Electronic Colloquium on Computational Complexity, Report No. 184 (2026)

The BRRY Analysis of the INW Pseudorandom Generator is Optimal

William M. Hoza
Department of Computer Science
The University of Chicago
williamhoza@uchicago.edu

Yakov Shalunov
Department of Computer Science
The University of Chicago
yasha@uchicago.edu

Abstract

Braverman, Rao, Raz, and Yehudayoff (SICOMP 2014) showed that there is an explicit pseudorandom generator (PRG) that fools standard-order regular read-once branching programs (ROBPs) with seed length

$$O(\log n \cdot \log\log n + \log n \cdot \log(wd/\varepsilon)),$$

where w is the width of the program, n is the length, d is the alphabet size, and ε is the error of the generator. To prove it, they prove a bound on the error of the INW generator (Impagliazzo, Nisan, and Wigderson, STOC 1994) in terms of the spectral expansion parameters of the expander graphs used to construct the generator. Then they plug in standard explicit constructions of sparse spectral expanders.

In this paper, we prove that Braverman, Rao, Raz, and Yehudayoff's analysis is optimal. That is, if some instantiation of the INW generator fools standard-order regular ROBPs and the proof of correctness doesn't use any properties of the underlying graphs except bounds on their spectral expansion parameters, then the seed length of the generator is at least

$$\Omega(\log n \cdot \log\log n + \log n \cdot \log(wd/\varepsilon)),$$

provided $w \in [6, 2^{n^{0.99}}]$, $\varepsilon \in [2^{-n^{0.99}}, 0.01]$, and $d \leq \text{poly}(n)$. A lower bound of $\Omega(\log n \cdot \log(w/\varepsilon))$ was already known even for the special case of fooling permutation ROBPs (Hoza, Pyne, and Vadhan, Algorithmica 2024). Our contribution is to prove that the $\log n \cdot \log\log n$ and $\log n \cdot \log d$ terms are unavoidable if one wishes to fool regular programs.

ISSN 1433-8092

1 Introduction

Randomness can be considered a computational resource, similar to time or space. A *pseudorandom generator* (PRG), formally defined below, uses a small number of truly random bits to produce a large number of "random enough" bits.

Definition 1.1 (PRGs)**.** A *PRG* is a function $G\colon \Omega \to [d]^n$ for some finite set Ω and some $n, d \in \mathbb{N}$. The parameter $\log|\Omega|$ is called the *seed length* of the generator.[1] Let $f\colon [d]^n \to \{0,1\}$. We say that G *fools* f with error $\varepsilon \in [0,1]$ if

$$\left| \Pr_{x\in[d]^n}[f(x)=1] - \Pr_{z\in\Omega}[f(G(z))=1] \right| \leq \varepsilon.$$

Let $\mathcal{F}$ be a class of functions $f\colon [d]^n \to \{0,1\}$. We say that G fools $\mathcal{F}$ with error ε if G fools every $f \in \mathcal{F}$ with error ε.

In this paper, we study the problem of fooling *standard-order read-once branching programs* (ROBPs). See Definition 2.1. A PRG that fools width-w ROBPs can be used to simulate any randomized $(\log w)$-space decision algorithm without significantly affecting its acceptance probability. A probabilistic argument shows that there exists a PRG that fools width-w length-n ROBPs over the alphabet $[d]$ with error ε and seed length $O(\log(wnd/\varepsilon))$, but a nonexplicit existence argument is not useful for typical applications. There is therefore a huge amount of interest in designing *explicit* PRGs that fool ROBPs.

Arguably, the most important case is when $w = \mathrm{poly}(n)$, $d = 2$, and $\varepsilon = 0.1$. In this regime, the best explicit construction is a classic PRG by Nisan with seed length $O(\log^2 n)$ [Nis92]. For width-2 or width-3 ROBPs, the seed length can be improved to $O(\log n)$ [SZ95; BDVY13; HH24; Kum25] or $\widetilde{O}(\log n)$ [MRT19] respectively, but it remains an open problem to fool width-4 ROBPs with seed length $o(\log^2 n)$.

1.1 Regular ROBPs

We say that an ROBP over the alphabet $[d]$ is *regular* if every vertex has exactly d incoming edges (except those in layer zero, which have zero incoming edges). Braverman, Rao, Raz, and Yehudayoff showed how to fool constant-width regular ROBPs with seed length $O(\log n \cdot \log\log n)$ [BRRY14]:

Theorem 1.2 (Fooling regular ROBPs [BRRY14])**.** *For every $w, n, d \in \mathbb{N}$ and $\varepsilon \in (0,1)$, there exists an explicit PRG that fools width-w length-n regular ROBPs over the alphabet $[d]$ with error ε and seed length*[2]

$$O(\log n \cdot \log\log n + \log n \cdot \log(wd/\varepsilon)).$$

We would like to improve Braverman, Rao, Raz, and Yehudayoff's seed length even further. Lee, Pyne, and Vadhan showed that every width-w length-n ROBP can be simulated by a regular ROBP of width $O(wn)$ [LPV23], so a better PRG for polynomial-width regular ROBPs would also yield improvements for fooling non-regular ROBPs. We also want to improve Braverman, Rao, Raz, and Yehudayoff's result in the constant-width regime. We especially want to improve the $O(\log n \cdot \log\log n)$ term to the optimal value $O(\log n)$. Indeed, there are precious few cases in which we can unconditionally fool some interesting model of computation with seed length $O(\log n)$, and

[1] Typically, we want $\Omega = \{0,1\}^s$ for some $s \in \mathbb{N}$, but in this work we allow non-integer seed lengths.

[2] Braverman, Rao, Raz, and Yehudayoff focused on the case $d = 2$ [BRRY14], but the generalization to larger d is straightforward.

Steinke writes that achieving this milestone for constant-width regular ROBPs over the binary alphabet is a "major objective" [Ste12]. Braverman, Rao, Raz, and Yehudayoff showed that there is an explicit "hitting set generator" for constant-width binary-alphabet regular ROBPs with seed length $O(\log n)$ [BRRY14], but the PRG problem remains open.

1.2 The INW Generator

Let us review Braverman, Rao, Raz, and Yehudayoff's approach, so we can understand where their seed length comes from and what it would take to improve it. The PRG they use to prove Theorem 1.2 is the Impagliazzo-Nisan-Wigderson (INW) PRG [INW94], which is defined recursively as follows.

- For the base case, define $G_0 \colon [d] \to [d]$ by $G_0(x) = x$.
- For the inductive step, suppose we have already defined $G_{i-1} \colon \Omega \to [d]^{2^{i-1}}$. Let H_i be a degree-D *expander graph* on the vertex set Ω. Define $G_i \colon \Omega \times [D] \to [d]^{2^i}$ by the formula $G_i(x, y) = (G_{i-1}(x), G_{i-1}(H_i[x, y]))$, where $H_i[x, y]$ denotes the y-th neighbor of x in H_i.

Let $\lambda(H)$ denote the spectral expansion parameter of a regular graph H, i.e., the second largest eigenvalue of the random walk matrix in absolute value. In their original paper [INW94], Impagliazzo, Nisan, and Wigderson prove that if $\lambda(H_i) \le \lambda$ for every i, then the generator $G_{\log n}$ fools width-w length-n ROBPs over the alphabet $[d]$[3] with error

$$\varepsilon = \lambda \cdot w \cdot n. \tag{1}$$

Therefore, we can achieve a desired error ε by choosing $\lambda = \frac{\varepsilon}{wn}$. One can take H_i to be an explicit graph of degree $D = \mathrm{poly}(1/\lambda)$, in which case $G_{\log n}$ has seed length $\log d + O(\log n \cdot \log(wn/\varepsilon))$.

To improve the seed length for regular programs, Braverman, Rao, Raz, and Yehudayoff improve Eq. (1). They show[4] that $G_{\log n}$ fools width-w length-n regular ROBPs over the alphabet $[d]$ with error

$$\varepsilon = \lambda \cdot \mathrm{poly}(w, d) \cdot \log n, \tag{2}$$

which is better than Eq. (1) if $w, d \ll n$. This improved error bound readily implies the seed length bound stated in Theorem 1.2.

1.3 Permutation ROBPs

There is a line of work showing how to further improve the analysis of the INW generator for the special case of fooling *permutation ROBPs*. A permutation ROBP is a regular ROBP in which at each vertex, all the incoming edges have distinct labels.

- Cohen, Doron, and Goldgraber recently showed that the INW generator $G_{\log n}$ fools width-w permutation ROBPs over the binary alphabet with error

$$\varepsilon = \lambda \cdot \mathrm{poly}(w),$$

[3] Impagliazzo, Nisan, and Wigderson also studied more general models [INW94].

[4] Actually, Braverman, Rao, Raz, and Yehudayoff present the generator and its analysis in terms of *extractors* rather than expander graphs. However, the extraction property they use holds for any expander graph, so their analysis can be understood as an analysis of the original, expander-based INW generator. See Hatami and Hoza's survey [HH24] for a direct, expander-based exposition of their result that does not go through the concept of extractors. Also, as mentioned previously, Braverman, Rao, Raz, and Yehudayoff only analyze the case $d = 2$, but the generalization to larger d is straightforward.

with no dependence on n [CDG26], improving several prior bounds in terms of the dependence on w [KNP11; De11; Ste12]. This implies that the $\log n \cdot \log\log n$ term in Braverman, Rao, Raz, and Yehudayoff's seed length can be improved to $O(\log n)$ in the special case of fooling permutation ROBPs over the binary alphabet.

- Meanwhile, Hoza, Pyne, and Vadhan showed that $G_{\log n}$ fools width-w permutation ROBPs over the alphabet $[d]$ with error
$$\varepsilon = O(\lambda \cdot w \cdot \log n),$$
with no dependence on the alphabet size d [HPV21].[5] This implies that the $\log n \cdot \log d$ term in Braverman, Rao, Raz, and Yehudayoff's seed length can be improved to $\log d$ in the special case of fooling permutation ROBPs.

