8 Conclusions

8.1 A Unified Theory of Pseudorandomness

In the course of this survey, we have seen that a wide variety of "pseudorandom objects" are very closely related, and indeed almost equivalent when viewed appropriately: list-decodable codes, averaging samplers, expander graphs, randomness extractors, (black-box) pseudorandom generator constructions, and hardness amplifiers. The power of these connections comes from the fact they allow us to translate intuitions and techniques that are very natural for one of these objects to others where they have been might be difficult to discover. Indeed, we have seen several examples of this, such the use of randomness extractors to construct near-optimal samplers (Problem 6.5), the use of Parvaresh–Vardy codes to construct bipartite expanders (Theorem 5.35) and lossless condensers (Theorem 6.34), and the use of the Nisan–Wigderson pseudorandom generator construction to construct extractors (Theorem 7.73).

Table 8.1 reviews the connections between these objects, by putting them in our list-decoding framework. Recall that we can view each of the objects as a function $\Gamma : [N] \times [D] \to [M]$ and can characterize their properties by bounding the sizes of sets of the form $\mathrm{LIST}_\Gamma(T, \alpha) = \{x : \Pr_y[\Gamma(x,y) \in T] > \alpha\}$, where $T \subset [M]$. (When $\alpha = 1$, we change

Table 8.1. Capturing Pseudorandom Objects $\Gamma : [N] \times [D] \to [M]$ by bounding sizes of $\mathrm{LIST}_\Gamma(T,\alpha) = \{x : \Pr_y[\Gamma(x,y) \in T] > \alpha\}$ for $T \subset [M]$. The parameter γ denotes a arbitrarily small positive constant. As usual, $N = 2^n$, $M = 2^m$, $D = 2^d$, and $K = 2^k$.

Object	$\Gamma(x,y)$	List-Decoding Problem	Decoding Time	Standard Parameters	Prop.
$(1 - 1/q - \varepsilon, K)$ list-decodable codes	$(y, \mathrm{Enc}(x)_y)$	$T = \{(y, r_y)\} \Rightarrow$ $\lvert\mathrm{LIST}(T, \mu(T) + \varepsilon)\rvert \le K$	$\mathrm{poly}(n, 1/\varepsilon)$	$q = O(1)$, $\varepsilon = \Omega(1)$, $M = Dq = O(n)$, $K = \mathrm{poly}(n)$	5.29
(t,k,ε) black-box hardness amplifiers	$(y, \mathrm{Amp}^f(y))$	$T = \{(y, r_y)\} \Rightarrow$ $\lvert\mathrm{LIST}(T, \mu(T) + \varepsilon)\rvert \le K$	$t = \mathrm{poly}(\log n, 1/\varepsilon)$ (local w/advice)	$d = O(\log n)$, $q = 2$, $M = Dq = \mathrm{poly}(n)$, $k = t = \mathrm{poly}(\log n, 1/\varepsilon) \le n^\gamma$	7.74
(δ,ε) averaging samplers	$\mathrm{Samp}(x)_y$	$\lvert\mathrm{LIST}(T, \mu(T) + \varepsilon)\rvert \le K$	don't care	$K = \delta N$, $D = \mathrm{poly}(1/\varepsilon, \log(1/\delta))$, $n = O(m + \log(1/\delta))$	5.30
(k,ε) extractors	$\mathrm{Ext}(x,y)$	$\lvert\mathrm{LIST}(T, \mu(T) + \varepsilon)\rvert \lesssim K$	don't care	$d = O(\log(n/\varepsilon))$, $k = O(m) \in [\mathrm{polylog}(n), \gamma n]$	6.23
(t,k,ε) black-box PRGs	$G^x(y)$	$\lvert\mathrm{LIST}(T, \mu(T) + \varepsilon)\rvert \le K$	$t = \mathrm{poly}(m)$ (local w/advice)	$\varepsilon = 1/m$, $k = t = \mathrm{poly}(m) \in [\mathrm{polylog}(n), n^\gamma]$	7.72
$(= K, A)$ expanders	y'th nbr of x	$\lvert T\rvert < AK \Rightarrow$ $\lvert\mathrm{LIST}(T, 1)\rvert < K$	don't care	$D = O(1)$, $A = 1 + \Omega(1)$, $K = N/2$, $M = N$	5.33
(δ,ε) hitting samplers	$\mathrm{Samp}(x)_y$	$\lvert T\rvert < (1-\varepsilon)M \Rightarrow$ $\lvert\mathrm{LIST}(T, \mu(T) + \varepsilon)\rvert \le K$	don't care	$K = \delta N$, $D = \mathrm{poly}(1/\varepsilon, \log(1/\delta))$, $n = O(m + \log(1/\delta))$	Prob. 4.7
$k \to_\varepsilon k + d$ (lossless) condensers	$\mathrm{Con}(x,y)$	$\lvert T\rvert < (1-\varepsilon)DK \Rightarrow$ $\lvert\mathrm{LIST}(T, 1)\rvert < K$	don't care	$d = O(\log(n/\varepsilon))$, $k = O(m) \in [\mathrm{polylog}(n), \gamma n]$	6.33

the inequality to an equality.) Other objects we have encountered, such as dispersers (Definition 6.19), black-box hitting-set generator constructions (Problem 7.8), and lossy condensers (Definition 6.32) can also be cast in the framework, but we leave this as an exercise.

In addition to illustrating the connections between the objects, Table 8.1 also brings out their differences. In some cases $\alpha = \mu(T) + \varepsilon$, and in other cases $\alpha = 1$. In some cases (samplers, extractors, PRG constructions), we consider all subsets $T \subset [M]$, and in other cases (codes, hardness amplifiers, expanders, condensers), we only consider T that are small and possibly have some additional structure, like being the graph of a received word. (Typically, the only way the graph structure is used is to determine that the size of T is D.) In some cases, we want efficient algorithms to construct $\mathrm{LIST}(T, \alpha)$ (even polylogarithmic-time local decoding algorithms), and in others, all we care about is its size.

The most common parameter ranges also vary quite widely among the objects. The typical relation between M and N ranges from logarithmic (codes) to polylogarithmic (hardness amplifiers) to a constant power (samplers) to a constant multiplicative factor (expanders). (Extractors, PRG constructions, and condensers all consider the full range between polylogarithmic and a constant power.) The typical value of D ranges from being a constant independent of N (expanders), to being logarithmic in N (codes), to being polylogarithmic in N (extractors, condensers, PRG constructions, hardness amplifiers). Despite these differences, we have seen that the connections are nevertheless quite powerful, and ideas or techniques developed for one of these objects are often useful for the others.

