Electronic Colloquium on Computational Complexity, Revision 1 of Report No. 143 (2021)

Hitting Sets for Regular Branching Programs

Andrej Bogdanov[*]
The Chinese University of Hong Kong
andrejb@cse.cuhk.edu.hk

William M. Hoza
Simons Institute
williamhoza@berkeley.edu

Gautam Prakriya
The Chinese University of Hong Kong
gautamprakriya@gmail.com

Edward Pyne[†]
Harvard University
epyne@college.harvard.edu

November 3, 2021

Abstract

We construct improved hitting set generators (HSGs) for ordered (read-once) regular branching programs in two parameter regimes. First, we construct an explicit ε-HSG for *unbounded-width* regular branching programs with a single accept state with seed length

$$\widetilde{O}(\log n \cdot \log(1/\varepsilon)),$$

where n is the length of the program. Second, we construct an explicit ε-HSG for width-w length-n regular branching programs with seed length

$$\widetilde{O}\left(\log n \cdot \left(\sqrt{\log(1/\varepsilon)} + \log w\right) + \log(1/\varepsilon)\right).$$

For context, the "baseline" in this area is the pseudorandom generator (PRG) by Nisan (Combinatorica 1992), which fools ordered (possibly non-regular) branching programs with seed length $O(\log(wn/\varepsilon) \cdot \log n)$. For regular programs, the state-of-the-art PRG, by Braverman, Rao, Raz, and Yehudayoff (FOCS 2010, SICOMP 2014), has seed length $\widetilde{O}(\log(w/\varepsilon) \cdot \log n)$, which beats Nisan's seed length when $\log(w/\varepsilon) = o(\log n)$. Taken together, our two new constructions beat Nisan's seed length in all parameter regimes except when $\log w$ and $\log(1/\varepsilon)$ are *both* $\Omega(\log n)$ (for the construction of HSGs for regular branching programs with a single accept vertex).

Extending work by Reingold, Trevisan, and Vadhan (STOC 2006), we furthermore show that an explicit HSG for regular branching programs with a single accept vertex with seed length $o(\log^2 n)$ in the regime $\log w = \Theta(\log(1/\varepsilon)) = \Theta(\log n)$ would imply improved HSGs for *general* ordered branching programs, which would be a major breakthrough in derandomization. Pyne and Vadhan (CCC 2021) recently obtained such parameters for the special case of permutation branching programs.

Keywords: pseudorandomness, space-bounded computation

[*] Supported by RGC GRF CUHK 14209419 and CUHK 14209920. Part of the work was completed at the Simons Institute for the Theory of Computing, UC Berkeley.

[†] Supported by NSF grant CCF-1763299.

ISSN 1433-8092

1 Introduction

Random choices can make computing easier sometimes, but random bits are not always available. We therefore want to understand when randomized algorithms have an inherent advantage and when randomness is unnecessary. In this work, we focus on the interplay between randomness and *space complexity*. Starting with the work of Ajtai, Komlós, and Szemerédi [AKS87], there have been three decades of work on the derandomization of space-bounded computation, with the goal of eventually proving that every halting decision algorithm can be derandomized with only a constant factor space blowup ($\mathbf{L} = \mathbf{BPL}$). As in previous work, we will use the following nonuniform model of space-bounded computation, which captures how a randomized small-space algorithm uses its random bits.

Definition 1.1. An **(ordered) branching program** B of **length** n and **width** w computes a function $B : \{0,1\}^n \to \{0,1\}$. On an input $x \in \{0,1\}^n$, 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 input bit x_t and updates its state according to a transition function $B_t : [w] \times \{0,1\} \to [w]$ by taking $v_t = B_t(v_{t-1}, x_t)$. Note that the transition function B_t can differ at each time step.

Moreover, there is a set $V_{\text{acc}} \subseteq [w]$ of accept states. Let v_n be the final state reached by the branching program on input x. If $v_n \in V_{\text{acc}}$ the branching program accepts, denoted $B(x) = 1$, and otherwise the program rejects, denoted $B(x) = 0$. We also consider branching programs restricted to having a single accept state, which is always denoted v_{acc}.

Arguably the most natural approach to derandomizing space-bounded computation is to design a *pseudorandom generator*, defined next. We let U_i denote the uniform distribution over $\{0,1\}^i$.

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

$$\left| \Pr_{x \leftarrow U_n}[f(x) = 1] - \Pr_{x \leftarrow U_s}[f(G(x)) = 1] \right| \leq \varepsilon.$$

We say that G **ε-fools** $\mathcal{F}$ if it is an ε-PRG for $\mathcal{F}$. The input length s is the **seed length** of the generator.

It can be shown via the probabilistic method that there is a (non-explicit) ε-PRG for ordered branching programs of length n and width w that has seed length $O(\log(nw/\varepsilon))$, and moreover this seed length is optimal. We say a generator G is **explicit** if the output is computable in space $O(s)$. Decades of work has focused on constructing explicit pseudorandom generators with parameters matching the probabilistic method. All results we subsequently discuss are explicit constructions. In 1990, Nisan [Nis92] constructed an ε-PRG for general ordered branching programs of length n and width w with seed length

$$O(\log n \cdot (\log n + \log w + \log(1/\varepsilon))).$$

Nisan's PRG is a factor of $O(\log n)$ from optimal, and achieves seed length $O(\log^2 n)$ when $w \leq \operatorname{poly}(n)$ and $\varepsilon \geq 1/\operatorname{poly}(n)$, rather than the optimal $O(\log n)$.

There has been extensive work analyzing restricted classes of branching programs with additional structure. We focus on the well-studied class of *regular* programs:

Definition 1.3. An **(ordered) regular branching program** of length n and width w is an ordered branching program where for every $t = 1, \ldots, n$ and every $v \in [w]$, there are exactly 2 pairs $(u, b) \in [w] \times \{0,1\}$ such that $B_t(u, b) = v$.

In 2010, Braverman, Rao, Raz and Yehudayoff [BRRY14] constructed a PRG for regular branching programs with near-optimal dependence on n, achieving seed length:[1]

$$O(\log n \cdot (\log\log n + \log w + \log(1/\varepsilon))).$$

(Subsequently, De also presented a PRG for regular programs, albeit with a somewhat inferior seed length [De11].) Braverman et al.'s result suggests two natural challenges regarding regular programs. The first challenge is to improve the $\log n \cdot \log w$ term in Nisan's seed length; this is necessary to beat Nisan's generator in the polynomial-width regime (e.g., $w = n$). The second challenge is to improve the $\log n \cdot \log(1/\varepsilon)$ term; this is necessary to beat Nisan's generator in the small-error regime (e.g., $\varepsilon = 1/n$).

Designing PRGs that meet these challenges seems to be quite difficult, but fortunately PRGs are not the only approach to derandomization. To address the challenges we will instead aim to construct *hitting set generators* (HSGs). An HSG (defined next) is a "one-sided" generalization of a PRG that is still valuable for derandomization.

Definition 1.4. Let $\mathcal{F}$ be a class of functions $f : \{0,1\}^n \to \{0,1\}$. An **$\varepsilon$-hitting set generator** (**ε-HSG**) for $\mathcal{F}$ is a function $H : \{0,1\}^s \to \{0,1\}^n$ such that for every $f \in \mathcal{F}$ where $\Pr_{x \leftarrow U_n}[f(x) = 1] > \varepsilon$, there exists $x \in \{0,1\}^s$ such that $f(H(x)) = 1$.

Note that with our definitions, an ε-PRG for a class $\mathcal{F}$ is an ε-HSG for $\mathcal{F}$. In many cases, there has been more success at developing HSGs than at developing PRGs. Indeed, in the context of regular branching programs, in addition to their PRG construction, Braverman, Rao, Raz, and Yehudayoff constructed an HSG with seed length $O(w \log n)$, achieving optimal seed length for constant width [BRRY14].[2]

1.1 Our Contributions

In this work, we present improved HSGs for regular branching programs. For our first result, we focus on improving the dependence on w, the width of the program. In fact, we study the intriguing setting of *unbounded-width* programs [MZ13, HPV21, PV21a, PV22]. We design an HSG for unbounded-width regular branching programs with a single accept vertex with a near-optimal dependence on the length of the program n.

Theorem 1.5. *Given $n \in \mathbb{N}$ and $\varepsilon \in (0, 1/2)$, there is an explicit ε-HSG for regular branching programs of length n and unbounded width with a single accept state that has seed length*

$$O(\log n \cdot (\log\log n + \log(1/\varepsilon))).$$

This result eliminates all dependence on w from the seed length of Braverman, Rao, Raz, and Yehudayoff's PRG [BRRY14], with the caveats that we only obtain an HSG and we assume that there is only one accept state. For regular branching programs of width $w = \mathrm{poly}(n)$ (the regime most relevant for the derandomization of space-bounded computation), Theorem 1.5 is the first explicit construction with seed length $o(\log^2 n)$. In the superpolynomial-width regime, the state of the art prior to our work was the analysis of Hoza, Pyne, and Vadhan [HPV21], which implies that

[1] They consider regular branching programs with a single accept state, but dividing ε by w to allow an arbitrary set of accept states does not change the seed length.

[2] The lack of dependence on ε can be explained by the observation of BRRY that every regular branching program that has nonzero acceptance probability has acceptance probability at least $1/2^{w-1}$, so WLOG $\varepsilon > 1/2^w$, i.e. $w > \log(1/\varepsilon)$.

Nisan's generator is an HSG for unbounded-width regular branching programs with a single accept vertex with seed length $O(\log n \cdot \log(n/\varepsilon))$.

More generally, if a program has a accept states and acceptance probability at least ε, then there must be an accept state that is reached with probability at least ε/a, so we obtain the following corollary:[3]

Corollary 1.6. *Given $n, a \in \mathbb{N}$ and $\varepsilon \in (0, 1/2)$, there is an explicit ε-HSG for regular branching programs of length n and unbounded width with a accept states that has seed length*

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

For our second result, we focus on improving the dependence on ε, the threshold of the HSG.

Theorem 1.7. *For every $w, n \in \mathbb{N}$ and $\varepsilon > 0$, there exists an explicit ε-HSG for width-w length-n regular branching programs with seed length*

$$O\left(\log n \cdot \left(\sqrt{\log(1/\varepsilon)} + \log w + \log\log n\right) + \log(1/\varepsilon)\right).$$

Comparing to the seed length of Braverman, Rao, Raz, and Yehudayoff's PRG [BRRY14], we improve the $\log n \cdot \log(1/\varepsilon)$ term to $\log n \cdot \sqrt{\log(1/\varepsilon)}$. Recall that Braverman et al. also constructed an HSG with seed length $O(w \log n)$, independent of ε. Our seed length has a much better dependence on w, so for example, when $w = 2^{O(\sqrt{\log n})}$ and $\varepsilon = 1/n$, our seed length is $\widetilde{O}(\log^{3/2} n)$, whereas prior work could not beat Nisan's $O(\log^2 n)$ seed length for that regime. Furthermore, since every nonzero width-w regular program has acceptance probability at least $2^{-(w-1)}$ [BRRY14], Theorem 1.7 implies that we can achieve seed length $\widetilde{O}\left(\sqrt{w}\log n + w\right)$, independent of ε. Theorem 1.7 makes progress on a problem posed by Hoza and Zuckerman [HZ20]: they asked for an HSG with seed length $\widetilde{O}(\log(n/\varepsilon))$ for regular branching programs of width $\operatorname{polylog} n$.

