Electronic Colloquium on Computational Complexity, Revision 3 of Report No. 64 (2026)

Improved Error Reduction for Weighted PRGs

Ben Chen* Gil Cohen† Dean Doron‡ Yuval Khaskelberg§ Amnon Ta-Shma¶

Abstract

We devise an error-reduction procedure that transforms a PRG for length-n, width-w read-once branching programs with error $1/\mathrm{poly}(n)$ and seed length s_0 into a weighted PRG with seed length

$$s_0 + O\left(\log\frac{1}{\varepsilon} + \log\log\left(\frac{\log w}{\log n}\right)\cdot\log w\right).$$

Using this reduction, we improve upon the state-of-the-art weighted PRG constructions of Hoza (RANDOM 2021) and Cheng and Wu (SODA 2026), achieving optimal dependence on the program's arity while matching the best known bounds in all other parameters.

Our motivation for obtaining optimal dependence on the arity stems from a result of Cheng and Hoza (CCC 2020, ToC 2022), who showed that a PRG with optimal arity and error dependence yields a PRG with seed length $O(\log^{3/2} n)$ (for, say, constant width), thereby breaking the long-standing log-squared barrier.

*Tel Aviv University. ben1chen@gmail.com. Funded by the Israel Science Foundation grant 443/22 and the Blavatnik fund.

†Tel Aviv University. gil@tauex.tau.ac.il. Funded by ERC starting grant 949499 and by the Israel Science Foundation grant 2989/24.

‡Ben Gurion University. deand@bgu.ac.il. Funded in part by the Israel Science Foundation grant 857/25.

§Tel Aviv University. khaskelberg@mail.tau.ac.il. Funded by ERC starting grant 949499.

¶Tel Aviv University. amnon@tauex.tau.ac.il. Funded by the Israel Science Foundation grant 443/22.

ISSN 1433-8092

Contents

1 Introduction

The question of whether $\mathbf{BPL} = \mathbf{L}$ is one of the central open problems in computational complexity theory [AKL+79, BCP83, Jun81]. It asks whether randomness provides additional computational power to space-bounded algorithms, or equivalently, whether every probabilistic logspace algorithm can be fully derandomized with only a constant-factor increase in space. Despite decades of intensive study, the problem remains open. Nonetheless, the prevailing belief is that $\mathbf{BPL} = \mathbf{L}$, a conjecture supported by plausible circuit lower bounds [KvM02, DT23, DPT24]. Importantly, unlike in the time-bounded setting, there are currently no known barriers that rule out unconditional derandomization of $\mathbf{BPL}$.

Progress toward this goal has been strikingly limited. Beyond a small number of landmark results, there has been essentially no improvement for several decades. Already in the work of Savitch [Sav70] and Borodin, Cook, and Pippenger [BCP83], it was shown that $\mathbf{BPL} \subseteq \mathbf{L}^2$. The only substantial strengthening of this bound was obtained by Saks and Zhou [SZ99], who proved that $\mathbf{BPL} \subseteq \mathbf{L}^{3/2}$. This result, established nearly 30 years ago, has remained essentially unimproved to this day, with only a quite recent sub-polynomial improvement due to Hoza [Hoz21].

The primary approach to resolving $\mathbf{BPL}$ vs. $\mathbf{L}$ is via the construction of pseudorandom generators (PRGs) for read-once branching programs (BPs), which constitute the standard nonuniform model associated with space-bounded Turing machines. We refer the reader to Definition 3.1 for a formal definition. For present purposes, it suffices to note that the length of a BP, denoted by n, corresponds to the number of random bits read by the algorithm (and is therefore bounded by its running time), while the width w captures the space usage and is, in fact, exponential in the space bound. For naive derandomization of $\mathbf{BPL}$ via PRGs, one should consider $w = \mathrm{poly}(n)$.

In a seminal paper, Nisan [Nis92] constructed an explicit ε-error PRG for BPs of length n and width w with seed length $O\big(\log n \cdot \log\big(\frac{nw}{\varepsilon}\big)\big)$. This construction—together with its expander-based variant due to Impagliazzo, Nisan, and Wigderson [INW94]—remains the state of the art in the general setting. An improvement in the regime $w \gg n$ was obtained by Armoni [Arm98] and was recently simplified in [CT25], shaving a factor of $\log\left(\frac{\log w}{\log(n/\varepsilon)}\right)$ from the seed length.

By contrast, an optimal PRG has seed length $O\big(\log\big(\frac{nw}{\varepsilon}\big)\big)$; if such a generator were also space-efficient, it would immediately imply $\mathbf{BPL} = \mathbf{L}$. Remarkably, despite more than three decades of subsequent work, Nisan's construction has not been improved. In fact, except for the case of $w = 3$ [MRT19], or when additional structural restrictions are imposed on the BPs (see, e.g., [KNP11, De11, BRRY14, DHH19, PV21, DMR+21, CHL+23]), no PRG is known to break the so-called log-squared barrier; that is, to achieve seed length $o(\log^2 n)$ for constant error ε, when $w = \mathrm{poly}(n)$, or even when the width is as small as $w = 4$.

Weighted pseudorandom generators

To bypass this long-standing stalemate, Braverman, Cohen, and Garg [BCG20] introduced a new type of pseudorandom object called a *weighted pseudorandom generator (WPRG)*. A WPRG relaxes

the classical definition of a PRG while still enabling the derandomization of two-sided error probabilistic algorithms. Specifically, in a WPRG each seed is assigned a weight—which may be positive, negative, and potentially unbounded—and the desired ε-approximation is achieved via a weighted sum rather than a simple average.

In [BCG20], the authors constructed a WPRG with an almost optimal dependence on the error parameter ε, achieving seed length $\widetilde{O}(\log n \cdot \log(nw) + \log(1/\varepsilon))$. While the error parameter is taken to be constant in the most naive PRG-based approach to derandomizing $\mathbf{BPL}$, it plays a crucial role in more refined derandomization procedures, most notably the Saks–Zhou algorithm. The latter involves multiple composition steps, necessitating small error and making the dependence on ε crucial. Indeed, Braverman et al. [BCG20] and Chattopadhyay and Liao [CL20] showed that one can prove $\mathbf{BPL} \subseteq \mathbf{L}^{4/3}$ even without breaking the log-squared barrier. Concretely, they showed that such a derandomization result would follow from the existence of a space-efficient WPRG with seed length $O(\log^2 n + \log(w/\varepsilon))$.

As observed in [BCG20], WPRGs readily yield hitting sets. Consequently, their result led to the first improved construction of hitting sets for BPs since the work of Nisan [Nis92], with a much simplified follow-up construction due to Hoza and Zuckerman [HZ20].

Since then, WPRGs have attracted significant attention. They have inspired simpler and improved constructions [Koz20, CL20, CDR+21, PV21, Hoz21], culminating in the WPRG of Hoza [Hoz21], which removes all double-logarithmic factors, obtaining seed length $O(\log n \cdot \log(nw) + \log(1/\varepsilon))$ and in the very recent state-of-the-art construction of Cheng and Wu [CW26], which achieves further improvements in the regime $w \gg n$.

These works incorporate ideas from fast Laplacian solvers and were also the basis for significantly improved generators for restricted models [HPV21, PV21, CHL+23]. The limitations of these techniques have been investigated in [CDM+25]. Developments in WPRGs have led to progress on several long-standing problems. Notably, they were leveraged by Hoza to improve the Saks–Zhou derandomization mentioned above [Hoz21], and have also enabled substantial advances on the closely related problem of iterated matrix multiplication [CDSTS23, PP23]. Moreover, the error reduction techniques have recently been used to show that $\mathbf{BPL}$ is contained in logspace-uniform $\mathbf{AC}^1$ [CW24].

Braverman, Cohen, and Garg [BCG20] emphasized the role of the error parameter ε, showing that simultaneously improving the seed-length dependence on both ε and w would yield a significant improvement to the Saks–Zhou algorithm, even without breaking the log-squared barrier. Moreover, such an improvement need only be achieved for WPRGs, rather than for standard PRGs. While [BCG20] and follow-up works significantly improved the dependence on ε, progress on the width parameter w has remained much more elusive. To date, the only work in this direction that we are aware of is due to Raz and Reingold [RR99], which presents an intriguing, albeit seemingly stalled, approach.

Cheng and Hoza [CH22] proposed an intriguing approach toward breaking the log-squared barrier, showing that seed length $O(\log^{3/2} n)$ can be obtained even without *directly* improving the

dependence on either n or w. Instead, such an improvement follows from *simultaneously* improving the dependence of PRGs on two secondary parameters: the error ε and the program's arity $|\Sigma|$. We discuss this result below. This underscores the somewhat surprising importance of arity dependence.

1.1 Our Result

The main result of this paper is a construction of WPRGs that achieves optimal dependence on both ε and Σ, while matching the state-of-the-art bounds in n and w. As in prior work, our WPRG is obtained via an error-reduction procedure.

Theorem 1.1 (see also Theorem 4.7)**.** *There exists an explicit reduction that transforms a PRG for length-n, width-w branching programs with seed length s_0 and error $\varepsilon_0 = 1/\mathrm{poly}(n)$ into a WPRG with error ε and seed length*

$$s_0 + O\left(\log\frac{1}{\varepsilon} + \log\log\left(\frac{\log w}{\log n}\right)\cdot\log w\right).$$

Instantiating Theorem 1.1 with Armoni's PRG [Arm98], we obtain the following corollary.

Corollary 1.2. *There exists an explicit WPRG for length-n, width-w branching programs over alphabet Σ, achieving error ε with seed length*

$$\log_2|\Sigma| + O\left(\log\frac{1}{\varepsilon} + \frac{\log^2 n + \log n\cdot\log w}{\max(1,\log\log w - \log\log n)} + \log\log\left(\frac{\log w}{\log n}\right)\cdot\log w\right).$$

Corollary 1.2 improves upon all known WPRGs for general branching programs, including in the regime $w = n^{O(1)}$, where the seed length simplifies to

$$\log_2|\Sigma| + O\big(\log(1/\varepsilon) + \log^2 n\big).$$

In particular, it matches the recent construction of Cheng and Wu [CW26] in the parameters n, w, and ε, while improving the dependence on Σ. Although this dependence is not made explicit in [CW26] (nor in most prior work), it is straightforward to verify that their construction—as well as several other recent WPRGs (particularly [Hoz21], which is also state-of-the-art in the regime $w = n^{O(1)}$)—exhibits suboptimal dependence on Σ, typically incurring an additional multiplicative factor of $\frac{\log(1/\varepsilon)}{\log n}$.

By contrast, some earlier WPRG constructions (e.g., [BCG20, CDR+21]), while slightly inferior in overall seed length, achieve optimal dependence on Σ. Thus, the contribution of Corollary 1.2 is to attain optimal dependence on Σ while matching the best known bounds in all other parameters. Moreover, our construction and its analysis appear to be simpler than those of [CW26], which rely on "weighted pseudorandom reductions"—a sequence of alphabet-reduction and length-reduction steps combined with different PRGs and WPRGs. Furthermore, in the regime $w = n^{O(1)}$, our construction is significantly simpler, and is comparable in simplicity to [Hoz21].

