Electronic Colloquium on Computational Complexity, Report No. 20 (2004)

The Complexity of Constructing Pseudorandom Generators from Hard Functions*
(Preliminary Version)

Emanuele Viola†
Division of Engineering and Applied Sciences
Harvard University
Cambridge, MA 02138
viola@eecs.harvard.edu

Abstract

We study the complexity of building pseudorandom generators (PRGs) from hard functions.

We show that, starting from a function $f : \{0,1\}^l \to \{0,1\}$ that is mildly *hard on average, i.e. every circuit of size $2^{\Omega(l)}$ fails to compute f on at least a $1/\mathrm{poly}(l)$ fraction of inputs, we can build a PRG $: \{0,1\}^{O(\log n)} \to \{0,1\}^n$ computable in $ATIME(O(1), \log n)$ = alternating time $O(\log n)$ with $O(1)$ alternations. Such a PRG implies $BP \cdot AC_0 = AC_0$ under $DLOGTIME$-uniformity.*

On the negative side, we prove a tight time-alternations tradeoff for black-box PRG constructions that are based on worst-case *hard functions. We also prove a tight time-alternations tradeoff for black-box* worst-case *hardness amplification, which is the problem of producing an average-case hard function starting from a worst-case hard one. In particular, we obtain that there is no black-box worst-case hardness amplification within the polynomial time hierarchy. These lower bounds are obtained by showing that constant depth circuits cannot compute extractors and list-decodable codes.*

1. Introduction

The fascinating connection between hardness and randomness was first noticed by Yao [43] and by Blum and Micali [7]. A decade later, Nisan and Wigderson [28] showed how to build pseudorandom generators (PRGs) from *strong average-case* hardness assumptions, namely the existence of functions that are hard on average for circuits. Since then, much research [28, 4, 14, 18] was devoted to relaxing this assumption to a *worst-case* one, namely the existence of functions of high circuit complexity. This research culminated in [18] where it is shown how to amplify the worst-case hardness of a function to the average-case hardness required in [28]. More direct proofs and improved results were obtained in [36, 17, 33, 40]. Finally, in [40] the 'right' trade-off between worst-case hardness and randomness was achieved for the full range of parameters.

*A preliminary version of this paper was published in *Proceedings of the 18th Annual Conference on Compuational Complexity*, IEEE, Aarhus, 2003, pp. 53-69.

†Research supported in part by NSF grant CCR-0133096.

ISSN 1433-8092

The problem we study: In this paper we address the following problem: What is the complexity of building a PRG from a hard function? There are at least two reasons for studying this problem. First, we want to understand the computational relationship between two fundamental quantities in theoretical computer science: hardness and randomness. Second, PRGs are a basic tool whose variety of applications justifies the quest for more and more efficient constructions.

Derandomization: An important application that demands more and more efficient PRGs is the *high-end derandomization* of a complexity class C, that is proving $BP \cdot C = C$. For such application we need PRGs with logarithmic seed length, which can be built starting from a function having exponential circuit complexity. For example, Impagliazzo and Wigderson [18] show that $BP \cdot P = P$ if $E := TIME(2^{O(n)})$ requires circuits of size $2^{\Omega(n)}$. This derandomization works as follows. We run the algorithm we want to derandomize using all the possible outputs of the PRG in place of true random bits. Then we decide according to majority vote. Since the seed length is logarithmic, this process is efficient, i.e. only gives a polynomial slow-down.

It is then clear that if we aim to derandomize a probabilistic complexity class $BP \cdot C$ using a PRG, then the PRG must be computable in C. Therefore, the lower the complexity class we want to derandomize, the more efficient the PRG must be. For example, already to derandomize $BP \cdot L$ (where $L := SPACE(\log n)$), one needs a more efficient PRG construction than the one given in [28] and used in [18] to derandomize $BP \cdot P$. This problem is solved by Klivans and van Melkebeek [22] who obtain $BP \cdot L = L$ under the assumption that deterministic linear space requires exponential size circuits.

In this paper we study more efficient PRG constructions that could be used to derandomize probabilistic classes below $BP \cdot L$. In particular, we aim to prove $BP \cdot AC_0 = AC_0$, where AC_0 denotes the class of constant depth circuits.

Note that the high-end derandomization $BP \cdot AC_0 = AC_0$ (or even $BP \cdot AC_0 \subseteq P$) is not known to hold unconditionally. In fact, the most efficient derandomization of $BP \cdot AC_0$ is the one obtained by Nisan [27] that runs in quasipolynomial time. Such derandomization is based on lower bounds for AC_0 circuits of size $2^{n^{\Omega(1)}}$, such as the ones shown by Håstad [13]. On the other hand, the high-end derandomization $BP \cdot AC_0 = AC_0$ (or even $BP \cdot AC_0 \subseteq P$) through a PRG requires lower bounds for AC_0 circuits of size $2^{\Omega(n)}$. But it is consistent with the current state of knowledge that every function in E has circuits of size $2^{o(n)}$ and depth 3 (cf., [16]).

In this paper, derandomization always means high-end derandomization.

PRGs in AC_0: The main technical question addressed in this paper is: Starting from a hard function, can you build a PRG in AC_0? To make sense of this question, one has to specify what is the uniformity condition for the circuits. This is because, under P-uniformity, the answer is clearly 'yes'. Indeed, since it is known how to build a PRG in P from a hard function, we can just hardwire all the outputs of the PRG in the AC_0 circuit[1], and then have the AC_0 circuit compute the PRG via a simple table look-up. *But the circuit is not REALLY computing the PRG! All the work is done by the Turing machine describing the circuit!*

This phenomenon, of a uniformity condition that hides the real power of circuits, is well-known in circuit complexity. There is a consensus that the 'right' uniformity condition for AC_0 is $DLOGTIME$-uniformity. Informally, a family of circuits is $DLOGTIME$-uniform if given indices to two gates one can decide their type and whether they are connected in linear time in the length of the indices (which is logarithmic time in the size of the circuit). The evidence that $DLOGTIME$-uniformity is the right uniformity condition for AC_0 comes from the fact that $DLOGTIME$-uniform AC_0 has several different and elegant characterizations [6, 41]. In particular, $DLOGTIME$-uniform AC_0 is equivalent to $ATIME(O(1), \log n)$: Alternating time $O(\log n)$ with $O(1)$ alternations. This is the logarithmic time hierarchy introduced by Sipser [35], a class within L.

[1] For this discussion we assume that the PRG has logarithmic seed length, so that it has only polynomially many outputs.

Table 1. A comparison of our main results with previous ones.

Hardness Amplification for functions : $\{0,1\}^l \to \{0,1\}$		**PRG construction from strong hardness** for PRG : $\{0,1\}^{O(\log n)} \to \{0,1\}^n$	**Derandomization**
Previous Results			
Worst-Case Hard $\Downarrow$ Strongly Hard		Complexity of PRG	Worst-Case Hardness Assumption $\Downarrow$
$TIME(2^{O(l)})$ [4, 14, 18] $SPACE(O(l))$ [22]		$TIME(n^{O(1)})$ [28] $SPACE(O(\log n))$ [22]	$BP \cdot P = P$ [18] $BP \cdot L = L$ [22]
Our Results			
Worst-Case Hard $\Downarrow$ Mildly Hard	Mildly Hard $\Downarrow$ Strongly Hard	Complexity of PRG	Mild Average-Case Hardness Assumption $\Downarrow$
Impossible in $ATIME(O(1), 2^{o(l)})$ if black-box [Corollary 7.5]	$ATIME(O(1), l)$ [Theorem 4.2]	$ATIME(O(1), \log n)$ [Theorem 4.1]	$BP \cdot ATIME(O(1), \log n)$ $\|$ $ATIME(O(1), \log n)$ [Theorems 4.3, 4.7]

Note that a PRG computable in $ATIME(O(1), \log n)$ allows us to derandomize $BP \cdot AC_0$ under $DLOGTIME$-uniformity. Previous to this paper the most uniform derandomization of $BP \cdot AC_0$ was under L-uniformity; that is, circuit families described by Turing machines running in logarithmic space. Such derandomization may be obtained using the techniques in [22, 21].

Our results: Our main results are summarized and compared to previous ones in Table 1.

On the positive side, we show that we can compute a PRG : $\{0,1\}^{O(\log n)} \to \{0,1\}^n$ in $ATIME(O(1), \log n)$ from a *mild average-case* hardness assumption, i.e. the existence of a function $f : \{0,1\}^l \to \{0,1\}$ such that for every circuit C of size $2^{\Omega(l)}$ we have $\Pr_x[C(x) \neq f(x)] \geq 1/\mathrm{poly}(l)$. The main new technical tool to achieve this is a construction of combinatorial designs that is computable in $ATIME(O(1), \log n)$. We also show that it is possible to amplify mild average-case hardness up to strong average-case hardness within $ATIME(O(1), l)$, where l is the input size of the mild average-case hard fuction.

On the negative side, we show a lower bound for PRG constructions from worst-case hard functions: We prove that there is no *black-box* PRG construction from worst-case hard functions that is computable in alternating time with $O(1)$ alternations, even if we allow time $n^{o(1)}$ where n is the output length of the PRG. Note that most known PRG constructions are black-box [18, 22, 17, 36, 33, 40]. We also show that there is no black-box worst-case to mild average-case hardness amplification computable in alternating time with $O(1)$ alternations, even if we allow time $2^{o(l)}$ where l is the input size of the worst-case hard function. Again, note that most known approaches [4, 36] are black-box. Our lower bound for hardness amplification implies that worst-case hardness amplification within the polynomial time hierarchy cannot be black-box.

Since [18], PRG constructions from worst-case hard functions have been simplified and strengthened [17, 36, 33, 40]. In particular, the latest constructions do not fall in the twofold paradigm of 'hardness amplification + Nisan-Wigderson PRG', but directly transform worst-case hardness into randomness. However, our results

suggest that the process of transforming worst-case hardness into randomness *is* twofold: Black-box worst-case hardness amplification is harder than black-box PRG constructions from mild average-case hardness.

We also study classes slightly larger than $BP \cdot AC_0$, such as $BP \cdot TC_0$, whose derandomization can be based on a worst-case hardness assumption. In addition, using results by Agrawal [1], we show that our derandomization results can be based on the weaker hardness assumption that there exists a function that is hard for constant depth circuits (whereas the discussion above refers to hardness against general circuits).

Our techniques: We now sketch the ideas behind our lower bounds. We prove them for constant depth circuits, the nonuniform analogue of alternating time with $O(1)$ alternations. Our lower bound for black-box PRG constructions employs the following ideas. First we use the fact, discovered by Trevisan [38] (see also [39, 32]), that black-box PRG constructions give rise to 'good' *extractors* [29]. Then we show that constant depth circuits cannot compute 'good' extractors. For this last point we use the notion of *noise sensitivity*, which is a measure of how likely the output of a function is to change when the input is perturbed with random noise. On the one hand we show that extractors are very sensitive to noise, while on the other hand we know that constant depth circuits are not (by results of Linial, Mansour and Nisan [23] and Boppana [8]). This dichotomy establishes our lower bound.

Our lower bound for black-box worst-case hardness amplifications proceeds along similar lines: First, following [36, 39], we show that black-box worst-case hardness amplifications give rise to 'good' list-decodable codes. Then we show that 'good' list-decodable codes are very sensitive to noise. Again, the lower bound follows from the fact that constant depth circuits are not very sensitive to noise. It should be noted that a certain lower bound for black-box worst-case hardness amplifications already follows from our previous results. Namely, if there is a black-box worst-case hardness amplification then combining this with our black-box PRG construction from mild average-case hardness one gets a black-box PRG construction from worst-case hardness, and our lower bound applies. However, we get a more general lower bound through a direct proof.

