Electronic Colloquium on Computational Complexity, Report No. 108 (2021)

Limitations of the Impagliazzo–Nisan–Wigderson Pseudorandom Generator against Permutation Branching Programs

Edward Pyne*
Harvard University
epyne@college.harvard.edu

Salil Vadhan†
Harvard University
salil_vadhan@harvard.edu

July 22, 2021

Abstract

The classic Impagliazzo–Nisan–Wigderson (INW) psesudorandom generator (PRG) (STOC '94) for space-bounded computation uses a seed of length $O(\log n \cdot \log(nwd/\varepsilon))$ to fool ordered branching programs of length n, width w, and alphabet size d to within error ε. A series of works have shown that the analysis of the INW generator can be improved for the class of *permutation* branching programs or the more general *regular* branching programs, improving the $O(\log^2 n)$ dependence on the length n to $O(\log n)$ or $\tilde{O}(\log n)$. However, when also considering the dependence on the other parameters, these analyses still fall short of the optimal PRG seed length $O(\log(nwd/\varepsilon))$.

In this paper, we prove that any "spectral analysis" of the INW generator requires seed length

$$\Omega\left(\log n \cdot \log\log(\min\{n,d\}) + \log n \cdot \log(w/\varepsilon) + \log d\right)$$

to fool ordered permutation branching programs of length n, width w, and alphabet size d to within error ε. By "spectral analysis" we mean an analysis of the INW generator that relies only on the spectral expansion of the graphs used to construct the generator; this encompasses all prior analyses of the INW generator. Our lower bound matches the upper bound of Braverman–Rao–Raz–Yehudayoff (FOCS 2010, SICOMP 2014) for regular branching programs of alphabet size $d = 2$ except for a gap between their $O(\log n \cdot \log\log n)$ term and our $O(\log n \cdot \log\log \min\{n,d\})$ term. It also matches the upper bounds of Koucký–Nimbhorkar–Pudlák (STOC 2011), De (CCC 2011), and Steinke (ECCC 2012) for constant-width ($w = O(1)$) permutation branching programs of alphabet size $d = 2$ to within a constant factor.

To fool permutation branching programs in the stronger measure of *spectral norm*, we prove that any spectral analysis of the INW generator requires a seed of length $\Omega(\log n \cdot \log\log n + \log n \cdot \log(1/\varepsilon) + \log d)$ when the width is at least polynomial in n ($w = n^{\Omega(1)}$), matching the recent upper bound of Hoza–Pyne–Vadhan (ITCS '21) to within a constant factor.

Keywords: pseudorandomness, space-bounded computation, spectral graph theory

*Supported by NSF grant CCF-1763299.
†Supported by NSF grant CCF-1763299 and a Simons Investigator Award.

ISSN 1433-8092

1 Introduction

Starting with the work of Babai, Nisan, and Szegedy [BNS92], there has been three decades of work of constructing and analyzing pseudorandom generators for space-bounded computation, motivated by obtaining unconditional derandomization (e.g. seeking to prove that BPL=L) and a variety of other applications (e.g. [Ind06, Siv02, HVV06, HHR11]). Although we still remain quite far from having pseudorandom generators that suffice for a full derandomization of space-bounded computation, there has been substantial progress on pseudorandom generators for restricted models of space-bounded computation. In particular, a series of works have shown that the analysis of the classic Impagliazzo–Nisan–Wigderson (INW) generator [INW94] can be significantly improved for restricted models (e.g. "permutation branching programs"), but these analyses have not matched the parameters of an optimal pseudorandom generator. In this work, we show that there are inherent limitations to the analysis of the INW generator for these restricted models, proving lower bounds that nearly match the known upper bounds.

1.1 Pseudorandom Generators for Space-Bounded Computation

Like previous work, we will work with the following nonuniform model of space-bounded computation.

Definition 1.1. An **ordered branching program (OBP)** B of **length** n, **width** w and **alphabet size** d computes a function $B : [w] \times [d]^n \to [w]$. On an input $\sigma \in [d]^n$, the branching program computes as follows. It starts at a fixed start state $v_0 \in [w]$. Then for $t = 1, \ldots, n$, it reads the next symbol σ_t and updates its state according to a transition function $B_t : [w] \times [d] \to [w]$ by taking $v_t = B_t(v_{t-1}, \sigma_t)$. Note that the transition function B_t can differ at each time step.

Moreover, there is a set of accept states $V_e \subseteq [w]$. Let u be the final state of the branching program on input σ. If $u \in V_e$ the branching program accepts, denoted $B(\sigma) = 1$, and otherwise the program rejects, denoted $B(\sigma) = 0$.

An ordered branching program can be viewed as a layered digraph, consisting of $(n + 1)$ layers of w vertices each, where for every $t = 1, \ldots, n$ and $v \in [w]$, the v'th vertex in layer $t - 1$ has d outgoing edges, going to the vertices $B_t(v, 1), B_t(v, 2), \ldots, B_t(v, d) \in [w]$ in layer t.

An ordered branching program corresponds to a streaming algorithm, in that the n input symbols from $[d]$ are each read only once, and in a fixed order. This is the relevant model for derandomization of space-bounded computation because a randomized space-bounded algorithm processes its random bits in a streaming fashion. Specifically, if on an input x, a randomized algorithm A uses space s and n random bits σ, the function $B_x(\sigma) = A(x; \sigma)$ can be computed by an ordered branching program of length n, width $w = 2^s$ and alphabet size 2. In particular, if A is a randomized logspace algorithm (i.e. a BPL algorithm), then $n = w = \mathrm{poly}(|x|)$.

The standard definition of pseudorandom generator is as follows.

Definition 1.2. Let $\mathcal{F}$ be a class of functions $f : [d]^n \to \{0, 1\}$. An **ε-pseudorandom generator (ε-PRG)** for $\mathcal{F}$ is a function $\mathrm{GEN} : \{0, 1\}^s \to [d]^n$ such that for every $f \in \mathcal{F}$,

$$\left| \mathop{\mathbb{E}}_{x \leftarrow U_{[d]^n}} [f(x)] - \mathop{\mathbb{E}}_{x \leftarrow U_{\{0,1\}^s}} [f(\mathrm{GEN}(x))] \right| \leq \varepsilon,$$

where U_S is the uniform distribution over the set S. The value s is the **seed length** of the PRG. We say a generator GEN is **explicit** if the ith symbol of output is computable in space $O(s)$. We say that GEN **ε-fools $\mathcal{F}$** if it is an ε-PRG for $\mathcal{F}$.

By the Probabilistic Method, it can be shown that there exist (non-explicit) ε-PRGs for the class of ordered branching programs of length n, width w, and alphabet size d with seed length $s = O(\log(nwd/\varepsilon))$, and it can be shown that this is optimal up to a constant factor (provided that $2^n \geq w$, $n, d, w \geq 2$, d is even, and $\varepsilon \leq 1/3$). We include these bounds for completeness in Appendix C. An explicit construction with such a seed length (even for $d = 2$ and $\varepsilon = 1/3$) would suffice to fully derandomize logspace computation (i.e. prove BPL=L).

The classic constructions of Nisan [Nis92] and Impagliazzo, Nisan, and Wigderson [INW94] give explicit PRGs with seed length $s = O(\log n \cdot \log(nwd/\varepsilon))$. For the case corresponding to derandomizing general

logspace computation, where d and ε are constant and w is polynomially related to n, we have $s = O(\log^2 n)$, quadratically worse than the optimal seed length of $s = O(\log n)$. Brody and Verbin [BV10] showed that these classic pseudorandom generators require seed length $\Omega(\log^2 n)$ even for width $w = 3$. Meka, Reingold, and Tal [MRT19] recently gave a completely different explicit construction of pseudorandom generator for width $w = 3$ with seed length $s = \tilde{O}(\log n \cdot \log(1/\varepsilon)))$, but for width $w = 4$ no explicit constructions with seed length $o(\log^2 n)$ are known.

1.2 Permutation Branching Programs

Motivated by the lack of progress on the general ordered branching program model, there has been extensive research on restricted models:

Definition 1.3. An **(ordered) regular branching program** of length n, width w, and alphabet size d is an ordered branching program where the associated layered digraph consists of regular bipartite graphs between every pair of consecutive layers. Equivalently, for every $t = 1, \ldots, n$ and every $v \in [w]$, there are exactly d pairs $(u, \sigma) \in [w] \times [d]$ such that $B_t(u, \sigma) = v$.

Definition 1.4. An **(ordered) permutation branching program** of length n, width w, and alphabet size d is an ordered branching program where for all $t \in [n]$ and $\sigma \in [d]$, $B_t(\cdot, \sigma)$ is a permutation on $[w]$.

Every ordered permutation branching program is a regular branching program, but not conversely.

A series of works has shown that the Impagliazzo–Nisan–Wigderson (INW) pseudorandom generator can be instantiated with smaller seed length for regular or permutation branching programs. First, Rozenman and Vadhan [RV05] analyzed the INW generator for carrying out random walks on d-regular w-vertex graphs, which correspond to regular branching programs in which all of the transition functions B_t are the same. They showed that if the graph is consistently labelled (equivalently, that we have a permutation branching program), then a seed length of $s = O(\log(nwd/\varepsilon))$ suffices for the random walk to get within distance ε of the uniform distribution on vertices, provided that the length n of the pseudorandom walk is polynomially larger than the mixing time of a truly random walk. (This "pseudo-mixing" property is nonstandard but has applications, including giving a simpler proof of Reingold's Theorem that Undirected Connectivity is in deterministic logspace [Rei08] and the construction of almost k-wise independent permutations [KNR05].)

Next, Braverman, Rao, Raz, Yehudayoff [BRRY10] analyzed the INW generator for regular branching programs of alphabet size $d = 2$, and achieved seed length $s = O(\log n \cdot \log\log n + \log n \cdot \log(w/\varepsilon))$, thereby improving the dependence on the length n from $O(\log^2 n)$ to $\tilde{O}(\log n)$ for the standard pseudorandomness property. For the case of *permutation* branching programs of *constant width* $w = O(1)$ and alphabet size $d = 2$, Koucký and Nimbhorkar and Pudlák [KNP11] further improved the seed length to $s = O_w(\log n \cdot \log(1/\varepsilon))$. The hidden constant in the $O_w(.)$ depended exponentially on the width w, but was subsequently improved to a polynomial by De [De11] and Steinke [Ste12].

Recently, Hoza, Pyne, and Vadhan [HPV21] turned their attention to permutation branching programs of *unbounded width*, and showed that the INW generator fools such programs in "spectral norm" with seed length $s = O(\log n \cdot \log\log n + \log n \cdot \log(1/\varepsilon) + \log d)$. Here, fooling in spectral norm means that the $w \times w$ matrix of probabilities of going from each initial state to each final state under the generator has distance at most ε in spectral norm from the same matrix under truly random inputs. ε-fooling in spectral norm can be shown to imply the standard notion of pseudorandomness for programs with a single accept state. Surprisingly, the seed length of [HPV21] even beats the Probabilistic Method; indeed they show that a random function requires seed length $\Omega(n)$ to fool permutation branching programs of unbounded width and a single accept vertex with high probability.

We summarize the aforementioned analyses of the INW generator in Table 1. Let us elaborate on how all of these results are instantations of the INW generator. Specifically, the INW generator can be viewed as a template for a recursive construction of a PRG, where a PRG INW_{i-1} generating $n_{i-1} = 2^{i-1}$ output symbols is used to construct a PRG INW_i generating $n_i = 2^i$ output symbols, by running INW_{i-1} twice on a pair of correlated seeds. The pair of seeds are chosen according to a random edge in an auxiliary expander graph H_i:

$$\mathrm{INW}_i(e) = \mathrm{INW}_{i-1}(x) \cdot \mathrm{INW}_{i-1}(y) \text{ for each edge } e = (x, y) \text{ of } H_i, \tag{1}$$

Model	Seed Length	Pseudorandomness	Reference
General	$O(\log n \cdot \log(nwd/\varepsilon))$	standard	[INW94]
Perm., same trans.	$O(\log(nwd/\varepsilon))$	pseudo-mixing	[RV05]
Regular, $d = 2$	$O(\log n \cdot \log\log n + \log n \cdot \log(w/\varepsilon))$	standard	[BRRY10]
Permutation, $d = 2$	$O_w(\log n \cdot \log(1/\varepsilon))$	standard	[KNP11, De11, Ste12]
Permutation	$O(\log n \cdot \log\log n + \log n \cdot \log(1/\varepsilon) + \log d)$	spectral	[HPV21]

Table 1: Spectral Analyses of the INW Generator

where $\cdot$ denotes concatenation. Thus different choices of the sequence of graphs $H_1, H_2, \ldots, H_{\log n}$ yield different instantiations of the INW generator. In all of the aforementioned works,[1] the pseudorandomness property of the generator is proven using only the spectral expansion properties of the graphs H_i, namely requiring that all of the nontrivial normalized eigenvalues of H_i have absolute value at most some value λ_i for $i = 1, \ldots, \log n$. We call such an analysis a *spectral analysis* of the INW generator. Given a spectral analysis of the INW generator, the degrees of the expanders H_i are then determined by the optimal relationship between expansion and degree $d_i = \text{poly}(1/\lambda_i)$, which in turn determines the seed length of the final generator, namely

$$s = \Theta\left(\log\left(d \cdot d_1 \cdot d_2 \cdots d_{\log n}\right)\right) = \Omega\left(\log d + \sum_{i=1}^{\log n} \log(1/\lambda_i)\right). \tag{2}$$

1.3 Our Results

Given the improved analyses of the INW generator described in Table 1, it is natural to wonder how much further these analyses can be pushed. In particular, can the INW generator ε-fool permutation branching programs of length n, width w, and alphabet size d with seed length matching the optimal seed length of $O(\log(nwd/\varepsilon))$? Our main result is that the answer is no:

Theorem 1.5 (informally stated)**.** *Any spectral analysis of the INW generator for ε-fooling permutation branching programs of length n, width w, and alphabet size d requires seed length*

$$s = \Omega\left(\log n \cdot \log\log(\min\{n, d\}) + \log n \cdot \log(w/\varepsilon) + \log d\right).$$

Notice that this lower bound nearly matches the upper bounds in Table 1. In particular, we match the upper bound of [BRRY10] for regular branching programs, except that we get a $\log n \cdot \log\log n$ term only when $d = n^{\Omega(1)}$ while they have such a term even when $d = 2$. We also match the upper bounds of [KNP11, De11, Ste12] for permutation branching programs of alphabet size $d = 2$ and constant width $w = O(1)$.