As discussed above, our motivation for achieving optimal dependence on the alphabet size stems from its implications for breaking the log-squared barrier [CH22]. In Section 5, we provide an alternative proof of this implication that requires truly optimal dependence on the alphabet, namely $\log_2 |\Sigma|$, as achieved in Corollary 1.2 for WPRGs. We prove

Theorem 1.3. *Assume there exists an explicit PRG with seed length*

$$\log_2 |\Sigma| + O(\log n \cdot \log(nw) + \log(1/\varepsilon)).$$

Then there exists an explicit PRG with seed length

$$\log_2 |\Sigma| + O\Big(\log^{3/2} n + \log n \cdot \log w + \sqrt{\log n} \cdot \log(1/\varepsilon)\Big).^{1}$$

The Cheng–Hoza reduction (Lemma 4.9 in [CH22]) is a recursive construction that uses randomness samplers at each step. Our construction is also recursive, but uses only the hypothesized PRG. We also note that compared to Theorem 1.3, the applies even when the hypothesized PRG has only $O(\log |\Sigma|)$ dependence on the alphabet size.

2 Proof Overview

In this section, we provide an informal overview of Theorem 1.1, and outline our alternative proof of the Cheng–Hoza result as given in Theorem 1.3. The reader may skip this part and proceed directly to the formal sections of the paper. Recall that this theorem takes a PRG for length-n, width-w BPs with error $\varepsilon_0 = 1/n$ and seed length s_0, and produces a WPRG with error ε and seed length

$$s_0 + O\left(\log \frac{1}{\varepsilon} + \log\log\left(\frac{\log w}{\log n}\right) \cdot \log w\right)$$

in the regime $\frac{1}{\varepsilon} > w > n$. When $w < n$ (and more generally, for $w = n^{O(1)}$), the more involved step of our construction can be skipped, yielding a seed length $s_0 + O(\log \frac{1}{\varepsilon})$, and resulting in a significantly simpler construction.

The first component we need for proving Theorem 1.1 is a variant of the Richardson iteration which we now turn to discuss.

2.1 Incremental and asymmetric Richardson iteration

The transformation underlying Theorem 1.1 uses the Richardson iteration, which is by now the standard tool for reducing the error of a PRG. However, our use of the Richardson iteration is more refined, and we begin by briefly describing it here (see Section 4.1 for the formal and complete treatment). For our purposes, the Richardson iteration—or rather, the variant of it used here—is

[1] Using error reduction, one can obtain linear dependence on $\log(1/\varepsilon)$ at the cost of producing a WPRG rather than a PRG (see Corollary 5.5).

an algorithm Rich that takes as input two WPRGs W_L and W_R for the same length n and width w, with corresponding errors ε_L and ε_R, and produces a new WPRG, denoted

$$\mathrm{Rich}(W_L, W_R),$$

with error guarantee $\varepsilon_L \varepsilon_R$ and seed length

$$s_L + O\left(\log \frac{1}{\varepsilon_L \varepsilon_R}\right).$$

In fact, both the error guarantee and the seed length are slightly more intricate (see Lemma 4.1), but for the purpose of this overview, it is best to suppress the exact dependence.

How our use of Richardson iteration differs. There are several differences between our implementation of the Richardson iteration and those in the literature. While these distinctions provide useful context, the following discussion is primarily intended for experts; the reader may proceed with the proof overview without focusing on these differences.

In a nutshell, prior works apply the Richardson iteration to a single PRG using several correlated samples, taken with both positive and negative weights, to achieve error cancellation and thus obtain a lower error. The correlated seeds in these works are generated using an auxiliary PRG for BPs. In contrast, our approach reduces the error of two PRGs (and even WPRGs) which may be imbalanced in their error guarantees and seed lengths. The correlated seeds are obtained via extractors rather than PRGs, allowing for finer control over the seed length of the reduction, which is crucial in the presence of asymmetric WPRGs.

A second difference is our incremental application of the Richardson iteration: rather than applying it in a "one-shot" manner to achieve the desired error, we observe that it is preferable to invoke Richardson on just two instances and then compose the resulting procedure sufficiently many times. This approach is more economical, as the correlated seeds need not meet the final error guarantee from the start. Instead, the early iterations can operate with more coarsely correlated seeds.

Getting back to the proof overview of Theorem 1.1, the transformation proceeds in two phases:

- Phase 1: Reduce the error from $1/n$ to $1/w$.[2]
- Phase 2. Reduce the error from $1/w$ to the desired error ε.

2.2 Phase 1 – amplifying from inverse-length to inverse-width error

For the first phase—which is the more involved part, and can be skipped in the regime $w = n^{O(1)}$, yielding a simpler construction. The key technical transformation is summarized in the following

[2]Strictly speaking, we start with a slightly smaller error, e.g., $1/n^c$ for some constant $c > 1$. We ignore this technicality here.

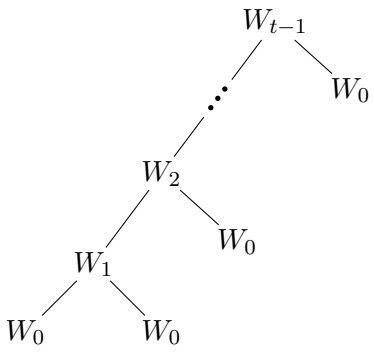

Figure 1: Recursive "comb" structure.

proposition (see Proposition 4.5 for the formal statement).

Proposition 2.1 (informal)**.** *For any integer $t \geq 1$, a WPRG with error guarantee $\varepsilon_0 < 1/n$ and seed length s_0 can be transformed into a WPRG with error $\varepsilon = \varepsilon_0^t$ and seed length*

$$s_0 + O\left(t^2 \cdot \log \frac{1}{\varepsilon_0} + \log w\right).$$

To be more precise, we assume a bound on the weights of the given WPRG as a function of the other parameters, in particular the error guarantee. This invariant, which we call *error-dominated* (see Definition 4.2), is preserved throughout our transformations, including the one given in Proposition 2.1. This ensures that the weights remain under control without the need to track them explicitly. In this informal overview, we will ignore them altogether.

The comb reduction. The proof of Proposition 2.1 proceeds by applying the Richardson iteration recursively in a "comb-like" manner, as depicted in Figure 1 (which may explain the name). More formally, given a WPRG W_0 with error ε_0 and seed length s_0, we define a sequence of WPRGs $W_1, \ldots, W_{t-1}$ by setting, for each $j \geq 1$,

$$W_j = \mathrm{Rich}\left(W_{j-1}, W_0\right).$$

It is easy to see that for each $j \geq 1$, the WPRG W_j has error guarantee $\varepsilon_j = \varepsilon_0^{j+1}$. Consequently, the seed length s_j satisfies the recurrence

$$s_j = s_{j-1} + O\left(\log \frac{1}{\varepsilon_0^{j+1}}\right),$$

which readily yields the seed length bound stated in Proposition 2.1. The additive $\log w$ term arises from a technical consideration that we suppress in this informal discussion.

Comb of combs. Applying Proposition 2.1 with $t = \frac{\log w}{\log(1/\varepsilon_0)}$ yields a WPRG with the desired error ε, albeit with a seed length that has a *quadratic* dependence on $\log w$. To understand how this can be avoided, we first ask what error we can "buy" with an additive $O(\log w)$ term in the seed – a cost we are willing to incur. Since the seed length in Proposition 2.1 is $s_0 + O(t^2 \cdot \log(1/\varepsilon_0))$, we can set

$$t = \sqrt{\frac{\log w}{\log(1/\varepsilon_0)}},$$

thereby increasing the seed length by an additive $O(\log w)$ term over s_0. The resulting error guaranteed by Proposition 2.1 is then $\varepsilon_1 = \varepsilon_0^{t_1}$. Let us (re)denote the resulting WPRG by W_1, which has seed length $s_1 = s_0 + O(\log w)$.

Generally, the error ε_1 is not sufficiently small, and so we repeat the process and construct a new WPRG $W_2 = \mathsf{Comb}_{t_2}(W_1)$ (hence the name "comb-of-combs"; see Figure 2 for an illustration.). Again, we wish to incur only an additive $O(\log w)$ increase in the seed length. Thus, we choose t_2 so that $t_2^2 \cdot \log(1/\varepsilon_1) = \log w$, which yields

$$t_2 = \left(\frac{\log w}{\log(1/\varepsilon_0)}\right)^{1/4} = \sqrt{t}.$$

The resulting error is then $\varepsilon_2 = \varepsilon_0^{t^{1+\frac{1}{2}}}$. Continuing in a recursive manner, we define $W_j = \mathsf{Comb}_{t_j}(W_{j-1})$, with $t_j = \sqrt{t_{j-1}}$. This results in error

$$\varepsilon_j = \varepsilon_0^{t^{1+\frac{1}{2}+\frac{1}{4}+\cdots+\frac{1}{2^{j-1}}}}$$

using seed length $s_0 + O(j \cdot \log w)$. Assuming now that $\varepsilon_0 = 1/n$, a calculation shows that for $\ell = \log\log\left(\frac{\log w}{\log n}\right)$, the WPRG W_ℓ achieves error $1/w$ and seed length $s_0 + O(\ell \cdot \log w)$, which is nearly linear in $\log w$, since ℓ is an extremely slowly growing function.

2.3 Phase 2 – amplifying from inverse-width error

Once the error is below $1/w$, or if we are already in the regime $w = n^{O(1)}$, there is no further benefit to using asymmetric Richardson iteration, although the incremental approach remains advantageous. To obtain the final WPRG, we therefore compose the Richardson iterations in a binary-tree-like fashion, with the initial WPRG placed at the leaves. More precisely, let W_0 denote the WPRG obtained in the previous sections, with error guarantee $1/w$ and seed length s_0. We then define, for $j = 1, \ldots, h = \log\left(\frac{\log(1/\varepsilon)}{\log w}\right)$,

$$W_j = \mathrm{Rich}(W_{j-1}, W_{j-1}).$$

It is easy to verify that W_h achieves the desired error guarantee ε and has seed length

$$s_0 + 2^h \log w = s_0 + O(\log(1/\varepsilon)).$$

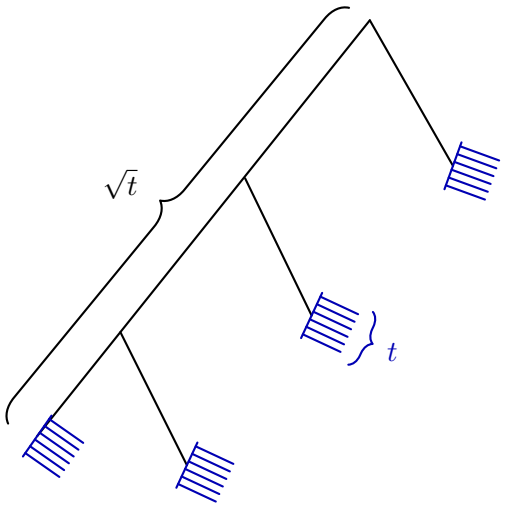

Figure 2: One level illustration of the comb of combs reduction.