Taken together, Theorems 1.5 and 1.7 improve the seed length of Nisan's construction for hitting regular branching programs with a single accept state when $\log(1/\varepsilon) = o(\log n)$ *or* $\log w = o(\log n)$, thereby identifying the regime $\log(1/\varepsilon) \sim \log w \sim \log n$ as the remaining target. An explicit HSG of seed length $o(\log^2 n)$ in that regime would also be an explicit HSG of seed length $o(\log^2 n)$ for polynomial-width regular branching programs with an *arbitrary* set of accept states. In turn, we show that such an HSG would imply a major advance in derandomizing space-bounded computation:

Theorem 1.8. *For every $n, w \in \mathbb{N}$ and $\varepsilon > 0$, there are values $w' = \operatorname{poly}(nw/\varepsilon)$ and $n' = O(n \log(nw/\varepsilon))$ such that if there is an explicit ε-HSG (resp. PRG) $G : \{0,1\}^s \to \{0,1\}^{n'}$ for width-w' length-n' regular branching programs, then there is an explicit $O(\varepsilon)$-HSG (resp. PRG) $G' : \{0,1\}^s \to \{0,1\}^n$ for width-w length-n ordered branching programs with the same seed length.*

The conclusion of Theorem 1.5 yields a derandomization of decision problems solvable in randomized logspace with one-sided error (the class **RL**). Cheng and Hoza [CH20] show more generally that a two-sided error derandomization (the class **BPL**) follows. Thus we obtain the following corollary:

Corollary 1.9. *Suppose there is an explicit ε-HSG of seed length $O(\log(nw/\varepsilon))$ for regular branching programs of length n and width w. Then* $\mathbf{BPL} = \mathbf{L}$.

[3] We remark that it is not possible to improve this dependence on a by any analysis of the INW generator that only uses the expansion properties of the auxiliary expanders [PV21b], even for HSGs.

1.2 Related work

Theorem 1.8 extends a result of Reingold, Trevisan, and Vadhan [RTV06]. They show an analogous transformation for PRGs over alphabet size $\mathrm{poly}(nw/\varepsilon)$. In contrast, Theorem 1.8 also applies to HSGs and works over the binary alphabet.

A number of results improve the seed length of Braverman et al. for the restricted class of "permutation" branching programs. A *permutation branching program* is a regular branching program with the further restriction that the transitions $B_t(\cdot, 1)$ and $B_t(\cdot, 0)$ are permutations of $[w]$ for every t. While most of these results are tailored to the constant-width regime [BV10, KNP11, De11, Ste12, RSV13], two exceptions construct generators for unbounded-width permutation branching programs with a single accept vertex:

- Hoza, Pyne, and Vadhan [HPV21] (building on work by Ahmadinejad, Kelner, Murtagh, Peebles, Sidford, and Vadhan [AKM$^+$20]) construct a PRG with seed length $\widetilde{O}(\log n \cdot \log(1/\varepsilon))$, and
- Pyne and Vadhan [PV21a] construct an HSG[4] with seed length $\widetilde{O}\left(\log n \cdot \sqrt{\log(n/\varepsilon)} + \log(1/\varepsilon)\right)$.

Our Theorem 1.5 matches the seed length of Hoza, Pyne, and Vadhan [HPV21] for the more general setting of regular programs. Our Theorem 1.7 can be viewed as an analogue of the result of Pyne and Vadhan [PV21a]. Unfortunately, our seed length includes an additional $O(\log n \cdot \log w)$ term. If this term were at all improved, we would obtain $o(\log^2 n)$ seed length HSGs for general branching programs via Theorem 1.8.

The arguments used for unbounded-width permutation branching programs [HPV21, PV21a] rely heavily on the permutation condition to leverage powerful results in spectral graph theory [RV05, CKK$^+$18, AKM$^+$20] that are not applicable in the regular setting.[5] In contrast, our proofs are combinatorial. In particular, our Theorem 1.7 is proved via combinatorial rather than spectral error-reduction methods, providing some hope that the aforementioned improvement in the $O(\log n \cdot \log w)$ term might not be out of reach.

1.3 Overview of Proofs

1.3.1 The Unanimity Program Model

The proofs of Theorem 1.5 and Theorem 1.7 both rely on a new generalization of ordered branching programs that we call *unanimity programs.* A unanimity program is defined like an ordered branching program, except that every vertex (not just those in the last layer) is labeled either "accept" or "reject." The program accepts if *every* vertex it visits is an accepting vertex; otherwise it rejects. More precisely:

Definition 1.10. An **(ordered) unanimity program** B of length n and width w starts at a fixed start state $v_0 \in [w]$. In each step $t \in [n]$, the program reads the next input symbol x_t and updates its state according to a transition function $B_t \colon [w] \times \{0, 1\} \to [w]$ by taking $v_t = B_t(v_{t-1}, x_t)$. For every $t \in \{0, 1, \ldots, n\}$, there is a set of accept states $V_{\mathrm{acc}}^{(t)} \subseteq [w]$. The program accepts, denoted $B(x) = 1$, if for every t, we have $v_t \in V_{\mathrm{acc}}^{(t)}$. Otherwise the program rejects, denoted $B(x) = 0$.

A unanimity program is **regular** if for every $t = 1, \ldots, n$ and every $v \in [w]$, there are exactly two pairs $(u, b) \in [w] \times \{0, 1\}$ such that $B_t(u, b) = v$. The program is a **permutation unanimity**

[4] Their result gives a more general object called "weighted PRG" or "pseudorandom pseudodistribution" [BCG20].

[5] Specifically, permutation branching programs have the property that the underlying graph remains regular – and hence the uniform distribution remains stationary – even when we restrict to a pseudorandom sequence of paths.

program if for every $t = 1, \dots, n$ and every $b \in \{0, 1\}$, the function $B_t(\cdot, b)$ is a permutation on $[w]$.

The standard definition of an ordered branching program is the special case that for $t < n$, we have $V^{(t)}_{\text{acc}} = [w]$. Throughout this paper, the phrase "branching program" will always refer to the standard model, whereas "unanimity program" will refer to the more general model.

A width-w unanimity program can trivially be simulated by a width-$(w+1)$ branching program. However, this simulation does not preserve regularity, and in fact it is not possible in general to simulate a regular unanimity program by a regular branching program of a similar width.[6]

Despite the fact that the unanimity model is strictly more powerful, we show (Lemma 2.1) that designing PRGs for regular unanimity programs is essentially equivalent to designing PRGs for regular branching programs (with an arbitrary set of accept states in the final layer). Therefore, known PRGs for regular branching programs [BRRY14, De11] automatically also fool this broader class of statistical tests. We use this lemma in two different ways to prove our two main results (Theorems 1.5 and 1.7).

1.3.2 The Large-Width Case

To prove Theorem 1.5, for every unbounded-width regular branching program B with a single accept state and every $\varepsilon > 0$, we construct an ε-"lower-approximator" for B. The lower approximator is a regular unanimity program B_L of width $O(1/\varepsilon)$ that accepts on a subset of the strings accepted by B, and has acceptance probability (under the uniform distribution) within ε of that of B.

Lemma 1.11. *Let $\varepsilon > 0$. Every regular (respectively permutation) branching program of length n and unbounded width, with a single accept state, is ε-lower approximated by a regular (respectively permutation) unanimity program of length n and width $2 \cdot \lfloor 1/\varepsilon \rfloor$.*

A standard argument shows that an HSG for a lower approximator of B is also an HSG for B, so given Lemma 1.11, it follows that the BRRY PRG for bounded-width regular branching programs [BRRY14] is our desired HSG for unbounded-width regular branching programs.

We prove Lemma 1.11 by a more careful analysis of a result of Hoza, Pyne, and Vadhan [HPV21], who prove that B has an ε-lower approximator of width $O(n/\varepsilon)$. The key observation behind our improvement is that regular branching programs cannot concentrate low probability events. More precisely, say that a vertex v of B is "negligible" if the probability of visiting v is at most ε when B reads a uniform random input. We show that the probability of visiting some negligible vertex and then accepting is at most ε, with no dependence on n. Thus, if we reject all inputs that visit negligible vertices (i.e., we impose a unanimity condition), then we get an ε-lower approximator. Furthermore, the "effective width" of the approximator (namely, the maximum number of *accepting* vertices in any individual layer) is at most $1/\varepsilon$. A regular unanimity program of effective width w_{eff} can be simulated by one of actual width $2w_{\text{eff}}$ (see Lemma 3.3), completing the proof.

1.3.3 The Low-Threshold Case

To prove Theorem 1.7, we follow the approach of Hoza and Zuckerman [HZ20]. They designed a method to convert any "moderate-error" PRG for ordered branching programs into an ε-HSG, where ε is potentially very small [HZ20]. (See also related work on error reduction for "weighted PRGs" [BCG20, CL20, CDR$^+$21, PV21a, Hoz21].) We develop a modified version of their framework that is suitable for the setting of regular programs.

[6] For example, the AND function on n bits can be computed by a width-2 permutation unanimity program, but it cannot be computed by a width-w regular branching program unless $w \geq n + 1$.

Let B be an ordered branching program that accepts with probability p. Let $K > 1$, and let S be the set of vertices from which the acceptance probability is at least Kp. The starting point of the construction is our Lemma 5.1, which states that on a uniformly random input, B has a moderately large $\Omega(1/K)$ chance of visiting some vertex in S. If B is regular, then the predicate of failing to visit S can be computed by a regular unanimity program of the same width. Therefore, when B reads a pseudorandom string produced by a moderate-error PRG for regular branching programs, there is still an $\Omega(1/K)$ chance of visiting S. The HSG guesses where to truncate the pseudorandom string to land in S and then repeats the process, increasing the acceptance probability to K^2p, then K^3p, etc., until eventually an accepting vertex is reached. To keep the overall seed length low, we recycle the PRG's seed from one iteration to the next using a hitter (also known as a disperser).

The upshot is that for any $\varepsilon < \varepsilon_0 < 0.1$, we can convert an ε_0-PRG for width-$(2w)$ regular branching programs into an ε-HSG for width-w regular branching programs. If the ε_0-PRG has seed length s, then our ε-HSG has seed length

$$O\left(s + \frac{\log(1/\varepsilon) \cdot \log n}{\log(1/\varepsilon_0)} + \log(wn/\varepsilon)\right).$$

We plug in the PRG construction by Braverman, Rao, Raz, and Yehudayoff [BRRY14] with error $\varepsilon_0 = 2^{-\sqrt{\log(1/\varepsilon)}}$ to complete the proof of Theorem 1.7.

Hoza and Zuckerman's original version of the reduction [HZ20] is similar. The key difference is that in each iteration of their setup, the probability of visiting their "target set" S of vertices is only $1/\operatorname{poly}(n)$. As a result, their version of the reduction requires the "moderate-error" PRG to have error $\varepsilon_0 < 1/\operatorname{poly}(n)$, which would be too small for our purposes. Our refined lemma allows us to greatly increase the probability of visiting S. Unlike in Hoza and Zuckerman's setting, however, the set S may now be spread across multiple layers of the branching program and thus has to be analyzed in the unanimity program model.

1.4 Other Results

Our approach for constructing HSGs for unbounded-width regular branching programs also works, mutatis mutandis, for some other unbounded-width models. For unbounded-width *permutation* branching programs with a single accept state, by plugging in the best PRGs for constant-width permutation branching programs [De11, Ste12], we achieve seed length $\log n \cdot \operatorname{poly}(1/\varepsilon)$, which is optimal when the threshold ε is constant.