We show that our time lower bounds are tight. On the other hand, if one insists on a black-box PRG construction based on worst-case hard functions that is computable in $ATIME(O(1), \log n)$, where n is the output length of the PRG, then one is forced to start with a hardness assumption so strong that worst-case and mild average-case hardness are equivalent. In such a case, no worst-case hardness amplification is needed, and one can build a PRG using our construction from mild average-case hard functions. The same tradeoff holds for hardness amplification: If one insists on black-box worst-case hardness amplification within $ATIME(O(1), l)$, where l is the input size of the hard function, then one is forced to start with a hardness assumption so strong that worst-case and mild average-case hardness are equivalent. In such a case, worst-case hardness amplification is vacuous.

Related work: There exist several other works addressing the complexity of PRGs. Some works study the complexity of building PRGs assuming that some specific problem is hard: Impagliazzo and Naor [15] show how to construct a PRG : $\{0,1\}^{n-\theta(\log n)} \rightarrow \{0,1\}^n$ in AC_0. Naor and Reingold [25] give PRG constructions based on number-theoretic hardness assumptions. Their PRGs are computable in TC_0.

Other works study which complexity classes can contain PRGs. Kharitonov, Goldberg and Yung [20] and Yu and Yung [44] prove strong negative results about the ability of various automata and other space-restricted devices to compute PRGs. Linial, Mansour and Nisan [23] prove that AC_0 cannot compute pseudorandom functions (an object related to PRGs). Cryan and Miltersen [9] consider the question of whether there are PRGs in NC_0.

However, none of the above works address the general question of what is the complexity of constructing a PRG from *any* hard function. Moreover, they do not deal with PRGs having logarithmic seed length, corresponding to high-end derandomization.

It should also be noted that space lower bounds for on-line computation of extractors and list-decodable codes are proved in [5]. However, these lower bounds hold only in the on-line model of computation and therefore are

incomparable with ours.

Organization: In Section 2 we give some preliminaries. In Section 3 we survey previous results about PRGs which are needed for the paper. In Section 4 we describe our results. In Section 5 we show how to construct a PRG computable in $ATIME(O(1), \log n)$ from a mild average-case hardness assumption. In Section 6 we prove our lower bound for black-box PRG constructions from worst-case hardness assumptions. We also discuss in which sense our results are tight. In Section 7 we prove our lower bound for black-box worst-case hardness amplification. In Section 8 we relax the hardness assumptions to the existence of functions hard for constant depth circuits. Finally, Section 9 discusses some open problems.

2. Preliminaries

Complexity: In this paper circuits have *unbounded fan-in*. The *size* of a circuit is the number of *edges* in the circuit. We denote by $AC_0[d]$ the class of circuits of depth d. $AC_0 := \cup_d AC_0[d]$. We denote by $TC_0[d]$ the class of circuits of depth d with majority gates. $TC_0 := \cup_d TC_0[d]$. We denote by CKT the class of circuits with no depth restriction.

Let $\mathcal{C}$ be a circuit class. We also think of $\mathcal{C}$ as a class of functions. We say that a function f is in $\mathcal{C}$ if f has $\mathcal{C}$-circuits of polynomial size. In all other cases we will explicitly specify the size of the circuits.

A family $\{C_n\}$ of circuits is $DLOGTIME$-uniform if the *direct connection language* of the circuit family can be decided in deterministic logarithmic time[2]. Where the direct connection language is the language of tuples (t, a, b, y) such that $|y| = n$, a and b are numbers of gates in C_n, gate b is a child of gate a and gate a is of type t. In this paper, *uniform always means $DLOGTIME$-uniform*.

We denote by $ATIME(O(1), l)$ the class of functions computable in time $O(l)$ with constant number of alternations by a multitape Turing machine. We use the following characterization of uniform AC_0 due to Barrington, Immerman and Straubing:

Theorem 2.1 ([6]). *A function $f : \{0,1\}^n \to \{0,1\}$ is in uniform AC_0 if and only if it is in $ATIME(O(1), \log n)$.*

Let $f : \{0,1\}^l \to \{0,1\}^{O(l)}$, and let $f(x)_i$ be the i-th bit of $f(x)$. We say that f is in $ATIME(O(1), l)$ if the language $\{(x, i) : f(x)_i = 1\}$ is in $ATIME(O(1), l)$. Note that if g is in $ATIME(O(1), l)$ then $g \circ f$ is in $ATIME(O(1), l)$: We can compute $g(f(x))$ by existentially guessing $y \in \{0,1\}^{O(l)}$, universally verifying that $y_i = f(x)_i$ for every i, and then simulating the machine for g on input y.

In this paper we need to show that some functions are in $ATIME(O(1), l)$. We sometimes make use of the following result, usually attributed to Nepomnjaščiĭ [26]:

Theorem 2.2 ([26]). *For any $\epsilon > 0$, if $f : \{0,1\}^l \to \{0,1\}$ is computable by an algorithm running in time $\mathrm{poly}(l)$ and using space $l^{1-\epsilon}$ then f is in $ATIME(O(1), l)$.*

We will occasionally consider the following complexity classes: We denote by $CTIME(O(1), l)$ the extension of $ATIME(O(1), l)$ where we also allow for *counting* quantifiers. This class was introduced by Wagner [42] and studied, among others, by Torán [37]. Along the same lines we denote by $A{\oplus}TIME(O(1), l)$ the extension of $ATIME(O(1), l)$ where we also allow for *parity* quantifiers. The same techniques in Theorem 2.1 give the following theorem.

Theorem 2.3. *Let $f : \{0,1\}^n \to \{0,1\}$. f is in uniform TC_0 if and only if it is in $CTIME(O(1), \log n)$. f is in uniform AC_0 with parity gates if and only if it is in $A{\oplus}TIME(O(1), \log n)$.*

[2] Logarithmic time Turing machines have a special address tape. On a given time step the machine has access to the bit of the input denoted by the contents of the address tape.

Let C be a complexity class. The class $BP \cdot C$ consists of the languages L for which there is $V \in C$ and a polynomial p such that $x \in L \Rightarrow \Pr_{y:|y|=p(|x|)}[V(x,y)=1] \geq 2/3$ and $x \notin L \Rightarrow \Pr_{y:|y|=p(|x|)}[V(x,y)=1] \leq 1/3$.

For background on circuit complexity and uniformity the reader may consult the survey by Allender and Wagner [3] and the textbook by Vollmer [41].

Hardness and pseudorandomness: We denote by U_l a random variable uniform on $\{0,1\}^l$.

Let $\mathcal{C}$ be a circuit class. A function $f : \{0,1\}^l \to \{0,1\}$ is (g,δ)-*hard* for $\mathcal{C}$ if for every circuit $C \in \mathcal{C}$ of size at most g we have:

$$\Pr[C(U_l) = f(U_l)] < \delta.$$

Worst-case hardness corresponds to $\delta = 1$. Our threshold for *average-case* hardness is *mild* average-case hardness, corresponding to δ at most $(1 - 1/l^c)$ for some c.

A function $G : \{0,1\}^u \to \{0,1\}^n$ is a (n,ϵ)-*pseudorandom generator* (PRG) against $\mathcal{C}$ if for all $C \in \mathcal{C}$ of size at most n we have:

$$\Big|\Pr[C(G(U_u)) = 1] - \Pr[C(U_n) = 1]\Big| \leq \epsilon.$$

A n-PRG is a $(n, 1/n)$-PRG. We refer to u as the *seed length* of G.

3. Previous Results

In this section we give the background about PRGs needed for the paper. We start with PRGs against CKT. Then we focus on PRGs against constant depth circuits.

3.1. PRGs against CKT

Nisan and Wigderson [28] show how to build PRGs from strong average-case hardness assumptions. We recall the definition of their PRG and state their result.

Definition 3.1 ([28]). *An (m,l) design of size n over a universe U is a collection $(S_1,...,S_n)$ of subsets of U, each of size l, such that for any $1 \leq i < j \leq n$, the intersection $S_i \cap S_j$ has size at most m.*

For a function $f : \{0,1\}^l \to \{0,1\}$, and a $(\log n, l)$ design of size n over a universe of size u, the Nisan-Wigderson PRG NW_f is defined as

$$\begin{aligned} NW_f &: \{0,1\}^u \to \{0,1\}^n \\ NW_f(x) &= f(x|_{S_1}) \circ \cdots \circ f(x|_{S_n}), \end{aligned}$$

where $x|_S$ is the string obtained from x by selecting the bits indexed by S.

Theorem 3.2 ([28]). *If there is a function $f : \{0,1\}^l \to \{0,1\}$ in E that is $(2^{\Omega(l)}, 1/2 + 2^{-\Omega(l)})$-hard for CKT then there is a n-PRG against CKT with seed length $O(\log n)$ and computable in time $\mathrm{poly}(n)$, and in particular $BP \cdot P = P$.*

Proof Idea: The PRG is NW_f for a family of $(\log n, c \log n)$ designs of size n over a universe of size $d \log n$, for some constants c, d. Specifically, one needs such a family for every given c and some d. Nisan and Wigderson show that these families are computable in time $\mathrm{poly}(n)$, and that NW_f is a n-PRG. The 'in particular' part is proved as follows: We run the algorithm we want to derandomize using all the possible outputs of the PRG in place of true random bits. Then we decide according to majority vote. ■

An important point to keep in mind is that, although we are assuming that f is in E, the PRG is computable in time $\mathrm{poly}(n)$. This comes from the fact that f is evaluated on inputs of length $O(\log n)$.

A major line of research in the last ten years has focused on relaxing the average-case hardness assumption in Theorem 3.2 to a worst-case one, that is, the existence of a function in E that is $(2^{\Omega(l)}, 1)$-hard for CKT. This was first achieved through the following *hardness amplifications within* E.

First, in [4], random self-reducibility of EXP-complete problems is used to convert a worst-case hard function to one with mild average-case hardness.

Theorem 3.3 ([4]). *If there is a function $f : \{0,1\}^l \to \{0,1\}$ in E that is $(2^{\Omega(l)}, 1)$-hard for CKT, then there is a function $f' \in E$ that is $(2^{\Omega(l)}, 1 - 1/\mathrm{poly}(l))$-hard for CKT.*

Proof Idea: f' is a small degree, multi-variate polynomial extension of f. For a suitable choice of parameters, the random self-reducibility of low-degree polynomials implies that f' has the required hardness. ■

Then in [14] mild average-case hardness is amplified to constant hardness.

Theorem 3.4 ([14]). *If there is a function $f : \{0,1\}^l \to \{0,1\}$ in E that is $(2^{\Omega(l)}, 1 - 1/\mathrm{poly}(l))$-hard for CKT, then there is a function $f' \in E$ that is $(2^{\Omega(l)}, 2/3)$-hard for CKT.*

Proof Idea: Let $f' : \{0,1\}^{O(l)} \to \{0,1\}$ be:

$$f'(a, r) := \langle f(x_1) \circ \cdots \circ f(x_l), r \rangle$$

where $|a| = O(l)$, $|r| = l$ and $x_1, ..., x_l$ are pairwise independent samples in $\{0,1\}^l$ obtained from seed a, and $\langle ., . \rangle$ denotes inner product mod 2. In other words, f' is the inner product of the random string r with l evaluations of f on pairwise independent inputs $x_1, ..., x_l$.

It is shown in [14] that, if we apply this transformation a constant number of times to f, then we obtain a function with constant hardness. ■

Finally, in [18] it is shown how to amplify constant hardness to the kind of hardness required in Theorem 3.2.

