_{i+1}) \oplus \cdots \oplus (\mathbf{X}_w \wedge \mathbf{H}_w). \tag{2}$$

11

We observe that $\mathcal{D}_0$ is the output distribution of our PRG, while $\mathcal{D}_w$ is a uniformly random string from $\{0,1\}^n$. Hence, it suffices to show that

$$|\mathbb{E}_{x \sim \mathcal{D}_0}[C(x)] - \mathbb{E}_{x \sim \mathcal{D}_w}[C(x)]| \le \varepsilon. \tag{3}$$

To show (3), it suffices to show for every $i \in \{1, \ldots, w\}$ that

$$|\mathbb{E}_{x \sim \mathcal{D}_{i-1}}[C(x)] - \mathbb{E}_{x \sim \mathcal{D}_i}[C(x)]| \le \varepsilon/w. \tag{4}$$

In the following, we prove (4). We observe that $\mathbf{H}_i$ is $2t$-wise $\frac{1}{w}$-bounded. Conditioning on an instantiation of $\mathbf{H}$, we have that $\mathbf{Z}_i := \mathbf{Y} \oplus \sum_{j<i}(\mathbf{U}_j \wedge \mathbf{H}_j) \oplus \sum_{j>i}(\mathbf{X}_j \wedge \mathbf{H}_j)$ is an $\varepsilon/(24m)^{2t}$-pseudorandom string for size-(m^2) CNF, because $\mathbf{Y}$ is. Let $\mathcal{F}$ be the family of all next-to-bottom layer sub-circuits of C. Denote by $\mathcal{E}$ the event "$\mathcal{F}|_{\mathbf{H}_i[\mathbf{Z}_i, \star]}$ does not have $\log(m)$-partial depth-t DT". Then it follows from Lemma 4 that

$$\Pr_{\mathbf{H}, \mathbf{Y}, \mathbf{U}_1, \ldots, \mathbf{U}_{i-1}, \mathbf{X}_{i+1}, \ldots, \mathbf{X}_w}[\mathcal{E}] \le 4m^{t/\log(m)}\left(24\frac{k}{w}\right)^t + \frac{\varepsilon \cdot (24m)^{t+\log(m)}}{(24m)^{2t}} \le \frac{\varepsilon}{2w}.$$

Conditioning on $\neg\mathcal{E}, \mathbf{H}, \mathbf{Y}, \mathbf{U}_1, \ldots, \mathbf{U}_{i-1}, \mathbf{X}_{i+1}, \ldots, \mathbf{X}_w$ and calculating $\mathbf{Z}_i$ as defined above, one can then write $C|_{\mathbf{H}_i[\mathbf{Z}_i, \star]}$ as a depth-t decision tree T where each leaf of T is labelled by an AC^0_{d-1}-circuit of size m^2 and bottom fan-in $\log(m)$. Let $\{\ell_1, \ldots, \ell_{2^t}\}$ enumerate the leaves of the decision tree. Each ℓ_j is associated with a size-m^2 depth-$(d-1)$ circuit, which is also denoted by $\ell_j : \{0,1\}^n \to \{0,1\}$ for brevity. Then one can write $C|_{(H_i)[Z_i, \star]}$ as

$$C|_{\mathbf{H}_i[Z_i, \star]}(y) = \sum_{j=1}^{2^t} \ell_j(y) \cdot \mathbb{1}\{T(y) \text{ reaches leaf } \ell_j\}.$$

Let's fix an index $j \in [2^t]$ for now. Note that $\ell_j(y) \cdot \mathbb{1}\{\text{reach } \ell_j \text{ on } y\}$ is itself a depth-$(d-1)$ circuit of size at most $2m^2$. By the construction of X_i we know that

$$|\mathbb{E}_{\mathbf{X}_i}[\ell_j(\mathbf{X}_i + Z_i) \cdot \mathbb{1}\{\text{reach } \ell_j \text{ on } \mathbf{X}_i + Z_i\}] - \mathbb{E}_{\mathbf{U}_i}[\ell_j(\mathbf{U}_i + Z_i) \cdot \mathbb{1}\{\text{reach } \ell_j \text{ on } \mathbf{U}_i + Z_i\}]| \le \frac{\varepsilon}{2^{t+1}w}.$$

Taking a summation over all leaves j, one gets

$$\left|\mathbb{E}_{\mathbf{X}_i}[C|_{\mathbf{H}_i[Z_i, \star]}(\mathbf{X}_i + Z_i)] - \mathbb{E}_{\mathbf{U}_i}[C|_{\mathbf{H}_i[Z_i, \star]}(\mathbf{U}_i + Z_i)]]\right| \le \frac{\varepsilon}{2w}.$$

Finally, one has

$$|\mathbb{E}_{x \sim \mathcal{D}_{i-1}}[C(x)] - \mathbb{E}_{x \sim \mathcal{D}_i}[C(x)]| \le \Pr[\neg\mathcal{E}] \cdot \frac{\varepsilon}{2w} + \Pr[\mathcal{E}] \le \frac{\varepsilon}{w},$$

proving (4).

$\square$

Wrapping-up. Given Theorem 3, we prove Theorem 1 by tuning parameters.

Proof of Theorem 1. For every $d \ge 3$ and every $\mathsf{AC}^0_d[m]$ circuit with unbounded bottom fan-in, we can interpret it as a depth-$(d+1)$ circuit with bottom fan-in being 1. Applying Theorem 3 in this case gives a PRG with seed length $O(\log^{d-1}(m)\log(m/\varepsilon)\log\log(m))$, as desired. For the case of $d = 2$, we use Theorem 2 directly. This completes the proof. $\square$

As a final remark, suppose we could have $O(\log(m)\log(m/\varepsilon))$-seed PRG for CNFs (namely, if we can shave the $\log\log m$ factor in Theorem 2). Then our construction implies PRG for AC^0_3 with seed length $O(\log^2(m)\log(m/\varepsilon))$, and further implies PRG for AC^0_d with seed length $O(\log^{d-1}(m)\log(m/\varepsilon))$, matching the lower bound barrier [Has89, Hås14].

5   Fully-Derandomized Switching Lemma

Before we show the proof of Lemma 4, we state and prove the simpler version of the classical switching lemma in this section. The following statement is the full version of Lemma 1 in Introduction. Lemma 1 follows from Lemma 5 trivially.

Lemma 5. *Let $k, m \geq 1$ be integers, and $\varepsilon, p > 0$ be reals. Let $F = \bigvee_{i=1}^{m} C_i$ be a size-m k-DNF over inputs $\{0,1\}^n$. For all $t \geq 1$, let $(\Lambda, \mathbf{x})$ be a joint random variable such that:*

- Λ *is a $(t+k)$-wise p-bounded subset of $[n]$.*

- *Conditioning on Λ, $\mathbf{x}$ is a random string that ε-fools CNF of size at most m.*

Consider the random restriction $F|_{\Lambda[\mathbf{x},\star]}$, we have:

$$\Pr_{\Lambda,\mathbf{x}}[\mathsf{DT}(F|_{\Lambda[\mathbf{x},\star]}) > t] \leq (10kp)^t + (4m)^{t+k} \cdot \varepsilon.$$

Understanding the proof of Lemma 5 is necessary to read the proof of Lemma 4. On the other hand, once Lemma 5 is established, we can prove Lemma 4 using a rather similar strategy. The rest of the section is devoted to the proof of Lemma 5.