Proposition 1.12. *Given $n \in \mathbb{N}$ and $\varepsilon > 0$, there is an explicit ε-HSG for permutation branching programs of length n and unbounded width with a single accept state that has seed length*

$$O(\log n \cdot \log(1/\varepsilon) \cdot (1/\varepsilon^4)).$$

The approach also works in the more challenging "unordered" model. For unordered permutation branching programs, we get near-optimal seed length for constant threshold ε.

Proposition 1.13. *Given $n \in \mathbb{N}$ and $\varepsilon > 0$, there exists an explicit ε-HSG for unbounded-width unordered permutation branching programs of length n with a single accept state that has seed length*

$$O(\log(n/\varepsilon) \cdot \log\log n \cdot (1/\varepsilon^4)).$$

We also get an improvement for unordered *regular* branching programs. In the unordered setting, it tends to be difficult to take advantage of regularity, because the regularity condition is not preserved under restrictions. This issue has forced some prior works to settle for fooling permutation programs [RSV13,CHHL19]. However, our reduction does not involve any restrictions, so we are not affected by the issue. For unbounded-width unordered regular branching programs, we get seed length $\widetilde{O}((\log^2 n)/\varepsilon)$, which admittedly is still far from optimal, but keep in mind that the state-of-the-art PRG for general polynomial-width unordered branching programs has seed length $O(\log^3 n)$ [FK18].

Proposition 1.14. *Given $n \in \mathbb{N}$ and $\varepsilon > 0$, there exists an explicit ε-HSG for unbounded-width unordered regular branching programs of length n with a single accept state that has seed length*

$$O(\log(n/\varepsilon) \cdot \log n \cdot \log\log n \cdot (1/\varepsilon)).$$

Our results for unordered branching programs (Propositions 1.13 and 1.14) rely on PRGs developed by prior work of Forbes and Kelley [FK18] and Chattopadhyay, Hatami, Hosseini and Lovett [CHHL19].

We observe that the BRRY [BRRY14] HSG for constant-width regular branching programs can be viewed more generally as a *co-HSG* for unbounded-width regular branching programs with a constant number of *accept* states.

Proposition 1.15. *Given $n, a \in \mathbb{N}$, the set $H = \{x \in \{0,1\}^n : \mathrm{wt}(x) \le a\}$ where $\mathrm{wt}(x)$ denotes the Hamming weight of x is a co-hitting set for regular branching programs of length n and unbounded width with a accept states. That is, for every regular branching program B with at most a accept states that is not the constant function $B(x) = 1$, there is $x \in H$ such that $B(x) = 0$.*

In contrast, [HPV21] show that a random function is not a co-HSG for regular branching programs of unbounded width and a single accept state unless the seed length is $\Omega(n)$.[7] Thus, we obtain a very simple explicit construction with exponentially shorter seed length than that obtained via the probabilistic method.

1.5 Preliminaries

First, we define notation relating to states and transitions in a branching program. Here, we adopt the perspective of a branching program as a directed graph, with edges from layer i to layer $i+1$ corresponding to the ith transition function.

Definition 1.16. For a branching program B of length n, let $V = V_0 \cup V_1 \cup \ldots \cup V_n$ be the vertex set of the branching program, where V_i holds the vertices corresponding to states in layer i. We will overload notation and consider the transition function as a map $B_t \colon V_{t-1} \times \{0,1\} \to V_t$ in addition to thinking of it as a map $B_t \colon [w] \times \{0,1\} \to [w]$. Similarly, we will often think of the start state v_0 as being an element of V_0 instead of an element of $[w]$, and similarly $V_{\mathrm{acc}} \subseteq V_n$ instead of $V_{\mathrm{acc}} \subseteq [w]$, etc. For $v \in V_i$ and $u \in V_j$ for $j > i$, we write $B[v, x] = u$ if the program transitions to state u starting from state v on input $x \in \{0,1\}^{j-i}$.

Next, we define notation for the probability of reaching a state from the start state, and notation for the probability of accepting from that state.

Definition 1.17. Let B be a branching program, let $v_0 \in V_0$ be the start state, and let $V_{\mathrm{acc}} \subseteq V_n$ be the set of accept states. For every state $v \in V_i$, let $p_{\to v} = \Pr[B[v_0, U_i] = v]$ be the probability v is reached from the start state over U_i, and let $p_{v\to} = \Pr[B[v, U_{n-i}] \in V_{\mathrm{acc}}]$ be the probability the program accepts over U_{n-i} starting from v, where we define $p_{\to v_0} = 1$ and $p_{\to v} = 0$ for all $v \in V_0 \setminus \{v_0\}$ and likewise $p_{v\to} = 1$ for $v \in V_{\mathrm{acc}}$ and $p_{v\to} = 0$ for $v \in V_n \setminus V_{\mathrm{acc}}$.

[7] Their result is stated as showing a random function is not a PRG, but the argument also rules out a co-HSG.

For a state v with transitions to u_1, u_2 (which are not necessarily distinct), the accept probability $p_{v\to}$ is exactly equal to $(p_{u_1\to} + p_{u_2\to})/2$. Next, we formally define the concept of a lower approximator.

Definition 1.18. Given a branching program B of length n and $\varepsilon > 0$, a length-n branching program B_L is an **ε-lower approximator of** B if $B_L^{-1}(1) \subseteq B^{-1}(1)$ and $|\Pr[B(U_n) = 1] - \Pr[B_L(U_n) = 1]| \leq \varepsilon$.

Finally, we define the unordered branching program model (although it is not the focus of this paper).

Definition 1.19. An **unordered (oblivious read-once) branching program** B consists of an ordered branching program B' and a permutation $\pi\colon [n] \to [n]$. It computes the function

$$B(x) = B'(x_{\pi(1)}, \ldots, x_{\pi(n)}).$$

We similarly define **unordered regular branching programs**, etc.

1.6 Organization

In Section 2 we prove that fooling regular unanimity programs is essentially equivalent to fooling regular branching programs. In Section 3, we show that regular branching programs with a single accept state can be ε-lower-approximated by regular unanimity programs of width $O(1/\varepsilon)$. In Section 4 we combine these two results and conclude Theorem 1.5 (our HSG for unbounded-width regular branching programs); we also prove our other results for unbounded-width programs in this section. In Section 5 we construct our low-threshold HSG. In Appendix A we prove Theorem A.4, establishing that pseudorandom objects for regular programs over a binary alphabet imply pseudorandom objects for regular programs over a D-ary alphabet. In Appendix B we re-prove the result of Reingold, Trevisan, and Vadhan [RTV06] in a formulation convenient for our use, and show their argument works for HSGs.

2 PRGs for Unanimity Programs

In this section, as outlined in Section 1.3.1, we prove that PRGs for regular branching programs also fool the more general model of regular *unanimity* programs.

Lemma 2.1. *Let $w, n \in \mathbb{N}$ and let $G\colon \{0,1\}^s \to \{0,1\}^n$. If G is an ε-PRG for width-$(2w)$ regular (respectively permutation) branching programs, then G is a (2ε)-PRG for width-w regular (respectively permutation) unanimity programs.*

Proof. Let B be a width-w length-n unanimity program. Let the layers of the program be $V_0, \ldots, V_n$ and let $v_0 \in V_0$ be the start state. For $t \in \{0, \ldots, n\}$, let $V_{\text{acc}}^{(t)} \subseteq V_t$ be the set of accepting vertices in layer t, and define a function $R_t\colon \{0,1\}^n \to \{0,1\}$ by

$$R_t(x) = 1 \iff B[v_0, x_{1..t}] \notin V_{\text{acc}}^{(t)}.$$

That is, $R_t(x)$ indicates whether $B(x)$ visits a reject state in layer t. Then

$$1 - B(x) = \bigvee_{t=0}^{n} R_t(x) = 2^{-n} \cdot \sum_{T \subseteq \{0,\ldots,n\}} \bigoplus_{t \in T} R_t(x), \tag{1}$$

because by the Fourier expansion of the OR function we have we have $\mathsf{OR}(y_0, \dots, y_n) = 2 \cdot \mathbb{E}_{T \subseteq \{0,\dots,n\}}\left[\bigoplus_{t \in T} y_t\right]$. For each $T \subseteq \{0, \dots, n\}$, define $B^{(T)}(x) = \bigoplus_{t \in T} R_t(x)$. Let us design a width-$(2w)$ branching program to compute $B^{(T)}$. The vertex set in layer $t \in \{0, \dots, n\}$ is given by $V_t^{(T)} = V_t \times \{0, 1\}$. The start state is (v_0, a) where

$$a = (0 \in T \wedge v_0 \notin V_{\text{acc}}^{(0)}).$$

For $t > 0$, the transition function $B_t^{(T)} \colon V_{t-1}^{(T)} \times \{0,1\} \to V_t^{(T)}$ is given by $B_t^{(T)}((v_{t-1}, a_{t-1}), b) = (v_t, a_t)$, where

$$\begin{aligned} v_t &= B_t(v_{t-1}, b) \\ a_t &= a_{t-1} \oplus (t \in T \wedge v_t \notin V_{\text{acc}}^{(t)}). \end{aligned}$$

In the final layer, the set of accept states is $V_{\text{acc}} = V_n \times \{1\}$. This program indeed computes $B^{(T)}(x)$, because in layer t, the program reaches the state (v_t, a_t), where $v_t = B[v_0, x_{1..t}]$ and $a_t = \bigoplus_{t \in T \cap \{0, \dots, t\}} R_t(x)$.

We claim that if B is a regular program, then so is $B^{(T)}$, and furthermore if B is a permutation program, then so is $B^{(T)}$. To prove it, fix some $t > 0$ and some $(v, a) \in V_t^{(T)}$. If B is regular, then $|B_t^{-1}(v)| = 2$, say $B_t^{-1}(v) = \{(u_0, b_0), (u_1, b_1)\}$. Define

$$a' = a \oplus (t \in T \wedge v \notin V_{\text{acc}}^{(t)}).$$

From the definition of $B^{(T)}$, we have

$$(B_t^{(T)})^{-1}((v, a)) = \{((u_0, a'), b_0), ((u_1, a'), b_1)\}.$$

In particular, $|(B_t^{(T)})^{-1}((v, a))| = 2$, showing that $B^{(T)}$ is regular. If furthermore B is a permutation program, then $b_0 \neq b_1$, which immediately implies that $B^{(T)}$ is a permutation program.

Consequently, G fools each function $B^{(T)}$ with error ε. By Equation 1, it follows that G fools $1 - B$ (and therefore B) with error $2^{-n} \cdot \sum_{T \subseteq \{0, \dots, n\}} \varepsilon = 2\varepsilon$. □

The idea of reducing disjunctions to parity functions (like what we do in Equation 1) is not new. Prior works have used a similar technique in other settings (e.g. [Wil18, Lee19, DHH20]).

3 Lower Approximators for Regular Branching Programs

In this section, we show that unbounded-width regular branching programs can be lower approximated by bounded-width regular unanimity programs. Hoza, Pyne, and Vadhan showed [HPV21, Theorem 4.1] that unbounded-width regular branching programs are ε-lower approximated by regular *branching* programs of width $O(n^2/\varepsilon)$, which is too large for our application.[8] We obtain a lower approximator of width $O(1/\varepsilon)$:

Lemma 1.11. *Let $\varepsilon > 0$. Every regular (respectively permutation) branching program of length n and unbounded width, with a single accept state, is ε-lower approximated by a regular (respectively permutation) unanimity program of length n and width $2 \cdot \lfloor 1/\varepsilon \rfloor$.*

[8] They also constructed lower approximators of width $O(n/\varepsilon)$. Those programs are not quite regular, but even setting aside issues of regularity, they are still too wide for our application.

