



# On Sums of INW Pseudorandom Generators

William M. Hoza   

Department of Computer Science, The University of Chicago, IL, USA

Zelin Lv   

Department of Computer Science, The University of Chicago, IL, USA

---

## Abstract

We study a new approach for constructing pseudorandom generators (PRGs) that fool constant-width standard-order read-once branching programs (ROBPs). Let  $X$  be the  $n$ -bit output distribution of the INW PRG (Impagliazzo, Nisan, and Wigderson, STOC 1994), instantiated using expansion parameter  $\lambda$ . We prove that the bitwise XOR of  $t$  independent copies of  $X$  fools width- $w$  programs with error  $n^{\log(w+1)} \cdot (\lambda \cdot \log n)^t$ . Notably, this error bound is meaningful even for relatively large values of  $\lambda$  such as  $\lambda = 1/O(\log n)$ .

Admittedly, our analysis does not yet imply any improvement in the bottom-line overall seed length required for fooling such programs – it just gives a new way of re-proving the well-known  $O(\log^2 n)$  bound. Furthermore, we prove that this shortcoming is not an artifact of our analysis, but rather is an intrinsic limitation of our “XOR of INW” approach. That is, no matter how many copies of the INW generator we XOR together, and no matter how we set the expansion parameters, if the generator fools width-3 programs and the proof of correctness does not use any properties of the expander graphs except their spectral expansion, then we prove that the seed length of the generator is inevitably  $\Omega(\log^2 n)$ .

Still, we hope that our work might be a step toward constructing near-optimal PRGs fooling constant-width ROBPs. We suggest that one could try running the INW PRG on  $t$  *correlated* seeds, sampled via another PRG, and taking the bitwise XOR of the outputs.

**2012 ACM Subject Classification** Theory of computation → Pseudorandomness and derandomization

**Keywords and phrases** INW generator, pseudorandomness, space-bounded computation, XOR Lemmas

**Digital Object Identifier** 10.4230/LIPIcs.APPROX/RANDOM.2025.67

**Category** RANDOM

**Related Version** *Full Version*: <https://eccc.weizmann.ac.il/report/2025/050/> [20]

**Acknowledgements** We thank Huacheng Yu for collaboration during the early stages of this project. We thank Gil Cohen and Dean Doron for valuable discussions. We thank Aaron Potechin for valuable discussions and helpful comments on a draft of this paper.

## 1 Introduction

### 1.1 Pseudorandom generators for space-bounded computation

Randomness can be considered a type of “fuel” for computation. We prefer to use as little randomness as possible, just like we prefer to minimize consumption of other types of “fuel.” A *pseudorandom generator* (PRG) is a way of decreasing the number of random bits used in computation.

► **Definition 1** (PRG). *Let  $X$  be a distribution over  $\{0, 1\}^n$  and let  $f: \{0, 1\}^n \rightarrow \{0, 1\}$ . We say that  $X$  fools  $f$  with error  $\varepsilon$  if*

$$|\mathbb{E}[f(X)] - \mathbb{E}[f]| \leq \varepsilon.$$



© William M. Hoza and Zelin Lv;

licensed under Creative Commons License CC-BY 4.0

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2025).

Editors: Alina Ene and Eshan Chattopadhyay; Article No. 67; pp. 67:1–67:24



Leibniz International Proceedings in Informatics

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

Here  $\mathbb{E}[f]$  is a shorthand for  $\mathbb{E}[f(U_{\{0,1\}^n})]$ , where, in general,  $U_{\mathcal{R}}$  denotes the uniform distribution over the finite set  $\mathcal{R}$ . A pseudorandom generator (PRG) is a function  $G: \mathcal{R} \rightarrow \{0,1\}^n$  for some finite set  $\mathcal{R}$ . We say that  $G$  fools  $f$  with error  $\varepsilon$  if  $G(U_{\mathcal{R}})$  fools  $f$  with error  $\varepsilon$ . The seed length of the PRG is the quantity  $\log |\mathcal{R}|$ .<sup>1</sup>

In this paper, we study PRGs in the context of space-bounded computation. If we wish to simulate a randomized space-bounded algorithm, then we ought to use a PRG that fools standard-order *read-once branching programs* (ROBPs), defined next.

► **Definition 2** (Standard-order RBP). *A width- $w$  length- $n$  standard-order read-once branching program (ROBP) is a directed acyclic multigraph. The vertices are arranged in  $n+1$  layers  $V_0, V_1, \dots, V_n$ , each consisting of  $w$  vertices. Each vertex in  $V_i$  where  $i < n$  has two outgoing edges leading to  $V_{i+1}$  labeled 0 and 1. One vertex  $v_0 \in V_0$  is designated as the start vertex, and each vertex  $v \in V_n$  is labeled with an output value  $q_v \in \{0,1\}$ . Each input  $x \in \{0,1\}^n$  selects a walk through the program, defined by starting at  $v_0$  and traversing the outgoing edge with label  $x_i$  in step  $i$ . This walk ends at some vertex  $v \in V_n$ . The program computes the function  $f$  given by  $f(x) = q_v$ .*

Polynomial-width standard-order ROBPs describe what log-space randomized algorithms do on a fixed input as a function of their random bits. In this paper, we focus on the constant-width case. There isn't necessarily a clear connection between constant-width ROBPs and uniform models of computation such as randomized Turing machines, but constant-width ROBPs still constitute an extremely interesting nonuniform model of space-bounded computation. The width-2 case is well-understood [44, 4, 19, 29]. In particular, Saks and Zuckerman showed in the 1990s that there is an explicit PRG that fools width-2 branching programs with seed length  $O(\log n)$  [44], which is optimal. Decades later, Meka, Reingold, and Tal showed that there is an explicit PRG that fools width-3 standard-order ROBPs with seed length  $\tilde{O}(\log n)$  [35]. However, the best seed length bound for width-4 programs is  $O(\log^2 n)$ , which is a special case of the following classic theorem.

► **Theorem 3** ([38]). *For every  $w, n \in \mathbb{N}$  and every  $\varepsilon \in (0,1)$ , there exists an explicit PRG that fools width- $w$  length- $n$  standard-order ROBPs with error  $\varepsilon$  and seed length  $O(\log(wn/\varepsilon) \cdot \log n)$ .*

The original proof of Theorem 3 is due to Nisan [38]. There is an alternative proof due to Impagliazzo, Nisan, and Wigderson [26] that is more relevant to the present paper. Impagliazzo, Nisan, and Wigderson inductively defined a sequence of PRGs  $G_0, G_1, \dots$ , where  $G_j: \mathcal{R}_j \rightarrow \{0,1\}^{2^j}$ , as follows.

1. Base case: Let  $\mathcal{R}_0 = \{0,1\}$  and  $G_0(x) = x$ .
2. Inductive step: Assume we have already constructed  $G_{j-1}$ . Let  $H_j$  be a  $D_j$ -regular undirected multigraph on the vertex set  $\mathcal{R}_{j-1}$ . Define  $\mathcal{R}_j = \mathcal{R}_{j-1} \times [D_j]$  and

$$G_j(x, y) = (G_{j-1}(x), G_{j-1}(H_j[x, y])).$$

Here  $H_j[x, y]$  denotes the  $y$ -th neighbor of the vertex  $x$  in  $H_j$ . Impagliazzo, Nisan, and Wigderson's original analysis [26] shows that if  $\lambda(H_j) \leq \lambda$  for every  $j$ , then  $G_{\log n}$  fools width- $w$  length- $n$  standard-order ROBPs with error  $\lambda \cdot w \cdot n$ . Here  $\lambda(H_j)$  denotes the *spectral expansion parameter* of  $H_j$ , which can be defined as the second largest eigenvalue of its transition probability matrix in absolute value. There are numerous explicit

<sup>1</sup> Usually we want the domain to have the form  $\mathcal{R} = \{0,1\}^s$  for some  $s \in \mathbb{N}$ , but in this work we allow arbitrary domains.

constructions of expander graphs  $H_j$  such that  $\lambda(H_j) \leq \lambda$  and  $\deg(H_j) \leq \text{poly}(1/\lambda)$ . (See, for example, Vadhan’s pseudorandomness survey [47].) If we use such expander graphs, then the seed length of  $G_{\log n}$  is  $O(\log n \cdot \log(1/\lambda))$ . Choosing  $\lambda = \frac{\epsilon}{wn}$  completes the proof of Theorem 3.

## 1.2 Instantiating the INW PRG with relatively mild expanders

It is a major open problem to design PRGs that fool constant-width standard-order ROBPs with seed length  $o(\log^2 n)$ . A natural thing to try is to simply increase the parameter  $\lambda$  in the INW construction. Indeed, even when  $\lambda$  is relatively large, it turns out that the INW generator still does a good job of fooling so-called “regular” and “permutation” ROBPs [8, 15, 28, 45, 22]. More generally, one can try using different values of  $\lambda$  at the different levels of the recursion, say  $\lambda_1, \dots, \lambda_{\log n}$ . For example, Rozenman and Vadhan showed how to use the INW generator, with  $\lambda_j = \Omega(1)$  for most but not all  $j$ , to solve the undirected  $s$ - $t$  connectivity problem in deterministic log-space [43], re-proving Reingold’s famous theorem [42].

Unfortunately, it turns out that no matter how we set the expansion parameters, the INW generator is provably too weak to fool constant-width standard-order ROBPs with seed length  $o(\log^2 n)$  [9, 23]. Making this statement precise is a little subtle, because “the” INW generator is actually a whole family of PRGs, even after we fix the expansion parameters  $\lambda_1, \dots, \lambda_{\log n}$ . After all, the definition of the PRG does not specify which specific expander graphs to use. For a vector  $\vec{\lambda} = (\lambda_1, \dots, \lambda_{\log n})$ , let us define  $\text{INW}(\vec{\lambda})$  to be the set of all PRGs  $G_{\log n}$  that can be constructed via the INW template using graphs  $H_1, \dots, H_{\log n}$  satisfying  $\lambda(H_j) \leq \lambda_j$  for every  $j$ . As a shorthand, let us also write  $\text{INW}(\lambda)$  for the special case  $\lambda_1 = \lambda_2 = \dots = \lambda_{\log n} = \lambda$  when  $n$  is clear from context. (See Definition 23 for a more detailed definition.) Then we have the following theorem due to Brody and Verbin [9], with details filled in by Hoza, Pyne, and Vadhan [23]:

► **Theorem 4** (Limitations of the INW generator [9, 23]). *Let  $\vec{\lambda} = (\lambda_1, \dots, \lambda_{\log n}) \in [0, 1]^{\log n}$ . If every PRG in the family  $\text{INW}(\vec{\lambda})$  fools width-3 standard-order ROBPs with error 0.99, then  $\lambda_j \leq 1/n^{\Omega(1)}$  for  $\Omega(\log n)$  values of  $j$ , and moreover every PRG in the family  $\text{INW}(\vec{\lambda})$  has seed length  $\Omega(\log^2 n)$ .*

Theorem 4 can be interpreted as saying that if one wishes to use the INW template to construct a PRG that fools width-3 standard-order ROBPs with seed length  $o(\log^2 n)$ , then the proof of correctness would have to exploit some property of the specific graphs  $H_1, \dots, H_{\log n}$  beyond their spectral expansion. It is not clear what property would be helpful.

## 1.3 A possible way forward: Unpredictability and Yao’s XOR Lemma

In this paper, we propose a new approach for constructing a near-optimal PRG fooling constant-width standard-order ROBPs. The approach is based on the classic notion of *unpredictability*. Specialized to the setting of standard-order ROBPs, unpredictability can be defined as follows.

► **Definition 5** (Unpredictability). *Let  $X$  be a distribution over  $\{0, 1\}^n$ . We say that  $X$  is  $\delta$ -unpredictable for width- $w$  standard-order ROBPs if, for every  $i \in [n]$  and every width- $w$  length- $(i-1)$  standard-order ROB  $f: \{0, 1\}^{i-1} \rightarrow \{0, 1\}$ , we have*

$$\Pr[f(X_1, X_2, \dots, X_{i-1}) = X_i] \leq \frac{1}{2} + \delta.$$

*If  $X = G(U_{\mathcal{R}})$  where  $G: \mathcal{R} \rightarrow \{0, 1\}^n$ , then we also describe  $G$  itself as being  $\delta$ -unpredictable for width- $w$  standard-order ROBPs.*

Interestingly, even though  $\text{INW}(1/\text{polylog } n)$  does not *fool* width-3 programs (Theorem 4), it turns out that  $\text{INW}(1/\text{polylog } n)$  is *unpredictable* for constant-width programs. More precisely, Fefferman, Shaltiel, Umans, and Viola showed that every PRG in the family  $\text{INW}(\lambda)$  is  $O(w \cdot \lambda \cdot \log n)$ -unpredictable for width- $w$  standard-order ROBPs [16].<sup>2</sup>

How can we leverage the unpredictability of the INW generator to construct a PRG that actually *fools* constant-width ROBPs? Fefferman, Shaltiel, Umans, and Viola suggested combining the INW generator with a randomness extractor [16]. We propose a different approach, inspired by Yao's *XOR Lemma* in circuit complexity. Yao's XOR Lemma is about the average-case hardness of computing a Boolean function, or more generally the hardness of guessing a Boolean random variable  $Y \in \{0, 1\}$  given some correlated random variable  $X \in \{0, 1\}^n$ . Let  $(X^{(1)}, Y^{(1)}), \dots, (X^{(t)}, Y^{(t)})$  be  $t$  independent copies of  $(X, Y)$ . Roughly speaking, Yao's XOR Lemma says that if it is "somewhat hard" to guess  $Y$  given  $X$ , then it is "very hard" to guess  $Y^{(1)} \oplus \dots \oplus Y^{(t)}$  given  $X^{(1)}, \dots, X^{(t)}$ . Here "somewhat hard" means that all small circuits have, e.g., a constant failure probability, and "very hard" means that all small circuits have a failure probability very close to  $1/2$ .

