Pseudorandomness When the Odds are Against You

Sergei Artemenko[*1], Russell Impagliazzo[†2], Valentine Kabanets[‡3], and Ronen Shaltiel[§4]

1 **Department of Computer Science, University of Haifa, Haifa, Israel**
sartemen@gmail.com
2 **Department of Computer Science, University of California, San Diego, USA**
russell@cs.ucsd.edu
3 **School of Computing Science, Simon Fraser University, Burnaby, Canada**
kabanets@cs.sfu.ca
4 **Department of Computer Science, University of Haifa, Haifa, Israel**
ronen@cs.haifa.ac.il

Abstract

Impagliazzo and Wigderson [25] showed that if E = DTIME($2^{O(n)}$) requires size $2^{\Omega(n)}$ circuits, then every time T constant-error randomized algorithm can be simulated deterministically in time poly(T). However, such polynomial slowdown is a deal breaker when $T = 2^{\alpha \cdot n}$, for a constant $\alpha > 0$, as is the case for some randomized algorithms for NP-complete problems. Paturi and Pudlak [30] observed that many such algorithms are obtained from randomized time T algorithms, for $T \leq 2^{o(n)}$, with large one-sided error $1-\epsilon$, for $\epsilon = 2^{-\alpha \cdot n}$, that are repeated $1/\epsilon$ times to yield a constant-error randomized algorithm running in time $T/\epsilon = 2^{(\alpha + o(1)) \cdot n}$.

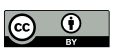
We show that if E requires size $2^{\Omega(n)}$ nondeterministic circuits, then there is a poly(n)-time ϵ-HSG (Hitting-Set Generator) $H\colon \{0,1\}^{O(\log n)+\log(1/\epsilon)} \to \{0,1\}^n$, implying that time T randomized algorithms with one-sided error $1-\epsilon$ can be simulated in deterministic time poly(T)/ϵ. In particular, under this hardness assumption, the fastest known constant-error randomized algorithm for k-SAT (for $k \geq 4$) by Paturi et al. [31] can be made deterministic with essentially the same time bound. This is the first hardness versus randomness tradeoff for algorithms for NP-complete problems. We address the necessity of our assumption by showing that HSGs with very low error imply hardness for nondeterministic circuits with "few" nondeterministic bits.

Applebaum et al. [2] showed that "black-box techniques" cannot achieve poly(n)-time computable ϵ-PRGs (Pseudo-Random Generators) for $\epsilon = n^{-\omega(1)}$, even if we assume hardness against circuits with oracle access to an arbitrary language in the polynomial time hierarchy. We introduce weaker variants of PRGs with *relative error*, that do follow under the latter hardness assumption. Specifically, we say that a function $G : \{0,1\}^r \to \{0,1\}^n$ is an (ϵ, δ)-re-PRG for a circuit C if $(1-\epsilon) \cdot \Pr[C(U_n) = 1] - \delta \leq \Pr[C(G(U_r) = 1] \leq (1+\epsilon) \cdot \Pr[C(U_n) = 1] + \delta$. We construct poly($n$)-time computable (ϵ, δ)-re-PRGs with arbitrary polynomial stretch, $\epsilon = n^{-O(1)}$ and $\delta = 2^{-n^{\Omega(1)}}$. We also construct PRGs with relative error that fool *non-boolean distinguishers* (in the sense introduced by Dubrov and Ishai [11]).

Our techniques use ideas from [30, 43, 2]. Common themes in our proofs are "composing" a PRG/HSG with a combinatorial object such as dispersers and extractors, and the use of nondeterministic reductions in the spirit of Feige and Lund [12].

1998 ACM Subject Classification F.1.2 Modes of Computation

[*] Research supported by ERC starting grant 279559.
[†] Research supported by the Simons Foundation and NSF grants #CNS-1523467 and CCF-121351.
[‡] Research supported by an NSERC Discovery grant.
[§] Research supported by BSF grant 2010120, ISF grant 864/11, and ERC starting grant 279559.



COMPUTATIONAL COMPLEXITY CONFERENCE

31st Conference on Computational Complexity (CCC 2016).
Editor: Ran Raz; Article No. 9; pp. 9:1–9:35

Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Keywords and phrases Derandomization, pseudorandom generator, hitting-set generator, relative error

Digital Object Identifier 10.4230/LIPIcs.CCC.2016.9

1 Introduction

Derandomization, the construction of deterministic algorithms from randomized algorithms, is an area where there are tight connections between lower bounds and algorithm design. Indeed, strong enough circuit lower bounds can be used to construct pseudo-random generators that can then be used to simulate randomized algorithms with only polynomial overhead. This is often summarized as saying, "Randomness is never essential for efficient algorithm design", if such lower bounds exists.

However, there are many algorithmic applications where a simulation with polynomial overhead is next to useless. For example, consider the best algorithms for different NP-complete problems such as different variations of SAT. Many of the best algorithms for these problems are in fact randomized or careful derandomizations of probabilistic algorithms [32, 31, 34, 22, 33, 1]. If the Exponential Time Hypothesis is true, these problems all require exponential time, so a polynomial slowdown might take an algorithm from best possible to worse than exhaustive search. On the other hand, as observed in [30], most of these randomized algorithms are in fact fast algorithms, but with only a very small success probability ϵ. [30] call such algorithms OPP algorithms, for *One-sided error Probabilistic Polynomial Time*. (These algorithms can then be repeated $O(1/\epsilon)$ times to yield a final randomized algorithm with constant success probability).

It is not to hard to see that OPP algorithms can be derandomized in time comparable to the running time of the final randomized algorithm, if we can construct efficient pseudorandom generators (or hitting set generators) which work for a very low error parameter ϵ using short seeds.

In this paper, we address the question of constructing such generators. We give constructions of pseudorandom generators and hitting-set generators that get essentially optimal simulations of OPP, and go beyond to also consider algorithms that have two-sided error that only slightly favors the correct answer. In order to get these generators, we need stronger lower bounds, lower bounds against nondeterministic circuits rather than deterministic circuits. However, we also show that such lower bounds are necessary for strong derandomization of OPP algorithms.

As we explain later, in some settings there are black-box impossibility results on constructing generators for very low error parameter. In this paper, we also introduce new notions of pseudorandom generator with "relative error" which can be used to replace low-error generators in certain settings. We give constructions of such generators, and discuss potential applications.

1.1 Pseudorandom generators and hitting-set generators

We start by reviewing the definitions of pseudorandom generators and hitting set generators.

▶ **Definition 1.1** (PRGs and HSGs)**.** Let $\mathcal{C}$ be a class of boolean functions $C : \{0,1\}^n \to \{0,1\}$. A function $G : \{0,1\}^r \to \{0,1\}^n$ is:

- an ϵ-PRG for $\mathcal{C}$ if for every C in $\mathcal{C}$, $|\Pr[C(G(U_r)) = 1] - \Pr[C(U_n) = 1]| \leq \epsilon$.
- an ϵ-HSG for $\mathcal{C}$ if for every C in $\mathcal{C}$ s.t. $\Pr[C(U_n) = 1] > \epsilon$, there exists $x \in \{0,1\}^r$ s.t. $C(G(x)) = 1$.

We will be interested in generators that fool circuits of size n^b for some fixed constant b, and run in time poly(n^b). In the case of logarithmic seed length ($r = O(\log n)$) this is often referred to as the Nisan-Wigderson setting. We will typically be interested in larger seed length (which is required to handle low error ϵ).

Such PRGs imply circuit lower bounds, and so in the current state of knowledge, we cannot construct them unconditionally. A long line of research [9, 45, 29, 5, 21, 25, 41, 24, 35, 44] is devoted to constructing such PRGs under the weakest possible hardness assumptions. An important milestone of this line of research is the hardness versus randomness tradeoff of Impagliazzo and Wigderson [25].

▶ **Definition 1.2** (E is hard for exponential size circuits)**.** We say that E is hard for exponential size circuits, if there exists a language L in $\mathrm{E} = \mathrm{DTIME}(2^{O(n)})$ and a constant $\beta > 0$, such that for every sufficiently large n, circuits of size $2^{\beta n}$ fail to compute the characteristic function of L on inputs of length n.

▶ **Theorem 1.3** ([25])**.** *If E is hard for exponential size circuits, then for every constant $b > 1$ there exists a constant $c > 1$ such that for every sufficiently large n, there is a function $G : \{0,1\}^r \to \{0,1\}^n$ that is an n^{-b}-PRG for size n^b circuits, with $r = c \log n$. Furthermore, G is computable in time poly(n^b).*

By a standard probabilistic argument, for every $\epsilon > 0$, there exists a (nonexplicit) ϵ-PRG with seed length $r = c \log n + O(\log(1/\epsilon))$ for size n^b circuits. In particular, if we shoot for PRGs with "polynomial stretch" (that is, $r = n^{\Omega(1)}$), we can expect to get error $\epsilon = 2^{-n^{\Omega(1)}}$ that is exponentially small. The known proofs of Theorem 1.3 do not achieve these parameters. In fact, they cannot achieve negligible error of $\epsilon = n^{-\omega(1)}$ if the constructed PRG runs in time poly(n), even if we allow large seed length $r = \Omega(n)$.[1] It is natural to ask if we can construct poly(n)-time computable ϵ-PRGs or ϵ-HSGs with $\epsilon = n^{-\omega(1)}$? Under what assumptions? Can we get ϵ to be exponentially small?

1.2 Limitations on deterministic reductions for PRGs and HSGs

There is a formal sense in which "black-box" proofs of Theorem 1.3 cannot achieve negligible ϵ [38, 18, 3]. It is instructive to explain this argument. Loosely speaking, "black-box" proofs are made of two components: The first is a construction, this is an oracle procedure $\mathrm{Con}^{(\cdot)}$ which implements the PRG $G(x) = \mathrm{Con}^f(x)$ given oracle access to the hard function f that is guaranteed in the hardness assumption. Note that as G runs in time poly(n), the construction cannot afford to query $f \in \mathrm{E}$ on inputs of length larger than $\ell = c \log n$ (for some constant $c > 1$). On inputs of this length, the maximal possible circuit complexity of f is at most $2^\ell = n^c$.

[1] In this paper we are mostly interested in PRGs that run in time poly(n), that is polynomial in the output length. Another natural notion is allowing PRGs to run in time exponential in the seed length (that is time $2^{O(r)}$). These notions differ in the case of "polynomial stretch" ($r = n^{\Omega(1)}$) which will be the setting that we will consider. PRGs which run in time exponential in the seed length make sense in most applications which run the PRGs over all 2^r seeds. An exception is the case of "SAT algorithms" where r may be $\alpha \cdot n$ for a constant $\alpha < 1$ that is close to 1, and there may be a substantial difference between running 2^r instantiations of a time poly(n) PRG, (which gives time less than 2^n) compared to 2^r instantiations of a PRG running in time larger than 2^r, which takes time at least $2^r \cdot 2^r > 2^n$. Other applications in which there is a big difference between $2^{O(r)}$ time PRGs and poly(n) time PRGs are applications that run the PRG only once. In such cases, it is typically important that the PRG run in time polynomial in the output length (so that the application runs in polynomial time). We will elaborate on several such applications in this paper.

The second component in the proof is a reduction $\text{Red}^{(\cdot)}$, which is given black-box access to a circuit D that is not fooled by the PRG, and implements a small circuit $C : \{0,1\}^\ell \to \{0,1\}$ for f (contradicting the hardness assumption). By the discussion above, in order to contradict the hardness assumption, the reduction must produce a circuit C of size less than n^c. However, note that in some sense, the reduction needs to distinguish between a useless function D that always answers zero, and a useful function D that is not fooled by the PRG, and answers one with probability ϵ. It intuitively follows that Red (which only has black-box access to D) must query D at least $1/\epsilon$ times. (This part of the argument can be made formal, see [38, 18, 3], and also applies for constructions of HSGs). In particular, the circuit C that is implemented by Red has size $\geq 1/\epsilon$. This gives $1/\epsilon \leq n^c$ which implies $\epsilon \geq n^{-c}$.

The same kind of limitations apply in the closely related problem of hardness amplification (there the goal is to start with a worst-case hard lower bound (such as the assumption E is hard for exponential size circuits) and produce an average-case hard function. An influential work of Feige and Lund [12] shows that nondeterministic reductions can be used to bypass these limitations. Specifically, we may relax the requirement that Red implements a (deterministic) circuit, and allow Red to implement a nondeterministic circuit. Indeed, nondeterminism allows Red to make exponentially many queries to D (on different "nondeterministic computations paths") circumventing the limitation above. The price we pay is that we need to assume a hardness assumption against nondeterministic circuits. This approach indeed leads to hardness amplification with negligible ϵ under hardness assumptions for nondeterministic circuits [43, 10, 2].

1.3 Hardness assumptions for nondeterministic circuits

We start by defining various notions of nondeterministic circuit.

▶ **Definition 1.4** (nondeterministic circuits with few nondeterministic bits)**.** We say that a function $f : \{0,1\}^n \to \{0,1\}$ is computed by a size s circuit D with k nondeterministic bits if there exists a size s deterministic circuit $C : \{0,1\}^n \times \{0,1\}^k \to \{0,1\}$ such that for every $x \in \{0,1\}^n$

$$f(x) = 1 \;\Leftrightarrow \exists y \in \{0,1\}^k \text{ s.t. } D(x,y) = 1.$$

▶ **Definition 1.5** (oracle circuits and Σ_i-circuits)**.** Given a boolean function $A(x)$, an A-circuit is a circuit that is allowed to use A gates (in addition to the standard gates). An NP-circuit is a SAT-circuit (where SAT is the satisfiability function) a Σ_i-circuit is an A-circuit where A is the canonical Σ_i^P-complete language. The size of all circuits is the total number of wires and gates.[2]

Note for example that an NP-circuit is different than a nondeterministic circuit. The former is a nonuniform analogue of P^{NP} (which contains coNP) while the latter is an analogue of NP. Hardness assumptions against nondeterministic/NP/Σ_i circuits appear in the literature in various contexts of derandomization [27, 28, 43, 15, 35, 19, 36, 6, 37, 10, 4, 2]. Typically, the assumption is of the following form: E is hard for exponential size circuits (where the type of circuits is one of the types discussed above). More specifically:

[2] An alternative approach is to define using the Karp-Lipton notation for Turing machines with advice. For $s \geq n$, a size $s^{\Theta(1)}$ deterministic circuit is equivalent to $\text{DTIME}(s^{\Theta(1)})/s^{\Theta(1)}$, a size $s^{\Theta(1)}$ nondeterministic circuit is equivalent to $\text{NTIME}(s^{\Theta(1)})/s^{\Theta(1)}$, a size $s^{\Theta(1)}$ NP-circuit is equivalent to $\text{DTIME}^{\text{NP}}(s^{\Theta(1)})/s^{\Theta(1)}$, a size $s^{\Theta(1)}$ nondeterministic NP-circuit is equivalent to $\text{NTIME}^{\text{NP}}(s^{\Theta(1)})/s^{\Theta(1)}$, and a size $s^{\Theta(1)}$ Σ_i-circuit is equivalent to $\text{DTIME}^{\Sigma_i^P}(s^{\Theta(1)})/s^{\Theta(1)}$.

▶ **Definition 1.6.** We say that E is hard for exponential size circuits of type X if there exists a problem L in $\mathrm{E} = \mathrm{DTIME}(2^{O(n)})$ and a constant $\beta > 0$, such that for every sufficiently large n, circuits of type X with size $2^{\beta n}$ fail to compute the characteristic function of L on inputs of length n.

Such assumptions can be seen as the nonuniform and scaled-up versions of assumptions of the form EXP $\neq$ NP or EXP $\neq \Sigma_2^{\mathrm{P}}$ (which are widely believed in complexity theory). As such, these assumptions are very strong, and yet plausible - the failure of one of these assumptions will force us to change our current view of the interplay between time, nonuniformity and nondeterminism.[3]

It is known that Theorem 1.3 extends to every type of circuits considered in Definitions 1.4, Definition 1.5 and their combinations.

▶ **Theorem 1.7** ([25, 27, 35, 36])**.** *For every $i \geq 0$, the statement of Theorem 1.3 also holds if we replace every occurrence of the word "circuits" by "Σ_i-circuits" or alternatively by "nondeterministic Σ_i-circuits".*

1.4 A construction of HSGs with low error

Our first result is a construction of a $\mathrm{poly}(n)$-time computable ϵ-HSG that works for small ϵ. We rely on the assumption that E is hard for exponential size nondeterministic circuits. (Note that by the earlier discussion we cannot expect to get this with hardness against deterministic circuits).

▶ **Theorem 1.8** (HSG with seed length $r = \log(1/\epsilon) + O(\log n)$)**.** *If E is hard for exponential size nondeterministic circuits then for every constant $b > 1$ there exists a constant $c > 1$ such that for every sufficiently large n, there is a function $G : \{0,1\}^r \to \{0,1\}^n$ that is an ϵ-HSG for size n^b circuits, with $r = \log(1/\epsilon) + c \log n$. Furthermore, G is computable in time poly(n^b).*

We stress that the seed length achieved in Theorem 1.8 matches that of nonexplicit HSGs that exist by a probabilistic argument: the dependence of r on ϵ is an additive factor of $1 \cdot \log(1/\epsilon)$. In some settings, achieving this correct dependence (with the right constant) for a polynomial time computable HSG is crucial (as seen in the example of the next section).[4]

1.5 Derandomizing randomized algorithms with large one sided error

By going over all seeds of the HSG, we can deterministically simulate randomized polynomial time algorithms with large one-sided error of $1 - \epsilon(n)$ in time $2^r \cdot \mathrm{poly}(n) = \mathrm{poly}(n)/\epsilon(n)$. This is stated precisely in the theorem below.