To get the width down to $O(1/\varepsilon)$, we make two changes to the proof by Hoza, Pyne, and Vadhan [HPV21]. First, rather than retaining the n/ε most important states at each layer, we prove that retaining only the $1/\varepsilon$ most important states suffices. Second, we show that the unanimity program model allows us to rewire edges that previously pointed to deleted vertices (in such a way that the approximator rejects whenever such edges are crossed) while only increasing the width by a constant factor, whereas Hoza, Pyne, and Vadhan paid another factor of n at this stage to obtain a regular branching program [HPV21].

We begin with the following claim, which shows that a regular branching program cannot concentrate low-probability events.

Claim 3.1. *Let $\varepsilon > 0$. Let B be a regular branching program, and let $V^\varepsilon = \{v : p_{\to v} \leq \varepsilon\}$ be the vertices of B that have at most ε probability of being reached over a uniformly random string. Then, for every $i \in \{0, 1, \ldots, n\}$ and $v \in V_i$, the probability of reaching v and visiting at least one vertex from V^ε along the way is at most ε. That is,*

$$\Pr_{x \leftarrow U_i}\left[(B[v_0, x] = v) \bigwedge \bigvee_{j=1}^{i} (B[v_0, x_{1..j}] \in V^\varepsilon)\right] \leq \varepsilon.$$

Proof. The proof is by induction on i. The property is trivially true for states in the 0th layer. Assuming it holds for layer i, consider $v \in V_{i+1}$. If $v \in V^\varepsilon$ the property holds since

$$\Pr_{x \leftarrow U_{i+1}}\left[(B[v_0, x] = v) \bigwedge \bigvee_{j=1}^{i+1} (B[v_0, x_{1..j}] \in V^\varepsilon)\right] = \Pr_{x \leftarrow U_{i+1}}[B[v_0, x] = v] = p_{\to v} \leq \varepsilon.$$

Otherwise,

$$\begin{aligned}
&\Pr_{x \leftarrow U_{i+1}}\left[(B[v_0, x] = v) \bigwedge \bigvee_{j=1}^{i+1} (B[v_0, x_{1..j}] \in V^\varepsilon)\right] \\
&= \Pr_{x \leftarrow U_{i+1}}\left[\left(\bigvee_{(u,b) \in B_{i+1}^{-1}(v)} (B[v_0, x_{1..i}] = u \wedge x_{i+1} = b)\right) \bigwedge \bigvee_{j=1}^{i} (B[v_0, x_{1..j}] \in V^\varepsilon)\right] \\
&= \sum_{(u,b) \in B_{i+1}^{-1}(v)} \frac{1}{2} \cdot \Pr_{x \leftarrow U_i}\left[(B[v_0, x] = u) \bigwedge \bigvee_{j=1}^{i} (B[v_0, x_{1..j}] \in V^\varepsilon)\right] \\
&\leq \sum_{(u,b) \in B_{i+1}^{-1}(v)} \frac{\varepsilon}{2} \\
&= \varepsilon,
\end{aligned}$$

where the last step uses regularity. □

Using Claim 3.1, we now show that a regular branching program with a single accept state is ε-lower approximated by a regular unanimity program with at most $O(1/\varepsilon)$ accept states in each layer. The maximum number of accept states in an individual layer of the program can be considered a measure of the "effective width" of the program.

Lemma 3.2. *Let $\varepsilon > 0$. Every regular (respectively permutation) branching program of length n and width w, with a single accept state, is ε-lower approximated by a regular (respectively permutation) unanimity program of length n and width w, with at most $\lfloor 1/\varepsilon \rfloor$ accept states in each layer.*

Proof. Let B be a regular (permutation) branching program of length n and width w, with a single accept state v_{acc}, and let $V^\varepsilon = \{v : p_{\to v} \leq \varepsilon\}$. We construct a unanimity program B_L that ε-lower approximates B. We take B_L to have the same set of states, transitions, and start state as B. For $i \in \{0, \ldots, n-1\}$, the set of accept states in layer i of B_L is $V_i \setminus V^\varepsilon$, i.e., the set of states reached with probability greater than ε. In layer n, the set of accept states is $\{v_{\text{acc}}\} \setminus V^\varepsilon$, i.e., the unique accept state of B (or the empty set if $\Pr[B(U_n) = 1] \leq \varepsilon$). By construction, each layer of B_L has at most $\lfloor 1/\varepsilon \rfloor$ accept states. Moreover, B_L is a regular (permutation) unanimity program if B is a regular (permutation) branching program. Finally, note that $B_L(x) = 1$ if and only if the path in B corresponding to the string x reaches v_{acc} without visiting any state in V^ε. Therefore, by Claim 3.1, B_L is an ε-lower approximator of B. □

Finally, we show that regular unanimity programs of "effective width" w_{eff} can be simulated by regular unanimity programs of actual width $2 \cdot w_{\text{eff}}$ (Lemma 3.3). Lemma 1.11 follows from Lemmas 3.2 and 3.3.

Lemma 3.3. *Let $w_{\text{eff}} \in \mathbb{N}$. Every regular (respectively permutation) unanimity program with at most w_{eff} accept states in each layer has an equivalent regular (respectively permutation) unanimity program of width $2 \cdot w_{\text{eff}}$.*

Proof. Let B be a regular (permutation) unanimity program of length n, with at most w_{eff} accept states in each layer. We may assume without loss of generality that B has exactly w_{eff} accept states in each layer[9] and that the set of accept states in each layer is $[w_{\text{eff}}]$. Assume also that the start state of B, v_0, is an accept state, otherwise the claim is trivial.

We claim that there exists a regular (permutation) program A of length n and width w_{eff} that includes every edge from an accept state of B to another accept state of B. That is, for $t > 0$, $B_t(u, b) = A_t(u, b)$ whenever $u \in [w_{\text{eff}}]$ and $B_t(u, b) \in [w_{\text{eff}}]$. Indeed, such a program A can be constructed greedily.

Now, for an input $x \in \{0,1\}^n$ and $t > 0$, let $v_t = B[v_0, x_{1..t}]$, i.e., v_t is the vertex that B reaches in layer t. Observe that $B(x) = 1$ if and only if $B_t(v_{t-1}, x_t) = A_t(v_{t-1}, x_t)$, for all $t > 0$. We construct a regular (permutation) unanimity program B' of length n and width $2 \cdot w_{\text{eff}}$ that accepts x if and only if $B_t(v_{t-1}, x_t) = A_t(v_{t-1}, x_t)$, for all $t > 0$.

Identify the state space $[2 \cdot w_{\text{eff}}]$ with $[w_{\text{eff}}] \times \{0, 1\}$. The start state of B' is $(v_0, 0)$, and the set of accept states in each layer is $[w_{\text{eff}}] \times \{0\}$. For $t > 0$, the transition function B'_t is given by $B'_t((u_{t-1}, a_{t-1}), b) = (u_t, a_t)$, where

$$\begin{aligned} u_t &= A_t(u_{t-1}, b) \\ a_t &= a_{t-1} \oplus \mathbf{1}[A_t(u_{t-1}, b) \neq B_t(u_{t-1}, b)]. \end{aligned}$$

One can verify that B' is a regular (permutation) unanimity program by an argument similar to the proof of Lemma 2.1. □

[9] This is because we can add w_{eff} dummy states (unreachable from the start state) to each layer, along with transitions between dummy states to maintain the regularity/permutation condition. We can then assign some of the dummy states to be accept states to ensure that each layer has exactly w_{eff} accept states.

4 Hitting Sets for Unbounded-Width Regular Branching Programs

Combining Lemmas 1.11 and 2.1, we get a general transfer theorem, which says that any PRG for width-$O(1/\varepsilon)$ regular programs is also an HSG for unbounded-width regular programs with a single accept vertex.

Theorem 4.1. *Let $n \in \mathbb{N}$ and $\varepsilon > 0$, and let $G\colon \{0,1\}^s \to \{0,1\}^n$. If G is an ε-PRG for width-$(4 \cdot \lfloor 1/\varepsilon \rfloor)$ regular (respectively permutation) branching programs, then G is a (3ε)-HSG for unbounded-width regular (respectively permutation) branching programs with a single accept vertex.*

Proof. Fix an arbitrary regular (resp. permutation) branching program B of length n and unbounded width with a single accept state where $\Pr[B(U_n) = 1] > 3\varepsilon$. Applying Lemma 1.11, there is a regular (resp. permutation) unanimity program B_L of length n and width $2 \cdot \lfloor 1/\varepsilon \rfloor$ such that B_L is an ε-lower approximator of B, i.e. $\Pr[B_L(U_n) = 1] > 2\varepsilon$ and $B_L^{-1}(1) \subseteq B^{-1}(1)$. By Lemma 2.1, G fools B_L with error 2ε. In particular, this implies that G hits B_L and thus B. □

Then Theorem 1.5 follows from the BRRY PRG [BRRY14], Proposition 1.12 follows from the PRG of Steinke [Ste12], and Proposition 1.14 and Proposition 1.13 follow from the PRGs of Forbes and Kelley [FK18] and Chattopadhyay, Hatami, Hosseini, and Lovett [CHHL19] respectively.[10]

Finally, we give a direct proof of Proposition 1.15, which we recall. The set is identical, and the proof of correctness is nearly identical, to the hitting set for width-w regular branching programs of Braverman, Rao, Raz, and Yehudayoff [BRRY14].

Proposition 1.15. *Given $n, a \in \mathbb{N}$, the set $H = \{x \in \{0,1\}^n : \mathrm{wt}(x) \le a\}$ where $\mathrm{wt}(x)$ denotes the Hamming weight of x is a co-hitting set for regular branching programs of length n and unbounded width with a accept states. That is, for every regular branching program B with at most a accept states that is not the constant function $B(x) = 1$, there is $x \in H$ such that $B(x) = 0$.*

Proof. Let B be an arbitrary regular branching program of length n and unbounded width with at most a accept states such that $B(x)$ is not the constant 1 function.

We say a state v is *doomed* if $p_{v\to} = 1$. We say that v is *important* if $B[v, 0]$ is doomed and $B[v, 1]$ is not, or vice versa. We claim that B has at most a layers with at least one important state. To prove this, first note that if there are k doomed states in V_i, there are at least k doomed states in V_{i+1}. This is because for doomed $v \in V_i$, by definition $B[v, 1]$ and $B[v, 0]$ are doomed, and states in V_{i+1} have in-degree at most 2. Furthermore, note that if there are k doomed states in V_i and a non-doomed $v \in V_i$ is important, the number of doomed states in V_{i+1} is at least $k + 1$, because there are at least $2k + 1$ transitions that must end at doomed states in V_{i+1}. We conclude by noting that there at at most a doomed states in V_n, so the claim follows.