To wrap up, starting with a PRG with error $1/n$ and seed length s_0, the overall seed length, accounting for both phases, is

$$s_0 + O\left(\log\frac{1}{\varepsilon} + \log\log\left(\frac{\log w}{\log n}\right)\cdot\log w\right),$$

as claimed in Theorem 1.1. Moreover, note that in the regime $w = n^{O(1)}$, only the second phase is invoked, yielding a seed length of $s_0 + O(\log\frac{1}{\varepsilon})$.

2.4 Improving Nisan's PRG via optimal alphabet and error dependence

In this section, we give an overview of our alternative proof of the Cheng–Hoza reduction, as given in Theorem 1.3, showing that optimal dependence on the alphabet and error suffices to break the log-squared barrier. More precisely, recall that the goal here is to break the log-squared dependence on the length n in Nisan's PRG, assuming the existence of a PRG with optimal alphabet and error dependence. Concretely, suppose we are given a PRG with seed length

$$\log|\Sigma| + O\left(\log^2 n + \log n\cdot\log w + \log\frac{1}{\varepsilon}\right).$$

Our goal in proving Theorem 1.3 is to show that there exists an efficient transformation which then yields a PRG with seed length

$$\log|\Sigma| + O\left(\log^{3/2} n + \log n\cdot\log w + \sqrt{\log n}\cdot\log\frac{1}{\varepsilon}\right).$$

As we bear no news regarding the dependence on the width w, indeed, our construction exhibits the same $\log n\cdot\log w$ dependence as all previous constructions, we ignore the width param-

eter for simplicity.[3] Under this simplification, we assume the existence of a PRG with seed length $\log|\Sigma|+O(\log^2 n+\log\frac{1}{\varepsilon})$, and aim to construct a PRG with seed length $\log|\Sigma|+O(\log^{3/2} n+\sqrt{\log n}\cdot\log\frac{1}{\varepsilon})$. The idea is to exploit the optimal dependence on Σ, as follows.

The PRG is constructed recursively with respect to the length n. Fix a parameter r (for the impatient reader, we note that we will take $r=\sqrt{\log n}$), and let $R=2^r$ and $t=\frac{\log n}{r}$. We construct a sequence of PRGs $G_0, G_1, \ldots, G_t$, all for the same width w and alphabet Σ, where

$$G_j\colon \{0,1\}^{d(j)} \to \Sigma^{R^j}.$$

Note that the final PRG G_t is for length-n BPs, and is the PRG produced by the transformation.

The first PRG, G_0, is simply the identity function, so $d(0)=\log_2|\Sigma|$. For $j\geq 1$ we define G_j by composing R concatenated copies of G_{j-1}. The naive approach would be to define

$$G_j(z)=G_{j-1}(z_1)\circ G_{j-1}(z_2)\circ\cdots\circ G_{j-1}(z_R),$$

where the seed z is partitioned into R disjoint blocks. But this results in an unacceptable blow-up in the seed length, $d(j)=R\cdot d(j-1)$, which ultimately yields a trivial PRG.

Instead, we use the hypothesized PRG to recycle the seeds of the R calls to G_{j-1}. In particular, let H_j denote the hypothesized PRG used to construct G_j. It has the form

$$H_j\colon \{0,1\}^{d(j)} \to \Sigma_j^R,$$

where $\Sigma_j=\{0,1\}^{d(j-1)}$. With H_j in place, we define G_j via "sparsified folding":[4]

$$G_j(z)=G_{j-1}\left(H_j(z)_1\right)\circ G_{j-1}\left(H_j(z)_2\right)\circ\cdots\circ G_{j-1}\left(H_j(z)_R\right).$$

Note that H_j is a PRG with very large alphabet size, exponential in the seed length of G_{j-1}, yet its output length is only R, which we will take to be much smaller. It is at this point that we capitalize on the excellent dependence on the alphabet size. Indeed, by our hypothesis, the seed length of H_j satisfies

$$d(j)=\underbrace{\log|\Sigma_j|}_{d(j-1)}+O\left(r^2+\log\frac{1}{\varepsilon_H}\right),$$

where ε_H is the error parameter of H_j (taken uniformly across the j-s). Solving the recurrence yields

$$d(t)=O\left(t\cdot\left(r^2+\log\frac{1}{\varepsilon_H}\right)\right)+\log|\Sigma|.$$

In Section 5, we prove that correlating the R seeds using a BP does indeed work, yielding a final

[3] Alternatively, the reader may think of the width w as a constant; recall that breaking the log-squared barrier remains open even for width $w=4$.

[4] A similar length reduction step is also used in [CW26].

error ε bounded by $n\varepsilon_H$. This, together with our choice $t = \frac{\log n}{r}$ yields that the seed length is

$$d(t) = O\left(\frac{\log n}{r} \cdot \left(r^2 + \log n + \log \frac{1}{\varepsilon}\right)\right) + \log |\Sigma|.$$

Optimizing by setting $r = \sqrt{\log n}$, we get the desired seed length.

3 Preliminaries

All logarithms in this paper are taken to base 2, unless otherwise stated. Whenever we write $x \sim A$ for A being a set, we mean that x is sampled uniformly at random from the flat distribution over A. We denote by U_n the random variable distributed uniformly over $\{0,1\}^n$.

We say $f\colon \Lambda_1 \to \Lambda_2^\star$ is explicit if it's computable in time $\mathrm{poly}(\log |\Lambda_1|)$. We mostly do not keep precise track of the space complexity of our error-reduction procedure. A suitable bound follows from standard results on the space complexity of compositions, together with the space complexity of the underlying PRG whose error is being reduced.

3.1 Matrices, Branching Programs, and Space Complexity

A matrix is Boolean if all its entries are in $\{0,1\}$, and stochastic if all its entries are nonnegative and the sum of each column is 1. Denote by $\mathrm{BSto}(w)$ the set of $w \times w$ boolean stochastic matrices. We will denote by $\|\cdot\|$ the induced ℓ_1 norm, i.e., $\|A\| = \max_j \sum_i |A_{i,j}|$.

We start by defining standard-order, read-once, branching programs.

Definition 3.1 (branching program)**.** *Let Σ be some alphabet and let $n, w \in \mathbb{N}$. An (n, w, Σ) branching program (BP) is a sequence $B = (B_1, \ldots, B_n)$, where each $B_i\colon \Sigma \to \mathrm{BSto}(w)$.*

For $i \le j$ we let $B_{i\to j}$ be the $(j - i + 1, \Sigma, w)$ BP $(B_i, \ldots, B_j)$.

Definition 3.2. *The* value *of an (n, w, Σ) BP $B = (B_1, \ldots, B_n)$ on $x = (x_1, \ldots, x_n) \in \Sigma^n$, denoted $B(x)$, is the realized $w \times w$ matrix of B when fed by x, i.e.,*

$$B(x) = B_n(x_n) \cdot B_{n-1}(x_{n-1}) \cdots B_1(x_1).$$

If B is the empty sequence, we set $\emptyset(x) = I_w$.

Definition 3.3 (weighted PRG)**.** *We say (G, μ) is an $(n, w, \Sigma, \Gamma, \varepsilon)$-WPRG against BPs with seed length s if:*

- *$G\colon \{0,1\}^s \to \Sigma^n$ and $\mu\colon \{0,1\}^s \to \mathbb{R}$,*
- *For every (n, w, Σ) BP $B = (B_1, \ldots, B_n)$, it holds that*

$$\left\| \mathbb{E}_{x\sim\{0,1\}^s}[\mu(x) \cdot B(G(x))] - \mathbb{E}_{x\sim\Sigma^n}[B(x)] \right\| \le \varepsilon,$$

- *For every* $x \in \{0,1\}^s$, $|\mu(x)| \leq \Gamma$.

When $\mu \equiv 1$, *the notion coincides with that of a PRG, in which case we omit the* Γ *from the tuple.*

For $1 \leq i \leq j \leq n$ we let $G_{i\to j} \colon \{0,1\}^s \to \Sigma^{j-i+1}$ be the corresponding truncation of G's output. Note that if (G,μ) is an $(n,w,\Sigma,\Gamma,\varepsilon)$ WPRG, then $(G_{i\to j},\mu)$ is an $(j-i+1,w,\Sigma,\Gamma,\varepsilon)$ WPRG. We will often omit the subscript $i \to j$ when the length of the corresponding BP is clear from context.

3.2 Known PRG Constructions

Theorem 3.4 ([INW94]). *For any positive integers* n, w, *any error parameter* $\varepsilon > 0$ *and any alphabet* Σ, *there exists an explicit* (n,Σ,w,ε) *PRG with seed length*

$$s = \log|\Sigma| + O\Big(\log n \cdot \log\Big(\frac{nw}{\varepsilon}\Big)\Big).$$

When $w \gg n$, Armoni's PRG [Arm98] improves upon the INW generator. When instantiated with modern extractors, it works for all ranges of parameters [KNW08], and has the optimal dependence on Σ [CT25].

Theorem 3.5 ([Arm98, KNW08, CT25]). *For any positive integers* n, w, *any error parameter* $\varepsilon > 0$ *and any alphabet* Σ, *there exists an explicit* (n,Σ,w,ε) *PRG with seed length*

$$s = \log|\Sigma| + O\left(\frac{\log(nw/\varepsilon)\log n}{\max(1,\log\log w - \log\log(n/\varepsilon))}\right).$$

3.3 Seeded Extractors

The *support* of a random variable X distributed over some domain Ω is the set $x \in \Omega$ for which $\Pr[X = x] \neq 0$, which we denote by $\mathrm{Supp}(X)$. The *total variation distance* (or, statistical distance) between two random variables X and Y over the same domain Ω is defined as $|X - Y| = \max_{A\subseteq\Omega}(\Pr[X \in A] - \Pr[Y \in A])$. Whenever $|X - Y| \leq \varepsilon$ we say that X is ε-close to Y and denote it by $X \approx_\varepsilon Y$.

For a function $f \colon \Omega_1 \to \Omega_2$ and a random variable X distributed over Ω_1, $f(X)$ is the random variable distributed over Ω_2 obtained by choosing x according to X and computing $f(x)$. For a set $A \subseteq \Omega_1$, $f(A) = \{f(x) : x \in A\}$. For every $f \colon \Omega_1 \to \Omega_2$ and two random variables X and Y distributed over Ω_1 it holds that $|f(X) - f(Y)| \leq |X - Y|$, and this is often referred to as the data-processing inequality.

Definition 3.6 (min-entropy). *Let* X *be a discrete random variable over a set* $\mathcal{X}$. *The* min-entropy *of* X, *denoted by* $H_\infty(X)$, *is defined as*

$$H_\infty(X) = \min_{x\in\mathcal{X}}(-\log\Pr[X = x]) = -\log\left(\max_{x\in\mathcal{X}}\Pr[X = x]\right). \tag{1}$$

Definition 3.7 ((n,k) source)**.** *An (n,k)* source *is a random variable X distributed over $\{0,1\}^n$ such that its min-entropy is at least k, i.e., $H_\infty(X) \geq k$. Equivalently, it is a distribution where for all $x \in \{0,1\}^n$,*

$$\Pr[X = x] \leq 2^{-k}. \tag{2}$$

Definition 3.8 (extractor)**.** *A function $\mathsf{Ext}\colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ is a (k,ε) (seeded) extractor if the following holds. For every (n,k) source X it holds that $\mathsf{Ext}(X,Y) \approx_\varepsilon U_m$, where Y is uniformly distributed over $\{0,1\}^d$ and is independent of X.*

We will use two extractors, one in the balanced regime, and one in the unbalanced regime.

