Electronic Colloquium on Computational Complexity, Report No. 121 (2022)

# Recent Progress on Derandomizing Space-Bounded Computation

William M. Hoza*
Simons Institute for the Theory of Computing
University of California, Berkeley
williamhoza@berkeley.edu

**Abstract**

Is randomness ever necessary for space-efficient computation? It is commonly conjectured that $\mathsf{L} = \mathsf{BPL}$, meaning that halting decision algorithms can always be derandomized without increasing their space complexity by more than a constant factor. In the past few years (say, from 2017 to 2022), there has been some exciting progress toward proving this conjecture. Thanks to recent work, we have new pseudorandom generators (PRGs), new black-box derandomization algorithms (generalizations of PRGs), and new non-black-box derandomization algorithms. This article is a survey of these recent developments. We organize the underlying techniques into four overlapping themes:

1. The *iterated pseudorandom restrictions* framework for designing PRGs, especially PRGs for functions computable by arbitrary-order read-once branching programs.
2. The *inverse Laplacian* perspective on derandomizing $\mathsf{BPL}$ and the related concept of local consistency.
3. *Error reduction* procedures, including methods of designing low-error weighted pseudorandom generators (WPRGs).
4. The continued use of *spectral expander graphs* in this domain via the derandomized square operation and the Impagliazzo-Nisan-Wigderson PRG (STOC 1994).

We give an overview of these ideas and their applications, and we discuss the challenges ahead.

*Part of this work was done while the author was visiting the Simons Institute for the Theory of Computing. This article is based on a presentation that the author first gave at Oberwolfach Workshop 2146 on Complexity Theory.



ISSN 1433-8092

# 1 Introduction

In an effort to solve problems as efficiently as possible, algorithm designers often introduce randomness into their algorithms. This paradigm is undoubtedly ingenious and beautiful. However, random bits can themselves be considered a computational "resource" that might be costly or unavailable. At best, randomization trades one type of inefficiency for another. We therefore want to distinguish between cases in which randomization gives an intrinsic advantage and cases in which algorithms can be derandomized with little to no penalty. In this article, we focus on the question of how randomization affects *space complexity*.

## 1.1 Randomized Space-Bounded Computation

Informally, $\mathsf{BPSPACE}(S)$ is everything that can be decided using randomness and $O(S)$ bits of space. More precisely, for a function $S\colon \mathbb{N} \to \mathbb{N}$, a language $L$ is in $\mathsf{BPSPACE}(S)$ if there exists a Turing machine $A$ with the following features.

1. The machine $A$ has three tapes: a read-only input tape, a read-write work tape, and a read-once "random tape" that is initially filled with uniform random bits.

2. For every $N \in \mathbb{N}$,[1] every input $\sigma \in \{0,1\}^N$, and every assignment to the random tape $x \in \{0,1\}^\infty$, the machine touches at most $O(S(N))$ cells of the work tape and eventually halts, outputting a Boolean value $A(\sigma, x) \in \{0,1\}$.

3. For every input $\sigma \in \{0,1\}^*$, we have

$$\sigma \in L \implies \Pr_x[A(\sigma,x)=1] \geq 2/3$$
$$\sigma \notin L \implies \Pr_x[A(\sigma,x)=1] \leq 1/3.$$

Let us assume that $S \geq \log N$, so the machine has enough space to store a pointer to an arbitrary location in its input. Note that we assume that the algorithm halts for *every* assignment to the random tape (not merely with high probability). One can show that it follows that the algorithm halts within $2^{O(S)}$ steps.[2] We use $\mathsf{BPL}$ to denote $\mathsf{BPSPACE}(\log N)$. The classes $\mathsf{RSPACE}(S)$ and $\mathsf{RL}$ are defined the same way, except that we only allow one-sided error.

These models were first studied by Aleliunas, Karp, Lipton, Lovász, and Rackoff [AKLLR79] more than four decades ago. They presented a randomized algorithm showing that the undirected connectivity problem is in $\mathsf{RL}$, and they asked whether $\mathsf{L} = \mathsf{RL}$. Today, thanks to Reingold's famous algorithm [Rei08], we know that the specific problem of undirected connectivity is indeed in $\mathsf{L}$. (Reingold's work [Rei08] is the climax of a long sequence of papers studying the space complexity of the undirected connectivity problem [BCDRT89; BR97; NSW92; ATWZ00; Tri08; Rei08; RV05].) It is commonly believed that more generally $\mathsf{L} = \mathsf{RL} = \mathsf{BPL}$. By a padding argument, if $\mathsf{L} = \mathsf{BPL}$, then $\mathsf{DSPACE}(S) = \mathsf{BPSPACE}(S)$ for every space-constructible $S \geq \log N$.

Superficially, this sounds like the same frustrating story that pervades complexity theory. "We have been studying these important complexity classes for many decades, and at this point we think

[1] In this article, we use uppercase $N$ to denote the length of the input to a space-bounded algorithm. We use lowercase $n$ to denote the number of random bits that the algorithm uses.

[2] Historically, there was more early interest in the alternative "non-halting" model in which we merely require the algorithm to halt with high probability [Gil77; Sim77; Sim81; Jun81; BCP83; Mic92; Sak96]. Indeed, in the older literature, notation along the lines of "$\mathsf{BPSPACE}(S)$" typically refers to the non-halting model, whereas the halting model is discussed using augmented notation such as "$\mathsf{BP_H SPACE}(S)$." Today, the halting model is standard.

we know the relationship between them, but we don't know how to prove it." The same can be said regarding $\mathsf{P}$ vs. $\mathsf{NP}$, or $\mathsf{P}$ vs. $\mathsf{BPP}$, or $\mathsf{L}$ vs. $\mathsf{P}$, or countless other fundamental problems.

However, there is a widespread feeling that *the* $\mathsf{L}$ *vs.* $\mathsf{BPL}$ *problem is different.* Compared to (say) the problem of proving $\mathsf{P} = \mathsf{BPP}$, there is a great deal of *optimism* about the possibility of unconditionally proving $\mathsf{L} = \mathsf{BPL}$. This optimism is sensible because the $\mathsf{BPL}$ model has a crucial weakness: the read-once random tape.

### 1.2 The Read-Once Assumption

In the definition of $\mathsf{BPL}$, the machine $A$ is only permitted to read each cell of the random tape a single time; the tape head can move right but not left. The motivation for this assumption is that we are modeling a problem-solving agent who has access to a single fair coin. The agent can see the outcome of the most recent coin flip, but if they want to know the outcome of a previous coin flip, they ought to have written it down at the time that it occurred (and paid for it in terms of space complexity).

As a consequence of the read-once assumption, the action of $A$ on its random bits can be modeled by a polynomial-width *read-once branching program* (ROBP), defined below.

**Definition 1** (ROBPs)**.** A *width-$w$ length-$n$ ROBP $f$* is defined by a start state $v_0 \in [w]$, a sequence of $n$ transition functions $f_1, \dots, f_n \colon [w] \times \{0,1\} \to [w]$, and a set of accepting states $V_{\text{acc}} \subseteq [w]$. An input $x \in \{0,1\}^n$ determines a sequence of states $v_0, v_1, \dots, v_n \in [w]$ by the rule $v_i = f_i(v_{i-1}, x_i)$ for $i > 0$. The output of the program is given by

$$f(x) = \begin{cases} 1 & \text{if } v_n \in V_{\text{acc}} \\ 0 & \text{if } v_n \notin V_{\text{acc}}. \end{cases} \tag{1}$$

Equivalently, we can think of $f$ as a directed graph with vertices arranged in $n+1$ layers, $V_0, \dots, V_n$, where $|V_i| = w$. For $i < n$, each vertex $u \in V_i$ has two outgoing edges leading to $V_{i+1}$, one labeled 0 and the other labeled 1. There is a designated start vertex $v_0 \in V_0$, and there is a set of designated accepting vertices $V_{\text{acc}} \subseteq V_n$. An input $x \in \{0,1\}^n$ is interpreted as a sequence of edge labels, identifying a path $(v_0, v_1, \dots, v_n) \in V_0 \times V_1 \times \cdots \times V_n$. The output of the program is once again given by Eq. (1).

(Note that in the context of derandomizing space-bounded computation, we assume by default that a "read-once branching program" reads its input bits in the standard order $x_1, x_2, \dots, x_n$.) If $A$ is a randomized, halting log-space algorithm and $\sigma$ is an input of length $N$, then the function $f(x) \stackrel{\text{def}}{=} A(\sigma, x)$ can be computed by a width-$n$ length-$n$ ROBP for a suitable value $n = \text{poly}(N)$; each state the program $f$ encodes a configuration of the machine $A$. An appealing approach to derandomizing $A$ is to design a *pseudorandom generator* (PRG) that fools ROBPs.

**Definition 2** (PRGs)**.** Let $\mathcal{F}$ be a class of functions $f \colon \{0,1\}^n \to \{0,1\}$, let $X$ be a distribution over $\{0,1\}^n$, and let $\varepsilon > 0$. We say that $X$ *fools $\mathcal{F}$ with error $\varepsilon$* if for every $f \in \mathcal{F}$,

$$|\Pr[f(X) = 1] - \Pr[f(U_n) = 1]| \leq \varepsilon,$$

where $U_n$ denotes the uniform distribution over $\{0,1\}^n$. An *$\varepsilon$-PRG* for $\mathcal{F}$ is a function $G \colon \{0,1\}^s \to \{0,1\}^n$ such that $G(U_s)$ fools $\mathcal{F}$ with error $\varepsilon$. The value $s$ is called the *seed length* of $G$.

If we could construct a PRG $G$ that 0.1-fools width-$n$ length-$n$ ROBPs with seed length $O(\log n)$ and space complexity $O(\log n)$, then we could conclude that $\mathsf{L} = \mathsf{BPL}$, because we could deterministically estimate the acceptance probability of an algorithm $A$ on an input $\sigma$ to within $\pm 0.1$ by computing $A(\sigma, G(x))$ for every seed $x$.