▶ **Theorem 1.9.** *Let A be a time $T(n) \geq n$ randomized algorithm that accepts some language L with one sided error of $1 - \epsilon(n)$. That is, for every sufficiently large n and $x \in \{0,1\}^n$:*

- $x \in L \Rightarrow \Pr[A(x) = 1] \geq \epsilon(n)$.

[3] Another advantage of constructions based on this type of assumptions is that any E-complete problem (and such problems are known) can be used to implement the constructions, and the correctness of the constructions (with that specific choice) follows from the assumption. We do not have to consider and evaluate various different candidate functions for the hardness assumption.

[4] We remark that when applying the probabilistic argument for PRGs we get an additive factor of $2 \cdot \log(1/\epsilon)$ whereas for HSGs it is possible to get $1 \cdot \log(1/\epsilon)$. This difference is crucial for the application of derandomizing OPP algorithms as we explain below.

- $x \notin L \Rightarrow \Pr[A(x) = 1] = 0$.

If E is hard for exponential size nondeterministic circuits then there is a deterministic algorithm running in time poly$(T(n))/\epsilon(n)$ that accepts L.

Note that if $T(n) \ll 1/\epsilon(n)$ then the slowdown is polynomial in $T(n)$ but *linear* in $1/\epsilon(n)$. As we explain below, in many algorithms in the literature, $T(n) = \text{poly}(n)$ and $\epsilon(n) = 2^{-\alpha \cdot n}$ for some constant $0 < \alpha < 1$. Note that even the more modest goal of amplifying the success probability of A to obtain a randomized algorithm with constant one-sided error, requires running time of $\text{poly}(T(n))/\epsilon(n)$ which is $\text{poly}(n) \cdot 2^{\alpha \cdot n}$ for the choices above. We achieve the same time with a deterministic algorithm.[5]

Moreover, if we were to amplify A, and then derandomize it using known hardness versus randomness tradeoffs, we would end up with a deterministic algorithm running in time at least $\text{poly}(n)/\epsilon(n)^c$ for a large constant c. Such a slowdown is a "deal breaker" if ϵ is very small (say $\epsilon = 2^{-\alpha \cdot n}$) for a constant α that is only slightly smaller than one. In the next section we observe that this is the case in many randomized k-SAT algorithms.

1.6 Deterministic k-SAT algorithms

Paturi and Pudlak [30] observed that many of the randomized algorithms in the literature for solving k-SAT and other NP-complete problems (in particular the algorithm of Paturi, Pudlak and Zane [32], Paturi et al. [31], Schöning [34]) are based on designing probabilistic polynomial-time (or subexponential-time) algorithms with one-sided error, whose success probability may be exponentially small. To improve the success probability to a constant, one repeats the original randomized algorithm the inverse of the success probability times. The running time of this new randomized algorithm is dominated by the inverse of the success probability of the original algorithm.

For example, suppose A is a SAT-algorithm running in time $T(n) = 2^{o(n)}$ that, given a satisfiable formula, produces a satisfying assignment with probability at least $\epsilon = 2^{-\alpha \cdot n}$ (for some constant $0 < \alpha < 1$). The algorithm with constant success probability is produced by repeating A $O(1/\epsilon)$ times, and so has the running time $2^{(\alpha + o(1)) \cdot n}$.

By Theorem 1.9, all such algorithms can be made deterministic (with essentially the same time bounds) under the assumption that E is hard for nondeterministic circuits. This is the first application of the hardness versus randomness paradigm that yields a nontrivial derandomization of these algorithms.

Some of these randomized algorithms (and in particular the PPZ algorithm [32] and Schöning's algorithm [34]) have deterministic versions. However the fastest known algorithms for k-SAT for $k \geq 4$ due to Paturi et al. [31] does not have a matching deterministic algorithm. We get the first derandomization result for these k-SAT algorithms from [31], based on circuit complexity assumptions.

For each $k \geq 4$, let us denote by $T_k^{\text{PPSZ}}(n) \leq 2^{o(n)}$ the running time of the randomized PPSZ algorithm [31], and let $2^{-\alpha_k^{\text{PPSZ}} \cdot n}$ (where $0 < \alpha_k^{\text{PPSZ}} < 1$ is a constant specified in [31]) be its success probability. The fastest known constant-error randomized algorithm for k-SAT, for $k \geq 4$, is obtained by repeating the above algorithm the inverse success probability number of times, resulting in the running time

$$2^{\alpha_k^{\text{PPSZ}} \cdot n} \cdot T_k^{\text{PPSZ}}(n) \leq 2^{\left(\alpha_k^{\text{PPSZ}} + o(1)\right) \cdot n}.$$

[5] The assumption in Theorem 1.9 can be relaxed to "E is hard for size $n^{\omega(1)}$ nondeterministic circuits" and then, the final running time will be $2^{T(n)^{o(1)}}/\epsilon(n)$.

Our approach gives the following result:

▶ **Theorem 1.10.** *If E is hard for nondeterministic circuits, then there are deterministic algorithms for k-SAT, for each $k \geq 4$, running in time*

$$T_k(n) = 2^{\alpha_k^{PPSZ} \cdot n} \cdot poly(T_k^{PPSZ}(n)) \leq 2^{\left(\alpha_k^{PPSZ} + o(1)\right) \cdot n}.$$

We remark that the assumption could be relaxed to E is hard for size $n^{\omega(1)}$ nondeterministic circuits, and then the deterministic time $T_k(n)$ for k-SAT would become

$$2^{\alpha_k^{\mathrm{PPSZ}} \cdot n} \cdot 2^{\left(T_k^{\mathrm{PPSZ}}(n)\right)^{o(1)}},$$

which is still at most $2^{\left(\alpha_k^{\mathrm{PPSZ}} + o(1)\right) \cdot n}$, as $T_k^{\mathrm{PPSZ}}(n) \leq n^{\beta(n)}$ for every $\beta(n) \in \omega(1)$.

1.7 Hardness assumptions implied by HSGs with low error

In Theorem 1.8 we show that hardness for nondeterministic circuits implies HSGs with low error. Is this assumption necessary? Is the converse statement true? We do not know the answer to these questions. However, we can show ϵ-HSGs for deterministic poly-size circuits are essentially equivalent to $\frac{1}{2}$-HSGs for a subclass of nondeterministic circuits: The class of poly-size nondeterministic circuits with approximately $\log(1/\epsilon)$ nondeterministic bits. The precise definitions and statements appear in Section 6. Note that as $n > r \geq \log(1/\epsilon)$, the circuits we are interested in are nondeterministic circuits with a sublinear number of nondeterministic bits. Using this connection, we can show that ϵ-HSGs with seed length $r = n^{o(1)} + O(\log(1/\epsilon))$ imply that E is hard for poly-size nondeterministic circuits with $o(n)$ nondeterministic bits.

▶ **Theorem 1.11.** *Let $\delta > 0$ be a constant. Assume that for every sufficiently large n, there is a 2^{-n^δ}-HSG $H : \{0,1\}^{O(n^\delta)} \to \{0,1\}^n$ for size $s \geq n$ circuits, and furthermore that the family of functions $H = \{H_n\}$ is computable in time exponential in the seed length, that is time $2^{O(n^\delta)}$. Then there exists a constant $\gamma > 0$ and a problem $L \in E$ such that, for every sufficiently large n', nondeterministic circuits of size $(\gamma n')^{1/\delta}$ with $\gamma \cdot n'$ nondeterministic bits fail to compute the characteristic function of L on inputs of length n'.*

1.8 Limitations on nondeterministic reductions for PRGs

Theorem 1.8 demonstrates that hardness assumption for nondeterministic circuits can yield polynomial time computable HSGs with low error. A recent result of Applebaum et al. [2] shows that these techniques cannot be extended to yield PRGs. We state this result informally below (the reader is referred to [2] for the formal model and precise statement).

▶ **Informal Theorem 1.12.** *For every $i \geq 0$, it is impossible to use "black-box reductions" to prove that the assumption that E is hard for exponential size Σ_i-circuits implies that for $\epsilon = n^{-\omega(1)}$, there is a poly$(n)$-time computable ϵ-PRG $G : \{0,1\}^{n-1} \to \{0,1\}^n$ for size n^2.*

While hardness against circuits with oracle to PH problems does not suffice, hardness for circuits with oracle to PSPACE problems does suffice. This follows by inspecting the correctness proofs of Theorem 1.3 (the one that seems easiest to handle is by Sudan, Trevisan and Vadhan [41]).

▶ **Theorem 1.13** (PRG with seed length $r = O(\log n) + \log(1/\epsilon)$)**.** *If E is hard for exponential size PSPACE-circuits then for every constant $b > 1$ there exists a constant $c > 1$ such that for every sufficiently large n, there is a function $G : \{0,1\}^r \to \{0,1\}^n$ that is an ϵ-PRG for size n^b circuits, with $c \cdot (\log n + \log(1/\epsilon))$. Furthermore, G is computable in time poly(n^b).*

In fact, something more precise can be said. The circuit model that comes up is the nonuniform class which corresponds to the fourth level of the counting hierarchy (which is contained in PSPACE).

1.9 Derandomizing randomized algorithms with large two sided error

By Theorem 1.12 we do not expect to construct ϵ-PRGs for small ϵ under the assumption that E is hard for Σ_1-circuits. Nevertheless, it turns out that we can use this assumption to extend Theorem 1.9 to the case of two-sided error.

▶ **Theorem 1.14.** *Let A be a time $T(n) \geq n$ randomized algorithm such that for every sufficiently large n and $x \in \{0,1\}^n$:*

- $x \in L \Rightarrow \Pr[A(x) = 1] \geq 2 \cdot \epsilon(n)$.
- $x \notin L \Rightarrow \Pr[A(x) = 1] \leq \epsilon(n)$.

If E is hard for exponential size Σ_1-circuits then there is a deterministic algorithm running in time $\frac{poly(T(n))}{\epsilon(n)^2}$ that accepts L.[6]

Note that even the more modest goal of amplifying the success probability of A to obtain a randomized algorithm with constant two-sided error, requires running time of $T(n)/\epsilon(n)^2$. We achieve roughly the same time with a deterministic algorithm. In fact, the conclusion of Theorem 1.14 is stronger than the one that follows if we were to run the PRG of Theorem 1.13 on all seeds. The latter approach would have given time $\frac{\text{poly}(n)}{\epsilon(n)^c}$ for a large constant c.

Loosely speaking, we avoid the limitations on PRGs by showing a derandomization procedure which runs the algorithm on 2^r "pseudorandom strings" (just like in PRGs). The key difference is that the "estimation of the success probability of $A(x)$" is not done by "averaging over all pseudorandom strings". This allows the procedure not to be fooled by a small fraction of "pseudorandom strings" that yield incorrect results.

1.10 Implications to derandomization of BPP_{path}

The class BPP_{path} defined by Han, Hemaspaandra and Thierauf [20] consists of polynomial time randomized algorithms $A(x)$ which are allowed to output "don't know". It is required that for every input x, conditioned on giving an answer, the probability that $A(x)$ answers correctly is at least $2/3$, and that the probability that $A(x)$ answers is larger than $\epsilon(n)$ for some $\epsilon(n) > 0$.

Han, Hemaspaandra and Thierauf [20] showed that this class is quite powerful and contains $\text{P}_{||}^{\text{NP}}$ which contains NP. (The subscript "$||$" in $\text{P}_{||}^{\text{NP}}$ means that the queries to the NP oracle are nonadaptive). Shaltiel and Umans [36] showed that BPP_{path} is equal to $\text{P}_{||}^{\text{NP}}$ if $\text{E}_{||}^{\text{NP}}$ is hard for exponential size nondeterministic circuits.

Theorem 1.14 allows us to give a deterministic simulation of BPP_{path} algorithms with running time depending on the parameter $\epsilon(n)$. More specifically, it follows that under the hardness assumption, every BPP_{path} algorithm that gives an answer with probability $\epsilon(n)$, can be simulated in deterministic time $\text{poly}(n)/\epsilon(n)^2$.

---

[6] The assumption in Theorem 1.14 can be improved to E is hard for exponential size nondeterministic circuit. This is because Shaltiel and Umans [36] showed that this assumption implies that E is hard for exponential size Σ_1-circuits which make nonadaptive queries to their oracle. This latter assumption is sufficient for our proof. We defer the details to the final version.

1.11 PRGs with relative error

Theorem 1.14 demonstrates that it is sometimes possible to achieve consequences of PRGs with low error, under hardness assumptions that do not seem to suffice for such PRGs. We now give another such example.

A useful property of a δ-PRG $G : \{0,1\}^r \to \{0,1\}^n$ with very small error $\delta = n^{-\omega(1)}$ is that it "preserves the probability of small events". By that we mean that for every circuit $C : \{0,1\}^n \to \{0,1\}$ such that $\Pr[C(U_n) = 1] \leq \delta$,

$$\Pr[C(G(U_r)) = 1] \leq \Pr[C(U_n) = 1] + \delta \leq 2\delta$$

which is still negligible. The notion of PRGs with "relative error" defined below captures this property. In the definition below, the reader should think of $\delta \ll \epsilon$, and recall $e^\epsilon \approx 1 + \epsilon$ for sufficiently small ϵ.

▶ **Definition 1.15** (re-PRGs)**.** Let p_1, p_2 be two numbers, we define a relation on p_1, p_2 by:

$$p_1 \overset{re}{\sim}_{(\epsilon,\delta)} p_2 \;\Leftrightarrow\; \max(p_1, p_2) \leq e^\epsilon \cdot \min(p_1, p_2) + \delta.$$

A function $G : \{0,1\}^r \to \{0,1\}^n$ is an (ϵ, δ)-re-PRG for a class $\mathcal{C}$ of functions $C : \{0,1\}^n \to \{0,1\}$ if for every C in the class $\mathcal{C}$,

$$\Pr[C(G(U_r)) = 1] \overset{re}{\sim}_{(\epsilon,\delta)} \Pr[C(U_n) = 1].$$

The use of the formalism above is inspired by the notion of (ϵ, δ)-differential privacy. An $(1, \delta)$-re-PRG indeed "preserves the probability of small events" and gives that $\Pr[C(U_n) = 1] \leq \delta$ implies $\Pr[C(G(U_r)) = 1] \leq e \cdot \delta$. It also immediately follows that:

▶ **Fact 1.16.** *If G is an (ϵ, δ)-re-PRG for $\mathcal{C}$ and $\delta \leq \epsilon$ then G is a 4ϵ-PRG for $\mathcal{C}$.*

Thus, an (ϵ, δ)-re-PRG with $\delta \ll \epsilon$ can be thought of as an ϵ-PRG which has the additional property that it preserves the probability of small events.

Our next result is a construction of poly(n)-time computable (ϵ, δ)-re-PRGs with arbitrary polynomial stretch, $\epsilon = n^{-O(1)}$ and exponentially small $\delta = 2^{-\sqrt{r}} = 2^{-n^{\Omega(1)}}$.

▶ **Theorem 1.17.** *If E is hard for exponential size Σ_3-circuits, then for every constants $b, e > 1$ and $\mu > 0$ there exists a constant $\gamma > 0$ such that and every sufficiently large n, there is a function $G : \{0,1\}^{r=n^\mu} \to \{0,1\}^n$ that is an $(n^{-b}, 2^{-\gamma\cdot\sqrt{r}})$-re-PRG for size n^b circuits. Furthermore, G is computable in time poly(n^b).*

We remark that we would have liked to achieve $\delta = 2^{-\Omega(r)}$ (rather than $\delta = 2^{-\Omega(\sqrt{r})}$) but we don't know how to achieve this.

1.12 Randomness reduction in Monte-Carlo constructions

In many famous explicit construction problems (such as constructing rigid matrices or generator matrices for linear codes matching the Gilbert-Varshamov bound) a random n bit string has the required property with overwhelming probability of $1 - \delta$ for exponentially small δ. It is often the case that we do not have poly(n)-time deterministic algorithm that produce an n bit string with the required property. An intermediate goal is to reduce the number of random bits used (while preserving exponentially small failure probability). Using re-PRGs, achieves this task for problems where checking whether a given an n bit string x satisfies the property can be decided in the polynomial time hierarchy (and note that the two aforementioned construction problems satisfy this requirement). This is stated formally below:

▶ **Theorem 1.18.** *Let $i \geq 0$ be a constant and let L be a language such that:*

- *There is a constant $\alpha > 0$ s.t. for every sufficiently large n, $\Pr_{X \leftarrow U_n}[X \in L] \geq 1 - \delta$ for $\delta = 2^{-n^\alpha}$.*
- *L is accepted by a family of poly-size Σ_i-circuits.*

If E is hard for exponential size Σ_{i+3}-circuits then there is a poly(n)-time algorithm B such that $\Pr[B(U_r) \in L] \geq 1 - 4 \cdot \delta$ with $r = O(\log(1/\delta)^2) = n^{2\alpha}$.

Note that standard PRGs (which under this assumption achieve error $\geq 1/\text{poly}(n)$) give a version of Theorem 1.18 with $\delta = 1/\text{poly}(n)$. Our result gives this tradeoff also for smaller values of δ. We remark that we would have liked the dependence of r on δ in Theorem 1.18 to be $r = O(\log(1/\delta))$, and this would follow if we can improve the parameter $\delta = 2^{-\Omega(\sqrt{r})}$ in Theorem 1.17 to $\delta = 2^{-\Omega(r)}$.