Finally, we show that the hitting set has a string that reaches a reject state. Consider an algorithm starting at $u = v_0 \in V_0$. At each step, if u is an important state, take the transition that leads to a non-doomed state, and otherwise take the 0 transition. Since B is not the constant function $B(x) = 1$, this procedure reaches a reject state, and by the claim we take at most a 1 transitions, so there is $x \in H$ such that $B(x) = 0$. Since B was arbitrary, we conclude. □

[10] Theorem 4.1 focuses on ordered programs, but the analogous theorem for the unordered programs follows. To see why, fix some ordered program B and some permutation $\pi\colon [n] \to [n]$. A generator G fools/hits $B(x_{\pi(1)}, \dots, x_{\pi(n)})$ if and only if the generator $G'(x) \overset{\text{def}}{=} (G(x)_{\pi(1)}, \dots, G(x)_{\pi(n)})$ fools/hits B, so we can apply the theorem to G' and draw conclusions about G.

5 Error Reduction for Regular Branching Programs

In this section, as outlined in Section 1.3.3, we use error reduction methods to construct an HSG for regular branching programs with seed length $\widetilde{O}\left(\log n \cdot \left(\sqrt{\log(1/\varepsilon)} + \log w\right) + \log(1/\varepsilon)\right)$. The first step is to show that for any ordered branching program, there is a noticeable chance of visiting a vertex from which the acceptance probability has gone up significantly, refining a lemma of Hoza and Zuckerman [HZ20]. We reiterate that in the following lemma, the set S is not guaranteed to be contained within a single layer.

Lemma 5.1. *Let $K > 1$ be a real number, and let B be a (possibly non-regular) branching program with $\mathbb{E}[B] = p \leq 1/K$. Let V be the set of vertices in B, and let $S = \{v \in V : p_{v\to} \geq Kp\}$. Then when B reads a uniform random input, the probability that it visits S is at least $\frac{1}{2K}$.*

Proof. Let $S' = \{v \in V : Kp \leq p_{v\to} < 2Kp\}$. Because B has degree 2, the acceptance probability $p_{v\to}$ can at most double when we move from a vertex to one of its outneighbors. Therefore, every accepting path from the start vertex v_0 must visit S'.

For each vertex $v \in S'$, define $g_v \colon \{0,1\}^n \to \{0,1\}$ by letting $g_v(x) = 1$ if and only if $B(x)$ visits v and v is the *first* vertex in S' that B visits. Then

$$\begin{aligned} p = \mathbb{E}[B] = \Pr_{x \leftarrow U_n}\left[\bigvee_{v \in S'} (B(x) = g_v(x) = 1)\right] &= \sum_{v \in S'} \Pr_{x \leftarrow U_n}[B(x) = g_v(x) = 1] \\ &= \sum_{v \in S'} \mathbb{E}[g_v] \cdot p_{v\to} \\ &< 2Kp \cdot \sum_{v \in S'} \mathbb{E}[g_v] \\ &= 2Kp \cdot \Pr_{x \leftarrow U_n}[B(x) \text{ visits } S']. \end{aligned}$$

Therefore, when B reads a uniform random input, there is at least a $1/(2K)$ chance that it visits $S' \subseteq S$. □

Now we present our HSG. The construction and analysis closely follow those of Hoza and Zuckerman [HZ20]; the main difference is that we need to invoke Lemma 2.1 (the equivalence between unanimity programs and branching programs) to argue that when B reads a pseudorandom input generated by the ε_0-PRG, there is still a noticeable chance of visiting the set S of Lemma 5.1.

Let $w, n \in \mathbb{N}$, let $\varepsilon_0 < 0.1$, let $G \colon \{0,1\}^s \to \{0,1\}^n$ be an ε_0-PRG for width-$(2w)$ regular branching programs, and let $K = \frac{1}{6\varepsilon_0} > 1$. Let $0 < \varepsilon < \varepsilon_0$; we will construct an ε-HSG for width-w length-n regular branching programs.

The construction uses a tool called a "hitter" [Gol11]. A (θ, δ)-hitter is a function $\mathsf{Hit} \colon \{0,1\}^\ell \times \{0,1\}^q \to \{0,1\}^s$ such that for every set $E \subseteq \{0,1\}^s$, if $|E| \geq \theta \cdot 2^s$, then

$$\Pr_{x \leftarrow U_\ell}[\exists y, \mathsf{Hit}(x, y) \in E] \geq 1 - \delta.$$

(A hitter is a one-sided version of a "sampler," and one can show that it is equivalent to the concept of a "disperser.") Let Hit be a (θ, δ)-hitter with threshold $\theta = \varepsilon_0$ and failure probability $\delta = \frac{1}{2wn}$. Our HSG G' is given by

$$G'(x, t, y_1, \ldots, y_t, n_1, \ldots, n_t) = G(\mathsf{Hit}(x, y_1))_{1..n_1} \circ \cdots \circ G(\mathsf{Hit}(x, y_t))_{1..n_t},$$

where $x \in \{0,1\}^\ell$, t is a positive integer with $t \leq \left\lceil \frac{\log(1/\varepsilon)}{\log K} \right\rceil$, $y_1, \ldots, y_t \in \{0,1\}^q$, and $n_1, \ldots, n_t$ are positive integers with $n_1 + \cdots + n_t = n$. Here $\circ$ denotes string concatenation.

Claim 5.2. *G' is an ε-HSG for width-w length-n regular branching programs.*

Proof. Let B be a regular branching program with $\mathbb{E}[B] > \varepsilon$, and let V be the set of vertices in B. For each vertex $u \in V$, we define a "target set" $S_u \subseteq V$ by the rule

$$S_u = \begin{cases} \{v \in V : p_{v\to} \geq K p_{u\to}\} & \text{if } p_{u\to} \leq 1/K \\ V_{\mathrm{acc}} & \text{if } p_{u\to} > 1/K. \end{cases}$$

Say u is in layer i of the program. We define a function $g_u\colon \{0,1\}^n \to \{0,1\}$ where $g_u(x)$ indicates whether B ever visits the target set S_u when we start at u and read x, i.e.,

$$g_u(x) = \bigvee_{j=0}^{n-i} (B[u, x_{1..j}] \in S_u).$$

By Lemma 5.1, $\mathbb{E}[g_u] \geq 1/(2K) = 3\varepsilon_0$. Furthermore, $1 - g_u$ can be computed by a width-w regular unanimity program. Therefore, by Lemma 2.1, G fools g_u with error $2\varepsilon_0$, so $\mathbb{E}[g_u(G(U_s))] \geq \varepsilon_0$. Let $E_u = \{z \in \{0,1\}^s : g_u(G(z)) = 1\}$. Then by the hitter condition,

$$\Pr_{x \leftarrow U_\ell}[\exists y, \mathsf{Hit}(x,y) \in E_u] \geq 1 - \frac{1}{2wn}.$$

By the union bound, therefore, there exists some $x_* \in \{0,1\}^\ell$ such that for every vertex $u \in V$, there exists a $y \in \{0,1\}^q$ such that $\mathsf{Hit}(x_*, y) \in E_u$.

Now we inductively define a sequence of vertices $u_0, u_1, \ldots$, a sequence of strings $y_1, y_2, \ldots$, and a sequence of positive integers $n_1, n_2, \ldots$ as follows. We begin with $u_0 = v_0$ (the start vertex of B). Assume that we have defined $u_0, u_1, \ldots, u_{i-1}$. Let y_i be such that $\mathsf{Hit}(x_*, y) \in E_{u_{i-1}}$. Recalling the definition of $E_{u_{i-1}}$, this means that if we start at u_{i-1} and read the string $G(\mathsf{Hit}(x_*, y))$, we visit the target set $S_{u_{i-1}}$. Let u_i be the first vertex in the target set $S_{u_{i-1}}$ that we visit, and let n_i be the number of steps from u_{i-1} to u_i. We terminate the process when we reach some vertex $u_i \in V_{\mathrm{acc}}$ in the final layer.

Let t be the number of iterations. In every iteration except possibly the last, the acceptance probability goes up by at least a factor of K, by the definition of the target set S_u. Therefore, $\varepsilon \cdot K^{t-1} < 1$, so $t < 1 + \frac{\log(1/\varepsilon)}{\log K}$. By construction,

$$B(G'(x_*, t, y_1, \ldots, y_t, n_1, \ldots, n_t)) = 1. \qquad \square$$

Proof of Theorem 1.7. The sampling algorithm by Bellare, Goldreich, and Goldwasser [BGG93] implies that for every $s \in \mathbb{N}$ and every $\theta, \delta > 0$, there is an explicit (θ, δ)-hitter $\mathsf{Hit}\colon \{0,1\}^\ell \times \{0,1\}^q \to \{0,1\}^s$ with $\ell = O(s + \log(1/\delta))$ and $q = O(\log(1/\theta) + \log\log(1/\delta))$. In our case, we get $\ell = O(s + \log(wn))$ and $q = O(\log(1/\varepsilon_0) + \log\log(wn))$. Therefore, the seed length of G' is bounded by

$$\begin{aligned} &\ell + O(\log\log(1/\varepsilon)) + \left(1 + \frac{\log(1/\varepsilon)}{\log K}\right) \cdot (q + \log n) \\ &\leq O\left(s + \frac{\log(1/\varepsilon) \cdot (\log n + \log\log w)}{\log(1/\varepsilon_0)} + \log(wn/\varepsilon)\right) \\ &\leq O\left(s + \frac{\log(1/\varepsilon) \cdot \log n}{\log(1/\varepsilon_0)} + \log(wn/\varepsilon)\right), \end{aligned}$$

where the last step holds without loss of generality because if $\log\log w > \log n$ then the claimed seed length is greater than n, which is trivial. Finally, we choose $\epsilon_0 = 2^{-\sqrt{\log(1/\varepsilon)}}$ and take G to be the BRRY PRG [BRRY14], which has seed length $s = O(\log n \cdot (\log(w/\varepsilon_0) + \log\log n))$. $\square$

6 Acknowledgements

We thank Sumegha Garg and Salil Vadhan for many helpful discussions, and Salil Vadhan and Chin Ho Lee for comments on a draft of this paper.

References

[AKM+20] AmirMahdi Ahmadinejad, Jonathan A. Kelner, Jack Murtagh, John Peebles, Aaron Sidford, and Salil P. Vadhan. High-precision estimation of random walks in small space. In *61st IEEE Annual Symposium on Foundations of Computer Science, FOCS 2020, Durham, NC, USA, November 16-19, 2020*, pages 1295–1306. IEEE, 2020.

[AKS87] Miklós Ajtai, János Komlós, and E. Szemerédi. Deterministic simulation in LOGSPACE. In *19th Annual ACM Symposium on Theory of Computing*, pages 132–140, New York City, 25–27 May 1987.

[BCG20] Mark Braverman, Gil Cohen, and Sumegha Garg. Pseudorandom pseudo-distributions with near-optimal error for read-once branching programs. *SIAM J. Comput.*, 49(5), 2020.

[BGG93] Mihir Bellare, Oded Goldreich, and Shafi Goldwasser. Randomness in interactive proofs. *Computational Complexity*, 3(4):319–354, 1993.

[BRRY14] Mark Braverman, Anup Rao, Ran Raz, and Amir Yehudayoff. Pseudorandom generators for regular branching programs. *SIAM J. Comput.*, 43(3):973–986, 2014.

[BV10] Joshua Brody and Elad Verbin. The coin problem and pseudorandomness for branching programs. In *FOCS*, pages 30–39. IEEE Computer Society, 2010.

[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.