### 1.2.1 PRGs and Lower Bounds

Some readers might have an intuition that says that designing unconditional PRGs is hopelessly difficult. This intuition is indeed sensible in many contexts. For example, consider the problem of designing a PRG that fools size-$n$ *multiple-read* branching programs, i.e., programs that may read their input bits any number of times. Such a PRG $G\colon \{0,1\}^s \to \{0,1\}^n$ would induce a corresponding "hard function" $h\colon \{0,1\}^{s+1} \to \{0,1\}$ that cannot be computed by size-$n$ branching programs.[3] If $G$ is computable in space $O(s)$, then so is $h$. Therefore, the problem of designing explicit PRGs for multiple-read branching programs is even harder than the problem of proving *branching program lower bounds* for explicit functions. Perhaps someday our grandchildren will manage to prove optimal lower bounds for branching programs, but until that day, we should probably consider optimal PRGs for read-many branching programs to be out of reach.[4]

The good news is that the *read-once* assumption is an absolute game-changer. In the read-once setting, optimal lower bounds are already known. For example, using standard communication complexity arguments, one can show that every ROBP computing the function

$$h(x_1, \dots, x_{2n}) = x_1 x_{n+1} \oplus x_2 x_{n+2} \oplus \cdots \oplus x_n x_{2n}$$

has width $2^{\Omega(n)}$. To design optimal PRGs for ROBPs, we "merely" need to bridge the gap between lower bounds and PRGs.[5] There is no clear "barrier" preventing us from designing optimal PRGs for ROBPs. This is one of the reasons that a proof that $\mathsf{L} = \mathsf{BPL}$ seems vastly more attainable than, say, a proof that $\mathsf{P} = \mathsf{BPP}$.

### 1.2.2 Nisan's PRG and Beyond

So far, we do have several explicit PRGs that unconditionally fool ROBPs, but they do not achieve the optimal seed length. Most famously, Nisan designed an explicit PRG that $\varepsilon$-fools width-$w$ length-$n$ ROBPs with seed length $O(\log(wn/\varepsilon) \cdot \log n)$ [Nis92]. The optimal seed length would be $\Theta(\log(wn/\varepsilon))$.

Admittedly, at this point it has been over three decades since Nisan's work [Nis92], and we still do not have explicit PRGs for polynomial-width ROBPs with seed length better than Nisan's $O(\log^2 n)$ bound. However, an extensive body of research on the $\mathsf{L}$ vs. $\mathsf{BPL}$ problem has shown how to "go beyond" Nisan's work [Nis92] in one sense or another. This rich and sophisticated literature is full of valuable insights that profoundly clarify the role of randomness in computing, even though the central questions remain open.

In the remainder of this article, we survey exciting progress that has been made on the $\mathsf{L}$ vs. $\mathsf{BPL}$ problem in just the past few years. (See Saks' survey [Sak96] for an overview of older work.) We structure our discussion around four recurring technical themes: the *iterated pseudorandom restrictions* framework (Section 2), the *inverse Laplacian* perspective (Section 3), *error reduction* procedures (Section 4), and *expander graphs* (Section 5).

[3] Specifically, $h$ is the indicator function of the set $\{G(x)_{1\dots s+1} : x \in \{0,1\}^s\}$.

[4] Currently, the best lower bound known is Nečiporuk's near-quadratic lower bound [Neč66]. Explicit PRGs for size-$n$ branching programs are known with a near-matching seed length of $\widetilde{O}(\sqrt{n})$ [IMZ19; HHTT22].

[5] Note that the Nisan-Wigderson reduction [NW94] does not work here, because it does not preserve the read-once property.

## 2 Iterated Pseudorandom Restrictions

### 2.1 Arbitrary-Order ROBPs

Nisan's classic PRG [Nis92] suffers from a strange weakness. It turns out that permuting the output bits does not, in general, preserve the pseudorandomness property [Tzu09]. In other words, define an *arbitrary-order ROBP* just like an ordinary ROBP (Definition 1), except that instead of reading the input bits in the standard order $x_1, x_2, \ldots, x_n$, it reads the input bits in the order $x_{\pi(1)}, x_{\pi(2)}, \ldots, x_{\pi(n)}$ for some permutation $\pi\colon [n] \to [n]$. Nisan's PRG does not fool arbitrary-order ROBPs [Tzu09].

An interesting line of work has shown how to construct alternative PRGs for ROBPs that work even in the arbitrary-order setting [BPW11; SVW17; CHRT18; MRT19; FK18]. We highlight a breakthrough paper by Forbes and Kelley [FK18]. Building on several earlier papers [RSV13; HLV18; CHRT18], Forbes and Kelley constructed two explicit PRGs for arbitrary-order ROBPs.

**Theorem 1** (PRGs for arbitrary-order ROBPs [FK18])**.** *For every $w, n \in \mathbb{N}$ and $\varepsilon > 0$, there exist explicit $\varepsilon$-PRGs for width-$w$ length-$n$ arbitrary-order ROBPs with seed lengths*

$$O(\log(wn/\varepsilon) \cdot \log^2 n) \tag{2}$$

*and*

$$\widetilde{O}(w \cdot \log(n/\varepsilon) \cdot \log n). \tag{3}$$

These seed lengths are only a little worse than Nisan's seed length [Nis92], yet the PRGs fool a more powerful model.

For our main application (derandomizing $\mathsf{BPL}$), it is no loss of generality to assume that the random bits are read in the standard order $x_1, x_2, x_3, \ldots$, so why study arbitrary-order ROBPs? One reason is that they capture other interesting models of computation such as read-once formulas [BPW11]. Another reason is that studying arbitrary-order ROBPs forces us to develop *new techniques* for fooling ROBPs. Indeed, the ideas underlying Forbes and Kelley's PRGs [FK18] are completely different than those underlying Nisan's PRG [Nis92]. Forbes and Kelley's PRGs [FK18] are based on the framework of *iterated pseudorandom restrictions* – our first "theme."

### 2.2 Forbes-Kelley Restrictions

Ajtai and Wigderson introduced the iterated restrictions framework in the context of pseudorandomness for $\mathsf{AC}^0$ circuits [AW89]. Much later, Gopalan, Meka, Reingold, Trevisan, and Vadhan brought the framework to the world of $\mathsf{L}$ vs. $\mathsf{BPL}$ [GMRTV12]. The idea is as follows. Our goal is to sample a string $X \in \{0,1\}^n$ that fools some function of interest $f\colon \{0,1\}^n \to \{0,1\}$. Our first step is to design a pseudorandom *restriction* $X \in \{0,1,\star\}^n$, i.e., we pseudorandomly assign values to a *pseudorandom subset* of the variables. We ensure that $X$ "preserves the expectation" of $f$, meaning that $X \circ U$ fools $f$, where $X \circ U$ denotes the string obtained by sampling $X$ and then replacing each $\star$ with a fresh truly random bit. Intuitively, designing such an $X$ is easier than designing a full PRG, because in the analysis, some helpful truly random bits ($U$) are sprinkled in among the pseudorandom bits.

After assigning values according to $X$, our remaining task is to fool the restricted function $f|_X$. Therefore, we repeat the process, i.e., we sample a restriction $X'$ that preserves the expectation of $f|_X$. Iterating in this way, we assign values to more and more variables. Eventually, we have assigned values to all the variables and hence we have a full PRG.

Forbes and Kelley's primary contribution is to show how to accomplish the first step, i.e., how to sample a pseudorandom restriction that assigns values to many variables while preserving the expectation of every bounded-width arbitrary-order ROBP [FK18]. Indeed, they prove the following.

**Theorem 2** (A pseudorandom restriction for ROBPs [FK18])**.** *Let $w, n \in \mathbb{N}$ and $\varepsilon > 0$, and let $k = 4\log(wn/\varepsilon)$. Let $D$ and $T$ be $k$-wise independent $n$-bit strings (with uniform marginals), let $U$ be uniform random over $\{0,1\}^n$, and assume that $D$, $T$, and $U$ are mutually independent. Then $D + T \wedge U$ fools width-$w$ length-$n$ ROBPs with error $\varepsilon$, where $+$ denotes bitwise* XOR *and $\wedge$ denotes bitwise* AND*.*

The distributions $D$ and $T$ define a restriction $X$ by letting $T$ indicate the $\star$ positions and using $D$ to assign values to the non-$\star$ positions. The statement that $X$ preserves the expectation of $f$ is equivalent to the statement that $D + T \wedge U$ fools $f$. The latter way of thinking can be called the "pseudorandomness plus noise" perspective [HLV18; LV20].

Theorem 2 is stated for ROBPs. It applies more generally to *arbitrary-order* ROBPs, because permuting variables preserves $k$-wise independence. Using standard constructions of $k$-wise independent random variables [Vad12], one can explicitly sample $D$ and $T$ using $O(k \log n) = O(\log(wn/\varepsilon) \cdot \log n)$ truly random bits. In expectation, each restriction assigns values to half of the living variables, so after roughly $\log n$ iterations, we should intuitively expect that all the variables have been assigned variables. With a little more care, one can indeed achieve an overall seed length of $O(\log(wn/\varepsilon) \cdot \log^2 n)$ (see Forbes and Kelley's work for details [FK18, Section 7]).

The proof of Theorem 2 is a beautiful application of Boolean Fourier analysis. Forbes and Kelley's techniques [FK18] work particularly well in the constant-width setting. By leveraging "Fourier growth bounds" for constant-width ROBPs [RSV13; SVW17; CHRT18], Forbes and Kelley obtained restrictions for constant-width ROBPs with better parameters. Recall that a distribution is "$\delta$-biased" if it fools parity functions with error $\delta/2$ [NN93].

**Theorem 3** (A pseudorandom restriction for constant-width ROBPs [FK18])**.** *Let $w \in \mathbb{N}$ be a constant. For every $n \in \mathbb{N}$ and $\varepsilon > 0$, there exists a value $\delta = \exp\left(-\widetilde{O}(\log(n/\varepsilon))\right)$ such that the following holds. Let $D$ and $T$ be $\delta$-biased random variables distributed over $\{0,1\}^n$, let $U$ be uniform random over $\{0,1\}^n$, and assume that $D$, $T$, and $U$ are mutually independent. Then $D + T \wedge U$ fools width-$w$ length-$n$ ROBPs with error $\varepsilon$, where $+$ denotes bitwise* XOR *and $\wedge$ denotes bitwise* AND*.*

Once again, the small-bias property is preserved under variable permutations. Using standard constructions of $\delta$-biased distributions [NN93; AGHP92], the random variables $D$ and $T$ of Theorem 3 can be sampled explicitly using $O(\log(n/\delta)) = \widetilde{O}(\log(n/\varepsilon))$ truly random bits. This leads to a PRG for constant-width arbitrary-order ROBPs with seed length $\widetilde{O}(\log(n/\varepsilon) \cdot \log n)$.

### 2.3 The Early Termination Technique

Forbes and Kelley's PRGs [FK18] are examples of restrictions-based PRGs with seed length $\mathrm{polylog}(n)$, similar to the seed length of Nisan's PRG [Nis92]. In some cases, we can use the iterated restrictions framework to get seed lengths as low as $\widetilde{O}(\log n)$ or even $O(\log n)$. The key idea is to show that after applying a few pseudorandom restrictions (say, $\mathrm{poly}(\log\log n)$ many), the function $f$ that we are trying to fool "simplifies" in some sense with high probability. When this occurs, we can *terminate the restriction process early*, and use some other approach to fool the restricted function, taking advantage of its simplicity.