1.4 Our Contributions: Limitations of the INW Generator

In light of the successful line of work on permutation ROBPs [KNP11; De11; Ste12; HPV21; CDG26], the most natural approach for getting an improved PRG that fools regular ROBPs would be to improve Eq. (2) (Braverman, Rao, Raz, and Yehudayoff's bound on the error of the INW generator in terms of the spectral expansion parameter λ). Our main result says *this approach cannot work*; Eq. (2) is essentially optimal.

More generally, we can consider an analysis of the INW generator that bounds the spectral expansion parameter of each graph H_i separately, say $\lambda(H_i) \le \lambda_i$ for some vector $\vec{\lambda} = (\lambda_1, \dots, \lambda_{\log n})$. For example, Rozenman and Vadhan showed how to use the INW generator to decide undirected connectivity in $O(\log n)$ space [RV05], re-proving Reingold's theorem [Rei08]. For their application of the INW generator, Rozenman and Vadhan use $H_1, \dots, H_{\log n}$ such that $\lambda(H_i)$ is a small constant for most i, and then $\lambda(H_i)$ rapidly decreases for the last few rounds [RV05]. Our main result says that even this kind of "varying-λ" analysis of the INW generator cannot improve Braverman, Rao, Raz, and Yehudayoff's result [BRRY14].

To be more precise, let $\mathsf{INW}(d, \vec{\lambda})$ denote the set of PRGs $G\colon \Omega \to [d]^n$ that can be constructed using the INW template, using graphs $H_1, \dots, H_{\log n}$ satisfying $\lambda(H_i) \le \lambda_i$ for every i. (See Definition 2.2 for details.) Then our main results are as follows.

Theorem 1.3 (The $\log n \cdot \log\log n$ term is unavoidable)**.** *Let n be a power of four and let $d \ge 2$. Let $\vec{\lambda} \in [0,1]^{\log n}$, and assume that every PRG in the set $\mathsf{INW}(d, \vec{\lambda})$ fools width-6 regular ROBPs over the alphabet $[d]$ with error $1/4$. Then $\lambda_i \le 1/(\log n)^{\Omega(1)}$ for $\Omega(\log n)$ values of i, and moreover, every PRG in the set $\mathsf{INW}(d, \vec{\lambda})$ has seed length $\Omega(\log n \cdot \log\log n)$.*

Theorem 1.4 (The $\log n \cdot \log d$ term is unavoidable)**.** *Let n be a power of two and let $2 \le d \le n$. Let $\vec{\lambda} \in [0,1]^{\log n}$, and assume that every PRG in the set $\mathsf{INW}(d, \vec{\lambda})$ fools width-4 regular ROBPs over the alphabet $[d]$ with error 0.01. Then $\lambda_i \le 1/d^{\Omega(1)}$ for all $i \le \frac{1}{2}\log n$, and moreover, every PRG in the set $\mathsf{INW}(d, \vec{\lambda})$ has seed length $\Omega(\log n \cdot \log d)$.*

Previous work by Hoza, Pyne, and Vadhan proves an $\Omega(\log n \cdot \log(w/\varepsilon))$ lower bound, even for the special case of fooling permutation ROBPs [HPV24]. Combining their bounds with ours shows that Braverman, Rao, Raz, and Yehudayoff's analysis of the INW generator [BRRY14] is optimal with respect to all four parameters (n, w, d, and ε):

[5] In fact, they show that this bound holds provided that the program has at most w accepting vertices in the final layer, regardless of the width of the program [HPV21].

Corollary 1.5 (The BRRY analysis of the INW generator is optimal)**.** *Let n be a power of four, let $6 \le w \le 2^{n^{0.99}}$, let $2 \le d \le n$,[6] and let $2^{-n^{0.99}} \le \varepsilon \le 0.01$. Let $\vec{\lambda} \in [0,1]^{\log n}$, and assume that every PRG in the set $\mathsf{INW}(d, \vec{\lambda})$ fools width-w regular ROBPs over the alphabet $[d]$ with error ε. Then every PRG in the set $\mathsf{INW}(d, \vec{\lambda})$ has seed length at least*

$$\Omega(\log n \cdot \log\log n + \log n \cdot \log(wd/\varepsilon)).$$

In summary, improving Braverman, Rao, Raz, and Yehudayoff's seed length will require either using graphs $H_1, \ldots, H_{\log n}$ with special properties beyond spectral expansion, or else deviating from the INW template altogether.

The binary-alphabet case ($d = 2$) is arguably most interesting, so Theorem 1.3 might be considered the more important of our two main results. However, there is a growing awareness that large-alphabet programs are also important, because they often arise at intermediate analysis steps even if the binary case is what we ultimately care about. See, for example, the discussion in Chen, Cohen, Doron, Khaskelberg, and Ta-Shma's recent work [CCDKT26]. We therefore believe that evading our $\Omega(\log n \cdot \log d)$ lower bound (Theorem 1.4) is also an important open problem.

1.5 Proof Overview

1.5.1 The $\Omega(\log n \cdot \log d)$ Lower Bound

The proof of our $\Omega(\log n \cdot \log d)$ lower bound (Theorem 1.4) is relatively simple. Assume for simplicity that $d \ll \sqrt{n}$. We use a width-4, length-2 "gadget program" h over the alphabet $[d]$ (Fig. 2) with the following properties.

- If we pick two inputs $x, y \in [d]$ independently and uniformly at random and feed them into h, the probability that the state changes parity is approximately $\frac{1}{2d}$.
- There is an expander graph $H = ([d], E)$, with $\lambda(H) \le O(1/\sqrt{d})$, so that if we pick an edge $(x, y) \in E$ uniformly at random and feed it into h, the probability that the state changes parity is approximately $\frac{1}{d}$.

To prove Theorem 1.4, we do $\Theta(d)$ independent trials of h to amplify the gap.

1.5.2 The $\Omega(\log n \cdot \log\log n)$ Lower Bound

The proof of our $\Omega(\log n \cdot \log\log n)$ lower bound (Theorem 1.3) is trickier. A key idea is to reason about a regular ROBP f by running the program "backward," from the final layer to the first layer. We start with the set of accepting vertices $V_{\text{acc}} \subseteq V_n$. Then we look at the last input bit x_n and ask, what is the set of vertices in V_{n-1} that have edges leading to V_{acc} labeled with x_n? We continue working our way backward through the program, one step at a time, keeping track of the set of vertices in that layer that lead to accept under x. Finally, we reach some set of vertices $S \subseteq V_0$. If S contains the start vertex, then $f(x) = 1$, and otherwise, $f(x) = 0$. Intuitively, the reason this perspective is useful is that, under the uniform distribution, regular and permutation programs have the same behavior when we run them forward, but not when we run them backward.

To prove the theorem, we construct a program R, of width 6 and length $\Theta(\log\log n)$, with the following properties.

[6] In the regime $d > n$, the optimal seed length is $\Theta(\log d + \log n \cdot \log(wn/\varepsilon))$. Proof: The upper bound is from the original INW paper [INW94]. Hoza, Pyne, and Vadhan prove an $\Omega(\log d + \log n \cdot \log(w/\varepsilon))$ lower bound, even for fooling permutation programs [HPV24]. Finally, the $\Omega(\log^2 n)$ term follows from the full version of Theorem 1.4 (Theorem 3.1 in Section 3).

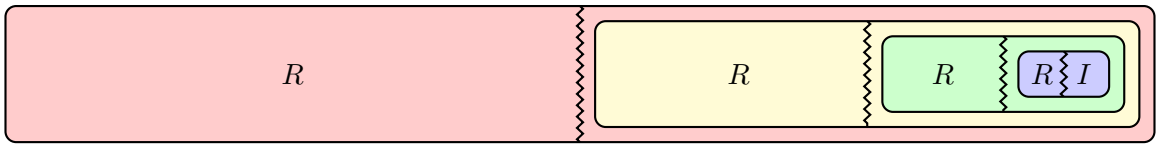

Figure 1: The program f that we use to prove Theorem 1.3. Under the INW generator, the left and right halves of each rounded rectangle are correlated via an expander. We depict only four copies of R in the diagram for simplicity, but in the real construction, the number of copies of R is $\Theta(\log n)$.

- If we start at a specific set of states $B \subseteq [6]$ and run R backward, then with some small probability $\delta \approx 1/(\log n)^{2/3}$, we will arrive at a specific set A. With the same probability δ, we will arrive at A^c. With the remaining probability, we will arrive at B or B^c. The same applies if we start at B^c.
- If we start at A or A^c and run R backward, then with probability $1 - \Theta(\delta)$, we will arrive at B or B^c.
- Finally, if we start at A and run R backward, then with probability $\Theta(\delta)$, we will arrive at $\varnothing$.

Using this program R, we construct our branching program f as depicted in Fig. 1.

On the one hand, suppose we start at B and run f backward under the uniform distribution. The only way we can escape $\{B, B^c, A, A^c\}$ is if some copy of R brings us from B or B^c to A or A^c, which only happens with probability $O(\delta)$, and then the very next copy of R brings us somewhere other than B or B^c, which, again, only happens with probability $O(\delta)$. By the union bound, the probability that this ever happens is $O(\log n \cdot \delta^2) = o(1)$. So with high probability, we end up at one of B, B^c, A, A^c.

On the other hand, we carefully construct an INW generator, with spectral expansion parameter $\lambda = 1/(\log n)^{0.1}$, such that the event of going from B, B^c to A in one copy of R is positively correlated with the event of going from A to $\varnothing$ in the next copy of R. Specifically, the probability of doing both is $\Theta(\lambda\delta) \gg 1/\log n$ instead of $\Theta(\delta^2) \ll 1/\log n$. Over the course of the whole program, these $\Theta(\lambda\delta)$ probabilities accumulate to give an $\Omega(1)$ chance of arriving in the set $\varnothing$. It readily follows that our INW generator does not fool f.

It remains to explain how we construct the program R. We use two constant-size gadgets, g and K. The gadget g (Fig. 3) is a width-6 length-1 program with the following properties.

- If we start at B or B^c and run g backward, we remain in B or B^c.
- If we start at A or A^c and run g backward, there is a 50% chance that we stay in A or A^c, and there is a 50% chance that we move to B or B^c.

