Structural Lower Bounds on Black-Box Constructions of Pseudorandom Functions

Amos Beimel[1], Tal Malkin[2], and Noam Mazor[3(✉)]

[1] Ben Gurion University, Be'er Sheva, Israel
amos.beimel@gmail.com
[2] Columbia University, New York, USA
tal@cs.columbia.edu
[3] Tel Aviv University, Tel Aviv, Israel
noammaz@gmail.com

Abstract. We address the black-box complexity of constructing pseudorandom functions (PRF) from pseudorandom generators (PRG). The celebrated GGM construction of Goldreich, Goldwasser, and Micali (Crypto 1984) provides such a construction, which (even when combined with Levin's domain-extension trick) has super-logarithmic depth. Despite many years and much effort, this remains essentially the best construction we have to date. On the negative side, one step is provided by the work of Miles and Viola (TCC 2011), which shows that a black-box construction which just calls the PRG once and outputs one of its output bits, cannot be a PRF.

In this work, we make significant further progress: we rule out black-box constructions of PRF from PRG that follow certain structural constraints, but may call the PRG adaptively polynomially many times. In particular, we define "tree constructions" which generalize the GGM structure: they apply the PRG G along a tree path, but allow for different choices of functions to compute the children of a node on the tree and to compute the next node on the computation path down the tree. We prove that a tree construction of logarithmic depth cannot be a PRF (while GGM is a tree construction of super-logarithmic depth). We also show several other results and discuss the special case of one-call constructions.

Our main results in fact rule out even *weak* PRF constructions with *one* output bit. We use the oracle separation methodology introduced by Gertner, Malkin, and Reingold (FOCS 2001), and show that for any candidate black-box construction F^G from G, there exists an oracle relative to which G is a PRG, but F^G is not a PRF.

A. Beimel—Part of this work was done while visiting the Simons Institute. Research partly supported by ISF grant 391/21.
T. Malkin—Part of this work was done while visiting the Simons Institute. Research supported by NSF CCF-2312242 and an Amazon Research Award.
N. Mazor—Part of this work was done while at Cornell Tech and while visiting the Simons Institute. Research partly supported by NSF CNS-2149305.

L. Reyzin and D. Stebila (Eds.): CRYPTO 2024, LNCS 14924, pp. 459–488, 2024.
https://doi.org/10.1007/978-3-031-68388-6_16

1 Introduction

Pseudorandom Functions (PRF) constitute one of the most important primitives in cryptography, used in almost every application of cryptography in theory and in practice, and with deep connections to complexity theory and learning theory. Classic results in cryptography prove that the existence of PRFs is equivalent to the existence of many other fundamental primitives such as one-way functions (OWF), pseudorandom generators (PRG), signatures, private key encryption, and many others, where equivalence is defined by the existence of a polynomial time reduction. However, these primitives are not all created equal, as the reductions often incur significant efficiency cost, for various notions of efficiency. For example, given a PRF it is easy to construct a PRG with similar parallel-time complexity, but the other direction is not known. There is also a wide gap between the efficiency of theoretical constructions of PRF and the corresponding designs used in practice (block ciphers such as AES). An important and intensively studied goal is to minimize the complexity of PRF constructions from minimal assumptions. In this paper, we focus on the complexity of constructing PRFs from PRGs.

The seminal result of Goldreich, Goldwasser, and Micali [GGM86] (referred to as the GGM construction hereafter) showed how PRFs can be constructed from PRG in a black box way. In particular, given a PRG $G\colon \{0,1\}^n \to \{0,1\}^{2n}$ they construct a PRF $\mathcal{F} = \left\{ f_k^G \colon \{0,1\}^{m(n)} \to \{0,1\}^n \right\}_{k\in\{0,1\}^n}$ for any polynomial input length $m(n)$. To evaluate $f_k^G(x)$, the construction sequentially applies G adaptively $|x|$ times (once per each bit of x) along a tree path. This results in a construction of depth linear in the length of the input – highly non-parallel. This can be improved to $\omega(\log n)$ depth by using Levin's domain extension technique [Lev87], a generic transformation which applies a pairwise independent function to the input before running it through the construction. This allows to start from a polynomial length input, shorten it to a super-logarithmic length input, and then run the construction (in our case GGM) on the shorter input. In more detail, if $\mathcal{H} = \left\{ h\colon \{0,1\}^{m'} \to \{0,1\}^m \right\}$ is a family of pairwise independent functions, and $\mathcal{F} = \left\{ f_k^G \right\}$ is a PRF, it is not hard to see that the family $\mathcal{F}' = \left\{ f'^G_{k,h} \right\}$ where $f'^G_{k,h}(x) = f_k^G(h(x))$ is also a PRF.

In the decades since the GGM construction was introduced, much effort was dedicated to trying to improve it (e.g., [NR99, NRR00, NR04, LW09, BMR10, BPR12, AR16]), including some results achieving PRFs in NC^1 (logarithmic depth circuits) from concrete assumptions like DDH [NR99] and LWR [BPR12]. Despite this, the above construction remains the best one we have to date from PRG. In terms of lower bounds, it is known via connections to learning and the natural proofs barrier, that PRFs cannot be constructed in certain low circuit complexity classes such as $\mathrm{AC}^0[2]$ (c.f. [Val84, LMN93, PW88, RR94]), but there are no known lower bounds on the required depth (or parallel efficiency) of a PRF constructed from PRG.

Indeed, the following question remains open (and stated as an open problem already by Naor and Reingold [NR99]): *Is there a black-box construction of PRF from PRG with logarithmic depth?* We can start by asking a much more basic question: *Is there a black-box construction of PRF from PRG that calls the PRG just one time?* Miles and Viola [MV11] make a step towards addressing this question, by ruling out such one-call constructions that consist of a projection, namely call the PRG once and just output one of its output bits.[1] Beyond this, even this basic question remains open.

Our goal is to address this large gap between the known positive results (black-box constructions of super-logarithmic depth and number of calls to the PRG), and the known negative results (only a very partial impossibility of a black-box construction with a single call). We provide some explanation to this state of affairs, by giving black-box separations ruling out a large class of black-box constructions. In particular, for any candidate construction $\mathcal{F}^G = \{f_k^G\}$ in this class, we show an oracle $\mathcal{O}$ relative to which there is a PRG G, but $\mathcal{F}^G$ is not a PRF: there exists an efficient algorithm $\text{Break}^{\mathcal{O}}$ that can distinguish a randomly chosen function in $\mathcal{F}^G$ from a truly random function. This follows the oracle separation methodology of Gertner, Malkin, and Reingold [GMR01], and rules out so-called fully-black-box constructions [RTV04], but does not rule out all relativizing reductions, since the adversary Break we design is specific for the given candidate construction.

1.1 Our Results

We start with a high level overview, followed by more details. We consider (purported) black-box constructions of a PRF from a PRG, where the PRF has super-logarithmic input length and one-bit output. Since we are showing negative results, this implies the same results hold for PRF with many output bits. Our results hold even when the given PRG has super-polynomial stretch.

We rule out black-box constructions of a PRF $\mathcal{F} = \{f_k\}$ from a PRG G that satisfy some structural conditions. Our main results rule out constructions of the form

$$f_k^G(x) = A^{\mathcal{Q}_{k,x}}(k, x),$$

where the oracle $\mathcal{Q}_{k,x}$ implements a function that calls G and depends on the input x, k in some constrained way, while the algorithm A is an arbitrary oracle aided algorithm (it can call the oracle adaptively, any number of times, and on any input, without restriction). We have two main results with different constraints on $\mathcal{Q}$.

Our first main result generalizes a result by Miles and Viola [MV11]. In our terminology, they rule out constructions where $\mathcal{Q}_{k,x}(s)$ applies $G(s)$ and returns one of its output bits (which one depends on k, x), and where A can only call

[1] Miles and Viola [MV11] considered the task of stretching the output of PRGs, but as noted in [MV15] their lower bound implies a lower bound on PRF constructions. The results in [MV15] additionally rules out, in our terminology, PRF constructions with non-adaptive calls and AC0 post-processing.

$\mathcal{Q}_{k,x}$ *once* and return the same output bit (namely the PRF applies a projection on the output of the PRG). In contrast, we rule out any construction where $\mathcal{Q}_{k,x}(s)$ applies $G(s)$ and then returns a longer "digest" (depending also on k, x), and where A is an arbitrary oracle-aided algorithm that can call $\mathcal{Q}_{k,x}$ adaptively and apply arbitrary post-processing. We require that the digest function applied by the oracle in every call does not use too many bits of x, k (at most logarithmically many), and that its output is not too long (bounded away from the security parameter by more than a logarithmic amount). We show that these requirements are necessary (so our result is tight in this sense).

Our second main result allows $\mathcal{Q}_{k,x}$ to apply what we call a tree construction, which generalizes a logarithmic-depth GGM structure, applying G at each level, but allowing different choices for the functions computing the children of a node at each level of the tree, and for the function computing the next node on the computation path for a given input. Thus, we show that even if you can call such a log-depth tree oracle $\mathcal{Q}$ polynomially many times on different root values, the resulting construction is not a PRF. In this sense this result is tight with GGM, which can be viewed as a single call to such a tree oracle $\mathcal{Q}$ with super-logarithmic depth.

Considering the special case of constructions that call the PRG just one time, our first result immediately rules out any such construction that calls G on some arbitrary function of x, k, then applies some digest on the output of G and on logarithmically many bits of x, k, and then applies arbitrary post-processing using the digest (which is not too long) and the input x, k. Here our constraints do not seem tight; completely ruling out any construction that calls the PRG one time remains an intriguing open problem.

We note that all the results above in fact rule out even constructions of *weak* PRF, which are PRF that should be indistinguishable from random functions by adversaries who get *random* (input,output) pairs (rather than being able to ask queries).

Finally, we show that any black-box construction of PRF $\{f_k\}$ from a PRG G cannot have the key k be much shorter than the length of the inputs that it calls G on: if G is called on n-bit inputs, k must be of length at least $n - O(\log n)$. Note that the first step in GGM applies $G(k)$, so there k is exactly the length of the input to G.

We provide more details on each of our results below. In the following, for a security parameter $n \in \mathbb{N}$, let

$$\mathcal{F} = \left\{ f_k \colon \{0,1\}^{m(n)} \to \{0,1\} \right\}_{k \in \{0,1\}^{\lambda(n)}}$$

be a candidate PRF construction with domain $\{0,1\}^{m(n)}$ of super-polynomial size and with a key of polynomial length $\lambda(n)$. For simplicity of presentation, here we assume that all calls to G in the construction are always on inputs of length n (the security parameter). In the full version of this paper we show how to generalize the proof for constructions that can call G on different input lengths.

Warmup: Projection Functions. Miles and Viola [MV11] showed that there is no black-box construction of a PRF from PRG such that

$$f_k^G(x) = G(s(k,x))_{i(k,x)},$$

where $i(k,x) \in [|G(s(k,x))|]$ is some index of a bit in the output of G (we call this a projection function).[2] Our first theorem makes this result stronger by allowing to add arbitrary post-processing.

Let $G_{i(k,x)}$ be the function defined by $G_{i(k,x)}(s) = G(s)_{i(k,x)}$. Miles and Viola [MV11] showed that there is no PRF construction such that $f_k^G(x)$ makes one call to $G_{i(k,x)}$ and outputs the result. We show that there is no black-box PRF construction such that $f_k^G(x) = A^{G_{i(k,x)}}$, where A is an arbitrary oracle-aided algorithm. That is, here A is allowed to make arbitrary number of (adaptive) calls to $G_{i(k,x)}$, and to apply arbitrary functions on the outputs.

Theorem 1.1. *For a function $G\colon \{0,1\}^n \to \{0,1\}^{n+r(n)}$ and an index $i \in [n + r(n)]$, let $G_i(s) = G(s)_i$. Let $i\colon \{0,1\}^{\lambda(n)} \times \{0,1\}^{m(n)} \to [n+r(n)]$ be a function.*

Then for any polynomial r there is no black-box PRF construction $\mathcal{F} = \{f_k\}$ from $r(n)$-bit stretch PRG, such that

$$f_k^G(x) = A^{G_{i(k,x)}}(k,x)$$

for any algorithm A.