We will first introduce the important concept of "canonical decision tree" ([Has89, TX13, ST19, Tal17]) in Section 5.1. We prove for the case that $\mathbf{x}$ is truly random (and only Λ is pseudorandom) in Section 5.2, and then argue how to prove for pseudorandom $\mathbf{x}$ in Section 5.3.

Throughout the whole section, we always use $F = \bigvee_{i=1}^{m} C_i$ to denote the k-DNF that we are analyzing. For every $i \in [m]$ let $V_i \subseteq [n]$ be the variables involved in the term C_i.

5.1   Canonical decision tree

For a DNF $F = \bigvee_{i=1}^{m} C_i$, denote by T_F the *canonical decision tree* of F, whose construction is shown in Algorithm 1. We use $\mathsf{CDT}(F)$ to denote the depth of the canonical decision tree for F. Since canonical decision tree is one particular decision tree of F, its depth must be no less than $\mathsf{DT}(F)$. To prove Lemma 5, we will analyze canonical decision trees, and show that with high probability over the random restriction ρ, we have $\mathsf{CDT}(F|_{\rho}) < t$.

Algorithm 1: Canonical Decision Tree

Input: A DNF $F = \bigvee_{i=1}^{m} C_i$, black-box access to a string $\alpha \in \{0,1\}^n$.

initialize:
> $j^* \leftarrow 0$.
> $x \leftarrow (\star)^n$.

while $j^* < m$ **do**
> Find the first $j > j^*$ such that $C_j(x) \not\equiv 0$. If no such j exists, exit the loop.
> $B_j \leftarrow$ the set of unknown variables in C_j.
> Query α_{B_j}.
> Set $x_{B_j} \leftarrow \alpha_{B_j}$.
> **if** $C_j(x) = 1$ **then**
>> **return** 1.
>
> $j^* \leftarrow j$.

return 0

5.2 Proof when x is truly random

Defining witness. Let $\rho = \Lambda[x, \star]$ being a bad restriction, under which $\mathsf{CDT}(F|_\rho) \geq t$. Consider simulating the canonical decision tree $T_{F|_\rho}$. We know that on some inputs $\alpha \in \{0,1\}^n$, $T_{F|_\rho}$ fails to output the decision after making $(t-1)$ queries. We choose one such α and simulate $T_{F|_\rho}$ until it makes at least t queries. The "running transcript" of $T_{F|_\rho}$ on α is naturally a witness to the fact that $\mathsf{CDT}(F_\rho) \geq t$. We formalize this idea in the following definition.

Definition 4. Let $F = \bigvee_{i=1}^m C_i$ be the k-DNF and $\rho \in \{0,1,\star\}^n$ be a restriction. Let $t \geq 1$. Consider a tuple $(r, \ell_i, s_i, B_i, \alpha_i)$, where:

1. $r \in [1, t]$ is an integer.

2. $(\ell_1, \ldots, \ell_r) \in [m]^r$ is a list of increasing indices.

3. $(s_1, \ldots, s_r)$ is a list of positive integers such that $s := \sum_{i=1}^r s_i \in [t, t+k-1]$.

4. $(B_1, \ldots, B_r)$ is a list of subsets of $[k]$. Moreover, for every $i \in [r]$, $|B_i| = s_i$.

5. $(\alpha_1, \ldots, \alpha_r)$ is a list of binary strings. For every $i \in [r]$, $|\alpha_i| = s_i$.

We call $(r, \ell_i, s_i, B_i, \alpha_i)$ a t-witness for ρ, if there exists $\alpha \in \{0,1\}^n$ such that:

- When we run $T_{F|_\rho}$ on α, for every $i \in [r]$, C_{ℓ_i} is the i-th term queried by $T_{F|_\rho}$.

- By the time $T_{F|_\rho}$ issues the i-th set of query, exactly s_i variables in C_{ℓ_i} are not known, and their "relative positions" in V_{ℓ_i} are specified by B_i.

- The response to the i-th set of query is α_i.

We define the *size* of the witness $(r, \ell_i, s_i, B_i, \alpha_i)$ as $s := \sum_{i=1}^r s_i$.

Notation convention. When B_i and α_i are associated with a term C_{ℓ_i}, sometimes we will slightly abuse notation by using B_i to refer to the set of variables it corresponds to in V_{ℓ_i} (Recall V_{ℓ_i} is the set of variables that appear in C_{ℓ_i}), and use α_i to denote a partial assignment $\alpha_i \in \{0,1,\star\}^n$ in which we only assign the B_i part of α_i and leave other coordinates unfixed. In this section, we will mostly use the term "witness" to denote t-witness, since we are always analyzing for a fixed t.

It is easy to see that if $\mathsf{DT}(F|_\rho) \geq t$, there must be a witness for for $\mathsf{CDT}(F|_\rho) \geq t$[6]. By now, a natural idea to bound the probability of picking a bad restriction would be (1) enumerating every possible witness, (2) calculating the probability of a random ρ having such a witness, and (3) union-bounding over them. Unfortunately this is too expensive for us: we have at least $\binom{m}{t}$ choices of the list (ℓ_i). In order for this approach to be meaningful, we have to bound the probability that a random restriction has a particular witness by m^{-t}, which seems very hard, if not impossible.

It turns out we can avoid the enumeration of (ℓ_i) part in the witness. To succinctly describe the idea, let us define partial witnesses first.

Definition 5. Let $F = \bigvee_{i=1}^m C_i$ be the k-DNF and $\rho \in \{0,1,\star\}^n$ be a restriction. Let $t \geq 1$. Consider a tuple (r, s_i, B_i, α_i). We call (r, s_i, B_i, α_i) a partial t-witness for ρ, if there exists $(\ell_1, \ldots, \ell_r)$ such that $(r, \ell_i, s_i, B_i, \alpha_i)$ is a t-witness for ρ.

---

[6]However, we note that the converse may not be true, since our witness can only refute the existence of shallow *canonical* decision trees.

Remark 1. *Here we make an important observation: if (r, s_i, B_i, α_i) is a partial witness for ρ, then there is only one valid list (ℓ_i) which makes $(r, \ell_i, s_i, B_i, \alpha_i)$ a witness for ρ (To see this, first note that ℓ_1 is fixed. After querying ℓ_1 and getting the response α_1, use induction). This observation will prove useful in Section 5.3 and Section 6.*

Then, the proof goes by enumerating r, s_i, B_i, α_i and bounding the following:

$$\Pr_\rho[\mathsf{DT}(F|_\rho) \geq t] \leq \sum_{(r, s_i, B_i, \alpha_i)} \Pr_\rho[(r, s_i, B_i, \alpha_i) \text{ is a partial witness for } \rho]. \tag{5}$$

Fixing an $s \in [t, t+k-1]$, there are at most $(4k)^s$ possible partial witnesses (r, s_i, B_i, α_i) of size s. For each of them, if we can bound the probability by, say, $(2p)^s$, then the lemma is proved.

