Using Nondeterminism to Amplify Hardness*

Alexander Healy †　　Salil Vadhan ‡　　Emanuele Viola §

Division of Engineering & Applied Sciences
Harvard University
Cambridge, Massachusetts

August 12, 2005

Abstract

We revisit the problem of hardness amplification in $\mathcal{NP}$, as recently studied by O'Donnell (JCSS '04). We prove that if $\mathcal{NP}$ has a balanced function f such that any circuit of size $s(n)$ fails to compute f on a $1/\operatorname{poly}(n)$ fraction of inputs, then $\mathcal{NP}$ has a function f' such that any circuit of size $s'(n) = s(\sqrt{n})^{\Omega(1)}$ fails to compute f' on a $1/2 - 1/s'(n)$ fraction of inputs. In particular,

1. If $s(n) = n^{\omega(1)}$, we amplify to hardness $1/2 - 1/n^{\omega(1)}$.
2. If $s(n) = 2^{n^{\Omega(1)}}$, we amplify to hardness $1/2 - 1/2^{n^{\Omega(1)}}$.
3. If $s(n) = 2^{\Omega(n)}$, we amplify to hardness $1/2 - 1/2^{\Omega(\sqrt{n})}$.

These improve the results of O'Donnell, which amplify to $1/2 - 1/\sqrt{n}$. O'Donnell also proved that no construction of a certain general form could amplify beyond $1/2 - 1/n$. We bypass this barrier by using both *derandomization* and *nondeterminism* in the construction of f'.

We also prove impossibility results demonstrating that both our use of nondeterminism and the hypothesis that f is balanced are necessary for "black-box" hardness amplification procedures (such as ours).

Keywords average-case complexity, hardness amplification, pseudorandom generators for space-bounded computation, noise stability.

---

*An extended abstract of this paper appeared in *STOC 2004* [HVV].

†Email: `ahealy@fas.harvard.edu`. Research supported in part by NSF grant CCR-0205423 and a Sandia Fellowship.

‡Email: `salil@eecs.harvard.edu` Research supported by NSF grant CCR-0133096, ONR grant N00014-04-1-0478, a Sloan Research Fellowship, and US-Israel BSF grant 2002246. Work done in part while a fellow at the Radcliffe Institute for Advanced Study at Harvard University.

§Email: `viola@eecs.harvard.edu`. Research supported by NSF grant CCR-0133096 and US-Israel BSF grant 2002246.

1 Introduction

Average-case complexity is a fundamental topic in complexity theory, whose study has at least two distinct motivations. On one hand, it may provide more meaningful explanations than worst-case complexity about the intractability of problem instances actually encountered in practice. On the other hand, it provides us with methods to generate hard instances, allowing us to harness intractability for useful ends such as cryptography and derandomization.

One of the goals of this area is to establish connections between average-case complexity and worst-case complexity. While this has been accomplished for high complexity classes such $\#\mathcal{P}$ and $\mathcal{EXP}$ (e.g. [Lip, BF, BFL, FL, CPS, STV, TV, Vio1]), it remains a major open question for $\mathcal{NP}$. In fact, there are results showing that such connections for $\mathcal{NP}$ are unlikely to be provable using the same kinds of techniques used for the high complexity classes [FF, Vio1, BT, Vio2].

A more modest goal is "hardness amplification," where we seek to establish connections between "mild" average-case complexity and "strong" average-case complexity. That is, given a problem for which a nonnegligible fraction of inputs are "hard," can we obtain a problem for which almost all inputs are hard? To make this precise, let us define "hard."

Definition 1.1. *For $\delta \in [0, 1/2]$, a function $f : \{0,1\}^n \to \{0,1\}$ is* δ-hard for size s *if every circuit of size s fails to compute f on at least a δ fraction of inputs.*

Note that the maximum value of the hardness parameter δ is $1/2$ because f is boolean (and so can trivially be computed with error probability at most $1/2$). This notion of hardness is fairly standard (e.g. in the literature on derandomization starting from [NW]), but we remark that it differs from Levin's notion of average-case complexity [Lev] in several ways. Most importantly, Levin's formulation corresponds to algorithms that always either give the correct answer or say "don't know," whereas we consider even "heuristic" algorithms that can make arbitrary errors. (See Impagliazzo's survey [Imp2].)

The *hardness amplification problem* is to convert a function f that is δ-hard for size s into a function f' that is $(1/2 - \epsilon)$-hard for size polynomially related to s. Commonly, $\delta = 1/\operatorname{poly}(n)$ and the aim is to make $\epsilon = \epsilon(n)$ vanish as quickly as possible.

The standard approach to hardness amplification employs Yao's XOR Lemma [Yao] (see [GNW]): Given a mildly hard-on-average function $f : \{0,1\}^n \to \{0,1\}$, we define $f' : \{0,1\}^{n \cdot k} \to \{0,1\}$ by

$$f'(x_1, \ldots, x_k) \stackrel{\text{def}}{=} f(x_1) \oplus f(x_2) \oplus \cdots \oplus f(x_k).$$

The XOR Lemma says that the hardness of f' approaches $1/2$ exponentially fast with k. More precisely:

Yao's XOR Lemma. *If f is δ-hard for size $s(n) \geq n^{\omega(1)}$ and $k \leq \operatorname{poly}(n)$, then f' is $(1/2 - 1/2^{\Omega(\delta k)} - 1/s')$-hard for size $s'(n \cdot k) = s(n)^{\Omega(1)}$.*

In particular, taking $k = \Theta(n/\delta)$, the amplified hardness is dominated by the $1/s'$ term. That is, we can amplify to hardness $(1/2 - \epsilon)$, where ϵ is polynomially related to the (reciprocal of the) circuit size for which f was hard. (Note, however, that we should measure $\epsilon = \epsilon(n')$ as a function of the new input length $n' = n \cdot k$, so when $k = n$, the hardness is actually $1/2 - 1/s(\sqrt{n'})^{\Omega(1)}$.)

However, if we are interested in hardness amplification within $\mathcal{NP}$ (i.e. f and f' are characteristic functions of languages in $\mathcal{NP}$), we cannot use the XOR lemma; it does not ensure that f' is in $\mathcal{NP}$ when f is in $\mathcal{NP}$. Hardness amplification within $\mathcal{NP}$ was first addressed in a recent paper of O'Donnell [O'D], which is the starting point for our work.

1.1 O'Donnell's Hardness Amplification

To ensure that the new function f' is in $\mathcal{NP}$ when f is in $\mathcal{NP}$, O'Donnell [O'D] was led to study constructions of the form

$$f'(x_1, \ldots, x_k) \stackrel{\text{def}}{=} C(f(x_1), f(x_2), \ldots, f(x_k)), \tag{1}$$

where C is an efficiently computable *monotone* function. The monotonicity of C ensures that f' is in $\mathcal{NP}$ when f is in $\mathcal{NP}$. But we are left with the task of choosing such a function C and proving that it indeed amplifies hardness.

Remarkably, O'Donnell was able to precisely characterize the amplification properties of Construction 1 in terms of a combinatorial property of the combining function C, called its *expected bias*. (The actual definition is not needed for this discussion, but can be found in Section 3.) By finding a monotone combining function in which this expected bias is small, he obtained the first positive result on hardness amplification in $\mathcal{NP}$:

O'Donnell's Theorem [O'D]. *If $\mathcal{NP}$ has a balanced function that is $1/\operatorname{poly}(n)$-hard for polynomial-size circuits, then $\mathcal{NP}$ has a function that is $(1/2 - 1/n^{1/2-\alpha})$-hard for polynomial-size circuits (where α is an arbitrarily small positive constant).*

However, the amplification provided by O'Donnell's theorem is not as strong as what the XOR Lemma gives. It is limited to $1/2 - 1/\sqrt{n}$, regardless of the circuit size s for which the original function is hard, even if s is exponentially large. The XOR Lemma, on the other hand, amplifies to $1/2 - 1/s^{\Omega(1)}$. O'Donnell showed that this difference is inherent — no construction of the form (1) with a monotone combining function C can always amplify hardness to better than $1/2 - 1/n$.[1]

1.2 Our Result

In this paper, we manage to amplify hardness within $\mathcal{NP}$ beyond the $1/2 - 1/n$ barrier:

Main Theorem. *If $\mathcal{NP}$ has a balanced function that is $1/\operatorname{poly}(n)$-hard for circuits of size $s(n)$, then $\mathcal{NP}$ has a function that is $(1/2 - 1/s'(n))$-hard for circuits of size $s'(n)$ for some function*[2] *$s'(n) = s(\sqrt{n})^{\Omega(1)}$. In particular,*

1. *If $s(n) = n^{\omega(1)}$, we amplify to hardness $1/2 - 1/n^{\omega(1)}$.*
2. *If $s(n) = 2^{n^{\Omega(1)}}$, we amplify to hardness $1/2 - 1/2^{n^{\Omega(1)}}$.*
3. *If $s(n) = 2^{\Omega(n)}$, we amplify to hardness $1/2 - 1/2^{\Omega(\sqrt{n})}$.*

Items 1–3 match the parameters of the Yao's XOR Lemma. However, subsequent "derandomizations" of the XOR Lemma [Imp1, IW1] actually amplify up to $1/2 - 1/2^{\Omega(n)}$ rather than just $1/2 - 1/2^{\Omega(\sqrt{n})}$ in the case $s(n) = 2^{\Omega(n)}$. This gap is *not* inherent in our approach and, as mentioned below, would be eliminated given a corresponding improvement in one of the tools we employ.

Of course, our construction cannot be of the form in Construction (1). Below we describe our two main points of departure.

[1] The gap between O'Donnell's positive result of $1/2 - 1/\sqrt{n}$ and his negative result of $1/2 - 1/n$ is not significant for what follows, and in particular, it will be subsumed by our improvements.

[2] In the rest of the paper, we more compactly write "for circuits of size $s'(n) = s(\sqrt{n})^{\Omega(1)}$," where the $\Omega(1)$ is to be interpreted as a fixed constant (possibly depending on previously quantified constants).

1.3 Techniques

To explain how we bypass it, we first look more closely at the source of the $1/2 - 1/n$ barrier. The actual barrier is $1/2 - 1/k$, where k is the input length of the monotone combining function C. (This is based on a result of [KKL], see [O'D].) Since in Construction (1), f' has input length $n' = n \cdot k \geq k$, it follows that we cannot amplify beyond $1/2 - 1/n'$.

Derandomization. Given the above, our first idea is to break the link between the input length of f' and the input length of the combining function C. We do this by *derandomizing* O'Donnell's construction. That is, the inputs $x_1, \ldots, x_k$ are no longer taken independently (as in Construction (1)), but are generated pseudorandomly from a short seed of length $n' \ll k$, which becomes the actual input to f'. Our method for generating the x_i's is based on combinatorial designs (as in the Nisan–Wigderson generator [NW]) and Nisan's pseudorandom generator for space-bounded computation [Nis2]; it reduces the input length of f' from $n \cdot k$ to $n' = O(n^2 + \log^2 k)$. We stress that this derandomization is unconditional, i.e. requires no additional complexity assumption. We also remark that it is the quadratic seed length of Nisan's generator that limits our amplification to $1/2 - 1/2^{\Omega(\sqrt{n})}$ rather than $1/2 - 1/2^{\Omega(n)}$ in Part 3 of our Main Theorem, and thus any improvement in Nisan's generator would yield a corresponding improvement in our result.

Similar derandomizations have previously been achieved for Yao's XOR Lemma by Impagliazzo [Imp1] and Impagliazzo and Wigderson [IW1]. The analysis of such derandomizations is typically tailored to a particular proof, and indeed both [Imp1, IW1] gave new proofs of the XOR Lemma for that purpose. In our case, we do not know how to derandomize O'Donnell's original proof, but instead manage to derandomize a different proof due to Trevisan [Tre].

Our derandomization allows for k to be larger than the input length of f', and hence we can go beyond the $1/2 - 1/n'$ barrier. Indeed, by taking k to be a sufficiently large polynomial, we amplify to $1/2 - 1/(n')^c$ for any constant c.

Using Nondeterminism. To amplify further, it is tempting to take k superpolynomial in the input length of f'. But then we run into a different problem: how do we ensure that f' is in $\mathcal{NP}$? The natural algorithm for f' requires running the algorithm for f on k inputs.

To overcome this difficulty, we observe that we need only give an efficient *nondeterministic* algorithm for f'. Each nondeterministic path may involve only polynomially many evaluations of f while the global outcome $f'(x)$ depends on exponentially many evaluations. To implement this idea, we exploit the specific structure of the combining function C. Namely, we (like O'Donnell) use the Tribes function of Ben-Or and Linial [BL], which is a monotone DNF with clauses of size $O(\log k)$. Thus, the nondeterministic algorithm for f' can simply guess a satisfied clause and (nondeterministically) evaluate f on the $O(\log k)$ corresponding inputs.

1.4 Other Results

We also present some complementary negative results:

- We show that the assumption that the original hard function is balanced is necessary, in the sense that no monotone "black-box" hardness amplification can amplify unbalanced functions of unknown bias (or even improve their bias).[3]

[3]We note that there do exist balanced $\mathcal{NP}$-complete problems, as observed by Barak [For], but this has no direct implication for us because we are studying the average-case complexity of $\mathcal{NP}$.

- We show that our use of nondeterminism is necessary, in the sense that any "black-box" hardness amplification in which each evaluation of f' is a monotone function of at most k evaluations of f can amplify hardness to at most $1/2 - 1/k$.

Informally, a "black-box" hardness amplification is one in which the construction of the amplified function f' from f only utilizes f as an oracle and is well-defined for any function f (regardless of whether or not it is in $\mathcal{NP}$). Moreover, the correctness of the construction is proved by a generic reduction that converts any *oracle* A (regardless of its circuit size) that computes f' well on average (e.g., with probability $1/2+\epsilon$ over random choice of input) into one that computes f much better on average (e.g., with probability $1-o(1)$ over random input). (A formal definition is given in Section 7.1.) We note that most results on hardness amplification against circuits, including ours, are black-box (though there have been some recent results using non-black-box techniques in hardness amplification against *uniform* algorithms; see [IW2, TV]).

