On the Complexity of Non-Adaptively Increasing the Stretch of Pseudorandom Generators

Eric Miles* and Emanuele Viola**

Northeastern University
{enmiles,viola}@ccs.neu.edu

Abstract. We study the complexity of black-box constructions of linear-stretch pseudorandom generators starting from a 1-bit stretch oracle generator G. We show that there is no construction which makes non-adaptive queries to G and then just outputs bits of the answers. The result extends to constructions that both work in the non-uniform setting and are only black-box in the primitive G (not the proof of correctness), in the sense that any such construction implies NP/poly $\neq$ P/poly. We then argue that not much more can be obtained using our techniques: via a modification of an argument of Reingold, Trevisan, and Vadhan (TCC '04), we prove in the non-uniform setting that there is a construction which only treats the primitive G as black-box, has polynomial stretch, makes non-adaptive queries to the oracle G, and outputs an affine function (i.e., parity or its complement) of the oracle query answers.

1 Introduction

The notion of a pseudorandom generator is fundamental to the study of both cryptography and computational complexity. An efficient algorithm $G : \{0,1\}^n \to \{0,1\}^{n+s}$ is a (cryptographic) pseudorandom generator (PRG) if no efficient adversary can distinguish a random output from a uniformly random string, except with some small advantage. That is, for all efficient adversaries A, we have $|\Pr[A(G(U_n)) = 1] - \Pr[A(U_{n+s}) = 1]| < \epsilon$.

A key parameter for any PRG is its stretch s, the difference between the output and input lengths. Any PRG must have stretch $s \geq 1$ to even satisfy the definition, but in fact such a small amount of stretch is not useful for any cryptographic or derandomization applications of which we are aware. For these, one typically needs the stretch to be larger, e.g. linear ($s = \Omega(n)$). An important and well-known result is that the existence of a PRG with stretch $s = 1$ implies the existence of a PRG with stretch $s = \text{poly}(n)$ for any desired polynomial. We begin by briefly recalling the construction that is typically used to prove this result (due to Goldreich and Micali; see [4] Sect. 3.3.2 for a more thorough treatment).

For a generator $G : \{0,1\}^\ell \to \{0,1\}^{\ell+1}$ and a positive integer k, let $G^k(x)$ denote the $(\ell + 1)$-bit string resulting from k iterative applications of G, each

* Supported by NSF grant CCF-0845003.
** Supported by NSF grant CCF-0845003.

time using the first ℓ bits of the previous output as input, and using x as input for the first invocation. Then, the "stretch-increasing" construction $H^{(\cdot)} : \{0,1\}^\ell \to \{0,1\}^m$ is defined as

$$H^G(x) := G^1(x)_{\ell+1} \circ G^2(x)_{\ell+1} \circ \cdots \circ G^m(x)_{\ell+1}. \tag{1}$$

That is, H iteratively queries G as described above, and outputs the final bit of each answer.

An aspect of the Goldreich-Micali construction that we would like to stress is that the queries H makes to its oracle are adaptive, in the sense that the ith query can be determined only after the answer to the $(i-1)$th query has been received. The presence of adaptivity in such constructions is of particular importance when considering the existence of cryptographic primitives in "low" complexity classes. The celebrated work of Applebaum et al. [1], in combination with the recent (non-adaptive) construction of Haitner et al. [7], demonstrates the existence of PRGs computable in NC^0 under the assumption that there exist one-way functions computable in NC^1. However, the resulting PRGs have *sub-linear* stretch, and the application of construction (1) would place them outside of NC^0.

Finally, we note that construction (1) only outputs bits of the answers it gets back from the queries, with no additional computation performed.

Informal discussion of our results. In this paper we study the complexity of increasing the stretch of cryptographic PRGs. We work in the setting of black-box constructions, which is explained in detail later. Informally, our first main result says that there can be no linear-stretch construction that makes non-adaptive queries and outputs only bits of its answers. For example, this rules out constructions which make non-adaptive queries and always output the last bit of the query answers, or even the entire query answers. Thus, linear-stretch constructions require either adaptive queries or postprocessing the queries in a more sophisticated way than just projecting. Our proof of this result is similar to one by Gennaro, Gertner, Katz, and Trevisan [3] who prove that constructions of generators with stretch s from one-way permutations must make $\geq \Omega(s/\log(\text{security}))$ queries. But note that our work is incomparable to theirs because we do not bound the number of queries (our constructions make one query per output bit).

Our second main result complements the first by showing that not much more can be obtained with the techniques in this paper. Specifically, we consider a special type of construction which is termed *weakly black-box* by Reingold, Trevisan, and Vadhan in [12], and for which we later advocate the alternative terminology *primitive black-box*. Then we extend an argument also in [12] to prove unconditionally, in the non-uniform setting, the existence of such constructions which have polynomial stretch, make non-adaptive queries, and just compute affine functions (parities or their complement) of the query answers. This complements the previous result because if instead of affine functions one only allows for projections, we show that such a construction implies NP/poly

$\neq$ P/poly. This means that, at least in the non-uniform setting, to extend our negative result to constructions with more postprocessing power requires different techniques from the ones in this paper. Our extension of the argument in [12] combines that argument with the Nisan-Wigderson generator [11].

Black-box constructions and formal statement of our results. To formally state our result, we first define black-box constructions. To explain and motivate the latter, we start by sketching the proof of correctness of the Goldreich-Micali Construction (1). Suppose there is an adversary A that distinguishes $H^G(U_\ell)$ from U_m with advantage greater than $\epsilon \cdot m$. Using a hybrid argument, one can show that there exists a $k \in [m]$ such that A distinguishes the distributions $U_{k-1} \circ \left(H^G(U_\ell)|_{[m-(k-1)]}\right)$ and $U_k \circ \left(H^G(U_\ell)|_{[m-k]}\right)$ with advantage greater than ϵ. Then, we define a probabilistic oracle circuit $C^{(\cdot)}$ as follows: on input $(x, b) \in \{0,1\}^\ell \times \{0,1\}$, $C^{G,A}$ computes $H^G(x)$ using its oracle to G, chooses $y \in \{0,1\}^{k-1}$ uniformly at random, and then outputs $A\left(y \circ b \circ H^G(x)|_{[m-k]}\right)$. Depending on whether (x, b) was chosen from $U_{\ell+1}$ or from $G(U_\ell)$, the input C gives to A will come from one of the two hybrid distributions that A can distinguish between, and so C distinguishes G with advantage greater than ϵ, contradicting G's pseudorandomness.

The above argument is an example of a *black-box reduction*: the argument applies to any (possibly hard to compute) functions G and A, provided that we are given oracle access to them.