Theorem 3.9 ([GW97])**.** *There exists a constant $c_{\mathsf{seed}} \geq 1$ such that the following holds. For every positive integer n, and any $\Delta < n$ and $\varepsilon > 0$, there exists an explicit $(k = n - \Delta, \varepsilon)$ extractor $\mathsf{Ext}\colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^n$, where $d = c_{\mathsf{seed}} \cdot (\Delta + \log(n/\varepsilon))$.*

Theorem 3.10 ([RVW02, CL20])**.** *There exist constants $c_{\mathsf{seed}}, c_{\mathsf{loss}} \geq 1$ such that the following holds. For every positive integer n, and any $\Delta < n$ and $\varepsilon > 0$, there exists an explicit $(k = n - \Delta, \varepsilon)$ extractor $\mathsf{Ext}\colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$, where $d = c_{\mathsf{seed}} \cdot \log(\Delta/\varepsilon)$ and $m = n - c_{\mathsf{loss}} \cdot (\Delta + \log \frac{1}{\varepsilon})$.*

4 Our Error Reduction Procedure

In this section we prove Theorem 1.1 and deduce Corollary 1.2. We begin with an informal overview of the Richardson iteration technique. This procedure depends on a parameter k that governs the quality of error reduction: roughly, an error-ε PRG is transformed into an error-ε^k WPRG. As discussed in the proof overview, in this work we observe that it is more economical to work with $k = 2$ and then compose the resulting WPRGs.

Our use of the Richardson iteration is asymmetric. In particular, in our error reduction we apply the iteration to WPRGs with varying error parameters. To set the stage, we first review, at an informal level, the Richardson iteration with parameter $k = 2$ in the symmetric setting. With this intuition in place, we then proceed in Section 4.1 to formally construct the Richardson iteration with parameter $k = 2$ for the asymmetric case.

Suppose we are given

- $w \times w$ sub-stochastic matrices $M_1, \ldots, M_n$, and,
- Crude approximations $\widetilde{M}_{i,j}$ to $M_i M_{i-1} \cdots M_j$ for all $1 \leq i < j \leq n$ (for $j = i$ we naturally take $\widetilde{M}_{i,i} = M_i$).

In the context of space bounded derandomization, [MRSV17] describe a method to use $M_1, \ldots, M_n$ to refine the crude approximations to much better approximations. First, they "plant" the matrix product problem within a matrix inversion problem. Specifically, they form the $(n+1) \times (n+1)$

block matrix

$$M = \begin{pmatrix} 0 & 0 & 0 & \cdots & 0 & 0 \\ M_1 & 0 & 0 & \cdots & 0 & 0 \\ 0 & M_2 & 0 & \ddots & \vdots & \vdots \\ \vdots & \vdots & \ddots & \ddots & 0 & 0 \\ 0 & 0 & \cdots & M_{n-1} & 0 & 0 \\ 0 & 0 & \cdots & 0 & M_n & 0 \end{pmatrix} \tag{3}$$

and define $L = I - M$. Then they observe that the inverse of L is:

$$(L^{-1})_{i,j} = \begin{cases} M_{i-1} \cdots M_j & \text{if } i > j, \\ I & \text{if } i = j, \\ 0 & \text{otherwise.} \end{cases}$$

Thus, the product we wish to approximate is a block within L^{-1}. We can use the crude approximations $\widetilde{M}_{i-1,j}$ to form the matrix

$$\widetilde{L^{-1}}_{i,j} = \begin{cases} \widetilde{M}_{i-1,j}, & \text{if } i > j, \\ I, & \text{if } i = j, \\ 0, & \text{if } i < j. \end{cases}$$

and $\widetilde{L^{-1}}$ is a crude approximation to L^{-1}. Prior works apply Richardson iteration to get a better approximation to L^{-1}. Specifically they take

$$\widehat{\widetilde{L^{-1}}} = 2\widetilde{L^{-1}} - \widetilde{L^{-1}} L \widetilde{L^{-1}}, \tag{4}$$

and it is immediate to check that

$$\left\| I - L\widehat{\widetilde{L^{-1}}} \right\| = \left\| I - L\widetilde{L^{-1}} \right\|^2.$$

Thus, $\widehat{\widetilde{L^{-1}}}$ yields a quadratically improved approximation of L^{-1} and therefore also of $M_1 \cdot \ldots \cdot M_n$. An explicit formula for the approximation of $M_1 \cdot \ldots \cdot M_n$ can be derived: For every $n \geq 3$,

$$\widehat{\widetilde{M}}_{n+1,1} = \sum_{r=1}^{n-2} \widetilde{M}_{n,r+2} M_{r+1} \widetilde{M}_{r,1} - \sum_{r=1}^{n-3} \widetilde{M}_{n,r+2} \widetilde{M}_{r+1,1}. \tag{5}$$

For example, for $n = 4$, the approximation of $M_4 M_3 M_2 M_1$ is

$$\widehat{\widetilde{M}}_{5,1} = \widetilde{M}_{4,3} M_2 M_1 + M_4 M_3 \widetilde{M}_{2,1} - \widetilde{M}_{4,3} \widetilde{M}_{2,1}.$$

Observe that Equation (5) is expressed as the difference of two sums. In the first (positive) sum, each summand is a product of three terms—two given approximations with an input matrix M_r sandwiched in the middle. In the second (negative) sum, the product omits this middle input matrix.

[PV21] and [CDR+21] independently notice that if the approximations $\widetilde{M}$ are obtained with a PRG, then the approximation $\widetilde{\widetilde{M}}$ can be realized with a WPRG [5]. A naive (and expensive) implementation of this would construct the WPRG for $\widetilde{\widetilde{M}}$ using two independent calls to the WPRGs for $\widetilde{M}$. A better idea is to recycle the seeds needed for the two calls to the WPRG of $\widetilde{M}$. In this work we explore what happens when we employ two different WPRGS, each with a different accuracy level, and we show that this leads to better WPRGs.

4.1 Asymmetric Richardson Iteration

In this section we make the above idea precise.

Construction. We are given two WPRGs, which we think of as a "left" WPRG and a "right" one, possibly with different parameters, and an extractor.

- An $(n, w, \Sigma, \Gamma_L, \varepsilon_L)$ WPRG (G^L, μ_L) with seed length s_L.
- An $(n, w, \Sigma, \Gamma_R, \varepsilon_R)$ WPRG (G^R, μ_R) with seed length s_R.
- A $(k, \varepsilon_{\mathsf{Ext}})$ extractor $\mathsf{Ext}\colon \{0,1\}^{s_L} \times \{0,1\}^{d_{\mathsf{Ext}}} \to \{0,1\}^{s_R}$, where $k = s_L - \log \frac{w}{\varepsilon_{\mathsf{Ext}}}$.

The construction of our new WPRG

$$\mathrm{Rich}_{\mathsf{Ext}}((G^L, \mu_L), (G^R, \mu_R)) = (G, \mu)$$

goes as follows. We interpret the set of seeds S as $[2n-5] \times \{0,1\}^{s_L} \times \{0,1\}^{d_{\mathsf{Ext}}} \times \{0,1\}^{d_{\mathsf{Ext}}}$. Given $(t, x_L, y_1, y_2) \in S$, we set $t' = t - n + 2$, and define

$$G(t, x_L, y_1, y_2) = \begin{cases} G^L_{1\to t}(x_L) \circ \mathsf{Ext}(x_L, y_2)_{[1,\log|\Sigma|]} \circ G^R_{t+2\to n}(\mathsf{Ext}(x_L, y_1)) & \text{if } t \le n-2, \\ G^L_{1\to t'+1}(x_L) \circ G^R_{t'+2\to n}(\mathsf{Ext}(x_L, y_1)) & \text{otherwise.} \end{cases}$$

$$\mu(t, x_L, y_1, y_2) = (2n-5) \cdot \begin{cases} \mu_L(x_L) \cdot \mu_R(\mathsf{Ext}(x_L, y_1)) & \text{if } t \le n-2, \\ -\mu_L(x_L) \cdot \mu_R(\mathsf{Ext}(x_L, y_1)) & \text{otherwise.} \end{cases}$$

Note that the construction exactly corresponds to Equation (5), where we use two different WPRGs for approximating the matrices. Notice that the first $n-2$ terms come with a positive sign, and the last $n-3$ terms with a negative sign.

[5] The use of weights here is essential, as some of terms in Equation (5) come with a negative weight.

Lemma 4.1. *Given (G^L, μ_L), (G^R, μ_R), and* Ext *as above, it holds that (G, μ) is an $(n, w, \Sigma, \Gamma, \varepsilon)$ WPRG, where:*

- *The seed length $s = \log |S|$ satisfies $s = \log(2n-5) + s_L + 2d_{\mathsf{Ext}}$,*
- *The weight of G is at most $\Gamma = c \cdot n\Gamma_L\Gamma_R$ for some constant $c \geq 1$, and,*
- *The error ε is bounded by $\varepsilon \leq 4(n+1)^2\varepsilon_L\varepsilon_R + \Gamma \cdot \varepsilon_{\mathsf{Ext}}$.*

Proof. The weighted generator (G, μ) treats each seed as a quadruple (t, x_L, y_1, y_2). Here $t \in [2n-5]$ specifies which of the $(n-2)$ "positive" or $(n-3)$ "negative" summands in the Richardson expansion is selected; $x_L \in \{0,1\}^{s_L}$ is drawn according to μ_L; and $y_1, y_2 \in \{0,1\}^{d_{\mathsf{Ext}}}$ are independent extractor seeds. Encoding t requires $\log(2n-5)$ bits, x_L requires s_L bits, and the two extractor seeds contribute $2\,d_{\mathsf{Ext}}$ bits in total, giving a seed length of

$$s = \log(2n-5) + s_L + 2\,d_{\mathsf{Ext}}.$$

Each output of (G, μ) has weight

$$\mu(t, x_L, y_1, y_2) = \pm(2n-5)\,\mu_L(x_L)\,\mu_R\big(\mathsf{Ext}(x_L, y_1)\big),$$

and therefore satisfies

$$|\mu(t, x_L, y_1, y_2)| \leq (2n-5)\,\Gamma_L\,\Gamma_R.$$

We move to analyzing the error of the weighted PRG.