Our framework also gives a new proof of the hardness amplification by Impagliazzo and Wigderson [IW1]. Our proof is simpler and in particular its analysis does not employ the Goldreich–Levin [GL] step.

1.5 Organization

The rest of the paper is organized as follows. In Section 2, we discuss some preliminaries. In Section 3, we review existing results on hardness amplification in $\mathcal{NP}$. In Section 4, we present our main results and new techniques. In Section 5 we treat the details of the proof of our main theorem. In Section 6 we show how we could amplify to $1/2 - 1/2^{\Omega(n)}$ given an improvement in the pseudorandom generator we use, and we also give a new proof of the hardness amplification by Impagliazzo and Wigderson [IW1]. In Section 7 we discuss some limitations of monotone hardness amplification; in particular we show a sense in which the hypothesis that the starting function be balanced is necessary, and also that the use of nondeterminism is necessary.

2 Preliminaries

We denote the uniform distribution on $\{0,1\}^n$ by U_n. If U_n occurs more than once in the same expression, it is understood that these all represent the same random variable; for example, $U_n \cdot f(U_n)$ denotes the random variable obtained by choosing X uniformly at random in $\{0,1\}^n$ and outputting $X \cdot f(X)$ (where $\cdot$ means concatenation).

Definition 2.1. *Let X and Y be two random variables taking values over the same set S. Then the* statistical difference *between X and Y, is*

$$\Delta(X,Y) \stackrel{\text{def}}{=} \max_{T \subseteq S} \Big| \Pr[X \in T] - \Pr[Y \in T] \Big|.$$

We view probabilistic functions as functions of two inputs, e.g. $h(x;r)$, the first being the input to the function and the second being the randomness. (Deterministic functions may be thought of as probabilistic functions that ignore the randomness.) For notational convenience, we will often omit the second input to a probabilistic function, e.g. writing $h(x)$ instead of $h(x;r)$, in which case we view $h(x)$ as the random variable $h(x;U_{|r|})$.

Definition 2.2. *The* bias *of a* 0-1 *random variable X is*

$$\mathrm{Bias}\,[X] \stackrel{\text{def}}{=} \Big| \Pr[X=0] - \Pr[X=1] \Big| = 2 \cdot \Delta(X, U_1).$$

Analogously, the bias *of a probabilistic function* $f : \{0,1\}^n \to \{0,1\}$ *is*

$$\mathrm{Bias}\,[f] \stackrel{\mathrm{def}}{=} \Big| \Pr[f(U_n) = 0] - \Pr[f(U_n) = 1] \Big|,$$

where the probabilities are taken over both the input chosen according to U_n *and the coin tosses of* f*. We say that* f *is* balanced *when* $\mathrm{Bias}\,[f] = 0$.

Note that the bias of a random variable is a quantity between 0 and 1.

We say that the random variables X and Y are *ϵ-indistinguishable for size s* if for every circuit C of size s,

$$\Big| \Pr_X[C(X) = 1] - \Pr_Y[C(Y) = 1] \Big| \le \epsilon.$$

We will routinely use the following connection between hardness and indistinguishability.

Lemma 2.3 ([Yao])**.** *Let* $h : \{0,1\}^n \to \{0,1\}$ *be any probabilistic function. If the distributions* $U_n \cdot h(U_n)$ *and* $U_n \cdot U_1$ *are ϵ-indistinguishable for size s then h is $(1/2 - \epsilon)$-hard for size $s - O(1)$. Conversely, if h is $(1/2 - \epsilon)$-hard for size s then the distributions* $U_n \cdot h(U_n)$ *and* $U_n \cdot U_1$ *are ϵ-indistinguishable for size $s - O(1)$.*

Finally, whenever we amplify the hardness of a function $f : \{0,1\}^n \to \{0,1\}$ that is hard for circuits of size $s(n)$, we assume that $s(n)$ is well-behaved in the sense that it is computable in time $\mathrm{poly}(n)$ and $s(cn) = s(n)^{O(1)}$, for all constants $c > 0$. Most natural functions smaller than 2^n, such as $n^k, 2^{\log^k n}, 2^{n^\epsilon}, 2^{\epsilon n}$, are well-behaved in this sense.

3 Overview of Previous Hardness Amplification in $\mathcal{NP}$

In this section we review the essential components of existing results on hardness amplification in $\mathcal{NP}$. We then discuss the limitations of these techniques. By the end of this section, we will have sketched the main result of O'Donnell [O'D], following the approach of Trevisan [Tre]. We outline this result in a way that will facilitate the presentation of our results in subsequent sections.

Let $f : \{0,1\}^n \to \{0,1\}$ be an average-case hard function, and let $C : \{0,1\}^k \to \{0,1\}$ be any function. In [O'D], O'Donnell studies the hardness of functions of the form

$$C \circ f^{\otimes k} : (\{0,1\}^n)^k \to \{0,1\}$$

where $f^{\otimes k}(x_1, \ldots, x_k) \stackrel{\mathrm{def}}{=} (f(x_1), \ldots, f(x_k))$, and $\circ$ denotes composition. That is,

$$(C \circ f^{\otimes k})(x_1, \ldots, x_k) \stackrel{\mathrm{def}}{=} C(f(x_1), \ldots, f(x_k)).$$

In order to ensure that $C \circ f^{\otimes k} \in \mathcal{NP}$ whenever $f \in \mathcal{NP}$, O'Donnell chooses C to be a polynomial-time computable *monotone* function. (Indeed, it is not hard to see that a monotone combination of $\mathcal{NP}$ functions is itself in $\mathcal{NP}$.)

O'Donnell characterizes the hardness of $C \circ f^{\otimes k}$ in terms of a combinatorial property of the combining function C, called its *expected bias* (which we define later).

We will now review the key steps in establishing this characterization and O'Donnell's final amplification theorem.

Step 1: Impagliazzo's hardcore sets. An important tool for establishing this connection is the hardcore set lemma of Impagliazzo [Imp1], which allows us to pass from computational hardness to information-theoretic hardness.

Definition 3.1. *We say that a (probabilistic) function $g : \{0,1\}^n \to \{0,1\}$ is* δ-random *if g is balanced and there exists a subset $H \subseteq \{0,1\}^n$ with $|H| = 2\delta 2^n$ such that $g(x) = U_1$ (i.e. a coin flip) for $x \in H$ and $g(x)$ is deterministic for $x \notin H$.*

Thus, a δ-random function has a set of relative size 2δ on which it is information-theoretically unpredictable. Note that in the above definition we require g to be balanced. This will be convenient when dealing with functions f that are balanced.

The following version of Impagliazzo hardcore set lemma says that any balanced δ-hard function $f : \{0,1\}^n \to \{0,1\}$ has a hardcore set $H \subseteq \{0,1\}^n$ of density $\approx 2\delta$ such that f is very hard-on-average on H. Thus, f looks like a δ-random function to small circuits (cf. Lemma 2.3). (Following subsequent works, our formulation of Impagliazzo's lemma differs from the original one in several respects.)

Lemma 3.2 ([Imp1, KS, STV, O'D])**.** *For any function $f : \{0,1\}^n \to \{0,1\}$ that is balanced and δ-hard for size s, there exists a δ'-random function $g : \{0,1\}^n \to \{0,1\}$ such that $X \cdot f(X)$ and $X \cdot g(X)$ are ϵ-indistinguishable for size $\Omega(s\epsilon^2/\log(1/\delta))$, with $\delta/2 \le \delta' \le \delta$, where $X \equiv U_n$.*

In particular, by a standard hybrid argument (see, e.g., [Gol3]),

$$X_1 \cdots X_k \cdot f(X_1) \cdots f(X_k) \text{ and } X_1 \cdots X_k \cdot g(X_1) \cdots g(X_k)$$

are $k\epsilon$-indistinguishable for size $\Omega(s\epsilon^2/\log(1/\delta))$, where the X_i's are uniform and independent.

Step 2: Expected Bias. By the above, proving the computational hardness of $C \circ f^{\otimes k}$ reduces to calculating the information-theoretic hardness of $C \circ g^{\otimes k}$ for some δ'-random g. It turns out that information-theoretic hardness can be characterized by the following quantity.

Definition 3.3. *Let $h : \{0,1\}^n \to \{0,1\}$ be any probabilistic function. We define the* expected bias *of h by*

$$\mathrm{ExpBias}[h] \stackrel{\text{def}}{=} \mathop{\mathrm{E}}_{x \leftarrow U_n}\left[\,\mathrm{Bias}[h(x)]\,\right],$$

where $\mathrm{Bias}[h(x)]$ *is taken over the coin tosses of h.*

It turns out that for any function $C : \{0,1\}^k \to \{0,1\}$ and any δ-random g, the quantity $\mathrm{ExpBias}\left[C \circ g^{\otimes k}\right]$ does not depend on the particular choice of the δ-random function g; indeed, it turns out to equal the quantity that O'Donnell [O'D] calls the "expected bias of C with respect to noise 2δ" and denotes by $\mathrm{ExpBias}_{2\delta}(C)$ in [O'D]. However, the more general notation we use will be useful in presenting our improvements.

The next lemma shows that information-theoretic hardness is equivalent to expected bias.

Lemma 3.4. *For any probabilistic function $h : \{0,1\}^n \to \{0,1\}$,*

$$\Delta(U_n \cdot h(U_n), U_n \cdot U_1) = \frac{1}{2}\,\mathrm{ExpBias}[h].$$

Proof. $\Delta(U_n \cdot h(U_n), U_n \cdot U_1) = \mathop{\mathrm{E}}_{x \leftarrow U_n}[\Delta(h(x), U_1)] = \mathop{\mathrm{E}}_{x \leftarrow U_n}[\mathrm{Bias}[h(x)]/2] = \mathrm{ExpBias}[h]/2.$ □

In particular, for any circuit C (regardless of its size) we have $\Big| \Pr[C(U_n \cdot h(U_n)) = 1] - \Pr[C(U_n \cdot U_1) = 1] \Big| \le \mathrm{ExpBias}\,[h]\,/2$, and thus by Lemma 2.3, h is $(1/2 - \mathrm{ExpBias}\,[h]\,/2)$-hard for circuits of any size.

Now we characterize the hardness of $C \circ f^{\otimes k}$ in terms of expected bias. Specifically, by taking, say, $\epsilon = 1/s^{1/3}$ in Lemma 3.2 and using Lemmas 2.3 and 3.4, one can prove the following (we defer the details until the proof of the more general Lemma 5.2).

Lemma 3.5 ([O'D])**.** *Let $f : \{0,1\}^n \to \{0,1\}$ be balanced and δ-hard for size s, and let $C : \{0,1\}^k \to \{0,1\}$ be any function. Then there exists a δ'-random function $g : \{0,1\}^n \to \{0,1\}$, with $\delta/2 \le \delta' \le \delta$, such that $C \circ f^{\otimes k} : (\{0,1\}^n)^k \to \{0,1\}$ has hardness*

$$\frac{1}{2} - \frac{\mathrm{ExpBias}\left[C \circ g^{\otimes k}\right]}{2} - \frac{k}{s^{1/3}}$$

for circuits of size $\Omega\left(s^{1/3}/\log(1/\delta)\right) - \mathrm{size}(C)$, where $\mathrm{size}(C)$ denotes the size of a smallest circuit computing C.

What makes this lemma so useful is that, as noted above, the quantity $\mathrm{ExpBias}\left[C \circ g^{\otimes k}\right]$ is independent of the choice of the δ-random function g (using the fact that g is balanced, by definition of δ-random); hence the hardness of $C \circ f^{\otimes k}$ depends only on the combining function C and the hardness parameter δ. Thus, understanding the hardness of $C \circ f^{\otimes k}$ is reduced to analyzing a combinatorial property of the combining function C.

Step 3: Noise Stability Unfortunately, it is often difficult to analyze the expected bias directly. However, the expected bias is closely related to the *noise stability*, a quantity that is more amenable to analysis and well-studied (see, e.g., [O'D], [MO]). The noise stability of a function is (up to normalization) the probability that the value of the function is the same on two correlated inputs x and $x + \eta$, where x is a random input and η a random vector of noise.

Definition 3.6. *The noise stability of C with respect to noise δ, denoted $\mathrm{NoiseStab}_\delta[C]$, is defined by*

$$\mathrm{NoiseStab}_\delta[C] \stackrel{\text{def}}{=} 2 \cdot \Pr_{x,\eta}[C(x) = C(x \oplus \eta)] - 1,$$

where x is random, η is a vector whose bits are independently one with probability δ and $\oplus$ denotes bitwise XOR.

The following lemma from [O'D] bounds the expected bias of $C \circ g^{\otimes k}$ (and hence the hardness in Lemma 3.5) in terms of the noise stability of C.

Lemma 3.7. *Let $g : \{0,1\}^n \to \{0,1\}$ be δ-random. Then*

$$\mathrm{ExpBias}\left[C \circ g^{\otimes k}\right] \le \sqrt{\mathrm{NoiseStab}_\delta[C]}.$$

Combining this with Lemma 3.5, we find that the hardness of $C \circ f^{\otimes k}$ is at least (roughly) $1/2 - \sqrt{\mathrm{NoiseStab}_\delta[C]}/2$. The next step is to exhibit a combining function C with a small noise stability (to ensure that the hardness of $C \circ f^{\otimes k}$ is as close to $1/2$ as possible). The following is shown in [O'D].