The witness searcher. Now, given a restriction ρ and a candidate partial witness (r, s_i, B_i, α_i), how can we decide if there is a list $(\ell_i)_i$ such that $(r, \ell_i, s_i, B_i, \alpha_i)$ constitutes a witness for ρ? We will do so by designing and using a simple procedure, which we call the witness searcher. More specifically, our searcher receives as input a restriction ρ, a partial witness (r, s_i, B_i, α_i) and an advice string $y \in \{0,1\}^n$. It outputs either a complete witness $(r, \ell_i, s_i, B_i, \alpha_i)$, or a special ERROR symbol. Its description in shown in Algorithm 2.

Algorithm 2: Witness Searcher

Input: A DNF $F = \bigvee_{i=1}^m C_i$, a restriction $\rho \in \{0, 1, \star\}^n$, a partial witness (r, s_i, B_i, α_i), an advice $y \in \{0,1\}^n$.

initialize:
> $z \leftarrow \rho \circ y$.
> $j^* \leftarrow 0$.
> $c \leftarrow 1$.

while $c \leq r$ **do**
> Find the first $j > j^*$ such that C_j is satisfied by z. If no such i exists, return ERROR.
> Set $\ell_c \leftarrow j$, and associate B_c, α_c with C_{ℓ_c}.
> Replace the B_c part of z with α_c. That is, $z \leftarrow \rho \circ \alpha_1 \circ \cdots \circ \alpha_c \circ y$.
> $c \leftarrow c + 1$.
> $j^* \leftarrow j$.

return $(r, \ell_i, s_i, B_i, \alpha_i)$

In the following, we use $\mathcal{S}$ to denote Algorithm 2.

Lemma 6. *If (r, s_i, b_i, α_i) is a partial witness for ρ, then there exists an advice y that makes $\mathcal{S}$ find $(\ell_i)_{i=1}^r$. More importantly, on a uniformly random $\mathbf{y} \sim \mathcal{U}_n$, $\mathcal{S}$ finds $(\ell_1, \ldots, \ell_r)$ with probability exactly 2^{-s}.*

Proof. If $(r, s_i, \ell_i, b_i, \alpha_i)$ is a witness for ρ, it records a running transcript of $T_{F|_\rho}$ on some input $\alpha \in \{0,1\}^n$. This mean that C_{ℓ_1} is the first term queried by $T_{F|_\rho}$, which implies:

1. Every term before C_{ℓ_1} is falsified by ρ (because $T_{F|_\rho}$ skipped them).

2. C_{ℓ_1} is consistent with ρ (because $T_{F|_\rho}$ was uncertain of the value of $C_{\ell_i}(\alpha)$.

Now, consider running $\mathcal{S}$ on (r, s_i, b_i, α_i), $\mathcal{S}$ will also skip all the terms before C_{ℓ_1}. At C_{ℓ_1}, since ρ is consistent with C_{ℓ_1}, the random string $\rho \circ \mathbf{y}$ satisfies C_{ℓ_1} with probability $2^{-|B_1|}$. Conditioning on this happened, in the later execution, we modify z and replace the part corresponding to B_1 with α_1.

At this point, we go back and inspect the execution of $T_{F|_\rho}$ on α. Since $(r, s_i, \ell_i, b_i, \alpha_i)$ is the running transcript for $T_{F|_\rho}$ on α, we know that α_1 was the response that $T_{F|_\rho}$ received from querying alive variables in C_{ℓ_1}. Since $T_{F|_\rho}$ issued its second bunch of queries to alive variables in C_{ℓ_2}, it implies that terms from C_{ℓ_1+1} to C_{ℓ_2-1} are all falsified by $\rho \circ \alpha_1$, and C_{ℓ_2} is consistent with $\rho \circ \alpha_1$. Then, it follows that $\rho \circ \alpha_1 \circ \mathbf{y}$ satisfies C_{ℓ_2} with probability $2^{-|B_2|}$. Conditioning on this happened, the searcher will proceed with string $\rho \circ \alpha_1 \circ \alpha_2 \circ y$ and we can again consider the execution of $T_{F|_\rho}$ after querying C_{ℓ_2}. We do this argument so on and so forth, until we have identified all of r indices $\ell_1, \ldots, \ell_r$ and exit the procedure. In summary, we have shown there exists y which makes $\mathcal{S}$ find the list $(\ell_i)_{i=1}^r$, and the probability of sampling such a y is

$$2^{-\sum_{i=1}^c |B_i|} = 2^{-s}$$

as desired. $\qquad\square$

Decoupling. If we inspect the execution of $\mathcal{S}$ carefully, we can notice that it only needs to know the string $z = \rho \circ y$ to work. In particular, it *does not* need to know which part of z is fixed in the restriction ρ. Therefore, we can revise $\mathcal{S}$ to get a searcher $\mathcal{S}'$: the input to $\mathcal{S}'$ is now a string z and (r, s_i, B_i, α_i). Otherwise it runs identically the same way as $\mathcal{S}$. We denote the output of $\mathcal{S}'$ as $\mathcal{S}'(z, (r, s_i, B_i, \alpha_i))$. By Lemma 6, we have

$$\Pr_{\rho \sim \mathcal{R}}[(r, s_i, B_i, \alpha_i) \text{ is a partial witness for } \rho]$$
$$\leq 2^s \cdot \Pr_{\rho \sim \mathcal{R}, y \sim \mathcal{U}_n}[\mathcal{S}'(\rho \circ y, (r, s_i, B_i, \alpha_i)) \text{ is a witness for } \rho].$$

Denote $\rho = \Lambda[x, \star]$. We observe that

$$\mathbb{1}\{ (r, \ell_i, s_i, B_i, \alpha_i) \text{ is a witness for } \rho\} \leq \mathbb{1}\left\{ \left(\bigcup_{j=1}^r B_j\right) \subseteq \Lambda \right\}. \tag{6}$$

For a $(t+k)$-wise p-bounded set $\mathbf{\Lambda}$, the event on the right hand side holds with probability $p^{\sum_j |B_j|}$. Then, we have

$$\Pr_{\rho \sim \mathbf{\Lambda}[\mathbf{x}, 0], y \sim \mathcal{U}_n}[\mathcal{S}'(\rho \circ y, (r, s_i, B_i, \alpha_i)) \text{ is a witness for } \rho]$$
$$= \mathbb{E}_{\mathbf{\Lambda}} \mathbb{E}_{x, y \sim \mathcal{U}_n} \left[\mathbb{1}\{\mathcal{S}'(\mathbf{\Lambda}[x, y], (r, s_i, B_i, \alpha_i)) \text{ is a witness for } \rho\}\right]$$
$$= \mathbb{E}_{z \sim \mathcal{U}_n} \left[\Pr_{\mathbf{\Lambda}}[\mathcal{S}'(z, (r, s_i, B_i, \alpha_i)) \text{ is a witness for } \rho]\right]$$
$$\leq p^s,$$

where the third equality is due to that $\Lambda[x, y]$ is distributed as $\mathcal{U}_n$ when $x, y \sim \mathcal{U}_n$, and the last inequality holds by (6) and the $(t+k)$-wise p-bounded property of $\mathbf{\Lambda}$.