First Main Result: Digest Functions. Our result is actually even stronger, as we can replace the function $G_{i(k,x)}$ ("projection") with any function $\mathcal{Q}_{k,x}(s) = P(G(s), L(k,x))$, as long as $|L(k,x)| \in O(\log n)$ and $|P(G(s), L(k,x))| \leq n - \omega(\log n)$ (a "digest" function). Note that projection functions are a special case of digest functions, as $G_{i(k,x)}(s)$ can be written as $P(G(s), L(k,x))$ for $L(k,x) = i(k,x)$ and $P(z,i) = z_i$.

Theorem 1.2. *Let $G\colon \{0,1\}^n \to \{0,1\}^{n+r(n)}$. Let L and P be functions such that for every k, x and s, $|L(k,x)| \in O(\log n)$ and $|P(G(s), L(k,x))| \leq n - \omega(\log r)$. Let $\mathcal{Q}_{L(k,x)}(s) = P(G(s), L(k,x))$. Then there is no black-box PRF construction $\mathcal{F} = \{f_k\}$ from a PRG, such that*

$$f_k^G(x) = A^{\mathcal{Q}_{L(k,x)}}(k,x)$$

for any algorithm A.

Tightness. Note that if we allow the stretch of the PRG to be super-polynomial, there is a simple black-box projection construction that uses the output of the PRG as the truth-table of the PRF, namely $f_k^G(x) = G(k)_x$ for $x \in [|G(k)|]$. Thus, for both the original [MV11] and our generalization in Theorem 1.1, we

[2] Their paper, and its journal version [MV15], has other results as well, in particular about increasing PRG stretch in a black box way; we focus here on the result relevant to our paper.

need to require the index function to be have output of logarithmic length (which in turn implies the stretch must be polynomial). Our more general result in Theorem 1.2 does not impose any restriction on the stretch of the PRG, but still requires the output of the function L to be of logarithmic length. This restriction is necessary, to avoid the same simple truth-table construction.

The restriction on the output length of the digest P is also necessary for our structural lower bound. Observe that when the output of P is allowed to be $n+1$ bits, there exists a PRF construction $f_k^G(x) = A^{\mathcal{Q}}(k,x)$ where $\mathcal{Q}(s)$ simply returns $P(G(s)) = G(s)_{\leq n+1}$,[3] and A runs the standard black-box construction of a PRF from a PRG with 1-bit stretch (first using the oracle to get a length doubling PRG, and then running the GGM construction). In fact, using the Goldreich-Levin theorem, such a construction exists even if we set $P(G(s)) = G(s)_{\leq n-\log n}$, as we can add logarithmically many hard core bits.

Second Main Result: Tree Constructions. The GGM construction has the following structure. For every key k and input x, $f_k^G(x)$ is defined by

$$f_k^G(x) = S(G(\dots G(S(G(k), x_1))\dots), x_n)_1,$$

where $S(z,b) = z_b$ for $z = z_0||z_1$ with $|z_0| = |z_1|$.

This construction can be seen as a binary tree, where each node is labeled with an n bit string: the root is labeled by with the key k, and to compute the label of a child of a node v, we query the PRG on the label of v, and then apply some function $S(G(v),0)$ and $S(G(v),1)$ to get the labels of the two children. To compute the function, we start from the root, and use x to determine the path to a leaf we take on the tree. The output is just the label of the leaf.

We generalize the above structure and define tree constructions of PRFs.

Definition 1.3. *We say that a black-box construction of a PRF $\mathcal{F} = \{f_k\}$ from a PRG G is a (t,c)-*Tree construction*, if there exist functions $L_1,\dots,L_t$ and $S_0,\dots,S_t$ such that for every key k and input x, $|L_i(k,x)| \leq c$, and,*

$$f_k^G(x) = S_t(G(\dots S_2(G(S_1(G(S_0(k,x)), L_1(k,x))), L_2(k,x)), \dots), L_t(k,x))_1.$$

Note that a tree construction generalizes GGM in several ways:

1. We allow trees with larger degree (2^c), as long as the degree is a constant. (GGM uses a binary tree, with $c = 1$.)
2. We allow the label of the root of the tree to be an arbitrary function S_0 of k and x. (In GGM it is $S_0(k,x) = k$.)
3. We allow the function S_i that chooses the label of the children of a node labeled by y based on the value of $G(y)$ to be arbitrary function. Moreover, we allow it to be different in each level i of the tree. (In GGM $S_i(z,b) = S(z,b) = z_b$ for every level.)
4. We use an arbitrary function $L_i(k,x)$ to select which child to choose as the next node on the path down the tree at level i. (In GGM it is $L_i(k,x) = x_i$.)

[3] Here $x_{\leq i}$ denotes the first i bits of x.

Thus, the GGM construction is a $(t = n, c = 1)$ tree construction, with the functions $L_1, \ldots, L_t, S_0, \ldots S_t$ specified above, and if we use Levin's domain extension together with GGM, it becomes a $(t = \omega(\log n), c = 1)$ tree construction.

We prove that for every choice of the functions L_i, S_i, there is no $(\log n, O(1))$ tree construction.

Theorem 1.4 (Tree constructions). *Let $c \in \mathbb{N}$ be a constant. Then there is no (t, c)-tree black-box PRF construction from $r(n)$-bit stretch PRG, with $t(n) \leq \log n - \log\log n - \omega(1)$.*

We then significantly generalize this result to rule out an arbitrary oracle-aided algorithm that can call such a $(\log n, O(1))$ tree construction adaptively, and apply arbitrary post-processing.

Definition 1.5. *We say that a function family $\mathcal{Q} = \{\mathcal{Q}_{k,x}\}$ is a (t, c)-*Tree oracle*, if there exist functions $L_1, \ldots, L_t$ and $S_0, \ldots, S_t$ such that for every key k and input x, $|L_i(k, x)| \leq c$, and,*

$$\mathcal{Q}_{k,x}(s) = S_t(G(\ldots S_2(G(S_1(G(s), L_1(k, x))), L_2(k, x)), \ldots), L_t(k, x))_1.$$

That is, $\mathcal{Q}_{k,x}(s)$ is the above tree construction, when replacing the root $S_0(k, x)$ with the input s.

Theorem 1.6. *Let $G\colon \{0,1\}^n \to \{0,1\}^{n+r(n)}$, $c \in \mathbb{N}$ be a constant, and $t(n) \leq \log n - \log\log n - \omega(1)$ be a function. Let $\mathcal{Q}$ be a (t, c) tree oracle. Then there is no black-box PRF construction $\mathcal{F} = \{f_k\}$ from a PRG, such that*

$$f_k^G(x) = A^{\mathcal{Q}_{k,x}}(k, x)$$

for any algorithm A.

Tightness. Theorem 1.4 is tight both in terms of c and t:

- For any $t = \omega(\log n)$, there exists a $(t, 1)$-tree PRF construction $\{f_k\}_{k \in \{0,1\}^{\lambda(n)}}$ from n-bits stretch PRG, with $\lambda(n) = n$. This is the GGM construction with Levin's domain extension (note that applying the domain extension to a tree construction still results in a tree construction).
- For any $c = c(n) = \omega(1)$, there exists a $(\log n, 2^c)$-tree PRF construction $\{f_k\}_{k \in \{0,1\}^{\lambda(n)}}$ from $(2^c \cdot n)$-bits stretch PRG, with $\lambda(n) = n$. This can be shown by considering a shallow 2^c-ary tree instead of the binary tree in the GGM construction.

Special Case for One-Call Constructions. We say that a black-box PRF construction is a *one-call* construction, if in order to evaluate f_k^G on a point x we only need to call the PRG G once. In other words, a one-call construction is a construction of the form

$$f_k^G(x) = P(G(S(k, x)), k, x),$$

where the function S selects the query to the PRG and the function P computes some arbitrary function of the key k, input x, and the output of the PRG.

Each of our two main results gives some lower bound on a restricted type of such constructions as a special case. In particular, each of these results implies that there is no construction of a PRF f_k^G such that

$$f_k^G(x) = P(G(S(k,x)), L(k,x))$$

where $|L(k,x)| \in O(\log n)$ (notice that here the output of P is one bit).

In fact, Theorem 1.2 implies something stronger: that there is no PRF construction of the form $f_k^G(x) = P_2(P_1(G(S(k,x)), L(k,x)), k, x)$ where here P_2 can be dependent arbitrarily on k and x, as long as the output of P_1 is at most $n - \omega(\log r)$ bits, and $|L(k,x)| \in O(\log n)$:

Corollary 1.7. *Let $G\colon \{0,1\}^n \to \{0,1\}^{n+r(n)}$. Let L, P_1, and P_2 be functions such that for every k, x and s, $|L(k,x)| \in O(\log n)$ and $|P_1(G(s), L(k,x))| \leq n - \omega(\log r)$. Then there is no black-box PRF construction $\mathcal{F} = \{f_k\}$ from a PRG, such that*

$$f_k^G(x) = P_2(P_1(G(s), L(k,x)), k, x).$$

Does there exist a one-call black-box construction of PRF from PRG? While intuitively the answer seems to be no, we do not know how to prove it in general. We leave this as a fascinating (and elusive) open problem.

Lower Bound on the Key Length. Our last result shows that in any black-box PRF construction, the key length must be roughly equal to the input length for the PRG. That is, the length of the key cannot be much shorter than n (unless the domain is of polynomial size).

Theorem 1.8 (Lower bound on the key-length). *There is no black-box PRF construction from PRG $G\colon \{0,1\}^n \to \{0,1\}^{n+r(n)}$ such that $\lambda(n) \leq n - \omega(\log n)$.*

The proof of Theorem 1.8 is given in the full version of this paper.

2 Proof Overview

We now give some overview of the proof. We start with explaining the proof for a special case of one-call constructions: black-box constructions $\mathcal{F}^G = \left\{f_k^G\right\}_{k\in\lambda(n)}$ of the form

$$f_k^G(x) = P(G(S(k,x)), L(k,x)),$$

where $|S(k,x)| = n$, $|L(k,x)| = O(\log n)$ and $|P(G(S(k,x)), L(k,x))| = 1$. We then explain how to generalize the proof to get our lower bound for tree constructions. For simplicity, assume that $\mathcal{F}^G$ is a PRF construction from a length-doubling PRG $G\colon \{0,1\}^n \to \{0,1\}^{2n}$. Our goal is to construct an oracle $\mathcal{O}$, with

respect to there exists such a PRG G, and an efficient (with respect to $\mathcal{O}$) algorithm Break that breaks the security of $\mathcal{F}^G$. That is, Break distinguishes a truly random function from a function sampled from the family $\mathcal{F}^G = \{f_k^G\}_{k\in\lambda(n)}$, but cannot be used to break the security of G.

Eliminating L. We start by using the technique of Miles and Viola [MV11] to eliminate the function L. Intuitively, their technique shows that when the output of L is short, and for some type of distinguishers Break, it is enough to consider constructions of the form

$$f_k^G(x) = P'(G(S(k,x))).$$

In more detail, since the output length of L is $O(\log n)$, there exists some value z such that $L(k,x) = z$ with noticeable probability. Fix such z, and assume that we can show that there exists some class $\mathcal{G}$ of PRGs, and a function $\widehat{f}$, such that for every key k and input x, and for every PRG $G \in \mathcal{G}$, it holds that $P(G(S(k,x)),z) = \widehat{f}(k,x)$. That is, $f_k^G(x)$ can be evaluated on every input for which $L(k,x) = z$ without calling to G. Then, we can consider a simple distinguisher Break that breaks the security of $\mathcal{F}^G$:

1. Sample $x_1,\ldots,x_\ell \leftarrow \{0,1\}^n$ for $\ell \gg 2^{L(k,x)}|k|$
2. Query $x_1,\ldots,x_\ell$ to get $y_1 = f(x_1),\ldots,y_\ell = f(x_\ell)$.
3. Break^f outputs 1 if there exists k such that:
   (a) There are at least $2|k|$ i's such that $L(k,x_i) = z$.
   (b) $\widehat{f}(k,x_i) = y_i$ for every $i \in [\ell]$ with $L(k,x_i) = z$.
   Otherwise, Break^f outputs 0.

It is not hard to show that for a random function f, Break^f outputs 1 with negligible probability, while it outputs 1 on f_k^G with $1/\text{poly}$ probability, for every PRG G from the class $\mathcal{G}$ and over a randomly chosen key k.