Lemma 3.8 ([O'D])**.** *For all $\delta > 0$, there exists a $k = \mathrm{poly}(1/\delta)$ and a polynomial-time computable monotone function $C : \{0,1\}^k \to \{0,1\}$ with $\mathrm{NoiseStab}_\delta[C] \le 1/k^{\Omega(1)}$.*

Finally, by combining Lemmas 3.5, 3.7 and 3.8, we obtain the following weaker version of O'Donnell's hardness amplification within $\mathcal{NP}$. (While in the introduction we mentioned a stronger version of O'Donnell's result, that amplifies up to hardness $1/2-1/m^{1/2-\alpha}$ for every constant $\alpha > 0$, the following version will suffice as a starting point for our work. The loss in the amplification in this version comes from the fact that we did not specify the constants in Lemma 3.8.)

Theorem 3.9 ([O'D]). *If there is a balanced function $f : \{0,1\}^n \to \{0,1\}$ in $\mathcal{NP}$ that is $1/\operatorname{poly}(n)$-hard for size $s(n)$, then there is a function $f' : \{0,1\}^m \to \{0,1\}$ in $\mathcal{NP}$ that is $(1/2-1/m^{\Omega(1)})$-hard for size $s(m^{\Omega(1)})^{\Omega(1)}$.*

Limitations of Direct Product Constructions.
O'Donnell also showed that Theorem 3.9 is essentially the best result that one can obtain using the techniques that we have described thus far. He showed that for all monotone combining functions C there is a δ-hard function f such that the hardness of $C \circ f^{\otimes k}$ is not much better than $1/2 - \mathrm{NoiseStab}_\delta[C]/2$ (assuming that C is easily computable). This is problematic because the noise stability of monotone functions cannot become too small. Specifically, by combining a result from [KKL] with a Fourier characterization of noise stability, O'Donnell [O'D] proves the following theorem.

Theorem 3.10 ([KKL, O'D]). *For every monotone function $C : \{0,1\}^k \to \{0,1\}$ and every $\delta > 0$,*

$$\mathrm{NoiseStab}_\delta[C] \geq (1-2\delta) \cdot \Omega\left(\frac{\log^2 k}{k}\right).$$

Therefore, for any monotone $C : \{0,1\}^k \to \{0,1\}$ there is a δ-hard f such that $C \circ f^{\otimes k}$ does *not* have hardness $1/2 - \mathrm{NoiseStab}_\delta[C]/2 \leq 1/2 - \Omega(1/k)$. Since $C \circ f^{\otimes k}$ takes inputs of length $m = n \cdot k \geq k$, this implies that we must employ a new technique to amplify beyond hardness $1/2 - \Omega(1/m)$.

4 Main Theorem and Overview

In this paper, we obtain the following improvement upon Theorem 3.9.

Theorem 4.1 (Main Theorem, restated). *If there is a balanced function $f : \{0,1\}^n \to \{0,1\}$ in $\mathcal{NP}$ that is $1/\operatorname{poly}(n)$-hard for size $s(n)$, then there is a function $f' : \{0,1\}^m \to \{0,1\}$ in $\mathcal{NP}$ that is $(1/2 - 1/s(\sqrt{m})^{\Omega(1)})$-hard for size $s(\sqrt{m})^{\Omega(1)}$.*[4]

We also show that the assumption that we start with a *balanced* function f is essential for a large class of hardness amplifications. Specifically, we show (Section 7.1) that no monotone black-box hardness amplification can amplify the hardness of functions whose bias is unknown. Most hardness amplifications, including the one in this paper, are black-box.

We now elaborate on the two main techniques that allow us to prove Theorem 4.1. As explained in the introduction, these two techniques are derandomization and nondeterminism.

[4] A comment is in order about the input lengths for which f' is hard. As it turns out, the hardness of f' on inputs of length m is related to the hardness of the original function f on inputs of length $\Theta(\sqrt{m})$. Thus if f is hard for all sufficiently large input lengths, then so is f'. Alternatively, if f is only hard infinitely often, then we may still conclude that f' is hard infinitely often.

4.1 Derandomization

As in the previous section, let $f : \{0,1\}^n \to \{0,1\}$ be our hard function and let $C : \{0,1\}^k \to \{0,1\}$ be a (monotone) combining function.

We will derandomize O'Donnell's construction using an appropriately "pseudorandom" generator.

Definition 4.2. *A* generator *is a function* $G : \{0,1\}^l \to (\{0,1\}^n)^k$*. We call* l *the* seed length *of* G*, and we often write* $G(\sigma) = X_1 \cdots X_k$*, with each* $X_i \in \{0,1\}^n$*.* G *is* explicitly computable *if given* σ *and* $1 \le i \le k$*, we can compute* X_i *in time* $\mathrm{poly}(l, \log k)$*, where* $G(\sigma) = X_1 \cdots X_k$*.*

Instead of using the function $C \circ f^{\otimes k} : (\{0,1\}^n)^k \to \{0,1\}$, we take a generator $G : \{0,1\}^l \to (\{0,1\}^n)^k$ (where $l \ll nk$) and use $(C \circ f^{\otimes k}) \circ G : \{0,1\}^l \to \{0,1\}$, i.e.,

$$(C \circ f^{\otimes k}) \circ G(\sigma) = C\big(f(X_1), \ldots, f(X_k)\big),$$

where $(X_1, \ldots, X_k) \in (\{0,1\}^n)^k$ is the output of $G(\sigma)$. This reduces the input length of the function to l. Therefore, if G is a "good" pseudorandom generator we would expect $(C \circ f^{\otimes k}) \circ G$ to be harder (with respect to its input length) than $C \circ f^{\otimes k}$. We will show that this is indeed the case, provided the generator G satisfies the following requirements:

1. **G is indistinguishability-preserving:** Analogously to Lemma 3.5, the generator G should be such that the computational hardness of $(C \circ f^{\otimes k}) \circ G$ is at least the information-theoretic hardness of $(C \circ g^{\otimes k}) \circ G$ for some δ-random function g – that is, at least $1/2 - \mathrm{ExpBias}\left[(C \circ g^{\otimes k}) \circ G\right]$. We will see that this can be achieved provided that G is *indistinguishability-preserving*; that is (analogously to the last part of Lemma 3.2),

$$\sigma \cdot f(X_1) \cdots f(X_k) \text{ and } \sigma \cdot g(X_1) \cdots g(X_k)$$

   should be indistinguishable, for some δ-random g, when $\sigma \xleftarrow{\mathrm{R}} \{0,1\}^l$ and $(X_1, \ldots, X_k) \in (\{0,1\}^n)^k$ is the output of G on input σ.

2. **G fools the expected bias:** G should be such that for any δ-random g, $\mathrm{ExpBias}\left[(C \circ g^{\otimes k}) \circ G\right]$ is approximately $\mathrm{ExpBias}\left[C \circ g^{\otimes k}\right]$, and thus, by Lemma 3.7:

$$\mathrm{ExpBias}\left[(C \circ g^{\otimes k}) \circ G\right] \le \sqrt{\mathrm{NoiseStab}_\delta[C] + \epsilon}, \tag{2}$$

   for a suitably small ϵ. Actually, we will not show that G fools the expected bias directly but instead will work with a related quantity (the expected collision probability), which will still suffice to show Inequality (2).

Informally, the effect of the two above requirements on the generator G is that the hardness of $(C \circ f^{\otimes k}) \circ G$ is roughly the hardness of $C \circ f^{\otimes k}$, while the input length is dramatically reduced from nk to l (the seed length of G). More precisely, as illustrated in Figure 1, the first requirement allows us to relate the hardness of $(C \circ f^{\otimes k}) \circ G$ to the information-theoretic hardness of $(C \circ g^{\otimes k}) \circ G$ (where g is a δ-random function); the second allows us to relate this information-theoretic hardness to the noise stability of the combining function C. In particular, if we employ the combining function from Lemma 3.8, we obtain hardness $1/2 - 1/k^{\Omega(1)}$. Thus, by choosing $k \gg l$, we bypass the barrier discussed at the end of the previous section.

Now we briefly describe how the above requirements on G are met. The first requirement is achieved through a generator that outputs *combinatorial designs*. This construction is essentially from Nisan and Wigderson [Nis1, NW] and has been used in many places, e.g. [IW1, STV].

The second requirement is achieved as follows. We show that if G is pseudorandom against space-bounded algorithms and the combining function C is computable in small space (with one-way access to its input), then Inequality (2) holds. We then use Nisan's *unconditional* pseudorandom generator against space-bounded algorithms [Nis2], and show that combining functions with low noise stability can in fact be computed in small space.[5] Note that we only use the pseudorandomness of the generator G to relate the expected bias with respect to G to a combinatorial property of the combining function C. In particular, it is *not* used to fool the circuits trying to compute the hard function. This is what allows us to use an unconditional generator against a relatively weak model of computation.

Our final generator, Γ, is the generator obtained by XORing a generator that is indistinguishability-preserving and a generator that fools the expected bias, yielding a generator that has both properties. The approach of XORing two generators in this way appeared in [IW1], and was subsequently used in [STV].

4.2 Using Nondeterminism

The derandomization described above gives hardness amplification up to $1/2-1/n^c$ *for any* constant c. This already improves upon the best previous result, namely Theorem 3.9. However, to go beyond this new techniques are required. The problem is that if we want C to be computable in time $\mathrm{poly}(n)$, we must take $k = \mathrm{poly}(n)$ and thus we amplify to at most $1/2-1/k = 1/2-1/\mathrm{poly}(n)$.

We solve this problem by taking full advantage of the power of $\mathcal{NP}$, namely nondeterminism. This allows us to use a function $C : \{0,1\}^k \to \{0,1\}$ which is computable in *nondeterministic* time $\mathrm{poly}(n, \log(k))$; thus, the amplified function will still be in *$\mathcal{NP}$ for k as large as 2^n*.

Conversely, in Section 7.2 we show that any non-adaptive monotone black-box hardness amplification that amplifies to hardness $1/2 - 1/n^{\omega(1)}$ cannot be computed in $\mathcal{P}$, i.e. the use of nondeterminism is essential.

We proceed by discussing the details of the derandomization (Sections 5.1, 5.2 and 5.3) and the use of nondeterminism (Section 5.4). The results obtained in these sections are summarized in Table 1. For clarity of exposition, we focus on the case where the original hard function f is balanced and is 1/3-hard. Hardness amplification from hardness $1/\mathrm{poly}(n)$ is discussed in Section 5.5, and hardness amplification of unbalanced functions is discussed in Section 7.1.

5 Proof of Main Theorem

In this section we prove our main theorem (i.e., Theorem 4.1).

5.1 Preserving Indistinguishability

The main result in this subsection is that if G is pseudorandom in an appropriate sense, then the hardness of $(C \circ f^{\otimes k}) \circ G$ is roughly

$$1/2 - \mathrm{ExpBias}\left[(C \circ g^{\otimes k}) \circ G\right]$$

[5] The same approach also works using the unconditional pseudorandom generator against constant-depth circuits of [Nis1] and showing that the combining function is computable by a small constant-depth circuit; however, the space generator gives us slightly better parameters.

Table 1: Hardness Amplification within $\mathcal{NP}$.

Functions : $\{0,1\}^n \to \{0,1\}$		
Amplification up to	**Technique**	**Reference**
$1/2 - 1/\sqrt{n}$	Direct Product	[O'D]
$1/2 - 1/n^c$, for *every* c	Derandomized Direct Product	Theorem 5.8
$1/2 - 1/2^{\Omega(\sqrt{n})}$	Derandomized Direct Product & Nondeterminism	Theorem 5.13

for some δ-random function g. As we noted in the previous section, it will be sufficient for G to be *indistinguishability-preserving*. We give the definition of indistinguishability-preserving and then our main result.

Definition 5.1. *A generator $G : \{0,1\}^l \to (\{0,1\}^n)^k$ is said to be* indistinguishability-preserving for size t *if for all (possibly probabilistic) functions $f_1, \ldots, f_k, g_1, \ldots, g_k$ the following holds:*

If for every $i, 1 \le i \le k$ the distributions

$$U_n \cdot f_i(U_n) \text{ and } U_n \cdot g_i(U_n)$$

are ϵ-indistinguishable for size s, then

$$\sigma \cdot f_1(X_1) \cdots f_k(X_k) \text{ and } \sigma \cdot g_1(X_1) \cdots g_k(X_k)$$

are $k\epsilon$-indistinguishable for size $s - t$, where σ is a random seed of length l and $X_1 \cdots X_k$ is the output of $G(\sigma)$.

The fact that in the above definition we consider k f_i's and k g_i's implies that an $indistinguishability-preserving$ generator stays $indistinguishability-preserving$ when XORed with any other generator (cf. the proof of Item 1 in Lemma 5.12). We will use this property in the proof of our main result.

Lemma 5.2. *Let $f : \{0,1\}^n \to \{0,1\}$ be δ-hard for size s, let $G : \{0,1\}^l \to (\{0,1\}^n)^k$ be a generator that is indistinguishability-preserving for size t and let $C : \{0,1\}^k \to \{0,1\}$ be any function. Then there exists a δ'-random g, with $\delta/2 \le \delta' \le \delta$ such that the function $(C \circ f^{\otimes k}) \circ G : \{0,1\}^l \to \{0,1\}$ has hardness*

$$\frac{1}{2} - \frac{\mathrm{ExpBias}\left[(C \circ g^{\otimes k}) \circ G\right]}{2} - \frac{k}{s^{1/3}}$$

for circuits of size $\Omega\left(s^{1/3}/\log(1/\delta)\right) - t - \mathrm{size}(C)$ where $\mathrm{size}(C)$ denotes the size of a smallest circuit computing C.

