A Direct PRF Construction from Kolmogorov Complexity

Yanyi Liu[1(✉)] and Rafael Pass[1,2]

[1] Cornell Tech, New York, USA
yl2866@cornell.edu
[2] Tel-Aviv University, Tel Aviv, Israel
rafaelp@tau.ac.il

Abstract. While classic results in the 1980s establish that one-way functions (OWF) imply the existence of pseudorandom generators (PRG) which in turn imply pseudorandom functions (PRF), the constructions (most notably the one from OWFs to PRGs) is complicated and inefficient.

Consequently, researchers have developed alternative *direct* constructions of PRFs from various different concrete hardness assumptions. In this work, we continue this thread of work and demonstrate the first direct construction of PRFs from average-case hardness of the time-bounded Kolmogorov complexity problem $\mathsf{MK}^t\mathsf{P}[s]$, where given a threshold, $s(\cdot)$, and a polynomial time-bound, $t(\cdot)$, $\mathsf{MK}^t\mathsf{P}[s]$ denotes the language consisting of strings x with t-bounded Kolmogorov complexity, $K^t(x)$, bounded by $s(|x|)$.

In more detail, we demonstrate a direct PRF construction with quasi-polynomial security from mild avg-case of hardness of $\mathsf{MK}^t\mathsf{P}[2^{O(\sqrt{\log n})}]$ w.r.t the uniform distribution. We note that by earlier results, this assumption is known to be equivalent to the existence of quasi-polynomially secure OWFs; as such, our results yield the first direct (quasi-polynomially secure) PRF construction from a natural hardness assumptions that also is known to be implied by (quasi-polynomially secure) PRFs.

Perhaps surprisingly, we show how to make use of the Nisan-Wigderson PRG construction to get a cryptographic, as opposed to a complexity-theoretic, PRG.

1 Introduction

Pseudorandom functions (PRFs), introduced by Goldwasser, Goldreich, and Micali [14] in the 1980s are one of the most important cryptographic primitives. Roughly speaking, a PRF is a family of efficiently computable functions

Y. Liu—Work done while visiting the Simons Institute (during the Meta-complexity program) and visiting University of Washington. Supported by a JP Morgan fellowship. R. Pass—Supported in part by NSF Award CNS 2149305, AFOSR Award FA9550-18-1-0267, AFOSR Award FA9550-23-1-0387, ISF Grant No. 2338/23 and an Algorand Foundation award. This material is based upon work supported by DARPA under Agreement No. HR00110C0086. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the United States Government, DARPA, AFOSR or the Algorand Foundation.

M. Joye and G. Leander (Eds.): EUROCRYPT 2024, LNCS 14654, pp. 375–406, 2024.
https://doi.org/10.1007/978-3-031-58737-5_14

$\{f_z\}_{z\in\{0,1\}^*}$, where given a randomly sampled seed z, the outputs of $f_z(x)$ on (adversarially-selected) inputs x is indistinguishable from the outputs of a truly random function $F(x)$ (with respect to a computationally bounded machine). Most notably they are a central object in private-key cryptography: they give simple and direct constructions of private-key encryption, message authentication, and identification (between parties with shared keys) [13,15,34]. In addition, they have also found many other applications in cryptography, including resettable security [5], oblivious RAM [17] (and more), and also in computational complexity [45] and computational learning theory [51]; the reader is referred to a nice survey by Bogdanov and Rosen [4].

While classic results in the 1980s by Hastad, Impagliazzo, Levin and Luby [23] and Goldreich, Goldwasser and Micali [14] established that one-way functions (OWFs) imply the existence of pseudorandom generators (PRGs) [23] which in turn imply pseudorandom functions (PRFs) [14], the constructions (most notably the one from OWFs to PRGs) is complicated and inefficient. Although there has been great progress over the last decades in improving the construction of PRGs [19,20,25,50], the currently best constructions of [36] has a seed length $\omega(n^3/\log n)$ and requires calling the underlying OWFs $\omega(n^3/\log^2 n)$ times.

Direct Constructions of PRFs. Consequently, towards the goal of developing practical and provably secure PRF constructions, researchers have developed alternative *direct* constructions of PRFs from various different concrete hardness assumptions—most notably based on the Decisional Diffie-Hellman (DDH) assumption [38,39], the hardness of factoring Blum-integers [40], and the hardness of lattice problems [3].

In this work, we continue this thread of work and demonstrate the first direct constructions of PRFs from average-case hardness of the standard time-bounded Kolmogorov complexity problem $\mathsf{MK}^t\mathsf{P}[s]$ [1,2,22,28,29,46], where given a threshold, $s(\cdot)$, and a polynomial time-bound, $t(\cdot)$, $\mathsf{MK}^t\mathsf{P}[s]$ denotes the language consisting of strings x with t-bounded Kolmogorov complexity, $K^t(x)$, bounded by $s(|x|)$. In more detail, we demonstrate a direct PRF construction with *quasi-polynomial security* from mild average-case of hardness of $\mathsf{MK}^t\mathsf{P}[2^{\sqrt{\log n}}]$ w.r.t the uniform distribution.[1] We note that by earlier results, this assumption is known to be equivalent to the existence of quasi-polynomially secure OWFs; as such, our results yield the first direct (quasi-polynomially secure) PRF construction from a natural hardness assumption that also is known to be implied by PRFs.

To explain this result in more detail, let us first recall the notion of (time-bounded) Kolmogorov complexity, and the notion of (mild) average-case hardness that we rely on.

The $\mathsf{MK}^t\mathsf{P}$ Problem. Given a truth table $x \in \{0,1\}^n$ of a Boolean function, what is the size of the smallest "program" that computes x? This problem has fascinated researchers since the 1950 [48,52,53], and various variants of it have been considered depending on how the notion of a program is formalized.

[1] The threshold $2^{\sqrt{\log n}}$ is the inverse of $2^{\log^2 n} = n^{\log n}$.

For instance, when the notion of a program is taken to be circuits (e.g., with AND,OR,NOT gates), then it corresponds to the Minimum Circuit Size problem (MCSP) [27,48], and when the notion of a program is taken to be a time-bounded Turing machine, then it corresponds to the Minimum Time-Bounded Kolmogorov complexity problem ($\mathsf{MK}^t\mathsf{P}$) [1,2,22,28,29,46]. Our focus here is on the latter scenario. Given a string x describing a truth table, let $K^t(x)$ denote the t-bounded Kolmogorov complexity of x—that is, the length of the shortest string Π such that for every $i \in [n]$, $U(\Pi, i) = x_i$ within time $t(|\Pi|)$, where U is a fixed Universal Turing machine.[2]

Given a threshold, $s(\cdot)$, and a polynomial time-bound, $t(\cdot)$, let $\mathsf{MK}^t\mathsf{P}[s]$ denote the language consisting of strings x such that $K^t(x) \leq s(|x|)$; $\mathsf{MK}^t\mathsf{P}[s]$ is clearly in NP, but it is unknown whether it is NP-complete. In [30], Liu and Pass recently showed that when the threshold $s(\cdot)$ is "large" (more precisely, when $s(n) = n - c \log n$, for some constant c), then mild *average-case hardness* of this language w.r.t. the uniform distribution of instances is equivalent to the existence of one-way functions (OWF).[3]

Even more recently, a different work by Liu and Pass [31] demonstrated that when the threshold is smaller then an appropriate notion of average-case hardness of the problem characterizes quasi-polynomial or sub-exponential one-way functions (depending on the threshold). In more detail, when the threshold s is small, $\mathsf{MK}^t\mathsf{P}[s]$ is a sparse language so it can never the average-case hard w.r.t. the uniform distribution (simply saying NO succeeds with overwhelming probability). To deal with this issue, [31] thus argued that the right notion of average-case case hardness of a sparse language ought to *condition* on both YES and NO instances. In more detail, we refer to a language $L \subset \{0,1\}^*$ as $D(\cdot)$-dense if for all $n \in \mathbb{N}$, $|L_n| = D(n)$, where $L_n = L \cap \{0,1\}^n$. We say that a $D(\cdot)$-dense language L is $\alpha(\cdot)$ *hard-on-average** *with respect to* $T(\cdot)$*-time attackers* (*(T,α)-HoA**) if for all probabilistic T-time heuristics $\mathcal{H}$, for all sufficiently large n, there exist $\mu \in \{0,1\}$ such that,

$$\Pr[x \leftarrow \{0,1\}^n : \mathcal{H}(x) = \mu \mid L(x) = \mu] < 1 - \alpha(n^*),$$

[2] There are many ways to define time-bounded Kolmogorov complexity. We here consider the "local compression" version—which corresponds to the above truth table compression problem—and where the running-time bound is a function of the length of the program. A different version of (time-bounded) Kolmogorov complexity instead considers the size of the shortest program that outputs the *whole* string x. This other notion refers to a "global compression" notion, but is less appealing from the point of view of truth table compression, as the running-time of the program can never be smaller than the length of the truth table x.

[3] Strictly speaking, [30] considered the "global compression" version of Kolmogorov complexity, but when the threshold is large, these notions are essentially equivalent, and the result from [30] directly applies also the "local compression" notion of Kolmogorov complexity considered here.

where $n^* = \log D(n)$. n^* is referred to as the *normalized input length*. In other words, there does not exist a T-time "heuristic" that decides L with probability $1 - \alpha(n^*)$ *conditioned* on YES (and NO) instances.[4]

[31] showed that for any $\delta > 0$, quasi-polynomially secure and subexponentially secure OWFs are characterized by $(n^\delta, O(1/n^3))$-average-case* hardness of $\mathsf{MK}^t\mathsf{P}[s]$ where the threshold are $s(n) = 2^{O(\sqrt{\log n})}$ and $s(n) = \mathsf{poly}\log n$ respectively. Intriguingly, their result—following a literature on so-called *hardness magnification* [7–10,37,42–44] shows that it suffices to assume *sublinear* hardness of these problems to provide those characterizations (when the threshold is sublinear). While the original result of [31] only showed equivalence in the non-uniform regime, it was recently shown how to also establish an equivalence also in the uniform regime [33]; additionally, [33] also (implicitly) show that the equivalence still holds if the error parameter becomes larger—it is (sufficient and) also necessary to assume $(n^\delta, \frac{1}{n^\beta})$-average-case* hardness of $\mathsf{MK}^t\mathsf{P}[s]$ for any $\beta > 0$.

In more detail, focusing on quasi-polynomially secure OWFs, we have the following. We say that a function f is a *quasi-polynomially secure* OWF if there exists a constant $c > 0$ such that f is $(T, 1/T)$-one-way for $T(n) = 2^{c \log^2 n}$.

Theorem 1 ([31,33]). *For any polynomial $t(n) \geq (1+\varepsilon)n, \varepsilon > 0$, any $\beta > 0$, $\delta > 0$, the following are equivalent:*

- *Quasi-polynomially secure (resp non-uniformly quasi-polynomially secure) OWFs exist.*
- *There exists a constant $c > 0$, $s(n) = 2^{c\sqrt{\log n}}$, such that $\mathsf{MK}^t\mathsf{P}[s]$ is $(n^\delta, \frac{1}{n^\beta})$-HoA* (resp non-uniformly HoA*).*

Our Main Result. Our main result is a direct construction of quasi-polynomially secure PRFs (with input domain $\{0,1\}^{\Omega(\log^2 k)}$ where k is the seed length) from the average-case* of hardness of $\mathsf{MK}^t\mathsf{P}[2^{O(\sqrt{\log n})}]$ w.r.t the uniform distribution and with respect to attackers of size n^3. (Formally proved in Sect. 5.3.)

Theorem 2 (Main Theorem). *Consider some threshold $s(n) = 2^{c\sqrt{\log n}}$, $c > 0$, polynomial $t(n) \geq (1+\varepsilon)n, \varepsilon > 0$, and any $\beta > 0$. Assume that $\mathsf{MK}^t\mathsf{P}[s]$ is $(n^3, \frac{1}{n^\beta})$-HoA* (resp. non-uniformly $(n^3, \frac{1}{n^\beta})$-HoA*). Then, there exists a quasi-polynomially secure (resp. non-uniformly quasi-polynomially secure) PRF h : $1^\lambda \times \{0,1\}^{\tilde{O}(\lambda^{1+\beta})} \times \{0,1\}^{\Omega(\log^2 \lambda)} \to \{0,1\}$.*

[4] The reason the error probability, α is a function of the logarithm of the number of YES-instances (i.e., the "normalized" input length, n^*, as opposed to just n (i.e., the logarithm of the number of instances) is to ensure that this notion meaningfully relaxes the notion of a one-sided heuristic, also for very sparse languages. If it wasn't, then a $1/n$-heuristic* could not make *any* mistakes on YES instances when the languages contains less than n YES-instances.

As noted above (see Theorem 1), the above assumption is equivalent to the existence of quasi-polynomially secure OWFs. Thus, Theorem 2 follows from Theorem 1 and the results of [14,23] (which show how to get a quasi-polynomially secure PRF from any quasi-polynomially secure PRG, with an explicit reduction). The point here is that we present a *direct* proof of this result, without passing through the result of [23], and thus as a result the construction becomes much simpler and more efficient. See the paragraph below discussing the efficiency of the construction.