The aforementioned examples of matrix rigidity and linear codes matching the Gilbert-Varshamov bound do not seem related to computational hardness assumptions. Assuming circuit lower bounds in order to handle them, may seem like an overkill. We remark that one can also apply Theorem 1.18 to solve explicit construction problem that are computational in nature. For example, the language L consisting of truth tables of functions $f : \{0,1\}^{\log n} \to \{0,1\}$ with almost maximal circuit complexity also satisfies the requirements in Theorem 1.18, and so, if E is hard for exponential size Σ_4-circuits, then there is a randomized polynomial time algorithm that uses $r = n^{2\alpha}$ random bits and generates an n-bit truth table of a function with almost maximal circuit complexity with probability at least $1 - 2^{-n^\alpha}$.

This approach can be useful to construct other "computational" pseudorandom objects, and is used to explicitly construct nonboolean PRGs (formally defined in the next section) with relative error, under hardness assumptions. This result is described in the next section.

1.13 PRGs with relative error for nonboolean distinguishers

Dubrov and Ishai [11] considered a generalization of PRGs which fools circuits that output many bits (and not just boolean circuits).

▶ **Definition 1.19** (nb-PRG)**.** Let ℓ be a parameter, and let $\mathcal{C}$ be a class of functions $C : \{0,1\}^n \to \{0,1\}^\ell$. A function $G : \{0,1\}^r \to \{0,1\}^n$ is an (ℓ, ϵ)-nb-PRG for $\mathcal{C}$ if for every C in $\mathcal{C}$, the probability distributions $C(G(U_r))$ and $C(U_n)$ are ϵ-close, meaning that for every function $D : \{0,1\}^\ell \to \{0,1\}$, $|\Pr[D(C(G(U_r))) = 1] - \Pr[D(C(U_n)) = 1]| \leq \epsilon$.

For every $\ell \geq 1$, an (ℓ, ϵ)-nb-PRG is in particular a $(1, \epsilon)$-nb-PRG which is easily seen to be equivalent to an ϵ-PRG. Thus, (ℓ, ϵ)-nb-PRGs are a generalization of ϵ-PRGs, and so the limitations of Theorem 1.12 apply to them.

The motivation for nb-PRGs is reducing the randomness complexity of sampling procedures. We now explain this application. Let P be a distribution over ℓ-bit strings, and let A be a sampling algorithm for it. That is, A is a poly(n)-time algorithm such that $A(U_n) = P$. An (ℓ, ϵ)-nb-PRG $G : \{0,1\}^r \to \{0,1\}^n$ for size n^b-circuits can be used to sample a distribution P' that is ϵ-close to P, using only $r < n$ random bits. This is because the sampling algorithm $B(U_r) = A(G(U_r))$ produces a distribution that is ϵ-close to $A(U_n)$.[7] Note that if we want B to run in time poly(n), we must require that G runs in time poly(n). Thus, this is another setting where we would like to have PRGs computable in time poly(n).

[7] It is important to note that if G is a standard PRG, we can only guarantee that $B(U_r)$ is computationally indistinguishable from $A(U_n)$, rather than statistically indistinguishable.

In this paper we consider a generalization of nb-PRGs with (ϵ, δ)-relative error.

▶ **Definition 1.20** (re-nb-PRG)**.** Let $\mathcal{C}$ be a class of functions $C : \{0,1\}^n \to \{0,1\}^\ell$. A function $G : \{0,1\}^r \to \{0,1\}^n$ is an (ℓ, ϵ, δ)-re-nb-PRG for $\mathcal{C}$ if for every C in $\mathcal{C}$, the probability distributions $C(G(U_r))$ and $C(U_n)$ are (ϵ, δ)-close in relative distance, meaning that for every function $D : \{0,1\}^\ell \to \{0,1\}$, $\Pr[D(C(G(U_r))) = 1] \overset{re}{\sim}_{(\epsilon,\delta)} \Pr[D(C(U_n)) = 1]$.

If we use re-nb-PRGs (rather than nb-PRGs) in the construction of the sampling algorithm B, then we "preserve probability of small events". That is for every $D : \{0,1\}^\ell \to \{0,1\}$, if $\Pr[D(P) = 1] \leq \delta$ then $\Pr[D(P') = 1] \leq O(\delta)$. This property is helpful in some applications.

Previous work by Applebaum et al. [2] (improving upon [11, 4]) gives $(\ell, n^{-O(1)})$-nb-PRGs with seed length $r = O(\ell + \log n)$ and $\epsilon = n^{-O(1)}$. This is under the assumption that E is hard for exponential size nondeterministic circuits. Note that $r \geq \ell$ is a trivial lower bound on the seed length. In this paper we construct (ϵ, δ)-re-nb-PRGs with $\epsilon = n^{-O(1)}$ and $r = 1 \cdot \ell + O(\log(1/\delta))^2$ for $\delta \geq 2^{-n^{\Omega(1)}}$. This is done under the stronger assumption that E is hard for exponential size Σ_6-circuits.

▶ **Theorem 1.21** (re-nb-PRG with seed length $1 \cdot \ell + O(\log(1/\delta))^2$)**.** *If E is hard for exponential size Σ_6-circuits then for every constants $b > 1$, $\alpha > 0$ there exists a constant $c > 1$ such that for every functions $\ell = \ell(n) \leq n$, $\delta = \delta(n) \leq 2^{-n^\alpha}$, and every sufficiently large n, there is a function $G : \{0,1\}^r \to \{0,1\}^n$ that is an (ℓ, ϵ, δ)-re-nb-PRG for circuits of size n^b with $\epsilon = n^{-b}$, and $r = \ell + c \cdot (\log(1/\delta))^2$.*

Note that the dependence of r on ℓ is an additive term of $1 \cdot \ell$. This is best possible, with the correct leading constant. We would have liked the dependence of r on δ to be an additive term of $O(\log(1/\delta))$. Once again, we would get this if we could improve the parameter $\delta = 2^{-\Omega(\sqrt{r})}$ in Theorem 1.17 to $\delta = 2^{-\Omega(r)}$. We also remark that the requirement that $\delta \leq 2^{-n^\alpha}$ may be omitted, and then $r = \ell + c \cdot ((\log(1/\delta))^2 + \log n)$.

1.14 Cryptographic applications of re-nb-PRGs

Dubrov and Ishai [11] observe that nb-PRGs can be used to reduce the randomness complexity of parties in multi-party cryptographic protocols. They consider the setup where honest parties run in polynomial time, and security is information theoretic (that is security is guaranteed even against unbounded adversaries). The precise details can be found in [11]. When using nb-PRGs, this application requires nb-PRGs with small error, as the probability of a security breach in the final protocol is additive in the error of the nb-PRG. However, assuming the probability of a security breach in the original protocol is at most δ (for some negligible δ), we can use $(\ell, 1, \delta)$-re-nb-PRGs to "preserve the probability of small events" and obtain a protocol with reduced randomness complexity, and where the probability of a security breach is at most $4 \cdot \delta$.

The key idea in the application above is that the nb-PRG is used to fool "honest parties" rather than "adversaries". This observation is crucial if we want to use NW-style PRGs in cryptography. More precisely, unlike "cryptographic PRGs" which fool circuits of superpolynomial size, NW-style PRGs (such as our re-nb-PRGs) only fool circuits of fixed polynomial size n^b and run in time $\text{poly}(n^b)$. Thus, they are unsuitable to fool cryptographic adversaries (which are more powerful than honest parties).

It is our hope that re-nb-PRGs may find other applications in cryptography. Toward this goal, we present the following toy example of a potential application of re-nb-PRGs: Suppose we are given a one-way function $f : \{0,1\}^{n^3} \to \{0,1\}^n$ that is computable in time n^b and circuits of very large size (say $s = 2^{n^{1/3}}$) cannot invert with probability larger than δ. Can

we reduce the input length of f to say $O(n)$ bits while preserving its security? Note that this only makes sense if we use tools that don't imply a stronger one-way function. We are not aware of such a conversion.

Nevertheless, using a poly(n)-time computable $(n, 1, \delta)$-re-nb-PRG $G : \{0,1\}^{O(n)} \to \{0,1\}^{n^3}$ for size n^b circuits (which we can achieve for $\delta = 2^{-\sqrt{n}}$ under Theorem 1.21) we can argue that $f'(x) = f(G(x))$ is a one-way function where the input length is reduced from n^3 to $O(n)$, and the security of f is preserved: circuits of size s can invert f' with probability at most $4 \cdot \delta$.

2 Overview of the technique

In this section we give a high level overview of the technique used to prove our results.

2.1 HSGs with low error

We assume that E is hard for exponential size nondeterministic circuits, and construct a poly(n)-time computable ϵ-HSG $G : \{0,1\}^{O(\log n)+\log(1/\epsilon)} \to \{0,1\}^n$ for circuits of fixed polynomial size. By Theorem 1.7 our assumption implies a poly(n)-time computable $\frac{1}{2}$-HSG $G' : \{0,1\}^{O(\log n)} \to \{0,1\}^{2n}$ for nondeterministic circuits of fixed polynomial size. It is standard that using $z \leftarrow U_{2n}$, we can produce $t = O(1/\epsilon) \leq 2^n$ pairwise independent random variables $Y_1(z), \ldots, Y_t(z)$ of length n. Furthermore, even though t may be super-polynomial, there is a polynomial time algorithm that given z, i, outputs the i'th variable $Y_i(z)$.

Our generator G will receive two seeds: a seed x for G', and an $i \in [t]$. It uses x to prepare a $2n$ bit long output string $z = G'(x)$, and then uses z as a seed to generate the i'th random variable $Y_i(z)$.

Let $D : \{0,1\}^n \to \{0,1\}$ be some fixed polynomial size deterministic circuit with $\Pr[D(U_n) = 1] \geq \epsilon$, and let $B = \{x : D(x) = 1\}$. By Chebyshev's inequality, pairwise independent variables have a "hitting property" for sets B of size at least $\epsilon \cdot 2^n$, meaning that with probability at least $2/3$ over choosing $z \leftarrow U_{2n}$, there exists an $i \in [t]$ such that $Y_i(z) \in B$ which means that $D(Y_i(z)) = 1$. Consider the nondeterministic circuit $C : \{0,1\}^{2n} \to \{0,1\}$, which given $z \in \{0,1\}^{2n}$ accepts iff $\exists i : D(Y_i(z)) = 1$. This is a fixed polynomial size nondeterministic circuit (and jumping ahead we mention that it uses $\log t = \log(1/\epsilon) + O(1)$ nondeterministic bits). We have that $\Pr_{z \leftarrow U_{2n}}[C(z) = 1] \geq 2/3$. Thus by the guarantee on G', there exists a seed x for G' such that $C(G'(x)) = 1$. This in turn means that there exists a seed (x, i) for G, such that $D(G(x, i)) = 1$ as required. The precise argument is given in Section 5.

The proof above uses the standard pairwise independent based randomness efficient amplification of success probability of randomized algorithms with a twist: The circuit C uses its nondeterminism to "speed up" the amplification as it does not have to explicitly go over all t options for i. A technically related (though somewhat different) idea was used by Paturi and Pudlak [30] in the context of "boosting" the success probability of hypothetical efficient randomized circuit-sat algorithms. There, given a circuit D, one considers a deterministic circuit D' which is hardwired with a "good string" z, and on input i, applies C on $Y_i(z)$. The key idea is that the input length of D' is $\log(1/\epsilon) < n$, and this is used to argue that feeding D' (rather than D) to the hypothetical circuit-sat algorithm, allows one to make progress.

2.2 Derandomization of randomized algorithms with large error

By going over all seeds of our ϵ-HSG we can derandomize one-sided error polynomial time algorithms with success probability ϵ and prove Theorem 1.9. We now explain that this

argument extends also to two-sided error algorithms of the form of Theorem 1.14. We make slight modifications in the construction above. This time we require that G' is a $\frac{1}{10}$-PRG for Σ_1-circuits, and by Theorem 1.7 such PRGs follow from the assumption that E is hard for exponential size Σ_1-circuits (and a more careful analysis allows an even weaker assumption). We also increase the number of pairwise independent variables from $O(1/\epsilon)$ to $t = O(1/\epsilon^2)$. We do this, as with this "query complexity", pairwise independent variables give an "averaging sampler", which means that for any set $B \subseteq \{0,1\}^n$, the fraction of Y_i's that land in B is with probability 9/10 close to the volume $\frac{|B|}{2^n}$ of B. Let G be the function obtained by these modifications.

We do not expect to prove that G is an ϵ-PRG, as by Theorem 1.12 such a proof will not be black-box. More concretely, the generator G' has an error of 1/10, and so a 1/10-fraction of its seeds may be useless, and we cannot hope that G has error $< 1/10$.

Let $D : \{0,1\}^n \to \{0,1\}$ be a fixed polynomial size circuit. In the two sided error case, we want to distinguish the case that $\Pr[D(U_n) = 1] \geq 2\epsilon$ from the case that $\Pr[D(U_n) = 1] \leq \epsilon$. We show how to use G in order to distinguish these two cases, in deterministic time $\text{poly}(n)/\epsilon^2$.

In analogy to the previous argument, we can show that with probability 9/10 over $z \leftarrow U_{2n}$, the estimate $p(z) = \frac{1}{t} \cdot |\{i : D(Y_i(z)) = 1\}|$ is very close to the acceptance probability of D. For simplicity, let us cheat and assume equality. The key observation is that by the classical results of [40, 39, 26] on approximate counting of NP witnesses (see Section 4.1 for a precise statement), $p(z)$ can be estimated by a fixed polynomial size Σ_1-circuit $C(z)$. Furthermore, this estimation is sufficiently accurate to distinguish the case that $p(z) \geq 2\epsilon$ from the case that $p(z) \leq \epsilon$. Similarly to the earlier argument, the fact that G' fools C, means that replacing $z \leftarrow U_{2n}$ with $G'(x) : x \leftarrow U_{O(\log n)}$ makes little difference. This means that by going over all $x \in \{0,1\}^{c \log n}$, and checking if for at least half of them $p(G'(x)) \geq 2\epsilon$, we can indeed distinguish the two cases. This takes time $\text{poly}(n)/\epsilon^2$ as required. The precise argument is given in Section 5.

2.3 HSGs with low error imply hardness for weak nondeterministic circuits

Let $G : \{0,1\}^r \to \{0,1\}^n$ be an ϵ-HSG for fixed polynomial size circuits. Note that in Section 2.1 we explained that such HSGs follow from $\frac{1}{2}$-HSGs for fixed polynomial size nondeterministic circuits with roughly $\log(1/\epsilon)$ nondeterministic bits. We now show that G implies such HSGs. Indeed consider, the function G' that outputs the first $n-k$ bits of the output of G, for $k = \log(1/\epsilon) - 1$. We show that G' is a $\frac{1}{2}$-HSG for fixed polynomial size circuits with k nondeterministic bits. Indeed, let $C : \{0,1\}^{n-k} \to \{0,1\}$ be such a circuit that accepts at least half of its inputs. This means that there exists a fixed polynomial size deterministic circuit $D : \{0,1\}^n \to \{0,1\}$, such that $C(x) = 1 \Leftrightarrow \exists y \in \{0,1\}^k$ s.t. $D(x,y) = 1$. The fact that C accepts half of its input implies that D accepts at least $\frac{1}{2} \cdot 2^{-k} = \epsilon$ fraction of the pairs $(x,y) \in \{0,1\}^n$, which implies that there exists a seed s for G such that $D(G(s)) = 1$. This in turn implies that $C(G'(s)) = 1$ as required.

There is an easy general transformation by Impagliazzo, Shaltiel and Wigderson [23] which transforms an HSG into a worst-case hard function in E. This transformation can be used to transform G' into a function that is hard for nondeterministic circuits with few nondeterministic bits. The precise argument is given in Section 6.

2.4 A construction of re-PRGs

Our starting point is a construction of Trevisan and Vadhan [43], which under the assumption that E is hard for exponential size Σ_1-circuits, gives a polynomial time computable function $f : \{0,1\}^n \to \{0,1\}^{n'=\Omega(n)}$ such that for every fixed polynomial size circuit $A : \{0,1\}^n \to \{0,1\}$, $\Pr_{V \leftarrow U_n}[A(V) = f(V)] \leq 2^{-n'/3}$.[8]

A natural approach toward constructing PRGs is to use the Goldreich-Levin theorem [14] to transform f into a *boolean* function g. Indeed, the standard way to do this is to define $g(v,y) = EC(f(v))_y$ where EC is a binary list-decodable code (and the GL theorem is for the special case of the Hadamard code). For this to work, we require that EC has an efficient list-decoding algorithm that can recover from distance $\frac{1}{2} - \epsilon$. In our setting, $\epsilon = 2^{-\Omega(n)}$, and we want a list-decoding algorithm implementable by a polynomial size circuit D. This is obviously impossible, as D needs to read at least $1/\epsilon$ positions in the "received word".

We may hope to circumvent this problem by allowing D to be a poly-size Σ_i-circuit. This will allow D to query the received word in exponentially many positions (on different computation paths). On the one hand, if we assume that E is hard for exponential size Σ_{i+1}-circuits, then the proof of Trevisan and Vadhan (which relativizes) gives security against Σ_i-circuits. However, the lower bounds of Applebaum et al. [2] show that even Σ_i-circuits cannot be used for this task of list decoding binary codes. Indeed, if we could get a boolean function g that cannot be computed with advantage better than $\epsilon = 2^{-\Omega(n)}$ over random guessing, we would obtain an $O(\epsilon)$-PRG by plugging g into the NW generator.

Instead, we shoot for a weaker conclusion, and try to show that the function g has the property that $G(x) = (x, g(x))$ is a $(2^{-\Omega(n)}, n^{-O(1)})$-re-PRG with one bit stretch. (Later, we will be able to get arbitrary stretch by plugging g in the NW-generator). That is, that for every fixed polynomial size circuit C, $\Pr[C(G(U_n)) = 1] \overset{re}{\sim}_{(n^{-O(1)},\delta)} \Pr[C(U_{n+1}) = 1]$ for $\delta = 2^{-\Omega(n)}$. We give a "list-decoding algorithm", that given C, constructs a Σ_2-circuit A that computes the function f too well. This allows us to choose $i = 2$ and start from the assumption that E is hard for exponential-size Σ_3-circuits.