For fooling with respect to spectral norm, we can get a lower bound of $\log n \cdot \log\log n$ whenever $w = n^{\Omega(1)}$, in particular matching the result of [HPV21] for unbounded-width permutation branching programs:

Theorem 1.6 (informally stated)**.** *For ε-fooling in spectral norm, any spectral analysis of the INW generator for permutation branching programs of length n, width w, and alphabet size d requires seed length*

$$s = \Omega\left(\log n \cdot \log\log(\min\{n, w\}) + \log n \cdot \log(1/\varepsilon) + \log d\right).$$

While our theorems are quite close to the upper bounds, they leave a few regimes where a spectral analysis of the INW generator could potentially yield an improved seed length. In particular, a couple of open questions stand out regarding the $\log n \cdot \log\log n$ in terms in the bounds:

- Can we achieve seed length $O(\log n \cdot \log(w/\varepsilon))$ for permutation (or even regular) branching programs of alphabet size $d = 2$? When the alphabet size is $d = 2$, the $\log\log(\min\{n, d\})$ term disappears in Theorem 1.5. However, the upper bound of [BRRY10] for regular branching programs still has an $O(\log n \cdot \log\log n)$ term, and the upper bounds of [KNP11, De11, Ste12] only achieve a polynomial dependence on the width w.

[1] Braverman et al. [BRRY10] analyze the INW generator constructed with *randomness extractors* [NZ96], but the extractor parameters they use follow from spectral expansion properties of the underlying graphs [GW97].

- Can we achieve seed length $O(\log n \cdot \log(1/\varepsilon))$ for permutation branching programs with a single accept vertex, alphabet size $d = 2$, and width $w = n$ (or even unbounded width)? The best upper bound for this model is [HPV21], which has a additional $O(\log n \cdot \log\log n)$ term. This term is necessary for fooling in spectral-norm by Theorem 1.6 but may not be necessary for the easier task of fooling programs with a single accept vertex.

A second opportunity for improvement is to go beyond spectral analysis of the INW generator, and exploit graphs H_i with additional properties. To indicate that there is some hope for this, we include an observation showing that there *exists* an instantiation of the INW generator that achieves optimal seed length, even against more general ordered branching programs:

Theorem 1.7. *For all $n, w, d \in \mathbb{N}$ and $\varepsilon > 0$, there exists a sequence of graphs $\mathcal{H}$ such that the INW generator constructed with this sequence ε-fools ordered branching programs of length n, width w and alphabet size d and has seed length $O(\log(nwd/\varepsilon))$.*

This is an application of the Probabilistic Method, and so does not give an explicit PRG.

Our lower bounds also say nothing about constructions that deviate from the template of the INW generator, and better seed lengths can potentially be obtained by modifying the INW generator or using it as a tool in more involved constructions. Examples include the pseudorandom generator for width 3 ordered branching programs [MRT19], which combines the INW generator with pseudorandom restrictions, and [BCG18, CL20, CDR+21, PV21, Hoz21], which construct "weighted pseudorandom generators" with a better dependence on the error by taking linear combinations of the INW generator (or blends of the Nisan and INW generator).

1.4 Techniques

Theorem 1.5 is really three separate lower bounds, which we state as separate theorems here to discuss the proof ideas separately. (The lower bound of $s = \Omega(\log d)$ is very simple, and we include it for completeness in Appendix C.)

Theorem 1.8 (informally stated)**.** *Any spectral analysis of the INW generator for $(1 - 1/w^{\Omega(1)})$-fooling permutation branching programs of length n, width w, and alphabet size $d = 2$ requires seed length $s = \Omega(\log n \cdot \log w)$.*

Note that the lower bound holds for a very large error parameter, namely $\varepsilon = 1 - 1/w^{\Omega(1)}$. In fact, it holds even for obtaining a *hitting-set generator*, where we Definition 1.2 is relaxed to only require that $\mathbb{E}_{x \leftarrow U_{[d]^n}}[f(x)] > \varepsilon$ implies that $\mathbb{E}_{x \leftarrow U_{\{0,1\}^s}}[f(\mathrm{GEN}(x))] > 0$.

To prove this Theorem 1.8, we show that most of the λ_i's parameterizing the INW generator must have $\lambda_i < 1/w^{\Omega(1)}$, which implies the seed-length lower bound by Equation (2). If that is not the case for some value of i, we construct an auxiliary graph H_i to use in the INW generator (with $\lambda(H_i) \le \lambda_i$) such that a permutation branching program only needs width $\mathrm{poly}(1/\lambda_i) \le w$ in order to perfectly distinguish a random edge in H_i from a pair of vertices in H_i that are disconnected. Specifically, we can take H_i to be an expander with degree $c_i = \mathrm{poly}(1/\lambda_i)$ and c_i^2 vertices. To be able to use such a graph in most levels in the INW generator, we may need to pad the number of vertices to a value larger than c_i. We do this by taking a tensor product with a complete graph, which retains both the expansion of H_i and the ability of a width w permutation branching program to distinguish edges and non-edges. We use complete graphs (with an appropriate edge labelling) for the remaining graphs H_j in the INW generator, and argue a permutation branching program of width w can still distinguish the output from uniform.

Theorem 1.9 (informally stated)**.** *Any spectral analysis of the INW generator for ε-fooling permutation branching programs of length n, width $w = 2$, and alphabet size $d = 2$ requires seed length $s = \Omega(\log n \cdot \log(1/\varepsilon))$.*

To prove Theorem 1.9 we use a construction from [RV05] used to show that the tightness of their analysis of the "derandomized square" operation on graphs. (Composing the INW generator with a permutation branching programs amounts to performing $\log n$ iterated derandomized square operations on the graph of the branching program.) Specifically, in order to show that each λ_i satisfies $\lambda_i = O(\varepsilon)$, we consider a graph

H_i that has a self-loop probability of λ_i but has $\lambda(H_i) \leq \lambda_i$. When the self-loop is taken, it means that two consecutive subsequences of the output of the INW generator of length 2^{i-1} are equal to each other, by Equation (1). Thus the permutation branching program of width 2 that computes the parity of the input bits on the union of those two subsequences will distinguish the output of the INW generator from uniform with advantage $\Omega(\lambda_i)$.

Theorem 1.10 (informally stated)**.** *Any spectral analysis of the INW generator for .1-fooling permutation branching programs of length n, width $w = 2$, and alphabet size d requires seed length $s = \Omega(\log n \cdot \log\log(\min\{n, d\}))$.*

To prove Theorem 1.10, we want to show that most of the λ_i's must satisfy $\lambda_i \leq O(1/\log n)$, where we assume without loss of generality that $d = n$. It suffices to prove that $\sum_{i=1}^{\log n} \lambda_i \leq O(1)$. To do this, we again consider graphs H_i that have a self-loop probability of λ_i, but rather than considering only one such graph, we use all of them in the INW generator. Intuitively, we want to show that the errors of $\Omega(\lambda_i)$ accumulate to lead to an overall error of $\Omega(\sum_i \lambda_i) > \varepsilon$. We consider a permutation branching program that corresponds to a random walk on a graph G with $w = 2$ vertices that has a self-loop probability of $1 - 1/d = 1 - 1/n$. A truly random walk of length n on G will end at its start vertex with probability at most $1 - n \cdot (1/n) \cdot (1 - 1/n)^{n-1} < .64$. We show that a pseudorandom walk using the INW generator with the graphs H_i will end at its start vertex with probability at least .75. Specifically, we choose our edge and vertex labellings carefully so that the self-loops in the graphs H_i cause random walks to backtrack with a high constant probability, so that it is as if we are typically doing random walks on G of length at most $n/4$.

Turning to Theorem 1.11, the only part of the lower bound that does not follow from the same arguments as above is the following:

Theorem 1.11 (informally stated)**.** *For 1/3-fooling in spectral norm, any spectral analysis of the INW generator for permutation branching programs of length n, width w, and alphabet size $d = 2$ requires seed length $s = \Omega(\log n \cdot \log\log(\min\{n, w\}))$.*

The proof of Theorem 1.11 is similar to that of Theorem 1.10, but instead of considering random walks on a 2-vertex graph G with large degree d, we use an graph G of degree 2 and a large number of vertices. Specifically we take G to be the undirected cycle on $w = \Theta(\sqrt{n})$ vertices. The key point is that the truly random walk on the cycle mixes in $n = \Theta(w^2)$ steps in spectral norm. So a truly random walk of length n will differ from complete mixing by at most, say 1/3, in spectral norm, but due to backtracking, the pseudorandom walks using the INW generator will differ from complete mixing by at least 2/3 in spectral norm.

1.5 Organization

In Section 2, we introduce formal definitions and give our general recipe for proving lower bounds. In Section 3, we prove Theorem 1.9, our lower bound in terms of the error of the pseudorandom generator. In Section 4, we show how the error incurred in different levels of the INW generator can accumulate, leading to Theorems 1.10 and Theorem 1.11. In Section 5, we prove Theorem 1.8, our lower bound in terms of the width. In Appendix A, we observe that this lower-bound technique gives stronger results for fooling general (e.g. non-regular) ordered branching programs, and in particular recovers the analysis of Brody and Verbin for bounds against width-3 OBPs. In Appendix B, we prove Theorem 1.7, establishing the existence of graphs enabling the INW generator to achieve optimal seed length.

2 Structure Of Lower Bounds

We now give the general approach to proving our lower bounds. To define spectral analysis, we introduce notation related to labeled graphs and distributions.

Definition 2.1 (One-Way Labeling [RV05])**.** A **one-way labeling** of a d-regular directed (multi)graph G assigns a label in $[d]$ to each edge (u, v) such that for every vertex u, the labels of the outgoing edges of u are distinct. For G with a one-way labeling, let $G[u, i]$ denote the vertex v such that (u, v) is labeled i.

Furthermore, for $\bar{y} = (y_1, \ldots, y_k) \in [d]^k$ let $G^k[x, \bar{y}]$ be the vertex obtained from following the sequence of edge labels $\bar{y}$, i.e. $G^k[x, \bar{y}] = G[G[\ldots G[x, y_1], \ldots, y_{k-1}], y_k]$.

For example, for all $w \in \mathbb{N}$, let J_w be the **complete graph** on w vertices with the one-way labeling $J_w[x, y] = y$ for all $x, y \in [w]$. We occasionally write J_* where the size of the graph is obvious from context. For the remainder of the paper all graphs have one-way labelings. All logs are base-2, and $[T] = \{0, \ldots, T-1\}$ for all $T \in \mathbb{N}$. In addition, we work with the random walk matrices of graphs, and the distribution induced by taking walks on graphs according to the output of a PRG.

Definition 2.2. For a d-regular labeled graph G on w vertices, let $\mathbf{W}_G[y] \in \{0,1\}^{w \times w}$ be the matrix where entry (u, v) is 1 if and only if $G[u, y] = v$. Furthermore, we can define the **random walk matrix of** G as $\overline{\mathbf{W}}_G = \mathbb{E}[\mathbf{W}_G[U_{[d]}]]$. Furthermore, for a function $\mathrm{GEN} : [S] \to [d]^k$, define

$$\overline{\mathbf{W}_{G^k} \circ \mathrm{GEN}} = \mathbb{E}[\mathbf{W}_{G^k}[\mathrm{GEN}(U_{[S]})]].$$

Note that with this notation, $\overline{\mathbf{W}}_{G^k} = (\overline{\mathbf{W}}_G)^k$ for every k.

