Simplifying Armoni's PRG

Ben Chen
Department of Computer Science, Tel Aviv University, Israel

Amnon Ta-Shma
Department of Computer Science, Tel Aviv University, Israel

Abstract

We propose a simple variant of the INW pseudo-random generator, where blocks have varying lengths, and prove it gives the same parameters as the more complicated construction of Armoni's PRG. This shows there is no need for the specialized PRGs of Nisan and Zuckerman and Armoni, and they can be obtained as simple variants of INW.

For the construction to work we need space-efficient extractors with tiny entropy loss. We use the extractors from [2] instead of [6] taking advantage of the very high min-entropy regime we work with. We remark that using these extractors has the additional benefit of making the dependence on the branching program alphabet Σ correct.

2012 ACM Subject Classification Theory of computation → Pseudorandomness and derandomization

Keywords and phrases PRG, ROBP, read-once, random, psuedorandom, armoni, derandomization

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2025.36

Category RANDOM

Funding *Ben Chen*: Israel Science Foundation (grant number 443/22).
Amnon Ta-Shma: Israel Science Foundation (grant number 443/22).

1 Introduction

In this paper we revisit the problem of constructing pseudo-random generators (PRGs) against bounded width read-once branching programs (ROBPs).[1] Nisan [9] constructed a PRG ε–fooling length t width w ROBP over Σ with seed length $\Theta(\log t \cdot \log \frac{wt|\Sigma|}{\varepsilon})$ using pair-wise independence. Impagliazzo, Nisan and Wigderson [7] gave a variant of the construction with a similar seed length, but using expanders, extractors or samplers instead of pair-wise independence. The INW PRG has seed length $\log|\Sigma| + \Theta(\log t \cdot \log \frac{wt}{\varepsilon})$.

For the special case when the width w of the ROBP is much larger then the length of the ROBP and the error, i.e., $t \leq \log^c w, \varepsilon \geq 2^{-\log^{0.9} w}$ for some constant c, Nisan and Zuckerman [10] constructed a different PRG with an optimal seed length $\Theta(\log \frac{wt}{\varepsilon})$. Armoni [1] recursively used the NZ construction, and with a careful analysis obtained a PRG with seed length $\Theta(\log t \cdot \frac{\log \frac{wt}{\varepsilon}}{\max\{\log \frac{\log w}{\log \frac{t}{\varepsilon}}, 1\}})$ for constant size Σ where $t \leq 2^{\log^{1-\varepsilon} w}$ for some constant $\varepsilon > 0$.

In [8] Armoni's PRG is instantiated with a specific space-explicit extractor removing the parameters limit, and the resulting seed length is $2\log|\Sigma| + \theta(\log t \cdot \frac{\log \frac{wt}{\varepsilon}}{\max\{\log \frac{\log w}{\log \frac{t}{\varepsilon}}, 1\}})$. Armoni's PRG with [8] bridges the gap between INW and NZ - it asymptotically matches INW when t/ε is large, and it matches NZ when t/ε is small compared to w.

In this paper we construct another PRG that matches the parameters obtained in Armoni. The surprising thing about our construction is that it avoids altogether the NZ PRG, and also avoids the recursion in Armoni, and instead it is a direct, simple variant of INW. Saying

[1] For a formal definition of ROBPand PRG see Definitions 1 and 2.

it differently, Armoni interpolates between two different PRG constructions: the NZ PRG and the INW PRG, and his construction is a recursive combination of both. Instead, we show that a simple variant of INW alone gives the same bounds for all parameter regimes.

To understand our new construction, we revisit the INW construction. The main building block behind the INW construction is showing that one can recycle the randomness given to a PRG, where the recycling is done using an expander, or more generally, a sampler or an extractor. Specifically, instead of applying twice a PRG with two independent seeds, one can apply it once (consuming s random bits) and then use a short seed (of length $d \ll s$) to recycle the s bits and get from it a new seed for the second application of the PRG. The constructions in [9, 7] preserve the same seed lengths for the two applications of the PRGs (we say, in short, that INW preserves the block length).