Definition 1 (Black-box stretch-increasing construction). *An oracle function $H^{(\cdot)} : \{0,1\}^n \to \{0,1\}^{n+s}$ is a black-box stretch-increasing construction with security reduction size t of a generator with stretch s and error ϵ from any one-bit-stretch oracle generator $G : \{0,1\}^\ell \to \{0,1\}^{\ell+1}$ with error δ if the following holds:*

For every 1-bit stretch generator $G : \{0,1\}^\ell \to \{0,1\}^{\ell+1}$ and every adversary A, if A distinguishes H^G with advantage ϵ, i.e.

$$\left|\Pr[A(H^G(U_n)) = 1] - \Pr[A(U_{n+s}) = 1]\right| \geq \epsilon$$

then there is an oracle circuit $C^{(\cdot)}$ of size t that, when given oracle access to both A and G, distinguishes G with advantage δ, i.e.

$$\left|\Pr[C^{A,G}(G(U_\ell)) = 1] - \Pr[C^{A,G}(U_{\ell+1}) = 1]\right| \geq \delta.$$

Poly-time computable stretch-increasing black-box constructions are useful in constructing efficient PRGs, because if we start with an oracle G that is a poly-time computable PRG with error $\delta(\ell) = 1/\ell^{\omega(1)}$ against circuits of size $s(\ell) = \ell^{\omega(1)}$, and we have $t, n = \text{poly}(\ell)$, then H^G is a poly-time computable PRG with error $\epsilon(n) = 1/n^{\omega(1)}$ against circuits of size $s'(n) = n^{\omega(1)}$. This can be easily seen by noticing that any circuit of size $\text{poly}(n)$ that distinguishes H^G with advantage $1/n^{O(1)}$ can be transformed into a circuit of size at most $\text{poly}(\ell)$ that distinguishes G with advantage $1/\ell^{O(1)}$, contradicting G's pseudorandomness.

We can now state our first main result.

Theorem 1 *For all sufficiently large ℓ and for $n \leq 2^{\sqrt{\ell}}$, there is no black-box construction $H^{(\cdot)} : \{0,1\}^n \to \{0,1\}^{n+s}$ of a generator with stretch $s \geq 5n/\log n$ and error $\epsilon \leq 1/4$ from any one-bit stretch generator $G : \{0,1\}^\ell \to \{0,1\}^{\ell+1}$ with error $\delta \geq 2^{-\sqrt{\ell}/30}$ and with security reduction size $t \leq 2^{\sqrt{\ell}/30}$ of the form*

$$H^G(x) := G(q_1(x))_{b_1(x)} \circ \cdots \circ G(q_{n+s}(x))_{b_{n+s}(x)}$$

where $q_i : \{0,1\}^n \to \{0,1\}^\ell$ specifies the i-th query and $b_i : \{0,1\}^n \to [\ell+1]$ specifies the bit of the i-th answer to output.

Note this holds even if the 1-bit stretch generator is hard against circuits (as opposed to uniform algorithms).

An interesting open problem is to understand whether a result like Theorem 1 holds for arbitrary polynomial-time postprocess of the query answers. We do not know how to solve this problem. However, as mentioned before we can show that the techniques in this paper are unlikely to prove a negative result, even if the postprocessing is just computing parities or their complements. For this, we consider a special type of construction, which is termed weakly black-box in [12]. Informally, a reduction is weakly black-box if the construction H treats the primitive G as a black-box, but outputs a generator in the real-world, i.e. the proof of correctness is arbitrary. We suggest the alternative terminology *primitive black-box*, to signify both that only the primitive (and not the adversary) is treated as a black-box, and that this is a "cruder" form of reduction.

We define next primitive constructions, working in the asymptotic setting because the following results are cleaner to state in that setting. We also note that our construction will hold for infinitely many input lengths (as opposed to sufficiently large input), and for conciseness we incorporate this into the definition.

Definition 2 (Primitive black-box stretch-increasing construction). *Let ℓ be a security parameter, and let $n = n(\ell)$ and $s = s(\ell)$ be functions of ℓ. A family of oracle functions $H^{(\cdot)} : \{0,1\}^n \to \{0,1\}^{n+s}$ is a primitive black-box stretch-increasing construction with stretch s from any family of one-bit-stretch generators $G : \{0,1\}^\ell \to \{0,1\}^{\ell+1}$ if the following holds for infinitely many input lengths ℓ:*

For every generator family $G : \{0,1\}^\ell \to \{0,1\}^{\ell+1}$, if there exists a constant c_0 and a circuit family A of size at most n^{c_0} which distinguishes H^G from uniform with advantage at least $1/n^{c_0}$, i.e.

$$\left|\Pr\left[A\left(H^G(U_n)\right) = 1\right] - \Pr\left[A\left(U_{n+s}\right) = 1\right]\right| \geq 1/n^{c_0}$$

then there exists a constant c_1 and a oracle circuit family $C^{(\cdot)}$ of size at most ℓ^{c_1} which distinguishes G from uniform with probability at least $1/\ell^{c_1}$, i.e.

$$\left|\Pr\left[C^G\left(G(U_\ell)\right) = 1\right] - \Pr\left[C^G\left(U_{\ell+1}\right) = 1\right]\right| \geq 1/\ell^{c_1}.$$

Our second main results proves the existence of a non-adaptive primitive black-box stretch-increasing construction of a slightly modified form which computes a polynomial-stretch PRG. The additional power that this construction has is the ability to compute parities or their complements of the query answers, rather than just projections. Recall from Definition 2 that primitive constructions only work for infinitely many input lengths.

Theorem 2 *Let $c > 1$ be any constant. Then, for $n = 17\ell^2$, there exists a primitive black-box stretch-increasing construction $H^{(\cdot)} : \{0,1\}^n \to \{0,1\}^{n^c}$ with stretch $s := n^c - n$ from any family of one bit stretch generators $G : \{0,1\}^\ell \to \{0,1\}^{\ell+1}$. In addition, $H^{(\cdot)}$ is computable by a* poly(n)*-sized circuit family, and has the form*

$$H^G(x) := \langle G(q_1(x)), r_1(x)\rangle \oplus t_1(x) \ \circ \ \cdots \ \circ \ \langle G(q_{n+s}(x)), r_{n+s}(x)\rangle \oplus t_{n+s}(x)$$

where $q_i : \{0,1\}^n \to \{0,1\}^\ell$ specifies the ith query, $r_i : \{0,1\}^n \to \{0,1\}^{\ell+1}$ specifies the parity function for the ith answer, and $t_i : \{0,1\}^n \to \{0,1\}$ specifies whether to flip the ith bit.

One weakness of the above theorem is that it works in the non-uniform setting. It is an open problem whether something like this can be proved in the uniform one.

To appreciate Theorem 2, we point out that the techniques used for our previous negative result (Theorem 1) "extend" to primitive constructions as well, in the sense that they can be used to show that any such construction implies NP/poly $\not\subseteq$ P/poly. Note that primitive black-box constructions cannot be ruled out without ruling out the existence of pseudorandom generators (if the latter exist, a construction can just ignore the oracle and output a pseudorandom generator).

Theorem 3 *Let $n = n(\ell) \leq 2^{\sqrt{\ell}}$ and $s = s(n) \geq 5n/\log n$. Let $H^{(\cdot)} : \{0,1\}^n \to \{0,1\}^{n+s}$ be a primitive black-box stretch-increasing construction with stretch s from any family of one-bit stretch generators $G : \{0,1\}^\ell \to \{0,1\}^{\ell+1}$. If H has the form*

$$H^G(x) := G(q_1(x))_{b_1(x)} \circ \cdots \circ G(q_{n+s}(x))_{b_{n+s}(x)}$$

and the q_i and b_i are computable by poly(n)*-sized circuits, then NP/poly $\not\subseteq$ P/poly.*

Note that the parameters in Theorem 2 are within the range of parameters considered by Theorem 3 – the only difference is the amount of postprocess.