The unified framework of Table 8.1 opens up a tantalizing possibility that we can also have a unified *construction* of all of these objects. In what follows, we will argue that this is possible ignoring explicitness/efficiency considerations (including local list-decoding). The bounds will be stated in terms of the tails of the binomial distribution:

Definition 8.1 (tails of binomial distributions). For $t \in \mathbb{N}$, $\mu, \alpha \in (0,1)$, define

$$\mathrm{Bin}(t, \mu, \alpha) = \Pr\left[\frac{1}{t}\sum_{i=1}^{t} X_i > \alpha\right],$$

where the X_is are iid $\{0,1\}$-valued random variables such that $\Pr[X_i = 1] = \mu$, and define

$$\mathrm{Bin}(t,\mu,1) = \Pr\left[\frac{1}{t}\sum_{i=1}^{t} X_i = 1\right] = \mu^t.$$

With this definition, we can give the following unified construction:

Theorem 8.2 (nonconstructive unified pseudorandom object). For every $N, M, D \in \mathbb{N}$, there exists a function $\Gamma : [N] \times [D] \to [M]$ such that for every $T \subset [M]$ and every $\alpha \in (0,1]$, we have

$$\begin{aligned}&|\mathrm{LIST}_\Gamma(T,\alpha)| \\ &\quad \le \min\left\{K \in \mathbb{N} : \binom{N}{K} \cdot \binom{M}{|T|} \cdot \mathrm{Bin}(D,\mu(T),\alpha)^K \le \frac{1}{4K^2|T|^2}\right\}.\end{aligned}$$

We leave the proof of Theorem 8.2 as an exercise and instead show how many of the nonconstructive bounds we've seen for various pseudorandom objects would simultaneously hold for the Γ of Theorem 8.2 by setting parameters appropriately:

- The nonconstructive bounds for expanders (Theorem 4.4) can be obtained by setting $M = N$, $\alpha = 1$, $\mu = \mu(T)$, $K = |T|/(D-2)$ (ignoring round-off errors), and noting that

$$\begin{aligned}&\binom{N}{K} \cdot \binom{N}{|T|} \cdot \mathrm{Bin}(D,\mu,1)^K \\ &\quad \le \left(\frac{Ne}{K}\right)^K \cdot \left(\frac{Ne}{|T|}\right)^{|T|} \cdot \mu^{DK} \\ &\quad = \left(\frac{(D-2)\cdot e}{\mu}\right)^K \cdot \left(\frac{e}{\mu}\right)^{(D-2)\cdot K} \cdot \mu^{D\cdot K} \\ &\quad = ((D-2)e^{D-1}\mu)^K \\ &\quad < 1/(4K^2|T|^2),\end{aligned}$$

  provided μ is sufficiently small. Thus, by Proposition 5.33, Γ defines a $(\gamma N, (D-2))$ vertex expander for a sufficiently small γ.

More generally, we see that for any constants $\gamma, A, D > 0$ and sufficiently large $N = M$, Γ will be a $(\gamma N, A)$ expander provided that $D > (H(\gamma) + H(\gamma A))/\gamma \log(1/\gamma A)$, where $H(x) = x\log(1/x) + (1-x)\log(1/(1-x))$ is the binary entropy function.[1] (To see this, use the fact that $\binom{N}{\beta N} = 2^{(H(\beta)+o(1))\cdot N}$ as $N \to \infty$.)

- For the nonconstructive bound on extractors (Theorem 6.14) and averaging samplers, we set $\alpha = \mu(T) + \varepsilon$ and use the bounds $\binom{N}{K} \le (Ne/K)^K$, $\binom{M}{T} \le 2^M$, and $\mathrm{Bin}(D, \mu(T), \mu(T) + \varepsilon) \le \exp(-\Omega(D\varepsilon^2))$ (by the Chernoff Bound of Theorem 2.21). Then the condition of Theorem 8.2 is satisfied provided that $D \ge (c/\varepsilon^2) \cdot \log(N/K)$ and $M \le \varepsilon^2 KD/c$ for a sufficiently large constant c.

To get the "strong" forms of pseudorandom objects, where $\Gamma(x,y) = (y, \Gamma'(x,y))$ for some function Γ', we need to consider the tails of sums of independent, but not necessarily identically distributed, binomial random variables:

Definition 8.3 (tails of Poisson binomial distributions). For $t, \mathbb{N}, \mu, \alpha \in (0,1)$, define

$$\mathrm{PBin}(t, \mu, \alpha, \eta) = \sup_{(Y_1,\ldots,Y_t)} \Pr\left[\frac{1}{t}\sum_{i=1}^{t} Y_i > \alpha\right],$$

and

$$\mathrm{PBin}(t, \mu, 1, \eta) = \sup_{(Y_1,\ldots,Y_t)} \Pr\left[\frac{1}{t}\sum_{i=1}^{t} Y_i = 1\right] = \mathrm{Bin}(t, \mu, 1),$$

where the suprema are taken over all sequences of independent Bernoulli random variables $Y_1, \ldots, Y_t$ such that $(1/t) \cdot \sum_{i=1}^{t} \mathrm{E}[Y_i] \le \mu$, and additionally $\mathrm{E}[Y_i]$ is an integer multiple of η for every i.

[1] The discussion after Theorem 4.4 states a slightly stronger result, only requiring $D > (H(\gamma) + H(\gamma A))/(H(\gamma) - \gamma A H(1/\gamma))$, but this is based on choosing a graph that is the union of D random perfect matchings, which does indeed have slightly better expansion properties than the model we are using now, of choosing D random neighbors for each left vertex.

Theorem 8.4 (nonconstructive unified pseudorandom object). For every $N,q,D \in \mathbb{N}$, there exists a function $\Gamma' : [N] \times [D] \to [q]$ such that if we define $\Gamma(x,y) = (y,\Gamma'(x,y))$, then for every $T \subset [D] \times [q]$ and every $\alpha \in (0,1]$, we have

$$|\mathrm{LIST}_\Gamma(T,\alpha)| \\ \leq \min\left\{K \in \mathbb{N} : \binom{N}{K} \cdot \binom{Dq}{|T|} \cdot \mathrm{PBin}(D,\mu(T),\alpha,1/q)^K \leq \frac{1}{4K^2|T|^2}\right\}.$$

- Nonconstructive bounds for strong vertex expanders, hitting samplers, and lossless condensers (matching the bounds for the non-strong ones) follow by noting that $\mathrm{PBin}(n,\mu,1,\eta) \leq \mathrm{Bin}(n,\mu,1)$. Indeed, writing μ_i for the expectation of Bernoulli random variable Y_i, then

$$\Pr\left[\frac{1}{t}\sum_{i=1}^{t} Y_i = 1\right] = \prod_{i=1}^{t} \mu_i \leq \left(\frac{1}{t}\sum_{i=1}^{t}\mu_i\right)^t \leq \mu^t,$$

  where the second-to-last inequality follows from the Arithmetic Mean–Geometric Mean Inequality.
- Nonconstructive bounds for strong extractors and averaging samplers (matching the bounds for the non-strong ones) follow because the Chernoff Bound (Theorem 2.21) also applies to nonidentical random variables: $\mathrm{PBin}(D,\mu(T),\mu(T)+\varepsilon,\eta) \leq \exp(-\Omega(D\varepsilon^2))$ for every $\eta > 0$.
- Nonconstructive bounds for list-decodable codes (Theorem 5.8) follow because the list-decodability of codes involves taking T to be the graph of a received word (cf. Proposition 5.29) and hence $\mu(T) = 1/q$, and $\mathrm{PBin}(D,1/q,1-\delta,1/q) = q^{H_q(\delta,D)\cdot D}/q^D$. To see the latter, let $\mathrm{E}[Y_i] = m_i/q$ for $m_i \in \mathbb{N}$ such that $\sum_i m_i \leq D$. so Y_i can be viewed as the indicator for the event the $R(i) \in \{1,\ldots,m_i\}$ for a uniformly random received word $R : [D] \to [q]$, and $\Pr[\sum_i Y_i > 1-\delta]$ is the probability that the graph $G(R) = \{(i,r(i)) : i \in [D]\}$ of the received word R intersects the set