Nisan and Zuckerman observe that the seed needed for the extractor has length $d = O(\log \frac{t}{\varepsilon})$ while the string it acts upon has length $s = \Omega(\log w)$, and so when $t/\varepsilon \ll w$, instead of applying the extractor once, it is more beneficial to apply it $u = O(\frac{\log w}{\log t/\varepsilon})$ times with independent seeds of length d. Extending these ideas, Armoni implements this idea recursively.

In this paper we suggest an alternative to NZ and Armoni where instead of taking a large u so that $ud = s$, we keep $u = 1$ and instead track down our losses. Next, we explain this idea in detail.

Suppose we start with a seed y of length s and we want to create from it two seeds for two independent applications of the PRG. The way INW achieves this is by carving out d bits from y and reserving them for the extractor application. Specifically, say $y = y_{left} \circ y_{right}$ where y_{right} has length d. Then, in INW, the first PRG is called with the seed y_{left}, and the second PRG is called with the seed $Ext(y_{left}, y_{right})$, where Ext is an extractor. Thus, the seed length for the first application is $s - d$ and the seed length for the second application is the extractor output size, and it suffers the losses the extractor suffers. These include:

- The amount of information the machine knows about y_{left} after seeing the output of the first PRG. This loss amounts to $\ell = O(\log \frac{tw}{\varepsilon})$.
- The extractor entropy loss, which is $2 \log 1/\varepsilon$ in non-explicit constructions. For the time being let us assume we can achieve a $O(\ell)$ loss explicitly, and then this loss is comparable with the first loss.

Thus, the seed loss in the first PRG application is $d = O(\log \frac{t}{\varepsilon})$, whereas the loss in the second PRG application is larger and is $O(\log \frac{tw}{\varepsilon})$. The two losses are quite different when $t/\varepsilon \ll w$.

We then do the obvious. We define a recursive construction, where starting with a seed length s we get two recursive calls, one with seed length $s - d$ and one with seed length $s - \ell$, where $d = O(\log \frac{t}{\varepsilon})$ and $\ell = O(\log \frac{wt}{\varepsilon})$. Doing the analysis we find out that we recover the bound $\log |\Sigma| + \Theta(\log t \cdot \frac{\ell}{\log(\frac{\ell}{d}+1)})$ in a much simpler way.

To summarize, when $t/\varepsilon \ll w$, it is much better to employ the INW strategy with varying block lengths, and doing so gives the correct parameters in a straight forward way. However, NZ and later Armoni try to preserve the block length throughout different calls. This forces NZ to use several calls of the extractor, and later forces Armoni to use specially crafted recursion (and a more careful analysis). What we show here is that all of that is unnecessary, and with varying block lengths the INW construction obtains the same parameters as NZ and Armoni.

For our construction to work, we require space-efficient extractors with small entropy loss. In general, there are non-explicit (k, ϵ) extractors $E : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ with entropy loss $2 \log 1/\varepsilon + O(1)$, i.e., $m = k + d - 2 \log 1/\varepsilon - O(1)$, and notice that the

Table 1 PRG for standard order ROBP.

Seed Size	Reference	Remarks
$\Theta(\log t \cdot \log \frac{wt\lvert\Sigma\rvert}{\varepsilon})$	[9]	
$\log\lvert\Sigma\rvert + \Theta(\log t \cdot \log \frac{wt}{\varepsilon})$	[7]	
$\Theta(\log w)$	[10]	When $\log w = (t/\varepsilon)^\beta$ for some $\beta > 0$
$2\log\lvert\Sigma\rvert + \Theta(\log t \cdot \frac{\log \frac{wt}{\varepsilon}}{\max\{\log \frac{\log w}{\log \frac{t}{\varepsilon}}, 1\}})$	[1]	With the extractors of [8]
$\log\lvert\Sigma\rvert + \Theta(\log t \cdot \frac{\log \frac{wt}{\varepsilon}}{\max\{\log \frac{\log w}{\log \frac{t}{\varepsilon}}, 1\}})$	This paper	Also, [1] with the extractor of Theorem 4

entropy loss is independent of k. Kane et al. in [8] instantiated Armoni's PRG with the GUV extractor [6, Theorem 5.10] which has $\Omega(k)$ entropy-loss, and they show it is space-efficient. However, $\Omega(k)$ loss is too much for us.