Our "list-decoding algorithm" builds on ideas by Trevisan and Vadhan [43] and Applebaum et al. [2]. For our purposes it is more intuitive to restate the construction of g in an equivalent way: We set $g(v,y) = E(f(v), y)$ where E is a strong extractor with error $\delta^{O(1)}$, which allows seed length and entropy threshold of $O(\log(1/\delta))$.

After a suitable averaging argument, we get that for a non-negligible fraction of good v, a circuit C that distinguishes $G(U_n)$ from U_{n+1}, can be used to distinguish $(Y, E(z^*, Y))$ from uniform for $z^* = f(v)$. The guarantee of strong extractors says that there cannot be more than $poly(1/\delta)$ strings $z \in \{0,1\}^{n'}$ for which this distinguishing is possible. (As the uniform distribution over these z's would be a source on which the extractor fails).

The key observation is that we can design a Σ_1-circuit $B_v(z)$ which uses approximate counting of NP witnesses and accept iff C distinguishes $(Y, E(z,Y))$ from uniform with relative distance. This is because we can use approximate counting to estimate the acceptance probability of C on these two distributions.[9] We have that $z^* = f(v)$ is one of the few z's that B_v accepts. We can guess $z^* = f(v)$ by using random sampling of NP-witnesses [26, 7] to uniformly sample an accepting input of B_v. This strategy can be seen as a Σ_2-circuit A that given v computes $f(v)$ with probability $\delta^{O(1)} = 2^{-\Omega(n)}$, contradicting the hardness of f.

[8] This is another example showing that nondeterministic reductions can achieve very low error.

[9] It is important to note that here we critically use the fact that C distinguishes with relative distance, and we cannot hope to do this for an additive distance of $2^{-\Omega(n)}$. This is the reason why constructing re-PRGs with small δ is easier than constructing δ-PRGs.

We obtain re-PRGs with polynomial stretch by plugging g into the NW-generator. The analysis of the NW-generator can be used (with suitable modifications) to argue that if $G(x) = (x, g(x))$ is an re-PRG then we obtain an re-PRG with larger output with closely related ϵ, δ. An inherent limitation of the NW-generator (that is discussed in detail in [24]) gives that the seed length is quadratic in the input length of g. This is the reason why we get that the seed length has quadratic dependence on $\log(1/\delta)$. The precise argument is given in Section 7.

2.5 A construction of re-nb-PRGs

We first show that an $(\epsilon, \delta \cdot 2^{-\ell})$-re-PRG for Σ_1-circuits is an $(\ell, O(\epsilon), O(\delta))$-re-nb-PRG for deterministic circuits. This implication appears in Section 8.

This means that our previous construction of re-PRGs can give re-nb-PRGs assuming E is hard for exponential size Σ_4-circuits. A disadvantage of this approach is that because of the quadratic loss mentioned above, we obtain seed length approximately $r = O(\ell + \log(1/\delta))^2$. Previous work on nb-PRGs [4, 2] already achieved seed length that is linear in ℓ which is optimal up to constants. We can obtain seed length $r = 1 \cdot \ell + O(\log(1/\delta))^2$. That is, we can remove the quadratic dependence on ℓ but not on $\log(1/\delta)$.

For this, we imitate an approach developed by Applebaum et al. [2], which we can now improve as we can use re-PRGs instead of standard PRGs. We first show that with probability $1 - \delta$, a random $\text{poly}(n^b)$-wise independent hash function $h : \{0,1\}^r \to \{0,1\}^n$ is an re-PRG for size n^b Σ_1-circuits, with excellent dependence of r on ϵ and δ. We then show that checking whether a given circuit h is not an re-PRG for Σ_1-circuits can be done by Σ_3-circuits. Loosely speaking, this is because a Σ_3-circuit can guess a Σ_1-circuit that is not fooled by h, and use approximate counting of NP-witnesses (which costs an NP-oracle) to check whether that circuit is not fooled by the given circuit. (Here again, it is crucial that the notion of distance is relative, so that approximate counting can be used).

Finally, we construct the re-nb-PRG G as follows: We use two seeds x_1 for h and x_2 for an $(n^{-O(1)}, \delta)$-re-PRG G' for Σ_3-circuits (that we have under the assumption that E is hard for exponential size Σ_6-circuits). G computes $G'(x_2)$ and use this to choose a hash function h from the family. The final output is $h(x_1)$.

We have that a random h from the family is an re-PRG for Σ_1-circuits with probability $1 - \delta$, and that Σ_3-circuits can check whether h is an re-PRG for Σ_1-circuits. As re-PRGs preserve the probability of small events, we conclude that with probability $1 - 4\delta$ over the choice of x_2 we obtain a hash function h that is an re-PRG for Σ_1-circuits (which we already showed is an re-nb-PRG for deterministic circuits). Therefore, G is an re-nb-PRG. The precise argument is given in Section 8.

3 Organization of the paper

In Section 4 we state the classical results on approximate counting and sampling of NP witnesses. We also define several notions of relative approximation and prove some useful lemmas regarding them. In Section 5 we construct HSGs with low error, and prove Theorem 1.8. We also show how to derandomize two sided error algorithms and prove Theorem 1.14. In Section 6 we show that HSGs with low error are essentially equivalent to $\frac{1}{2}$-HSGs for nondeterministic circuits with few nondeterministic bits. We also prove Theorem 1.11 and show that HSGs with low error imply lower bounds for nondeterministic circuits with few nondeterministic bits. In Section 7 we give our construction of re-PRGs. In Section 8 we show how to use re-PRGs in order to construct re-nb-PRGs.

4 Preliminaries

4.1 Approximate counting and uniform sampling of NP witnesses

We use the classical result on approximate counting and uniform sampling of NP-witnesses [40, 39, 26, 7], which we now state in a way that is convenient for our application.

▶ **Definition 4.1** (relative approximation)**.** We say that a number p is an ϵ-relative approximation to q if $e^{-\epsilon} \cdot q \leq p \leq e^{\epsilon} \cdot q$.

▶ **Theorem 4.2** (approximate counting [40, 39, 26])**.** *For every $i \geq 0$, every sufficiently large s and every $0 < \epsilon \leq 1$, there is a Σ_{i+1}-circuit of size poly(s/ϵ) that given a Σ_i-circuit C of size s outputs an ϵ-relative approximation of $|\{x : C(x) = 1\}|$.*

▶ **Theorem 4.3** (uniform sampling [26, 7])**.** *For every $i \geq 0$, every sufficiently large s and every $\delta > 0$, there is a randomized size poly$(s, \log(1/\delta))$ Σ_{i+1}-circuit A that given a Σ_i-circuit $C : \{0,1\}^n \to \{0,1\}$ of size $s \geq n$ outputs a value in $\{0,1\}^n \cup \{\perp\}$ such that for every size s Σ_i-circuit, $\Pr[A(C) = \perp] \leq \delta$ and the distribution $(A(C)|A(C) \neq \perp)$ is uniform over $\{x : C(x) = 1\}$.*

4.2 Notions of relative error

In Section 1 we defined a notion of relative distance between two numbers which we notate by $p_1 \overset{re}{\sim}_{(\epsilon,\delta)} p_2$. This notion was used in the definition of re-PRGs and re-nb-PRGs. In this section we discuss properties of this distance, as well as related notions of distance.

▶ **Definition 4.4.** Let p_1, p_2 be two numbers, and let $p_{\max} = max(p_1, p_2)$ and $p_{\min} = min(p_1, p_2)$. We say that p_1, p_2 are

- ϵ-**close** if $p_{\max} - p_{\min} \leq \epsilon$, and use the notation $p_1 \overset{ad}{\sim}_{\epsilon} p_2$.
- (ϵ, δ)-**relative-close** if $p_{\max} \leq e^{\epsilon} \cdot p_{\min} + \delta$, and use the notation $p_1 \overset{re}{\sim}_{(\epsilon,\delta)} p_2$.
- (ϵ, δ)-**relative-threshold-close** if $p_{\max} \leq \delta$ or $p_{\max} \leq e^{\epsilon} \cdot p_{\min}$, and use the notation $p_1 \overset{rt}{\sim}_{(\epsilon,\delta)} p_2$.

The three notions above can be used to define distance between probability distributions. Thus, for example, if X, Y are distributions over a finite set Ω, we write $X \overset{re}{\sim}_{(\epsilon,\delta)} Y$ if for every function $D : \Omega \to \{0,1\}$, $\Pr[D(X) = 1] \overset{re}{\sim}_{(\epsilon,\delta)} \Pr[D(Y) = 1]$.

It is easy to verify the following relationships between the three notions, by using the approximations $1 + x \leq e^x \leq 1 + 3x$ and $1 - x \leq e^{-x} \leq 1 - x/3$ which hold for $0 \leq x \leq 1$

▶ **Lemma 4.5.** *For every numbers $0 \leq p_1, p_2 \leq 1$, and $0 \leq \epsilon, \delta \leq 1$*

- $p_1 \overset{re}{\sim}_{(\epsilon,\delta)} p_2 \Rightarrow p_1 \overset{ad}{\sim}_{3\epsilon+\delta} p_2$.
- $p_1 \overset{rt}{\sim}_{(\epsilon,\delta)} p_2 \Rightarrow p_1 \overset{re}{\sim}_{(\epsilon,\delta)} p_2$.
- *For $\epsilon \leq \frac{1}{2}$,* $p_1 \overset{re}{\sim}_{(\epsilon,\delta)} p_2 \Rightarrow p_1 \overset{rt}{\sim}_{(4\epsilon,4\delta/\epsilon)} p_2$.
- $p_1 \overset{rt}{\sim}_{(\epsilon,\delta)} p_2 \Rightarrow p_1 \overset{ad}{\sim}_{3\epsilon+\delta} p_2$.
- $p_1 \overset{ad}{\sim}_{\delta} p_2 \Rightarrow p_1 \overset{rt}{\sim}_{(\epsilon,3\delta/\epsilon)} p_2$.

In our constructions of re-PRGs and re-nb-PRGs, we will shoot for $\epsilon = n^{-O(1)}$ and $\delta = 2^{-n^{\Omega(1)}}$. Note that by Lemma 4.5, for these choices, the notions of "relative-close" and "relative-threshold-close" are equivalent. It turns out that for our purposes, the notion of "relative-threshold-close" is easier to work with. For this reason we now redefine the notions of re-PRGs and re-nb-PRGs using the notion of relative-threshold-close instead

of relative-close. The definitions are identical except that we use "relative-threshold-close" instead of "relative-close".

▶ **Definition 4.6** (rt-PRGs). A function $G : \{0,1\}^r \to \{0,1\}^n$ is an (ϵ,ρ)-rt-PRG for a class $\mathcal{C}$ of functions $C : \{0,1\}^n \to \{0,1\}$ if for every C in the class $\mathcal{C}$,

$$\Pr[C(G(U_r)) = 1] \overset{rt}{\sim}_{(\epsilon,\rho)} \Pr[C(U_n) = 1].$$

▶ **Definition 4.7** (rt-nb-PRG). Let $\mathcal{C}$ be a class of boolean functions $C : \{0,1\}^n \to \{0,1\}^\ell$. A function $G : \{0,1\}^r \to \{0,1\}^n$ is an (ℓ,ϵ,ρ)-rt-nb-PRG for $\mathcal{C}$ if for every C in $\mathcal{C}$, the probability distributions $C(G(U_r)) \overset{rt}{\sim}_{(\epsilon,\rho)} C(U_n)$.

By the discussion above it immediately follows that:

▶ **Fact 4.8** (rt-PRGs are re-PRGs). *An (ϵ,ρ)-rt-PRG is also an (ϵ,ρ)-re-PRG, and an (ϵ,ρ)-rt-nb-PRG is also an (ϵ,ρ)-re-nb-PRG.*

In the remainder of the paper we will only discuss rt-PRGs.

4.3 Some useful technical lemmas on relative error

The next lemma shows that if we can approximate two quantities p_1, p_2 using an η-relative approximation, for $\eta < \epsilon/10$ then when we can essentially tell if $p_1 \overset{rt}{\not\sim}_{(\epsilon,\rho)} p_2$.

▶ **Lemma 4.9.** *Let $0 \le p_1, p_2 \le 1$ and let p'_1, p'_2 be η-relative approximations of p_1, p_2 respectively. Let $T(p'_1, p'_2)$ be a test that accepts iff $\max(p'_1, p'_2) \ge \rho \cdot e^{-\eta}$ and $\frac{\max(p'_1,p'_2)}{\min(p'_1,p'_2)} \ge e^{\epsilon-2\eta}$. Then,*

- $p_1 \overset{rt}{\not\sim}_{(\epsilon,\rho)} p_2 \Rightarrow T(p'_1, p'_2)$ *accepts.*
- $T(p'_1, p'_2)$ *accepts* $\Rightarrow p_1 \overset{rt}{\not\sim}_{(\epsilon-4\eta,\rho\cdot e^{-2\eta})} p_2$.

We also need the following technical lemma that allows us to perform "Markov style arguments" with relative distance.

▶ **Lemma 4.10.** *Let R, W be independent random variables, and let ψ_1, ψ_2 be boolean functions. Assume that*

$$\Pr[\psi_1(R,W) = 1] \overset{rt}{\not\sim}_{(\epsilon,\rho)} \Pr[\psi_2(R,W) = 1].$$

Let $\epsilon' = \epsilon/10$ and $\rho' = \rho \cdot \epsilon/10$, and let $G = \left\{ r : \Pr[\psi_1(r,W) = 1] \overset{rt}{\not\sim}_{(\epsilon',\rho')} \Pr[\psi_2(r,W) = 1] \right\}$. Then $\Pr[R \in G] \ge \rho \cdot \epsilon/10$.

Proof. Let $a_r = \Pr[\psi_1(r,W) = 1]$, $b_r = \Pr[\psi_2(r,W) = 1]$ and $p_r = \Pr[R = r]$. We can write $a = \Pr[\psi_1(R,W) = 1] = \sum_r p(r) a_r$ and $b = \Pr[\psi_2(R,W) = 1] = \sum_r p(r) b_r$. Assume w.l.o.g. that $a > b$ and we know that $a > e^\epsilon b$ and $a > \rho$. We conclude that $a - b = \sum_r p(r)(a_r - b_r) > \max((e^\epsilon - 1)b, \rho - b)$ and assume that $(a_r - b_r)$ is positive for all r (otherwise we take only positive terms and increase the sum).

Let $A = \{r : a_r > e^{\epsilon'} b_r \wedge a_r > \rho'\}$, $B = \{r : a_r \le \rho'\}$, $C = \{r : a_r \le e^{\epsilon'} b_r \wedge a_r > \rho'\}$.

$$\sum_{r\in A} p(r) \ge \sum_{r\in A} p(r)(a_r - b_r) > \max((e^\epsilon - 1)b, \rho - b) - \sum_{r\in B} p(r)(a_r - b_r) - \sum_{r\in C} p(r)(a_r - b_r).$$

Goal: Construct a poly(n)-time computable ϵ-HSG, $G : \{0,1\}^r \to \{0,1\}^n$ for circuits of size n^b with $r = 1 \cdot \log(1/\epsilon) + O(b \cdot \log(\cdot n))$.

Assumption: E is hard for exponential size nondeterministic circuits.

Parameters:

- b, n - We are shooting to fool circuits of size n^b.
- $\epsilon \geq 2^{-(n-1)}$ - the required error.

Ingredients:

- We make use of a $\frac{1}{2}$-HSG for nondeterministic circuits. Specifically, let $b' = a \cdot b$ for a constant $a > 1$ to be chosen later. By Theorem 1.7, there exists a constant $c > 1$ such that the assumption that E is hard for exponential size nondeterministic circuits, implies that for every sufficiently large n, there is a poly(n)-time computable $G' : \{0,1\}^{c \log n} \to \{0,1\}^{2n}$ that is a $\frac{1}{2}$-HSG for size $n^{b'}$ nondeterministic circuits.
- A hitter, that is a function hitter : $\{0,1\}^{2n} \to (\{0,1\}^n)^{4/\epsilon}$ such that for every $B \subseteq \{0,1\}^n$, $\Pr_{Z \leftarrow U_{2n}}[\exists i : \text{ hitter}(Z)_i \in B] \geq \frac{2}{3}$. It is standard that this is achieved by the "pair-wise independent hitter" that uses its $2n$ bit input to sample $4/\epsilon$ pairwise independent n bit variables, (see e.g., [13]). Moreover, given (z, i), hitter$(z)_i$ can be computed in time poly(n).

The HSG: Define $G : \{0,1\}^{c \log n + \log(4/\epsilon)} \to \{0,1\}^n$ by $G(x, i) = \text{hitter}(G'(x))_i$.

Figure 1 An HSG for low error.

Since $\sum_{r \in B} p(r)(a_r - b_r) \leq \rho'$ and $\sum_{r \in C} p(r)(a_r - b_r) \leq (e^{\epsilon'} - 1)b$ we conclude

$$\sum_{r \in A} p(r) > \max((e^{\epsilon} - e^{\epsilon'})b - \rho', \rho - e^{\epsilon'} b - \rho').$$

If $b < 0.25\rho$ then $\rho - e^{\epsilon'} b - \rho' > \rho(1 - e/4 - 1/10) > 0.22\rho$. If $b \geq 0.25\rho$ then $(e^{\epsilon} - e^{\epsilon'})b - \rho' > (0.9 \cdot 0.25 - 0.1)\epsilon\rho = 0.125\epsilon\rho$. So we can conclude that $\Pr[R \in G] = \sum_{r \in G} p(r) \geq \sum_{r \in A} p(r) > 0.125\epsilon\rho$ since $A \subseteq G$. ◀

5 Derandomization of poly-time randomized algorithms with large error

In this section we prove construct the low-error HSG of Theorem 1.8 and show how to extend the argument to handle two-sided error algorithms, proving Theorem 1.14.