Proof. By Lemma 3.2, there exists a δ'-random function g with $\delta/2 \le \delta' \le \delta$, such that $U_n \cdot f(U_n)$ and $U_n \cdot g(U_n)$ are ϵ-indistinguishable for size $\Omega(s\epsilon^2/\log(1/\delta))$. Since G is an indistinguishability-preserving for size t by assumption, this implies that

$$\sigma \cdot f(X_1) \cdots f(X_k) \text{ and } \sigma \cdot g(X_1) \cdots g(X_k)$$

are $k\epsilon$-indistinguishable for size $\Omega(s\epsilon^2/\log(1/\delta)) - t$, where here and below σ denotes a uniform random seed in $\{0,1\}^l$ and $X_1 \cdots X_k$ will denote the output of $G(\sigma)$. This in turn implies that

$$\sigma \cdot C(f(X_1) \cdots f(X_k)) \text{ and } \sigma \cdot C(g(X_1) \cdots g(X_k))$$

(i.e., $\sigma \cdot (C \circ f^{\otimes k}) \circ G(\sigma)$ and $\sigma \cdot (C \circ g^{\otimes k}) \circ G(\sigma))$ are $k\epsilon$-indistinguishable for size $\Omega(s\epsilon^2/\log(1/\delta)) - t - \mathrm{size}(C)$. By Lemma 3.4,

$$\sigma \cdot (C \circ g^{\otimes k}) \circ G \text{ and } \sigma \cdot U_1$$

are $(\mathrm{ExpBias}\left[(C \circ g^{\otimes k}) \circ G\right]/2)$-indistinguishable for any size. Therefore, we have that

$$\sigma \cdot (C \circ f^{\otimes k}) \circ G \text{ and } \sigma \cdot U_1$$

are $(\mathrm{ExpBias}\left[(C \circ g^{\otimes k}) \circ G\right]/2 + k\epsilon)$-indistinguishable for size $\Omega(s\epsilon^2/\log(1/\delta)) - t - \mathrm{size}(C)$. The result follows by setting $\epsilon = 1/s^{1/3}$ and applying Lemma 2.3. □

In particular, we note that the *identity generator* $G : \{0,1\}^{nk} \to (\{0,1\}^n)^k$, i.e. $G(x) = x$, is indistinguishability-preserving for size 0 (by a hybrid argument, see, e.g., [Gol3]), and thus Lemma 3.5 is a corollary of Lemma 5.2. However, the identity generator has seed-length nk and is therefore a very poor pseudorandom generator. Fortunately, there are indistinguishability-preserving pseudorandom generators with much shorter seeds which will allow us to use Lemma 5.2 to obtain much stronger hardness amplifications.

Lemma 5.3. *There is a constant c such that for every $n \ge 2$ and every $k = k(n)$ there is an explicitly computable generator $NW_k : \{0,1\}^l \to (\{0,1\}^n)^k$ with seed length $l = c \cdot n^2$ that is indistinguishability-preserving for size k^2.*

Proof. The generator is the main component of the generator by Nisan and Nisan and Wigderson [Nis1, NW], and is based on combinatorial designs. Specifically, we let $S_1, \ldots, S_k \subseteq [l]$ be an explicit family of sets such that $|S_i| = n$ for all i, and $|S_i \cap S_j| \le \log k$ for all $i \ne j$. Nisan [Nis1] gives an explicit construction of such sets with $l = O(n^2)$. Then the generator $NW_k : \{0,1\}^l \to (\{0,1\}^n)^k$ is defined by

$$NW_k(\sigma) := (\sigma|_{S_1}, \ldots, \sigma|_{S_k}),$$

where $\sigma|_{S_i} \in \{0,1\}^n$ denotes the projection of σ onto the coordinates indexed by the set S_i.

The proof that this generator is indistinguishability preserving for size k^2 follows the arguments in [NW, STV]. For completeness, we sketch the proof here. Suppose that we have a circuit C of size $s - k^2$ distinguishing the distributions $\sigma \cdot f_1(\sigma|_{S_1}) \cdots f_k(\sigma|_{S_k})$ and $\sigma \cdot g_1(\sigma|_{S_1}) \cdots g_k(\sigma|_{S_k})$ with advantage greater than $k \cdot \epsilon$. For $i = 0, \ldots, k$, let H_i be the hybrid distribution

$$H_i = \sigma \cdot g_1(\sigma|_{S_1}) \cdots g_i(\sigma|_{S_i}) \cdot f_{i+1}(\sigma|_{S_{i+1}}) \cdots f_k(\sigma|_{S_k}).$$

Then there must exist an $i \in \{0, \ldots, k\}$ such that C distinguishes H_i from H_{i+1} with advantage greater than ϵ. The only difference between H_i and H_{i+1} is that H_i has the component $f_{i+1}(\sigma|_{S_{i+1}})$ while H_{i+1} has $g_{i+1}(\sigma|_{S_{i+1}})$. By averaging, we may fix all the bits of σ outside of S_{i+1} (as well as the randomness of f_j, g_j for $j \neq i+1$ if they are probabilistic functions) while preserving the advantage of C. Thus C distinguishes between two distributions of the form

$$\tau \cdot h_1(\tau)h_2(\tau)\cdots h_i(\tau)f_{i+1}(\tau)h_{i+2}(\tau)\cdots h_k(\tau),$$

and

$$\tau \cdot h_1(\tau)h_2(\tau)\cdots h_i(\tau)g_{i+1}(\tau)h_{i+2}(\tau)\cdots h_k(\tau),$$

where τ is uniform in $\{0,1\}^n$ and each h_j is a function of at most $|S_j \cap S_{i+1}|$ bits of τ. Then each h_j can be computed by a circuit of size smaller than $2^{|S_j \cap S_{i+1}|} \leq k$. Combining these $k-1$ circuits with C, we get a distinguisher between $\tau \cdot f_{i+1}(\tau)$ and $\tau \cdot g_{i+1}(\tau)$ of size $|C| + (k-1) \cdot k < s$ and advantage greater than ϵ. □

5.2 Fooling the Expected Bias

In this subsection we prove a derandomized version of Lemma 3.7. Informally, we show that if C is computable in a restricted model of computation and G "fools" that restricted model of computation, then for any δ-random function g:

$$\mathrm{ExpBias}\left[(C \circ g^{\otimes k}) \circ G\right] \leq \sqrt{\mathrm{NoiseStab}_\delta[C] + \epsilon}.$$

The restricted model of computation we consider is that of nonuniform space-bounded algorithms which make one pass through the input, reading it in blocks of length n. These are formally modeled by the following kind of branching programs.

Definition 5.4. *A (probabilistic, read-once, oblivious)* branching program *of* size s *with* block-size n *is a finite state machine with s states, over the alphabet $\{0,1\}^n$ (with a fixed start state, and an arbitrary number of accepting states). Each edge is labelled with a symbol in $\{0,1\}^n$. For every state a and symbol $\alpha \in \{0,1\}^n$, the edges leaving a and labelled with α are assigned a probability distribution. Then computation proceeds as follows. The input is read sequentially, one block of n bits at a time. If the machine is in state a and it reads α, then it chooses an edge leaving a and labelled with α according to its probability, and moves along it.*

Now we formally define pseudorandom generators against branching programs.

Definition 5.5. *A generator $G : \{0,1\}^l \to (\{0,1\}^n)^k$ is* ϵ-pseudorandom against branching programs of size s and block-size n *if for every branching program* B *of size s and block-size n:*

$$\left|\Pr[\mathrm{B}(G(U_l)) = 1] - \Pr[\mathrm{B}(U_{nk}) = 1]\right| \leq \epsilon.$$

In [Nis2], Nisan builds an unconditional pseudorandom generator against branching programs. Its parameters (specialized for our purposes) are given in the following theorem.

Theorem 5.6 ([Nis2])**.** *For every n and $k \leq 2^n$, there exists a generator*

$$N_k : \{0,1\}^l \to (\{0,1\}^n)^k$$

such that:

- *N_k is 2^{-n}-pseudorandom against branching programs of size 2^n and block-size n.*
- *N_k has seed length $l = O(n \log k)$.*

- N_k *is explicitly computable.*

Note that Nisan [Nis2] does not mention *probabilistic* branching programs. However, if there is a probabilistic branching program distinguishing the output of the generator from uniform, then by a fixing of the coin tosses of the branching program there is a *determinisitic* branching program that distinguishes the output of the generator from uniform.

We now state the derandomized version of Lemma 3.7.

Lemma 5.7. *Let*

- $g : \{0,1\}^n \to \{0,1\}$ *be a* δ*-random function,*
- $C : \{0,1\}^k \to \{0,1\}$ *be computable by a branching program of size* t *and block-size* 1*,*
- $G : \{0,1\}^l \to (\{0,1\}^n)^k$ *be* $\epsilon/2$*-pseudorandom against branching programs of size* t^2 *and block-size* n*.*

Then $\mathrm{ExpBias}\left[(C \circ g^{\otimes k}) \circ G\right] \leq \sqrt{\mathrm{NoiseStab}_\delta[C] + \epsilon}$.

Proof. We will not show that G fools the expected bias, but rather the following related quantity. For a probabilistic boolean function $h(x;r)$ we define its (normalized) *expected collision probability* as

$$\mathrm{ExpCP}[h] \stackrel{\text{def}}{=} \mathop{\mathrm{E}}_{x}\big[2 \cdot \Pr_{r,r'}[h(x;r) = h(x;r')] - 1\big].$$

The same reasoning that proves Lemma 3.7 also shows that for every probabilistic boolean function h:

$$\mathrm{ExpBias}\,[h] \leq \sqrt{\mathrm{ExpCP}[h]}. \tag{3}$$

More specifically, Inequality (3) holds because

$$\begin{aligned} \mathrm{ExpBias}\,[h] &= \mathop{\mathrm{E}}_{x \leftarrow U_n}\big[\,\mathrm{Bias}\,[h(x)]\,\big] \\ &\leq \sqrt{\mathop{\mathrm{E}}_{x \leftarrow U_n}\big[\,\mathrm{Bias}\,[h(x)]^2\,\big]} \qquad \text{(by Cauchy-Schwartz)} \\ &= \sqrt{\mathrm{ExpCP}[h]} \end{aligned}$$

Let $h(x;r) : (\{0,1\}^n)^k \to \{0,1\}$ be the probabilistic function $C \circ g^{\otimes k}$. Even though h is defined in terms of g, it turns out that its expected collision probability is the same for all δ-random functions g. Specifically, for $x = (x_1, \ldots, x_k)$, the only dependence of the collision probability $\Pr_{r,r'}[h(x;r) = h(x;r')]$ on x_i comes from whether $g(x_i)$ is a coin flip (which occurs with probability δ over the choice of x_i), $g(x_i) = 1$ (which occurs with probability $(1-\delta)/2$), or $g(x_i) = 0$ (which occurs with probability $(1-\delta)/2$). In the case where $g(x_i)$ is a coin flip, then the i'th bits of the two inputs fed to C (i.e. $g(x_i;r)$ and $g(x_i;r')$) are random and independent, and otherwise they are equal and fixed (according to $g(x_i)$). It can be verified that this corresponds precisely to the definition of noise stability, so we have:

$$\mathrm{ExpCP}[h] = \mathrm{NoiseStab}_\delta[C]. \tag{4}$$

Now we construct a probabilistic branching program $M : (\{0,1\}^n)^k \to \{0,1\}$ of size t^2 and block-size n such that for every $x \in (\{0,1\}^n)^k$:

$$\Pr[M(x) = 1] = \Pr_{r,r'}[h(x;r) = h(x;r')].$$

To do this, we first note that, using the branching program for C, we can build a probabilistic branching program of size t with block-size n which computes $C \circ g^{\otimes k}$: The states of the branching program are the same as those of the branching program for C, and we define the transitions as follows. Upon reading symbol $\alpha \in \{0,1\}^n$ in state s, if $g(\alpha) = 0$ (resp. $g(\alpha) = 1$), we deterministically go to the state given by the 0-transition (resp., 1-transition) of C from state s, and if $g(\alpha)$ is a coin flip, then we put equal probability on these two transitions.

Then, to obtain M, run two *independent* copies of this branching program (i.e., using independent choices for the probabilistic state transitions) and accept if and only if both of them accept or both of them reject. Now,

$$\begin{aligned}
&\left| \mathrm{ExpCP}[(C \circ g^{\otimes k}) \circ G] - \mathrm{NoiseStab}_\delta[C] \right| \\
&= \left| \mathrm{ExpCP}[(C \circ g^{\otimes k}) \circ G] - \mathrm{ExpCP}[C \circ g^{\otimes k}] \right| \quad \text{(by (4))} \\
&= 2 \cdot \left| \Pr[M \circ G(U_l) = 1] - \Pr[M(U_{n \cdot k}) = 1] \right| \\
&\leq \epsilon. \quad \text{(by pseudorandomness of } G\text{)}
\end{aligned}$$

The lemma follows combining this with Equation (3). □

5.3 Amplification up to $1/2 - 1/\operatorname{poly}$

In this subsection we sketch our hardness amplification up to $1/2 - 1/n^c$, for every c:

Theorem 5.8. *If there is a balanced function $f : \{0,1\}^n \to \{0,1\}$ in $\mathcal{NP}$ that is $(1/3)$-hard for size $s(n) \geq n^{\omega(1)}$, then for every $c > 0$ there is a function $f' : \{0,1\}^m \to \{0,1\}$ in $\mathcal{NP}$ that is $(1/2 - 1/m^c)$-hard for size $(s(\sqrt{m}))^{\Omega(1)}$.*

To amplify we use the Tribes function of Ben-Or and Linial [BL], a monotone read-once DNF.

Definition 5.9. *The* Tribes *function on k bits is:*

$$\mathrm{Tribes}_k(x_1, \ldots, x_k) \stackrel{\text{def}}{=}$$
$$(x_1 \wedge \ldots \wedge x_b) \vee (x_{b+1} \wedge \ldots \wedge x_{2b}) \vee \ldots \vee (x_{k-b+1} \wedge \ldots \wedge x_k)$$

where there are k/b clauses each of size b, and b is the largest integer such that $(1 - 2^{-b})^{k/b} \geq 1/2$. Note that this makes $b = O(\log k)$.

The Tribes DNF has very low noise stability when perturbed with constant noise.

