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

Hitting Sets For Regular Branching Programs

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

October 13, 2021

Abstract

We construct an explicit ε-hitting set generator (HSG) for regular ordered branching programs of length n and *unbounded width* with a single accept state that has seed length

$$O(\log(n)(\log\log(n) + \log(1/\varepsilon))). \tag{1}$$

Previously, the best known seed length for regular branching programs of width w with a single accept state was by Braverman, Rao, Raz and Yehudayoff (FOCS 2010, SICOMP 2014) and Hoza Pyne and Vadhan (ITCS 2021), which gave

$$O(\log(n)(\log\log(n) + \min\{\log(w), \log(n)\} + \log(1/\varepsilon))).$$

We also give a simple co-HSG for the model with optimal seed length $O(\log n)$.

For the more restricted model of *permutation* branching programs, Hoza Pyne and Vadhan (ITCS 2021) constructed a PRG with seed length matching (1), and then Pyne and Vadhan (CCC 2021) developed an error-reduction procedure that gave an HSG (in fact a "weighted PRG") with seed length $\widetilde{O}(\log(n)\sqrt{\log(n/\varepsilon)} + \log(1/\varepsilon))$. We show that if an analogous error-reduction result could be obtained for our HSG, there is an explicit HSG for general ordered branching programs of width $w = n$ with seed length $\widetilde{O}(\log^{3/2} n)$, improving on the $O(\log^2 n)$ seed length of Nisan (Combinatorica 1992).

Keywords: pseudorandomness, space-bounded computation

*Supported by NSF grant CCF-1763299.

ISSN 1433-8092

1 Introduction

Starting with the work of Babai, Nisan, and Szegedy [BNS92], there has been three decades of work on constructing and analyzing pseudorandom generators and variants for space-bounded computation, motivated by obtaining unconditional derandomization (e.g. seeking to prove that BPL = L) and a variety of other applications (e.g. [Ind06, Siv02, HVV06, HHR11]). 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. For $l \in \mathbb{N}$ we write $[l]$ to denote $\{0, \ldots, l-1\}$.

Definition 1.1. An **ordered branching program (OBP)** 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 symbol 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_{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 $B(x) = 0$. We also consider branching programs restricted to having a single accept state, which is always denoted v_{acc}.

We focus our attention on branching programs with additional structure:

Definition 1.2. 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 $B_t(u, b) = v$.

Ordered regular branching programs are a powerful model with connections to space-bounded derandomization, which we will detail later. We now recall the formal definition of hitting set and pseudorandom generators. We let U_i denote the uniform distribution over $\{0,1\}^i$ and U_S be the uniform distribution over the set S.

Definition 1.3. 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 $f(H(x)) = 1$.

Definition 1.4. 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 $f \in \mathcal{F}$,

$$\left| \Pr_{x \leftarrow U_{\{0,1\}^n}}[f(x) = 1] - \Pr_{x \leftarrow U_{\{0,1\}^s}}[f(G(x)) = 1] \right| \leq \varepsilon.$$

We say that G **ε-fools** $\mathcal{F}$ if it is an ε-PRG for $\mathcal{F}$. The value s is the **seed length** of the PRG (resp. HSG). We say a generator G is **explicit** if the ith bit of output is computable in space $O(s)$.

Note that with our definition, an ε-PRG for a class $\mathcal{F}$ is an ε-HSG for $\mathcal{F}$. It can be shown via the probabilistic method that there is a (nonexplicit) 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. Decades of work has focused on constructing explicit hitting set and pseudorandom generators for branching programs, motivated by the derandomization of space-bounded computation. All results we subsequently discuss are explicit constructions. In 1990, Nisan [Nis92] constructed a PRG for general ordered branching programs:

Theorem 1.5 ([Nis92])**.** *There exists an explicit ε-PRG for ordered branching programs of length n and width w that has seed length*

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

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

In 2010, Braverman, Rao, Raz and Yehudayoff [BRRY10] achieved near-optimal dependence on n for *regular* branching programs:[1]

Theorem 1.6 ([BRRY10])**.** *There exists an explicit ε-PRG for regular branching programs of length n and width w that has seed length*

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

However, if we consider $w = \text{poly}(n)$ or $\varepsilon = 1/\text{poly}(n)$, the seed length remains $O(\log^2 n)$ as Nisan's PRG. In addition, they obtained an HSG for regular branching programs with seed length $O(w \log(n))$.[2]

In 2020, Hoza, Pyne and Vadhan [HPV21] were able to eliminate the dependence on the width w for *permutation* branching programs with a single accept state. Permutation branching programs are a restricted class of regular branching programs where the transitions $B_t(\cdot, 1)$ and $B_t(\cdot, 0)$ corresponding to each fixed input bit form a permutation on $[w]$.

Theorem 1.7 ([HPV21])**.** *There exists an explicit ε-PRG for permutation branching programs of length n and unbounded width with a single accept state that has seed length*

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

In addition, their work implies an explicit ε-HSG for regular branching programs of unbounded width with a single accept state that has seed length

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

However, neither result beats Nisan's $O(\log^2 n)$ barrier where $w = \text{poly}(n)$ and we allow an arbitrary set of accept states.

Recently, Pyne and Vadhan [PV21b] obtained an improved hitting set generator for permutation branching programs of unbounded width:

Theorem 1.8 ([PV21b])**.** *There exists an explicit ε-HSG for permutation branching programs of length n and unbounded width with a single accept state that has seed length*

$$\tilde{O}(\log(n)\sqrt{\log(n/\varepsilon)} + \log(1/\varepsilon)).$$

Here, if we consider $w = \text{poly}(n)$ and an arbitrary set of accept states, their HSG obtains seed length $\tilde{O}(\log^{3/2} n)$, beating Nisan in the regime motivated by space-bounded derandomization. This result is obtained by an "error reduction" procedure applied to the PRG of Hoza et al. (Theorem 1.7).

[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 Braverman et al. 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. $w > \log(1/\varepsilon)$.

1.1 Our Contribution

In the case of regular branching programs with a single accept state, we eliminate the dependence on w of Braverman et al. (Theorem 1.6), at the cost of only obtaining a hitting set generator.

Theorem 1.9. *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*

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

Note that this result matches the seed length of the PRG of Hoza et al. (Theorem 1.7). For regular branching programs of width $w = \mathrm{poly}(n)$ (the regime most relevant for the derandomization of space-bounded computation), Theorem 1.9 is the first explicit construction with seed length $o(\log^2 n)$.

Since a regular branching program with a accept states can be written as a sum of a regular branching programs each with a single accept state, we obtain the following corollary:[3]

Corollary 1.10. *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*

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

To motivate our focus on regular branching programs of large width with a single accept state, we prove that reducing the error of our construction analogously to how Pyne and Vadhan (Theorem 1.8) did to the [HPV21] PRG (Theorem 1.7) would imply major advances in the derandomization of space-bounded computation. We achieve this by extending results of Reingold Trevisan and Vadhan [RTV06] to show a reduction from general ordered branching programs to regular branching programs.

Theorem 1.11. *Suppose there is an explicit $\varepsilon/20$-PRG (resp. HSG, weighted PRG) $G : \{0,1\}^s \to \{0,1\}^t$ for regular branching programs of length $t = \mathrm{poly}(n \log(w/\varepsilon))$ and width $\mathrm{poly}(nw/\varepsilon)$. Then there is an explicit ε-PRG (resp. HSG, weighted PRG) $G' : \{0,1\}^s \to \{0,1\}^n$ for ordered branching programs of length n and width w.*

The main novelties of this reduction over that of [RTV06] are that it applies to HSGs and WPRGs, not just PRGs, and that it only requires a PRG for regular branching programs over the binary alphabet, whereas the [RTV06] reduction requires one over an alphabet of size $\mathrm{poly}(nw/\varepsilon)$.

Combining Theorem 1.11 with prior work, we read off several corollaries. We note an analogue of the error-reduction result of Pyne and Vadhan (Theorem 1.8) would imply hitting sets for general ordered branching programs beating the $O(\log^2 n)$ barrier of Nisan.