Theorem 3.5 ([18]). *If there is a function $f : \{0,1\}^l \to \{0,1\}$ in E that is $(2^{\Omega(l)}, 2/3)$-hard for CKT, then there is a function $f' \in E$ which is $(2^{\Omega(l)}, 1/2 + 2^{-\Omega(l)})$-hard for CKT.*

Proof Idea: Let $f' : \{0,1\}^{O(l)} \to \{0,1\}$ be:

$$f'(x, r, v_1, p) := \langle f(x|_{S_1} \oplus v_1) \circ \cdots \circ f(x|_{S_l} \oplus v_l), r \rangle,$$

where $\oplus$ denotes bit-wise XOR, $(S_1, \ldots, S_l)$ is a (l, cl) design of size l over a universe of size dl, for some c, d as in Theorem 3.2, and $(v_1, \ldots, v_l)$ is a walk in an expander graph over $\{0,1\}^{O(l)}$ with constant degree and bounded second largest eigenvalue. This walk is obtained starting at v_1 and walking according to p. ■

Combining all these results, one gets:

Theorem 3.6 ([28, 4, 14, 18]). *If there is a function $f : \{0,1\}^l \to \{0,1\}$ in E that is $(2^{\Omega(l)}, 1)$-hard for CKT then there is a n-PRG against CKT with seed length $O(\log n)$ and computable in time $\mathrm{poly}(n)$, and in particular $BP \cdot P = P$.*

After [18] PRGs constructions from worst-case hard functions have been simplified and strengthened [17, 36, 33, 40]. In particular, last constructions do not fall in the twofold paradigm 'hardness amplification + NW PRG', but directly transform worst-case hardness into pseudorandomness. However, our results suggest that transforming worst-case hardness into pseudorandomness is a substantially harder task than transforming mild average-case hardness into pseudorandomness. Therefore we use the earlier constructions that allow us to investigate the fine structure of hardness amplification.

Klivans and van Melkebeek [22] prove a space-bounded analogue of Theorem 3.6. They show how to amplify hardness within linear space, then they give a more efficient implementation of the NW PRG. We summarize their final result in the following theorem.

Theorem 3.7 ([22]). *If there is a function $f : \{0,1\}^l \to \{0,1\}$ in $SPACE(l)$ that is $(2^{\Omega(l)}, 1)$-hard for CKT, then there is a n-PRG against CKT with seed length $O(\log n)$ and computable in $SPACE(\log n)$, and in particular $BP \cdot L = L$.*

3.2. PRGs against Constant Depth Circuits

A natural question, addressed by Agrawal [1], is: What are the hardness assumptions needed for constructing PRGs against more restricted classes of circuits? As pointed out in [1], in all the proofs of correctness of the above constructions the depth only increases by a constant factor, *provided that the circuits have majority gates*. This gives the following result:

Theorem 3.8 ([28, 4, 14, 18, 1]). *There is a constant c such that if there is a function $f : \{0,1\}^l \to \{0,1\}$ in E that is $(2^{\Omega(l)}, 1)$-hard for $TC_0[d + c]$, then there is a n-PRG against $TC_0[d]$ with seed length $O(\log n)$ and computable in time $\mathrm{poly}(n)$. In particular, if for every d there is a function $f : \{0,1\}^l \to \{0,1\}$ in E that is $(2^{\Omega(l)}, 1)$-hard for $TC_0[d]$, then $BP \cdot TC_0 = TC_0$.*

Now we focus on PRGs against AC_0. Note that Theorem 3.8 does not immediately translate to AC_0 because it is known that AC_0 cannot compute majority [10].

Nisan [27] builds an *unconditional* PRG against AC_0, using the results by Håstad [13] on the average-case hardness of the function *parity*$(x_1 \dots x_l) := (\sum_i x_i) \bmod 2$.

Theorem 3.9 ([27]). *For every d there is a n-PRG against $AC_0[d]$ with seed length $\log^{O(1)} n$, and computable in time $\mathrm{poly}(n)$.*

Proof Idea: The PRG is NW_{parity} for a family of $(\log n, \log^c n)$ designs of size n over a universe of size $\log^e n$, for some constants c, e. Specifically, one needs such a family for every given c and some e. Nisan shows how to construct such families in time $\mathrm{poly}(n)$. ■

Although Nisan's PRG does not rely on any complexity assumption, it has polylogarithmic seed length, and therefore it cannot be used directly to obtain $BP \cdot AC_0 = AC_0$.

One can try to build, under the assumption that some function is hard for AC_0, a PRG with logarithmic seed length against AC_0, following the construction in Section 3.1. The difficulty in this approach is that the proof of correctness of the construction in Theorem 3.5 (and other approaches like [36]) does not carry through in AC_0 [1].

This problem is discussed and then solved by Agrawal [1]:

Theorem 3.10 ([1]). *There is a constant c such that if there is a function $f : \{0,1\}^l \to \{0,1\}$ in E that is $(2^{\Omega(l)}, 1)$-hard for $AC_0[c \cdot d]$, then there is a $(n, 1/\log^{O(1)} n)$-PRG against $AC_0[d]$ with seed length $O(\log n)$ and computable in time $\mathrm{poly}(n)$.*

Proof Idea: Agrawal's PRG is obtained combining a conditional PRG G with Nisan's unconditional PRG from Theorem 3.9. Since Nisan's PRG has polylogarithmic seed length, we can get a combined PRG with logarithmic seed length if G has only polynomial stretch (i.e. $G : \{0,1\}^l \to \{0,1\}^{l^{O(1)}}$). Now, to build such a G we can use exactly the same construction in Section 3.1: Agrawal shows that, since the stretch of G is only polynomial, all the proofs of correctness carry through in AC_0. ■

Note that Theorem 3.10 gives a $(n, 1/\log^{O(1)} n)$-PRG instead of a n-PRG. However, this is sufficient for derandomization purposes.

As we already mentioned, a PRG with logarithmic seed length allows us to derandomize an algorithm *provided that we can compute majority*. While it is known that majority cannot be computed in AC_0 [13], Klivans [21] notices that one can use Ajtai's construction [2] to *approximately* compute majority in AC_0, which is enough for the derandomization to go through.

This gives the following corollary.

Corollary 3.11 ([1, 21]). *If for every d there is a function $f : \{0,1\}^l \to \{0,1\}$ in E that is $(2^{\Omega(l)}, 1)$-hard for $AC_0[d]$, then $BP \cdot AC_0 = AC_0$.*

Before ending our survey, a comment about the uniformity conditions is in order. Theorem 3.8 gives, conditionally, $BP \cdot TC_0 = TC_0$, and Corollary 3.11 gives, conditionally, $BP \cdot AC_0 = AC_0$. Note we did not specify the uniformity condition for these circuit classes. To our knowledge, previous to this paper the best result in this direction was that these derandomizations hold for L-uniform circuit families. That is, circuit families described by a Turing machine running in logarithmic space. This may be obtained using the techniques in [22, 21].

4. Our Results

In this section we describe our results. The main ones are summarized and compared to previous results in Table 1.

Our goal is to cast the results in Section 3 to alternating time with $O(1)$ alternations: We aim to construct a PRG computable in $ATIME(O(1), \log n)$ which could then be used to derandomize $BP \cdot ATIME(O(1), \log n)$, i.e. proving $BP \cdot ATIME(O(1), \log n) = ATIME(O(1), \log n)$. Note that, because of Theorem 2.1, this derandomization corresponds to the derandomization of uniform AC_0, i.e. uniform $BP \cdot AC_0 = \text{uniform}\, AC_0$. Similar considerations apply to our other derandomizations because of Theorem 2.3.

Improving on the complexity of the design construction in the NW PRG, we obtain the following:

Theorem 4.1. *If there is a function $f : \{0,1\}^l \to \{0,1\}$ in $ATIME(O(1), l)$ that is $(2^{\Omega(l)}, 1/2 + 2^{-\Omega(l)})$-hard for CKT then there is a n-PRG against CKT with seed length $O(\log n)$ and computable in $ATIME(O(1), \log n)$, and $BP \cdot ATIME(O(1), \log n) = ATIME(O(1), \log n)$.*

In analogy with the results discussed in Section 3.1, to relax the average-case hardness assumption in Theorem 4.1 we study hardness amplification in the linear exponential analogue of $ATIME(O(1), \log n)$, that is, linear alternating time with $O(1)$ alternations.

We notice that the constructions in Theorems 3.4 and 3.5 can be carried out within linear alternating time with $O(1)$ alternations.

Theorem 4.2. *If there is a function $f : \{0,1\}^l \to \{0,1\}$ in $ATIME(O(1), l)$ that is $(2^{\Omega(l)}, 1 - 1/\text{poly}(l))$-hard for CKT, then there is a function $f' \in ATIME(O(1), l)$ which is $(2^{\Omega(l)}, 1/2 + 2^{-\Omega(l)})$-hard for CKT.*

Combining Theorems 4.1 and 4.2 we can build a PRG computable in $ATIME(O(1), \log n)$ from a function of mild average-case hardness.

Theorem 4.3. *If there is a function $f : \{0,1\}^l \to \{0,1\}$ in $ATIME(O(1), l)$ that is $(2^{\Omega(l)}, 1-1/\mathrm{poly}(l))$-hard for CKT then there is a n-PRG against CKT with seed length $O(\log n)$ and computable in $ATIME(O(1), \log n)$, and $BP \cdot ATIME(O(1), \log n) = ATIME(O(1), \log n)$.*

Theorems 4.1, 4.2 and 4.3 are stated formally and proved in Section 5.

On the negative side, we show a lower bound for black-box PRG constructions starting from hard functions.

Theorem 4.4 (Informal). *There is no black-box PRG construction $G : \{0,1\}^{O(\log n)} \to \{0,1\}^n$ from worst-case hard functions such that G is computable in alternating time with $O(1)$ alternations, even if we allow time $n^{o(1)}$.*

Note that most constructions are black-box [18, 22, 17, 36, 33, 40].

It is interesting to note that the bottleneck is indeed worst-case hardness amplification:

Theorem 4.5 (Informal). *There is no black-box worst-case to mild average-case hardness amplification computable in alternating time with $O(1)$ alternations, even if we allow subexponential time.*

Note that Theorem 4.5 implies that there is no black-box worst-case hardness amplification in the polynomial time hierarchy.

Again, note that most known approaches [4, 36] are black-box.

Theorems 4.4 and 4.5 are tight in the following sense: The only settings of parameters which are not ruled out correspond either to computational resources that essentially allow for the worst-case hardness amplification in Theorem 3.3, that combined with Theorem 4.3 gives a PRG construction from worst-case hard functions, or else they correspond to hardness assumptions so strong that worst-case hardness and mild average-case hardness *collapse*, in which case no worst-case hardness amplification is needed, and to get a PRG one can apply directly Theorem 4.3.

It should be noted that the lower bounds in Theorems 4.4 and 4.5 hold for constant depth circuits, the nonuniform analogue of alternating time with $O(1)$ alternations.

Theorems 4.4 and 4.5 are proved in Sections 6 and 7, respectively.

We note that worst-case to average-case hardness amplification becomes feasible if one allows one parity gate. This allows us to build a PRG computable in $A\oplus TIME(O(1), \log n)$ from a worst-case hardness assumption.

Theorem 4.6. *If there is a function $f : \{0,1\}^l \to \{0,1\}$ in $A\oplus TIME(O(1), l)$ that is $(2^{\Omega(l)}, 1)$-hard for CKT then there is a n-PRG against CKT with seed length $O(\log n)$ and computable in $A\oplus TIME(O(1), \log n)$, and $BP \cdot A\oplus TIME(O(1), \log n) = A\oplus TIME(O(1), \log n)$.*

Theorem 4.6 is proved in Section 6.1.