This "early termination" technique was introduced by Gopalan, Meka, Reingold, Trevisan, and Vadhan [GMRTV12], and it has turned out to be useful for quite a few PRG problems [GMRTV12; MRT19; DHH19; Lee19; LV20; DHH20; DMRTV21]. Let us briefly discuss three examples.

- Gopalan, Meka, Reingold, Trevisan, and Vadhan designed an explicit PRG for *read-once* $\mathsf{CNF}$ *formulas* with near-optimal seed length $\widetilde{O}(\log(n/\varepsilon))$ [GMRTV12].

- Doron, Hatami, and Hoza designed an explicit PRG for *read-once* $\mathsf{AC}^0$ *formulas* with near-optimal seed length $\widetilde{O}(\log(n/\varepsilon))$ [DHH19].

- Doron, Meka, Reingold, Tal, and Vadhan designed an explicit PRG for constant-width arbitrary-order *monotone ROBPs* (defined below) with near-optimal seed length $\widetilde{O}(\log(n/\varepsilon))$ [DMRTV21].

**Definition 3** (Monotone ROBPs)**.** Let $f$ be a width-$w$ length-$n$ ROBP with transition functions $f_1, \dots, f_n \colon [w] \times \{0,1\} \to [w]$. We say that $f$ is *monotone* if, for each $i \in [n]$ and each bit $b \in \{0,1\}$, the transition function $f_i(\cdot, b)$ is a monotone function $[w] \to [w]$ [MZ13; DMRTV21].

It turns out that constant-width arbitrary-order monotone ROBPs can simulate read-once $\mathsf{AC}^0$ formulas [CSV15; DMRTV21]. In turn, obviously read-once $\mathsf{AC}^0$ formulas generalize read-once $\mathsf{CNF}$ formulas. Thus, the classes fooled by the three PRGs mentioned above form a hierarchy:

$$\text{read-once } \mathsf{CNF}\text{s} \subseteq \text{read-once } \mathsf{AC}^0 \subseteq \text{constant-width arbitrary-order monotone ROBPs.}$$

Over time, we are gradually figuring out how to fool *more and more powerful classes* with near-optimal seed length, building our way up toward the class of general arbitrary-order ROBPs. This type of progress (steadily improving the class of functions fooled) has turned out to be more feasible than insisting on fooling *all* ROBPs and trying to improve the seed length.

Recall that Forbes and Kelley's work (Theorem 3) shows how to assign values to half the input variables of a constant-width ROBP at a cost of only $\widetilde{O}(\log(n/\varepsilon))$ truly random bits. To get a full PRG in the monotone case, Doron, Meka, Reingold, Tal, and Vadhan show that after a few Forbes-Kelley restrictions, monotone ROBPs are likely to simplify [DMRTV21]. Roughly speaking, the notion of simplification is that the *width* of the program steadily decreases until function is trivial.

We remark that Doron, Meka, Reingold, Tal, and Vadhan's work [DMRTV21] is one example where *techniques designed for the arbitrary-order case have turned out to be useful even for the standard-order case.* (The monotone ROBP model was first introduced by Meka and Zuckerman [MZ13], who were not concerned with issues of variable ordering. They presented PRGs for the standard-order case [MZ13]; in the constant-width regime, their seed length matches Nisan's [Nis92].) This demonstrates the counterintuitive wisdom of working on problems that are even *more* difficult than the problems that we care about most.

### 2.4 A Challenge: Parity Gates

The iterated restrictions paradigm is flexible and powerful, especially when it is combined with the early termination technique. All of the recent work using these techniques is certainly exciting and encouraging. Unfortunately, however, we still do not have a clear path toward fooling all constant-width ROBPs (let alone polynomial-width ROBPs) with near-logarithmic seed length. Indeed, it seems that this line of work is perhaps "running out of steam."

To understand the limitations of these techniques, observe that for any ROBP $f \colon \{0,1\}^n \to \{0,1\}$, we can define a more complicated function $g \colon \{0,1\}^{n^2} \to \{0,1\}$ by block-composing with the parity function, i.e.,

$$g(x_{11}, \dots, x_{nn}) = f\left(\bigoplus_{i=1}^{n} x_{1i}, \bigoplus_{i=1}^{n} x_{2i}, \dots, \bigoplus_{i=1}^{n} x_{ni}\right).$$

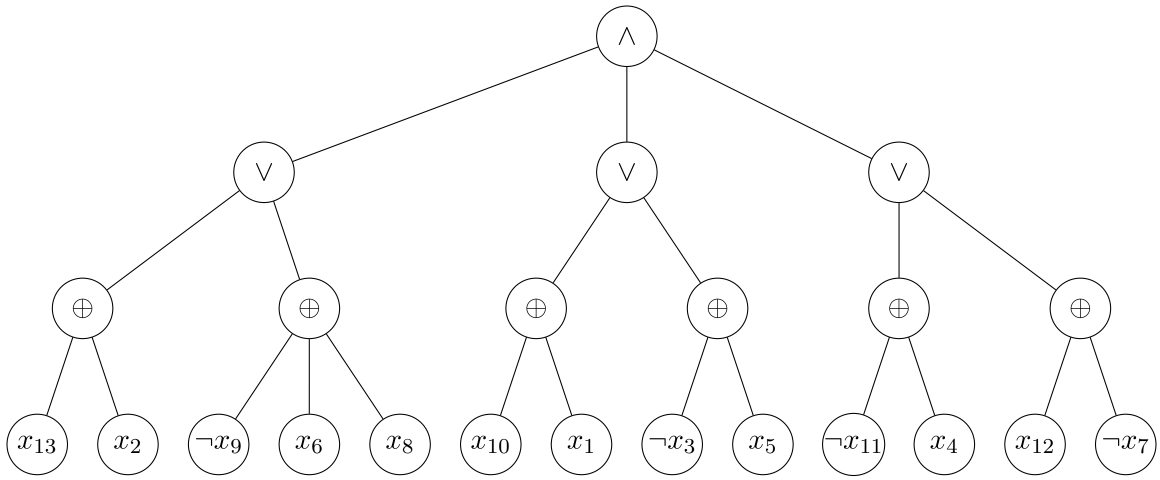


Figure 1: We would like to design explicit PRGs for constant-width (arbitary-order) ROBPs with near-optimal seed length $\widetilde{O}(\log(n/\varepsilon))$. The class of read-once $\mathsf{AC}^0$ formulas with parity gates is a challenging special case. Indeed, the case of read-once $\mathsf{AND} \circ \mathsf{OR} \circ \mathsf{PARITY}$ formulas already seems formidable.

If the initial ROBP $f$ has width $w = O(1)$, then $g$ can be computed by an ROBP of width $2w = O(1)$, but the early termination technique seems to break down when we try to apply it to $g$. It seems that (pseudo)random restrictions have very little effect on $g$, because a restriction of the parity function is always either the parity function or its complement. Fooling a typical restriction of $g$ is thus at least as difficult as fooling $f$.

More concretely, consider the problem of fooling read-once $\mathsf{AC}^0$ formulas *with parity gates* (Fig. 1). Doron, Hatami, and Hoza gave an explicit PRG for this class with seed length $\widetilde{O}(t + \log(n/\varepsilon))$ where $t$ is the number of parity gates in the formula [DHH19]. For the depth-2 case, we have explicit PRGs with near-optimal seed length [LV20; MRT19; Lee19], and in fact with "partially optimal" seed length $O(\log n) + \widetilde{O}(\log(1/\varepsilon))$ [DHH20]. However, when the depth is a large constant and the number of parity gates is unbounded, it seems quite difficult to achieve seed length $\widetilde{O}(\log n)$.

# 3 The Inverse Laplacian Perspective

In light of challenges such as that discussed in Section 2.4, it is worthwhile to take a step back and ask whether we truly need to design better PRGs for ROBPs. After all, our primary goal is derandomizing space-bounded computation. In this section, we discuss a non-PRG-based approach to proving $\mathsf{L} = \mathsf{BPL}$.

## 3.1 The Matrix of Expectations of Subprograms

To derandomize $\mathsf{BPL}$, it suffices to design a deterministic log-space algorithm that is given a width-$n$ length-$n$ ROBP $f$ and estimates $\mathbb{E}[f]$ to within a small additive error. There is no need to treat $f$ as a black box; it is permissible to inspect the transitions of $f$ and try to thereby gain some advantage. Since we are only concerned with space complexity, if we intend to estimate the expectation of the program, we might as well estimate the expectations of all subprograms, too.

**Definition 4** (Subprograms)**.** Suppose $f$ is a width-$w$ length-$n$ ROBP with layers $V_0, \dots, V_n$. Let $u \in V_i$ and $v \in V_j$ be vertices with $i \leq j$. We define the *subprogram* $f_{u \to v}$ to be the width-$w$

length-$(j-i)$ ROBP on layers $V_i, V_{i+1}, \ldots, V_j$ obtained from $f$ by designating $u$ as the start vertex and $v$ as the unique accepting vertex.

Let us collect all the expectations of these subprograms $\mathbb{E}[f_{u \to v}]$ in an $m \times m$ matrix $P$, where $m$ is the number of vertices in $f$, namely $m = w \cdot (n+1)$. That is, for every pair of vertices $u, v$ in $f$, if $u \in V_i$ and $v \in V_j$, then

$$P_{u,v} = \begin{cases} \mathbb{E}[f_{u \to v}] & \text{if } i \leq j \\ 0 & \text{if } i > j. \end{cases} \tag{4}$$

The following problem is essentially complete for $\mathsf{BPL}$:

- **Input:** A width-$n$ length-$n$ ROBP $f$.
- **Output:** A matrix $\widehat{P}$ that approximates the matrix of expectations of subprograms ($P$) to within additive entrywise error 0.1.

(By "essentially complete for $\mathsf{BPL}$," we mean that a *decision version* of the problem is complete for the *promise version* of $\mathsf{BPL}$ with respect to deterministic log-space reductions. These technicalities do not seem to be important.)

### 3.2 The Inverse Laplacian of an ROBP

To try to approximate $P$, we can start by computing the *random walk matrix* $W$. By definition, for each pair of vertices $u, v$ in $f$, the entry $W_{u,v}$ gives the probability of arriving at $v$ when we start at $u$ and take a single random step.