Constructing approximate inverses. Let B be an (n, w, Σ) BP with transition matrices $M_1, \ldots, M_n$. Let L be the corresponding $(n+1)w \times (n+1)w$ block matrix defined in Equation (3) whose inverse L^{-1} encodes all partial products $M_i \cdots M_{j+1}$. We form two approximate inverses $L^{-1(L)}$ and $L^{-1(R)}$ by replacing each block $M_{i,j}$ with

$$\widetilde{M}^{(L)}_{i,j} = \mathop{\mathbb{E}}_{x \sim \{0,1\}^{s_L}}\Big[\mu_L(x) \cdot B_{i \to j}\big(G^L(x)\big)\Big] \quad \text{or} \quad \widetilde{M}^{(R)}_{i,j} = \mathop{\mathbb{E}}_{x \sim \{0,1\}^{s_R}}\Big[\mu_R(x) \cdot B_{i \to j}\big(G^R(x)\big)\Big],$$

respectively. Recalling that (G^L, μ_L) and (G^R, μ_R) have errors ε_L and ε_R, we have

$$\big\|M_{i,j} - \widetilde{M}^{(L)}_{i,j}\big\| \;\leq\; \varepsilon_L, \quad \big\|M_{i,j} - \widetilde{M}^{(R)}_{i,j}\big\| \;\leq\; \varepsilon_R.$$

Summing these errors over the $n+1$ block rows gives

$$\big\|L^{-1} - L^{-1(L)}\big\| \;\leq\; (n+1)\,\varepsilon_L, \quad \big\|L^{-1} - L^{-1(R)}\big\| \;\leq\; (n+1)\,\varepsilon_R.$$

As $\|L\| \leq 2$,

$$\big\|I - L^{-1(L)}L\big\| \;\leq\; 2(n+1)\,\varepsilon_L, \qquad \big\|I - L^{-1(R)}L\big\| \;\leq\; 2(n+1)\,\varepsilon_R.$$

Asymmetric Richardson. "Vanilla" Richardson refines a *single* crude inverse $A \approx L^{-1}$ via

$$A \longmapsto 2A - ALA,$$

which satisfies $\|I - (2A - ALA)L\| = \|I - AL\|^2$. In principle, introducing two approximations looks unnecessary–one could just reuse the more accurate one. In our setting, however, the two appearances of $\widetilde{L^{-1}}$ are generated from *different seeds*: the left term is driven by x_L, while the right term is obtained from x_L through an extractor and therefore may have different accuracy. This asymmetry naturally leads us to use a Richardson-type update that mixes two approximations,

$$R = L^{-1(L)} + L^{-1(R)} - L^{-1(L)} L L^{-1(R)}.$$

A short calculation shows that

$$I - RL = (I - L^{-1(L)}L)(I - L^{-1(R)}L).$$

Consequently,

$$\|I - RL\| \le \|I - L^{-1(L)}L\| \, \|I - L^{-1(R)}L\| \le 4(n+1)^2 \varepsilon_L \varepsilon_R,$$

and in particular the bottom-left block satisfies

$$\||R[n+1,1] - M_n \cdots M_1\| \le 4(n+1)^2 \varepsilon_L \varepsilon_R.$$

Uncorrelated seeds. Assume that the seed x_R for G_R and the symbol $\sigma \in \Sigma$ are sampled independently from x_L and uniformly. Given $t \in [2n-5]$, we consider the above two cases. When $t \in [n-2]$, letting $r = t$, the generator outputs $G^L_{1\to r}(x_L) \circ \sigma \circ G^R_{r+2\to n}(x_R)$. Then,

$$\mathbb{E}_{x_L, x_R, \sigma}\Big[\mu_L(x_L)\, \mu_R(x_R)\, B\big(G^L_{1\to r}(x_L) \circ \sigma \circ G^R_{r+2\to n}(x_R)\big)\Big] = \widetilde{M}^{(L)}_{n,r+2}\, M_{r+1}\, \widetilde{M}^{(R)}_{r,1}.$$

When $t > n-2$, we denote r to be such that $t = n - 2 + r$, and now

$$\mathbb{E}_{x_L, x_R}\Big[\mu_L(x_L)\, \mu_R(x_R)\, B\big(G^L_{1\to r+1}(x_L) \circ G^R_{r+2\to n}(x_R)\big)\Big] = -\,\widetilde{M}^{(L)}_{n,r+2}\, \widetilde{M}^{(R)}_{r+1,1}.$$

Averaging over $t \in [2n-5]$ and multiplying by $2n-5$ yields exactly

$$\sum_{r=1}^{n-2} \widetilde{M}^{(L)}_{n,r+2}\, M_{r+1}\, \widetilde{M}^{(R)}_{r,1} - \sum_{r=1}^{n-3} \widetilde{M}^{(L)}_{n,r+2}\, \widetilde{M}^{(R)}_{r+1,1},$$

which is the $(n+1,1)$-block of the Richardson matrix R. Thus, when x_R and σ are drawn independently, the weighted generator's output matches $R[n+1,1]$.

Correlating x_R and σ with x_L via two extractor calls. We now replace the independent choices (x_R, σ) by

$$x_R = \mathsf{Ext}(x_L, y_1) \in \{0,1\}^{s_R}, \quad \sigma = \mathsf{Ext}(x_L, y_2)_{[1,\log|\Sigma|]} \in \Sigma,$$

where y_1, y_2 are independent uniform seeds (and independent of x_L). We bound the difference in the ℓ_1 norm of the $(n{+}1, 1)$-block contribution.

We use the standard seed-recycling argument for width-w ROBPs. Conditioning on reaching the state $q \in [w]$ after reading x_L reveals at most, roughly, $\log w$ bits of information; Formally, except with probability at most $\varepsilon_{\mathsf{Ext}}$, this conditioning reduces the *min-entropy* of the left seed by at most $\log(w/\varepsilon_{\mathsf{Ext}})$. Setting $\Delta := \log(w/\varepsilon_{\mathsf{Ext}})$, and using our extractor $\mathsf{Ext} : \{0,1\}^{s_L} \times \{0,1\}^{d_{\mathsf{Ext}}} \to \{0,1\}^{s_R}$ with entropy loss Δ and error $\varepsilon_{\mathsf{Ext}}$, the condition

$$s_L - \Delta \geq s_L - \log \frac{w}{\varepsilon_{\mathsf{Ext}}} = k$$

guarantees that $\mathsf{Ext}(x_L, U)$ is $\varepsilon_{\mathsf{Ext}}$-close to uniform even after conditioning on q. Consequently, for any functions f_L, f_R,

$$\left\| \mathbb{E}_{x_L,y}\big[B\big(f_L(x_L) \circ f_R(\mathsf{Ext}(x_L, y))\big)\big] - \mathbb{E}_{x_L,x_R}\big[B\big(f_L(x_L) \circ f_R(x_R)\big)\big] \right\| \leq 2\varepsilon_{\mathsf{Ext}}.$$

Taking the weights into account, recalling that $|\mu_L| \leq \Gamma_L$ and $|\mu_R| \leq \Gamma_R$, we have

$$\begin{aligned}\Big\| &\mathbb{E}_{x_L,y}\big[(2n-5)\mu_L(x_L)\mu_R(\mathsf{Ext}(x_L, y)) \cdot B(f_L(x_L) \circ f_R(\mathsf{Ext}(x_L, y)))\big] \\ &\qquad - \mathbb{E}_{x_L,x_R}\big[(2n-5)\mu_L(x_L)\mu_R(x_R) \cdot B(f_L(x_L) \circ f_R(x_R))\big] \Big\| \leq c \cdot (2n-5)\Gamma_L\Gamma_R\varepsilon_{\mathsf{Ext}}\end{aligned}$$

for some constant $c \geq 1$. The ultimate error bound is obtained by the triangle inequality.

□

To conclude this section, we define a useful invariant which will be helpful towards establishing our WPRG.

Definition 4.2. *We say that an $(n, w, \Sigma, \Gamma, \varepsilon)$ WPRG is* error-dominated *if*

$$\Gamma \leq \frac{1}{c(n+1)^3\varepsilon},$$

where c is the same constant from Lemma 4.1.

Note that error-dominated WPRGs necessarily have low error. We record the following observation.

Claim 4.3. *Keeping the notation above, particularly the constant c from Lemma 4.1, assume that both (G^L, μ_L) and (G^R, μ_R) are error-dominated, and that*

$$\varepsilon_{\mathsf{Ext}} \leq \frac{n^2\varepsilon_L\varepsilon_R}{c\Gamma}.$$

Then, (G, μ) is error-dominated.

Proof. By Lemma 4.1, the composed generator satisfies

$$\Gamma = cn\,\Gamma_L\Gamma_R \qquad \text{and} \qquad \varepsilon \le 4(n+1)^2\varepsilon_L\varepsilon_R + \Gamma\varepsilon_{\mathsf{Ext}}.$$

Using $\varepsilon_{\mathsf{Ext}} \le \frac{n\varepsilon_L\varepsilon_R}{c\Gamma}$, we get

$$\varepsilon \le \big(4(n+1)^2 + n^2\big)\varepsilon_L\varepsilon_R \le 5(n+1)^2\varepsilon_L\varepsilon_R.$$

Since (G^L, μ_L) and (G^R, μ_R) are error-dominated,

$$\Gamma_L \le \frac{1}{c(n+1)^3\varepsilon_L} \qquad \text{and} \qquad \Gamma_R \le \frac{1}{c(n+1)^3\varepsilon_R}.$$

Therefore,

$$\Gamma \le cn\Gamma_L\Gamma_R \le \frac{cn}{c^2(n+1)^6}\cdot\frac{1}{\varepsilon_L\varepsilon_R} \le \frac{cn}{c^2(n+1)^6}\cdot\frac{5(n+1)^2}{\varepsilon} = \frac{n}{c(n+1)^4}\cdot\frac{1}{\varepsilon} \le \frac{1}{c(n+1)^3\varepsilon}.$$

□

4.2 A Comb Reduction

In this section we define a comb-like construction, that composes our asymmetric Richardson construction given in Section 4.1 as follows. For some base (error-dominated) WPRG W_0, and a sequence of extractors $\mathsf{Ext}_1, \ldots, \mathsf{Ext}_t$, we define W_i recursively as

$$W_i = \mathrm{Rich}_{\mathsf{Ext}_i}(W_{i-1}, W_0).$$

We denote by $W_0 = (G_0, \mu_0)$ our $(n, w, \Sigma, \Gamma_0, \varepsilon_0)$ error-dominated WPRG, and by s_0 its seed length. We proceed to set the parameters of our extractors, each of the t extractors

$$\mathsf{Ext}_i \colon \{0,1\}^{s(i)} \times \{0,1\}^{d(i)} \to \{0,1\}^{s_0}$$

is given in Theorem 3.10 and set to have error $\varepsilon_{\mathsf{Ext}}(i)$, where the $\varepsilon_{\mathsf{Ext}}(i)$'s are as follows. Define, for each $i \in [t]$, $\varepsilon(i) = (5(n+1)^2\varepsilon_0)^i\varepsilon_0$. Note that the sequence of $\varepsilon(i)$-s satisfy $\varepsilon(i) = 5(n+1)^2\varepsilon(i-1)\cdot\varepsilon_0$. For each $i \in [t]$ set

$$\varepsilon_{\mathsf{Ext}}(i) = \frac{n^2\varepsilon_0}{c\Gamma(i)}\cdot\varepsilon(i-1),$$

where $\Gamma(i) = (cn\Gamma_0)^i\Gamma_0$. Note that the $\Gamma(i)$-s satisfy the recurrence relation $\Gamma(i) = cn\Gamma(i-1)\Gamma_0$.