Wrapping-up. We finish the proof by enumerating all (r, s_i, B_i, α_i) and taking a summation.

$$\Pr_{\rho \sim \mathbf{\Lambda}[\mathbf{x},\star]}[\mathsf{DT}(F|_\rho) \geq t] \leq \sum_{(r,s_i,B_i,\alpha_i)} \Pr_{\rho \sim \mathbf{\Lambda}[\mathbf{x},\star]}[(r,s_i,B_i,\alpha_i) \text{ is a partial witness for } \rho]$$

$$\leq \sum_{(r,s_i,B_i,\alpha_i)} 2^s \cdot \mathbb{E}_{z \in \mathcal{U}_n}\left[\Pr_{\mathbf{\Lambda}}[\mathcal{S}'(z,(r,s_i,B_i,\alpha_i)) \text{ is a witness for } \mathbf{\Lambda}[\mathbf{x},\star]]\right]$$

$$\leq \sum_{s=t}^{t+k} 2^s (4k)^s p^s$$

$$\leq (10kp)^t. \tag{7}$$

5.3 Proof when x is pseudorandom

Now we consider the case that $\mathbf{x}$ is not truly random. There could even be correlation between $\mathbf{x}$ and $\mathbf{\Lambda}$. Our only requirement for $\mathbf{x}$ is that, for every fixed $\mathbf{\Lambda}$, $\mathbf{x}$ is a pseudorandom string that ε-fools CNFs of size at most m. First of all, by Remark 1, the following equation is established.

$$\mathbb{1}\{(r,s_i,B_i,\alpha_i) \text{ is a partial witness for } \rho\} = \sum_{\ell_i} \mathbb{1}\{(r,\ell_i,s_i,B_i,\alpha_i) \text{ is a witness for } \rho\}. \tag{8}$$

If we inspect the deduction in (7), it actually shows the following:

$$\sum_{(r,s_i,B_i,\alpha_i)} \sum_{\ell_i} \Pr_{\rho \sim \mathbf{\Lambda}[\mathcal{U}_n,\star]}[(r,\ell_i,s_i,B_i,\alpha_i) \text{ is a witness for } \rho] \leq (10kp)^t. \tag{9}$$

Next, fixing the set Λ and a tuple of $(r,\ell_i,s_i,B_i,\alpha_i)$, we consider the predicate

$$h_\Lambda^{(r,\ell_i,s_i,B_i,\alpha_i)}(x) := \mathbb{1}\{(r,\ell_i,s_i,B_i,\alpha_i) \text{ is a witness for } \Lambda[x,\star]\}.$$

We omit the superscript and subscript of h when they are clear from context. We claim that h can be implemented by a CNF of size at most m. To see this, note that $h(x) = 1$ if all of the following hold.

- For every $j < \ell_1$, C_j is falsified by $\Lambda[x,\star]$. For this to be true, there is at least one variable in C_j that is not chosen by Λ and is assigned to the opposite value to its appearance in C_j. Note that we can use an OR over relevant variables or their negations to verify if C_j is falsified.

- For $j = \ell_1$, C_{ℓ_1} cannot be falsified by $\Lambda[x,\star]$, which means all variables in C_{ℓ_1} that are not chosen by Λ are assigned so that it does not contradict C_{ℓ_1}. We can use an AND over those variables to verify this.

- For every $\ell_1 < j < \ell_2$, C_j is falsified by $\Lambda[x,\alpha_1]$, which means that either C_j is falsified by α_1, or there is at least one variable in C_j that is not chosen by Λ and is assigned to the opposite value. If C_j is falsified by α_1, there is nothing to verify. Otherwise, we can verify it by an OR over relevant variables.

- The same reasoning applies to C_j for every $j \geq \ell_2$. Finally, after we finish the verification for C_{ℓ_r}, we are done.

In summary, we can implement h by a CNF (AND of ORs). It is obvious that the size of the CNF is no larger than the size of F, which is m. Therefore, we conclude that $\mathbf{x}$ ε-fools h. That is,

$$\left| \Pr_{x \sim \mathcal{U}_n}[h_\Lambda^{(r,\ell_i,s_i,B_i,\alpha_i)}(x)] - \Pr_{x \sim \mathbf{x}}[h_\Lambda^{(r,\ell_i,s_i,B_i,\alpha_i)}(x)] \right| \leq \varepsilon. \tag{10}$$

Finally, we have

$$\begin{aligned}
\Pr_{\rho \sim \Lambda[\mathbf{x},\star]}[\mathsf{DT}(F|_\rho) \geq t] &\leq \sum_{(r,s_i,B_i,\alpha_i)} \Pr_{\rho \sim \Lambda[\mathbf{x},\star]}[(r,s_i,B_i,\alpha_i) \text{ is a partial witness for } \rho] \\
&\leq \sum_{(r,s_i,B_i,\alpha_i)} \sum_{\ell_i} \Pr_{\rho \sim \Lambda[\mathbf{x},\star]}[(r,\ell_i,s_i,B_i,\alpha_i) \text{ is a witness for } \rho] \\
&\leq \sum_{(r,s_i,B_i,\alpha_i)} \sum_{\ell_i} \varepsilon + \Pr_{\rho \sim \Lambda[\mathcal{U}_n,\star]}[(r,\ell_i,s_i,B_i,\alpha_i) \text{ is a witness for } \rho] \\
&\leq (10kp)^t + (4m)^{t+k} \cdot \varepsilon.
\end{aligned}$$

Here, the first line is due to the argument in Section 5.2. The second line holds by (8). The third line is due to (10). The last line holds because (9) and there are at most $(4m)^{k+t}$ possible tuples $(r,\ell_i,s_i,B_i,\alpha_i)$.

6 Derandomizing the Multi-Switching Lemma

In this section, we prove Lemma 4. Recall the statement.