The hope now is that the class $\mathcal{G}$ is large enough such that it contains a PRG that is secure against attackers with oracle access to Break. In our construction we will choose $\mathcal{G}$ such that we will be able to convert any PRG G into a PRG $G' \in \mathcal{G}$, in a black-box way. That is, if Break can be used to break the security of G', then it is possible to break the security of G. The proof now follows by the fact that a random function is a PRG with respect to any oracle with high probability [Imp11,GGKT05] (and thus there exists an oracle which is a secure PRG against Break).

Constructing $\widehat{f}$. To summarize, so far we showed that if we can prove that for some function $\widehat{f}$, and for every PRG G from a large enough class of PRGs $\mathcal{G}$, it holds that

$$P'(G(S(k,x)),z) = \widehat{f}(k,x),$$

then it follows that there is no black-box PRG construction of the form

$$f_k^G(x) = P(G(S(k,x)),L(k,x)).$$

In the following, let $P'(G(S(k,x))) = P(G(S(k,x)),z)$. While it is enough for us to consider P' with an output of one bit, we explain how to construct such $\widehat{f}$ with $\widehat{f}(k,x) = P'(G(S(k,x)))$ for any function P' with an output of at most $n/2$ bits, as this will be useful for us later.[4] Fix such a function $P' \colon \{0,1\}^{2n} \to \{0,1\}^{n/2}$. As a first step, assume that P' can be completed to a permutation. That is, assume that there exists some permutation $R \colon \{0,1\}^{2n} \to \{0,1\}^{2n}$, such that $P'(y) = R(y)_{\leq n/2}$ for every $y \in \{0,1\}^{2n}$. In this case, we claim it is easy to construct the function $\widehat{f}$ and the class $\mathcal{G}$.

Indeed, consider the permutation $\pi = R^{-1}$. For every PRG G, the function $G' = \pi \circ G$ (namely, $G'(s) = \pi(G(s))$) is still a PRG.[5] More importantly, for every such G' we get that

$$P'(G'(S(k,x))) = R(G'(S(k,x)))_{\leq n/2} = R(\pi(G(S(k,x))))_{\leq n/2} = G(S(k,x))_{\leq n/2}.$$

That is, the output of P' is the first $n/2$ output bits of G. We can now choose a PRG G such that $G(s)_{\leq n/2} = s_{n/2}$, and we get that

$$P'(G'(S(k,x))) = S(k,x)_{\leq n/2} \tag{1}$$

which implies what we wanted to show, by taking $\widehat{f}(k,x) = S(k,x)_{\leq n/2}$. We remark that Eq. (1) holds for any PRG G' such that

$$G'(s) = \pi(s_{\leq n/2} || G(s_{>n/2}),$$

and that G' is a PRG if G is, as we wanted to show. This concludes the proof for the case that P' can be completed to a permutation.

Dealing with an Arbitrary Function P'. We are left to deal with the case in which $\boldsymbol{P'}$ cannot be completed to a permutation. Recall that by choosing π to be the inverse of the function P' (or actually the inverse of the permutation R), we were able to show that $P'(\pi(y)) = y_{n/2}$ for every y. This allowed us to compute the output of P' only using the first $n/2$ bits of y. While we cannot find such an inverse π for any function P', we show it is possible to find a "pseudo-inverse" – a function π which is close to being a permutation, such that the first i bits of the output of $P'(\pi(y))$ can be computed by roughly the first i bits of y. This is stated in the following lemma.

Lemma 2.1 (Pseudo-inverse lemma). *Let $n \in \mathbb{N}$ be a number, $w = \omega(\log n)$ and $f \colon \{0,1\}^n \to \{0,1\}^n$ be a function. Then there exists a function $\pi \colon \{0,1\}^n \to \{0,1\}^n$, and functions $\{f_i\}_{i \in [n-w]}$ such that:*

1. $\mathrm{SD}(U_n, \pi(U_n)) \leq \mathrm{neg}(n)$
2. *For every $i \in [n-w]$, $f(\pi(x))_{\leq i} = f_i(x_{<i+w})$.*

[4] In the actual proof we show how to do it for outputs of length $n - \omega(\log n)$.

[5] We assume here that G is secure against adversaries with oracle access to π. We can construct such oracle PRG using [Imp11, GGKT05].

We remark that by the first condition above, $\pi \circ G$ is a PRG for any PRG G. For simplicity of this presentation, in the following we assume (the generally false assumption) that the above claim holds when setting w to be 0.

Back to our lower bound, by taking π to be the pseudo-inverse of P', we can finish the proof. Consider the same class of PRGs $\mathcal{G}$, and let $\widehat{f}(k,x) = f_{n/2}(S_0(k,x)_{\leq n/2})$, when $f_{n/2}$ is the function promised by Lemma 2.1 for $i = n/2$. We get that

$$\begin{aligned} P'(G'(S_0(k,x)) &= P'(\pi(S_0(k,x)_{\leq n/2}||G(S_0(k,x)_{>n/2})) \\ &= f_{n/2}(S_0(k,x)_{\leq n/2}) \\ &= \widehat{f}(k,x) \end{aligned}$$

which concludes the proof. We explain how we prove Lemma 2.1 in Lemma 2.2. In the following, we explain how to use Lemma 2.1 to generalize the above proof to work for tree constructions.

2.1 Tree Construction

We now explain how to use the same techniques as described above to prove our lower bound for tree constructions.

Fix functions $S_0, \ldots, S_t$ and $L_1 \ldots, L_t$ as in the definition of tree construction. As discussed above, since the total length of $L_1(k,x)|| \ldots ||L_t(k,x)$ is at most $t = O(\log n)$, we can fix its value. It is thus enough to show that there exists a function $\widehat{f}$ such that

$$\widehat{f}(k,x) = S_t(G(\ldots S_1(G(S_0(k,x))) \ldots))$$

for every PRG G from some large class $\mathcal{G}$.

We first observe that when all of the functions S_i are equal ($S_t = \cdots = S_1 = S$ for some function S) then this is easy to show. Indeed, set $P' = S$. Then our proof above shows that for the right choice of G, the first $n/2$ bits of the output of $S(G(S_0(k,x)))$ are only dependent on the first $n/2$ bits of $S_0(k,x)$.[6] More generally, it holds that $S(G(y))_{\leq n/2} = \tau(y_{n/2})$ for some function τ. Applying $S \circ G$ again, we get that

$$S(G(S(G(y))))_{\leq n/2}) = \tau(S(G(y))_{\leq n/2}) = \tau(\tau(y_{\leq n/2})).$$

More generally, the output of such depth t tree-construction is equal to

$$\tau^t(S_0(k,x)_{\leq n/2}),$$

and thus can be computed without calling to G.

Of course, this is not the case when the functions $S_1, \ldots, S_t$ are not all equal. Yet, we can use a similar idea. First, we observe that the same proof works also when the function τ is different in each level. That is, if we can

[6] Actually, $n/2 + \omega(\log n)$, but we ignore it for this presentation.

show that for any PRG $G \in \mathcal{G}$, and for every $i \in [t]$, there exists τ_i such that $S_i(G(y))_{\leq n/2} = \tau_i(y_{n/2})$. Then we will get what we wanted by considering $\widehat{f}(k,x) = \tau_t(\tau_{t-1}(\ldots\tau_1(S(k,x)_{\leq n/2})\ldots))$.[7]

But this is still too much to ask for. Indeed, consider the concatenation of the prefixes of the functions $S_1, \ldots, S_t$

$$S(y) := S_1(y)_{\leq n/2}||\ldots||S_t(y)_{\leq n/2}.$$

Then it can be the case that $S(y)_{\leq 2n} = y$. In this case we cannot hope to compute $S(G(z))$ only from the first $n/2$ bits of z, or without calling to G. Thus such $\tau_1, \ldots, \tau_t$ cannot exists.

However, the above requirement is still stronger than what we really need. Recall that we want to show that we can find $\widehat{f}$ such that

$$\widehat{f}(k,x) = S_t(G(\ldots S_1(G(S_0(k,x)))\ldots)),$$

and where the output of S_t is only one bit long. Assume that for every i, we can show that there exists τ_i such that $S_i(G(y))_{\leq 2^{t-i}} = \tau_i(y_{\leq 2\cdot 2^{t-i}})$. That is, the first bit of the output of S_t only depends on the first two bits of the output of S_{t-1}, which in turn only depends on the first 4 output bits of S_{t-2} and so on. The point is that when $t \leq \log n/4$, we get that the output of $\widehat{f}$ can be computed from the first $n/2$ bits of $S_0(k,x)$, which is exactly what we wanted to show.

So we now want to find an (almost) permutation π such that for every $i \in [t]$

$$S_i(\pi(y))_{\leq 2^{t-i}} = \tau_i(y_{\leq 2\cdot 2^{t-i}})$$

for some function τ_i. We can do it again using the pseudo-inverse lemma. Assume that $t \leq \log n/4$, and consider the function

$$P'(z) = S_t(z)_1||S_{t-1}(z)_{\leq 2}||S_{t-2}(z)_{\leq 4}||\ldots||S_2(z)_{\leq n/8}(z)||S_1(z)_{\leq n/4}(z).$$

Let π be the function promised by the psuedo-inverse. The crux of this choice is that, by the pseudo-inverse lemma, for every z, it holds that $P'(\pi(z))_{\leq i}$ only depends on the first i bits of z. By our construction of P', we get that for every i, $S_{t-i}(\pi(z))_{\leq 2^i}$ is only a function of the first

$$\left|S_t(z)_1||S_{t-1}(z)_{\leq 2}||\ldots||S_{t-i}(z)_{\leq 2^i}(z)\right| = \sum_{t\geq j\geq t-i} 2^{t-j} = \sum_{0\leq j\leq i} 2^j \leq 2^{i+1}$$

bits of z, as we wanted to get.

[7] This is already enough to get some lower bound. As Tianqi Yang commented to us, if we assume that the PRF construction is secure even when using different PRG in each one of the levels, we can construct each one of the functions τ_i by applying the pseudo-inverse lemma separately for each S_i. Specifically, letting π_i be the pseudo-inverse of S_i, and taking the PRG G_i to be $\pi_i \circ G$, we can get that $S_t(G_t(S_{t-1}(G_{t-1}(\ldots S_0(k,x)\ldots)))) = \widehat{f}(k,x)$ for some function $\widehat{f}$ (and for any $t \in o(n/\log n)$). Interestingly, the GGM construction has this type of security.

2.2 Pseudo-Inverse Lemma Proof Overview

We now give some intuition for how to prove Lemma 2.1. In the formal proof (Sect. 6) we take a different and more direct path, and define π more explicitly. Yet, the general approach is the same.

As explained above, in the special case in which f is a permutation, we can take $\pi = f^{-1}$. In this case $f(\pi(x)) = x$, and thus $f(\pi(x))_{\leq i} = x_{\leq i}$. We now show how to generalize it for any function.

We start with some notations. For every $i \in [n]$ and every prefix $a \in \{0,1\}^i$ of an image of f, let $f^{-1}(a)$ be the set of all inputs x such that $f(x)_{\leq i} = a$. We will construct π together with a set of inputs $\mathcal{S}_a \subseteq \{0,1\}^n$ for every such a, such that for every $z \in \mathcal{S}_a$ it will hold that $f(\pi(z))_{\leq i} = a$. In other words, we construct π such that $\pi(\mathcal{S}_a) \subseteq f^{-1}(a)$. Moreover, we will construct $\mathcal{S}_a$ in a way that allows us to determine for any z whether $z \in \mathcal{S}_a$ only by the first $|a| + w$ first bits of z. This will promise that the second property of Lemma 2.1 holds.

Since we want the above to hold for any prefix $a \in \{0,1\}^*$, it must hold for every such a that $\mathcal{S}_a = \mathcal{S}_{a0} \cup \mathcal{S}_{a1}$, and that $\mathcal{S}_{a0}$ and $\mathcal{S}_{a1}$ are disjoint. Thus, $\mathcal{S}_{a0}$ and $\mathcal{S}_{a1}$ are a partition of $\mathcal{S}_a$. In the following we construct such sets inductively: we start with the construction of the set $\mathcal{S}_\epsilon = \{0,1\}^n$, and for every prefix a explain how to split the set $\mathcal{S}_a$ into the sets $\mathcal{S}_{a0}$ and $\mathcal{S}_{a1}$. If by this construction we will be able to show that for every image $y \in \{0,1\}^n$ of f, the set $\mathcal{S}_y$ is of the same size as $f^{-1}(y)$, then we can construct a *permutation* π that fulfills the second property of Lemma 2.1. Indeed, we can choose such π that maps arbitrarily between elements in $\mathcal{S}_y$ to $f^{-1}(y)$ for every y. By the sizes of these sets we get that π is a permutation. By construction we also have that we can compute $f(\pi(z))_{\leq i}$ by checking for which sets $\{\mathcal{S}_a\}_{a \in \{0,1\}^i}$ z belongs. By construction again, this can be done by only considering the first $i + w$ bits of z.