Meanwhile, the gadget K (Fig. 4) is a width-6 length-3 program with the following properties.

- If we start at B or B^c and run K backward, we move to A or A^c.
- If we start at A and run K backward, then with constant probability, we arrive at $\varnothing$.

Given these two gadgets, we set $R = g^m K g^m$ for a suitable value $m = \Theta(\log \log n)$.

2 Preliminaries

Definition 2.1 (Branching Programs)**.** A *branching program* (BP) B of length ℓ and width w over alphabet $[d]$ and input length n is a layered graph with $\ell+1$ layers $V_0, \ldots, V_\ell$ which is out-regular of degree d (except vertices in V_ℓ, which have out-degree 0) and where $|V_i| \leq w$ for each i. A vertex $v_{\text{init}} \in V_0$ is designated the initial vertex and a subset $V_{\text{acc}} \subseteq V_\ell$ are labeled as accepting vertices.

Each vertex $v \in V_i$ (for $i < \ell$) is labeled $x_v \in \{x_1, \ldots, x_n\}$ and each outgoing edge is labeled with a distinct symbol from $[d]$. $B : [d]^n \to \{0,1\}$ is evaluated by starting at the initial vertex $v_0 = v_{\text{init}}$ and then performing ℓ transitions from v_i to $v_{i+1} = B[v_i, x_{v_i}]$. We output 1 if $v_\ell \in V_{\text{acc}}$ and 0 otherwise.

A branching-program is called *read-once* if each variable appears at most once in every computation path. We call such a program an ROBP. An ROBP is called *standard-order* if $\ell = n$ and the label for every vertex $v \in V_i$ is x_{i+1}. We denote by $\tau_{i+1,c} : V_i \to V_{i+1}$ for $i < \ell$, $c \in [d]$ the transition map for symbol c at layer i.

A branching program is called *regular* if, as a graph, it is in-regular of degree d (except vertices in V_0, which have in-degree 0). A read-once branching program is called a *permutation* ROBP if $\tau_{i,c}$ is a bijection for each i, c.

Given two branching programs P and Q of the same width, we will denote by PQ the program given by identifying the first layer of Q with the final layer of P. In keeping with the multiplicative notation, P^k will refer to the program P concatenated with itself k times.

We use J_N to refer to the complete graph on N vertices with self-loops and its transition probability matrix, the $N \times N$ matrix with entries $1/N$. Where the number of vertices is unambiguous, we sometimes refer to this graph/matrix just as J.

2.1 The INW generator

Definition 2.2 (INW generator)**.** Given a sequence of d_i-regular graphs $H_i = (V_i, E_i)$ where $i \in [t]$ and $V_i = E_{i-1}$ for all $i > 1$, we define the INW generator

$$\mathsf{GEN}_{H_1,\ldots,H_t} : V_1 \times \prod_{i \in [t]} [d_i] \to V_1^{2^t}$$

recursively by letting $\mathsf{GEN}_\varnothing : V_1 \to V_1$ be the identity function and

$$\mathsf{GEN}_{H_1,\ldots,H_i}(x,y) := (\mathsf{GEN}_{H_1,\ldots,H_{i-1}}(x), \mathsf{GEN}_{H_1,\ldots,H_{i-1}}(H_i[x,y])).$$

We will denote $|V_1|$ by d_0.

Given such a generator GEN, let $s(\mathsf{GEN}) = \log_2(\prod_i d_i)$ be the seed-length of the generator.

We define $\mathsf{INW}(d_0, \lambda_1, \ldots, \lambda_t)$ to be the family of all generators $\mathsf{GEN}_{H_1,\ldots,H_t}$ where $V_1 = [d_0]$ and $\lambda(H_i) \leq \lambda_i$.

We will prove two theorems that say, if the expansion coefficients $\lambda_1, \ldots, \lambda_t$ are large, then we can construct a branching program f and a generator $\mathsf{GEN} \in \mathsf{INW}(d, \vec{\lambda})$ such that GEN does not fool f. Given those theorems, to complete the proofs of our main results, we will need to argue that if the expansion coefficients $\lambda_1, \ldots, \lambda_t$ are small, then every generator in $\mathsf{INW}(d, \vec{\lambda})$ has a large seed length.

Heuristically, the minimum seed length of a generator in $\mathsf{INW}(d, \vec{\lambda})$ should be approximately $2\sum_i \log(1/\lambda_i)$, because this is approximately the seed length when $H_1, \ldots, H_t$ are Ramanujan graphs. The following two lemmas provide formal versions of this heuristic strong enough for our theorems.

Lemma 2.3 ([HPV24])**.** *Given d and $\vec{\lambda} = \lambda_1, \ldots, \lambda_t$, there is a set $S \subseteq [t]$ with $|S| \le 2\log t + 2\log\log(1/\lambda_{\min})$ where $\lambda_{\min} = \min \vec{\lambda}$ such that for all $\mathsf{GEN} \in \mathsf{INW}(d, \vec{\lambda})$,*

$$s(\mathsf{GEN}) \ge \Omega\left(\sum_{i \in [t] \setminus S} \log(1/\lambda_i)\right).$$

Lemma 2.4. *Given d and $\vec{\lambda} = \lambda_1, \ldots, \lambda_t$, if there exists $\varepsilon \in (0,1)$ and a set $I \subseteq [t]$ with $|I| = \alpha t \ge 6\log t$ such that $\sum_{i \in I} \lambda_i \le t^{1-\varepsilon}$ then for all $\mathsf{GEN} \in \mathsf{INW}(d, \vec{\lambda})$, $s(\mathsf{GEN}) \ge \Omega(\alpha\varepsilon \cdot t \log t)$.*

Lemma 2.4 is proven in Appendix A using Lemma 2.3. In our usage, α and ε are constants.

3 Dependence on alphabet size

In this section, we prove Theorem 1.4, which says that the INW generator's seed length must be $\Omega(\log n \cdot \log d)$ assuming $d \le n$. More generally, we prove the following theorem, which is a version of Theorem 1.4 in which d can be arbitrarily large:

Theorem 3.1 (The $\log n \cdot \log d$ term is unavoidable)**.** *Let n be a power of two and let $2 \le d$. Let $\vec{\lambda} \in [0,1]^{\log n}$, and assume that every PRG in the set $\mathsf{INW}(d, \vec{\lambda})$ fools width-4 regular ROBPs over the alphabet $[d]$ with error 0.01. Then $\lambda_i \le 1/\min\{n, d\}^{\Omega(1)}$ for all $i \le \frac{1}{2}\log n$, and moreover, every PRG in the set $\mathsf{INW}(d, \vec{\lambda})$ has seed length $\Omega(\log n \cdot \log \min\{n, d\})$.*

Proof. We will prove the contrapositive. Suppose there is some $i \le \frac{1}{2}\log n$ such that $\lambda_i \ge 8/\sqrt{d}$ and $\lambda_i \ge 8/n^{1/4}$. Under this assumption, we will construct a generator $G \in \mathsf{INW}(d, \vec{\lambda})$ and a width-4 regular program f such that G does not fool f with error 0.01.

Decompose $[d]$ into disjoint sets $[d] = \Sigma_1 \cup \Sigma_2 \cup \Gamma$, where $|\Sigma_1| = |\Sigma_2| = \lfloor d/2 \rfloor$ and $|\Gamma| = d \bmod 2$. Let $s = \lceil \frac{d}{\sqrt{n}} \rceil$, and further decompose $\Sigma_1 = \Sigma_1^{\text{many}} \cup \Sigma_1^{\text{few}}$ and $\Sigma_2 = \Sigma_2^{\text{many}} \cup \Sigma_2^{\text{few}}$, where[7]

$$|\Sigma_1^{\text{many}}| = |\Sigma_2^{\text{many}}| = \lfloor d/2 \rfloor - s$$
$$|\Sigma_1^{\text{few}}| = |\Sigma_2^{\text{few}}| = s.$$

Let h denote the width-4 length-2 gadget depicted in Fig. 2. Let $r = \lfloor \frac{d}{3s} \rfloor$. Our program consists f consists of r consecutive copies of the gadget h, with $2^{i-1} - 1$ identity layers inserted after each layer of each copy of h. The start state is 1, and the accepting states are 2 and 4. This construction is legal, because the total number of layers is

$$r \cdot 2^i \le \frac{\sqrt{n}}{3} \cdot 2^{\frac{1}{2}\log n} < n.$$

Let us upper bound the acceptance probability of f under the uniform distribution. The start state is odd and the accepting states are even. For the parity of the state to change, it is necessary that at least one copy of h reads a symbol in $\Sigma_1 \cup \Gamma$ followed by a symbol in Σ_1^{few}. Therefore, by the union bound,

$$\mathbb{E}[f] \le r \cdot \frac{|\Sigma_1 \cup \Gamma|}{d} \cdot \frac{|\Sigma_1^{\text{few}}|}{d} \le \frac{\lceil d/2 \rceil}{3d} \le \frac{1}{6} + \frac{1}{6d}.$$

[7] This is well-defined, because $8/n^{1/4} \le \lambda_i \le 1$, hence $n \ge 8^4$, which implies $s \le \lfloor d/2 \rfloor$.