1.1 More related work

The earlier work [13] (which was later extended by [10]) analyzes a type of pseudorandom generator construction that is very similar to ours. The constructions in [13] make non-adaptive queries to an oracle one-way function, and then apply an arbitrary unbounded-fan-in constant-depth circuit (AC^0) to the outputs;

[13] shows that such constructions cannot have linear stretch. At first glance this construction is incomparable to Theorem 1, because it starts from a weaker primitive (one-way function instead of one-bit stretch generator) but on the other hand allows for AC^0 postprocessing instead of just projections.

However, it was pointed out to us by Benny Applebaum that a strengthening of Theorem 1 follows from [13] when combined with the works [1] and [7]. Specifically, a version of Theorem 1 holds even if the construction H is allowed to apply an AC^0 circuit to the output of the one-bit stretch oracle PRG G (rather than just taking projections). We now elaborate on this improvement. (We also remark that at the moment this establishes a strengthened negative result only for constructions that start from a uniform hardness assumption, because Theorem 1.1 in [13] is only proved for those.)

Let $H^{(\cdot)} : \{0,1\}^n \to \{0,1\}^{n+s}$ be a a black-box construction of a PRG from a one-bit stretch PRG of the form $H^G(x) := C_x(G(q_1(x)), \ldots, G(q_{\mathrm{poly}(n)}(x)))$, where C_x is an AC^0 circuit generated arbitrarily from x and the functions q_i are arbitrary as before. Let $G^{(\cdot)}_{\mathrm{HRV}} : \{0,1\}^\ell \to \{0,1\}^{\ell+1}$ be the black-box construction of a PRG from a OWF given by [7] Theorem 6.1. This construction has the form $G^f_{\mathrm{HRV}}(x) := C'(x, f(x'_1), \ldots, f(x'_t))$ where C' is an NC^1 circuit and the x'_i are disjoint projections of the input x. Then, we can apply the compiler from [1] (cf. Remark 6.7 in that work) to obtain a black-box construction $G^{(\cdot)}_{\mathrm{AIK}} : \{0,1\}^\ell \to \{0,1\}^{\ell+1}$ of a PRG from a OWF of the form $G^f_{\mathrm{AIK}}(x) := C''(x, f(x'_1), \ldots, f(x'_t))$, where now C'' is an NC^0 circuit (and thus is also an AC^0 circuit). (For both G_{HRV} and G_{AIK} the seed length is $\ell = \mathrm{poly}(m)$, where m is the input length of the oracle OWF, though the compiler from [1] does increase the seed length.) Finally, by combining H and G_{AIK}, we obtain a black-box construction $H^{(\cdot)}_* : \{0,1\}^n \to \{0,1\}^{n+s}$ of a PRG from a OWF which has the form $H^f_*(x) := C'''_x(f(q_1(x)), \ldots, f(q_{\mathrm{poly}(n)}(x)))$ where C'''_x is an AC^0 circuit. This is a contradiction to Theorem 1.1 of [13] when the oracle $f : \{0,1\}^m \to \{0,1\}^k$ has $\log^{\omega(1)} m < k \leq m^{O(1)}$ and the stretch s is greater than $n \cdot \log^{O(1)} m/k = o(n)$.

Finally, we mention that in a concurrent work, Bronson, Juma, and Papakonstantinou [2] also study non-adaptive black-box PRG constructions and obtain results which are incomparable to ours.

1.2 Techniques

We now explain the ideas behind the proof of Theorem 1. For simplicity, we first explain our proof in the case in which the construction always outputs the same bit of the answers, say the first bit (i.e., $b_i(x) = 1$ for every i, in Theorem 1). We start by considering a non-explicit oracle pseudorandom generator $G : \{0,1\}^\ell \to \{0,1\}^{\ell+1}$ that is hard to break even for circuits that have access to G. Such oracles are obtained in an unpublished manuscript of Impagliazzo [9] and in a work by Zimand [15]. (They work in a slightly different setting, however, obtaining PRGs with high probability in the random oracle model, where we instead require an unconditional (but non-explicit) generator that is

secure against adversaries which can query it. For completeness we present a streamlined version of their arguments in Sect. 4, and for the moment continue with the description of the proof of Theorem 1.) By padding, we can modify our oracle to have the extra property that $G(x)_1 = x_1$ for every x. Now, the point is that the construction doesn't need to query the oracle, since each output bit $G(q_i(x))_{b_i(x)}$ can be replaced with $q_i(x)_1$. So we can consider an adversary A that breaks the construction H by simply checking, given a challenge $z \in \{0,1\}^m$, whether there exists an input to H that produces z. This breaks H as soon as the output length is $\geq |x| + 1$. Since H doesn't use G anymore, neither does the adversary A. Hence the ability to access A does not compromise the security of G, contradicting Definition 1.

To obtain the result for primitive constructions, we observe that A can be computed in NP/poly, and hence under the assumption that NP/poly = P/poly we obtain a distinguisher.

Moreover, this simplified argument says nothing about, for example, a construction which outputs the entirety of the answers received from the oracle, which a priori may seem to be a plausible candidate. To generalize our result to constructions that output different bits (i.e. not always the first one), we identify a set of indices $T \subseteq [\ell + 1]$ of size $\ell(1 - \Theta(1/\log \ell))$, such that for most input strings $x \in \{0,1\}^n$, most of the bits $b_i(x)$ chosen by H fall inside T. We exploit this fact by designing an oracle PRG G that reveals the first $|T|$ bits of its input on the set T; that is, $G(x)|_T = x_1 x_2 \cdots x_{|T|}$ for every input x. We then design an (inefficient) adversary A that distinguishes H^G from uniform by examining, for every $x \in \{0,1\}^n$, only the bits i such that $b_i(x) \in T$, and checking if each bit matches the corresponding bit from the query $q_i(x)$. This turns out to break H as soon as the the output length is $\geq |x| + \Omega(|x|/\log |x|)$ (we do not attempt to optimize this value and content ourselves with anything sublinear). On the other hand, A depends on G just because of the knowledge of the set T, which means that oracle access to A does not compromise the security of G, again contradicting 1.

We now explain the proof of Theorem 2. Here we follow closely an argument of Reingold, Trevisan, and Vadhan in [12]. We proceed by case analysis, depending on the existence or non-existence of one-way functions (OWFs). For our argument, we define OWFs as computable by a family of poly-size circuits and hard to invert by any family of poly-size circuits.

If OWFs exist, we use the result of Håstad et al. [8] that efficiently computable PRGs also exist; the construction then ignores its oracle and simply outputs the PRG, by letting $t_i(x)$ be the ith output bit of the PRG, and setting $r_i = 0$, for every i.

If OWFs do not exist, this means that the oracle cannot be computable by poly-size circuits (since it is assumed to be hard to invert). We can then use Goldreich-Levin [6] to transform the oracle into a Boolean function that is hard to compute by any family of poly-size circuits. Until now this is the argument in [12]. (Actually [12] is more involved because it works even in the uniform setting.) Our contribution is to apply at this point the Nisan-Wigderson

construction [11] to get a PRG. Since this construction is non-adaptive and has arbitrary polynomial stretch, and the hard function given by Goldreich-Levin is just the inner product of the oracle with a random vector, this has the desired form.

We note that, as is well-known, the proof of correctness of the Nisan-Wigderson construction requires non-uniformity, and this is what prevents this result to apply to the uniform setting.

Organization In Sect. 2 we prove Theorems 1 and 3, the two negative results. In Sect. 3 we prove Theorem 2, the complementing positive result. Finally, in Sect. 4 we construct the one-bit-stretch oracle generator used in Sect. 2.