In general, the best *explicit* extractors have $\Omega(\frac{k}{poly(\log k))})$ entropy-loss [5, 11]. What saves us here is that we do not need general extractors but rather extractors working in the very-high min-entropy regime, where $\Delta = n - k$ is small. In this regieme one can split the source to a large block followed by a second small block of length $O(\Delta)$, and then use a block-wise extractor. This was implemented in [2]. The resulting extractor is cited in Theorem 4, has low entropy-loss and is space-efficient.

Replacing the lossy extractor used in [8] with the small entropy-loss extractor of [2] has other benefits. It turns out this also reduces the additive dependence on $|\Sigma|$ from $2\log|\Sigma|$ to $\log|\Sigma|$. We summarize the parameters obtained by the previous constructions and by our construction in Table 1.

We remark that while the improvement in the dependence of Σ may seem insignificant, it can help simplify certain constructions. A notable example is the recent work of Cheng and Wu [3] which employs an iterative process of alternating steps of length and alphabet reductions. Using our PRG (and [2] extractor) in the length reduction, the alphabet reduction becomes unnecessary.

2 Preliminaries

$[k]$ denotes the set $\{1, \ldots, k\}$. For a $k \times k$ matrix M and $i, j \in [k]$, $M[i, j]$ is the value of M at the i'th row and j'th column. $\|M\|$ is the spectral norm of M. For every $f : [w] \to [w]$ there is a corresponding $w \times w$ boolean matrix M_f such that $M_f[i, j] = 1$ iff $f(j) = i$. We denote the set of such matrices by $SBM_{w\times w}$ (stochastic, boolean matrices).

▶ **Definition 1** (ROBP). *Let Σ be an arbitrary subset, $w, t \in \mathbb{N}$. B is a width w length t read once branching program (*ROBP*) on alphabet Σ if it is a sequence of t functions $(B_1, B_2, ..., B_t)$, with $B_i : \Sigma \to SBM_{w\times w}$. The* evaluation *of B on input $\sigma_1, ...\sigma_t \in \Sigma^t$ is the linear operator $B(\sigma_1, ..., \sigma_t) \stackrel{\text{def}}{=} B_t(\sigma_t) \cdot ... \cdot B_1(\sigma_1)$. We also say B is a $(w, t, \Sigma) -$* ROBP.

▶ **Definition 2** (PRG). *Let Σ be an arbitrary subset and $s, t \in \mathbb{N}$. A (s, t, Σ) pseudo random generator is a function $PRG : \{0,1\}^s \to \Sigma^t$. For $\epsilon > 0$ a we say PRG ϵ-fools $(w, t, \Sigma)-$*ROBP *if for every $(w, t, \Sigma) -$* ROBP *B we have:*

$$\|\mathbf{E}_{\sigma\in\Sigma^t} B(\sigma) - \mathbf{E}_{x\in\{0,1\}^s} B(PRG(x))\| \leq \epsilon$$

2.1 Extractors

U_n denotes the uniform distribution on n bits. The *min-entropy* of a random source X, denoted $\mathcal{H}_\infty(X)$, is $\mathcal{H}_\infty(X) \stackrel{\text{def}}{=} \min_{\omega \in \text{Supp}(X)} \log \frac{1}{\Pr_{x\sim X}(x=\omega)}$. The *statistical distance* between two random random variables defined over a domain Ω is $SD_\Omega(X,Y) \stackrel{\text{def}}{=} \frac{1}{2}\sum_{\omega\in\Omega} |\Pr_{x\sim X}[x = \omega] - \Pr_{y\sim Y}[y=\omega]|$. The statistical distance can be equivalently defined as $SD_\Omega(X,Y) \stackrel{\text{def}}{=} \max_{f:\Omega\to\{0,1\}} |\mathbf{E}_{x\sim X} f(x) - \mathbf{E}_{y\sim Y} f(y)|$.