$T' = \{(i,j) : i \in [D], j \leq m_i\}$ in more than $(1-\delta)D$ positions. For a set $S \subset T'$, the probability that $G(R) \cap T' = S$ is at most $(1/q)^{|S|} \cdot (1-1/q)^{D-|S|}$. (This is exactly the probability in case S contains only pairs (i,j) with D distinct values of i, otherwise the probability is zero.) Thus,

$$\begin{aligned}\Pr\left[\sum_i Y_i > 1-\delta\right] &\leq \sum_{s>(1-\delta)D} \binom{\sum_i m_i}{s} (1/q)^s \cdot (1-1/q)^{D-s} \\ &\leq \frac{1}{q^D} \cdot \sum_{s>(1-\delta)D} \binom{D}{s} (q-1)^s \\ &= \frac{q^{H_q(\delta,D)\cdot D}}{q^D}.\end{aligned}$$

Now, given this bound on $\mathrm{PBin}(\cdot)$ and Proposition 5.29, Theorem 8.4 gives us a (δ,K) list-decodable code $\mathrm{Enc} : [N] \to [q]^D$ provided that $\binom{N}{K} \cdot \binom{Dq}{D} \cdot (\frac{q^{H_q(\delta,D)\cdot D}}{q^D})^K \leq \frac{1}{4K^2D^2}$. Using the bounds $\binom{N}{K} \leq N^K$ and $\binom{Dq}{D} \leq (eq)^D$, we see that we get a list-decodable code provided that the rate $\rho = \log N / D \log q$ is smaller than $H_q(\delta,D) - O(1/K) - O((\log KD)/(KD\log q)$. This matches the nonconstructive bound for list-decodable codes (Theorem 5.8) up to the dependence of the error term on the list size K.

Given the above, a natural long-term goal for the theory of pseudorandomness is the following:

Open Problem 8.5. Can we give explicit constructions of functions Γ that nearly match the bounds of Theorems 8.2 and 8.4? Can this be done with efficient (local) list-decoding algorithms?

8.2 Other Topics in Pseudorandomness

In this section, we survey some additional topics in the theory of pseudorandomness that we did not cover in depth, each of which could merit at least an entire section on its own. Sections 7.2 and 7.9

contained a detailed survey of cryptographic pseudorandomness, so we do not discuss it again here.

8.2.1 Pseudorandomness for Space-Bounded Computation

As discussed in Section 4.4, unlike for **BPP** and **RP**, we do know very nontrivial derandomizations **RL** (and **BPL**). These derandomizations are obtained constructions of pseudorandom generators $G : \{0,1\}^d \to \{0,1\}^m$ such that no randomized $(\log m)$-space algorithm can distinguish $G(U_d)$ from U_m. In order to get derandomizations that are correct on every input x, we require pseudorandom generators that fool *nonuniform* space-bounded algorithms. On the other hand, since randomized space-bounded algorithms get their random bits as a stream of coin tosses, we only need to fool space-bounded distinguishers that read each of their input bits once, in order. Thus, we consider distinguishers that are *(oblivious, read-once) branching programs*, which maintain a state $s_i \in [w]$ after reading i input bits, and determine the next state $s_{i+1} \in [w]$ as a function of s_i and the $(i+1)$th input bit. The number w of available states at each time step is called the *width* of the branching program, and corresponds to a space bound of $\log w$ bits. A generator $G : \{0,1\}^{d(m)} \to \{0,1\}^m$ that is computable in space $O(d(m))$ and such that $G(U_{d(m)})$ cannot be distinguished from U_m by oblivious, read-once branching programs of width m implies that $\mathbf{BPL} \subset \mathbf{DSPACE}(O(d(\text{poly}(m))))$.

The fact that pseudorandom generators imply lower bounds (Problem 7.1) applies to this context too, but fortunately we do know exponential width lower bounds for *oblivious, read-once* branching programs (e.g., computing the inner-product modulo 2 of two ℓ-bit strings requires width $2^{\Omega(\ell)}$). On the other hand, we cannot simply plug such lower bounds into the Nisan–Wigderson generator construction (Theorem 7.24), because the reductions used do not preserve the read-once property (and we do not know superpolynomial lower-bounds for branching programs that can read each input bit many times).

Nevertheless, a series of papers starting with Ajtai, Komlós, and Szemerédi [11] have given *unconditional* pseudorandom generators for space-bounded computation. The pseudorandom generator of

Nisan [299] uses a seed of length $O(\log^2 m)$ to produce m bits that are pseudorandom to oblivious, read-once branching programs of width m. At first, this only seems to imply that $\mathbf{RL} \subset \mathbf{L}^2$, which already follows from Savitch's Theorem [349] that $\mathbf{NL} \subset \mathbf{L}^2$. Nevertheless, Nisan's generator and its additional properties has been used in more sophisticated ways to obtain highly nontrivial derandomizations of **RL**. Specifically, Nisan [300] used it to show that every problem in **RL** can be solved simultaneously in polynomial time and $O(\log^2 n)$ space, and Saks and Zhou [344] used it to prove that $\mathbf{RL} \subset \mathbf{L}^{3/2}$. Another important pseudorandom generator for space-bounded computation is that of Nisan and Zuckerman [303], which uses a seed of length $O(\log m)$ to produce $\log^k m$ bits that are pseudorandom to oblivious, read-once branching programs of width m. None of these results have been improved in nearly two decades.

However, substantially better generators have been constructed for restricted classes of oblivious read-once branching programs. Specifically, there are pseudorandom generators or hitting-set generators (see Problem 7.8) stretching a seed of length $\tilde{O}(\log m)$ to m bits that fool combinatorial rectangles (which check membership in a "rectangle" $S_1 \times S_2 \times \cdots \times S_{m/\log m}$, where each $S_i \subset \{0,1\}^{\log m}$) [136, 264, 31, 271, 179], branching programs of width 2 and 3 [345, 73, 363, 179], constant-width *regular* branching programs (where the transition function at each layer is regular) [82, 84], and constant-width *permutation* branching programs (where each input string induces a permutation of the states at each layer) [338, 250, 113, 373]. However, the following remains open:

Open Problem 8.6. Is there an explicit pseudorandom generator $G : \{0,1\}^{o(\log^2 m)} \to \{0,1\}^m$ whose output distribution is pseudorandom to oblivious, read-once branching programs of width 4?