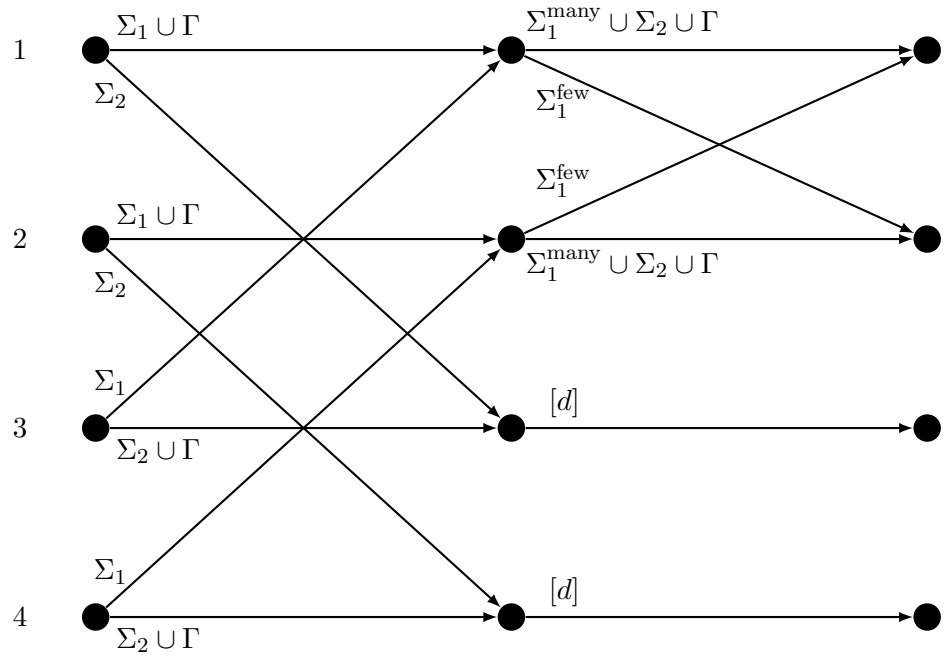

Figure 2: The gadget h. An arrow labeled with a set $S \subseteq [d]$ represents $|S|$ parallel edges, each labeled with a distinct symbol from S.

Now let us construct an INW generator under which f has a significantly higher acceptance probability. Let $u = \mathbf{1}_{\Sigma_1^{\text{many}}} - \mathbf{1}_{\Sigma_2^{\text{many}}}$ and $v = \mathbf{1}_{\Sigma_1^{\text{few}}} - \mathbf{1}_{\Sigma_2^{\text{few}}}$, and let H be the d-vertex graph with the random walk matrix

$$H = J + \frac{uv^\top + vu^\top}{d}.$$

Then H is an undirected d-regular multigraph, because the matrix is symmetric, every row sums to 1, and every entry is 0, $1/d$, or $2/d$. We have $\lambda(H) = \frac{1}{d}\|uv^\top + vu^\top\|_{\text{op}}$. To bound the operator norm, let x be a unit test vector. Since u and v are orthogonal, we can write $x = au + bv + y$ where $\|au\|_2^2 + \|bv\|_2^2 \le 1$ and y is orthogonal to both u and v. Therefore,

$$\begin{aligned}\|(uv^\top + vu^\top)x\|_2^2 = b^2 \cdot (v^\top v)^2 \cdot \|u\|_2^2 + a^2 \cdot (u^\top u)^2 \cdot \|v\|_2^2 &= \|u\|_2^2 \cdot \|v\|_2^2 \cdot (\|au\|_2^2 + \|bv\|_2^2) \\ &\le \|u\|_2^2 \cdot \|v\|_2^2,\end{aligned}$$

so

$$\lambda(H) \le \frac{\|u\|_2 \cdot \|v\|_2}{d} = \frac{2\sqrt{(\lfloor d/2 \rfloor - s) \cdot s}}{d} \le \sqrt{\frac{2s}{d}}.$$

Our INW generator is $\mathsf{GEN}_{H_1,\ldots,H_t}$, where $H_j = J$ for $j \ne i$ and $H_i = H \otimes J$. This generator is in the set $\mathsf{INW}(d, \vec{\lambda})$, because

$$\lambda_i \ge 8 \cdot \max\left\{\frac{1}{\sqrt{d}}, \frac{1}{n^{1/4}}\right\} \ge \cdot\sqrt{64 \cdot \max\left\{\frac{1}{d}, \frac{1}{\sqrt{n}}\right\}} \ge \sqrt{32 \cdot \left(\frac{1}{d} + \frac{1}{\sqrt{n}}\right)} \ge \sqrt{\frac{32s}{d}}. \tag{3}$$

By construction, when f reads a sequence sampled from $\mathsf{GEN}_{H_1,\ldots,H_t}$, each copy of the gadget h reads an independent uniform edge $(x, y) \in E(H)$. If one such trial satisfies $(x, y) \in \Sigma_1 \times \Sigma_1^{\text{few}}$ and

all others satisfy $(x,y) \notin (\Sigma_1 \cup \Gamma) \times \Sigma_1^{\text{few}}$, then the program will accept. For each trial, we have

$$\begin{aligned}\Pr_{(x,y)\in E(H)}[(x,y) \in \Sigma_1 \times \Sigma_1^{\text{few}}] = \frac{\mathbf{1}_{\Sigma_1}^\top H \mathbf{1}_{\Sigma_1^{\text{few}}}}{d} &= \frac{\lfloor d/2 \rfloor \cdot s}{d^2} + \frac{\mathbf{1}_{\Sigma_1}^\top uv^\top \mathbf{1}_{\Sigma_1^{\text{few}}} + \mathbf{1}_{\Sigma_1}^\top vu^\top \mathbf{1}_{\Sigma_1^{\text{few}}}}{d^2} \\ &= \frac{\lfloor d/2 \rfloor \cdot s}{d^2} + \frac{(\lfloor d/2 \rfloor - s) \cdot s + 0}{d^2} \\ &\geq \frac{s}{d} \cdot \left(1 - \frac{2s}{d}\right)\end{aligned}$$

and

$$\Pr_{(x,y)\in E(H)}[(x,y) \in (\Sigma_1 \cup \Gamma) \times \Sigma_1^{\text{few}}] = \frac{\lceil d/2 \rceil \cdot s}{d^2} + \frac{(\lfloor d/2 \rfloor - s) \cdot s}{d^2} \leq \frac{s}{d}.$$

Therefore,

$$\begin{aligned}\mathbb{E}[f \circ \mathsf{GEN}_{H_1,\ldots,H_t}] &\geq r \cdot \frac{s}{d} \cdot \left(1 - \frac{2s}{d}\right) \cdot \left(1 - \frac{s}{d}\right)^{r-1} \\ &\geq \left(\frac{d}{3s} - 1\right) \cdot \frac{s}{d} \cdot \left(1 - \frac{2s}{d}\right) \cdot \left(1 - \frac{rs}{d}\right) \\ &\geq \left(\frac{1}{3} - \frac{s}{d}\right) \cdot \left(1 - \frac{2s}{d}\right) \cdot \frac{2}{3}.\end{aligned}$$

By Eq. (3) and the fact that $\lambda_i \in [0,1]$, we have $s/d \leq 1/32$. Therefore,

$$\mathbb{E}[f] \leq \frac{1}{6} + \frac{1}{6d} \leq \frac{1}{6} + \frac{s}{6d} \leq \frac{1}{6} + \frac{1}{6 \cdot 32} = 0.17187\ldots$$

whereas

$$\mathbb{E}[f \circ \mathsf{GEN}_{H_1,\ldots,H_t}] \geq \left(1 - \frac{1}{16}\right) \cdot \left(\frac{1}{3} - \frac{1}{32}\right) \cdot \frac{2}{3} = 0.18880\ldots$$

Thus, $\mathsf{GEN}_{H_1,\ldots,H_t}$ does not fool f with error 0.01. In other words, if every generator in $\mathsf{INW}(d, \vec{\lambda})$ fools width-4 regular programs with error 0.01, then for every $i \leq \frac{1}{2}\log n$, we have

$$\lambda_i \leq \lambda_* := 8 \max\left\{\frac{1}{\sqrt{d}}, \frac{1}{n^{1/4}}\right\} = \frac{1}{\min\{n,d\}^{\Omega(1)}}.$$

Finally, for the seed length bound, we use Lemma 2.3. Let $\vec{\mu} = (\mu_1, \ldots, \mu_{\log n})$, where $\mu_i = \lambda_*$ for $i \leq \frac{1}{2}\log n$ and $\mu_i = 1$ for $i > \frac{1}{2}\log n$. Then $\mathsf{INW}(d, \vec{\lambda}) \subseteq \mathsf{INW}(d, \vec{\mu})$, and by Lemma 2.3, every generator in $\mathsf{INW}(d, \vec{\mu})$ has seed length at least

$$\Omega(\log n \cdot \log(1/\lambda_*)) = \Omega(\log n \cdot \log \min\{n,d\}). \qquad \square$$

4 Dependence on length

In this section, we prove Theorem 1.3, which says that the INW generator's seed length must be $\Omega(\log n \cdot \log\log n)$ to fool width-6 regular standard-order ROBPs. For convenience, assume $n = 2^{2t}$ for some t. (If not, we can pad with identity layers, picking up a factor of at most 4 in the length.)

We prove Theorem 1.3 using an intermediate result, which says that if a certain collection of expansion parameters has a large sum, then the error can be large. Combining with Lemma 2.4 will complete the proof of Theorem 1.3.

Theorem 4.1. *Given $n = 2^{2t}$, $d \geq 2$, and $\lambda_1, \ldots, \lambda_{2t} < \frac{1}{2}$, if $\sum_{i=1}^{\lfloor t/2 \rfloor} \lambda_{t+2i} \geq (\log n)^{3/4}$, then there exists a width-6 regular standard order ROBP f over the alphabet $[d]$ and a sequence of expanders $\vec{H} = H_1, \ldots, H_{\log n}$ satisfying $\lambda(H_i) \leq \lambda_i$ such that*

$$\left|\mathbb{E}\left[f \circ \mathsf{GEN}_{\vec{H}}\right] - \mathbb{E}[f]\right| > \frac{1}{4}.$$

Proof of Theorem 1.3 given Theorem 4.1. We apply Theorem 4.1 in contrapositive form to $\vec{\lambda}' = \vec{\lambda}/3$. Since $\mathsf{INW}(d, \vec{\lambda}/3) \subseteq \mathsf{INW}(d, \vec{\lambda})$, we conclude that $\sum_{i \in I} \lambda_i/3 < (\log n)^{3/4}$ for

$$I = \{(\log n)/2 + 2i : i \in [(\log n)/4]\}.$$

This in turn implies $\sum_{i \in I} \lambda_i < 3(\log n)^{3/4}$, and for large enough n this is at most $(\log n)^{4/5}$.