As far as we know, our results thus yield the first direct PRF construction from a natural hardness assumption that also is known to be implied by PRFs.

A Non-Black-Box Security Reduction. We highlight that whereas we provide an explicit security reduction when proving the security of our PRF, the reduction is *non-black box*. In particular, we are relying on the fact that the PRF attacker is computationally bounded and has a small description. For this reason, the proof for the uniform case does not directly imply security in the non-uniform setting and we need to work a bit harder to demonstrate security also in the non-uniform setting. Roughly speaking, the non-black box nature of the reduction stems from the fact that we will use the attacker to "compress" the instance string x (in order to determine if its K^t complexity is small).

On the PRF Domain Size. We highlight that our PRF only has a $O(\log^2 \lambda)$ bit input domain where λ is the security parameter. Such (small domain) PRFs suffices for typical applications of PRFs [13] (e.g., to CPA-secure private-key encryption [18]). Of course we can always extend the domain using the standard tree construction [13] (but at a loss in efficiency). Alternatively, we note that if we consider hardness of $\mathsf{MK}^t\mathsf{P}[s]$ with an even smaller threshold $s(n) = \mathsf{poly}\log(n)$ (which is equivalent to subexponentially secure OWFs), then the construction directly extends to give a PRF with a λ^ϵ-bit input domain; see Theorem 18 for a generalized version of Theorem 2, Corollary 19, and Corollary 20.

On the Efficiency on the PRFs. We present an explicit reduction from an attacker that breaks the security of our PRF to breaking $\mathsf{MK}^t\mathsf{P}[s]$ on a particular instance $x \in \{0,1\}^n$. The efficiency of the PRF is a function of the time-bound t and the length of the threshold $\lambda = s(n)$ (which roughly equals the "normalized input length" of $\mathsf{MK}^t\mathsf{P}[s]$). In particular, its running time is $\tilde{O}(\lambda^\beta) \cdot t(\lambda)$ where β is any constant > 0, and the seed length is $\tilde{O}(\lambda^{1+\beta})$. At first it would seem that the efficiency of the construction is "too good to be true"—since $\lambda = s(n)$ is *sublinear* in the length n of the instance $x \in \{0,1\}^n$ we reduce security from. The point, however, is that the $\mathsf{MK}^t\mathsf{P}[s]$ problem can be trivially decided in time $2^{s(n)}$ (and is generally conjectured to be hard for time $2^{\Omega(s(n))}$—this is referred to as the *Perebor Conjecture* [48])—so the "fair" way to measure efficiency is in terms of the threshold s (and not the instance length n), and this is why we let the security parameter be defined as $\lambda = s(n)$ (instead of defining it as n).

Comparisons with Existing PRFs. Let us briefly compare our PRF with existing constructions. We first focus our attention on a "baseline" PRF, in which we apply the generic transformation of [14,23] (from OWFs to PRFs) to the

OWF construction of [31]. Given that both (our and the baseline) constructions base their security on the hardness of $\mathsf{MK}^t\mathsf{P}[s]$, we here consider a linear running time bound $t(n) = O(n)$, and $(n^3, \frac{1}{n^\beta})$-average-case* hardness of $\mathsf{MK}^t\mathsf{P}[s]$ for some arbitrarily small constant $\beta > 0$. In this setting, the [31] OWF has both the running time and the seed length of $\tilde{O}(\lambda^{1+\beta})$. However, recall that even the start-of-the-art OWF-to-PRG construction [36] incurs a $\tilde{O}(\lambda^3)$ blow up in both the seed length and the running time, whereas our PRF construction achieves runtime $\tilde{O}(\lambda^{1+\beta})$ and seed length $\tilde{O}(\lambda^{1+\beta})$.

We also provide comparisons with the PRFs constructed in [3,38], and [40]. We highlight that these constructions are based on hardness assumptions that are not known to be also implied by PRFs. Nevertheless, we still give an efficiency comparison, the results of which are summarized in Table 1.

Table 1. Efficiency comparisons with existing PRFs. (All assumptions are quasi-polynomially hard, $\beta > 0$ is an arbitrarily small constant.)

	Seed Length	Runtime	Input Length
LWE [3]	$\tilde{O}(\lambda^2)$	$\tilde{O}(\lambda^2)$	$\log^2 \lambda$
R-LWE [3]	$\tilde{O}(\lambda)$	$\tilde{O}(\lambda)$	$\log^2 \lambda$
DDH [38], Factoring [40]	$O(\lambda)$	$\tilde{O}(\lambda)$	$\log^2 \lambda$
DDH [38], Factoring [38]	$O(\lambda)$	$\tilde{O}(\lambda^2)$	λ
Ours	$\tilde{O}(\lambda^{1+\beta})$	$\tilde{O}(\lambda^{1+\beta})$	$\log^2 \lambda$

Notice that [3] only gets PRF with input length $\log^2 \lambda$ (rather than λ), just as we do, from quasi-poly assumptions.

We, as well as [3], can always use standard domain extension (incurring an overhead of λ in terms of running time) to extend the domain to λ bits. In essence, all of the constructions (except for the LWE one) have the same efficiency. Ours is β worse in the exponent for any arbitrary small $\beta > 0$, and the LWE based construction is strictly worse, losing a factor of λ.

1.1 Construction Overview

We here provide a brief outline of the construction. Towards this, let us first briefly review the Nisan-Wigderson PRG [41].

The NW Construction. The construction NW starts off with a function f that is assumed to be average-case hard to compute (with probability better than $1/2 + \delta$) over random inputs $\in \{0,1\}^\ell$, and is parameterized by a collection of sets of indexes $\mathcal{I} = \{I_i\}_{i \in [m]}$ referred to as the *designs*; these design sets I_i, $|I_i| = \ell$ are efficiently computable given i and in their simplest implementation involve evaluating a polynomial "indexed" by i—more details on this construction below.[5]

[5] This polynomial-based design construction appeared in [41], and is used to obtain the so-called "low-end" derandomization. There are many other constructions of

Next, given a seed y, to compute the bit i of output of the PRG NW, we simply apply f to the "projected" input y_{I_i} defined as y restricted to the indexes in I_i:

$$\mathsf{NW}^f_{\mathcal{I}}(y) = f(y_{I_1}) \dots f(y_{I_m})$$

See Fig. 1 for an illustrative example of the NW construction (in which we will employ the design construction that we describe in the next paragraph).

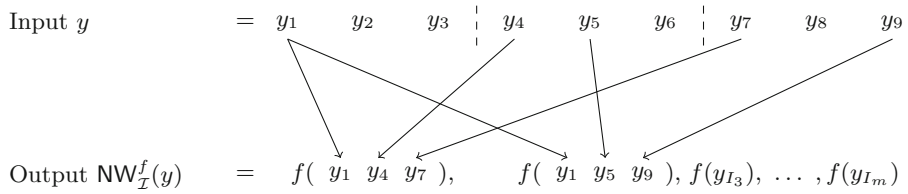

Fig. 1. An illustrative example of the NW construction with function $f : \{0,1\}^\ell \to \{0,1\}$ and designs $\mathcal{I}$. In this example, $\ell = q = 3$. The design construction is based on polynomials, where $I_1 = \{1,4,7\}$ corresponds to the polynomial $p_1(x) = 0$, and $I_2 = \{1,5,9\}$ corresponds to the polynomial $p_2(x) = x$.

For concreteness, let us recall the design construction of [41] that we will rely on. Let ℓ denote the input length to f, and let q denote a prime (or the smallest power of 2) such that $q \geq \ell$. We consider designs with universe $[d]$ where $d = \ell q$. The seed length of the NW generator, $|y|$, will also be picked to be d. We interpret each number $j \in [d]$ as a tuple in $[\ell] \times [q]$. Each set $I_i \subseteq [d]$ is defined to be the set $\{j_k = (k, p(k))\}_{k \in [\ell]}$ where p is the i-th polynomial of $\mathbf{F}_q$ (according to some canonical enumeration of all polynomials in $\mathbf{F}_q$, in a degree increasing order). Notice that each I_i contains ℓ elements, and thus the length of the projected seed y_{I_i} is equal to the input length of f.

The NW Reconstruction Procedure. It will be useful to review the security reduction of the NW PRG. Roughly speaking, [41] show that any distinguisher D (of the output of the PRG and a random string) with advantage ε, can be used to approximate f with probability $1/2 + O(\varepsilon/m)$ using a small (but larger than $m \cdot m^{1/\log \ell}$) bits of non-uniform advice z. This is referred to as the NW reconstruction procedure. (In particular, if f cannot be approximated by programs of size $|D| + |z|$ this yields a contradiction.)

The NW Construction as a PRF. Let us highlight one particular feature of the NW PRG when using the *above* design construction (based on polynomials). Given the seed y and the output index i, we can efficiently—i.e., in time polynomial in $(|i|, \ell)$ rather than in (i, ℓ)—compute the set I_i. Besides this, computing the ith output bit of the generator only needs one call to the underlying

designs, but, for our purposes, the polynomial-based one is useful as it is efficient in $|i|$.

hard function f. In other words, the output bits of the NW PRG are *locally computable*. As a consequence, if we are able to show that the above generator actually is a PRG with superpolynomial stretch (and f is efficient), then it actually also yields a PRF; this will be the approach that we will rely on.

Our PRF. Consider some running time bound $t(n) \geq (1+\varepsilon')n$, $\varepsilon' > 0$, threshold function $s(n) = 2^{\sqrt{\log n}}$, and hardness parameter $\alpha(\lambda) = 1/\lambda^{\beta}$. We construct a PRF $h : \{0,1\}^{\tilde{O}(\lambda^{1+\beta})} \times \{0,1\}^{\Omega(\log^2 \lambda)} \to \{0,1\}$ whose security can be based on α-average-case* hardness of $\mathsf{MK}^t\mathsf{P}[s]$ of input length n such that $s(n) = \lambda$.[6] To simplify notation, we here abuse the notation and specify h rather as a PRG of the form $h : \{0,1\}^{\tilde{O}(\lambda^{1+\beta})} \to \{0,1\}^{n=2^{\log^2 \lambda}}$, but note that due to our use of the NW PRG, each bit of the output h is locally computable (i.e., efficiently as a function of the seed and the index i) so this actually yields the desired PRF.

The construction of the (PRG-representation of) h proceeds in the following three steps.

- We start by appealing to the [30,31] OWF construction. Given a seed $\Pi' \in \{0,1\}^{\lambda+1}$, and an input $i \in \{0,1\}^{\log n}$, the function f removes all '0' from the end of Π' (if any), and removes an extra '1' from the end.[7] Let Π denote the remaining string. f simply outputs the bit produced by Π after running on input i for $t(|\Pi|)$ steps. In more detail,

$$f(\Pi', i) = U(\Pi(i), 1^{t(|\Pi|)})$$

  where U is the universal Turing machine fixed in the definition of K^t.
- Next, relying on the Nisan-Wigderson generator, we define the function $g(\Pi', y)$ as

$$g(\Pi', y) = \mathsf{NW}^{f(\Pi',\cdot)}_{\mathcal{I}}(y)$$

  where $\Pi' \in \{0,1\}^{\lambda+1}$, $f(\Pi', \cdot)$ is a function $\{0,1\}^{\log n} \to \{0,1\}$, $y \in \{0,1\}^d$, $d = O(\log^2 n) = O(\log^4 \lambda)$, and $\mathcal{I}$ is the polynomial-based design as introduced above.[8] The output size of the NW generator is set to be $m = n^{\varepsilon} = 2^{\varepsilon \log^2 \lambda}$ (where $\varepsilon = 1/8$).

  In more detail, g divides its seed into Π' and y, instantiates the NW generator with hard function $f(\Pi', \cdot)$, and simply computes the NW generator on seed y. The seed length of g is $|\Pi'| + |y| = (\lambda + 1) + O(\log^4 \lambda) = O(\lambda)$.
- Finally, we amplify the security of g, by considering the function h that takes the XOR of g on many independent seed. In more detail, h is defined as

$$h(\Pi'_1, y_1, \ldots, \Pi'_\gamma, y_\gamma) = g(\Pi'_1, y_1) \oplus \ldots \oplus g(\Pi'_\gamma, y_\gamma)$$

[6] In this proof overview, we assume that n is a power of 2 for simplicity.

[7] The [30] construction was slightly different; the above modified version has a slightly tighter analysis and shaves $\log \lambda$ bits in the input length.

[8] The $O(\log^2 n) = O(\log^4 \lambda)$ seed length of the PRG comes from our choice of designs. We note that there are explicitly computable designs that support our use of the NW PRG with only $O(\log n)$ seed length. See [21] and [11, Lemma A.2]. It may be that those designs lead to a more practical construction.

where $\gamma = \log n/\alpha = \lambda^{\beta} \log^2 \lambda$.