Lemma 5.10 ([O'D, MO])**.** *For every constant $\delta > 0$,*

$$\mathrm{NoiseStab}_\delta[\mathrm{Tribes}_k] \leq \frac{1}{k^{\Omega(1)}}.$$

A key step in our result is that Tribes_k is easily computable by a branching program of size $O(k)$, and therefore we can use Lemma 5.7 to fool its expected bias.

We now define the generator we will use in our derandomized direct product construction.

Definition 5.11. *Given n and $k \leq 2^n$, define the generator $\Gamma_k : \{0,1\}^m \to (\{0,1\}^n)^k$ as follows:*

$$\Gamma_k(x, y) \stackrel{\text{def}}{=} NW_k(x) \oplus N_k(y),$$

where $\oplus$ denotes bitwise XOR.

We recall the properties of Γ we are interested in:

Lemma 5.12. *The following hold:*

1. Γ_k *is indistinguishability-preserving for size* k^2.
2. Γ_k *is* 2^{-n}*-pseudorandom against branching programs of size* 2^n *and block-size* n.
3. Γ_k *has seed length* $m = O(n^2)$.
4. Γ_k *is explicitly computable (see Definition 4.2 for the definition of explicit).*

Proof. Item (1) follows from Lemma 5.3 and the fact that an indistinguishability-preserving generator XORed with any fixed string is still indistinguishability-preserving. More specifically, suppose, for the sake of contradiction, that $\Gamma_k(x, y) = NW_k(x) \oplus N_k(y)$ is not indistinguishability-preserving. Then there are functions $f_1, \ldots, f_k$ and $g_1, \ldots, g_k$ such that for every i the distributions $U_n \cdot f_i(U_n)$ and $U_n \cdot g_i(U_n)$ are indistinguishable, yet the distributions

$$(x, y) \cdot f_1(NW_1(x) \oplus N_1(y)) \cdots f_k(NW_k(x) \oplus N_k(y))$$

and

$$(x, y) \cdot g_1(NW_1(x) \oplus N_1(y)) \cdots g_k(NW_k(x) \oplus N_k(y))$$

are distinguishable (for random x, y). Then, by averaging, they are distinguishable for some fixed value of $y = \tilde{y}$. Thus, we obtain that

$$x \cdot f'_1(NW_1(x)) \cdots f'_k(NW_k(x))$$

and

$$x \cdot g'_1(NW_1(x)) \cdots g'_k(NW_k(x))$$

are distinguishable (for random x), where $f'_i(z) = f_i(z \oplus N_i(\tilde{y}))$, $g'_i(z) = g_i(z \oplus N_i(\tilde{y}))$. (Note that we hardwire the fixed part of the seed $\tilde{y}$ in the distinguisher.) Now observe that the indistinguishability of $U_n \cdot f_i(U_n)$ and $U_n \cdot g_i(U_n)$ implies the indistinguishability of $U_n \cdot f'_i(U_n)$ and $U_n \cdot g'_i(U_n)$, because the mapping $T(u \cdot v) = (u \oplus N_i(\tilde{y})) \cdot v$ transforms the latter pair of distributions to the former. (There is no loss in the circuit size assuming that circuits have input gates for both the input variables and their negations.) But this is a contradiction because NW is indistinguishability-preserving.

Item (2) follow from Theorem 5.6 and the fact that XORing with any fixed string (in particular, $NW_k(x)$ for any x) preserves pseudorandomness against branching programs.

Item (3) is an immediate consequence of the seed lengths of NW_k (Lemma 5.3) and N_k (Theorem 5.6).

Item (4) follows from the fact that NW_k is explicit (Lemma 5.3) and N_k is explicit (Theorem 5.6). $\square$

Proof of Theorem 5.8. Given $f : \{0,1\}^n \to \{0,1\}$ that is δ-hard for size $s(n)$ (for $\delta = 1/3$) and a constant c, let $k = n^{c'}$ for $c' = O(c)$ to be determined later. Consider the function $f' : \{0,1\}^m \to \{0,1\}$ defined by

$$f' \stackrel{\text{def}}{=} (\text{Tribes}_k \circ f^{\otimes k}) \circ \Gamma_k.$$

Note that $f' \in \mathcal{NP}$ since $f \in \mathcal{NP}$, Tribes is monotone and both Γ and Tribes are efficiently computable.

We now analyze the hardness of f'. Since Γ_k is indistinguishability-preserving for size k^2 by Lemma 5.12, Lemma 5.2 implies that there is a δ'-random function g (for $\delta/2 \leq \delta' \leq \delta$) such that f' has hardness

$$\frac{1}{2} - \frac{\mathrm{ExpBias}\left[(\mathrm{Tribes}_k \circ g^{\otimes k}) \circ \Gamma_k\right]}{2} - \frac{k}{s(n)^{1/3}} \tag{5}$$

for circuits of size $\Omega\left(s(n)^{1/3}\right) - k^2 - \mathrm{size}(\mathrm{Tribes}_k)$. Next we bound the hardness. By Lemma 5.12, we know that Γ_k is 2^{-n}-pseudorandom against branching programs of size 2^n and block-size n. In particular, since $k = \mathrm{poly}(n)$, Γ_k is $1/k$-pseudorandom against branching programs of size $9k$ and block-size n. Since, as we noted before, Tribes_k is easily computable by a branching program of size $O(k)$, we can apply Lemma 5.7 (noting that $O(k)^2 = \mathrm{poly}(n) \ll 2^n$) in order to bound $\mathrm{ExpBias}\left[(\mathrm{Tribes}_k \circ g^{\otimes k}) \circ \Gamma_k\right]$ by $\sqrt{\mathrm{NoiseStab}_{\delta'}[\mathrm{Tribes}_k] + 2/k}$. And the noise stability inside the square root is at most $1/k^{\Omega(1)}$ by Lemma 5.10. Since $k = \mathrm{poly}(n)$ and $s(n) = n^{\omega(1)}$, the $k/s^{1/3}$ term in the hardness (5) is negligible and we obtain hardness at least $1/2 - 1/k^{\Omega(1)}$.

We now bound the circuit size: Since Tribes_k is computable by circuits of size $O(k)$, and $s(n) = n^{\omega(1)}$, the size is at least $s(n)^{\Omega(1)}$.

Now note that f' has input length $m = m(n) = O(n^2)$ by Lemma 5.12. Strictly speaking, we have only defined f' for certain input lengths; however, it is easy to extend the function to every input length, by simply defining $f'(x) \stackrel{\text{def}}{=} f'(x')$ where x' consists of the first $m(n)$ bits of x and n is the largest integer such that $m(n) \leq |x|$. It is easy to check that f' still has hardness $1/2 - 1/k^{\Omega(1)} = 1/2 - 1/n^{\Omega(c')}$. The result then follows by an appropriate choice of $c' = O(c)$. □

5.4 Using Nondeterminism

In this subsection we discuss how to use nondeterminism to get the following theorem.

Theorem 5.13. *If there is a balanced function $f : \{0,1\}^n \to \{0,1\}$ in $\mathcal{NP}$ that is $(1/3)$-hard for size $s(n)$, then there is a function $f' : \{0,1\}^m \to \{0,1\}$ in $\mathcal{NP}$ that is $(1/2 - 1/s(\sqrt{m})^{\Omega(1)})$-hard for size $s(\sqrt{m})^{\Omega(1)}$.*

Our main observation is that Tribes_k is a DNF with clause size $O(\log k)$, and therefore it is computable in nondeterministic time $\mathrm{poly}(n)$ *even when k is superpolynomial in n*:

Lemma 5.14. *Let $f : \{0,1\}^n \to \{0,1\}$ be in $\mathcal{NP}$, and let $G_k : \{0,1\}^l \to (\{0,1\}^n)^k$ be any explicitly computable generator (see Definition 4.2) with $l \geq n$. Then the function $f' \stackrel{\text{def}}{=} (\mathrm{Tribes}_k \circ f^{\otimes k}) \circ G_k$ is computable in $\mathcal{NP}$ for every $k = k(n) \leq 2^n$.*

Proof. We compute $f'(\sigma)$ nondeterministically as follows: Guess a clause $v_i \wedge v_{i+1} \wedge \cdots \wedge v_j$ in Tribes_k. Accept if for every h s.t. $i \leq h \leq j$ we have $f(X_h) = 1$, where $G(\sigma) = (X_1, \ldots, X_k)$ and the values $f(X_h)$ are computed using the $\mathcal{NP}$ algorithm for f.

It can be verified that this algorithm has an accepting computation path on input σ iff $f'(\sigma) = 1$. Note that the clauses have size logarithmic in k, which is polynomial in n. Moreover, G is explicitly computable. The result follows. □

Now the proof of Theorem 5.13 proceeds along the same lines as the proof of Theorem 5.8, setting $k \stackrel{\text{def}}{=} s(n)^{\Omega(1)}$.

5.5 Amplifying from Hardness $1/\operatorname{poly}$

Our amplification from hardness $\Omega(1)$ to $1/2-\epsilon$ (Theorem 5.8) can be combined with O'Donnell's amplification from hardness $1/\operatorname{poly}$ to hardness $\Omega(1)$ to obtain an amplification from $1/\operatorname{poly}$ to $1/2-\epsilon$. However, since O'Donnell's construction blows up the input length polynomially, we would only obtain $\epsilon = 1/s(n^{\Omega(1)})$ (where the hidden constant depends on the initial polynomial hardness) rather than $\epsilon = 1/s(\sqrt{n})^{\Omega(1)}$ (as in Theorem 5.8). Thus we show here how to amplify directly from $1/\operatorname{poly}$ to $1/2-\epsilon$ using our approach. For this we need a combining function C that is more involved than the Tribes function. The properties of C that are needed in the proof of Theorem 4.1 are captured by the following lemma.

Lemma 5.15. *For every $\delta(n) = 1/n^{O(1)}$, there is a sequence of functions $C_k : \{0,1\}^k \to \{0,1\}$, such that for every $k = k(n)$ with $n^{\omega(1)} \le k \le 2^n$, the following hold:*

1. $\mathrm{NoiseStab}_\delta[C_k] \le 1/k^{\Omega(1)}$.
2. *For every $f : \{0,1\}^n \to \{0,1\}$ in $\mathcal{NP}$ and every explicitly computable generator (see Definition 4.2) $G_k : \{0,1\}^l \to (\{0,1\}^n)^k$ with $l \ge n$, the function $(C_k \circ f^{\otimes k}) \circ G_k$ is in $\mathcal{NP}$.*
3. *C_k can be computed by a branching program of size $\operatorname{poly}(n) \cdot k$, and also by a circuit of size $\operatorname{poly}(n) \cdot k$.*

Before proving Lemma 5.15, let us see how it can be used to prove our main theorem.

Theorem 5.16 (Thm. 4.1, restated)**.** *If there is a balanced function $f : \{0,1\}^n \to \{0,1\}$ in $\mathcal{NP}$ that is $1/\operatorname{poly}(n)$-hard for size $s(n)$, then there is a function $f' : \{0,1\}^m \to \{0,1\}$ in $\mathcal{NP}$ that is $(1/2 - 1/s(\sqrt{m})^{\Omega(1)})$-hard for size $s(\sqrt{m})^{\Omega(1)}$.*

Proof. Let $f : \{0,1\}^n \to \{0,1\}$ be a balanced function in $\mathcal{NP}$ that is $\delta = \delta(n)$-hard for size $s(n)$, where $\delta \ge 1/n^{O(1)}$. Let $k = k(n) \stackrel{\text{def}}{=} s(n)^{1/7}$ and let C_k be the function guaranteed by Lemma 5.15. Let Γ_k be the the generator from Definition 5.11. Consider the function $f' : \{0,1\}^m \to \{0,1\}$ defined by $f' \stackrel{\text{def}}{=} (C_k \circ f^{\otimes k}) \circ \Gamma_k$. Note that $f' \in \mathcal{NP}$ by Item 2 in Lemma 5.15.

We now analyze the hardness of f'. Since Γ_k is indistinguishability-preserving for size k^2 (by Lemma 5.12), Lemma 5.2 implies that there is a δ'-random function g (for $\delta/2 \le \delta' \le \delta$) such that f' has hardness

$$\alpha(m) = \frac{1}{2} - \frac{\mathrm{ExpBias}\left[(C_k \circ g^{\otimes k}) \circ \Gamma_k\right]}{2} - \frac{k}{s(n)^{1/3}} \tag{6}$$

for circuits of size

$$s'(m) = \Omega\left(\frac{s(n)^{1/3}}{\log(1/\delta)}\right) - k^2 - \mathrm{size}(C_k).$$

We first bound the hardness $\alpha(m)$. By Lemma 5.12, we know that Γ_k is 2^{-n}-pseudorandom against branching programs of size 2^n and block-size n. Since the branching program for computing C_k has size $\operatorname{poly}(n) \cdot k$, and $(\operatorname{poly}(n) \cdot k)^2 \ll 2^n$ (by our choice of $k(n)$), we may apply Lemma 5.7 in order to bound $\mathrm{ExpBias}\left[(C_k \circ g^{\otimes k}) \circ \Gamma_k\right]$ by $\sqrt{\mathrm{NoiseStab}_{\delta'}[C_k] + 2/2^n}$. This noise stability is at most $1/k^{\Omega(1)}$ by Item 1 in Lemma 5.15. Using the fact that $k = s(n)^{1/7}$, we have

$$\alpha(m) \ge \frac{1}{2} - \frac{\sqrt{1/k^{\Omega(1)} - 2/2^n}}{2} - \frac{k}{s(n)^{1/3}} = \frac{1}{2} - \frac{1}{s(n)^{\Omega(1)}}.$$

We now bound the circuit size $s'(m)$. Since C_k is computable by a circuit of size $\mathrm{poly}(n) \cdot k$ (by Item 3 in Lemma 5.15) and $\log(1/\delta) = O(\log n)$ and $s(n) = n^{\omega(1)}$, we have

$$s'(m) = \Omega\left(\frac{s(n)^{1/3}}{\log n}\right) - s(n)^{2/7} - \mathrm{poly}(n) = s(n)^{\Omega(1)}.$$

To conclude, we note that f' has input length $m = O(n^2)$ by Lemma 5.12, so $s(n) = s(\Omega(\sqrt{m})) = s(\sqrt{m})^{\Omega(1)}$, and we indeed obtain hardness $\alpha(m) = 1/2 - 1/s(\sqrt{m})^{\Omega(1)}$ for size $s'(m) = s(\sqrt{m})^{\Omega(1)}$. Strictly speaking, we have only defined f' for certain input lengths; however, it is easy to extend the function to every input length, cf. the end of the proof of Theorem 5.8. □

The rest of this subsection is devoted to the proof of Lemma 5.15. Recall that amplification from hardness $\Omega(1)$ (Theorem 5.8) relies on the fact that the Tribes DNF has low noise stability with respect to noise parameter $\delta = \Omega(1)$ (i.e., Lemma 5.10). Similarly, to amplify from hardness $1/\mathrm{poly}(n)$ we need to employ a combining function that has low noise stability with respect to noise $1/\mathrm{poly}(n)$. To this end, following [O'D], we employ the recursive-majorities function, RMaj_r. Let Maj denote the majority function.