By Lemma 4.1, the input lengths satisfy $s(1) = s_0 + c_{\mathsf{seed}}\log(w/\varepsilon_{\mathsf{Ext}}(1))$, and for each $2 \le i \le t$,

$$s(i+1) = s(i) + 2c_{\mathsf{seed}}\log(\log w/\varepsilon_{\mathsf{Ext}}(i+1)) + \log(2n-5).$$

We record that:

Claim 4.4. *For every $i \in [t]$ it holds that*

$$\log \frac{1}{\varepsilon_{\mathsf{Ext}}(i)} = O\left(i \log \frac{n\Gamma_0}{\varepsilon_0}\right),$$
$$s(i) = s_0 + O\left(\log w + i^2 \log \frac{n\Gamma_0}{\varepsilon_0}\right),$$

and

$$d(i) = O\left(\log \log w + i \log \frac{n\Gamma_0}{\varepsilon_0}\right).$$

Proof. By definition of $\varepsilon_{\mathsf{Ext}}(i)$ and error domination from Claim 4.3:

$$\varepsilon_{\mathsf{Ext}}(i) = \frac{\varepsilon(i-1)\varepsilon_0 n^2}{c\Gamma(i)} \geq \frac{1}{c} \cdot \varepsilon(i-1)\varepsilon_0 n^2 \cdot c(n+1)^3 \varepsilon(i) = \frac{(n+1)n^2\varepsilon(i)^2}{5},$$

so

$$\log \frac{1}{\varepsilon_{\mathsf{Ext}}(i)} = O\left(i \cdot \log \frac{n\Gamma_0}{\varepsilon_0}\right).$$

For the WPRG's seed length, we have

$$\begin{aligned} s(i) &= s_0 + s(1) + (i-1) \cdot \log(2n-5) + 2c_{\mathsf{seed}} \sum_{k=2}^{i} \log \frac{\log w}{\varepsilon_{\mathsf{Ext}}(k+1)} \\ &= s_0 + O\left(\log w + i \log \log w + i^2 \log \frac{n\Gamma_0}{\varepsilon_0}\right). \end{aligned}$$

By the construction given in Section 4.1, the entropy loss of each extractor is $\Delta(i) = \log(w/\varepsilon_{\mathsf{Ext}}(i))$. Thus, we can indeed output s_0 bits as

$$s(i) \geq s_0 + c_{\mathsf{loss}} \left(\Delta(i) + \log \frac{1}{\varepsilon_{\mathsf{Ext}}(i)}\right).$$

We thus get that

$$d(i) = O\left(\log \log w + \log \frac{1}{\varepsilon_{\mathsf{Ext}}(i)}\right) = O\left(\log \log w + i \log \frac{n\Gamma_0}{\varepsilon_0}\right).$$

□

We denote this section's transformation, which on input W_0 construct W_t, as $\mathsf{Comb}_t(W_0)$. The parameters of the construction are summarized in the following theorem, whose proof readily follows from Lemma 4.1, Claim 4.3, and Claim 4.4.

Proposition 4.5. *Keeping the above notation, assuming $\varepsilon_0 \leq 1/n^5$ and that $s_0 \geq \log|\Sigma| + \Omega(\log(w/\varepsilon_0))$,*

$\mathsf{Comb}_t(W_0)$ *is an error-dominated* $(n, w, \Sigma, \Gamma_t, \varepsilon_t)$ *WPRG with seed length*

$$s_0 + O\left(t^2 \log \frac{1}{\varepsilon_0} + \log w\right),$$

and error $\varepsilon_t = \varepsilon_0^{t/2}$.

We emphasize that error domination of the WPRG is preserved, and moreover, that if W_0 is explicit, W_t is explicit as well.

4.3 Amplifying from Inverse-Length to Inverse-Width Error

To reduce the error from $n^{-\Omega(1)}$ to $w^{-\Omega(1)}$, we define a "comb-of-combs" construction as follows. Letting W_0 be an error dominated $(n, w, \Sigma, \Gamma_0, \varepsilon_0)$ WPRG that satisfies the assumptions of Proposition 4.5, for some $t_1, \ldots, t_\ell$, we define the sequence of WPRGs defined by

$$W_i = \mathsf{Comb}_{t_i}(W_{i-1}),$$

and set $\mathsf{CoC}_{t_1,\ldots,t_\ell}(W_0) = W_\ell$.

Concretely, we start with $\varepsilon_0 = 1/n^5$, choose

$$\ell = \log\log\left(\frac{\log w}{\log n}\right),$$

and for each $i \in [\ell]$, we take

$$t_i = 2\left(\frac{\log w}{\log n}\right)^{2^{-i}}.$$

Under this choice of parameters, $\mathsf{CoC}_{t_1,\ldots,t_\ell}(W_0)$ satisfies the following.

Proposition 4.6. *Keeping the above notation,*

$$\mathsf{CoC}_{t_1,\ldots,t_\ell}(W_0)$$

is an explicit error-dominated $(n, w, \Sigma, \Gamma, \varepsilon)$ *WPRG with* $\varepsilon = 1/w$*, seed length* $s_0 + O(\ell \log w)$.

Proof. First, note that the fact that W_ℓ is error dominated already follows from Proposition 4.5, and we proceed to analyzing the error. Letting ε_i be the error of W_i, by Proposition 4.5 we know that $\varepsilon_i = \varepsilon_{i-1}^{t_i/2}$. By our choice of t_i-s, denoting $r = \frac{\log w}{\log n}$, we get that

$$\varepsilon_\ell = \varepsilon_0^{2^{-\ell}\prod_{i\in[\ell]} t_i} = \varepsilon_0^{\prod_{i\in[\ell]} r^{2^{-i}}} = \varepsilon_0^{r^{1-2^{-\ell}}}.$$

Now, $2^\ell = \log r$, so $r^{1-2^{-\ell}} = r/2$. Recalling that $\varepsilon_0 = n^{-5}$, we get $\varepsilon_0^{r/2} = n^{-5r/2} \leq 1/w$.

For the seed length, denote by s_i the seed length of W_i. By Proposition 4.5, we know that each $s_i = s_{i-1} + O(t_i^2 \log(1/\varepsilon_{i-1}))$. Fixing some $i \in [\ell]$, note from the above that

$$t_i^2 \cdot \log \frac{1}{\varepsilon_{i-1}} = t_i^2 \cdot r^{1-2^{-(i-1)}} \log \frac{1}{\varepsilon_0} = 4r^{2^{-i+1}} \cdot r^{1-2^{-i+1}} \log \frac{1}{\varepsilon_0} = 20 \log n \cdot r = 20 \log w.$$

Thus, $s_i = s_{i-1} + O(\log w)$ and we're done with the seed length caclculation. Finally, note that in order to invoke Proposition 4.5, we need to make sure that $s_i \geq \log |\Sigma| + \Omega(\log(w/\varepsilon_i))$, which follows from the fact that $t_i^2 \log(1/\varepsilon_{i-1}) = \Omega(\log(1/\varepsilon_i))$ and that $s_0 \geq \log |\Sigma| + \Omega(\log w)$. □

4.4 Amplifying to an Arbitrary Error

Once we achieved an error of $1/w$, we reduce the error to an arbitrary desired error ε by recursively applying the Richardson iteration on the same WPRG, i.e., at the base of the recursion we use CoC above on the given base-WPRG W. More formally, we start with $V_0 = \mathsf{CoC}_{t_1,\ldots,t_\ell}(W)$ as above, set

$$h = \log \left(\frac{\log(1/\varepsilon)}{\log(w/6n^2)} \right),$$

and for each $i \in [h]$, define

$$V_i = \mathrm{Rich}_{\mathsf{Ext}_i}(V_{i-1}, V_{i-1}),$$

where the extractors $\mathsf{Ext}_1, \ldots, \mathsf{Ext}_h$ are defined as follows. We denote the seed length of each $(n, w, \Sigma, \Gamma_i, \varepsilon_i)$ WPRG V_i by s_i, and the seed length of W by $s(W)$. Each

$$\mathsf{Ext}_i \colon \{0,1\}^{s_{i-1}} \times \{0,1\}^{d_i} \to \{0,1\}^{s_{i-1}}$$

is taken to be the $(k_i, \varepsilon_{\mathsf{Ext}_i})$ extractor of Theorem 3.9 as in Section 4.2, where $k_i = s_{i-1} - \log(w/\varepsilon_{\mathsf{Ext}_i})$, and $\varepsilon_{\mathsf{Ext}_i} = \frac{n^2 \varepsilon_{i-1}^2}{c\Gamma_i}$. By Lemma 4.1, we have the recurrence relation relating the seed lengths of the WPRGs

$$s_i = s_{i-1} + 2d_i + O(\log n) = s_{i-1} + O(\log(1/\varepsilon_{\mathsf{Ext}_i})),$$

and the recursive relation of the errors is $\varepsilon_i \leq 6n^2 \varepsilon_{i-1}^2$. Solving for the recursion, recalling $\varepsilon_0 = 1/w$, we get

$$\varepsilon_i \leq \left(\frac{6n^2}{w} \right)^{2^i}.$$

Our choice of h guarantees a final error of $\varepsilon_h \leq \varepsilon$.

By Theorem 3.9, the seed length

$$d_i = O\left(\log(w/\varepsilon_{\mathsf{Ext}_i})\right) = O(\log(1/\varepsilon_i)) = O\left(2^i \log w\right),$$

and so the recurrence simplifies to $s_i = s_{i-1} + O(2^i \log w)$, yielding

$$s_h = s_0 + O(2^h \log w) = s_0 + O(\log(1/\varepsilon)).$$

By Proposition 4.6,

$$s_0 = s(W) + O\left(\log\log\left(\frac{\log w}{\log n}\right) \cdot \log w\right),$$

and so the final seed length is

$$s_h = s(W) + O\left(\log\frac{1}{\varepsilon} + \log\log\left(\frac{\log w}{\log n}\right) \cdot \log w\right).$$

We summarize this in the following theorem, which is the formal version of Theorem 1.1 whose proof follows by the above calculations, also noting that the constructed WPRGs remain error-dominated and explicit.

Theorem 4.7. *Keeping the above notation, let W be an explicit error-dominated $(n, w, \Sigma, \Gamma_0, \varepsilon_0 = 1/n^5)$ WPRG with seed length $s(W)$. Then, V_h is an error-dominated $(n, w, \Sigma, \Gamma, \varepsilon)$ WPRG with seed length*

$$s(W) + O\left(\log\frac{1}{\varepsilon} + \log\log\left(\frac{\log w}{\log n}\right) \cdot \log w\right).$$

Finally, for our base-WPRG we'll choose Armoni's explicit PRG from Theorem 3.5.

Corollary 4.8. *For any positive integers n, w such that $w > n$, any $\varepsilon > 0$, and any alphabet Σ, there exists an explicit $(n, w, \Sigma, \Gamma, \varepsilon)$ WPRG with seed length*

$$\log|\Sigma| + O\left(\frac{\log^2 n + \log n \cdot \log w}{\log\log w - \log\log n} + \log\log\left(\frac{\log w}{\log n}\right) \cdot \log w + \log\frac{1}{\varepsilon}\right).$$

and $\Gamma \leq \max\left\{1, \frac{1}{\mathrm{poly}(n)\cdot\varepsilon}\right\}$. In particular, when $w = 2^{n^\alpha}$ for some constant $\alpha \in (0, 1)$, the seed length above becomes

$$\log|\Sigma| + O\left((\log\log\log w) \cdot \log w + \log\frac{1}{\varepsilon}\right).$$

When w is only polynomial in n, we can perform the error reduction step of Section 4.4 directly, and for the base-PRG, we can use the simpler INW generator (see Theorem 3.4) and get the following corollary.