8.2.2 Derandomization vs. Lower Bounds

Derandomization from Uniform Assumptions. The construction of pseudorandom generators we have seen (Theorem 7.63) requires nonuniform circuit lower bounds for functions in **E**, and it is of interest to find constructions that only require uniform hardness assumptions. One way to obtain such results is to show that uniform hardness assumptions imply nonuniform assumptions. Indeed, the Karp–Lipton Theorems in complexity theory [235] show that certain strong uniform lower bounds imply nonuniform lower bounds. For example, if an **EXP**-complete problem cannot be solved by a uniform algorithm in the second level of the polynomial-time hierarchy (like **NP** but with two nondeterministic quantifiers, cf. Section 3.3), then it also cannot be solved by polynomial-sized circuits. Subsequent strengthenings of the Karp–Lipton Theorem (based on the theory of interactive proofs) show that if $\mathbf{EXP} \neq \mathbf{MA}$ (where **MA** is like **NP** but allows probabilistic verification of witnesses), then $\mathbf{EXP} \not\subseteq \mathbf{P/poly}$ [42]; consequently one gets pseudorandom generators with arbitrary polynomial stretch (secure for infinitely many input lengths) under the assumption $\mathbf{EXP} \neq \mathbf{MA}$ [43].

Assuming the Karp–Lipton Theorem cannot be further improved (e.g., to show $\mathbf{EXP} \not\subseteq \mathbf{P/poly} \Leftrightarrow \mathbf{EXP} \neq \mathbf{BPP}$), from a uniform assumption such as $\mathbf{EXP} \neq \mathbf{BPP}$, we can only hope to construct pseudorandom generators that are secure against *uniform distinguishers* (because pseudorandom generators secure against nonuniform distinguishers imply nonuniform lower bounds, by Problem 7.1). In the context of cryptographic pseudorandom generators, uniform results were typically developed together with or soon after the corresponding nonuniform results. Indeed, analogously to Theorem 7.11, it is known that cryptographic pseudorandom generators that are secure against uniform distinguishers exist if and only if there exist one-way functions that are hard to invert by uniform algorithms [197]. For noncryptographic, mildly explicit pseudorandom generators as in Theorem 7.63 and Corollary 7.64, an obstacle is that black-box constructions (Definition 7.65) require nonuniform advice in the reduction. (See discussion at the end of Section 7.7.2. This obstacle is avoided in the case of cryptographic pseudorandom generators, because the appropriate definition of black-box construction from one-way functions gives the reduction oracle access to the one-way function in addition to the distinguisher, since one-way functions are supposed to be efficiently computable by definition.)

Thus, we must turn to *non-black-box* constructions, in which we make more use of the fact that the hard function f is computable in $\mathbf{E}$ and/or the fact that the distinguisher T is computable by an efficient probabilistic algorithm, not just to deduce that G^f is mildly explicit and Red^T is efficient. In fact, f and T need not even be used as oracles; we can make use of the code of the programs computing these functions (e.g., to reduce f to an $\mathbf{E}$-complete problem). While at first it may seem difficult to take advantage of non-black-box constructions, this was eventually accomplished by Impagliazzo and Wigderson [216]. They showed that if $\mathbf{EXP} \neq \mathbf{BPP}$, then there are mildly explicit pseudorandom generators with polynomial stretch that are secure against uniform probabilistic distinguishers (for infinitely many input lengths), and hence **BPP** has subexponential-time average-case derandomizations (by Problem 7.9). (See [397] for a precise statement regarding the construction of pseudorandom generators.) This is a uniform analogue of the "low-end" nonuniform result in Corollary 7.64 (Item 3). Analogues for the high-end bounds (Items 1, 2) remain open. For example:

Open Problem 8.7. Does $\mathbf{EXP} \not\subseteq \mathbf{BPSUBEXP}$ imply mildly explicit generators $G : \{0,1\}^{\mathrm{polylog}(m)} \to \{0,1\}^m$ whose input is pseudorandom to every uniform probabilistic algorithm running in time m (for infinitely many m)?

Such a result is known if we replace **EXP** by **PSPACE** [397].

For derandomizing **AM** instead of **BPP**, both high-end and low-end uniform results have been obtained [195, 358]. These results utilize hard functions in $\mathbf{E}$, unlike the nonuniform results which only require hard functions in $\mathbf{NE} \cap \mathbf{co\text{-}NE}$ (cf. Theorem 7.68).

Derandomization Implies Circuit Lower Bounds. Since uniform hardness assumptions and PRGs against uniform distinguishers only seem to imply average-case derandomizations (Problem 7.9), it is tempting to conjecture that *worst-case* derandomizations imply (or are even equivalent to) nonuniform circuit lower bounds. A result of this type was first given implicitly by Buhrman, Fortnow, and

Thierauf [86] and then explicitly and in stronger form by Impagliazzo, Kabanets, and Wigderson [212]. Specifically, these results show that if **MA** (like **NP**, but with probabilistic verification of witnesses) can be derandomized (e.g., $\mathbf{MA} = \mathbf{NP}$ or even $\mathbf{MA} \subset \mathbf{NSUBEXP}$), then $\mathbf{NEXP} \not\subseteq \mathbf{P/poly}$. Derandomization of **prBPP** implies derandomization of **MA**, so this also implies that if $\mathbf{prBPP} = \mathbf{prP}$ or even $\mathbf{prBPP} \subset \mathbf{prSUBEXP}$, then $\mathbf{NEXP} \not\subseteq \mathbf{P/poly}$. This result falls short of giving a converse to Corollary 7.64 (Item 3) because the circuit lower bounds are for **NEXP** rather than **EXP**. (Corollary 7.64, as well as most of the other derandomization results we've seen, apply equally well to **prBPP** as to **BPP**.) In addition, the result does not give exponential circuit lower bounds even if we assume full derandomization ($\mathbf{prBPP} = \mathbf{prP}$). However, Santhanam [347] shows that $\mathbf{prBPP} = \mathbf{prP}$ implies that for every constant k, there is a language in **NP** that does not have circuits of size n^k, which can be viewed as a "scaled down" version of the statement that **NE** requires circuits of size $2^{\Omega(n)}$.[2]

Thus, the following remain open:

Open Problem 8.8. Does $\mathbf{prBPP} = \mathbf{prP}$ imply $\mathbf{E} \not\subseteq \mathbf{P/poly}$ (equivalently $\mathbf{EXP} \not\subseteq \mathbf{P/poly}$, by Problem 7.2)?

Open Problem 8.9. Does $\mathbf{prBPP} = \mathbf{prP}$ imply that **NEXP** has a problem requiring nonuniform boolean circuits of size $2^{\ell^{\Omega(1)}}$ on inputs of length ℓ?

By the result of [212] and Corollary 2.31, finding a deterministic polynomial-time algorithm for the **prBPP**-complete problem $[+\varepsilon]$-Approx Circuit Average implies superpolynomial circuit lower bounds for **NEXP**. Unfortunately, we do not know a wide variety of natural problems that are complete for **prBPP** (unlike **NP**). Nevertheless, Kabanets and Impagliazzo [227] showed that finding a deterministic polynomial-time algorithm for Polynomial Identity

[2] Indeed, by a standard "padding" or "translation" argument in complexity theory, if for some constant k, every language in **NP** had circuits of size n^k, then every language in **NE** would have circuits of size $2^{o(n)}$.