Definition 2.3. For a d-regular digraph G on w vertices, define the **spectral expansion** of G as $\lambda(G) = \max_{x : x \perp 1} \|x\overline{\mathbf{W}}_G\|_2 / \|x\|_2$.

We now formally define the INW PRG.

Definition 2.4. Given $d_0 \in \mathbb{N}$ and a set of graphs $\mathcal{H} = (H_1, \ldots, H_\ell)$ where $\deg(H_i) = d_i$ and $|H_i| = \prod_{j=0}^{i-1} d_j$, the **INW generator constructed with** $\mathcal{H}$, denoted $\mathrm{INW}_{\mathcal{H}}$ or INW_ℓ when the family is clear, is the function $\mathrm{INW}_{\mathcal{H}} : [d_0] \times \cdots \times [d_\ell] \to [d_0]^{2^\ell}$ defined recursively where for $x \in [d_0]$ we have $\mathrm{INW}_0(x) = x$ and for $(x, y) \in ([d_0] \times \cdots \times [d_i], [d_{i+1}])$ we have

$$\mathrm{INW}_{i+1}(x, y) = (\mathrm{INW}_i(x), \mathrm{INW}_i(H_{i+1}[x, y])).$$

The seed length of $\mathrm{INW}_{\mathcal{H}}$ is thus $\left\lceil \log\left(\prod_{i=0}^{\ell} d_i\right)\right\rceil$.

We then define an analysis of the INW PRG that only "knows about" the spectral gap of the auxiliary graphs. For the remainder of the paper (with the exception of Appendix B) we assume all auxiliary graphs H are undirected, so we can assume $\mathbf{W}_H$ has a basis of eigenvalues.

Definition 2.5. For $d_0 \in \mathbb{N}$ and $\lambda_1, \ldots, \lambda_\ell \geq 0$, let $\mathrm{INW}(d_0, \lambda_1, \ldots, \lambda_\ell)$ be the set of INW PRGs $\mathrm{GEN} : [S] \to [d_0]^{2^\ell}$ constructed with auxilliary undirected regular graphs $H_1, \ldots, H_\ell$ where $\lambda(H_i) \leq \lambda_i$ for all i. We say $\mathrm{INW}(d_0, \lambda_1, \ldots, \lambda_\ell)$ ε-fools a class of functions $\mathcal{F}$ if every $\mathrm{GEN} \in \mathrm{INW}(d_0, \lambda_1, \ldots, \lambda_\ell)$ ε-fools every $f \in \mathcal{F}$. Furthermore, define $s_{\mathrm{INW}}(d_0, \lambda_1, \ldots, \lambda_\ell)$ as the minimal seed length of all PRGs in $\mathrm{INW}(d_0, \lambda_1, \ldots, \lambda_\ell)$. We call the set $(\lambda_1, \ldots, \lambda_\ell)$ a **constraint**, and say a family of graphs $(H_1, \ldots, H_\ell)$ **satisfies** the constraint if $\lambda(H_i) \leq \lambda_i$ for all i.

Given a family of INW PRGs, we can derive a lower bound on the seed length via the relation between degree and maximum expansion, as given by the following standard fact.

Proposition 2.6 (see e.g. Trevisan)**.** *Let G be an undirected d-regular graph on V vertices. Then*

$$\lambda(G) \geq \frac{1}{\sqrt{d}} \sqrt{\frac{V - d}{V - 1}}.$$

In particular $d \geq \min\{2/\lambda(G)^2, (V+1)/2\}$.

That is, the degree must be at least polynomially related to $1/\lambda(G)$ (as assumed in the seed-length calculation in Equation (2)), unless d is very close to the number of vertices. To deal with the latter case in our seed-length lower bounds, we will remove the terms corresponding to λ_i's where the $2/\lambda(G)^2 > (V+1)/2$, yielding the following:

Lemma 2.7. *Given* $\mathrm{INW}(d_0, \lambda_1, \ldots, \lambda_\ell)$ *where* $\lambda_i \geq 1/2^t$ *for all* i*, there is a set* $S \subseteq \{1, \ldots, \ell\}$ *with* $|S| \leq 2\log(\ell t)$ *such that*

$$s_{\mathrm{INW}}(d_0, \lambda_1, \ldots, \lambda_\ell) = \Omega\left(\sum_{i \in \{1,\ldots,\ell\} \setminus S} \log(1/\lambda_i)\right).$$

Proof. Recall that $s_{\mathrm{INW}}(d_0, \lambda_1, \ldots, \lambda_\ell) = \lceil \log d_0 \cdot m_p \rceil$, where m_p is the minimum product of degrees over all sets of auxiliary graphs $H_1, \ldots, H_\ell$ with the required spectral expansion. Let $H_1, \ldots, H_\ell$ be the family of undirected regular graphs with minimal product of degrees over all families satisfying the constraint. We claim without loss of generality at most $2\log(\ell t)$ such graphs are dense, in that they satisfy $2/\lambda(H_i)^2 > (|H_i|+1)/2$. Otherwise since $|H_1| \geq 2$ and $|H_i| = d_0 \cdot \prod_{j=1}^{i} \deg(H_j)$ the seed length is trivially $(3/2)^{2\log(\ell t)} = \Omega(\ell \cdot t)$ which already exceeds the lower bound. For all graphs H_i where this does not hold, we have $\deg(H_i) \geq 2/\lambda(H_i)^2 \geq 2/\lambda_i^2$ by Proposition 2.6. Then taking S^c to be the indices of these graphs, we have

$$m_p \geq \prod_{i \in S^c} \deg(H_i) \geq \prod_{i \in S^c} \frac{1}{\lambda_i^2}$$

and so we bound $s_{\mathrm{INW}}(d_0, \lambda_1, \ldots, \lambda_\ell) = \lceil \log d_0 \cdot m_p \rceil$ as desired. □

3 Dependence on Error

In this section, we prove Theorem 1.9, establishing a lower bound on the seed length as a function of the error of the generator.

Theorem 3.1 (Formal Statement of Theorem 1.9)**.** *For every* $n = 2^\ell$ *and* $\varepsilon \geq 2^{-n^{.1}}$ *and* $\lambda_1, \ldots, \lambda_\ell \geq 0$*, if* $\mathrm{INW}(2, \lambda_1, \ldots, \lambda_\ell)$ *ε-fools ordered permutation branching programs of length n, width 2, and alphabet size 2, then* $s_{\mathrm{INW}}(2, \lambda_1, \ldots, \lambda_\ell) = \Omega(\log n \cdot \log(1/\varepsilon))$.

This follows as a consequence of the main lemma, which shows that constructing an ε-biased space using the spectral INW generator requires constraining all spectral gaps to be $O(1/\varepsilon)$.

Lemma 3.2. *For all* $\varepsilon > 0$*, for every constraint* $(\lambda_1, \ldots, \lambda_\ell)$ *where there is* r *such that* $\lambda_r > 2\varepsilon$*, there is a family of auxilliary graphs* $\mathcal{H} = (H_1, \ldots, H_\ell)$ *where* $\lambda(H_i) \leq \lambda_i$ *and a alphabet size 2, width 2, length* $n = 2^\ell$ *permutation branching program B such that* $\mathrm{INW}_{\mathcal{H}}$ *fails to ε-fool B.*

To prove the lemma, we define convex combinations of graphs on the same vertex set.

Definition 3.3. For G, G' arbitrary d-regular graphs on n vertices, and $\lambda = a/b \in \mathbb{Q} \cap [0,1]$, let $H = \lambda G + (1-\lambda)G'$ be the $d \cdot b$-regular directed graph on n vertices where for $x \in [n]$ and $(y, c) \in [d] \times [b]$:

$$H[x, (y, c)] = \begin{cases} G[x, y] & c < a \\ G'[x, y] & c \geq a \end{cases}$$

We remark that with this definition, $\overline{\mathbf{W}}_H = \lambda \overline{\mathbf{W}}_G + (1-\lambda)\overline{\mathbf{W}}_{G'}$. We implicitly extend this to convex combinations of graphs with non-equal degrees d, d' by duplicating edges so both graphs have degree $\mathrm{LCM}(d, d')$.

We can then construct a bad family of graphs and a distinguisher.

Proof of Lemma 3.2. Take $\mu = 2\varepsilon$ and $K = 2^{2^{r-1}}$ and define

$$H = \mu I_K + (1-\mu) J_K$$

where I_K is the 1-regular graph on K vertices with a self-loop on each vertex. Then define the family $\mathcal{H} = (J_2, \ldots, J_{2^{2^{r-1}}}, H, J_*, \ldots, J_*)$. It is clear $\mathcal{H}$ satisfies the constraint.

Now let B be the length n, width 2, alphabet size 2 permutation branching program where

$$B(\sigma) = \bigoplus_{i=1}^{2^r} \sigma_i.$$

Is is clear that $\Pr[B(U_{\{0,1\}^n}) = 1] = 1/2$. Furthermore, for every seed $\sigma = (x, u, *)$ we have

$$\mathrm{INW}_{\mathcal{H}}(\sigma)_{1..2^r} = \mathrm{INW}_r((x,u)) = (\mathrm{INW}_{r-1}(x), \mathrm{INW}_{r-1}(H[x,u])).$$

From our definition of H, with probability $1-\mu$ over the random seed σ the first 2^r bits output are (x,y) where (x,y) is distributed uniformly over $\{0,1\}^{2^r}$, and with probability μ the first 2^r bits of output are (x,x), which has parity zero for all x. Therefore letting $[S]$ be the seed length of $\mathrm{INW}_{\mathcal{H}}$,

$$\Pr[B(\mathrm{INW}_{\mathcal{H}}(U_{[S]})) = 1] \geq \frac{1}{2}(1-\mu) + 1\mu > 1/2 + \varepsilon$$

so $\mathrm{INW}_{\mathcal{H}}$ fails to ε-fool B. □

We can then prove Theorem 3.1 by combining Lemma 3.2 and Lemma 2.7.

Proof of Theorem 3.1. Applying Lemma 3.2, every family $\mathrm{INW}(2, \lambda_1, \ldots, \lambda_\ell)$ that ε-fools length n, width w, alphabet size 2 permutation branching programs has $\lambda_i < 2\varepsilon$ for all i, so we obtain $s_{\mathrm{INW}}(d_0, \lambda_1, \ldots, \lambda_\ell) = \Omega(\log(1/\varepsilon)(\log(n) - 2\log\log(n/\varepsilon)))$ via Lemma 2.7, which simplifies to $\Omega(\log n \cdot \log(1/\varepsilon))$ with the assumption $\varepsilon \geq 2^{-n^{.1}}$. □

4 Accumulation of Error

In this section, we prove the lower bounds on seed length $\Omega(\log n \cdot \log\log \min\{n, d\})$ and $\Omega(\log n \cdot \log\log \min\{n, w\})$ from Theorems 1.5 and 1.6, respectively. As discussed in the introduction, in both of these lower bounds, we wish to show that that the error $\Omega(\lambda_i)$ demonstrated in Section 3 actually accumulates to give an error of $\varepsilon = \Omega(\sum_i \lambda_i)$, which will imply that most of the λ_i's are $O(1/\log n)$ and hence we require seed length $\Omega(\log n \cdot \log\log n)$. For the standard notion of pseudorandomness, we will be able to argue this when the alphabet size of the branching programs is polynomially related to n, and for fooling in spectral norm, we will be able to argue it when the width of the branching program is polynomially related to n.

4.1 The INW PRG On Reversible Graphs

We will analyze the distribution of the output of the INW PRG over graphs, taking the transition function of the branching program to equal that of the graph. We recall the connection between consistently labeled graphs and permutation branching programs.

Definition 4.1. A d-regular labeled graph G on w vertices is **consistently labeled** if $G[v,i] = G[v',i]$ implies $v = v'$ for all $v, v' \in [w]$, $i \in [d]$. Equivalently, each edge label $i \in [d]$ defines a permutation over $[w]$.

Remark 4.2. Given a d-regular consistently labeled graph G on w vertices and $n \in \mathbb{N}$, the branching program G^n of length n, width w and alphabet size d with transition functions $G_1(v,b) = \ldots = G_n(v,b) = G[v,b]$ is a permutation branching program.

To prove these results, we introduce a graph property that will be satisfied by the graphs we use as our distinguishing permutation branching programs. Furthermore, given such a graph we construct a family of expanders such that the INW PRG behaves as if it is taking walks that are a constant factor shorter than truly random, and are thus distinguishable.

Definition 4.3. A d-regular labeled graph G on w vertices is **reversible** if there exists a matching $\pi : [d] \to [d]$ such that for every edge label $\sigma \in [d]$ and vertex $v \in [w]$ we have $G^2[v, (\sigma, \pi(\sigma))] = G[G[v,\sigma], \pi(\sigma)] = v$, i.e. $\mathbf{W}_{G^2}[(\sigma, \pi(\sigma))] = \mathbf{I}$ for all σ.

