Optimal Error Pseudodistributions for Read-Once Branching Programs

Eshan Chattopadhyay
Department of Computer Science, Cornell University, Ithaca, NY, USA
eshanc@cornell.edu

Jyun-Jie Liao
Department of Computer Science, Cornell University, Ithaca, NY, USA
jjliao@cs.cornell.edu

Abstract

In a seminal work, Nisan (Combinatorica'92) constructed a pseudorandom generator for length n and width w read-once branching programs with seed length $O(\log n \cdot \log(nw) + \log n \cdot \log(1/\varepsilon))$ and error ε. It remains a central question to reduce the seed length to $O(\log(nw/\varepsilon))$, which would prove that $\mathbf{BPL} = \mathbf{L}$. However, there has been no improvement on Nisan's construction for the case $n = w$, which is most relevant to space-bounded derandomization.

Recently, in a beautiful work, Braverman, Cohen and Garg (STOC'18) introduced the notion of a *pseudorandom pseudo-distribution* (PRPD) and gave an explicit construction of a PRPD with seed length $\tilde{O}(\log n \cdot \log(nw) + \log(1/\varepsilon))$. A PRPD is a relaxation of a pseudorandom generator, which suffices for derandomizing $\mathbf{BPL}$ and also implies a hitting set. Unfortunately, their construction is quite involved and complicated. Hoza and Zuckerman (FOCS'18) later constructed a much simpler hitting set generator with seed length $O(\log n \cdot \log(nw) + \log(1/\varepsilon))$, but their techniques are restricted to hitting sets.

In this work, we construct a PRPD with seed length

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

This improves upon the construction by Braverman, Cogen and Garg by a $O(\log\log(1/\varepsilon))$ factor, and is optimal in the small error regime. In addition, we believe our construction and analysis to be simpler than the work of Braverman, Cohen and Garg.

2012 ACM Subject Classification Theory of computation → Pseudorandomness and derandomization

Keywords and phrases Derandomization, explicit constructions, space-bounded computation

Digital Object Identifier 10.4230/LIPIcs.CCC.2020.25

Funding *Eshan Chattopadhyay*: Supported by NSF grant CCF-1849899.
Jyun-Jie Liao: Supported by NSF grant CCF-1849899.

Acknowledgements We thank anonymous reviewers for helpful comments.

1 Introduction

A major challenge in computational complexity is to understand to what extent randomness is useful for efficient computation. It is widely believed that randomness does not provide substantial savings in time and space for algorithms. Indeed, under plausible assumption, every randomized algorithm for decision problem can be made deterministic with only a polynomial factor slowdown in time ($\mathbf{BPP} = \mathbf{P}$) [16] or a constant factor blowup in space ($\mathbf{BPL} = \mathbf{L}$) [20].

However, it remains open for decades to prove these results unconditionally. For derandomization in the time-bounded setting, it is known that proving $\mathbf{BPP} = \mathbf{P}$ implies circuit lower bounds which seem much beyond reach with current proof techniques [18]. However no

© Eshan Chattopadhyay and Jyun-Jie Liao;
licensed under Creative Commons License CC-BY

such implications are known for the space-bounded setting, and there has been some progress. Savitch's theorem [30] implies that $\mathbf{RL} \subseteq \mathbf{L}^2$. Borodin, Cook, Pippenger [3] and Jung [17] proved that $\mathbf{PL} \subseteq \mathbf{L}^2$, which implies $\mathbf{BPL} \subseteq \mathbf{L}^2$. Nisan [23, 24] constructed a pseudorandom generator for log-space computation with seed length $O(\log^2 n)$, and used it to show that $\mathbf{BPL}$ can be simulated with $O(\log^2 n)$ space and polynomial time. Saks and Zhou [29] used Nisan's generator in a non-trivial way to show that $\mathbf{BPL} \subseteq \mathbf{L}^{3/2}$, which remains the best known result so far. We refer the interested reader to the beautiful survey by Saks [28] for more background and relevant prior work.

We introduce the notion of a read-once branching programs, which is a non-uniform model for capturing algorithms that use limited memory.

▶ **Definition 1** (Read-once branching program)**.** *A (n, w)-read-once branching program (ROBP) B is a directed graph on the vertex set $V = \bigcup_{i=0}^{n} V_i$, where each set V_i contains w nodes. Every edge in this directed graph is labeled either 0 or 1. For every $i < n$, and every node $v \in V_i$, there exists exactly two edges starting from v, one with label 0 and the other with label 1. Every edge starting from a node in V_i connects to a node in V_{i+1}. We say n is the* length *of B, w is the* width *of B and V_i is the i-th* layer *of B.*

Moreover, there exists exactly one starting state $s \in V_0$, and exactly one accepting state $t \in V_n$. For every $x = (x_1, \ldots, x_n) \in \{0,1\}^n$, we define $B(x) = 1$ if starting from s we will reach t following the edges labeled by $x_1, \ldots, x_n$. Otherwise we define $B(x) = 0$.

It is well-known the computation of a probabilistic Turing machine that uses space S and tosses n coins, on a given input y, can be carried out by a $(n, 2^{O(S)})$-ROBP B_y. In particular, if the string $x \in \{0,1\}^n$ corresponds to the n coin tosses, then $B_y(x)$ is the output of the Turing machine.

A standard derandomization technique is via *pseudorandom generators*. We define this notion for the class of ROBPs.

▶ **Definition 2** (Pseudorandom generator)**.** *A function $G : \{0,1\}^s \to \{0,1\}^n$ is a (n, w, ε)-pseudorandom generator (PRG) if for every (n, w)-ROBP B,*

$$\left| \mathop{\mathbb{E}}_{x \in \{0,1\}^n} [B(x)] - \mathop{\mathbb{E}}_{r \in \{0,1\}^s} [B(G(r))] \right| \leq \varepsilon.$$

The seed length *of G is s. G is explicit if G is computable in $O(s)$ space.*

To derandomimze space-bounded computation given an explicit (n, w, ε)-PRG, one can enumerate $B(G(r))$ for every $r \in \{0,1\}^s$ with $O(s)$ additional space to compute an ε-approximation of the quantity $\mathbb{E}_x [B(x)]$.

Nisan [23] constructed a (n, w, ε)-PRG with seed length $O(\log n \cdot \log(nw/\varepsilon))$, which implies $\mathbf{BPL} \subseteq \mathbf{L}^2$. While there is a lot of progress in constructing PRG with better seed length for restricted family of ROBP (see, e.g., [25, 1, 6, 2, 5, 21, 9, 31, 22] and references therein), Nisan's generator and its variants [23, 15, 26] remain the best-known generators in the general case.

1.1 Pseudorandom pseudodistribution

Recently, a beautiful work of Braverman, Cohen and Garg [4] introduced the notion of a *pseudorandom pseudodistribution* (PRPD) that relaxes the definition of a PRG.

▶ **Definition 3** (Pseudorandom pseudodistribution). *A pair of functions* $(G, \rho) : \{0,1\}^s \to \{0,1\}^n \times \mathbb{R}$ *generates a* (n, w, ε)*-pseudorandom pseudodistribution (PRPD) if for every* (n, w)*-ROBP* B*,*

$$\left| \mathop{\mathbb{E}}_{x \in \{0,1\}^n} [B(x)] - \mathop{\mathbb{E}}_{r \in \{0,1\}^s} [\rho(r) \cdot B(G(r))] \right| \leq \varepsilon.$$

We say s *is the* seed length *of* (G, ρ)*. We say* (G, ρ) *is* k*-bounded if* $|\rho(x)| \leq k$ *for every* $x \in \{0,1\}^s$*. We say* (G, ρ) *is explicit if they are computable in space* $O(s)$*.*

Note that a (n, w, ε)-PRG G of seed length s with a constant function $\rho(x) = 1$ generates a 1-bounded (n, w, ε)-PRPD. Similar to a PRG, it is possible to derandomize **BPL** by enumerating all seeds of a PRPD and computing an ε-approximation for $\mathbb{E}_x [B(x)]$. In [4] they observe that given (G, ρ) which generates an (n, w, ε)-PRPD, the function G itself is an ε-hitting set generator for (n, w)-ROBP.

The main result in [4] is an explicit construction of a (n, w, ε)-PRPD with seed length

$$O\left((\log n \cdot \log(nw) + \log(1/\varepsilon)) \cdot \log\log(nw/\varepsilon)\right),$$

which is $\mathrm{poly}(nw/\varepsilon)$-bounded.[1] This improves on the seed-length of Nisan's generator and provides near optimal dependence on error.

Unfortunately, the construction and analysis in [4] is highly complicated. Hoza and Zuckerman [13] provided a dramatically simpler hitting set generator with slightly improved seed length. However, it is not clear how to extend their techniques for constructing a PRPD (or PRG).

1.2 Main result

In this paper, we construct a PRPD with optimal dependence on error (up to constants).

▶ **Theorem 4.** *There exists an explicit* (n, w, ε)*-PRPD generator* (G, ρ) *with seed length*

$$O\left(\log n \cdot \log(nw) \cdot \log\log(nw) + \log(1/\varepsilon)\right),$$

which is $\mathrm{poly}(1/\varepsilon)$*-bounded.*

This improves upon the construction in [4] by a factor of $O(\log\log(1/\varepsilon))$, for any $\varepsilon < n^{-\Omega(\log(nw)\log\log(nw))}$.

As observed in [4], the small-error regime is well motivated for application to derandomizing space-bounded computation. In particular, Saks and Zhou [29] instantiated Nisan's PRG with error $n^{-\omega(1)}$ to obtain the result $\mathbf{BPL} \subseteq \mathbf{L}^{3/2}$. We note that one can replace the PRG in the Saks-Zhou scheme with a PRPD which is $\mathrm{poly}(w, 1/\epsilon)$-bounded, and hence improvements to our result will lead to improved derandomization of **BPL**. We sketch a proof in Appendix A.

Our construction uses a strategy similar to [4] with the following key differences.

[1] Note that in [4], they define $\sum_r \rho(r) B(G(r))$ to be the approximation of $\mathbb{E}_x [B(x)]$. Here we define $\mathbb{E}_r [\rho(r) B(G(r))]$ to be the approximation instead to emphasize the possible loss when plugged into the Saks-Zhou scheme. (See Appendix A for more details.) Therefore a k-bounded PRPD in their definition is actually $2^s k$-bounded in our definition. Nevertheless, it is possible to show that their construction is still $\mathrm{poly}(nw/\varepsilon)$-bounded with our definition.

- The construction in [4] has a more *bottom-up* nature: their construction follows the binary tree structure in Nisan's generator [23], but in each node they maintain a sophisticated "leveled matrix representation" (LMR) which consists of many pieces of small-norm matrices, and they show how to combine pieces in two LMRs one by one to form a LMR in the upper level. Our construction follows the binary tree structure in Nisan's generator, but has a more *top-down* spirit. We give a clean recursive formula which generates a "robust PRPD" for (n, w)-PRPD given robust PRPDs for $(n/2, w)$-ROBP, where a robust PRPD is a family of pseudodistributions such that the approximation error of pseudodistribution drawn from this family is small on average. (A formal definition can be found in Definition 32.) The top-down nature of our construction significantly simplifies the construction and analysis.
- Following [4], we use an averaging sampler in our recursive construction, but we further observe that we can apply a simple "flattening" operation to limit the growth of seed length. With this observation, we not only improve the seed length but also simplify the construction and analysis by avoiding some special case treatments that are necessary in [4]. (Specifically, we do not need the special multiplication rule "outer product" in [4].)

Independent work

Independent work of Cheng and Hoza [7] remarkably prove that a hitting set generator (HSG) for ROBPs can be used for derandomizing **BPL**. Their first result shows that every (n, w)-ROBP f can be deterministically approximated within error ε with an explicit HSG for $(\mathrm{poly}(\frac{nw}{\varepsilon}), \mathrm{poly}(\frac{nw}{\varepsilon}))$-ROBP with seed length s. The space complexity of their first derandomization is $O(s + \log(nw/\varepsilon))$. Their second result shows that every (n, w)-ROBP f can be deterministically approximated within error ε with an explicit HSG for $(n, \mathrm{poly}(w))$-ROBP with seed length s. Their second derandomization has space complexity $O(s + w \log(n/\varepsilon))$, and only requires black-box access to f.

Their first result does not imply better derandomization algorithms with the state-of-art HSGs so far. Plugging in the HSG from [13], their second result gives a black-box derandomization algorithm for (n, w)-ROBP in space $O(\log(n) \log(nw) + w \log(n/\varepsilon))$. This is better than the black-box derandomization with our PRPD for the restricted case of $w = O(1)$. We note that an advantage of PRPDs (over hitting sets) is that they are applicable in the Saks and Zhou's scheme [29] (as mentioned in Appendix A, when applied with Armoni's sampler trick [1]).

Organization

In Section 2, we present the matrix representation of ROBPs, see how a pseudodistribution can be interpreted as matrices, and introduce some basic rules for translating between matrix operations and operations on pseudodistribution. We use Section 3 to present an outline of our main construction and proof. Section 4 contains necessary preliminaries. In Section 5, we formally prove several lemmas about using samplers on approximate matrix multiplication. In Section 6, we present and prove correctness of our main construction. We conclude with possible future directions in Section 7.

2 ROBPs and Matrices

We introduce the matrix representation of ROBPs and some related definitions that are useful in the rest of the paper. First, we setup some notation.

Notation: Given two strings x, y, we use $x\|y$ to denote the concatenation of x and y. For every $n \in \mathbb{N}$, we use $[n]$ to denote the set $\{1, 2, \ldots, n\}$. We denote a collection of objects A_i^j with subscript $i \in S$ and superscript $j \in T$ by $[A]_S^T$ for short.

Given a (n, w)-ROBP B with layers $V_0, \ldots, V_n$, we can represent the transition from layer V_{t-1} to V_t by two stochastic matrices M_t^0 and M_t^1 as follows: suppose layer V_j consists of the nodes $\{v_{j,1}, \ldots, v_{j,w}\}$. The entry $(M_t^0)_{i,j} = 1$ if and only if there exist a 0-labeled edge from $v_{t-1,i}$ to $v_{t,j}$ (else $(M_t^0)_{i,j} = 0$). The matrix M_t^1 is defined similarly according to the edges that labeled 1 between layers V_{t-1} and V_t. More generally, we can also represents multi-step transition by a stochastic matrix. That is, for every $0 \le a \le b \le n$, and every $r = (r_{a+1}, \ldots, r_b) \in \{0,1\}^{b-a}$, we can define

$$M_{a..b}^r = \prod_{t=a+1}^{b} M_t^{r_t}$$

which corresponds to the transition matrix from layer a to layer b following the path labeled by r. Note that every row of $M_{a,b}^r$ contains exactly one 1, and the other entries are 0.

An n-step random walk starting from the first layer can be represented with the following matrix:

$$M_{0..n} = \frac{1}{2^n} \sum_{r \in \{0,1\}^n} M_{0..n}^r = \prod_{t=1}^{n} \frac{1}{2} \left(M_t^0 + M_t^1\right).$$

By definition of M_t^0, M_t^1 one can observe that the (i, j) entry of $M_{0..n}$ is the probability that a random walk from $v_{0,i} \in V_0$ reaches $v_{n,j} \in V_n$. Therefore, suppose $v_{0,i} \in V_0$ is the starting state of B, $v_{n,j} \in V_n$ is the accepting state of B, then $\mathbb{E}_x [B(x)]$ equals the (i, j) entry of $M_{0..n}$.