What is not completely satisfactory in the above derandomization results is that our hardness assumptions are qualitatively stronger than the corresponding derandomizations. For example, consider Theorem 4.3. The nonuniform analogue of $ATIME(O(1), \log n)$ is AC_0, not CKT. So one wants the same conclusions under the weaker assumption of a hard function for AC_0. Using Agrawal's construction presented in Theorem 3.10, we obtain the following theorem.

Theorem 4.7. *There is a constant c such that if there is a function $f : \{0,1\}^l \to \{0,1\}$ in $ATIME(O(1), l)$ that is $(2^{\Omega(l)}, 1 - 1/l^b)$-hard for $AC_0[c \cdot \max(b, d)]$, then there is a $(n, 1/\log^{O(1)} n)$-PRG against $AC_0[d]$ with logarithmic seed length and computable in $ATIME(O(1), \log n)$.*

In particular, if there is a constant b such that for every d there is a function in $ATIME(O(1), l)$ that is $(2^{\Omega(l)}, 1 - 1/l^b)$-hard for $AC_0[d]$, then $BP \cdot ATIME(O(1), \log n) = ATIME(O(1), \log n)$.

Finally, we point out the following derandomization of $BP \cdot CTIME(O(1), \log n)$ under worst-case hardness assumptions for TC_0.

Theorem 4.8. *There is a constant c such that if there is a function $f : \{0,1\}^l \to \{0,1\}$ in $CTIME(O(1), l)$ that is $(2^{\Omega(l)}, 1)$-hard for $TC_0[c+d]$, then there is a n-PRG against $TC_0[d]$ with seed length $O(\log n)$ and computable in $CTIME(O(1), \log n)$.*

In particular, if for every d there is a function $f : \{0,1\}^l \to \{0,1\}$ in $CTIME(O(1), l)$ that is $(2^{\Omega(l)}, 1)$-hard for $TC_0[d]$, then $BP \cdot CTIME(O(1), \log n) = CTIME(O(1), \log n)$.

Theorems 4.7 and 4.8 are proved in Section 8.

5. Average-Case Hardness vs. Randomness

In this section we show how to build a n-PRG $G : \{0,1\}^{O(\log n)} \to \{0,1\}^n$ against CKT computable in $ATIME(O(1), \log n)$ starting from a function $f : \{0,1\}^l \to \{0,1\}$ in $ATIME(O(1), l)$ that is $(2^{\Omega(l)}, 1 - 1/\mathrm{poly}(l))$-hard for CKT. In particular, we prove Theorems 4.1, 4.2 and 4.3.

Here our main new technical contribution is the construction of the family of designs to be used in the NW PRG, which we now discuss.

First we show how to compute pairwise independent samples over $\{0,1\}^l$ in $ATIME(O(1), l)$. A matrix T with entries in $\{0,1\}$ is *Toeplitz* if it is constant on diagonals. It is well known (cf., [11]) that if we choose a random $l \times l$ Toeplitz matrix T and a random vector $U \in \{0,1\}^l$, then the 2^l random variables $\{Tx + U : x \in \{0,1\}^l\}$ are pairwise independent over $\{0,1\}^l$.

Clearly, a $l \times l$ Toeplitz matrix T is uniquely determined by the string $t \in \{0,1\}^{2l-1}$ of its values on the first row and on the first column. The following lemma states that we can compute pairwise independent samples over $\{0,1\}^l$ in $ATIME(O(1), l)$.

Lemma 5.1. *There is a machine $A(t, x, u)$ which computes $Tx + u$ in $ATIME(O(1), l)$ for $|x| = |u| = l$, $|t| = 2l - 1$ and T the Toeplitz matrix determined by t.*

Proof: Recall what we need to show is that, given t, x, u and i, we can compute the i-th bit of $Tx + u$ in $ATIME(O(1), l)$. We actually show that it can be computed in deterministic time $O(l)$. It is easy to see that the i-th bit of $Tx + u$ is

$$\langle t_i \dots t_{i+l-1}, x \rangle + u_i.$$

Note the inner product is over l bits, and therefore can be computed in time $O(l)$. ■

We now show our design construction.

Lemma 5.2. *For every constant c there is a constant d such that there is a family $\{D_n\}$ of $(\log n, c \log n)$ designs of size n over a universe of size $d \log n$ with the following property: There is a machine in $ATIME(O(1), \log n)$ such that, given n and $k \leq n$, computes the characteristic vector of the k-th set in D_n.*

Proof: Let $l := \log n$. First we show the existence with a probabilistic argument. Then we show how to derandomize the argument. Finally, we show how the derandomization is implementable in $ATIME(O(1), l)$.

Existence: We view the universe as cl blocks of b elements each, i.e. let $d := cb$, for some b we specify later.

Let us choose $S_1, \dots, S_n$ at random from the sets which have exactly one element in each block. Notice the size of these sets is cl, as required.

For every $i \neq j$, by a union bound:

$$\Pr[|S_i \cap S_j| \geq l] \leq \binom{cl}{l} \left(\frac{1}{b}\right)^l \leq \left(\frac{ecl}{l}\right)^l \left(\frac{1}{b}\right)^l \leq \left(\frac{ec}{b}\right)^l.$$

Taking $b := 4ec$ the latter equals $1/n^2$.

So, by a union bound:

$$\Pr[\exists i < j : |S_i \cap S_j| \geq l] \leq \sum_{i<j} \Pr[|S_i \cap S_j| \geq l] \leq \binom{n}{2} \frac{1}{n^2} < \frac{1}{2} < 1.$$

Therefore such designs exist.

Derandomization: Note that the analysis goes through even if the sets are just pairwise independent. We use this below to show that $D_n \in ATIME(O(1), \log n)$.

$ATIME(O(1), l)$*:* Each string $s \in \{0,1\}^{(\log b)\cdot cl}$ represents a set S with one element in each block in the following natural way: View s as cl blocks of $\log b$ bits each, the i-th block of s is an index to an element in the i-th block of b elements in our universe. We can easily build a machine T running in time $O(l)$ computing this transformation, i.e. $T(s) \in \{0,1\}^{b\cdot cl}$ is the characteristic vector of the set with one element in each block which $s \in \{0,1\}^{(\log b)\cdot cl}$ represents.

Let $A \in ATIME(O(1), l)$ be the machine given by Lemma 5.1 that, given a and i, computes the i-th pairwise independent sample over $\{0,1\}^{(\log b)\cdot cl}$ according to a. Note we can check in $ATIME(O(1), l)$ if the samples corresponding to some a form a design:

$$\forall i \neq j \in \{0,1\}^l \left| T(A(a,i)) \cap T(A(a,j)) \right| \leq l.$$

We already know that A and T are in $ATIME(O(1), l)$. Note that computing the intersection size is feasible since we are dealing with strings of length $O(l)$.

To put our hands on some particular design, we can existentially guess a string a^* and universally verify that it is the lexicographically first string whose samples correspond to a design. The characteristic vector of the k-th set in D_n is then $T(A(a^*, k))$. ■

Remark 5.3. Our construction of designs is a mix of the constructions in [31] and [22]: We choose the sets with one element in each block, as in [31], and we derandomize the argument through pairwise independence, as in [22]. Neither the construction in [31] nor the one in [22] seems to be easily implementable in $ATIME(O(1), \log n)$: [31] seems to require polynomial space in the size of the design because of the method of conditional probabilities. [22] needs to associate to a number $x \leq \binom{c\log n}{d\log n}$ the x-th subset of $\{1, \ldots, c\log n\}$ of size $d\log n$. This latter operation can be easily computed in $SPACE(\log n)$, going through all the subsets, but we do not know if it can be computed in $ATIME(O(1), \log n)$. Moreover, the analysis of our construction is simpler than the analysis in [22].

Other constructions of designs, based on error correcting codes, are obtained by Salil Vadhan (credited in the journal version of [12]) and by Luca Trevisan and Hoeteck Wee (personal communication, Sept. 2002). We do not know if these constructions are computable in $ATIME(O(1), \log n)$.

Plugging this design construction into NW PRG we obtain the following:

Theorem 4.1 (restated). *If there is a function $f : \{0,1\}^l \to \{0,1\}$ in $ATIME(O(1), l)$ that is $(2^{\Omega(l)}, 1/2 + 2^{-\Omega(l)})$-hard for CKT then there is a n-PRG against CKT with seed length $O(\log n)$ and computable in $ATIME(O(1), \log n)$, and $BP \cdot ATIME(O(1), \log n) = ATIME(O(1), \log n)$.*

Proof: The PRG is NW_f, with the design construction from Lemma 5.2. The correctness of this construction has already been proved in [28]. The fact that $NW_f \in ATIME(O(1), \log n)$ follows from Lemma 5.2 and the fact that $f \in ATIME(O(1), \log n)$. In analogy with Corollary 3.11, to obtain $BP \cdot ATIME(O(1), \log n) = ATIME(O(1), \log n)$ we use Ajtai's construction for approximate majority [2]. (In [2] the construction is given in terms of first-order definability, but this coincides with $ATIME(O(1), \log n)$ [6, 41].) ■

Along the lines of the previous results discussed in Section 3.1, we now want to relax the average-case hardness assumption. Therefore we now prove some results about hardness amplification within $ATIME(O(1), l)$. These hardness amplifications will allow us to start from a function with mild average-case hardness. See Sections 6 and 7 for a discussion of worst-case hardness assumptions.

Using the same construction in Theorem 3.4 we can amplify from mild hardness to constant hardness:

Theorem 5.4. *If there is a function $f : \{0,1\}^l \to \{0,1\}$ in $ATIME(O(1), l)$ that is $(2^{\Omega(l)}, 1 - 1/\mathrm{poly}(l))$-hard for CKT, then there is a function $f' \in ATIME(O(1), l)$ that is $(2^{\Omega(l)}, 2/3)$-hard for CKT.*

Proof: We use the construction in Theorem 3.4. The correctness of this construction has already been proved in [14], so it is only left to see that $f' \in ATIME(O(1), l)$. This follows from the construction of a pairwise independent sample space given in Lemma 5.1. ■

Using the same construction in Theorem 3.5 we can amplify from constant hardness to exponential hardness.

Theorem 5.5. *If there is a function $f : \{0,1\}^l \to \{0,1\}$ in $ATIME(O(1), l)$ that is $(2^{\Omega(l)}, 2/3)$-hard for CKT, then there is a function $f' \in ATIME(O(1), l)$ which is $(2^{\Omega(l)}, 1/2 + 2^{-\Omega(l)})$-hard for CKT.*

Proof: We use the construction in Theorem 3.5. The correctness of this construction has already been proved in [18], so it is only left to see that $f' \in ATIME(O(1), l)$. Lemma 5.2 shows how to compute the required designs in $ATIME(O(1), l)$.

It is only left to

show how to compute walks on expanders in $ATIME(O(1), l)$. This problem, for the parameters of interest here, has already been solved by Ajtai [2], using the expander construction by Lubotzky, Phillips and Sarnak [24].

Theorem 5.6 ([24, 2]). *There is a constant $\alpha, 0 < \alpha < 1$, such that for every prime n congruent to 1 modulo 4 there is a 6-regular graph G_n on n vertices with second largest eigenvalue at most α. Moreover, there is a machine in $ATIME(O(1), \log n)$ such that, given a prime n congruent to 1 modulo 4, $x \in G_n$ and p, with $|p| \leq O(\log n)$, computes the node in G_n reached starting from x and following the path specified by p.*

■

Combining the above two hardness amplifications we get the following theorem.