Then we apply Lemma 2.4 to $\vec{\lambda}$, with $d = d$, $\alpha = \frac{|I|}{\log n} = \frac{1}{4}$, and $\varepsilon = 1/5 = 1 - 4/5$, to conclude that $s(\mathsf{GEN}) \geq \Omega(\log n \cdot \log\log n)$ for every $\mathsf{GEN} \in \mathsf{INW}(d, \vec{\lambda})$, as desired.

Further, $\lambda_i \geq 0$, there is some $J \subseteq I$ with $|J| \geq |I|/2 = \Omega(\log n)$ such $\lambda_i \leq 6(\log n)^{-1/4} \leq (\log n)^{-1/5}$ for all $i \in J$. □

The rest of this section is dedicated to proving Theorem 4.1.

4.1 Expander lemma

A critical ingredient for our proof is an expander that near-quadratically inflates the probability of a particular event (that a randomly sampled edge spans two given sets) while roughly preserving the probabilities of other events.

Lemma 4.2. *Let V be a finite set and let $S, T \subseteq V$ be independent,*[8] *with $\min(|S|, |T|) = \varepsilon|V|$, $\max(|S|, |T|) = c\varepsilon|V|$, $0 < \varepsilon \leq c\varepsilon \leq 1/3$. There exists a regular graph $H_{S \times T} = (V, E_{S \times T})$ such that:*

- $$\Pr_{(u,v) \in E_{S \times T}}[(u, v) \in S \times T] \geq \varepsilon.$$
- *If S' is disjoint from S and T' is independent of S, then*
$$\Pr_{(u,v) \in E_{S \times T}}[(u, v) \in S' \times T'] \leq O(c \cdot \Pr_{u \in V}[u \in S'] \cdot \Pr_{v \in V}[v \in T']).$$
- *If T' is disjoint from T, then*
$$\Pr_{(u,v) \in E_{S \times T}}[(u, v) \in S \times T'] \leq \Pr_{u \in V}[u \in S] \cdot \Pr_{v \in V}[v \in T'] \cdot (1 + O(\varepsilon)).$$

The proof is somewhat technical and can be found in Appendix B.

Corollary 4.3. *Let V be a finite set and let $S, T \subseteq V$ be independent with ε and c as in Lemma 4.2 and let $\lambda \in [0, 1]$ be rational. Then the graph $H = (V, E)$ corresponding to the transition probability matrix $(1 - \lambda)J + \lambda H_{S \times T}$ is a λ-spectral expander such that:*

- $$\Pr_{(u,v) \in E}[(u, v) \in S \times T] \geq \lambda\varepsilon.$$

[8] That is, if $v \in V$ is uniformly random, then the events $v \in S$ and $v \in T$ are independent.

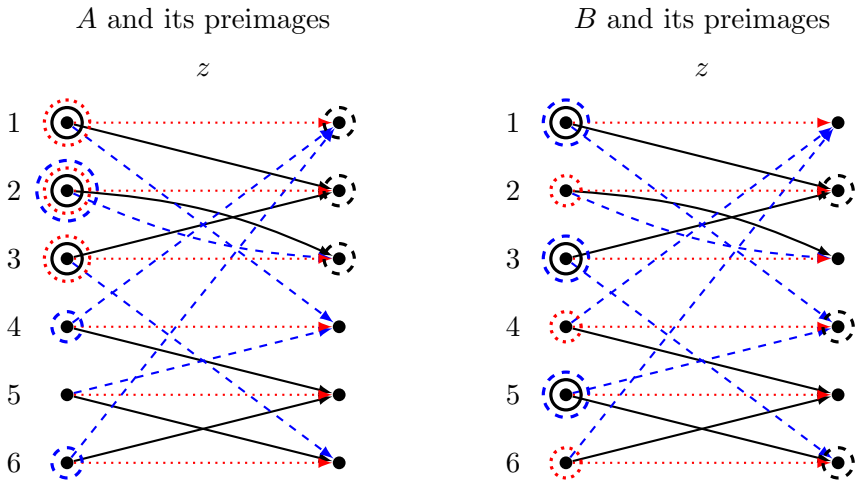

Figure 3: The width-6, one-layer gadget g. Solid black arrows represent symbols in Σ_1, dashed blue arrows represent symbols in Σ_2, and dotted red arrows represent symbols in Γ. For convenience, two copies of g are depicted; the set A and its preimages are highlighted in the left copy of g, while B and its preimages are highlighted in the right copy of g.

- *If S' is disjoint from S and T' is independent of S, then*

$$\Pr_{(u,v)\in E}[(u,v) \in S' \times T'] \leq \Pr_{u\in V}[u \in S'] \cdot \Pr_{v\in V}[v \in T'] \cdot (1 + O(\lambda c)).$$

- *If T' is disjoint from T, then*

$$\Pr_{(u,v)\in E}[(u,v) \in S \times T'] \leq \Pr_{u\in V}[u \in S] \cdot \Pr_{v\in V}[v \in T'] \cdot (1 + O(\lambda \varepsilon)).$$

Proof. This is immediate from Lemma 4.2. □

4.2 Gadget programs

In this section, it will be useful to temporarily forget about start and accept vertices and consider branching programs P of length n and width w instead as functions $P : [d]^n \to [w]^{[w]}$ where $P(x) : [w] \to [w]$ (which we will often write as P_x) describes where initial vertex v ends up on input x. We will write $P_x^{-1} : 2^{[w]} \to 2^{[w]}$ to be the inverse of P_x. As a slight abuse of notation, we will write $P^{-1} : [d]^n \to (2^{[w]})^{2^{[w]}}$ to be the function $x \mapsto P_x^{-1}$, i.e., the "reverse program" of P whose vertices represent sets of vertices in the corresponding layer in P. We define $P(x; v)$ to be $P_x(v)$ and $P^{-1}(x; X)$ to be $P_x^{-1}(X)$.

We will note that (as a general property of inverse functions) $f_x^{-1}(S^c) = (f_x^{-1}(S))^c$.

Let $A = \{1, 2, 3\}$, $B = \{2, 4, 6\}$. Let $\alpha = \lfloor d/2 \rfloor / d \in [1/3, 1/2]$ and $\beta = \lceil d/2 \rceil / d \in [1/2, 2/3]$. Decompose $[d]$ into disjoint sets $[d] = \Sigma_1 \cup \Sigma_2 \cup \Gamma$, where $|\Sigma_1| = |\Sigma_2| = \lfloor d/2 \rfloor$ and $|\Gamma| = d \bmod 2$.

For ease of understanding, it may be helpful for the reader to assume for simplicity that d is even and $\alpha = \beta = 1/2$.

As discussed in Section 1.5, we will construct our branching program f by using many copies of a shorter program R. To build the gadget R, we will need the following two subgadgets.

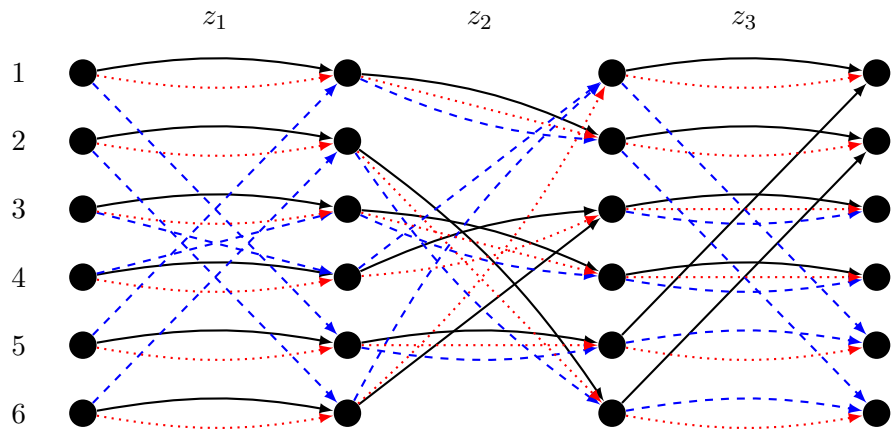

Figure 4: The width-6, three-layer gadget $K = QP$. The first depicted layer is Q, and the last two depicted layers form P. Solid black arrows represent symbols in Σ_1, dashed blue arrows represent symbols in Σ_2, and dotted red arrows represent symbols in Γ.

Lemma 4.4. *For any $\delta \in (0,1)$ and degree d, there exists a width 6, length $O(\log(1/\delta))$ regular ROBP L such that:*

- $L_x^{-1}(B) \in \{B, B^c\}$ *for all* x,
- $L_x^{-1}(A) \in \{A, B, B^c\}$ *for all* x,
- $\Pr_x[L_x^{-1}(A) = A] \in (\delta/3, \delta]$

Proof. Let g be the one-layer program where

$$\tau_x = \begin{cases} (2,3,2,5,6,5) & \text{if } x \in \Sigma_1 \\ (4,3,6,1,4,1) & \text{if } x \in \Sigma_2 \\ \text{id} & \text{if } x \in \Gamma \end{cases}.$$

Then $g^{-1}_{\Sigma_1 \cup \Gamma}(A) = A$, $g^{-1}_{\Sigma_2}(A) = B$, and $g_*^{-1}(B) \in \{B, B^c\}$ (see Fig. 3). Therefore, $g_x^{-m}(A) \in \{A, B, B^c\}$ and $\Pr_x[g_x^{-m}(A) = A] = \beta^m$ (this is the event $x \in (\Sigma_1 \cup \Gamma)^m$). Let $L = g^m$ where $m = \lceil \log_\beta(1/\delta_*) \rceil$ and $\delta = \beta^m$. □

Lemma 4.5. *For any degree d, there exists a width 6, length 3 regular ROBP K such that:*

- $K_x^{-1}(A) = \varnothing$ *for all* $x \in [d] \times \Sigma_2^2$,
- $K_x^{-1}(B) = A$ *for all* $x \in (\Sigma_1 \cup \Gamma) \times [d]^2$, *and*
- $K_x^{-1}(B) = A^c$ *for all* $x \in \Sigma_2 \times [d]^2$.