Recall that a generator of a (n, w, ε)-PRPD is a pair of function (G, ρ) such that for every (n, w)-ROBP B,

$$\left| \mathop{\mathbb{E}}_{r} [\rho(r) \cdot B(G(r))] - \mathop{\mathbb{E}}_{x \in \{0,1\}^n} [B(x)] \right| \le \varepsilon.$$

Equivalently, for every transition matrices $M_1^0, M_1^1, \ldots, M_n^0, M_n^1$, we have

$$\left\| \mathop{\mathbb{E}}_{r} \left[\rho(r) \cdot M_{0..n}^{G(r)} \right] - M_{0..n} \right\|_{\max} \le \varepsilon,$$

where $\|A\|_{\max}$ denotes $\max_{i,j} |A(i, j)|$.

Therefore it is natural to represents a PRPD (G, ρ) with a mapping $\mathcal{G} : \{0,1\}^s \to \mathbb{R}^{w \times w}$ where $\mathcal{G}(r) = \rho(r) \cdot M_{0..n}^{G(r)}$. More generally, we will use a notation similar to the "matrix bundle sequence" (MBS) introduced in [4] to represent a PRPD.

▶ **Definition 5.** *Consider a (n, w)-ROBP $[M]_{[n]}^{\{0,1\}}$ and a pair of functions $(G, \rho) : \{0,1\}^{s_{\text{out}}} \times [S_{\text{in}}] \to \{0,1\}^n \times \mathbb{R}$. The* matrix form *of (G, ρ) on $[M]_{[n]}^{\{0,1\}}$ is a mapping $\mathcal{A} : \{0,1\}^{s_{\text{out}}} \times [S_{\text{in}}] \to \mathbb{R}^{w \times w}$ such that for every $x \in \{0,1\}^{s_{\text{out}}}$ and $y \in [S_{\text{in}}]$,*

$$\mathcal{A}(x, y) = \rho(x, y) \cdot M_{0..n}^{G(x,y)}.$$

For every $x \in \{0,1\}^{s_{\text{out}}}$ we abuse the notation and define

$$\mathcal{A}(x) = \mathop{\mathbb{E}}_{y} [\mathcal{A}(x, y)].$$

Besides, we define $\langle \mathcal{A} \rangle = \mathbb{E}_{x,y} [\mathcal{A}(x, y)]$. We say s_{out} is the outer seed length *of $\mathcal{A}$, denoted by $s_{\text{out}}(\mathcal{A})$, and S_{in} is the* inner size *of $\mathcal{A}$, denoted by $S_{\text{in}}(\mathcal{A})$. We also define $s_{\text{in}}(\mathcal{A}) = \lceil \log S_{\text{in}} \rceil$ to be the* inner seed length *of $\mathcal{A}$, and $s(\mathcal{A}) = s_{\text{out}}(\mathcal{A}) + s_{\text{in}}(\mathcal{A})$ to be the* seed length *of $\mathcal{A}$.*

▶ Remark 6. For every fixed x, the collection $\{\mathcal{A}(x,y) : y \in [S_{\text{in}}]\}$ corresponds to the "matrix bundle" in [4]. This should be treated as a collection of matrices which "realizes" the matrix $\mathcal{A}(x)$. The whole structure $\mathcal{A}$ corresponds to the "matrix bundle sequence" in [4], and should be treated as a uniform distribution over the set $\{\mathcal{A}(x) : x \in \{0,1\}^{s_{\text{out}}}\}$.

When the ROBP $[M]_{[n]}^{\{0,1\}}$ is clear in the context, we will use the matrix form $\mathcal{A}$ to represent the pseudodistribution (G,ρ) directly. We will apply arithmetic operations on matrices $\mathcal{A}(x)$, and these operations can be easily translated back to operations on pseudodistributions as follows.

▶ **Definition 7.** *Consider a (n,w)-ROBP $[M]_{[n]}^{\{0,1\}}$, and a pair of function $(F,\sigma) : [S] \to \{0,1\}^n \times \mathbb{R}$. The matrix that is* realized *by (F,σ) on $M_{0..n}$ is $\mathbb{E}_{i\in[S]}\left[\sigma(i)\cdot M_{0..n}^{F(i)}\right]$. We say S is the* size *of (F,σ).*

Scaling the matrix corresponds to scaling the coefficients in the pseudodistribution.

▷ Claim 8. Consider a (n,w)-ROBP $[M]_{[n]}^{\{0,1\}}$, let A be a matrix realized by matrix bundle (F_A,σ_A) on $M_{0..n}$. Then cA is realized by a matrix bundle (F'_A,σ'_A) of size S_A s.t. $F'_A = F_A$ and $\sigma'_A(x) = c\sigma_A(x)$ for every $x \in [S]$.

The summation on matrices corresponds to re-weighting and union on pseudodistributions.

▷ Claim 9. Consider a (n,w)-ROBP $[M]_{[n]}^{\{0,1\}}$, let A be a matrix realized by matrix bundle (F_A,σ_A) of size S_A on $M_{0..n}$ and B be a matrix realized by matrix bundle (F_B,σ_B) of size S_B on $M_{0..n}$. Then $A+B$ is realized by a matrix bundle (F',σ') of size S_A+S_B on $M_{0..n}$ s.t.

$$F'(x) = \begin{cases} F_A(x) & \text{if } x \le S_A \\ F_B(x-S_A) & \text{if } x > S_A \end{cases} \text{ and } \sigma'(x) = \begin{cases} \frac{S_A+S_B}{S_B}\cdot\sigma_A(x) & \text{if } x \le S_A \\ \frac{S_A+S_B}{S_B}\cdot\sigma_B(x-S_A) & \text{if } x > S_A \end{cases}$$

The multiplication on matrices corresponds to concatenation of pseudodistributions.

▷ Claim 10. Consider a (n,w)-ROBP $[M]_{[n]}^{\{0,1\}}$, let A be a matrix realized by matrix bundle (F_A,σ_A) of size S_A on $M_{0..n/2}$ and B be a matrix realized by matrix bundle (F_B,σ_B) of size S_B on $M_{n/2..n}$. Fix a bijection $\pi : [S_A]\times[S_B] \to [S_A\cdot S_B]$. Then AB is realized by a matrix bundle (F',σ') of size $S_A\cdot S_B$ s.t. for every $a \in [S_A], b \in [S_B]$,

$$F'(\pi(a,b)) = F_A(a)\|F_B(b) \text{ and } \sigma'(\pi(a,b)) = \sigma(a)\cdot\sigma(b).$$

3 Proof Overview

In this section we give an outline of our construction and proof. In Section 3.1, we briefly recap how a sampler is used in [4] to achieve better seed length in the small-error regime. We discuss our construction ideas in Section 3.2.

3.1 The sampler argument

Nisan's generator and its variants recursively use a lemma of the following form.

▶ **Lemma 11.** *Consider a (n,w)-ROBP $[M]_{[n]}^{\{0,1\}}$. Let $\mathcal{A}$ be the matrix form of a distribution on $M_{0..n/2}$, and $\mathcal{B}$ be the matrix form of a distribution on $M_{n/2..n}$. Suppose $s(\mathcal{A}) = s(\mathcal{B}) = s$. Then there exists a distribution whose matrix form $\mathcal{C}$ on $M_{0..n}$ of seed length $s + O(\log(w/\delta))$ such that*

$$\|\langle\mathcal{C}\rangle - \langle\mathcal{A}\rangle\langle\mathcal{B}\rangle\|_{\max} \le \delta.$$

This lemma is usually achieved with a pseudorandom object. For example, the INW generator [15] uses a bipartite expander with degree $\mathrm{poly}(w/\delta)$ to construct the distribution $\mathcal{C}$ in the above lemma. That is, for every edge (x, y) in the expander G, they add $\mathcal{A}(x)\mathcal{B}(y)$ into $\mathcal{C}$. A similar lemma can also be obtained with universal hash functions [23] or seeded extractors [26]. By recursively constructing good approximations of $M_{0..n/2}$ and $M_{n/2..n}$ and applying Lemma 11, one can obtain a PRG which has seed length $O(\log n \cdot \log(nw/\varepsilon))$ (δ is taken to be ε/n because of a union bound). Observe that in such constructions, one needs to pay $O(\log(1/\varepsilon))$ (in seed length) per level of recursion.

The crucial idea in [4] is to amortize this cost over all $\log n$ levels. What makes this possible is the following argument, which we will refer to as the *sampler argument*. First we define the notion of an averaging sampler.

▶ **Definition 12.** *A function $g : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ is an (ε, δ)-(averaging) sampler if for every function $f : \{0,1\}^m \to [0,1]$,*

$$\Pr_{x\in\{0,1\}^n}\left[\left|\mathop{\mathbb{E}}_{s\in\{0,1\}^d}\left[f(g(x,s))\right] - \mathop{\mathbb{E}}_{y\in\{0,1\}^m}\left[f(y)\right]\right| \le \varepsilon\right] \ge 1-\delta.$$

The crucial observation in [4] is that if one uses a sampler to prove Lemma 11, the error actually scales with the norm of one of the matrix forms.

▶ **Lemma 13** ([4])**.** *Consider a (n, w)-ROBP with matrix representation $[M]_{[n]}^{\{0,1\}}$. Let $\mathcal{A}$ and $\mathcal{B}$ be (pseudo)distributions in matrix form on $M_{0..n/2}$ and $M_{n/2..n}$ respectively. Let $n = s_{\mathrm{out}}(\mathcal{A})$, $m = s_{\mathrm{out}}(\mathcal{B})$. Suppose $\forall x \in \{0,1\}^n, \|\mathcal{A}(x)\| \le 1$ and $\forall y \in \{0,1\}^m, \|\mathcal{B}(y)\| \le 1$. Let $g : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ be a (ε, δ) sampler. Then there exists a (pseudo)distribution $\mathcal{C}$ such that*

$$\|\langle C\rangle - \langle A\rangle \langle B\rangle\| \le O\left(w^2\left(\delta + \varepsilon \mathop{\mathbb{E}}_x\left[\|\mathcal{A}(x)\|\right]\right)\right).$$

Besides, $\mathcal{C}$ has outer seed length $n = s_{\mathrm{out}}(\mathcal{A})$, and for every $x \in \{0,1\}^n$,

$$\mathcal{C}(x) = \mathop{\mathbb{E}}_s\left[\mathcal{A}(x)\mathcal{B}\left(g(x,s)\right)\right].$$

Note that $s_{\mathrm{in}}(\mathcal{C}) = s_{\mathrm{in}}(\mathcal{A}) + s_{\mathrm{in}}(\mathcal{B}) + d$.

The intuition behind this approximation is as follows. If we want to compute the matrix product precisely, we take every $\mathcal{A}(x)$ and multiply it with $\mathbb{E}_y[\mathcal{B}(y)]$. However, with the help of sampler, we can use x as our seed to select some samples from $\mathcal{B}$, and take their average as an estimate of $\mathbb{E}_y[\mathcal{B}(y)]$. The error of this approximation comes in two different way. For those x which are not good choices of a seed for the sampler, the samples chosen with such an x can deviate from the average arbitrarily. However, only δ fraction of x can be bad, so they incur at most δ error. The second kind of error is the estimation error between average of samples $\mathbb{E}_s[\mathcal{B}(g(x,s))]$ and the real average $\mathbb{E}_y[\mathcal{B}(y)]$, which can be at most ε. Since this gets multiplied with $\mathcal{A}(x)$, this kind of error actually scales with $\|\mathcal{A}(x)\|$. Although the first kind of error (which is δ) does not benefit from $\|\mathcal{A}\|$ being small, in [4] they observe that, the parameter δ has almost no influence on the seed length in some cases. To discuss this more precisely, we first recall explicit constructions of samplers.

▶ **Lemma 14** ([27, 10])**.** *For every $\delta, \varepsilon > 0$ and integer m, there exists a space efficient (ε, δ)-sampler $f : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ s.t. $d = O(\log\log(1/\delta) + \log(1/\varepsilon))$ and $n = m + O(\log(1/\delta)) + O(\log(1/\varepsilon))$.*

Note that in Lemma 13, $s(\mathcal{C}) = s(\mathcal{A}) + d + s_{\mathrm{in}}(\mathcal{B})$. Therefore if $n \ge m + O(\log(1/\delta)) + O(\log(1/\varepsilon))$, δ has almost no impact on the seed length.

To use the above ideas, it boils down to working with matrices with small norm, and making sure that every multiplication is "unbalanced" enough so that δ has no impact. [4] applies a delicate telescoping sum trick (which they called "delta sampler") to divide an ε-approximation into a base approximation with $1/\mathrm{poly}(n)$ error and several "correcting terms" which have small norm. By carefully choosing the samplers and discarding all the non-necessary terms, they roughly ensure the following properties: first, a matrix with large seed length must have small norm; second, every matrix multiplication is unbalanced enough so that δ has no impact on the seed length.

With these properties and the sampler argument, they show that the total seed length is bounded by $\tilde{O}(\log(1/\varepsilon) + \log n \log(nw))$.

3.2 Our construction

In executing the ideas sketched above, the construction and analysis in [4] turns out to be quite complicated and involved. One thing which complicates the construction and analysis is its *bottom-up* nature. That is, when multiplying two terms, they create more terms with the telescoping sum trick. Moreover, in the telescoping sum trick one needs to choose the parameters of each sampler very carefully to make sure the seed length of each term does not exceed its "smallness".

Our first step toward a simpler construction is the following *top-down* formula, which we will apply recursively to compute an approximation of $M_{0..n}$:

▶ **Lemma 15.** *Let $\|\cdot\|$ be a sub-multiplicative matrix norm, and A, B be two matrices s.t. $\|A\|, \|B\| \le 1$. Let $k \in \mathbb{N}$ and $\gamma < 1$. For every $0 \le i \le k$, let A_i be a γ^{i+1}-approximation of A, and let B_i be a γ^{i+1}-approximation of B. Then*

$$\sum_{i=0}^{k} A_i B_{k-i} - \sum_{i=0}^{k-1} A_i B_{k-1-i}$$

is a $((k+2)\gamma^{k+1} + (k+1)\gamma^{k+2})$-approximation of AB.

Proof. We have,

$$\begin{aligned}
&\left\| (\sum_{i=0}^{k} A_i B_{k-i} - \sum_{i=0}^{k-1} A_i B_{k-1-i}) - AB \right\| \\
&= \left\| \sum_{i=0}^{k} (A - A_i)(B - B_{k-i}) - \sum_{i=0}^{k-1} (A - A_i)(B - B_{k-1-i}) + (A_k - A)B + A(B_k - B) \right\| \\
&\le \sum_{i=0}^{k} \|A - A_i\| \cdot \|B - B_{k-i}\| + \sum_{i=0}^{k-1} \|A - A_i\| \cdot \|B - B_{k-1-i}\| \\
&+ \|A_k - A\| \cdot \|B\| + \|A\| \cdot \|B_k - B\| \\
&\le (k+2)\gamma^{k+1} + (k+1)\gamma^{k+2}
\end{aligned}$$

◀

This formula shares an important property with the BCG construction: we never need a γ^k-approximation (which implies large seed length) on both sides simultaneously. The benefit of our top-down formula is that we are treating the PRPD as one object instead of the sum of many different terms. One obvious effect of such treatment is we don't need to analyze the "smallness" of each term and the accuracy of the whole PRPD separately.