Theorem 4.2 (restated). *If there is a function $f : \{0,1\}^l \to \{0,1\}$ in $ATIME(O(1), l)$ that is $(2^{\Omega(l)}, 1 - 1/\mathrm{poly}(l))$-hard for CKT, then there is a function $f' \in ATIME(O(1), l)$ which is $(2^{\Omega(l)}, 1/2 + 2^{-\Omega(l)})$-hard for CKT.*

This allows us to build a PRG computable in $ATIME(O(1), \log n)$ from a function of mild average-case hardness.

Theorem 4.3 (restated). *If there is a function $f : \{0,1\}^l \to \{0,1\}$ in $ATIME(O(1), l)$ that is $(2^{\Omega(l)}, 1 - 1/\mathrm{poly}(l))$-hard for CKT then there is a n-PRG against CKT with seed length $O(\log n)$ and computable in $ATIME(O(1), \log n)$, and $BP \cdot ATIME(O(1), \log n) = ATIME(O(1), \log n)$.*

6. PRGs from Worst-Case Hardness

In this section we discuss PRG constructions from worst-case hardness assumptions, and in particular we prove a formal version of Theorem 4.4, establishing a lower bound for black-box PRG constructions from worst-case hardness assumptions. In Section 6.1 we discuss the tightness of our lower bound and we also prove Theorem 4.6.

To show our lower bound for black-box PRG constructions we proceed in two steps: First we use the fact, discovered by Trevisan [38] (see also [39, 32]), that black-box PRG constructions give rise to 'good' *extractors*. Then we show that 'good' extractors are not computable by constant depth circuits (the nonuniform analogue of alternating time with $O(1)$ alternations). To explain the intuition behind this last step we need the notion of *noise sensitivity*. Roughly speaking, the noise sensitivity of a function is a measure of how likely the output of the function is to change when one perturbs the input with random noise. We show that 'good' extractors are very sensitive to noise. Since constant depth circuits are not [23, 8], we obtain our lower bound.

We now proceed to turn the above sketch into a formal proof.

Definition 6.1. *An oracle algorithm $G^f : \{0,1\}^u \to \{0,1\}^n$ is a (l, s, ε) black-box PRG construction if for every $f : \{0,1\}^l \to \{0,1\}$ and for every $A : \{0,1\}^n \to \{0,1\}$ such that*

$$\left| \Pr[A(G^f(U_u)) = 1] - \Pr[A(U_n) = 1] \right| \geq \varepsilon$$

there is an oracle circuit C of size at most s such that $C^A(x) = f(x)$ for every x.

Note in the above definition we did not specify the type of the circuit C (e.g. $CKT, AC_0, \ldots$) because it does not play a role in this section. Also note that if $G^f : \{0,1\}^u \to \{0,1\}^n$ is a (l, s, ε) black-box PRG construction then for every function $f : \{0,1\}^l \to \{0,1\}$ we have that if f is $(ns, 1)$-hard then G^f is a n-PRG.

We note that most of the known PRG constructions are black-box [18, 22, 17, 36, 33, 40]. For example, in this notation the PRG construction in Theorem 3.6 is a $(O(\log n), n^\gamma, 1/n)$-black-box PRG construction, for some $0 < \gamma < 1$ (see, e.g., [38]). This construction also gives $u = O(\log n)$, which is what one needs for high-end derandomization. However, our lower bound applies regardless of this.

We now define extractors.

The *min-entropy* of a random variable X is defined as $H_\infty(X) := \min_x \log(1/\Pr[X = x])$.

Definition 6.2 ([29]). *$E : \{0,1\}^h \times \{0,1\}^u \to \{0,1\}^n$ is a (k, ε) extractor if for every random variable X of min-entropy at least k, and for every $T \subseteq \{0,1\}^n$:*

$$\left| \Pr[E(X, U_u) \in T] - \Pr[U_n \in T] \right| \leq \varepsilon.$$

We call $T \subseteq \{0,1\}^n$ a test *and $y \in \{0,1\}^u$ a* seed*.*

Trevisan [38] shows that black-box PRG constructions are extractors (see also [39, 32]). For completeness, we now state and prove this result.

Theorem 6.3 ([38]). *Let $G^f : \{0,1\}^u \to \{0,1\}^n$ be a (l, s, ε)-black-box PRG construction. Then $E : \{0,1\}^{2^l} \times \{0,1\}^u \to \{0,1\}^n$ defined as $E(x, y) := G^x(y)$ is a $(O(s \log s) + \log(1/\varepsilon), 2\epsilon)$ extractor.*

Proof: Let X be a random variable and $T \subseteq \{0,1\}^n$ such that

$$\left| \Pr_{X, U_u}[E(X, U_u) \in T] - \Pr_{U_n}[U_n \in T] \right| > 2\varepsilon.$$

Then, using the triangle inequality:

$$\Pr_X \left[\left| \Pr_{U_u}[E(X, U_u) \in T] - \Pr_{U_n}[U_n \in T] \right| \geq \varepsilon \right] > \varepsilon.$$

Since for every x such that $|\Pr_{U_u}[E(x, U_u) \in T] - \Pr_{U_n}[U_n \in T]| \geq \varepsilon$ there must exist an oracle circuit of size at most s such that $C^T = x$, the number of such x is bounded by the number of oracle circuits of size at most s. There are at most $2^{O(s \log s)}$ such circuits. Therefore X lands in a set of size at most $2^{O(s \log s)}$ with probability bigger than ε, and so $H_\infty(X) < O(s \log s) + \log(1/\varepsilon)$. ■

The following theorem states that constant depth circuits cannot compute good extractors for min-entropy $k \leq n^{1-\Omega(1)}$.

Theorem 6.4. *Fix a constant $\epsilon < 1$. Let $E : \{0,1\}^h \times \{0,1\}^u \to \{0,1\}^n$ be a (k, ϵ) extractor, with $u \leq n/4$, and let E be computable by a circuit of size g and depth d. Then:*

$$\log^{d-1} g \geq \Omega\left(\frac{h}{k}\right).$$

Before proving Theorem 6.4 note that, in combination with Theorem 6.3, it yields the following lower bound for black-box PRG constructions.

Corollary 6.5 (Formal version of Theorem 4.4). *Let $\varepsilon := 1/4$. Let G be a (l, s, ε) black-box PRG construction, and let G^f be computable in $ATIME(d,t)^f$. Then:*

$$t \geq \Omega\left(\frac{2^l}{s \log s}\right)^{1/(d+O(1))}.$$

In particular, for any $\gamma < 1$, there is no $(O(\log n), n^\gamma, \epsilon)$-black-box PRG construction computable in $ATIME(O(1), n^{o(1)})$.

Proof: By standard techniques [10], the $ATIME(d,t)^f$ computation can be carried out by a circuit of depth $d + O(1)$ and size $2^{O(t)}$, where we view the oracle as part of the input. The result then follows from Theorems 6.3 and 6.4. ■

To prove Theorem 6.4 we make use of the following fact about low noise sensitivity of constant depth circuits, which we deduce combining results in [19, 8, 30].

Lemma 6.6. *Let $C : \{0,1\}^n \to \{0,1\}$ be a circuit of size g and depth d. Let $X \in \{0,1\}^n$ be a random input and let $\tilde{X}$ be obtained from X by flipping each bit independently with probability $\delta < 1/2$. Then:*

$$\Pr_{X,\tilde{X}}[C(X) \neq C(\tilde{X})] \leq O(\delta \log^{d-1} g).$$

The proof of Lemma 6.6 requires a detour into Fourier analysis and therefore we defer it to the appendix.

The following easy lemma states that the amount of noise by which we perturb X corresponds to the min-entropy of $\tilde{X}$.

Lemma 6.7. *For any $x \in \{0,1\}^h$, let $\tilde{X}$ be obtained from x by flipping each bit independently with probability $k/h \leq 1/2$. Then $H_\infty(\tilde{X}) \geq \Omega(k)$.*

Proof: Since $k/h < 1/2$ it is easy to see that $H_\infty(\tilde{X}) \geq \log\left(1/\Pr[\tilde{X} = x]\right) = \log\left(1/(1-k/n)^n\right)$. The result then follows easily. ■

Proof of Theorem 6.4: For $z, z' \in \{0,1\}^n$ let $\Delta(z,z')$ denote the *relative* Hamming distance, i.e. $\Pr_i[z_i \neq z'_i]$. Let $E(x,y)_i$ denote the i-th bit of $E(x,y)$. Let X be chosen at random in $\{0,1\}^h$, and $\tilde{X}$ obtained from X flipping each bit independently with probability $O(k/h)$ so that $H_\infty(\tilde{X}) \geq k$ by Lemma 6.7.

We seek a test that distinguishes $E(\tilde{X}, U_u)$ from U_n. The main ideas are the following: For every seed y, we expect $\Delta(E(X,y), E(\tilde{X},y))$ to be 'small' by the low average sensitivity of constant depth circuits (Lemma 6.6). We can fix $X = x$ maintaining this property. Now we can tell whether a sample z comes from $E(\tilde{X}, U_u)$, rather than being truly random, checking whether there is a seed y such that $\Delta(E(x,y), z)$ is 'small'.

Fix a seed y and a position $i \in \{1, \ldots, n\}$. By Lemma 6.6:

$$\Pr_{X,\tilde{X}}[E(X,y)_i \neq E(\tilde{X},y)_i] \leq \frac{O(k \log^{d-1} g)}{h}.$$

By linearity of expectation:

$$\mathop{\mathrm{E}}_{X,\tilde{X},y}\Big[\Delta(E(X,y), E(\tilde{X},y))\Big] \leq \frac{O(k \log^{d-1} g)}{h}.$$

By averaging there must exist a fixed x such that

$$\mathop{\mathrm{E}}_{\tilde{X},y}\Big[\Delta(E(x,y), E(\tilde{X},y))\Big] \leq \frac{O(k \log^{d-1} g)}{h},$$

where $\tilde{X}$ is generated by flipping the bits of x. Pick a small constant $\xi > 0$. By Markov inequality:

$$\Pr_{\tilde{X},y}\Big[\Delta(E(x,y), E(\tilde{X},y)) \geq \xi\Big] \leq \frac{O(k \log^{d-1} g)}{h \cdot \xi}.$$

We are now ready to define the test that will distinguish the output of the extractor from U_n:

$$T := \Big\{z \in \{0,1\}^n : \exists y \in \{0,1\}^u \;\; \Delta(E(x,y), z) \leq \xi\Big\}.$$

By what we have said above:

$$\Pr_{\tilde{X},y}[E(\tilde{X},y) \in T] \geq 1 - \frac{O(k \log^{d-1} g)}{h \cdot \xi}.$$

We now show that a truly random sample will pass the test with very low probability. Fix a seed y.

$$\Pr_{U_n}\Big[\Delta(E(x,y), U_n) \leq \xi\Big] \leq \frac{V_n(\xi)}{2^n} \leq \frac{2^{H(\xi)n}}{2^n} \leq 2^{-n/2}.$$

Where $V_n(\xi)$ is the size of a Hamming ball in $\{0,1\}^n$ of radius ξn, $H(x) = -x \log x - (1-x)\log(1-x)$ is the binary entropy function, and $H(\xi) < 1/2$ for sufficiently small ξ. Since there are 2^u seeds, using $u \leq n/4$ and a union bound:

$$\Pr_{U_n}[U_n \in T] \leq 2^u \cdot 2^{-n/2} \leq 2^{-n/4} = o(1).$$

Since $H_\infty(\tilde{X}) \geq k$, and E is a (k, ϵ) extractor, we have:

$$1 - \frac{O(k \log^{d-1} g)}{h \cdot \xi} \leq \epsilon + o(1).$$

■

6.1. Tightness

In this section we study in more detail the consequences of Corollary 6.5. Recall that it established the following tradeoff for a (l, s, ε) black-box PRG construction G such that G^f is computable in $ATIME(d, t)^f$:

$$t \geq \Omega\left(\frac{2^l}{s \log s}\right)^{1/(d+O(1))}.$$

We investigate what happens in the following two cases:

- The PRG construction is computable in $ATIME(O(1), l)$.
- The PRG construction is based on the existence of a function $f : \{0,1\}^l \to \{0,1\}$ that is $(2^{\Omega(l)}, 1)$-hard for CKT.