Maurer and Tessaro observed that Yao's XOR Lemma implies that the bitwise XOR operation amplifies cryptographic unpredictability [34]. In more detail, let  $X$  be a distribution that is  $\delta$ -unpredictable for small circuits for a relatively large value of  $\delta$  such as  $\delta = 0.1$ . Let  $X' = X^{(1)} \oplus \dots \oplus X^{(t)}$ , where  $X^{(1)}, \dots, X^{(t)}$  are independent copies of  $X$ . Yao's XOR Lemma implies that if a small circuit attempts to guess the  $i$ -th bit of  $X'$  given the first  $i - 1$  bits of each copy  $X^{(1)}, \dots, X^{(t)}$ , then its success probability will be very close to  $1/2$ . If the circuit is only given the first  $i - 1$  bits of  $X'$ , then the prediction task becomes even more difficult. Thus,  $X'$  is  $\delta'$ -unpredictable for small circuits where  $\delta' \ll \delta$ .

Intuitively, we expect the same phenomenon to occur in the ROBP setting. If  $X'$  is the bitwise XOR of  $t$  independent copies of some distribution  $X$  that is  $\delta$ -unpredictable for constant-width standard-order ROBPs, then we expect  $X'$  to be  $\delta'$ -unpredictable for such programs, where  $\delta' \approx \delta^t$ . By Yao's so-called "distinguisher-to-predictor lemma," this would imply that  $X'$  actually *fools* such programs with error  $\delta' \cdot n$ . Based on these considerations, we propose the following two-step approach for constructing near-optimal PRGs fooling constant-width standard-order ROBPs.

- **Step 1:** Let  $G \in \text{INW}(1/\text{polylog } n)$  be a PRG with seed length  $s = \tilde{O}(\log n)$ . Prove that if we sample  $t \approx \log n$  seeds  $Z^{(1)}, \dots, Z^{(t)}$  independently and uniformly at random, then the bitwise XOR  $G(Z^{(1)}) \oplus \dots \oplus G(Z^{(t)})$  fools constant-width standard-order ROBPs.
- **Step 2:** Derandomize the proof of Step 1. That is, prove that  $G(Z^{(1)}) \oplus \dots \oplus G(Z^{(t)})$  fools constant-width standard-order ROBPs even if the seeds  $Z^{(1)}, \dots, Z^{(t)}$  are not independent, but rather they are generated in some pseudorandom manner using only  $\tilde{O}(\log n)$  truly random bits.

To be clear, "Step 1" is not in any way trivial, even if we ignore "Step 2." Taking a bitwise XOR of several independent samples from a distribution does not *always* improve its pseudorandomness properties. For example, if  $X$  is  $k$ -wise uniform and uniform over a subspace of  $\mathbb{F}_2^n$  (see Section 2.8), then the bitwise XOR of many copies of  $X$  has the same distribution as a single copy of  $X$ . See also Theorem 10.

<sup>2</sup> One way to prove this statement is to use the notion of *weight* introduced by Braverman, Rao, Raz, and Yehudayoff [8]. For any next-bit predictor  $f: \{0, 1\}^{i-1} \rightarrow \{0, 1\}$ , we can define a test  $g: \{0, 1\}^i \rightarrow \{0, 1\}$  that checks whether  $f$  succeeds:  $g(x_1, \dots, x_i) = f(x_1, \dots, x_{i-1}) \oplus x_i$ . If  $f$  can be computed by a width- $w$  standard-order ROBP, then  $g$  can be computed by a width- $w$  standard-order ROBP with weight two. From here, Braverman, Rao, Raz, and Yehudayoff's analysis shows that  $\text{INW}(\lambda)$  fools  $g$  with error  $O(w \cdot \lambda \cdot \log n)$  [8].

## 1.4 Sums of INW generators fool constant-width ROBPs

One of the main results of this paper is to accomplish “Step 1” described above. To state our results more precisely, let us make the following definition.

► **Definition 6** (XOR of INW generators). *Let  $n, t \in \mathbb{N}$  where  $n$  is a power of two, and let  $\Lambda \in [0, 1]^{t \times \log n}$ . Let  $\text{INW}^{\oplus t}(\Lambda)$  denote the set of all PRGs  $G: \mathcal{R}_1 \times \dots \times \mathcal{R}_t \rightarrow \{0, 1\}^n$  of the form*

$$G(z^{(1)}, \dots, z^{(t)}) = G^{(1)}(z^{(1)}) \oplus \dots \oplus G^{(t)}(z^{(t)}),$$

where  $G^{(i)} \in \text{INW}(\Lambda_i)$  for every  $i \in [t]$ . Here  $\Lambda_i$  denotes the  $i$ -th row of  $\Lambda$ . As a shorthand, we also write  $\text{INW}^{\oplus t}(\lambda)$  if every entry of  $\Lambda$  is equal to  $\lambda$  and  $n$  is clear from context.

We prove the following.

► **Theorem 7** (INW<sup>⊕t</sup> fools constant-width ROBPs). *Let  $n, w, t \in \mathbb{N}$  where  $n$  is a power of two, and let  $\Lambda \in [0, 1]^{t \times \log n}$ . Every PRG in the family  $\text{INW}^{\oplus t}(\Lambda)$  fools width- $w$  length- $n$  standard-order ROBPs with error*

$$n^{\log(w+1)} \cdot \prod_{i=1}^t \sum_{j=1}^{\log n} \Lambda_{i,j}.$$

For example, Theorem 7 implies that there is a value  $t = \Theta(\frac{\log w \cdot \log n + \log(1/\varepsilon)}{\log \log n})$  such that  $\text{INW}^{\oplus t}(1/\log^2 n)$  fools width- $w$  length- $n$  standard-order ROBPs with error  $\varepsilon$ . By plugging in explicit expanders of degree  $\text{poly}(1/\lambda)$ , we get an explicit PRG that fools width- $w$  length- $n$  standard-order ROBPs with error  $\varepsilon$  and seed length

$$O(\log^2 n \cdot \log w + \log n \cdot \log(1/\varepsilon)),$$

re-proving Theorem 3 in the case  $w = O(1)$ . For context, prior to our work, there were already several different known ways to construct PRGs that fool constant-width standard-order ROBPs (and more general models) using a seed of length  $O(\log^2 n)$  [38, 26, 2, 41, 18, 7] or  $\tilde{O}(\log^2 n)$  [17]. Our Theorem 7 adds to this list.

We hope that there is value in having yet another proof of the  $O(\log^2 n)$  bound, but of course the real goal is to construct a PRG with seed length  $o(\log^2 n)$ . As discussed previously, we believe that the most promising approach (“Step 2” in our proposal) is to “derandomize” Theorem 7, i.e., prove that a similar error bound holds even if the  $t$  seeds for the INW generators are chosen in some pseudorandom manner instead of sampling them independently. So far, we have not figured out how to make such an approach work. However, we would like to point out several “success stories” describing cases in which prior researchers managed to solve similar problems:

- There are known derandomized versions of Yao’s XOR Lemma [25, 27, 12].
- The first asymptotically optimal “small-bias” generator, due to Naor and Naor [37], works by plugging several correlated seeds into a weak initial generator and taking the bitwise XOR of the results [37].
- There are several constructions of low-error “weighted pseudorandom generators” and “hitting set generators” for space-bounded computation that work by plugging several correlated seeds into a “moderate-error” initial PRG and then combining the results in some manner [24, 40, 14, 21, 5, 11, 10, 13].

We find these success stories encouraging.

## 1.5 Summing fewer copies of the INW generator does not work

Instead of analyzing correlated seeds, we can also consider a different and more straightforward approach for improving the seed length of our PRG. What happens if we take an XOR of fewer copies of the INW generator while still using relatively mild expander graphs? For example, what if we try  $\text{INW}^{\oplus 2}(1/\text{polylog } n)$ ?

We are inspired to ask this question because of a line of work on the bitwise XOR of small-bias distributions. Recall that by definition, a distribution  $X$  over  $\{0,1\}^n$  is  $\varepsilon$ -biased if, for every nonempty set  $S \subseteq [n]$ , we have  $|\Pr[\oplus_{i \in S} X_i = 0] - \Pr[\oplus_{i \in S} X_i = 1]| \leq \varepsilon$ . A sequence of works has shown that the bitwise XOR of  $d$  independent small-bias distributions fools degree- $d$  polynomials over  $\mathbb{F}_2$ , whereas the XOR of only  $d - 1$  small-bias distributions does not (unless the bias parameter is extremely small) [6, 32, 48, 33]. This demonstrates that a bitwise XOR of a small number of “weak” PRGs can sometimes be quite strong. Indeed, the XOR of just two small-bias distributions has been proposed as a candidate PRG for ROBPs [36, 31].

Unfortunately, we prove that  $\text{INW}^{\oplus 2}(1/\text{polylog } n)$  is not capable of fooling width-3 standard-order ROBPs. More generally, no matter how many or how few copies of the INW generator we XOR together, and no matter how we set the expansion parameters, if the resulting PRG fools width-3 programs, then the seed length is inevitably  $\Omega(\log^2 n)$ :

► **Theorem 8** (Limitations of  $\text{INW}^{\oplus t}$ ). *Let  $n$  be a power of two, let  $t \in \mathbb{N}$ , and let  $\Lambda \in [0, 1]^{t \times \log n}$ . If every PRG in the family  $\text{INW}^{\oplus t}(\Lambda)$  fools width-3 standard-order ROBPs with error 0.99, then every PRG in the family  $\text{INW}^{\oplus t}(\Lambda)$  has seed length  $\Omega(\log^2 n)$ .*

## 1.6 Increasing $t$ vs. decreasing $\lambda$ : Two incomparable PRG paradigms

Motivated by Theorems 7 and 8, we wish to better understand how our new  $\text{INW}^{\oplus t}$  generator compares to the classic INW generator. To be more precise, suppose that for some relatively large value of  $\lambda$ , it turns out that  $\text{INW}(\lambda)$  is too weak to fool some class of functions. To strengthen the PRG, one could try using  $\text{INW}^{\oplus t}(\lambda)$  for some  $t > 1$ , or one could try using  $\text{INW}(\lambda')$  for some  $\lambda' < \lambda$ . Which approach is better?

We prove that these two approaches are in fact *incomparable* in general. There are cases in which one approach is better, and there are cases in which the other approach is better. We state these results precisely in the following two theorems. The first theorem describes a case in which summing multiple generators is the better approach.

► **Theorem 9** (A case where XORing is cheaper than using heavy-duty expanders).

1. For every  $\lambda \in (0, 1)$ , every PRG in  $\text{INW}^{\oplus 2}(\lambda)$  fools quadratic polynomials over  $\mathbb{F}_2$  with error  $O(\sqrt{\lambda})$ .
2. Let  $\vec{\lambda} \in [0, 1]^{\log n}$  where  $n$  is a power of two. If every PRG in  $\text{INW}(\vec{\lambda})$  fools quadratic polynomials over  $\mathbb{F}_2$  with error 0.49, then  $\lambda_j \leq 2^{-\Omega(2^j)}$  for every  $j \geq 4$ , and moreover every PRG in  $\text{INW}(\vec{\lambda})$  has seed length  $\Omega(n)$ .<sup>3</sup>

Item 1 in the theorem above is an immediate consequence of prior work [28, 15, 45, 6, 32, 48]. The main content of the theorem is Item 2. Theorem 9 demonstrates the strength of the “XOR of INW” paradigm. We hope that future researchers can capitalize on the strengths

<sup>3</sup> We remark that the distinguishers we construct are *read-once* quadratic polynomials, and hence they can be computed by width-4 ROBPs that read their input bits in some nonstandard order. This is an example showing that the INW generator does not fool arbitrary-order ROBPs. Tzur previously showed that Nisan’s PRG has this same weakness [46], but to the best of our knowledge, ours is the first such example regarding the INW generator. We thank Adin Gitig for pointing out this gap in the literature.



of the  $\text{INW}^{\oplus t}$  generator to develop better PRGs for ROBPs. Next, let us discuss a case in which summing multiple INW generators is worse than simply using one INW generator with a slightly smaller  $\lambda$  value.

► **Theorem 10** (A case where using heavy-duty expanders is cheaper than XORing). *Let  $n$  be a power of two and let  $\lambda \in (\frac{1000}{n}, \frac{1}{2})$ . There exist  $\lambda' = \Omega(\lambda^2)$  and  $f: \{0, 1\}^n \rightarrow \{0, 1\}$  such that the following hold.*

1. *Every PRG in  $\text{INW}(\lambda')$  fools  $f$  with error 0.01.*
2. *Let  $t \in \mathbb{N}$ . If every PRG in  $\text{INW}^{\oplus t}(\lambda)$  fools  $f$  with error 0.99, then  $t \geq \Omega(\lambda \cdot n / \log n)$ , and moreover every PRG in  $\text{INW}^{\oplus t}(\lambda)$  has seed length  $\Omega(n \cdot \lambda \cdot \log(1/\lambda))$ .*

Theorem 10 is valuable because it helps us to interpret Theorem 7. As a reminder, Theorem 7 says that constant-width ROBPs are fooled by a sum of  $\text{INW}(1/\text{polylog } n)$  generators. In a sense, Theorem 10 shows that Theorem 7 “could have been false” and was not “inevitable.” Indeed, Theorem 10 shows that in general, the fact that a function is fooled by  $\text{INW}(1/\text{poly}(n))$  does not automatically imply that it is fooled by a sum of  $\text{INW}(1/\text{polylog } n)$  generators. Consequently, Theorem 7 reveals a *new weakness* of the constant-width RBP model, above and beyond the well-known weakness of being fooled by  $\text{INW}(1/\text{poly}(n))$ . We hope that future researchers can further exploit the weaknesses of such programs to fool them with a shorter seed.