Testing implies that either **NEXP** $\not\subseteq$ **P/poly** or that the Permanent does not have polynomial-sized arithmetic circuits, both of which are long-standing open problems in complexity theory. (See [1] for a simpler and somewhat stronger proof.) Polynomial Identity Testing is in **co-RP**, by Theorem 2.12, but is not known to be complete for any randomized complexity class. Of course, it is also of interest to find additional complete problems for **prBPP**:

Open Problem 8.10. Find combinatorial or algebraic complete problems for any randomized complexity classes (e.g., **prBPP**, **prRP**, **prAM**, **BPP**, **RP**, **ZPP**).

Derandomizations of **prAM** are also known to imply circuit lower bounds, which are stronger than what the aforementioned results give from derandomizations of **prBPP** in that they either yield exponential-size bounds [38] or give lower bounds for nondeterministic circuits [39].

One interesting interpretation of many of these results is to show that if derandomization is possible via any means (for all of **prBPP** or **prAM**), then it can be done in a canonical way — via pseudorandom generators (because these results show that derandomization implies circuit lower bounds, which in turn imply pseudorandom generators via Theorem 7.63). A recent work by Goldreich [163] directly proves equivalences between various kinds of derandomizations of **prBPP** (e.g., worst-case or average-case) and various forms of pseudorandom generators, without going through circuit lower bounds. (See Problem 7.10.)

Another reason for the interest in these results is that they suggest derandomization as a potential approach to proving circuit lower bounds. Indeed, derandomization results have played a role in some state-of-the-art lower bounds, namely the result of Buhrman, Fortnow, and Thierauf [86] that **MAEXP** (the exponential-time analogue of **MA**) is not in **P/poly**, and the result of Williams [418, 419] that **NEXP** is not in **ACC** (which is defined like $\mathbf{AC^0}$ but also allowing unbounded fan-in gates that test whether their inputs sum to zero modulo m, for any constant m). The result of Kabanets and Impagliazzo [227] (along with [7]) has also been one of the motivations for the recent line of work on derandomizing Polynomial Identity Testing for low-depth arithmetic circuits. (See the Notes and References of Section 2.)

8.2.3 Hardness Amplification

As discussed in Section 7.6.1, *hardness amplification* is the task of taking a computational problem that is mildly hard on average and making it much more hard on average. Hardness amplification was introduced by Yao, in oral presentations of his paper [421]. Specifically, he suggested the "Direct Product" construction $f'(x_1,\ldots,x_k) = (f(x_1),\ldots,f(x_k))$ to convert a weak one-way function f (one that is mildly hard to invert) into a standard "strong" one-way function f' (satisfying Definition 7.10), and an "XOR Lemma" showing that if a Boolean function f is mildly hard to compute, then $f'(x_1,\ldots,x_k) = f(x_1) \oplus f(x_2) \oplus \cdots f(x_k)$ is very hard on average to compute. These were tools used in his proof that weak one-way permutations imply pseudorandom generators. These results have been generalized and strengthened in a number of ways:

Quantitative Bounds: It is of interest to have tight bounds on the hardness of f' as a function of the hardness of f and k. For the Direct Product construction, if f is δ average-case-hard, then intuitively we expect f' to be roughly $(1-(1-\delta)^k)$ average-case-hard, corresponding to the fact that an efficient algorithm trying to compute f' should have probability at most $1-\delta$ of solving each of the k instances correctly. Similarly, in the case of the XOR Lemma, we expect that if f is $(1-\delta)/2$ average-case-hard, then f' should be roughly $(1-\delta^k)/2$-average-case hard. Levin [259] proved a version of the XOR Lemma that gives such a tight bound, and his proof technique also extends to the Direct Product construction (see [171]).

Derandomization: Goldreich, Impagliazzo, Levin, Venkatesan, and Zuckerman [166] gave derandomized hardness amplification results for converting weak one-way permutations and weak regular one-way functions into strong ones, where the inputs x_i are not independent but are generated in some pseudorandom way from a short seed that is the input to f'. Impagliazzo and Wigderson [208, 215] gave derandomized versions of the XOR Lemma and a Direct Product Lemma (for hard-to-compute boolean functions). These results were used in the first proof, due to [215], that $\mathbf{P} = \mathbf{BPP}$ if $\mathbf{E}$ requires exponential-size circuits. Recall that the proof we saw in Section 7 avoids hardness amplification, and instead goes directly from worst-case hardness to strong average-case hardness via locally list-decodable codes (following [381]). Nevertheless, hardness amplification is still of interest because it can be implemented in lower complexity than worst-case-to-average-case amplification. Indeed, hardness amplification (starting from mild average-case hardness) can be implemented in polynomial time with oracle access to f, whereas Viola [411] has shown that black-box worst-case-to-average-case amplification (per Definition 7.74) cannot be implemented in the polynomial-time hierarchy (due to needing to compute a list-decodable encoding of the truth table of f). Indeed, another line of work has investigated hardness amplification for functions in **NP**.

Hardness Amplification in NP: The study of this topic was initiated in the work of O'Donnell [304]. The goal is to show that if **NP** has a function f that is mildly hard on average, then it has a function f' that is very hard on average. Yao's XOR Lemma does not prove this because f being in **NP** does not imply that $f'(x_1,\ldots,x_k) = f(x_1) \oplus \cdots \oplus f(x_k)$ is also in **NP**, assuming that $\mathbf{NP} \neq \mathbf{co\text{-}NP}$ (which is commonly conjectured). Thus, O'Donnell examines constructions of the form $f'(x_1,\ldots,x_k) = C(f(x_1),\ldots,f(x_k))$, where C is an efficiently computable and *monotone* combining function (i.e., changing an input of C from 0 to 1 cannot change the output from 1 to 0). He characterizes the amplification properties of C in terms of its "noise stability," thereby connecting the study of hardness amplification with the analysis of boolean functions (see [305] for more on this topic). He uses this connection to find monotone functions C with nearly optimal amplification properties, namely ones that will ensure that the function f' is roughly $(1/2 - O(1/\sqrt{k}))$-hard if it is obtained by combining k evaluations of f. Contrast this bound with the XOR Lemma, where C is the (non-monotone) parity function and ensures that f' is $(1/2 - 1/2^{\Omega(k)})$-hard. Healy, Vadhan, and Viola [202] showed how to derandomize O'Donnell's construction, so that the inputs $x_1,\ldots,x_k$ can be generated in a correlated way by a much shorter input to f'. This allows for taking k to be exponential in the input length of f', and for certain combining functions C, the function f' is still in **NP** (using the ability of a nondeterministic computation to compute exponential-sized ORs). As a result, assuming that f is mildly hard for nonuniform algorithms running in time $2^{\Omega(n)}$ (the "high end"), they obtain $f' \in \mathbf{NP}$ that is $(1/2 - 1/2^{(n')^{1/2}})$-hard where n' is the input length of f'. A quantitative improvement was given by [273, 179], replacing $2^{(n')^{1/2}}$ with $2^{n'/\mathrm{polylog}(n')}$, but it remains open to achieve the optimal bound of $2^{\Omega(n')}$.