Reminder of Lemma 4. *Let $t,w,k,n,m \geq 1$ be integers. Let $p,\varepsilon > 0$ be reals. Let $\mathcal{F} = \{F_1,\ldots,F_m\}$ be a list of size-m k-DNFs on inputs $\{0,1\}^n$. Let $(\Lambda,\mathbf{x})$ be a joint random variable satisfying the following:*

- *Λ is a $(t+k)$-wise p-bounded subset of $[n]$,*
- *Conditioning on Λ, $\mathbf{x}$ is a random string that ε-fools size-(m^2) CNF.*

Then we have

$$\Pr_{T \sim \Lambda, x \sim \mathbf{x}}[\mathcal{F}|_{\Lambda[x,\star]} \text{ does not have } w\text{-partial depth-}t \text{ DT}] \leq 4m^{t/w}(24pk)^t + (24m)^{t+k} \cdot \varepsilon.$$

The rest of the section is devoted to the proof of Lemma 4. Section 6.1 states some preliminary tools and includes a proof overview. Section 6.2 considers the case that $\mathbf{x}$ is truly random. It extends the idea in Section 5.2 and has a rather similar structure. Finally in Section 6.3, we consider the case that $\mathbf{x}$ is pseudorandom, and finish the proof.

6.1 Preliminaries

The canonical partial decision tree. Let ρ be a random restriction under which $\mathcal{F}|_\rho$ fails to have a w-partial depth-t decision tree. Let $\beta \in \{0,1\}^n$ be an unknown string that our decision tree shall query. We consider the following attempt (see Algorithm 3) to construct a partial decision tree for $\mathcal{F}|_\rho$. Here $z \in \{0,1\}^n$ is an auxiliary string to be chosen. Every choice of $z \in \{0,1\}^n$ yields

a different partial decision tree for $\mathcal{F}$. In [ST19], the construction was called "canonical partial decision tree".

Algorithm 3: Canonical Partial Decision Tree

Input: A list of DNFs $\mathcal{F} = \{F_1, \ldots, F_m\}$, black-box access to a string $\beta \in \{0,1\}^n$, and an auxiliary string $z \in \{0,1\}^n$.

initialize:
$\quad x \leftarrow (\star)^n$.
$\quad j \leftarrow 1$.
$\quad$ counter $\leftarrow 0$.

while counter $< t$ **do**
$\quad$ Find the smallest $i \geq j$ such that $\mathsf{DT}(F_i|_x) > w$. If no such i exists, exit the loop.
$\quad y \leftarrow (\star)^n$.
$\quad I \leftarrow \emptyset$.
$\quad$ **while** $F_i|_{x \circ y}(\star)$ *is not constant and* counter $< t$ **do**
$\quad\quad C_{i,q} \leftarrow$ the term that $T_{F_i|_{x \circ y}}$ will query.
$\quad\quad B_{i,q} \leftarrow$ the set of unknown variables in $C_{i,q}|_{x \circ y}$.
$\quad\quad y_{B_{i,q}} \leftarrow z_{B_{i,q}}$.
$\quad\quad I \leftarrow I \cup B_{i,q}$.
$\quad\quad$ counter $\leftarrow$ counter $+ |B_{i,q}|$.
$\quad$ Query β_I, and set $x_I \leftarrow \beta_I$.
$\quad j \leftarrow i$.

return x

Note that in Algorithm 3, it is possible that for a formula F_i to get picked by the outer "while" loop more than once.

Proof overview. Before we dive into the formal proof, we try to give some (over-simplified) intuition here. If $\mathcal{F}$ does not have w-partial DT of depth t, then Algorithm 3 fails to construct a partial decision tree of depth t on every auxiliary string z. However, we will only consider some "adversarially chosen" z's. That is, we only consider those z's that trick Algorithm 3 to make at least w queries on each chosen formula. By doing so, we are guaranteed that Algorithm 3 only chooses $\frac{t}{w}$ formulae in total.

Then, we can pay a factor of $m^{\frac{t}{w}}$ to enumerate a subset of $\frac{t}{w}$ formulae in $\mathcal{F}$, and calculate the probability that Algorithm 3 "gets stuck" on those formulae. Having fixed those $\frac{t}{w}$ formulae, we can use ideas similar to Section 5.2, and bound the probability by $O(pk)^t$. In summary, we get

$$\Pr[\text{bad restriction}] \leq m^{\frac{t}{w}} \cdot O(pk)^t.$$

Here we only considered the case that the restriction string $\mathbf{x}$ is purely random. For the case that both $\mathbf{\Lambda}$ and $\mathbf{x}$ are pseudorandom, we use tricks similar to Section 5.3 and pay another additive factor.

6.2 Proof when x is truly random

We start the proof by considering the case that $\mathbf{x}$ is a truly random string. Fix ρ to be a bad restriction, under which $\mathcal{F}|_\rho$ fails to have a w-partial depth-t decision tree. It implies that on every $z \in \{0,1\}^n$, Algorithm 3 fails to construct a partial decision tree for $\mathcal{F}$. However, for ease of our analysis, we will only consider a special class of string z. We give the following definition.

Definition 6. Let ρ be a bad restriction for $\mathcal{F}$. We call $z \in \{0,1\}^n$ a *powerful* (w,t)-*refutation* for ρ, if there exists $\beta \in \{0,1\}^n$ satisfying the following: Algorithm 3 on input $(\mathcal{F}|_\rho, \beta, z)$ makes at least t queries. Moreover, at each time we query β_I, it is guaranteed that $|I| \geq w$.

Powerful refutation always exists for bad restrictions, as shown in the following.

Lemma 7. *If $\mathcal{F}|_\rho$ does not have w-partial decision tree of depth t, then there exists a powerful* (w,t)-*refutation for ρ.*

Proof. If $\mathcal{F}_\rho$ does not have w-partial decision tree of depth t, then it means Algorithm 3 fails to construct a w-partial DT of depth t for every $z \in \{0,1\}^n$. Then we construct a powerful refutation z, together with its associated adversarial input β, in the following way.

- Initially, we set $z = \beta = (\star)^n$. Then we monitor the execution of Algorithm 3 on $(\mathcal{F}, \beta, z)$, and gradually fill in z, β when they are accessed by Algorithm 3.

- Whenever Algorithm 3 chooses one formula $F_i|_{\rho \circ x}$ in the outer while-loop, we do the following:

  - When running inside the inner while-loop, Algorithm 3 needs to consult the auxiliary string z. We can fill in relevant variables of z in such a way that Algorithm 3 consults at least w bits from z in the loop. Since $\mathsf{DT}(F_i|_{\rho \circ x}) \geq w$, this is always possible.

  - After finishing the inner loop, Algorithm 3 will make a query to β_I. At this point, we set β_I in such a way that $\mathcal{F}|_{\rho \circ x \circ \beta}$ does not have w-partial DT of depth $(t - \text{counter})$. This is always possible as long as $\mathcal{F}_{\rho \circ x}$ does not have w-partial DT of depth $(t - \text{counter} + |I|)$, which holds by induction.

- After Algorithm 3 returns, we fill in the remaining bits of β, z with zeros.

$\square$

Having established Lemma 7, we come up with the following definition of "global witness" naturally.