In particular, $K_x^{-1}(B) \in \{A, A^c\}$ for all x.

Proof. Let $K = QP$ be the width-6, three-layer program depicted in Fig. 4. K consists of a 2-layer program P which has $P^{-1}_{\Sigma_2^2}(A) = \varnothing$ and $P^{-1}_{[d]^2}(B) = A$ preceded by a one-layer program Q which has $Q^{-1}_{\Sigma_1 \cup \Gamma}(A) = A$ and $Q^{-1}_{\Sigma_2}(A) = A^c$. □

Using L and K, we can construct R.

Lemma 4.6. *For any $\delta \in (0,1)$ and degree d, there exists a regular ROBP R of width 6 and length $O(\log(1/\delta))$ such that:*

- $\Pr_x[R_x^{-1}(B) = A], \Pr_x[R_x^{-1}(B) = A^c] \in [\alpha\delta/3, \beta\delta]$ *and* $R_x^{-1}(B) \in \{A, A^c, B, B^c\}$*;*
- $\Pr_x[R_x^{-1}(A) \in \{B, B^c\}] = 1 - \Theta(\delta)$*; and*
- $\Pr[R^{-1}(A) = \varnothing] \in [\alpha^2\delta/3, \delta]$.

Proof. Let $R = LKL$. The first bullet follows from the fact that $\Pr[K^{-1}(B) = A], \Pr[K^{-1}(B) = A^c] \geq \alpha$ and $K^{-1}(B) \in \{A, A^c\}$.

The second bullet and third bullets are immediate. □

4.3 Program and generator construction

To prove Theorem 4.1, we will inductively construct an INW generator G_i and a regular ROBP f_i such that the sum of the λ_i is small (forcing a large seed length), and yet f_i distinguishes the output of G_i from uniform randomness.

Let $\delta_* = t^{-2/3}$ and let $\delta = \Theta(\delta_*)$ and R be as guaranteed by Lemma 4.6. Recall that $n = 2^{2t}$. Let $s_0 = 2^t$. We will recurse for t levels, doubling the length of the generator output and program at each level. We will divide the first s_0 digits of the generator input and generator output into t "blocks," each of size at least $|R| = O(\log(1/\delta))$. Since for sufficiently large n, $O(\log(1/\delta)) = O(\log t) \ll 2^t/t = s_0/t$, such a partition exists. We will write elements of $[d]^{s_0} = \prod_{i=1}^t [d]^{b_i}$ as $(a^{(1)}, \ldots, a^{(t)})$ where $a^{(i)}$ has length b_i.

4.3.1 Program construction

We will define $f_0 : [d]^{s_0} \to [6]^{[6]}$ to be s_0 width-6 identity layers. We will then recursively define f_{i+1} given f_i by constructing an appropriate length-$s_0 \cdot 2^i$, width-6 $f_{\text{left},i}$ and then define $f_{i+1} = f_{\text{left},i} f_i$. For future sections, it will be convenient at times to suppress the dependence on i write $f_{\text{left}} = f_{\text{left},i}$ and $f_{\text{right}} = f_i$.

We define f_{left} as

$$f_{\text{left}}(a^{(1)}, a^{(2)}, \ldots, a^{(t)}, y) = R(a^{(t-i)}).$$

That is, $f_{\text{left}} = I_0 R I_1$ where I_0 and I_1 are identity blocks of appropriate length.

4.4 Generator construction

For the first t rounds of the INW construction, we use the complete graph J as our expander, so after these first t rounds, our "PRG" is simply the identity function on $[d]^{s_0}$. For convenience, we will shift indices, denoting this identity function as $G_0 \colon [d]^{s_0} \to [d]^{s_0}$ rather than G_t. We also shift the indices of the λ_i's correspondingly.

Matching the program construction, we will recurse for t more levels. Let $G_i : V_i \to [d]^{k_i}$ and note that the set of seeds for G_i has the form $V_i = [d]^{s_0} \times U_i$ for some set U_i. By induction, we assume that $G_i \colon [d]^{s_0} \times U_i \to [d]^{k_i}$ has the form

$$G_i(a^{(1)}, a^{(2)}, \ldots, a^{(t)}, e) = (a^{(1)}, a^{(2)}, \ldots, a^{(t-i)}, E_i(a^{(t-i+1)}, \ldots, a^{(t)}, e)). \tag{4}$$

for an appropriate function E_i.

Thus, events of the form $\{x \in V_i : \text{“some condition on } f_{\text{left}}(G_i(x))\text{”}\}$ (which depend only on block $t-i$, $[d]^{[(t-i)\cdot b]\setminus[(t-i-1)\cdot b]} \times U_i$) are independent of events $\{x \in V_i : \text{“some condition on } f_{\text{right}}(G_i(x))\text{”}\}$ (which depend only on the last i blocks, $[d]^{([t\cdot b]\setminus[(t-i)\cdot b])} \times U_i$).

We will define outcomes (over the universe of seeds $x \in V_i$) of $f_i^{-1}(G_i(x); B)$ as follows: $\perp_i$ corresponds to the event that after some instance of R (when applying the program in reverse), we are in a state that is not A, A^c, B, B^c, or $\varnothing$. Note that $\perp$ depends on the whole “trace” of $f_i^{-1}(x; B)$ and not merely on its output. We then define:

$$[S]_i := \{x \in V_i \setminus \perp_i : f_i^{-1}(G_i(x); B) = S\}.$$

We additionally define the sets

$$\mathcal{N}_i = \left\{x \in [d]^{s_0} \times U_i : f_{\text{left}}^{-1}(G_i(x); A) = \varnothing\right\}.$$

Note that $[A]_i$ is a predicate on $f_{\text{right}} \circ G_i$ and $\mathcal{N}_i$ is a predicate on $f_{\text{left}} \circ G_i$, so they are independent.

Our expander H_{i+1} will then be $(1-\lambda_{i+1}) \cdot J + \lambda_{i+1} \cdot H_{[A]_i \times \mathcal{N}_i}$ from Corollary 4.3. G_{i+1} is then defined using the expander H_{i+1} in typical INW fashion as

$$G_{i+1}(x, \ell) = (G_i(x), G_i(H_{i+1}[x, \ell]))$$

The inductive step for Eq. (4) then follows from the fact $[A]_i$ and $\mathcal{N}_i$ are independent of the first $t-i-1$ blocks and so those blocks are uncorrelated with $E_{i+1}(a^{(t-i)}, \ldots, a^{(t)}, e, \ell)$.

4.5 Program and generator joint analysis

In order to prove Theorem 4.1, we will need to show that with each layer of recursion, the likelihood of ending up in the state $\varnothing$ as we move backwards grows quickly. This will then imply that the expectation $f_t \circ G_t$ is small. This is stated as Lemma 4.12 and proven in this subsection.

We will show that $\frac{\delta}{100} \le \Pr[[A]_i], \Pr[\mathcal{N}_i] \le \delta$. Note that $[A]_i$ and $\mathcal{N}_i$ are sets of seeds and these are probabilities over uniformly chosen seeds, not over uniform inputs. The lower bound on $\Pr[[A]_i]$ is shown inductively.

Lemma 4.7. $\Pr[[A]_i \cup [A^c]_i] \le O(\delta)$.

Proof. Entering the left-most R (backward), if we are not in $\perp$, then we are in A, A^c, B, B^c, or $\varnothing$. Therefore, if $\mathcal{A}_i \cup \mathcal{A}_i^c$ occurs, then the left-most R takes at least one of A, A^c, B, or B^c to A or A^c (when it is run backward). The probability of the latter event is $O(\delta)$, because under our INW generator G_i, each copy of R receives a uniform input. □

Lemma 4.8. $\Pr[\perp_i] \le O(i\delta^2)$.

Proof. There are three ways that $\perp_i$ can occur:

- $\perp_{i-1}$ occurs.
- $[A^c]_{i-1}$ occurs and then $f_{\text{left}}^{-1}(A^c)$ causes $\perp_i$. Now we will apply Corollary 4.3 with $\varepsilon = \delta/100$, $c = 100$, $S = [A]_{i-1}$, $T = \mathcal{N}_{i-1}$, $S' = [A^c]_{i-1}$, and $T' = \left\{x \in V_{i-1} : f_{\text{left}}^{-1}(G_{i-1}(x); A^c) \text{ leads to } \perp\right\}$. Note that $\Pr_x[f_{\text{left}}^{-1}(G_{i-1}(x), A^c) \notin \{B, B^c\}]$ (and thus in particular, the probability of T') is at most $O(\delta)$.

  First, we need to establish that the preconditions of the corollary are satisfied. S and T are independent by construction. Observe that S and S' are clearly disjoint, since $f_{i-1}^{-1}(G_{i-1}(x); A)$

cannot be both A and A^c. Further, T' is independent of S since S is a condition on the last $i-1$ blocks of x while T' is a condition on block $t-i$. Finally, $\min(\Pr[[A]_{i-1}], \Pr[\mathcal{N}_{i-1}]) \geq \frac{\delta}{40}$ by induction

Thus, by bullet 2 of the Corollary 4.3,

$$\Pr_{H_{i+1}}[S' \times T'] \leq \Pr[S'] \cdot \Pr[T'] \cdot (1 + O(100\lambda_{i+1})) \leq O(\delta^2).$$

- Finally, we can have $[A]_{i-1}$ occur and then $f_{\text{left}}^{-1}(A)$ causes $\perp_i$. When we apply R backward starting at A, the probability of something other than $A, A^c, B, B^c, \varnothing$ is $O(\delta)$. By bullet 3 of Corollary 4.3, $[A]_{i-1} \times \left\{x \in V_{i-1} : f_{\text{left}}^{-1}(G_{i-1}(x); A) \text{ leads to } \perp\right\}$ occurs with probability at most

$$O(\delta) \cdot O(\delta) \cdot (1 + O(\lambda\delta)) \leq O(\delta^2).$$

Overall, $\Pr[\perp_i] \leq \Pr[\perp_{i-1}] + O(\delta^2)$. □

Lemma 4.9. $\Pr[[B]_i \cup [B^c]_i \cup \varnothing_i] \geq 1 - O(\delta + i\delta^2)$.