2 Black-box stretch-increasing constructions

In this section we prove Theorems 1 and 3. We use the following definition of an oracle pseudorandom generator.

Definition 3 (Oracle pseudorandom generator). *Let $G : \{0,1\}^n \to \{0,1\}^{n+s}$ be a function. G is a (T, ϵ)-pseudorandom generator if $s \geq 1$ and for every oracle circuit C of size at most T, we have $\left|\Pr[C^G(G(U_n)) = 1] - [C^G(U_{n+s}) = 1]\right| < \epsilon$. The quantity on the left-hand side of the inequality is referred to as C's advantage in distinguishing G, and s is referred to as G's stretch.*

The key property we require of our one-bit stretch oracle G, stated in the next theorem, is that it reveals a large portion of its input, i.e. most of the output bits are simply copied from the input.

Theorem 4 *Let $\ell, d \in \mathbb{N}$ be sufficiently large with $d \leq \ell/2$. Then, for any subset $T \subseteq [\ell + 1]$ with $|T| = \ell - d$ and any oracle A, there exists a generator $G : \{0,1\}^\ell \to \{0,1\}^{\ell+1}$ such that*

1. G is a $(2^{d/30}, 2^{-d/30})$-PRG against adversaries with oracle access to A (and G).

2. For every input $x \in \{0,1\}^\ell$, $G(x)|_T = x_1 x_2 \cdots x_{\ell-d}$.

We defer the proof of this theorem to Sect. 4, and instead start by showing how it is used to prove the main theorem. First, we need a simple technical lemma showing that for any stretch-increasing construction of the specified form, we can find a large set of indices inside which most $b_i(x)$ fall for most choices of x.

Lemma 1. *Let $n, d, s, \ell \in \mathbb{N}$ with $d < \ell$. Let $\{b_i : \{0,1\}^n \to [\ell + 1]\}_{i \in [n+s]}$ be a collection of $n + s$ functions. Then, there exists a set $T \subseteq [\ell + 1]$ of size $\ell - d$ such that*

$$\Pr_x \left[|\{i : b_i(x) \in T\}| \geq (n+s) \cdot \left(1 - \frac{4(d+1)}{\ell+1}\right)\right] \geq \frac{3}{4}.$$

Proof. Let $S \subseteq [\ell+1]$ denote a random subset of size $d+1$. We have $\Pr_{x,i,S}[b_i(x) \in S] = (d+1)/(\ell+1)$, and so we can fix some S so that $\Pr_{x,i}[b_i(x) \in S] \leq (d+1)/(\ell+1)$. This can be restated as $\mathbb{E}_x\left[\Pr_i\left[b_i(x) \in S\right]\right] \leq (d+1)/(\ell+1)$, and so by Markov's inequality we have $\Pr_x[\Pr_i[b_i(x) \in S] \geq 4(d+1)/(\ell+1)] \leq 1/4$. Letting $T := [\ell+1] \setminus S$ completes the proof.

We now prove Theorem 1, restated for convenience.

Theorem 1 *For all sufficiently large ℓ and for $n \leq 2^{\sqrt{\ell}}$, there is no black-box construction $H^{(\cdot)} : \{0,1\}^n \to \{0,1\}^{n+s}$ of a generator with stretch $s \geq 5n/\log n$ and error $\epsilon \leq 1/4$ from any one-bit stretch generator $G : \{0,1\}^\ell \to \{0,1\}^{\ell+1}$ with error $\delta \geq 2^{-\sqrt{\ell}/30}$ and with security reduction size $t \leq 2^{\sqrt{\ell}/30}$ of the form*

$$H^G(x) := G(q_1(x))_{b_1(x)} \circ \cdots \circ G(q_{n+s}(x))_{b_{n+s}(x)}$$

where $q_i : \{0,1\}^n \to \{0,1\}^\ell$ specifies the i-th query and $b_i : \{0,1\}^n \to [\ell+1]$ specifies the bit of the i-th answer to output.

Proof. Let $H^{(\cdot)}$ be a construction of the specified form. Fix a parameter $d := \ell/\log n$. Fix $T \subseteq [\ell+1]$ to be the subset of size $\ell - d$ guaranteed by Lemma 1. For each $x \in \{0,1\}^n$, let I_x denote the set $\{i : b_i(x) \in T\} \subseteq [n+s]$. Using $s = 5n/\log n$, the chosen value for d, and the fact that $|I_x|$ is an integer, the bound from Lemma 1 can be restated as $\Pr_x\left[|I_x| \geq n+1\right] \geq 3/4$ for sufficiently large n and ℓ. In the remainder of the proof, we refer to x such that $|I_x| \geq n+1$ as *good*.

Let T^{-1} denote a transformation such that $T^{-1}(j) = k$ if j is the kth smallest element of T (this is simply to provide a mapping from G's output bits to the corresponding revealed input bits). The adversary $A : \{0,1\}^{n+s} \to \{0,1\}$ is defined as the function which accepts exactly the set

$$\{z : \exists x \in \{0,1\}^n \text{ such that } x \text{ is good and } \forall i \in I_x, z_i = q_i(x)_{T^{-1}(b_i(x))}\}.$$

Let $G : \{0,1\}^\ell \to \{0,1\}^{\ell+1}$ be the PRG guaranteed by Theorem 4 using these choices of T and A. We claim that A distinguishes $H^G(U_n)$ from U_{n+s} with advantage at least $1/4$. To see this, consider z which is a uniformly chosen output of H^G, i.e. $z = H^G(x)$ for $x \leftarrow U_n$. Because x is good with probability at least $3/4$, and because $H^G(x)_i = q_i(x)_{T^{-1}(b_i(x))}$ for all $i \in I_x$ by item 2 of Theorem 4, we have $\Pr[A(H^G(U_n)) = 1] \geq 3/4$. Conversely, for the case where A's input is chosen from U_{n+s}, we have the following calculation:

$$\begin{aligned}
\Pr_{z \leftarrow U_{n+s}}[A(z) = 1] &= \Pr_z\left[\exists x : \ x \text{ is good} \wedge \forall i \in I_x : z_i = q_i(x)_{T^{-1}(b_i(x))}\right] \\
&\leq \sum_{\substack{x \in \{0,1\}^n \\ x \text{ is good}}} \Pr_z\left[\forall i \in I_x : z_i = q_i(x)_{T^{-1}(b_i(x))}\right] \\
&\leq \sum_{\substack{x \in \{0,1\}^n \\ x \text{ is good}}} 2^{-(n+1)} \\
&\leq \frac{1}{2}.
\end{aligned}$$

(The second inequality follows from the fact that $|I_x| \geq n+1$ for x that are good.)

Finally, note that item 1 in Theorem 4 (along with the choice of d and the upper bound on n) implies that there is no oracle circuit C of size at most $2^{\sqrt{\ell}/30}$ such that $C^{A,G}$ distinguishes G with advantage at least $2^{-\sqrt{\ell}/30}$. Therefore, H does not meet the conditions of Definition 1 for the stated parameters.

Next, we show that this theorem can be extended to the primitive black-box setting.