In this top-down formula, we do not explicitly maintain small-norm matrices as in [4]. However, observe that in the proof of Lemma 15, we are using the fact that $A_k - A$ is a small norm matrix. Our goal is to apply the sampler argument (Lemma 13) on these "implicit" small-norm matrices. The following is our main technical lemma.

▶ **Lemma 16** (main lemma, informal). *Let $A, B \in \mathbb{R}^{w\times w}$, $k \in \mathbb{N}$ and $\gamma < 1$. Suppose for every $i \le k$ there exists pseudodistribution $\mathcal{A}_i, \mathcal{B}_i$ such that $\mathbb{E}_x\left[\|\mathcal{A}_i(x) - A\|\right] \le \gamma^{i+1}$, $\mathbb{E}_x\left[\|\mathcal{B}_i(x) - B\|\right] \le \gamma^{i+1}$, and $\|\mathcal{A}_i(x)\|, \|\mathcal{B}_i(x)\| \le 1$ for every x. Then there exists a pseudodistribution $\mathcal{C}_k$ such that*

$$\mathop{\mathbb{E}}_{x}\left[\|\mathcal{C}_k(x) - AB\| \le O(\gamma)^{k+1}\right],$$

where $\mathcal{C}_k(x) = \sum_{i+j=k} A_{x,i}B_{x,j} - \sum_{i+j=k-1} A_{x,i}B_{x,j}$. $A_{x,i}$ and $B_{x,i}$ are defined as follows.

- *If $i > \lceil k/2\rceil$, $A_{x,i} = \mathcal{A}_i(x)$ and $B_{x,i} = \mathcal{B}_i(x)$.*
- *If $i \le \lceil k/2\rceil$, $A_{x,i} = \mathbb{E}_s\left[\overline{\mathcal{A}_i}(g_i(x,s))\right]$ and $B_{x,i} = \mathbb{E}_s\left[\overline{\mathcal{B}_i}(g_i(x,s))\right]$, where g_i is a $(\gamma^{i+1}, \gamma^{k+1})$-sampler, and $\overline{\mathcal{A}_i}, \overline{\mathcal{B}_i}$ denote the "flattened" form of $\mathcal{A}_i$ and $\mathcal{B}_i$.*

We leave the explanation of "flattened" for later and explain the intuition behind the lemma first. Our goal is to construct $\mathcal{C}_k$ such that $\mathcal{C}_k(x)$ is a good approximation of AB *on average* over x. We know that $\mathcal{A}_i$ and $\mathcal{B}_i$ are γ^{i+1}-approximation of A and B on average. Our hope is to use x to draw samples A_i and B_i from $\mathcal{A}_i$ and $\mathcal{B}_i$, and apply the formula in Lemma 15 to get a good approximation of AB. In particular, a natural choice would be setting $A_{x,i} = \mathcal{A}_i(x)$ and $B_{x,i} = \mathcal{B}_i(x)$ for every $i \le k$. However, if there exists a term $A_{x,i}B_{x,j}$ such that $A_{x,i}$ and $B_{x,j}$ are both bad approximation for a large enough fraction of x, we cannot guarantee to get a $O(\gamma^{k+1})$-approximation on average.

To avoid the above case, for every $i \le \lceil k/2\rceil$ we use a sampler to approximate $\langle\mathcal{A}_i\rangle$ and $\langle\mathcal{B}_i\rangle$. This ensure that the chosen samples $A_{x,i}$ and $B_{x,i}$ are good *with high probability*. This guarantees that in each term $A_{x,i}B_{x,j}$, at least one of $A_{x,i}$ or $B_{x,j}$ will be a good choice with high probability over x. If $A_{x,i}$ is a good choice with high probability, we can apply the average-case guarantee on $B_{x,i}$ to get an average-case guarantee for $\mathcal{C}_k$, and vice versa. (Indeed, this is the sampler argument.) Therefore we can ensure that $\mathcal{C}_k(x)$ is good on average. Note that we only apply a sampler on $\mathcal{A}_i$ (or $\mathcal{B}_i$) when $i \le \lceil k/2\rceil$, which means $\mathcal{A}_i$ (or $\mathcal{B}_i$) has small seed length. Therefore we don't need to add too much redundant seed to make the sampler argument work.

In executing the above sketched idea, we run into the following problem: in each multiplication, the inner seed on both sides aggregates to the upper level. If we start with pseudodistributions with non-zero inner seed in the bottom level, the inner seed would become $\Omega(n)$ in the topmost level. Therefore we need a way to limit the aggregation of inner seed.

In [4], they run into a similar problem. To deal with this, they apply a different multiplication rule, "outer product", in some special cases to deal with this. However, the outer product does not seem applicable in our construction. Nevertheless, we observe that whenever we use a sampler to select matrix $A_{x,i}$, we only care about whether $\langle\mathcal{A}_i\rangle$ is close to A, and we don't need most of $\mathcal{A}_i(x)$ to be close to A anymore. Therefore we will "flatten" $\mathcal{A}_i$ whenever we apply a sampler. That is, recall that each $\mathcal{A}_i(x)$ is realized by the average of some matrices, $\mathbb{E}_y\left[\mathcal{A}_i(x,y)\right]$. We define the flattened form of $\mathcal{A}_i$, denoted by $\overline{\mathcal{A}_i}$, such that $\overline{\mathcal{A}_i}(x\|y) = \mathcal{A}_i(x,y)$. Observe that $\langle\overline{\mathcal{A}_i}\rangle = \langle\mathcal{A}_i\rangle$ and $s_{\text{in}}(\overline{\mathcal{A}_i}) = 0$. This guarantees that the inner seed length of $\mathcal{A}_i$ will not aggregate in $\mathcal{C}_k$. Moreover, while the flattening will increase the outer seed length of $\overline{\mathcal{A}_i}$, this is almost for free since we only flatten $\mathcal{A}_i$ when $i \le \lceil k/2\rceil$, i.e. when $\mathcal{A}_i$ has relatively small seed length. As a result, this operation also helps us save a $O(\log\log(1/\varepsilon))$ factor in the seed length.

We conclude by briefly discussing the seed length analysis. First note that we set $\gamma = 1/\mathrm{poly}(n)$ to make sure that the error is affordable after a union bound. Now consider the inner seed length. Consider a term A_iB_j such that $i \geq j$. In this term, part of the inner seed of $\mathcal{C}$ is passed to $\mathcal{A}_i$, and the other is used for the sampler on B_j. Since the seed length of the sampler only needs to be as large as the "precision gap" between $\mathcal{A}_i$ and $\mathcal{C}_k$, the inner seed length of $\mathcal{C}_k$ can be maintained at roughly $O(k\log(1/\gamma)) = O(\log(1/\varepsilon))$. However, after each multiplication, there's actually a $O(\log(nw/\gamma)) = O(\log(nw))$ additive overhead. Note that this is necessary since the $k = 0$ case degenerates to the INW generator. Therefore after $\log n$ levels of recursion, the inner seed length will be $O(\log(1/\varepsilon) + \log n \cdot \log(nw))$.

Besides, we also need the outer seed length of $\mathcal{C}_k$ to be long enough so that we can apply a sampler on $\mathcal{A}_{\lceil k/2\rceil}$ and $\mathcal{B}_{\lceil k/2\rceil}$. The seed length caused by approximation accuracy ε can be bounded similarly as the inner seed length. However, the $O(\log n \cdot \log(nw))$ inner seed length will be added to the outer seed length several times, because of the flattening operation. Nevertheless, since we only do flattening for $\mathcal{A}_i$ and $\mathcal{B}_i$ where $i \leq \lceil k/2\rceil$, this ensures that the flattening operation happens at most $\log k$ times. So the total outer seed length will be bounded by $O(\log(1/\varepsilon) + \log k \cdot \log n \cdot \log(nw)) = O(\log(1/\varepsilon) + \log\log(1/\varepsilon) \cdot \log n \cdot \log(nw))$, which is bounded by $O(\log(1/\varepsilon) + \log\log(nw) \cdot \log n \cdot \log(nw))$ since $O(\log(1/\varepsilon))$ is the dominating term when $\log(1/\varepsilon) \geq \log^3(nw)$.

4 Preliminaries

4.1 Averaging samplers

▶ **Definition 17.** *A function $g : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ is a (ε, δ)* (averaging) sampler *if for every function $f : \{0,1\}^m \to [0,1]$,*

$$\Pr_{x\in\{0,1\}^n}\left[\left|\mathop{\mathbb{E}}_{s\in\{0,1\}^d}[f(g(x,s))] - \mathop{\mathbb{E}}_{y\in\{0,1\}^m}[f(y)]\right| \leq \varepsilon\right] \geq 1-\delta.$$

It's easy to show that samplers also work for f with general range by scaling and shifting.

▷ Claim 18. Let $g : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ be a (ε,δ)-sampler, and let $\ell < r \in \mathbb{R}$. Then for every $f : \{0,1\}^m \to [\ell, r]$,

$$\Pr_{x\in\{0,1\}^n}\left[\left|\mathop{\mathbb{E}}_{s\in\{0,1\}^d}[f(g(x,s))] - \mathop{\mathbb{E}}_{y\in\{0,1\}^m}[f(y)]\right| \leq \varepsilon(r-\ell)\right] \geq 1-\delta.$$

Proof. Let f' be the function such that $f'(y) = (f(y)-\ell)/(r-\ell)$. Observe that the range of f' is in $[0,1]$. By definition of sampler,

$$\Pr_{x\in\{0,1\}^n}\left[\left|\mathop{\mathbb{E}}_{s\in\{0,1\}^d}[f'(g(x,s))] - \mathop{\mathbb{E}}_{y\in\{0,1\}^m}[f'(y)]\right| \leq \varepsilon\right] \geq 1-\delta.$$

By multiplying $(r-\ell)$ on both sides of the inequality inside the probability above we prove the claim. ◁

In our construction, we will use the following sampler which is explicitly computable with small space.

▶ **Lemma 19** ([27, 10])**.** *For every $\delta, \varepsilon > 0$ and integer m, there exists a (ε,δ)-sampler $f : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ s.t. $d = O(\log\log(1/\delta) + \log(1/\varepsilon))$ and $n = m + O(\log(1/\delta)) + O(\log(1/\varepsilon))$. Moreover, for every x, y, $f(x,y)$ can be computed in space $O(m + \log(1/\delta) + \log(1/\varepsilon))$.*

▶ Remark 20. The original sampler in [27] has a restriction on ε. Such a restriction will cause a $2^{O(\log^*(nw/\varepsilon))}$ factor in our construction, as in [4]. However, [27] pointed out that the restriction is inherited from the extractor in [33], which breaks down when the error is extremely small. As observed in [10], this restriction can be removed by plugging in a more recent extractor construction in [12]. Note that there exists a space-efficient implementation of [12] in [19], so the resulting sampler is also space-efficient. For completeness we include a proof in Appendix B.

4.2 Matrix norms

As in [4], we will use the infinity norm in this paper.

▶ **Definition 21.** *For every matrix* $A \in \mathbb{R}^{w\times w}$, $\|A\| = \max_i \sum_j |A_{i,j}|$.

We record some well known properties of the infinity norm.

▷ Claim 22. Let $A, B \in \mathbb{R}^{w\times w}$, $c \in \mathbb{R}$. Then

- $\|cA\| = |c|\,\|A\|$
- $\|A\| + \|B\| \le \|A + B\|$
- $\|AB\| \le \|A\|\,\|B\|$
- $\max_{i,j} |A_{i,j}| \le \|A\|$
- If A is stochastic, then $\|A\| = 1$

Note that for any (n, w)-ROBP represented by $w \times w$ matrices $M_{[n]}^{\{0,1\}}$, $\|M_{i..j}\| = 1$ for every $0 \le i \le j \le n$.

5 Approximate Matrix Multiplication via Samplers

In this section we formally prove the sampler arguments which will be used in our construction. Our proof strategy resembles that of [4], with the following two crucial differences. First, we will define two different notions of "smallness" for our flattening idea. Second, in our construction we need the case where we use samplers to select matrices on both sides (Lemma 27).

We will consider mappings $\mathcal{A} : \{0,1\}^n \to \mathbb{R}^{w\times w}$ which correspond to the implicit small norm matrices we discussed in the previous section. Borrowing notation from Definition 5, we use $\langle\mathcal{A}\rangle$ to denote $\mathbb{E}_x[\mathcal{A}(x)]$. First we define two different norms for the mapping $\mathcal{A}$. The *robust norm* is similar to the notion of "smallness" in [4], i.e. the average of norm of $\mathcal{A}(x)$, while the *norm* of $\mathcal{A}$ is simply the norm of $\langle\mathcal{A}\rangle$, i.e. the norm of average of $\mathcal{A}(x)$.

▶ **Definition 23.** *For every function* $\mathcal{A} : \{0,1\}^n \to \mathbb{R}^{w\times w}$, *we define the* norm *of* $\mathcal{A}$ *to be* $\|\mathcal{A}\| = \left\|\mathbb{E}_{x\in\{0,1\}^n}[\mathcal{A}(x)]\right\|$, *and the* robust norm *of* $\mathcal{A}$ *to be* $\|\mathcal{A}\|_r = \mathbb{E}_{x\in\{0,1\}^n}[\|\mathcal{A}(x)\|]$. *Besides, we define the weight of* $\mathcal{A}$ *to be* $\mu(\mathcal{A}) = \max_x \|\mathcal{A}(x)\|$.

▷ Claim 24. $\|\mathcal{A}\| \le \|\mathcal{A}\|_r \le \mu(\mathcal{A})$.

Proof. $\|\mathcal{A}\| \le \|\mathcal{A}\|_r$ is by sub-additivity of $\|\cdot\|$, and $\|\mathcal{A}\|_r \le \mu(\mathcal{A})$ since $\|\mathcal{A}\|_r$ is the average of values no larger than $\mu(\mathcal{A})$. ◁

Next we show a simple lemma which will be used later. That is, a sampler for functions with range $[0, 1]$ is also a sampler for matrix-valued functions, where the error is measured with infinity norm.