It might be instructive to compare  $\text{INW}^{\oplus t}(\lambda)$  with a different family of PRGs. Let  $\mathcal{G}_t(\lambda)$  denote the set of generators one can construct by the following recursive paradigm:

$$\begin{aligned} G_0(x) &= x \\ G_{i+1}(x, y) &= (G_i(x), G_i(H_{i+1}^t[x, y])), \end{aligned}$$

where  $H_{i+1}$  is a graph satisfying  $\lambda(H_{i+1}) \leq \lambda$ . This is the same as the definition of the INW generator, except that we use a  $t$ -th power of an expander graph instead of using a generic expander graph. Then  $\mathcal{G}_t(\lambda) \subseteq \text{INW}(\lambda^t)$ , so a theorem saying “Every PRG in  $\mathcal{G}_t(\lambda)$  fools width- $w$  length- $n$  standard-order ROBPs with error  $\lambda^t \cdot w \cdot n$ ” would be true but not interesting. In contrast, Theorem 10 implies that  $\text{INW}^{\oplus t}(\lambda) \not\subseteq \text{INW}(\lambda^t)$ , and we believe that Theorem 7 is saying something new and interesting.

## 1.7 Proof techniques

### 1.7.1 Our proof that $\text{INW}^{\oplus t}(\lambda)$ fools ROBPs, even if $\lambda$ is relatively large

In this section, we give a brief informal overview of our proof of Theorem 7, our positive result on using  $\text{INW}^{\oplus t}(\lambda)$  to fool constant-width standard-order ROBPs. For simplicity’s sake, let us focus on the case that  $\Lambda_{i,j} = \lambda$  for every  $i, j$ , and let us assume that we take the XOR of  $t$  copies of the *same* generator in  $\text{INW}(\lambda)$ . The analysis is based on the following alternative, equivalent description of the resulting PRG. We inductively define a sequence of PRGs  $G_0, G_1, G_2, \dots$ , where  $G_i: \mathcal{R}_i \rightarrow \{0, 1\}^{2^i}$ , as follows.

1. Base case: Let  $\mathcal{R}_0 = \{0, 1\}$  and let  $G_0(x_1, \dots, x_t) = x_1 \oplus \dots \oplus x_t$ .
2. Inductive step: Assume we have already constructed  $G_{j-1}$ . Let  $H_j$  be a  $D_j$ -regular expander graph on the vertex set  $\mathcal{R}_{j-1}$  satisfying  $\lambda(H_j) \leq \lambda$ . Define  $\mathcal{R}_j = \mathcal{R}_{j-1} \times [D_j]$  and

$$G_j((x_1, y_1), \dots, (x_t, y_t)) = (G_{j-1}(x_1, \dots, x_t), G_{j-1}(H_j[x_1, y_1], \dots, H_j[x_t, y_t])). \quad (1)$$

In effect, Equation (1) says that at each stage of the recursion, we use a *tensor product of expander graphs*  $(H_j \otimes \cdots \otimes H_j)$  to recycle the seed  $(x_1, \dots, x_t)$ . Note that tensoring does *not* improve expansion:  $\lambda(H_j \otimes \cdots \otimes H_j) = \lambda(H_j)$ .

Our job is to show that  $G_j$  fools width- $w$  standard-order ROBPs. For simplicity's sake, let us focus on showing that the last bit of the output of  $G_j$  is  $\delta$ -unpredictable for such programs where  $\delta = w^j \cdot (j\lambda)^t$ . Let  $f$  be a width- $w$  program that attempts to predict the last bit of the output of  $G_j$ , given all the previous bits. The idea is to write the transition probability matrix of  $H_j \otimes \cdots \otimes H_j$  in the form

$$(J + E) \otimes (J + E) \otimes \cdots \otimes (J + E).$$

Here  $J$  is the transition matrix of the complete graph with self-loops, and  $E$  is an “error matrix” with operator norm at most  $\lambda$ . After expanding, we get a sum of terms, each of which has the form  $J^{\otimes k} \otimes E^{\otimes(t-k)}$  (after reordering if necessary).

To analyze such a term, we can first consider any fixing of the last  $t - k$  parts of the seed, allowing us to focus on the  $J^{\otimes k}$  factor. This  $J^{\otimes k}$  factor corresponds to running  $\text{INW}(\lambda)^{\oplus k}$  twice, on two independent seeds, and concatenating the results. Since  $f$  is trying to predict the last bit, the first copy of  $\text{INW}(\lambda)^{\oplus k}$  is completely irrelevant; it does not give  $f$  any advantage whatsoever. Meanwhile, by induction, the second copy is  $\varepsilon$ -unpredictable where  $\varepsilon = w^{j-1} \cdot ((j-1) \cdot \lambda)^k$ . Meanwhile, the matrix  $E^{\otimes(t-k)}$  has operator norm at most  $\lambda^{t-k}$ . As it turns out, this has the effect of dampening the inductive advantage bound by a factor of  $w \cdot \lambda^{t-k}$ , so that overall the advantage from the  $J^{\otimes k} \otimes E^{\otimes(t-k)}$  term is at most  $w^j \cdot (j-1)^k \cdot \lambda^t$ . Summing over all terms, we get an overall advantage bound of  $w^j \cdot \lambda^t \cdot \sum_{k=0}^t \binom{t}{k} (j-1)^k = w^j \cdot (j\lambda)^t$ .

In the full proof, in order to optimize parameters and to make the proof as simple as possible, we ultimately do not formally use the notion of unpredictability, but the idea remains the same.

### 1.7.2 Our proof that sums of INW PRGs with seed length $o(\log^2 n)$ do not fool ROBPs

Next, let us discuss our proof that if the entries of  $\Lambda$  are large enough that there exists a generator in  $\text{INW}^{\oplus t}(\Lambda)$  with seed length  $o(\log^2 n)$ , then the entries of  $\Lambda$  are so large that there exists a generator in  $\text{INW}^{\oplus t}(\Lambda)$  that does not fool width-3 programs (Theorem 8). For simplicity's sake, let us focus on the case  $t = 2$ . The proof builds on the works of Brody and Verbin [9], Hoza, Pyne, and Vadhan [23], and Lee and Viola [31]. We construct our  $\text{INW}^{\oplus 2}$  generator using Cayley graphs over the group  $\mathbb{F}_2^n$ , i.e., expander graphs of the form  $H[x, y] = x \oplus G(y)$  where  $G$  is a small-bias generator. By using both Cayley graphs and complete graphs where appropriate, we ensure that the output of our  $\text{INW}^{\oplus 2}$  generator has many substrings of the form

$$(x \oplus x', x \oplus x' \oplus G(y) \oplus G(y')),$$

where  $x, x', G(y), G(y') \in \{0, 1\}^m$  for some  $m = \Theta(\log n)$ .

We instantiate  $G$  using a small-bias generator constructed by Lee and Viola [31]. Their generator outputs *noisy codewords*, i.e., the output is  $a + b$  where  $a$  is a random element of a subspace  $\mathcal{C}^\perp \subseteq \mathbb{F}_2^m$  and  $b$  is a random vector of low Hamming weight. Crucially, Lee and Viola observe that the sum of two noisy codewords is another noisy codeword. Consequently, letting  $z = x \oplus x'$ , the output of our  $\text{INW}^{\oplus 2}$  generator has many substrings of the form

$$(z, \text{something close to } \mathcal{C}^\perp \oplus z).$$



The specific parameters ensure that the description above is nontrivial, i.e., there is some  $v_*$  such that such a substring is never equal to  $(0^m, v_*)$ . Our width-3 distinguisher checks all the relevant regions of its input and accepts if it ever sees the specific substring  $(0^m, v_*)$ . With some tweaks, it turns out that this approach is strong enough to prove Theorem 8.

### 1.7.3 Our proof that INW with small $\lambda$ is incomparable with $\text{INW}^{\oplus t}$ with large $\lambda$

Next, let us briefly explain how we construct an INW generator  $G$  that does not fool quadratic polynomials over  $\mathbb{F}_2$  (Item 2 in Theorem 9). The construction is based on the inner product function  $\text{IP}$ . We construct an expander graph  $H = (V, E)$  such that  $\text{IP}(x, y) = 0$  for every  $(x, y) \in E$ . The construction is essentially  $E = \text{IP}^{-1}(0)$ , except that we must redirect a few edges to ensure that the graph is regular. Using this expander graph  $H$ , we can construct an INW generator whose output is always rejected by  $\text{IP}$ . In contrast,  $\mathbb{E}[\text{IP}] = 1/2 - o(1)$  under the uniform distribution.

Finally, let us briefly discuss the proof of Theorem 10 (a case where  $\text{INW}^{\oplus t}(\lambda)$  performs poorly, but  $\text{INW}(\lambda')$  performs well when  $\lambda'$  is slightly smaller than  $\lambda$ ). We use Cayley graphs to construct an  $\text{INW}^{\oplus t}(\lambda)$  generator that has many  $(2m)$ -bit substrings of the form

$$\left( \bigoplus_{i=1}^t x_i, \bigoplus_{i=1}^t (x_i \oplus G_i(y_i)), \bigoplus_{i=1}^t x'_i, \bigoplus_{i=1}^t (x'_i \oplus G_i(y'_i)) \right),$$

where  $G_1, \dots, G_t$  are small-bias generators, and, crucially, we ensure that  $y_1$  and  $y'_1$  disagree somewhere in their last  $\log(1/\lambda)$  bits.

Our distinguisher begins by canceling out the  $x$ 's to compute the sums  $\bigoplus_{i=1}^t G_i(y_i)$  and  $\bigoplus_{i=1}^t G_i(y'_i)$ . Next, the distinguisher inverts these sum-of-small-bias generators to compute the underlying seeds  $y_1, \dots, y_t, y'_1, \dots, y'_t$ . (We use a probabilistic argument to show that there exist small-bias generators  $G_1, \dots, G_t$  for which this inversion procedure is possible.) Finally, the distinguisher rejects if the last  $\log(1/\lambda)$  bits of  $y_1$  agree with the last  $\log(1/\lambda)$  bits of  $y'_1$ , and otherwise it accepts.

By construction, the distinguisher accepts every output of our  $\text{INW}^{\oplus t}(\lambda)$  generator. In contrast, we show that it has low acceptance probability under any  $\text{INW}(\lambda')$  generator, where  $\lambda'$  is just a little smaller than  $\lambda$ . The proof is based on two ideas.

- In later rounds of the INW recursion, we can use standard techniques based on “communication bottlenecks” to argue that the expander graph in the INW construction does not introduce much error. In particular, after processing  $\bigoplus_i x_i$  and  $\bigoplus_i (x_i \oplus G_i(y_i))$ , the distinguisher only needs to remember the last  $\log(1/\lambda)$  bits of the computed seed  $y_1$ . As a result, the acceptance probability is close to what it would be if we used independent seeds instead of correlated seeds in these later rounds of the INW construction.
- Our remaining task is to bound our distinguisher’s acceptance probability under a concatenation of many independent  $\text{INW}(\lambda')$  generators, each of which outputs  $m$  bits. We are not able to say anything about the distribution of the distinguisher’s computed  $y_1$  seed. However, simply because they are independent and identically distributed, there is a noticeable chance that the distinguisher’s computed  $y_1$  and  $y'_1$  seeds happen to agree in their last  $\log(1/\lambda)$  bits. By doing  $O(1/\lambda)$  independent trials, we ensure that the overall acceptance probability is low.

The proofs of Theorems 9 and 10 are omitted from this extended abstract, but they can be found in the full version of this paper [20].

## 1.8 Additional related work

Assadi and N established a general XOR lemma for streaming algorithms [3] (see also work by Lee, Pyne, and Vadhan [30]). However, their XOR lemma does not imply anything about  $\text{INW}^{\oplus t}$ , because they focus on the scenario in which the streaming algorithm sees  $t$  instances of a problem in sequence, one after another, instead of seeing the bitwise XOR of the instances.

Several prior works have proved limitations on the power of sums of small-bias distributions to fool various types of tests [6, 33, 36, 4, 31]. Our negative result about using  $\text{INW}^{\oplus t}$  to fool ROBPs (Theorem 8) is in a similar spirit, and indeed the proof builds on Lee and Viola's work [31] as mentioned previously.