Theorem 3 *Let $n = n(\ell) \leq 2^{\sqrt{\ell}}$ and $s = s(n) \geq 5n/\log n$. Let $H^{(\cdot)} : \{0,1\}^n \to \{0,1\}^{n+s}$ be a primitive black-box stretch-increasing construction with stretch s from any family of one-bit stretch generators $G : \{0,1\}^\ell \to \{0,1\}^{\ell+1}$. If H has the form*

$$H^G(x) := G(q_1(x))_{b_1(x)} \circ \cdots \circ G(q_{n+s}(x))_{b_{n+s}(x)}$$

and the q_i and b_i are computable by poly(n)*-sized circuits, then NP/poly $\not\subseteq$ P/poly.*

Proof. Let H be a primitive black-box stretch-increasing construction of the specified form. Let G and I_x be defined as in Theorem 1 (the oracle A against which G is secure is not relevant here). Because the q_i, b_i functions are computable by poly(n)-size circuits, there is a poly(n)-size circuit family which computes the string $H^G(x)|_{I_x}$ on input x, while making *no* oracle calls to G. As a result, we can define a non-deterministic poly(n)-size circuit family which distinguishes H^G from uniform with advantage $1/4$: on input $z \in \{0,1\}^{n+s}$, the circuit non-deterministically guesses $x \in \{0,1\}^n$, and accepts iff $|I_x| \geq n+1$ and $z|_{I_x} = H^G(x)|_{I_x}$. The proof that this is indeed a distinguisher for H^G is identical to the argument given for Theorem 1.

Now assume for contradiction that NP/poly = P/poly, i.e. that every non-deterministic circuit family can be simulated by a deterministic circuit family with only a polynomial increase in size. Then, there is a poly(n)-size deterministic circuit family which distinguishes H^G from uniform with noticeable advantage. By the definition of a primitive black-box construction, there must also be such a circuit family that distinguishes G, contradicting G's pseudorandomness.

3 A non-adaptive primitive black-box construction

In this section we prove Theorem 2, showing that there exists a non-adaptive primitive black-box reduction with slightly more post-processing power (namely the ability to compute inner products) that computes a polynomial-stretch PRG. We remind the reader that this construction produces a PRG on infinitely many input lengths. For the sake of brevity, we state two standard definitions that are used in this section and the next.

Definition 4 (Hard to invert). *Let $f : \{0,1\}^n \to \{0,1\}^m$ be a function. f is (T,ϵ)-hard to invert if for every oracle circuit C of size at most T, we have $\Pr[f(C^f(f(U_n))) = f(U_n)] < \epsilon$.*

Definition 5 (Hard to compute). *Let $f : \{0,1\}^n \to \{0,1\}$ be a Boolean function. f is (T, ϵ)-hard to compute if for every circuit C of size at most T, we have* $\Pr[C(U_n) = f(U_n)] < 1/2 + \epsilon$.

We now state the prior results that we will use, and prove a simple lemma. In what follows, we will sometimes make the assumption that "OWFs do not exist", which means that, for any family of functions $f : \{0,1\}^\ell \to \{0,1\}^{\mathrm{poly}(\ell)}$ that is $(p(\ell), 1/p(\ell))$-hard to invert for all polynomials p and sufficiently large ℓ, every $\mathrm{poly}(\ell)$-sized circuit family fails to compute f on infinitely many input lengths. This corresponds to one-way functions computable by circuits and hard to invert by circuits.

Theorem 5 ([8]). *Assume that there exists a family of functions $G : \{0,1\}^\ell \to \{0,1\}^{\ell+1}$ that is computable by a $\mathrm{poly}(\ell)$-size circuit family and is $(p(\ell), 1/p(\ell))$-hard to invert for all polynomials p and sufficiently large ℓ. Then, for any constant c, there exists a family of generators $H : \{0,1\}^n \to \{0,1\}^{n^c}$ that is computable by a $\mathrm{poly}(n)$-size circuit family and is $(p(n), 1/p(n))$-pseudorandom for all polynomials p and sufficiently large n.*

A version of the following result was proved in [12] for the uniform computation model. This proof, which relies on non-uniformity, is a bit simpler.

Lemma 2. *Assume that OWFs do not exist and let $G : \{0,1\}^\ell \to \{0,1\}^{\ell+1}$ be a generator family. If G is $(p(\ell), 1/p(\ell))$-pseudorandom for all polynomials p and sufficiently large ℓ, then the Boolean function family $f(x, r) := \langle G(x), r\rangle$ is $(p(\ell), 1/p(\ell))$-hard to compute for all polynomials p and infinitely many input lengths.*

Proof. First, we show that if G is $(p(\ell), 1/p(\ell))$-pseudorandom for all polynomials p and sufficiently large ℓ, then it is also $(p(\ell), 1/p(\ell))$-hard to invert for all polynomials p and sufficiently large ℓ. Let C be a $\mathrm{poly}(\ell)$-size circuit family which, for sufficiently large ℓ, inverts G with probability $= \epsilon$ for some $\epsilon = 1/\mathrm{poly}(\ell)$. Then, define an adversary $A : \{0,1\}^{\ell+1} \to \{0,1\}$ as follows: on input y, A computes $x = C(y)$, uses its oracle to G to check if $G(x) = y$, and outputs 1 iff this holds. We clearly have $\Pr[A(G(U_\ell)) = 1] = \epsilon$. Let $T \subseteq \mathrm{Im}(G)$ be the set of outputs that C inverts, and note that $\sum_{y \in T} \Pr[G(U_\ell) = y] = \epsilon$. For each $y \in T$ we have $\Pr[G(U_\ell) = y] \geq 1/2^\ell$, and so $|T|/2^\ell \leq \epsilon$. Then, since A will only output 1 on inputs that C can invert and since no string outside $\mathrm{Im}(G)$ can be inverted, we have $\Pr[A(U_{\ell+1}) = 1] = |T|/2^{\ell+1} \leq \epsilon/2$, and thus A distinguishes G from uniform with advantage $\geq \epsilon/2 = 1/\mathrm{poly}(\ell)$.

Now, assume for contradiction that there exists a polynomial p and a circuit family C of size $p(\ell)$ which computes f correctly with probability at least $1/2 + 1/p(\ell)$ over the input, for sufficiently large ℓ. Then by the Goldreich-Levin theorem [6], there exists a polynomial p' and a circuit family C' of size $p'(\ell)$ such that $\Pr[C'(U_\ell) = G(U_\ell)] \geq 1/p'(\ell)$, for sufficiently large ℓ. Notice that (the function computed by) C' can only be inverted on strictly less than a $1 - 1/(2p'(\ell))$ fraction of inputs by $\mathrm{poly}(\ell)$-size circuits, because any circuit which inverts C' on

a $1-1/(2p'(\ell))$ fraction of inputs also inverts G on at least a $1/(2p'(\ell))$ fraction of inputs. However, using the standard direct product construction (originally due to Yao [14]; see also [4] Theorem 2.3.2), this implies the existence of a one-way function, contradicting the assumption that OWFs do not exist.

By virtue of the above proof, Theorem 2 actually establishes a primitive black-box stretch-increasing construction which works when the oracle is any hard-to-invert function, and not only the special case of one-bit-stretch PRGs.

In order to apply the Nisan-Wigderson construction, we recall the notion of designs.

Definition 6 (Design). *A collection of sets $S_1, \ldots, S_d \subseteq [n]$ is an (n, d, ℓ, α)-design if*

1. $\forall i : \ |S_i| = \ell$.
2. $\forall i \neq j : \ |S_i \cap S_j| \leq \alpha$.