Computing $W$ is trivial: $W_{u,v}$ is half the number of edges from $u$ to $v$. Furthermore, there is a simple formula for the matrix of expectations of subprograms ($P$) in terms of the random walk matrix ($W$):

$$P = W^0 + W^1 + W^2 + \cdots + W^n. \tag{5}$$

Indeed, $(W^t)_{u,v}$ is the probability that a $t$-step random walk from $u$ arrives at $v$. Therefore, if $u \in V_i$ and $v \in V_j$, then

$$(W^t)_{u,v} = \begin{cases} \mathbb{E}[f_{u \to v}] & \text{if } j - i = t \\ 0 & \text{otherwise.} \end{cases} \tag{6}$$

Eq. (5) follows. Eq. (6) also shows that $W^{n+1} = 0$, which means that we can simplify Eq. (5) using the geometric series formula:

$$P = (I - W)^{-1}. \tag{7}$$

The matrix $I - W$ is called the (directed) *Laplacian matrix* of the program $f$ and denoted $L$. Thus, we are looking for an *approximate inverse Laplacian* $\widehat{P} \approx L^{-1}$. This way of thinking – the "inverse Laplacian perspective" – was introduced most clearly in work by Ahmadinejad, Kelner, Murtagh, Peebles, Sidford, and Vadhan [AKMPSV20], and it is our second technical "theme."

### 3.3 Local Consistency

A key benefit of the inverse Laplacian perspective is that it suggests a new way of thinking about error. Suppose that someone gives us a candidate matrix $\widehat{P}$. Is $\widehat{P}$ a good approximation to $P$? We cannot directly compare the entries of $\widehat{P}$ to those of $P$, because we do not know $P$ (remember, approximating $P$ is essentially $\mathsf{BPL}$-complete). However, we *can* compute the error *after multiplying by the Laplacian matrix*. That is, we can compare $\widehat{P}L$ to the identity matrix. Define $E$ to be the error matrix $E = I - \widehat{P}L$.

This error matrix $E$ has a natural probabilistic interpretation. If $u \in V_i$ and $v \in V_j$ where $i < j$, then one can show that

$$E_{u,v} = \underbrace{\left( \sum_{s \in V_{j-1}} \widehat{P}_{u,s} \cdot W_{s,v} \right)}_{(*)} - \widehat{P}_{u,v}.$$

The entry $E_{u,v}$ measures the difference between two different methods of using $\widehat{P}$ to estimate $\mathbb{E}[f_{u \to v}]$. The first method is to simply consult the $(u, v)$ entry of $\widehat{P}$, since after all $\widehat{P}$ is intended to be an approximation to $P$. The second method is to look at $\widehat{P}$'s estimates for the probabilities of arriving at vertices in the layer $V_{j-1}$ that *precedes* $v$, and then propagate those probabilities forward by a single step, leading to quantity $(*)$.

Thus, $E$ measures the extent to which $\widehat{P}$ is locally *consistent with itself*; we refer to $E$ as the matrix of *local consistency errors*. The term "local consistency" was introduced by Cheng and Hoza [CH20]; the connection between local consistency and the Laplacian matrix was observed by subsequent papers [CDRST21; PV21b; Hoz21].

### 3.4 One-Sided vs. Two-Sided Derandomization

Cheng and Hoza used the concept of local consistency to prove a new conditional derandomization of $\mathsf{BPL}$ [CH20]. Recall that a *hitting set generator* (HSG) is a one-sided generalization of a PRG.

**Definition 5** (HSGs)**.** Let $\mathcal{F}$ be a class of functions $f : \{0,1\}^n \to \{0,1\}$ and let $\varepsilon > 0$. An *$\varepsilon$-HSG* for $\mathcal{F}$ is a function $G : \{0,1\}^s \to \{0,1\}^n$ such that for every $f \in \mathcal{F}$,

$$\text{if } \Pr[f(U_n) = 1] \geq \varepsilon, \text{ then } \exists x \text{ such that } f(G(x)) = 1.$$

HSGs are potentially much easier to construct than PRGs, so it is worthwhile to ask, what would be the applications of optimal explicit HSGs? Working through the definitions, one can easily show that an optimal explicit HSG for ROBPs would imply $\mathsf{L} = \mathsf{RL}$ (one-sided derandomization). Cheng and Hoza showed that it would also imply the stronger statement $\mathsf{L} = \mathsf{BPL}$ (two-sided derandomization) [CH20].

**Theorem 4** (HSGs would derandomize $\mathsf{BPL}$ [CH20])**.** *Assume that for every $n \in \mathbb{N}$, there is a $\frac{1}{2}$-HSG for width-$n$ length-$n$ ROBPs that has seed length $O(\log n)$ and that is computable in space $O(\log n)$. Then $\mathsf{L} = \mathsf{BPL}$.*

To prove Theorem 4, briefly, suppose we are given a width-$n$ length-$n$ ROBP $f$. Let $G$ be an HSG with output length $n^c$, where $c$ is a large enough constant. For each seed $x$, we think of $G(x)$ as a long stream of random bits and use it to compute a matrix $\widehat{P}^{(x)}$ that is a candidate approximation to the matrix $P$ of expectations of subprograms of $f$. Using the hitting property of $G$, one can show that there is at least one "good seed" $x$ such that $\widehat{P}^{(x)} \approx P$. To identify such a seed algorithmically, we find an $x$ such that $\widehat{P}^{(x)}$ has good local consistency.

We remark that an analogous theorem for time-bounded derandomization has been known for decades [ACR98; ACRT99; BF99; GVW11]. In fact, Buhrman and Fortnow showed generically that derandomizing the promise version of $\mathsf{RP}$ would imply $\mathsf{P} = \mathsf{BPP}$, regardless of whether the derandomization is via an HSG [BF99]. An interesting open problem is to prove the analogous theorem for the space-bounded setting, generalizing Theorem 4.

# 4 Error Reduction Procedures

In the previous section, we introduced the inverse Laplacian perspective, and we discussed one application (the derandomization of BPL using a hypothetical HSG). There are several other applications of the inverse Laplacian perspective. These other applications take advantage of the rich literature on *fast, randomized* algorithms for approximately solving Laplacian systems of equations, starting with Spielman and Teng's seminal work [ST04]. Most especially, these other applications work by importing *error reduction* techniques – our third "theme" – to the space-bounded derandomization setting.

## 4.1 Non-Black-Box Error Reduction

As our first example, let us discuss a theorem by Ahmadinejad, Kelner, Murtagh, Peebles, Sidford, and Vadhan [AKMPSV20] (strengthening prior work by Hoza and Zuckerman [HZ20]). Their theorem shows how to generically decrease the error of space-bounded derandomization algorithms. In the following, think of $\varepsilon$ as negligibly small compared to $n$, such as perhaps $\varepsilon = 2^{-\operatorname{polylog}(n)}$.

**Theorem 5** (Error reduction for non-black-box derandomization [AKMPSV20])**.** *Let $S\colon \mathbb{N} \to \mathbb{N}$ be a function. Assume that given a width-$n$ length-$n$ ROBP $f$, it is possible to deterministically compute $\mathbb{E}[f]$ to within $\pm 1/n^3$ in space $S(n)$. Then given $f$ and $\varepsilon > 0$, it is possible to deterministically compute $\mathbb{E}[f]$ to within $\pm\varepsilon$ in space*

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

Let us sketch the proof of Theorem 5, which uses the inverse Laplacian perspective. Let $P$ be the matrix of expectations of subprograms of $f$. Using the given $S(n)$-space algorithm, we can construct a matrix $\widehat{P}$ such that $\|P - \widehat{P}\|_\infty \le O(1/n)$. Let $W$ be the random walk matrix of $f$, let $L = I - W$ be the Laplacian matrix, and let $E = I - \widehat{P}L$ be the error matrix after multiplying by $L$ (aka the matrix of local consistency errors). Then, we define a new approximation matrix $\widehat{P}'$ by the formula

$$\widehat{P}' = \widehat{P} + E\widehat{P} + E^2\widehat{P} + \cdots + E^m\widehat{P}$$

for a suitably chosen parameter $m$. (Intuitively, we start with $\widehat{P}$, and then we add a sequence of finer and finer "correction terms" $E\widehat{P}, E^2\widehat{P}, \ldots, E^m\widehat{P}$.) Let us measure the quality of this new approximation. The key, again, is to measure quality *after* multiplying by the Laplacian matrix, which causes a telescoping sum:

$$\widehat{P}'L = (I - E) + E \cdot (I - E) + E^2 \cdot (I - E) + \cdots + E^m \cdot (I - E) = I - E^m.$$

Amazingly, we have managed to replace $E$ with $E^m$, which intuitively should mean that the errors are getting much smaller. This technique for decreasing the error of an approximate matrix inverse is called *preconditioned Richardson iteration.*

Ultimately, what we care about is entrywise closeness to $P$. We can bound the entrywise errors using the submultiplicative $\|\cdot\|_\infty$ matrix norm:

$$\begin{aligned} \left\|\widehat{P}' - P\right\|_\infty = \left\|\left(\widehat{P}'L - I\right) \cdot P\right\|_\infty = \|E^m \cdot P\|_\infty &\le \|E\|_\infty^m \cdot \|P\|_\infty \\ &= \left\|\left(P - \widehat{P}\right) \cdot L\right\|_\infty^m \cdot \|P\|_\infty \\ &\le \left(\left\|P - \widehat{P}\right\|_\infty \cdot \|L\|_\infty\right)^m \cdot \|P\|_\infty \\ &\le O(1/n)^m \cdot O(n), \end{aligned}$$

which is at most $\varepsilon$ if we choose a suitable value $m = O(\log(1/\varepsilon)/\log n)$. One can compute $\widehat{P}'$ deterministically in space $O(S(n) + \log n \cdot \log m)$, completing the proof of Theorem 5.

### 4.2 Weighted Pseudorandom Generators (WPRGs)

The parameters of Theorem 5 are impressive; we pay very little penalty for error reduction, even when the target error $\varepsilon$ is extremely small. The algorithm of Theorem 5 is non-black-box, because we must inspect the graph structure of the given ROBP to compute the matrices $W$, $L$, $E$, etc.

As discussed previously, non-black-box algorithms are ultimately sufficient for proving $\mathsf{L} = \mathsf{BPL}$. However, black-box algorithms are stronger, and they tend to be more useful as building blocks inside larger algorithms. What is the best way to compute $\mathbb{E}[f]$ to within a tiny additive error $\varepsilon$ if we only have *query* access to an ROBP $f$? An $\varepsilon$-PRG clearly suffices for this task, but could there be an easier approach? This motivates the intriguing concept of a *weighted pseudorandom generator* (WPRG), introduced by Braverman, Cohen, and Garg [BCG20].