Corollary 4.9. *For any positive integers n, w such that $w = n^{O(1)}$, any $\varepsilon > 0$, and any alphabet Σ, there exists an explicit $(n, w, \Sigma, \Gamma, \varepsilon)$ WPRG with seed length*

$$\log|\Sigma| + O\left(\log^2 n + \log\frac{1}{\varepsilon}\right)$$

and $\Gamma \leq \max\left\{1, \frac{1}{\mathrm{poly}(n)\cdot\varepsilon}\right\}$.

5 Better PRGs via Optimal Alphabet and Error Dependence

In this section we give an alternative to the Cheng-Hoza reduction as stated in Theorem 1.3. For convenience, we restate the assumption of the theorem.

Assumption 1. *For any $n, w \in \mathbb{N}$, alphabet Σ, and error parameter $\varepsilon > 0$, there exists an explicit $(n, w, \varepsilon, \Sigma)$ PRG with seed length*

$$d_H(n, \Sigma) = \log |\Sigma| + O\left(\log^2 n + \log n \cdot \log w + \log \frac{1}{\varepsilon}\right).$$

Note that in the above notation, $d_H(n, \Sigma)$ suppresses the dependence on w and ε, as these remain fixed throughout the reduction.

We construct our PRG in a recursive manner, via "sparsified folding" as informally discussed in Section 2.4: Suppose we want to fool a BP of length $R_0 \cdot R_1$, and we have a PRG G_0 with seed length d_0 that fools length R_0, and a PRG G_1 that fools length R_1 and alphabet $\{0,1\}^{d_0}$ with seed length d_1. Then, one can use G_1 to produce R_1 (correlated) seeds for G_0, namely, the concatenated PRG G has seed length d_1, and for any $z \in \{0,1\}^{d_1}$,

$$G(z) = G_0(G_1(z)_1) \circ G_0(G_1(z)_2) \circ \ldots \circ G_0(G_1(z)_{R_1}) \in \{0,1\}^{R_0 \cdot R_1}.$$

We instantiate this idea recursively, using as the inner PRG the one hypothesized in Assumption 1, as follows.

The construction. Extending this idea to length n, we fix some integer $r < \log n$, and denote $R = 2^r$. Also, let $t = \frac{\log n}{r}$, so $R^t = n$. For $i \in \{0, 1, \ldots, t\}$, we define

$$G_i \colon \{0,1\}^{d(i)} \to \Sigma_0^{R^i}$$

as follows. The PRG G_0 is just the identity over Σ, which we find convenient to denote as Σ_0 throughout the proof. Thus, $d(0) = \log |\Sigma_0|$. Given some $i \geq 1$, identify Σ_i with $\{0,1\}^{d(i-1)}$, and let $H_i \colon \{0,1\}^{d(i)} \to \Sigma_i^R$ be the PRG from Assumption 1, with seed length $d(i) = d_H(R, \Sigma_i)$ and error ε_H to be chosen later on. Then, for $z \in \{0,1\}^{d(i)}$, we define

$$G_i(z) = G_{i-1}(H_i(z)_1) \circ \ldots \circ G_{i-1}(H_i(z)_R) \in \Sigma_0^{R^i}.$$

Seed length. First, let us determine the seed length of G_i. By our assumption,

$$d(i) = d_H(R, \Sigma_i) = d(i-1) + O\left(r^2 + r \cdot \log w + \log \frac{1}{\varepsilon_H}\right),$$

and $d(0) = \log|\Sigma_0|$. Thus, the seed length for length n is given by

$$d(t) = \log|\Sigma_0| + O\left(r \cdot \log n + \log n \cdot \log w + \frac{\log n}{r} \cdot \log\frac{1}{\varepsilon_H}\right). \tag{6}$$

Correctness. Complementing the correctness proof amounts to tracking the error throughout the recursion. To this end, let $\varepsilon(i)$ be the error of G_i. The following lemma will be the crux of our correctness proof.

Lemma 5.1. *For $i \geq 1$,*

$$\varepsilon(i) \leq R \cdot \varepsilon(i-1) + \varepsilon_H.$$

Proof. Fix an (R^i, w, Σ_0) BP B. We wish to bound

$$\left\|\mathbb{E}\left[B\left(G_i(U_{d(i)})\right)\right] - \mathbb{E}\left[B\left(U_{\Sigma_0^{R^i}}\right)\right]\right\| \triangleq \|\delta - \mu\|.$$

Denote

$$G_{\mathsf{hyb}} = G_{i-1}(U^{(1)}) \circ \ldots \circ G_{i-1}(U^{(R)})$$

where all the $U^{(i)}$-s are uniform over Σ_i and independent of one another. Denote $\Delta_0 = \|\mu - \mathbb{E}[B(G_{\mathsf{hyb}})]\|$ and $\Delta_1 = \|\mathbb{E}[B(G_{\mathsf{hyb}})] - \delta\|$, noticing that $\|\delta - \mu\| \leq \Delta_0 + \Delta_1$.

Claim 5.2. *It holds that $\Delta_1 \leq \varepsilon_H$.*

Proof. Let B^{F} be the (R, w, Σ_i) BP that is the outcome of a "folding" operation of B with respect to G_{i-1}. That is, every R^{i-1} consecutive layers are transformed into a single layer, where the transitions within each layer are determined according to the PRG G_{i-1}. Notice that $B^{\mathsf{F}}(U) = B(G_{\mathsf{hyb}})$, and that

$$\delta = \mathbb{E}[B^{\mathsf{F}}(H_i(U_{d(i)}))].$$

Since H_i fools B^{F}, it readily follows that $\Delta_1 \leq \varepsilon_H$. □

Claim 5.3. *It holds that $\Delta_0 \leq R \cdot \varepsilon(i-1)$.*

Proof. The proof goes via a standard hybrid argument. For $j \in \{0, \ldots, R\}$, denote

$$G^j_{\mathsf{hyb}} = G_{i-1}(U^{(1)}) \circ \ldots \circ G_{i-1}(U^{(j)}) \circ \widetilde{U}^{(j+1)} \circ \ldots \circ \widetilde{U}^{(R)},$$

where the $\widetilde{U}$-s are uniform over $\Sigma_0^{R^{i-1}}$ and independent of one another, and of the $U^{(j)}$-s. Notice that $G_{\mathsf{hyb}} = G^{(R)}_{\mathsf{hyb}}$ and $G^{(0)}_{\mathsf{hyb}}$ is the uniform distribution over $\Sigma_0^{R^i}$. Now,

$$\Delta_0 \leq \sum_{j=1}^{R} \left\|B\left(G^{(j)}_{\mathsf{hyb}}\right) - B\left(G^{(j-1)}_{\mathsf{hyb}}\right)\right\| \triangleq \sum_{j=1}^{R} \Delta_0^j.$$

Fix some $j \in [R]$ and denote by $S = \widetilde{U}^{(j+1)} \circ \ldots \circ \widetilde{U}^{(R)}$ the length-$(R-j)R^{i-1}$ suffix of both $G_{\mathsf{hyb}}^{(j)}$ and $G_{\mathsf{hyb}}^{(j-1)}$. For brevity, let $T = R^{i-1}$. Now,

$$\begin{aligned}\Delta_0^j &= \left\| \mathbb{E}_{x\sim S}\left[B_{jT+1\to RT}(x)\right] \cdot B_{1\to jT}\left((G_{\mathsf{hyb}}^{(j)})_{[1,j]}\right) - \mathbb{E}_{x\sim S}\left[B_{jT+1\to RT}(x)\right] \cdot B_{1\to jT}\left((G_{\mathsf{hyb}}^{(j-1)})_{[1,j]}\right)\right\| \\ &\leq \left\|\mathbb{E}_{x\sim S}\left[B_{jT+1\to RT}(x)\right]\right\| \cdot \left\| B_{1\to jT}\left((G_{\mathsf{hyb}}^{(j)})_{[1,j]}\right) - B_{1\to jT}\left((G_{\mathsf{hyb}}^{(j-1)})_{[1,j]}\right)\right\|,\end{aligned}$$

and the first term is bounded by 1. We repeat the same argument for the prefix. Namely, let $P = G_{i-1}(U^{(1)}) \circ \ldots \circ G_{i-1}(U^{(j-1)})$, and we can bound

$$\Delta_0^j \leq \left\|B_{1\to(j-1)T}(P)\right\| \cdot \left\| B_{(j-1)T+1\to jT}\left(G_{i-1}(U^{(j)})\right) - B_{(j-1)T+1\to jT}\left(\widetilde{U}^{(j)}\right)\right\|.$$

The first term is bounded by 1, and second one is bounded by $\varepsilon(i-1)$, since G_{i-1} fools the corresponding BP.

□

□

Lemma 5.1 tells us that

$$\varepsilon(t) \leq \varepsilon_H \cdot \sum_{i=0}^{t-1} R^i \leq n \cdot \varepsilon_H.$$

Choosing $\varepsilon_H = \frac{\varepsilon}{n}$ in Equation (6), we get an overall seed length of

$$\log|\Sigma_0| + O\left(r \cdot \log n + \log n \cdot \log w + \frac{\log^2 n}{r} + \frac{\log n}{r} \cdot \log\frac{1}{\varepsilon}\right).$$

For $r = \sqrt{\log n}$, this gives

$$\log|\Sigma_0| + O\left(\log^{3/2} n + \log n \cdot \log w + \sqrt{\log n} \cdot \log\frac{1}{\varepsilon}\right),$$

which establishes Theorem 1.3.

Space complexity analysis

For completeness, we give a naive analysis of the generator's space complexity.

Claim 5.4. *Assume the PRG given in Assumption 1 uses space $S(n, w, s, \varepsilon)$, where $s = \log|\Sigma|$. Then, the PRG of Theorem 1.3, with seed length d, can be computed in space*

$$O(\log\log|\Sigma| + \log^{3/2} n) + \sqrt{\log n} \cdot O\Big(S\Big(2^{\sqrt{\log n}}, w, d, \varepsilon/n\Big)\Big).$$

In particular, if, say, $S(n, w, s, \varepsilon) = O(\log^2 n + \log(w/\varepsilon)) + \mathrm{polylog}(s)$, the space complexity of the PRG of Theorem 1.3 is linear in its seed, i.e., $O(d)$.