[CH20] Kuan Cheng and William M. Hoza. Hitting sets give two-sided derandomization of small space. In Shubhangi Saraf, editor, *35th Computational Complexity Conference, CCC 2020, July 28-31, 2020, Saarbrücken, Germany (Virtual Conference)*, volume 169 of *LIPIcs*, pages 10:1–10:25. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020.

[CHHL19] Eshan Chattopadhyay, Pooya Hatami, Kaave Hosseini, and Shachar Lovett. Pseudorandom generators from polarizing random walks. *Theory Comput.*, 15:1–26, 2019.

[CKK+18] Michael B Cohen, Jonathan Kelner, Rasmus Kyng, John Peebles, Richard Peng, Anup B Rao, and Aaron Sidford. Solving directed laplacian systems in nearly-linear time through sparse lu factorizations. In *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 898–909. IEEE, 2018.

[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.

[DHH20] Dean Doron, Pooya Hatami, and William M. Hoza. Log-Seed Pseudorandom Generators via Iterated Restrictions. In Shubhangi Saraf, editor, *35th Computational Complexity Conference (CCC 2020)*, volume 169 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 6:1–6:36, Dagstuhl, Germany, 2020. Schloss Dagstuhl–Leibniz-Zentrum für Informatik.

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

[Gol11] Oded Goldreich. A sample of samplers: A computational perspective on sampling. In Oded Goldreich, editor, *Studies in Complexity and Cryptography. Miscellanea on the Interplay between Randomness and Computation - In Collaboration with Lidor Avigad, Mihir Bellare, Zvika Brakerski, Shafi Goldwasser, Shai Halevi, Tali Kaufman, Leonid Levin, Noam Nisan, Dana Ron, Madhu Sudan, Luca Trevisan, Salil Vadhan, Avi Wigderson, David Zuckerman*, volume 6650 of *Lecture Notes in Computer Science*, pages 302–332. Springer, 2011.

[Hoz21] William M. Hoza. Better pseudodistributions and derandomization for space-bounded computation. In Mary Wootters and Laura Sanità, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2021, August 16-18, 2021, University of Washington, Seattle, Washington, USA (Virtual Conference)*, volume 207 of *LIPIcs*, pages 28:1–28:23. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 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.

[HZ20] William M. Hoza and David Zuckerman. Simple optimal hitting sets for small-success RL. *SIAM J. Comput.*, 49(4):811–820, 2020.

[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.

[Lee19] Chin Ho Lee. Fourier Bounds and Pseudorandom Generators for Product Tests. In Amir Shpilka, editor, *34th Computational Complexity Conference (CCC 2019)*, volume 137 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 7:1–7:25, Dagstuhl, Germany, 2019. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.

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

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

[PV21a] 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.

[PV21b] Edward Pyne and Salil P. Vadhan. Limitations of the impagliazzo-nisan-wigderson pseudorandom generator against permutation branching programs. In Chi-Yeh Chen, Wing-Kai Hon, Ling-Ju Hung, and Chia-Wei Lee, editors, *Computing and Combinatorics - 27th International Conference, COCOON 2021, Tainan, Taiwan, October 24-26, 2021, Proceedings*, volume 13025 of *Lecture Notes in Computer Science*, pages 3–12. Springer, 2021.

[PV22] Edward Pyne and Salil Vadhan. Deterministic Approximation of Random Walks via Queries in Graphs of Unbounded Size. To Appear, Symposium in Simplicity in Algorithms, SOSA, 2022.

[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].

[RTV06] Omer Reingold, Luca Trevisan, and Salil Vadhan. Pseudorandom walks in regular digraphs and the RL vs. L problem. In *Proceedings of the 38th Annual ACM Symposium on Theory of Computing (STOC '06)*, pages 457–466, 21–23 May 2006. Preliminary version as *ECCC* TR05-22, February 2005.

[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.

[RVW02] Omer Reingold, Salil Vadhan, and Avi Wigderson. Entropy waves, the zig-zag graph product, and new constant-degree expanders. *Annals of Mathematics*, 155(1), January 2002.

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

[Wil18] R. Ryan Williams. Counting Solutions to Polynomial Systems via Reductions. In Raimund Seidel, editor, *1st Symposium on Simplicity in Algorithms (SOSA 2018)*, volume 61 of *OpenAccess Series in Informatics (OASIcs)*, pages 6:1–6:15, Dagstuhl, Germany, 2018. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.

A Reductions From Large Alphabets

In this section, we prove that PRGs and HSGs for regular branching programs over a binary alphabet imply PRGs and HSGs for regular branching programs over larger alphabets, with mild degradation in parameters. Together with the modified proof of Reingold, Trevisan, and Vadhan in Appendix B, this suffices to establish Theorem 1.8 (the reduction from general ordered branching programs to the regular case).

To do so, we first define branching programs of higher degree.

Definition A.1. An **(ordered) branching program** B of length n, width w, and **degree / alphabet size** D computes a function $B : [D]^n \to \{0,1\}$. On an input $x \in [D]^n$, 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 input symbol x_t and updates its state according to a transition function $B_t : [w] \times [D] \to [w]$ by taking $v_t = B_t(v_{t-1}, x_t)$. As in the $D = 2$ case, there is a set V_{acc} of accept states. Let v_n be the final state reached by the branching program on input x. If $v_n \in V_{\text{acc}}$ the branching program accepts, denoted $B(x) = 1$, and otherwise the program rejects, denoted $B(x) = 0$. The program B is **regular** if for every $t \in 1, \ldots, n$ and $v \in [w]$ there are exactly D pairs $(u, \sigma) \in [w] \times [D]$ such that $B_t(u, \sigma) = v$.

We define notation for states and transitions in branching programs analogously to the $D = 2$ case. In particular, for a degree D branching program B we write $B[v, x] = u$ if B reaches state $u \in V_j$ from state $v \in V_i$ over input $x \in [D]^{j-i}$.

Finally, we formally define HSGs and PRGs over larger alphabets. We use U_S to denote the uniform distribution over the set S.

Definition A.2. 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 $H : \{0,1\}^s \to [D]^n$ such that for every $f \in \mathcal{F}$ where $\Pr_{x \leftarrow U_{[D]^n}}[f(x) = 1] > \varepsilon$, there exists $x \in \{0,1\}^s$ such that $f(H(x)) = 1$.

Definition A.3. 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 $G : \{0,1\}^s \to [D]^n$ such that for every $f \in \mathcal{F}$,

$$\left| \Pr_{x \leftarrow U_{[D]^n}}[f(x) = 1] - \Pr_{x \leftarrow U_s}[f(G(x)) = 1] \right| \le \varepsilon.$$

We can now state our main theorem for transferring pseudorandom objects over a binary alphabet into pseudorandom objects over larger alphabets:

Theorem A.4. *Given $n, w, D \in \mathbb{N}$ and $\varepsilon > 0$, there exist values $w' = O(wn^2D/\varepsilon)$ and $n' = O(n \log(nD/\varepsilon))$ and an explicit function $p\colon \{0,1\}^{n'} \to [D]^n$ such that if $G : \{0,1\}^s \to \{0,1\}^{n'}$ is an ε-PRG (resp. HSG) for regular branching programs of length n', width w', and degree 2, then $p \circ G$ is a (3ε)-PRG (resp. HSG) for regular branching programs of length n, width w and degree D.*

A.1 Overview of Proof of Theorem A.4

Theorem A.4 follows from a pair of reductions (Lemmas A.5 and A.6). Here we focus on the case where the initial object is a PRG for simplicity.

First, we show how to convert a PRG over the binary alphabet into a PRG over the alphabet $[R]$, where R is an arbitrary power of two, say $R = 2^r$ where $r \in \mathbb{N}$. To establish this, we take an arbitrary regular branching program B of length n and width w over the alphabet $[R]$. We define a new program $B'\colon \{0,1\}^{nr} \to \{0,1\}$ that simulates B as follows. The state space is $[w] \times \{0,1\}^r$.

Each input in $\{0,1\}^{nr}$ is divided into n blocks of r bits. When B' is in state (u, x) and it reads bit i of block t, it replaces the ith bit of x with the input bit. When it finishes reading a block, it furthermore updates u according to the transition function of the original function $u \leftarrow B_t(u, x)$, and updates x in such a way that regularity is maintained. In effect, the new program stores every block of r bits into an auxiliary component of the state and then uses this register (viewed as a number in R) to make the appropriate transition in the original program. This increases the width by a factor of R but exactly preserves the computed function, so G ε-fooling B' implies $p \circ G$ ε-fools B, where p simply maps each block of r bits to a number in $[R]$.

Second, we show how to convert a PRG for regular branching programs over the alphabet $[R]$ into a PRG over the alphabet $[D]$, where D is arbitrary (not necessarily a power of two) and R is a sufficiently large power of two. We let $p(x) = x \mod D$ where we apply the mod function entrywise. To show $p \circ G$ fools regular branching programs over the alphabet $[D]$, we let m be the largest multiple of D less than R. Given a regular branching program $B : [R]^n \to \{0,1\}$, we can compute $B'(x) = (B \circ p)(x) \wedge \{x \leq m\}$ by a regular branching program that G is required to fool. Furthermore, the condition $x \leq m$ is satisfied with probability at least $1 - \varepsilon$ over uniformly random input, and there is a regular branching program that tests if its input satisfies $x \leq m$ (that G is required to fool).

A.2 Proof of Theorem A.4

We now precisely state and prove the pair of reductions outlined in the preceding section. The first reduction transforms a binary PRG into a PRG for alphabets of size arbitrary powers of two.

Lemma A.5 (Generator for degree two $\implies$ generator for degree any power of two)**.** *Given $n, w, R \in \mathbb{N}$ and $\varepsilon > 0$ where $R = 2^r$, there is an explicit map $p : \{0,1\}^{nr} \to [R]^n$ such that if $G : \{0,1\}^s \to \{0,1\}^{nr}$ is an ε-PRG (resp. HSG) for regular branching programs of length nr, degree 2 and width wR, then $p \circ G$ is an ε-PRG (resp. HSG) for regular branching programs of length n, degree R and width w.*

Proof. Let $p_e : \{0,1\}^r \to [R]$ be an explicit bijection and define

$$p(x) = (p_e(x_1, \ldots, x_r), \ldots, p_e(x_{(n-1)r}, \ldots, x_n)).$$

Note that p maps uniformly random input to uniformly random output.

Now fix an arbitrary regular branching program B of length n, width w, and degree R. For every transition function $B_t : [w] \times [R] \to [w]$, define $\mathrm{Rot}_t : [w] \times \{0,1\}^r \to [w] \times \{0,1\}^r$ such that Rot_t is injective, and $\mathrm{Rot}_t(u, x) = (v, y) \implies B_t(u, p_e(x)) = v$. Such a function exists since $|B_t^{-1}(v)| = R$ for every v. (We use the notation "Rot" because the function is closely connected to the concept of the "rotation map" of a regular digraph [RVW02, RV05].) Then define a new branching program $B' : \{0,1\}^{nr} \to \{0,1\}$ where the states in each layer are $[w] \times \{0,1\}^r$. For $t \in [n]$ and $i \in [r]$ we define the transition function as

$$B'_{r(t-1)+i}((u, x), b) = \begin{cases} (u, x^{i \leftarrow b}) & i < r \\ \mathrm{Rot}_t(u, x^{i \leftarrow b}) & i = r, \end{cases}$$

where $x^{i \leftarrow b}$ denotes replacing the ith bit of x with b. Then B' is regular, because the operation of replacing the ith bit is regular. By choosing the start state to be $(v_0, 0^r)$ and marking as accept all states (u, x) where u is an accept state in B, we obtain that $B \circ p = B'$.