5.1 An HSG for low error

We first consider the case of one-sided error algorithms which can be derandomized using hitting-set generators. The following theorem gives a construction of HSGs and implies Theorem 1.8 and Theorem 1.9.

▶ **Theorem 5.1** (HSG with seed length $\log(1/\epsilon) + O(\log n)$)**.** *Let $b > 1$ be a constant, and let $\epsilon = \epsilon(n) \geq 2^{-(n-1)}$. Assume that E is hard for exponential size nondeterministic circuits. Let G be the function constructed in Figure 1, with the parameters chosen there. Then there exists a constant $c > 1$ such that for every sufficiently large n, $G : \{0,1\}^{\log(1/\epsilon) + c \log n} \to \{0,1\}^n$ is an ϵ-HSG for size n^b circuits. Furthermore, G is computable in time poly(n^b).*

Proof. Let $D : \{0,1\}^n \to \{0,1\}$ be a size n^b circuit, let $B = \{y : D(y) = 1\}$, and assume that $|B| \geq \epsilon \cdot 2^n$. Let $T = \left\{z \in \{0,1\}^{2n} : \exists i : \text{ hitter}(z)_i \in B\right\}$. By the properties of hitter

$|T| \geq \frac{2}{3} \cdot 2^{2n}$. Note that by the definition of T, there exists a nondeterministic circuit $C : \{0,1\}^{2n} \to \{0,1\}$ of size $\text{poly}(n) + n^b$ such that $C(z) = 1$ iff $z \in T$. We can choose the constant a to be sufficiently large so that $n^{b'} = n^{a \cdot b}$ is larger than the size of C. It follows that G' fools C and in particular, there exists $x \in \{0,1\}^{c \log n}$ such that $C(G'(x)) = 1$ which implies $D(G(x,i)) = D(\text{hitter}(G'(x)_i) = 1$ as required. ◀

5.2 Extending the argument to 2-sided error randomized algorithms

In this section we prove Theorem 1.14. Our first step is to modify the construction in Figure 1 to use a PRG instead of an HSG and an averaging sampler instead of a hitter. Specifically, we replace hitter : $\{0,1\}^{2n} \to (\{0,1\}^n)^{4/\epsilon}$ with a function samp : $\{0,1\}^{2n} \to (\{0,1\}^n)^t$ for $t = O(1/\epsilon^2)$. that has the property that for every $B \subseteq \{0,1\}^n$ with $|B| \geq \frac{2}{3} \cdot 2^n$,

$$\Pr_{Z \leftarrow U_{2n}} \left[\left| \frac{|\{i : \text{samp}(Z)_i \in B\}|}{t} - \frac{|B|}{2^n} \right| \geq \frac{\epsilon}{10} \right] \leq \frac{1}{10}.$$

It is standard that this is achieved by the "pair-wise independent sampler" that uses its $2n$ bit input to sample $t = O(1/\epsilon)^2$ pairwise independent n bit variables, (see e.g., [13]). We also require that G' is a 1/10-PRG for Σ_1-circuits of size $n^{b'}$ (rather than a $\frac{1}{2}$-HSG for nondeterministic circuits of size $n^{b'}$). This follows just the same from Theorem 1.7, if we strengthen the assumption, and assume that E is hard for exponential size Σ_1-circuits. We repeat the construction of Figure 1 with these choices, and let G denote the final function G obtained in Figure 1 with the modifications above. We prove the following extension of Theorem 5.1.

▶ **Theorem 5.2.** *Let $b > 1$ be a constant, and let $\epsilon = \epsilon(n) \geq 2^{-n/2}$. Assume that E is hard for exponential size Σ_1-circuits. Let G be the function constructed in Figure 1, with the parameters chosen there and the modifications explained above. Then there exists a constant $c > 1$ such that for every sufficiently large n, $G : \{0,1\}^{c \log n} \times \{0,1\}^{\log t} \to \{0,1\}^n$ satisfies that for every size n^b circuit $D : \{0,1\}^n \to \{0,1\}$:*

- *If $\Pr[D(U_n) = 1] \geq 2 \cdot \epsilon$ then $\Pr_{X \leftarrow U_{c \log n}} \left[\frac{|\{i : D(G(X,i)) = 1\}|}{t} \geq \frac{3}{2} \cdot \epsilon \right] \geq \frac{4}{5}$.*
- *If $\Pr[D(U_n) = 1] \leq \epsilon$ then $\Pr_{X \leftarrow U_{c \log n}} \left[\frac{|\{i : D(G(X,i)) = 1\}|}{t} \geq \frac{3}{2} \cdot \epsilon \right] \leq \frac{1}{5}$.*

Furthermore, G is computable in time poly(n^b).

By the discussion in Section 1.8 we cannot use black-box techniques to construct poly-time computable PRGs with low error under the assumption of Theorem 5.2. Theorem 5.2 does not contradict the discussion as the constructed object G is not a PRG. It is not the case that $G(U_r)$ indistinguishable from uniform with very low error. Nevertheless, the guarantee on G suffices for derandomization in time exponential in the seed length (as is the case in PRGs).

Proof. (of Theorem 5.2) Let $D : \{0,1\}^n \to \{0,1\}$ be a size n^b circuit, let $B = \{y : D(y) = 1\}$. Let

$$T = \left\{ z \in \{0,1\}^{2n} : \left| \frac{|\{i : \text{samp}(z)_i \in B\}|}{t} - \frac{|B|}{2^n} \right| \leq \frac{\epsilon}{10} \right\}.$$

By the properties of samp, $|T| \geq \frac{9}{10} \cdot 2^{2n}$. It follows that:

- If $\Pr[D(U_n) = 1] \geq 2 \cdot \epsilon$ then $\Pr_{Z \leftarrow U_{2n}} \left[\frac{|\{i : D(\text{samp}(Z)_i) = 1\}|}{t} \geq \frac{7}{4} \cdot \epsilon \right] \geq \frac{9}{10}$.
- If $\Pr[D(U_n) = 1] \leq \epsilon$ then $\Pr_{Z \leftarrow U_{2n}} \left[\frac{|\{i : D(\text{samp}(Z)_i) = 1\}|}{t} \leq \frac{5}{4} \cdot \epsilon \right] \geq \frac{9}{10}$.

Consider the Σ_1-circuit $C : \{0,1\}^{2n} \to \{0,1\}$ that works as follows: given input $z \in \{0,1\}^{2n}$, C uses Theorem 4.2 to compute an η-relative approximation p' of $p = |\{i \in [t] : D(\mathrm{samp}(z)_i) = 1\}|$, for $\eta = 1/100$. The circuit C accepts iff $p' \geq t \cdot \frac{3}{2} \cdot \epsilon$. It follows that C is a Σ_1-circuit of size $\mathrm{poly}(n^b)$. The quality of approximation is sufficient to distinguish the case that $p \geq \frac{7}{4} \cdot \epsilon$ and $p \leq \frac{5}{4} \cdot \epsilon$.

We can choose the constant a to be sufficiently large so that $n^{b'} = n^{a \cdot b}$ is larger than the size of C. It follows that G' fools C, and the theorem follows.[10] ◀

We are now ready to prove Theorem 1.14.

Proof of Theorem 1.14. Let $T(n) \geq n$ be a bound on the running time of A. Given an input $x \in \{0,1\}^n$, we consider the circuit $D_x : \{0,1\}^{T(n)} \to \{0,1\}$, which given y simulates $A(x)$ using y as random coins. We apply Theorem 5.2 to obtain a constant c and a function

$$G : \{0,1\}^{c \log T(n)} \times \{0,1\}^{\log t(n)} \to \{0,1\}^{T(n)},$$

where $t(n) = O(1/\epsilon(n)^2)$. By applying the guarantee of Theorem 5.2 on D_x we get that

- $x \in L \Rightarrow \Pr[D_x(U_n) = 1] \geq 2 \cdot \epsilon] \Rightarrow \Pr_{S \leftarrow U_{c \log T(n)}} \left[\frac{|\{i : D_x(G(S,i)) = 1\}|}{t(n)} \geq \frac{3}{2} \cdot \epsilon(n) \right] \geq \frac{4}{5}$.
- $x \notin L \Rightarrow$ If $\Pr[D(U_n) = 1] \leq \epsilon] \Rightarrow \Pr_{S \leftarrow U_{c \log T(n)}} \left[\frac{|\{i : D_x(G(S,i)) = 1\}|}{t(n)} \geq \frac{3}{2} \cdot \epsilon(n) \right] \leq \frac{1}{5}$.

The deterministic algorithm works as follows: We go over all $s \in \{0,1\}^{c \log T(n)}$. For each one we count the number of $i \in [t(n)]$ for which D_x accepts $G(s,i)$. If the fraction of s, such that

$$\frac{|\{i : D_x(G(s,i)) = 1\}|}{t(n)} \geq \frac{3}{2} \cdot \epsilon(n)$$

is larger than $\frac{4}{5}$, we accept. The correctness of this simulation follows. The running time is $t(n) \cdot \mathrm{poly}(T(n)) = \mathrm{poly}(T(n))/\epsilon(n)^2$. ◀

6 On minimal hardness assumptions for HSGs with low error

We are using hardness for nondeterministic circuits in Theorem 1.8 which constructs an ϵ-HSG with very low error. Is this assumption necessary?

In this section we address this question. We will consider a natural intermediate model of circuits that are stronger than deterministic circuits, and weaker than nondeterministic circuits, namely nondeterministic circuits that use $k \leq n$ nondeterministic bits (as defined in Definition 1.4).

6.1 $\frac{1}{2}$-HSGs for nondeterministic circuits with few nondeterministic bits

An inspection of our construction of HSG in Figure 1 reveals that the assumption that E is hard for exponential size nondeterministic circuits was only used to obtain a $\frac{1}{2}$-HSG for nondeterministic circuits with a number of nondeterministic bits that is roughly $k = \log(1/\epsilon)$. More precisely, our construction gives the following general conversion.

[10] Note that the circuit C uses its oracle only to perform approximate counting. It can be verified that this can be done by a circuit C that makes nonadaptive queries to its oracle. This means that for this argument it is sufficient that G' fools circuits of this type, and by the "downward collapse theorem" of Shaltiel and Umans [36], coupled with Theorem 1.7, such PRGs follow under the seemingly weaker assumption that E is hard for exponential size nondeterministic circuits.

▶ **Theorem 6.1.** *For every constant $b > 1$ there is a constant $b' > b$ such that for every sufficiently large n, if $G : \{0,1\}^r \to \{0,1\}^{2n}$ is a $\frac{1}{2}$-HSG for size $n^{b'}$ nondeterministic circuits that use $k = \log(1/\epsilon) + O(1)$ nondeterministic bits, then there is a function $G' : \{0,1\}^{r+k} \to \{0,1\}^n$ that is an ϵ-HSG for circuits of size n^b. Furthermore, G can be computed in time poly(n) with one oracle call to G.*

We can show the following partial converse:

▶ **Lemma 6.2.** *Let $G : \{0,1\}^r \to \{0,1\}^n$ be an ϵ-HSG for circuits of size s, and let $G' : \{0,1\}^r \to \{0,1\}^{n-k}$ be $G'(x) = G(x)|_{1,\ldots,n-k}$ for $k = \log(1/\epsilon) - 1$. G' is a $\frac{1}{2}$-HSG for size s nondeterministic circuits with k nondeterministic bits.*

Proof. Let $C : \{0,1\}^{n-k} \to \{0,1\}$ be a size s nondeterministic circuit with k nondeterministic bits, which accepts at least half of its inputs. That is, there exists a deterministic circuit $D : \{0,1\}^{n-k} \times \{0,1\}^k \to \{0,1\}$ of size s such that: for every $x \in \{0,1\}^{n-k}$,

$$C(x) = 1 \;\Leftrightarrow\; \exists y \in \{0,1\}^k \text{ s.t. } D(x,y) = 1,$$

and $\Pr[C(U_{n-k}) = 1] \geq \frac{1}{2}$. It follows that $\Pr[D(U_n) = 1] \geq \frac{1}{2} \cdot 2^{-k} = \epsilon$ (here we view D as a circuit with n input bits). Thus, by the guarantee of the HSG, there exists an $s' \in \{0,1\}^r$ such that $D(G(s')) = 1$. Denote $G(s') = (x', y')$ so that $G'(s') = x'$. It follows that $D(x', y') = 1$ which implies that $C(x') = 1$, and we have that $C(G'(s')) = 1$. ◀

This means that in the case that $r = \Omega(\log(1/\epsilon))$ the notions of $\frac{1}{2}$-HSG for nondeterministic circuits with $k = \log(1/\epsilon)$ bits of nondeterminism, and the notion of ϵ-HSGs for standard circuits are essentially equivalent (in the sense that conversions between them incur only slight penalties in seed length and circuit size).

Consequently, if we are interested in low error HSGs for deterministic circuits that have polynomial stretch, we should be interested in HSGs against the class of polynomial size nondeterministic circuits on n bits with $\gamma \cdot n$ nondeterministic bits, for a small $\gamma > 0$.

6.2 Hardness assumptions that imply HSGs for circuits with weak nondeterminism

How hard is it to construct $\frac{1}{2}$-HSGs for poly-size nondeterministic circuits with $\gamma \cdot n$ nondeterministic bits? Given the success of the hardness versus randomness paradigm exhibited in Theorems 1.3 and Theorem 1.7, we can hope that hardness for this circuit class, translates into pseudorandomness for this circuit class. If this is the case, we can start from the assumption that there exists a $\gamma > 0$ such that E is hard for exponential size nondeterministic circuits that use γn nondeterministic bits.

An inspection of the hardness versus randomness tradeoffs in the literature reveal that they do not give such a result. Loosely speaking, because of the need for a hybrid argument, the reductions need to "perform decoding" from error less than $1/n$ and make a super-linear number queries to the distinguisher circuit. This overall means that we require hardness against circuits with a super-linear number of nondeterministic bits.

This is a pity because it trivially follows that nondeterministic circuits of size s with k nondeterministic bits can be simulated by deterministic circuits of size $s \cdot 2^k$, and this implies that:

▶ **Fact 6.3.** *If E is hard for exponential size circuits, then there exists a $\gamma > 0$ such that E is hard for exponential size nondeterministic circuits that use γn nondeterministic bits.*

Consequently, a hardness versus randomness tradeoff of the form we hope for, would be able to start from the assumption that E is hard for exponential size circuits. By the discussion in the introduction, such hardness versus randomness tradeoffs cannot have black-box proofs.

6.3 Hardness assumptions implied by HSGs with low error

In the previous section we observed that it is unlikely that we can prove that hardness for nondeterministic circuits with few nondeterministic bits implies HSGs with low error against deterministic circuits. We are able to prove the other direction. Specifically, we show that HSGs with low error and polynomial stretch, that run in time exponential in their seed length, imply that E is hard for polynomial size nondeterministic circuits with $\Omega(n)$ nondeterministic bits on inputs of length n. The following is a restatement of Theorem 1.11

▶ **Theorem 6.4.** *Let $\delta > 0$ be a constant. Assume that for every sufficiently large n, there is a 2^{-n^δ}-HSG $H : \{0,1\}^{O(n^\delta)} \to \{0,1\}^n$ for size $s \geq n$ circuits, and furthermore that the family of functions $H = \{H_n\}$ is computable in time exponential in the seed length, that is time $2^{O(n^\delta)}$. Then, there exists a constant $\gamma > 0$, and a problem $L \in E$ such that for every sufficiently large n', nondeterministic circuits of size $(\gamma n')^{1/\delta}$ with $\gamma \cdot n'$ nondeterministic bits fail to compute the characteristic function of L on inputs of length n'.*

Our current state of knowledge doesn't give us any reason to think that HSGs with $\epsilon = 1/n$ imply the same conclusion.

We use the following simple argument from [23] to prove the following:

▶ **Theorem 6.5.** *Let $H : \{0,1\}^r \to \{0,1\}^n$ be a $\frac{1}{2}$-HSG for size s nondeterministic circuits that use k bits. Let $f : \{0,1\}^{r+2} \to \{0,1\}$ be the function*

$$f(y) = 0 \;\Leftrightarrow\; \exists z \in \{0,1\}^{n-(r+2)}, \exists x \in \{0,1\}^n \text{ s.t. } H(x) = y \circ z$$

where "$\circ$" denotes concatenation. f cannot be computed by size s-circuits that use k bits of nondeterminism.

Proof. A circuit $C : \{0,1\}^{r+2} \to \{0,1\}$ computing f, can be thought of a circuit $C : \{0,1\}^n \to \{0,1\}$ (that only looks at the $r+2$ prefix of its input). It is immediate that $\Pr[C(U_n) = 1] \geq \frac{3}{4}$ and yet C answers zero on all outputs of G. ◀

Theorem 6.4 now follows by converting the low error HSG into a $\frac{1}{2}$-HSG for nondeterministic circuits with few nondeterministic bits, and then using Theorem 6.5 to convert the HSG into a hard function.