Definition 5.17. *The* RMaj_r *function on* 3^r *bits is defined recursively by:*

$$\begin{aligned}
\mathrm{RMaj}_1(x_1, x_2, x_3) &\stackrel{\mathrm{def}}{=} \mathrm{Maj}(x_1, x_2, x_3) \\
\mathrm{RMaj}_r(x_1, \ldots, x_{3^r}) &\stackrel{\mathrm{def}}{=} \mathrm{RMaj}_{r-1}\left(\mathrm{Maj}(x_1, x_2, x_3), \ldots, \mathrm{Maj}(x_{3^r-2}, x_{3^r-1}, x_{3^r})\right)
\end{aligned}$$

The following lemma quantifies the noise stability of RMaj_r.

Lemma 5.18 ([O'D], Proposition 11)**.** *There is a constant c such that for every $\delta > 0$ and every $r \geq c \cdot \log(1/\delta)$, we have*

$$\mathrm{NoiseStab}_\delta[\mathrm{RMaj}_r] \leq \frac{1}{4}.$$

Note that if $r = O(\log n)$ then RMaj_r is a function of $3^r = \mathrm{poly}(n)$ bits.

However, when $r = O(\log n)$, RMaj_r does not have sufficiently low noise stability to be used on its own. For this reason, we will combine RMaj with Tribes. (The same combination of RMaj and Tribes is employed by O'Donnell [O'D], albeit for a different setting of parameters.)

Proof of Lemma 5.15. Given n and $\delta = \delta(n) \geq 1/n^{O(1)}$, let $r \stackrel{\mathrm{def}}{=} c \cdot \log(1/\delta)$ for a constant c to be chosen later. Assume, without loss of generality, that r and $k/3^r$ are integers. The function $C_k : \{0,1\}^k \to \{0,1\}$ is defined as follows

$$C_k \stackrel{\mathrm{def}}{=} \mathrm{Tribes}_{k/3^r} \circ \mathrm{RMaj}_r^{\otimes k/3^r}.$$

We now prove that C_k satisfies the required properties.

1. We will use the following result from [O'D].

   Lemma 5.19 ([O'D], Proposition 8)**.** *If h is a balanced boolean function and $\varphi : \{0,1\}^\ell \to \{0,1\}$ is any boolean function, then*

   $$\mathrm{NoiseStab}_\delta[\varphi \circ h^{\otimes \ell}] = \mathrm{NoiseStab}_{\frac{1}{2} - \frac{\mathrm{NoiseStab}_\delta[h]}{2}}[\varphi].$$

Letting c be a sufficiently large constant (recall that $r = c \cdot \log(1/\delta)$), by Lemma 5.18 we have that $\mathrm{NoiseStab}_\delta[\mathrm{RMaj}_r]/2 \geq 1/2 - 1/8 \geq 3/8$. Now note that RMaj_r is balanced because taking the bitwise complement of an input x also negates the value of $\mathrm{RMaj}_r(x)$. Hence, by Lemma 5.19,

$$\mathrm{NoiseStab}_\delta[\mathrm{Tribes}_{k/3^r} \circ \mathrm{RMaj}_r^{\otimes k/3^r}] = \mathrm{NoiseStab}_{3/8}[\mathrm{Tribes}_{k/3^r}] \leq \frac{1}{(k/3^r)^{\Omega(1)}} = \frac{1}{k^{\Omega(1)}},$$

where the last two equalities use Lemma 5.10 and the fact that $k = n^{\omega(1)}$ and $r = O(\log n)$.

2. The proof is similar to the proof of Lemma 5.14. To compute $(\mathrm{Tribes}_{k/3^r} \circ \mathrm{RMaj}_r^{\otimes k/3^r} \circ f^{\otimes k}) \circ G_k$, we guess a clause of the $\mathrm{Tribes}_{k/3^r}$ and verify that all the RMaj_r evaluations feeding into it are satisfied (using the $\mathcal{NP}$ algorithm for f). The only additional observation is that each of the recursive majorities depends only on $3^r = \mathrm{poly}(n)$ bits of the input, and hence can be computed in time polynomial in n.

3. As noted earlier, $\mathrm{Tribes}_{k/3^r}$ is easily computable by a branching program of size $O(k)$. RMaj_r, on the other hand, can be computed by a branching program of size $\mathrm{poly}(n)$. Indeed, $\mathrm{Maj}(x_1, x_2, x_3)$ is clearly computable by a branching program of constant size c, and thus $\mathrm{RMaj}_r = \mathrm{RMaj}_{r-1}\left(\mathrm{Maj}(x_1, x_2, x_3), \ldots, \mathrm{Maj}(x_{3^r-2}, x_{3^r-1}, x_{3^r})\right)$ can be computed by a branching program whose size is at most c times the size of RMaj_{r-1}. By induction it follows that RMaj_r can be computed in size $c^r = \mathrm{poly}(n)$.

   By composing the branching program of size $O(k)$ for Tribes with the branching program of size $\mathrm{poly}(n)$ for RMaj, we can compute C_k by a branching program of size $\mathrm{poly}(n) \cdot k$.

□

A natural question is whether one can amplify from hardness $1/n^{\omega(1)}$. A modification of the [Vio1] negative result about hardness amplification given in [LTW] shows that this task cannot be achieved by any black-box hardness amplification computable in the polynomial-time hierarchy (and thus in particular cannot be achieved by any black-box hardness amplification computable in $\mathcal{NP}$). (See Definition 7.2 for the definition of black-box hardness amplification.)

6 Extensions

6.1 On the Possibility of Amplifying Hardness up to $1/2 - 1/2^{\Omega(n)}$

Even when starting from a function that is δ-hard for size $2^{\Omega(n)}$, our results (Theorem 4.1) only amplify hardness up to $1/2 - 1/2^{\Omega(\sqrt{n})}$ (rather than $1/2 - 1/2^{\Omega(n)}$). In this section we discuss the possibility of amplifying hardness in $\mathcal{NP}$ up to $1/2 - 1/2^{\Omega(n)}$, when starting with a function that is δ-hard for size $2^{\Omega(n)}$. The problem is that the seed length of our generator in Lemma 5.12 is *quadratic* in n, rather than linear. To amplify hardness up $1/2 - 1/2^{\Omega(n)}$ we need a generator (for every $k = 2^{\Omega(n)}$) with the same properties of the one in Lemma 5.12, but with linear seed length.

Recall our generator is the XOR of an indistinguishability-preserving generator and a generator that is pseudorandom against branching programs. While it is an open problem to exhibit a generator with linear seed length that is pseudorandom against branching programs, an indistinguishability-preserving generator with linear seed length is given by the following lemma.

Lemma 6.1. *For every constant γ, $0 < \gamma < 1$, there is a constant c such that for every n there is an explicitly computable generator $NW'_{2^{n/c}} : \{0,1\}^l \to (\{0,1\}^n)^{2^{n/c}}$ with seed length $l = c \cdot n$ that is indistinguishability-preserving for size $2^{\gamma \cdot n}$.*

The generator is due to Nisan and Wigderson [NW] and Impagliazzo and Wigderson [IW1]. The approach is the same as for the generator used in Lemma 5.3, except we now require a design consisting of $2^{\Omega(n)}$ sets of size n in a universe of size $O(n)$, with pairwise intersections of size at most $\gamma n/2$. An explicit construction of such a design is given in [GV].[6]

Theorem 6.2. *Suppose that there exists an explicit generator $N'_{2^n} : \{0,1\}^l \to (\{0,1\}^n)^{2^n}$ that is 2^{-n}-pseudorandom against branching programs of size 2^n and block-size n and that has seed length $l = O(n)$. Then the following holds: If there is a balanced function $f : \{0,1\}^n \to \{0,1\}$ in $\mathcal{NP}$ that is $1/\operatorname{poly}(n)$-hard for size $2^{\Omega(n)}$, then there is a function $f' : \{0,1\}^m \to \{0,1\}$ in $\mathcal{NP}$ that is $(1/2 - 1/2^{\Omega(n)})$-hard for size $2^{\Omega(n)}$.*

A slightly more careful analysis shows that all the branching programs considered in our constructions have *width*[7] much smaller than their size (specifically, the branching program for Tribes has constant width, and the one for the function in Lemma 5.15, i.e. Tribes composed with RMaj, has width $\operatorname{poly}(n)$ independent of k). Thus to prove the conclusion in the statement of Theorem 6.2 it would suffice to exhibit an explicit pseudorandom generator that fools such restricted branching programs.

For amplifying from constant hardness, it suffices to instead have a generator fooling *constant-depth circuits* of size 2^n with seed length $O(n)$. (The generator of Nisan [Nis1] has seed length $\operatorname{poly}(n)$.) The reason is that our proof that PRGs versus branching programs "fool" the expected bias also works for PRGs versus constant-depth circuits, provided that the combining function is computable in constant depth. The Tribes function is depth 2 by definition (but the recursive majorities RMaj is not constant-depth, and hence this would only amplify from constant hardness).

More generally, we only need, for every constant $\gamma > 0$, a generator $G : \{0,1\}^{O(n)} \to (\{0,1\}^n)^k$ where $k = 2^{\gamma n}$ such that for every δ-random function g,

$$\operatorname{ExpBias}\left[(C_k \circ g^{\otimes k}) \circ G\right] = 2^{-\Omega(n)},$$

where, for example, $C_k = \mathrm{Tribes}_k$ (when δ is constant). As in the proof of Lemma 3.7, in proving such a statement, it may be convenient to work instead with the (polynomially related) expected collision probability. An important property of $C_k = \mathrm{Tribes}_k$ we used in bounding the expected bias with respect to G is that it gives expected bias $2^{-\Omega(n)}$ if G is replaced with a truly random generator (i.e. using seed length $n \cdot k$) and δ is constant. One might try to use a different monotone combining function with this property, provided it can also be evaluated in nondeterministic time $\operatorname{poly}(n)$.

6.2 Impagliazzo and Wigderson's Hardness Amplification

Our framework gives a new proof of the hardness amplification by Impagliazzo and Wigderson [IW1]. Impagliazzo and Wigderson [IW1] show that if $\mathcal{E} \stackrel{\text{def}}{=} DTIME\left(2^{O(n)}\right)$ contains a function $f : \{0,1\}^n \to \{0,1\}$ that requires (in the worst-case) circuits of size $2^{\Omega(n)}$, then $\mathcal{E}$ contains a function f' that is $\left(1/2 - 1/2^{\Omega(n)}\right)$-hard for circuits of size $2^{\Omega(n)}$. The main contribution in [IW1] is amplification from constant hardness, e.g. 1/3, to $\left(1/2 - 1/2^{\Omega(n)}\right)$ (amplification from worst-case hardness and constant hardness was essentially already established in [BFNW, Imp1]). The

[6] Alternatively, we can use the randomized algorithm described in [IW1] that computes such sets $S_1, \ldots, S_M$ with probability exponentially close to 1 using $O(n)$ random bits. This is sufficient for constructing an indistinguishability-preserving generator.

[7] The width of a branching program is the maximum, over i, of the number of states that are reachable after reading i symbols.

improvement over the standard Yao XOR Lemma is that the input length of the amplified function increases only by a constant factor. In this section, we sketch a simple proof of this result using the framework developed in earlier sections. While other, more recent hardness amplifications achieving the same result for $\mathcal{E}$ are known [STV, SU, Uma], the original one by Impagliazzo and Wigderson is still interesting because it can be implemented in "low" complexity classes, such as the polynomial-time hierarchy, while the others cannot (due to the fact that they actually amplify from worst-case hardness [Vio1]).

The construction of [IW1] uses an *expander-walk* generator $W_k : \{0,1\}^l \to (\{0,1\}^n)^k$, which uses its seed of length $l = n + O(k)$ to do a random walk of length k (started at a random vertex) in a constant-degree expander graph on 2^n vertices. More background on such generators can be found in [Gol2, Sec 3.6.3]. The construction of [IW1] XORs the expander-walk generator with the (first k outputs of the) indistinguishability-preserving generator from Lemma 6.1:

Definition 6.3. *Let $k = c \cdot n$ for a constant $c > 1$. Let $NW''_k : \{0,1\}^{O(n)} \to (\{0,1\}^n)^k$ be a generator that is indistinguishability-preserving for size $2^{n/c}$ as given by Lemma 6.1. The generator $IW_k : \{0,1\}^l \to (\{0,1\}^n)^k$ is defined as*

$$IW_k(x, y) \stackrel{\text{def}}{=} NW''_k(x) \oplus W_k(y).$$

The seed length of IW_k is $l = O(n)$.