Note that the seed length of h is $O(\lambda) \cdot \gamma \leq O(\lambda^{1+\beta} \log^2 \lambda)$, and the output size will be $n^{\varepsilon} = 2^{\varepsilon \log^2 \lambda}$ (so its corresponding PRF has input domain $\{0,1\}^{\Omega(\log^2 \lambda)}$). Also notice that the running time of our PRF is $O(t(\lambda) \cdot \lambda^{\beta} \log^2 \lambda)$. (The reader is refer to Sect. 5.2 for a more detailed presentation of our PRF.)

We remark that there are two aspects of the construction that a-priori seem strange:

- In the derandomization literature, NW generators are used to fool attackers with smaller running time than the time needed to compute the hard function, whereas we are dealing with attackers that are more powerful than what is required to run the construction;
- The above PRG construction (without the amplification step), is very similar to the derandom-ization-style PRG[9] constructions based on the hardness of time-bounded Kolmogorov complexity problems appearing in [32]. These constructions, however, just as most more recent NW generators [26,47] are instantiated with a hard function encoded by error-correcting codes (ECCs) in order to make sure the reconstruction procedure is able to actually compress the input (since the "plain" NW reconstruction procedure only shows how the function can be computed with non-trivial probability). Note that using ECCs would not work in our setting as to get a PRF—we would need the ECC to be locally encodable, and such ECC cannot exist.

Both of the above differences mean that we cannot rely on simply the "standard" NW proof. However, we shall show that since we are starting off with average-case hardness of a *particular* hard function (i.e., based on $\mathsf{MK}^t\mathsf{P}[s]$), we are able to overcome these issues.

1.2 Proof Overview

We briefly explain why the above construction is secure, assuming α-average-case* hardness of $\mathsf{MK}^t\mathsf{P}[s]$ where $\alpha(\lambda) = 1/\lambda^{\beta}$ and $s(n) = 2^{\sqrt{\log n}}$. The proof will proceed in two steps. First, we will show that the function g above (i.e., applying just the plain NW generator on the sampled function) satisfies a weak PRG property. Next, we show that the general XOR construction used in the final PRG h amplifies such a weak PRG into a standard one. Finally, as noted above, since each bit is locally computable, the PRG actually yields a PRF.

Let us start by describing the notion of a weak family of PRGs and how they can be amplified into a standard PRG using the construction employed in function h.

Weak Families of PRGs. We refer to a family of functions $\{g_j\}_{j \in \{0,1\}^{\lambda}}$ as a *α-weak family of PRGs* if for every efficient distinguisher D, it holds that with probability $\alpha(\lambda)$ over $j \leftarrow \mathcal{U}_{\lambda}$, D distinguishes $g_j(\mathcal{U}_d)$ from uniform with at most negligible probability.

[9] In fact, hitting set generators.

In essence, with reasonable probability over the index, we get a full-fledged (strong) PRG. We remark that this notion is different from previous notions of weak PRGs [12,35] in the aspect that we consider a family of functions, whereas these other definitions consider a single PRG whose output is only weakly indistinguishable from random. A weak family of PRGs also yields such a weak PRG when the expansion is long enough so as to recover the index (which indeed will be the case for us). The issue is that the weak PRG it gives has indistinguishability gap of $1 - \alpha$, but the results of [12,35] only apply when the indistinguishability gap is smaller than $1/2$. Consequently, we directly leverage the fact that the once a "good" index of the family has been sampled, the PRG is actually fully indistinguishable from random. Relying on this, we next present a general proof that such weak families of PRGs can be amplified into a standard PRG[10] using the XOR construction and while relying on proof techniques similar to those used in Yao's classic hardness amplification theorem [16,54].

Showing that g is a Weak Family of PRGs. The central part of our paper is demonstrating that the function g can be viewed as a $\Omega(\alpha)$-weak family of PRGs assuming α-average-case* hardness of $\mathsf{MK}^t\mathsf{P}[s]$. In more detail, we define a weak family of PRGs $g'_{\Pi'}(y) = g(\Pi', y)$, and assume for contradiction that there exists a distinguisher that breaks its security for $1 - \alpha/O(1)$ fraction of index Π' of the family. We aim to use this distinguisher to decide $\mathsf{MK}^t\mathsf{P}[s]$ on average (conditioned on both YES- and NO-instances). The general approach will combine ideas from [6,24,30–32]. Roughly speaking, to decide an instance x, we view x as a hard function to use in the NW generator (as in [24,32], following [6]), except that we do not rely on an ECC as done by [24,32]), and feed the output of this generator to the attacker. If the attacker succeeds in distinguishing, we answer YES and otherwise NO. The key point is that, intuitively, if the distinguisher succeeds, then by the NW reconstruction argument, we can get a short description of program that approximates x with non-trivial probability, and intuitively, most NO instances do not have such short approximate description. Let us make two observations here: (1) the reason we do not need to rely on an ECC is exactly the fact that NO instances do not have even short *approximate* descriptions (as we shall soon argue), (2) the use of the NW reconstruction algorithm (similar to its use in [6,24,32]) is what makes the reduction non-black box since we are relying on the fact that the distinguisher has a small description.

We next need to argue that on random YES instances, the distinguisher actually works; this will rely on a probabilistic domination argument from [30,31] and it is here that we require the original distinguisher (breaking the PRGs g') to succeed for a large fraction of indices.

In more detail, assume for contradiction that there exists a distinguisher D that breaks the $(\alpha/O(1))$-weak family of PRGs g'; that is, over at least a $1 - \alpha/O(1)$ fraction of the indices Π', D breaks $g_{\Pi'}$ with a non-negligible advantage.

[10] For the ease of presentation, we only consider standard (polynomially secure) PRGs and negligible distinguishing advantage. Our proof will also show that the PRG (after amplification) is quasi-polynomially secure, as desired.

Notice that we can assume D breaks each such $g_{\Pi'}$ with advantage at least $\frac{1}{2^{\varepsilon \log^2 \lambda}}$ (which is negligible in λ) and also notice that $\frac{1}{2^{\varepsilon \log^2 \lambda}} = \frac{1}{n^{\varepsilon}}$. We will use D to decide $\mathsf{MK}^t\mathsf{P}[s]$ with probability at least $1 - \alpha$ conditioned on both YES instances and NO instances. Let us assume for simplicity in this proof overview that D is a deterministic uniform distinguisher. (In the actual proof, we will deal with probabilistic distinguishers, or even non-uniform algorithms.)

First note that by the security (of the reconstruction procedure) of the Nisan-Wigderson generator [41] (see also [6,24,32]), if x is a random NO instance, then with very high probability (much larger than $1 - \alpha$) over x, D cannot succeed in distinguishing between $\mathcal{U}_{n^\varepsilon}$ and $\mathsf{NW}^x(\mathcal{U}_d)$—the reason for this is that if D could do so, then the Nisan-Wigderson reconstruction procedure would approximate x over at least a $1/2 + 1/n^{2\varepsilon}$ fraction of coordinates (but in a randomized local fashion). In addition, the reconstruction procedure only requires $O(n^{2\varepsilon})$ bits as advice, so we have managed to approximately compress x. Observe that the distribution of a random NO instance is statistically close to the uniform distribution, so it remains to show that such an approximate local compression is impossible for almost all of random strings $x \in \{0,1\}^n$.

Random Strings Cannot be Approximately Compressed. It is well-known that by a standard counting argument, most of random strings cannot be exactly compressed (and thus have high Kolmogorov-complexity). Proving that they cannot be locally $(1/2 + 1/n^{2\varepsilon})$-approximated (by randomized programs of size $n^{2\varepsilon}$) requires a slightly more delicate argument. We will show that a (fixed) randomized program Π can only $(1/2 + 1/n^{2\varepsilon})$-approximate a uniform random string x with very small probability; the proof is then concluded using a union bound over all programs (of size $n^{2\varepsilon}$).

Let P denote the set of strings $x \in \{0,1\}^n$ such that given x, the randomized program Π on input a random index $i \in [n]$ (with fresh randomness) computes x_i with probability $\geq 1/2 + 1/n^{2\varepsilon}$. Our goal is to show that $|P|$ is very small.

Towards this, the key idea is to derandomize Π, and next simply argue that most random string will be far from the output of the deterministic version of Π. At first it would seem that we can simply fix the best random tape for Π; the issue, however, is that Π may succeed to compute different bits i with different random tapes. To overcome this issue, we use an averaging argument to show that for each $x \in P$, there exists at least a $\frac{1}{2n^{2\varepsilon}}$ fraction of random tapes r such that the output of the deterministic machine Π_r (the program Π with its random tape fixed to r) is $(1/2 + 1/(2n^{2\varepsilon}))$-close to x. From this it follows that over random x, r, the probability that Π_r produces a string that is $(1/2 + 1/(2n^{2\varepsilon}))$-close to x is at least

$$\frac{1}{2n^{2\varepsilon}} \cdot \frac{|P|}{2^n}$$

We are now ready to derandomize Π. Observe that if we fix the random tape of Π to a string r^* that maximizes the above probability (over a random x), we have that the output of Π_{r^*} is $(1/2 + 1/(2n^{2\varepsilon}))$-close to x is at least

$$\frac{1}{2n^{2\varepsilon}} \cdot \frac{|P|}{2^n}$$

over random $x \in \{0,1\}^n$.

Next, notice that the deterministic program Π_{r^*} will output a fixed string, but by a Chernoff bound, it follows that a *random* string is $(1/2 + 1/n^{2\varepsilon})$-close to it with probability at most

$$e^{-\Omega(n^{1-4\varepsilon})}$$

(To see this, note that each bit of the random string matches the fixed string independently with probability 1/2, so the distance between them is simply the sum of n independent random variables with expectation 1/2.) Therefore, we have that

$$\frac{|P|}{2^n} \leq 2n^{2\varepsilon} e^{-\Omega(n^{1-4\varepsilon})}$$

Finally, by taking a Union bound over all randomized program of size at most $n^{2\varepsilon}$, we conclude that the probability that a random string x can be approximately compressed is at most

$$2^{n^{2\varepsilon}} \cdot 2n^{2\varepsilon} e^{-\Omega(n^{1-4\varepsilon})} = 2n^{2\varepsilon} \cdot 2^{-\Omega(n^{1-4\varepsilon})+n^{2\varepsilon}}$$

In addition, the above probability will be negligible when $2\varepsilon < 1 - 4\varepsilon$ (which will hold since $\varepsilon = 1/8$).

Why the Distinguisher D Works on YES Instances. On the other hand, when x is a random YES instance of $\mathsf{MK}^t\mathsf{P}[s]$, we claim that D will manage to distinguish between $\mathsf{NW}^x(\mathcal{U}_d)$ and $\mathcal{U}_{n^\varepsilon}$, for at least a $1-\alpha$ faction of x. Observe that if x is a YES instance, then x will appear as the truth table of the function $f(\Pi', \cdot)$ for some $\Pi' \in \{0,1\}^{\leq \lambda}$ (e.g., the K^t-witness of x), and thus $\mathsf{NW}^x(\mathcal{U}_d)$ will always be one PRG inside the family of PRGs that g' defines. Since D has to break a $1 - \alpha/O(1)$ fraction of PRGs in the family, it would seem that D has to distinguish between $\mathsf{NW}^x(\mathcal{U}_d)$ and $\mathcal{U}_{n^\varepsilon}$ for at least a $1-\alpha$ fraction of x. However, the distribution of a random YES instance is very different from how the truth table of $f(\Pi', \cdot)$ is distributed in our construction of g. So we have no guarantee how the distinguisher D performs if we sample x from the random-YES distribution. To deal with this issue, we rely on an argument in [30, 31] to show that the $f(\Pi', \cdot)$-truth-table distribution "dominates" the random-YES distribution, and therefore if D succeeds with probability $1 - \alpha/O(1)$ over the $f(\Pi', \cdot)$-truth-table distribution, D must succeed also with probability $1 - (\alpha/O(1)) \cdot O(1) \geq 1 - \alpha$ over the random-YES distribution. This concludes that we can always decide $\mathsf{MK}^t\mathsf{P}[s]$ with probability $\geq 1-\alpha$ conditioned on both YES and NO instances, and concludes the construction of a $(\alpha/O(1))$-weak family of PRGs.

2 Preliminaries

For any two random variables X and Y defined over some set $\mathcal{V}$, we let $\mathsf{SD}(X,Y) = \max_{T \subseteq \mathcal{V}} |\Pr[X \in T] - \Pr[Y \in T]| = \frac{1}{2}\sum_{v \in \mathcal{V}} |\Pr[X = v] - \Pr[Y = v]|$ denote the *statistical distance* between X and Y. It will be helpful to note

that the expression is maximixed when the "distinguisher" $T = \{v : \Pr[X = v] > \Pr[Y = v]\}$.

Let $\mathcal{U}_n$ denote the uniform distribution over $\{0,1\}^n$, for any $n \in \mathbb{N}$.