Proof of Theorem 6.4. Let c be the constant hidden in the seed length of H, and let n be sufficiently large. By Lemma 6.2 we have that $H' : \{0,1\}^r \to \{0,1\}^{n-k}$ is a $\frac{1}{2}$-HSG for size $s \geq n$, nondeterministic circuits with k bits of nondeterminism, for $k = n^\delta - 1$. Let $n' = c \cdot n^\delta + 2$, and let $f : \{0,1\}^{n'} \to \{0,1\}$ be the function obtained by applying Theorem 6.5 on H'. We have that f cannot be computed by size s-circuits that use k bits of nondeterminism. Note that $s \geq n \geq \Omega(n')^{1/\delta}$, and $k \geq \Omega(n')$. Let L be the language of the decision problem f. It follows that $L \in \mathrm{E}$ as f can be computed by running over all $2^{O(n^\delta)}$ seeds of G and computing G, and this takes time $2^{O(n^\delta)} = 2^{O(n')}$. ◀

7 A construction of an re-PRG

In this section we construct re-PRGs and prove Theorem 1.17. We will actually construct rt-PRGs which are in particular re-PRGs. We begin by constructing a poly(n)-time computable function $g : \{0,1\}^n \to \{0,1\}$, such that the function $G : \{0,1\}^n \to \{0,1\}^{n+1}$ defined by $G(x) = (x, g(x))$ is an $(1/\text{poly}(n), 2^{-\Omega(n)})$-rt-PRG for fixed polynomial size circuits. This construction is given in Section 7.1 and builds on ideas from [43, 2]. In order to obtain an re-PRG with polynomial stretch we apply the Nisan-Wigderson generator [29] where the function g plays the role of the hard function. The analysis of the NW-generator can be used (with some modifications) to argue that this yields an rt-PRG.

7.1 rt-PRGs with one bit stretch

7.1.1 The construction

We use the following result by Trevisan and Vadhan [43].

▶ **Theorem 7.1** ([43]). *Let $i \geq 0$. If E is hard for exponential size Σ_{i+1}-circuits, then for every constant $b > 1$, there exists some constant $\alpha > 0$ such that for every sufficiently large n, there is a function $f : \{0,1\}^n \to \{0,1\}^{m=\alpha \cdot n}$ such that for every size n^b, Σ_i-circuit A, $\Pr_{X \leftarrow U_n}[A(X) = f(X)] \leq 2^{-m/3}$. Furthermore, f is computable in time poly(n^b).*

▶ Remark. Theorem 7.1 is not stated in this form in [43]. Nevertheless, it is implicit in the work of [43] as we now explain.

Lemma 5.1 in [43] states that if a circuit C computes a degree d multivariate polynomial $p : \mathbb{F}^t \to \mathbb{F}$ (over a field $\mathbb{F}$ of size q) correctly on an ϵ fraction of its inputs (and if certain conditions on the parameters t, d, q and ϵ are met) then there exists a Σ_j-circuit C' that computes p correctly on all inputs, and the size of C' is polynomial in the size of C and in $t, d, \log q$ (but does not depend on ϵ). The lemma claims this for $j = 2$, but we will later observe that this holds also for $j = 1$.

The parameters in the lemma allow ϵ which is roughly $\sqrt{d/q}$ and thinking of p as a boolean function outputting $\log q$ bits, this allows ϵ to approach $2^{-\frac{1}{2} \cdot \log q}$ if $d \ll q$.

By using the "low degree extension" it is standard that E has complete problems that can be represented as such low degree polynomials. More precisely, given an input length n, we consider a restriction of an E complete problem to inputs of length $\ell = c \log n$ and perform the low degree extension with $d = 2^{\gamma \cdot \ell}$, where $\gamma > 0$ is a small constant, $t = O(1/\gamma)$ and huge field size $q = 2^{n/t}$ so that the input length of p (in bits) is $t \log q = n$ and the output length is $\log q = \Omega(n)$. It follows that p can be computed in time $2^{O(c\ell)} = n^{O(c)}$ and if we assume that E is hard for Σ_j-circuits then p cannot be computed by circuits of size $2^{\beta \cdot \ell} = n^{\beta c}$, which we can control by choosing c. The reader can also find this argument in the proof of Theorem 5.5 in [43].

From this, Lemma 5.1 in [43] gives that p (interpreted as a function with boolean input and output) is a function that is hard on average. This proves a version of Theorem 7.1 in which Σ_1 is replaced by Σ_2. The stronger statement (for Σ_1) follows because Lemma 5.1 in [43] also holds for $j = 1$. This was stated in an earlier version of [43] and follows by a more efficient implementation of the proof.

More specifically, the proof of Lemma 5.1 constructs the circuit C' by first specifying a randomized procedure which for every input $x \in \mathbb{F}^t$ computes $p(x)$ correctly with probability say $2/3$. The procedure requires "nonuniform advice" about p in the form of a point $z \in \mathbb{F}^t$ and its evaluation $p(z)$ (the existence of a "good point" z is shown in the proof). The

Goal: Construct a poly(n)-time computable $(n^{-b}, 2^{-\Omega(n)})$-rt-PRG, $G : \{0,1\}^n \to \{0,1\}^{n+1}$ for Σ_i-circuits of size n^b.

Assumption: E is hard for exponential size Σ_{i+3}-circuits.

Parameters:

- i, b, n - We are shooting to fool Σ_i-circuits of size n^b.
- We set $\rho = 2^{-\mu n}$ for a constant $0 < \mu < \frac{1}{2}$ to be specified later.
- We set $t = \nu n$ for a constant $\nu > \frac{1}{2}$ to be specified later.

Ingredients:

- We make use of an exponentially hard on average function due to Trevisan and Vadhan [43]. Specifically, let $b' = a \cdot b$ for a constant $a > 1$ to be chosen later. By Theorem 7.1, there exists a constant $\alpha > 0$ such that under the assumption that E is hard for exponential size Σ_{i+3}-circuits, for every sufficiently large n, there is a polynomial time computable function $f : \{0,1\}^t \to \{0,1\}^{t'=\alpha \cdot t}$ such that for every size $n^{b'}$, Σ_{i+2}-circuit A, $\Pr_{X \leftarrow U_t}[A(X) = f(X)] \leq 2^{-t'/3}$.
- A strong $(k = O(\log(1/\rho)), \rho/2000)$-extractor, $E : \{0,1\}^{t'} \times \{0,1\}^d \to \{0,1\}$, with seed length $d = O(\log(1/\rho))$. There are constructions of polynomial time computable extractors with these parameters [17]. (In fact, such extractors immediately follow from binary poly-time $(1/2 - \Omega(\rho), \text{poly}(1/\rho))$-list-decodable codes with rate poly(ρ) [42]. We have that $d = O(\log(1/\rho) = e \cdot \mu \cdot n$ for some constant $e > 1$. We now specify the constant ν (which was used to define $t = \nu \cdot n$) so that the equality $n = t + d$ holds, we will later choose $\mu > 0$ to be sufficiently small so that we indeed have that $\nu > 1/2$ as promised.

The rt-PRG: Define $g : \{0,1\}^t \times \{0,1\}^d \to \{0,1\}$, by $g(v, y) = E(f(v), y)$. The final generator $G : \{0,1\}^n \to \{0,1\}^{n+1}$ is given by $G(v, y) = (v, y, g(v, y))$.

Figure 2 An rt-PRG with one bit stretch.

computation of the procedure can be expressed in the following form: Given x, the procedure nondeterministically guesses polynomially many strings $h_1, \ldots, h_\ell$ of polynomial length. For each one it prepares a circuit T_i which depends on h_i (and also on $x, z, p(z)$). The procedure then uses approximate counting to verify that sufficiently many of the T_i's accept more than a fixed number of inputs.

Indeed, such a computation can be described (after removing the random coins) by a nondeterministic circuit that uses an NP-oracle to perform approximate counting. Overall, this gives a Σ_2-circuit.

However, approximate counting is only used to check that a given circuit accepts more than a fixed number of inputs. The problem of checking that the number of accepting inputs is larger than a fixed quantity is easier than approximating the number of accepting inputs: it can be solved by an Arthur-Merlin protocol as shown by Goldwasser and Sipser [16]. With this implementation, the entire procedure can be seen (after removing the randomness) as a Σ_1-circuit.

Our construction is given in Figure 2. Note that an intuitive overview is given in Section 2.

▶ **Theorem 7.2** (rt-PRG with one bit stretch)**.** *Let $i \geq 0, b > 1$ be constants. Assume that E is hard for exponential size Σ_{i+3} circuits. Let g, G be the functions constructed in Figure 2, with the parameters chosen there. Then there exists a constant $\mu > 0$ such that for every sufficiently large n, $G(x) = (x, g(x))$ is an (n^{-b}, ρ)-rt-PRG for size n^b, Σ_i-circuits, for $\rho = 2^{-\mu \cdot n}$. Furthermore, g is computable in time poly(n^b).*

7.1.2 Proof of Theorem 7.2

We start by proving the following lemma (which captures the role that strong seeded extractors play in the proof).

▶ **Lemma 7.3.** *Let* $E : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}$ *be a* (k,ρ)*-strong extractor, and let* $T : \{0,1\}^d \times \{0,1\} \to \{0,1\}$ *be a function. Let*

$$Z = \left\{ z : \Pr_{Y \leftarrow U_d}[T(Y, E(z,Y)) = 1] \overset{ad}{\not\sim}_{\rho} \Pr_{Y \leftarrow U_d, W \leftarrow U_1}[T(Y,W) = 1] \right\}.$$

Then $|Z| \leq 2 \cdot 2^k$.

Proof. We can partition Z into two sets according to which of the two terms in the definition of Z is the maximal one. If $|Z| > 2 \cdot 2^k$ then we can assume w.l.o.g

$$|\{z : \Pr_{Y \leftarrow U_d}[T(Y, E(z,Y)) = 1] - \Pr_{Y \leftarrow U_d, W \leftarrow U_1}[T(Y,W) = 1] > \rho\}| > 2^k.$$

Let X be a random variable that is uniformly distributed over Z and note that $H_\infty(X) > k$. E is a (k,ρ)-strong extractor which implies that $(Y, E(X,Y))$ is ρ-close to (Y,W). This implies that $|\Pr[T(Y, E(X,Y)) = 1] - \Pr[T(Y,W) = 1]| \leq \rho$, and we get a contradiction. ◀

We are now ready to prove Theorem 7.2. We assume (for contradiction) that G is not a (n^{-b},ρ)-rt-PRG for size n^b Σ_i-circuits, and our goal is to show that there is a Σ_{i+2}-circuit A of size $n^{b'}$ such that $\Pr_{X \leftarrow U_t}[A(X) = f(X)] > 2^{-t'/3}$.

Our assumption says that there exists a size n^b Σ_i-circuit C such that $\Pr[C(G(U_n)) = 1] \overset{rt}{\not\sim}_{(n^{-b},\rho)} \Pr[C(U_{n+1}) = 1]$. For the remainder of the proof, let us consider a probability space that consists of three independent random variables: $V \leftarrow U_t$, $Y \leftarrow U_d$ and $U \leftarrow U_1$. By the construction of G we have that:

$$\Pr[C(V, Y, E(f(V), Y)) = 1] \overset{rt}{\not\sim}_{(n^{-b},\rho)} \Pr[C(V,Y,U) = 1]$$

We now apply Lemma 4.10. For this purpose we set $R = V$, $W = (Y,U)$. We use the notation $W_1 = Y$ and $W_2 = U$. Let $\psi_1(r,w) = C(r, w_1, E(f(r), w_1))$ and $\psi_2(r,w) = C(r, w_1, w_2)$. We apply the lemma and obtain that:

▶ **Claim 7.4.** *Let* $\epsilon' = n^{-b}/10, \rho' = \rho \cdot n^{-b}/10$ *and let*

$$G = \left\{ v \in \{0,1\}^t : \Pr[C(v, Y, E(f(v), Y)) = 1] \overset{rt}{\not\sim}_{(\epsilon',\rho')} \Pr[C(v,Y,U) = 1] \right\}.$$

It follows that $\Pr[V \in G] \geq \rho \cdot n^{-b}/10$.

Recall that our goal is to contradict the hardness of f by showing the existence of a suitable circuit A that compute $f(v)$ too well given a random v. For every $v \in G$, we define $T_v(y,b) = C(v,y,b)$ so that T_v distinguishes $(Y, E(f(v),Y))$ from (Y,U) in the sense that $\Pr[T_v(Y, E(f(v),Y)) = 1] \overset{rt}{\not\sim}_{(\epsilon',\rho')} \Pr[T_v(Y,U) = 1]$. By Lemma 4.5, this implies that $\Pr[T_v(Y, E(f(v),Y)) = 1] \overset{ad}{\not\sim}_{\rho' \cdot \epsilon'/3} \Pr[T_v(Y,U) = 1]$. This is helpful because by Lemma 7.3 there aren't that many $z \in \{0,1\}^{t'}$ for which T_v distinguishes $(Y, E(z,Y))$ from (Y,U), and yet $z^* = f(v)$ is one of those z's. We will use this property to construct a small Σ_{i+2} circuit A that given a $v \in G$, produces $f(v)$ with probability $\rho^{O(1)}$, and this will be a contradiction. The description of A appears in Figure 3. We first present A as a randomized circuit that tosses coins, and will later fix its coins to give a circuit that does not toss coins. The correctness of A will follow from the following claims.

Goal: A size $n^{b'}$ Σ_{i+2}-circuit C that computes f with probability $2^{-t'/3}$.

Description: On input $v \in \{0,1\}^t$, A computes as follows:

- Prepare the Σ_i-circuit $T_v(y,b) = C(v,y,b)$
- Prepare the Σ_{i+1}-circuit B_v, defined as follows:
  - On input $z \in \{0,1\}^{t'}$, B_v computes as follows:
  - B_v prepares the circuits $D_1^v : \{0,1\}^d \to \{0,1\}$ and $D_2^v : \{0,1\}^{d+1} \to \{0,1\}$ defined by $D_1(y) = T_v(y, E(z,y))$ and $D_2(y,b) = T_v(y,b)$. Note that these are circuits of size $\text{poly}(n^b)$.
  - Let p_1, p_2 be the number of accepting inputs of D_1^v, D_2^v respectively. Let $\eta = \epsilon'/10 = n^{-b}/100$. B uses Theorem 4.2 to compute η-relative approximations p_1', p_2' to p_1, p_2. Note that by Theorem 4.2, this can be done in a size $\text{poly}(n^b)$ Σ_{i+1}-circuit.
  - B_v accepts z if $\max(p_1', p_2') \geq \rho' e^{-\eta}$ and $\max(p_1', p_2')/\min(p_1', p_2') \geq e^{\epsilon' - 2\eta}$. This choice is made so that by Lemma 4.9:
    - $p_1 \overset{rt}{\not\sim}_{(\epsilon',\rho')} p_2 \Rightarrow B_v$ accepts z, and
    - B_v accepts $z \Rightarrow p_1 \overset{rt}{\not\sim}_{(\epsilon'',\rho'')} p_2$, where $\epsilon'' = \epsilon' - 4\eta \geq \epsilon'/2$ and $\rho'' = \rho' \cdot e^{-2\eta} \geq \rho'/2$.
- The circuit A uses Theorem 4.3 to sample an accepting input z of B_v (with error $\delta = 2^{-n}$). Note that this can be done with a size $\text{poly}(n^b)$ Σ_{i+2}-circuit. Overall, the size of A is $\text{poly}(n^b)$. Recall that in Figure 2 we have chosen $b' = a \cdot b$ for an unspecified constant a. Note that A is indeed a Σ_{i+2}-circuit, and we can now choose a to be sufficiently large so that the size of A is $\text{poly}(n^b) = n^{a \cdot b} = n^{b'}$.
- Finally, the circuit A outputs z.

Figure 3 Circuit A: A Σ_{i+2}-circuit that computes f correctly with noticeable probability.

▶ **Claim 7.5.** *For every $v \in G$, B_v accepts $f(v)$.*

Proof. Fix some $v \in G$. By Claim 7.4 and the definition of T_v, $\Pr[T_v(Y, E(f(v), Y)) = 1] \overset{rt}{\not\sim}_{(\epsilon',\rho')} \Pr[T_v(Y,U) = 1]$. When B_v receives $z = f(v)$ as input, it prepares the circuit $D_1(y) = T_v(y, E(f(v), y))$ and $D_2(y,b) = T_v(y,b)$. Recall that p_1, p_2 are the number of accepting inputs of these two circuits. Therefore, we have that $p_1 \overset{rt}{\not\sim}_{(\epsilon',\rho')} p_2$, and by construction, B_v accepts z. ◀

▶ **Claim 7.6.** *For every $v \in G$, B_v accepts at most $2^{k+1} = (\frac{1}{\rho})^{O(1)}$ inputs.*

Proof. By construction, if B_v accepts an input z, then the quantities p_1, p_2 in Figure 3, satisfy $p_1 \overset{rt}{\not\sim}_{(\epsilon' - 4\eta, \rho' \cdot e^{-2\eta})} p_2$. By Lemma 4.5, this implies that $p_1 \overset{ad}{\not\sim}_{\epsilon'' \cdot \rho''/3} p_2$, and note that

$$\epsilon'' \cdot \rho''/3 \geq \epsilon' \cdot \rho'/12 \geq n^{-2b} \cdot \rho/1200 \geq \rho/2000.$$

Therefore, if B_v accepts z, then $\Pr[T_v(Y, E(z,Y)) = 1] \overset{ad}{\not\sim}_{\rho/2000} \Pr[T_v(Y,U) = 1]$ and by Lemma 7.3, there are at most $2 \cdot 2^k = \text{poly}(1/\rho)$ such strings z. ◀

The two claims above give that

▶ **Claim 7.7.** *For every $v \in G$, $\Pr[A(v) = f(v)] \geq 2^{-(k+1)} = \rho^{O(1)}$ (where the probability is over the coin tosses of A).*

It follows that

$$\Pr[A(V) = f(V)] \geq \Pr[V \in G] \cdot \Pr[A(V) = f(V) | V \in G] \geq$$
$$\rho \cdot n^{-b}/10 \cdot \rho^{O(1)} = \rho^{O(1)} = 2^{-O(\mu \cdot n)} \geq 2^{-t'/3},$$

where the last step follows because we can choose $\mu > 0$ to be sufficiently small so that

$$t'/3 = \alpha \cdot t/3 = \alpha \cdot \nu \cdot n/3 \geq \alpha \cdot n/6 \geq O(\mu n).$$

Finally, by an averaging argument, we can hardwire the random coins of A to produce a non-randomized circuit with the same success probability. Thus, the circuit A contradicts the assumption that f is $2^{-t'/3}$-incomputable by Σ_{i+2}-circuits of size $n^{b'}$.