Given a function f that is $1/3$-hard for size $s = 2^{\Omega(n)}$, the Impagliazzo–Wigderson amplification defines

$$f' \stackrel{\text{def}}{=} (XOR \circ f^{\otimes k}) \circ IW_k : \{0,1\}^{O(n)} \to \{0,1\},$$

where $k = c \cdot n$ for a constant c that depends on the hidden constant in the $s = 2^{\Omega(n)}$. They prove the following about this construction.

Theorem 6.4 ([IW1])**.** *If there is a function $f : \{0,1\}^n \to \{0,1\}$ in $\mathcal{E}$ that is $1/3$-hard for size $2^{\Omega(n)}$, then there is a function $f' : \{0,1\}^m \to \{0,1\}$ in $\mathcal{E}$ that is $(1/2 - 2^{-\Omega(m)})$-hard for size $2^{\Omega(m)}$.*

Proof. By Theorem 5.2 there exists a δ'-random function $g : \{0,1\}^n \to \{0,1\}$, where δ' is a constant, such that the hardness of $f' : \{0,1\}^{O(n)} \to \{0,1\}$ is $1/2 - \text{ExpBias}\left[(XOR \circ g^{\otimes k}) \circ IW_k\right] - 2^{-\Omega(n)}$ for circuits of size $2^{\Omega(n)}$.

We now bound the hardness. Whenever some $IW_i(x)$ falls in the set of inputs of density $2 \cdot \delta'$ where the output of g is a coin flip, the bias of $(XOR \circ g^{\otimes k}) \circ IW_k$ is 0. Therefore

$$\text{ExpBias}\left[(XOR \circ g^{\otimes k}) \circ IW_k\right] \le \Pr_x[\forall i : IW_i(x) \notin H] \le 2^{-\Omega(n)},$$

where in the last inequality we use standard hitting properties of expander walks (see e.g. [Gol1] for a proof), and take c to be a sufficiently large constant. □

7 Limitations of Monotone Hardness Amplification

7.1 On the hypothesis that f is balanced

The hardness amplification results in the previous sections start from balanced functions. In this section we study this hypothesis. Our main finding is that, while this hypothesis is not necessary for hardness amplification within $\mathcal{NP}/\text{poly}$ (i.e., non-deterministic polynomial size circuits), it is likely to be necessary for hardness amplification within $\mathcal{NP}$.

To see that this hypothesis is not necessary for amplification within $\mathcal{NP}/\operatorname{poly}$, note that if the quantity $\Pr_x[f(x) = 1]$ of the original hard function $f : \{0,1\}^n \to \{0,1\}$ is known, then we can easily pad f to obtain a balanced function $\bar{f} : \{0,1\}^{n+1} \to \{0,1\}$:

$$\bar{f}(x,p) \stackrel{\text{def}}{=} \begin{cases} f(x) & \text{if } p = 0 \\ 0 & \text{if } p = 1 \text{ and } x \le \Pr_x[f(x) = 1] \cdot 2^n \\ 1 & \text{if } p = 1 \text{ and } x > \Pr_x[f(x) = 1] \cdot 2^n \end{cases}$$

It is easy to see that $\bar{f}$ is $1/\operatorname{poly}(n)$-hard if f is. Since a circuit can (non-uniformly) know $\Pr_x[f(x) = 1]$, the following hardness amplification within $\mathcal{NP}/\operatorname{poly}$ is a corollary to the proof of Theorem 4.1.

Corollary 7.1. *If there is a function $f : \{0,1\}^n \to \{0,1\}$ in $\mathcal{NP}/\operatorname{poly}$ that is $1/\operatorname{poly}(n)$-hard for size $s(n)$, then there is a function $f' : \{0,1\}^m \to \{0,1\}$ in $\mathcal{NP}/\operatorname{poly}$ that is $(1/2 - 1/s(\sqrt{m})^{\Omega(1)})$-hard for size $s(\sqrt{m})^{\Omega(1)}$.*

Now we return to hardness amplification within $\mathcal{NP}$. First we note that, in our results, to amplify the hardness of $f : \{0,1\}^n \to \{0,1\}$ up to $1/2 - \epsilon$ it is only necessary that $\text{Bias}\,[f] \le \epsilon^c$ for some universal constant c. The argument is standard and can be found, for example, in [Tre].

Combining this observation with the above padding technique, O'Donnell constructs several candidate hard functions, one for each "guess" of the bias of the original hard function. He then combines them in a single function using a different input length for each candidate; this gives a function that is very hard on average for infinitely many input lengths. However, this approach, even in conjunction with derandomization and nondeterminism, cannot give better hardness than $1/2 - 1/n$. (Roughly speaking, if we want to amplify to $1/2 - \epsilon$, then we will have at least $1/\epsilon$ different candidates and thus the "hard" candidate may have input length $n \ge 1/\epsilon$, which means $1/2 - \epsilon \le 1/2 - 1/n$.)

To what extent can we amplifiy the hardness of functions whose bias is *unknown*? Non-monotone hardness amplifications, such as Yao's XOR Lemma, work regardless of the bias of the original hard function. However, in the rest of this section we show that, for hardness amplifications that are *monotone* and *black-box*, this is impossible. In particular, we show that black-box monotone hardness amplifications cannot amplify the hardness beyond the bias of the original function.

We now formalize the notion of black-box monotone hardness amplification and then state our negative result.

Definition 7.2. *An oracle algorithm $Amp : \{0,1\}^l \to \{0,1\}$ is a* black-box β-bias $[\delta \mapsto (1/2 - \epsilon)]$-hardness amplification for length n and size s *if for every $f : \{0,1\}^n \to \{0,1\}$ such that* $\text{Bias}\,[f] \le \beta$ *and for every $A : \{0,1\}^l \to \{0,1\}$ such that*

$$\Pr[A(U_l) \ne Amp^f(U_l)] \le 1/2 - \epsilon,$$

there is an oracle circuit C of size at most s such that

$$\Pr[C^A(U_n) \ne f(U_n)] \le \delta.$$

Amp is monotone *if for every x, $Amp^f(x)$ is a monotone function of the truth table of f.*

Note that if *Amp* is as in Definition 7.2 and if f is δ-hard for size s' and $\text{Bias}\,[f] \le \beta$, then Amp^f is $(1/2 - \epsilon)$-hard for size s'/s: if there were a circuit A of size s'/s computing Amp^f with error probability at most $1/2 - \epsilon$, then C^A would be a circuit of size $s \cdot (s'/s) = s'$ computing f with error probability at most δ, contradicting the hardness of f. The term "black box" refers to the fact that the definition requires this to hold for *every* f and A, regardless of whether or not f is in $\mathcal{NP}$ and A is a small circuit.

The following theorem shows that any *monotone* black-box hardness amplification up to $1/2 - \epsilon$ must start from functions of bias $\beta \approx \epsilon$.

Theorem 7.3. *For any constant $\gamma > 0$, if Amp is a monotone black-box β-bias $[\delta \mapsto (1/2 - \epsilon)]$-hardness amplification for length n and size $s \leq 2^{n/3}$ such that $1/2 - 4\epsilon > \delta + \gamma$, then $\beta \leq 8\epsilon + O(2^{-n})$.*

The main ideas for proving this bound are the same as in the negative result for black-box hardness amplification in [Vio1]: First we show that the above kind of hardness amplification satisfies certain coding-like properties. Roughly, *Amp* can be seen as a kind of list-decodable code where the distance property is guaranteed only for δ-distant messages with bias at most β (cf. [Tre]). Then we show that monotone functions fail to satisfy these properties. The limitation we prove on monotone functions relies on the following corollary to the Kruskal-Katona theorem (see [And], Theorem 7.3.1).

Lemma 7.4. *Let $\mathcal{S} = \{S_1, \ldots, S_m\}$ be a collection of m subsets $S_i \subseteq \{1, \ldots, N\}$, where $|S_i| = t$. If $m \geq \binom{N-1}{t} = \left(1 - \frac{t}{N}\right)\binom{N}{t}$ then for every integer $t' < t$*

$$\left|\{S : |S| = t', S \subseteq S_i \text{ for some } i\}\right| \geq \binom{N-1}{t'} = \left(1 - \frac{t'}{N}\right)\binom{N}{t'}.$$

Let $\mathcal{F}_p$ be the uniform distribution on functions f whose truth-tables have relative Hamming weight exactly p, i.e. $\Pr_x[f(x) = 1] = p$. We use the above Lemma 7.4 to prove the following fact.

Lemma 7.5. *Let $A : \{0,1\}^l \to \{0,1\}$ be a an oracle function such that for every $x \in \{0,1\}^l$, $A^f(x)$ is a* monotone *function of the truth-table of $f : \{0,1\}^n \to \{0,1\}$. (For example, any monotone black-box hardness amplification Amp satisfies this condition.) Let τ be an integer multiple of $1/2^n$. Then there is $p \in \{1/2 - \tau, 1/2 + \tau\}$ such that:*

$$\mathrm{E}_{U_l}\left[\mathrm{Bias}_{F \leftarrow \mathcal{F}_p}[A^F(U_l)]\right] \geq \tau.$$

Proof. We show that for every fixed $x \in \{0,1\}^l$ there is a $p \in \{1/2 - \tau, 1/2 + \tau\}$ such that $\mathrm{Bias}_{F \leftarrow \mathcal{F}_p}\left[A^F(x)\right] \geq 2\tau$. The theorem then follows easily. Fix x. For every p define S_p as the set of functions f in $\mathcal{F}_p$ such that $A^f(x) = 0$. Note that

$$\mathrm{Bias}_{F \leftarrow \mathcal{F}_p}\left[A^F(x)\right] = \left|1 - \frac{2|S_p|}{|\mathcal{F}_p|}\right|.$$

If $|S_{1/2+\tau}| < (1/2 - \tau) \cdot |\mathcal{F}_{1/2+\tau}|$ then $\mathrm{Bias}_{F \leftarrow \mathcal{F}_{1/2+\tau}}\left[A^F(x)\right] > 2\tau$ and we are done. Otherwise,

$$|S_{1/2+\tau}| \geq (1/2 - \tau) \cdot |\mathcal{F}_{1/2+\tau}| = (1 - (1/2 + \tau)) \cdot \binom{2^n}{(1/2 + \tau)2^n}.$$

View the elements $f \in S_p$ as subsets of $\{1, \ldots, 2^n\}$ of size exactly $p2^n$ in the natural way; i.e. the subset associated with f is the set of inputs x such that $f(x) = 1$. Note that if $f' \subseteq f \subseteq \{1, \ldots, 2^n\}$ and $A^f(x) = 0$ then by the monotonicity of A, $A^{f'}(x) = 0$. Therefore, by Lemma 7.4,

$$|S_{1/2-\tau}| \geq (1 - (1/2 - \tau)) \cdot \binom{2^n}{(1/2 - \tau)2^n} = (1/2 + \tau) \cdot |\mathcal{F}_{1/2-\tau}|,$$

and so $\mathrm{Bias}_{F \leftarrow \mathcal{F}_{1/2-\tau}}\left[A^F(x)\right] \geq 2\tau$. □

The following lemma captures the coding-like properties of monotone, black-box hardness amplifications — it shows that it is very unlikely that Amp^f for a "random f" will land in any fixed Hamming ball of radius $1/2-\epsilon$. For two functions f_1, f_2, let *Dist* denote the relative Hamming distance of their truth tables, i.e. $Dist(f_1, f_2) \stackrel{\text{def}}{=} \Pr_x[f_1(x) \neq f_2(x)]$.

Lemma 7.6. *Let $\gamma > 0$ be any fixed constant. Let Amp be a monotone black-box 8ϵ-bias $[\delta \mapsto (1/2-\epsilon)]$-hardness amplification for length n and size $s \leq 2^{n/3}$, where $1/2 - 4\epsilon > \delta + \gamma$, and let $\epsilon > 0$ be an integer multiple of $1/2^n$. Then for both p in $\{1/2-4\epsilon, 1/2+4\epsilon\}$ and every function G:*

$$\Pr_{F \leftarrow \mathcal{F}_p}[Dist(G, Amp^F) \leq 1/2 - \epsilon] \leq 2^{-\Omega(2^n)}.$$

Proof. Let $N \stackrel{\text{def}}{=} 2^n$. For every function f of bias at most 8ϵ such that $Dist(G, Amp^f) \leq 1/2 - \epsilon$, there must exist a circuit of size s, with oracle access to G, that computes f with error at most δ. Therefore, since there are $2^{O(s \log s)}$ circuits of size s and no more than $2^{H(\delta)N}$ functions that are at distance at most δ from f, there are at most $2^{O(s \log s)} 2^{H(\delta)N}$ such functions. Thus, when we restrict our attention to the $\binom{N}{pN}$ functions in $\mathcal{F}_p$ (for $p \in \{1/2-4\epsilon, 1/2+4\epsilon\}$), we have:

$$\begin{aligned}
\Pr_{F \leftarrow \mathcal{F}_p}[Dist(G, Amp^F) \leq 1/2 - \epsilon] &\leq \frac{2^{O(s \log s)} \cdot 2^{H(\delta)N}}{\binom{N}{pN}} \\
&\leq 2^{O(s \log s)} \cdot (N+1) \cdot 2^{(H(\delta) - H(p))N} \\
&\leq 2^{O(s \log s)} \cdot (N+1) \cdot 2^{(H(\delta) - H(1/2-4\epsilon))N} \\
&\leq 2^{-\Omega(N)},
\end{aligned}$$

where the second inequality follows from the fact that $\binom{N}{pN} \geq 2^{H(p)N}/(N+1)$,[8] and the last inequality uses the fact that $1/2 - 4\epsilon > \delta + \gamma$ implies $H(1/2 - 4\epsilon) > H(\delta) + \Omega(1)$. $\square$

Proof of Theorem 7.3. We assume that ϵ is a positive integer multiple of $1/2^n$ and show that $\beta \leq 8\epsilon$. The theorem then follows for general ϵ by rounding it up to the next integer multiple of $1/2^n$. We show that $\beta = 8\epsilon$ is impossible, and therefore that $\beta < 8\epsilon$ (because any β'-bias hardness amplification is clearly also a β-bias hardness amplification for every $\beta \leq \beta'$).