**Definition 6** (WPRGs)**.** Let $\mathcal{F}$ be a class of functions $f\colon \{0,1\}^n \to \{0,1\}$ and let $\varepsilon > 0$. An *$\varepsilon$-WPRG* for $\mathcal{F}$ is a pair $(G, \rho)$, where $G\colon \{0,1\}^s \to \{0,1\}^n$ and $\rho\colon \{0,1\}^s \to \mathbb{R}$, such that for every $f \in \mathcal{F}$,

$$\left| \mathop{\mathbb{E}}_{x\sim U_n}[f(x)] - \mathop{\mathbb{E}}_{x\sim U_s}[f(G(x)) \cdot \rho(x)] \right| \leq \varepsilon. \tag{8}$$

A PRG is the special case $\rho \equiv 1$. Crucially, Definition 6 allows for $\rho(x) < 0$, which opens the door for the possibility of *error cancellation* in Eq. (8). One can think of these negative weights as effectively introducing a kind of "negative probability" into the picture; WPRGs are also known as *pseudorandom pseudodistribution generators*.[6]

One can show that if $(G, \rho)$ is an $\varepsilon$-WPRG for $\mathcal{F}$, then $G$ is an $\varepsilon'$-HSG for $\mathcal{F}$ for every $\varepsilon' > \varepsilon$. Thus, we have a hierarchy,

$$\text{PRG} \implies \text{WPRG} \implies \text{HSG}.$$

When they introduced the concept of a WPRG, Braverman, Cohen, and Garg presented an explicit construction of an $\varepsilon$-WPRG for polynomial-width ROBPs [BCG20] with seed length

$$\widetilde{O}(\log^2 n + \log(1/\varepsilon)). \tag{9}$$

For comparison, recall that Nisan's PRG fools polynomial-width ROBPs with seed length $O(\log^2 n + \log n \cdot \log(1/\varepsilon))$ [Nis92]. Thus, Braverman, Cohen, and Garg's seed length [BCG20] is superior when $\varepsilon$ is very small (again, the case $\varepsilon = 2^{-\operatorname{polylog}(n)}$ is good to have in mind). Prior to their work [BCG20], it was not even known how to construct an HSG with the seed length that they achieve.

Braverman, Cohen, and Garg's work [BCG20] is quite complex. This spurred a search for simpler approaches [HZ20; CL20; CDRST21; PV21b; Hoz21]. In addition to achieving improved simplicity, this line of work was also able to remove the lower-order terms hiding under the $\widetilde{O}$ in Eq. (9).

**Theorem 6** (Optimal-error WPRGs [Hoz21])**.** *For every $w, n \in \mathbb{N}$ and $\varepsilon > 0$, there is an explicit $\varepsilon$-WPRG for width-$w$ length-$n$ ROBPs with seed length $O(\log(wn) \cdot \log n + \log(1/\varepsilon))$.*

To prove Theorem 6, we start with Nisan's PRG with error $1/\operatorname{poly}(nw)$ and seed length $O(\log(wn) \cdot \log n)$. Then, we use the *preconditioned Richardson iteration* technique that we discussed

[6] Braverman, Cohen, and Garg coined the term "pseudorandom pseudodistribution" [BCG20]. The alternative term "weighted pseudorandom generator" was introduced later, by Cohen, Doron, Renard, Sberlo, and Ta-Shma [CDRST21].

in Section 4.1 to decrease the error of the PRG. Implementing this technique is not completely straightforward, because we are in the black-box setting, and hence we can no longer compute the matrices $W$, $L$, $E$, etc. However, two independent papers (one by Cohen, Doron, Renard, Sberlo, and Ta-Shma [CDRST21] and the other by Pyne and Vadhan [PV21b]) contributed the insight that one can set up the WPRG *construction* in such a way that preconditioned Richardson iteration happens in the *analysis*. Finally, to achieve the seed length of Theorem 6, we combine these ideas with a suitable sampler trick [Hoz21].

In general, starting from an explicit PRG for width-$w$ length-$n$ ROBPs with error $1/(wn)^c$ and seed length $s$ (for a suitable constant $c > 1$), we get an explicit WPRG for width-$w$ length-$n$ ROBPs with arbitrarily small error $\varepsilon$ and seed length $O(s + \log(1/\varepsilon))$ [Hoz21]. There are other, related error reduction procedures that achieve slightly better parameters in some cases [HZ20; CDRST21; PV21b]. For example, consider ROBPs of width $w$ and length $\log^c w$ for a constant $c \in \mathbb{N}$. Nisan and Zuckerman showed how to fool these short, wide programs with seed length $O(\log w)$ and a relatively large error such as $2^{-(\log w)^{0.99}}$. By applying an error-reduction procedure to the PRG, Hoza and Zuckerman designed an explicit $\varepsilon$-HSG for these programs with asymptotically optimal seed length $O(\log(w/\varepsilon))$, even when $\varepsilon$ is small. It remains an interesting open problem to match this seed length with a WPRG.

### 4.3 Improving the Saks-Zhou Algorithm

Let us now discuss an unconditional application of low-error WPRGs. Recall our original derandomization goal: we want to deterministically decide languages in $\mathsf{BPSPACE}(S)$, for $S \geq \log N$, using as little space as possible.

Savitch's theorem implies that $\mathsf{RSPACE}(S) \subseteq \mathsf{DSPACE}(S^2)$. The more general inclusion $\mathsf{BPSPACE}(S) \subseteq \mathsf{DSPACE}(S^2)$ follows from early work on the non-halting version of $\mathsf{BPSPACE}(S)$ [BCP83; Jun81]. Later, Saks and Zhou used Nisan's PRG [Nis92] in a sophisticated way to prove $\mathsf{BPSPACE}(S) \subseteq \mathsf{DSPACE}(S^{3/2})$ [SZ99]. Now, decades later, we can finally improve Saks and Zhou's bound.

**Theorem 7** (Improved derandomization of $\mathsf{BPSPACE}$ [Hoz21])**.** *Let $S \colon \mathbb{N} \to \mathbb{N}$ be a function satisfying $S(N) \geq \log N$. Then*

$$\mathsf{BPSPACE}(S) \subseteq \mathsf{DSPACE}\left(\frac{S^{3/2}}{\sqrt{\log S}}\right). \tag{10}$$

Admittedly, the bound of Eq. (10) is only barely better than Saks and Zhou's $O(S^{3/2})$ bound [SZ99]. Still, Theorem 7 potentially has some "psychological" value, because it demonstrates that Saks and Zhou's result [SZ99] is not the "end of the road." There is no particular reason to think that Theorem 7 is the end of the road either. No compelling barriers to further progress are known; humanity has no real excuse for having not yet proven $\mathsf{L} = \mathsf{BPL}$.

The starting point for proving Theorem 7 is work by Armoni from more than two decades ago [Arm98]. Armoni designed an explicit $\varepsilon$-PRG for width-$w$ length-$n$ ROBPs based on a generalization of Nisan and Zuckerman's techniques [NZ96]. Armoni's seed length is slightly better than Nisan's seed length [Nis92] in the regime $n \ll w$ and $\varepsilon \gg 1/w$ [Arm98]. By combining his PRG with recent error reduction techniques [CDRST21; PV21b], we get an explicit WPRG with a seed length that is slightly better than Nisan's seed length [Nis92] in the regime $n \ll w$, *even for low error* such as $\varepsilon = 1/\operatorname{poly}(w)$.

The original Saks-Zhou algorithm [SZ99] uses Nisan's PRG with parameters in this regime ($n \ll w$ and $\varepsilon = 1/\operatorname{poly}(w)$) as a subroutine. Armoni showed how to use a generic PRG in

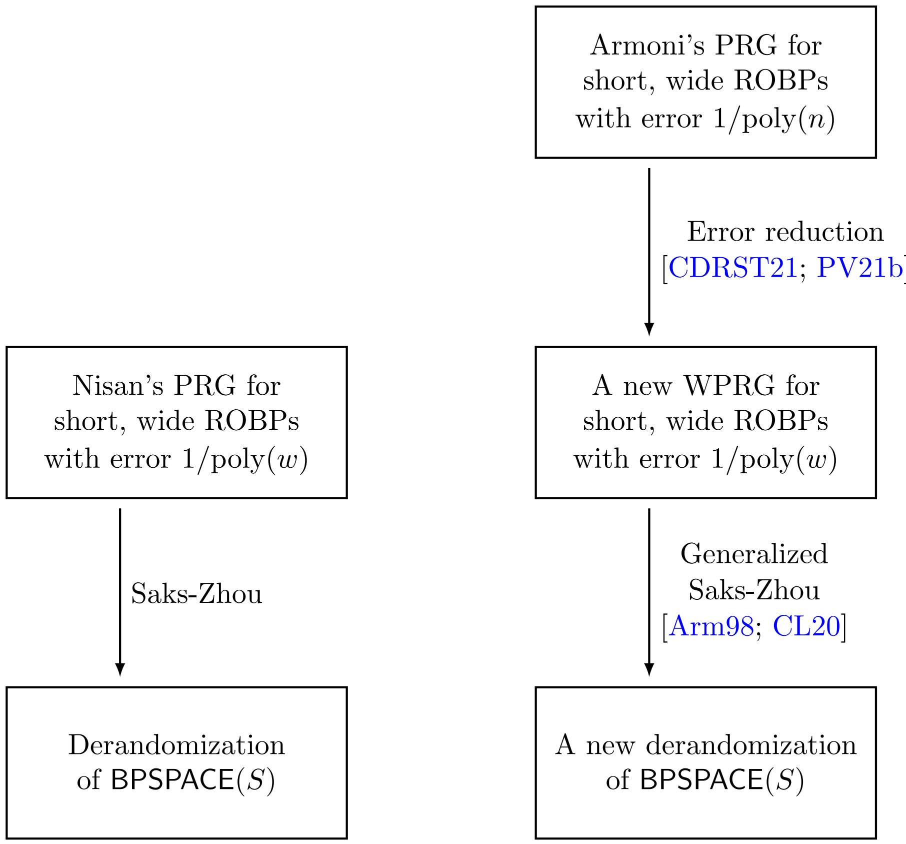


Figure 2: Saks and Zhou's derandomization of BPSPACE [SZ99] (left) vs. the new and improved derandomization of BPSPACE (Theorem 7, right).

place of Nisan's PRG [Arm98], and Chattopadhyay and Liao showed more generally how to use WPRGs [CL20], building off an earlier suggestion by Braverman, Cohen, and Garg [BCG20]. Combining these results proves Theorem 7. (See Fig. 2.)

Cohen, Doron, and Sberlo recently designed an algorithm that improves on Saks and Zhou's work [SZ99] in a different direction [CDS22]. Consider the following natural computational problem.

- **Input:** A value $n \in \mathbb{N}$ and a stochastic matrix $M \in \mathbb{R}^{w \times w}$, where each entry has bit complexity $O(\log(wn))$.
- **Output:** A matrix that approximates $M^n$ to additive entrywise error 0.1.

When we restrict to the case $n = w$, the problem above is essentially complete for BPL. One can think of the Saks-Zhou algorithm as a method of solving the problem in space $O(\log(wn) \cdot \sqrt{\log n})$. Cohen, Doron, and Sberlo show how to solve the problem in space $\widetilde{O}(\log w \cdot \sqrt{\log n} + \log n)$ [CDS22], which is a significant improvement in the regime $n \gg w$. Their algorithm combines the Saks-Zhou algorithm with Richardson iteration, but in a different way than the proof of Theorem 7.