Lemma 3 ([11]). *For any integers d and ℓ such that $\log d \leq \ell \leq d$, there exists a $\mathrm{poly}(d)$-time constructible collection $S_1, \ldots, S_d$ which is a $(4\ell^2, d, \ell, \log d)$-design.*

We now give the proof of Theorem 2.

Theorem 2 *Let $c > 1$ be any constant. Then, for $n = 17\ell^2$, there exists a primitive black-box stretch-increasing construction $H^{(\cdot)} : \{0,1\}^n \to \{0,1\}^{n^c}$ with stretch $s := n^c - n$ from any family of one bit stretch generators $G : \{0,1\}^\ell \to \{0,1\}^{\ell+1}$. In addition, $H^{(\cdot)}$ is computable by a $\mathrm{poly}(n)$-sized circuit family, and has the form*

$$H^G(x) := \langle G(q_1(x)), r_1(x)\rangle \oplus t_1(x) \ \circ \ \cdots \ \circ \ \langle G(q_{n+s}(x)), r_{n+s}(x)\rangle \oplus t_{n+s}(x)$$

where $q_i : \{0,1\}^n \to \{0,1\}^\ell$ specifies the ith query, $r_i : \{0,1\}^n \to \{0,1\}^{\ell+1}$ specifies the parity function for the ith answer, and $t_i : \{0,1\}^n \to \{0,1\}$ specifies whether to flip the ith bit.

Proof. Assume that OWFs exist, and let $H' : \{0,1\}^n \to \{0,1\}^{n^c}$ be the generator guaranteed by Theorem 5. Then, the construction $H^{(\cdot)}$ is $H^G(z) := H'(z)$ for any oracle G. (Note that this can be achieved in the form stated in the theorem by setting $r_i(z) = 0^{\ell+1}$ for all i and z, and choosing the t_i appropriately to compute each bit of H'.)

Now assume that OWFs do not exist. Let $G : \{0,1\}^\ell \to \{0,1\}^{\ell+1}$ be any generator family, and define $f : \{0,1\}^{2\ell+1} \to \{0,1\}$ as $f(x,r) := \langle G(x), r\rangle$. Fix a constant $c > 1$, and define $n = 4(2\ell+1)^2$ (which is at most $17\ell^2$ for sufficiently large ℓ). Let $S_1, \ldots, S_{n^c}$ be the $(n, n^c, 2\ell+1, c\log n)$ design guaranteed by Lemma 3. Then, the construction $H^G : \{0,1\}^n \to \{0,1\}^{n^c}$ is defined as

$$H^G(z) := f(z|_{S_1}) \circ \cdots \circ f(z|_{S_{n^c}}).$$

If there exists a polynomial p and a circuit family of size $p(\ell)$ which distinguishes G from uniform with advantage at least $1/p(\ell)$, then the theorem is

trivially true. Thus, we can take G to be $(p(\ell), 1/p(\ell))$-pseudorandom for all polynomials p and sufficiently large ℓ. Assume for contradiction that there exists a constant c_0 and a circuit family A of size n^{c_0} that distinguishes $H^G(U_n)$ from U_{n^c} with advantage $1/n^{c_0}$. Using the equivalence of distinguishing and next-bit predicting [14], this implies the existence of an $i \in [n^c]$ and a circuit family $A' : \{0,1\}^{i-1} \to \{0,1\}$ of size $n^{O(c_0)}$ such that $\Pr\left[A'(H^G(U_n)|_{[i-1]}) = H^G(U_n)_i\right] \geq 1/2 + 1/n^{c+c_0}$. Separating out the part of the input indexed by S_i, this can be rewritten as

$$\Pr_{(x,y)\leftarrow(U_{2\ell+1},U_n)}\left[A'(H^G(z)|_{[i-1]}) = H^G(z)_i\right] \geq 1/2 + 1/n^{c+c_0}, \tag{2}$$

where $z \in \{0,1\}^n$ is defined by $z|_{S_i} = x$ and $z|_{\overline{S_i}} = y|_{\overline{S_i}}$. By an averaging argument, there is a way to fix $y \in \{0,1\}^n$ such that (2) holds; from here on we assume that this y is fixed. For each $j \in [i-1]$, define the function $f_j : \{0,1\}^{2\ell+1} \to \{0,1\}$ as $f_j(x) := f(z)$, where now z is defined by $z|_{S_i \cap S_j} = x_1 x_2 \cdots x_{|S_i \cap S_j|}$ and $z|_{\overline{S_i \cap S_j}} = y|_{\overline{S_i \cap S_j}}$. Note that since $S_i \cap S_j \leq c \log n$ and y is fixed, each f_j is computable by a circuit family of size $\mathrm{poly}(n) = \mathrm{poly}(\ell)$. Finally, define the circuit family $A'' : \{0,1\}^{2\ell+1} \to \{0,1\}$ as $A''(x) := A'(f_1(x), \ldots, f_{i-1}(x))$. It can be easily checked that A'' has size $\mathrm{poly}(\ell)$ and correctly computes f on a random input with probability at least $1/2 + 1/n^{c+c_0}$, contradicting Lemma 2.

4 Constructing the oracle generator

In this section we prove Theorem 4 (restated for convenience), which gives the one-bit-stretch oracle generator used in the proofs of our negative results (Theorems 1 and 3).

Theorem 4 *Let $\ell, d \in \mathbb{N}$ be sufficiently large with $d \leq \ell/2$. Then, for any subset $T \subseteq [\ell+1]$ with $|T| = \ell - d$ and any oracle A, there exists a generator $G : \{0,1\}^\ell \to \{0,1\}^{\ell+1}$ such that*

1. G is a $(2^{d/30}, 2^{-d/30})$-PRG against adversaries with oracle access to A (and G).

2. For every input $x \in \{0,1\}^\ell$, $G(x)|_T = x_1 x_2 \cdots x_{\ell-d}$.

On constructing the oracle. A direct proof that a random function $G : \{0,1\}^\ell \to \{0,1\}^{\ell+1}$ is a pseudorandom generator even for circuits that have oracle access to G does not seem immediate to us. The existence of such oracles is shown via an indirect route in an unpublished manuscript of Impagliazzo [9] and – in a slightly different scenario – in a work by Zimand [15]. Both works proceed by considering an oracle one-way function, and then applying standard constructions of generators from one-way functions (for which one can now use [8] or [7]).

We proceed by first considering a hard-to-invert oracle permutation π, and then using the Goldreich-Levin hardcore bit [6] to get one bit of stretch. This

approach will have security exponential in the input length of π, and so we can apply π to the relatively few ($\Theta(\ell/\log \ell)$) bits outside of $|T|$, and then use padding to get a generator G on ℓ bits that reveals most of its input

We know of two ways to demonstrate the existence of such a permutation π. One is via a theorem in [3] which uses a clever encoding argument to prove that a random permutation is hard to invert with very high probability. They show that if there exists a small circuit which inverts a permutation π on some fraction of inputs, then π can be succinctly encoded when the circuit is given as advice. Then, since only a small number of permutations have succinct encodings, the probability that a random π can be sufficiently inverted by a fixed circuit is small, and a union bound over circuits gives the result.