2.1 Time-Bounded Kolmogorov Complexity

We define the notion of t-time-bounded Kolmogorov complexity that we rely on. We consider some universal Turing machine U that can emulate any Turing machine M with polynomial overhead. The universal Turing machine U receives as input a description/program $\Pi \in \{0,1\}^* = (M, w)$ where M is a Turing machine and $w \in \{0,1\}^*$ is an input to M; we let $U(\Pi(i), 1^{t(|\Pi|)})$ denote the output of $M(w, i)$ when emulated on U for $t(|\Pi|)$ steps.

Definition 3. *Let U be a universal Turing machine and $t(\cdot)$ be a polynomial. Define*

$$K^t(x) = \min_{\Pi \in \{0,1\}^*} \{|\Pi| : \forall i \in [|x|], U(\Pi(i), 1^{t(|\Pi|)}) = x_i\}.$$

We remark that the notion of time-bounded Kolmogorov complexity has been defined in a lot of different ways [2,28,29,46,48]; the definition we consider here is the "local compression" version [31] where the program Π is required to efficiently output each individual bit x_i of the string x, given i as input.

Let $\mathsf{MK}^t\mathsf{P}[s(n)]$ be a language consisting of strings x with K^t-complexity at most $s(|x|)$. We recall the following fact about (time-bounded) Kolmogorov complexity.

Fact 4 ([31]) *There exists a constant c such that for every polynomial $t(n) \geq (1+\varepsilon)n, \varepsilon > 0$, the following holds:*

(1) For every $x \in \{0,1\}^*$, $K^t(x) \leq |x| + c$;
(2) For every $n \in \mathbb{N}$, every function $0 < s(n) < n$, $2^{\lfloor s(n) \rfloor - c} \leq |\mathsf{MK}^t\mathsf{P}[s(n)] \cap \{0,1\}^n| \leq 2^{\lfloor s(n) \rfloor + 1}$.

2.2 Average-Case* Hardness

We introduce the notion of average-case* hardness, defined in [31]. On a high-level, average-case* hardness provides a meaningful notion of average-case hardness w.r.t. two-sided error heuristics for *sparse* languages. Before describing the definition, let us first define the density of a language: We say that a language $L \subset \{0,1\}^*$ is $D(\cdot)$-dense if for all $n \in \mathbb{N}$, $|L_n| = D(n)$, where $L_n = L \cap \{0,1\}^n$. Now we are ready to define the notion of average-case* hardness.

Definition 5 (Average-case* Hardness). *We say that a $D(\cdot)$-dense language L is $\alpha(\cdot)$* hard-on-average* (resp non-uniformly $\alpha(\cdot)$ hard-on-average*) with respect to $T(\cdot)$-time attackers ((T,α)-HoA* (resp non-uniformly (T,α)-HoA*)) *if for all probabilistic T-time (resp non-uniform T-time) heuristics $\mathcal{H}$, for all sufficiently large n, there exist $\mu \in \{0,1\}$ such that,*

$$\Pr[x \leftarrow \{0,1\}^n : \mathcal{H}(x) = \mu \mid L(x) = \mu] < 1 - \alpha(n^*),$$

where $n^* = \log D(n)$. n^* is referred to as the normalized input length.

In other words, there does not exist a T-time "heuristic*" that decides L with probability $1 - \alpha(n^*)$ *conditioned* on YES (and NO) instances. We refer the reader to [31] for how average-case* hardness meaningfully generalizes (and lies between) the two standard notions of average-case hardness (errorless and 2-sided error average-case hardness).

In this work, we are interested in the average-case* hardness of $\mathsf{MK}^t\mathsf{P}[s]$. We note that the normalized input length n^* of $\mathsf{MK}^t\mathsf{P}[s]$ on input length n is roughly as large as the threshold $s(n)$.

Claim 1. *Let c be the constant as in Fact 4. For any polynomial $t(\cdot)$, any function $0 < s(n) < n$, for any input length n, it follows that $s(n) - c \leq n^* \leq s(n) + 1$.*

Proof: This claim immediately follows from Fact 4. ∎

2.3 One-Way Functions and $\mathsf{MK}^t\mathsf{P}[s]$

We recall the standard definitions of one-way functions.

Definition 6. *Let $f : \{0,1\}^* \to \{0,1\}^*$ be a polynomial-time computable function. f is said to be a (T, ε)*-one-way function ((T, ε)-OWF) *(resp non-uniformly secure (T, ε)-OWF) if for any probabilistic $T(n)$-time (resp non-uniform $T(n)$-time) algorithm $\mathcal{A}$, for all sufficiently large $n \in \mathbb{N}$,*

$$\Pr[x \leftarrow \{0,1\}^n; y = f(x) : \mathcal{A}(1^n, y) \in f^{-1}(f(x))] < \varepsilon(n)$$

In addition, we say that f is *quasi-polynomially secure* if it is $(T, 1/T)$-one-way for some constant $c > 0$ and some function $T(n) = 2^{c \log^2 n}$. We say that f is *subexponentially secure* if it is $(T, 1/T)$-one-way for some constant $\varepsilon > 0$ and some function $T(n) = 2^{n^\varepsilon}$.

As mentioned in the introduction, [31] showed that OWFs are characterized by average-case* hardness of $\mathsf{MK}^t\mathsf{P}[s]$ and how small the threshold is determines how hard (or secure) the OWF we obtain. As also mentioned, while the original result of [31] only showed equivalence in the non-uniform regime, it was recently shown how to also establish an equivalence also in the uniform regime [33]; additionally, [33] also (implicitly) show that the equivalence still holds if the error parameter becomes larger—it is (sufficient and) also necessary to assume $(n^\delta, \frac{1}{n^\beta})$-average-case* hardness of $\mathsf{MK}^t\mathsf{P}[s]$ for any $\beta > 0$. We here formally state their results.

Theorem 7 ([31,33]) *For any polynomial $t(n) \geq (1+\varepsilon)n, \varepsilon > 0$, any $\beta > 0$, any $\delta > 0$, the following statements hold:*

- *Quasi-polynomially secure (resp non-uniformly quasi-polynomially secure) OWFs exist iff there exists a constant $c > 0$, $s(n) = 2^{c\sqrt{\log n}}$, such that $\mathsf{MK}^t\mathsf{P}[s]$ is $(n^\delta, \frac{1}{n^\beta})$-HoA* (resp non-uniformly HoA*).*
- *Subexponentially secure (resp non-uniformly subexponentially secure) OWFs exist iff there exists a constant $c > 0$, $s(n) = \log^c(n)$, such that $\mathsf{MK}^t\mathsf{P}[s]$ is $(n^\delta, \frac{1}{n^\beta})$-HoA* (resp non-uniformly HoA*).*

2.4 Pseudorandom Generators and Pseudorandom Functions

We recall the standard definitions of pseudorandom generators (PRGs).

Definition 8. *Let* $g : 1^\lambda \times \{0,1\}^{d(\lambda)} \to \{0,1\}^{m(\lambda)}$ *be a function.* g *is said to be a* $(T(\cdot), \varepsilon(\cdot))$-pseudorandom generator ((T,ε)-PRG) *(resp non-uniformly secure* (T,ε)*-PRG) if for any probabilistic* $T(\cdot)$*-time (resp non-uniform* $T(\cdot)$*-time) algorithm* $\mathcal{A}$ *(whose running time is* $T(\cdot)$ *in the length of its first input), for all sufficiently large* λ*,*

$$|\Pr[x \leftarrow \{0,1\}^{d(\lambda)} : \mathcal{A}(1^\lambda, g(x)) = 1] - \Pr[y \leftarrow \{0,1\}^{m(\lambda)} : \mathcal{A}(1^\lambda, y) = 1]| \leq \varepsilon(\lambda).$$

In addition, we say that g *is* locally-computable *if each bit of the output of* g *can be computed in time* $\mathsf{poly}(d(\lambda), \log m(\lambda))$*.*

We note that to simplify notations, we relax the efficiency requirement on g, since we also consider locally computable PRGs that output quasi-polynomially (or sub-exponentially) many bits (where it will be guaranteed that each bit of the output can be computed efficiently) (and such functions are inherently inefficient due to its output length).

We turn to introducing pseudorandom functions.

Definition 9. *Let* $f : 1^\lambda \times \{0,1\}^{d(\lambda)} \times \{0,1\}^{k(\lambda)} \to \{0,1\}$ *be a polynomial-time computable function.* f *is said to be a* $(T(\cdot), \varepsilon(\cdot))$-pseudorandom function (T,ε)-PRF *(resp non-uniformly secure* (T,ε)*-PRF) if for any probabilistic* $T(\cdot)$*-time (resp non-uniform* T*-time) algorithm* $\mathcal{A}$*, for all sufficiently large* n*,*

$$|\Pr[x \leftarrow \{0,1\}^{d(\lambda)} : \mathcal{A}^{f(x,\cdot)}(1^\lambda) = 1] - \Pr[f' \leftarrow \mathcal{F} : \mathcal{A}^{f'}(1^\lambda) = 1]| \leq \varepsilon(n)$$

where $\mathcal{F} = \{f' : \{0,1\}^{k(\lambda)} \to \{0,1\}\}$*.*

In addition, we say that f is *quasi-polynomially secure* if there exists a constant $c > 0$, $T(\lambda) = 2^{c \log^2 \lambda}$, f is a $(T, 1/T)$-PRF. We say that f is *subexponentially secure* if there exists a constant $\varepsilon > 0$, $T(\lambda) = 2^{\lambda^\varepsilon}$, f is a $(T, 1/T)$-PRF.

3 Weak Family of PRGs and Security Amplification

In this section, we introduce the notion of weak family of PRGs and prove a security amplification lemma for such PRGs.

Roughly speaking, a weak family of PRGs $\{g'_j(1^\lambda)\}_{j \in \{0,1\}^{d_1(\lambda)}}$ is a family of functions such that for any distinguisher D, there is at least some fraction of functions in the family whose output is pseudorandom. In addition, we say that the weak family has *running time* $t(\lambda)$ if there exists a function g such that $g(1^\lambda, j, y) = g'_j(1^\lambda, y)$ and $g(1^\lambda, j, y)$ runs in time $t(\lambda)$.

Definition 10. *We say that a family of functions* $g' : \{g'_j(1^\lambda) : \{0,1\}^{d_2(\lambda)} \to \{0,1\}^{m(\lambda)}\}_{j \in \{0,1\}^{d_1(\lambda)}}$ *is a* α-weak family of (T,ε)-pseudorandom generator (α-weak family of (T,ε)-PRGs) *if for all* $T(\lambda)$*-time distinguisher* D *(such that* D

runs in time $T(\lambda)$ when its first input is 1^λ), for all sufficiently large $n \in \mathbb{N}$, it holds that

$$\Pr[j \leftarrow \{0,1\}^{d_1(\lambda)} : \mathsf{Adv}_{D,\lambda}(g'_j(\mathcal{U}_{d_2(\lambda)}),\ \mathcal{U}_{m(\lambda)}) \leq \varepsilon(\lambda)] \geq \alpha(\lambda)$$

where for any random variables X, Y, $\mathsf{Adv}_{D,\lambda}(X,Y)$ is defined to be

$$\mathsf{Adv}_{D,\lambda}(X,Y) \stackrel{\text{def}}{=} |\Pr[r_1 \leftarrow X : D(1^\lambda, r_1) = 1] - \Pr[r_2 \leftarrow Y : D(1^\lambda, r_2) = 1]|$$

We refer to j as the index *and y as the* seed.

We say that g' is non-uniformly secure if the above holds for all non-uniform $T(\lambda)$-time distinguisher.

Notice that we will also consider PRGs whose output length is super polynomial in its seed length. Although such PRGs are inherently not efficiently computable, we will require them to be locally computable.

We show that weak family of PRGs can be amplified to obtain a full-fledged PRG (as long as PRGs in the weak family are expanding enough).

Lemma 11. *Let $\{g'_j(1^\lambda) : \{0,1\}^{d_2(\lambda)} \to \{0,1\}^{m(\lambda)}\}_{j \in \{0,1\}^{d_1(\lambda)}}$ be a α-weak family of (T,ε)-PRG with running time $t(\lambda)$. Let $\gamma(\lambda)$ be a (time-constructible) parallel repetition parameter, and let $h : 1^\lambda \times \{0,1\}^{(d_1(\lambda)+d_2(\lambda))\gamma(\lambda)} \to \{0,1\}^{m(\lambda)}$ be the xor of γ-fold repetition of g':*

$$h(j_1, y_1, \ldots, j_\gamma, y_\gamma) = g'_{j_1}(y_1) \oplus \ldots \oplus g'_{j_\gamma}(y_\gamma)$$

It holds that h is a $(T'(\lambda), \varepsilon'(\lambda))$-PRG where $T'(\lambda) = T(\lambda) - \gamma(\lambda)t(\lambda)$, $\varepsilon'(\lambda) = 2\max\{(1-\alpha(\lambda))^{\gamma(\lambda)}, \varepsilon(\lambda)\cdot\gamma(\lambda)\}$. In addition, the lemma also holds in the non-uniform setting.