# 5 Spectral Expander Graphs

Let us consider one more natural problem that is essentially complete for BPL.

- **Input:** A directed graph $G$, two vertices $s$ and $t$, and a positive integer $k$, represented in unary.
- **Output:** The probability that a $k$-step random walk starting at $s$ ends at $t$, to within an additive error of 0.1.

An appealing special case is when $G$ is undirected. As mentioned previously, Reingold designed a deterministic log-space algorithm to determine whether there *exists* a path from $s$ to $t$ in an undirected graph $G$ [Rei08], which, intuitively, corresponds to the case $k = \infty$. A recent line of work has studied the case that $k$ is finite, and in particular, $k$ might be smaller than the mixing time of $G$ [MRSV21a; MRSV21b; AKMPSV20]. For any $k$, Ahmadinejad, Kelner, Murtagh, Peebles, Sidford, and Vadhan gave an algorithm for computing $k$-step random walk probabilities in undirected graphs, even to within inverse polynomial error, that runs in near-logarithmic space [AKMPSV20].

**Theorem 8** (Estimating random-walk probabilities in undirected graphs [AKMPSV20])**.** *Given an undirected graph (or, more generally, an Eulerian digraph) $G$, two vertices $s$ and $t$, and a positive integer $k$ represented in unary, it is possible to deterministically compute the probability that a length-$k$ random walk starting at $s$ arrives at $t$ to within additive error $1/\operatorname{poly}(N)$ in space $\widetilde{O}(\log N)$, where $N$ is the bit-length of the input.*

One of the (many) ideas in the proof of Theorem 8 is to use *spectral expander graphs* to take a certain type of pseudorandom walk through $G$ instead of a truly random walk. There is a long history of using expanders as tools for space-bounded derandomization, going back to work by Ajtai, Komlós, and Szemerédi [AKS87]. Modern work on $\mathsf{L}$ vs. $\mathsf{BPL}$ continues to develop new ways of using and analyzing expanders – our fourth technical "theme."

## 5.1 The Derandomized Square

In more detail, the proof of Theorem 5 uses expanders via Rozenman and Vadhan's *derandomized square* operation [RV05]. For simplicity, consider a $D$-regular undirected graph $G$. A single step in the "square graph" $G^2$ corresponds to two steps in the original graph $G$. Effectively, this means that squaring $G$ places a clique on the $D$ neighbors of each vertex $v$. The idea of the derandomized square is to instead place an expander graph on the neighbors of $v$, thereby producing a sparse approximation to $G^2$.

In Rozenman and Vadhan's original paper, they prove that the *spectral expansion* of the derandomized square is almost as good as that of $G^2$ [RV05]. They use this bound to derive an alternative proof that undirected connectivity is in $\mathsf{L}$ [RV05]. Recent work [MRSV21a; MRSV21b; AKMPSV20] shows that the derandomized square approximates $G^2$ in much stronger senses. The proof of Theorem 8 combines this analysis with several other techniques, including the inverse Laplacian perspective and error reduction methods.

## 5.2 The INW Generator

The derandomized square operation also has connections to the PRG approach to $\mathsf{L}$ vs. $\mathsf{BPL}$, and in particular to a PRG by Impagliazzo, Nisan, and Wigderson [INW94] (the "INW generator"). The INW generator samples $n$ pseudorandom bits as follows:

1. Recursively construct a PRG $G \colon \{0,1\}^s \to \{0,1\}^{n/2}$.
2. Sample a uniform random vertex $X$ and a uniform random neighbor $Y$ in an expander graph on $2^s$ vertices.

3. Output the concatenation $G(X) \circ G(Y)$.

Several decades after its introduction [INW94], we are still learning more and more about what the INW generator is capable of. It has been shown to work particularly well for *regular* and *permutation* ROBPs.

**Definition 7** (Regular and permutation ROBPs)**.** Let $f$ be a width-$w$ length-$n$ ROBP with transition functions $f_1, \ldots, f_n \colon [w] \times \{0,1\} \to [w]$. We say that $f$ is a *permutation ROBP* if, for every $i \in [n]$ and every $b \in \{0,1\}$, the function $f_i(\cdot, b)$ is a permutation on $[w]$. More generally, we say that $f$ is *regular* if, for every $i \in [n]$ and every $u \in [w]$, we have $|f_i^{-1}(u)| = 2$.

Regular and permutation ROBPs have been studied extensively over the course of roughly the past decade [BRRY14; BV10; De11; KNP11; Ste12; RSV13; CHHL19; HPV21; PV21a; PV21b; CPT21; PV22; BHPP22; GV22; LPV22]. We now have various types of pseudorandomness results for regular and permutation ROBPs that are superior to the best corresponding results for general ROBPs. In many cases, the proofs consist of improved analyses of the classic INW construction [INW94] (with modified parameters). In other cases, the INW generator is one of multiple ingredients.

### 5.3 Unbounded-Width ROBPs

The first few papers on regular and permutation ROBPs [BRRY14; BV10; De11; KNP11; Ste12; RSV13] focused on constant-width programs. Arguably the most important case is that of polynomial-width programs. The trend recently has been to study the intriguing setting of *unbounded-width* programs [HPV21; PV21a; PV21b; PV22; BHPP22; GV22; LPV22].

Without further constraints, unbounded-width permutation ROBPs are too powerful to be interesting: they can compute all Boolean functions. Therefore, we assume that the program has a bounded number of *accepting states* in the final layer. Admittedly, width is a more natural complexity measure than the number of accepting states, but programs with a bounded number of accepting states turn out to be mathematically interesting. Even with just *one* accept state, unbounded-width permutation ROBPs can compute doubly-exponentially many distinct functions:

**Lemma 1** ([HPV21])**.** *Let $n \in \mathbb{N}$ be a positive even integer, and let $\pi \colon \{0,1\}^{n/2} \to \{0,1\}^{n/2}$ be a permutation. There exists a width-$(2^{n/2})$ length-$n$ permutation ROBP $f$ computing the following function:*

$$f(x, y) = 1 \iff \pi(x) = y.$$

(Briefly, to prove Lemma 1, we use the state space $\{0,1\}^{n/2}$. The all-zeroes state is the start state and the unique accepting state. We XOR $x$ into our state, then apply $\pi$ to our state, then XOR $y$ into our state.) On the other hand, one can check that the majority function on three bits cannot be computed by a regular ROBP with a single accept vertex, no matter how wide the program is. Thus, these strange unbounded-width models have both dramatic strengths and dramatic weaknesses.

One of the most striking results in this space is the following theorem by Pyne and Vadhan [PV21b].

**Theorem 9** (WPRGs for unbounded-width permutation ROBPs [PV21b])**.** *For every $n \in \mathbb{N}$ and $\varepsilon > 0$, there is an explicit $\varepsilon$-WPRG for unbounded-width permutation ROBPs with a single accept state with seed length*

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

Theorem 9 has implications for the more conventional setting of bounded-width permutation ROBPs. Every $\varepsilon$-WPRG for programs with one accepting state automatically $(\varepsilon a)$-fools programs with $a$ accepting states. Therefore, Theorem 9 implies an explicit WPRG for width-$n$ length-$n$ permutation ROBPs (with any number of accepting vertices) with error $1/n$ and seed length $\widetilde{O}(\log^{3/2} n)$, compared to Nisan's $O(\log^2 n)$ bound.

Theorem 9 also helps to clarify the importance of weights. When $\varepsilon = 1/n$, the seed length in Theorem 9 is $\widetilde{O}(\log^{3/2} n)$. In contrast, Hoza, Pyne, and Vadhan proved that every *unweighted* PRG that $(1/n)$-fools unbounded-width permutation ROBPs with a single accept vertex must have seed length $\Omega(\log^2 n)$ [HPV21]. Therefore, in at least one fairly-natural setting, WPRGs are *intrinsically more powerful* than traditional PRGs.

The proof of Theorem 9 uses the INW generator, the inverse Laplacian perspective, and error reduction techniques, among other ideas.

## 5.4 The Permutation Case and the Monotone Case: Opposite Extremes

Why study regular and permutation ROBPs? The main reason is the hope that studying these special cases will lead to improvements in the general case. Indeed, there is a reduction showing that good PRGs or HSGs for polynomial-width *regular* ROBPs imply good PRGs or HSGs for *all* polynomial-width ROBPs [RTV06; BHPP22].[7]

In the constant-width setting, no formal reduction is known. However, one can argue intuitively that permutation ROBPs and monotone ROBPs are "opposites" of one another. In a permutation ROBP, edges with the same label never collide, whereas in a monotone ROBP, the only way that a layer can do any nontrivial computation is by introducing collisions.

We have one set of techniques that works well for permutation ROBPs (spectral expanders and the INW generator), and we have another set of techniques that works well for monotone ROBPs (iterated restrictions with early termination). Might it be possible to combine these two sets of techniques to get a PRG that fools all width-$w$ ROBPs with seed length $\widetilde{O}(\log n)$ when $w$ is a constant? The idea might sound a bit naïve or fantastical, especially considering the difficulty discussed in Section 2.4. Remarkably, however, Meka, Reingold, and Tal proved that the answer is yes for the case $w = 3$ [MRT19].

**Theorem 10** (PRGs for width-3 ROBPs [MRT19])**.** *For every $n \in \mathbb{N}$ and $\varepsilon > 0$, there is an explicit $\varepsilon$-PRG for width-3 ROBPs with seed length $\widetilde{O}(\log n \cdot \log(1/\varepsilon))$.*

To prove Theorem 10, Meka, Reingold, and Tal first show how to sample pseudorandom restrictions that preserve the expectation of width-3 ROBPs. For this first step, one can alternatively use Forbes and Kelley's analysis (Theorem 3), which works more generally for width-$w$ ROBPs where $w$ is small. (The papers of Forbes and Kelley [FK18] and Meka, Reingold, and Tal [MRT19] are independent.)

Next, Meka, Reingold, and Tal show that width-3 ROBPs simplify after a few pseudorandom restrictions [MRT19]. And what does "simplify" mean in this context? Roughly speaking, they show that the program becomes *more and more permutation-like* as the restrictions are applied. After $\mathrm{poly}(\log\log(n/\varepsilon))$ many restrictions, they terminate the restriction process and apply the INW generator [INW94] as the final step. Building on Braverman, Rao, Raz, and Yehudayoff's analysis [BRRY14], they show that the INW generator fools these "highly permutation-like" ROBPs with a short seed length [MRT19].

[7] Note that Theorem 8 implies a *non-black-box* algorithm for estimating the expectation of a given regular ROBP in near-logarithmic space. Unfortunately, the reduction from the general case to the regular case does not work in the non-black-box setting.

It remains an open problem to design an explicit PRG (or WPRG or HSG) for width-4 ROBPs with seed length $o(\log^2 n)$.