## 2 Preliminaries

### 2.1 Read-once evaluation programs (ROEPs)

Ultimately, the model of computation we are interested in is the standard-order ROBP model (Definition 2). At intermediate stages of our argument, we will use a slight generalization of the ROBP model called the “read-once evaluation program” (ROEP) model, introduced by Braveman, Rao, Raz, and Yehudayoff [8]. An ROEP is simply an ROBP with fractional output values:

► **Definition 11** (Standard-order ROEP [8]). *A width- $w$  length- $n$  standard-order read-once evaluation program (ROEP) is defined just like a width- $w$  length- $n$  standard-order ROBP (Definition 2), except that the output values  $q_v$  are permitted to be any values in  $[0, 1]$ . Thus, the program computes a function  $f: \{0, 1\}^n \rightarrow [0, 1]$ . We say that a distribution  $X$  over  $\{0, 1\}^n$  fools  $f$  with error  $\varepsilon$  if  $|\mathbb{E}[f(X)] - \mathbb{E}[f]| \leq \varepsilon$ .*

### 2.2 Graphs

For any graph  $H$ , we use  $V(H)$  to denote the vertex set of  $H$ . As discussed in the introduction, we use the notation  $H[x, y]$  to denote the  $y$ -th neighbor of the vertex  $x$  in the graph  $H$ . This is well-defined provided that  $H$  is a *labeled* graph, defined as follows.

► **Definition 12** (Graph labeling). *Let  $H = (V, E)$  be a  $D$ -regular directed multigraph. We say that  $H$  is labeled if for every vertex  $x$ , the outgoing edges are labeled  $1, \dots, D$ . In this case, we write  $H[x, y]$  to denote the vertex reached from  $x$  by traversing the outgoing edge labeled  $y$ . If  $H$  is a  $D$ -regular undirected multigraph, then we identify  $H$  with the symmetric digraph obtained by replacing each undirected edge  $\{x, x'\}$  with two directed edges:  $(x, x')$  and  $(x', x)$ . If we say that  $H$  is “labeled,” we allow the two edges  $(x, x')$  and  $(x', x)$  to have distinct labels.*

We write  $K_R$  to denote the complete graph on  $R$  vertices without self-loops, and we write  $J_R$  to denote the complete graph on  $R$  vertices with self-loops. We write  $J$  or  $J_*$  if the number of vertices is clear from context. Several occurrences of “ $J_*$ ” in a single equation might represent complete graphs on several different numbers of vertices.

If  $H$  is a  $D$ -regular undirected multigraph on  $R$  vertices, its *transition probability matrix* is the matrix  $M \in [0, 1]^{R \times R}$  defined by letting  $M_{u,v} = \frac{e(u,v)}{D}$ , where  $e(u,v)$  denotes the number of edges from  $u$  to  $v$ . We often abuse notation by identifying a graph  $H$  with its transition probability matrix. For example, we use  $J_R$  to denote the  $R \times R$  matrix in which every entry is equal to  $1/R$ .

### 2.3 Spectral expansion

For a matrix  $M \in \mathbb{R}^{R \times R}$ , we use the notation  $\|M\|_{\text{op}}$  to denote the *operator norm* of  $M$ , i.e.,

$$\|M\|_{\text{op}} = \max_{\substack{x \in \mathbb{R}^R \\ \|x\|_2=1}} \|xM\|_2.$$

► **Definition 13** (Expansion parameter). *Let  $H$  be a regular undirected multigraph. The expansion parameter  $\lambda(H)$  is defined as*

$$\lambda(H) = \|H - J\|_{\text{op}}.$$

*Equivalently, one can define  $\lambda(H)$  to be the second-largest eigenvalue of  $H$  in absolute value.*

For example,  $\lambda(J) = 0$ . The complete graph without self-loops also has quite a good expansion parameter:

► **Fact 14** (Expansion parameter of the complete graph without self-loops). *For every  $R \in \mathbb{N}$ , we have  $\lambda(K_R) = 1/(R-1)$ .*

**Proof sketch.** The following  $R$  vectors are linearly independent eigenvectors of  $K_R$ :

- The all-ones vector, which has eigenvalue 1;
- Vectors of the form  $(1, 0, 0, \dots, 0, -1, 0, 0, \dots, 0)$ , each of which has eigenvalue  $-1/(R-1)$ . ◀

*Cayley graphs* are a more interesting class of expanders. The general definition is as follows.

► **Definition 15** (Cayley graph). *Let  $V$  be a group, let  $\mathcal{R}$  be a finite set, and let  $G: \mathcal{R} \rightarrow V$  be a function. The Cayley graph  $\text{Cay}(V, G)$  is a labeled  $|\mathcal{R}|$ -regular directed multigraph on the vertex set  $V$  defined by*

$$\text{Cay}(V, G)[x, y] = x \cdot G(y), \tag{2}$$

*where  $\cdot$  is the group operation.*

We will only use this definition in the special case that  $V = \mathbb{F}_2^n$ , so Equation (2) becomes

$$\text{Cay}(\mathbb{F}_2^n, G)[x, y] = x \oplus G(y).$$

It is well-known that if  $G$  is a small-bias generator, then  $\text{Cay}(\mathbb{F}_2^n, G)$  is an expander. We include the proof for completeness' sake.

► **Lemma 16** (Expanders from small-bias generators). *Let  $n \in \mathbb{N}$  and let  $G: \mathcal{R} \rightarrow \mathbb{F}_2^n$  be a  $\lambda$ -biased generator. Then  $\lambda(\text{Cay}(\mathbb{F}_2^n, G)) \leq \lambda$ .*

**Proof.** Let  $A$  be the  $2^n \times 2^n$  transition probability matrix of  $\text{Cay}(\mathbb{F}_2^n, G)$ . Then

$$A = \frac{1}{|\mathcal{R}|} \sum_{y \in \mathcal{R}} A^{(y)},$$

where  $A_{x, x'}^{(y)} = 1$  if  $x' = x \oplus G(y)$  and 0 otherwise.

The  $2^n$  eigenvectors of  $A^{(y)}$  are the *character functions*  $\chi_a$  (viewed as vectors indexed by  $\mathbb{F}_2^n$ ). Indeed,

$$(\chi_a A^{(y)})_x = \sum_{x'} \chi_a(x') A_{x', x}^{(y)} = \chi_a(x \oplus G(y)) = \chi_a(x) \cdot \chi_a(G(y)),$$

because the only  $x'$  for which  $A^{(y)}(x', x)$  is nonzero is  $x' = x \oplus G(y)$ . Hence  $\chi_a$  is an eigenvector of  $A^{(y)}$  with eigenvalue  $\chi_a(G(y))$ , and by linearity  $\chi_a$  is an eigenvector of  $A$  with eigenvalue

$$\lambda_a = \frac{1}{|\mathcal{R}|} \sum_{y \in \mathcal{R}} \chi_a(G(y)) = \mathbb{E}_{y \in \mathcal{R}} [\chi_a(G(y))].$$

Then we have all  $2^n$  eigenvalues of  $A$ . When  $a = 0$ , one finds  $\lambda_0 = 1$ , and so

$$\lambda(\text{Cay}(\mathbb{F}_2^n, G)) = \max_{a \neq 0} |\mathbb{E}_{y \in \mathcal{R}} [\chi_a(G(y))]|.$$

Since  $G$  is an  $\lambda$ -biased generator,  $|\mathbb{E}_{y \sim \mathcal{R}} [\chi_a(G(y))]| \leq \lambda$ , for all nonzero  $a \in \mathbb{F}_2^n$ . ◀

## 2.4 Lower bound on the degree of expander graphs

To prove our seed length lower bounds, we rely on the following standard fact.

► **Proposition 17** (Expander graph degree lower bound). *Let  $H$  be an undirected  $D$ -regular graph on  $R$  vertices. Then*

$$\lambda(H) \geq \sqrt{\frac{1}{D} \cdot \frac{R-D}{R-1}}.$$

In particular,  $D \geq \min\{1/(2 \cdot \lambda(H)^2), (R+1)/2\}$ , and if  $D = 1$ , then  $\lambda(H) = 1$ .

A proof of Proposition 17 can be found in the full version of this paper [20].

## 2.5 Tensor products

► **Definition 18** (Tensor product of graphs). *Given a pair of labeled graphs  $H_1, H_2$  on  $R_1, R_2$  vertices with degrees  $D_1, D_2$  respectively, define the tensor product  $H_1 \otimes H_2$  to be the  $(D_1 \cdot D_2)$ -regular graph on  $R_1 \cdot R_2$  vertices with neighbor relation  $(H_1 \otimes H_2)[(u_1, u_2), (e_1, e_2)] = (H_1[u_1, e_1], H_2[u_2, e_2])$ .*

► **Proposition 19** (Spectral expansion of a tensor product). *Let  $H_1, H_2$  be undirected regular graphs. Then  $\lambda(H_1 \otimes H_2) = \max(\lambda(H_1), \lambda(H_2))$ .*

We also use the notation  $M \otimes M'$  to denote the tensor product of matrices, aka the Kronecker product.

► **Fact 20** (Operator norm of tensor product). *For any two matrices  $M, M'$ , we have*

$$\|M \otimes M'\|_{\text{op}} = \|M\|_{\text{op}} \cdot \|M'\|_{\text{op}}.$$

## 2.6 Expander mixing lemma

We will use the following weak version of the famous “expander mixing lemma.”

► **Lemma 21** (Expander mixing lemma). *Let  $H = (V, E)$  be a regular undirected multigraph. Let  $f, g: V \rightarrow \{0, 1\}$ . Sample a uniform random vertex  $X$ , then sample a uniform random neighbor  $Y$  of  $X$ . Then*

$$|\mathbb{E}[f(X) \cdot g(Y)] - \mathbb{E}[f] \cdot \mathbb{E}[g]| \leq \lambda(H).$$

For a proof, see, e.g., Vadhan’s pseudorandomness survey [47].

## 2.7 INW generators

We now present a more precise definition of the INW generator, in case the informal definition in the introduction was not sufficiently clear.

► **Definition 22** (Permissible families of graphs). *Let  $n$  be a power of two, let  $D_1, \dots, D_{\log n} \in \mathbb{N}$ , let  $H_i$  be a labeled  $D_i$ -regular undirected multigraph for every  $i \in [\log n]$ , and let  $\vec{H} = (H_1, \dots, H_{\log n})$ . We say that  $\vec{H}$  is permissible if  $V(H_1) = [2]$  and  $V(H_{i+1}) = V(H_i) \times [D_i]$  for every  $i \in [\log n - 1]$ , where  $V(H)$  denotes the vertex set of  $H$ .*

*More generally, suppose  $\mathcal{H}$  is a  $t \times \log n$  matrix of labeled regular undirected multigraphs:*

$$\mathcal{H} = \begin{bmatrix} H_{1,1} & \dots & H_{1,\log n} \\ \vdots & \ddots & \vdots \\ H_{t,1} & \dots & H_{t,\log n} \end{bmatrix}.$$

*We say that  $\mathcal{H}$  is permissible if each row is permissible.*

► **Definition 23** (INW generators). *Let  $\vec{H} = (H_1, \dots, H_{\log n})$  be a permissible family of labeled regular undirected multigraphs. We define  $\text{INW}_{\vec{H}}: V(H_{\log n}) \rightarrow \{0, 1\}^n$  recursively by the formulas*

$$\begin{aligned} \text{INW}_{()}(x) &= x \\ \text{INW}_{(H_1, \dots, H_j)}(x, y) &= (\text{INW}_{(H_1, \dots, H_{j-1})}(x), \text{INW}_{(H_1, \dots, H_{j-1})}(H_j[x, y])). \end{aligned}$$

*More generally, let  $\mathcal{H}$  be a  $t \times \log n$  matrix of labeled regular undirected multigraphs, say*

$$\mathcal{H} = \begin{bmatrix} H_{1,1} & \dots & H_{1,\log n} \\ \vdots & \ddots & \vdots \\ H_{t,1} & \dots & H_{t,\log n} \end{bmatrix},$$

*and assume that  $\mathcal{H}$  is permissible. We define  $\text{INW}_{\mathcal{H}}^{\oplus t}: V(H_{1,\log n}) \times \dots \times V(H_{t,\log n}) \rightarrow \{0, 1\}^n$  by the formula*

$$\text{INW}_{\mathcal{H}}^{\oplus t}(x_1, \dots, x_t) = \text{INW}_{\mathcal{H}_1}(x_1) \oplus \dots \oplus \text{INW}_{\mathcal{H}_t}(x_t),$$

*where  $\mathcal{H}_1, \dots, \mathcal{H}_t$  are the rows of  $\mathcal{H}$ .*

*For a vector  $\vec{\lambda} \in [0, 1]^{\log n}$ , we define*

$$\text{INW}(\vec{\lambda}) = \left\{ \text{INW}_{\vec{H}}: \vec{H} \text{ is a } 1 \times \log n \text{ permissible family and } \lambda(H_j) \leq \lambda_j \text{ for all } j \right\}.$$

*When  $n$  is clear from context, we use  $\text{INW}(\lambda)$  as a shorthand for the case that  $\lambda_j = \lambda$  for every  $j$ . Similarly, for a matrix  $\Lambda \in [0, 1]^{t \times \log n}$ , we define*

$$\text{INW}^{\oplus t}(\Lambda) = \{ \text{INW}_{\mathcal{H}}: \mathcal{H} \text{ is a } t \times \log n \text{ permissible family and } \lambda(H_{i,j}) \leq \Lambda_{i,j} \text{ for all } i, j \}.$$

*When  $n$  is clear from context, we use  $\text{INW}^{\oplus t}(\lambda)$  as a shorthand for the case that  $\Lambda_{i,j} = \lambda$  for every  $i, j$ .*

## 2.8 Binary linear code

In this section we briefly review the basic concepts of the coding theory.

► **Definition 24** (Binary Linear Code). A binary linear code  $\mathcal{C}$  of block length  $m$  is a subspace of  $\mathbb{F}_2^m$ , where  $\mathbb{F}_2$  is the field with two elements. The minimum distance  $d$  of  $\mathcal{C}$  is the smallest Hamming weight among all nonzero codewords in  $\mathcal{C}$ .