Corollary 1.12. *Suppose 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 $\tilde{O}(\log(n)\sqrt{\log(n/\varepsilon)} + \log(1/\varepsilon))$. Then there is an explicit ε-HSG for ordered branching programs of length n and width w with seed length*

$$\tilde{O}(\log(n)\sqrt{\log(nw/\varepsilon)} + \log(w/\varepsilon)).$$

[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 [PV21a], even for HSGs.

Recall that BPL is the set of languages decided by randomized logspace machines with two-sided error, and L is the set of languages decided by deterministic logspace machines. The construction of Saks and Zhou [SZ99], generalized by Armoni [Arm98], shows that a PRG improving on Nisan's dependence on the width w implies an improved derandomization of BPL. Braverman, Cohen and Garg [BCG18] and Chattopadhyay and Liao [CL20] showed that a *weighted* PRG suffices. Using Theorem 1.11, we deduce that a weighted PRG for *regular* BPs suffices:

Corollary 1.13. *Suppose there is an explicit ε-weighted PRG for regular branching programs of length n and unbounded width with a single accept state that has seed length $O(\log^2 n + \log(1/\varepsilon))$. Then BPL $\subseteq L^{4/3}$.*

Recently, Cheng and Hoza [CH20] proved that optimal HSGs for ordered branching programs imply derandomization of logspace algorithms with *two*-sided error. Again using Theorem 1.11, we deduce that an HSG for regular BPs suffices:

Corollary 1.14. *Suppose there is an explicit ε-HSG for regular branching programs of length n and width w with seed length $O(\log(nw/\varepsilon))$. Then BPL = L.*

It is natural to wonder if our HSG (Theorem 1.9) can be strengthened to be a PRG. To this end, we observe that the Braverman et al. [BRRY10] *co-HSG* for constant width regular branching programs extends to regular BPs with a constant number of accept states.

Proposition 1.15. *Given $n, a \in \mathbb{N}$, the set $H = \{\sigma \in \{0,1\}^n : \mathrm{wt}(\sigma) \leq a\}$ where $\mathrm{wt}(\sigma)$ denotes the Hamming weight of σ 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 $\sigma \in H$ such that $B(\sigma) = 0$.*

In contrast, Hoza et al. [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 $\Omega(n)$. Thus, we obtain a very simple explicit construction with exponentially shorter seed length than that obtained via the probabilistic method.

1.2 Perspective

The recent progress on pseudorandomness for permutation branching programs [HPV21, PV21b] has heavily relied on the permutation property of such programs to apply powerful results in spectral graph theory [RV05, AKM+20, CKK+18]. (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.) Regular branching programs have seemed much more difficult to handle, reinforced by the result of [RTV06] (cf. Theorem 1.11) showing that pseudorandom generators for them can be used to derandomize all space-bounded computation. Our work gives hope that this barrier can be bypassed, as we show that the parameters of the [HPV21] PRG (Theorem 1.7) for permutation branching programs can be matched by a HSG for regular branching programs (Theorem 1.9), and we prove this using purely combinatorial arguments rather than spectral graph theory. If we can match the error reduction of [PV21b] (Theorem 1.8) for regular branching programs, then we will obtain a major improvement for HSGs for general ordered branching programs of polynomial width, obtaining seed length $\tilde{O}(\log^{3/2} n)$ versus the $O(\log^2 n)$ of Nisan's PRG (Theorem 1.5).

1.3 Overview of Proof of Theorem 1.9

The proof of Theorem 1.9 consists of two separate results which may be of independent interest. Both results relate to a model of computation intermediate between regular and general branching programs, which we formally define now.

Definition 1.16. An **(ordered) regular plus sudden reject branching program** B of length n and width w has its set of states as $\{[w], \perp\}$ in each layer. For every $t \in 1, \dots, n$, for every $v \in [w]$ there are at most two pairs $(u, b) \in [w] \times \{0, 1\}$ such that $B_t(u, b) = v$. Furthermore, $B_t(\perp, b) = \perp$ for all t, b. There is a single accept state $v_{\text{acc}} \in V_n$ and $v_{\text{acc}} \neq \perp$.

In effect, a regular plus sudden reject BP consists of a width-w regular BP with a reject sink $\perp$ added to each layer; transitions that previously reached states in $[w]$ in the next layer can be redirected to $\perp$. A regular plus sudden reject program of width w is also a general ordered branching program of width $w + 1$.

In Section 2, we show that for every $\varepsilon > 0$, every unbounded width regular branching program B with a single accept state has a ε-"lower approximator" of width $w = O(1/\varepsilon)$. This is a regular plus sudden reject branching program B_L that accepts on a susbet of the strings accepted by B, and has acceptance probability (under the uniform distribution) within ε of that of B.

Proposition 1.17. *Given $\varepsilon > 0$ and a regular branching program of length n and unbounded width with a single accept state B, there is a length n, width $\lceil 1/\varepsilon \rceil$ regular plus sudden reject branching program B_ε such that B_ε is an ε-lower approximator of B.*

We obtain Proposition 1.17 by a more careful analysis of a result of Hoza et al. [HPV21], who prove that unbounded-width regular branching programs have ε-lower approximators of width $w = O(n/\varepsilon)$. The key observation behind our improvement is that regular branching programs cannot concentrate low probability events. We show that removing *all* states in all layers with probability at most ε of being reached under a uniformly random input decreases the probability of reaching the accept state by at most ε (wiith no dependence on n). Thus, we can restrict a branching program by discarding all but the most likely $\lceil 1/\varepsilon \rceil$ states in each layer without cutting the accept probability by more than ε. Every transition that would reach a discarded state is instead sent to $\perp$, which is why we require a sudden reject state.

Due to these lower approximators not being regular branching programs, we would naively need to use a generator for general ordered branching programs to fool them. Unfortunately, existing generators for that model have seed length $\Omega(\log^2 n)$ even for $w = 4$ and $\varepsilon = \Omega(1)$. To circumvent this issue, we show in Section 3 that the INW PRG fools such "morally regular" branching programs with seed length matching that of the result of Braverman et al. [BRRY10] for regular branching programs:

Theorem 1.18. *Given $n, w \in \mathbb{N}$ and $\varepsilon \in (0, 1/2)$, there is an explicit ε-PRG for regular plus sudden reject branching programs of length n and width w that has seed length*

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

We show this result using a modification of the approach of Braverman et al. [BRRY10]. For states u, v connected by a transition in the branching program, they define the *weight* of the transition as $|p_{u\to} - p_{v\to}|$, where $p_{u\to}$ is the probability of accepting from u over uniformly random input, and likewise for v. They show for a regular branching program of width w, the sum of the weights of every transition is $O(w)$, which crucially has no dependence on n. We extend this bound on weight to regular plus sudden reject branching programs, via a modification of their "pebble

game." We then appeal to their result that the Impagliazzo–Nisan–Wigderson (INW) generator fools programs with weight $O(w)$ with the desired seed length.

Then to prove Theorem 1.9, a standard argument shows that an ε-PRG against an ε-lower approximator B_L of B is also a a 3ε-HSG for B. By Proposition 1.17, we can take B_L to have width $O(1/\varepsilon)$. Thus by Theorem 1.18, we get a PRG against B_L (and hence HSG against B) with seed length

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

as desired.

1.4 Overview of Proof of Theorem 1.11

Our "transfer theorem" Theorem 1.11 is a modification of a result of Reingold Trevisan and Vadhan [RTV06]. We give a version of their proof in the notation of branching programs in Appendix B:

Theorem 1.19 (Variant of [RTV06])**.** *Given $n, w \in \mathbb{N}$ and $\varepsilon > 0$, there is $D' = O((\varepsilon/nw)^4)$ such that if $G : \{0,1\}^s \to [D']^n$ is an explicit ε-PRG (resp. HSG) for regular branching programs of length n, width $(nw/\varepsilon)^8$ and degree D', there is an explicit 10ε-PRG (resp. HSG) $G' : \{0,1\}^s \to \{0,1\}^n$ for ordered branching programs of length n and width w.*

Outside the appendix we assume all branching programs are of degree $d = 2$, and we give the formal definitions of higher degree branching programs, and pseudorandom objects for such, in Appendix A. To complete the proof of Theorem 1.11, we prove that sufficiently good pseudorandom objects for regular branching programs over a binary alphabet imply equivalent constructions for regular branching programs over larger alphabets. Specifically:

Theorem 1.20. *Given $n, w, d \in \mathbb{N}$ and $\varepsilon > 0$, there is $t = O(n \log(nd/\varepsilon))$ such that if there is an explicit ε-PRG (resp. HSG) $G : \{0,1\}^s \to \{0,1\}^t$ for regular branching programs of length t, width $10wn^2d/\varepsilon$ and degree 2, there is an explicit 4ε-PRG (resp. HSG) $G'' : \{0,1\}^s \to [d]^n$ for regular branching programs of length n, width w and degree d.*

The proof of Theorem 1.20 consists of a pair of reductions, and here we focus on the case where the initial object is a PRG for simplicity.