▶ **Definition 3** (extractor). *Let $n, d, k, m \in \mathbb{N}$ and $\epsilon > 0$. A function $Ext : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ is a (k,ϵ)* extractor *if for every random variable X over $\{0,1\}^n$ with $\mathcal{H}_\infty(X) \geq k$ it holds that*

$$SD_{\{0,1\}^m}(Ext(X,U_d),U_m) \leq \epsilon$$

▶ **Theorem 4** ([2] High min-entropy extractor with a small entropy loss). *For $n > k$, $\epsilon > 0$, there is a family of extractors $E_{high} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ that is a (k,ϵ) extractor with $d = O(\log(n-k) + \log\frac{1}{\epsilon})$ and entropy loss $n + d - m \leq O(n - k + \log 1/\varepsilon)$. Furthermore, $E_{high}(x,y)$ can be computed in time poly$(n \log\frac{1}{\epsilon})$ and space $O(n - k + \log n + \log\frac{1}{\varepsilon})$ for all x, y.*

Cohen et al. in [4] used a more careful space analysis to show a different space bound $O(\log m + \log m \log\frac{1}{\varepsilon})$. For our analysis the bound of Chattopadhyay and Liao is sufficient.

3 The new PRG

As explained in the introduction, we apply the INW approach, each time replacing a seed with two different (shorter) seeds. Unlike INW the two seeds have *different* lengths. Specifically, if we start with an initial seed of length s, then we can only pass $s - d$ bits to the first application of the PRG, because we need to keep d independent bits for the recycling step. When we recycle the randomness, say with an extractor, and get a new seed for the second PRG, we can recover these d bits but we have two new losses:

- An entropy loss of order $O(\log\frac{1}{\varepsilon'})$, where ε' is the extractor error, which we take to be $\Theta(\frac{\varepsilon}{t})$, where ε is the final error, and t is the final number of blocks, and,
- A $\log\frac{w}{\varepsilon'}$ loss, that is due because of the information collected in the width w branching program after seeing the output of the first PRG.

Thus, for the second application we lose $\ell = O(\log\frac{wt}{\varepsilon})$ bits. As $d = O(\log\frac{t}{\varepsilon})$ it is significantly smaller than ℓ when $w \gg \frac{t}{\varepsilon}$. To summarize, we replace a length s seed, with two seeds, one of length $s - d$ and the other of length $s - \ell$ where $d = O(\log\frac{t}{\varepsilon})$ and $\ell = O(\log\frac{wt}{\varepsilon})$, where the constants behind the big O notation are essentially determined by the seed length and the entropy loss of the explicit extractor that we use (plus an additive $\log\frac{w}{\varepsilon'}$ added to ℓ).

Having the recycling building block, we use it recursively and define a sequence of PRGs. Applying this idea in a tree-like construction, we have the following construction:

▶ **Construction 5** (INW with varying block lengths). *Given a set Σ of size 2^σ, parameters $d, \ell \in \mathbb{N}$ and a family*

$$\{Ext_s : \{0,1\}^{s-d} \times \{0,1\}^d \to \{0,1\}^{s-\ell}\}_{s\in\mathbb{N}}$$

define a family $\{P_s : \{0,1\}^s \to \Sigma^{t(s)}\}_{s\in\mathbb{N}}$ of PRG*s by*

$$P_s(x \circ y) = \begin{cases} P_{s-d}(x) \circ P_{s-\ell}(Ext_s(x,y)) & \text{If } |x \circ y| \geq \sigma + \ell \\ \text{The first } \sigma \text{ bits of } x \circ y & \text{If } \sigma \leq |x \circ y| < \sigma + \ell \end{cases}$$

where $t(s) = 1$ for $\sigma \leq s < \sigma + \ell$ and $t(s) = t(s-d) + t(s-\ell)$ for $s \geq \sigma + \ell$.

▶ **Theorem 6.** *Let $w, t \in \mathbb{N}$, $\varepsilon > 0$, Σ a set of size 2^σ. There is a large enough constant c s.t. setting $d = c \log \frac{t}{\varepsilon}$, $\ell = c \log \frac{wt}{\varepsilon}$, and assuming a family $\{Ext_s : \{0,1\}^{s-d} \times \{0,1\}^d \to \{0,1\}^{s-\ell}\}_{s\in\mathbb{N}}$ of $(s - d - \log \frac{w}{\varepsilon'}, \varepsilon')$ extractors, for $\varepsilon' = \frac{\varepsilon}{6t}$, we have that $\{P_s\}_{s\in\mathbb{N}}$ as in Construction 5 is a $(s,t,\Sigma)-$*PRG *ε-fooling $(w,t,\Sigma)-$*ROBP *with $s = \sigma + \Theta\left(\frac{\log \frac{wt}{\varepsilon} \log t}{\log\left(2 + \frac{\log w}{\log \frac{t}{\varepsilon}}\right)}\right)$.*

We need to analyze the output length of the generator and to prove correctness. We start by analyzing the output length of P_s as a function of the seed length s.