Note that to obtain a result analogous to Theorem 3.6 for $ATIME(O(1), \log n)$ one needs both the above items.

PRG construction computable in $ATIME(O(1), l)$: If one wants $t = O(l)$ then $s \geq 2^l/l^{O(1)}$. In particular, the function $f : \{0,1\}^l \to \{0,1\}$ we start with must be hard for circuits of size at least $2^l/l^{O(1)}$. However, the next easy proposition shows that for such big sizes mild average-case and worst-case hardness *collapse*! Consequently, under such an assumption no worst-case to average-case hardness amplification is needed, and to get a PRG one could apply directly Theorem 4.3 or Theorem 4.7.

We state the next proposition for both CKT and AC_0 since in Section 8 we discuss derandomization under hardness assumptions for AC_0.

Proposition 6.8. *There is a constant k such that if $f : \{0,1\}^l \to \{0,1\}$ is $(k \cdot 2^l/l^{c-1}, 1)$-hard for CKT (respectively, $AC_0[d+k]$) then f is $(2^l/l^c, 1 - 1/l^c)$-hard for CKT (respectively, $AC_0[d]$).*

Proof: Suppose not. Let C be a circuit of size at most $\frac{2^l}{l^c}$ (and depth d) such that

$$\Pr[C(U_l) \neq f(U_l)] < \frac{1}{l^c}.$$

Then there are at most $\frac{2^l}{l^c}$ inputs x such that $C(x) \neq f(x)$. We can build a circuit C' of size at most $2l \cdot \frac{2^l}{l^c}$ such that, given x, decides whether $C(x) \neq f(x)$ (recall our size measure is the number of edges). C' does a simple lookup table: For every x such that $C(x) \neq f(x)$ there is an AND gate with l connections to the corresponding input bits (or their negations). After this layer of AND gates we put an OR gate with $\frac{2^l}{l^c}$ connections. It is easy to see that such a circuit correctly decides whether $C(x) \neq f(x)$ and has size at most $2 \cdot \frac{2^l}{l^{c-1}}$ (and depth 2).

Combining C and C' with a XOR we obtain a circuit of size at most $4 \cdot \frac{2^l}{l^{c-1}}$ (and depth $d+3$) computing f everywhere. Contradiction (for $k = 4$). ■

PRG construction based on $f : \{0,1\}^l \to \{0,1\}$ that is $(2^{\Omega(l)}, 1)$-hard: If one wants $s = 2^{\epsilon l}$ then $t \geq 2^{\Omega(l/d)}$. We now prove that these resources are also sufficient. Our approach is showing that they allow for computing worst-case to mild average-case hardness amplification. One can then obtain a PRG construction from worst-case hard functions combining this hardness amplification with the construction in Theorem 4.3.

To show that these resources are sufficient for computing worst-case to mild average-case hardness amplification we examine the construction in Theorem 3.3. First we show that *one* parity quantifier is sufficient for it, then we note that this parity quantifier can be simulated in $ATIME(d, 2^{O(l/d)})$.

Theorem 6.9. *If there is a function $f : \{0,1\}^l \to \{0,1\}$ in $ATIME(O(1), l)$ that is $(2^{\Omega(l)}, 1)$-hard for CKT, then there is a function $f' \in \oplus \cdot ATIME(O(1), l)$ which is $(2^{\Omega(l)}, 1 - 1/\mathrm{poly}(l))$-hard for CKT.*

Proof: We use the same construction in Theorem 3.3. Let us recall it. Fix a field F of size $4l^2$. Let H be the first (lexicographically) $\sqrt{|F|}$ elements of F.

Let $k := l/\log|H|$, so that f can be seen as mapping H^k to $\{0,1\}$. Let $\hat{f} : F^k \to F$ be:

$$\hat{f}(x_1, \ldots, x_k) := \sum_{h_1,\ldots,h_k \in H} f(h_1, \ldots, h_k)\delta_{h_1}(x_1) \cdot \cdots \cdot \delta_{h_k}(x_k)$$

where for $h \in H$ and $x \in F$,

$$\delta_h(x) := \prod_{h' \in H, h' \neq h} \frac{x - h'}{h - h'}.$$

As pointed out in [4] (see also [1]), $\hat{f}$ has the required hardness, but $\hat{f}$ is not yet our final function since it is not boolean. Define $f'(x, i) := \hat{f}(x)_i$, note $|i| = O(\log l)$. It is easy to see that this final transformation preserves mild hardness.

Thus we only need to show that $f' \in \oplus \cdot ATIME(O(1), l)$.

First we show that given $h \in H, x \in F$ we can compute $\delta_h(x)$ in $ATIME(O(1), l)$. To compute $\delta_h(x)$ we need to perform $\mathrm{poly}(l)$ field operations. Note that the field F can be found, and operated with, in time $\mathrm{poly}\log l^{O(1)} = \mathrm{poly}\log l$ [34]. Moreover, we can use the same space for all the field operations, for a total of $\mathrm{poly}\log l$ space. By Theorem 2.2 we can compute $\delta_h(x)$ in $ATIME(O(1), l)$.

Similarly, given $h_1, \ldots, h_k, x_1, \ldots, x_k$, we can compute $f(h_1, \ldots, h_k) \cdot \delta_{h_1}(x_1) \cdot \cdots \cdot \delta_{h_k}(x_k)$ in $ATIME(O(1), l)$. In fact, $f \in ATIME(O(1), l)$ by assumption and the total time to compute $\delta_{h_1}(x_1) \cdot \cdots \cdot \delta_{h_k}(x_k)$ is still $\mathrm{poly}(l)$, and moreover we can reuse the same space for the δ's. So by Theorem 2.2 this product can be computed in $ATIME(O(1), l)$.

What is left to do is to sum over all 2^l possible $h_1, \ldots, h_k$. Now note we can assume that the characteristic of F is 2, so addition equals XOR. In particular, the i-th output bit of f' is the parity of the i-th bit of $f(h_1, \ldots, h_k)\delta_{h_1}(x_1) \cdot \cdots \cdot \delta_{h_k}(x_k)$ over all 2^l possible $h_1, \ldots, h_k$. Thus $f' \in \oplus \cdot ATIME(O(1), l)$. ■

Note that the parity quantifier in the above computation ranges over 2^l bits. The following easy lemma states that this parity can be computed in $ATIME(d, 2^{O(l/d)})$.

Lemma 6.10. *For every integer $d \geq 1$, the parity of n bits is in $ATIME(d, n^{O(1/d)})$.*

Proof Sketch: The idea is trading alternations for time taking advantage of the associativity of parity. Namely, we partition the input into $n^{O(1/d)}$ pieces, existentially guess the parity of each, and universally verify that each guess is correct (recursively with the same algorithm). ■

Therefore $ATIME(d + O(1), 2^{O(l/d)})$ is essentially necessary and sufficient for PRG constructions from worst-case hard functions.

Combining Theorems 4.3 and 6.9 we obtain Theorem 4.6.

Theorem 4.6 (restated). *If there is a function $f : \{0,1\}^l \to \{0,1\}$ in $A\oplus TIME(O(1), l)$ that is $(2^{\Omega(l)}, 1)$-hard for CKT then there is a n-PRG against CKT with seed length $O(\log n)$ and computable in $A\oplus TIME(O(1), \log n)$, and $BP \cdot A\oplus TIME(O(1), \log n) = A\oplus TIME(O(1), \log n)$.*

7. Worst-Case Hardness Amplification

In this section we discuss worst-case hardness amplification. In particular, we prove a formal version of Theorem 4.5, establishing a lower bound for black-box worst-case hardness amplifications. As mentioned in the introduction, a certain lower bound for black-box worst-case hardness amplifications already follows from our previous results. Namely, if there is a black-box worst-case hardness amplification then combining this with our black-box PRG construction from mild average-case hardness (Theorem 4.3) one gets a black-box PRG construction from worst-case hardness, and the lower bound in Corollary 6.5 applies. In this section we give a direct proof of a lower bound for black-box worst-case hardness amplification. This direct proof yields a more general lower bound than what one can get using the above approach.

The general ideas in our lower bound are the same we employed in our lower bound for black-box PRG constructions in Section 6, with the exception that 'extractors' will be replaced with '*list-decodable codes*': First we show that every black-box hardness amplification gives rise to a 'good' list-decodable code. Then we show that 'good' list-decodable codes are very sensitive to noise. Since constant depth circuits are not, we get our lower bound.

We now proceed to turn the above sketch into a formal proof.

Definition 7.1. *An oracle algorithm $Amp : \{0,1\}^{l'} \to \{0,1\}$ is a (l, δ, s)-black-box worst-case hardness amplification if for every $f : \{0,1\}^{l} \to \{0,1\}$ and for every $A : \{0,1\}^{l'} \to \{0,1\}$ such that*

$$\Pr[A(U_{l'}) = Amp^f(U_{l'})] \geq \delta,$$

there is an oracle circuit C of size at most s such that $C^A(x) = f(x)$.

Note in the above definition we did not specify the type of the circuit C (e.g. $CKT, AC_0, \ldots$) because it does not play a role in this section. Also note that, if Amp is a (l, δ, s)-black-box worst-case hardness amplification, then for every function $f : \{0,1\}^l \to \{0,1\}$ we have that if f is $(s', 1)$-hard then Amp^f is $(s'/s, \delta)$-hard.

We note that most of the known hardness amplification techniques are black-box. In particular, all those described in Section 3.1 are. (While we have only defined black-box worst-case to average-case hardness amplification, it is easy to extend the definition to the setting of average-case to average-case hardness amplification.) For example, in this notation the hardness amplification in Theorem 3.3 is a $(l, 1 - 1/\mathrm{poly}(l), \mathrm{poly}(l))$-black-box hardness amplification. It should also be noted that in this hardness amplification the input length increases only by a constant factor, i.e. $Amp^f : \{0,1\}^{O(l)} \to \{0,1\}$. While this is what one needs for high-end derandomization [18], our lower bound applies regardless of this.

We give the definition of list-decodable codes:

Definition 7.2. *A code $C : \{0,1\}^n \to \{0,1\}^{\bar{n}}$ is* (δ, ρ)-list-decodable *if for every $\bar{x} \in \{0,1\}^{\bar{n}}$:*

$$\left| \left\{ y \in \{0,1\}^n : \Delta(\bar{x}, C(y)) \leq \delta \right\} \right| \leq \rho$$

where Δ is the relative Hamming distance: $\Delta(\bar{x}, \bar{y}) := \Pr_i[\bar{x}_i \neq \bar{y}_i]$. We refer to $x \in \{0,1\}^n$ as messages *and to $C(x), x \in \{0,1\}^n$, as* codewords.

Let Amp be a black-box worst-case hardness amplification. The following lemma, implicit in [36, 39], states that if we consider the truth table of a function f as a message and the truth table of Amp^f as a codeword, then Amp can be seen as an encoding algorithm.