7.2 rt-PRGs with polynomial stretch

We now use the NW-generator to transform the 1-bit stretch rt-PRG into an rt-PRG with polynomial stretch.

7.2.1 Using the NW-generator

We review the construction of the NW-generator.

▶ **Definition 7.8** (Design). A collection $\Delta = (S_1, \ldots, S_n)$ of subsets of $[r]$ is an (r, ℓ, u)-design if

- For every $i \in [n]$, $|S_i| = \ell$.
- For every distinct $i, j \in [n]$, $|S_i \cap S_j| \leq u$.

▶ **Definition 7.9** (NW-generator). Let $\Delta = (S_1, \ldots, S_n)$ be an (r, ℓ, u)-design, and let $g : \{0,1\}^\ell \to \{0,1\}$ be a function. For $y \in \{0,1\}^r$, we define $x_i(y) = y|_{S_i}$ and $z_i(y) = g(x_i(y))$. Let,

$$\mathrm{NW}_g^\Delta(y) = z_1(y), \ldots, z_n(y).$$

Theorem 1.17 follows from the next theorem.

▶ **Theorem 7.10.** *Let Δ be an (r, ℓ, u)-design with $u = c \cdot \log n$. If $G(x) = (x, g(x))$ is an $(\frac{\epsilon}{20n}, \frac{\rho \cdot \epsilon}{30n})$-rt-PRG for size $n^{c+1} + n^b + O(n)$ circuits, then $NW_g^\Delta : \{0,1\}^r \to \{0,1\}^n$ is an (ϵ, ρ)-rt-PRG for size n^b circuits.*

We need the following notation for the proof.

▶ **Definition 7.11.** Given $x \in \{0,1\}^\ell$, $v \in \{0,1\}^{r-\ell}$ and $i \in [n]$ let $y^{(i)}(x, v)$ denote the r-bit string y obtained by "placing" the bits of x in the ℓ indices of y that are in S_i, and using v to fill the remaining $r - \ell$ positions.

of Theorem 7.10. In this proof g and Δ are fixed, and so, to avoid clutter we write NW instead of NW_g^Δ. Assume for contradiction that NW is not an (ϵ, ρ)-rt-PRG for size n^b circuits. That is, that there exists a circuit D of size n^b such that

$$\Pr[D(\mathrm{NW}(U_r)) = 1] \overset{rt}{\not\sim}_{(\epsilon,\rho)} \Pr[D(U_n) = 1].$$

▶ **Claim 7.12** ("Relative error hybrid argument"). *Consider a probability space consisting of independent random variables $Y \leftarrow U_r$ and $B_1, \ldots, B_n \leftarrow U_1$, and define*

$$H_i = z_1(Y), \ldots, z_i(Y), B_{i+1}, \ldots, B_n,$$

so that $H_0 = U_n$ and $H_n = NW(Y)$. There exists $0 \leq i < n$ such that

$$\Pr[D(H_i) = 1] \overset{rt}{\not\sim}_{(\epsilon/2n, \rho \cdot e^{-\epsilon}/2)} Pr[D(H_{i+1}) = 1]$$

Proof. Let $p_i = \Pr[D(H_i) = 1]$. We have that $p_0 \overset{rt}{\not\sim}_{(\epsilon,\rho)} p_n$ which indeed implies that there exists an i such that $p_i \overset{rt}{\not\sim}_{(\epsilon/2n,\rho\cdot e^{-\epsilon}/2)} p_{i+1}$. ◀

We have that

$$\Pr[D(z_1(Y),\ldots,z_i(Y),B_{i+1},\ldots,B_n)=1] \overset{rt}{\not\sim}_{(\epsilon/2n,\rho\cdot e^{-\epsilon}/2)}$$
$$\Pr[D(z_1(Y),\ldots,z_{i+1}(Y),B_{i+2},\ldots,B_n)=1$$

We can imagine that the experiment of choosing $Y \leftarrow U_r$ is performed by choosing independently $X \leftarrow U_\ell$ and $V \leftarrow U_r$ and setting $Y = y^{(i+1)}(X,V)$. We now apply Lemma 4.10 setting $R = (V, B_{i+2},\ldots,B_n)$ and $W = (X, B_{i+1})$. We conclude that there exists a fixing $(v, b_{i+2},\ldots,b_n)$ for R such that

$$\Pr[D(z_1(y^{(i+1)}(X,v)),\ldots,z_i(y^{(i+1)}(X,v)),B_{i+1},b_{i+2},\ldots,b_n)=1] \overset{rt}{\not\sim}_{(\epsilon/20n,\rho\cdot e^{-\epsilon}\cdot\epsilon/20n)}$$

$$\Pr[D(z_1(y^{(i+1)}(X,v)),\ldots,z_{i+1}(y^{(i+1)}(X,v)),b_{i+2},\ldots,b_n)=1].$$

Note that by definition, $z_{i+1}(y^{(i+1)}(X,v)) = g(X)$. Furthermore, note that as Δ is a (r,ℓ,u)-design, for $j \le i$, $z_j(y^{(i+1)}(X,v)$ depends only on u bits of X, and therefore, can be computed by a circuit $C_j(X)$ of size 2^u. We now define a circuit C as follows:

$$C(x,b) = D(C_1(x),\ldots,C_i(x),b,b_{i+2},\ldots,b_n).$$

Substituting in the expression above, we have that:

$$\Pr[C(X,g(X))=1] \overset{rt}{\not\sim}_{(\epsilon/20n,\rho\cdot e^{-\epsilon}\cdot\epsilon/20n)} \Pr[C(X,B_{i+1})=1]$$

Note that C is of size $n\cdot 2^u + n^b + O(n) = n^{c+1} + n^b + O(n)$ and that $\rho\cdot e^{-\epsilon}\cdot\epsilon/20n \ge \rho\cdot\epsilon/30n$. ◀

7.2.2 Putting it together: Proof of Theorem 1.17

We assume that E is hard for exponential size Σ_{i+3}-circuits. Let $e, b > 1$ be some constants. Nisan and Wigderson [29] showed that there exists a constant $c > 1$ such that for every sufficiently large n, there is an (r,ℓ,u) design with $n = r^e$ sets, that has $r = O(\ell^2)$ and $u = c\log n$. Note that $n = O(\ell^{2e})$ and so, by Theorem 7.2 there are polynomial time computable functions $g : \{0,1\}^\ell \to \{0,1\}$ and $G(x) = (x, g(x))$ such that G is a $(n^{-b'},\rho)$-rt-PRG for size $n^{b'}$ Σ_i-circuits, where b' is a constant that we choose later, and $\rho = 2^{-\Omega(\ell)} = 2^{-\Omega(\sqrt{r})}$. We can choose the constant b' to be sufficiently large so that by Theorem 7.10 we have that NW_g^Δ is an (ϵ',ρ')-rt-PRG for size n^b Σ_i-circuits, with $\epsilon' = n^{-b'}\cdot 20n \le n^{-b}$ and $\rho' = \rho\cdot 30n/n^{-b'} = \rho^{\Omega(1)} = 2^{-\Omega(\sqrt{r})}$ for sufficiently large n. This gives a re-PRG with the same parameters.

8 A construction of re-nb-PRGs

In this section we construct rt-nb-PRGs which imply re-nb-PRGs.

8.1 rt-PRGs for Σ_1-circuits are rt-nb-PRGs

We first show that sufficiently strong rt-PRGs for Σ_1-circuits are rt-nb-PRGs. More specifically that (ϵ, ρ)-rt-PRGs with $\rho = O(2^{-\ell} \cdot \rho' \cdot \epsilon)$ for Σ_1-circuits are $(\ell, O(\epsilon), \rho')$-rt-nb-PRGs (for standard circuits).

▶ **Lemma 8.1.** *There exists a constant $c > 1$ such that for every constant $b > 1$, and for every sufficiently large n, if $G : \{0,1\}^r \to \{0,1\}^n$ is an (ϵ, ρ)-rt-PRG for Σ_1-circuits of size n^{bc} with $\rho \leq 2^{-(\ell+t)}$ then G is a $(\ell, 4 \cdot \epsilon, \rho')$-rt-nb-PRG for circuits of size n^b for $\rho' = O(2^{-t}/\epsilon)$.*

Proof. Let $C : \{0,1\}^n \to \{0,1\}^\ell$ be a size n^b circuit. We consider the circuit $A : \{0,1\}^\ell \to \{0,1\}$ that on input $z \in \{0,1\}^\ell$ computes a 1/10-relative approximation to the quantity $\Pr[C(U_n) = z]$ and accepts if and only if the approximation is smaller than 2ρ. By Theorem 4.2, A can be implemented by a Σ_1-circuit of size $\text{poly}(n^b)$. We use A also to denote the set of inputs accepted by A. Note that for every $z \in A$, $\Pr[C(U_n) = z] < 4\rho$. It follows that $\Pr[C(U_n) \in A] \leq 2^\ell \cdot 4\rho = 2^{-(t-2)}$. By the pseudorandomness of G, this implies that $\Pr[C(G(U_r)) \in A] \leq e^\epsilon 2^{-(t-2)}$.

Let $H = \{z : \Pr[C(U_n) = z] \geq 4\rho\}$. We have that $H \cap A = \emptyset$. For every $z \in \{0,1\}^\ell$, we can consider the circuit $T_z(x)$ which accepts iff $C(x) = z$. This circuit is fooled by G. For $z \notin A$, $\Pr[C(U_n) = z] \geq \rho$, and we have that $\Pr[C(U_n) = z] \overset{rt}{\sim}_{(\epsilon,0)} \Pr[C(G(U_r)) = z]$. This in turn implies that for every T such that $T \cap A = \emptyset$, $\Pr[C(U_n) \in T] \overset{rt}{\sim}_{(\epsilon,0)} \Pr[C(G(U_r)) \in T]$. We will show that:

▶ **Claim 8.2.** *For every $D \subseteq \{0,1\}^\ell$, $\Pr[C(U_n) \in D] \overset{re}{\sim}_{(\epsilon,\delta)} \Pr[C(G(U_r)) \in D]$ for $\delta = 2^{-(t-2)}$.*

Proof. We have that:

$$\begin{aligned}\Pr[C(U_n) \in D] &= \Pr[C(U_n) \in D \setminus H] + \Pr[C(U_n) \in D \cap H] \\ &\leq 4\rho \cdot 2^\ell + e^\epsilon \cdot \Pr[C(G(U_r)) \in D \cap H] \\ &\leq 2^{-(t-2)} + e^\epsilon \cdot \Pr[C(G(U_r)) \in D \cap H]\end{aligned}$$

We also have that:

$$\begin{aligned}\Pr[C(U_n) \in D] &\geq \Pr[C(U_n) \in D \setminus A] \\ &\geq e^{-\epsilon} \cdot \Pr[C(G(U_r)) \in D \setminus A] \\ &= e^{-\epsilon} \cdot (\Pr[C(G(U_r)) \in D] - \Pr[C(G(U_r)) \in D \cap A]) \\ &\geq e^{-\epsilon} \cdot (\Pr[C(G(U_r)) \in D] - e^\epsilon \cdot 2^{-(t-2)}) \\ &= e^{-\epsilon} \cdot \Pr[C(G(U_r)) \in D] - 2^{-(t-2)}\end{aligned}$$

◀

The lemma now follows because by Lemma 4.5 for $\epsilon \leq \frac{1}{2}$, $p_1 \overset{re}{\sim}_{(\epsilon,\delta)} p_2 \Rightarrow p_1 \overset{rt}{\sim}_{(4\epsilon, 4\delta/\epsilon)} p_2$.

◀

Using the rt-PRG of Theorem 1.17 we obtain the following rt-nb-PRG.

▶ **Theorem 8.3.** *Let $b, e > 1$ be constants, and $\ell = \ell(n) \leq n$, $\rho = \rho(n)$ be functions. If E is hard for exponential size Σ_4-circuits, then there is a polynomial time computable $G : \{0,1\}^r \to \{0,1\}^n$, such that for every sufficiently large n, G is an (ℓ, n^{-b}, ρ)-rt-nb-PRG for circuits of size n^b, with $r = O((\ell + \log(1/\rho))^2)$.*

This is disappointing as previous work on (non-relative) nb-PRGs achieves a better dependence on ℓ in the form of $r = O(\ell)$. We would like to also achieve a linear dependence of r on ℓ.

8.2 An rt-nb-PRG with seed length $r = \ell + O(\log(1/\rho))^2$

We will use the approach of Theorem 1.18 to achieve a construction with shorter seed length. Specifically, we design a poly(n) time randomized procedure P that produces circuit that is with high probability an rt-PRG for Σ_1-circuits that has excellent seed length. We then show that checking whether a given circuit is an rt-PRG for Σ_1-circuits can be done in the polynomial time hierarchy. This means that our rt-PRG G' of the earlier section can be used to produce a circuit h that with high probability is an rt-PRG for Σ_1-circuits. This in turn implies that it is an rt-nb-PRG for standard circuits. Our final rt-PRGs takes two seeds, x_1, x_2. It first constructs h by applying $P(G'(x_1))$ and then outputs $h(x_2)$.

8.2.1 A random hash function is an rt-PRG

We use the following standard construction of t-wise independent hash functions (that is based on degree $t-1$ polynomials).

▶ **Theorem 8.4** (t-wise independent hash functions)**.** *For every n, m, t there is a family $\mathcal{H}^t_{n,m}$ of at most $2^{d=t\cdot\max(n,m)}$ functions from n bits to m bits, such that for every distinct $x_1, \ldots, x_t \in \{0,1\}^n$, the random variables $h(x_1), \ldots, h(x_t)$ defined over the experiment $h \leftarrow \mathcal{H}^t_{n,m}$ are uniformly distributed and t-wise independent. Furthermore, there is a polynomial time algorithm that given the d bit description s of a hash function $h_s \in \mathcal{H}^t_{n,m}$, and an input $x \in \{0,1\}^n$, computes $h_s(x)$.*

A standard probabilistic argument shows that for any class $\mathcal{C}$ with 2^k functions, a random function $G : \{0,1\}^r \to \{0,1\}^n$ is w.h.p. a PRG for $\mathcal{C}$ with $r \approx \log k$. In the theorem below, we repeat this argument and show that it also applies for rt-PRGs and achieves an excellent dependence on ρ.

▶ **Theorem 8.5.** *Let $\mathcal{C}$ be a family of at most 2^k boolean functions on n bits. Let $t = 2(k+3) + 2n$ and $r = \log k + \log n + 2\log(1/\epsilon) + \log(1/\rho) + c$ for a sufficiently large universal constant c. With probability at least $1 - 2^{-n}$ over $h \leftarrow \mathcal{H}^t_{r,n}$, we obtain a function $h : \{0,1\}^r \to \{0,1\}^n$ that is an (ϵ, ρ)-rt-PRG for $\mathcal{C}$.*

Proof. Let $C : \{0,1\}^n \to \{0,1\}$ be a function in $\mathcal{C}$, and let $\mu = \Pr[C(U_n) = 1]$. Let X_C be the random variable that counts the number of $s \in \{0,1\}^r$ such that $C(h(s)) = 1$. The random variable X_C is a sum of 2^r, t-wise independent variables. We have that $E(X_C) = 2^r \cdot \mu$. By the t-wise independent "Chernoff style" bound of Bellare and Rompel [8] we have that for even t,

$$\Pr[|X_C - E(X_C)| \geq A] \leq 8 \cdot \left(\frac{t \cdot E(X_C) + t^2}{A^2}\right)^{t/2}$$

Let $A = \frac{1}{3} \cdot \epsilon \cdot 2^r \cdot \max(\mu, \rho)$. This choice is made so that $\frac{X_C}{2^r} \overset{rt}{\not\sim}_{(\epsilon,\rho)} \mu$ implies that $|X_C - E(X_C)| \geq A$. By our choice of parameters it follows that:

$$\frac{t \cdot E(X_C) + t^2}{A^2} \leq \frac{t \cdot \mu \cdot 2^r + t^2}{\frac{1}{9} \cdot \epsilon^2 \cdot 2^{2r} \cdot \max(\mu, \rho)^2} \leq \frac{t}{\frac{1}{9} \cdot \epsilon^2 \cdot 2^r \cdot \rho} + \left(\frac{t}{\frac{1}{3} \cdot \epsilon \cdot 2^r \cdot \rho}\right)^2 \leq \frac{1}{2}$$

where the last inequality follows for a sufficiently large constant c in the definition of r, and

using our choice of parameters. It follows that

$$\begin{aligned}\Pr[\frac{X_C}{2^r} \overset{rt}{\not\sim}_{(\epsilon,\rho)} \mu] &\leq 8 \cdot \left(\frac{t \cdot E(X_C) + t^2}{A^2}\right)^{t/2} \\ &\leq 8 \cdot \left(\frac{1}{2}\right)^{t/2} \\ &\leq 2^{-n} \cdot 2^{-k}\end{aligned}$$

where the last inequality follows by our choice of t. By a union bound over all 2^k functions C in $\mathcal{C}$ we have that with probability $1 - 2^{-n}$ we obtain an rt-PRG. ◀

▶ **Corollary 8.6.** *For every $b > 1$, and for every sufficiently large n, and every $\epsilon, \rho \geq 2^{-n}$, there is a randomized Turing Machine P running in time poly(n^b) that with probability $1-2^{-n}$ produces a circuit $h : \{0,1\}^r \to \{0,1\}^n$ that is an (ϵ, ρ)-rt-PRG for Σ_1-circuits of size n^b, with $r = O(b \log n) + 2\log(1/\epsilon) + \log(1/\rho)$.*

8.2.2 The complexity of checking if a given circuit is an rt-PRG

We consider the problem of checking whether a given circuit is an rt-PRG. We would like to show that this problem is in the polynomial time hierarchy. The following formulation as a promise problem makes this possible, and will suffice for our needs.