Proof. At each iteration $i \in [t]$, given $z \in \{0,1\}^{d(i)}$ and $j \in [R^i]$, determining the j-th bit of $G_i(z)$ amounts to computing $\sigma = H_i(z)_t$ for some $t = t(j)$, and $G_{i-1}(\sigma)$. Note that t is computable in space $O(i \log R)$. Letting $\mathsf{space}(i)$ denote the space complexity of G_i, we can use space-efficient composition (see, e.g., [Gol08, Chapter 5]) and get that

$$\begin{aligned}\mathsf{space}(i) &= O(S(R, w, \log|\Sigma_i|, \varepsilon_H)) + \mathsf{space}(i-1) + O(i \log R) \\ &= \mathsf{space}(i-1) + O\Big(i\sqrt{\log n} + S\Big(2^{\sqrt{\log n}}, w, d(i-1), \varepsilon/n\Big)\Big),\end{aligned}$$

where $\mathsf{space}(0) = O(\log\log|\Sigma|)$. Recalling that $d(i) = O(i\log(n/\varepsilon) + i\sqrt{\log n}\log w)$, the claim follows. □

A low-error WPRG

We can now use the (hypothetical) PRG from Theorem 1.3 as a "base PRG" for the error reduction framework and "re-liberate" ε, at the expense of getting a WPRG rather than a PRG. As discussed in the introduction, the framework in [PV21, CDR+21] gives us a WPRG with seed length $\log|\Sigma| + O(\log^{3/2} n + \log n \cdot \log w) + \widetilde{O}(\log(w/\varepsilon))$. Our current work lets us get rid of the doubly-logarithmic factors in $1/\varepsilon$ while keeping the good dependence on Σ. In particular, Theorem 1.1 together with Theorem 1.3 gives us the following result.

Corollary 5.5. *Assuming Assumption 1, for any $n, w \in \mathbb{N}$, alphabet Σ, and error parameter $\varepsilon > 0$, there exists an explicit $(n, w, \Sigma, \Gamma, \varepsilon)$ WPRG with seed length*

$$d = \log|\Sigma| + O\left(\log^{3/2} n + \log n \cdot \log w + \log\frac{1}{\varepsilon} + \log\log\left(\frac{\log w}{\log n}\right) \cdot \log w\right).$$

This matches, up to triple-logarithmic factor, the state-of-the-art *space complexity* of (non black-box) algorithms for estimating the acceptable probability of (n, w, Σ) BPs [PP23, CDSTS23].

WPRGs with bounded weights

We note (and skip the rather technical proof), that Theorem 1.3 also extends to the case that the PRG family of Assumption 1 is a family of *WPRGs* with weights bounded by 1 in absolute value. However, such a WPRG readily yields a PRG, as we now show.

Lemma 5.6. *Let $W = (G, \mu)$ be an $(n, w, \Sigma, \Gamma, \varepsilon)$ WPRG against BPs with $\Gamma = 1$ (that is, $\mu(x) \in [-1, 1]$ for every seed x). Then, it also holds that G is an $(n, w, \Sigma, 2\varepsilon)$ PRG against BPs.*

Proof. Let s denote the seed length of W (and of G). First, consider the trivial width-1 BP T that always accepts. The WPRG W fools T, so

$$|\mathbb{E}_{x\sim U_s}[\mu(x) \cdot T(G(x))] - \mathbb{E}_{x\sim\Sigma^n}[T(x)]| = |\mathbb{E}_{x\sim U_s}[\mu(x)] - 1| \le \varepsilon,$$

and thus $\mathbb{E}[\mu(x)] \in [1-\varepsilon, 1+\varepsilon]$.

Next, fix any (n, w, Σ) BP B, and let $p = \mathbb{E}_{x\sim\Sigma^n}[B(x)]$. On the one hand,

$$\begin{aligned}\mathbb{E}_{x\sim U_s}[B(G(x))] &= \mathbb{E}_{x\sim U_s}[\mu(x)B(G(x))] + \mathbb{E}_{x\sim U_s}[(1-\mu(x))B(G(x))] \\ &\leq \mathbb{E}_{x\sim U_s}[\mu(x)B(G(x))] + \mathbb{E}_{x\sim U_s}[(1-\mu(x))] \leq (p+\varepsilon)+\varepsilon = p+2\varepsilon,\end{aligned}$$

where in the first inequality we used the fact that $1-\mu(x) \geq 0$. Similarly, on the other hand, as $\mu(x) \leq 1$ and $B(G(x)) \geq 0$ for all x,

$$\mathbb{E}_{x\sim U_s}[B(G(x))] \geq \mathbb{E}_{x\sim U_s}[\mu(x)B(G(x))] \geq p-\varepsilon,$$

as desired. □

Acknowledgment

We thank William Hoza for bringing to our attention Lemma 4.9 in [CH22], which already proves one of the theorems stated in an earlier version of this paper.

References

[AKL+79] Romas Aleliunas, Richard M. Karp, Richard J. Lipton, Laszlo Lovasz, and Charles Rackoff. Random walks, universal traversal sequences, and the complexity of maze problems. In *Annual Symposium on Foundations of Computer (FOCS)*, pages 218–223. IEEE, 1979.

[Arm98] Roy Armoni. On the derandomization of space-bounded computations. In *International Workshop on Randomization and Approximation Techniques in Computer Science (RANDOM)*, volume 1518 of *LNCS*, pages 47–59. Springer, 1998.

[BCG20] Mark Braverman, Gil Cohen, and Sumegha Garg. Pseudorandom pseudo-distributions with near-optimal error for read-once branching programs. *SIAM Journal on Computing*, 49(5):STOC18–242–STOC18–299, 2020.

[BCP83] Allan Borodin, Stephen Cook, and Nicholas Pippenger. Parallel computation for well-endowed rings and space-bounded probabilistic machines. *Information and Control*, 58(1-3):113–136, 1983.

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

[CDM+25] Gil Cohen, Dean Doron, Tomer Manket, Edward Pyne, Yichuan Wang, and Tal Yankovitz. A study of error reduction polynomials. In *Electronic Colloquium on Computational Complexity (ECCC)*, 2025. Manuscript.

[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 *Computational Complexity Conference (CCC)*, pages 22:1–22:17. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2021.

[CDSTS23] Gil Cohen, Dean Doron, Ori Sberlo, and Amnon Ta-Shma. Approximating iterated multiplication of stochastic matrices in small space. In *Annual Symposium on Theory of Computing (STOC)*, pages 35–45. ACM, 2023.

[CH22] Kuan Cheng and William M. Hoza. Hitting sets give two-sided derandomization of small space. *Theory of Computing*, 18(21):1–32, 2022.

[CHL+23] Lijie Chen, William M. Hoza, Xin Lyu, Avishay Tal, and Hongxun Wu. Weighted pseudorandom generators via inverse analysis of random walks and shortcutting. In *Annual Symposium on Foundations of Computer Science (FOCS)*, pages 1224–1239. IEEE, 2023.

[CL20] Eshan Chattopadhyay and Jyun-Jie Liao. Optimal error pseudodistributions for read-once branching programs. In *Computational Complexity Conference (CCC)*, pages 25:1–25:27. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2020.

[CT25] Ben Chen and Amnon Ta-Shma. Simplifying Armoni's PRG. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM)*, pages 36:1–36:8. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2025.

[CW24] Kuan Cheng and Yichuan Wang. $\mathbf{BPL} \subseteq \mathbf{L} - \mathbf{AC}^1$. In *Computational Complexity Conference (CCC)*, pages 32:1–32:14. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2024.

[CW26] Kuan Cheng and Ruiyang Wu. Weighted pseudorandom generators for read-once branching programs via weighted pseudorandom reductions. In *Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 3423–3458. SIAM, 2026.

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

[DHH19] Dean Doron, Pooya Hatami, and William M. Hoza. Near-optimal pseudorandom generators for constant-depth read-once formulas. In *34th Computational Complexity Conference (CCC 2019)*, pages 16:1–16:34. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2019.

[DMR+21] Dean Doron, Raghu Meka, Omer Reingold, Avishay Tal, and Salil Vadhan. Pseudorandom generators for read-once monotone branching programs. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM)*, pages 58:1–58:21. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2021.

[DPT24] Dean Doron, Edward Pyne, and Roei Tell. Opening up the distinguisher: A hardness to randomness approach $\mathbf{BPL} = \mathbf{L}$ that uses properties of $\mathbf{BPL}$. In *Annual Symposium on Theory of Computing (STOC)*, pages 2039–2049. ACM, 2024.

[DT23] Dean Doron and Roei Tell. Derandomization with minimal memory footprint. In *Computational Complexity Conference (CCC)*, pages 11–1. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2023.

[Gol08] Oded Goldreich. *Computational Complexity: A Conceptual Perspective*. Cambridge University Press, 2008.

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

[Hoz21] William M. Hoza. Better pseudodistributions and derandomization for space-bounded computation. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM)*, pages 28:1–28:23. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2021.

[HPV21] William M. Hoza, Edward Pyne, and Salil Vadhan. Pseudorandom generators for unbounded-width permutation branching programs. In *Innovations in Theoretical Computer Science Conference (ITCS)*, 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 $\mathbf{RL}$. *SIAM Journal on Computing*, 49(4):811–820, 2020.

[INW94] Russell Impagliazzo, Noam Nisan, and Avi Wigderson. Pseudorandomness for network algorithms. In *Annual Symposium on Theory of Computing (STOC)*, pages 356–364. ACM, 1994.

[Jun81] H. Jung. Relationships between probabilistic and deterministic tape complexity. In *Mathematical Foundations of Computer Science (MFCS)*, volume 118 of *LNCS*, pages 339–346. Springer, 1981.

[KNP11] Michal Koucký, Prajakta Nimbhorkar, and Pavel Pudlák. Pseudorandom generators for group products. In *Annual Symposium on Theory of Computing (STOC)*, pages 263–272. ACM, 2011.

[KNW08] Daniel M. Kane, Jelani Nelson, and David P. Woodruff. Revisiting norm estimation in data streams. *arXiv preprint arXiv:0811.3648*, 2008.

[Koz20] Daniel Kozlov. Simplifying the BCG PRPD for space bounded computation, 2020. Available at https://www.cs.tau.ac.il/~amnon/Students/daniel.kozlov.pdf.

[KvM02] Adam R. Klivans and Dieter van Melkebeek. Graph nonisomorphism has subexponential size proofs unless the polynomial-time hierarchy collapses. *SIAM Journal on Computing*, 31(5):1501–1526, 2002.

[MRSV17] Jack Murtagh, Omer Reingold, Aaron Sidford, and Salil Vadhan. Derandomization beyond connectivity: Undirected laplacian systems in nearly logarithmic space. In *Annual Symposium on the Foundations of Computer Science (FOCS)*, pages 801–812. IEEE, 2017.

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

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

[PP23] Aaron Putterman and Edward Pyne. Near-optimal derandomization of medium-width branching programs. In *Annual Symposium on Theory of Computing (STOC)*, pages 23–34. ACM, 2023.

[PV21] Edward Pyne and Salil Vadhan. Pseudodistributions that beat all pseudorandom generators. In *Computational Complexity Conference (CCC)*, pages 33:1–33:15. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2021.

[RR99] Ran Raz and Omer Reingold. On recycling the randomness of states in space bounded computation. In *31st Annual Symposium on Theory of Computing (STOC 1999)*, pages 159–168. ACM, 1999.

[RVW02] Omer Reingold, Salil Vadhan, and Avi Wigderson. Entropy waves, the zig-zag graph product, and new constant-degree expanders. *Annals of Mathematics*, pages 157–187, 2002.

[Sav70] Walter J. Savitch. Relationships between nondeterministic and deterministic tape complexities. *Journal of Computer and System Sciences*, 4(2):177–192, 1970.

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

ECCC ISSN 1433-8092

https://eccc.weizmann.ac.il