▶ **Lemma 25.** *For every function $\mathcal{A} : \{0,1\}^m \to \mathbb{R}^{w\times w}$ and every (ε,δ)-sampler $g : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$,*

$$\Pr_{x\in\{0,1\}^n}\left[\left\|\mathop{\mathbb{E}}_{s\in\{0,1\}^d}\left[\mathcal{A}(g(x,s))\right]-\langle\mathcal{A}\rangle\right\| \le 2w\mu(\mathcal{A})\varepsilon\right] \ge 1-w^2\delta.$$

Proof. Let $\mathcal{E}(y) = \mathcal{A}(y) - \langle\mathcal{A}\rangle$. For every $i,j\in[w]$, observe that

$$\max_y \mathcal{E}(y)_{i,j} - \min_y \mathcal{E}(y)_{i,j} = \max_y \mathcal{A}(y)_{i,j} - \min_y \mathcal{A}(y)_{i,j}$$

By the property of sampler it follows that

$$\Pr_{x\in\{0,1\}^n}\left[\left|\mathop{\mathbb{E}}_{s}\left[\mathcal{E}(g(x,s))_{i,j}\right]\right| \le 2\varepsilon\mu(\mathcal{A})\right] \ge 1-\delta.$$

Using a union bound,

$$\Pr_{x\in\{0,1\}^n}\left[\forall i,j\in[w], \left|\mathop{\mathbb{E}}_{s}\left[\mathcal{E}(g(x,s))_{i,j}\right]\right| \le 2\varepsilon\mu(\mathcal{A})\right] \ge 1-w^2\delta.$$

Thus by definition of the infinity norm, we can conclude that

$$\Pr_{x\in\{0,1\}^n}\left[\left\|\mathop{\mathbb{E}}_{s\in\{0,1\}^d}\left[\mathcal{E}(g(x,s))\right]\right\| \le 2w\mu(\mathcal{A})\varepsilon\right] \ge 1-w^2\delta.$$

which by sub-additivity of $\|\cdot\|$ implies

$$\Pr_{x\in\{0,1\}^n}\left[\left\|\mathop{\mathbb{E}}_{s}\left[\mathcal{A}(g(x,s))\right]\right\| \le \|\mathcal{A}\| + 2w\mu(\mathcal{A})\varepsilon\right] \ge 1-w^2\delta. \quad ◀$$

▶ **Corollary 26.** *For every function $\mathcal{A} : \{0,1\}^m \to \mathbb{R}^{w\times w}$ and every (ε,δ)-sampler $g : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$,*

$$\Pr_{x\in\{0,1\}^n}\left[\left\|\mathop{\mathbb{E}}_{s\in\{0,1\}^d}\left[\mathcal{A}(g(x,s))\right]\right\| \le \|\mathcal{A}\| + 2w\mu(\mathcal{A})\varepsilon\right] \ge 1-w^2\delta.$$

Proof. By sub-additivity of $\|\cdot\|$, $\left\|\mathbb{E}_{s\in\{0,1\}^d}\left[\mathcal{A}(g(x,s))\right]-\langle A\rangle\right\| \le 2w\mu(\mathcal{A})\varepsilon$ implies $\left\|\mathbb{E}_{s\in\{0,1\}^d}\left[\mathcal{A}(g(x,s))\right]\right\| \le \|\langle\mathcal{A}\rangle\| + 2w\mu(\mathcal{A})\varepsilon$. The claim now directly follows from Lemma 25. ◀

Now we introduce three different matrix multiplication rules. The first one is applying a sampler on both sides, and the second and third are applying sampler on only one side.

▶ **Lemma 27** (symmetric product)**.** *Consider $\mathcal{A} : \{0,1\}^n \to \mathbb{R}^{w\times w}$ and $\mathcal{B} : \{0,1\}^m \to \mathbb{R}^{w\times w}$. Let $f : \{0,1\}^k \times \{0,1\}^{d_A} \to \{0,1\}^n$ be a (δ,ε_A) sampler, and $g : \{0,1\}^k \times \{0,1\}^{d_B} \to \{0,1\}^m$ be a (δ,ε_B) sampler. Then*

$$\mathop{\mathbb{E}}_{z}\left[\left\|\mathop{\mathbb{E}}_{x,y}\left[\mathcal{A}(f(z,x))\mathcal{B}(g(z,y))\right]\right\|\right] \le 2w^2\delta\mu(\mathcal{A})\mu(\mathcal{B}) + \left(\|\mathcal{A}\| + 2w\mu(\mathcal{A})\varepsilon_A\right)\left(\|\mathcal{B}\| + 2w\mu(\mathcal{B})\varepsilon_B\right).$$

Proof. Let

$$E_A = \left\{z : \left\|\mathop{\mathbb{E}}_{x}\left[\mathcal{A}(f(z,x))\right]\right\| > \|\mathcal{A}\| + 2w\mu(\mathcal{A})\varepsilon_A\right\},$$

and

$$E_B = \left\{z : \left\|\mathop{\mathbb{E}}_{y}\left[\mathcal{B}(g(z,y)\right]\right\| > \|\mathcal{B}\| + 2w\mu(\mathcal{B})\varepsilon_B\right\}.$$

Define $E = E_A \cup E_B$. By Lemma 26 and union bound, $\Pr_z[z \in E] < 2w^2\delta$. Therefore

$$\begin{aligned}
\mathop{\mathbb{E}}_z\left[\left\|\mathop{\mathbb{E}}_{x,y}[\mathcal{A}(f(z,x))\mathcal{B}(g(z,y))]\right\|\right] &= \Pr[z \in E]\mathop{\mathbb{E}}_{z\in E}\left[\left\|\mathop{\mathbb{E}}_{x,y}[\mathcal{A}(f(z,x))\mathcal{B}(g(z,y))]\right\|\right] \\
&\quad + \Pr[z \notin E]\mathop{\mathbb{E}}_{z\notin E}\left[\left\|\mathop{\mathbb{E}}_{x}[\mathcal{A}(f(z,x))]\mathop{\mathbb{E}}_{y}[\mathcal{B}(g(z,y))]\right\|\right] \\
&\le 2w^2\delta\mu(\mathcal{A})\mu(\mathcal{B}) + \mathop{\mathbb{E}}_{z\notin E}\left[\left\|\mathop{\mathbb{E}}_{x}[\mathcal{A}(f(z,x))]\right\|\left\|\mathop{\mathbb{E}}_{y}[\mathcal{B}(g(z,y))]\right\|\right] \\
&\le 2w^2\delta\mu(\mathcal{A})\mu(\mathcal{B}) + \left(\|\mathcal{A}\| + 2w\mu(\mathcal{A})\varepsilon_A\right)\left(\|\mathcal{B}\| + 2w\mu(\mathcal{B})\varepsilon_B\right).
\end{aligned}$$

The second last inequality is by the fact that $\|\cdot\|$ is non-negative and sub-multiplicative. ◀

▶ **Lemma 28** (left product). *Consider $\mathcal{A} : \{0,1\}^k \to \mathbb{R}^{w\times w}$ and $\mathcal{B} : \{0,1\}^m \to \mathbb{R}^{w\times w}$. Let $g : \{0,1\}^k \times \{0,1\}^{d_B} \to \{0,1\}^m$ be a (δ, ε_B) sampler. Then*

$$\mathop{\mathbb{E}}_z\left[\left\|\mathop{\mathbb{E}}_{y}[\mathcal{A}(z)\mathcal{B}(g(z,y))]\right\|\right] \le w^2\delta\mu(\mathcal{A})\mu(\mathcal{B}) + \|\mathcal{A}\|_r\left(\|\mathcal{B}\| + 2w\mu(\mathcal{B})\varepsilon_B\right).$$

Proof. Let

$$E = \left\{z : \left\|\mathop{\mathbb{E}}_{y}[\mathcal{B}(g(z,y)]\right\| > \|\mathcal{B}\| + 2w\mu(\mathcal{B})\varepsilon_B\right\}.$$

By Lemma 26, $\Pr_z[z \in E] < w^2\delta$. Therefore

$$\begin{aligned}
\mathop{\mathbb{E}}_z\left[\left\|\mathop{\mathbb{E}}_{y}[\mathcal{A}(z)\mathcal{B}(g(z,y))]\right\|\right] &= \Pr[z \in E]\mathop{\mathbb{E}}_{z\in E}\left[\left\|\mathop{\mathbb{E}}_{y}[\mathcal{A}(z)\mathcal{B}(g(z,y))]\right\|\right] \\
&\quad + \Pr[z \notin E]\mathop{\mathbb{E}}_{z\notin E}\left[\left\|\mathop{\mathbb{E}}_{y}[\mathcal{A}(z)\mathcal{B}(g(z,y))]\right\|\right] \\
&\le w^2\delta\mu(\mathcal{A})\mu(\mathcal{B}) + \Pr[z \notin E]\cdot\mathop{\mathbb{E}}_{z\notin E}\left[\|\mathcal{A}(z)\|\left\|\mathop{\mathbb{E}}_{y}[\mathcal{B}(g(z,y))]\right\|\right] \\
&\le w^2\delta\mu(\mathcal{A})\mu(\mathcal{B}) + \Pr[z \notin E]\mathop{\mathbb{E}}_{z\notin E}[\|\mathcal{A}(z)\|]\cdot(\|\mathcal{B}\| + 2w\mu(\mathcal{B})\varepsilon_B) \\
&\le w^2\delta\mu(\mathcal{A})\mu(\mathcal{B}) + \|\mathcal{A}\|_r\left(\|\mathcal{B}\| + 2w\mu(\mathcal{B})\varepsilon_B\right).
\end{aligned}$$

The third last inequality is by sub-multiplicativity of $\|\cdot\|$, the second last inequality is by non-negativity of $\|\cdot\|$, and the last inequality is by the fact that

$$\Pr[z \notin E]\cdot\mathop{\mathbb{E}}_{z\notin E}[\|\mathcal{A}(z)\|] = \mathop{\mathbb{E}}_z[\|\mathcal{A}(z)\|\cdot\mathbb{1}(z \notin E)] \le \|\mathcal{A}\|_r.$$ ◀

▶ **Lemma 29** (right product). *Consider $\mathcal{A} : \{0,1\}^k \to \mathbb{R}^{w\times w}$ and $\mathcal{B} : \{0,1\}^m \to \mathbb{R}^{w\times w}$. Let $f : \{0,1\}^k \times \{0,1\}^{d_A} \to \{0,1\}^n$ be a (δ, ε_A) sampler. Then*

$$\mathop{\mathbb{E}}_z\left[\left\|\mathop{\mathbb{E}}_{x}[\mathcal{A}(f(z,x))\mathcal{B}(z)]\right\|\right] \le w^2\delta\mu(\mathcal{A})\mu(\mathcal{B}) + \left(\|\mathcal{A}\| + 2w\mu(\mathcal{A})\varepsilon_A\right)\|\mathcal{B}\|_r.$$

Proof. Let

$$E = \left\{z : \left\|\mathop{\mathbb{E}}_{x}[\mathcal{A}(f(z,x)]\right\| > \|\mathcal{A}\| + 2w\mu(\mathcal{A})\varepsilon_A\right\}.$$

By Lemma 26, $\Pr_z[z \in E] < w^2\delta$. Therefore

$$\begin{aligned}
\mathop{\mathbb{E}}_{z}\left[\left\|\mathop{\mathbb{E}}_{x}\left[\mathcal{A}(f(z,x))\mathcal{B}(z)\right]\right\|\right] &= \Pr\left[z\in E\right]\mathop{\mathbb{E}}_{z\in E}\left[\left\|\mathop{\mathbb{E}}_{x}\left[\mathcal{A}(f(z,x))\mathcal{B}(z)\right]\right\|\right] \\
&\quad+\Pr\left[z\notin E\right]\mathop{\mathbb{E}}_{z\notin E}\left[\left\|\mathop{\mathbb{E}}_{x}\left[\mathcal{A}(f(z,x))\mathcal{B}(z)\right]\right\|\right] \\
&\le w^2\delta\mu(\mathcal{A})\mu(\mathcal{B})+\Pr\left[z\notin E\right]\cdot\mathop{\mathbb{E}}_{z\notin E}\left[\left\|\mathop{\mathbb{E}}_{x}\left[\mathcal{A}(f(z,x))\right]\right\|\left\|\mathcal{B}(z)\right\|\right] \\
&\le w^2\delta\mu(\mathcal{A})\mu(\mathcal{B})+\left(\left\|\mathcal{A}\right\|+2w\mu(\mathcal{A})\varepsilon_A\right)\cdot\Pr\left[z\notin E\right]\mathop{\mathbb{E}}_{z\notin E}\left[\left\|\mathcal{B}(z)\right\|\right] \\
&\le w^2\delta\mu(\mathcal{A})\mu(\mathcal{B})+\left(\left\|\mathcal{A}\right\|+2w\mu(\mathcal{A})\varepsilon_A\right)\left\|\mathcal{B}\right\|_r. \qquad \blacktriangleleft
\end{aligned}$$

6 Main Construction

In this section we show our main construction and prove its correctness. We first introduce several definitions.

▶ **Definition 30.** *For every mapping $\mathcal{A}:\{0,1\}^n\to\mathbb{R}^{w\times w}$ and every matrix $A\in\mathbb{R}^{w\times w}$, we define $\mathcal{A}-A$ to be the mapping s.t. $(\mathcal{A}-A)(x)=\mathcal{A}(x)-A$.*

▶ **Definition 31.** *Consider $A\in\mathbb{R}^{w\times w}$ and $\mathcal{A}:\{0,1\}^n\to\mathbb{R}^{w\times w}$. $\mathcal{A}$ is a* ε-approximator *of A if $\left\|\mathbb{E}_x\left[\mathcal{A}(x)\right]-A\right\|\le\varepsilon$, i.e. $\left\|\mathcal{A}-A\right\|\le\varepsilon$. $\mathcal{A}$ is a* ε-robust approximator *of A if $\mathbb{E}_x\left[\left\|\mathcal{A}(x)-A\right\|\right]\le\varepsilon$, i.e. $\left\|\mathcal{A}-A\right\|_r\le\varepsilon$.*

Now we define a robust PRPD. Note that a (n,w,ε)-robust PRPD (G,ρ) is also a $\mu(G,\rho)$-bounded (n,w,ε)-PRPD.