Furthermore, given a matching π and an edge sequence $\sigma = (\sigma_1, \ldots, \sigma_m)$, define $\pi(\sigma) \stackrel{\mathrm{def}}{=} (\pi(\sigma_m), \ldots, \pi(\sigma_1))$. Then reversibility extends to arbitrary edge sequences.

Lemma 4.4. *Given a d-regular reversible graph G with matching π, for every vertex v and edge sequence $\sigma \in [d]^m$, $G^{2m}[v, (\sigma, \pi(\sigma))] = v$.*

Proof. This follows from induction on m. The case $m = 1$ is clear from the definition, and assuming it holds for $m-1$, fix arbitrary v and $\sigma \in [d]^m$. We have

$$\begin{aligned} G^{2m}[v, (\sigma, \pi(\sigma))] &= G^{m-1}[G[G[G^{m-1}[v, \sigma_{1..m-1}], \sigma_m], \pi(\sigma_m)], \pi(\sigma_{1..m-1})] \\ &= G^{m-1}[G^{m-1}[v, \sigma_{1..m-1}], \pi(\sigma_{1..m-1})] \\ &= v \end{aligned}$$

so the inductive step holds. □

Now given a reversible graph and a constraint set that is not too restrictive, we can construct an INW PRG that performs in a well-behaved fashion. Intuitively, each generator output will consist of a mixture of truly random steps and steps that are "backtracked" and thus do not contribute to mixing. These backtracked steps will wipe out at least 3/4 of progress with high probability over the seed.

Lemma 4.5. *Let G be a d-regular reversible graph and $(\lambda_i)_\ell$ a constraint where $\sum_{i=1}^{\ell} \lambda_i > 8$. Then there is a family of auxiliary graphs $\mathcal{H} = (H_1, \ldots, H_\ell)$ where $\lambda(H_i) \le \lambda_i$ such that $\mathrm{INW}_{\mathcal{H}}$ satisfies:*

- $\overline{\mathbf{W}_{G^{2^\ell}} \circ \mathrm{INW}_{\mathcal{H}}}$ *is a convex combination of* $\overline{\mathbf{W}}_G^0, \ldots, \overline{\mathbf{W}}_G^{2^\ell}$.
- *In this convex combination, the sum of coefficients on* $\overline{\mathbf{W}}_G^0, \ldots, \overline{\mathbf{W}}_G^{2^\ell/4}$ *is at least .99.*

Our strategy is to use the fact that the graph is reversible to cause PRG outputs to backtrack with high probability. To do so, we define a property that each level of our PRG construction will satisfy.

Definition 4.6. Given a matching $\pi : [d] \to [d]$, a generator $\mathrm{GEN} : [S] \to [d]^r$ is **balanced with respect to** π if for all $v \in [d]^r$ we have $|\mathrm{GEN}^{-1}(v)| = |\mathrm{GEN}^{-1}(\pi(v))|$.

We are now prepared to prove the lemma. We iteratively construct the PRG to comply with the constraint, while backtracking as many steps as possible.

Proof of Lemma 4.5. Let $\mathrm{INW}_0 : [d] \to [d]$ be the trivial PRG that outputs its input. At each step we maintain that INW_i is balanced with respect to π, which is clearly satisfied for level 0.

Given $\mathrm{INW}_i : [S_i] \to [d]^{2^i}$, we show how to construct $\mathrm{INW}_{i+1} : [S_{i+1}] \to [d]^{2^{i+1}}$. For every output $t \in [d]^{2^i}$, let $[S_i] \supseteq R_t = \mathrm{INW}_i^{-1}(t)$. We have by assumption that $|R_t| = |R_{\pi(t)}|$ for all t. Let M_t be an arbitrary matching between R_t and $R_{\pi(t)}$ and define $M = \bigcup_{t \in [d]^{2^i}} M_t$. Then define

$$H_{i+1} = \lambda_{i+1} M + (1 - \lambda_{i+1}) J_*.$$

And define INW_{i+1} using this graph. Then the INW PRG constructed with this graph remains balanced.

Claim 4.7. $\mathrm{INW}_{i+1}(x, u) = (\mathrm{INW}_i(x), \mathrm{INW}_i(H_{i+1}[x, u]))$ *is balanced with respect to* π.

Proof. Fix an arbitrary output $(a, b) \in [d]^{2^i} \times [d]^{2^i}$.

- Let h be the number of seeds producing output (a, b) from neighbor relations in M. If $h > 0$ we have $\pi(a) = b$, so $(a, b) = (a, \pi(a))$. But then $\pi(a, \pi(a)) = (a, \pi(a))$, so the matching is vacuously balanced.
- Let k be the number of seeds producing output (a, b) from neighbor relations in J. By the inductive hypothesis $|R_{\pi(a)}| = |R_a|$ and $|R_{\pi(b)}| = |R_b|$ so there are also k seeds in J producing $(\pi(b), \pi(a)) = \pi(a, b)$ so outputs corresponding to neighbor relations in J are balanced.

Then since the sum of balanced functions is a balanced function we conclude. □

Finally, $\lambda(H_i) \le \lambda_i \cdot \lambda(M) \le \lambda_i$ since the complete graph falls out, so the family satisfies the constraint. We then analyze the distribution of outputs of the PRG. Since INW_0 is the trivial PRG we have $\overline{\mathbf{W}_G \circ \mathrm{INW}_0} = \overline{\mathbf{W}}_G$.

Claim 4.8. *For all $i \in [\ell]$,*

$$\overline{\mathbf{W}_{G^{2^{i+1}}} \circ \mathrm{INW}_{i+1}} = \lambda_{i+1} \mathbf{I}_{|G|} + (1 - \lambda_{i+1})(\overline{\mathbf{W}_{G^{2^i}} \circ \mathrm{INW}_i})^2.$$

Proof. Fixing arbitrary $v \in |G|$, we compute the distribution of $G[v, \mathrm{INW}_{i+1}(\sigma)]$ over a random seed $\sigma = (x, u) \leftarrow U_{[S_{i+1}]}$ of INW_{i+1}. From our construction of H_{i+1}, with probability λ_{i+1} over the random seed this corresponds to a neighbor in the matching M, so we have

$$\mathrm{INW}_{i+1}(\sigma) = (\mathrm{INW}_i(x), \mathrm{INW}_i(H_{i+1}[x,u])) = (t, \pi(t))$$

for some $t \in [d]^{2^i}$, and so $G[v, \mathrm{INW}_{i+1}(\sigma)] = v$. Otherwise, with probability $1 - \lambda_{i+1}$ over the random seed $H_{i+1}[x,u]$ corresponds to a neighbor in J_*, so we have

$$\mathrm{INW}_{i+1}(\sigma) = (\mathrm{INW}_i(x), \mathrm{INW}_i(H_{i+1}[x,u])) = (\mathrm{INW}_i(x), \mathrm{INW}_i(y))$$

with x, y independent and uniformly distributed over $U_{[S_i]}$, so the result follows. □

It is clear by induction on the above relation that $\overline{\mathbf{W}_{G^{2^\ell}} \circ \mathrm{INW}_\ell}$ defines a convex combination over $\overline{\mathbf{W}}_G^0, \ldots, \overline{\mathbf{W}}_G^{2^i}$, so it remains to bound the coefficients. To do so, for every seed σ of INW_ℓ let $r(\sigma)$ denote the number of symbols of output of $\mathrm{INW}_\ell(\sigma)$ that are not part of some section of output of the form $(t, \pi(t))$. Showing $\Pr[r(U_{[S_\ell]}) \le 2^\ell/4] > 99/100$ suffices to bound the coefficients. We say the jth symbol of the output of $\mathrm{INW}_\ell(\sigma)$ is matched if any of the ℓ neighbor relations corresponding to the jth symbol are contained in some matching M. We show for all j, with probability at least 999/1000 the jth symbol of output of $\mathrm{INW}_\ell(U_{[S_\ell]})$ is matched. Thus by linearity of expectation $\mathbb{E}[r(U_{[S_\ell]})] < 2^\ell/1000$ and the result follows from Markov.

Let $\mu = \frac{1}{\ell}\sum_{i=1}^{\ell} \lambda_i \ge 8/\ell$, and observe μ is the average probability a neighbor relation is contained in a matching. Then the probability the jth symbol of output was never matched is bounded above by

$$\prod_{i=1}^{\ell}(1-\lambda_i) \le (1-\mu)^\ell \le e^{-\ell\mu} \le 1/1000$$

where the first inequality is AM-GM, so the expected number of non-matched symbols of output is bounded above by $2^\ell/1000$ as desired. □

4.2 Branching Programs of Large Alphabet Size

Theorem 4.9 (Formal Statement of Theorem 1.10)**.** *For every $n = 2^\ell$ and $\lambda_1, \ldots, \lambda_\ell \ge 0$, if $\mathrm{INW}(d, \lambda_1, \ldots, \lambda_\ell)$ 1/10-fools ordered permutation branching programs of length n, width 2, and alphabet size d, then $s_{\mathrm{INW}}(d, \lambda_1, \ldots, \lambda_\ell) = \Omega(\log n \cdot \log\log(\min\{n, d\}))$.*

We can now prove the key lemmas, the first being the constraint against polynomial alphabet size permutation branching programs of width 2. Recall that $\mathrm{Bin}(m, p, t)$ is the probability of obtaining t heads from m iid Bernoulli(p) draws.

Lemma 4.10. *For every $n = 2^\ell$ and every constraint $(\lambda_1, \ldots, \lambda_\ell)$ where $\sum_{i=1}^{\ell} \lambda_i > 8$, there is a family of auxilliary graphs $\mathcal{H} = (H_1, \ldots, H_\ell)$ where $\lambda(H_i) \le \lambda_i$ and a length n, width 2, alphabet size n permutation branching program B such that the INW generator constructed with $\mathcal{H}$ fails to 1/10-fool B.*

Proof. Let G be the directed 2-cycle with $n-1$ self loops on each vertex. We will work with walks of length n over this graph, equivalent to computation on the length n permutation branching program $B = G^n$ as in Remark 4.2.

It is easy to see that G is reversible (in fact with π the identity function), so we apply Lemma 4.5 with G and $(\lambda_1, \ldots, \lambda_\ell)$ and obtain a PRG $\mathrm{INW}_{\mathcal{H}}$ where $\mathcal{H}$ satisfies the constraint.

To obtain the separation, we examine the probability that a random output of $\mathrm{INW}_{\mathcal{H}}$ ends at state 0 from state 0 (i.e. $(\overline{\mathbf{W}_{G^n} \circ \mathrm{INW}_{\mathcal{H}}})_{0,0}$), compared to the equivalent probability over truly random input (i.e. $(\overline{\mathbf{W}}_{G^n})_{0,0} = (\overline{\mathbf{W}}_G^n)_{0,0}$).

The probability a random walk of length n from state 0 in G ends at state 0 is lower bounded by 1 minus the probably such a walk takes exactly one non-self loop step. Therefore,

$$(\overline{\mathbf{W}}_{G^n})_{0,0} \le 1 - \mathrm{Bin}(n, 1/n, 1) = 1 - \frac{1}{n} n \left(1 - \frac{1}{n}\right)^{n-1} \le .633.$$

Intuitively, in the PRG output no backtracked section can possibly change the parity of the state, so $(\overline{\mathbf{W}_{G^n} \circ \mathrm{INW}_{\mathcal{H}}})_{0,0}$ is at least the probability that none of the non-backtracked steps (which are truly random) traverse edges in the cycle. Formally, for all $m \in \mathbb{N}$ we have $(\overline{\mathbf{W}}_G^m)_{0,0} \geq \mathrm{Bin}(m, 1/n, 0)$. Since this bound is monotonically decreasing with m, we lower bound $(\overline{\mathbf{W}_{G^n} \circ \mathrm{INW}_{\mathcal{H}}})_{0,0}$ by Lemma 4.5:

$$\begin{aligned}(\overline{\mathbf{W}_{G^n} \circ \mathrm{INW}_{\mathcal{H}}})_{0,0} &\geq \frac{1}{100}\mathrm{Bin}(n, 1/n, 0) + \frac{99}{100}\mathrm{Bin}(n/4, 1/n, 0)\\ &\geq \frac{99}{100}(1-(1-1/n)^{n/4})\\ &\geq .75\end{aligned}$$

Therefore $(\overline{\mathbf{W}_{G^n} \circ \mathrm{INW}_{\mathcal{H}}})_{0,0} - (\overline{\mathbf{W}}_{G^n})_{0,0} \geq .11$ and we have an $\Omega(1)$ separation as desired. □

We can then use this lemma to prove Theorem 4.9.