Uniform Reductions: Another line of work has sought to give hardness amplification results for uniform algorithms, similarly to the work on derandomization from uniform assumptions described in Section 8.2.2. Like with cryptographic pseudorandom generators, most of the hardness amplification results in the cryptographic setting, such as Yao's original hardness amplification for one-way functions [421], also apply to uniform algorithms. In the noncryptographic setting, a difficulty is that black-box hardness amplification corresponds to error-correcting codes that can be decoded from very large distances, such as $1/2 - \varepsilon$ in the case of binary codes, and at these distances unique decoding is impossible, so one must turn to list decoding and use some nonuniform advice to select the correct decoding from the list. (See Definition 7.74 and the discussion after it. For amplification from mild average-case hardness rather than worst-case hardness, the coding-theoretic interpretation is that the decoding algorithm only needs to recover a string that is very close to the original message, rather than exactly equal to the message [209, 390]; this also requires list decoding for natural settings of parameters.) However, unlike the case of pseudorandom generator constructions, here the number of candidates in the list can be relatively small (e.g., $\mathrm{poly}(1/\varepsilon)$), so a reasonable goal for a uniform algorithm is to produce a list of possible decodings, as in our definition of locally list-decodable codes (Definition 7.54). As observed in [216, 397, 390], if we are interested in amplifying hardness for functions in natural complexity classes such as **NP** or **E**, we can use (non-black-box) "checkability" properties of the initial function f to select a good decoding from the list, and thereby obtain a fully uniform hardness amplification result.

Uniform local list-decoding algorithms for the Reed–Muller Code were given by [35, 381] (as covered in Section 7), and were used to give uniform worst-case-to-average-case hardness amplification results for **E** and other complexity classes in [397]. Trevisan [390, 392] initiated the study of uniform hardness amplification from mild average-case hardness, giving uniform analogues of some of the hardness amplification results from [208, 304]. Impagliazzo, Jaiswal, Kabanets, and Wigderson [210, 211] gave nearly optimal uniform Direct Product Theorems and XOR Lemmas. The existing uniform amplification results still do not quite match the nonuniform results in two respects. First, the derandomizations are not quite as strong; in particular, it is not known how to achieve an optimal "high end" result, converting a function on n-bit inputs that is mildly average-case hard against algorithms that run in time $2^{\Omega(n)}$ into a function on $O(n)$-bit inputs that is $(1/2 - 1/2^{\Omega(n)})$-hard against time $2^{\Omega(n)}$. (In the nonuniform setting, this was achieved by [215].) Second, for hardness amplification in **NP**, the existing uniform results only amplify a function that is mildly hard against algorithms running in time t to ones that are $(1/2 - 1/(\log t)^{\Omega(1)})$-hard, rather than $(1/2 - 1/t^{\Omega(1)})$-hard, which is achieved in the nonuniform case by [304, 202]. See [88] for a coding-theoretic approach to closing this gap.

Other Cryptographic Primitives: There has also been a large body of work on security amplification for other kinds of cryptographic primitives and interactive protocols (where the goal of the adversary is much more complex than just computing or inverting a function). Describing this body of work is beyond the scope of this survey, so we simply refer the interested reader to [119, 105, 388] and the references therein.

A key component of many hardness amplification results mentioned above is the Hardcore Theorem of Impagliazzo [208] and variants. In its basic form, this theorem states that if a function f is mildly hard on average, then there is a *hardcore set* of inputs, of noticeable density, on which the function is very hard on average. Thus, intuitively, hardness amplification occurs when we evaluate a function many times because we are likely to hit the hardcore set. Since Impagliazzo's original paper, there have been a number of papers giving quantitative improvements to the Hardcore Theorem [243, 206, 45]. In addition to its applications in hardness amplification, the Hardcore Theorem has been shown to be closely related to "boosting" in machine learning [243], "dense model theorems" that have applications in additive number theory and leakage-resilient cryptography [127, 329, 395], computational analogues of entropy [49, 106, 145, 257, 329, 381, 401], and "regularity lemmas" in the spirit of Szemerédi's Regularity Lemma in graph theory [396]. One downside of using the Hardcore Theorem is that the (black-box) reductions in its proofs inherently require a lot of nonuniform advice [274]. As shown by Holenstein [206], this nonuniformity can often be avoided in cryptographic settings, where one can efficiently sample random pairs $(x, f(x))$; his "uniform" analogue of the Hardcore Theorem and variants have played a key role in simplifying and improving the construction of cryptographic pseudorandom generators from one-way functions [198, 206, 401].

8.2.4 Deterministic Extractors

As mentioned in Section 6.5, the study of deterministic extractors has remained active even after the introduction of seeded extractors. One reason is that many applications (such as in cryptography, distributed computing, and Monte Carlo simulation) really require high-quality random bits, we often only have access to physical sources of randomness, and the trick of enumerating the seeds of a seeded extractor does not work in these contexts. Another is that deterministic extractors for certain classes of sources often have other applications of interest (even in contexts where we allow sources of truly random bits). For these reasons, after nearly a decade of work on simulating **BPP** with weak sources and seeded extractors, there was a resurgence of interest in deterministic extractors for various classes of sources [90, 398].

One important class of sources are those that consist of a small number of independent k-sources, first studied by Chor and Goldreich [96] (following [346, 409], who gave extractors for several independent unpredictable-bit sources). In addition to their motivation for obtaining high-quality randomness, extractors for 2 independent sources are of interest because of connections to communication complexity and to Ramsey theory. (Textbooks on these topics are [253] and [182], respectively, and their connections to extractors can be found in [409, 96] and the references below.) In particular, a disperser for 2 independent k-sources of length n is *equivalent* to a *bipartite Ramsey graph* — a bipartite graph with N vertices on each side that contains no $K \times K$ bipartite clique or $K \times K$ bipartite independent set (for $N = 2^n$ and $K = 2^k$). Giving explicit constructions of Ramsey graphs that approach $K = O(\log N)$ bound given by the probabilistic method [132] is a long-standing open problem posed by Erdős [133].

Chor and Goldreich [96] gave extractors for 2 independent k-sources when $k > n/2$ (e.g., inner-product modulo 2), and there was no improvement in this bound for nearly 2 decades. Substantial progress began again with the work of Barak, Impagliazzo, and Wigderson [46], who used new results in arithmetic combinatorics to construct extractors for a constant number of independent k-sources when $k = \delta n$ for an arbitrarily small constant $\delta > 0$. Specifically, they used the Sum–Product Theorem over finite fields [79], which says that for p prime and every subset $A \subset \mathbb{F}_p$ whose size is not too close to p, either the set $A + A$ of pairwise sums or the set $A \cdot A$ of pairwise products is of size significantly larger than $|A|$. Arithmetic (and additive) combinatorics have now been found to be closely connected to many topics in pseudorandomness and theoretical computer science more broadly; see the survey of Trevisan [394] and the talks and lecture notes from the minicourse [50].

A series of subsequent works improved [46] using techniques based on seeded extractors and condensers, various forms of composition, and other new ideas, achieving in particular explicit extractors for 3 sources of min-entropy $k = \delta n$ [47], extractors for a constant number of sources of min-entropy $k = n^{\delta}$ [318] (notably making no use of arithmetic combinatorics), and dispersers for 2 sources of min-entropy $k = n^{o(1)}$ [48].