3.1 The seed length

Recall that $t(s)$ is the number of Σ symbols the PRG P_s outputs. Conversely, let us denote by $s(t)$ the seed length needed to output t symbols, i.e., the minimal integer such that $t(s_t) \geq t$. Then,

▶ **Lemma 7** (seed size). $s(t) = \sigma + \Theta(\frac{\ell \log t}{\log(\frac{\ell}{d}+1)})$.

To gain intuition, think of the recursion in Construction 5 as a tree, where at the root we have our initial seed, and every non-leaf vertex with seed length $s > \sigma + \ell$ has two children: a left child with seed length $s - d$, and a right child with seed length $s - \ell$. The leaves are vertices with $\sigma \leq s < \sigma + \ell$. A path in the tree is a sequence of left and right steps from the root to a leaf. Unlike the PRGs of [9, 7, 1], where all paths have the same length, in our construction different paths have different lengths.

Proof. Without a loss of generality assume ℓ is an integer multiple of d and $s - \sigma$ is an integer multiple of ℓ.

$$t(s) = \sum_{\substack{k_L, k_R \in \mathbb{N} \\ k_L \cdot d + k_R \cdot \ell + \sigma = s}} \binom{k_L + k_R}{k_R}$$

where k_L (resp. k_R) is the number of left (resp. right) steps in the path. While we need a lower bound on $t(s)$ we also derive a matching upper bound.

For the upper bound we notice that $k_L \leq \frac{s-\sigma}{d}$ and $k_R \leq \frac{s-\sigma}{\ell}$. As $\binom{a+b}{b}$ is an increasing monotone function for each parameter a and b when the other is fixed, we conclude that

$$\binom{k_L + k_R}{k_R} \leq \binom{\frac{s-\sigma}{d} + \frac{s-\sigma}{\ell}}{\frac{s-\sigma}{\ell}}$$

Also notice that there are at most $\frac{s-\sigma}{\ell}$ legal assignments for K_R.

For the lower bound we notice that $k_L = \frac{s-\sigma}{2d}$ and $k_R = \frac{s-\sigma}{2\ell}$ is a legal assignment. Thus,

$$\binom{\frac{s-\sigma}{2d}+\frac{s-\sigma}{2\ell}}{\frac{s-\sigma}{2\ell}} \leq t(s) \leq \frac{s-\sigma}{\ell} \cdot \binom{\frac{s-\sigma}{d}+\frac{s-\sigma}{\ell}}{\frac{s-\sigma}{\ell}}$$

Using $\frac{s-\sigma}{\ell} \leq \log t(s)$ and $(\frac{n}{k})^k \leq \binom{n}{k} \leq (\frac{en}{k})^k$, we get

$$t(s) \geq \left(\frac{\frac{s-\sigma}{2d}+\frac{s-\sigma}{2\ell}}{\frac{s-\sigma}{2\ell}}\right)^{\frac{s-\sigma}{2\ell}} = \left(\frac{\ell}{d}+1\right)^{\frac{s-\sigma}{2\ell}}$$

$$t(s) \leq \log t(s) \cdot \left(e \cdot \frac{\frac{s-\sigma}{d}+\frac{s-\sigma}{\ell}}{\frac{s-\sigma}{\ell}}\right)^{\frac{s-\sigma}{\ell}} = \log t(s) \cdot \left(\frac{e\ell}{d}+e\right)^{\frac{s-\sigma}{\ell}}$$