Theorem 4.9 (Formal Statement of Theorem 1.10)**.** *For every $n = 2^\ell$ and $\lambda_1, \ldots, \lambda_\ell \geq 0$, if $\mathrm{INW}(d, \lambda_1, \ldots, \lambda_\ell)$ 1/10-fools ordered permutation branching programs of length n, width 2, and alphabet size d, then $s_{\mathrm{INW}}(d, \lambda_1, \ldots, \lambda_\ell) = \Omega(\log n \cdot \log\log(\min\{n, d\}))$.*

Proof. Let $t = \log(\min\{n, d\})$ and fix some constraint $(\lambda_1, \ldots, \lambda_\ell)$ such that $\mathrm{INW}(d, \lambda_1, \ldots, \lambda_\ell)$ 1/10-fools the model.

Claim 4.11. *Every block $(\lambda_i, \ldots, \lambda_{i+t})$ satisfies $\sum_{j=i}^{i+t} \lambda_j < 8$.*

Proof. Note that given $(\lambda_i, \ldots, \lambda_{i+t})$ with $\sum_{j=i}^{i+t} \lambda_i \geq 8$, Lemma 4.10 gives a length $2^t \leq n$, width 2, alphabet size $2^t \leq d$ permutation branching program B and a family of auxiliary graphs $\mathcal{H} = (H_i, \ldots, H_{i+t})$ satisfying the constraint such that $\mathrm{INW}_{\mathcal{H}}$ fails to 1/10-fool B, so it remains to show how to embed this into a length-n, alphabet size d-construction. If $2^t = n$ (so $d \geq n$) this follows from simply identifying symbol $k \in [d]$ with $k \mod 2^t$. This will change the probability of switching states in a truly random step from $1/n$ to at least $1/2n$, and so after adjusting constant factors does not affect the separation.

Now if $2^t = d \leq n$, we must modify B to be length n and the family of auxiliary graphs to be size ℓ. The new program B' will ignore the final $n - 2^t$ layers, and we can take $H_j = J_*$ for all $\ell \geq j > i + t$. To handle lower levels, we again use the tensor product trick. Letting the first $i - 1$ levels of the modified family be $J_1, \ldots, J_{2^{2^i-1}}$, we modify B such that it only reads the *first* symbol of each block of length 2^i, with identity transitions on all other symbols. For the graphs, take $H_j \leftarrow H_j \otimes J_*$ for all $j \in [i, \ldots, i + t]$. Then letting $\mathcal{H}'$ be the modified family, the distribution of outputs of $\mathrm{INW}_{\mathcal{H}'}$ on symbols $1, 1 + 2^i, \ldots, 1 + 2^i 2^t$ will be identical to that of $\mathrm{INW}_{\mathcal{H}}$ on the original distinguishing construction, so $\mathrm{INW}_{\mathcal{H}'}$ will fail to 1/10-fool the alphabet size d, length n modified program. □

Now as an immediate consequence of the claim we have $\sum_{i=1}^{\ell} \lambda_i \leq 8\ell/t$, so at least 1/2 of the constraints satisfy $\lambda_i < 16/t$ by Markov, so we obtain $s_{\mathrm{INW}}(d, \lambda_1, \ldots, \lambda_\ell) = \Omega(\log(t) \cdot (\log n - \log\log n)) = \Omega(\log n \cdot \log\log(\min(n, d))$ via Lemma 2.7. □

4.3 Branching Programs of Large Width

Theorem 4.12 (Formal Statement of Theorem 1.11)**.** *For every $n = 2^\ell$ and $\lambda_1, \ldots, \lambda_\ell \geq 0$, if $\mathrm{INW}(2, \lambda_1, \ldots, \lambda_\ell)$ 1/3-fools ordered permutation branching programs of length n, width w, and alphabet size 2 with respect to spectral norm, then $s_{\mathrm{INW}}(2, \lambda_1, \ldots, \lambda_\ell) = \Omega(\log n \cdot \log\log(\min\{n, w\}))$.*

To prove this, we introduce notation for distributions over a branching program, using the notation of Reingold Steinke and Vadhan [RSV13].

Definition 4.13. Given a length n, width w, alphabet size d branching program B with transition functions $B_1, \ldots, B_n$, for $t \in [n]$ let $\mathbf{B}_t : [d] \to \mathbb{R}^{w \times w}$ be defined where $\mathbf{B}_t[s]_{i,j} = 1$ if $B_t(i, s) = j$ and 0 otherwise. For $0 \leq i < j \leq n$ let $\mathbf{B}_{i..j}$ be defined as $\mathbf{B}_{i..j}[s_{i+1} \ldots s_j] = \mathbf{B}_{i+1}[s_{i+1}] \cdots \mathbf{B}_j[s_j]$, and let $\mathbf{B} = \mathbf{B}_{0..n}$. For a function $\mathrm{GEN} : [S] \to [d]^n$, define the **distribution of B on** GEN as $\overline{\mathbf{B} \circ \mathrm{GEN}} = \mathbb{E}[\mathbf{B}[\mathrm{GEN}(U_{[S]})]]$.

Note that this definition exactly matches Definition 2.2 when the branching program is equal to G^n for a consistently labeled graph G (with $B_t = G$ for all steps t as in Remark 4.2). We can then define fooling with respect to a norm.

Definition 4.14. Let $\|\cdot\|$ be a norm on $w \times w$ real matrices and $\mathcal{B}$ a set of (n, w) branching programs. We say a function $\mathrm{GEN} : \{0,1\}^s \to [d]^n$ **ε-fools $\mathcal{B}$ with respect to** $\|\cdot\|$ if for every $B \in \mathcal{B}$ we have

$$\left\|\overline{\mathbf{B} \circ \mathrm{GEN}} - \overline{\mathbf{B} \circ U_{[d]^n}}\right\| \leq \varepsilon.$$

To use this definition, we need to select a matrix norm. We will work with two different norms on matrices $\mathbf{A} \in \mathbb{R}^{w \times w}$. Note that throughout the paper, all vectors are *row* vectors. Some examples include:

- $\|\mathbf{A}\|_2 = \max_{x \in \mathbb{R}^w - \{0\}} \|x\mathbf{A}\|_2 / \|x\|_2$. We call this the **spectral norm**, and it is what we obtain bounds against.
- $\|\mathbf{A}\|_1 = \max_{x \in \mathbb{R}^w - \{0\}} \|x\mathbf{A}\|_1 / \|x\|_1 = \max_i \|\mathbf{A}_{i,\cdot}\|$ where $\mathbf{A}_{i,\cdot}$ is the ith row of $\mathbf{A}$.
- $\|\mathbf{A}\|_{\max} = \max_{i,j} |\mathbf{A}_{i,j}|$.

We remark that fooling in ℓ_1 norm is equivalent to the conventional notion of fooling programs with an arbitrary set of accept vertices, and fooling in max-norm is equivalent to fooling programs with a *single* accept vertex. We work with ℓ_1 norm in Appendix B, whereas here we obtain bounds against spectral norm.

We now prove the main lemma for spectral fooling of polynomial width permutation BPs over a binary alphabet.

Lemma 4.15. *For every $n = 2^\ell$ and every constraint $(\lambda_1, \ldots, \lambda_\ell)$ where $\sum_{i=1}^{\ell} \lambda_i > 8$, there is a family of auxilliary graphs $\mathcal{H} = (H_1, \ldots, H_\ell)$ where $\lambda(H_i) \leq \lambda_i$ and a length n, width n, alphabet size 2 permutation branching program B such that the INW generator constructed with $\mathcal{H}$ fails to $1/3$-fool B with respect to spectral norm.*

Proof. Our distinguishing permutation branching program is again a consistently labeled graph, with transitions equal at every layer, as in Remark 4.2. For every $m \in \mathbb{N}$, let C_m be the 2-regular consistently labeled undirected m-cycle and let v_2 be a normalized eigenvector of $\overline{\mathbf{W}}_{C_m}$ with second largest eigenvalue. For an $m \times m$ matrix $\mathbf{A}$, let $\lambda_2(\mathbf{A}) \stackrel{\text{def}}{=} v_2 \mathbf{A} v_2^T$, and recall that $\lambda_2(\overline{\mathbf{W}}_{C_m}) = \cos(2\pi/m) = 1 - \Theta(1/m^2)$. We will apply this definition to matrices $\mathbf{A} \in \mathrm{span}\{\mathbf{I}, \overline{\mathbf{W}}_{C_m}, \overline{\mathbf{W}}_{C_m}^2, \ldots\}$; v_2 is an eigenvector of all these matrices, but it is not always the second eigenvector. Nonetheless, it is convenient for us to measure expansion with respect to v_2. Note that when v_2 is an eigenvector of $\mathbf{A}$, $\lambda_2(\mathbf{A}^k) = \lambda_2(\mathbf{A})^k$ for all k.

Recall that $\lambda_2(\overline{\mathbf{W}}_{C_m}^n) = \lambda_2(\overline{\mathbf{W}}_{C_m})^n = (1 - \Theta(1/m^2))^n$. Now given n, choose $w = \Theta(\sqrt{n})$ to be some integer such that random walks of length n are $1/3$ mixed with respect to λ_2, but walks of length $n/2$ are not. Formally let $w = \min_{m \in \mathbb{N}}(1/9 \leq \lambda_2(\overline{\mathbf{W}}_{C_m}^n) < 1/3)$.

We then observe that $G = C_w$ is reversible, so we apply Lemma 4.5 with G and $(\lambda_1, \ldots, \lambda_\ell)$ and obtain a PRG $\mathrm{INW}_{\mathcal{H}}$ where $\mathcal{H}$ satisfies the constraint.

Intuitively, "wasting" a constant fraction of steps by not making progress on mixing is enough to distinguish INW output from truly random in spectral norm. Since $\lambda_2(\overline{\mathbf{W}}_{C_w}^a) \leq \lambda_2(\overline{\mathbf{W}}_{C_w}^b)$ for all $a \geq b$, we again obtain a lower bound by Lemma 4.5:

$$\lambda_2(\overline{\mathbf{W}_{C_w^n} \circ \mathrm{INW}_{\mathcal{H}}}) \geq \frac{1}{100}\lambda_2(\overline{\mathbf{W}}_{C_w}^n) + \frac{99}{100}\lambda_2(\overline{\mathbf{W}}_{C_w}^{n/4}) = \frac{1}{100}\lambda_2(\overline{\mathbf{W}}_{C_w})^n + \frac{99}{100}\lambda_2(\overline{\mathbf{W}}_{C_w})^{n/4}.$$

But then

$$\begin{aligned}
\left\|\overline{\mathbf{W}_{C_w^n} \circ \mathrm{INW}_{\mathcal{H}}} - \overline{\mathbf{W}_{C_w^n} \circ U_{\{0,1\}^n}}\right\|_2 &\geq v_2(\overline{\mathbf{W}_{C_w^n} \circ \mathrm{INW}_{\mathcal{H}}} - \overline{\mathbf{W}}_{C_w}^n)v_2^T \\
&= \lambda_2(\overline{\mathbf{W}_{C_w^n} \circ \mathrm{INW}_{\mathcal{H}}}) - \lambda_2(\overline{\mathbf{W}}_{C_w}^n) \\
&\geq \frac{1}{100}\lambda_2(\overline{\mathbf{W}}_{C_w})^n + \frac{99}{100}\lambda_2(\overline{\mathbf{W}}_{C_w})^{n/4} - \lambda_2(\overline{\mathbf{W}}_{C_w})^n \\
&\geq \min_{x \in [1/9, 1/3)} (x^{1/4} - x)\frac{99}{100} \\
&\geq .34.
\end{aligned}$$

Where the final line follows from a numerical calculation, so we have the desired separation. □

We can then use this lemma to prove Theorem 4.12.

Theorem 4.12 (Formal Statement of Theorem 1.11)**.** *For every $n = 2^\ell$ and $\lambda_1, \dots, \lambda_\ell \geq 0$, if* $\mathrm{INW}(2, \lambda_1, \dots, \lambda_\ell)$ *1/3-fools ordered permutation branching programs of length n, width w, and alphabet size 2 with respect to spectral norm, then $s_{\mathrm{INW}}(2, \lambda_1, \dots, \lambda_\ell) = \Omega(\log n \cdot \log\log(\min\{n, w\}))$.*

Proof. Let $t = \log(\min\{n, w\})$ and fix some constraint $(\lambda_1, \dots, \lambda_\ell)$ such that $\mathrm{INW}(d, \lambda_1, \dots, \lambda_\ell)$ 1/3-fools the model.

Claim 4.16. *Every block $(\lambda_i, \dots, \lambda_{i+t})$ satisfies $\sum_{j=i}^{i+t} \lambda_j < 8$.*

The proof of this is essentially identical to that of Claim 4.11, except that embedding a width $2^t < w$ branching program into a width w BP is direct. Then as an immediate consequence of the claim we have $\sum_{i=1}^{\ell} \lambda_i \leq 8\ell/t$, so at least 1/2 of the constraints satisfy $\lambda_i < 16/t$ by Markov, so we obtain $s_{\mathrm{INW}}(2, \lambda_1, \dots, \lambda_\ell) = \Omega(\log(t) \cdot (\log n - \log\log n)) = \Omega(\log n \cdot \log\log(\min(n, w))$ via Lemma 2.7. □

5 Dependence on Width

In this section, we prove Theorem 1.8, establishing a lower bound on the seed length as a function of the width of the permutation branching program. Since we prove the INW generator does not even hit the distinguisher, we recall the formal definition of a hitting set generator.