► **Definition 25** (Binary Entropy Function). For  $0 \leq \delta \leq 1$ , the binary entropy function is defined as

$$H(\delta) = -\delta \log_2(\delta) - (1 - \delta) \log_2(1 - \delta),$$

with the convention that  $0 \log_2 0 = 0$ .

► **Theorem 26** (GV bound for binary linear codes). For every  $m, k \in \mathbb{N}$  such that  $k \leq m/2$ , there exists a binary linear code of block length  $m$  with minimum distance  $k+1$  and dimension at least  $\lfloor (1 - H(k/m)) \cdot m \rfloor$ .

► **Corollary 27**. For every  $\delta \in (0, 1/2)$ , there is a subspace  $\mathcal{C}^\perp \subseteq \mathbb{F}_2^m$  of dimension at most  $\lceil H(\delta) \cdot m \rceil$  such that the uniform distribution over  $\mathcal{C}^\perp$  is  $(\delta m)$ -wise uniform distribution, which means every  $\lfloor \delta m \rfloor$  bits of this distribution are uniform over  $\{0, 1\}^{\lfloor \delta m \rfloor}$ .

The proofs of Theorem 26 and Corollary 27 can be found in the full version of this paper [20].

### 3 Proof that sums of INW generators fool constant-width ROBPs

In this section, we present the proof of Theorem 7, which says that  $\text{INW}^{\oplus t}(\Lambda)$  fools width- $w$  programs with error  $n^{\log(w+1)} \cdot \prod_{i=1}^t \sum_{j=1}^{\log n} \Lambda_{i,j}$ . The proof is based on the notion of a *robust PRG*. Roughly speaking, a robust PRG is a multi-seed PRG that still works even if some seeds are fixed to arbitrary values, with an error bound that depends on the number of random seeds. The precise definition is as follows.

► **Definition 28** (Robust PRG). Let  $\mathcal{R}_1, \dots, \mathcal{R}_t$  be finite sets, let  $G: \mathcal{R}_1 \times \dots \times \mathcal{R}_t \rightarrow \{0, 1\}^n$ , let  $W \geq 0$ , let  $\vec{\varepsilon} = (\varepsilon_1, \dots, \varepsilon_t) \in (0, 1)^t$ , and let  $f: \{0, 1\}^n \rightarrow \mathbb{R}$ . We say that  $G$  robustly  $(W, \vec{\varepsilon})$ -fools  $f$  if the following holds. For every  $A \subseteq [t]$  and every  $x_{[t] \setminus A} \in \prod_{i \in [t] \setminus A} \mathcal{R}_i$ , if we sample  $x_A \in \prod_{i \in A} \mathcal{R}_i$  uniformly at random, then

$$\left| \mathbb{E}_{x_A} [f(G(x_1, \dots, x_t))] - \mathbb{E}[f] \right| \leq W \cdot \prod_{i \in A} \varepsilon_i.$$

For example, if  $\mathcal{R}_1 = \dots = \mathcal{R}_t = \{0, 1\}$  and  $G(x_1, \dots, x_t) = x_1 \oplus \dots \oplus x_t$ , then  $G$  robustly  $(1, \vec{0})$ -fools every function  $f: \{0, 1\} \rightarrow [0, 1]$ . This example will serve as the base case of our analysis of  $\text{INW}^{\oplus t}$ . The inductive step will be based on the following lemma, which shows how to double the output length of a robust PRG fooling ROEPs.

► **Lemma 29** (Inductive step in the analysis of  $\text{INW}^{\oplus t}$ ). Let  $n, w \in \mathbb{N}$  where  $n$  is even. Let  $\mathcal{R}_1, \dots, \mathcal{R}_t$  be finite sets. Let  $G: \mathcal{R}_1 \times \dots \times \mathcal{R}_t \rightarrow \{0, 1\}^{n/2}$ . Let  $W \geq 0$  and  $\vec{\varepsilon} \in (0, 1)^t$ , and assume that  $G$  robustly  $(W, \vec{\varepsilon})$ -fools all width- $w$  length- $(n/2)$  standard-order ROEPs. For each  $i \in [t]$ , let  $H^{(i)}$  be a  $D_i$ -regular multigraph on the vertex set  $\mathcal{R}_i$ . Let  $\vec{\lambda} = (\lambda(H^{(1)}), \dots, \lambda(H^{(t)}))$ . Define  $G': (\mathcal{R}_1 \times [D_1]) \times \dots \times (\mathcal{R}_t \times [D_t]) \rightarrow \{0, 1\}^n$  by the formula

$$G'((x_1, y_1), \dots, (x_t, y_t)) = (G(x_1, \dots, x_t), G(H^{(1)}[x_1, y_1], \dots, H^{(t)}[x_t, y_t])).$$

Then  $G'$  robustly  $(W \cdot (w + 1), \vec{\varepsilon} + \vec{\lambda})$ -fools all width- $w$  length- $n$  standard-order ROEPs.



**Proof.** Let  $f: \{0, 1\}^n \rightarrow [0, 1]$  be a width- $w$  length- $n$  standard-order ROEP. Fix any set  $A \subseteq [t]$ . For every  $i \in [t] \setminus A$ , let  $x_i, x'_i \in \mathcal{R}_i$  be arbitrary fixed values. We also write  $X_i = x_i$  and  $X'_i = x'_i$  for every  $i \in [t] \setminus A$ . Meanwhile, for every  $i \in A$ , sample  $X_i \in \mathcal{R}_i$  and  $Y_i \in [D_i]$  independently and uniformly at random, and let  $X'_i = H^{(i)}[X_i, Y_i]$ . Let  $X = (X_1, \dots, X_t)$  and  $X' = (X'_1, \dots, X'_t)$ . Our goal is to show that  $(G(X), G(X'))$  fools  $f$  with error  $W \cdot (w + 1) \cdot \prod_{i \in R} (\varepsilon_i + \lambda_i)$ .

For each vertex  $u$  in the middle layer of  $f$ , define  $f_{\rightarrow u}: \{0, 1\}^{n/2} \rightarrow \{0, 1\}$  by letting  $f_{\rightarrow u}(z)$  indicate whether  $f$  reaches  $u$  when it reads  $z$ . Furthermore, define  $f_{u \rightarrow}: \{0, 1\}^{n/2} \rightarrow [0, 1]$  by letting  $f_{u \rightarrow}(z)$  be the label of the vertex reached in the final layer if we start at  $u$  and read  $z$ . Let  $\mu_{u \rightarrow} = \mathbb{E}[f_{u \rightarrow}]$  and  $\bar{f}_{u \rightarrow} = f_{u \rightarrow} - \mu_{u \rightarrow}$ . Then for any  $z, z' \in \{0, 1\}^{n/2}$ , we have

$$f(z, z') = \sum_{u \in [w]} f_{\rightarrow u}(z) \cdot f_{u \rightarrow}(z') = \left( \sum_{u \in [w]} f_{\rightarrow u}(z) \cdot \bar{f}_{u \rightarrow}(z') \right) + \left( \sum_{u \in [w]} f_{\rightarrow u}(z) \cdot \mu_{u \rightarrow} \right).$$

The second sum above is computable by a width- $w$  standard-order ROEP  $f_{\text{left}}: \{0, 1\}^{\frac{n}{2}} \rightarrow [0, 1]$  that ignores the second half of its input. By assumption,  $(G(X), G(X'))$  fools  $f_{\text{left}}$  with error  $W \cdot \prod_{i \in R} \varepsilon_i$ .

Now consider a single term in the first sum,  $f_{\rightarrow u}(z) \cdot \bar{f}_{u \rightarrow}(z')$ . Under the uniform distribution, we have

$$\mathbb{E}[f_{\rightarrow u} \cdot \bar{f}_{u \rightarrow}] = \mathbb{E}[f_{\rightarrow u}] \cdot \mathbb{E}[\bar{f}_{u \rightarrow}] = \mathbb{E}[f_{\rightarrow u}] \cdot (\mu_{u \rightarrow} - \mu_{u \rightarrow}) = 0.$$

Now we analyze the expectation under the pseudorandom distribution  $(G(X), G(X'))$ . We begin by writing the expectation as a sum. For convenience, for any set  $S \subseteq [t]$ , let us use the notation  $\mathcal{R}_S$  to denote the Cartesian product  $\prod_{i \in S} \mathcal{R}_i$ . Furthermore, let us identify the graph  $H^{(i)}$  with its transition probability matrix. Then we have

$$\begin{aligned} & \mathbb{E}[f_{\rightarrow u}(G(X)) \cdot \bar{f}_{u \rightarrow}(G(X'))] \\ &= \sum_{x_A, x'_A \in \mathcal{R}_A} \Pr[X = x \text{ and } X' = x'] \cdot f_{\rightarrow u}(G(x)) \cdot \bar{f}_{u \rightarrow}(G(x')) \\ &= \sum_{x_A, x'_A \in \mathcal{R}_A} \left( \prod_{i \in A} \frac{H_{x_i, x'_i}}{|\mathcal{R}_i|} \right) \cdot f_{\rightarrow u}(G(x)) \cdot \bar{f}_{u \rightarrow}(G(x')), \end{aligned}$$

where the notation  $x$  denotes the vector  $x = (x_1, \dots, x_t)$ , and similarly  $x' = (x'_1, \dots, x'_t)$ . Next, we use the decomposition  $H^{(i)} = J_{|\mathcal{R}_i|} + E^{(i)}$ , where  $J_{|\mathcal{R}_i|}$  has  $1/|\mathcal{R}_i|$  in every entry and  $E^{(i)}$  is some matrix with operator norm  $\lambda_i$ . Applying this decomposition entrywise, we get

$$\begin{aligned} & \mathbb{E}[f_{\rightarrow u}(G(X)) \cdot \bar{f}_{u \rightarrow}(G(X'))] \\ &= \sum_{x_A, x'_A \in \mathcal{R}_A} \left( \prod_{i \in A} \left( \frac{1}{|\mathcal{R}_i|^2} + \frac{E_{x_i, x'_i}^{(i)}}{|\mathcal{R}_i|} \right) \right) \cdot f_{\rightarrow u}(G(x)) \cdot \bar{f}_{u \rightarrow}(G(x')) \\ &= \sum_{A=S \sqcup T} \sum_{x_T, x'_T \in \mathcal{R}_T} \left( \prod_{i \in T} \frac{E_{x_i, x'_i}^{(i)}}{|\mathcal{R}_i|} \right) \mathbb{E}_{x_S, x'_S \in \mathcal{R}_S} [f_{\rightarrow u}(G(x)) \cdot \bar{f}_{u \rightarrow}(G(x'))], \end{aligned}$$

where the outer sum is over all partitions of  $A$  into two disjoint sets,  $S$  and  $T$ . The product  $\prod_{i \in T} E_{x_i, x'_i}^{(i)}$  is exactly the  $(x_T, x'_T)$  entry in the tensor product matrix  $\bigotimes_{i \in T} E^{(i)}$ . Meanwhile, the expectation

$$\mathbb{E}_{x_S, x'_S \in \mathcal{R}_S} [f_{\rightarrow u}(G(x)) \cdot \bar{f}_{u \rightarrow}(G(x'))]$$

splits as a product of expectations. Thus, we get

$$\begin{aligned} & \mathbb{E}[f_{\rightarrow u}(G(X)) \cdot \bar{f}_{u \rightarrow}(G(X'))] \\ &= \sum_{A=S \sqcup T} \frac{1}{|\mathcal{R}_T|} \cdot \sum_{x_T, x'_T \in \mathcal{R}_T} \left( \bigotimes_{i \in T} E^{(i)} \right)_{x_T, x'_T} \cdot \mathbb{E}_{x_S \in \mathcal{R}_S} [f_{\rightarrow u}(G(x))] \cdot \mathbb{E}_{x'_S \in \mathcal{R}_S} [\bar{f}_{u \rightarrow}(G(x'))]. \end{aligned}$$

We can think of the first expectation,  $\mathbb{E}_{x_S \in \mathcal{R}_S} [f_{\rightarrow u}(G(x))]$ , as a single entry  $a_{x_T}$  in a long vector  $a \in \mathbb{R}^{\mathcal{R}_T}$ . Similarly, we think of the second expectation,  $\mathbb{E}_{x'_S \in \mathcal{R}_S} [\bar{f}_{u \rightarrow}(G(x'))]$ , as a single entry  $b_{x'_T}$  in a long vector  $b \in \mathbb{R}^{\mathcal{R}_T}$ . In this way, the inner sum above becomes a vector-matrix-vector product:

$$\begin{aligned} |\mathbb{E}[f_{\rightarrow u}(G(X)) \cdot \bar{f}_{u \rightarrow}(G(X'))]| &= \left| \sum_{A=S \sqcup T} \frac{1}{|\mathcal{R}_T|} \cdot a^\top \cdot E^{\otimes T} \cdot b \right| \\ &\leq \sum_{A=S \sqcup T} \frac{1}{|\mathcal{R}_T|} \cdot \|a\|_2 \cdot \left\| \bigotimes_{i \in T} E^{(i)} \right\|_{\text{op}} \cdot \|b\|_2 \\ &\leq \sum_{A=S \sqcup T} \|a\|_\infty \cdot \|b\|_\infty \cdot \prod_{i \in T} \|E^{(i)}\|_{\text{op}} \\ &= \sum_{A=S \sqcup T} \|a\|_\infty \cdot \|b\|_\infty \cdot \prod_{i \in T} \lambda_i. \end{aligned}$$