Constructing the Sets $\mathcal{S}_a$. To finish the proof we need to explain how to construct such sets. For simplicity, in the following we assume that for every i and $a \in \{0,1\}^i$, it holds that $\left|f^{-1}(a)\right| = c_a \cdot 2^{n-i-w}$ for some integer $c_a \in \mathbb{N}$. That is, the number of inputs that f maps to a prefix a of length i is a multiplication of 2^{n-i-w}. Observe that if f is a permutation, then $\left|f^{-1}(a)\right| = 2^{n-i}$, and this condition holds.

We will show that we can construct $\mathcal{S}_a$ such that $|\mathcal{S}_a| = \left|f^{-1}(a)\right| = c_a \cdot 2^{n-i-w}$ for every a. Since the same holds for any image y, this finishes the proof for this simplified case. We show the above in induction on the length of the prefix. First notice that $\mathcal{S}_\epsilon = \{0,1\}^n = f^{-1}(\epsilon)$. Fix i and $a \in \{0,1\}^i$. Let $\mathcal{S}_a$ be a set of size $c_a \cdot 2^{n-i-w}$. We show how to construct $\mathcal{S}_{a0}$ and $\mathcal{S}_{a1}$.

First, since the membership in $\mathcal{S}_a$ can be determined by the first $i + w$ bits of z, for every $z \in \mathcal{S}_a$ there are 2^{n-i-w} strings z' such that $z'_{\leq i+w} = z_{\leq i+w}$ and $z' \in \mathcal{S}_a$. This means that $\mathcal{S}_a$ can be partitioned into c_a sets $\mathcal{S}^q_a$, indexed by $q \in \{0,1\}^{i+w}$, such that $\mathcal{S}^q_a$ is a set of size 2^{n-i-w} of all the strings in $\{0,1\}^n$ with prefix q. Furthermore, we can partition each such $\mathcal{S}^q_a$ into equal size sets $\mathcal{S}^{q0}_a$ and $\mathcal{S}^{q1}_a$, of all the strings with prefix $q0$ and $q1$ respectively. We get $2c_a$ such sets, of size exactly $2^{n-i-1-w}$ each. Moreover, membership in each such set can be determined by a $i + 1 + w$-length prefix.

Next, By assumption, for every $b \in \{0,1\}$, $\left|f^{-1}(ab)\right| = c_{ab} \cdot 2^{n-i-1-w}$, where $c_{a0} + c_{a1} = 2c_a$ (since $\left|f^{-1}(a0)\right| + \left|f^{-1}(a1)\right| = \left|f^{-1}(a)\right|$). We can thus simply take the set $\mathcal{S}_{a0}$ to be the union of c_{a0} of the sets $\mathcal{S}_a^{qb}$, and the set $\mathcal{S}_{a1}$ to be the union of the rest. It is not hard to see that this construction fulfills the requirements we wanted to have.

2.3 Using Arbitrary Oracle-Aided Post-processing

We now briefly explain how to generalize the above proofs to get Theorems 1.2 and 1.6. Specifically, in Theorems 1.2 and 1.6 the PRF construction has the form $f_k^G(x) = A^{\mathcal{Q}_{L(k,x)}}$, which allows making arbitrary number of calls to the oracle and to apply arbitrary post-processing. Yet, by the same proof we presented above, we can show that (for some value of $L(k,x) = z$), the answer to each query to the oracle $\mathcal{Q}_{L(k,x)}$ can be simulated without querying G. Thus, we can replace the oracle $\mathcal{Q}_{L(k,x)}$ with an alternative oracle that is independent of the choice of the PRG G. As explained in the above proof, this implies that we can break the PRF security without breaking the PRG.

2.4 Limitation of Our Methods

We next discuss some limitations of our proof technique. In all of the above results we can only deal with digest function L that outputs $O(\log n)$ bits (in the tree construction this corresponds to the total length of $L_1(k,x)||\ldots||L_t(k,x)$). Below we give two reasons for this barrier.

The first reason is that our lower bounds hold even if the PRG has super-polynomial stretch $r(n)$. As discussed above, it is easy to construct a PRF from such a PRG, by making one call and using a digest function L that outputs $\log r(n) = \omega(\log n)$ bits. Specifically, $f_k^G(x) = G(k)_x$ is a PRF.

The second reason is related to the choice of the attacker Break. In our proof Break is chosen before the PRG G, and Break does not make any direct calls to G. Moreover, G is chosen from a large family of function $\mathcal{G}$. We observe that for such attacker Break (and even when Break makes polynomial many calls to G), and when $\mathcal{G}$ is "large" enough, there is a PRF construction that fools Break, even when the stretch of the PRG is small. Specifically, assume that over a random choice of G from the family $\mathcal{G}$, it holds that the min-entropy of $G(x)$ is large given the entire truth table of G except for $G(x)$. Namely, assume that $H_\infty(G(x) \mid G(\{0,1\}^n \setminus \{x\})) \geq \omega(\log n)$. Then Break cannot distinguish between

$$f_{k_1,k_2}^G(x) = Ext(G(k_1 \oplus x), k_2)$$

to a random function, where Ext is a strong seeded extractor with seed of length $|k_2| \in \omega(\log n)$ (and such extractors exists [Vad+12]), and when G is uniformly sampled from the family $\mathcal{G}$. This holds since the answer $\mathsf{Ext}(G(k_1 \oplus x), k_2)$ for every query of Break is (almost) uniformly distributed, even given the entire view of Break so far.

A similar (and more general) barrier was shown by Miles and Viola [MV11]. Roughly speaking, [MV11] showed that when one-way functions do not exist, a function that is hard to invert (and in particular any PRG candidate), must be a function that is hard to compute. Then, [MV11] showed it is possible to use the Nisan and Wigderson [NW94] PRG construction, to get a PRF with a simple structure that is secure against any adversary Break that does not have oracle access to the PRG G. On the other hand, when one-way functions exist, there is a PRF construction that does not use the oracle to the PRG G at all. Back to our proof, when the family $\mathcal{G}$ is large enough, it will contain a function that is hard to compute, and thus there exists a PRG in the family $\mathcal{G}$ which can be used to construct a PRF.

2.5 Lower Bound on the Key Length

We finally explain how to prove our lower bound on the key-length. Fix a black-box PRF construction $\mathcal{F}$ with a key of length $\lambda(n) \leq n - \omega(\log n)$, and let $\Omega \subseteq \{0,1\}^{m(n)}$ be an arbitrary set of n elements from the domain of $\mathcal{F}$. We will construct a PRG G and an efficient algorithm that breaks $\mathcal{F}^G$ by querying all the elements in Ω. Toward this, let $Z\colon \{0,1\}^n \to \{0,1\}^{n+r(n)}$ be the zero function. That is $Z(q) = 0^{n+r(n)}$ for every $q \in \mathbb{N}$. Let $\mathcal{S}$ be the set of all queries made by $f_k^Z(x)$ for any possible key and for every $x \in \Omega$. Namely,

$$\mathcal{S} = \left\{q\colon f_k^Z(x) \text{ queries } Z(q) \text{ on some } x \in \Omega_n, k \in \{0,1\}^{\lambda(n)}\right\}.$$

The idea is that for any PRG G such that $G(q) = 0^{n+r(n)}$ for every $q \in \mathcal{S}$, it holds that $f_k^G(x) = f_k^Z(x)$. Thus, for every such PRG and for every $x \in \Omega$ we can compute $f_k^G(x)$ without calling to G, and therefore we can distinguish f_k^G from a random function.

On the other hand, it holds that the size of $\mathcal{S}$ is at most $\text{poly}(n) \cdot |\Omega| \cdot 2^{\lambda(n)} = 2^n \cdot \text{neg}(n)$. This implies that given a PRG G, we can construct a new PRG G', such that $G'(q) = 0^{n+r(n)}$ if $q \in \mathcal{S}$ or $G'(q) = G(q)$ otherwise. By the negligible size of $\mathcal{S}$ we get that G' is secure if G is.

3 Preliminaries

3.1 Notations

All logarithms are taken in base 2. We use calligraphic letters to denote sets and distributions, uppercase for random variables, and lowercase for values and functions. For $n \in \mathbb{N}$, let $[n] := \{1, \ldots, n\}$. Given a vector $v \in \Sigma^n$, let v_i denote its i^{th} entry, let $v_{<i} = (v_1, \ldots, v_{i-1})$ and $v_{\leq i} = (v_1, \ldots, v_i)$. For $x, y \in \{0,1\}^*$, we let xy and $x||y$ denote the concatenation of the strings x an y.

Let poly stand for the set of all polynomials. Let PPT stand for probabilistic poly-time. We say that an oracle-aided algorithm is q-query algorithm if it makes at most $q(n)$ queries to the oracle on any input of length n.

3.2 Distributions and Random Variables

When unambiguous, we will naturally view a random variable as its marginal distribution. The support of a finite distribution $\mathcal{P}$ is defined by $\text{Supp}(\mathcal{P}) := \{x\colon \Pr_{\mathcal{P}}[x] > 0\}$. For a (discrete) distribution $\mathcal{P}$, let $x \leftarrow \mathcal{P}$ denote that x was sampled according to $\mathcal{P}$. Similarly, for a set $\mathcal{S}$, let $x \leftarrow \mathcal{S}$ denote that x is drawn uniformly from $\mathcal{S}$. We use U_n to denote the uniform distribution over $\{0,1\}^n$. The *statistical distance* (also known as, *variation distance*) of two distributions $\mathcal{P}$ and $\mathcal{Q}$ over a discrete domain $\mathcal{X}$ is defined by $\text{SD}(\mathcal{P},\mathcal{Q}) := \max_{\mathcal{S}\subseteq\mathcal{X}} |\mathcal{P}(\mathcal{S}) - \mathcal{Q}(\mathcal{S})| = \frac{1}{2}\sum_{x\in\mathcal{S}} |\mathcal{P}(x) - \mathcal{Q}(x)|$. We use $\mathcal{P} \approx_\epsilon \mathcal{Q}$ to denote that $SD(\mathcal{P},\mathcal{Q}) \leq \epsilon$.

We will make use of the following two inequalities.

Fact 3.1 (Chernoff bound). *Let $A_1, ..., A_n$ be independent random variables s.t. $A_i \in \{0,1\}$. Let $\widehat{A} = \Sigma_{i=1}^n A_i$ and $\mu = \mathrm{E}\left[\widehat{A}\right]$. For every $\epsilon \in [0,1]$ It holds that:*

$$\Pr\left[\left|\widehat{A} - \mu\right| \geq \epsilon \cdot \mu\right] \leq 2 \cdot e^{-\epsilon^2 \cdot \mu/3}.$$

Claim 3.2. *Let $\mathcal{S}$ be a set of size at least k, and let $X_1, \ldots, X_t \leftarrow \mathcal{S}$ be t independent, uniformly distributed, random variables over $\mathcal{S}$. Then*

$$\Pr[|\{X_1, \ldots, X_t\}| < k] \leq k \cdot (1/2)^{t/k}$$

Proof. First, notice that for every fixed t, the probability of interest is smaller when $|\mathcal{S}|$ is larger. Thus, we can assume without loss of generality that $\mathcal{S} = [k]$. For every $j \in [k]$ let χ_j be the indicator that $X_i = j$ for some $i \in [t]$. Then

$$\Pr[|\{X_1, \ldots, X_t\}| < k] = \Pr[\exists j \in [k] \text{ s.t. } \chi_j = 0].$$

By observing that

$$\Pr[\chi_1 = 0] = (1 - 1/k)^t \leq (1/2)^{t/k},$$

we get that

$$\Pr[|\{X_1, \ldots, X_t\}| < k] \leq k \cdot (1/2)^{t/k}$$

as we wanted to show.

3.3 Pseudorandom Generators and Functions

We next define pseudorandom generators.