Suppose, for the sake of contradiction, that $\beta = 8\epsilon$.

By Lemma 7.5, we may choose $p \in \{1/2-4\epsilon, 1/2+4\epsilon\}$ such that

$$\mathrm{E}_{U_l}\left[\mathrm{Bias}_{F \leftarrow \mathcal{F}_p}[Amp^F(U_l)]\right] \geq 4\epsilon.$$

Define the function $G(x) \stackrel{\text{def}}{=} \mathrm{Maj}_{F \leftarrow \mathcal{F}_p} Amp^F(x)$, and consider

$$\Pr_{U_l, F \leftarrow \mathcal{F}_p}[Amp^F(U_l) \neq G(U_l)]. \tag{7}$$

[8] Actually, $\binom{N}{pN}$ is asymptotically $\Theta\left(2^{H(p)N}/\sqrt{N}\right)$, but for our purpose the easier estimate stated suffices.

We have:

$$\begin{aligned}
& \Pr_{U_l, F \leftarrow \mathcal{F}_p}[Amp^F(U_l) \neq G(U_l)] \\
= \quad & \mathrm{E}_{U_l}\left[\Pr_{F \leftarrow \mathcal{F}_p}[Amp^F(U_l) \neq G(U_l)]\right] \\
= \quad & \mathrm{E}_{U_l}\left[\frac{1}{2} - \frac{\mathrm{Bias}_{F \leftarrow \mathcal{F}_p}[Amp^F(U_l)]}{2}\right] \qquad \text{(by def. of bias and } G\text{)} \\
= \quad & \frac{1}{2} - \frac{\mathrm{E}_{U_l}\left[\mathrm{Bias}_{F \leftarrow \mathcal{F}_p}[Amp^F(U_l)]\right]}{2} \\
\leq \quad & \frac{1}{2} - 2\epsilon. \qquad \text{(by the choice of } p\text{)}
\end{aligned}$$

On the other hand, Lemma 7.6 implies that quantity (7) is at least $1/2 - \epsilon - 2^{\Omega(2^n)}$ (note that the hypothesis of the lemma is satisfied by our assumption that $1/2 - 4\epsilon \geq \delta + \gamma$).

Combining the two bounds, we have that

$$1/2 - 2\epsilon \geq \Pr_{U_l, F \leftarrow \mathcal{F}_p}[Amp^F(U_l) \neq G(U_l)] \geq 1/2 - \epsilon - 2^{-\Omega(2^n)},$$

which is a contradiction for sufficiently large n (by the assumption that ϵ is a positive multiple of $1/2^n$). □

7.2 Nondeterminism is necessary

In this subsection we show that *deterministic*, monotone, non-adaptive black-box hardness amplifications cannot amplify hardness beyond $1/2 - 1/\operatorname{poly}(n)$. Thus, the use of *nondeterminism* in our results (Section 5.4) seems necessary. Note that most hardness amplifications, including the one in this paper, are black-box and non-adaptive.

O'Donnell [O'D] proves that any monotone "direct product construction" (i.e. $f'(x_1, \ldots, x_k) = C(f(x_1), \ldots, f(x_k))$, as in Equation 1) cannot amplify to hardness better than $1/2 - 1/n$, assuming that the amplification works for all functions f (not necessarily in $\mathcal{NP}$). We relax the assumption that the hardness amplification is a direct product construction (allowing any monotone nonadaptive oracle algorithm $f' = Amp^f$). On the other hand, we require that the reduction proving its correctness is also black-box (as formalized in Definition 7.2).

We prove our bound even for hardness amplifications that amplify only balanced functions (i.e. $\beta = 0$ in Definition 7.2).

Theorem 7.7. *For every constant $\delta < 1/2$, if Amp is a black-box 0-bias $[\delta \mapsto (1/2 - \epsilon)]$-hardness amplification for length n and size $s \leq 2^{n/3}$ such that for every x, $Amp^f(x)$ is a monotone function of $k \leq 2^{n/3}$ values of f, then*

$$\epsilon \geq \Omega\left(\frac{\log^2 k}{k}\right).$$

The proof of this result follows closely the proof of the negative result on hardness amplification in [Vio1]. The main difference is here we use bounds on the noise stability of monotone functions rather than constant depth circuits.

The following lemma is similar to Lemma 7.6. The only difference is in considering functions F at distance η from f; this will correspond to perturbing the monotone amplification-function with noise having parameter η.

Lemma 7.8. *Let Amp be as in Definition 7.2 with $\beta = 0$ and $s \leq 2^{n/3}$. Then for any constant $\delta < 1/2$ there is a constant $\eta < 1/2$ such that, for sufficiently large n, the following holds: If $f : \{0,1\}^n \to \{0,1\}$ is any fixed balanced function and $F : \{0,1\}^n \to \{0,1\}$ is a random balanced function such that $Dist(f, F) = \eta$, then*

$$\Pr_F[Dist(Amp^f, Amp^F) \leq 1/2 - \epsilon] \leq \epsilon.$$

Proof. Let $N \stackrel{\text{def}}{=} 2^n$. It is easy to see that F is uniform on a set of size $\binom{N/2}{\eta N/2}^2$. The rest of the proof is like the proof of Lemma 7.6:

$$\begin{aligned}
\Pr[Dist(Amp^f, Amp^F) \leq 1/2 - \epsilon] &\leq \frac{2^{O(s \log s)} 2^{H(\delta)N}}{\binom{N/2}{\eta N/2}^2} \\
&\leq 2^{O(s \log s)} \cdot (N/2 + 1)^2 \cdot 2^{(H(\delta) - H(\eta))N} \\
&\leq \epsilon,
\end{aligned}$$

where the last inequality holds for a suitable choice of $\eta < 1/2$, using the fact that $\delta < 1/2$ is a constant and that $s \leq 2^{n/3}$. □

Proof of Theorem 7.7. Let η be the constant in Lemma 7.8. The idea is to consider

$$\Pr_{U_l, F, F'}[Amp^F(U_l) \neq Amp^{F'}(U_l)], \tag{8}$$

where F is a random balanced function and F' is a random balanced function such that $Dist(F, F') = \eta$.

By the above lemma, the probability (8) is at least $1/2 - 2\epsilon$.

On the other hand, for every fixed x, $Amp^F(x)$ is a monotone function depending only on k bits of the truth-table of the function F. Since k is small compared to 2^n, the distribution (F, F') induces on the input of $Amp^F(x)$ a distribution very close to $(U_k, U_k \oplus \mu)$, where μ is a noise vector with parameter η. Specifically, it can be verified that the statistical difference between these two distributions is at most $O(k^2/(\eta 2^n))$. Because this value is dominated by $\log^2 k/k$ when $k \leq 2^{n/3}$, and because $Amp^F(x)$ is a *monotone* function of k bits, we may apply Theorem 3.10 to conclude that the probability (8) is at most $1/2 - O(\log^2 k/k)$.

Combining the two bounds, we have that $1/2 - O(\log^2 k/k) \geq 1/2 - 2\epsilon$ and the results follows. □

8 Acknowledgments

We thank Ryan O'Donnell and Rocco Servedio for an email exchange about noise stability. We thank Richard Stanley for pointing out the Kruskal-Katona theorem. We also thank Oded Goldreich, Luca Trevisan, Avi Wigderson, and the anonymous reviewers for helpful suggestions.

References

[And] I. Anderson. *Combinatorics of finite sets.* Dover Publications Inc., Mineola, NY, 2002. Corrected reprint of the 1989 edition.

[BFL] L. Babai, L. Fortnow, and C. Lund. Nondeterministic exponential time has two-prover interactive protocols. *Computational Complexity*, 1(1):3–40, 1991.

[BFNW] 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.

[BF] D. Beaver and J. Feigenbaum. Hiding Instances in Multioracle Queries. In *7th Annual Symposium on Theoretical Aspects of Computer Science*, volume 415 of *Lecture Notes in Computer Science*, pages 37–48. Springer, 1990.

[BL] M. Ben-Or and N. Linial. Collective Coin-Flipping. In S. Micali, editor, *Randomness and Computation*, pages 91–115. Academic Press, New York, 1990.

[BT] A. Bogdanov and L. Trevisan. On Worst-Case to Average-Case Reductions for NP Problems. In *44th Annual Symposium on Foundations of Computer Science*, pages 308–317. IEEE, 2003.

[CPS] J.-Y. Cai, A. Pavan, and D. Sivakumar. On the Hardness of the Permanent. In *16th International Symposium on Theoretical Aspects of Computer Science*, Lecture Notes in Computer Science, Volume 1563, pages 90–99. Springer-Verlag, 1999.

[FL] U. Feige and C. Lund. On the Hardness of Computing the Permanent of Random Matrices. *Computational Complexity*, 6(2):101–132, 1996.

[FF] J. Feigenbaum and L. Fortnow. Random-Self-Reducibility of Complete Sets. *SIAM J. on Computing*, 22(5):994–1005, 1993.

[For] L. Fortnow. Balanced NP sets. *Computational Complexity Weblog*, 2003. `http://weblog.fortnow.com/archive/2003_09_07_archive.html`.

[Gol1] O. Goldreich. A Sample of Samplers - A Computational Perspective on Sampling (survey). *Electronic Colloquium on Computational Complexity (ECCC)*, 4(020), 1997.

[Gol2] O. Goldreich. *Modern cryptography, probabilistic proofs and pseudorandomness*, volume 17 of *Algorithms and Combinatorics*. Springer-Verlag, Berlin, 1999.

[Gol3] O. Goldreich. *Foundations of Cryptography: Volume 1, Basic Tools*. Cambridge University Press, Cambridge, 2001.

[GL] O. Goldreich and L. A. Levin. A Hard-Core Predicate for all One-Way Functions. In *Proceedings of the Twenty First Annual ACM Symposium on Theory of Computing*, pages 25–32, 1989.

[GNW] O. Goldreich, N. Nisan, and A. Wigderson. On Yao's XOR lemma. Technical Report TR95–050, Electronic Colloquium on Computational Complexity, 1995. `http://www.eccc.uni-trier.de/eccc`.

[GV] D. Gutfreund and E. Viola. Fooling Parity Tests with Parity Gates. In *Proceedings of the Eight International Workshop on Randomization and Computation (RANDOM)*, Lecture Notes in Computer Science, Volume 3122, pages 381–392. Springer-Verlag, 2004.

[HVV] A. Healy, S. Vadhan, and E. Viola. Using nondeterminism to amplify hardness. In *Proceedings of the Thirty-Six Annual ACM Symposium on the Theory of Computing*, pages 192–201, 2004.

[Imp1] R. Impagliazzo. Hard-core distributions for somewhat hard problems. In *36th Annual Symposium on Foundations of Computer Science*, pages 538–545. IEEE, 1995.

[Imp2] R. Impagliazzo. A Personal View of Average-Case Complexity. In *Proceedings of the Tenth Annual Structure in Complexity Theory Conference*, pages 134–147. IEEE, 1995.

[IW1] 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, 1997.

[IW2] R. Impagliazzo and A. Wigderson. Randomness vs time: derandomization under a uniform assumption. *J. Comput. System Sci.*, 63(4):672–688, 2001. Special issue on FOCS 98.

[KKL] 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. IEEE, 1988.

[KS] A. Klivans and R. A. Servedio. Boosting and Hard-Core Sets. *Machine Learning*, 53(3):217–238, 2003.

[Lev] L. A. Levin. Average Case Complete Problems. *SIAM J. on Computing*, 15(1):285–286, 1986.

[Lip] R. Lipton. New Directions in Testing. In *Proceedings of DIMACS Workshop on Distributed Computing and Cryptography*, volume 2, pages 191–202. ACM/AMS, 1991.

[LTW] C.-J. Lu, S.-C. Tsai, and H.-L. Wu. On the Complexity of Hardness Amplification. In *Proceedings of the Twentieth Annual Conference on Computational Complexity*, pages 170–182. IEEE, 2005.

[MO] E. Mossel and R. O'Donnell. On the noise sensitivity of monotone functions. *Random Struct. Algorithms*, 23(3):333–350, 2003.

[Nis1] N. Nisan. Pseudorandom bits for constant depth circuits. *Combinatorica*, 11(1):63–70, 1991.

[Nis2] N. Nisan. Pseudorandom Generators for Space-bounded Computation. *Combinatorica*, 12(4):449–461, 1992.

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

[O'D] R. O'Donnell. Hardness Amplification Within NP. *J. Comput. Syst. Sci.*, 69(1):68–94, 2004.

[SU] R. Shaltiel and C. Umans. Simple extractors for all min-entropies and a new pseudorandom generator. *J. ACM*, 52(2):172–216, 2005.

[STV] 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 (1999).

[Tre] L. Trevisan. List Decoding Using the XOR Lemma. In *44th Annual Symposium on Foundations of Computer Science*, pages 126–135. IEEE, 2003.

[TV] L. Trevisan and S. Vadhan. Pseudorandomness and Average-Case Complexity via Uniform Reductions. In *Proceedings of the 17th Annual IEEE Conference on Computational Complexity*, pages 129–138. IEEE, 2002.

[Uma] 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.

[Vio1] E. Viola. The Complexity of Constructing Pseudorandom Generators from Hard Functions. *Comput. Complexity*, 13(3-4):147–188, 2004.

[Vio2] E. Viola. On Constructing Parallel Pseudorandom Generators from One-Way Functions. In *Proceedings of the Twentieth Annual Conference on Computational Complexity*, pages 183– 197. IEEE, 2005.

[Yao] A. C. Yao. Theory and Applications of Trapdoor Functions (Extended Abstract). In *23rd Annual Symposium on Foundations of Computer Science*, pages 80–91. IEEE, 1982.