Definition 5.1. Let $\mathcal{F}$ be a class of functions $f : [d]^n \to \{0, 1\}$. An **ε-hitting set generator** (ε-**HSG**) for $\mathcal{F}$ is a function $\mathrm{GEN} : \{0,1\}^s \to [d]^n$ such that for every $f \in \mathcal{F}$ where $\mathbb{E}_{x \leftarrow U_{[d]^n}}[f(x)] > \varepsilon$, there exists $y \in \{0,1\}^s$ such that $f(\mathrm{GEN}(y)) = 1$.

We are now prepared to give the formal statement.

Theorem 5.2 (Formal Statement of Theorem 1.8)**.** *For every $n = 2^\ell$ and $w \leq 2^{n^{.1}}$ and $\lambda_1, \dots, \lambda_\ell \geq 0$, if* $\mathrm{INW}(2, \lambda_1, \dots, \lambda_\ell)$ *is a $1 - w^{-1/8}$-hitting set generator for ordered permutation branching programs (with arbitrary sets of accept vertices) of length n, width w, and alphabet size 2, then $s_{\mathrm{INW}}(2, \lambda_1, \dots, \lambda_\ell) = \Omega(\log n \cdot \log w)$.*

The proof proceeds by showing that any INW PRG must constrain almost all spectral gaps to be at most $1/w^{\Omega(1)}$. To do this, we establish that if there is a constraint where $\lambda_r > 1/w^c$ for some $c > 0$, there is a graph E on $\sqrt{w}$ vertices and a permutation branching program that perfectly distinguishes between a pair of edges in E and random vertices. To enable this to work for all levels of the PRG, we tensor E with a large complete graph. We now state the main lemma.

Lemma 5.3. *There exists $c > 0$ such that for all $\ell \in \mathbb{N}$ and $w \geq w_0$, for every constraint $(\lambda_1, \dots, \lambda_\ell)$ where there exists $r \geq \log\log w$ such that $\lambda_r > 1/w^c$, there is a family of auxiliary graphs $\mathcal{H} = (H_1, \dots, H_r)$ where $\lambda(H_i) \leq \lambda_i$ and a alphabet size 2, width w, length 2^r permutation branching program B such that $\Pr[B(U_{\{0,1\}^{2^r}}) = 1] \geq 1 - w^{-1/8}$ and INW_r fails to hit B.*

To prove this, we define the tensor product of graphs, and recall a basic fact about their expansion, as we will construct the auxiliary graphs via tensoring a small expander with the complete graph.

Definition 5.4. Given a pair of labeled graphs G, H on w_1, w_2 vertices with degrees d_1, d_2 respectively, define the **tensor product** $G \otimes H$ to be the $d_1 d_2$-regular graph on $w_1 w_2$ vertices with neighbor relation $(G \otimes H)[(u, v), (e_1, e_2)] = (G[u, e_1], H[v, e_2])$.

Proposition 5.5 (see e.g. [Vad12, Lemma 4.33])**.** *Let G, H be undirected regular graphs. Then $\lambda(G \otimes H) = \max(\lambda(G), \lambda(H))$.*

We further recall the existence of expanders that are not too dense.

Proposition 5.6 (see e.g. [AR94])**.** *There are global constants $c > 0$ and $v_0 \in \mathbb{N}$ such that for every $S = 2^{8s} \geq v_0$, there is an undirected regular graph E on S vertices such that $\deg(E) < \sqrt{S}$ and $\lambda(E) < 1/S^c$.*

The v_0 requirement can in practice be ignored since in all cases we obtain bounds that are asymptotic in the number of vertices of the graph. Counterintutively, our *lower bound* relies on the existence of expanders with an *upper bound* on their degree.

Proof of Lemma 5.3. Let $2^s = S$ be the power of two satisfying $w^{1/4} \leq S < w^{1/2}$. Let E be the graph on S vertices with $\deg(E) = D$ where $D < \sqrt{S}$ and $\lambda(E) \leq 1/w^c$ obtained from Proposition 5.6, where we choose $w_0 = v_0^4$ to guarantee $S \geq v_0$. Then for some $K = 2^k \geq S$ to be chosen later define

$$H_K = E \otimes J_{K/S}.$$

Given $v \in [K]$ and $(i, j) \in [D] \times [K/S]$, the neighbor relation of H_K decomposes as:

$$H_K[v, (i, j)] = (E[v_s, i], J_{K/S}[v', j])$$

where v_s denotes the s bit prefix of v and v' the $k - s$ bit suffix. Choose $K = 2^{2^{r-1}}$ and let the family be $\mathcal{H} = (J_2, \ldots, J_{2^{2^{r-1}}}, H_K)$.

Claim 5.7. *For all $i \in [1, \ldots, r]$, $\lambda(H_i) \leq \lambda_i$.*

Proof. For all $i \neq r$ we have $\lambda(H_i) = 0 \leq \lambda_i$, and for $i = r$ we have we have from Proposition 5.5

$$\lambda(H_K) = \lambda(E \otimes J_{K/S}) = \lambda(E) \leq 1/w^c < \lambda_r. \qquad \square$$

We now construct a branching program of width $S^2 < w$ and length $2k = 2^r \leq n$ that checks neighbor relations in E. Write the input to INW_r as $\sigma = (x, i, *)$ where $x \in [K]$, $i \in [D]$ and $*$ is all subsequent bits of seed. Then the PRG output can be written as

$$\begin{aligned} \mathrm{INW}_r(\sigma) &= (\mathrm{INW}_{r-1}(x), \mathrm{INW}_{r-1}(H_K[x, (i, *)])) \\ &= (x, H_K[x, (i, *)]) \\ &= (x_s, x', E[x_s, i], J[x', *]]). \end{aligned}$$

Note that from our choice of labeling for the complete graphs, INW_{r-1} is the trivial PRG on 2^{r-1} bits. We can now build a permutation branching program of length 2^r, denoted B, to store the bits of output determined by E. Label the states in each layer as $(v, w) \in [S] \times [S]$ with start state $(0, 0)$. We define the branching program B as follows. Write the input to B as $(x_s, x', y_s, y') \in \{0,1\}^s \times \{0,1\}^{k-s} \times \{0,1\}^s \times \{0,1\}^{k-s}$. Then define the transition function as:

$$B_t((v, w), b) = \begin{cases} (v + b \cdot 2^{t-1}, w) & t \in \{1, \ldots, s\} \\ (v, w + b \cdot 2^{t-k-1}) & t \in \{k+1, \ldots, k+s\} \\ (v, w) & \text{otherwise.} \end{cases}$$

Observe that on input (x_s, x', y_s, y'), B reaches state (x_s, y_s) in the final layer. Then in layer $2k = 2^r$, mark state (v, w) as an accept state if and only if $(v, w) \notin E$.

Claim 5.8. INW_r *is not a $1 - w^{-1/8}$-HSG for B.*

Proof. All states (v, w) in the final layer have equal probability of being reached over $U_{\{0,1\}^{2k}}$. Then since for each $v \in [S]$ there are at most D reject vertices $(v, \cdot)$ and we have $D < \sqrt{S}$ we have $\Pr[B(U_{\{0,1\}^{2^r}}) = 1] > 1 - (S\sqrt{S})/S^2 \geq 1 - w^{-1/8}$. But $B(\mathrm{INW}_r(\sigma))$ reaches final state $(x_s, E[x_s, i])$ which by construction is not an accept vertex for all $\sigma = (x, i, *)$, so INW_r fails to hit B and we have the desired separation. $\square$

Therefore we have a family $\mathcal{H} = (H_1, \ldots, H_r)$ satisfying the constraint and a distinguishing permutation BP with the desired properties. $\square$

We now apply this lemma to prove the theorem.

Proof of Theorem 5.2. Applying Lemma 5.3, every family $\mathrm{INW}(2, \lambda_1, \ldots, \lambda_\ell)$ that is a $1 - w^{-1/8}$-HSG for permutation branching programs of length n, width w and alphabet size 2 must have $\lambda_i < 1/w^{\Omega(1)}$ for all $i \in [\log\log w, \ldots, \ell]$. This follows from the lemma, since given a family of graphs $\mathcal{H} = (H_1, \ldots, H_r)$ and a length 2^r PBP B, we can add $n - 2^r$ identity layers to B, and by taking $H_m = J_*$ for $\ell \geq m > k$ obtain a generator INW_ℓ that satisfies the subsequent constraints and behaves identically to INW_r on B. Thus we obtain $s_{\mathrm{INW}}(2, \lambda_1, \ldots, \lambda_\ell) = \Omega(\log w(\log n - 2\log\log(nw) - \log\log(w))$ via Lemma 2.7, which simplifies to $\Omega(\log n \cdot \log w)$ with the assumption $w \leq 2^{n^{.1}}$. □

6 Acknowledgements

We thank Ronen Shaltiel for asking a question at ITCS 2021 that prompted us to write this paper. S.V. thanks Omer Reingold and Luca Trevisan for discussions many years ago that provided some of the ideas in this paper, in particular the tensor product construction used in the proof of Theorem 1.8 and the probabilistic existence proof in Theorem 1.7.

References

[AR94] Noga Alon and Yuval Roichman. Random Cayley graphs and expanders. *Random Structures & Algorithms*, 5(2):271–284, 1994.

[BCG18] Mark Braverman, Gil Cohen, and Sumegha Garg. Hitting sets with near-optimal error for read-once branching programs. In Ilias Diakonikolas, David Kempe, and Monika Henzinger, editors, *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2018, Los Angeles, CA, USA, June 25-29, 2018*, pages 353–362. ACM, 2018.

[BNS92] László Babai, Noam Nisan, and Márió Szegedy. Multiparty protocols, pseudorandom generators for logspace, and time-space trade-offs. *Journal of Computer and System Sciences*, 45(2):204–232, 1992. Twenty-first Symposium on the Theory of Computing (Seattle, WA, 1989).

[BRRY10] Mark Braverman, Anup Rao, Ran Raz, and Amir Yehudayoff. Pseudorandom generators for regular branching programs. In *FOCS* [IEE10], pages 40–47.

[BV10] Joshua Brody and Elad Verbin. The coin problem and pseudorandomness for branching programs. In *FOCS* [IEE10], pages 30–39.

[CDR+21] Gil Cohen, Dean Doron, Oren Renard, Ori Sberlo, and Amnon Ta-Shma. Error Reduction for Weighted PRGs Against Read Once Branching Programs. In Valentine Kabanets, editor, *36th Computational Complexity Conference (CCC 2021)*, volume 200 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 22:1–22:17, Dagstuhl, Germany, 2021. Schloss Dagstuhl – Leibniz-Zentrum für Informatik.

[CL20] Eshan Chattopadhyay and Jyun-Jie Liao. Optimal error pseudodistributions for read-once branching programs. In Shubhangi Saraf, editor, *35th Computational Complexity Conference, CCC 2020, July 28-31, 2020, Saarbrücken, Germany (Virtual Conference)*, volume 169 of *LIPIcs*, pages 25:1–25:27. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020.

[De11] Anindya De. Pseudorandomness for permutation and regular branching programs. In *IEEE Conference on Computational Complexity*, pages 221–231. IEEE Computer Society, 2011.

[GW97] Oded Goldreich and Avi Wigderson. Tiny families of functions with random properties: A quality-size trade-off for hashing. *Random Structures & Algorithms*, 11(4):315–343, 1997.

[HHR11] Iftach Haitner, Danny Harnik, and Omer Reingold. On the power of the randomized iterate. *SIAM Journal on Computing*, 40(6):1486–1528, 2011.

[Hoz21] William Hoza. Better pseudodistributions and derandomization for space-bounded computation. ECCC preprint TR21-019, 2021.

[HPV21] William M. Hoza, Edward Pyne, and Salil P. Vadhan. Pseudorandom generators for unbounded-width permutation branching programs. In James R. Lee, editor, *12th Innovations in Theoretical Computer Science Conference, ITCS 2021, January 6-8, 2021, Virtual Conference*, volume 185 of *LIPIcs*, pages 7:1–7:20. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021.

[HVV06] Alexander Healy, Salil Vadhan, and Emanuele Viola. Using nondeterminism to amplify hardness. *SIAM Journal on Computing*, 35(4):903–931 (electronic), 2006.

[IEE10] IEEE. *51th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2010, October 23-26, 2010, Las Vegas, Nevada, USA*. IEEE Computer Society, 2010.