▶ **Definition 32.** *$(G,\rho):\{0,1\}^{s_{\text{out}}}\times\{0,1\}^{s_{\text{in}}}\times[\mu]\to\{0,1\}^n\times\mathbb{R}$ is a (n,w,ε)-*robust PRPD *if for every (n,w)-ROBP and its matrix representation $[M]_{[n]}^{\{0,1\}}$ the following holds. Let $\mathcal{A}:\{0,1\}^{s_{\text{out}}}\times\{0,1\}^{s_{\text{in}}}\to\mathbb{R}^{w\times w}$ denote the mapping*

$$\mathcal{A}(x,y)=\mathop{\mathbb{E}}_{i\in[\mu]}\left[\rho(x,y,i)\cdot M_{0..n}^{G(x,y,i)}\right].$$

- *Every $\rho(x,y,i)$ is either μ or $-\mu$. In other word, $\mathcal{A}(x,y)$ is the summation of transition matrices with coefficient ±1.*
- *Let $\widehat{\mathcal{A}}$ denote the mapping $\widehat{\mathcal{A}}(x)=\mathbb{E}_y\left[\mathcal{A}(x,y)\right]$. Then $\widehat{\mathcal{A}}$ is a ε-robust approximator for $M_{0..n}$.*

We say μ is the weight of (G,ρ), denoted by $\mu(G,\rho)$. s_{out} is the outer seed length *of (G,ρ), denoted by $s_{\text{out}}(G,\rho)$. s_{in} is the* inner seed length *of (G,ρ), denoted by $s_{\text{in}}(G,\rho)$. We write $s(G,\rho)=s_{\text{out}}(G,\rho)+s_{\text{in}}(G,\rho)$ for short. We say (G,ρ) is explicit if it can be computed in $O(s(G,\rho))$ space.*

We say $\mathcal{A}$ is the matrix form *of (G,ρ) on $M_{0..n}$, and the definition of $s_{\text{out}},s_{\text{in}},\mu$ on (G,ρ) also apply to $\mathcal{A}$. We say $\widehat{\mathcal{A}}$ is the* robust matrix form *of (G,ρ) on $M_{0..n}$.*

▶ Remark 33. The above definition is similar to Definition 5, but each matrix $\mathcal{A}(x,y)$ is realized with μ matrices instead of one matrix. These μ matrices will never be separated even after flattening. We do this in order to ensure that the matrix form always take bit-strings as input. This ensures that we can increase the outer and inner seed length of $\mathcal{A}$ arbitrarily: we can construct the new mapping $\mathcal{A}':\{0,1\}^{s'_{\text{out}}}\times\{0,1\}^{s'_{\text{in}}}$ such that $\mathcal{A}'(x,y)=\mathcal{A}(x_p,y_p)$ where x_p is the length-$s_{\text{out}}(\mathcal{A})$ prefix of x and y_p is the length-$s_{\text{in}}(\mathcal{A})$ prefix of y. In other word, $\mathcal{A}'$ computes the output only with prefix of necessary length of the input, and ignore the remaining bits. It is easy to verify that $\mathcal{A}'$ is also the matrix form of a (n,w,ε)-robust PRPD.

The following is some additional basic properties about robust PRPD and its flattened form.

▷ Claim 34. Let $(G,\rho):\{0,1\}^{s_{\text{out}}}\times\{0,1\}^{s_{\text{in}}}\times[\mu]\to\{0,1\}^n\times\mathbb{R}$ be a (n,w,ε)-*robust PRPD*. For every (n,w)-ROBP $M_1^0,M_1^1,\dots,M_n^0,M_n^1$ the following holds.

- Let $\widehat{\mathcal{A}}$ be the robust matrix form of (G,ρ) on $M_{0..n}$. Then $\mu(\widehat{\mathcal{A}})\le\mu(G,\rho)$.
- Let $\mathcal{A}$ denote the matrix form of (G,ρ) on $M_{0..n}$. Let $\overline{\mathcal{A}}:\{0,1\}^{s_{\text{out}}+s_{\text{in}}}\to\mathbb{R}^{w\times w}$ denote the mapping $\overline{\mathcal{A}}(x\|y)=\mathcal{A}(x,y)$. We say $\overline{\mathcal{A}}$ is the *flattened matrix form* of (G,ρ) on $M_{0..n}$. Then $\overline{\mathcal{A}}$ is an ε-approximator for $M_{0..n}$, and $\mu(\overline{\mathcal{A}})\le\mu(G,\rho)$.

Proof. Recall that for every string $r\in\{0,1\}^n$, $\|M^r_{0..n}\|=1$. By sub-additivity of $\|\cdot\|$ we have $\|\mathcal{A}(x,y)\|\le\mu(G,\rho)$ for every x,y, which implies $\mu(\overline{\mathcal{A}})\le\mu(G,\rho)$. By sub-additivity and scalability of $\|\cdot\|$, we have $\mu(\mathcal{A}')\le\mu(\mathcal{A})$. To show that $\overline{\mathcal{A}}$ is a ε-approimxator of $M_{0..n}$, observe that $\mathcal{A}'$ is also an ε-approximator of $M_{0..n}$ by Claim 24, and note that $\langle\mathcal{A}\rangle=\langle\mathcal{A}'\rangle$. ◁

Now we prove our main lemma. The following lemma allows us to construct robust PRPDs for $(2m,w)$ ROBPs from robust PRPDs for (m,w) ROBPs, without increasing the seed length too much. We will recursively apply this lemma for $\log n$ levels to get a (n,w,ε)-robust PRPD. The basic idea is as described in Lemma 16.

▶ **Lemma 35.** *Suppose there exists $s_{\text{out}},s_{\text{in}}$ such that the following conditions hold.*

- *For every $0\le i\le k$, there exists a (m,w,γ^{i+1})-robust PRPD (G_i,ρ_i) s.t. $\mu(G_i,\rho_i)\le\binom{m-1}{i}$ and $s_{\text{out}}(G,\rho)\le s_{\text{out}}$. Moreover, for every $0\le i\le\lceil k/2\rceil$, $s(G_i,\rho_i)\le s_{\text{out}}$.*
- *For every $i\le\lceil k/2\rceil$, there exists a (ε_i,δ)-sampler $g_i:\{0,1\}^{s_{\text{out}}}\times\{0,1\}^{d_i}\to\{0,1\}^{s(G_i,\rho_i)}$, where $\varepsilon_i\le\gamma^{i+1}/(w\cdot\binom{m-1}{i})$ and $\delta\le\gamma^{k+1}/(w^2\cdot\binom{2m-1}{i})$.*
- *For every $i\ge j\ge0$ s.t. $i+j\le k$, if $j\le i\le\lceil k/2\rceil$, then $d_i+d_j\le s_{\text{in}}$. If $i>\lceil k/2\rceil$, then $s_{\text{in}}(G_i,\rho_i)+d_j\le s_{\text{in}}$.*

Then there exists a $(2m,w,(11\gamma)^{k+1})$-robust PRPD (G,ρ) s.t. $s_{\text{out}}(G,\rho)=s_{\text{out}}$, $s_{\text{in}}(G,\rho)=s_{\text{in}}$ and $\mu(G,\rho)\le\binom{2m-1}{k}$.

Proof. Fix any $(2m,w)$-ROBP with matrix representation $M^{\{0,1\}}_{[2m]}$. Let $A=M_{0..m}$ and $B=M_{m..2m}$. For every $0\le i\le k$, let $\mathcal{A}_i,\widehat{\mathcal{A}}_i,\overline{\mathcal{A}}_i$ denote the matrix form, robust matrix form and flattened matrix form of (G,ρ) on $M_{0..m}$ respectively. Let $\mathcal{B}_i,\widehat{\mathcal{B}}_i,\overline{\mathcal{B}}_i$ denote the matrix form, robust matrix form and flattened matrix form of (G,ρ) on $M_{m..2m}$ respectively. By definition, $\widehat{\mathcal{A}}_i$ and $\widehat{\mathcal{B}}_i$ are γ^{i+1}-robust approximator for A and B respectively. By Claim 34, $\overline{\mathcal{A}}_i$ and $\overline{\mathcal{B}}_i$ are γ^{i+1}-approximator for A and B respectively. Moreover, we will increase the outer seed length of $\mathcal{A}_i$ and $\mathcal{B}_i$ to match the length of the given input when necessary. (See Remark 33)

Now for every x,y we define a mapping $\mathcal{C}_k:\{0,1\}^{s_{\text{out}}}\times\{0,1\}^{s_{\text{in}}}\to\mathbb{R}^{w\times w}$ as follows. Note that $\mathcal{C}_k$ corresponds to the matrix form of (G,ρ) on $M_{0..2m}$.

(1) For every $0\le i\le\lceil k/2\rceil$, let a_i be the prefix of y of length d_i and b_i be the suffix of y of length d_i. Define $A_{x,y,i}=\overline{\mathcal{A}_i}(g_i(x,a_i))$ and $B_{x,y,i}=\overline{\mathcal{B}_i}(g_i(x,b_i))$.

(2) For every $\lceil k/2\rceil<i\le k$, let a_i be the prefix of y of length $s_{\text{in}}(\mathcal{A}_i)$ and b_i be the suffix of y of length $s_{\text{in}}(\mathcal{B}_i)$. Define $A_{x,y,i}=\mathcal{A}_i(x,a_i)$ and $B_{x,y,i}=\mathcal{B}_i(x,b_i)$.

(3) Define $\mathcal{C}_k(x,y)=\sum_{i+j=k}A_{x,y,i}B_{x,y,j}-\sum_{i+j=k-1}A_{x,y,i}B_{x,y,j}$.

Note that for every $i+j\le k$, prefix a_i and suffix b_j of y never overlap.

By expanding every $A_{x,y,i}B_{x,y,j}$ term with distributive law, we can see that each small term in $A_{x,y,i}B_{x,y,j}$ has coefficient ±1, which satisfies the first condition of robust PRPD. Moreover, the total number of terms after expanding is

$$\mu(\mathcal{C}_k)\le\sum_{i+j=k}\binom{m-1}{i}\cdot\binom{m-1}{j}+\sum_{i+j=k-1}\binom{m-1}{i}\cdot\binom{m-1}{j}=\binom{2m-1}{k}.$$

It remains to show that $\mathcal{C}_k$ satisfies the second condition of robust PRPD, i.e. $\mathbb{E}_y[\mathcal{C}_k(x,y)]$ is a good approximation of $M_{0..2m} = AB$ on average over x. Observe that

$$\begin{aligned}
\mathop{\mathbb{E}}_x\left[\left\|\mathop{\mathbb{E}}_y[\mathcal{C}_k(x,y)] - AB\right\|\right] &= \mathop{\mathbb{E}}_x\left[\left\|\mathop{\mathbb{E}}_y[\mathcal{C}_k(x,y) - AB]\right\|\right] \\
&\le \sum_{i+j=k} \mathop{\mathbb{E}}_x\left[\left\|\mathop{\mathbb{E}}_y[(A_{x,y,i} - A)(B_{x,y,j} - B)]\right\|\right] \\
&+ \sum_{i+j=k-1} \mathop{\mathbb{E}}_x\left[\left\|\mathop{\mathbb{E}}_y[(A_{x,y,i} - A)(B_{x,y,j} - B)]\right\|\right] \\
&+ \mathop{\mathbb{E}}_x\left[\left\|\mathop{\mathbb{E}}_y[(A_{x,y,k} - A)B]\right\|\right] + \mathop{\mathbb{E}}_x\left[\left\|\mathop{\mathbb{E}}_y[A(B_{x,y,k} - B)]\right\|\right],
\end{aligned}$$

by decomposing $\mathcal{C}_k(x,y) - AB$ with the equation in the proof of Lemma 15 and applying sub-additivity of $\|\cdot\|$.

First we consider the last two terms. Since $\|B\| = 1$, by sub-multiplicativity we have

$$\mathop{\mathbb{E}}_x\left[\left\|\mathop{\mathbb{E}}_y[(A_{x,y,k} - A)B]\right\|\right] \le \mathop{\mathbb{E}}_x\left[\left\|\mathop{\mathbb{E}}_y[A_{x,y,k} - A]\right\|\right].$$

Now consider two cases. If $k \ge 2$, then

$$\mathop{\mathbb{E}}_x\left[\left\|\mathop{\mathbb{E}}_y[A_{x,y,k} - A]\right\|\right] = \mathop{\mathbb{E}}_x\left[\left\|\widehat{\mathcal{A}}_k(x) - A\right\|\right] \le \gamma^{k+1}$$

by definition. If $k < 2$, then

$$\mathop{\mathbb{E}}_x\left[\left\|\mathop{\mathbb{E}}_y[A_{x,y,k} - A]\cdot B\right\|\right] = \mathop{\mathbb{E}}_x\left[\left\|\mathop{\mathbb{E}}_{a_k}[\overline{\mathcal{A}_k}(g_k(x,a_k)) - A]\right\|\cdot B\right].$$

Apply Lemma 29 on $\overline{\mathcal{A}_k} - A$ and the dummy mapping $\mathcal{B}$ s.t. $\mathcal{B}(x) = B$ for every x, we can derive that the above formula is bounded by $w^2\delta\binom{m-1}{k} + 3\gamma^{i+1}$. For the term $\mathbb{E}_x\left[\|\mathbb{E}_y[A(B_{x,y,k} - B)]\|\right]$ we can get the same bound with a similar proof.

Now consider the terms in the form $\mathbb{E}_x\left[\|\mathbb{E}_y[(A_{x,y,i} - A)(B_{x,y,j} - B)]\|\right]$. First consider the case $i, j \le \lceil k/2\rceil$. Then

$$\begin{aligned}
&\mathop{\mathbb{E}}_x\left[\left\|\mathop{\mathbb{E}}_y[(A_{x,y,i} - A)(B_{x,y,j} - B)]\right\|\right] \\
&= \mathop{\mathbb{E}}_x\left[\left\|\mathop{\mathbb{E}}_{a_i}[\overline{\mathcal{A}_i}(g_i(x,a_i)) - A]\mathop{\mathbb{E}}_{b_j}[\overline{\mathcal{B}_k}(g_j(x,b_j)) - B]\right\|\right] \text{ (since } a_i, b_j \text{ don't overlap)} \\
&\le 2w^2\delta\cdot\binom{m-1}{i}\cdot\binom{m-1}{j} + 9\gamma^{i+j+2}. \text{ (by Lemma 27)}
\end{aligned}$$

Next consider the case $i > \lceil k/2\rceil, j \le \lceil k/2\rceil$. Then

$$\begin{aligned}
&\mathop{\mathbb{E}}_x\left[\left\|\mathop{\mathbb{E}}_y[(A_{x,y,i} - A)(B_{x,y,j} - B)]\right\|\right] \\
&= \mathop{\mathbb{E}}_x\left[\left\|\widehat{\mathcal{A}}_i(x)\cdot\mathop{\mathbb{E}}_{b_j}[\overline{\mathcal{B}_k}(g_j(x,b_j)) - B]\right\|\right] \text{ (since } a_i, b_j \text{ don't overlap)} \\
&\le w^2\delta\cdot\binom{m-1}{i}\cdot\binom{m-1}{j} + 3\gamma^{i+j+2}. \text{ (by Lemma 29)}
\end{aligned}$$

Similarly for the case that $i \le \lceil k/2 \rceil, j > \lceil k/2 \rceil$ we can show that

$$\mathop{\mathbb{E}}_{x}\left[\left\|\mathop{\mathbb{E}}_{y}\left[(A_{x,y,i}-A)(B_{x,y,j}-B)\right]\right\|\right] \le w^2\delta \cdot \binom{m-1}{i}\cdot\binom{m-1}{j} + 3\gamma^{i+j+2}$$

by Lemma 28. Finally, note that the case $i, j > \lceil k/2 \rceil$ does not exist because $i + j \le k$.