Definition 7. Let t, w be two integers. Consider a list of k-DNFs $\mathcal{F} = \{F_1, \ldots, F_m\}$. Suppose $\rho \in \{0, 1, \star\}^n$ is a restriction. Let $(R, L_i, S_i, W_i, \beta_i)$ be a tuple, where

1. $1 \leq R \leq \frac{t}{w}$ is an integer;

2. $1 \leq L_1 \leq L_2 \leq \cdots \leq L_R \leq m$ is a list of R non-decreasing indices;

3. $S_1 \ldots, S_R$ is a list of R integers such that $\sum_{i=1}^R S_i \in [t, t+k]$;

4. $W_1, \ldots, W_R$ is a list of witnesses (as per Definition 4). For every $i \in [R]$, W_i has size S_i;

5. $\beta_1, \ldots, \beta_R$ are R strings where $|\beta_i| = S_i$ for every $i \in [R]$.

We call the tuple a (w,t)-global witness for ρ, if it satisfies the following.

1. Set $\rho_1 = \rho$. W_1 is a S_1-witness for $F_{L_1}|_{\rho_1}$.

2. For every $i \geq 2$, let $I_{i-1} \subseteq [n]$ be the set of variables involved in W_{i-1}. Note that $|I_{i-1}| = S_{i-1}$ since the size of W_{i-1} is S_{i-1}. Identify β_{i-1} as a partial assignment in $\{0, 1, \star\}^n$ where only the part $\beta_{i-1, I_{i-1}}$ is set and other coordinates are filled in with $\star$. Construct $\rho_i = \rho_{i-1} \circ \beta_{i-1}$. Then W_i is a S_i-witness for $F_{L_i}|_{\rho_i}$.

The size of the global witness is defined as $\sum_{i=1}^{R} S_i$.

As a corollary of Lemma 7, we have:

Corollary 1. *Consider a list of k-DNFs $\mathcal{F} = \{F_1, \ldots, F_m\}$. Suppose $\rho \in \{0, 1, \star\}^n$ is a restriction such that $\mathcal{F}|_\rho$ does not have w-partial depth-t decision tree. Then there exists a (w, t)-global witness for ρ.*

Proof. By Lemma 7, there exists a powerful (w, t)-refutation for $\mathcal{F}|_\rho$. Take one such refutation z with its adversarial input β. Inspect the execution of Algorithm 3 on $(\mathcal{F}|_\rho, \beta, z)$ and record the transcript. The transcript contains the desired tuple. $\square$

Similar to what we have done in Section 5.2, we define the global partial witness.

Definition 8. Let t, w be two integers. Consider a list of k-DNFs $\mathcal{F} = \{F_1, \ldots, F_m\}$. Suppose $\rho \in \{0, 1, \star\}^n$ is a restriction. Let $(R, L_i, S_i, P_i, \beta_i)$ be a tuple, where

1. $1 \leq R \leq \frac{t}{w}$ is an integer;

2. $1 \leq L_1 \leq L_2 \leq \cdots \leq L_R \leq m$ is a list of R non-decreasing indices;

3. $S_1, \ldots, S_R$ is a list of R integers such that $\sum_{i=1}^{R} S_i \in [t, t + k]$;

4. $P_1, \ldots, P_R$ is a list of partial witnesses. For every $i \in [R]$, P_i has size S_i.

5. $\beta_1, \ldots, \beta_R$ are R strings where $|\beta_i| = S_i$ for every $i \in [R]$.

We call $(R, L_i, S_i, P_i, \beta_i)$ a (w, t)-global partial witness for ρ, if we can complete P_i to get a witness W_i for every $i \in [R]$, such that $(R, L_i, S_i, W_i, \beta_i)$ is a global witness for ρ.

Remark 2. *Again, it is easy to see by induction that, for any global partial witness, there is exactly one way to complete it and get a global witness.*

By a simple counting, it turns out for every $S \in [t, t+k]$, there are at most $2m^{t/w}(12k)^S$ possible global partial witnesses of size S:

1. First, $2m^{t/w}$ ways to choose at most $\frac{t}{w}$ formulae in $\mathcal{F}$.

2. Then, 3^S ways to partition the s units of "query budget" into R partial witnesses as $S = \sum_{i=1}^{R} S_i$, and further partition the budget for each partial witness as $S_i = \sum_{j=1}^{r_i} s_{i,j}$.

3. Then, at most k^S ways to construct sets $B_{i,j}$ for each partial witness.

4. Finally, there are 4^S ways to choose β_i's, as well as $\alpha_{i,j}$'s in each partial witness.

Fixing a global partial witness $(R, L_i, S_i, P_i, \beta_i)$, we try to bound the probability that a random ρ has this witness by $(2p)^k$. Using the witness searcher (Algorithm 2) as a subroutine, we design

21

a global witness searcher first. Again, the global witness searcher needs access to an advice string $y \in \{0,1\}^n$. See Algorithm 4.

Algorithm 4: Global Witness Searcher

Input: A list of DNFs $\mathcal{F} = \{F_1, \ldots, F_m\}$, a restriction $\rho \in \{0, 1, \star\}^n$, a global partial witness $(R, L_i, S_i, P_i, \beta_i)$, and an advice $y \in \{0,1\}^n$.

initialize:
> $z \leftarrow \rho \circ y$.
> $c \leftarrow 1$.
> $\rho^{(1)} \leftarrow \rho$.

while $c \leq R$ **do**
> Run Algorithm 2 on $(F_{L_c}, \rho^{(c)}, P_c, y)$. If it reports ERROR, report ERROR and
> terminate the procedure. Otherwise let W_c be the witness returned.
> $I_c \leftarrow$ the set of variables involved in W_c.
> Identify β_c as a partial assignment, where only β_{I_c} is fixed.
> $\rho^{(c+1)} \leftarrow \rho^{(c)} \circ \beta_c$.
> $c \leftarrow c + 1$.

return x

In the following, we use $\mathcal{S}$ to denote Algorithm 4 for brevity. Based on Lemma 6, we prove:

Lemma 8. *If $(R, L_i, S_i, P_i, \beta_i)$ is indeed a global partial witness for ρ, then there exists an advice y that makes $\mathcal{S}$ find $(R, L_i, S_i, P_i, \beta_i)$. More importantly, on a uniformly random $\mathbf{y} \sim \mathcal{U}_n$, $\mathcal{S}$ finds $(\ell_1, \ldots, \ell_r)$ with probability exactly 2^{-S}, where $S := \sum_{i=1}^R S_i$ is the size of the global witness.*

Proof. We examine the execution of $\mathcal{S}$. It first runs Algorithm 2 as a subroutine, trying to find W_1 for P_1. With probability 2^{-S_1} over $\mathbf{y}$, Algorithm 2 succeeds in finding W_1. Conditioning on this happened, we have only committed the assignment of $\mathbf{y}_{I_1}$, and the rest part of $\mathbf{y}$ is still uniformly random. Also note that we set $\rho^{(2)}_{I_1}$ to β_1, which will hide $\mathbf{y}_{I_1}$ in the later execution. This enables us to do an induction and finish the proof. $\qquad\square$