Lemma 7.3. *Let Amp be a (l, δ, s)-black-box worst-case hardness amplification. Then $Enc : \{0,1\}^{2^l} \to \{0,1\}^{\bar{n}}$ defined as $Enc(f) := Amp^f$ is $(1 - \delta, 2^{O(s \cdot \log s)})$-list-decodable.*

Proof: Consider $A \in \{0,1\}^{\bar{n}}$. By definition of hardness amplification, for every f such that $\Pr_{x\in\{0,1\}^{\bar{n}}}[A(x) = Amp^f(x)] \geq \delta$, there is an oracle circuit C of size at most s such that $C^A(x) = f(x)$. Therefore the number of such codewords is bounded by the number of oracle circuits. Noting that there are at most $2^{O(s\cdot\log s)}$ oracle circuits of size at most s, and that $\Pr_{x\in\{0,1\}^{\bar{n}}}[A(x) = Amp^f(x)] = 1 - \Delta(A, Amp^f)$, completes the proof. ■

The following theorem states that constant depth circuits cannot compute list-decodable codes even for very weak parameters.

Theorem 7.4. *There is a constant $\gamma, 0 < \gamma < 1$, such that the following holds. Let $C : \{0,1\}^n \to \{0,1\}^{\bar{n}}$ be a $(\delta, 2^m)$-list-decodable code, with $m \leq \gamma n$. If C can be computed by a circuit of size g and depth d, then*

$$\log^{d-1} g \geq \Omega\left(\frac{n\delta}{m}\right).$$

Before proving Theorem 7.4 note that, in combination with Lemma 7.3, it yields the following lower bound for black-box hardness amplification.

Corollary 7.5 (Formal version of Theorem 4.5). *Suppose Amp is a $(l, 1-\delta, s)$ black-box hardness amplification, and suppose that Amp^f is in $ATIME(d,t)^f$. Then:*

$$t \geq \Omega\left(\frac{2^l\delta}{s\log s}\right)^{1/(d+O(1))}.$$

In particular, for any constants $c > 0, \epsilon < 1$, there is no $(l, 1 - 1/l^c, 2^{l^\epsilon})$-black-box worst-case hardness amplification computable in $ATIME(O(1), 2^{o(l)})$.

Note that Corollary 7.5 implies that there is no black-box worst-case hardness amplification in the polynomial time hierarchy.

Proof: By standard techniques [10], the $ATIME(d,t)^f$ computation can be carried out by a circuit of depth $d + O(1)$ and size $2^{O(t)}$, where we view the oracle as part of the input. The result then follows from Lemma 7.3 and Theorem 7.4. ■

Remark 7.6. Corollary 7.5 is tight in the same way as Corollary 6.5: The only settings of parameters that are not ruled out either allow for the construction in Theorem 6.9, or else correspond to hardness assumptions so strong that worst-case hardness and average-case hardness collapse, and therefore worst-case hardness amplification is vacuous (see Section 6.1).

We now prove Theorem 7.4. The proof is very similar to the proof of Theorem 6.4, and again makes use of Lemma 6.6 and Lemma 6.7.

Proof of Theorem 7.4: Let $C_i(x)$ denote the i-th bit of $C(x)$. Let X be chosen at random in $\{0,1\}^n$, and $\tilde{X}$ obtained from X flipping each bit independently with probability $O(m/n)$ so that $H_\infty(\tilde{X}) \geq m + 1$ by Lemma 6.7.

The idea in the proof is to consider the quantity

$$\Pr_{i,X,\tilde{X}}[C_i(X) \neq C_i(\tilde{X})]$$

and to bound it using (1) the assumption that C is $(\delta, 2^m)$-list-decodable and (2) the low average sensitivity of AC_0 circuits (Lemma 6.6).

For every x, the list-decodability assumption tells us that there are at most 2^m messages whose codewords are at distance at most δ from $C(x)$. Fix any such message. Since $H_\infty(\tilde{X}) \geq m+1$, the probability that $\tilde{X}$ is equal to this message is at most $2^{-(m+1)}$. Therefore, by a union bound:

$$\Pr_{X,\tilde{X}}[\Delta(C(X), C(\tilde{X})) \leq \delta] \leq 2^m \cdot 2^{-(m+1)} = 1/2.$$

Therefore:

$$\begin{aligned} \Pr_{i,X,\tilde{X}}[C_i(X) \neq C_i(\tilde{X})] &\geq \Pr_{i,X,\tilde{X}}[C_i(X) \neq C_i(\tilde{X}) | \Delta(C(X), C(\tilde{X})) > \delta] \cdot \Pr_{X,\tilde{X}}[\Delta(C(X), C(\tilde{X})) > \delta] \\ &\geq \delta \cdot \frac{1}{2}. \end{aligned} \tag{1}$$

On the other hand, by Lemma 6.6 we have

$$\Pr_{i,X,\tilde{X}}[C_i(X) \neq C_i(\tilde{X})] \leq m \cdot \frac{O(\log^{d-1} g)}{n}. \tag{2}$$

The theorem follows putting together Bounds (1) and (2). ■

8. Derandomization from Weaker Assumptions

In this section we work on relaxing the hardness assumptions needed in our derandomization results. In particular, we prove Theorems 4.7 and 4.8.

We start with the latter. As explained in Section 3.2, Agrawal [1] notices that all the proofs of correctness of the constructions described in Section 3.1 carry through against TC_0 circuits. Combining this with Theorem 4.6, and recalling that counting quantifiers can simulate parity quantifiers, one gets the following theorem.

Theorem 4.8 (restated). *There is a constant c such that if there is a function $f : \{0,1\}^l \to \{0,1\}$ in $CTIME(O(1), l)$ that is $(2^{\Omega(l)}, 1)$-hard for $TC_0[c+d]$, then there is a n-PRG against $TC_0[d]$ with seed length $O(\log n)$ and computable in $CTIME(O(1), \log n)$.*

In particular, if for every d there is a function $f : \{0,1\}^l \to \{0,1\}$ in $CTIME(O(1), l)$ that is $(2^{\Omega(l)}, 1)$-hard for $TC_0[d]$, then $BP \cdot CTIME(O(1), \log n) = CTIME(O(1), \log n)$.

Now we focus on Theorem 4.7. For proving it we use the same construction in Theorem 3.10, but without the worst-case to average-case hardness amplification step in Theorem 3.3. The correctness of this construction has already been proved in [1], so we only need to show that has an implementation in $ATIME(O(1), \log n)$.

Recall this construction combined the conditional PRG from Section 3.1 with Nisan's unconditional PRG (Theorem 3.9). We have already shown in Section 5 how to compute the conditional PRG. So it is only left to discuss Nisan's unconditional PRG. In particular, we need to show that the following items are computable in $ATIME(O(1), \log n)$:

- Family of $(\log n, \log^c n)$ designs of size n over a universe of size $\log^d n$, for every given value of $c \geq 1$ and some $d \geq c$.
- *parity* over $\log^c n$ bits, for every given value of c.

The result about parity has been proved in Lemma 6.10. We now show that the design construction in [27] is computable in $ATIME(O(1), \log n)$.

Lemma 8.1. *For every constant c there is a constant d such that there is a family $\{D_n\}$ of $(\log n, \log^c n)$ designs of size n over a universe of size $\log^d n$ with the following property: There is a machine in $ATIME(O(1), \log n)$ such that, given n and $k \leq n$, computes the characteristic vector of the k-th set in D_n.*

Proof: Let $l := \log n$. Let us first recall the construction in [27]. Let l^c be the cardinality of a field F. Let $d := 2c$, i.e. the universe size is $|F|^2 = l^d$. Given a string i of length l, we view the string as the coefficients of a univariate polynomial $\hat{i}$ with coefficients in F. The corresponding set is

$$S_i := \{a \circ \hat{i}(a) : a \in F\}.$$

It is pointed out in [27] that $S_1, \ldots, S_n$ is a (l, l^c) design. Thus we only need to show that it is computable in $ATIME(O(1), l)$.

Our task is, given k and an element j of the universe, decide whether $j \in S_k$ in $ATIME(O(1), l)$. Let $j|_{c \log l}$ be the first $c \log l$ bits of j. Now, $j \in S_k$ if and only if $j = j|_{c \log l} \circ \hat{k}(j|_{c \log l})$. To compute $\hat{k}(j|_{c \log l})$ we need to perform $\mathrm{poly}(l)$ field operations. Note that the field F can be found, and operated with, in time $\mathrm{poly} \log l^{O(1)} = \mathrm{poly} \log l$ [34]. Moreover, we can use the same space for all the field operations, for a total of $\mathrm{poly} \log l$ space. Consequently, we can decide whether $j \in S_k$ in $ATIME(O(1), l)$ by Theorem 2.2. ■

This completes the proof of Theorem 4.7.

Theorem 4.7 (restated). *There is a constant c such that if there is a function $f : \{0,1\}^l \to \{0,1\}$ in $ATIME(O(1), l)$ that is $(2^{\Omega(l)}, 1 - 1/l^b)$-hard for $AC_0[c \cdot \max(b, d)]$, then there is a $(n, 1/\log^{O(1)} n)$-PRG against $AC_0[d]$ with logarithmic seed length and computable in $ATIME(O(1), \log n)$.*

In particular, if there is a constant b such that for every d there is a function in $ATIME(O(1), l)$ that is $(2^{\Omega(l)}, 1 - 1/l^b)$-hard for $AC_0[d]$, then $BP \cdot ATIME(O(1), \log n) = ATIME(O(1), \log n)$.

9. Open Problems

Notice that gap between Theorems 4.6 and 4.8. While *parity* quantifiers are sufficient for a PRG construction from a worst-case hard function, we seem to need hardness against circuits with *majority* gates for proving its correctness. It is true that one can still derandomize $BP \cdot AC_0$ from hardness assumptions for AC_0, using the construction in Theorem 3.10. However, this construction requires a strong unconditional PRG, and no such result is known for AC_0 with parity gates. Can one derandomize $BP \cdot AC_0$ with parity gates from a worst-case hardness assumption for AC_0 with parity gates? Essentially the same question was independently raised by Eric Allender and Sambuddha Roy (personal communication, Nov. 2002).

Can we, under some complexity assumption for AC_0, build a $(n, 1/n)$-PRG against AC_0 with seed length $O(\log n)$? (Theorem 3.10 and Theorem 4.7 only give a $(n, 1/\log^{O(1)} n)$-PRG.)

10. Acknowledgements

Adam Klivans pointed out [1] to us. Madhu Sudan suggested the sensitivity approach for proving lower bounds for AC_0. Thanks to Ronen Shaltiel for a helpful conversation. Many thanks to Salil Vadhan for encouragement, illuminating discussions and suggesting the problem. We also thank Oded Goldreich and the anonymous referees for the useful comments.

References

[1] M. Agrawal. Hard sets and pseudo-random generators for constant depth circuits. In *Twenty First Foundations of Software Technology and Theoretical Computer Science, December 13-15, Bangalore, India*, pages 58–69. 2001.

[2] M. Ajtai. Approximate counting with uniform constant-depth circuits. In *Advances in computational complexity theory (New Brunswick, NJ, 1990)*, pages 1–20. Amer. Math. Soc., Providence, RI, 1993.

[3] E. W. Allender and K. W. Wagner. Counting hierarchies: Polynomial time and constant depth circuits. *Bulletin of the European Association for Theoretical Computer Science*, 40:182–194, 1990.

[4] L. Babai, L. Fortnow, N. Nisan, and A. Wigderson. BPP has subexponential time simulations unless EXPTIME has publishable proofs. *Computational Complexity*, 3(4):307–318, 1993.

[5] Z. Bar-Yossef, O. Reingold, R. Shaltiel, and L. Trevisan. Streaming through combinatorial objects. In *Seventeenth Annual IEEE Conference on Computational Complexity*. IEEE Computer Soc., Los Alamitos, CA, 2002.