Taking the summation of all the cases, we get

$$\begin{aligned}
&\mathop{\mathbb{E}}_{x}\left[\left\|\mathop{\mathbb{E}}_{y}\left[\mathcal{C}_k(x,y)\right]-AB\right\|\right] \\
&\le 2w^2\delta\cdot\left(\sum_{i+j=k}\binom{m-1}{i}\binom{m-1}{j}+\sum_{i+j=k-1}\binom{m-1}{i}\binom{m-1}{j}+\binom{m-1}{k}\right) \\
&+(k+1)\cdot 9\gamma^{k+2}+k\cdot 9\gamma^{k+1}+2\cdot 3\gamma^{k+1} \\
&\le 4w^2\delta\cdot\binom{2m-1}{k}+(9k+9)\gamma^{k+2}+(9k+6)\gamma^{k+1} \\
&\le (10k+11)\gamma^{k+1} \\
&\le (11\gamma)^{k+1}.
\end{aligned}$$

Moreover, note that $AB = M_{0..2m}$, and the construction of $\mathcal{C}_k$ does not depend on the matrices $M_{[2m]}^{\{0,1\}}$. (See Section 2 for how the arithmetic operations in $\mathcal{C}_k(x,y)$ are translated back to operations on pseudo-distributions.) Therefore there exists a $(2m, w, (11\gamma)^{k+1})$-robust PRPD (G,ρ). ◀

Finally we analyze the seed length of the recursive construction, and present the main theorem.

▶ **Theorem 36.** *There exists an explicit (n, w, ε)-robust PRPD (G,ρ) such that*

- $s_{\text{out}}(G,\rho) = O\left(\log(1/\varepsilon)+\log n\log(nw)\log\left(\frac{\log(1/\varepsilon)}{\log n}\right)\right)$
- $s_{\text{in}}(G,\rho) = O\left(\log(1/\varepsilon)+\log n\log(nw)\log\left(\frac{\log(1/\varepsilon)}{\log n}\right)\right)$
- $\mu(G,\rho) = \text{poly}(1/\varepsilon)$

Moreover, for every B the approximator $\mathcal{G}$ has the same corresponding pseudodistribution.

Proof. Let c be the constant such that for every $\varepsilon, \delta > 0$ there exists a (ε,δ)-sampler $g : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ such that $n = m + c\log(1/\varepsilon) + c\log(1/\delta)$ and $d = c\log(1/\varepsilon) + c\log\log(1/\delta)$, as guaranteed in Lemma 19. WLOG assume that n is a power of 2. Define $\gamma = 1/n^4$. For every $0 \le h \le \log n$, every $k \ge 0$, we will inductively prove that there exists a $(2^h, w, (11^h\gamma)^{k+1})$-robust PRPD $(G_{h,k}, \rho_{h,k})$ with the following parameters.

- If $k \le 1$, $s_{\text{out}}(G_{h,k},\rho_{h,k}) \le h\cdot(3ck\log(n/\gamma)+7c\log(w/\gamma))$
- If $k > 1$, $s_{\text{out}}(G_{h,k},\rho_{h,k}) \le 4ck\log(n/\gamma)+(\lceil\log k\rceil+1)\cdot h\cdot(10c\log(w/\gamma))$
- If $k \le 1$, $s_{\text{in}}(G_{h,k},\rho_{h,k}) \le ck\log(n/\gamma)+4c\log(w/\gamma)$
- If $k > 1$, $s_{\text{in}}(G_{h,k},\rho_{h,k}) \le ck\log(n/\gamma)+h\cdot(4c\log(kw/\gamma))$
- $\mu(G_{h,k},\rho_{h,k}) \le \max(1,\binom{2^h-1}{k})$

We will write $s_{\text{out},h,k} = s_{\text{out}}(G_{h,k},\rho_{h,k})$ and $s_{\text{in},h,k} = s_{\text{out}}(G_{h,k},\rho_{h,k})$ for short. First consider the terminal case $2k \ge 2^h$ or $h = 0$. In this case we simply take $s_{\text{out},h,k} = 0$, $s_{\text{in},h,k} = 2^h \le 2k$ and $\mu(G_{h,k},\rho_{h,k}) = 1$ s.t. $G_{h,k}(x,y,i) = y$ and $\rho_{h,k}(x,y,i) = 1$. For the other cases, we show that we can get the intended parameters by constructing $(G_{h,k},\rho_{h,k})$ with the recursion in Lemma 35. Note that based on the induction hypothesis we can assume $\mathcal{G}_{a,h-1,k}$ and $\mathcal{G}_{a+2^{h-1},h-1,k}$ have exactly the same parameters, so we consider the

parameter of $\mathcal{G}_{a,h-1,k}$ only. We have seen that the bound for $\mu(G_{h,k}, \rho_{h,k})$ is correct. First we show that the bound for $s_{\text{in},h,k}$ is correct. Recall that in the recursion we take parameters $d_i = c\log(1/\varepsilon_i) + c\log\log(1/\delta) \leq ci\log(n/\gamma) + 2c\log(knw/\gamma)$, based on the fact that $\binom{2^h-1}{i} \leq n^i$. Now consider the restriction on $s_{\text{in}}(\mathcal{G}_k)$ in our recursion. For $i + j \leq k$ and $j \leq i \leq \lceil k/2 \rceil$, we need

$$d_i + d_j \leq ck\log(n/\gamma) + 4c\log(knw/\gamma) \leq s_{\text{in},h,k}$$

which is true. For $i + j \leq k$ and $i > \lceil k/2 \rceil$, we need

$$\begin{aligned} s_{\text{in},h-1,i} + d_j &\leq ci\log(1/\gamma) + (h-1)\cdot 4c\log(inw/\gamma) + (cj\log(1/\gamma) + 2c\log(knw/\gamma)) \\ &\leq ck\log(1/\gamma) + h\cdot 4c\log(knw/\gamma) \\ &\leq s_{\text{in},h,k} \end{aligned}$$

which is also true. Moreover, observe that when $k \leq 1$ it is always the case that $i, j \leq \lceil k/2 \rceil$. Therefore the third condition is also true. Finally we show that the bound for $s_{\text{out},h,k}$ is also correct. First observe that the restriction $s_{\text{out},h-1,i} \leq s_{\text{out},h,k}$ is trivially true. Then the only condition left is that for every $i \leq \lceil k/2 \rceil$,

$$s_{\text{out},h-1,i} + s_{\text{in},h-1,i} + c\log(1/\delta) + c\log(1/\varepsilon_i) \leq s_{\text{out},h,k}.$$

Since $s_{\text{out},h-1,i} \leq s_{\text{out},h-1,\lceil k/2 \rceil}$ and $s_{\text{in},h-1,i} \leq s_{\text{in},h-1,\lceil k/2 \rceil}$ for every i, it suffices to show that

$$s_{\text{out},h-1,\lceil k/2 \rceil} + s_{\text{in},h-1,\lceil k/2 \rceil} + c\log(1/\delta) + c\log(1/\varepsilon_{\lceil k/2 \rceil}) \leq s_{\text{out},h,k}.$$

First we consider $k \leq 1$, which is the case that $\lceil k/2 \rceil = k$. Then

$$\begin{aligned} &s_{\text{out},h-1,\lceil k/2 \rceil} + s_{\text{in},h-1,\lceil k/2 \rceil} + c\log(1/\delta) + c\log(1/\varepsilon_{\lceil k/2 \rceil}) \\ &\leq s_{\text{out}}(\mathcal{G}_{a,h-1,k}) + 3ck\log(n/\gamma) + 7c\log(n/\gamma) \\ &\leq h\cdot(3ck\log(n/\gamma) + 7c\log(n/\gamma)) \\ &\leq s_{\text{out},h,k}. \end{aligned}$$

Finally we consider the case $k > 1$. Observe that

$$\begin{aligned} &s_{\text{out},h-1,\lceil k/2 \rceil} + s_{\text{in},h-1,\lceil k/2 \rceil} + c\log(1/\delta) + c\log(1/\varepsilon_{\lceil k/2 \rceil}) \\ &\leq s_{\text{out},h-1,\lceil k/2 \rceil} + s_{\text{in},h-1,\lceil k/2 \rceil} + \frac{3k+1}{2}\cdot c\log(n/\gamma) + 3c\log(w/\gamma) \\ &\leq s_{\text{out},h-1,\lceil k/2 \rceil} + (2k+1)\cdot c\log(n/\gamma) + (h-1)\cdot 4c\log(w/\gamma) + 7c\log(w/\gamma) \\ &\leq 4c\cdot\frac{k+1}{2}\cdot\log(n/\gamma) + \left(\lceil\log\lceil\frac{k}{2}\rceil\rceil + 1\right)\cdot(h-1)\cdot(10c\log(nw/\gamma)) \\ &+ (2k+1)\cdot c\log(n/\gamma) + (h-1)\cdot 4c\log(w/\gamma) + 7c\log(w/\gamma) \\ &\leq 4ck\log(n/\gamma) + \left(\lceil\log\lceil\frac{k}{2}\rceil\rceil + 1\right)\cdot(h-1)\cdot(10c\log(nw/\gamma)) + h\cdot 10c\log(nw/\gamma) \\ &\leq s_{\text{out},h,k}. \end{aligned}$$

In the last inequality we use the fact that $\lceil\log k\rceil = \lceil\log(\lceil k/2\rceil)\rceil + 1$ for every $k > 1$.

Finally, note that $(11^{\log n}\gamma) = n^{\log_2 11}\cdot n^{-4} \leq n^{-0.5}$. By taking $h = \log n$ and $k = \frac{\log(1/\varepsilon)}{\log(1/n^{0.5})}$, we get a (n, w, ε)-robust PRPD. ◀

▶ Remark 37. To get the seed length we claimed in Theorem 4, observe that the $\log(1/\varepsilon)$ term is dominating when $\log(1/\varepsilon) \geq \log^3(nw)$. Therefore we can simply replace the $\log\log(1/\varepsilon)$ factor on the $O(\log n\log(nw))$ term with $\log\log(nw)$.

7 Discussion and Open Questions

We discuss some natural questions that arise from our work.

- In our construction, we applied the sampler argument in [4] without constructing small-norm matrices explicitly. This is probably hinting that negative weight is not essentially required for the sampler argument. Is it possible to apply the sampler argument to construct a PRG (instead of PRPD) with improved dependency on error?
- Is there an explicit PRPD which matches the seed length of the hitting set generator in [13], i.e. $O(\log(w/\varepsilon))$ when $n = \operatorname{poly}\log(w)$? A possible direction is to adapt our construction to a t-ary recursion tree where $t = \log^{1-\Omega(1)}(n)$ instead of a binary tree, as in [25, 1]. However, a direct adaption requires us to apply samplers on $(t-1)$-children in each recursion, and for every sampler we need to pay some randomness for "inner seed" which cannot be recycled. In our construction we see that the inner seed of a sampler contains a $\log w$ term. Therefore in each recursion we need to pay at least $(t-1)\log w$ which is too expensive. Is it possible to make the sampler argument work with a shorter inner seed?
- Is it possible to improve the seed length to $\tilde{O}(\log^2 n + \log(w/\varepsilon))$, even in some restricted settings? We note that there are two things which cause the $\Omega(\log n \cdot \log w)$ term in our construction. The first one is the inner seed of sampler, which is related to the question above. The second one is the restriction on the outer seed length, which is analogous to "entropy loss" if we view the samplers as extractors. Note that [26] shows how to "recycle entropy" in the INW generator in some restricted settings, but it is not clear how to apply the extractor-type analysis of INW generator in our construction.

References

1. Roy Armoni. On the derandomization of space-bounded computations. In Michael Luby, José D. P. Rolim, and Maria J. Serna, editors, *Randomization and Approximation Techniques in Computer Science, Second International Workshop, RANDOM'98, Barcelona, Spain, October 8-10, 1998, Proceedings*, volume 1518 of *Lecture Notes in Computer Science*, pages 47–59. Springer, 1998. doi:10.1007/3-540-49543-6_5.
2. Andrej Bogdanov, Zeev Dvir, Elad Verbin, and Amir Yehudayoff. Pseudorandomness for width-2 branching programs. *Theory of Computing*, 9:283–293, 2013. doi:10.4086/toc.2013.v009a007.
3. Allan Borodin, Stephen A. Cook, and Nicholas Pippenger. Parallel computation for well-endowed rings and space-bounded probabilistic machines. *Information and Control*, 58(1-3):113–136, 1983. doi:10.1016/S0019-9958(83)80060-6.
4. 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. doi:10.1145/3188745.3188780.
5. Mark Braverman, Anup Rao, Ran Raz, and Amir Yehudayoff. Pseudorandom generators for regular branching programs. *SIAM J. Comput.*, 43(3):973–986, 2014. doi:10.1137/120875673.
6. Joshua Brody and Elad Verbin. The coin problem and pseudorandomness for branching programs. In *51th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2010, October 23-26, 2010, Las Vegas, Nevada, USA*, pages 30–39. IEEE Computer Society, 2010. doi:10.1109/FOCS.2010.10.
7. Kuan Cheng and William Hoza. Hitting sets give two-sided derandomization of small space. *Electronic Colloquium on Computational Complexity (ECCC)*, 2020. URL: https://eccc.weizmann.ac.il/report/2020/016/.

8 Benny Chor and Oded Goldreich. Unbiased bits from sources of weak randomness and probabilistic communication complexity. *SIAM J. Comput.*, 17(2):230–261, 1988. doi:10.1137/0217015.

9 Anindya De. Pseudorandomness for permutation and regular branching programs. In *Proceedings of the 26th Annual IEEE Conference on Computational Complexity, CCC 2011, San Jose, California, USA, June 8-10, 2011*, pages 221–231. IEEE Computer Society, 2011. doi:10.1109/CCC.2011.23.

10 Oded Goldreich. A sample of samplers: A computational perspective on sampling. In Oded Goldreich, editor, *Studies in Complexity and Cryptography. Miscellanea on the Interplay between Randomness and Computation - In Collaboration with Lidor Avigad, Mihir Bellare, Zvika Brakerski, Shafi Goldwasser, Shai Halevi, Tali Kaufman, Leonid Levin, Noam Nisan, Dana Ron, Madhu Sudan, Luca Trevisan, Salil Vadhan, Avi Wigderson, David Zuckerman*, volume 6650 of *Lecture Notes in Computer Science*, pages 302–332. Springer, 2011. doi:10.1007/978-3-642-22670-0_24.

11 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. doi:10.1002/(SICI)1098-2418(199712)11:4<315::AID-RSA3>3.0.CO;2-1.

12 Venkatesan Guruswami, Christopher Umans, and Salil P. Vadhan. Unbalanced expanders and randomness extractors from parvaresh-vardy codes. *J. ACM*, 56(4):20:1–20:34, 2009. doi:10.1145/1538902.1538904.

13 William Hoza and David Zuckerman. Simple optimal hitting sets for small-success RL. In Mikkel Thorup, editor, *59th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2018, Paris, France, October 7-9, 2018*, pages 59–64. IEEE Computer Society, 2018. doi:10.1109/FOCS.2018.00015.

14 William M. Hoza and Chris Umans. Targeted pseudorandom generators, simulation advice generators, and derandomizing logspace. In Hamed Hatami, Pierre McKenzie, and Valerie King, editors, *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017, Montreal, QC, Canada, June 19-23, 2017*, pages 629–640. ACM, 2017. doi:10.1145/3055399.3055414.

15 Russell Impagliazzo, Noam Nisan, and Avi Wigderson. Pseudorandomness for network algorithms. In Frank Thomson Leighton and Michael T. Goodrich, editors, *Proceedings of the Twenty-Sixth Annual ACM Symposium on Theory of Computing, 23-25 May 1994, Montréal, Québec, Canada*, pages 356–364. ACM, 1994. doi:10.1145/195058.195190.