Definition 3.3 (Pseudorandom generator (PRG)). *An efficiently computable function G is an $r(n)$-bit stretch PRG if for every $n \in \mathbb{N}$ and $x \in \{0,1\}^*$, $|G(x)| = |x| + r(|x|)$, and for every efficient algorithm A, there exists a negligible function ν such that*

$$\left|\Pr_{x\leftarrow\{0,1\}^n}[A(1^n, G(x)) = 1] - \Pr_{y\leftarrow\{0,1\}^{n+r(n)}}[A(1^n, y) = 1]\right| \leq \nu(n).$$

Let $\mathcal{F}_{m,1}$ be the family of all functions from $\{0,1\}^m$ to $\{0,1\}$. In this work we show lower bounds on (semi) black-box constructions of pseudorandom functions from PRGs. We now define such black-box constructions.

Definition 3.4 (Black-Box PRF construction). *An efficient oracle-aided function family*

$$\mathcal{F} = \left\{ f_k^{(\cdot)} \colon \{0,1\}^{m(n)} \to \{0,1\} \right\}_{n \in \mathbb{N}, k \in \{0,1\}^{\lambda(n)}}$$

is a black-box PRF construction *from $r(n)$-bit stretch PRG if for every $G = \left\{ G_n \colon \{0,1\}^n \to \{0,1\}^{n+r(n)} \right\}_{n \in \mathbb{N}}$, every $q \in poly$, every q-query oracle-aided algorithm* Break *and every constant $c \in \mathbb{N}$ such that*

$$\left| \Pr_{k \leftarrow \{0,1\}^{\lambda(n)}} \left[\text{Break}^{f_k^G}(1^n) = 1 \right] - \Pr_{f \leftarrow \mathcal{F}_{m(n),1}} \left[\text{Break}^{f}(1^n) = 1 \right] \right| \geq 1/n^c$$

for infinitely many n's, there exists an efficient oracle-aided algorithm A and a constant $c' \in \mathbb{N}$ such that

$$\left| \Pr_{x \leftarrow \{0,1\}^n} \left[A^{\text{Break},G}(1^n, G(x)) = 1 \right] - \Pr_{y \leftarrow \{0,1\}^{n+r(n)}} \left[A^{\text{Break},G}(1^n, y) = 1 \right] \right| \geq 1/n^{c'}$$

for infinitely many n's.

$\mathcal{F}$ is a Black-Box weak PRF construction *if the same holds for all algorithms* Break *that query i.i.d. uniform queries to the oracle f.*

In our proof we will use the following well-known lemma, which states that there exists an oracle with respect to which PRGs exist.

Lemma 3.5. *Let $f \colon \{0,1\}^* \to \{0,1\}^*$, $w \colon \mathbb{N} \to \mathbb{N}$ and $r \colon \mathbb{N} \to \mathbb{N}$ be functions such that $n + r(n) \in 2^{o(w(n))}$. Then there exists a (possibly inefficient) function $G = \left\{ G_n \colon \{0,1\}^{w(n)} \to \{0,1\}^{n+r(n)} \right\}_{n \in \mathbb{N}}$ such that the following holds. For every efficient oracle-aided algorithm A, there exists a negligible function ν such that*

$$\left| \Pr_{x \leftarrow \{0,1\}^{w(n)}} \left[A^{f,G}(1^n, G(1^n, x)) = 1 \right] - \Pr_{y \leftarrow \{0,1\}^{n+r(n)}} \left[A^{f,G}(1^n, y) = 1 \right] \right| \leq \nu(n).$$

The proof of Lemma 3.5, which is given in the full version of this paper follows easily from the work of Gennaro, Gertner, Katz, and Trevisan [GGKT05] (see also [MV11,HHRS07]).

4 Structural Lower Bounds on PRF Constructions

In this part we present our structural lower bound on black-box PRF constructions, and derive our main lower bounds on Tree constructions and one-call constructions. For simplicity, in the following we assume that on security parameter

n, the PRF construction queries the PRG G only on inputs with length n. In the full version of this paper we generalize our results to get rid of this assumption.

To state our lower bound, we start with defining *sequential oracles*. Roughly speaking, a collection of functions $\{Q_z\}_{z\in\{0,1\}^*}$ is a sequential oracle if for some functions $P_1,\ldots,P_t$,

$$Q_z(s) = P_t(G(\ldots G(P_1(G(s),z))\ldots),z).$$

In other words, $\{Q_z\}_{z\in\{0,1\}^*}$ is a sequential oracle if for $p_0(s,z)=s$ and $p_i(s,z)=P_i(G(p_{i-1}),z)$ it holds that $Q_z(s)=p_t(s,z)$. In the definition below we let t be the depth of the function (that is, the number of adaptive calls to G), and v the length of the output of P_t. For simplicity we let $P(i,y,z)=P_i(y,z)$.

Definition 4.1 (Sequential oracle). *Let $t=t(n),\ell=\ell(n),v=v(n)$ and $r=r(n)$ be functions. A collection of oracles $\{Q_{n,z}\}_{n\in\mathbb{N},z\in\{0,1\}^{\ell(n)}}$ is a $(t(n),v(n))$-*sequential oracle *if for every $n\in\mathbb{N}$ there exists a function $P\colon [t(n)]\times\{0,1\}^{n+r(n)}\times\{0,1\}^{\ell(n)}\to\{0,1\}^n$ such that the following holds for every function $G\colon\{0,1\}^n\to\{0,1\}^{n+r(n)}$ and for every $n\in\mathbb{N}$.*

For $z\in\{0,1\}^{\ell(n)}$ and $s\in\{0,1\}^n$, let $p_0^G(s,z)=s$, and for every $i\in[t(n)]$, let $p_i^G(s,z)=P(i,G(p_{i-1}^G(s,z)),z)$. Then

$$Q_{n,z}^G(s) = p_{t(n)}^G(s,z)_{\le v(n)}.$$

For example, a tree construction can be written as $f_k^G(x)=Q_{n,L(k,x)}^G(S(k,x))$ for some functions L and S. Our goal is to show that when t and ℓ are not too large, there is no such tree construction. We actually show something stronger - that for every choice of function L, the function $Q_{n,L(k,x)}^G$ alone is *useless* to construct PRF. That is, there is no PRF construction that only uses $Q_{n,L(k,x)}^G$ as an oracle.

Theorem 4.2. *Let $c\in\mathbb{N}$ be a constant, and let $w,m,\lambda,r,\ell,v,t\colon\mathbb{N}\to\mathbb{N}$ be functions, such that $w(n)\in\omega(\log n+\log r(n))$, $\lambda\in$ poly, $\ell(n)\le c\log n$,*

$$m(n)\ge\ell(n)+\log(\lambda(n)+\ell(n)+10)+1$$

and

$$t(n)\le\log(1+\frac{n-w(n)}{\max\{v(n),w(n)\}}).$$

Then, for every $(t(n),v(n))$-sequential oracle $\{Q_{n,z}\}_{n\in\mathbb{N},z\in\ell(n)}$, every function family

$$L=\left\{L_n\colon\{0,1\}^{\lambda(n)}\times\{0,1\}^{m(n)}\to\{0,1\}^n\right\}_{n\in\mathbb{N}}$$

and for every oracle-aided algorithm A, there is no Black-Box (weak) PRF construction

$$\mathcal{F}=\left\{f_k\colon\{0,1\}^{m(n)}\to\{0,1\}\right\}_{k\in\{0,1\}^{\lambda(n)}}$$

from $r(n)$-bit stretch PRG, such that for all k,x,

$$f_k^G(x)=A^{Q_{n,L(k,x)}^G}(1^n,k,x).$$

We prove Theorem 4.2 in Sect. 5, but first we use it to derive lower bounds on two extreme cases of sequential oracles: Tree and depth-one oracles.

4.1 Tree Constructions

A tree construction is a black-box construction with sequential calls to the PRG, in which every call is only dependent on the output of the previous call, together with some small number of bits from the input and key. This type of construction is formally defined below.

Definition 4.3 (Tree construction). *A black-box construction of a PRF*

$$\mathcal{F} = \left\{ f_k \colon \{0,1\}^{m(n)} \to \{0,1\} \right\}_{k \in \{0,1\}^{\lambda(n)}}$$

from $r(n)$-bit stretch PRG is a (t,c)-Tree construction, if for every $n \in \mathbb{N}$, there exist functions $S_0 \colon \{0,1\}^{\lambda(n)} \times \{0,1\}^{m(n)} \to \{0,1\}^n, S \colon [t(n)] \times \{0,1\}^{n+r(n)} \times \{0,1\}^{c(n)} \to \{0,1\}^n$ and $L \colon [t(n)] \times \{0,1\}^{\lambda(n)} \times \{0,1\}^{m(n)} \to \{0,1\}^{c(n)}$ such that the following holds for every $x \in \{0,1\}^{m(n)}$, $k \in \{0,1\}^{\lambda(n)}$ and $G \colon \{0,1\}^n \to \{0,1\}^{n+r(n)}$.

Let $s_0 = S_0(k,x)$, and for every $i \in [t(n)]$, let $s_i = S(i, G(s_{i-1}), L(i,k,x))$ it holds that $f_k^G(x) = (s_{t(n)})_{\leq 1}$.

As a direct corollary of Theorem 4.2 we get a lower bound on the depth of every tree construction.

Corollary 4.4 (Lower bound for tree constructions – Theorem 1.4, restated). *Let $c \in \mathbb{N}$ be a constant, and let $\lambda \in poly$, $r \colon \mathbb{N} \to \mathbb{N}$, and $m(n) \geq c \cdot \log n + \log(\lambda(n) + c \log n + 10) + 1$ be functions. Then there is no (t,c)-Tree (weak) PRF construction $\mathcal{F} = \left\{ f_k \colon \{0,1\}^{m(n)} \to \{0,1\} \right\}_{k \in \{0,1\}^{\lambda(n)}}$ from $r(n)$-bit stretch PRG, with $t(n) \leq \log n - \log(\log n + \log r(n)) - \omega(1)$.*

Proof. The proof is by showing the (c,t)-Tree construction with constant c and $t(n) \leq \log n - \log(\log n + \log r(n)) - \alpha(n)$, for $\alpha(n) = \omega(1)$ can be implemented with one call to a $(1, t(n))$-sequential oracle. Specifically, let $w(n) = 2^{\alpha(n)} \cdot (\log n + \log r(n))$, $L'(k,x) = L(1,k,x)||\ldots||L(t(n),k,x)$, and let $\ell(n) = |L'(k,x)| = c \cdot t(n) \leq c \log n$. Let $P(i,y,z) = S(i,y,z_{ci+1,\ldots,c(i+1)})$, so that $P(i, G(s_{i-1}), L'(k,x)) = S(i, G(s_{i-1}), L_i(k,x))$ for every i.

Moreover, the algorithm A that given k,x calls to $Q^G_{L'(k,x)}$ with $S_0(k,x)$ and output the first bit of $P^G_{t(n)}(s, L'(k,x))$, implements $\mathcal{F}$.

A similar proof shows that there is no PRF construction even given an oracle to the tree construction.

Corollary 4.5 (Theorem 1.6, restated). *Let c, m, λ, r, ℓ be as in Theorem 4.2. Then, for every $(\log n - \log(\log n + \log r(n)) - \omega(1), 1)$-sequential oracle*

$$\{Q_{n,z}\}_{n \in \mathbb{N}, z \in \ell(n)},$$

and for every oracle-aided algorithm A, there is no Black-Box (weak) PRF construction

$$\mathcal{F} = \left\{ f_k \colon \{0,1\}^{m(n)} \to \{0,1\} \right\}_{k \in \{0,1\}^{\lambda(n)}}$$

from $r(n)$-bit stretch PRG, such that for all k, x,

$$f_k^G(x) = A^{Q^G_{n,L(k,x)}}(1^n, k, x).$$

4.2 Digest Functions

Directly from Theorem 4.2 we get the following theorem, which implies our result for one-call constructions.

Corollary 4.6 (Theorem 1.2, restated). *Let c, m, λ, r, ℓ be as in Theorem 4.2. Then, for every $(1, n-\omega(\log n + \log r(n))$-sequential oracle $\{Q_{n,z}\}_{n \in \mathbb{N}, z \in \ell(n)}$, and for every oracle-aided algorithm A, there is no Black-Box (weak) PRF construction*

$$\mathcal{F} = \left\{ f_k \colon \{0,1\}^{m(n)} \to \{0,1\} \right\}_{k \in \{0,1\}^{\lambda(n)}}$$

from $r(n)$-bit stretch PRG, such that for all k, x,

$$f_k^G(x) = A^{Q^G_{n,L(k,x)}}(1^n, k, x).$$

And as a special case we get the corollary for projection functions.