[Ind06] Piotr Indyk. Stable distributions, pseudorandom generators, embeddings, and data stream computation. *Journal of the ACM*, 53(3):307–323, 2006.

[INW94] Russell Impagliazzo, Noam Nisan, and Avi Wigderson. Pseudorandomness for network algorithms. In *Proceedings of the Twenty-Sixth Annual ACM Symposium on the Theory of Computing*, pages 356–364, Montréal, Québec, Canada, 23–25 May 1994.

[KNP11] Michal Koucký, Prajakta Nimbhorkar, and Pavel Pudlák. Pseudorandom generators for group products: extended abstract. In Lance Fortnow and Salil P. Vadhan, editors, *STOC*, pages 263–272. ACM, 2011.

[KNR05] Eyal Kaplan, Moni Naor, and Omer Reingold. Derandomized constructions of k-wise (almost) independent permutations. In *Proceedings of the 8th International Workshop on Randomization and Computation (RANDOM '05)*, number 3624 in Lecture Notes in Computer Science, pages 354 – 365, Berkeley, CA, August 2005. Springer.

[MRT19] Raghu Meka, Omer Reingold, and Avishay Tal. Pseudorandom generators for width-3 branching programs. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*, pages 626–637. ACM, 2019.

[Nis92] Noam Nisan. Pseudorandom generators for space-bounded computation. *Combinatorica*, 12(4):449–461, 1992.

[NZ96] Noam Nisan and David Zuckerman. Randomness is linear in space. *Journal of Computer and System Sciences*, 52(1):43–52, February 1996.

[PV21] Edward Pyne and Salil Vadhan. Pseudodistributions That Beat All Pseudorandom Generators (Extended Abstract). In Valentine Kabanets, editor, *36th Computational Complexity Conference (CCC 2021)*, volume 200 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 33:1–33:15, Dagstuhl, Germany, 2021. Schloss Dagstuhl – Leibniz-Zentrum für Informatik.

[Rei08] Omer Reingold. Undirected connectivity in log-space. *Journal of the ACM*, 55(4):Art. 17, 24, 2008.

[RSV13] Omer Reingold, Thomas Steinke, and Salil Vadhan. Pseudorandomness for regular branching programs via Fourier analysis. In Sofya Raskhodnikova and José Rolim, editors, *Proceedings of the 17th International Workshop on Randomization and Computation (RANDOM '13)*, volume 8096 of *Lecture Notes in Computer Science*, pages 655–670. Springer-Verlag, 21–23 August 2013. Full version posted as ECCC TR13-086 and arXiv:1306.3004 [cs.CC].

[RV05] Eyal Rozenman and Salil Vadhan. Derandomized squaring of graphs. In *Proceedings of the 8th International Workshop on Randomization and Computation (RANDOM '05)*, number 3624 in Lecture Notes in Computer Science, pages 436–447, Berkeley, CA, August 2005. Springer.

[Siv02] D. Sivakumar. Algorithmic derandomization via complexity theory. In *Proceedings of the Thirty-Fourth Annual ACM Symposium on Theory of Computing*, pages 619–626 (electronic), New York, 2002. ACM.

[Ste12] Thomas Steinke. Pseudorandomness for permutation branching programs without the group theory. Technical Report TR12-083, Electronic Colloquium on Computational Complexity (ECCC), July 2012.

[Vad12] Salil P Vadhan. Pseudorandomness. *Foundations and Trends® in Theoretical Computer Science*, 7(1–3):1–336, 2012.

A General Branching Programs

We note that our approach to Theorem 5.2 gives stronger results for fooling general (e.g. non-regular) ordered branching programs, and recovers the analysis of Brody and Verbin for bounds against width-3 OBPs. Both results use the error amplification technique of Brody and Verbin [BV10], and we include them only as a note.

Theorem A.1. *For every $n = 2^\ell$ and $3 \le w \le 2^{n^{.05}}$ and $\lambda_1, \dots, \lambda_\ell \ge 0$, if $\mathrm{INW}(2, \lambda_1, \dots, \lambda_\ell)$ is a $1 - o_{nw}(1)$-HSG against ordered branching programs of length n, width w, and alphabet size 2, then $s_{\mathrm{INW}}(2, \lambda_1, \dots, \lambda_\ell) = \Omega(\log n \cdot \log(nw))$.*

To prove the theorem, we boost the error of Lemma 5.2 and show how to use the edge checking argument for $w = 3$, if the distinguishing program does not have to be a permutation BP.

Corollary A.2. *There exists $c > 0$ such that for all $n = 2^\ell \in \mathbb{N}$ and $w \ge w_0$, for every constraint $(\lambda_1, \dots, \lambda_\ell)$ where there exists $\ell/2 \ge r \ge \log\log w$ such that $\lambda_r > 1/w^c$, there is a family of auxiliary graphs $\mathcal{H} = (H_1, \dots, H_\ell)$ where $\lambda(H_i) \le \lambda_i$ and a length n, width w, alphabet size 2 branching program B such that $\Pr[B(U_{\{0,1\}^n}) = 1] \ge 1 - 2^{-\log(w)\sqrt{n}/8}$ and $\mathrm{INW}_{\mathcal{H}}$ fails to hit B.*

Proof. Let w_0 and c be the same as in Lemma 5.3, and let the family of auxiliary graphs $H_1, \dots, H_r$ and the length 2^r PBP B be those obtained from applying Lemma 5.3 with $r = r$. Furthermore, for $\ell \ge m > r$ take $H_m = J_*$, so it is clear that the family satisfies the entire constraint.

Rather than checking only the bits of input determined by the output of INW_r (and thus neighbor relation in E), we check *all* such sections and compute the OR. We construct an ordered branching program B' of width w computing the function $f : \{0,1\}^n \to \{0,1\}$ defined as

$$f(\sigma) = \bigvee_{i=0}^{n/2^r - 1} B(\sigma_{1+2^r \cdot i \dots 2^r(i+1)}).$$

Once can see that we can construct a branching program computing f with $n/2^r$ successive copies of B, plus a single additional state $\perp$. At the final transition in each copy, all transitions previously leading to accept states are instead sent to $\perp$, and transitions leading to non-accept states are sent to state $(0,0)$. Finally $\perp$ is always wired to itself and marked as the accept vertex in layer n.

Since on input $\mathrm{INW}_{\mathcal{H}}(s)$ each clause computes $B(\mathrm{INW}_r(s_i))$ for some input s_i, we have that $\mathrm{INW}_{\mathcal{H}}$ fails to hit B'. However, truly random input will satisfy each such clause with probability at least $1 - w^{-1/8}$ by Lemma 5.3, the clauses are independent, and there at least $n/2^r \ge \sqrt{n}$ such clauses. Therefore we have

$$\Pr[B'(U_{\{0,1\}^n}) = 1] \ge 1 - 2^{-\log(w)\sqrt{n}/8}.$$

So we obtain a super-constant (with respect to n and w) HSG separation. □

We can further modify the idea to obtain an $\Omega(\log^2 n)$ lower bound, even for *width*-3 OBPs. The method is similar to the (sketched) proof in [BRRY10] that ordered branching programs can distinguish coins slightly biased towards 1, and is essentially the argument of Brody and Verbin [BV10].

Lemma A.3. *There exists $c > 0$ such that for all $n = 2^\ell \in \mathbb{N}$, for every constraint $(\lambda_1, \dots, \lambda_\ell)$ where there exists $\ell/4 \ge r \ge \log\log n$ such that $\lambda_r > 1/n^c$, there is a family of auxiliary graphs $\mathcal{H} = (H_1, \dots, H_\ell)$ where $\lambda(H_i) \le \lambda_i$ and a length n, width 3, alphabet size 2 branching program B such that $\Pr[B(U_{\{0,1\}^n}) = 1] \ge 1 - 2^{-\sqrt{n}}$ and $\mathrm{INW}_{\mathcal{H}}$ fails to hit B.*

Proof. Let $2^s = S$ be the power of two satisfying $n^{1/8} \leq S < n^{1/4}$. Let E be the graph on S vertices with $\deg(E) = D$ where $D < \sqrt{S}$ and $\lambda(E) \leq 1/n^c$ obtained from Proposition 5.6, where we assume $S \geq v_0$ since the result is asymptotic in n. Then for some $K = 2^k \geq S$ to be chosen later define

$$H_K = E \otimes J_{K/S}.$$

Given $v \in [K]$ and $(i,j) \in [D] \times [K/S]$, the neighbor relation of H_K decomposes as:

$$H_K[v,(i,j)] = (E[v_s, i], J[v', j])$$

where v_s denotes the s bit prefix of v and v' the $k-s$ bit suffix. Choose $K = 2^{2^{r-1}}$ and let the family be $\mathcal{H} = (J_2, \ldots, J_{2^{2^{r-1}}}, H_K, J_*, \ldots, J_*)$. Verifying that $\mathcal{H}$ satisfies the constraint is identical to the proof of Lemma 5.3.

Then choose some $(x,y) \in [S] \times [S]$ such that $(x,y) \notin E$ and create the length n, width 3 branching program B computing the function $f : \{0,1\}^n \to \{0,1\}$ defined as

$$f(\sigma) = \bigvee_{i=0}^{n/2^r - 1} (\sigma_{2^r \cdot i + 1 \ldots 2^r \cdot i + s} = x \wedge \sigma_{2^r \cdot i + k + 1 \ldots 2^r \cdot i + k + s} = y).$$

Since (x,y) is chosen such that it will never be output by $\mathrm{INW}_{\mathcal{H}}$ in the bits of output corresponding to neighbor relations in E, and these bits are precisely those checked by B, we have that $\mathrm{INW}_{\mathcal{H}}$ fails to hit B. However, a truly random input satisfies each clause with probability $(1 - 2^{-2s}) = (1 - 2^{-\log(n)/2})$ and since there are $n/2^r \geq n^{3/4}$ such clauses, the accept probability is at least

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

So we obtain a super-constant HSG separation. □

We can then prove the theorem:

Proof of Theorem A.1. Applying Corollary A.2 and Lemma A.3, every family $\mathrm{INW}(2, \lambda_1, \ldots, \lambda_\ell)$ that $\min(1 - 2^{-\log(w)\sqrt{n}/8}, 1 - \exp(-n^{1/4}))$-fools ordered branching programs of length n, width $w \geq 3$ and alphabet size 2 must have $\lambda_i < \min(1/w^{\Omega(1)}, 1/n^{\Omega(1)})$ for all $i \in [\log\log w, \ldots, \ell/4]$, so we obtain $s_{\mathrm{INW}}(2, \lambda_1, \ldots, \lambda_\ell) = \Omega(\log^2 n + \log w(\log(n)/4 - 2\log\log(nw) - \log\log(w))$ via Lemma 2.7, which simplifies to $\Omega(\log^2 n + \log n \cdot \log(w))$ with the assumption $w \leq 2^{n^{.05}}$. □

B Existence of Optimal INW Generators

Despite spectral analysis of INW PRGs already reaching lower bounds in multiple cases, and clearly being incapable of giving a full derandomization of space bounded computation, there *exists* an INW PRG with asymptotically optimal seed length.

Theorem B.1. *For all d, w and $n = 2^\ell \in \mathbb{N}$, there is a family of graphs $\mathcal{H} = (H_1, \ldots, H_\ell)$ such that $\mathrm{INW}_{\mathcal{H}}$ ε-fools ordered branching programs of length n, width w and alphabet size d, and this generator has seed length $O(\log(nwd/\varepsilon))$.*

The proof is slightly more involved than showing a random function is a PRG with high probability, but proceeds using essentially the same idea. For the first $O(\log\log(nwd/\varepsilon))$ levels, we construct the generator with complete graphs (i.e. the trivial PRG on $O(\log(nwd/\varepsilon))$ bits). At higher levels, we choose a graph such that the INW generator constructed with this graph is a good approximation of the concatenation of two lower level generators over every branching program.

We require a basic Chernoff bound:

Proposition B.2. *Let $X_1, \ldots, X_r$ be independent $[0,1]$ random variables and let $X = \sum_{i=1}^r X_i$ and $\mu = \mathbb{E}\, X$. Then for $0 \leq \delta \leq 1$,*

$$\Pr[|X - \mu| > \delta\mu] \leq 2\exp(-\delta^2\mu/3).$$

We first prove a lemma on the existence of good graphs for all sufficiently high levels of the INW generator. We view the previous levels of the PRG simply as a fixed function with sufficient seed length, and use the probabilistic method to find a *one*-outregular digraph such that the INW generator constructed with this graph (and the fixed function as a base) approximates the concatenation of two copies of the fixed function.