We defer the proof of Lemma 11 to the full version due to space limit.

4 Unapproximability of Random Strings for Small Programs

It is well known that most random strings cannot be (exactly) produced by small programs (and thus have high Kolmogorov complexity). In this section, we show that most random strings cannot even be approximated with slightly non-trivial probability by small programs. We say that a program approximates a string with probability $\frac{1}{2}+\delta$ if over a random index i (and randomness of the program), the program on input i outputs the bit on the i-th coordinate of the string with probability $\frac{1}{2}+\delta$.

We are going to prove that most random strings of length n cannot be approximated with probability $\geq 1/2 + 1/n^\varepsilon$ by programs of size n^δ.

Lemma 12. *For any constants $\varepsilon, \delta > 0$, $\delta < 1 - 2\varepsilon$, there exists constants n_0 such that for all $n \geq n_0$, with probability at least*

$$1 - n^\varepsilon \cdot 2^{-n^{1-2\varepsilon}/2 + n^\delta + 2}$$

over a uniform random string $x \in \{0,1\}^n$, it holds that no (probabilistic) program Π (that always terminates on every random tape) of description length $\leq n^\delta$ can approximate the string x with probability $1/2 + 1/n^\varepsilon$; that is

$$\Pr[i \leftarrow [n] : \Pi(i) = x_i] \geq \frac{1}{2} + \frac{1}{n^\varepsilon}$$

where the probability is also taken over the internal randomness of Π.

Proof: Let Π_r denote the deterministic machine of Π with random tape fixed to be r. Since Π always terminates, let t_Π denote the running time of the program Π. Fix any such program $\Pi \in \{0,1\}^*$, $|\Pi| \leq n^\delta$, we claim that there is no more than a $n^\varepsilon 2^{-n^{1-2\varepsilon}/2+1}$ fraction of strings $x \in \{0,1\}^n$ such that Π approximates x with probability $\frac{1}{2} + \frac{1}{n^\varepsilon}$. Formally, we claim that

$$\Pr\left[x \leftarrow \{0,1\}^n : \Pr_{r \leftarrow \{0,1\}^{t_\Pi}, i \leftarrow [n]} [\Pi_r(i) = x_i] \geq \frac{1}{2} + \frac{1}{n^\varepsilon}\right] \leq n^\varepsilon 2^{-n^{1-2\varepsilon}/2+1} \quad (1)$$

If this holds, by taking a union bound over all Π, we have that

$$\Pr\Big[x \leftarrow \{0,1\}^n : \exists \Pi, |\Pi| \leq n^\delta, \Pr_{r \leftarrow \{0,1\}^{t_\Pi}, i \leftarrow [n]} [\Pi_r(i) = x_i] \geq \frac{1}{2} + \frac{1}{n^\varepsilon}\Big] \leq n^\varepsilon 2^{-n^{1-2\varepsilon}/2+n^\delta+2}$$

since (by a standard counting argument) there are at most $2^{n^\delta+1}$ such programs Π, which proves the lemma.

We proceed to proving Inequality 1. Notice that if Π approximates x with probability $\frac{1}{2} + \frac{1}{n^\varepsilon}$, by a standard averaging argument, there are at least a $\frac{1}{2n^\varepsilon}$ fraction of r such that the deterministic machine Π_r approximate x with probability $\frac{1}{2} + \frac{1}{2n^\varepsilon}$. Thus, the LHS of Inequality 1 is at most

$$\begin{aligned}
&\Pr\left[x \leftarrow \{0,1\}^n : \Pr_{r \leftarrow \{0,1\}^{t_\Pi}} \left[\Pr_{i \leftarrow [n]} [\Pi_r(i) = x_i] \geq \frac{1}{2} + \frac{1}{2n^\varepsilon}\right] \geq \frac{1}{2n^\varepsilon}\right] \\
\leq &2n^\varepsilon \Pr\left[x \leftarrow \{0,1\}^n, r \leftarrow \{0,1\}^{t_\Pi} : \Pr_{i \leftarrow [n]} [\Pi_r(i) = x_i] \geq \frac{1}{2} + \frac{1}{2n^\varepsilon}\right] \\
\leq &2n^\varepsilon \Pr\left[x \leftarrow \{0,1\}^n : \Pr_{i \leftarrow [n]} [\Pi_{r^*}(i) = x_i] \geq \frac{1}{2} + \frac{1}{2n^\varepsilon}\right] \quad (2)
\end{aligned}$$

where r^* is picked to be the random tape that maximizes the probability. Notice that the deterministic program Π_{r^*} can output only a single string. Let z_i be the random variable such that $z_i = 1$ iff $\Pi_{r^*}(i) = x_i$. Since x is a random string, the random variables z_i are independently distributed with mean $1/2$. It follows

that

$$\begin{aligned}&\Pr\left[x \leftarrow \{0,1\}^n : \Pr_{i\leftarrow[n]}[\Pi_{r^*}(i) = x_i] \geq \frac{1}{2} + \frac{1}{2n^\varepsilon}\right]\\ =&\Pr\left[x \leftarrow \{0,1\}^n : \sum_{i\in[n]} z_i \geq \frac{n}{2} + \frac{n}{2n^\varepsilon}\right]\\ \leq& e^{-n^{1-2\varepsilon}/2} \leq 2^{-n^{1-2\varepsilon}/2}\end{aligned}$$

where the (second) last step follows from the Chernoff bound. Plugging this into the Inequality 2, the claim follows. ■

5 PRF Construction from $\mathsf{MK}^t\mathsf{P}$

In this section, we show how to construct a PRF from hardness of $\mathsf{MK}^t\mathsf{P}[s]$.

5.1 Tools

Let us briefly introduce the technical tools needed in our construction. We start by recalling the construction of the Nisan-Wigderson (NW) PRG [41].

Definition 13 (NW generator [41]). *Let $\mathcal{I} = (I_1, \ldots, I_m)$ be a family of subsets of $[d]$ with each $|I_j| = \ell$ and let $f : \{0,1\}^\ell \to \{0,1\}$ be a function. The $(\mathcal{I}, f)$-NW generator is the function $\mathsf{NW}^f_{\mathcal{I}} : \{0,1\}^d \to \{0,1\}^m$ that takes any string $y \in \{0,1\}^d$ as a seed and outputs*

$$\mathsf{NW}^f_{\mathcal{I}}(y) = f(y_{I_1}) \ldots f(y_{I_m})$$

where for any set $I_j = \{i_1, \ldots, i_\ell\} \subseteq [d]$, any $y = y_1 y_2 \ldots y_d$, $y_{I_j} \stackrel{\text{def}}{=} y_{i_1} y_{i_2} \ldots y_{i_\ell}$ denotes the "projection" of string y onto coordinates in I_j.

The core ingredient of the Nisan-Wigderson construction is a combinatorial design which will be used as the family of subsets in a NW generator.

Definition 14 (Combinatorial designs [41]). *A family of sets $\mathcal{I} = \{I_1, \ldots, I_m \subseteq [d]\}$ is said to be a (d, ℓ, κ)-*design *if for every $i \in [m]$, $|I_i| = \ell$, and for every $j \in [m], j \neq i$, $|I_i \cap I_j| \leq \kappa$.*

We rely on the following explicit design generation algorithm.

Definition 15 (Explicit design generation [41]). *Let* $\mathsf{DesignNWGen}$ *be the following algorithms. On input $\ell, \kappa, i \in \mathbb{N}$ such that $\kappa < \ell, 1 \leq i \leq \ell^\kappa$, let q be the smallest power of 2 such that $q \geq \ell$. The algorithm* $\mathsf{DesignNWGen}$ *will output a set $I \subseteq [d]$ such that $|I| = \ell$, where $d = \ell q$. The algorithm proceeds as follows:*

- *Let $p(\cdot)$ be the i-th polynomial on $\mathbf{F}_q$ of degree κ (with respect to the canonical enumeration of all polynomials of degree κ on $\mathbf{F}_q$).*

– *Consider each number in* $[d] = [\ell \times q]$ *as a pair of numbers in* $[\ell] \times [q]$. *The set* I *will be defined to be*

$$I = \{(a, p(a)) \mid a \in [\ell]\}$$

where we abuse the notation and view a *as also a field element in* $\mathbf{F}_q$.

Recall that the above construction gives us a good combinatorial design.

Lemma 16 ([41]). *For any* $\ell, \kappa, m \in \mathbb{N}$, $m \leq \ell^\kappa$, *let* DesignNWGen *be the algorithm and* $d \in \mathbb{N}$ *as in Definition 15. Let* $\mathcal{I} = \{I_i = \mathsf{DesignNWGen}(\ell, \kappa, i)\}_{i \in [m]}$. *It holds that* $\mathcal{I}$ *is a* (d, ℓ, κ)*-design.*

The following version of the reconstruction theorem will be useful for us.

Lemma 17 (Implicit in [26,41], explicit in [32]; see also [49]). *There exists a* PPT *algorithm* NWRecon *such that the following conditions hold.*

– *Input: the truth table of a function* $f : \{0,1\}^\ell \to \{0,1\}$, *a* (d, ℓ, κ)*-design* $\mathcal{I} = \{I_1, \ldots, I_m\}$, *and a distinguishing gap parameter* $1^{\varepsilon^{-1}}$.
– *Given oracle access to a (probabilistic) oracle* $D \subseteq \{0,1\}^m$ *such that*

$$\left|\Pr[y \leftarrow \{0,1\}^d : D(\mathsf{NW}^f_{\mathcal{I}}(y)) = 1] - \Pr[w \leftarrow \{0,1\}^m : D(w) = 1]\right| \geq \varepsilon.$$

– *Output: a (deterministic) program* M *of description length* $\leq m \cdot 2^\kappa + m + d + O(\log d)$ *such that*

$$\Pr[p \leftarrow [2^\ell] : \Pi(p) = f(p)] \geq \frac{1}{2} + \frac{\varepsilon}{2m}$$

where $\Pi = M^D(\mathcal{I})$ *and the probability is also taken over the internal randomness of* D.

5.2 The PRF Construction

We present our PRF construction from the average-case* hardness of $\mathsf{MK}^t\mathsf{P}$. Consider any (monotonically increasing) polynomial $t(\cdot)$, $t(n) \geq (1+\varepsilon)n$, $\varepsilon > 0$.

Let λ be a program size parameter (which will also serve as a security parameter in the construction), $k = k(\lambda)$ denote the PRF input length parameter, and let $\gamma = \gamma(\lambda)$ be a parallel repetition parameter. To base the security of our construction on the $\alpha(\cdot)$-average-case* hardness of $\mathsf{MK}^t\mathsf{P}[s]$, we state the concrete choices of the parameters: We consider input length n such that $s(n) = \lambda$, and we pick $k = 1/8 \log n$, $\gamma = O(\log n / \alpha(\lambda))$.

Our goal is to construct a PRF with input domain $\{0,1\}^k$. (Towards this, we will be needing a Nisan-Widgerson PRG of output length $m = 2^k$.) The construction relies on the following ingredients. (We refer the reader to Sect. 5.1 for definitions of the technical tools used in this construction.)

- We will rely on the following decoding procedure Dec that maps any string $\Pi' \in \{0,1\}^*$ to a string $\Pi \in \{0,1\}^{<|\Pi'|}$. Π is obtained by removing all '0' in the end of Π' (if any), and then removing an additional '1'.
- We define a function f that receives a seed $\Pi' \in \{0,1\}^{\lambda+1}$ and an input $i \in \{0,1\}^{k'}$, computes $\Pi = \mathsf{Dec}(\Pi')$ and interprets it as a program, runs the program Π on input i for $t(|\Pi|)$ steps, and outputs what the program outputs, where k' is picked to be $8k$. Formally,

$$f(\Pi', i) = U(\Pi(i), 1^{t(|\Pi|)})$$

  where U is the universal Turing machine we fixed in the definition of K^t.
- We will be needing a combinatorial design to instantiate a Nisan-Wigderson PRG. Let $\ell = k'$, $\kappa = k$, $m = 2^k$. For each $i \in [2^k]$, let I_i be the set generated by $\mathsf{DesignNWGen}$ on input ℓ, κ, i. Let

$$\mathcal{I} = (I_1, I_2, \ldots, I_m)$$

  And note that $\mathcal{I}$ will be a (d, ℓ, κ)-design (by Lemma 16) where $d = \ell q$ and q is the smallest power of 2 such that $q \geq \ell$. Also notice that $\mathsf{DesignNWGen}$ on input ℓ, κ will be able to generate at most ℓ^κ sets, and we here need 2^k sets. Due to our choice of parameters, it holds that $\ell^\kappa \geq 2^\kappa = 2^k = m$.
- Define $g(\Pi', y)$ as the Nisan-Wigderson generator instantiated with the function $f(\Pi', \cdot)$ and the design $\mathcal{I}$. In more detail, for any seed $\Pi' \in \{0,1\}^{\lambda+1}$, $y \in \{0,1\}^d$, $g(\Pi', y)$ is defined as

$$g(\Pi', y) = \mathsf{NW}_{\mathcal{I}}^{f(\Pi', \cdot)}(y)$$

  And note that the output size is m. Moreover, we define a family of function $\{g'_{\Pi'} \stackrel{\text{def}}{=} g(\Pi', \cdot)\}_{\Pi' \in \{0,1\}^{\lambda+1}}$ where we simply view Π' as an index (that indices a NW generator) and y as the seed of $g'_{\Pi'}$.