To conclude, we break into cases depending on the base pseudorandom object:

(G is an ε-HSG): Assuming that $\Pr[B(U_{[R]^n}) = 1] > \varepsilon$ then

$$\Pr[B'(U_{nr}) = 1] = \Pr[B(U_{[R]^n}) = 1] > \varepsilon$$

and so there is some x such that $B'(G(x)) = 1$ and thus $B((p \circ G)(x)) = 1$.

(G is an ε-PRG): We have

$$\left| \Pr_{x \leftarrow U_s}[B((p \circ G)(x)) = 1] - \Pr_{x \leftarrow U_{[R]^n}}[B(x) = 1] \right| = \left| \Pr_{x \leftarrow U_s}[B'(G(x)) = 1] - \Pr_{x \leftarrow U_{nr}}[B'(x) = 1] \right| \leq \varepsilon.$$

In both cases since B was arbitrary we obtain the desired result. □

We remark that the preceding component of the reduction does not preserve the property of B being a *permutation* branching program, since the "overwriting the ith bit" operation does not produce a permutation branching program. We next show that PRGs and HSGs over large alphabets imply PRGs and HSGs over smaller alphabets, including alphabet sizes that are not powers of two.

Lemma A.6 (Generator for large degree $\implies$ generator for small degree)**.** *Given $n, w, d \in \mathbb{N}$ and $\varepsilon > 0$, there is $R_0 = O(nD/\varepsilon)$ such that for every $R \geq R_0$, there is an explicit function $p : [R]^n \to [D]^n$ such that if $G : \{0,1\}^s \to [R]^n$ is an ε-PRG (resp. HSG) for regular branching programs of length n, degree R and width $w \cdot (n+1)$, then $p \circ G$ is a (3ε)-PRG (resp. HSG) for regular branching programs of length n, degree D and width w.*

Proof. Let R be large enough that $\frac{D}{R}n < \varepsilon$. Then let $p_e : [R] \to [D]$ be defined as $p_e(x_i) = x_i \mod D$ and define

$$p(x) = (p_e(x_1), \ldots, p_e(x_n)).$$

Let $m \leq R$ be the largest multiple of D not greater than R. For $x \in [R]^n$, we write $x \leq m$ if for all i, $x_i \leq m$. Let $\rho = \Pr_{x \leftarrow U_{[R]^n}}[x \not\leq m]$, and observe $\rho \leq n \cdot D/R < \varepsilon$, and furthermore there exists a length n, width $n \leq w$, degree R regular branching program Q where $Q(x) = \mathbf{1}[x \not\leq m]$.

Now fix an arbitrary regular branching program $B : [D]^n \to \{0,1\}$ of width w with states $V_0, \ldots, V_n$. Let $B' : [R]^n \to \{0,1\}$ be a branching program of length n, degree R and width $w(n+1)$. We identify the states of B' with $V_i \cup ([n] \times V_i)$. The states in V_i simulate B, while the other states are dummy rejection states. We now define the transition function B'_i. For $v \in V_i$ define

$$B'_i(v, \sigma) = \begin{cases} B_i(v, \sigma \mod D) & \sigma \leq m \\ (v, i) & \text{otherwise.} \end{cases}$$

Next, for $(v, j) \in (V_i \times [n])$ define

$$B'_i((v, j), \sigma) = \begin{cases} (v, j) & \sigma \leq m \text{ or } j \neq i \\ v & \text{otherwise.} \end{cases}$$

The accept states of B' are the accept states of B. It can be seen that B' is regular. Note that $B[v, \sigma \mod D] = B[v, p_e(\sigma)]$, so B' computes the function

$$B'(x) = B(p(x)) \cdot \mathbf{1}[x \leq m]. \tag{2}$$

Furthermore, p maps the conditional distribution $(x \leftarrow U_{[R]^n} | x \leq m)$ to the uniform distribution $U_{[D]^n}$. Therefore,

$$\begin{aligned}
&\left| \Pr_{x \leftarrow U_{[R]^n}}[B'(x) = 1] - \Pr_{x \leftarrow U_{[D]^n}}[B(x) = 1] \right| \\
&\quad = \left| \Pr_{x \leftarrow U_{[R]^n}}[B'(x) = 1 | x \not\leq m] \cdot \rho + \Pr_{x \leftarrow U_{[R]^n}}[B'(x) = 1 | x \leq m] \cdot (1-\rho) - \Pr_{x \leftarrow U_{[D]^n}}[B(x) = 1] \right| \\
&\quad \leq \rho + \left| \Pr_{x \leftarrow U_{[R]^n}}[B'(x) = 1 | x \leq m] - \Pr_{x \leftarrow U_{[D]^n}}[B(x) = 1] \right| \\
&\quad \leq \varepsilon + 0. \qquad (3)
\end{aligned}$$

To conclude, we break into cases depending on the base pseudorandom object:

(G is an ε-HSG): If $\Pr[B(U_{[D]^n}) = 1] > 2\varepsilon$, then $\Pr[B'(U_{[R]^n}) = 1] > 2\varepsilon - \varepsilon$ by Equation 3. Thus by assumption on G there is some x where $B'(G(x)) = 1$. Since $B'(x) = 0$ on all $x \not\leq m$, we have $1 = B'(G(x)) = B((p \circ G)(x))$, i.e. $p \circ G$ hits B.

(G is an ε-PRG): We have that G ε-fools Q by assumption, so $\Pr_{x \leftarrow U_s}[G(x) \not\leq m] \leq \varepsilon + \varepsilon$. Then by Equation 2, we obtain:

$$\left| \Pr_{x \leftarrow U_s}[B'(G(x)) = 1] - \Pr_{x \leftarrow U_s}[B((p \circ G)(x)) = 1] \right| \leq \Pr_{x \leftarrow U_s}[G(x) \not\leq m] \leq 2\varepsilon. \qquad (4)$$

We finish by repeated application of the triangle inequality:

$$\begin{aligned}
&\left| \Pr_{x \leftarrow U_s}[B(p \circ G(x)) = 1] - \Pr_{x \leftarrow U_{[D]^n}}[B(x) = 1] \right| \\
&\quad \leq \left| \Pr_{x \leftarrow U_s}[B(p \circ G(x)) = 1] - \Pr_{x \leftarrow U_{[R]^n}}[B'(x) = 1] \right| + \varepsilon \qquad \text{(Equation 3)} \\
&\quad \leq \left| \Pr_{x \leftarrow U_s}[B'(G(x)) = 1] - \Pr_{x \leftarrow U_{[R]^n}}[B'(x) = 1] \right| + 2\varepsilon \qquad \text{(Equation 4)} \\
&\quad \leq 3\varepsilon \qquad \text{(Assumption).} \quad \square
\end{aligned}$$

B Transfer to General Branching Programs

The original formulation of the result of Reingold, Trevisan and Vadhan stated that a "pseudoconverging walk generator" (an object implied by a PRG) with sufficiently short seed implies **BPL** = **L**. We extend their results to HSGs, and derive the degradation in parameters in the notation of branching programs.

Theorem B.1 (Variant of [RTV06])**.** *Given $n, w \in \mathbb{N}$ and $\varepsilon > 0$, there are values $D' = O(n^3 w/\varepsilon^3)$ and $w' = O(n^6 \cdot w^2/\varepsilon^5)$ and an explicit map $p : [D']^n \to \{0,1\}^n$ such that if $G : \{0,1\}^s \to [D']^n$ is an ε-PRG (resp. HSG) for regular branching programs of length n, width w' and degree D', then $p \circ G$ is a (16ε)-PRG (resp. HSG) for (possibly non-regular) branching programs of length n and width w.*

First, we show that generators for regular programs imply generators for "almost-regular" programs, and then we show that generators for almost-regular programs imply generators for non-regular programs.

Definition B.2. Let B be a width-w length-n ordered branching program of alphabet size D and let $\rho > 0$. We say that B is **ρ-almost-regular** if for every $t \in [n]$ and every $v \in [w]$, we have

$$|B_t^{-1}(v)| \leq D \cdot (1+\rho).$$

Lemma B.3 (Generator for regular programs $\implies$ generator for almost-regular programs)**.** *Let $n, w, D \in \mathbb{N}$ and $\varepsilon, \rho > 0$. Let $w' = w \cdot (n+1)$ and let $D' = \lfloor D \cdot (1+\rho) \rfloor$. There is an explicit map $p \colon [D']^n \to [D]^n$ such that if $G \colon \{0,1\}^s \to [D']^n$ is an ε-PRG (resp. HSG) for width-w' length-n regular branching programs of degree D', then $p \circ G$ is a $(2\varepsilon + 2\rho n)$-PRG (resp. HSG) for width-w length-n ρ-almost-regular branching programs of degree D.*

Proof. The map p operates symbol-by-symbol according to the rule

$$p(x)_i = \min\{D, x_i\}$$

for $i \in [n]$. To prove that this works, let B be a width-w length-n ρ-almost-regular branching program of degree D. We will construct a regular branching program B' of degree D' that computes the function

$$B'(x) = \begin{cases} B(x) & \text{if } x \in [D]^n \\ 0 & \text{otherwise.} \end{cases} \tag{5}$$

In particular, this function satisfies

$$\begin{aligned} \left| \Pr_{x \leftarrow U_{[D']^n}}[B'(x) = 1] - \Pr_{x \leftarrow U_{[D]^n}}[B(x) = 1] \right| &= \left| \Pr_{x \leftarrow U_{[D]^n}}[B(x) = 1] \cdot \left(\Pr_{x \leftarrow U_{[D']^n}}[x \in [D]^n] - 1 \right) \right| \\ &\leq \Pr_{x \leftarrow U_{[D']^n}}[x \notin [D]^n] \\ &\leq n \cdot \frac{D' - D}{D'} \\ &\leq n \cdot \rho. \end{aligned} \tag{6}$$

The construction is similar to the proof of Lemma 3.3. First, we claim that there exists a width-w length-n regular branching program A over the alphabet $[D']$ such that for every $t \in [n]$, every $u \in [w]$, and every $\sigma \in [D] \subseteq [D']$, we have $A_t(u, \sigma) = B_t(u, \sigma)$. Indeed, such an A can be constructed greedily, since every vertex of B has at most D' incoming edges. Now, identify the state space $[w']$ with $[w] \times \mathbb{Z}_{n+1}$, where $\mathbb{Z}_{n+1}$ is the additive group of integers modulo $n+1$. The transition function B'_t is given by $B'_t((u_{t-1}, a_{t-1}), \sigma) = (u_t, a_t)$, where

$$\begin{aligned} u_t &= A_t(u_{t-1}, \sigma) \\ a_t &= \begin{cases} a_t & \text{if } \sigma \in [D] \\ a_t + 1 & \text{if } \sigma \in [D'] \setminus [D]. \end{cases} \end{aligned}$$

The start state is $(v_0, 0)$ where v_0 is the start state of B, and the set of accept states is $V_{\text{acc}} \times \{0\}$ where V_{acc} is the set of accept states of B. Equation 5 is clear. The program B' is regular because A is regular. To conclude, we break into cases depending on the base generator:

(G is an ε-HSG): Suppose $\mathbb{E}[B] > \varepsilon + n\rho$. By Equation 6, we have $\mathbb{E}[B'] > \varepsilon$. Therefore, G hits B'. By Equation 5, this implies that there is some seed x such that $G(x) \in [D]^n$ and $B(G(x)) = 1$. Therefore, $B((p \circ G)(x)) = 1$, i.e., $p \circ G$ hits B.