First, given $R = 2^r \in \mathbb{N}$ we show that given an explicit ε-PRG $G : \{0,1\}^s \to \{0,1\}^n$ for regular branching programs of length n and width w over a binary alphabet, there is an explicit ε-PRG for regular branching programs of length n/r and width w/R over the alphabet $[R]$. To establish this, we take an arbitrary regular branching program B of length n/r and width w/R over the alphabet $[R]$. We blow up each state in layer i of B into a "cloud" of $R/2$ states in layer ir of a new regular branching program $B' : \{0,1\}^{(n/r)r} \to \{0,1\}$. Then for every transition $B_i(v,x) = u$ in the original branching program, we add a gadget in layers $ir, \ldots (i+1)r$ of B' that transitions from *every* state in cloud $C(v)$ to *some* state in cloud $C(u)$ on input x, viewing $x \in [R]$ as a binary string in $\{0,1\}^r$. We then mark all states in the clouds corresponding to accept states in B as accept states in B'. This increases the width (and number of accept states) by a factor of $R/2$ but exactly preserves the computed function, so G ε-fooling B' implies $q \circ G$ ε-fools B, where q simply maps each block of r bits to a number in $[R]$.

Second, given $d \in \mathbb{N}$ (which need not be a power of 2) we show that given an explicit ε-PRG $G : \{0,1\}^s \to [R]^n$ for regular branching programs of length n, width w and degree R where $R = O(nd/\varepsilon)$ is a sufficiently large power of 2, there is an explicit 4ε-PRG for regular branching programs of length n, width w and degree d. Here we take an arbitrary regular branching program

$B : [d]^n \to \{0,1\}$ of width w and duplicate each transition $\lfloor R/d \rfloor$ times to create a new regular program $B' : [R]^n \to \{0,1\}$. This leaves at most d "extra" symbols in $[R]$ that have not been assigned, so for every state in B we wire the transitions corresponding to these symbols to nw new dummy states that always reject. Letting $p : [R]^n \to [d]^n$ be the function that maps each symbol to its preimage under the blowup (and maps the extra symbols arbitrarily), we wish to show that $p \circ G$ fools B.

To show this claim, we call $x \in [R]^N$ BAD if it contains an extra symbol. For every x that is not BAD, we have $B(p(x)) = B'(x)$ by construction. Furthermore, a random string $x \leftarrow U_{[R]^n}$ is BAD with probability at most $n(d/R) < \varepsilon$, and there exists a regular branching program Q of width $n \leq w$. Since G ε-fools Q, G has at most a 2ε fraction of BAD outputs, and so a chain of inequalities shows that $p \circ G$ 4ε-fools B. Combining these two reductions, we obtain the desired result.

1.5 Preliminaries

First, we define notation relating to states and transitions in a branching program.

Definition 1.21. For a branching program B of length n, let $V_0, V_1, \ldots, V_n$ be the sets of states in layers $0, 1, \ldots, n$ respectively. 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 the accept state from the start state, and notation for the probability of accepting from that state.

Definition 1.22. For a branching program B with start state v_0 and accept state v_{acc}, 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 $p_{v\to} = \Pr[B[v, U_{n-i}] = v_{\text{acc}}]$ be the probability the program reaches the accept state 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 $p_{v_{\text{acc}}\to} = 1$ and $p_{v\to} = 0$ for $v \in V_n \setminus v_{\text{acc}}$.

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 define notation for the strings accepted by a given branching program.

Definition 1.23. Given a branching program B of length n, let the **accept set** of B be defined as $A(B) = \{x \in \{0,1\}^n : B[v_0, x] \in V_{\text{acc}}\}$.

Definition 1.24. Given two branching programs B, B' of length n, let

$$A_\Delta(B, B') = \frac{|A(B) \Delta A(B')|}{2^n}$$

be the relative mass of the symmetric difference of $A(B)$ and $A(B')$.

We require the following basic proposition:

Proposition 1.25. *Given two branching programs B, B' of length n, $|\Pr[B(U_n) = 1] - \Pr[B'(U_n) = 1]| \leq A_\Delta(B, B')$.*

With this, we formally define the concept of a lower approximator.

Definition 1.26. 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 $A(B_L) \subseteq A(B)$ and $A_\Delta(B, B_L) \leq \varepsilon$.

1.6 Organization

In Section 2, we show that regular branching programs with a single accept state can be approximated from below by regular plus sudden reject branching programs of width $O(1/\delta)$, where δ is the approximation parameter. In Section 3 we prove the INW PRG fools regular plus sudden reject branching programs. In Section 4 we combine these two results and conclude Theorem 1.9, and give a short proof of Proposition 1.15. In Appendix A we prove Theorem 1.20, 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 Lower Approximators for Regular Branching Programs

We first show that regular branching programs of unbounded width have bounded-width lower approximators. Hoza et al. [HPV21] showed (Theorem 4.1) that unbounded-width regular branching programs have *regular* ε-lower approximators of width $O(n^2/\varepsilon)$, which is too large for our application.[4] To obtain a lower approximator of width $O(1/\varepsilon)$, we make two changes. 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 compress the unimportant states into a single state $\perp$. This results in a non-regular lower approximator, though it retains enough structure that it is easy to fool.

Proposition 1.17. *Given $\varepsilon > 0$ and a regular branching program of length n and unbounded width with a single accept state B, there is a length n, width $\lceil 1/\varepsilon \rceil$ regular plus sudden reject branching program B_ε such that B_ε is an ε-lower approximator of B.*

Proof. First note that if $p_{\to v_{\text{acc}}} \leq \varepsilon$ the result is immediate, so we subsequently assume this is not the case. Let $V_\varepsilon = \{v : p_{\to v} \leq \varepsilon\}$ be the set of states of B that have at most ε probability of being reached over random input. We claim that for every state of B, the probability that a random input reaches this state from v_0 after passing through *at least one* element of V_ε is at most ε. Formally:

Claim 2.1. *For every $i \in \{1, \ldots, n\}$ and $v \in V_i$,*

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

Proof. We show this via induction on i. The property is vacuously 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

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

[4] They also show unbounded-width regular programs have regular plus sudden reject lower approximators of width $O(n/\varepsilon)$, which is likewise too wide.