Lemma B.3. *For every $d, w, n \in \mathbb{N}$ and $\varepsilon > 0$, there is $L = \Theta(\mathrm{poly}(ndw/\varepsilon))$ such that given a fixed function $F : [S] \to [d]^r$ with $S \geq L$ and $m \leq n/2$, there exists a 1-outregular digraph H on S vertices such that, defining $F'(x) = (F(x), F(H[x, 0]))$ and $(F, F)(x, y) = (F(x), F(y))$, for every alphabet size d, width w, length $2m$ ordered branching program $\mathbf{B}$ we have*

$$\left\| \overline{\mathbf{B} \circ F'} - \overline{\mathbf{B} \circ (F, F)} \right\|_1 \leq \varepsilon/n.$$

Proof. Let $U_{\mathcal{G}}$ be the uniform distribution over 1-outregular graphs on S vertices where for each vertex $s \in [S]$ we include an edge (s, z) where $z \leftarrow U_{[S]}$.

Now fix an arbitrary ordered branching program B of width w and length $2m$, an arbitrary start state v, and an arbitrary set of accept vertices A. If $\alpha \stackrel{\text{def}}{=} \sum_{j \in A} (\overline{\mathbf{B} \circ (F, F)})_{v,j} < 1/2$ take $A \leftarrow A^c$. This is without loss of generality, since an additive approximation of the reject probability implies an equal approximation of the accept probability. We will define random variables on the space of $1-$outregular digraphs H on S vertices in terms of this tuple (B, v, A).

For all $s \in [S]$, define the random variable X_s (as a function of H) as

$$X_s = \sum_{j \in A} \mathbf{B}[(F(s), F(H[s, 0]))]_{v,j}$$

i.e. X_s evaluates to one if and only if $(F(s), F(H[s, 0]))$ causes B to reach a vertex in A from start vertex v. Recalling the definition of F' given a fixed H, we have

$$\mathop{\mathbb{E}}_{s}[X_s] = \sum_{j \in A} (\overline{\mathbf{B} \circ F'})_{v,j}.$$

Furthermore, we have that $X_s \in [0, 1]$, and the X_s are independent for all s. Using that the neighbor of every vertex s is distributed uniformly over the set of graphs:

$$\begin{aligned}
\mathop{\mathbb{E}}_{H}\left[\mathop{\mathbb{E}}_{s}[X_s]\right] &= \mathop{\mathbb{E}}_{s}\left[\mathop{\mathbb{E}}_{H}[X_s]\right] \\
&= \mathop{\mathbb{E}}_{s}\left[\sum_{j \in A} \mathbf{B}[(F(s), F(U_{[S]}))]_{v,j}\right] \\
&= \sum_{j \in A} \mathbf{B}[(F(U_{[S]}), F(U_{[S]}))]_{v,j} \\
&= \sum_{j \in A} (\overline{\mathbf{B} \circ (F, F)})_{v,j} \\
&= \alpha
\end{aligned}$$

So showing there is a graph H such that $|\mathbb{E}_s[X_s] - \alpha| \leq \varepsilon$ for all tuples (B, v, A) is precisely equivalent to the desired ℓ_1 approximation. We now apply the probabilistic method to show there exists such a good H. Recalling Proposition B.2 and rearranging, we obtain:

$$\Pr_{H}\left[\left|\mathop{\mathbb{E}}_{s}[X_s] - \alpha\right| > \delta\alpha\right] \leq 2\exp(-\delta^2 S\alpha/3)$$

Then since $\alpha \in [1/2, 1]$ we bound our additive error by ε by choosing $\delta = \varepsilon$. Then the bound becomes:

$$\Pr_{H}\left[\left|\mathop{\mathbb{E}}_{s}[X_s] - \alpha\right| > \varepsilon\right] \leq 2\exp(-(\varepsilon^2/6) \cdot S).$$

Note that we obtain an exponential decrease in the failure probability in the *existing* number of seeds. Then for a random H, the probability $\mathbb{E}_s[X_s]$ fails to ε-approximate α for *any* of the $2^{\mathrm{poly}(nwd)}$ tuples (B, v, A) is bounded by $2\exp(\mathrm{poly}(nwd) - (\varepsilon^2/6)\cdot S)$. Taking $L = \mathrm{poly}(nwd/\varepsilon)$ to be sufficiently large and using $S \geq L$, we obtain that the failure probability is strictly below 1, so there is a graph H that is good for all tuples. Letting $F'(x) = (F(x), F(H[x,0]))$, for all $2m, w, d$ branching programs B, $v \in [w]$ and $A \subset [w]$,

$$\left| \sum_{j \in A} (\overline{\mathbf{B} \circ F'})_{v,j} - \sum_{j \in A} (\overline{\mathbf{B} \circ (F,F)})_{v,j} \right| \leq \varepsilon$$

and as both rows are distributions this implies the desired ℓ_1 approximation. □

We can then use this to prove the main theorem.

Theorem B.1. *For all d, w and $n = 2^\ell \in \mathbb{N}$, there is a family of graphs $\mathcal{H} = (H_1, \ldots, H_\ell)$ such that $\mathrm{INW}_{\mathcal{H}}$ ε-fools ordered branching programs of length n, width w and alphabet size d, and this generator has seed length $O(\log(nwd/\varepsilon))$.*

Proof. Let $L = \mathrm{poly}(nwd/\varepsilon)$ be the value guaranteed to exist by Lemma B.3 with the same n, w, d and ε and define r as the smallest integer such that $2^{2^r} \geq L$. For $i \in \{1, \ldots, r\}$ let $H_i = J_{2^{2^{i-1}}}$. Then for $r \leq k \leq \ell$, given $\mathcal{H}_k = (H_1, \ldots, H_k)$ let H_{k+1} be the 1-outregular graph obtained from applying Lemma B.3 with the same n, w, d and ε and $F = \mathrm{INW}_{\mathcal{H}_k}$.

Then fix an arbitrary ordered branching program $\mathbf{B}$ of width w, alphabet size d and length 2^ℓ. For all $x, y \in \{0, \ldots, 2^\ell\}$ and $i \in \{0, \ldots, \ell\}$, let $a = (y-x)/2^i$ and define

$$\mathbf{T}_{x,y,i} = \overline{\mathbf{B}_{x..y} \circ (\underbrace{\mathrm{INW}_i, \ldots, \mathrm{INW}_i}_{a \text{ times}})}.$$

Where $\mathbf{T}_{a,a,i} = \mathbf{I}_w$ and the matrix is not defined if $y - x$ is not divisible by 2^i. Then

$$\begin{aligned}
&\left\| \overline{\mathbf{B} \circ \mathrm{INW}_\ell} - \overline{\mathbf{B} \circ U_{[d]^n}} \right\|_1 \\
&\quad = \left\| \mathbf{T}_{0,n,\ell} - \mathbf{T}_{0,n,0} \right\|_1 \\
&\quad \leq \sum_{i=1}^{\ell} \left\| \mathbf{T}_{0,n,i} - \mathbf{T}_{0,n,i-1} \right\|_1 \\
&\quad \leq \sum_{i=1}^{\ell} \sum_{k=1}^{n/2^i} \left\| \mathbf{T}_{0,2^i(k-1),i} (\mathbf{T}_{2^i(k-1),2^ik,i} - \mathbf{T}_{2^i(k-1),2^ik,i-1}) \mathbf{T}_{2^ik,n,i-1} \right\|_1 \\
&\quad \leq \sum_{i=1}^{\ell} \sum_{k=1}^{n/2^i} \left\| \mathbf{T}_{0,2^i(k-1),i} \right\|_1 \cdot \left\| \mathbf{T}_{2^i(k-1),2^ik,i} - \mathbf{T}_{2^i(k-1),2^ik,i-1} \right\|_1 \cdot \left\| \mathbf{T}_{2^ik,n,i-1} \right\|_1 \\
&\quad = \sum_{i=1}^{\ell} \sum_{k=1}^{n/2^i} \left\| \mathbf{T}_{2^i(k-1),2^ik,i} - \mathbf{T}_{2^i(k-1),2^ik,i-1} \right\|_1 \\
&\quad = \sum_{i=1}^{\ell} \sum_{k=1}^{n/2^i} \left\| \overline{\mathbf{B}_{2^i(k-1)..2^ik} \circ \mathrm{INW}_i} - \overline{\mathbf{B}_{2^i(k-1)..2^ik} \circ (\mathrm{INW}_{i-1}, \mathrm{INW}_{i-1})} \right\|_1 \\
&\quad \leq \sum_{i=1}^{\ell} \frac{n}{2^i} \cdot \frac{\varepsilon}{n} \\
&\quad \leq n \frac{\varepsilon}{n}
\end{aligned}$$

Where the penultimate line follows from Lemma B.3. Since $\mathbf{B}$ was arbitrary we obtain that $\mathrm{INW}_{\mathcal{H}}$ is an ε-PRG for the class with respect to $\|\cdot\|_1$ norm, which implies the conventional notion of fooling. The seed length is clearly $\log(L) = O(\log(nwd/\varepsilon))$. □

C Seed Length Lower Bound

We include a proof of the lower bound for seed length for pseudorandom generators against permutation branching programs for completeness.

Proposition C.1. *Given $n, d \in \mathbb{N}$ with d even and $\varepsilon > 0$, let $G : \{0,1\}^s \to [d]^n$ be an ε-PRG for permutation branching programs of length n and alphabet size d with a single accept vertex. Then $s = \Omega(\log(nd/\varepsilon))$, provided $2^{-n} \leq \varepsilon \leq 1/3$. Furthermore, the same lower bound holds if G is an ε-spectral PRG.*

Proof. First, note that for any matrix $M \in \mathbb{R}^{w \times w}$, we have $\|M\|_{\max} \leq \|M\|_2$. From this, we obtain that lower bounds against δ-fooling a program with a single accept vertex imply the equivalent lower bounds against δ-fooling in spectral norm, since

$$\delta \leq \left(\overline{\mathbf{B} \circ G} - \overline{\mathbf{B} \circ U_n}\right)_{v_0, v_{\text{acc}}} \leq \left\|\overline{\mathbf{B} \circ G} - \overline{\mathbf{B} \circ U_n}\right\|_2.$$

For the bounds against n and ε, let $v : [d] \to \{0,1\}$ be a balanced function.

1. If $s \leq \log(d) - 1$, there are at most $d/2$ distinct first symbols output by G. Then there exists a width-2 permutation branching program B with a single accept vertex that accepts input σ if and only if σ_1 is not output by G. This program satisfies $\Pr[B(U_{[d]^n}) = 1] \geq 1/2$ but $\Pr[B(G(U_s)) = 1] = 0$ by construction, a contradiction.

2. If $s \leq \log(1/\varepsilon) - 1$, since $s \leq n - 1$ (where we use the assumption $\varepsilon \geq 2^{-n}$) there is some set of coordinates $S \subseteq [n]$ such that

   $$|\{x : \oplus_{i \in S} v(G(x)_i) = 1\}| \neq |\{x : \oplus_{i \in S} v(G(x)_i) = 0\}|,$$

   which follows from the fact that the only 0-biased distribution over all subsets of $\{0,1\}^n$ is U_n. Then there is a width-2 permutation program B that computes

   $$B(\sigma) = \bigoplus_{i \in S} v(\sigma_i),$$

   and we have $\Pr[B(U_{[d]^n}) = 1] = 1/2$ but $|\Pr[B(G(U_s)) = 1] - 1/2| \geq 1/2^s \geq 2\varepsilon$, so G fails to be an ε-PRG.

3. Finally, for the dependence on n we recall the proof of [HPV21]:

   If $2^s \leq n - 1$, there is some nonzero vector $z \in \mathbb{F}_2^n$ such that for every x,

   $$\sum_{i=1}^{n} z_i \cdot v(G(x)_i) = 0.$$

   The function $B(x) = \sum_{i=1}^{n} z_i \cdot b(x_i) = 1 \pmod 2$ can be computed by a width-2 alphabet size-d permutation branching program with a single accept vertex, and $\Pr[B(U_{[d]^n}) = 1] \geq 1/2$, but $\Pr[B(G(U_s)) = 1] = 0$, a contradiction.

□

Finally, we note that by allowing an arbitrary set of accept vertices, we also obtain a lower bound in terms of width:

Proposition C.2. *Given $n, w, d \in \mathbb{N}$ with d even, if $G : \{0,1\}^s \to [d]^n$ is a 1/3-PRG for permutation branching programs of width w with an arbitrary set of accept vertices then $s = \Omega(\log w)$, provided $w \leq 2^n$.*

Proof. Again let $v : [d] \to \{0,1\}$ be a balanced function. If $s \leq \log(w) - 1$, there are at most $w/2$ distinct PRG outputs in the first $\log(w)$ symbols. Then there exists a width-w, length n permutation branching program B computing the function

$$B(\sigma) = \sum_{i=1}^{\log w} 2^{i-1} \cdot v(\sigma_i)$$

where we use $\log w \leq n$. By marking states in the final layer as accepting if they are not reached by an output of G, we obtain $\Pr[B(U_{[d]^n}) = 1] \geq 1/2$, since applying v cannot increase the number of distinct PRG outputs, yet $\Pr[B(G(U_s)) = 1] = 0$, a contradiction. □

ECCC ISSN 1433-8092
https://eccc.weizmann.ac.il