The second way, and the one that we use here, is an arguably more direct argument showing that any fixed circuit with access to a fixed auxiliary oracle has negligible probability (over the choice of permutation) of sufficiently inverting the permutation. This method is from [9] and [15] (though they consider general length-preserving functions rather than permutations), and hinges on a combinatorial trick which originally appeared in [5]. Briefly, it is shown that for a fixed circuit C, the expected number of subsets of size k that are inverted by C is not too large. Then, Markov's inequality is used to show that the probability that C inverts *any* set of size $m \approx k^2$ is small, since to do so C would have to invert *each* of its $\binom{m}{k}$ subsets of size k (this is the combinatorial trick).

We now turn to the formal proof of Theorem 4. There are two main ingredients; the first is the well-known Goldreich-Levin hard-core bit theorem [6]. It can be checked that the standard proof of this theorem relativizes; we omit the details.

Theorem 6. *Let $f : \{0,1\}^d \to \{0,1\}^m$ be a function, and let A be any oracle. Let C be an oracle circuit of size T such that $\Pr[C^A(f(U_d), U'_d) = \langle U_d, U'_d \rangle] \geq 1/2 + \epsilon$. Then, for d sufficiently large, there exists an oracle circuit B of size at most $\alpha \cdot T \cdot (d/\epsilon)^2$ (where α is a universal constant) such that $\Pr[B^A(f(U_d)) = U_d] \geq \epsilon^3/8d$.*

The second ingredient is the fact that there exist permutations π which are hard to invert even for adversaries that have access to π and to an arbitrary fixed auxiliary oracle.

Theorem 7. *Let $d \in \mathbb{N}$ be sufficiently large. Then for any oracle A, there exists a permutation $\pi : \{0,1\}^d \to \{0,1\}^d$ that is $(2^{d/5}, 2^{-d/5})$-hard to invert against adversaries with oracle access to π and A.*

Before giving the proof, we state and prove two lemmas. The aforementioned combinatorial trick, due to [5], is given by the following lemma.

Lemma 4. *Let U be a finite set, let $\Gamma = \{\phi : U \to \{0,1\}\}$ be a family of predicates on U, and let p_k be an upper bound on the probability that ϕ chosen uniformly from Γ returns true for every element in a subset of size k, i.e.*

$$\forall K \subseteq U, |K| = k : \Pr_{\phi \leftarrow \Gamma}\left[\prod_{x \in K} \phi(x) = 1\right] \leq p_k.$$

Then, for any m such that $k \leq m \leq |U|$, we have

$$\Pr_{\phi \leftarrow \Gamma}\left[\exists M \subseteq U, |M| \geq m : \prod_{x \in M} \phi(x) = 1\right] \leq \frac{\binom{|U|}{k} \cdot p_k}{\binom{m}{k}}.$$

Proof. Let $\phi(X)$ denote $\prod_{x \in X} \phi(x)$. We have $\mathbb{E}\left[|\{K \subseteq U : |K| = k \text{ and } \phi(K) = 1\}|\right] \leq \binom{|U|}{k} \cdot p_k$ by linearity of expectation. Then the lemma follows from double counting, because for any set $M \subseteq U$ of size m, $\phi(M) = 1$ iff $\phi(K) = 1$ for every one of the $\binom{m}{k}$ subsets $K \subseteq M$ of size k.

We now explain why this lemma is helpful. Following [9] and [15], we bound the probability (over the permutation π) that a fixed circuit C of size s inverts a fixed set K of size k; this is done by considering the probability that any k out of the at most ks distinct queries made by C on inputs from K are mapped by π to K; specifically, we bound

$$p_k \leq \binom{ks}{k} \cdot \left(\frac{k}{|U|}\right)^k \approx \frac{s^k}{\binom{|U|}{k}}.$$

The factor of s^k means that we cannot use a union bound over all $\binom{|U|}{k}$ subsets of size k. So we instead use Lemma 4, choosing m so that $\binom{m}{k} \approx s^{2.3k}$, which makes the probability of inverting a set of size m small enough to use a union bound over all circuits.

We also require a bound on the number of oracle circuits of a given size.

Lemma 5. *There are at most $2^{s(3+4\log s)}$ oracle circuits of size s which have access to two oracles π and A.*

Proof. We define the size of a circuit to be the number of wires it has; this is also an upper bound on the number of gates. For each wire in the circuit, we must specify two things:

- which gate it is an output of (or if it is an input wire) and which position it is in for this gate
- which gate it is an input of (or if it is an output wire) and which position it is in for this gate

Note that the positions are relevant for wires incident on oracle gates, as the functions computed by these gates may not be symmetric. Specifying either incident gate for a given wire takes $\log s$ bits (as there are at most s gates), and likewise each position can be specified with $\log s$ bits. Therefore, each of the s wires can be specified with $4 \log s$ bits. Finally, for each gate, we must specify which of the five types it is ($\wedge, \vee, \neg, \pi$-oracle or A-oracle), which takes three bits.

Proof (Proof of Theorem 7). We will in fact show that a random π has the desired property with probability at least $1 - 2^{-2^{d/4}}$. Fix an oracle A and an oracle circuit C of size s. Fix a subset $K \subseteq \{0,1\}^d$ of size k; we will first bound

the probability that C inverts all of K. Let Q_x^π denote the set of at most s distinct queries that $C^{A,\pi}(x)$ makes to π (for some choice of x and π), and let $Q_K^\pi := \bigcup_{x\in K} Q_x^\pi$. We assume without loss of generality that the last query that C makes to π is the string that C outputs (this is justified because any circuit which does not query its output string can be modified into one that does with an increase in size that is so small as to not affect the union bound below).

A necessary condition for C to invert all of K is that $\pi^{-1}(x) \in Q_K^\pi$ for all $x \in K$. Since $|Q_K^\pi| \le ks$, we can bound this by

$$\begin{aligned}\Pr_\pi\left[\forall x \in K : \pi^{-1}(x) \in Q_K^\pi\right] &\le \Pr_\pi\left[\exists X \subseteq Q_K^\pi : \bigcup_{x\in X} \pi(x) = K\right] \\ &\le \binom{ks}{k} \cdot \left(\frac{k}{2^d}\right)\left(\frac{k-1}{2^d-1}\right)\cdots\left(\frac{1}{2^d-k+1}\right) \\ &\le \left(\frac{eks}{2^d}\right)^k.\end{aligned}$$

We now apply Lemma 4 in the obvious way: U is $\{0,1\}^d$, and there is a predicate $\phi_\pi \in \Gamma$ for each permutation π, where $\phi_\pi(x) = 1$ iff $C^{A,\pi}(x) = \pi^{-1}(x)$. By the lemma, the probability that there exists a set M of size $m \ge k$ such that C inverts every element of M is bounded from above by $(e^2 \cdot k \cdot s/m)^k$. Choosing $k = 2^{d/3}, m = 2^{4d/5}$ and $s = 2^{d/5}$, this is bounded by $2^{-2^{d/3}}$ for sufficiently large d. By Lemma 5, there are at most $2^{2^{d/5}\cdot\Theta(d)}$ circuits of size $2^{d/5}$, and so the probability over the choice of π that there *exists* a circuit of size $2^{d/5}$ which inverts a set of size at least $2^{4d/5}$ is at most $2^{-2^{d/3}+2^{d/5}\cdot\Theta(d)} < 2^{-2^{d/4}}$ for sufficiently large d. Therefore, π is $(2^{d/5}, 2^{-d/5})$-hard to invert with probability at least $1 - 2^{-2^{d/4}}$.

We may now give the proof of Theorem 4.