Now we are ready to describe our PRF construction.

- The PRF construction h is defined as a function $h : \{0,1\}^{(\lambda+1+d)\gamma} \times \{0,1\}^k \to \{0,1\}$.
- h receives as the seed a string $z = (\Pi'_1, y_1, \ldots, \Pi'_\gamma, y_\gamma)$ of length $(\lambda + 1 + d)\gamma$, where for each $j \in [\gamma]$, Π'_j is of length $\lambda + 1$ and each y_j is of length d.
- On input $i \in \{0,1\}^k$, the output of h is defined to be

$$h(z, i) = g(\Pi'_1, y_1)[i] \oplus \ldots \oplus g(\Pi'_\gamma, y_\gamma)[i]$$

  where $g(\Pi'_j, y_j)[i]$ denote the i-th bit of $g(\Pi'_j, y_j)$, for any $j \in [\gamma]$.

The seed length of the construction is $(\lambda + 1 + d)\gamma \leq (\lambda + O(k^2)) \cdot \gamma$. We next analyze the running time of our construction. f_Π emulates the program Π for $t(|\Pi|)$ steps, so f_Π runs in time $O(t(|\Pi|)) \leq O(t(\lambda))$ (since t is monotonically increasing). To compute $g(\Pi', y)$, for each $i \in [2^k]$, we need to invoke $\mathsf{DesignNWGen}$ on input (ℓ, k, i), which runs in time $O(\ell^2) = O(k^2)$, and we also

need to compute $f(\Pi', y_{I_i})$, which runs in time $O(t(\lambda))$. Thus, $g(\Pi', y)$ takes $O(2^k(t(\lambda)+\ell^2))$.

In order to analyze the running time of h, we notice that $g(\Pi', y)$ can be computed *locally* – on each index $i \in [2^k]$, $g(\Pi', y)[i]$ can be computed in time $O(t(\lambda)+\ell^2)$. $h(z,i)$ will take the xor of γ independent instances of $g(\Pi', y)[i]$. Thus, the running time of h is $O((t(\lambda)+\ell^2)\gamma)$.

5.3 Security of the PRF Construction

Theorem 18. *Consider any (monotonically increasing) polynomial $t(n) \geq (1+\varepsilon)n$, $\varepsilon > 0$, any threshold function $s(n) = n^{o(1)}$, and its inverse $n_s(\cdot) = s^{-1}(\cdot)$, and any hardness parameter $\alpha(n) = \frac{1}{n^\beta}$, $\beta > 0$. Assume $\mathsf{MK}^t\mathsf{P}[s(n)]$ is (n^3, α)-HoA* (resp non-uniformly (n^3, α)-HoA*).*

Then, there exist constants $\gamma_0 > 0, \delta > 0$ such that for parameters $n' = n_s(\lambda)$, $k(\lambda) = 1/8 \log n'$, $\gamma(\lambda) = \gamma_0 \cdot \log n'/\alpha(\lambda)$, it holds that the function $h : 1^\lambda \times \{0,1\}^{(\lambda+1+2\log^2 n')\gamma} \times \{0,1\}^{k(\lambda)} \to \{0,1\}$ (constructed in Sect. 5.2) is a $(n'^\delta, \frac{1}{n'^\delta})$-PRF (resp non-uniformly secure $(n'^\delta, \frac{1}{n'^\delta})$-PRF).

Notice that there are two immediate corollaries of Theorem 18, by considering threshold $s(n) = 2^{O(\sqrt{\log n})}$ (from which we obtain Corollary 19, where $n' = 2^{\Omega(\log^2 \lambda)}$, and the assumption is equivalent to quasi-polynomially secure OWFs; the PRF domain is $\{0,1\}^{\Omega(\log^2 \lambda)}$) and threshold $s(n) = \mathsf{poly}\log(n)$ (from which we obtain Corollary 20, where $n' = 2^{\lambda^{1/c}}$, and the assumption is equivalent to subexponentially secure OWFs; the PRF domain is $\{0,1\}^{\lambda^{1/c}}$). The reader is referred to Sect. 2.3 for equivalence between OWFs and average-case hardness of $\mathsf{MK}^t\mathsf{P}[s]$.

Corollary 19 (PRF with input length $\Omega(\log^2 \lambda)$). *Consider a threshold function $s(n) = 2^{\delta\sqrt{\log n}}, \delta > 0$, polynomial $t(n) \geq (1+\varepsilon)n, \varepsilon > 0$, and any $\beta > 0$. Assume that $\mathsf{MK}^t\mathsf{P}[s]$ is $(n^3, \frac{1}{n^\beta})$-HoA* (resp. non-uniformly $(n^3, \frac{1}{n^\beta})$-HoA*).*

Then, the function $h : 1^\lambda \times \{0,1\}^{\tilde{O}(\lambda^{1+\beta})} \times \{0,1\}^{\log^2 \lambda/(8\delta)} \to \{0,1\}$ (constructed in Sect. 5.2) is a quasi-polynomially secure (resp. non-uniformly quasi-polynomially secure) PRF.

Corollary 20 (PRF with input length $\Omega(\lambda^{1/c})$). *Consider a threshold function $s(n) = \log^c n$, $c > 2$, polynomial $t(n) \geq (1+\varepsilon)n, \varepsilon > 0$, and any $\beta > 0$. Assume that $\mathsf{MK}^t\mathsf{P}[s]$ is $(n^3, \frac{1}{n^\beta})$-HoA* (resp. non-uniformly $(n^3, \frac{1}{n^\beta})$-HoA*).*

Then, the function $h : 1^\lambda \times \{0,1\}^{\tilde{O}(\lambda^{1+\beta+1/c})} \times \{0,1\}^{\lambda^{1/c}/8} \to \{0,1\}$ (constructed in Sect. 5.2) is a sub-exponentially secure (resp. non-uniformly sub-exponentially secure) PRF.

We proceed to proving the Theorem 18. In what follows, we consider any polynomial $t(n) \geq (1+\varepsilon)n$, $\varepsilon > 0$, any threshold function $s(\cdot)$ (with its inverse denoted by $n_s(\cdot) = s^{-1}(\cdot)$) such that $s(n) = n^{o(1)}$, and any hardness parameter $\alpha(n) = \frac{1}{n^\beta}$, $\beta > 0$. We will show that h is a PRF assuming hardness of $\mathsf{MK}^t\mathsf{P}[s]$.

Switching Distributions. Recall that the α-average-case* hardness of $\mathsf{MK}^t\mathsf{P}[s]$ considers hardness of $\mathsf{MK}^t\mathsf{P}[s]$ over the uniform distribution (conditioned on both YES and NO instances), whereas our PRF security game concerns the performance of the distinguisher over the pseudorandom function distribution vs. its performance over the random function distribution. We start by showing that our hardness assumption implies hardness of $\mathsf{MK}^t\mathsf{P}[s]$ over distributions that are easier to work with.

Let us first introduce the distributions that are needed in our proof. We will be needing the notion of (t,s)-universal distribution (ensemble) $\{D_{\mathsf{univ},n}\}_{n\in\mathbb{N}}$, defined as follows.[11] $D_{\mathsf{univ},n}$ will pick a uniform random string $\Pi \in \{\{0,1\}^{\leq s(n)} \cup \epsilon\}$[12], and interprets it as a program. $D_{\mathsf{univ},n}$ will output $x \in \{0,1\}^n$, where each x_i, $i \in [n]$, is the bit produced by running the program Π on input i after $t(|\Pi|)$ steps. Formally, $x_i = U(\Pi(i), 1^{t(|\Pi|)})$ where U is the universal Turing machine we consider. The other distribution we need is the uniform distribution.

We proceed to introducing the notion of average-case* hardness with respect to two distributions, $\mathcal{D}_Y$ and $\mathcal{D}_N$, defined similarly to Definition 5 but considering more general distributions. Roughly speaking, this requires no attacker can simultaneously output 1 with high probability over $\mathcal{D}_Y$ and output 0 over $\mathcal{D}_N$.

Definition 21 (Average-case* Hardness w.r.t. $\mathcal{D}_Y$ and $\mathcal{D}_N$). *We say that a $D(\cdot)$-dense language L is $\alpha(\cdot)$* hard-on-average* *for T-time attackers with respect to $\mathcal{D}_Y = \{D_{Y,n}\}_{n\in\mathbb{N}}$ and $\mathcal{D}_N = \{D_{N,n}\}_{n\in\mathbb{N}}$ ((T,α)-HoA* w.r.t. $\mathcal{D}_Y$ and $\mathcal{D}_N$) (resp non-uniformly (T,α)-HoA* w.r.t. $\mathcal{D}_Y$ and $\mathcal{D}_N$) if for all probabilistic T-time (resp non-uniform T-time) heuristics $\mathcal{H}$, for all sufficiently large n, it holds that either*

$$\Pr[x \leftarrow D_{Y,n} : \mathcal{H}(x) = 1] < 1 - \alpha(n^*),$$

or

$$\Pr[x \leftarrow D_{N,n} : \mathcal{H}(x) = 0] < 1 - \alpha(n^*).$$

where $n^* = \log D(n)$. n^* is referred to as the normalized input length.

We will show that average-case* hardness of $\mathsf{MK}^t\mathsf{P}[s]$ over uniform distribution implies average-case* hardness of $\mathsf{MK}^t\mathsf{P}[s]$ over the distributions that we are interested in.

Lemma 22. *There exists a constant $c' > 0$ such that for any threshold function $s(n) \leq n/10$, any polynomial $t(n) \geq (1+\varepsilon)n$, $\varepsilon > 0$, any hardness parameter $\alpha(\lambda) = \frac{1}{\lambda^\beta}$, $\beta > 0$, the following holds.*

Assume that $\mathsf{MK}^t\mathsf{P}[s]$ is (T,α)-HoA*. Then, $\mathsf{MK}^t\mathsf{P}[s]$ is (T,α')-HoA* w.r.t. the (t,s)-universal distribution and the uniform distribution where $\alpha' = \alpha/c'$. Moreover, the lemma also holds in the non-uniform setting.

[11] Note that there are many ways of defining the universal distribution, and we here consider the definition that is most relevant to us.

[12] For technical reason, we also consider the empty string ϵ.

Proof: Let $c' = 2^{c+1}$ be a constant where c is the constant as in Fact 4. Let $\alpha'(n) = \alpha(n)/c'$. For any input length n of $\mathsf{MK}^t\mathsf{P}[s]$, let n^* be the "normalized input length". Assume for contradiction that there exists a T-time heuristic* H such that H breaks the $(T, \alpha/c')$-HoA* of $\mathsf{MK}^t\mathsf{P}[s]$ w.r.t. the universal distribution and the uniform distribution; that is, for infinitely many $n \in \mathbb{N}$, the following two conditions hold simultaneously: (1) H will output 1 with probability at least $1 - \alpha'(n^*)$ over $D_{\mathsf{univ},n}$ and (2) H will output 0 with probability at least $1 - \alpha'(n^*)$ over $\mathcal{U}_n$. We will show that the attacker H will also break the (T, α) average-case* hardness of $\mathsf{MK}^t\mathsf{P}[s]$. Fix some sufficiently large $n \in \mathbb{N}$ such that our heuristic* H succeeds on inputs of length n.

We first show that H will output 1 with probability at least $1 - \alpha(n^*)$ over a uniform random YES instance $\in \mathsf{MK}^t\mathsf{P}[s]$. Suppose not, and we have that H fails (to output 1) with probability $> \alpha(n^*)$. By Fact 4, there are at least $2^{s(n)-c}$ YES instances in $\mathsf{MK}^t\mathsf{P}[s]$. Therefore, for any $x \in \mathsf{MK}^t\mathsf{P}[s] \cap \{0,1\}^n$, with probability at most

$$\frac{1}{2^{s(n)-c}}$$

a random YES instance will hit x. On the other hand, since $x \in \mathsf{MK}^t\mathsf{P}[s]$, there exists a program Π, $|\Pi| \leq s(n)$, such that on input i, Π will output x_i within time $t(|\Pi|)$, for each $i \in [n]$. It follows that x will be sampled with probability at least

$$\frac{1}{2^{s(n)+1}}$$

in the universal distribution (since the universal distribution will pick a random program of length $\leq s(n)$ and there are $2^{s(n)+1}$ such programs (including the empty string); x will be sampled from the distribution when the program Π is picked). Thus, H must also fail over the universal distribution with probability at least

$$\begin{aligned}
&\Pr[x \leftarrow D_{\mathsf{univ},n} : H(x) \neq 1] \\
=&\sum_{x \in \mathsf{MK}^t\mathsf{P}[s]} \Pr[D_{\mathsf{univ},n} = x] \cdot \Pr[H(x) \neq 1] \\
\geq&\sum_{x \in \mathsf{MK}^t\mathsf{P}[s]} \frac{1}{2^{c+1}} \frac{1}{2^{s(n)-c}} \Pr[H(x) \neq 1] \\
\geq&\sum_{x \in \mathsf{MK}^t\mathsf{P}[s]} \frac{1}{2^{c+1}} \Pr[x' \leftarrow \{0,1\}^n \cap \mathsf{MK}^t\mathsf{P}[s] : x' = x] \cdot \Pr[H(x) \neq 1] \\
>&\frac{1}{2^{c+1}} \alpha(n^*) \\
=&\alpha'(n^*)
\end{aligned}$$

which contradicts to the condition (1).