Proof. This follows immediately from the previous two lemmas. □

Lemma 4.10. *If* $\Pr[\varnothing_{i-1}] \leq p$ *and* $\lambda_i < \frac{1}{2}$, *then* $\Pr[[A]_i] \geq \delta/18 \cdot (1 - p - O(\delta + i \cdot \delta^2))$.

Proof. If $\Pr[\varnothing_{i-1}] \leq p$ then by Lemma 4.9, $\Pr[[B]_{i-1} \cup [B^c]_{i-1}] \geq 1 - p - O(\delta + i\delta^2)$.

From here, with probability $1 - \lambda_i$ we are in the J case of the expander and the probability of arriving at A is at least $\frac{\delta}{9}$ by construction of R (bullet one of Lemma 4.6).

Thus, with at least probability $(1 - p - O(\delta + i\delta^2)) \cdot (1 - \lambda_i) \cdot \delta/9$, we end up in A. □

The remaining lemmas rely on our specific value of $\delta = \Theta(t^{-2/3})$ and assume t is sufficiently large.

Lemma 4.11. *If* $\lambda_j < 1/2$ *for all* j, $\delta/100 \leq \Pr[[A]_i], \Pr[\mathcal{N}_i] \leq \delta$.

Proof. Since $t = \Theta(\delta^{-3/2}) = o(\delta^{-2})$ and we will only need to recurse until $\Pr[\varnothing] = 4/5$, so this Lemma 4.10 ensures that

$$\Pr[[A]_i] \geq \delta \cdot 1/18 \cdot (1 - 4/5 - o(1)) \geq \delta/100$$

for each i. Further, we observe that $\Pr[\mathcal{N}] \in [\alpha^2\delta, \delta]$ by Lemma 4.6. Together with Lemma 4.7, this gives $\delta \geq \Pr[[A]], \Pr[\mathcal{N}] \geq \frac{\delta}{100}$. □

Lemma 4.12. *Suppose* $\lambda_j < 1/2$ *for all* j. *If* $\Pr[\varnothing_{i-1}] \leq \frac{4}{5}$ *then* $\Pr[\varnothing_{i+1}] \geq \Pr[\varnothing_{i-1}] + \Omega(\lambda_{i+1} \cdot \delta)$.

Proof. By Section 4.5, $\Pr[[A]_i], \Pr[\mathcal{N}_i] \geq \delta/100$. Then by the construction of the expander H_{i+1} via Corollary 4.3, we have that $\Pr[\mathcal{N} \times [A]_i] \geq \frac{1}{100} \cdot \lambda_{i+1} \cdot \delta$. □

4.6 Proof of length-dependence

Lemma 4.13. *There exists $v \in [6]$ such that*

$$\Pr_x[f_t(x; v) \in B] \geq \frac{1}{2} - o(1)$$

Proof. Let $\Omega = \{A, A^c, B, B^c\}$ and enumerate the instances of R as R_1 through R_t from right to left and let $X_i \subseteq [6]$ be the value of $f_i^{-1}(x; B)$ (note that X_i is a random variable) for $i \geq 0$.

Let $\mathcal{R}_{i,X,Y}$ be the event that $X_{i-1} = X \in \{B, B^c\}$, $X_i = Y \in \{A, A^c\}$, and $X_{i+1} \notin \{B, B^c\}$. Let $\mathcal{R}_i = \bigcup_{X,Y} \mathcal{R}_{i,X,Y}$ and let $\mathcal{R}$ be the event that there is some i such that $X_i \notin \Omega$.[9]

Observe that $\mathcal{R} \subseteq \bigcup_i \mathcal{R}_i$: let j be the smallest value such that $X_{j+1} \notin \Omega$. Then $X_j \in \Omega \setminus \{B, B^c\} = \{A, A^c\}$ since $R_x^{-1}(B), R_x^{-1}(B^c) \in \Omega$ for all x. Let $i \leq j$ be the largest such value such that $X_{i-1} \in \{B, B^c\}$ (such an i must exist since $X_0 = B$). Then since $i \leq j$ and i is maximal, $X_i \in \Omega \setminus \{B, B^c\} = \{A, A^c\}$ and since i is maximal, $X_{i+1} \notin \{B, B^c\}$. This is contained in event $\mathcal{R}_i$.

Observe that by a union bound

$$\Pr[\mathcal{R}_i] \leq \sum_{X,Y} \Pr[\mathcal{R}_{i,X,Y}] = \sum_{X,Y} \Pr_x[R_x^{-1}(X) = Y] \cdot \Pr[R_x^{-1}(Y) \notin \{B, B^c\}] = \sum_{X,Y} O(\delta^2) = O(\delta^2)$$

where the second equality holds application of bullet 1 of Lemma 4.6 to the first term and bullet 2 to the second term.

By a second union bound, we conclude that

$$\mathcal{R} \leq \sum_{i=1}^{t} \Pr[\mathcal{R}_i] \leq O(t\delta^2) = O(\delta^{1/2}) = o(1).$$

We now observe that $\Pr_{x,v}[f_t(x; v) \in B | \mathcal{R}_t^c] = \frac{1}{2}$ since if $\mathcal{R}_t$ doesn't occur then $f_t^{-1}(x; B) \in \Omega$ and $|X| = 3$ for all $X \in \Omega$. Since $\Pr[\mathcal{R}_t] \leq \Pr[\mathcal{R}] = o(1)$,

$$\Pr_{x,v}[f_t(x; v) \in B] \geq \frac{1}{2} \cdot (1 - o(1)) = \frac{1}{2} - o(1).$$

Since the best choice must be at least as good as the average choice, there then exists $v \in [6]$ such that $\Pr[f_t(x; v) \in B] \geq \frac{1}{2} - o(1)$. □

Proof of Theorem 4.1. Let $I \subseteq \{t + i : i \in [t], i \text{ even}\}$ be the set of indices summed in the theorem statement. Then, by assumption, $\sum_{i \in I} \lambda_i \geq t^{3/4}$ (recall that $t = 1/2 \cdot \log n$).

We choose $\delta_* = t^{-2/3}$ so δ as guaranteed by Lemma 4.6 satisfies $t = \Theta(\delta^{-3/2})$. Observe then that $\delta \cdot \sum_{i \in I} \lambda_i \geq \delta \cdot t^{3/4} = \omega(1)$ as $t \to \infty$. Applying Lemma 4.12 iteratively, we conclude that for sufficiently large n:

$$\Pr[\varnothing_t] \geq \min\left(\frac{4}{5}, \sum_{i \in I} \Omega(\delta\lambda_i)\right) = \min\left(\frac{4}{5}, \omega(1)\right) = \frac{4}{5}.$$

Let f be the branching program f_t with initial vertex v from Lemma 4.13 and B as the accept set. Note that $f(x) = 1$ if and only if $v \in f_t^{-1}(x; B)$.

[9] This is analogous to the event $\perp_t$, however, $\perp_t$ is an event over *generator seeds* whereas $\mathcal{R}$ is an event over program inputs.

Then

$$\Pr[f(G_t(x)) = 0] = \Pr_x[v \notin f_t^{-1}(G_t(x); B)] \geq \Pr_x[f_t^{-1}(G_t(x); B) = \varnothing] \geq \Pr_x[\varnothing_t] \geq \frac{4}{5}$$

and so $\mathbb{E}[f \circ G_t] \leq \frac{1}{5}$.

Further, by Lemma 4.13, we have that $\mathbb{E}[f] = \Pr_x[f_t(x; v) \in B] \geq 1/2 - o(1)$.

Since $1/2 - 1/5 - o(1) > 1/4$, this ensures that G_t does not 1/4-fool f. By construction, however, G_t is an INW generator with expanders $H_1, \ldots, H_{2t}$ where $H_i = J_{2^i}$ has $\lambda(H_i) = 0 \leq \lambda_i$ for $i \leq t$ and H_{t+i} has the form $(1 - \lambda_{t+i})J + \lambda_{t+i}H'$ and so has $\lambda(H_{t+i}) \leq \lambda_{t+i}$. Thus, $G_t \in \mathsf{INW}(d, \vec{\lambda})$.

□

AI Disclosure

This work was done with significant AI assistance. Most crucially, the width-6 gadget g in the proof of Theorem 1.3 came from ChatGPT 5.5 Pro. ChatGPT 5.6 Pro was later used to assist with a number of minor generalizations and bug corrections. The authors take responsibility for all claims made in the paper.

References

[BDVY13] Andrej Bogdanov, Zeev Dvir, Elad Verbin, and Amir Yehudayoff. "Pseudorandomness for width-2 branching programs". In: *Theory Comput.* 9 (2013), pp. 283–292. DOI: 10.4086/toc.2013.v009a007.

[BRRY14] Mark Braverman, Anup Rao, Ran Raz, and Amir Yehudayoff. "Pseudorandom generators for regular branching programs". In: *SIAM J. Comput.* 43.3 (2014), pp. 973–986. ISSN: 0097-5397. DOI: 10.1137/120875673.

[CCDKT26] Ben Chen, Gil Cohen, Dean Doron, Yuval Khaskelberg, and Amnon Ta-Shma. *Improved Error Reduction for Weighted PRGs*. ECCC preprint TR26-064. 2026. URL: https://eccc.weizmann.ac.il/report/2026/064/.

[CDG26] Gil Cohen, Dean Doron, and Noam Goldgraber. *A Forward-Backward Weight Analysis of INW for Permutation Branching Programs*. ECCC preprint TR26-123. 2026. URL: https://eccc.weizmann.ac.il/report/2026/123/.

[De11] Anindya De. "Pseudorandomness for Permutation and Regular Branching Programs". In: *Proceedings of the 2011 IEEE 26th Annual Conference on Computational Complexity (CCC)*. USA: IEEE Computer Society, 2011, 221–231. ISBN: 9780769544113. DOI: 10.1109/CCC.2011.23.