Proof (Proof of Theorem 4). Let the oracle A and the subset T be given. Recall that $|T| = \ell - d$, and let $\pi : \{0,1\}^d \to \{0,1\}^d$ be the permutation guaranteed by Theorem 7 which is $(2^{d/5}, 2^{-d/5})$-hard to invert against adversaries with oracle access to π and A. Then, the generator G treats its input $x \in \{0,1\}^\ell$ as $(x_1, x_2, x_3) \in \{0,1\}^{\ell-2d} \times \{0,1\}^d \times \{0,1\}^d$, and outputs the $(\ell+1)$-bit string defined as follows:

$$G(x)|_{[\ell+1]\setminus T} = \pi(x_3) \circ \langle x_3, x_2\rangle \qquad\qquad G(x)|_T = x_1 \circ x_2.$$

Now assume for contradiction that there exists an oracle circuit $C : \{0,1\}^{\ell+1} \to \{0,1\}$ of size at most $2^{d/30}$ such that $\Pr[C^{A,G}(G(U_\ell)) = 1] - \Pr[C^{A,G}(U_{\ell+1}) = 1] \ge 2^{-d/30}$ (dropping the absolute value w.l.o.g.). Because the permutation π is the only part of G's output which may be "difficult" to compute, we can take C to have oracles (A, π) instead of (A, G) at the cost of increasing C's size by a factor of poly(d). We construct a probabilistic oracle circuit $IP : \{0,1\}^d \times \{0,1\}^d \to \{0,1\}$ which, on input (x, y), tries to compute $\langle \pi^{-1}(x), y\rangle$. $IP^{A,\pi}(x,y)$ performs the following steps:

1. chooses a random string $z \in \{0,1\}^{\ell-2d}$ and a random bit $b \in \{0,1\}$
2. constructs the $(\ell+1)$-bit string w defined by $w|_{[\ell+1]\setminus T} = x \circ b,\ w|_T = z \circ y$
3. computes $C^{A,\pi}(w)$ and outputs $C^{A,\pi}(w) \oplus 1 \oplus b$

We clearly have $|IP| \leq |C| \cdot \mathrm{poly}(d) \leq 2^{d/30} \cdot \mathrm{poly}(d)$. Consider the behavior of $IP^{A,\pi}$ on a uniformly random input (x,y). It is easy to see that the string w is distributed according to $U_{\ell+1}$. If we condition on the chosen bit b being equal to $\langle \pi^{-1}(x), y\rangle$ (which happens with probability $1/2$), then w is distributed according to $G(U_\ell)$. For brevity, let E_{IP} denote the event $IP^{A,\pi}(x,y) = \langle \pi^{-1}(x), y\rangle$, and let E_b denote the event $b = \langle \pi^{-1}(x), y\rangle$. Then,

$$\begin{aligned}
\Pr[E_{IP}] &= \frac{1}{2}\left(\Pr[E_{IP} \mid E_b] + \Pr[E_{IP} \mid \overline{E_b}]\right) \\
&= \frac{1}{2}\left(\Pr[C^{A,\pi}(w) = 1 \mid E_b] + \left(1 - \Pr[C^{A,\pi}(w) = 1 \mid \overline{E_b}]\right)\right) \\
&= 1/2 + \Pr[C^{A,\pi}(w) = 1 \mid E_b] - \Pr[C^{A,\pi}(w) = 1] \\
&= 1/2 + \Pr[C^{A,\pi}(G(U_\ell)) = 1] - \Pr[C^{A,\pi}(U_{\ell+1}) = 1] \\
&\geq 1/2 + 2^{-d/30}.
\end{aligned}$$

The probabilities are over both (x,y) and the internal randomness of IP; by a standard averaging argument, we can fix the internal randomness of IP to get a deterministic circuit which computes $\langle \pi^{-1}(x), y\rangle$ on a random (x,y) with the same success probability. Then for sufficiently large d, Theorem 6 gives an oracle circuit of size at most $2^{d/30} \cdot \mathrm{poly}(d) \cdot O(d^2 \cdot 2^{2d/30}) \leq 2^{d/5}$ that, when given access to A and π, inverts π with probability at least $2^{-3d/30}/8d \geq 2^{-d/5}$ over its input, contradicting the hardness of π.

Acknowledgements. We are very grateful to Benny Applebaum for several useful comments, and especially for pointing out the strengthening of Theorem 1 and allowing us to include a proof in §1.1. We also would like to thank Russell Impagliazzo for sharing [9] with us, and the anonymous TCC referees for helpful feedback.

References

1. Benny Applebaum, Yuval Ishai, and Eyal Kushilevitz. Cryptography in NC^0. *SIAM J. Comput.*, 36(4):845–888, 2006.
2. Josh Bronson, Ali Juma, and Periklis A. Papakonstantinou. Limits on the stretch of non-adaptive constructions of pseudo-random generators. In *8th Theory of Cryptography Conference (TCC)*, 2011.
3. Rosario Gennaro, Yael Gertner, Jonathan Katz, and Luca Trevisan. Bounds on the efficiency of generic cryptographic constructions. *SIAM J. Comput.*, 35(1):217–246, 2005.
4. Oded Goldreich. *Foundations of Cryptography: Volume 1, Basic Tools*. Cambridge University Press, 2001.
5. Oded Goldreich, Hugo Krawczyk, and Michael Luby. On the existence of pseudo-random generators. *SIAM J. Comput.*, 22(6):1163–1175, 1993.

6. Oded Goldreich and Leonid Levin. A hard-core predicate for all one-way functions. In *21st Annual ACM Symposium on Theory of Computing (STOC)*, pages 25–32, 1989.
7. Iftach Haitner, Omer Reingold, and Salil P. Vadhan. Efficiency improvements in constructing pseudorandom generators from one-way functions. In *42nd ACM Symposium on Theory of Computing (STOC)*, pages 437–446, 2010.
8. Johan Håstad, Russell Impagliazzo, Leonid A. Levin, and Michael Luby. A pseudorandom generator from any one-way function. *SIAM J. Comput.*, 28(4):1364–1396 (electronic), 1999.
9. Russell Impagliazzo. Very strong one-way functions and pseudo-random generators exist relative to a random oracle. Manuscript, 1996.
10. Chi-Jen Lu. On the complexity of parallel hardness amplification for one-way functions. In *3rd Theory of Cryptography Conference (TCC)*, pages 462–481, 2006.
11. Noam Nisan and Avi Wigderson. Hardness vs randomness. *J. Computer & Systems Sciences*, 49(2):149–167, 1994.
12. Omer Reingold, Luca Trevisan, and Salil Vadhan. Notions of reducibility between cryptographic primitives. In *Proceedings of the 1st Theory of Cryptography Conference (Feb 19-21, 2004: Cambridge, MA, USA)*. Springer-Verlag, 2004.
13. Emanuele Viola. On constructing parallel pseudorandom generators from one-way functions. In *20th Annual Conference on Computational Complexity (CCC)*, pages 183–197. IEEE, 2005.
14. Andrew Yao. Theory and applications of trapdoor functions. In *23rd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 80–91. IEEE, 1982.
15. Marius Zimand. Efficient privatization of random bits. In *"Randomized Algorithms" satellite workshop of the 23rd International Symposium on Mathematical Foundations of Computer Science*, 1998. Available at http://triton.towson.edu/∼mzimand/pub/rand-privat.ps.