We turn to proving that, on input a random NO instance of $\mathsf{MK}^t\mathsf{P}[s]$, H will output 0 with probability at least $1 - \alpha(n^*)$. This follows from the fact that the

random NO distribution is statistically close the uniform distribution. In more detail, let

$$Z_1 = \mathcal{U}_n$$

be the uniform distribution over n-bit strings. And let

$$Z_2 = \{x \leftarrow \{0,1\}^n : x \notin \mathsf{MK}^t\mathsf{P}[s]\}$$

be the distribution of a random NO instance. Recall that by condition (2), the probability that H outputs 1 over Z_1 is at least

$$\Pr[x \leftarrow Z_1 : H(x) = 1] \leq \frac{1}{\alpha'(n^*)} = \alpha(n^*)/c'$$

We then show that statistical distance between Z_1 and Z_2 is at most $2^{-n+s(n)+1}$. By Fact 4, there are at most $2^{s(n)+1}$ n-bit strings that are YES-instances of $\mathsf{MK}^t\mathsf{P}[s]$, thus there are at most $2^{s(n)+1}$ points that have higher probability mass in Z_1 than in Z_2, and the difference in probability mass for each such point is exactly 2^{-n}. By the observation noted after the definition of statistical distance[13], it follows that the statistical distance is upper bounded by

$$\frac{1}{2^{n-s(n)-1}} \leq \frac{1}{2^{n^*+1}} \leq \frac{\alpha(n^*)}{2}.$$

(The first inequality holds since (1) we are only considering threshold functions such that $s(n) \leq n/10$, and thus $n - s(n) + 1 \geq n - n/10 + 1$; (2) recall that $n^* \leq s(n) + 1$ by Claim 1. The second inequality holds since $\alpha(n) = \frac{1}{n^\beta}$ for any constant $\beta > 0$.) Thus, the probability that H outputs 1 over Z_2 is at most

$$\begin{aligned}
&\Pr[x \leftarrow Z_2 : H(x) = 1] \\
\leq &\Pr[x \leftarrow Z_1 : H(x) = 1] + \mathsf{SD}(Z_1, Z_2) \\
\leq &\alpha(n^*)/c' + \alpha(n^*)/2 < \alpha(n^*)
\end{aligned}$$

Finally, notice that there are infinitely many such input lengths n, and we conclude that H also breaks the (T, α) average-case* hardness of $\mathsf{MK}^t\mathsf{P}[s]$.

Note that our proof only makes black-box use of the heuristic* H, and we conclude that it also holds in the non-uniform setting. ∎

Constructing a Weak Family of PRGs. We turn to showing that the function $g(\Pi', y)$ (specifically, the family g') we construct in Sect. 5.2 will be a weak family of PRGs.

Let us briefly recall the construction and introduce the parameters. We consider any polynomial $t(n) \geq (1+\varepsilon)n$, $\varepsilon > 0$. Our security parameter is denoted by λ. (We will base the security of our construction on the hardness of $\mathsf{MK}^t\mathsf{P}[s]$ with respect to input length n such that $n = n_s(\lambda)$.) We will consider a PRF input length parameter k satisfying $k = k(\lambda) = 1/8 \log n_s(\lambda)$. Let f be the function, g' be the family, $k' = k'(\lambda), \ell = \ell(\lambda), d = d(\lambda), m = m(\lambda)$ be the parameters as defined in Sect. 5.2. We are going to show that g' is a weak family of PRGs.

[13] That is, that the optimal distinguisher is $T = \{\omega : \Pr[Z_1 = \omega] > \Pr[Z_2 = \omega]\}$.

Lemma 23 *Let $t, s, s_n, k, k', \ell, d, m, f, g$ be as above, $s(n) \leq n/2$. Let $\alpha'(\lambda) = \frac{1}{c'\lambda^\beta}$ be a hardness parameter (for some constant $\beta > 0, c' > 0$).*

Assume $\mathsf{MK}^t\mathsf{P}[s]$ is (n^3, α')-HoA w.r.t. the (t, s)-universal distribution and the uniform distribution. Then, $\{g'_{\Pi'}(1^\lambda) : \{0,1\}^{d(\lambda)} \to \{0,1\}^{m(\lambda)}\}_{\Pi' \in \{0,1\}^{\lambda+1}}$ is a $(\alpha'(\lambda)/4)$-weak family of $(T'(\lambda), \varepsilon'(\lambda))$-PRGs where $T'(\lambda) = 2^{2k(\lambda)}$, $\varepsilon'(\lambda) = \frac{1}{2^{k(\lambda)}}$.*

In addition, the lemma also holds in the non-uniform setting.

Proof: We suppose for contradiction that there exists a distinguisher D that breaks the weak family of PRGs g'. Since D is a good distinguisher, it follows that there exist infinitely many $\lambda \in \mathbb{N}$ such that for at least a $1 - \alpha'(\lambda)/4$ fraction of its index $\Pi' \in \{0,1\}^{\lambda+1}$,

$$|\Pr[D(1^\lambda, g'_{\Pi'}(\mathcal{U}_d)) = 1] - \Pr[D(1^\lambda, \mathcal{U}_m) = 1]| \geq \varepsilon'(\lambda)$$

We will use this distinguisher D to construct an heuristic* H breaking the average-case* hardness of $\mathsf{MK}^t\mathsf{P}[s]$.

Our heuristic* H, on input a string $x \in \{0,1\}^n$, will proceed as follows.

- H first computes the security parameter $\lambda = s(n)$, the input length parameters $k = k(\lambda)$ and $k' = 8k$, and the other parameters needed in the construction of g.
- Let x' be the first $2^{k'}$ bits of x, and H will view x' as the truth table of a function $f_{x'}$ $\{0,1\}^{k'} \to \{0,1\}$.
- H will instantiate the NW generator NW with the function $f_{x'}$ (and the design $\mathcal{I}$ as in Sect. 5.2), and check whether the distinguisher D (on input the security parameter λ) will distinguish $\mathsf{NW}^{x'}_{\mathcal{I}}(\mathcal{U}_d)$ from random with advantage at least $\varepsilon'(\lambda)/2$.
- In more detail, let ρ_x denote the random variable $D(1^\lambda, \mathsf{NW}^{x'}_{\mathcal{I}}(\mathcal{U}_d))$ and θ denote the random variable $D(1^\lambda, \mathcal{U}_m)$. H will estimate $\mathbb{E}[\rho_x]$ by drawing $4\frac{1}{(\varepsilon'(\lambda)/8)^2}\log(\frac{1}{\alpha'(\lambda)/8})$ samples from ρ_x and take the average. Let ρ^* be the random variable of the average value. (Note that by the Hoeffding's Inequality, ρ^* is guaranteed to be (additively) $(\varepsilon'(\lambda)/8)$-close to $\mathbb{E}[\rho_x]$ with probability $\geq 1 - \alpha'(\lambda)/8$.) We repeat the above procedure to also estimate $\mathbb{E}[\theta]$ and denote the average by θ^*.
- Finally, H will output 1 if $|\rho^* - \theta^*| \geq \varepsilon'(\lambda)/2$.

We turn to analyzing the running time of H. Drawing a sample from ρ_x requires to run $\mathsf{NW}^{x'}_{\mathcal{I}}(\mathcal{U}_d)$. Using the same analysis as in Sect. 5.2 for $g(\Pi', y)$, we conclude that $\mathsf{NW}^{x'}_{\mathcal{I}}(\mathcal{U}_d)$ runs in time $O(2^k \cdot (n + \ell^2)) \leq O(n^2)$. In addition, we need to evaluate the distinguisher D, which takes time $T'(\lambda)$. Note that drawing a sample from θ takes at most as much time as sampling from ρ_x, we conclude that H runs in time $4\frac{1}{(\varepsilon'(\lambda)/8)^2}\log(\frac{1}{\alpha'(\lambda)/8}) \cdot 2(O(n^2) + T'(\lambda)) \leq O(2^{2k}\log\frac{1}{\alpha'(\lambda)}) \cdot O(2^{2k} + n^2) \leq n^3$.

We proceed to showing that H is a good heuristic*: H will output 1 with probability at least $1 - \alpha'(n^*)$ over the distribution for YES instances, H will

output 0 also with probability at least $1 - \alpha'(n^*)$ over the distribution for NO instances, and this holds for infinitely many input lengths n (where n^* is the "normalized" input length as in Definition 21). Fix some sufficiently large security parameter λ on which the distinguisher D breaks the weak family of PRGs g', and consider an input length n such that $n = n_s(\lambda)$.

We first analyze how our algorithm performs over the (t, s)-universal distribution $D_{\mathsf{univ},n}$ on input length n. It is helpful here to introduce a new notation $\mathsf{tt}(\cdot)$: For any binary function f, let $\mathsf{tt}(f)$ denote its truth table, and let $\mathsf{tt}_n(f)$ denote the n-bit prefix of the truth table. Consider the following two distribution:

- $\{x' = [x]_{2^{k'}} : x \leftarrow D_{\mathsf{univ},n}\}$, and
- $\{\mathsf{tt}_{2^{k'}}(f(\Pi', \cdot)) : \Pi' \leftarrow \{0,1\}^{\lambda+1}\}$

where $[x]_{2^{k'}}$ denotes the $(2^{k'})$-bit of x. Observe that (1) the above two distributions are identically distributed, (2) $g'_{\Pi'}(\mathcal{U}_d)$ will be identical to $\mathsf{NW}^{x'}_{\mathcal{I}}(\mathcal{U}_d)$ as long as $\mathsf{tt}_{2^{k'}}(f(\Pi, \cdot)) = x'$, and (3) over a random program Π' sampled from the second distribution, with probability at least $1 - \alpha'(\lambda)/4$, the distinguisher will distinguish $g'_{\Pi'}(\mathcal{U}_d)$ from random with advantage at least $\varepsilon'(\lambda)$ (which follows from the fact that D is a good distinguisher breaking g'). We conclude that (3) will still hold if we replace $f(\Pi', \cdot)$ by $f_{x'}$, and thus with probability at least $1 - \alpha'(\lambda)/4$ over $x \leftarrow D_{\mathsf{univ},n}$, D will distinguish $\mathsf{NW}^{x'}_{\mathcal{I}}(\mathcal{U}_d)$ from random and it holds that

$$|\mathsf{E}[\rho_x] - \mathsf{E}[\theta]| \geq \varepsilon'(\lambda) \tag{3}$$

Recall that ρ^* (resp θ^*) is our empirical estimation of $E[\rho_x]$ (resp $\mathsf{E}[\theta]$). As argued before, using Hoeffding's Inequality (and taking a Union Bound), we can show that except for probability $\alpha'(\lambda)/4$, the two estimations will be close to their expectations with difference $\leq \varepsilon'(\lambda)/8$. If so, it follows from Eq. 3 that the difference between their estimations should be at least

$$|\rho^* - \theta^*| \geq |\mathsf{E}[\rho_x] - \mathsf{E}[\theta]| - \varepsilon'(\lambda)/4 \geq \varepsilon'(\lambda)/2 \tag{4}$$

By applying the Union Bound again (taking into account that Eq. 3 holds with high probability and our estimations are accurate with high probability), it follows Eq. 4 holds with probability at least

$$1 - \alpha'(\lambda)/4 - \alpha'(\lambda)/4 = 1 - \frac{1}{2\lambda^{\beta}c'} \geq 1 - \frac{1}{(n^*)^{\beta}c'} = 1 - \alpha(n^*)$$

where the first inequality holds since recall that $\alpha'(\lambda) = \frac{1}{\lambda^{\beta}c'}$ for some constants β, c', and the second inequality holds since, by Claim 1, $\lambda = s(n) \leq n^* - 1$. Finally, note that if Eq. 4 holds, our algorithm H will output 1, which concludes that H outputs 1 with probability $\geq 1 - \alpha(n^*)$ over the YES distribution.