16 Russell Impagliazzo and Avi Wigderson. $P = BPP$ if E requires exponential circuits: Derandomizing the XOR lemma. In Frank Thomson Leighton and Peter W. Shor, editors, *Proceedings of the Twenty-Ninth Annual ACM Symposium on the Theory of Computing, El Paso, Texas, USA, May 4-6, 1997*, pages 220–229. ACM, 1997. doi:10.1145/258533.258590.

17 H. Jung. Relationships between probabilistic and deterministic tape complexity. In Jozef Gruska and Michal Chytil, editors, *Mathematical Foundations of Computer Science 1981, Strbske Pleso, Czechoslovakia, August 31 - September 4, 1981, Proceedings*, volume 118 of *Lecture Notes in Computer Science*, pages 339–346. Springer, 1981. doi:10.1007/3-540-10856-4_101.

18 Valentine Kabanets and Russell Impagliazzo. Derandomizing polynomial identity tests means proving circuit lower bounds. *Computational Complexity*, 13(1-2):1–46, 2004. doi:10.1007/s00037-004-0182-6.

19 Daniel M. Kane, Jelani Nelson, and David P. Woodruff. Revisiting norm estimation in data streams. *CoRR*, abs/0811.3648, 2008. arXiv:0811.3648.

20 Adam R. Klivans and Dieter van Melkebeek. Graph nonisomorphism has subexponential size proofs unless the polynomial-time hierarchy collapses. *SIAM J. Comput.*, 31(5):1501–1526, 2002. doi:10.1137/S0097539700389652.

21 Michal Koucký, Prajakta Nimbhorkar, and Pavel Pudlák. Pseudorandom generators for group products: extended abstract. In Lance Fortnow and Salil P. Vadhan, editors, *Proceedings of the 43rd ACM Symposium on Theory of Computing, STOC 2011, San Jose, CA, USA, 6-8 June 2011*, pages 263–272. ACM, 2011. doi:10.1145/1993636.1993672.

22 Raghu Meka, Omer Reingold, and Avishay Tal. Pseudorandom generators for width-3 branching programs. In Moses Charikar and Edith Cohen, editors, *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing, STOC 2019, Phoenix, AZ, USA, June 23-26, 2019*, pages 626–637. ACM, 2019. doi:10.1145/3313276.3316319.

23 Noam Nisan. Pseudorandom generators for space-bounded computation. *Combinatorica*, 12(4):449–461, 1992. doi:10.1007/BF01305237.

24 Noam Nisan. RL <= SC. *Computational Complexity*, 4:1–11, 1994. doi:10.1007/BF01205052.

25 Noam Nisan and David Zuckerman. Randomness is linear in space. *J. Comput. Syst. Sci.*, 52(1):43–52, 1996. doi:10.1006/jcss.1996.0004.

26 Ran Raz and Omer Reingold. On recycling the randomness of states in space bounded computation. In Jeffrey Scott Vitter, Lawrence L. Larmore, and Frank Thomson Leighton, editors, *Proceedings of the Thirty-First Annual ACM Symposium on Theory of Computing, May 1-4, 1999, Atlanta, Georgia, USA*, pages 159–168. ACM, 1999. doi:10.1145/301250.301294.

27 Omer Reingold, Salil P. Vadhan, and Avi Wigderson. Entropy waves, the zig-zag graph product, and new constant-degree expanders and extractors. *Electronic Colloquium on Computational Complexity (ECCC)*, 8(18), 2001. URL: http://eccc.hpi-web.de/eccc-reports/2001/TR01-018/index.html.

28 Michael Saks. Randomization and derandomization in space-bounded computation. In *Proceedings of Computational Complexity (Formerly Structure in Complexity Theory)*, pages 128–149. IEEE, 1996.

29 Michael E. Saks and Shiyu Zhou. BP $_{\mathrm{h}}$space(s) subseteq dspace(s$^{3/2}$). *J. Comput. Syst. Sci.*, 58(2):376–403, 1999. doi:10.1006/jcss.1998.1616.

30 Walter J. Savitch. Relationships between nondeterministic and deterministic tape complexities. *J. Comput. Syst. Sci.*, 4(2):177–192, 1970. doi:10.1016/S0022-0000(70)80006-X.

31 Thomas Steinke. Pseudorandomness for permutation branching programs without the group theory. *Electronic Colloquium on Computational Complexity (ECCC)*, 19:83, 2012. URL: http://eccc.hpi-web.de/report/2012/083.

32 Salil P. Vadhan. Pseudorandomness. *Foundations and Trends in Theoretical Computer Science*, 7(1-3):1–336, 2012. doi:10.1561/0400000010.

33 David Zuckerman. Randomness-optimal oblivious sampling. *Random Struct. Algorithms*, 11(4):345–367, 1997. doi:10.1002/(SICI)1098-2418(199712)11:4<345::AID-RSA4>3.0.CO;2-Z.

A Using PRPDs in the Saks-Zhou Scheme

In this section, we will briefly introduce Saks and Zhou's proof for $\mathbf{BPL} \subseteq \mathbf{L}^{3/2}$ [29] and Armoni's trick for replacing Nisan's PRG with any PRG in this proof [1]. Then we will see why a poly(nw/ε)-bounded PRPD suffices for this scheme. Since our purpose here is to go over the possible difference between using PRGs and PRPDs in this scheme, we will only include a sketch of Saks and Zhou's proof. We recommend interested readers to check [29, 1] for formal proofs and also [14] for a beautiful summary.

A.1 Saks and Zhou's Scheme

It is well-known that derandomizing **BPL** can be reduced to approximating M^n where M is any $n \times n$ stochastic matrix. The first step of Saks and Zhou is to turn M^n into the following recursive computation:

▶ **Fact 1.** *Let n_1, n_2 be integers such that $n_1^{n_2} = n$. Define $M_0 = M$, and $M_i = M_{i-1}^{n_1}$ for every positive integer i. Then $M_{n_2} = M^n$.*

To approximate M_{n_2}, it suffices to compute $M_i = M_{i-1}^{n_1}$ with small enough error in each step. However, if we need s bits of space to approximate the n_1-th power of a stochastic matrix, we will need $O(sn_2)$ bits of space in total. This doesn't really save any space (over approximating M^n directly) if we approximate $M_{i-1}^{n_1}$ with PRGs such as Nisan's generators. The first idea of Saks and Zhou is to utilize the "high probability" property of Nisan's generator:

▶ **Lemma 38** ([23]). *For every n, w, ε there exists an algorithm $\widehat{\mathrm{Pow}}_n$ which takes a $w \times w$ (sub)stochastic matrix M and a string $y \in \{0,1\}^{O(\log n \log(nw/\varepsilon))}$ as input, and outputs a $w \times w$ matrix such that*

$$\Pr_y\left[\left\|\widehat{\mathrm{Pow}}_n(M,y) - M^n\right\|_{\max} < \varepsilon\right] \geq 1-\varepsilon$$

in space $O(log(nw/\varepsilon))$.

In other word, derandomization with Nisan's generator has the following structure. First it fixes an "offline randomness" $y \in \{0,1\}^r$ and considers it as a part of input. Then it takes s bits of additional "processing space" to compute an approximation of M^n. Then the output will be a good approximation with high probability over y. (This is called an "offline randomized algorithm" in [29].) Furthermore $s \ll r$. With these properties, the main idea of Saks and Zhou is to reuse the same offline randomness for each level of recursion. If computing M^{n_1} takes r bits of offline randomness and s bits of processing space, then computing M^n will take r bits of offline randomness and $O(sn_2)$ bits of processing space. The space complexity would be $O(r + sn_2)$ which is better than approximating M^n directly since the offline randomness part was the original bottleneck.

However, there's a problem in this construction: if we compute $\widehat{M_1} = \widehat{\mathrm{Pow}}_{n_1}(M,y)$, and try to use $\widehat{\mathrm{Pow}}_{n_1}(\widehat{M_1},y)$ to approximate $M_2 = M_1^{n_1}$, it might be possible that $\widehat{\mathrm{Pow}}_{n_1}(\widehat{M_1},y)$ is always a bad approximation because $\widehat{M_1}$ depends on y. To resolve this issue, the second idea of Saks and Zhou is to break the dependency with a randomized rounding operation. We will borrow the name "snap" from [14] for this operation.

▶ **Definition 39.** *Given value $x \in \mathbb{R}$, string $y \in \{0,1\}^d$, define*

$$\mathrm{Snap}_d(x,y) = \max(\lfloor x \cdot 2^d - 2^{-d}y \rfloor \cdot 2^{-d}, 0).$$

For a $w \times w$ matrix M, define $\mathrm{Snap}_d(M,y)$ to be the matrix M' such that $M'_{i,j} = \mathrm{Snap}_d(M_{i,j},y)$ for every $i,j \in [w]$.

In other word, in a snap operation, we randomly perturb the matrix with a offset in $[0, 2^{-2d}]$, then round the entries down to d bits of precision. It's not hard to prove the following lemma:

▶ **Lemma 40** ([29]). *For any matrix M, M' such that $\|M - M'\|_{\max} \leq \varepsilon$,*

$$\Pr_y[\mathrm{Snap}_d(M,y) \neq \mathrm{Snap}_d(M',y)] \leq w^2(2^d\varepsilon + 2^{-d}).$$

Proof. The snap operation is equivalent to randomly choose a grid of length 2^{-d} and round each value to the closest grid point the left. Therefore two values a, b are rounded to different points only if there is a grid point between them, which happens with probability at most $2^d|a-b| + 2^{-d}$. By union bound and the fact that $\|M - M'\|_{\max} \leq \varepsilon$ the lemma follows. ◀

With the lemma above, we can see that by taking $\widehat{M_1} = \mathrm{Snap}_d(\widehat{\mathrm{Pow}}_{n_1}(M, y), z)$ instead, $\widehat{M_1}$ will be equivalent to $\mathrm{Snap}_d(M^{n_1}, z)$ with high probability, which is independent of y. Therefore we can use y as the offline randomness to compute the n_1-th power of $\widehat{M_1}$. Moreover, if the rounding precision is high enough, the snapped matrix is still a good approximation. Finally we get Saks-Zhou theorem:

▶ **Lemma 41** ([29]). *Let n_1, n_2 be integers such that $n_1^{n_2} = n$. Suppose there exists an offline randomized algorithm $\widehat{\mathrm{Pow}}_{n_1}$ which takes r bits of randomness and s bits of processing space such that for every substochastic matrix M,*

$$\Pr_x \left[\left\| \widehat{\mathrm{Pow}}_{n_1}(M, y) - M^{n_1} \right\|_{\max} \leq \varepsilon \right] \geq 1 - \varepsilon.$$

Now consider uniform random bits $y \in \{0,1\}^r$ and $z_1, z_2, \ldots, z_{n_2} \in \{0,1\}^d$. Let $\widehat{M_0} = M$, and $\widehat{M_i} = \mathrm{Snap}_d(\widehat{\mathrm{Pow}}_{n_1}(\widehat{M_{i-1}}, y), z_i)$ for every $i \in [n_2]$. Then with probability at least $1 - O(w^2 n_2(2^d\varepsilon + 2^{-d}))$ over $y, z_1, \ldots, z_{n_2}$,

$$\left\| \widehat{M_{n_2}} - M^n \right\| \leq nw2^{-d+1}.$$

Moreover, the space complexity of computing $\widehat{M_{n_2}}$ is $O(r + n_2(s + d))$.

Proof sketch. Define $\overline{M_0} = M$, $\overline{M_i} = \mathrm{Snap}((\overline{M_{i-1}})^{n_1}, z_i)$. By union bound, the following events happen simultaneously with probability $1 - O(w^2 n_2(2^d\varepsilon + 2^{-d}))$:

1. For every $i \in [n_2]$, $\left\| \widehat{\mathrm{Pow}}_{n_1}(\overline{M_{i-1}}, y) - (\overline{M_{i-1}})^{n_1} \right\|_{\max} \leq \varepsilon$.
2. For every $i \in [n_2]$, conditioned on $\widehat{M_{i-1}} = \overline{M_{i-1}}$ and $\left\| \widehat{\mathrm{Pow}}_{n_1}(\overline{M_{i-1}}, y) - \overline{M_{i-1}}^{n_1} \right\|_{\max} \leq \varepsilon$, $\widehat{M_i} = \overline{M_i}$.

When the above events occur, we have $\widehat{M_{n_2}} = \overline{M_{n_2}}$. Moreover, note that for every $i \in [n_2]$

$$\left\| \overline{M_i} - (\overline{M_{i-1}})^{n_1} \right\|_{\max} \leq 2^{-d+1}.$$

To see why this is true, observe that in a snap operation we change the given value by at most 2^{-2d} from perturbation and 2^{-d} from rounding. [2] This implies

$$\left\| \overline{M_i} - (\overline{M_{i-1}})^{n_1} \right\| \leq 2^{-d+1},$$

where $\|\cdot\|$ denotes the matrix infinity norm. By Lemma 5.4 in [29],

$$\left\| \overline{M_{n_2}} - M^n \right\| \leq nw2^{-d+1}.$$

For the space complexity, observe that we can compute $\widehat{M_{n_2}}$ with n_2 levels of recursive calls, and each recursive call takes $O(s + d)$ bits. Moreover, we need r bits to store the offline randomness. Therefore the space complexity is $O(n_2(s + d) + r)$ ◀

If we take $n_2 = \sqrt{\log n}$, $n_1 = 2^{\sqrt{\log n}}$, $d = O(\log(n))$ and $\varepsilon = 2^{-2d}$ and plugging in Nisan's generator, the above lemma shows that $\mathbf{BPL} \subseteq \mathbf{L}^{3/2}$.

[2] Note that capping the lowest possible value to be 0 can only reduce the error, because the snapped value was non-negative.

A.2 Armoni's Trick

We saw that in Saks and Zhou's proof, we need a "offline randomized algorithm" for substochastic matrix exponentiation such that when given r bits of randomness as additional input, the algorithm only requires additional $s \ll r$ bits of space to compute a good approximation with high probability. This is in fact the only place where we need PRGs in Saks and Zhou's proof. However, not every PRG has such property, so it might be hard to tell whether an improvement over Nisan's PRG will actually give a better derandomization for **BPL**. Fortunately, Armoni [1] observed that one can turn any PRG into a derandomization algorithm with the required property by simply composing the PRG with an averaging sampler.

Before we go through Armoni's claim, first we generalize Lemma 41 for a larger class of algorithms $\widehat{\mathrm{Pow}}$.