[6] D. A. M. Barrington, N. Immerman, and H. Straubing. On uniformity within NC^1. *J. Comput. System Sci.*, 41(3):274–306, 1990.

[7] M. Blum and S. Micali. How to generate cryptographically strong sequences of pseudo-random bits. *SIAM J. on Computing*, 13(4):850–864, Nov. 1984.

[8] R. B. Boppana. The average sensitivity of bounded-depth circuits. *Inform. Process. Lett.*, 63(5):257–261, 1997.

[9] M. Cryan and P. B. Miltersen. On pseudorandom generators in NC^0. In *26th International Symposium on Mathematical Foundations of Computer Science (MFCS 01)*, pages 272–284. Springer-Verlag, 2001.

[10] M. L. Furst, J. B. Saxe, and M. Sipser. Parity, circuits, and the polynomial-time hierarchy. *Mathematical Systems Theory*, 17(1):13–27, April 1984.

[11] O. Goldreich. A sample of samplers - a computational perspective on sampling (survey). *Electronic Colloquium on Computational Complexity (ECCC)*, 4(020), 1997.

[12] T. Hartman and R. Raz. On the distribution of the number of roots of polynomials and explicit logspace extractors. In *Proceedings of the Fourth International Workshop on Randomization and Approximation Techniques in Computer Science*, pages 3–22, July 14 2000.

[13] J. Hastad. *Computational Limitations for Small Depth Circuits*. PhD thesis, M.I.T., 1986.

[14] R. Impagliazzo. Hard-core distributions for somewhat hard problems. In *36th Annual Symposium on Foundations of Computer Science*, pages 538–545, Milwaukee, Wisconsin, 23–25 Oct. 1995. IEEE.

[15] R. Impagliazzo and M. Naor. Effi cient cryptographic schemes provably as secure as subset sum. *Journal of Cryptology: the journal of the International Association for Cryptologic Research*, 9(4):199–216, Fall 1996.

[16] R. Impagliazzo, R. Paturi, and F. Zane. Which problems have strongly exponential complexity? *J. Computer & Systems Sciences*, 63(4):512–530, Dec 2001.

[17] R. Impagliazzo, R. Shaltiel, and A. Wigderson. Extractors and pseudo-random generators with optimal seed length. In *Proceedings of the Thirty-second Annual ACM Symposium on Theory of Computing*, pages 1–10, Portland, Oregon, May 2000. See also ECCC TR00-009.

[18] R. Impagliazzo and A. Wigderson. $P = BPP$ if E requires exponential circuits: Derandomizing the XOR lemma. In *Proceedings of the Twenty-Ninth Annual ACM Symposium on Theory of Computing*, pages 220–229, El Paso, Texas, 4–6 May 1997.

[19] J. Kahn, G. Kalai, and N. Linial. The influence of variables on Boolean functions (extended abstract). In *29th Annual Symposium on Foundations of Computer Science*, pages 68–80, White Plains, New York, 24–26 Oct. 1988. IEEE.

[20] M. Kharitonov, A. V. Goldberg, and M. Yung. Lower bounds for pseudorandom number generators. In *30th Annual Symposium on Foundations of Computer Science*, pages 242–247, Research Triangle Park, North Carolina, 30 Oct.–1 Nov. 1989. IEEE.

[21] A. R. Klivans. On the derandomization of constant depth circuits. In *Proceedings of the Fifth International Workshop on Randomization and Approximation Techniques in Computer Science*, August 18–20 2001.

[22] A. R. Klivans and D. van Melkebeek. Graph nonisomorphism has subexponential size proofs unless the polynomial-time hierarchy collapses. In *Annual ACM Symposium on Theory of Computing (Atlanta, GA, 1999)*, pages 659–667 (electronic). ACM, New York, 1999.

[23] N. Linial, Y. Mansour, and N. Nisan. Constant depth circuits, Fourier transform, and learnability. *J. Assoc. Comput. Mach.*, 40(3):607–620, 1993.

[24] A. Lubotzky, R. Phillips, and P. Sarnak. Ramanujan graphs. *Combinatorica*, 8(3):261–277, 1988.

[25] M. Naor and O. Reingold. Number-theoretic constructions of effi cient pseudo-random functions. In *38th Annual Symposium on Foundations of Computer Science*, pages 458–467, Miami Beach, Florida, 20–22 Oct. 1997. IEEE.

[26] V. Nepomnjaščii. Rudimentary predicates and turing calculations. *Soviet Mathematics-Doklady*, 11(6):1462–1465, 1970.

[27] N. Nisan. Pseudorandom bits for constant depth circuits. *Combinatorica*, 11(1):63–70, 1991.

[28] N. Nisan and A. Wigderson. Hardness vs randomness. *J. Comput. Syst. Sci.*, 49(2):149–167, Oct. 1994.

[29] N. Nisan and D. Zuckerman. Randomness is linear in space. *J. Comput. Syst. Sci.*, 52(1):43–52, Feb. 1996.
[30] R. O'Donnell. Hardness amplification within NP. In *Proceedings of the 34th Annual ACM Symposium on Theory of Computing*, pages 751–760. ACM, May 2002.
[31] R. Raz, O. Reingold, and S. Vadhan. Extracting all the randomness and reducing the error in Trevisan's extractors. In *Annual ACM Symposium on Theory of Computing (Atlanta, GA, 1999)*, pages 149–158 (electronic). ACM, New York, 1999.
[32] R. Shaltiel. Recent developments in explicit constructions of extractors. *Bulletin of the European Association for Theoretical Computer Science*, 77:182–194, 2002.
[33] R. Shaltiel and C. Umans. Simple extractors for all min-entropies and a new pseudo-random generator. In *42nd Annual Symposium on Foundations of Computer Science*. IEEE, 14–17 Oct. 2001.
[34] V. Shoup. New algorithms for finding irreducible polynomials over finite fields. *Math. Comp.*, 54(189):435–447, 1990.
[35] M. Sipser. Borel sets and circuit complexity. In *Proceedings of the Fifteenth Annual ACM Symposium on Theory of Computing*, pages 61–69, Boston, Massachusetts, 25–27 Apr. 1983.
[36] M. Sudan, L. Trevisan, and S. Vadhan. Pseudorandom generators without the XOR lemma. *J. Comput. System Sci.*, 62(2):236–266, 2001. Special issue on the Fourteenth Annual IEEE Conference on Computational Complexity (Atlanta, GA, 1999).
[37] J. Torán. *Structural properties of the counting hierarchies*. PhD thesis, Facultat d'Informatica de Barcelona, Barcelona, Spain, 1988.
[38] L. Trevisan. Extractors and pseudorandom generators. *Journal of the ACM*, 48(4):860–879, 2001.
[39] L. Trevisan and S. Vadhan. Pseudorandomness and average-case complexity via uniform reductions. In *Seventeenth Annual IEEE Conference on Computational Complexity*, pages 129–138. IEEE Computer Soc., Los Alamitos, CA, 2002.
[40] C. Umans. Pseudo-random generators for all hardnesses. In *Proceedings of the 34th ACM Symposium on Theory of Computing*, pages 627–634. ACM Press, 2002.
[41] H. Vollmer. *Introduction to circuit complexity*. Springer-Verlag, Berlin, 1999.
[42] K. W. Wagner. The complexity of combinatorial problems with succinct input representation. *Acta Inform.*, 23(3):325–356, 1986.
[43] A. C. Yao. Theory and applications of trapdoor functions (extended abstract). In *23rd Annual Symposium on Foundations of Computer Science*, pages 80–91, Chicago, Illinois, 3–5 Nov. 1982. IEEE.
[44] X. Yu and M. Yung. Space lower-bounds for pseudorandom-generators. In *Ninth Annual Structure in Complexity Theory Conference*, pages 186–197. IEEE Computer Soc., Los Alamitos, CA, 1994.

A Noise sensitivity of AC_0

In this section we prove Lemma 6.6.

Lemma 6.6 (restated). *Let $C : \{0,1\}^n \to \{0,1\}$ be a circuit of size g and depth d. Let $X \in \{0,1\}^n$ be a random input and let $\tilde{X}$ be obtained from X by flipping each bit independently with probability $\delta < 1/2$. Then:*

$$\Pr_{X,\tilde{X}}[C(X) \neq C(\tilde{X})] \leq O(\delta \log^{d-1} g).$$

Although it is well known that constant depth circuits have small noise sensitivity, the bound we need is not stated anywhere, and to prove it we need to introduce the Fourier machinery and then combine several results.

We now set up the usual Fourier machinery, see e.g. [23] for details. Whenever we discuss Fourier coefficients, we will use $\{+1,-1\}$ instead of $\{0,1\}$. Let $f : \{+1,-1\}^n \to \{+1,-1\}$ be any boolean function. Then f has a unique representation as a multilinear polynomial in $x_1, \ldots, x_n$ of total degree at most n. The S-th Fourier coefficient of f, denoted $\hat{f}(S)$, is the coefficient of the monomial $\prod_{i \in S} x_i$ in this polynomial. We also have, by Parseval's identity:

$$\sum_{S \subseteq [n]} \hat{f}(S)^2 = 1,$$

where $[n] := \{1, \ldots, n\}$.

The first result we need is a characterization of the noise sensitivity of a function in terms of its Fourier coefficients. Such a characterization is given by O'Donnell [30].

Lemma A.1 ([30]). *Let $f : \{+1,-1\}^n \to \{+1,-1\}$ be any boolean function. Let $X \in \{+1,-1\}^n$ be a random input and let $\tilde{X}$ be obtained from X by flipping each bit independently with probability $\delta < 1/2$. Then:*

$$\Pr_{X,\tilde{X}}[f(X) \neq f(\tilde{X})] = \frac{1}{2} - \frac{1}{2} \sum_{S \subseteq [n]} (1-2\delta)^{|S|} \hat{f}(S)^2.$$

The second result we need is a bound on the Fourier coefficients of functions computed by constant depth circuits. In [8], Boppana gives a tight bound on the *average sensitivity* of constant depth circuits. Combining this bound with the characterization of average sensitivity in terms of Fourier coefficients given by Kahn, Kalai and Linial [19] (see also [23]), we obtain the following bound.

Lemma A.2. *[19, 8] Let f be computable by a circuit of size g and depth d. Then*

$$\sum_{S \subseteq [n]} |S| \hat{f}(S)^2 \leq O(\log^{d-1} g).$$

We can now prove Lemma 6.6.

Proof of Lemma 6.6: Let $f : \{+1,-1\}^n \to \{+1,-1\}$ be the function computed by C. Then:

$$\begin{aligned}
\Pr_{X,\tilde{X}}[C(X) \neq C(\tilde{X})] &= \frac{1}{2} - \frac{1}{2} \sum_{S \subseteq [n]} (1-2\delta)^{|S|} \hat{f}(S)^2 \qquad \text{(by Lemma A.1)} \\
&\leq \frac{1}{2} - \frac{1}{2} \sum_{S \subseteq [n]} (1-2\delta|S|) \hat{f}(S)^2 \qquad \text{(by Bernoulli's inequality)} \\
&= \frac{1}{2} \sum_{S \subseteq [n]} 2\delta|S| \hat{f}(S)^2 \qquad \text{(by Parseval's identity)} \\
&\leq O(\delta \log^{d-1} g) \qquad \text{(by Lemma A.2)}
\end{aligned}$$

■

ECCC ISSN 1433-8092
http://www.eccc.uni-trier.de/eccc
ftp://ftp.eccc.uni-trier.de/pub/eccc
ftpmail@ftp.eccc.uni-trier.de, subject 'help eccc'