[HH24] Pooya Hatami and William Hoza. "Paradigms for Unconditional Pseudorandom Generators". In: *Foundations and Trends in Theoretical Computer Science* 16.1-2 (2024), pp. 1–210. ISSN: 1551-305X. DOI: 10.1561/0400000109.

[HPV21] 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)*. 2021, 7:1–7:20. DOI: 10.4230/LIPIcs.ITCS.2021.7.

[HPV24] William M. Hoza, Edward Pyne, and Salil Vadhan. "Limitations of the Impagliazzo-Nisan-Wigderson pseudorandom generator against permutation branching programs". In: *Algorithmica* 86.10 (2024), pp. 3153–3185. ISSN: 0178-4617,1432-0541. DOI: 10.1007/s00453-024-01251-2.

[INW94] Russell Impagliazzo, Noam Nisan, and Avi Wigderson. "Pseudorandomness for network algorithms". In: *Proceedings of the 26th Annual Symposium on Theory of Computing (STOC)*. 1994, 356–364. DOI: 10.1145/195058.195190.

[KNP11] Michal Koucký, Prajakta Nimbhorkar, and Pavel Pudlák. "Pseudorandom Generators for Group Products". In: *Proceedings of the 43rd Symposium on Theory of Computing (STOC)*. 2011, pp. 263–272. DOI: 10.1145/1993636.1993672.

[Kum25] Vinayak M. Kumar. "New Pseudorandom Generators and Correlation Bounds Using Extractors". In: *Proceedings of the 16th Innovations in Theoretical Computer Science Conference (ITCS)*. Vol. 325. 2025, 68:1–68:23. DOI: 10.4230/LIPIcs.ITCS.2025.68.

[LPV23] 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)*. 2023, 44:1–44:22. DOI: 10.4230/LIPIcs.APPROX/RANDOM.2023.44.

[MRT19] 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)*. 2019, pp. 626–637. DOI: 10.1145/3313276.3316319.

[Nis92] Noam Nisan. "Pseudorandom generators for space-bounded computation". In: *Combinatorica* 12.4 (1992), pp. 449–461. ISSN: 0209-9683. DOI: 10.1007/BF01305237.

[Rei08] Omer Reingold. "Undirected connectivity in log-space". In: *J. ACM* 55.4 (2008), Art. 17, 24. ISSN: 0004-5411. DOI: 10.1145/1391289.1391291.

[RV05] Eyal Rozenman and Salil Vadhan. "Derandomized squaring of graphs". In: *Proceedings of the 9th International Workshop on Randomization and Computation (RANDOM)*. 2005, pp. 436–447. DOI: 10.1007/11538462_37.

[Ste12] 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/.

[SZ95] Michael Saks and David Zuckerman. Unpublished. 1995.

A Seed length lemma

Proof of Lemma 2.4. Define $\gamma_i = 1$ if $i \notin I$ and $\gamma_i = \max(\lambda_i, t^{-\varepsilon})$ if $i \in I$. Note that $\lambda_i \le \gamma_i$ and so $\mathsf{INW}(\vec{\lambda}) \subseteq \mathsf{INW}(\vec{\gamma})$.

Now observe that $\gamma_{\min} \ge t^{-\varepsilon}$ by construction, so $\log\log(1/\gamma_{\min}) \le \log\log t^{\varepsilon} = \log\log t + \log\varepsilon$. Applying Lemma 2.3 gives us that the exceptional set has size $S \subseteq [t]$ with $|S| \le 2\log t + 2\log\log t - 2\log(1/\varepsilon) \le 3\log t$.

Let $J := I \setminus S$ and observe that $|J| \ge \alpha t - 3\log t \ge \frac{\alpha}{2} \cdot t$.[10] Since $\max(\lambda_i, t^{-\varepsilon}) \le \lambda_i + t^{-\varepsilon}$, $J \subseteq I$, and $|I| \le t$,

$$\sum_{i \in J} \gamma_i \le \sum_{i \in I} (\lambda_i + t^{-\varepsilon}) \le t^{1-\varepsilon} + t^{1-\varepsilon} = 2t^{1-\varepsilon}$$

[10] $\alpha t \ge 6\log t$ is assumed in the statement of Lemma 2.4.

Now consider $J_{\text{big}} = \left\{i \in J : \gamma_i > \frac{8}{\alpha} \cdot t^{-\varepsilon}\right\}$. Observe then that

$$\frac{8}{\alpha} \cdot t^{-\varepsilon} \cdot |J_{\text{big}}| \leq \sum_{i \in J_{\text{big}}} \gamma_i \leq \sum_{i \in J} \gamma_i \leq 2t^{1-\varepsilon}$$

Thus, $|J_{\text{big}}| \leq \frac{\alpha}{4} \cdot t \leq \frac{1}{2}|J|$. Let $J_{\text{small}} = J \setminus J_{\text{big}}$. Then we have

- $J_{\text{small}} \subseteq [t] \setminus S$ (where S is from Lemma 2.3),
- $|J_{\text{small}}| \geq \frac{\alpha}{4} \cdot t$, and
- $\forall i \in J_{\text{small}}, \gamma_i \leq \frac{8}{\alpha} \cdot t^{-\varepsilon}$.

Thus,

$$\sum_{i \in [t] \setminus S} \log(1/\gamma_i) \geq \sum_{i \in K} \log(1/\gamma_i) \geq \sum_{i \in J_{\text{small}}} \log(8\alpha^{-1}t^{\varepsilon}) \geq \frac{\alpha}{4} \cdot t \cdot \left(\varepsilon \log t + \log \frac{8}{\alpha}\right)$$

Applying Lemma 2.3 gives the desired result. □

B Expander lemma

Proof of Lemma 4.2. Let $N = |V|$ and let

$$C = S \cap T.$$

Since S and T are independent,

$$|C| = \frac{|S||T|}{N} = c\varepsilon^2 N.$$

The two sets $S \setminus T$ and $T \setminus S$ therefore have sizes

$$\varepsilon(1 - c\varepsilon)N \qquad \text{and} \qquad c\varepsilon(1 - \varepsilon)N$$

in some order. Moreover,

$$c\varepsilon(1 - \varepsilon) \geq \varepsilon(1 - c\varepsilon),$$

because $c \geq 1$.

Choose

$$A \subseteq S \setminus T \qquad \text{and} \qquad B \subseteq T \setminus S$$

such that

$$|A| = |B| = \varepsilon(1 - c\varepsilon)N.$$

Thus, one of A and B is the entire exclusive part of the smaller of S and T. Put

$$S_* = A \cup C, \qquad T_* = B \cup C, \qquad D = V \setminus (A \cup B \cup C).$$

Then

$$|S_*| = |T_*| = \varepsilon N, \qquad S_* \cap T_* = C,$$

and

$$\frac{|D|}{N} = 1 - 2\varepsilon + c\varepsilon^2.$$

Define $H_{S \times T}$ as follows:

- From $u \in A$, go to a uniformly random vertex in B.
- From $u \in B$, go to a uniformly random vertex in A.
- From $u \in C$, go to a uniformly random vertex in C.
- From $u \in D$, go to a uniformly random vertex in D.

Every vertex in A transitions into $B \subseteq T$, and every vertex in C transitions into $C \subseteq T$. Since $A \cup C \subseteq S$,

$$\begin{aligned}\Pr_{(u,v)\in E_{S\times T}}[(u,v) \in S \times T] &\geq \Pr_{u\in V}[u \in A \cup C] \\ &= \varepsilon(1 - c\varepsilon) + c\varepsilon^2 \\ &= \varepsilon.\end{aligned}$$

Now suppose that S' is disjoint from S and T' is independent of S. Since $A, C \subseteq S$, every starting vertex in S' belongs to either B or D. Consequently,

$$\Pr_{(u,v)\in E_{S\times T}}[(u,v) \in S' \times T'] = \Pr[S' \cap B] \cdot \frac{|A \cap T'|}{|A|} + \Pr[S' \cap D] \cdot \frac{|D \cap T'|}{|D|}.$$

Since $A \subseteq S$ and S and T' are independent,

$$\begin{aligned}\frac{|A \cap T'|}{|A|} &\leq \frac{\Pr[S \cap T']}{\Pr[A]} \\ &= \frac{\Pr[S]\Pr[T']}{\varepsilon(1 - c\varepsilon)} \\ &\leq \Pr[T'] \cdot \frac{c}{1 - c\varepsilon}.\end{aligned}$$

Also,

$$\frac{|D \cap T'|}{|D|} \leq \Pr[T'] \cdot \frac{1}{1 - 2\varepsilon + c\varepsilon^2}.$$

It follows that

$$\Pr_{(u,v)\in E_{S\times T}}[(u,v) \in S' \times T'] \leq \Pr[S']\Pr[T'] \cdot \max\left\{\frac{c}{1 - c\varepsilon}, \frac{1}{1 - 2\varepsilon + c\varepsilon^2}\right\}.$$

Since $c\varepsilon \leq \frac{1}{3}$, $\frac{c}{1-c\varepsilon} \leq \frac{3}{2} \cdot c$. Further, $\varepsilon \leq c\varepsilon \leq 1/3$ and $\frac{1}{1-2\varepsilon+c\varepsilon^2} \leq \frac{1}{1-2\varepsilon} \leq 1 + 6\varepsilon$. Thus, the max expression is at most $\max\{\frac{3}{2} \cdot c, 1 + 6\varepsilon\} \leq \max\{\frac{3}{2} \cdot c, 3\} \leq 3c$.

Finally, suppose that T' is disjoint from T. Transitions from A go into $B \subseteq T$, transitions from C stay in $C \subseteq T$, and $B \cap S = \varnothing$. Hence only starting vertices in $S \cap D$ can contribute. Therefore,

$$\begin{aligned}\Pr_{(u,v)\in E_{S\times T}}[(u,v) \in S \times T'] &= \Pr[S \cap D] \cdot \frac{|D \cap T'|}{|D|} \\ &\leq \Pr[S]\Pr[T'] \cdot \frac{1}{1 - 2\varepsilon + c\varepsilon^2},\end{aligned}$$

proving the third bullet. □

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