## 6 Conclusions

We continue to make steady, substantial progress toward proving $\mathsf{L} = \mathsf{BPL}$. The past few years alone have yielded many exciting results and developments. The problem remains challenging, but there does not seem to be any firm obstacle preventing further breakthroughs.

## 7 Acknowledgments

I thank Edward Pyne for helpful comments on a draft of this article.

## References

[ACR98] Alexander E. Andreev, Andrea E. F. Clementi, and José D. P. Rolim. "A New General Derandomization Method". In: *J. ACM* 45.1 (1998), pp. 179–213. ISSN: 0004-5411. DOI: 10.1145/273865.273933.

[ACRT99] Alexander E. Andreev, Andrea E. F. Clementi, José D. P. Rolim, and Luca Trevisan. "Weak Random Sources, Hitting Sets, and BPP Simulations". In: *SIAM J. Comput.* 28.6 (1999), pp. 2103–2116. ISSN: 0097-5397. DOI: 10.1137/S0097539797325636.

[AGHP92] Noga Alon, Oded Goldreich, Johan Håstad, and René Peralta. "Simple Constructions of Almost $k$-wise Independent Random Variables". In: *Random Structures Algorithms* 3.3 (1992), pp. 289–304. ISSN: 1042-9832. DOI: 10.1002/rsa.3240030308.

[AKLLR79] Romas Aleliunas, Richard M. Karp, Richard J. Lipton, László Lovász, and Charles Rackoff. "Random walks, universal traversal sequences, and the complexity of maze problems". In: *Proceedings of the 20th Symposium on Foundations of Computer Science (FOCS)*. 1979, pp. 218–223. DOI: https://doi.org/10.1109/SFCS.1979.34.

[AKMPSV20] AmirMahdi Ahmadinejad, Jonathan Kelner, Jack Murtagh, John Peebles, Aaron Sidford, and Salil Vadhan. "High-precision estimation of random walks in small space". In: *Proceedings of the 61st Symposium on Foundations of Computer Science*. 2020, pp. 1295–1306. DOI: 10.1109/FOCS46700.2020.00123.

[AKS87] Miklós Ajtai, János Komlós, and Endre Szemerédi. "Deterministic Simulation in LOGSPACE". In: *Proceedings of the 19th Symposium on Theory of Computing (STOC)*. 1987, pp. 132–140. ISBN: 0897912217. DOI: 10.1145/28395.28410.

[Arm98] Roy Armoni. "On the Derandomization of Space-Bounded Computations". In: *Proceedings of the 2nd International Workshop on Randomization and Approximation Techniques in Computer Science (RANDOM)*. 1998, pp. 47–59. DOI: 10.1007/3-540-49543-6_5.

[ATWZ00] Roy Armoni, Amnon Ta-Shma, Avi Wigderson, and Shiyu Zhou. "An $O(\log(n)^{4/3})$ space algorithm for $(s,t)$ connectivity in undirected graphs". In: *J. ACM* 47.2 (2000), pp. 294–311. ISSN: 0004-5411. DOI: 10.1145/333979.333984.

[AW89] Miklós Ajtai and Avi Wigderson. "Deterministic Simulation of Probabilistic Constant-Depth Circuits". In: *Advances in Computing Research – Randomness and Computation* 5 (1989), pp. 199–23.

[BCDRT89] Allan Borodin, Stephen A. Cook, Patrick W. Dymond, Walter L. Ruzzo, and Martin Tompa. "Two applications of inductive counting for complementation problems". In: *SIAM J. Comput.* 18.3 (1989), pp. 559–578. ISSN: 0097-5397. DOI: 10.1137/0218038.

[BCG20] Mark Braverman, Gil Cohen, and Sumegha Garg. "Pseudorandom Pseudodistributions with Near-Optimal Error for Read-Once Branching Programs". In: *SIAM Journal on Computing* 49.5 (2020), STOC18–242–STOC18–299. DOI: 10.1137/18M1197734.

[BCP83] A. Borodin, S. Cook, and N. Pippenger. "Parallel Computation for Well-Endowed Rings and Space-Bounded Probabilistic Machines". In: *Inform. and Control* 58.1-3 (1983), pp. 113–136. ISSN: 0019-9958. DOI: 10.1016/S0019-9958(83)80060-6.

[BF99] Harry Buhrman and Lance Fortnow. "One-Sided Versus Two-Sided Error in Probabilistic Computation". In: *Proceedings of the 16th Symposium on Theoretical Aspects of Computer Science (STACS)*. 1999, pp. 100–109. DOI: 10.1007/3-540-49116-3_9.

[BHPP22] Andrej Bogdanov, William M. Hoza, Gautam Prakriya, and Edward Pyne. "Hitting Sets for Regular Branching Programs". In: *Proceedings of the 37th Computational Complexity Conference (CCC 2022)*. 2022, 3:1–3:22. ISBN: 978-3-95977-241-9. DOI: 10.4230/LIPIcs.CCC.2022.3.

[BPW11] Andrej Bogdanov, Periklis A. Papakonstantinou, and Andrew Wan. "Pseudorandomness for Read-Once Formulas". In: *Proceedings of the 52nd Symposium on Foundations of Computer Science (FOCS)*. 2011, pp. 240–246. DOI: 10.1109/FOCS.2011.57.

[BR97] Greg Barnes and Walter L. Ruzzo. "Undirected $s$-$t$ connectivity in polynomial time and sublinear space". In: *Comput. Complexity* 6.1 (1997), pp. 1–28. ISSN: 1016-3328. DOI: 10.1007/BF01202039.

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

[BV10] Joshua Brody and Elad Verbin. "The Coin Problem and Pseudorandomness for Branching Programs". In: *Proceedings of the 51st Symposium on Foundations of Computer Science (FOCS)*. 2010, pp. 30–39. DOI: 10.1109/FOCS.2010.10. URL: https://www.cs.swarthmore.edu/~brody/papers/CoinProblemFullVersion.pdf.

[CDRST21] Gil Cohen, Dean Doron, Oren Renard, Ori Sberlo, and Amnon Ta-Shma. "Error reduction for weighted PRGs against read once branching programs". In: *Proceedings of the 36th Computational Complexity Conference*. 2021, 22:1–22:17. DOI: 10.4230/LIPIcs.CCC.2021.22.

[CDS22] Gil Cohen, Dean Doron, and Ori Sberlo. *Approximating Large Powers of Stochastic Matrices in Small Space*. ECCC preprint TR22-008. 2022. URL: https://eccc.weizmann.ac.il/report/2022/008/.

[CH20] Kuan Cheng and William M. Hoza. "Hitting Sets Give Two-Sided Derandomization of Small Space". In: *Proceedings of the 35th Computational Complexity Conference (CCC)*. 2020, 10:1–10:25. ISBN: 978-3-95977-156-6. DOI: 10.4230/LIPIcs.CCC.2020.10.

[CHHL19] Eshan Chattopadhyay, Pooya Hatami, Kaave Hosseini, and Shachar Lovett. "Pseudorandom Generators from Polarizing Random Walks". In: *Theory Comput.* 15 (2019), Paper No. 10. DOI: 10.4086/toc.2019.v015a010.

[CHRT18] Eshan Chattopadhyay, Pooya Hatami, Omer Reingold, and Avishay Tal. "Improved Pseudorandomness for Unordered Branching Programs through Local Monotonicity". In: *Proceedings of the 50th Symposium on Theory of Computing (STOC)*. 2018, pp. 363–375. DOI: 10.1145/3188745.3188800.

[CL20] Eshan Chattopadhyay and Jyun-Jie Liao. "Optimal Error Pseudodistributions for Read-Once Branching Programs". In: *Proceedings of the 35th Computational Complexity Conference (CCC)*. 2020, 25:1–25:27. ISBN: 978-3-95977-156-6. DOI: 10.4230/LIPIcs.CCC.2020.25.

[CPT21] Gil Cohen, Noam Peri, and Amnon Ta-Shma. "Expander random walks: a Fourier-analytic approach". In: *Proceedings of the 53rd Annual Symposium on Theory of Computing (STOC)*. 2021, pp. 1643–1655. DOI: 10.1145/3406325.3451049.

[CSV15] Sitan Chen, Thomas Steinke, and Salil Vadhan. *Pseudorandomness for Read-Once, Constant-Depth Circuits*. arXiv preprint 1504.04675. 2015. URL: https://arxiv.org/abs/1504.04675.

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

[DHH19] Dean Doron, Pooya Hatami, and William M. Hoza. "Near-Optimal Pseudorandom Generators for Constant-Depth Read-Once Formulas". In: *Proceedings of the 34th Computational Complexity Conference (CCC)*. 2019, 16:1–16:34. DOI: 10.4230/LIPIcs.CCC.2019.16.

[DHH20] Dean Doron, Pooya Hatami, and William M. Hoza. "Log-Seed Pseudorandom Generators via Iterated Restrictions". In: *Proceedings of the 35th Computational Complexity Conference (CCC)*. 2020, 6:1–6:36. ISBN: 978-3-95977-156-6. DOI: 10.4230/LIPIcs.CCC.2020.6.

[DMRTV21] Dean Doron, Raghu Meka, Omer Reingold, Avishay Tal, and Salil Vadhan. "Pseudorandom generators for read-once monotone branching programs". In: *Proceedings of the 25th International Conference on Randomization and Computation (RANDOM)*. 2021, 58:1–58:21. DOI: 10.4230/LIPIcs.APPROX/RANDOM.2021.58.

[FK18] Michael A. Forbes and Zander Kelley. "Pseudorandom Generators for Read-Once Branching Programs, in Any Order". In: *Proceedings of the 59th Symposium on Foundations of Computer Science (FOCS)*. 2018, pp. 946–955. DOI: 10.1109/FOCS.2018.00093.

[Gil77] John Gill. "Computational Complexity of Probabilistic Turing Machines". In: *SIAM J. Comput.* 6.4 (1977), pp. 675–695. ISSN: 0097-5397. DOI: 10.1137/0206049.

[GMRTV12] Parikshit Gopalan, Raghu Meka, Omer Reingold, Luca Trevisan, and Salil Vadhan. "Better Pseudorandom Generators from Milder Pseudorandom Restrictions". In: *Proceedings of the 53rd Symposium on Foundations of Computer Science (FOCS)*. 2012, pp. 120–129. DOI: 10.1109/FOCS.2012.77.

[GV22] Louis Golowich and Salil Vadhan. "Pseudorandomness of Expander Random Walks for Symmetric Functions and Permutation Branching Programs". In: *Proceedings of the 37th Computational Complexity Conference (CCC)*. 2022, 27:1–27:13. ISBN: 978-3-95977-241-9. DOI: 10.4230/LIPIcs.CCC.2022.27.