The latter result is a substantial improvement over the previous best explicit construction of Ramsey graphs [141], which avoided cliques and independent sets of size $K = 2^{\sqrt{n}}$ and only applied to the nonbipartite case. Even with all of this progress, the following remains open:

Open Problem 8.11. For every constant $\delta > 0$, construct an explicit function $\mathrm{Ext} : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ such that for every two independent δn-sources X, Y, $\mathrm{Ext}(X,Y)$ is ε-close to uniform on $\{0,1\}$.

Another line of work considers classes of sources defined by some measure of "complexity," for example capturing the complexity of generating a random sample of the source (from independent coin tosses). Explicit extractors have been constructed for a variety of models of space-bounded sources [69, 231, 246, 247, 398, 408]. In particular, Kamp, Rao, Vadhan, and Zuckerman [231] show how to deterministically extract $\Omega(n)$ bits from any source of min-entropy $k = \Omega(n)$ generated by a (nonuniform) algorithm with $O(n)$ space; this results exploits connections to extracting from multiple independent sources (as discussed above) as well as from bit-fixing sources (see Section 6.5). Trevisan and Vadhan [398] suggest to consider sources generated by small Boolean circuits, and show that under strong complexity assumptions (similar to, but stronger than, the one in Theorem 7.68), there is a deterministic extractor computable in time $\mathrm{poly}(s)$ that extracts randomness from sources of min-entropy $k = (1 - \Omega(1))n$ generated by circuits of size s, with error $\varepsilon = 1/s$. It is necessary that the extractor has higher computational complexity than the source,[3] but the min-entropy and error bounds can potentially be much better:

Open Problem 8.12. Under plausible complexity assumptions, show that there exists a deterministic extractor computable in time $\mathrm{poly}(s)$ that extracts randomness from sources of min-entropy k generated by circuits of size s, with error ε, for $k \leq n/2$ and/or $\varepsilon = s^{-\omega(1)}$. Alternatively, give evidence that no such extractor exists.

[3] Interestingly, for *condensers*, it no longer seems necessary that the extractor has higher complexity than the source [120].

De and Watson [114] and Viola [414, 415] have recently obtained unconditional extractors for restricted classes of sampling circuits, such as $\mathbf{NC^0}$ (where each output bit depends on a constant number of input bits) [114, 414] $\mathbf{AC^0}$ [414], even for sublinear min-entropy. These results are based on a close connection between constructing extractors for a class of circuits and finding explicit distributions that are hard for circuits in the class to sample, the latter a topic that was also studied for the purpose of proving data structure lower bounds [415].

It is also natural to look at sources that are low complexity in an algebraic sense. The simplest such model is that of *affine sources*, which are uniform over affine subspaces of $\mathbb{F}^n$ for a finite field $\mathbb{F}$. If the subspace has dimension k, then the source is a flat source of min-entropy $k \log |\mathbb{F}|$. For the case of $\mathbb{F} = \mathbb{Z}_2$, there are now explicit extractors for affine sources of sublinear min-entropy $k = O(n)$ [78, 422, 262] and dispersers for affine sources of subpolynomial min-entropy $k = n^{o(1)}$ [353]. For large fields $\mathbb{F}$ (i.e., $|\mathbb{F}| = \text{poly}(n)$), there are extractors for subspaces of every dimension $k \geq 1$ [147]. There have also been works on extractors for sources described by polynomials of degree larger than 1, either as the output distribution of a low-degree polynomial map [123] or the zero set of low-degree polynomial (i.e., an algebraic variety) [122].

We remark that several of the state-of-art constructions for independent sources and affine sources, such as [47, 48, 353], are quite complicated, using sophisticated compositions and/or machinery from arithmetic combinatorics. It is of interest to find simpler constructions; some progress has been made for affine sources in [60, 262] and for independent sources in [62] (where the latter in fact gives a reductions from constructing 2-source extractors to constructing affine extractors).

8.2.5 Algebraic Pseudorandomness

Algebraic Measures of Pseudorandomness. In this survey, we have mostly focused on statistical and computational measures of (pseudo)randomness, such as pairwise independence and computational indistinguishability, respectively. It is also of interest to consider more algebraic measures, because they can be often related to statistical and/or computational measures, may be convenient for analyzing algebraic constructions of pseudorandom objects, and can have applications of their own (e.g., in additive or arithmetic combinatorics).

One of the most basic and important algebraic measures of pseudorandomness is that of a *small-bias* space, introduced by Naor and Naor [296]. Here a random variable $X = (X_1,\ldots,X_n)$ taking values in $\{0,1\}^n$ is said to be *ε-biased* iff for every nonempty subset $S \subset [n]$, we have $(1-\varepsilon)/2 \leq \Pr[\oplus_{i\in S} X_i = 1] \leq (1+\varepsilon)/2$. Naor and Naor [296] presented an explicit generator $G: \{0,1\}^d \to \{0,1\}^n$ of seed length $d = O(\log(n/\varepsilon))$ such that $G(U_d)$ is ε-biased; thus G is a pseudorandom generator fooling all "parity tests." (Simpler constructions with better constants are in [19].) This generator has found a variety of applications, such as almost k-wise independent hash functions [296, 19] (cf. Problem 3.4), pseudorandom generators for small-width branching programs [345, 363, 179], derandomization of specific algorithms [296], and almost-linear-length probabilistically checkable proofs [61, 59].

Small-bias generators have several equivalent formulations. When viewed appropriately, they are equivalent to *linear* error-correcting codes (as in Problem 5.4) over $\mathbb{F}_2$ in which every nonzero codeword has Hamming weight between $(1-\varepsilon)/2$ and $(1+\varepsilon)/2$ [296, 19]. They are also equivalent to expanders with spectral expansion $1-\varepsilon$ that have algebraic structure of a *Cayley graph* over the group $G = \mathbb{Z}_2^n$ [24]. (In general, when G is a group and $S \subset G$, the Cayley graph is $|S|$-regular digraph on vertex set G, where the neighbors of vertex g are $\{gs : s \in S\}$.) Finally, the small-bias property is equivalent to X being a distribution on the group $G = \mathbb{Z}_2^n$ all of whose nontrivial Fourier coefficients are at most ε [296]. Specifically, the fact that the $(1-\varepsilon)/2 \leq \Pr[\oplus_{i\in S} X_i = 1] \leq (1+\varepsilon)/2$ is equivalent to requiring that $|\mathrm{E}[\chi_S(X)]| \leq \varepsilon$, where $\chi_S(x) = (-1)^{\oplus_{i\in S} x_i}$ is the Fourier character of $G = \mathbb{Z}_2^n$ indexed by the set S. (For background on Fourier analysis over finite groups, see the book by Terras [387], and for a survey of applications in theoretical computer science, see [115] and the references therein.) Thus, we see that even in an algebraic context, different types of pseudorandom objects (generators, codes, and expanders) are equivalent.