▶ **Definition 42.** *We say an offline randomized algorithm requires s bits of* sensitive processing space *and t bits of* reusable processing space *if*

- *During the execution of this algorithm, only t bits of processing space is required.*
- *Before each time a bit is read from the real input (not including the offline randomness), only s bits of processing space is being used at the time.*

In the above definition, think of each input bit as generated from a recursive call. Thus the "reusable processing space" can be interpreted as "recursion-friendly processing space", which can be erased before every recursive call. With this new definition we can generalize Lemma 41 as follows:

▶ **Lemma 43** ([29], generalized)**.** *Let n_1, n_2 be integers such that $n_1^{n_2} = n$. Suppose there exists an offline randomized algorithm $\widehat{\mathrm{Pow}}_{n_1}$ which takes r bits of randomness, s bits of sensitive processing space and t bits of reusable processing space, such that for every substochastic matrix M,*

$$\Pr_x\left[\left\|\widehat{\mathrm{Pow}}_{n_1}(M,y) - M^{n_1}\right\|_{\max} \le \varepsilon\right] \ge 1-\varepsilon.$$

Now consider uniform random bits $y \in \{0,1\}^r$ and $z_1, z_2, \ldots, z_{n_2} \in \{0,1\}^d$. Let $\widehat{M_0} = M$, and $\widehat{M_i} = \mathrm{Snap}_d(\widehat{\mathrm{Pow}}_{n_1}(\widehat{M_{i-1}}, y), z_i)$ for every $i \in [n_2]$. Then with probability at least $1 - O(w^2 n_2(2^d\varepsilon + 2^{-d}))$ over $y, z_1, \ldots, z_{n_2}$,

$$\left\|\widehat{M_{n_2}} - M^n\right\| \le nw2^{-d+1}.$$

Moreover, the space complexity of computing $\widehat{M_{n_2}}$ is $O(r + t + n_2(s+d))$.

We omit the proof because it's Exactly the same as Lemma 41.

For technicality, we also need to define a ROBP with larger "step size".

▶ **Definition 44.** *A (n,w,d)-ROBP is a ROBP of n layers, w nodes in each layer, and 2^d branches from each node.*

That is, a (n,w,d)-ROBP is a ROBP which can read d bits at once. Note that derandomizing (n,w,d)-ROBP corresponds to derandomizing the exponentiation of a stochastic matrix which has d bits of precision in each entry.

Now we are ready to introduce Armoni's Lemma.

▶ **Lemma 45** ([1])**.** *Suppose there exists an explicit PRG for $(n, w+1, \log(3nw/\varepsilon))$-ROBP with error $\varepsilon/3$ which has seed length s. Then there exists an offline randomized algorithm which approximates the n-th power of any substochastic matrix within error ε with probability*

at least $1-\varepsilon$*. Moreover, such algorithm requires* $s+O(\log(w/\varepsilon))$ *bits of randomness,* $O(s+O(\log(w/\varepsilon)))$ *bits of reusable processing space and* $O(\log(nw/\varepsilon))$ *bits of sensitive processing space.*

Proof. Given an input M, first we round each entry down to $d=\log(3nw/\varepsilon)$ bits of precision. Then we will get a substochastic matrix M' such that each entry of M' is a multiple of 2^{-d}, and $\|M-M'\|_{\max}\le\varepsilon/3nw$. Then we have

$$\|M^n-(M')^n\|_{\max}\le\|M^n-(M')^n\|\le n\,\|M-M'\|\le nw\,\|M-M'\|_{\max}\le\frac{\varepsilon}{3}.$$

Then we construct a $(n,w+1,d)$-ROBP B as follows. For each $t\in[n]$, we connect k edges from node $(t-1,i)$ to node (t,j) if $M'_{i,j}=k\cdot 2^{-d}$. Then for each node $(t-1,i)$ which doesn't have 2^d outgoing edges yet, we connect more edges from $(t-1,i)$ to a dummy node $(t,w+1)$. For each dummy node we connect 2^d edges to the dummy node in the next layers. It is easy to observe that $(M'^n)_{i,j}$ is exactly the probability that we start a random walk from $(0,i)$ and reach (n,j). Now for every $i,j\in[w]$, define $B_{i,j}(x)$ to be the indicator for whether we will reach (t,j) if we start from $(0,i)$ and follow $x\in(\{0,1\}^d)^n$. Then $\mathbb{E}_x\left[B_{i,j}(x)\right]=(M'^n)_{i.j}$. Take the given PRG G, we have

$$\left|\mathop{\mathbb{E}}_r\left[B_{i,j}(G(r))\right]-\mathop{\mathbb{E}}_x\left[B_{i,j}(x)\right]\right|\le\frac{\varepsilon}{3}.$$

Now define the offline randomized algorithm $\widehat{\mathrm{Pow}}$ to be

$$\widehat{\mathrm{Pow}}(M,y)_{i,j}=\mathop{\mathbb{E}}_z\left[B_{i,j}(G(\mathrm{Samp}(y,z)))\right],$$

where Samp is a $(\varepsilon/3,\varepsilon/w^2)$-sampler. By definition of sampler, with probability at least $(1-(\varepsilon/w^2))$ over the choice of y, we have

$$\left|\widehat{\mathrm{Pow}}(M,y)_{i,j}-\mathop{\mathbb{E}}_r\left[B_{i,j}(G(r))\right]\right|\le\frac{\varepsilon}{3}.$$

By union bound, with probability at least $(1-\varepsilon)$,

$$\left\|\widehat{\mathrm{Pow}}(M,y)_{i,j}-\mathop{\mathbb{E}}_r\left[B_{i,j}(G(r))\right]\right\|_{\max}\le\frac{\varepsilon}{3}$$

for every $i,j\in[w]$. Therefore by triangle inequality we have

$$\left\|\widehat{\mathrm{Pow}}(M,y)-M^n\right\|_{\max}\le\varepsilon$$

with probability at least $1-\varepsilon$.

Finally we compute the complexity of $\widehat{\mathrm{Pow}}$. By Lemma 19, the required randomness in this offline randomized algorithm is $s+O(\log(1/\varepsilon)+\log\log(w/\varepsilon))$. The required processing space is the processing space for samplers and PRGs. Observe that the only sensitive data is the second input for sampler (i.e. z); the current node in the ROBP, which takes $\log(nw)$ bits to store; and a d-bit block in $G(\mathrm{Samp}(y,z))$ indicating which entry of M we should check. Therefore the required sensitive processing space is only $O(\log(nw/\varepsilon))$ bits. ◀

With Armoni's sampler trick, if we have any PRG for $(n,w+1,\log(3nw/\varepsilon))$-ROBP, we can always plug it into the Saks-Zhou scheme regardless of whether it has the high-probability property. Specifically, as suggested in [4], if we have a PRG of seed length $O(\log^2(n)+\log^{4/3}(w/\varepsilon))$, we can even prove that $\mathbf{BPL}\subseteq\mathbf{L}^{4/3}$.

A.3 Saks-Zhou-Armoni Scheme with PRPDs

Finally we see how to apply a PRPD in the above scheme.

▶ **Lemma 46.** *Suppose there exists an explicit* $\mathrm{poly}(nw/\varepsilon)$*-bounded PRPD* (G, ρ) *for* $(n, w+1, \log(3nw/\varepsilon))$*-ROBP with error* $\varepsilon/3$ *which has seed length* s*. Then there exists an offline randomized algorithm which approximates the* n*-th power of any substochastic matrix within error* ε *with probability at least* $1-\varepsilon$*. Moreover, such algorithm requires* $s + O(\log(w/\varepsilon))$ *bits of randomness,* $O(s + O(\log(w/\varepsilon)))$ *bits of reusable processing space and* $O(\log(nw/\varepsilon))$ *bits of sensitive processing space.*

Proof. The proof is basically the same as Lemma 45, with the following two difference.

- $\widehat{\mathrm{Pow}}(M, y)_{i,j}$ is defined as $\mathbb{E}_z\left[\rho(\mathrm{Samp}(y, z)) \cdot B_{i,j}(G(\mathrm{Samp}(y, z)))\right]$ instead.
- If (G, ρ) is k-bounded, then we will choose Samp as a $(\varepsilon/6k, \varepsilon/w^2)$ sampler instead.

It's not hard to verify the correctness. (With Claim 18 which shows that samplers can also be used for functions with output range $[-k, k]$.) The required sensitive processing space is increased to $O(\log(nw/\varepsilon) + \log(k))$, which is still $O(\log(nw/\varepsilon))$ if $k = \mathrm{poly}(nw/\varepsilon)$. ◀

One may notice that there might have negative output in our new definition of $\widehat{\mathrm{Pow}}$. However, this is not a problem when applying Saks-Zhou argument because we only rely on the non-negativeness of matrices $\overline{M_i}$, which is independent of the approximation algorithm we use. With the above lemma we have the following corollary, which better motivates the problem of getting improved seed length for PRPDs:

▶ **Corollary 47.** *If there exists a* $\mathrm{poly}(nw/\varepsilon)$*-bounded explicit PRPD for* (n, w, d)*-ROBP with error* ε *which has seed length* $O(\log^2(n) + (\log(w/\varepsilon) + d)^{4/3})$*, then* $\mathbf{BPL} \subseteq \mathbf{L}^{4/3}$*.*

Proof. Apply the Saks-Zhou scheme (Lemma 43), and take $n_1 = 2^{\log^{2/3}(n)}$, $n_2 = \log^{1/3}(n)$, $d = 10\log(n)$ and $\varepsilon = 2^{-2d}$. The required subprocedure $\widehat{\mathrm{Pow}}$ would be approximating the n_1-th power of $n \times n$ substochastic matrices within error ε. By Lemma 46, there exists an offline randomized algorithm which approximates M^{n_1} within error $\varepsilon = 2^{-2d} = \mathrm{poly}(1/n)$, which requires sensitive processing space $O(\log(n))$ and offline randomness + reusable processing space $O(\log^2(n_1) + \log^{4/3} n) = O(\log^{4/3}(n))$. Therefore the total space complexity is $O(\log(n) \cdot n_2 + \log^{4/3}(n)) = O(\log^{4/3}(n))$. ◀

▶ Remark 48. Note that while we only construct PRPDs for (n, w)-ROBP in this paper, it is possible to adapt our construction to get PRPDs for (n, w, d)-ROBP with seed length $O(\log n \log(nw) \log\log(nw) + \log(1/\varepsilon) + d)$: simply replace the base case with a sampler with d-bit output. Since it doesn't imply better derandomization for **BPL** anyway, we keep $d = 1$ for simplicity.

B Proof of Lemma 19

▶ **Lemma 49** (Lemma 19, restated. [27, 10])**.** *For every* $\delta, \varepsilon > 0$ *and integer* m*, there exists a* (ε, δ)*-sampler* $f : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ *s.t.* $d = O(\log\log(1/\delta) + \log(1/\varepsilon))$ *and* $n = m + O(\log(1/\delta)) + O(\log(1/\varepsilon))$*. Moreover, for every* x, y*,* $f(x, y)$ *can be computed in space* $O(m + \log(1/\delta) + \log(1/\varepsilon))$*.*

We will use the equivalence between seeded randomness extractor and oblivious sampler by Zuckerman [33]. To achieve the parameter we need, we need a "high-entropy seeded extractor" such that the seed length only depends on entropy loss but not the length of source. We will use the standard "block-source" construction for high-entropy extractor

which can be found in [11, 27]. For simplicity, we will use simple composition instead of zig-zag composition [27] because we are not aiming for optimal entropy loss. We will use the following standard lemmas for the extractor construction. Some of the following lemmas are implicit in their original source, and we recommend the readers to see [12, 32] for a proof.

▶ **Definition 50** ([8]). *(X_1, X_2) is a (k_1, k_2)-*block source *if X_1 is a k_1-source, and for every $x_1 \in \mathrm{Supp}(X)$, X_2 conditioned on $X_1 = x_1$ is a k_2-source.*

▶ **Lemma 51** ([11]). *Let $X \in \{0,1\}^n$ be a $(n - \Delta)$ source. Then for every integer $0 \le t \le n$, X is ε-close to a $(t - \Delta, n - t - \Delta - \log(1/\varepsilon))$-block source (X_1, X_2) where $X_1 \in \{0,1\}^t$ and $X_2 \in \{0,1\}^{n-t}$.*

▶ **Lemma 52** ([25]). *Let $E_1 : \{0,1\}^{n_1} \times \{0,1\}^d \to \{0,1\}^{d_2}$ be a (k_1, ε_1) extractor and $E_2 : \{0,1\}^{n_2} \times \{0,1\}^{d_2} \to \{0,1\}^m$ be a (k_2, ε_2) extractor. Define $E((x_1, x_2), s) = E_1(x_2, E_2(x_1, s))$. Then for every (k_1, k_2)-block source $(X_1, X_2) \in \{0,1\}^{n_1} \times \{0,1\}^{n_2}$, $E((X_1, X_2), U_d)$ is $(\varepsilon_1 + \varepsilon_2)$-close to uniform.*

▶ **Lemma 53** ([11]). *For every $\varepsilon, \Delta > 0$ and integer n there exists a $(n - \Delta, \varepsilon)$ extractor $E : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^n$ with $d = O(\Delta + \log(1/\varepsilon))$, and for every x, y, $E(x, y)$ can be computed in space $O(n + \log(1/\varepsilon))$.*

▶ **Lemma 54** ([12, 19]). *For every $\varepsilon > 0$, integer $m > 0$ and $n \ge 2m$, there exists a $(2m, \varepsilon)$ extractor $E : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ with $d = O(\log m + \log(1/\varepsilon))$, and for every x, y, $E(x, y)$ can be computed in space $O(m + \log(1/\varepsilon))$.*

▶ **Lemma 55** ([33]). *Every $(n - \log(1/\delta) - 1, \varepsilon)$-extractor is a (ε, δ)-sampler.*

Now we show how to construct the sampler we need, and that it is indeed space efficient.

Proof. Let $\Delta = \log(1/\delta) + 1$. Let $E_1 : \{0,1\}^m \times \{0,1\}^{d_1} \to \{0,1\}^m$ be an $(m - \Delta, \varepsilon/3)$-extractor from Lemma 53, w.l.o.g. assume that $d_1 \ge \Delta + \log(3/\varepsilon)$. Then let $E_2 : \{0,1\}^{3d_1} \times \{0,1\}^d \to \{0,1\}^{d_1}$ be an $(2d_1, \varepsilon/3)$-extractor from Lemma 54. Then we claim that $E : \{0,1\}^{m+3d_1} \times \{0,1\}^d \to \{0,1\}^m$, defined as $E((x_1, x_2), s) = E_1(x_1, E_2(x_2, s))$, is a $(m + 3d_1 - \Delta, \varepsilon)$ extractor, and hence a (ε, δ) sampler by Lemma 55.

To prove the claim, consider any $(m + 3d_1 - \Delta)$-source X. By Lemma 51, X is $(\varepsilon/3)$-close to a $(m - \Delta, 3d_1 - \Delta - \log(3/\varepsilon))$-block source $(X_1, X_2) \in \{0,1\}^{3d_1} \times \{0,1\}^m$. By Lemma 52, $E_1(X_1, E_2(X_2, U_d))$ is $2\varepsilon/3$-close to uniform. Since $E(X, U_d)$ is $\varepsilon/3$-close to $E_1(X_1, E_2(X_2, U_d))$, by triangle inequality it is ε-close to uniform. Moreover, $d = O(\log(d_1/\varepsilon)) = O(\log\log(1/\delta) + \log(1/\varepsilon))$, $n = m + 3d_1 = m + O(\log(1/\delta) + \log(1/\varepsilon))$, and the required space to compute E is $O(m + d_1 + \log(1/\varepsilon)) = O(m + \log(1/\varepsilon) + \log(1/\delta))$. ◀