Corollary 4.7 (Theorem 1.1, restated). *Let c, m, λ be as in Theorem 4.2, and let $r \in$ poly. Let $i = \left\{ i_n \colon \{0,1\}^{\lambda(n)} \times \{0,1\}^{m(n)} \to [n + r(n)] \right\}$, and let $\mathcal{Q}^G_{n,i_n(k,x)}(s) = G(s)_{i_n(k,x)}$. Then, for every oracle-aided algorithm A, there is no Black-Box (weak) PRF construction*

$$\mathcal{F} = \left\{ f_k \colon \{0,1\}^{m(n)} \to \{0,1\} \right\}_{k \in \{0,1\}^{\lambda(n)}}$$

from $r(n)$-bit stretch PRG, such that for all k, x,

$$f_k^G(x) = A^{Q^G_{n,i_n(k,x)}}(1^n, k, x).$$

5 Proving Theorem 4.2

In this section we prove Theorem 4.2. We actually prove a stronger statement, with respect to a slightly stronger oracle, defined next.

Definition 5.1 (Augmented sequential oracle). *Let $t = t(n), \ell = \ell(n)$ and $r = r(n)$ be functions, and $v = v(n) = (v_1^n, \ldots, v_{t(n)}^n) \in \mathbb{N}^{t(n)}$ be a vector. A collection of oracles $\{Q_{n,z}\}_{n \in \mathbb{N}, z \in \{0,1\}^{\ell(n)}}$ is a v-sequential oracle if for every $n \in \mathbb{N}$ there exists a function $P \colon [t(n)] \times \{0,1\}^{n+r(n)} \times \{0,1\}^{\ell(n)} \to \{0,1\}^n$*

such that the following holds for every function $G\colon \{0,1\}^n \to \{0,1\}^{n+r(n)}$. *For* $z \in \{0,1\}^{\ell(n)}$ *and* $s \in \{0,1\}^n$, *let* $p_0^G(s,z) = s$, *and for every* $i \in [t(n)]$, *let* $p_i^G(s,z) = P(i, G(p_{i-1}^G(s,z)), z)$. *Then*

$$Q_{n,z}^G(s) = p_1^G(s,z)_{\le v_1} || \ldots || p_{t(n)}^G(s,z)_{\le v_{t(n)}}.$$

That is, in augmented sequential oracles, we also output some inner values computed during the computation of the output of the sequential oracle. We prove the following lemma.

Lemma 5.2. *Let* $c, w, m, \lambda, r, \ell$ *and* t *be as in Theorem 4.2. For every* $n \in \mathbb{N}$, *let* $v(n) = (v_1^n, \ldots, v_{t(n)}^n) \in \mathbb{N}^{t(n)}$ *be numbers such that*

$$v_i^n \ge w(n) + \sum_{t(n) \ge j > i} v_j^n, \qquad \text{and,} \qquad \sum_{i \in [t(n)]} v_i^n \le n - w(n),$$

and let $L_n\colon \{0,1\}^{\lambda(n)} \times \{0,1\}^{m(n)} \to \{0,1\}^{\ell(n)}$. *Then, for every sequential oracle* $\{Q_{n,z}\}_{n \in \mathbb{N}, z \in \ell(n)}$, *and for every oracle-aided algorithm* A, *there is no Black-Box (weak) PRF construction*

$$\mathcal{F} = \left\{ f_k\colon \{0,1\}^{m(n)} \to \{0,1\} \right\}_{k \in \{0,1\}^{\lambda(n)}}$$

from $r(n)$*-bit stretch PRG, such that for all* k, x,

$$f_k^G(x) = A^{Q_{n, L_n(k,x)}^G}(1^n, k, x).$$

Theorem 4.2 follows from Lemma 5.2 simply by choosing the right parameters.

Proof. (Proof of Theorem 4.2). Let $w'(n) = \max\{v(n), w(n)\}$, and observe that $w'(n) \in \omega(\log n + \log r(n))$. For every $n \in \mathbb{N}, i \in [t(n)]$, let $v_i^n = w'(n) \cdot 2^{t(n)-i}$. It follows that

$$\begin{aligned} v_i^n &= w'(n) \cdot 2^{t(n)-i} = w'(n) \cdot (1 + \sum_{t(n) \ge j > i} 2^{t(n)-j}) \\ &= w'(n) + \sum_{t(n) \ge j > i} v_j^n \ge w(n) + \sum_{t(n) \ge j > i} v_j^n \end{aligned}$$

and that $\sum_{i \in [t(n)]} v_i^n = w'(n)(2^{t(n)} - 1)$. Thus, $v(n) = (v_1^n, \ldots, v_{t(n)}^n)$ satisfies the conditions of Lemma 5.2, as long as

$$w'(n)(2^{t(n)} - 1) \le n - w(n)$$

which holds for

$$t(n) \le \log(1 + \frac{n - w(n)}{w'(n)}) = \log(1 + \frac{n - w(n)}{\max\{v(n), w(n)\}}).$$

Moreover, the v-augmented sequential oracle contains the first $v(n)$ bits of $p_{t(n)}^G(s,z)$, which is the output of the $(t(n), v(n))$-sequential oracle which is defined by the same function P. Thus, the v-augmented sequential oracle is strictly stronger than the $t(n)$-sequential oracle.

5.1 Proving Lemma 5.2

We now prove Lemma 5.2. Assume toward a contradiction that there exists a black-box PRF construction $\mathcal{F} = \left\{ f_k^{(\cdot)} \colon \{0,1\}^{m(n)} \to \{0,1\} \right\}_{k \in \lambda(n)}$ from $r(n)$-bit stretch PRG with the structure described in Lemma 5.2.

To prove Lemma 5.2, we will show that there exists an oracle $\mathcal{O}$, such that with respect to the oracle, there exists a PRG G, but $\mathcal{F}^G$ is not a PRF. In more detail, we will show that for some $z_n \in \{0,1\}^{\ell(n)}$, and for some PRG G, $Q_{z_n}^G$ can be computed on every input without calling to G. It follows that an adversary that only attempts to break the security of the PRF $\mathcal{F}$ on keys and inputs for which $L(k,x) = z_n$ cannot be used to break the security of G. We construct such an adversary below.

We next describe the distinguisher that breaks the security of $\mathcal{F}$.
Let $c, w, m, \lambda, r, \ell, t, v$ be as in Lemma 5.2. Let $\{Q_z\}_{n \in \mathbb{N}, z \in \{0,1\}^{\ell(n)}}$ be the v-sequential oracle used by $\mathcal{F}$, and for every $n \in \mathbb{N}$, let P_n be the function given by the definition of sequential oracle. Let A and L_n be the algorithm and function such that $f_k^G(x) = A^{Q^G_{n,L_n(k,x)}}(1^n, k, x)$ for every G, k and x. In the following, when n is clear from the context, we use L, P and Q_z to denote L_n, P_n and $Q_{n,z}$.

We start with the following simple claim, which states that there exists some value z such that $L(k,x)$ is equal to z with a good probability, over a random choice of the key k and the input x. Our distingusher will try to break the security of $\mathcal{F}$ on inputs k, x for which $L(k,x) = z$.

Claim 5.3. *For every $n \in \mathbb{N}$, there exists $z_n \in \{0,1\}^{\ell(n)}$ such that*

$$\Pr_{k \leftarrow \{0,1\}^{\lambda(n)}} \left[\Pr_{x \leftarrow \{0,1\}^{m(n)}} [L(k,x) = z_n] \geq 2^{-\ell(n)-1} \right] \geq 2^{-\ell(n)-1}.$$

The claim follows directly from the assumption that the output of $L(k,x)$ is short.

Proof. Let $z \in \{0,1\}^{\ell(n)}$ be the value that maximize

$$\Pr_{k \leftarrow \{0,1\}^{\lambda(n)}, x \leftarrow \{0,1\}^{m(n)}} [L(k,x) = z].$$

Then $\Pr_{k \leftarrow \{0,1\}^{\lambda(n)}, x \leftarrow \{0,1\}^{m(n)}} [L(k,x) = z] \geq 2^{-\ell(n)}$.

We get that

$$\begin{aligned}
2^{-\ell} &\leq \Pr_{k \leftarrow \{0,1\}^{\lambda(n)}, x \leftarrow \{0,1\}^{m(n)}} [L(k,x) = z] \\
&= \mathrm{E}_{k \leftarrow \{0,1\}^{\lambda(n)}} \left[\Pr_{x \leftarrow \{0,1\}^{m(n)}} [L(k,x) = z] \right] \\
&\leq \Pr_{k \leftarrow \{0,1\}^{\lambda(n)}} \left[\Pr_{x \leftarrow \{0,1\}^{m(n)}} [L(k,x) = z] \geq 2^{-\ell(n)-1} \right] \cdot 1 \\
&\quad + \Pr_{k \leftarrow \{0,1\}^{\lambda(n)}} \left[\Pr_{x \leftarrow \{0,1\}^{m(n)}} [L(k,x) = z] < 2^{-\ell(n)-1} \right] \cdot 2^{-\ell(n)-1} \\
&\leq \Pr_{k \leftarrow \{0,1\}^{\lambda(n)}} \left[\Pr_{x \leftarrow \{0,1\}^{m(n)}} [L(k,x) = z] \geq 2^{-\ell(n)-1} \right] + 2^{-\ell(n)-1}
\end{aligned}$$

which implies the claim.

In the following, for every $n \in \mathbb{N}$ let z_n be the value promised by Claim 5.3. The following lemma is the main crux of the proof.

Lemma 5.4. *There exist a function family*

$$\pi = \left\{ \pi_n \colon \{0,1\}^{n+r(n)} \to \{0,1\}^{n+r(n)} \right\}_{n \in \mathbb{N}}$$

and a function $\widehat{Q}$ *such that the following holds for every function* $G \colon \{0,1\}^n \to \{0,1\}^{n+r(n)}$ *with* $G(s)_{\leq n-w(n)} = s_{\leq n-w(n)}$*, and for every* $n \in \mathbb{N}$*.*

1. $SD(U_{n+r(n)}, \pi_n(U_{n+r(n)})) \leq 2 \cdot 2^{-w(n)}$*, and*
2. $Q_{z_n}^{\pi_n \circ G}(s) = \widehat{Q}(s)$ *for every* $s \in \{0,1\}^n$*.*

As we will show later, the first property promises that for every PRG G (which is secure against adversaries with oracle access to π), $\pi \circ G$ is also a PRG. The second property promises that for any PRG G that outputs the first $n - n^\epsilon$ bits of its seed, the output of the oracle $Q_{z_n}^{G'}$ can be computed without evaluating G' when using $G' = \pi \circ G$ as the black-box PRG. These two properties together allow us to break the security of any PRF construction that uses $Q_{z_n}^{G'}$ without breaking the security of G.

Lemma 5.4 is proven in Sect. 5.3, but first let us use Lemma 5.4 to define the distinguisher that breaks the security of the PRF. Let π and $\widehat{Q}$ be the function families promised by Lemma 5.4. We next define the distinguisher.

Algorithm 5.5. (Break)
Input: 1^n
Oracle: $f \colon \{0,1\}^{m(n)} \to \{0,1\}$
Operation:

1. *Randomly choose* $p(n) = 2^{\ell(n)+10} \cdot (\lambda(n) + \ell(n) + 10)^3$ *points* $x_1, \ldots, x_{p(n)} \leftarrow \{0,1\}^{m(n)}$.
2. *Computes* $y_i = f(x_i)$ *for every* $i \in [p(n)]$.
3. *For every* $k \in \{0,1\}^{\lambda(n)}$*, let* $\mathcal{G}_k = \{x_i \colon L(k, x_i) = z_n\}$.
4. *If for some* $k \in \{0,1\}^{\lambda(n)}$*, it holds that* $|\mathcal{G}_k| > \lambda(n) + \ell(n) + 10$*, and* $f(x)$ *is equal to* $A^{\widehat{Q}}(k, x)$ *for every* $x \in \mathcal{G}_k$*, return* 1*. Otherwise return* 0*.*

Lemma 5.2 follows from the following two claims. The first states that Break breaks the security of the PRF $\mathcal{F}$, when using a certain type of PRGs.