Otherwise, let $u_1, u_0 \in V_i$ be the (not necessarily distinct) states in layer i with 1 and 0 transitions to v respectively. Then

$$\begin{aligned}
&\Pr_{x \leftarrow U_{i+1}}\left[(B[v_0, x] = v) \bigwedge \bigvee_{j=1}^{i+1} (B[v_0, x_{1..j}] \in V^\varepsilon)\right] \\
&\quad = \Pr_{x \leftarrow U_{i+1}}\left[[(B[v_0, x_{1..i}] = u_1 \wedge x_{i+1} = 1) \vee (B[v_0, x_{1..i}] = u_0 \wedge x_{i+1} = 0)] \bigwedge \bigvee_{j=1}^{i} (B[v_0, x_{1..j}] \in V^\varepsilon)\right] \\
&\quad = \frac{1}{2} \cdot \Pr_{x \leftarrow U_i}\left[(B[v_0, x] = u_1) \bigwedge \bigvee_{j=1}^{i} (B[v_0, x_{1..j}] \in V^\varepsilon)\right] \\
&\qquad\qquad + \frac{1}{2} \cdot \Pr_{x \leftarrow U_i}\left[(B[v_0, x] = u_0) \bigwedge \bigvee_{j=1}^{i} (B[v_0, x_{1..j}] \in V^\varepsilon)\right] \\
&\quad \leq \frac{\varepsilon}{2} + \frac{\varepsilon}{2}
\end{aligned}$$

so the claim follows. □

We conclude by creating B_ε from B by wiring all transitions that would reach states in V_ε to the sudden reject state $\perp$ in the relevant layer, and removing all states in V_ε. Since there are at most $\lceil 1/\varepsilon \rceil$ states in each layer not in V_ε (and there can be made to be exactly $\lceil 1/\varepsilon \rceil$ such states by adding dummy states with no in transitions and both out transitions wired to $\perp$), and the non-removed states have at most 2 in transitions, this produces a regular plus sudden reject branching program of the desired width. Furthermore, the program is a lower approximator since the only added state is $\perp$. To see that $A_\Delta(B_\varepsilon, B) \leq \varepsilon$, note that for x where $B(x) = 1$ and $B_\varepsilon(x) = 0$ it must be the case that $B[v_0, x] = v_{\text{acc}}$ and B hits some element of V_ε on x, so applying Claim 2.1 we obtain that there are at most $\varepsilon \cdot 2^n$ such strings x. □

As a corollary, we obtain that ε-hitting sets for branching programs of width $O(1/\varepsilon)$ also constitute $O(\varepsilon)$-hitting sets for regular programs of unbounded width.

Corollary 2.2. *Let H be an $\varepsilon/2$-hitting set generator for ordered branching programs of length n and width $1 + \lceil 2/\varepsilon \rceil$. Then H is an ε-hitting set generator for regular branching programs of length n and unbounded width with a single accept state.*

Proof. Given an arbitrary regular branching program B such that $\Pr[B(U_n) = 1] > \varepsilon$, let B_L be the length n, width $\lceil 2/\varepsilon \rceil + 1$, $\varepsilon/2$-lower approximator ordered branching program that is guaranteed to exist by Proposition 1.17. We have

$$\Pr[B_L(U_n) = 1] = \Pr[B(U_n) = 1] + (\Pr[B_L(U_n) = 1] - \Pr[B(U_n) = 1]) > \varepsilon - A_\Delta(B, B_L) \geq \varepsilon/2$$

where the first inequality follows from Proposition 1.25. By assumption H hits B_L and therefore B, and since B was arbitrary we are done. □

This establishes that better HSGs for ordered branching programs in the constant width, constant error case imply improved constant-error hitting sets for regular branching programs of unbounded width with a single accept state. Unfortunately, even for general ordered branching programs of width $w = 4$ the best known explicit HSG remains Nisan's PRG, with seed length $\Theta(\log^2 n)$. To obtain seed length $o(\log^2 n)$, we must exploit the specific structure of regular plus sudden reject branching programs.

3 Fooling Regular Plus Sudden Reject Branching Programs

We now show that the lower-approximators produced by Proposition 1.17 can be fooled by the Impagliazzo–Nisan–Wigderson [INW94] generator. We recall the precise statement:

Theorem 1.18. *Given $n, w \in \mathbb{N}$ and $\varepsilon \in (0, 1/2)$, there is an explicit ε-PRG for regular plus sudden reject branching programs of length n and width w that has seed length*

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

Our approach is similar to that of Baverman, Rao, Raz and Yehudayoff [BRRY10], in that we show the error incurred by the generator does not accumulate linearly over the length of the program. In fact, we show this by modifying the "pebble game" of [BRRY10], and appealing to their results that the INW generator fools all programs with low "weight". The weight of a branching program can be thought of as the sum of the importance of the transitions, and we formally define it now. Recall that $p_{v\to}$ is the probability of accepting from state v over truly random input. In the notation of Braverman, Rao, Raz and Yehudayoff, this is denoted $p(v)$.

Definition 3.1. Given a (not necessarily regular) branching program B, let

$$\text{weight}(B) = \sum_{(u,v,b):B[u,b]=v} |p_{u\to} - p_{v\to}|.$$

Furthermore, the weight of a specific transition from state u to state v is $|p_{u\to} - p_{v\to}|$.

Braverman et al. prove that for B an arbitrary regular branching program of width w, $\text{weight}(B) \leq 2(w+1)$, which crucially has no dependence on n. The set of branching programs we work with are not regular, but are almost regular, in that the only state with in-degree greater than 2 is always wired to itself and always rejects in the final layer. We show that the weight of regular plus sudden reject branching programs still does not depend on n. This is despite the fact that a general ordered branching program of width 3 can have weight $\Omega(n)$.

Lemma 3.2. *Let B be a regular plus sudden reject branching program of width w. Then* $\text{weight}(B) \leq 6w + 2$.

To prove this, we introduce a modification of the pebble game of Braverman et al. [BRRY10]:

Definition 3.3. For every $w \in \mathbb{N}$, the (modified) **pebble game of size** w features an initial configuration of $2w$ pebbles $0 = q_0 = q_1 = \ldots = q_{2w-3}, 1 = q_{2w-2} = q_{2w-1}$, where q_i denotes the initial position of pebble i. At each step, a valid move consists of either moving a pair of pebbles at positions $a < b$ to position $(a+b)/2$, which scores $b - a$, or moving a pebble to 0, which scores nothing. The **value** of the game is the supremum over every valid sequence of moves of the score obtained by those moves.

Our modification consists of adding the ability to move pebbles to the origin, which scores nothing. We first bound the value of the game, then show that the weight of a regular plus sudden reject branching program of width w is bounded by the value of the game of size w.

Lemma 3.4. *The value of the pebble game of size w is at most $6w + 2$.*

Proof. Let $\{q_i\}_{i\in[2w]}$ be the positions of the pebbles at some point in the game. Given these positions, define $P = \sum_{i,j\in[2w]} |q_i - q_j|$ as the *potential* of the configuration. Note that $P \geq 0$ always, and $P = P_0 \leq 2w$ at the start of the game.

We claim that a move that scores δ decreases the potential of the game by at least δ. To see this, note that for pebbles r_1, r_2 at positions $a < b$ moved to $(a+b)/2$, for a pebble at position $c \leq a$ or $b \leq c$ the sum of distances to r_1 and r_2 is unchanged. For a pebble at position $a < c < b$, the sum of distances to r_1 and r_2 decreases. To see this, let $x = (c-a)$ and $y = (b-a)/2$, and WLOG $x < y$. Then the sum of distances from r_1, r_2 to the pebble at c starts at $x + (2y - x) = 2y$ and ends at $2(y-x) < 2y$. Finally, the distance between r_1 and r_2 decreases by exactly $\delta = |b-a|$.

Next, note that moves that are not moving a pebble to zero do not change the sum $D = \sum_{i\in[2w]} q_i$ of distances from zero. Note that $D = D_0 \leq 2$ at the start of the game and $D \geq 0$ always. Furthermore, if a move to 0 increases P by γ, it decreases D by at least $\gamma/2w$. Thus moves to zero can increase the potential P by at most $4w$ over the entire game.

Thus every move can either score δ and decrease P by at least δ (and leave D fixed), or increase P by γ and decrease D by at least $\gamma/4w$ (and leave the score fixed). Thus, we bound the maximum possible score of every sequence of moves by $4w \cdot D_0 + P_0 \leq 4w + 2(w+1)$ as desired. □

With this, we are prepared to prove Lemma 3.2, by giving a sequence of moves in the pebble game whose score is exactly equal to the weight of the program.

Proof of Lemma 3.2. We model the weight of the program using the pebble game. For all $j \in [w]$, let $q_{2j} = q_{2j-1} = p_{v_j\to} \in [0,1]$ where v_j is the jth state in $V_n \setminus \perp$, which corresponds to the initial configuration of the pebble game of size w (and without loss of generality assume $v_w = v_{\text{acc}}$). We do not place pebbles corresponding to $\perp$.