Decoupling. Now, we can observe that $\mathcal{S}$ only needs to know $z = \rho \circ y$ to work. In particular, it does not need to know which part of z is fixed in ρ. Therefore, we can revise $\mathcal{S}$ to get a searcher $\mathcal{S}'$: the input to $\mathcal{S}'$ is now a string z and a global partial witness $(R, L_i, S_i, P_i, \beta_i)$. Otherwise it runs identically the same way as $\mathcal{S}$. We denote the output of $\mathcal{S}'$ as $\mathcal{S}'(z, (R, L_i, S_i, P_i, \beta_i))$. By Lemma 8, we have

$$\Pr_{\rho \sim \mathcal{R}}[(R, L_i, S_i, P_i, \beta_i) \text{ is a global partial witness for } \rho]$$

$$\leq 2^S \cdot \Pr_{\rho \sim \mathcal{R}, y \sim \mathcal{U}_n}[\mathcal{S}'(\rho \circ y, (R, L_i, S_i, P_i, \beta_i)) \text{ is a global witness for } \rho]$$

We observe that

$$\mathbb{1}\{ (R, L_i, S_i, W_i, \beta_i) \text{ is a global witness for } \rho\} \leq \mathbb{1}\left\{ \left(\bigcup_{j=1}^R I_j \right) \subseteq \Lambda \right\}. \tag{11}$$

For a $(t + k)$-wise p-bounded $\mathbf{\Lambda}$, the event on the right hand side holds with probability $p^{\sum_j |I_j|}$.

Then, we have

$$\Pr_{\rho \sim \mathbf{\Lambda}[\mathbf{x},0], y \sim \mathcal{U}_n} [\ \mathcal{S}'(\rho \circ y, (R, L_i, S_i, P_i, \beta_i)) \text{ is a global witness for } \rho]$$

$$= \mathbb{E}_{\mathbf{\Lambda}} \mathbb{E}_{x,y \sim \mathcal{U}_n} \left[\mathbb{1}\{\mathcal{S}'(\mathbf{\Lambda}[x,y], (R, L_i, S_i, P_i, \beta_i)) \text{ is a global witness for } \rho\} \right]$$

$$= \mathbb{E}_{z \sim \mathcal{U}_n} \left[\Pr_{\mathbf{\Lambda}}[\mathcal{S}'(z, (R, L_i, S_i, P_i, \beta_i)) \text{ is a global witness for } \rho] \right]$$

$$\leq p^S,$$

where the third line is due to that $\Lambda[x,y]$ is distributed as $\mathcal{U}_n$ when $x, y \sim \mathcal{U}_n$, and the last inequality holds by (11) and the $(t+k)$-wise p-bounded property of Λ.

Wrapping-up. We finish the proof by enumerating all $(R, L_i, S_i, P_i, \beta_i)$ and taking a summation.

$$\Pr_{\rho \sim \mathbf{\Lambda}[\mathbf{x},\star]}[\mathcal{F}|_\rho \text{ does not have } w\text{-partial depth-}t \text{ decision tree}]$$

$$\leq \sum_{(R,L_i,S_i,P_i,\beta_i)} \Pr_{\rho \sim \mathbf{\Lambda}[\mathbf{x},\star]}[(R, L_i, S_i, P_i, \beta_i) \text{ is a global partial witness for } \rho]$$

$$\leq \sum_{(R,L_i,S_i,P_i,\beta_i)} 2^S \cdot \mathbb{E}_{z \in \mathcal{U}_n} \left[\Pr_{\mathbf{\Lambda}}[\mathcal{S}'(z, (R, L_i, S_i, P_i, \beta_i)) \text{ is a global witness for } \mathbf{\Lambda}[\mathbf{x},\star]] \right]$$

$$\leq \sum_{S=t}^{t+k} 2m^{t/w} 2^S (12k)^S p^S$$

$$\leq 4m^{t/w}(24kp)^t. \tag{12}$$

6.3 Proof when x is pseudorandom

Now we consider the case that $\mathbf{x}$ is not truly random. Our only requirement for $\mathbf{x}$ is that, for every fixed $\mathbf{\Lambda}$, $\mathbf{x}$ is a pseudorandom string that ε-fools CNFs of size at most m. First of all, by Remark 2, the following equation is established.

$$\mathbb{1}\{(R, L_i, S_i, P_i, \beta_i) \text{ is a global partial witness for } \rho\}$$

$$= \sum_{W_i:\text{completion of } P_i} \mathbb{1}\{(R, L_i, S_i, W_i, \beta_i) \text{ is a global witness for } \rho\}. \tag{13}$$

Then, the deduction in (12) implies that

$$\sum_{(R,L_i,S_i,W_i,\beta_i)} \Pr_{\rho \sim \mathbf{\Lambda}[\mathcal{U}_n,\star]}[(R, L_i, S_i, W_i, \beta_i) \text{ is a global witness for } \rho] \leq 4m^{t/w}(24kp)^t. \tag{14}$$

Next, fixing a tuple $(R, L_i, S_i, W_i, \beta_i)$ and an instantiation of Λ, we consider the predicate

$$h_{\Lambda}^{(R,L_i,S_i,W_i,\beta_i)}(x) := \mathbb{1}\{(R, L_i, S_i, W_i, \beta_i) \text{ is a global witness for } \Lambda[x,\star]\}.$$

We omit the superscript and subscript of h when they are clear from context. We claim that h can be implemented by a CNF of size at most $\sum_{i=1}^m \mathsf{size}(F_i) \leq m^2$. To see this, note that $h(x) = 1$ if for every $i \in [R]$, W_i is a witness for $\rho \circ \beta_1 \circ \cdots \circ \beta_{i-1}$. By argument in Section 6.3, this can be verified

using a CNF of size $\mathsf{size}(F_{L_i})$. Therefore, we can evaluate h using an AND over R CNFs, which is itself a larger CNF. Therefore, we conclude that $\mathbf{x}\ \varepsilon$-fools h. That is,

$$\left| \Pr_{x \sim \mathcal{U}_n}[h_\Lambda^{(R,L_i,S_i,W_i,\beta_i)}(x)] - \Pr_{x \sim \mathbf{x}}[h_\Lambda^{(R,L_i,S_i,W_i,\beta_i)}(x)] \right| \le \varepsilon. \tag{15}$$

Finally, we have

$$\Pr_{\rho \sim \Lambda[\mathbf{x},\star]}[\mathcal{F}|_\rho \text{ does not have } w\text{-partial depth-}t \text{ decision tree}]$$

$$\le \sum_{(R,L_i,S_i,W_i,\beta_i)} \Pr_{\rho \sim \Lambda[\mathbf{x},\star]}[(R,L_i,S_i,W_i,\beta_i) \text{ is a global witness for } \rho]$$

$$\le \sum_{(R,L_i,S_i,W_i,\beta_i)} \varepsilon + \Pr_{\rho \sim \Lambda[\mathcal{U}_n,\star]}[(R,L_i,S_i,W_i,\beta_i) \text{ is a global witness for } \rho]$$

$$\le 4m^{t/w}(24kp)^t + (24m)^{t+k} \cdot \varepsilon.$$

Here, the first line is due to the argument in Section 6.2 and observation (13). The second line is due to (15). The last line holds because (14) and there are at most $(24m)^{t+k}$ possible tuples (R,L_i,S_i,W_i,β_i).