(G is an ε-PRG): One can construct a regular width-$(n+1)$ branching program Q of degree D' such that $Q(x) = 1 \iff x \notin [D]^n$. Observe that $B' \leq B \circ p \leq B' + Q$. Therefore,

$$\begin{aligned}\left| \Pr_{x \leftarrow U_s}[B((p \circ G)(x)) = 1] - \Pr_{x \leftarrow U_s}[B'(G(x)) = 1] \right| &\leq \Pr_{x \leftarrow U_s}[Q(G(x)) = 1] \\ &\leq \Pr_{x \leftarrow U_{[D']^n}}[Q(x) = 1] + \varepsilon \\ &\leq n\rho + \varepsilon.\end{aligned}$$

Furthermore, since G fools B', we have

$$\left| \Pr_{x \leftarrow U_s}[B'(G(x)) = 1] - \Pr_{x \leftarrow U_{[D']^n}}[B'(x) = 1] \right| \leq \varepsilon.$$

Together with Equation 6 and the triangle inequality, this implies

$$\left| \Pr_{x \leftarrow U_s}[B((p \circ G)(x)) = 1] - \Pr_{x \leftarrow U_{[D]^n}}[B(x) = 1] \right| \leq 2\varepsilon + 2n\rho$$

as claimed. □

Lemma B.4 (Generator for almost-regular programs $\implies$ generator for non-regular programs)**.** *Given $n, w \in \mathbb{N}$ and $\varepsilon, \rho > 0$, there are values $D = O(nw/(\varepsilon\rho^2))$ and $w' = O(n^2w^2/(\varepsilon^2\rho^3))$ and an explicit function $p : [D]^n \to \{0,1\}^n$ such that if $G \colon \{0,1\}^s \to [D]^n$ is an ε-PRG (resp. HSG) for width-w' length-n ρ-almost-regular branching programs of degree D, then $p \circ G$ is a (4ε)-PRG (resp. HSG) for width-w length-n (possibly non-regular) branching programs.*

Proof. Let $T = \lceil (nw/\varepsilon) \cdot (2/\rho + 1) \rceil$, and let D be the smallest even integer with $D \geq (4T+4)/\rho$. Let $\phi \colon [D] \to \{0,1\} \times [D/2]$ be a bijection, and let $p : [D]^n \to \{0,1\}^n$ project each symbol to its first bit, i.e., $p(x)_i = \phi(x_i)_1$.

Let B be an ordered branching program of length n and width w over the binary alphabet. We define a new branching program $B' \colon [D]^n \to \{0,1\}$ of width $T \cdot (D+1)$ and length n as follows. We identify the state space $[T \cdot (D+1)]$ with $[T] \cup ([T] \times [D])$. Informally, the states in $[T]$ carry a simulation of B, while the states in $[T] \times [D]$ are dummy rejection states to handle certain rare events.

Precisely, we associate each vertex $v \in V_t$ of B with a "cloud" $C(v) \subseteq [T]$ as follows. Let $S = \lfloor T \cdot p_{\to v} \rfloor$. If $S \leq 2/\rho$, let $C(v) = \emptyset$, and otherwise let $|C(v)| = S$. Since $\sum_{v \in V_t} \lfloor T \cdot p_{\to v} \rfloor \leq T$, the clouds $C(v)$ can be chosen in such a way that they are disjoint for $v \in V_t$. Number the states in $C(v)$ as $C(v) = \{C(v)_1, \ldots, C(v)_S\}$, and extend the notation by defining $C(v)_i = C(v)_{i \bmod S}$ when $i > S$. Mark an arbitrary state in $C(v_0)$ as the start state of B', and if v is an accept state of B, then mark all states in the cloud $C(v)$ as accept states.

Now we define transitions. Fix some $t \in [n]$ and some symbol $\sigma \in [D]$.

- For a state $u' \in [T]$:
  - Let $(b, i) = \phi(\sigma) \in \{0,1\} \times [D/2]$. First suppose that u' is a member of a cloud, say $u' \in C(u)$ where $u \in V_{t-1}$, and suppose furthermore that $C(B[u,b]) \neq \emptyset$. In this case, we define
  $$B'_t(u', \sigma) = C(B[u,b])_i \in [T].$$
  - Otherwise, we define
  $$B'_t(u', \sigma) = (u', \sigma) \in [T] \times [D].$$

- For a state of the form $(u', j) \in [T] \times [D]$, we define
$$B'_t((u', j), \sigma) = (u', j).$$

In summary, each edge (u, v) of B is "lifted" to a collection of edges from the cloud $C(u)$ to the cloud $C(v)$ whenever possible, and furthermore these edges are distributed as evenly as possible. We use dummy rejection states to deal with the case $C(v) = \emptyset$.

Unfortunately, due to roundoff errors, B' is not necessarily regular. However, we next show that B' is *almost* regular.

Claim B.5. *The branching program B' is ρ-almost-regular.*

Proof. We begin by analyzing states in clouds. Let $t \in [n]$, and let $v' \in C(v)$ where $v \in V_t$. Let $(u_1, v), \ldots, (u_r, v)$ be the edges incoming to v in B. In B', all edges to the cloud $C(v)$ come from the clouds $C(u_1), \ldots, C(u_r)$. For each $i \in [r]$, for each vertex $u'_i \in C(u_i)$, there are $D/2$ edges from u'_i to the cloud $C(v)$, and those edges are distributed as evenly as possible among the members of $C(v)$. In particular, if we let $e(u'_i, v')$ denote the number of edges from u'_i to v', we have
$$e(u'_i, v') \leq \frac{D/2}{|C(v)|} + 1.$$

Therefore, summing up,
$$\begin{aligned}
\deg^-(v') = \sum_{i=1}^{r} \sum_{u'_i \in C(u_i)} e(u'_i, v') &\leq \sum_{i=1}^{r} |C(u_i)| \cdot \left(\frac{D/2}{|C(v)|} + 1\right) \\
&\leq \left(\frac{D/2}{|C(v)|} + 1\right) \cdot T \cdot \sum_{i=1}^{r} p_{\to u_i} \\
&= \left(\frac{D/2}{|C(v)|} + 1\right) \cdot 2T \cdot p_{\to v} \\
&\leq \left(\frac{D/2}{|C(v)|} + 1\right) \cdot 2 \cdot (|C(v)| + 1) \\
&= D \cdot \left(1 + \frac{1}{|C(v)|} + \frac{2|C(v)| + 2}{D}\right) \\
&\leq D \cdot \left(1 + \frac{1}{|C(v)|} + \frac{2T + 2}{D}\right).
\end{aligned}$$

Since $C(v)$ is nonempty, $|C(v)| > 2/\rho$, so plugging in the definition of D, we get a bound of $D \cdot (1 + \rho)$.

Next, we consider states outside clouds. By construction, a state $v' \in [T]$ that is not in a cloud has *zero* incoming edges. Finally, a state $(v', j) \in [T] \times [D]$ has at most $D + 1 \leq D \cdot (1 + \rho)$ incoming edges, namely the D transitions $B'_t((v', j)\sigma) = (v', j)$ and possibly the one additional transition $B'_t(v', j) = (v', j)$. □

Thus B' is a ρ-almost-regular program of degree D and length n and width $T \cdot (D + 1)$, so G is an ε-PRG (resp. ε-HSG) for B'. Next, we show that the program B' approximately simulates B (or rather $B \circ p$). Define the function $Q \colon [D]^n \to \{0, 1\}$ by the rule
$$Q(x) = 1 \iff \text{on input } p(x),\ B \text{ visits some } v \text{ such that } C(v) = \emptyset.$$

Claim B.6. *We have $B' \leq B \circ p \leq B' + Q$.*

Proof. Let $x \in [D]^n$, and let $v_0, \ldots, v_n$ be the sequence of vertices that B visits on input $p(x)$, i.e., $v_i = B[v_0, p(x)_{1..i}]$. Looking at the definition of B', we see that $B'(x)$ visits vertices in the clouds $C(v_0), C(v_1), \ldots, C(v_n)$, unless at some point one of these clouds is empty, in which case B' transitions to a reject vertex and we have $B'(x) = 0$. Therefore, if $Q(x) = 0$, then $B'(x) = B(p(x))$, and meanwhile if $Q(x) = 1$, then $B'(x) = 0$. □

Consequently, under the uniform distribution, we have

$$\begin{aligned} \left| \Pr_{x \leftarrow U_{[D]^n}}[B'(x) = 1] - \Pr_{x \leftarrow U_n}[B(x) = 1] \right| &= \left| \Pr_{x \leftarrow U_{[D]^n}}[B'(x) = 1] - \Pr_{x \leftarrow U_{[D]^n}}[(B \circ p)(x) = 1] \right| \\ &\leq \Pr_{x \leftarrow U_{[D]^n}}[Q(x) = 1]. \end{aligned} \tag{7}$$

By the union bound, we have

$$\begin{aligned} \Pr_{x \leftarrow U_{[D]^n}}[Q(x) = 1] &\leq \sum_{v : \lfloor T \cdot p_{\to v} \rfloor \leq 2/\rho} p_{\to v} \\ &\leq nw \cdot \frac{2/\rho + 1}{T} \\ &\leq \varepsilon. \end{aligned} \tag{8}$$

To conclude, we break into cases depending on the base pseudorandom object:

(G is an ε-HSG): If $\Pr[B(U_n) = 1] > 2\varepsilon$ then $\Pr_{x \leftarrow U_{[D]^n}}[B'(x) = 1] > \varepsilon$ by Equations 7 and 8. Then by assumption on G there is x such that $B'(G(x)) = 1$, which implies $B((p \circ G)(x)) = 1$ by Claim B.6.

(G is an ε-PRG): We can construct a branching program that computes Q that is very similar to B'. Namely, we modify B' by setting the accept states in the final layer to be $[T] \times [D]$ (leaving the transitions unchanged). That branching program is ρ-almost-regular, so G fools Q with error ε. Therefore, by Claim B.6 and Equation 8,

$$\left| \Pr_{x \leftarrow U_s}[B((p \circ G)(x)) = 1] - \Pr_{x \leftarrow U_s}[B'(G(x)) = 1] \right| \leq \Pr_{x \leftarrow U_s}[Q(G(x)) = 1] \leq 2\varepsilon. \tag{9}$$

Then we conclude by a chain of inequalities:

$$\begin{aligned} &\left| \Pr_{x \leftarrow U_s}[B((p \circ G)(x)) = 1] - \Pr_{x \leftarrow U_{\{0,1\}^n}}[B(x) = 1] \right| \\ &\quad \leq \left| \Pr_{x \leftarrow U_s}[B((p \circ G)(x)) = 1] - \Pr_{x \leftarrow U_{[D]^n}}[B'(x) = 1] \right| + \varepsilon && \text{(Equations 7 and 8)} \\ &\quad \leq \left| \Pr_{x \leftarrow U_s}[B'(G(x)) = 1] - \Pr_{x \leftarrow U_{[D']^n}}[B'(x) = 1] \right| + 3\varepsilon && \text{(Equation 9)} \\ &\quad \leq 4\varepsilon && \text{(Assumption).} \end{aligned}$$

□

Theorem B.1 follows immediately from Lemmas B.3 and B.4 by choosing $\rho = \varepsilon/n$.