Assume that there are 2 pebbles at $p_{v_j\to} \in [0,1]$ for all $v_j \in V_i \setminus \perp$, which is initially satisfied with $i = n$. If there are t total transitions from states in V_{i-1} to $\perp$, there are t states (with multiplicity) in V_i with fewer than two in-transitions. For every $v \in V_i$ missing at least one in-transition, move the corresponding number of pebbles from $p_{v\to}$ to $p_{\perp\to} = 0$ (which scores nothing). Now consider arbitrary $u \in V_{i-1}$. If u has transitions to $v_1 \neq \perp$ and $v_2 \neq \perp$, move pebbles from $p_{v_1\to}, p_{v_2\to}$ to $(p_{v_1\to} + p_{v_2\to})/2 = p_{u\to}$. Otherwise, for the one or two transitions to $\perp$, use one or two of the pebbles at $p_{\to\perp} = 0$ to make the corresponding move (handling the other transition normally if relevant) to $p_{u\to}$. As $p_{\perp\to} = 0$ in every layer, transitions from $\perp$ to itself have weight zero and can be ignored. Performing these moves for all $u \in V_{i-1} \setminus \perp$, we obtain two pebbles on $p_{u\to}$ for all $u \in V_{i-1} \setminus \perp$, and the score obtained via this sequence of moves is precisely the weight of all transitions from layer $i-1$ to layer i. By iterating this procedure from layer n to layer 0, we obtain a sequence of moves in the pebble game that scores exactly weight(B). But then weight(B) is upper bounded by the value of the pebble game of size w, and we conclude by Lemma 3.4. □

We now recall the main theorem of Braverman et al. [BRRY10], slightly restating it for our application:

Theorem 3.5 (Theorem 5 [BRRY10])**.** *Given $i \in \mathbb{N}$ and $\beta \in (0, 1/2)$, there exists an explicit (INW) PRG $G : \{0,1\}^s \to \{0,1\}^{2^i}$ with error parameter β and seed length $s = O(i \cdot \log(1/\beta))$. Then for every (not necessarily regular) branching program B of width w and length 2^i we have*

$$|\Pr[B(U_{2^i}) = 1] - \Pr[B(G(U_s)) = 1]| \leq i \cdot (w+1) \cdot \beta \cdot \text{weight}(B).$$

Then the proof of Theorem 1.18 is direct:

Theorem 1.18. *Given $n, w \in \mathbb{N}$ and $\varepsilon \in (0, 1/2)$, there is an explicit ε-PRG for regular plus sudden reject branching programs of length n and width w that has seed length*

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

Proof. Since branching programs can ignore bits, without loss of generality assume n is a power of 2. Let $G : \{0,1\}^s \to \{0,1\}^n$ be the explicit PRG of Theorem 3.5 with $\beta = \varepsilon/14w^2\log(n)$ and $i = \log(n)$ and seed length $s = O(\log(n)(\log\log(n) + \log(w) + \log(1/\varepsilon)))$.

Now fix an arbitrary regular plus sudden reject branching program B of length n and width w. We have that

$$\begin{aligned} \Pr[B(U_n) = 1] - \Pr[B(G(U_s)) = 1]| &\leq \log(n) \cdot (w+2) \cdot \text{weight}(B) \cdot \beta && \text{(Theorem 3.5)} \\ &\leq \log(n) \cdot (w+2) \cdot 7w \cdot \beta && \text{(Lemma 3.4)} \\ &\leq \varepsilon \end{aligned}$$

and since B was arbitrary we otain the desired result. □

4 Putting It All Together

We have that every regular branching program with a single accept state can be ε-lower approximated by a regular plus sudden reject branching program of width $w = O(1/\varepsilon)$, and the INW generator fools regular plus sudden reject branching programs with seed length $\tilde{O}(\log(n)\log(w/\varepsilon))$, so the proof of Theorem 1.9 is direct.

Theorem 1.9. *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*

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

Proof. Let $G : \{0,1\}^s \to \{0,1\}^n$ be the explicit generator obtained from Theorem 1.18 with $n = n$ and $w = \lceil 2/\varepsilon \rceil$ and $\varepsilon = \varepsilon/4$ with seed length $s = O(\log(n)(\log\log(n) + \log(1/\varepsilon)))$.

Now fix an arbitrary regular branching program B of length n with a single accept state where $\Pr[B(U_n) = 1] > \varepsilon$. Let B_L be the $\varepsilon/2$-lower approximator of B that is guaranteed to exist by Proposition 1.17. We have that B_L is a regular plus sudden reject branching program of length n and width $\lceil 2/\varepsilon \rceil$. Furthermore $\Pr[B_L(U_n) = 1] > \varepsilon - \varepsilon/2$ by Proposition 1.25 and the reverse triangle inequality. We have that G $\varepsilon/4$-fools B_L, i.e.

$$|\Pr[B_L(U_n) = 1] - \Pr[B_L(G(U_s)) = 1]| \leq \varepsilon/4,$$

therefore

$$\Pr[B_L(G(U_s)) = 1] > \varepsilon/2 - \varepsilon/4 \geq 0.$$

So there is some string σ such that $B_L(G(\sigma)) = 1$ and thus G hits B_L and thus B. As B was arbitrary, we conclude. □

We now prove Corollary 1.10. The proof consists of noticing that a program with a accept states and probability of accepting at least ε must have some accept state with probability of being reached at least ε/a, and it suffices to hit this state.

Corollary 1.10. *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*

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

Proof. Given a regular branching program B of unbounded width and a accept states $V_{\text{acc}} = \{v_1, \ldots, v_a\} \subseteq V_n$ such that $\Pr[B(U_n) = 1] > \varepsilon$, we have that B can be written as $B(x) = \sum_{i=1}^{a} B_i(x)$, where B_i is the program with identical structure to B with single accept state $v_{\text{acc}} = v_i$. Then by linearity of expectation

$$\sum_{i=1}^{a} \Pr[B_i(U_n) = 1] = \Pr[B(U_n) = 1] > \varepsilon$$

and since $\Pr[B_i(U_n) = 1]$ is non-negative for all i there is some i_0 such that $\Pr[B_{i_0}(U_n) = 1] > \varepsilon/a$. Then applying Theorem 1.9 with error ε/a, we obtain that G hits B_{i_0} and thus B, and seed length and explicitness are as claimed. □

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 et al [BRRY10].

Proposition 1.15. *Given $n, a \in \mathbb{N}$, the set $H = \{\sigma \in \{0,1\}^n : \text{wt}(\sigma) \leq a\}$ where $\text{wt}(\sigma)$ denotes the Hamming weight of σ 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 $\sigma \in H$ such that $B(\sigma) = 0$.*

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 $v \in V_i$ is *doomed* if for every $x \in \{0,1\}^{n-i}$ we have that $B[v, x] \in V_{\text{acc}}$. We say a state $v \in V_i$ 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 $\sigma \in H$ such that $B(\sigma) = 0$. Since B was arbitrary, we conclude. □

5 Acknowledgements

I 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.

[Arm98] Roy Armoni. On the derandomization of space-bounded computations. In *Randomization and approximation techniques in computer science (Barcelona, 1998)*, volume 1518 of *Lecture Notes in Comput. Sci.*, pages 47–59. Springer, Berlin, 1998.

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

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

[BRRY10] Mark Braverman, Anup Rao, Ran Raz, and Amir Yehudayoff. Pseudorandom generators for regular branching programs. In *FOCS*, pages 40–47. IEEE Computer Society, 2010.

[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.

[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.

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

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

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

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

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

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

[PV21a] Edward Pyne and Salil Vadhan. Limitations of Spectral Analysis of the INW Generator. In *To Appear, Computing and Combinatorics - 27th International Conference, COCOON*, Lecture Notes in Computer Science. Springer, 2021.

[PV21b] 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.

[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.

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

[SZ99] Michael Saks and Shiyu Zhou. $\mathrm{BP_HSPACE}(S) \subseteq \mathrm{DSPACE}(S^{3/2})$. *Journal of Computer and System Sciences*, 58(2):376–403, 1999.

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. We remark that our reduction holds with identical parameters given a *weighted* PRG as the initial pseudorandom object.

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

Definition A.1. An **ordered branching program (OBP)** 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 $v_n \in V_{\mathrm{acc}}$ 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 regular branching programs over larger alphabets.