[GVW11] Oded Goldreich, Salil Vadhan, and Avi Wigderson. "Simplified Derandomization of BPP Using a Hitting Set Generator". In: *Studies in Complexity and Cryptography. Miscellanea on the Interplay between Randomness and Computation.* Vol. 6650. Lecture Notes in Comput. Sci. Springer, Heidelberg, 2011, pp. 59–67. DOI: 10.1007/978-3-642-22670-0_8.

[HHTT22] Pooya Hatami, William M. Hoza, Avishay Tal, and Roei Tell. "Fooling constant-depth threshold circuits". In: *Proceedings of the 62nd Annual Symposium on Foundations of Computer Science (FOCS)*. 2022 (albeit "FOCS 2021"), pp. 104–115. DOI: 10.1109/FOCS52979.2021.00019.

[HLV18] Elad Haramaty, Chin Ho Lee, and Emanuele Viola. "Bounded Independence Plus Noise Fools Products". In: *SIAM J. Comput.* 47.2 (2018), pp. 493–523. ISSN: 0097-5397. DOI: 10.1137/17M1129088.

[Hoz21] William M. Hoza. "Better pseudodistributions and derandomization for space-bounded computation". In: *Proceedings of the 25th International Conference on Randomization and Computation (RANDOM)*. 2021, 28:1–28:23. DOI: 10.4230/LIPIcs.APPROX/RANDOM.2021.28.

[HPV21] William M. Hoza, Edward Pyne, and Salil Vadhan. "Pseudorandom Generators for Unbounded-Width Permutation Branching Programs". In: *Proceedings of the 12th Innovations in Theoretical Computer Science Conference (ITCS)*. 2021, 7:1–7:20. ISBN: 978-3-95977-177-1. DOI: 10.4230/LIPIcs.ITCS.2021.7.

[HZ20] William M. Hoza and David Zuckerman. "Simple optimal hitting sets for small-success **RL**". In: *SIAM J. Comput.* 49.4 (2020), pp. 811–820. ISSN: 0097-5397. DOI: 10.1137/19M1268707.

[IMZ19] Russell Impagliazzo, Raghu Meka, and David Zuckerman. "Pseudorandomness from shrinkage". In: *J. ACM* 66.2 (2019), Art. 11, 16. ISSN: 0004-5411. DOI: 10.1145/3230630.

[INW94] Russell Impagliazzo, Noam Nisan, and Avi Wigderson. "Pseudorandomness for Network Algorithms". In: *Proceedings of the 26th Symposium on Theory of Computing (STOC)*. 1994, pp. 356–364. ISBN: 0897916638. DOI: 10.1145/195058.195190.

[Jun81] H. Jung. "Relationships between probabilistic and deterministic tape complexity". In: *Proceedings of the 10th Symposium on Mathematical Foundations of Computer Science (MFCS)*. 1981, pp. 339–346. DOI: 10.1007/3-540-10856-4_101.

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

[Lee19] Chin Ho Lee. "Fourier Bounds and Pseudorandom Generators for Product Tests". In: *Proceedings of the 34th Computational Complexity Conference (CCC)*. 2019, 7:1–7:25. DOI: 10.4230/LIPIcs.CCC.2019.7.

[LPV22] Chin Ho Lee, Edward Pyne, and Salil Vadhan. *Fourier Growth of Regular Branching Programs*. ECCC preprint TR22-034. 2022. URL: https://eccc.weizmann.ac.il/report/2022/034/.

[LV20] Chin Ho Lee and Emanuele Viola. "More on bounded independence plus noise: pseudorandom generators for read-once polynomials". In: *Theory Comput.* 16 (2020), Paper No. 7, 50. DOI: 10.4086/toc.2020.v016a007.

[Mic92] Pascal Michel. "A survey of space complexity". In: *Theoret. Comput. Sci.* 101.1 (1992), pp. 99–132. ISSN: 0304-3975. DOI: 10.1016/0304-3975(92)90151-5.

[MRSV21a] Jack Murtagh, Omer Reingold, Aaron Sidford, and Salil Vadhan. "Derandomization beyond connectivity: undirected Laplacian systems in nearly logarithmic space". In: *SIAM J. Comput.* 50.6 (2021), pp. 1892–1922. ISSN: 0097-5397. DOI: 10.1137/20M134109X.

[MRSV21b] Jack Murtagh, Omer Reingold, Aaron Sidford, and Salil Vadhan. "Deterministic approximation of random walks in small space". In: *Theory Comput.* 17 (2021), Paper No. 4, 35. DOI: 10.4086/toc.2021.v017a004.

[MRT19] Raghu Meka, Omer Reingold, and Avishay Tal. "Pseudorandom Generators for Width-3 Branching Programs". In: *Proceedings of the 51st Symposium on Theory of Computing (STOC)*. 2019, pp. 626–637. DOI: 10.1145/3313276.3316319.

[MZ13] Raghu Meka and David Zuckerman. "Pseudorandom Generators for Polynomial Threshold Functions". In: *SIAM J. Comput.* 42.3 (2013), pp. 1275–1301. ISSN: 0097-5397. DOI: 10.1137/100811623.

[Neč66] È. I. Nečiporuk. "On a Boolean function". In: *Dokl. Akad. Nauk SSSR* 169 (1966), pp. 765–766. ISSN: 0002-3264.

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

[NN93] Joseph Naor and Moni Naor. "Small-Bias Probability Spaces: Efficient Constructions and Applications". In: *SIAM J. Comput.* 22.4 (1993), pp. 838–856. ISSN: 0097-5397. DOI: 10.1137/0222053.

[NSW92] N. Nisan, E. Szemeredi, and A. Wigderson. "Undirected connectivity in $O(\log^{1.5} n)$ space". In: *Proceedings of the 33rd Annual Symposium on Foundations of Computer Science (FOCS)*. 1992, pp. 24–29. DOI: 10.1109/SFCS.1992.267822.

[NW94] Noam Nisan and Avi Wigderson. "Hardness vs. randomness". In: *J. Comput. System Sci.* 49.2 (1994), pp. 149–167. ISSN: 0022-0000. DOI: 10.1016/S0022-0000(05)80043-1. URL: https://doi.org/10.1016/S0022-0000(05)80043-1.

[NZ96] Noam Nisan and David Zuckerman. "Randomness is Linear in Space". In: *J. Comput. System Sci.* 52.1 (1996), pp. 43–52. ISSN: 0022-0000. DOI: 10.1006/jcss.1996.0004.

[PV21a] Edward Pyne and Salil Vadhan. "Limitations of the Impagliazzo-Nisan-Wigderson Pseudorandom Generator Against Permutation Branching Programs". In: *Proceedings of the 27th International Computing and Combinatorics Conference (COCOON)*. 2021, pp. 3–12. DOI: 10.1007/978-3-030-89543-3_1.

[PV21b] Edward Pyne and Salil Vadhan. "Pseudodistributions that beat all pseudorandom generators (extended abstract)". In: *Proceedings of the 36th Computational Complexity Conference (CCC)*. 2021, 33:1–33:15. DOI: 10.4230/LIPIcs.CCC.2021.33.

[PV22] Edward Pyne and Salil Vadhan. "Deterministic Approximation of Random Walks via Queries in Graphs of Unbounded Size". In: *Proceedings of the 5th Symposium on Simplicity in Algorithms (SOSA)*. 2022, pp. 57–67. DOI: 10.1137/1.9781611977066.5.

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

[RSV13] Omer Reingold, Thomas Steinke, and Salil Vadhan. "Pseudorandomness for Regular Branching Programs via Fourier Analysis". In: *Proceedings of the 17th International Workshop on Randomization and Computation (RANDOM)*. 2013, pp. 655–670. DOI: 10.1007/978-3-642-40328-6_45.

[RTV06] Omer Reingold, Luca Trevisan, and Salil Vadhan. "Pseudorandom Walks on Regular Digraphs and the **RL** vs. **L** Problem". In: *Proceedings of the 38th Symposium on Theory of Computing (STOC)*. 2006, pp. 457–466. DOI: 10.1145/1132516.1132583.

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

[Sak96] Michael Saks. "Randomization and Derandomization in Space-Bounded Computation". In: *Proceedings of the 11th Conference on Computational Complexity (CCC)*. 1996, pp. 128–149. DOI: 10.1109/CCC.1996.507676.

[Sim77] Janos Simon. "On the difference between one and many (preliminary version)". In: *Proceedings of the 4th International Colloquium on Automata, Languages and Programming (ICALP)*. 1977, pp. 480–491. DOI: 10.1007/3-540-08342-1_37.

[Sim81] Janos Simon. "On tape-bounded probabilistic Turing machine acceptors". In: *Theoret. Comput. Sci.* 16.1 (1981), pp. 75–91. ISSN: 0304-3975. DOI: 10.1016/0304-3975(81)90032-3.

[ST04] Daniel A. Spielman and Shang-Hua Teng. "Nearly-linear time algorithms for graph partitioning, graph sparsification, and solving linear systems". In: *Proceedings of the 36th Annual ACM Symposium on Theory of Computing (STOC)*. ACM, New York, 2004, pp. 81–90. DOI: 10.1145/1007352.1007372.

[Ste12] Thomas Steinke. *Pseudorandomness for Permutation Branching Programs Without the Group Theory*. ECCC preprint TR12-083. 2012. URL: https://eccc.weizmann.ac.il/report/2012/083/.

[SVW17] Thomas Steinke, Salil Vadhan, and Andrew Wan. "Pseudorandomness and Fourier-Growth Bounds for Width-3 Branching Programs". In: *Theory Comput.* 13 (2017), Paper No. 12. DOI: 10.4086/toc.2017.v013a012.

[SZ99] Michael Saks and Shiyu Zhou. "$\mathrm{BP_H SPACE}(S) \subseteq \mathrm{DSPACE}(S^{3/2})$". In: *J. Comput. System Sci.* 58.2 (1999), pp. 376–403. ISSN: 0022-0000. DOI: 10.1006/jcss.1998.1616.

[Tri08] Vladimir Trifonov. "An $O(\log n \log\log n)$ space algorithm for undirected *st*-connectivity". In: *SIAM J. Comput.* 38.2 (2008), pp. 449–483. ISSN: 0097-5397. DOI: 10.1137/050642381.

[Tzu09] Yoav Tzur. “Notions of Weak Pseudorandomness and $GF(2^n)$-Polynomials”. M.Sc. thesis. Weizmann Institute of Science, 2009. URL: https://eccc.weizmann.ac.il/static/books/Notions_of_Weak_Pseudorandomness/.

[Vad12] Salil P. Vadhan. “Pseudorandomness”. In: *Foundations and Trends® in Theoretical Computer Science* 7.1–3 (2012), pp. 1–336. ISSN: 1551-305X. DOI: 10.1561/0400000010.



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