Lemma 5.6. *Let* $G \colon \{0,1\}^n \to \{0,1\}^{n+r(n)}$ *be a function such that for every* $n \in \mathbb{N}$ *and* $s \in \{0,1\}^n$*, it holds that* $G_n(s)_{\leq n-w(n)} = s_{\leq n-w(n)}$*. Let* $G' \colon \{0,1\}^n \to \{0,1\}^{n+r(n)}$ *be the function defined by* $G'(s) = \pi_n(G(s))$ *for* $s \in \{0,1\}^n$*. Then,*

$$\left| \Pr_{f \leftarrow F^{G'}} \left[\text{Break}^f(1^n) = 1 \right] - \Pr_{f \leftarrow \mathcal{F}_{m(n),1}} \left[\text{Break}^f(1^n) = 1 \right] \right| \geq 2^{-\ell(n)-10}.$$

The second lemma states that there is some oracle $\mathcal{O}$ with respect to Break is efficiently computable and there exists a PRG of the form needed in Lemma 5.6 with respect to $\mathcal{O}$.

Lemma 5.7. *There exists a function*

$$G = \left\{ G_n \colon \{0,1\}^{w(n)} \to \{0,1\}^{w(n)+r(n)} \right\}_{n\in\mathbb{N}}$$

such that G is a PRG with respect to the oracle $\mathcal{O} = (\pi, \text{Break}, G)$.

Proof. (Proof of Lemma 5.7). Immediate by Lemma 3.5.

Lemma 5.6 is proven below, but first we use them to prove Lemma 5.2.

Proof. (Proof of Lemma 5.2). For every $n \in \mathbb{N}$, let z_n be the value promised by Claim 5.3 and $\pi = \{\pi_n\}_{n\in\mathbb{N}}$ be the function promised by Lemma 5.4. Let $\mathcal{O} = (\pi, \text{Break}, G)$ be the oracle promised by Lemma 5.7.

Let $\widehat{G} \colon \{0,1\}^n \to \{0,1\}^{n+r(n)}$ be the function family defined by

$$\widehat{G}_n(s) = s_{\leq n-n^\epsilon} || G(s_{>n-n^\epsilon}).$$

Clearly, $\widehat{G}$ is a PRG if G is, Moreover, since $\pi = \{\pi_n\}_{n_\mathbb{N}}$ is efficiently computable with respect to $\mathcal{O}$, $G' = \pi \circ \widehat{G}$ is a PRG. Indeed, otherwise let D be a distinguisher that breaks the security of G'. We claim that $D' = D \circ \pi_n$ breaks the security of $\widehat{G}$. To see this, observe that

$$\begin{aligned}\Pr_{y\leftarrow\{0,1\}^n}\left[D'(\widehat{G}(y)) = 1\right] &= \Pr_{y\leftarrow\{0,1\}^n}\left[D(\pi_n(\widehat{G}(y))) = 1\right] \\ &= \Pr_{y\leftarrow\{0,1\}^n}[D(G'(y)) = 1]\end{aligned}$$

and that, since π_n is close to be a permutation,

$$\begin{aligned}\Pr_{y\leftarrow\{0,1\}^{n+r(n)}}[D'(y) = 1] &= \Pr_{y\leftarrow\{0,1\}^{n+r(n)}}[D(\pi_n(y) = 1] \\ &= \Pr_{y\leftarrow\{0,1\}^{n+r(n)}}[D(y) = 1] \pm \text{neg}(n).\end{aligned}$$

Thus D' breaks $\widehat{G}$ with roughly the same advantage of D breaking G', and therefore the advantage must be negligible.

Since G' is a PRG with respect to the oracle $\mathcal{O} = (\pi, \text{Break}, G)$, it is also a PRG with respect to the (weaker) oracle $\mathcal{O}' = (\text{Break}, G')$.

On the other hand, Break is efficiently computable with respect to $\mathcal{O}'$, and by Lemma 5.6, Break breaks the security of $\mathcal{F}$ when using G' as the oracle PRG. Thus, $\mathcal{F}$ is not a black-box construction of a PRF.

5.2 Proving Lemma 5.6

Lemma 5.6 follows directly by the following two claims.

Claim 5.8.

$$\Pr_{f \leftarrow \mathcal{F}_{m(n),1}}\left[\text{Break}^{f}(1^n) = 1\right] \leq 2^{-\ell(n)-10}$$

Proof. Let $\widehat{f}_k(x) = A^{\widehat{Q}_{L(k,x)}}(k,x)$. The probability that a random function agrees with $\widehat{f}_k$ on $\lambda(n)+\ell(n)+10$ inputs, for any fixed k, is at most $2^{-\lambda(n)-\ell(n)-10}$. The claim thus follows by the union bound over all possible $2^{\lambda(n)}$ keys.

Claim 5.9. *Let G' be as in Lemma 5.6. Then*

$$\Pr_{k \leftarrow \{0,1\}^{\lambda(n)}}\left[\text{Break}^{f_k^{G'}}(1^n) = 1\right] \geq 2^{-\ell(n)-5}$$

Proof. By Lemma 5.4, for every k and x with $L(k,x) = z_n$, it holds that $f_k^{G'}(x) = A^{\widehat{Q}_{L(k,x)}}(k,x)$. Thus, it is enough to show that with good probability over the choice of the key k, the size of the set $\mathcal{G}_k$ is larger than $\lambda(n)+\ell(n)+10$ with a good probability. That is,

$$\Pr_{k \leftarrow \{0,1\}^{\lambda(n)}, x_1,\ldots,x_{p(n)} \leftarrow \{0,1\}^{m(n)}}[|\mathcal{G}_k| \geq \lambda(n)+\ell(n)+10] \geq 2^{-\ell(n)-5}. \quad (2)$$

By Claim 5.3, with probability at least $2^{-\ell(n)-1}$ over the choice of k, the probability that for a random x it is the case that $L(k,x) = z_n$ is at least $2^{-\ell(n)-1}$. It is thus enough to show that for every such k, the size of $\mathcal{G}_k$ is at least $\lambda(n)+\ell(n)+10$ with probability at least 2^{-4}.

Fix such k, and let $X_1,\ldots,X_{p(n)}$ be the random variables taking the value of the queries made by Break in a random execution. Let

$$\mathcal{GX} = \left\{x \in \{0,1\}^{m(n)} \colon L(k,x) = z_n\right\}$$

be the set of all x's such that $L(k,x) = z_n$, and let $\mathcal{GI} = \{i \in [p(n)] \colon X_i \in \mathcal{GX}\}$ be the (random) set of all indexes such that $X_i \in \mathcal{G}_k$ in the execution of Break.

By our assumption on k and a simple use of Chernoff, $|\mathcal{GI}| \geq (\lambda(n)+\ell(n)+10)^2$ with probability at least $1/2$. Moreover, for every $i \in [p(n)]$, given the event that $i \in \mathcal{GI}$, X_i uniformly distributed over the set $\mathcal{GX}$. We thus get that for $q(n) = (\lambda(n)+\ell(n)+10)^2$,

$$\begin{aligned}
&\Pr[|\mathcal{G}_k| \geq \lambda(n)+\ell(n)+10] = \Pr[|\{X_i \colon i \in \mathcal{GI}\}| \geq \lambda(n)+\ell(n)+10] \\
&\geq \Pr[|\{X_i \colon i \in \mathcal{GI}\}| \geq \lambda(n)+\ell(n)+10 \wedge |\mathcal{GI}| \geq q(n)] \\
&\geq \Pr[|\{X_i \colon i \in \mathcal{GI}\}| \geq \lambda(n)+\ell(n)+10 \mid |\mathcal{GI}| \geq q(n)] \cdot 1/2 \\
&\geq 1/2 \cdot \Pr_{Y_1,\ldots,Y_{q(n)} \leftarrow \mathcal{GX}}\left[\left|\{Y_1,\ldots,Y_{q(n)}\}\right| \geq \lambda(n)+\ell(n)+10\right] \\
&\geq 1/4
\end{aligned}$$

where the last inequality follows by Claim 3.2, since by our bound on $m(n)$ and our assumption on k, $|\mathcal{GX}| \geq 2^{m(n)} \cdot 2^{-\ell(n)-1} \geq \lambda(n)+\ell(n)+10$.

We are now ready to prove Lemma 5.6.

Proof (Proof of Lemma 5.6). Immediate from Claim 5.8 and Claim 5.9.

5.3 Proving Lemma 5.4

In this part we prove Lemma 5.4. We will make use in the following lemma.

Lemma 5.10 (Pseudo-inverse lemma). *Let $n, w \in \mathbb{N}$ be numbers and $f\colon \{0,1\}^n \to \{0,1\}^{n-w}$ be a function. Then there exists a function $\pi\colon \{0,1\}^n \to \{0,1\}^n$, and functions $\{f_i\}_{i\in[n-w]}$ such that:*

1. *$SD(U_n, \pi(U_n)) \leq 2 \cdot 2^{-w}$*
2. *For every $i \in [n-w]$, $f(\pi(x))_{\leq i} = f_i(x_{<i+w})$.*

Lemma 5.10 is proven in Sect. 6. In the following we use Lemma 5.10 to prove Lemma 5.4.

Proof (Proof of Lemma 5.4). We start with the construction of π. Fix n and for every $i \in [t(n)]$, and $y \in \{0,1\}^{n+r(n)}$, let $S_i(y) = P(i, y, z_n)$. Let S'_i be the function defined by $S'_i(y) = S_i(y)_{\leq v_i}$. Finally, let $S = S'_{t(n)}||S'_{t(n)-1}||\ldots||S'_1$.

By Lemma 5.10 with respect to $f = S$ and $w = w(n)$, there exists a function $\pi_n\colon \{0,1\}^{r(n)} \to \{0,1\}^{r(n)}$ such that

$$SD(U_{r(n)}, \pi_n(U_{r(n)})) \leq 2 \cdot 2^{-w(n)},$$

and, $S'_{t(n)}(\pi_n(x)), \ldots, S'_i(\pi_n(x))$ can be computed by the first

$$w(n) + \sum_{i \leq j \leq t(n)} v_j \leq v_{i-1}$$

bits of x. That is, there exist functions $\widehat{S}_1, \ldots, \widehat{S}_{t(n)}$ such that $S'_i(\pi_n(x)) = \widehat{S}_i(x_{\leq v_{i-1}})$ for every $i \in [t(n)]$.

We next use $\widehat{S}_1, \ldots, \widehat{S}_{t(n)}$ to construct the function $\widehat{Q}$. Let $\widehat{p}_0^{\pi_n \circ G}(s) = s$, and for every $i \in [t(n)]$, let $\widehat{p}_i^{\pi_n \circ G}(s) = \widehat{S}_i(\widehat{p}_{i-1}^{G}(s))$. Define

$$\widehat{Q}(s) := \widehat{p}_1^{\pi_n \circ G}(s)||\ldots||\widehat{p}_{t(n)}^{\pi_n \circ G}(s).$$

Next, let $p_0^{\pi_n \circ G}(s) = s$, and for every $i \in [t(n)]$, let

$$p_i^{\pi_n \circ G}(s) = S_i(\pi_n(G(p_{i-1}^{\pi_n \circ G}(s)))).$$

To finish the proof of the lemma, we need to show that for any $G\colon \{0,1\}^n \to \{0,1\}^{r(n)}$ with $G(s)_{\leq n-w(n)} = s_{\leq n-w(n)}$, and for every s, it holds that

$$\widehat{Q}(s) = Q_{z_n}^{\pi_n \circ G}(s),$$

or more explicitly, we need to show that

$$\widehat{p}_1^{\pi_n \circ G}(s)||\ldots||\widehat{p}_{t(n)}^{\pi_n \circ G}(s) = p_1^{\pi_n \circ G}(s)_{\leq v_1}||\ldots||p_{t(n)}^{\pi_n \circ G}(s)_{\leq v_{t(n)}}. \tag{3}$$

To establish Eq. (3), in the following we prove with induction on $0 \leq i \leq t(n)$ that $\widehat{p}_i^{\pi_n \circ G}(s) = p_i^{\pi_n \circ G}(s)_{\leq v_i}$.