These different views of small-bias spaces suggest different generalizations. One is that of linear codes over larger finite fields, which have been studied extensively in coding theory and of which we've seen some examples (Problem 5.4). Another is to consider Cayley graphs and Fourier analysis over groups G other than $\mathbb{Z}_2^n$. Over abelian groups, this generalization is fairly direct; the Fourier coefficients of a small-bias space on a group G are exactly the eigenvalues of a corresponding Cayley graph over the group G. For many abelian groups, including those of the form $G = \mathbb{Z}_p^n$ for prime p, there are known explicit constructions of small-bias generators with seed length $O(\log\log|G| + \log(1/\varepsilon))$, corresponding to explicit Cayley expanders of spectral expansion $1 - \varepsilon$ and degree $\mathrm{poly}(\log|G|, 1/\varepsilon)$ [240, 9, 135, 40, 321, 21]. However, there are benefits in working with nonabelian groups, as Cayley expanders over abelian groups G require degree $\Omega(\log N)$, where $N = |G|$ is the number of vertices [24]. The logarithmic degree lower bound for expanders over abelian groups also holds for the more general notion of *Schreier graphs*, where the group G acts as permutations on the vertex set V, and we connect a vertex $v \in V$ to $s(v)$ for every $s \in S$ for some subset $S \subset G$. On the other hand, many of the algebraic constructions of constant-degree expanders, including Ramanujan graphs and the others described in Section 4.3.1, are obtained as Cayley graphs or Schreier graphs over nonabelian groups. The spectral expansion of these constructions can be analyzed via group representation theory, which is more involved than Fourier analysis over abelian groups, because it deals with matrix-valued (rather than scalar-valued) functions. See the surveys by Hoory, Linial, and Wigderson [207] and Lubotzky [276], the lecture notes of Tao [386], and the notes for Section 4 for more on this and other approaches to analyzing the expansion of Cayley and Schreier graphs.

In the above, we think of a small-bias generator (according to the original definition) as producing pseudorandom elements of the group $G = \mathbb{Z}_2^n$, and consider generalizations to different groups G. An alternative view is that the small-bias generator produces a sequence of bits, and the group $G = \mathbb{Z}_2^n$ only arises in defining what it means for the distribution on bit-strings to be pseudorandom. More generally, we can take $G = H^n$ for any finite group H, and consider a random variable $X = (X_1,\ldots,X_n)$ taking values in $\{0,1\}^n$ to be pseudorandom if for every $(h_1,\ldots,h_n) \in H^n$, the distribution of $h_1^{X_1}h_2^{X_2}\cdots h_n^{X_n}$ is statistically close to $h_1^{R_1}h_2^{R_2}\cdots h_n^{R_n}$ where $R = (R_1,\ldots,R_n)$ is uniformly distributed in $\{0,1\}^n$. This notion is of interest in part because the function $f_h(x_1,\dots,x_n) = h_1^{x_1}\cdots h_n^{x_n}$ can be computed by a read-once branching program of width $|H|$ (see Section 8.2.1). Thus constructing generators fooling such "group product programs" are a natural warm-up to constructing pseudorandom generators for space-bounded computation, and in fact are equivalent to constructing generators for permutation branching programs in the case of constant width [250].

Yet another type of generalization is obtained by expanding the class of distinguishers from linear functions (which over $\mathbb{F}_2^n$ are simply parities) to higher-degree polynomials. A series of recent results has shown that the sum of d small-bias spaces (over $\mathbb{F}^n$ for any finite field $\mathbb{F}$) cannot be distinguished from uniform by polynomials of degree d [75, 270, 413]. These results were inspired by and influenced work on the Gowers uniformity norms from arithmetic combinatorics [180, 181], which can be viewed as providing a "higher-order Fourier analysis" and have found numerous applications in theoretical computer science. (See [394, 50].)

Explicit Constructions via Polynomial Evaluation. As we've seen in this survey, algebra also provides powerful tools for constructing pseudorandom objects whose definitions make no reference to algebra. One particularly useful paradigm in such constructions is polynomial evaluation. Specifically, we construct a pseudorandom object $\Gamma : \mathbb{F}^n \times \mathbb{E} \to \mathbb{F}^m$, where $\mathbb{F}$ is a finite field and $\mathbb{E} \subset \mathbb{F}^t$ is a set of evaluation points, by setting $\Gamma(f,y) = (f_1(y),\dots,f_m(y))$, where we view $f \in \mathbb{F}^n$ as specifying a low-degree polynomial in t variables, from which we construct m related low-degree polynomials $f_1,\dots,f_m$ that we evaluate at the seed y. Reed–Solomon and Reed–Muller Codes (Constructions 5.14 and 5.16) correspond to the case where $m = 1$ and we take $f_1 = f$ to be a univariate or multivariate polynomial, respectively.[4] In Parvaresh–Vardy Codes (Construction 5.21), we took f to be a univariate polynomial and obtained the f_is by powering f modulo an irreducible polynomial. In Folded Reed–Solomon Codes

[4] In our presentation of Reed–Solomon and Reed–Muller codes, we took $\mathbb{E} = \mathbb{F}^t$, but many of the properties of these codes also hold for appropriately chosen subsets of evaluation points.

(Construction 5.23), $f_i(Y) = f(\gamma^{i-1}Y)$, so evaluating f_i at y amounts to evaluating f at the related point $\gamma^{i-1}y$. More generally, if our construction $\Gamma(f,y)$ is obtained by evaluating f at points $g_1(y),\ldots,g_m(y)$ for linear (or low-degree) functions $g_1,\ldots,g_m$, we can also view it as evaluating the polynomials $f_1 = f \circ g_1,\ldots,f_m = f \circ g_m$ at the seed y.

Prior to Parvaresh–Vardy codes, constructions of this type were used for extractors and pseudorandom generators. Specifically, Miltersen and Vinodchandran [289] show that if we take f to describe a multivariate polynomial (via low-degree extension) and evaluate it on the points of a random axis-parallel line through y, we obtain a hitting-set generator construction against nondeterministic circuits (assuming f has appropriate worst-case hardness for nondeterministic circuits); this construction has played a key role in derandomizations of **AM** under uniform assumptions [195, 358].[5] Ta-Shma, Zuckerman, and Safra [384] showed that a similar construction yields randomness extractors with seed length $(1+O(1))\log n$ for polynomially small min-entropy and polynomial entropy loss. Shaltiel and Umans [356, 399] showed that evaluating the t-variate polynomial f at the points $y, \gamma y, \gamma^2 y, \ldots, \gamma^{m-1}y$, where γ is a primitive element of the field of size $|\mathbb{F}|^t$ (which we associate with $\mathbb{F}^t$) yields both a very good extractor construction and an optimal construction of pseudorandom generators from worst-case hard functions. Notice that this construction is precisely a multivariate analogue of Folded Reed–Solomon Codes (which came afterwards [188]). Recently, Kopparty, Saraf, and Yekhanin [249] have introduced yet another useful way of obtaining the "related polynomials" $f_1,\ldots,f_m$, namely by taking derivatives of f; this has yielded the first codes with rate approaching 1 while being locally (list-)decodable in sublinear time [249, 248] as well as codes matching the optimal rate-distance tradeoff of Folded Reed–Solomon Codes [193, 248]. All of this suggests that polynomial evaluation may be a promising approach to obtaining a unified and near-optimal construction of pseudorandom objects (Open Problem 8.5).

[5] The constructions described here have some additional components in addition to the basic polynomial evaluation framework $\Gamma(f,y) = (f_1(y),\ldots,f_m(y))$, for example the seed should also specify the axis along which the line is parallel in [289] and a position in an "inner encoding" in [384, 356, 399]. We ignore these components in this informal discussion.