The function  $f_{\rightarrow u}$  is  $\{0, 1\}$ -valued, so  $\|a\|_\infty \leq 1$ . Meanwhile, for any fixing of  $x'_T \in \mathcal{R}_T$ , the entry  $b_{x'_T}$  is precisely the error when we sample  $x'_S \in \mathcal{R}_S$  uniformly at random and try to use  $G(x')$  to fool  $f_{u \rightarrow}$ , a width- $w$  length- $(n/2)$  standard-order ROEP. By assumption,  $\|b\|_\infty \leq W \cdot \prod_{i \in S} \varepsilon_i$ . Therefore,

$$|\mathbb{E}[f_{\rightarrow u}(G(X)) \cdot \bar{f}_{u \rightarrow}(G(X'))]| \leq W \cdot \sum_{A=S \sqcup T} \left( \prod_{i \in S} \varepsilon_i \right) \cdot \left( \prod_{i \in T} \lambda_i \right) = A \cdot \prod_{i \in A} (\varepsilon_i + \lambda_i).$$

Consequently, summing up all the errors, we get

$$|\mathbb{E}[f(G(X), G(X'))] - \mathbb{E}[f]| \leq \left( \sum_{u \in [w]} W \cdot \prod_{i \in A} (\varepsilon_i + \lambda_i) \right) + W \cdot \prod_{i \in A} \varepsilon_i \leq W(w+1) \cdot \prod_{i \in A} (\varepsilon_i + \lambda_i). \blacktriangleleft$$

**Proof of Theorem 7.** Let  $G_{\log n}$  be any PRG in the family  $\text{INW}^{\oplus t}(\Lambda)$ . We will prove by induction on  $n$  that  $G_{\log n}$  robustly  $(W, \vec{\varepsilon})$ -fools width- $w$  length- $n$  standard-order ROEPs, where  $W = (w+1)^{\log n}$  and  $\varepsilon_i = \sum_{j=1}^{\log n} \Lambda_{i,j}$ . For the base case, if  $n = 1$ , then there is exactly one PRG in the family  $\text{INW}^{\oplus t}(\Lambda)$ , namely  $G_0: \{0, 1\}^t \rightarrow \{0, 1\}$  is given by  $G_0(x) = x_1 \oplus \dots \oplus x_t$ . This PRG indeed robustly  $(1, \vec{0})$ -fools all functions  $f: \{0, 1\} \rightarrow [0, 1]$ . Now, for the inductive step, suppose  $n > 1$ . By definition of  $\text{INW}^{\oplus t}(\Lambda)$ , the PRG  $G_{\log n}$  has the form  $G_{\log n}(a_1, \dots, a_t) = G_{\log n}^{(1)}(a_1) \oplus \dots \oplus G_{\log n}^{(t)}(a_t)$ , where  $G_{\log n}^{(i)} \in \text{INW}(\Lambda_i)$  for every  $i$ . By definition of  $\text{INW}(\Lambda_i)$ , the seed  $a_i$  is a pair  $(x_i, y_i)$ , and the generator  $G_{\log n}^{(i)}$  has the form

$$G_{\log n}^{(i)}(x_i, y_i) = (G_{\log n-1}^{(i)}(x_i), G_{\log n-1}^{(i)}(H^{(i)}[x_i, y_i]))$$

for some  $G_{\log n-1}^{(i)} \in \text{INW}((\Lambda_{i,1}, \dots, \Lambda_{i, \log n-1}))$  and some  $\Lambda_{i, \log n}$ -spectral expander  $H^{(i)}$ . Define a PRG  $G_{\log n-1}$  by the rule

$$G_{\log n-1}(x_1, \dots, x_t) = G_{\log n-1}^{(1)}(x_1) \oplus \dots \oplus G_{\log n-1}^{(t)}(x_t).$$

Then  $G_{\log n-1} \in \text{INW}^{\oplus t}(\Lambda')$ , where  $\Lambda'$  consists of all but the last column of  $\Lambda$ . By induction,  $G_{\log n-1}$  robustly  $((w+1)^{\log n-1}, \bar{\alpha})$ -fools width- $w$  length- $n$  standard-order ROEPs, where  $\alpha_i = \sum_{j=1}^{\log n-1} \Lambda_{i,j}$ . Working through the definitions, we see that the PRG  $G_{\log n}$  can be written as

$$G_{\log n}((x_1, y_1), \dots, (x_t, y_t)) = (G_{\log n-1}(x_1, \dots, x_t), G_{\log n-1}(H^{(1)}[x_1, y_1], \dots, H^{(t)}[x_t, y_t])).$$

Applying Lemma 29 completes the inductive step.

Finally, since  $G_{\log n}$  robustly  $(W, \bar{\varepsilon})$ -fools width- $w$  length- $n$  standard-order ROEPs, it follows that  $G_{\log n}$  fools width- $w$  length- $n$  standard-order ROBPs with error  $\varepsilon$ , where

$$\varepsilon = W \cdot \prod_{i=1}^t \varepsilon_i = n^{\log(w+1)} \cdot \prod_{i=1}^t \sum_{j=1}^{\log n} \Lambda_{i,j}. \quad \blacktriangleleft$$

## 4 Seed length lower bound for fooling ROBPs

In this section, we prove Theorem 8, which states that no matter how many (or how few) copies of the INW generator we XOR together, and regardless of how we set the expansion parameters, if the resulting PRG fools width-3 programs, then the seed length is inevitably  $\Omega(\log^2 n)$ .

### 4.1 Sums of small-bias distributions

We begin by analyzing a family of a small-bias distributions introduced by Lee and Viola [31]. This construction guarantees that summing independent samples from these distributions does not substantially increase the overall number of distinct strings.

► **Definition 30** (Sum of sets). For  $S, T \subseteq \{0, 1\}^m$ ,  $S + T = \{s \oplus t \mid s \in S, t \in T\}$ .

► **Lemma 31** (Small-bias distributions with a small sum set). For every  $m, t \in \mathbb{N}$  and every  $\varepsilon_1, \dots, \varepsilon_t \in (0, 1]$  such that  $\lceil \ln(1/\varepsilon_1) \rceil + \dots + \lceil \ln(1/\varepsilon_t) \rceil < \frac{1}{625} \cdot m$ , there exist distributions  $D_1, \dots, D_t$  over  $\{0, 1\}^m$  such that  $D_i$  is  $\varepsilon_i$ -biased for every  $i$ , and

$$|\text{Supp}(D_1) + \dots + \text{Supp}(D_t)| < 2 \cdot 2^{m/2}.$$

Furthermore, the probability mass function of each  $D_i$  only takes on rational values.

**Proof.** For each  $i \in [t]$ , we construct  $D_i$  by taking the bitwise XOR  $X \oplus Y_i$ , where  $X$  is distributed uniformly over  $\mathcal{C}^\perp$  which is  $(\frac{m}{25})$ -wise uniform constructed by Corollary 27, and  $Y_i$  is an independent “noise vector” constructed as follows. Repeat the following process  $r_i$  times independently, where  $r_i = \lceil 25 \ln(1/\varepsilon_i) \rceil$ : choose a position uniformly at random from  $[m]$ , and set it to a uniform bit. The remaining bits of  $Y_i$  are zero. Note that  $X$  is uniform over a subspace of  $\mathbb{F}_2^m$  and the probability of getting a particular noise vector  $y$  is a multiple of  $(1/m)^{r_i}$ , which is a rational number.

First we prove that each  $D_i$  is  $\varepsilon_i$ -biased. For any character function  $\chi_S$  with  $|S| < \frac{m}{25}$ , since  $X$  and  $Y_i$  are independent and  $X$  is  $(\frac{m}{25})$ -wise uniform,  $\chi_S$  is perfectly fooled by  $D_i$ . Next, consider any character function  $\chi_S$ , where  $|S| \geq \frac{m}{25}$ . In this case, the bias is nonzero only if none of the elements in  $S$  are selected by the random noise  $Y_i$ . So the bias is at most

$$(1 - |S|/m)^{r_i} \leq \exp(-r_i \cdot |S|/m) \leq \varepsilon_i.$$

Thus, we have shown that each  $D_i$  is  $\varepsilon_i$ -biased.

By the closure property of linear subspaces,

$$\begin{aligned} |\text{Supp}(D_1) + \cdots + \text{Supp}(D_t)| &= |\text{Supp}(X) + \text{Supp}(Y_1) + \cdots + \text{Supp}(X) + \text{Supp}(Y_t)| \\ &= |\mathcal{C}^\perp + \cdots + \mathcal{C}^\perp + \text{Supp}(Y_1) + \cdots + \text{Supp}(Y_t)| \\ &= |\mathcal{C}^\perp + \text{Supp}(Y_1) + \cdots + \text{Supp}(Y_t)|. \end{aligned}$$

The set  $\mathcal{C}^\perp + \text{Supp}(Y_1) + \cdots + \text{Supp}(Y_t)$  is precisely the set of all binary strings within Hamming distance at most  $r_1 + \cdots + r_t \leq \lceil 25 \ln(1/\varepsilon_1) \rceil + \cdots + \lceil 25 \ln(1/\varepsilon_t) \rceil < \frac{m}{25}$  from  $\mathcal{C}^\perp$ . Therefore,

$$\begin{aligned} |\mathcal{X} + \text{Supp}(Y_1) + \cdots + \text{Supp}(Y_t)| &\leq |\mathcal{C}^\perp| \cdot \binom{m}{\leq m/25} \leq 2^{\lceil H(\frac{1}{25}) \cdot m \rceil} \cdot 2^{H(\frac{1}{25}) \cdot m} \\ &< 2 \cdot 2^{m/2}. \end{aligned} \quad \blacktriangleleft$$

► **Remark 32** (The benefit of noisy codewords). In the proof of Lemma 31, we use small-bias distributions based on noisy codewords [31]. A more straightforward approach for minimizing  $|\text{Supp}(D_1) + \cdots + \text{Supp}(D_t)|$  would be to simply make  $|\text{Supp}(D_i)|$  as small as possible and then use the trivial bound

$$|\text{Supp}(D_1) + \cdots + \text{Supp}(D_t)| \leq \prod_{i=1}^t |\text{Supp}(D_i)|.$$

This approach is too weak to prove Lemma 31. Indeed, every small-bias distribution  $D_i$  satisfies  $|\text{Supp}(D_i)| \geq \Omega(m)$  [1], so we inevitably have  $\prod_{i=1}^t |\text{Supp}(D_i)| \geq \Omega(m)^t$ , so the bound is trivial when  $t \geq 1.01 \cdot m / \log m$ . In contrast, Lemma 31 is meaningful even when  $t = \Theta(m)$ .

## 4.2 Constructing an $\text{INW}^{\oplus t}$ generator that does not fool ROBPs

Recall that the seed length of  $\text{INW}_{\mathcal{H}}^{\oplus t}$  is  $\sum_{i \in [t], j \in [\log n]} \log(\deg(H_{i,j}))$ . Our goal is to show that if every PRG in  $\text{INW}^{\oplus t}(\Lambda)$  fools width-3 programs, then every PRG in  $\text{INW}^{\oplus t}(\Lambda)$  has seed length  $\Omega(\log^2 n)$ . Indeed, we will show that for each PRG in  $\text{INW}^{\oplus t}(\Lambda)$ , the seed length grows by  $\Omega(\log n)$  in each of  $\Omega(\log n)$  of the rounds of the INW recursion. The lemma below makes this precise, formulated in the contrapositive.

► **Lemma 33** (One-round seed length lower bound for fooling width-3 programs). *Let  $n$  be a sufficiently large power of two, let  $t \in \mathbb{N}$ , let  $\Lambda \in [0, 1]^{t \times \log n}$ , and let  $j_* \in [\log \log n, \frac{1}{4} \log n]$ . Assume that there exists a  $t \times \log n$  permissible family of labeled regular undirected multigraphs  $\mathcal{H} = \{H_{i,j}\}$  such that  $\lambda(H_{i,j}) \leq \Lambda_{i,j}$  for every  $i, j$  and  $\log(\deg(H_{1,j_*})) + \cdots + \log(\deg(H_{t,j_*})) < \frac{1}{20000} \cdot \log n$ . Then there exists another  $t \times \log n$  permissible family of labeled regular undirected multigraphs  $\mathcal{H}' = \{H'_{i,j}\}$  such that  $\lambda(H'_{i,j}) \leq \Lambda_{i,j}$  for every  $i, j$ , along with a length- $n$ , width-3, standard-order ROBP  $B$  such that  $\Pr[B(U_{\{0,1\}^n}) = 1] \geq 1 - \exp(-n^{1/4})$ , but  $B(\text{INW}_{\mathcal{H}'}^{\oplus t}(x)) = 0$  for every seed  $x$ .*

**Proof.** We first partition  $[t]$  based on whether  $\deg(H_{i,j_*}) \geq 1/(2\Lambda_{i,j_*}^2)$ . Let  $T_1 \subseteq [t]$  be the indices such that  $\deg(H_{i,j_*}) \geq 1/(2\Lambda_{i,j_*}^2)$  and let  $T_2 = [t] \setminus T_1$ , which means for all  $i \in T_2$ ,  $\deg(H_{i,j_*}) \geq (|H_{i,j_*}| + 1)/2$  by Proposition 17. Without loss of generality, we assume that  $T_1 = [t_1]$  and  $T_2 = [t] \setminus [t_1]$ . Note that if  $\Lambda_{i,j_*} = 0$  for some  $i \in [t]$ , then  $i \in T_2$ .