Thus, for $s_{up} = \sigma + \frac{2\ell \log t}{\log(\frac{\ell}{d}+1)}$ we have $t(s_{up}) \geq (\frac{\ell}{d}+1)^{\frac{s_{up}-\sigma}{2\ell}} = t$ and therefore $s(t) \leq s_{up}$. Similarly, for $s_{low} = \sigma + \frac{\ell \log(t/\log t)}{\log(\frac{e\ell}{d}+e)}$ we have $\frac{t(s_{low})}{\log t(s_{low})} \leq (\frac{e\ell}{d}+e)^{\frac{s_{low}-\sigma}{\ell}} = \frac{t}{\log t}$ and therefore $s(t) \geq s_{low}$. I.e.,

$$\frac{\ell \log \frac{t}{\log t}}{\log(\frac{e\ell}{d}+e)} \leq s(t) - \sigma \leq \frac{2\ell \log t}{\log(\frac{\ell}{d}+1)}$$

We conclude that

$$s(t) = \sigma + \Theta\left(\frac{\ell \log t}{\log(\frac{\ell}{d}+1)}\right),$$

proving Lemma 7. ◀

3.2 Correctness

▶ **Lemma 8** (Following [7]). *Suppose*

- P_1 *is a* $(s_1, t_1, \Sigma) - \mathsf{PRG}$ *that* ϵ_1*-fools* $(w, t_1, \Sigma) - \mathsf{ROBP}$*, and,*
- P_2 *is a* $(s_2, t_2, \Sigma) - \mathsf{PRG}$ *that* ϵ_2*-fools* $(w, t_2, \Sigma) - \mathsf{ROBP}$*,*

for some $s_1, s_2, t_1, t_2, d \in \mathbb{N}$ *and* $\epsilon_1, \epsilon_2 > 0$*. Further assume*

- $E : \{0,1\}^{s_1} \times \{0,1\}^d \to \{0,1\}^{s_2}$ *is a* $(s_1 - \log \frac{w}{\varepsilon'}, \varepsilon')$ *extractor.*

Then $G : \{0,1\}^{s_1} \times \{0,1\}^d \to \Sigma^{t_1+t_2}$ *defined by*

$$G(x,y) = P_1(x) \circ P_2(E(x,y))$$

$(3\varepsilon' + \epsilon_1 + \epsilon_2)$*-fools* $(w, t_1+t_2, \Sigma) - \mathsf{ROBP}$*.*

The main difference between Lemma 8 and the corresponding lemma in [7] is that P_1 and P_2 take different input lengths. For completeness we give the proof:

Proof. Let B be a $(w, t = t_1 + t_2, \Sigma)$-ROBP. The main claim is

▷ Claim 9. $\|\mathbf{E}_{x \sim U_{s_1}, y \sim U_d} B(P_1(x) \circ P_2(E(x,y))) - \mathbf{E}_{x \sim U_{s_1}, y \sim U_{s_2}} B(P_1(x) \circ P_2(y))\| \leq 3\varepsilon'$.

Proof. Let v be the state in layer t_1 of B reached after taking a walk on B according to $P_1(x)$. We split our expectation into two cases.

- We reach a vertex v that has at most $\frac{2^{s_1} \cdot \varepsilon'}{w}$ sources. Using the union bound over the states, we see that the probability over x of this event is at most ε'. This gives an error of at most ε'.

- We reach a vertex v that has at least $\frac{2^{s_1}\cdot\varepsilon'}{w}$ sources. In this case, the min-entropy of x conditioned on v is at least $s_1 - \log\frac{w}{\varepsilon'}$. Since E is an $(s_1 - \log\frac{w}{\varepsilon'}, \varepsilon')$ extractor, from the adversary point of view, the distributions $E(x, U_d)$ and U_{s_2} are ε' statistically close, and therefore this adds at most $2\varepsilon'$ to the distance.

This proves Claim 9. ◁

Thus,

$$\|\mathbf{E}_{x,y}B(P_1(x)\circ P_2(E(x,y))) - \mathbf{E}_{\sigma\in\Sigma^t}B(\sigma)\| \leq$$
$$\|\mathbf{E}_{x,y}B(P_1(x)\circ P_2(E(x,y))) - \mathbf{E}_{x,y}B(P_1(x)\circ P_2(y))\| +$$
$$\|\mathbf{E}_{x,y}B(P_1(x)\circ P_2(y)) - \mathbf{E}_{\sigma\in\Sigma^{t_1},y}B(\sigma\circ P_2(y))\| +$$
$$\|\mathbf{E}_{\sigma\in\Sigma^{t_1},y}B(\sigma\circ P_2(y)) - \mathbf{E}_{\sigma\in\Sigma^t}B(\sigma)\|$$