▶ **Definition 8.7.** Let $\mathsf{DC}^{i,s,s'}_{\epsilon,\rho}$ denote the following promise problem:
Input: a circuit $G : \{0,1\}^r \to \{0,1\}^n$ of size s.
Yes instances: G is not an (ϵ, ρ)-rt-PRG for Σ_i-circuits of size s'.
No instances: G is an $(\epsilon/2, \rho \cdot (1-\epsilon))$-rt-PRG for Σ_i-circuits of size s'.

▶ **Theorem 8.8.** *For every $i \geq 0$, $0 < \epsilon, \rho \leq 1$, and $r \leq n \leq s' \leq s$ there is a nondeterministic Σ_{i+2}-circuit of size poly$(r, n, s, 1/\epsilon)$ which solves $\mathsf{DC}^{i,s,s'}_{\epsilon,\rho}$.*

Proof. We consider the following nondeterministic Σ_{i+1}-circuit A: when given the circuit G as input, the circuit A guesses a Σ_i-circuit $C : \{0,1\}^n \to \{0,1\}$ of size s'. Let $\eta = \epsilon/10$. Using Theorem 4.2, A computes η-relative approximations p'_1, p'_2 of the quantities of $p_1 = \Pr[C(U_n) = 1]$ and $p_2 = \Pr[C(G(U_n) = 1]$. A then applies the test $T(p'_1, p'_2)$ of Lemma 4.9 and outputs its outcome. By Lemma 4.9:

- $p_1 \overset{rt}{\not\sim}_{(\epsilon,\rho)} p_2 \Rightarrow T(p'_1, p'_2)$ accepts.
- $T(p'_1, p'_2)$ accepts $\Rightarrow p_1 \overset{rt}{\not\sim}_{(\epsilon-4\eta,\rho\cdot e^{-2\eta})} p_2$.

The theorem follows by our choice of η. ◀

The key is that the size of the circuit above does not depend on ρ, and note that if the circuit rejects G, then G is an (ϵ, ρ)-rt-PRG.

8.3 rt-nb-PRGs with small seed length

The following Theorem implies Theorem 1.21.

▶ **Theorem 8.9** (rt-nb-PRG with seed length $1 \cdot \ell + O(\log(1/\rho))^2$)**.** *Let $b > 1$ and $\alpha > 0$ be constants and $\ell = \ell(n) \leq n$, $\rho = \rho(n) \leq 2^{-n^\alpha}$. Assume that E is hard for exponential size Σ_6-circuits. Let G be the function constructed in Figure 4, with the parameters chosen there. Then for every sufficiently large n, there is a polynomial time computable (ℓ, n^{-b}, ρ)-rt-nb-PRG $G : \{0,1\}^r \to \{0,1\}^n$ for size n^b circuits, with $r = \ell + O(\log(1/\rho))^2$.*

Goal: Construct a poly(n)-time computable (ℓ, n^{-b}, ρ)-rt-nb-PRG, $G : \{0,1\}^r \to \{0,1\}^n$ for circuits $C : \{0,1\}^n \to \{0,1\}^\ell$ of size n^b.

Assumption: E is hard for exponential size Σ_6-circuits.

Parameters:

- ℓ, b, n - We are shooting to fool circuits $C : \{0,1\}^n \to \{0,1\}^\ell$ of size n^b.
- We require that $\rho \leq 2^{-n^\alpha}$ for some constant $\alpha > 0$. (This is done to simplify the presentation).

Ingredients:

- We make use of the Turing Machine P of Corollary 8.6. Specifically, let $b' = a \cdot b$ for a sufficiently large constant $a > 1$ to be chosen later. By Corollary 8.6 there is a randomized Turing Machine P running in time poly$(n^{b'})$ which produces a circuit $h : \{0,1\}^r \to \{0,1\}^n$, that with probability $1 - 2^{-n}$ is an $(n^{-b'}, \rho' = 2^{-\ell} \cdot \rho \cdot n^{-3b'})$-rt-PRG for Σ_1-circuits of size $n^{b'}$, and $r_1 = \ell + O(b' \cdot \log n) + \log(1/\rho)$.
- We also make use of the rt-PRG of Theorem 1.17. Specifically, let $b'' = a \cdot b'$ and recall that a sufficiently large constant $a > 1$ will be chosen later. By Theorem 1.17 the hardness assumption that E is hard for exponential size Σ_6-circuits implies that there is a poly$(n^{b''})$ computable $(\frac{1}{2}, \rho')$-rt-PRG $G' : \{0,1\}^{r_2} \to \{0,1\}^{n^{b''}}$ for size $n^{b''}$ Σ_3-circuits, with $r_2 = O(1/\rho))^2$. (Here we use the fact that $\rho \leq 2^{-n^\alpha}$ so that $\log(1/\rho) \geq n^\alpha$).

The rt-nb-PRG:

- Let $r = r_1 + r_2 = \ell + O(b' \cdot \log n) + O(\log(1/\rho))^2 = \ell + O(\log(1/\rho))^2$. Given $x \in \{0,1\}^r$ interpret it as $(x_1, x_2) \in \{0,1\}^{r_1} \times \{0,1\}^{r_2}$.
- Run the procedure P using the string $G'(x_2)$ as random coins. (Note that we can choose the constant a to be sufficiently large so that $n^{b''} = n^{ab'}$ is larger than the number of coins required by P). The procedure P produces a circuit $h : \{0,1\}^{r_1} \to \{0,1\}^n$.
- $G(x)$ outputs $h(x_1)$.

Figure 4 An rt-nb-PRG with seed length $\approx 1 \cdot \ell$.

Proof of Theorem 8.9. We first argue, that when G applies P to obtain a circuit h, then w.h.p. it obtains an rt-PRG. Specifically,

▶ **Claim 8.10.** *With probability $1 - 2\rho'$ over $x_2 \leftarrow U_{r_2}$, the circuit $h : \{0,1\}^{r_1} \to \{0,1\}^n$ obtained by $P(G'(x_2))$ is a $(n^{-b'}, \rho')$-rt-PRG for size $n^{b'}$ Σ_1-circuits.*

Proof. (of claim) Let $s' = n^{b'}$ and $s = \text{poly}(n^{b'})$ be a bound on the size of h. By Theorem 8.3 we have that the promise problem $\mathsf{DC}^{1,s,s'}_{n^{-2b'},\rho'}$ is solved by a nondeterministic Σ_3-circuit T of size poly$(n^{b'})$. Recall that if T rejects a given circuit, then this circuit is a $(n^{-2b'}, \rho')$-rt-PRG for Σ_1-circuits of size $s' = n^{b'}$. Let $D(z) = T(P(z))$ and note that D can be implemented by a Σ_3-circuit of size poly$(n^{b'})$. The parameters of the generator G' were chosen so that it fools D. More specifically, by choosing a to be sufficiently large we have that the size of D is smaller than $n^{b''} = n^{a \cdot b'}$. By the guarantee on P, we know that the probability that D accepts a uniform input is at most 2^{-n}. As G' is a $(\frac{1}{2}, \rho')$-PRG it follows that the probability that D accepts $G'(U_{r_2})$ is at most $e^{\frac{1}{2}} \cdot \rho' \leq 2\rho'$. The claim follows. ◀

We have that with probability $1 - 2\rho'$ over the choice of x_2, G output $h(U_{r_1})$ for h that is a $(n^{-b'}, \rho')$-rt-PRG for Σ_1-circuits of size $n^{b'}$. This implies that G is a $(O(n^{-b'}), O(\rho'/n^{-b'}))$-rt-PRG for size $n^{b'}$ Σ_1-circuits. A trivial (albeit somewhat wasteful) way to see this is to use Lemma 4.5 to transform the guarantee on $\overset{rt}{\sim}_{(,)}$ to $\overset{re}{\sim}_{(,)}$ which makes the calculation straightforward, and then transform back. This is why we have $n^{-b'}$ in the denominator.

Finally, we choose a to be sufficiently large so that by Lemma 8.1 an rt-PRG against Σ_1-circuits of size $n^{b'} = n^{ab}$ is a rt-nb-PRG for circuits of size n^b. Applying the lemma, we get that G is an $(O(n^{-b'}), O(\rho'/n^{-2b'}))$-rt-nb-PRG for circuits of size n^b. The Theorem follows as by our choice of parameters $O(n^{-b'})$ can be made smaller than n^{-b} and $O(\rho'/n^{-2b'})$ is smaller than ρ. ◀

9 Open Problems

Theorem 1.8 is the first hardness versus randomness tradeoff that is applicable to randomized algorithm solving NP complete problem. It is interesting to find more instances where this approach can be used to efficiently derandomize algorithms for other NP complete problems. Is it possible to give hardness versus randomness tradeoffs for general randomized algorithms (that are not necessarily OPP)?

The dependence of the seed length on the parameter δ in Theorem 1.17 is additive in $\log(1/\delta)^2$ can this be reduced to $\log(1/\delta)$? As explained in the introduction, this will give improvements in applications of the theorem.

Can we find more applications of re-PRGS and re-nb-PRGs? It will be especially interesting to find cryptographic applications in computational settings as discussed in Section 1.14.

Acknowledgements. This work was done in part while the last three authors were visiting Simons Institute for the Theory of Computing. We are grateful to anonymous referees for helpful comments.

References

1 Noga Alon, Raphael Yuster, and Uri Zwick. Color-coding. *J. ACM*, 42(4):844–856, 1995.
2 Benny Applebaum, Sergei Artemenko, Ronen Shaltiel, and Guang Yang. Incompressible functions, relative-error extractors, and the power of nondeterministic reductions (extended abstract). In *Conference on Computational Complexity*, volume 33 of *LIPIcs*, pages 582–600. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2015.
3 Sergei Artemenko and Ronen Shaltiel. Lower bounds on the query complexity of non-uniform and adaptive reductions showing hardness amplification. *Computational Complexity*, 23(1):43–83, 2014.
4 Sergei Artemenko and Ronen Shaltiel. Pseudorandom generators with optimal seed length for non-boolean poly-size circuits. In *STOC*, pages 99–108. ACM, 2014.
5 László Babai, Lance Fortnow, Noam Nisan, and Avi Wigderson. BPP has subexponential time simulations unless EXPTIME has publishable proofs. *Computational Complexity*, 3:307–318, 1993.
6 Boaz Barak, Shien Jin Ong, and Salil P. Vadhan. Derandomization in cryptography. *SIAM J. Comput.*, 37(2):380–400, 2007.
7 Mihir Bellare, Oded Goldreich, and Erez Petrank. Uniform generation of NP-witnesses using an NP-oracle. *Inf. Comput.*, 163(2):510–526, 2000.
8 Mihir Bellare and John Rompel. Randomness-efficient oblivious sampling. In *FOCS*, pages 276–287. IEEE Computer Society, 1994.
9 Manuel Blum and Silvio Micali. How to generate cryptographically strong sequences of pseudo-random bits. *SIAM J. Comput.*, 13(4):850–864, 1984.
10 Andrew Drucker. Nondeterministic direct product reductions and the success probability of SAT solvers. In *FOCS*, pages 736–745. IEEE Computer Society, 2013.

11 Bella Dubrov and Yuval Ishai. On the randomness complexity of efficient sampling. In *STOC*, pages 711–720. ACM, 2006.

12 Uriel Feige and Carsten Lund. On the hardness of computing the permanent of random matrices. *Computational Complexity*, 6(2):101–132, 1997.

13 Oded Goldreich. A sample of samplers: A computational perspective on sampling. In *Studies in Complexity and Cryptography*, volume 6650 of *Lecture Notes in Computer Science*, pages 302–332. Springer, 2011.

14 Oded Goldreich and Leonid A. Levin. A hard-core predicate for all one-way functions. In *STOC*, pages 25–32. ACM, 1989.

15 Oded Goldreich and Avi Wigderson. Derandomization that is rarely wrong from short advice that is typically good. In *RANDOM*, volume 2483 of *Lecture Notes in Computer Science*, pages 209–223. Springer, 2002.

16 Shafi Goldwasser and Michael Sipser. Private coins versus public coins in interactive proof systems. In *STOC*, pages 59–68. ACM, 1986.

17 Venkatesan Guruswami, Christopher Umans, and Salil P. Vadhan. Unbalanced expanders and randomness extractors from parvaresh–vardy codes. *J. ACM*, 56(4), 2009.

18 Dan Gutfreund and Guy N. Rothblum. The complexity of local list decoding. In *APPROX-RANDOM*, volume 5171 of *Lecture Notes in Computer Science*, pages 455–468. Springer, 2008.

19 Dan Gutfreund, Ronen Shaltiel, and Amnon Ta-Shma. Uniform hardness versus randomness tradeoffs for arthur-merlin games. *Computational Complexity*, 12(3-4):85–130, 2003.

20 Yenjo Han, Lane A. Hemaspaandra, and Thomas Thierauf. Threshold computation and cryptographic security. *SIAM J. Comput.*, 26(1):59–78, 1997.

21 Russell Impagliazzo. Hard-core distributions for somewhat hard problems. In *FOCS*, pages 538–545. IEEE Computer Society, 1995.

22 Russell Impagliazzo, William Matthews, and Ramamohan Paturi. A satisfiability algorithm for ac^0. In *SODA*, pages 961–972. SIAM, 2012.

23 Russell Impagliazzo, Ronen Shaltiel, and Avi Wigderson. Near-optimal conversion of hardness into pseudo-randomness. In *FOCS*, pages 181–190. IEEE Computer Society, 1999.

24 Russell Impagliazzo, Ronen Shaltiel, and Avi Wigderson. Reducing the seed length in the nisan-wigderson generator. *Combinatorica*, 26(6):647–681, 2006.

25 Russell Impagliazzo and Avi Wigderson. $P = BPP$ if E requires exponential circuits: Derandomizing the XOR lemma. In *STOC*, pages 220–229. ACM, 1997.

26 Mark Jerrum, Leslie G. Valiant, and Vijay V. Vazirani. Random generation of combinatorial structures from a uniform distribution. *Theor. Comput. Sci.*, 43:169–188, 1986.

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

28 Peter Bro Miltersen and N. V. Vinodchandran. Derandomizing arthur-merlin games using hitting sets. *Computational Complexity*, 14(3):256–279, 2005.

29 Noam Nisan and Avi Wigderson. Hardness vs randomness. *J. Comput. Syst. Sci.*, 49(2):149–167, 1994.

30 Ramamohan Paturi and Pavel Pudlák. On the complexity of circuit satisfiability. In *STOC*, pages 241–250. ACM, 2010.

31 Ramamohan Paturi, Pavel Pudlák, Michael E. Saks, and Francis Zane. An improved exponential-time algorithm for k-SAT. *J. ACM*, 52(3):337–364, 2005.

32 Ramamohan Paturi, Pavel Pudlák, and Francis Zane. Satisfiability coding lemma. *Chicago J. Theor. Comput. Sci.*, 1999, 1999.

33 Rahul Santhanam. Fighting perebor: New and improved algorithms for formula and QBF satisfiability. In *FOCS*, pages 183–192. IEEE Computer Society, 2010.

34 Uwe Schöning. A probabilistic algorithm for k-SAT and constraint satisfaction problems. In *FOCS*, pages 410–414. IEEE Computer Society, 1999.

35 Ronen Shaltiel and Christopher Umans. Simple extractors for all min-entropies and a new pseudorandom generator. *J. ACM*, 52(2):172–216, 2005.

36 Ronen Shaltiel and Christopher Umans. Pseudorandomness for approximate counting and sampling. *Computational Complexity*, 15(4):298–341, 2006.

37 Ronen Shaltiel and Christopher Umans. Low-end uniform hardness versus randomness tradeoffs for AM. *SIAM J. Comput.*, 39(3):1006–1037, 2009.

38 Ronen Shaltiel and Emanuele Viola. Hardness amplification proofs require majority. *SIAM J. Comput.*, 39(7):3122–3154, 2010.

39 Michael Sipser. A complexity theoretic approach to randomness. In *STOC*, pages 330–335. ACM, 1983.

40 Larry J. Stockmeyer. The complexity of approximate counting (preliminary version). In *STOC*, pages 118–126. ACM, 1983.

41 Madhu Sudan, Luca Trevisan, and Salil P. Vadhan. Pseudorandom generators without the XOR lemma. *J. Comput. Syst. Sci.*, 62(2):236–266, 2001.

42 Luca Trevisan. Extractors and pseudorandom generators. *J. ACM*, 48(4):860–879, 2001.

43 Luca Trevisan and Salil P. Vadhan. Extracting randomness from samplable distributions. In *FOCS*, pages 32–42. IEEE Computer Society, 2000.

44 Christopher Umans. Pseudo-random generators for all hardnesses. *J. Comput. Syst. Sci.*, 67(2):419–440, 2003.

45 Andrew Chi-Chih Yao. Theory and applications of trapdoor functions (extended abstract). In *FOCS*, pages 80–91. IEEE Computer Society, 1982.