Let  $M$  be a power of two satisfying  $n^{1/8} \leq M < n^{1/4}$ , and let  $m = \log M$ . Now we can construct a new family of graphs  $\mathcal{H}'$ . For indices  $i \in T_1$ , we use Lemma 31 to construct a family of distributions  $D_i, i \in T_1$  over  $\{0, 1\}^m$  such that each  $D_i$  is  $\Lambda_{i,j_*}$ -biased. By Proposition 17, we know that for any graph  $H_{i,j_*}$  such that  $\deg(H_{i,j_*}) = 1$ , we have  $\lambda(H_{i,j_*}) = \Lambda_{i,j_*} = 1$ . Thus,

$$\begin{aligned}
\sum_{i \in T_1} \lceil \log(1/\Lambda_{i,j_*}) \rceil &= \sum_{i \in T_1, \Lambda_{i,j_*} \neq 1} \lceil \log(1/\Lambda_{i,j_*}) \rceil + \sum_{i \in T_1, \Lambda_{i,j_*} = 1} \lceil \log(1/\Lambda_{i,j_*}) \rceil \\
&\leq \sum_{i \in T_1, \Lambda_{i,j_*} \neq 1} \lceil \log(2 \cdot \deg(H_{i,j_*})) \rceil + 0 \\
&\leq 2 \cdot \sum_{i \in [t]} \log(\deg(H_{i,j_*})) \leq \frac{1}{10000} \cdot \log n < \frac{1}{1000} \cdot m,
\end{aligned}$$

where we used the fact that if  $\Lambda_{i,j_*} \neq 1$ , then  $\deg(H_{i,j_*}) \geq 2$ . Let  $Z_i = \text{Cay}(\mathbb{F}_2^m, \text{SB}_i)$ , where  $\text{SB}_i: K_i \rightarrow \{0, 1\}^m$  is some generator such that  $\text{SB}_i(U_{K_i}) = D_i$ . (Such a generator exists, because all probabilities under  $D_i$  are rational numbers.) This graph satisfies  $\lambda(Z_i) \leq \Lambda_{i,j_*}$  by Lemma 16, and each vertex in  $Z_i$  has  $|\text{Supp}(D_i)|$  distinct neighbors. Let  $q = 2^{j_*-1}$  and  $Q = 2^q$ , and we define

$$W_i = Z_i \otimes J_{Q/M}.$$

Recall that  $\mathcal{H}'_1, \dots, \mathcal{H}'_t$  denote the rows of  $\mathcal{H}'$ . For each  $i \in T_1$ , let

$$\mathcal{H}'_i = [J_2, \dots, J_{2^{2^{j_*}-2}}, W_i, J_*, \dots, J_*],$$

that is, for all indices  $j' \neq j_*$ ,  $\mathcal{H}'_{i,j'}$  is the complete graph of appropriate size, and in  $j_*$ -th index, we use our  $W_i$  constructed before.

For each  $i' \in T_2$ , let

$$\mathcal{H}'_{i'} = [H_{i',1}, H_{i',2}, \dots, H_{i',j_*-1}, H_{i',j_*}, J_*, \dots, J_*],$$

i.e. the graphs up to  $j_*$  are exactly same as the corresponding graphs in  $\mathcal{H}$ , and after  $j_*$ , we use the complete graphs of appropriate size. By combining  $\mathcal{H}'_i, i \in T_1$  and  $\mathcal{H}'_{i'}, i' \in T_2$  accordingly, we get a new family graphs  $\mathcal{H}'$ . Since  $\mathcal{H}$  is a family of graphs that satisfy the constraint  $\Lambda$ , so is  $\mathcal{H}'$ .

By the definition of  $\text{INW}_{\mathcal{H}'}^{\oplus t}$ ,

$$\begin{aligned}
\text{INW}_{\mathcal{H}'}^{\oplus t}(x_1^0, \dots, x_t^0) &= \text{INW}_{\mathcal{H}'_1}(x_1^0) \oplus \dots \oplus \text{INW}_{\mathcal{H}'_t}(x_t^0) = G_{\log n}^{(1)}(x_1^0) \oplus \dots \oplus G_{\log n}^{(t)}(x_t^0) \\
&= (G_{\log n-1}^{(1)}(x'_1), G_{\log n-1}^{(1)}(H'_{1,\log n}[x'_1, y'_1])) \oplus \dots \oplus \\
&\quad (G_{\log n-1}^{(t)}(x'_t), G_{\log n-1}^{(t)}(H'_{t,\log n}[x'_t, y'_t])),
\end{aligned}$$

where  $x'_i$  and  $y'_i$  are the substrings of  $x_i$  of appropriate length, and each  $G_{\log n}^{(i)}$  as a INW generator in  $\text{INW}(\Lambda_i)$ , so that  $G_j^{(i)}$ ,  $j \in [\log n]$ , is defined recursively. By recursively expanding these generators until the  $j_*$ -th round, where we can express the output of  $\text{INW}_{\mathcal{H}'}^{\oplus t}$  as  $n/2^{j_*}$  independent copies, as  $\mathcal{H}'_{i,j'}$  are complete graphs for  $i \in [t]$  and  $j_* < j' \leq \log n$ , of the following form

$$\begin{aligned}
&G_{j_*}^{(1)}(x_1, y_1) \oplus \dots \oplus G_{j_*}^{(t)}(x_t, y_t) \\
&= (G_{j_*-1}^{(1)}(x_1), G_{j_*-1}^{(1)}(H'_{1,j_*}[x_1, y_1])) \oplus \dots \oplus (G_{j_*-1}^{(t)}(x_t), G_{j_*-1}^{(t)}(H'_{t,j_*}[x_t, y_t])) \\
&= (x_1 \oplus \dots \oplus x_{t_1} \oplus G_{j_*-1}^{(t_1+1)}(x_{t_1+1}) \oplus \dots \oplus G_{j_*-1}^{(t)}(x_t), \\
&\quad H'_{1,j_*}[x_1, y_1] \oplus \dots \oplus H'_{t_1,j_*}[x_{t_1}, y_{t_1}] \\
&\quad \oplus G_{j_*-1}^{(t_1+1)}(H'_{t_1+1,j_*}[x_{t_1+1}, y_{t_1+1}]) \oplus \dots \oplus G_{j_*-1}^{(t)}(H'_{t,j_*}[x_t, y_t])),
\end{aligned}$$

as  $G_{j_*-1}^{(i)}$  is the identity function for  $i \in T_1$ , where  $x_i, y_i$  are the corresponding input strings.

Let  $R'_i$  be the set of all possible  $G_{j_*-1}^{(i)}(x_i) \oplus G_{j_*-1}^{(i)}(H'_{i,j_*}[x_i, y_i])$  where we allow  $x_i$  to range over all vertices *and* we allow  $y_i$  to range over all edge labels, and let  $R_i = \{x_1, \dots, x_m \mid x \in R'_i\}$ , which is the set of first  $m$  bits of strings in  $R'_i$ . In the following, we are going to show that  $|\sum_{i=1}^t R_i| < 2^m$ , hence there is at least one  $v_* \in \{0, 1\}^m$  such that  $v_* \notin \sum_{i=1}^t R_i$ . Our approach is to bound

$$\left| \sum_{i=1}^t R_i \right| \leq \left| \sum_{i \in T_1} R_i \right| \cdot \prod_{i \in T_2} |R_i|.$$

We will bound  $|\sum_{i \in T_1} R_i|$  and  $\prod_{i \in T_2} |R_i|$  separately.

For each  $i \in T_1$ , let  $x_i^{(1)} \in \{0, 1\}^m$  be the first  $m$  bits of  $x_i$ , and  $x_i^{(2)} \in \{0, 1\}^{q-m}$  to be the substring of  $x_i$  after  $x_i^{(1)}$ , so that  $x_i = (x_i^{(1)}, x_i^{(2)})$ . For  $y_i$ , since  $W_i = Z_i \otimes J$ , we have  $y_i = (y_i^{(1)}, y_i^{(2)})$  where  $y_i^{(1)} \in K_i$  and  $y_i^{(2)} \in \{0, 1\}^{q-m}$ . Thus, for  $i \in T_1$ , we have

$$\begin{aligned} x_i \oplus H'_{i,j_*}[x_i, y_i] &= (x_i^{(1)}, x_i^{(2)}) \oplus H'_{i,j_*}[(x_i^{(1)}, x_i^{(2)}), (y_i^{(1)}, y_i^{(2)})] \\ &= (x_i^{(1)}, x_i^{(2)}) \oplus W_i[(x_i^{(1)}, x_i^{(2)}), (y_i^{(1)}, y_i^{(2)})] \\ &= (x_i^{(1)}, x_i^{(2)}) \oplus (Z_i[x_i^{(1)}, y_i^{(1)}], J_{Q/M}[x_i^{(2)}, y_i^{(2)}]) \\ &= (x_i^{(1)}, x_i^{(2)}) \oplus (Z_i[x_i^{(1)}, y_i^{(1)}], y_i^{(2)}). \end{aligned}$$

By our construction of  $Z_i$ , for any choice of  $x_i^{(1)}, y_i^{(1)}$ , we have  $x_i^{(1)} \oplus Z_i[x_i^{(1)}, y_i^{(1)}] \in \text{Supp}(D_i)$ , and

$$\bigoplus_{i \in T_1} x_i^{(1)} \oplus Z_i[x_i^{(1)}, y_i^{(1)}] \in \sum_{i \in T_1} \text{Supp}(D_i).$$

Therefore, by Lemma 31, we have

$$\left| \sum_{i \in T_1} R_i \right| \leq \left| \sum_{i \in T_1} \text{Supp}(D_i) \right| \leq 2 \cdot 2^{m/2}.$$

For  $i \in T_2$ , we have

$$G_{j_*-1}^{(i)}(x_i) \oplus G_{j_*-1}^{(i)}(H'_{i,j_*}[x_i, y_i]) = G_{j_*-1}^{(i)}(x_i) \oplus G_{j_*-1}^{(i)}(H_{i,j_*}[x_i, y_i]).$$

For  $i \in T_2$ ,  $\deg(H_{i,j_*}) \geq \frac{|H_{i,j_*}|+1}{2}$  and  $x_i$  is a vertex label in  $H_{i,j_*}$ ,

$$|R_i| \leq |H_{i,j_*}| \cdot \deg(H_{i,j_*}) \leq 2(\deg(H_{i,j_*}) - 1) \cdot \deg(H_{i,j_*}) \leq 2 \deg(H_{i,j_*})^2.$$

We know that  $\log(\deg(H_{1,j_*})) + \dots + \log(\deg(H_{t,j_*})) < \frac{1}{20000} \cdot \log n < 0.001 \cdot \log n$ . By Proposition 17, for any undirected graph  $h$  with degree 1, we know  $\lambda(h) = 1$ , so that for any  $i \in T_2$ ,  $\deg(H_{i,j_*}) \geq 2$ , which means  $|T_2| < 0.001 \cdot \log n$ . Therefore,

$$\prod_{i \in T_2} |R_i| \leq 2^{|T_2|} \cdot \prod_{i \in T_2} \deg(H_{i,j_*})^2 \leq 2^{0.001 \cdot \log n} \cdot n^{0.01} \leq n^{0.02} \leq 2^{0.2m}.$$

By the bounds above, the number of possible choices for  $v$  is at most

$$\left| \sum_{i \in T_1} R_i \right| \cdot \prod_{i \in T_2} |R_i| \leq 2 \cdot 2^{m/2+0.2m} < 2^m.$$

Therefore, there is at least one  $v_* \in \{0, 1\}^m$  such that  $v_* \notin \sum_{i=1}^t R_i$ .



Encoding this missing element to a width-3 branching program  $B$  computing the function  $f : \{0, 1\}^n \rightarrow \{0, 1\}$  as

$$f(x) = \bigvee_{i=0}^{n/2^{j^*}-1} (x_{2^{j^*} \cdot i + 1 \dots 2^{j^*} \cdot i + m} = 0^m \wedge x_{2^{j^*} \cdot i + q + 1 \dots 2^{j^*} \cdot i + q + m} = v_*),$$

we know that  $B(\text{INW}_{\mathcal{H}'}^{\oplus t}(x)) = 0$  for any input seed  $x$ . However, for a truly random input satisfies each clause with probability  $2^{-2m} \geq 1/\sqrt{n}$ , and since there are  $n/2^{j^*} \geq n^{3/4}$  such clauses, the acceptance probability is

$$\Pr[B(U_{\{0,1\}^n}) = 1] = 1 - \Pr[B(U_{\{0,1\}^n}) = 0] \geq 1 - (1 - 1/\sqrt{n})^{n^{3/4}} \geq 1 - \exp(-n^{1/4}). \blacktriangleleft$$

We can then prove Theorem 8.