The first expression is bounded by $3\varepsilon'$ by Claim 9, the second expression by ε_1 because P_1 fools $(w, t_1, \Sigma) - \mathsf{ROBP}$ and the third by ε_2 because P_2 fools $(w, t_2, \Sigma) - \mathsf{ROBP}$, completing the proof of Lemma 8. ◀

Once we know how the error accumulates in a single application we can deduce how it accumulates throughout the tree:

▶ **Lemma 10** (error accumulation). *P_s ε''-fools $(w, t(s), \Sigma) - \mathsf{ROBP}$ for $\varepsilon'' = 3(t(s)-1)\cdot\varepsilon'$.*

Proof. (of Lemma 10) By induction on s. For $s < \sigma + \ell$ the PRG returns truly uniform bits. Let us prove for $s \geq \sigma + \ell$. By Lemma 8, $\epsilon_s \leq \epsilon_{s-d} + \epsilon_{s-\ell} + 3\varepsilon'$. By induction $\epsilon_s \leq 3\varepsilon'(t(s-d)-1) + 3\varepsilon'(t(s-\ell)-1) + 3\varepsilon' = 3\varepsilon'(t(s-d) + t(s-\ell) - 1)$. The proof is complete using $t(s) = t(s-d) + t(s-\ell)$. ◀

Lemma 7 together with Lemma 8 prove Theorem 6, because $\varepsilon'' \leq 3t(s)\varepsilon' \leq 6t\varepsilon' \leq \varepsilon$.

References

1. Roy Armoni. On the derandomization of space-bounded computations. In *International Workshop on Randomization and Approximation Techniques in Computer Science*, pages 47–59. Springer, 1998. doi:10.1007/3-540-49543-6_5.
2. Eshan Chattopadhyay and Jyun-Jie Liao. Optimal error pseudodistributions for read-once branching programs. *arXiv preprint arXiv:2002.07208*, 2020. arXiv:2002.07208.
3. Kuan Cheng and Ruiyang Wu. Weighted pseudorandom generators for read-once branching programs via weighted pseudorandom reductions. *arXiv preprint arXiv:2502.08272*, 2025. doi:10.48550/arXiv.2502.08272.
4. Gil Cohen, Dean Doron, Ori Sberlo, and Amnon Ta-Shma. Approximating iterated multiplication of stochastic matrices in small space. In *Proceedings of the 55th Annual ACM Symposium on Theory of Computing*, pages 35–45, 2023. doi:10.1145/3564246.3585181.
5. Zeev Dvir, Swastik Kopparty, Shubhangi Saraf, and Madhu Sudan. Extensions to the method of multiplicities, with applications to kakeya sets and mergers. *SIAM Journal on Computing*, 42(6):2305–2328, 2013. doi:10.1137/100783704.
6. Venkatesan Guruswami, Christopher Umans, and Salil Vadhan. Unbalanced expanders and randomness extractors from parvaresh–vardy codes. *Journal of the ACM (JACM)*, 56(4):1–34, 2009. doi:10.1145/1538902.1538904.
7. Russell Impagliazzo, Noam Nisan, and Avi Wigderson. Pseudorandomness for network algorithms. In *Proceedings of the twenty-sixth annual ACM symposium on Theory of computing*, pages 356–364, 1994. doi:10.1145/195058.195190.
8. Daniel M Kane, Jelani Nelson, and David P Woodruff. Revisiting norm estimation in data streams. *arXiv preprint arXiv:0811.3648*, 2008. arXiv:0811.3648.

9 Noam Nisan. Pseudorandom generators for space-bounded computations. In *Proceedings of the twenty-second annual ACM symposium on Theory of computing*, pages 204–212, 1990. doi:10.1145/100216.100242.

10 Noam Nisan and David Zuckerman. Randomness is linear in space. *Journal of Computer and System Sciences*, 52(1):43–52, 1996. doi:10.1006/JCSS.1996.0004.

11 Amnon Ta-Shma and Christopher Umans. Better condensers and new extractors from parvaresh-vardy codes. In *2012 IEEE 27th Conference on Computational Complexity*, pages 309–315. IEEE, 2012. doi:10.1109/CCC.2012.25.