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

Finally, we formally define HSGs and PRGs over larger alphabets.

Definition A.3. 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.4. 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 $f \in \mathcal{F}$,

$$\left| \Pr_{x \leftarrow U_{[d]^n}}[f(x) = 1] - \Pr_{x \leftarrow U_{\{0,1\}^s}}[f(G(x)) = 1] \right| \leq \varepsilon.$$

The value s is the **seed length** of the PRG (resp. HSG). We say a generator G is **explicit** if the ith symbol of output is computable in space $O(s)$.

We define notation for states and transitions in branching programs analogously for 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 $x \in [d]^{j-i}$. We can then state our main theorem for transferring pseudorandom objects over a binary alphabet into pseudorandom objects over larger alphabets. We state it in terms of bounded-width branching programs, but an equivalent result holds when we restrict the number of accept states rather than the width.

Theorem 1.20. *Given $n, w, d \in \mathbb{N}$ and $\varepsilon > 0$, there is $t = O(n \log(nd/\varepsilon))$ such that if there is an explicit ε-PRG (resp. HSG) $G : \{0,1\}^s \to \{0,1\}^t$ for regular branching programs of length t, width $10wn^2d/\varepsilon$ and degree 2, there is an explicit 4ε-PRG (resp. HSG) $G'' : \{0,1\}^s \to [d]^n$ for regular branching programs of length n, width w and degree d.*

We now state and prove the pair of reductions which together imply the theorem. We first transform a binary PRG into a PRG for alphabets of size arbitrary powers of two, and then transform a PRG for a sufficiently large power of two into a PRG for the desired lower degree.

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

Proof. Let $q_e : \{0,1\}^r \to [R]$ be defined as $q_e(x_1, \ldots x_r) = \sum_{i=1}^{r} x_i 2^{i-1}$ and define the function $q(x) = (q_e(x_1, \ldots, x_r), \ldots, q_e(x_{(t-1)r}, \ldots, x_t))$. Now fix an arbitrary branching program B of length $t = \lfloor n/d \rfloor$, width w/R and degree R with a accept states. For each state $v \in V_i$ of B, we blow up v into a cloud $C(v)$ of $R/2$ states in layer ir of a new program $B' : \{0,1\}^{tr} \to \{0,1\}$ of width $(w/R)(R/2) \leq w$. To connect these clouds, we rely on the following gadget:

Claim A.6. *For every $K = 2^k$ for $k \in \mathbb{N}$, there is a regular branching program M of length k and degree 2 with $K/2$ states in layers $0, \ldots, k-1$ and K states $u_0, \ldots, u_{K-1}$ in layer k (each with a single in-transition) such that for every state v in layer 0 and $\sigma \in \{0,1\}^k$, we have $M[v, \sigma] = u_\sigma \in V_k$ where we view σ as a number in $[K]$.*

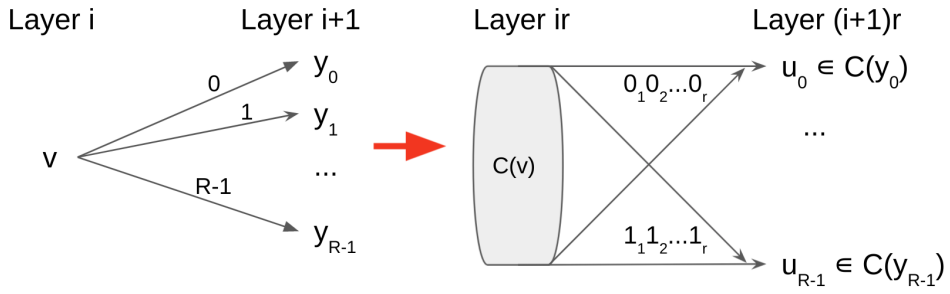

Figure 1: The blowup of state v in B into a cloud of states $C(v)$ in B' via the gadget of Claim A.6.

Proof. We proceed by induction. For $k = 1$ the statement is simply a program with one state v in layer 0 and states u_1, u_0 in layer 1 where $B[v, b] = u_b$ for $b \in \{0, 1\}$.

Now assume we have the gadget of size k and consider $k + 1$. Let the states in layer 0 be $v_0, \ldots, v_{K-1}$ and in layer 1 be $y_0, \ldots, y_{K/2-1}, z_0, \ldots, z_{K/2-1}$. Then for $b \in \{0, 1\}$ define the transition function from layer 0 to layer 1 as:

$$M_1(v_i, b) = \begin{cases} y_{i \mod 2^{k-2}} & b = 0 \\ z_{i \mod 2^{k-2}} & b = 1 \end{cases}$$

Then on $y_0, \ldots, y_{K/2-1}$ we place the gadget of size K on the remaining layers and likewise for $z_0, \ldots, z_{K/2-1}$, so we obtain the desired construction. □

In effect, M reaches a state in the final layer determined by the $\{0, 1\}^k$ bits of input, with no dependence on the initial state.

Now fixing an arbitrary cloud $C(v)$ in layer ir of B', let $y_j = B[v, j]$ for all $j \in [R]$. In B', we place the program M of size $k = r$ on layers $ir, \ldots, (i + 1)r$ where the initial states are $C(v)$ and the final states are arbitrary (not necessarily distinct) states in $C(y_0), \ldots, C(y_{R-1})$. Since for every w, elements of $C(w)$ participate in the final layer of gadgets R times and thus have R total in-transitions, we can apply this construction to all states in B so the blowup B' is regular. Therefore, if $B[v, j] = y_j$, for every state u in the cloud $C(v)$ we have $B'[u, j] \in C(y_j)$, viewing j as an r-bit binary string. We illustrate this process in Figure 1.

We then mark as accept states all states in B' corresponding to clouds of accept states of B (and WLOG pad B' to length n with identity layers), and so by construction $B \circ q = B'$ and q maps uniform input to uniform output. To conclude, we break into cases depending on the base pseudorandom object:

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

$$\Pr[B'(U_{\{0,1\}^n}) = 1] = \Pr[B(U_{[R]^t}) = 1] > \varepsilon$$

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

(G is an ε-PRG): We have

$$\begin{aligned}
&\left|\Pr_{x\leftarrow U_s}[B(q\circ G(x))=1]-\Pr_{x\leftarrow U_{[R]^t}}[B(x)=1]\right| \\
&\quad=\left|\Pr_{x\leftarrow U_s}[(B\circ q)(G(x))=1]-\Pr_{x\leftarrow U_{\{0,1\}^n}}[B'(x)=1]\right| \\
&\quad=\left|\Pr_{x\leftarrow U_s}[B'(G(x))=1]-\Pr_{x\leftarrow U_{\{0,1\}^n}}[B'(x)=1]\right| \\
&\quad\leq\varepsilon.
\end{aligned}$$

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

We remark that this component of the reduction does not preserve the property of B being a *permutation* branching program, since the gadgets are not permutation programs. We then show that PRGs and HSGs for alphabets of size powers of two imply PRGs and HSGs for other alphabet sizes.

Lemma A.7. *Given $n, w, d \in \mathbb{N}$ and $\varepsilon > 0$, there is $R_0 = O(nd/\varepsilon)$ such that for every $2^r = R \geq R_0$, 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, then $p \circ G$ is an explicit 4ε-PRG (resp. HSG) for regular branching programs of length n, degree d and width $w/(n+1)$.*

Proof. Let $R = O(nd/\varepsilon)$ be an arbitrary power of two such 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 let $p(x) = (p_e(x_1), \ldots, p_e(x_n))$.

Let $m \leq R$ be the largest multiple of d not greater than R. For $x \in [R]^n$, we say x is BAD if there exists i where $x_i > m$. Let $\rho = \Pr_{x\leftarrow U_{[R]^n}}[x \text{ is BAD}]$, and observe $\rho \leq n(d/R) < \varepsilon$, and furthermore there exists a length n, width $n \leq w$, degree R regular branching program Q where $Q(x) = \mathbb{I}[x \text{ is BAD}]$.