► **Theorem 34** (Restatement of Theorem 8). *Let  $n$  be a power of two, let  $t \in \mathbb{N}$ , and let  $\Lambda \in [0, 1]^{t \times \log n}$ . If every PRG in the family  $\text{INW}^{\oplus t}(\Lambda)$  fools width-3 standard-order ROBPs with error 0.99, then every PRG in the family  $\text{INW}^{\oplus t}(\Lambda)$  has seed length  $\Omega(\log^2 n)$ .*

**Proof.** For every  $t \in \mathbb{N}$ , Lemma 33 implies that for every PRG in the family  $\text{INW}^{\oplus t}(\Lambda)$  and for each  $j \in [\log \log n, \log n]$ ,

$$\log(\deg(H_{1,j})) + \dots + \log(\deg(H_{t,j})) \geq \frac{1}{20000} \cdot \log n.$$

Consequently, the overall seed length is at least

$$\begin{aligned} \sum_{j \in [\log \log n, \log n]} (\log(\deg(H_{1,j})) + \dots + \log(\deg(H_{t,j}))) &\geq \sum_{j \in [\log \log n, \log n]} \frac{1}{20000} \cdot \log n \\ &= \Omega(\log^2 n). \quad \blacktriangleleft \end{aligned}$$

## 5 Directions for future research

It would be very interesting to prove a general “XOR lemma” saying that taking the bitwise XOR of many copies of a distribution amplifies its unpredictability for bounded-width ROBPs. Proving such a general lemma might be the key to analyzing derandomized sums of INW generators.

There is a well-known variant of the INW generator in which we use an extractor to recycle the seed at each stage instead of using an expander, i.e., the recursive step is  $G_{j+1}(x, y) = (G_j(x), G_j(\text{Ext}(x, y)))$ . This construction and its analysis are similar to the Nisan-Zuckerman PRG [39]. Intriguingly, it is not clear how to carry out the proof of Theorem 7 using extractors instead of expanders. Conceivably, an extractor-based proof might be more amenable to derandomization.

We would also like to highlight the problem of determining the optimal dependence on  $w$  in Theorem 7. Can the  $n^{\log(w+1)}$  term be improved to  $\text{poly}(n, w)$ ?

---

## References

- 1 Noga Alon, Oded Goldreich, Johan Håstad, and René Peralta. Simple constructions of almost  $k$ -wise independent random variables. *Random Structures Algorithms*, 3(3):289–304, 1992. doi:10.1002/rsa.3240030308.
- 2 Roy Armoni. On the derandomization of space-bounded computations. In *Proceedings of the 2nd Workshop on Randomization and Approximation Techniques in Computer Science (RANDOM)*, pages 47–59, 1998. doi:10.1007/3-540-49543-6\_5.

- 3 Sepehr Assadi and Vishvajeet N. Graph streaming lower bounds for parameter estimation and property testing via a streaming XOR lemma. In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing (STOC)*, pages 612–625, 2021. doi:10.1145/3406325.3451110.
- 4 Andrej Bogdanov, Zeev Dvir, Elad Verbin, and Amir Yehudayoff. Pseudorandomness for width-2 branching programs. *Theory Comput.*, 9:283–292, 2013. doi:10.4086/toc.2013.v009a007.
- 5 Andrej Bogdanov, William M. Hoza, Gautam Prakriya, and Edward Pyne. Hitting Sets for Regular Branching Programs. In *Proceedings of the 37th Computational Complexity Conference (CCC)*, pages 3:1–3:22, 2022. doi:10.4230/LIPIcs.CCC.2022.3.
- 6 Andrej Bogdanov and Emanuele Viola. Pseudorandom bits for polynomials. *SIAM J. Comput.*, 39(6):2464–2486, April 2010. doi:10.1137/070712109.
- 7 Mark Braverman, Gil Cohen, and Sumegha Garg. Pseudorandom pseudo-distributions with near-optimal error for read-once branching programs. *SIAM J. Comput.*, 49(5):STOC18–242–STOC18–299, 2020. doi:10.1137/18M1197734.
- 8 Mark Braverman, Anup Rao, Ran Raz, and Amir Yehudayoff. Pseudorandom generators for regular branching programs. *SIAM J. Comput.*, 43(3):973–986, 2014. doi:10.1137/120875673.
- 9 Joshua Brody and Elad Verbin. The coin problem and pseudorandomness for branching programs. In *2010 IEEE 51st Annual Symposium on Foundations of Computer Science*, pages 30–39, 2010. doi:10.1109/FOCS.2010.10.
- 10 Eshan Chattopadhyay and Jyun-Jie Liao. Recursive Error Reduction for Regular Branching Programs. In *15th Innovations in Theoretical Computer Science Conference (ITCS)*, pages 29:1–29:20, 2024. doi:10.4230/LIPIcs.ITCS.2024.29.
- 11 Lijie Chen, William M. Hoza, Xin Lyu, Avishay Tal, and Hongxun Wu. Weighted pseudorandom generators via inverse analysis of random walks and shortcutting. In *Proceedings of the 64th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 1224–1239, 2023. doi:10.1109/FOCS57990.2023.00072.
- 12 Lijie Chen and Xin Lyu. Inverse-exponential correlation bounds and extremely rigid matrices from a new derandomized XOR lemma. In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing (STOC)*, pages 761–771, 2021. doi:10.1145/3406325.3451132.
- 13 Kuan Cheng and Ruiyang Wu. Weighted pseudorandom generators for read-once branching programs via weighted pseudorandom reductions, 2025. doi:10.48550/arXiv.2502.08272.
- 14 Gil Cohen, Dean Doron, Oren Renard, Ori Sberlo, and Amnon Ta-Shma. Error Reduction for Weighted PRGs Against Read Once Branching Programs. In *Proceedings of the 36th Computational Complexity Conference (CCC)*, pages 22:1–22:17, 2021. doi:10.4230/LIPIcs.CCC.2021.22.
- 15 Anindya De. Pseudorandomness for permutation and regular branching programs. In *Proceedings of the 2011 IEEE 26th Annual Conference on Computational Complexity, CCC '11*, pages 221–231, USA, 2011. IEEE Computer Society. doi:10.1109/CCC.2011.23.
- 16 Bill Fefferman, Ronen Shaltiel, Christopher Umans, and Emanuele Viola. On beating the hybrid argument. *Theory Comput.*, 9:809–843, 2013. doi:10.4086/toc.2013.v009a026.
- 17 Michael A. Forbes and Zander Kelley. Pseudorandom generators for read-once branching programs, in any order. In *Proceedings of the 59th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 946–955, 2018. doi:10.1109/FOCS.2018.00093.
- 18 Anat Ganor and Ran Raz. Space Pseudorandom Generators by Communication Complexity Lower Bounds. In Klaus Jansen, José Rolim, Nikhil R. Devanur, and Cristopher Moore, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2014)*, volume 28 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 692–703, Dagstuhl, Germany, 2014. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.APPROX-RANDOM.2014.692.
- 19 Pooya Hatami and William Hoza. Paradigms for unconditional pseudorandom generators. *Found. Trends Theor. Comput. Sci.*, 16(1-2):1–210, 2024. doi:10.1561/0400000109.

- 20 William Hoza and Zelin Lv. On sums of inw pseudorandom generators. ECCC preprint TR25-050, 2025. URL: <https://eccc.weizmann.ac.il/report/2025/050/>.
- 21 William M. Hoza. Better Pseudodistributions and Derandomization for Space-Bounded Computation. In *Proceedings of the 25th International Conference on Randomization and Computation (RANDOM)*, pages 28:1–28:23, 2021. doi:10.4230/LIPIcs.APPROX/RANDOM.2021.28.
- 22 William M. Hoza, Edward Pyne, and Salil Vadhan. Pseudorandom Generators for Unbounded-Width Permutation Branching Programs. In *12th Innovations in Theoretical Computer Science Conference (ITCS)*, pages 7:1–7:20, 2021. doi:10.4230/LIPIcs.ITCS.2021.7.
- 23 William M. Hoza, Edward Pyne, and Salil Vadhan. Limitations of the Impagliazzo-Nisan-Wigderson pseudorandom generator against permutation branching programs. *Algorithmica*, 2024, 2024. URL: <https://link.springer.com/article/10.1007/s00453-024-01251-2>.
- 24 William M. Hoza and David Zuckerman. Simple optimal hitting sets for small-success  $\mathbb{RL}$ . *SIAM J. Comput.*, 49(4):811–820, 2020. doi:10.1137/19M1268707.
- 25 Russell Impagliazzo. Hard-core distributions for somewhat hard problems. In *Proceedings of the 36th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 538–545, 1995. doi:10.1109/SFCS.1995.492584.
- 26 Russell Impagliazzo, Noam Nisan, and Avi Wigderson. Pseudorandomness for network algorithms. In *Proceedings of the 26th Annual Symposium on Theory of Computing (STOC)*, pages 356–364, 1994. doi:10.1145/195058.195190.
- 27 Russell Impagliazzo and Avi Wigderson.  $P = BPP$  if  $E$  requires exponential circuits: derandomizing the XOR lemma. In *Proceedings of the 29th Annual ACM Symposium on Theory of Computing (STOC)*, pages 220–229, 1997. doi:10.1145/258533.258590.
- 28 Michal Koucký, Prajakta Nimbhorkar, and Pavel Pudlák. Pseudorandom generators for group products: extended abstract. In *Proceedings of the Forty-Third Annual ACM Symposium on Theory of Computing, STOC '11*, pages 263–272, New York, NY, USA, 2011. Association for Computing Machinery. doi:10.1145/1993636.1993672.
- 29 Vinayak M. Kumar. New Pseudorandom Generators and Correlation Bounds Using Extractors. In *Proceedings of the 16th Innovations in Theoretical Computer Science Conference (ITCS)*, volume 325, pages 68:1–68:23, 2025. doi:10.4230/LIPIcs.ITCS.2025.68.
- 30 Chin Ho Lee, Edward Pyne, and Salil Vadhan. On the Power of Regular and Permutation Branching Programs. In *Proceedings of the 27th International Conference on Randomization and Computation (RANDOM)*, pages 44:1–44:22, 2023. doi:10.4230/LIPIcs.APPROX/RANDOM.2023.44.
- 31 Chin Ho Lee and Emanuele Viola. Some limitations of the sum of small-bias distributions. *Theory of Computing*, 13(16):1–23, 2017. doi:10.4086/toc.2017.v013a016.
- 32 Shachar Lovett. Unconditional pseudorandom generators for low-degree polynomials. *Theory of Computing*, 5(3):69–82, 2009. doi:10.4086/toc.2009.v005a003.
- 33 Shachar Lovett and Yoav Tzur. Explicit lower bound for fooling polynomials by the sum of small-bias generators. ECCC preprint TR09-088, 2009. URL: <https://eccc.weizmann.ac.il/report/2009/088/>.
- 34 Ueli Maurer and Stefano Tessaro. Computational indistinguishability amplification: tight product theorems for system composition. In *Proceedings of the 29th International Cryptology Conference (CRYPTO)*, pages 355–373, 2009. doi:10.1007/978-3-642-03356-8\_21.
- 35 Raghu Meka, Omer Reingold, and Avishay Tal. Pseudorandom generators for width-3 branching programs. In *Proceedings of the 51st Annual Symposium on Theory of Computing (STOC)*, pages 626–637, 2019. doi:10.1145/3313276.3316319.
- 36 Raghu Meka and David Zuckerman. Small-bias spaces for group products. In *Proceedings of the 13th International Workshop on Randomization and Computation (RANDOM)*, pages 658–672, 2009. doi:10.1007/978-3-642-03685-9\_49.
- 37 Joseph Naor and Moni Naor. Small-bias probability spaces: efficient constructions and applications. *SIAM J. Comput.*, 22(4):838–856, 1993. doi:10.1137/0222053.

- 38 Noam Nisan. Pseudorandom generators for space-bounded computation. *Combinatorica*, 12(4):449–461, 1992. doi:10.1007/BF01305237.
- 39 Noam Nisan and David Zuckerman. Randomness is linear in space. *J. Comput. System Sci.*, 52(1):43–52, 1996. doi:10.1006/jcss.1996.0004.
- 40 Edward Pyne and Salil Vadhan. Pseudodistributions That Beat All Pseudorandom Generators (Extended Abstract). In *Proceedings of the 36th Computational Complexity Conference (CCC)*, pages 33:1–33:15, 2021. doi:10.4230/LIPIcs.CCC.2021.33.
- 41 Ran Raz and Omer Reingold. On recycling the randomness of states in space bounded computation. In *The 31st Annual ACM Symposium on Theory of Computing (STOC)*, pages 159–168, 1999. doi:10.1145/301250.301294.
- 42 Omer Reingold. Undirected connectivity in log-space. *J. ACM*, 55(4):Art. 17, 24, 2008. doi:10.1145/1391289.1391291.
- 43 Eyal Rozenman and Salil Vadhan. Derandomized squaring of graphs. In *Proceedings of the 9th International Workshop on Randomization and Computation (RANDOM)*, pages 436–447, 2005. doi:10.1007/11538462\_37.
- 44 Michael Saks and David Zuckerman, 1995. Unpublished.
- 45 Thomas Steinke. Pseudorandomness for permutation branching programs without the group theory. ECCC preprint TR12-083, 2012. URL: <https://eccc.weizmann.ac.il/report/2012/083/>.
- 46 Yoav Tzur. Notions of weak pseudorandomness and  $GF(2^n)$ -polynomials. M.Sc. thesis, Weizmann Institute of Science, 2009. URL: [https://eccc.weizmann.ac.il/static/books/Notions\\_of\\_Weak\\_Pseudorandomness/](https://eccc.weizmann.ac.il/static/books/Notions_of_Weak_Pseudorandomness/).
- 47 Salil P. Vadhan. Pseudorandomness. *Foundations and Trends in Theoretical Computer Science*, 7(1–3):1–336, 2012. doi:10.1561/04000000010.
- 48 Emanuele Viola. The sum of  $d$  small-bias generators fools polynomials of degree  $d$ . *Comput. Complexity*, 18(2):209–217, 2009. doi:10.1007/s00037-009-0273-5.