For $i = 0$ the above holds trivially, as by definition $\widehat{p}_0^{\pi_n \circ G}(s) = s = p_0^{\pi_n \circ G}(s)$. Next, assume that the above holds for $i - 1$. We get that

$$\begin{aligned} p_i^{\pi_n \circ G}(s)_{\leq v_i} &= S_i(\pi_n(G(p_{i-1}^{\pi_n \circ G}(s))))_{\leq v_i} = \widehat{S}_i(G(p_{i-1}^{\pi_n \circ G}(s))_{\leq v_{i-1}}) \\ &= \widehat{S}_i((p_{i-1}^{\pi_n \circ G}(s))_{\leq v_{i-1}}), \end{aligned} \tag{4}$$

where the first equality holds by the definition of $p_i^{\pi_n \circ G}(s)$, the second by the definition of $\widehat{S}_i$, and the last equality holds since by our choice of G it holds that $G(s)_{\leq v_{i-1}} = s_{\leq v_{i-1}}$ (recall that $v_{i-1} \leq n - w(n)$). Next, by the induction hypothesis we get that

$$\widehat{S}_i((p_{i-1}^{\pi_n \circ G}(s))_{\leq v_{i-1}}) = \widehat{S}_i((\widehat{p}_{i-1}^{\pi_n \circ G}(s)) = \widehat{p}_i^{\pi_n \circ G}(s), \tag{5}$$

where the last equality holds by the definition of $\widehat{p}_i^{\pi_n \circ G}(s)$. We now get the claim by combining Eqs. (4) and (5).

6 Proving the Pseudo-Inverse Lemma

In this section we prove Lemma 5.10, restated below.

Lemma 6.1 (Pseudo-inverse lemma, restated). *Let $n, w \in \mathbb{N}$ be numbers and $f\colon \{0,1\}^n \to \{0,1\}^{n-w}$ be a function. Then there exists a function $\pi\colon \{0,1\}^n \to \{0,1\}^n$, and functions $\{f_i\}_{i \in [n-w]}$ such that:*

1. *$SD(U_n, \pi(U_n)) \leq 2 \cdot 2^{-w}$*
2. *For every $i \in [n-w]$, $f(\pi(x))_{\leq i} = f_i(x_{<i+w})$.*

Proof (Proof of Lemma 5.10). We start with the construction of π. Fix n and w, and let $\epsilon = n \cdot 2^{-w}$. In the following we construct functions $\pi_1, \pi_2\colon \{0,1\}^n \to \{0,1\}^n$, and take $\pi = \pi_1 \circ \pi_2$ (namely, $\pi(x) = \pi_1(\pi_2(x))$). We will construct π_1 and π_2 such that π_1 will be a (perfect) permutation, while π_2 will only be close to a permutation, in the sense that

$$SD(U_n, \pi_2(U_n)) \leq \epsilon.$$

This will be enough to get that

$$U_n = \pi_2(U_n) \approx_\epsilon \pi_2(\pi_1(U_n)) = \pi(U_n)$$

as stated in the lemma.

We will construct π_1 such that $f' := f \circ \pi_1$ will be a monotone function. That is, for every $x_1 \leq x_2$ it holds that $f'(x_1) \leq f'(x_2)$, where here $\leq$ is with respect to the lexicographic order. We will then construct π_2 that fulfils both of the requirements of the lemma, with respect to this monotone function f'.

Constructing π_1 such that $f \circ \pi_1$ is monotone is straightforward. Let $y_1 \leq \cdots \leq y_r$ be all of the images of f in a lexicographic order. Define π_1 as follows: map the first $\left|f^{-1}(y_1)\right|$ elements in $\{0,1\}^n$ (according to the lexicographic order) to the set $f^{-1}(y_1)$. Then map the next $\left|f^{-1}(y_1)\right|$ elements of $\{0,1\}^n$ to $f^{-1}(y_2)$, and so on. In the last step, map the last $\left|f^{-1}(y_r)\right|$ elements of $\{0,1\}^n$ to $f^{-1}(y_r)$. It is not hard to see that π_1 is a permutation and that $f' = f \circ \pi_1$ is monotone.

We next define π_2. To do so, for every x, let $i(x)$ be the minimal index i such that

$$f'(x_{i+w}||0^{n-i-w})_{\leq i} \neq f'(x_{i+w}||1^{n-i-w})_{\leq i}.$$

or $i(x) = \perp$ if not such index exists. That is, $i(x)$ (if not $\perp$) is the first index such that we cannot predict the first i bits of $f'(x)$ from the first $i+w$ bits of x (recall that since f' is monotone, if $f'(x_{i+w}||0^{n-i-w})_{\leq i} = f'(x_{i+w}||1^{n-i-w})_{\leq i}$, then $f'(x_{i+w}||0^{n-i-w})_{\leq i} = f'(x_{i+w}||z)_{\leq i}$ for any $z \in \{0,1\}^{n-w-i}$). We now define π_2 as follows: for any x such that $i(x) = \perp$, let $\pi_2(x) = x$. Otherwise, let $\pi_2(x) = x_{\leq i(x)+w}||0^{n-i(x)-w}$.

To finish the proof of the lemma, we need to show that (1), π_2 is close to be a permutation, and (2), that there are functions $\{f_i\}$ such that $f(\pi(x))_{\leq i} = f'(\pi_2(x))_{\leq i} = f_i(x_{\leq i+w})$. To show that π_2 is a permutation, it is enough to prove that the number of $x \in \{0,1\}^n$ such that $\pi_2(x) \neq x$ is at most $2^n \cdot \epsilon$. That is, it is enough to show that there are at most $2^n \cdot \epsilon$ strings $x \in \{0,1\}^n$ such that $i(x) \neq \perp$.

To see the above, observe that by the monotonicity of f', for any $i \in [n-w]$, and for any prefix $a \in \{0,1\}^i$, there is at most one $z \in \{0,1\}^{i+w}$ such that $f(z||0^{n-i-w})_{\leq i} = a$ but $f(z||1^{n-i-w})_{\leq i} \neq a$. Indeed, take the minimal such z. If $f(z||0^{n-i-w})_{\leq i} = a$ but $f(z||1^{n-i-w})_{\leq i} \neq a$ then it must hold that $f(z||1^{n-i-w})_{\leq i} > a$, which means that $f(z'||0^{n-i-w})_{\leq i} > a$ for any $z' > z$.

Using the above observation, we get that there are at most 2^i strings $z \in \{0,1\}^{i+w}$ for which $f(z||0^{n-i-w})_{\leq i} \neq f(z||1^{n-i-w})_{\leq i}$. This implies, by definition of $i(x)$, that there are at most 2^i strings $x \in \{0,1\}^n$ such that $i(x) = i$. Summing over all possible values of $i \in [n-w]$, we get that there are at most $\sum_{i \leq n-w} 2^i = 2^{n-w+1} = \epsilon \cdot 2^n$ strings x with $i(x) \neq \perp$, which implies that π_2 is indeed close to be a permutation.

Finally, we need to construct the functions $\{f_i\}_{i \in [n-w]}$. For every $i \in [n-w]$, and $z \in \{0,1\}^{i+w}$ let $f_i(z)$ be defined as follows. If z is a prefix of some string $x \in \{0,1\}^n$ such that $i(x) \leq i$, then let $f_i(z) = f'(z_{\leq i(x)+w}||0^{n-i(x)-w})$. Otherwise, let $f_i(z) = f'(z||0^{n-i-w})$.

Now observe that by the definition of $i(\cdot)$, we can determine if $i(x) \leq i$ by looking on the first $i+w$ bits of x. Moreover, if $i(x) \leq i$, then we can determine the value of $i(x)$ by looking on the same first $i+w$ bits of x. Thus, if $i(x) \leq i$ then $f_i(x_{\leq i+w}) = f'(x_{\leq i(x)+w}||0^{n-i(x)-w}) = f'(\pi_2(x))$. On the other hand, when $i(x) \geq i$ or $i(x) = \perp$, we have again that $f_i(x_{\leq i+w}) = f'(x_{\leq i+w}||0^{n-i-w}) = f'(\pi_2(x))$, which concludes the proof.

Acknowledgment. We thank Yanyi Liu, Tamer Mour, and Tianqi Yang for very useful discussions.

References

[AR16] Applebaum, B., Raykov, P.: Fast pseudorandom functions based on expander graphs. In: Hirt, M., Smith, A. (eds.) TCC 2016. LNCS, vol. 9985, pp. 27–56. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-53641-4_2

[BMR10] Boneh, D., Montgomery, H.W., Raghunathan, A.: Algebraic pseudorandom functions with improved efficiency from the augmented cascade. In: Proceedings of the 17th ACM Conference on Computer and Communications Security, pp. 131–140 (2010)

[BPR12] Banerjee, A., Peikert, C., Rosen, A.: Pseudorandom functions and lattices. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 719–737. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-29011-4_42

[GGKT05] Gennaro, R., Gertner, Y., Katz, J., Trevisan, L.: Bounds on the efficiency of generic cryptographic constructions. SIAM J. Comput. **35**(1), 217–246 (2005)

[GGM86] Goldreich, O., Goldwasser, S., Micali, S.: How to construct random functions. J. ACM (JACM) **33**(4), 792–807 (1986)

[GMR01] Gertner, Y., Malkin, T., Reingold, O.: On the impossibility of basing trapdoor functions on trapdoor predicates. In: Proceedings 42nd IEEE Symposium on Foundations of Computer Science, pp. 126–135. IEEE (2001)

[HHRS07] Haitner, I., Hoch, J.J., Reingold, O., Segev, G.: Finding collisions in interactive protocols-a tight lower bound on the round complexity of statistically-hiding commitments. In: 48th Annual IEEE Symposium on Foundations of Computer Science (FOCS'07), pp. 669–679. IEEE (2007)

[Imp11] Impagliazzo, R.: Relativized separations of worst-case and averagecase complexities for NP. In: 2011 IEEE 26th Annual Conference on Computational Complexity, pp. 104–114. IEEE (2011)

[Lev87] Levin, L.A.: One way functions and pseudorandom generators. Combinatorica **7**(4), 357–363 (1987)

[LMN93] Linial, N., Mansour, Y., Nisan, N.: Constant depth circuits, Fourier transform, and learnability. J. ACM (JACM) **40**(3), 607–620 (1993)

[LW09] Lewko, A.B., Waters, B.: Efficient pseudorandom functions from the decisional linear assumption and weaker variants. In: Proceedings of the 16th ACM Conference on Computer and Communications Security, pp. 112–120 (2009)

[MV11] Miles, E., Viola, E.: On the complexity of non-adaptively increasing the stretch of pseudorandom generators. In: Ishai, Y. (ed.) TCC 2011. LNCS, vol. 6597, pp. 522–539. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-19571-6_31

[MV15] Miles, E., Viola, E.: On the complexity of constructing pseudorandom functions (especially when they don't exist). J. Cryptol. **28**(3), 509–532 (2015)

[NR04] Naor, M., Reingold, O.: Number-theoretic constructions of efficient pseudorandom functions. J. ACM (JACM) **51**(2), 231–262 (2004)

[NR99] Naor, M., Reingold, O.: Synthesizers and their application to the parallel construction of pseudo-random functions. J. Comput. Syst. Sci. **58**(2), 336–375 (1999)

[NRR00] Naor, M., Reingold, O., Rosen, A.: Pseudo-random functions and factoring. In: Proceedings of the Thirty-Second Annual ACM Symposium on Theory of Computing, pp. 11–20 (2000)

[NW94] Nisan, N., Wigderson, A.: Hardness vs randomness. J. Comput. Syst. Sci. **49**(2), 149–167 (1994)

[PW88] Pitt, L., Warmuth, M.K.: Reductions among prediction problems: on the difficulty of predicting automata. In: 1988 Structure in Complexity Theory Third Annual Conference, pp. 60–61. IEEE Computer Society (1988)

[RR94] Razborov, A.A., Rudich, S.: Natural proofs. In: Proceedings of the Twenty-Sixth Annual ACM Symposium on Theory of Computing, pp. 204–213 (1994)

[RTV04] Reingold, O., Trevisan, L., Vadhan, S.: Notions of reducibility between cryptographic primitives. In: Naor, M. (ed.) TCC 2004. LNCS, vol. 2951, pp. 1–20. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-24638-1_1

[Vad+12] Vadhan, S.P., et al.: Pseudorandomness. Found. Trends® Theor. Comput. Sci. **7**(1–3), 1–336 (2012)

[Val84] Valiant, L.G.: A theory of the learnable. Commun. ACM **27**(11), 1134–1142 (1984)