Now fix an arbitrary regular branching program $B : [d]^n \to \{0,1\}$ of width w. Let $B' : [R]^n \to \{0,1\}$ be the regular branching program of length n, degree R and width $w + nw$ where, for every state $v \in V_i$ of B, we add an auxiliary state $v^\perp$ that is always wired to itself except in layer i, and marked as reject in the final layer. Furthermore, we have

$$B'[v,\sigma] = \begin{cases} B[v, \sigma \mod d] & \sigma \leq m \\ v^\perp & \text{otherwise.} \end{cases}$$

Note that $B[v, \sigma \mod d] = B[v, p_e(\sigma)]$, so for every $x \in [R]^n$ that is not BAD we have $B'(x) = B(p(x))$ and p maps uniform non-BAD inputs to uniform outputs. Therefore we have

$$\begin{aligned}
(*) &= \left|\Pr_{x\leftarrow U_{[R]^n}}[B'(x)=1]-\Pr_{x\leftarrow U_{[d]^n}}[B(x)=1]\right| \\
&= \left|\Pr_{x\leftarrow U_{[R]^n}}[B'(x)=1|x \text{ is BAD}]\cdot\rho+\Pr_{x\leftarrow U_{[R]^n}}[B'(x)=1|x \text{ is not BAD}]\cdot(1-\rho)-\Pr_{x\leftarrow U_{[d]^n}}[B(x)=1]\right| \\
&\leq \rho+\left|\Pr_{x\leftarrow U_{[R]^n}}[B'(x)=1|x \text{ is not BAD}]-\Pr_{x\leftarrow U_{[d]^n}}[B(x)=1]\right| \\
&\leq \varepsilon + 0.
\end{aligned}$$

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

(G is an ε-HSG): We have that if $\Pr[B(U_{[d]^n}) = 1] > 2\varepsilon$ then $\Pr[B'(U_{[R]^n}) = 1] > 2\varepsilon - \varepsilon$ by $(*)$. Thus by assumption on G there is some x where $B'(G(x)) = 1$. Since B' always rejects on inputs that are BAD, we have $1 = B'(G(x)) = B(p \circ G(x))$.

(G is an ε-PRG): We have that G ε-fools Q by assumption, so $\Pr_{x \leftarrow U_s}[G(x) \text{ is BAD}] \leq \varepsilon + \varepsilon$.[5] Then using the structure of B' and an equivalent argument to (*) we obtain:

$$(**) = \left| \Pr_{x \leftarrow U_s}[B'(G(x)) = 1] - \Pr_{x \leftarrow U_s}[B(p \circ G(x)) = 1] \right| \leq \Pr_{x \leftarrow U_s}[G(x) \text{ is BAD}] \leq 2\varepsilon.$$

We finish by repeated application of the triangle inequality:

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

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

We can then prove the main transfer result:

Theorem 1.20. *Given $n, w, d \in \mathbb{N}$ and $\varepsilon > 0$, there is $t = O(n \log(nd/\varepsilon))$ such that if there is an explicit ε-PRG (resp. HSG) $G : \{0,1\}^s \to \{0,1\}^t$ for regular branching programs of length t, width $10wn^2d/\varepsilon$ and degree 2, there is an explicit 4ε-PRG (resp. HSG) $G'' : \{0,1\}^s \to [d]^n$ for regular branching programs of length n, width w and degree d.*

Proof. Let $R \leq 2nd/\varepsilon$ be the constant in Lemma A.7 with $n = n, w = w$ and $\varepsilon = \varepsilon$ and let $t = \lceil n \log R \rceil$. We apply Lemma A.5 to G with $n = t, w = 10wn^2d/\varepsilon$ and $R = R$ and obtain an explicit ε-PRG (resp. HSG) $G' : \{0,1\}^s \to [R]^n$ for regular branching programs of length n, width $10wn$ and degree R. Then applying Lemma A.7 to G' with $n = t, w = 10wn$ and $\varepsilon = \varepsilon$, we obtain an explicit 4ε-PRG (resp. HSG) $G'' : \{0,1\}^s \to [d]^n$ for regular branching programs of length n, width $10wn/(n+1) \geq w$ and degree d. □

[5] A nearly identical bound establishes a matching transfer result for weighted PRGs.

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 weighted PRGs, and derive the degradation in parameters in the notation of branching programs.

Theorem 1.19 (Variant of [RTV06])**.** *Given $n, w \in \mathbb{N}$ and $\varepsilon > 0$, there is $D' = O((\varepsilon/nw)^4)$ such that if $G : \{0,1\}^s \to [D']^n$ is an explicit ε-PRG (resp. HSG) for regular branching programs of length n, width $(nw/\varepsilon)^8$ and degree D', there is an explicit 10ε-PRG (resp. HSG) $G' : \{0,1\}^s \to \{0,1\}^n$ for ordered branching programs of length n and width w.*

Proof. Let B be an ordered branching program of length n and width w. Let $T = (nw/\varepsilon)^2$ and $D = (nw/\varepsilon)^4$ (and WLOG assume D is a power of 2). For every state $v \in V_k$ of B, let $S = \lfloor T \cdot p_{\to v} \rfloor$. If $S \leq nw/\varepsilon$, do not insert states, and otherwise blow up v into a cloud $C(v)$ of S states in a new branching program $B' : [D]^n \to \{0,1\}$. This results in (at present) at most $T \cdot w$ states in each layer of B'. If v is an accept state, mark all states in the blowup $C(v)$ as accept states, and mark an arbitrary state in $C(v_{\text{acc}})$ as the start state.

We now define transitions. Fixing state $u \in V_k$ where $B[u,1] = v_1$ and $B[u,0] = v_0$, let $t_1 = |C(v_1)|$ and $t_0 = |C(v_0)|$, for $u' \in C(u)$ define the transition function:

$$B'_k(u', \sigma) = \begin{cases} C(v_1)_i & \sigma_1 = 1 \wedge ((\sigma - D/2) \mod t_1) = i \\ C(v_0)_i & \sigma_1 = 0 \wedge (\sigma \mod t_0) = i \end{cases}$$

Where σ_1 is the first bit of σ viewed as a string in $\{0,1\}^{\log D}$. Furthermore, if $t_1 = 0$ or $t_0 = 0$ (i.e. v_1 or v_0 was not blown up into a cloud) instead send all relevant transitions to a dummy reject state $u'^{\perp}$. This results in a branching program B' of degree D and width at most $4Tn^2w$. Unfortunately, as B' has roundoff errors, it is not yet a regular branching program, which we fix by adding symbols to the alphabet.

Claim B.1. *For every state $v' \in C(v)$ in B', the number of in-transitions entering this state is at most $D(1+\rho)$ where $\rho = 3\varepsilon/n^2w$.*

Proof. We can bound the number of in-transitions using the definition of the blowup. Let the states in B with out-transitions to b be denoted $u_1, \ldots, u_r$ (where we double-count states with two such transitions). Then:

$$\begin{aligned}
\sum_{i=1}^{r} \sum_{u'_i \in C(u_i)} |\{\sigma : B'[u'_i, \sigma] = v'\}| &\leq \sum_{i=1}^{r} |C(u_i)| \cdot \left(\frac{D/2}{|C(v)|} + 1 \right) \\
&\leq 2T + \sum_{i=1}^{r} \lfloor T \cdot p_{\to u_i} \rfloor \cdot \left(\frac{D/2}{\lfloor T \cdot (\sum_{i=1}^{r} p_{\to u_i}/2) \rfloor} \right) \\
&\leq 2T + \left(\sum_{i=1}^{r} T \cdot p_{\to u_i} \right) \cdot \frac{D}{T \cdot (\sum_{i=1}^{r} p_{\to u_i}) - 1} \\
&\leq 2T + D(1 + 2\varepsilon/nw) \\
&\leq D(1+\rho).
\end{aligned}$$

Where we use that if $C(v)$ is non-empty it has at least nw/ε states. □

Thus every state in the blowup has in-degree at most $D' = D(1+\rho)$. We then blow up B' into a new *regular* branching program $B'' : [D']^n \to \{0,1\}$, where the transitions corresponding to the first D symbols are unchanged, and the symbols in $[D'] \setminus [D]$ are wired to $n(4Tn^2w)$ new dummy states that always reject. Note that B'' is a regular branching program of length n, width $8Tn^3w \leq (nw/\varepsilon)^8$, and degree $D' = O((nw/\varepsilon)^4)$ where D' does not depend on B, only the input parameters. Thus, G is required to be ε pseudorandom against B''.