Acknowledgements

References

[AW89]   Miklós Ajtai and Avi Wigderson. Deterministic simulation of probabilistic constant depth circuits. *Adv. Comput. Res.*, 5:199–222, 1989.

[BIS12]   Paul Beame, Russell Impagliazzo, and Srikanth Srinivasan. Approximating ac^0 by small height decision trees and a deterministic algorithm for #ac^0sat. In *Computational Complexity Conference*, pages 117–125. IEEE Computer Society, 2012.

[Bra10]   Mark Braverman. Polylogarithmic independence fools AC^0 circuits. *J. ACM*, 57(5):28:1–28:10, 2010.

[CSS18]   Ruiwen Chen, Rahul Santhanam, and Srikanth Srinivasan. Average-case lower bounds and satisfiability algorithms for small threshold circuits. *Theory Comput.*, 14(1):1–55, 2018.

[DETT10] Anindya De, Omid Etesami, Luca Trevisan, and Madhur Tulsiani. Improved pseudorandom generators for depth 2 circuits. In Maria J. Serna, Ronen Shaltiel, Klaus Jansen, and José D. P. Rolim, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, 13th International Workshop, APPROX 2010, and 14th International Workshop, RANDOM 2010, Barcelona, Spain, September 1-3, 2010. Proceedings*, volume 6302 of *Lecture Notes in Computer Science*, pages 504–517. Springer, 2010.

[DMR⁺21] Dean Doron, Raghu Meka, Omer Reingold, Avishay Tal, and Salil P. Vadhan. Monotone branching programs: Pseudorandomness and circuit complexity. *Electron. Colloquium Comput. Complex.*, page 18, 2021.

[FK18]   Michael A. Forbes and Zander Kelley. Pseudorandom generators for read-once branching programs, in any order. In Mikkel Thorup, editor, *59th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2018, Paris, France, October 7-9, 2018*, pages 946–955. IEEE Computer Society, 2018.

[GMR⁺12] Parikshit Gopalan, Raghu Meka, Omer Reingold, Luca Trevisan, and Salil P. Vadhan. Better pseudorandom generators from milder pseudorandom restrictions. In *53rd Annual IEEE Symposium on Foundations of Computer Science, FOCS 2012, New Brunswick, NJ, USA, October 20-23, 2012*, pages 120–129. IEEE Computer Society, 2012.

[GW13]   Oded Goldreich and Avi Wigderson. On the size of depth-three boolean circuits for computing multilinear functions. *Electron. Colloquium Comput. Complex.*, page 43, 2013.

[Has89]   John Hastad. Almost optimal lower bounds for small depth circuits. *Adv. Comput. Res.*, 5:143–170, 1989.

[Hås14]   Johan Håstad. On the correlation of parity and small-depth circuits. *SIAM J. Comput.*, 43(5):1699–1708, 2014.

[HHTT21] Pooya Hatami, William Hoza, Avishay Tal, and Roei Tell. Fooling constant-depth threshold circuits. *Electron. Colloquium Comput. Complex.*, page 2, 2021.

[HLV18]   Elad Haramaty, Chin Ho Lee, and Emanuele Viola. Bounded independence plus noise fools products. *SIAM J. Comput.*, 47(2):493–523, 2018.

[HS19]   Prahladh Harsha and Srikanth Srinivasan. On polynomial approximations to AC. *Random Struct. Algorithms*, 54(2):289–303, 2019.

[IMP12]   Russell Impagliazzo, William Matthews, and Ramamohan Paturi. A satisfiability algorithm for ac⁰. In *SODA*, pages 961–972. SIAM, 2012.

[IMZ12]   Russell Impagliazzo, Raghu Meka, and David Zuckerman. Pseudorandomness from shrinkage. In *53rd Annual IEEE Symposium on Foundations of Computer Science, FOCS 2012, New Brunswick, NJ, USA, October 20-23, 2012*, pages 111–119. IEEE Computer Society, 2012.

[Kel21]   Zander Kelley. An improved derandomization of the switching lemma. In Samir Khuller and Virginia Vassilevska Williams, editors, *STOC '21: 53rd Annual ACM SIGACT Symposium on Theory of Computing, Virtual Event, Italy, June 21-25, 2021*, pages 272–282. ACM, 2021.

[LMN93]   Nathan Linial, Yishay Mansour, and Noam Nisan. Constant depth circuits, fourier transform, and learnability. *J. ACM*, 40(3):607–620, 1993.

[LV20]   Chin Ho Lee and Emanuele Viola. More on bounded independence plus noise: Pseudorandom generators for read-once polynomials. *Theory Comput.*, 16:1–50, 2020.

[MRT19]   Raghu Meka, Omer Reingold, and Avishay Tal. Pseudorandom generators for width-3 branching programs. In Moses Charikar and Edith Cohen, editors, *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing, STOC 2019, Phoenix, AZ, USA, June 23-26, 2019*, pages 626–637. ACM, 2019.

[MZ13]   Raghu Meka and David Zuckerman. Pseudorandom generators for polynomial threshold functions. *SIAM J. Comput.*, 42(3):1275–1301, 2013.

[NW94]   Noam Nisan and Avi Wigderson. Hardness vs randomness. *J. Comput. Syst. Sci.*, 49(2):149–167, 1994.

[ST19]   Rocco A. Servedio and Li-Yang Tan. Improved pseudorandom generators from pseudorandom multi-switching lemmas. In Dimitris Achlioptas and László A. Végh, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2019, September 20-22, 2019, Massachusetts Institute of Technology, Cambridge, MA, USA*, volume 145 of *LIPIcs*, pages 45:1–45:23. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019.

[Tal17]   Avishay Tal. Tight bounds on the fourier spectrum of AC0. In Ryan O'Donnell, editor, *32nd Computational Complexity Conference, CCC 2017, July 6-9, 2017, Riga, Latvia*, volume 79 of *LIPIcs*, pages 15:1–15:31. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2017.

[TX13]   Luca Trevisan and Tongke Xue. A derandomized switching lemma and an improved derandomization of AC0. In *Proceedings of the 28th Conference on Computational Complexity, CCC 2013, K.lo Alto, California, USA, 5-7 June, 2013*, pages 242–247. IEEE Computer Society, 2013.

[Vad12]   Salil P. Vadhan. Pseudorandomness. *Found. Trends Theor. Comput. Sci.*, 7(1-3):1–336, 2012.

[Vio21]   Emanuele Viola. AC0 unpredictability. *ACM Trans. Comput. Theory*, 13(1):5:1–5:8, 2021.