We next show that this blowup does not change the probability of transitions between clusters too much. For the remainder of the proof, we work with clouds of states in B''.

Claim B.2. *For every v where $C(v) \neq \emptyset$, let $u' \in C(u)$ be such that $B[u,b] = v$. Then*

$$\left| \sum_{v' \in C(v)} \Pr_{\sigma \leftarrow U_{[D']}} [B''[u',\sigma] = v' | \sigma_1 = b] - \frac{1}{2} \right| \leq \rho.$$

Proof. In B' exactly $1/2$ the transitions from u' with first bit b reach states in $C(v)$ by construction. Then in the blowup from B' to B'' we add at most a ρ fraction of extra symbols, so we change the probability of transitioning to $v' \in C(v)$ by at most an equal amount. □

It remains to show that G fooling B'' implies a modification of the generator fools B. We first show the blowup process does not modify the accept probability too much.

Claim B.3. *We have*

$$\left| \Pr_{x \leftarrow U_n} [B(x) = 1] - \Pr_{x \leftarrow U_{[D']^n}} [B''(x) = 1] \right| \leq 3\varepsilon.$$

Proof. For every state $v \in V_k$ in the original branching program B, let

$$\mathrm{err}_v = \left| p_{\to v} - \sum_{v' \in C(v)} p_{\to v'} \right|.$$

Recall $p_{\to v}$ is the probability of transitioning from v_0 to state v in B over uniformly random input, and likewise for $p_{\to v'}$ for state v' in B''. We prove that for $v \in V_k$ with in transitions from $u_1, \ldots, u_r$ (where we double count states with two such transitions) we have

$$\mathrm{err}_v \leq \frac{1}{2} \sum_{i=1}^{r} \mathrm{err}_{u_i} + r\rho.$$

Note that since err_u appears in exactly two summations in layer k for every state u in layer $k-1$, the total error increases by an *additive* $w\rho$ at each layer. Then $\sum_{v \in V_a} \mathrm{err}_v \leq nw\rho$ and our choice of ρ completes the claim.

First, suppose $C(v) = \emptyset$. Then we have $Tp_{\to v} \leq 2nw/\varepsilon$, and by our choice of T we obtain

$\mathrm{err}_v = |p_{\to v} - 0| \le \rho$ so the claim is proved. Otherwise if $C(v)$ is nonempty we have:

$$\begin{aligned}
\mathrm{err}_v &= \left| p_{\to v} - \sum_{v' \in C(v)} p_{\to v'} \right| \\
&= \left| p_{\to v} - \sum_{v' \in C(v)} \sum_{i=1}^{r} \sum_{u'_i \in C(u_i)} p_{\to u'_i} \Pr_{\sigma \leftarrow U_{[D']}} [B'[u'_i, \sigma] = v'] \right| \\
&\le \left| \sum_{i=1}^{r} p_{\to u_i}/2 - \sum_{i=1}^{r} \sum_{u'_i \in C(u_i)} p_{\to u'_i} \sum_{v' \in C(v)} \Pr_{\sigma \leftarrow U_{[D']}} [B'[u'_i, \sigma] = v'] \right| \\
&\le \sum_{i=1}^{r} \left| p_{\to u_i}/2 - \sum_{u'_i \in C(u_i)} p_{\to u'_i}/2 \right| + r\rho \qquad \text{(Claim B.2)} \\
&\le \frac{1}{2} \sum_{i=1}^{r} \mathrm{err}_{u_i} + r\rho. \qquad \square
\end{aligned}$$

Next, we show that the blowup process does not alter the performance of G too much. Let $P : [D']^n \to \{0,1\}^n$ be the explicit function that projects each symbol in $[D]$ onto its first bit (and maps other symbols arbitrarily).

Claim B.4. *We have that $B'' \le B \circ P$, and if G is a PRG then*[6]

$$\left| \Pr_{x \leftarrow U_s} [B(P \circ G(x)) = 1] - \Pr_{x \leftarrow U_s} [B''(G(x)) = 1] \right| \le 6\varepsilon.$$

Proof. For $x \in [D']^n$, we call x BAD if it contains a symbol in $[D'] \setminus [D]$. To prove the first component of the claim, suppose $B''(x) = 1$. Then x is not BAD, so the sequence of states obtained in evaluating B'' on x, denoted $v'_0, v'_1, \ldots, v'_n$, is contained in a sequence of clouds $C(v_0), \ldots, C(v_n)$ where v_n is marked as accept in B. Thus evaluating B on $P(x)$ results in the sequence of states $v_0, \ldots, v_n$ by construction of P and transitions between clouds in B'', so $B(P(x)) = 1$.

To prove the second component of the claim, we call $x \in [D']^n$ NEG if $B''(x) = 0$ whereas $B(P(x)) = 1$. For x that is NEG, it must be the case that $B''(x)$ attempts to transition to a cloud that does not exist and instead goes to a dummy state. From Claim B.3 we have that this occurs with probability at most 6ε over uniformly random input (since the corresponding states are hit with probability at most $(\varepsilon/nw)nw$ in B). Furthermore, there are regular branching programs Q_1, Q_2 of length n, width $(nw/\varepsilon)^8$, and degree D' that accept if x is BAD or NEG respectively, and G is required to fool both. Thus, we have

$$\begin{aligned}
\left| \Pr_{x \leftarrow U_s} [B(P \circ G(x)) = 1] - \Pr_{x \leftarrow U_{[D]^n}} [B''(G(x)) = 1] \right| &\le \Pr_{x \leftarrow U_s} [G(x) \text{ is BAD}] + \Pr_{x \leftarrow U_s} [G(x) \text{ is NEG}] \\
&\le n\rho + \varepsilon + 3\varepsilon + \varepsilon \\
&\le 6\varepsilon. \qquad \square
\end{aligned}$$

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

(G is an ε-HSG): We have that if $\Pr[B(U_n) = 1] > 10\varepsilon$ then $\Pr_{x \leftarrow U_{[D']^n}}[B''(x) = 1] > \varepsilon$ by Claim B.3. Then by assumption on G there is x such that $B''(G(x)) = 1$. Thus $1 = B(P \circ G(x)) = B''(G(x))$ by Claim B.4.

[6] Again, a nearly identical bound establishes an equivalent result for weighted PRGs.

(G is an ε-PRG):

$$
\begin{aligned}
&\left|\Pr_{x\leftarrow U_s}[B(P\circ G(x))=1]-\Pr_{x\leftarrow U_{\{0,1\}^n}}[B(x)=1]\right| \\
&\quad\le\left|\Pr_{x\leftarrow U_s}[B(P\circ G(x))=1]-\Pr_{x\leftarrow U_{[D']^n}}[B''(x)=1]\right| \\
&\qquad\qquad+\left|\Pr_{x\leftarrow U_{[D']^n}}[B''(x)=1]\Pr_{x\leftarrow U_n}[B(x)=1]\right| \\
&\quad\le\left|\Pr_{x\leftarrow U_s}[B(P\circ G(x))=1]-\Pr_{x\leftarrow U_{[D]^n}}[B''(x)=1]\right|+3\varepsilon && \text{(Claim B.3)} \\
&\quad\le\left|\Pr_{x\leftarrow U_s}[B(P\circ G(x))=1]-\Pr_{x\leftarrow U_s}[B''(G(x))=1]\right| \\
&\qquad\qquad+\left|\Pr_{x\leftarrow U_s}[B''(G(x))=1-\Pr_{x\leftarrow U_{[D']^n}}[B''(x)=1]\right|+3\varepsilon \\
&\quad\le\left|\Pr_{x\leftarrow U_s}[B(P\circ G(x))=1]-\Pr_{x\leftarrow U_{[D]^n}}[B''(G(x))=1]\right|+4\varepsilon && \text{(Assumption)} \\
&\quad\le 10\varepsilon && \text{(Claim B.4).}
\end{aligned}
$$

In both cases since B was arbitrary, we set $G'=P\circ G$ and obtain the desired result. □

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