We move on to proving that our algorithm H will output 0 with probability at least $1 - \alpha'(n^*)$ over the uniform distribution over $x \in \{0,1\}^n$. We refer to a string $x \in \{0,1\}^n$ as being *bad* if our distinguisher D distinguishes $\mathsf{NW}^{x'}_{\mathcal{I}}(\mathcal{U}_d)$ (where x' is the first $2^{k'}$ bits of x) from random with at least $\varepsilon'(\lambda)/4$; that is,

$$|\mathsf{E}[\rho_x] - \mathsf{E}[\theta]| \geq \varepsilon'(\lambda)/4 \tag{5}$$

Notice that if a string x is *not* bad, using the same Chernoff/Hoeffding-type argument we did for YES instances, it follows that $H(x)$ will output 0 with probability at least $1 - \alpha'(\lambda)/4 = 1 - \alpha'(s(n))/4 \geq 1 - \alpha'(n^*)/2$, as desired. Thus, we will show that the faction of bad strings over $\{0,1\}^n$ is very small. We consider any bad string $x \in \{0,1\}^n$. It follows from Eq. 5 that $D(1^\lambda)$ will distinguish $\mathsf{NW}_{\mathcal{I}}^{x'}(\mathcal{U}_d)$ from random with advantage

$$|\Pr[D(1^\lambda, \mathsf{NW}_{\mathcal{I}}^{x'}(\mathcal{U}_d)) = 1] - \Pr[D(1^\lambda, \mathcal{U}_m) = 1]| \geq \varepsilon'(\lambda)/4$$

Recall from Lemma 16 that $\mathcal{I}$ is a (d, ℓ, κ)-design. By Lemma 17, the Nisan-Wigderson reconstruction algorithm $\mathsf{NWRecon}^{D(1^\lambda,\cdot)}(x')$ will output (with high probability) a program M that given oracle access to D approximates the function $f_{x'}$ where $\mathsf{tt}(f_{x'}) = x'$ (and therefore, approximates the prefix of the string x). In more detail, the program M is of description length $\leq m2^\kappa + m + d + O(\log d)$, and the oracle-aided program $M' = M^{D(1^\lambda,\cdot)}(\mathcal{I})$ will satisfy that

$$\begin{aligned}
&\Pr[p \leftarrow [2^{k'}] : M'(p) = x_p] \\
&= \Pr[p \leftarrow [2^{k'}] : M'(p) = f_{x'}(p)] \\
&\geq \frac{1}{2} + \frac{\varepsilon'(\lambda)}{8\,m} = \frac{1}{2} + \frac{1}{8 \cdot 2^k \cdot 2^k} \geq \frac{1}{2} + \frac{1}{2^{(5/16)k'}}
\end{aligned}$$

where the last inequality holds when $k' = 8k, k = 1/8 \log n$ is sufficiently large. We will further argue that M' has a small description length: Consider an implementation of M' with the program M, the code of D, parameters λ, ℓ, k hardwired in it. It first invokes the design generation algorithm to generate the design $\mathcal{I}$. It will then simulate $M^{D(1^\lambda,\cdot)}(\mathcal{I})$ and will output whatever M outputs. Notice that hardwiring the code of D takes either $O(1)$ bits (when D is a uniform attacker), or $O(2^{2k})$ bits (when D is a non-uniform attacker since D runs in time 2^{2k}), and storing the parameters takes $O(\log \lambda) + O(\log d)$ bits. So when k is sufficiently large, the description length of M' is at most

$$\begin{aligned}
&m2^\kappa + m + d + O(2^{2k}) + O(\log d) + O(\log \lambda) \\
&= 2^k \cdot 2^k + 2^k + O(k^2) + O(2^{2k}) + O(\log k^2) + O(\log s(n)) \\
&\leq O(2^{2k}) + 2^k + O(k^2) \leq 2^{(5/16)k'}
\end{aligned}$$

due to our choice of parameters (where $d = O(\ell^2) = O(k^2)$ and $k' = 8k$). Thus, we conclude that for any bad $x \in \{0,1\}^n$, its $(2^{k'})$-bit prefix can be approximated by a program (i.e., M') of description length $\leq 2^{(5/16)k'}$ with probability at least $\frac{1}{2} + \frac{1}{2^{(5/16)k'}}$. By Lemma 12, a random string x is bad with probability at most

$$\begin{aligned}
&2^{(5/16)k'} \cdot \exp(-2^{(1-2(5/16))k'}/2 + 2^{(5/16)k'} + 2) \\
&= 2^{(5/16)k'} \cdot \exp(-2^{(6/16)k'}/2 + 2^{(5/16)k'} + 2) \\
&= n^{5/16} \cdot \exp(-n^{6/16}/2 + n^{5/16} + 2) \\
&\leq 2^{-n^{5/16}} \leq \alpha'(n^*)/2
\end{aligned}$$

where $\exp(\cdot)$ denotes $2^{(\cdot)}$ and the last inequality holds when n is sufficiently large since $n^* \leq n$ and $\alpha'(n^*) = \frac{1}{(n^*)^\beta c'}$ for some constants $\beta, c' > 0$. Taken this together with the fact that H will output 0 with probability at least $1-\alpha'(n^*)/2$ when the input string x is not bad, we conclude that $H(x)$ outputs 0 over the uniform distribution with probability at least $1-\alpha'(n^*)$, which finishes the proof for the NO instances. ■

Remark 1 (A note on non-black box nature of the reduction). We remark that the proof of Lemma 23 implicitly defines a reduction R that breaks the average-case* hardness of $\mathsf{MK}^t\mathsf{P}[s]$ given any *"efficient"* machine D that breaks the weak family of PRGs. Although the reduction only accesses D as a black-box, the reduction is actually *non-black box* because in the analysis of the reduction, we are relying on the fact that D has a relatively short description—this is instrumental on argue that we succeed on NO instances (where D is used to approximately compress the instance x).

Amplifying Weak Families of PRGs. We proceed to proving that the construction h in Sect. 5.2 will be a PRF assuming that g' is a weak family of PRGs. In Sect. 3, we have shown that weak families of PRGs can be amplified by taking the xor of the independent outputs. Notice that if we consider the function h' that takes as input a seed z and outputs the truth table of $h(z,\cdot)$, this function is the xor of the function g'. By Lemma 11, we conclude that h' is a PRG, and it follows that h will be a PRF since each bit on the truth table of $h(z,\cdot)$ can be computed explicitly (as argued in Sect. 5.2).

Returning to Proving Theorem 18. We here present a formal proof of Theorem 18.

Proof: [of Theorem 18] Let t, s, n_s, k, γ as in the theorem statement. Let c' be the constant as in Lemma 22. We first show that h will be a PRF with desired security if we assume $\mathsf{MK}^t\mathsf{P}[s]$ is (n^3, α)-HoA*, and we will argue that this proof implicitly defines a security reduction we need.

We pick the constant γ_0 to be $4c'$ and the constant $\delta = 1/16$. It follows (from Lemma 22) that $\mathsf{MK}^t\mathsf{P}[s]$ is $(n^3, \alpha/c')$-HoA* w.r.t. the (t,s)-universal distribution and the uniform distribution. Let $n' = n_s(\lambda)$. Then by Lemma 23, g' is a $(\alpha(\lambda)/(4c'))$-weak family of $(T'(\lambda), \varepsilon'(\lambda))$-PRG where $T'(\lambda) = 2^{2k} = n'^{1/4}$ and $\varepsilon'(\lambda) = \frac{1}{2^k} = \frac{1}{n'^{1/8}}$. Recall that g' runs in time $t'(\lambda) \stackrel{\text{def}}{=} 2^k(t(\lambda) + O(\ell^2)) = O(2^k t(\lambda))$.

We will rely on Lemma 11 to show that h is a PRF. However, Lemma 11 is only stated with respect to PRGs (instead of PRFs). As mentioned before, it suffices to show that h, being viewed as a PRG (by considering the function outputting the truth table of h on each seed as a pseudorandom string), is a PRG. If so, it follows that h will be a PRF since the pseudorandom string can be computed locally (as argued in Sect. 5.2). By Lemma 11, we have that h is a $(T''(\lambda), \varepsilon''(\lambda))$-PRG (when being viewed as a PRG) where $T''(\lambda) = T'(\lambda) - \gamma t'(\lambda)$

and $\varepsilon''(\lambda) = 2\max\{(1-\alpha(\lambda)/(4c'))^{\gamma}, \gamma\varepsilon'(\lambda)\}$. Notice that

$$T''(\lambda) \geq n'^{1/4} - \gamma O(2^k t(\lambda)) \geq n'^{1/4} - \gamma_0 \log n'/\alpha(\lambda) \cdot O(2^k t(\lambda)) \geq n'^{\delta}$$

since (1) we only consider $\alpha(\lambda) = 1/\lambda^{\beta}$, $\beta > 0$, and $\lambda = s(n') = n'^{o(1)}$ (taken together, this implies that $1/\alpha(\lambda) = n'^{o(1)}$), and (2) $t(\lambda) = n'^{o(1)}$ since t is a polynomial. We turn to prove that $\varepsilon''(\lambda)$ is also small. $\varepsilon''(\lambda)$ is the maximal of the two values, and we will prove that each of the values will be upper bounded by $\frac{1}{n'^{1/16}} = \frac{1}{n'^{\delta}}$. Observe that on one hand,

$$2(1-\alpha(\lambda)/(4c'))^{\gamma} = 2(1-\alpha(\lambda)/(4c'))^{4c'/\alpha(\lambda)\cdot\log n'} \leq 2(1/e)^{\log n'} \leq \frac{1}{n'^{1/16}}$$

And on the other hand,

$$2\gamma\frac{1}{n'^{1/8}} = 2(4c'/\alpha(\lambda)\log n')\frac{1}{n'^{1/8}} \leq \frac{1}{n'^{1/16}}$$

since as argued above, $1/\alpha(\lambda) = n^{o(1)}$. This concludes that h is a $(n'^{\delta}, \frac{1}{n'^{\delta}})$-PRG.

Notice that our security proof (presented above, going through Lemma 22, Lemma 23, and Lemma 11) defines a security reduction that, given an attacker A that breaks the PRF h on security parameter 1^{λ}, breaks the hardness of $\mathsf{MK}^t\mathsf{P}[s]$ on input length $\lambda = s(n)$.

Also notice that the proof also works in the non-uniform setting since the lemmas needed in the proof all hold in the non-uniform setting. ∎

References

1. Allender, E.: When worlds collide: Derandomization, lower bounds, and kolmogorov complexity. In: International Conference on Foundations of Software Technology and Theoretical Computer Science, pp. 1–15. Springer (2001)
2. Allender, E., Buhrman, H., Koucký, M., Van Melkebeek, D., Ronneburger, D.: Power from random strings. SIAM J. Comput. **35**(6), 1467–1493 (2006)
3. Banerjee, A., Peikert, C., Rosen, A.: Pseudorandom functions and lattices. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques, pp. 719–737. Springer (2012)
4. Bogdanov, Andrej, Rosen, Alon: Pseudorandom functions: three decades later. In: Tutorials on the Foundations of Cryptography. ISC, pp. 79–158. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-57048-8_3
5. Canetti, R., Goldreich, O., Goldwasser, S., Micali, S.: Resettable zero-knowledge (extended abstract). In: STOC 2000, pp. 235–244 (2000). https://doi.org/10.1145/335305.335334
6. Carmosino, M.L., Impagliazzo, R., Kabanets, V., Kolokolova, A.: Learning algorithms from natural proofs. In: 31st Conference on Computational Complexity (CCC 2016). Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik (2016)
7. Chen, L., Hirahara, S., Oliveira, I.C., Pich, J., Rajgopal, N., Santhanam, R.: Beyond natural proofs: Hardness magnification and locality. In: 11th Innovations in Theoretical Computer Science Conference (ITCS 2020). Schloss Dagstuhl-Leibniz-Zentrum für Informatik (2020)

8. Chen, L., Jin, C., Williams, R.R.: Hardness magnification for all sparse np languages. In: 2019 IEEE 60th Annual Symposium on Foundations of Computer Science (FOCS), pp. 1240–1255. IEEE (2019)
9. Chen, L., McKay, D.M., Murray, C.D., Williams, R.R.: Relations and equivalences between circuit lower bounds and karp-lipton theorems. In: 34th Computational Complexity Conference (CCC 2019). Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik (2019)
10. Chen, L., Tell, R.: Bootstrapping results for threshold circuits "just beyond" known lower bounds. In: Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing, pp. 34–41 (2019)
11. Chen, L., Tell, R.: Hardness vs randomness, revised: uniform, non-black-box, and instance-wise. Electronic Colloquium on Computational Complexity (2021). https://eccc.weizmann.ac.il/report/2021/080/l
12. Dodis, Y., Impagliazzo, R., Jaiswal, R., Kabanets, V.: Security amplification for interactive cryptographic primitives. In: Theory of Cryptography: 6th Theory of Cryptography Conference, TCC 2009, San Francisco, CA, USA, March 15-17, 2009. Proceedings 6, pp. 128–145. Springer (2009)
13. Goldreich, O.: Foundations of Cryptography — Basic Tools. Cambridge University Press (2001)
14. Goldreich, O., Goldwasser, S., Micali, S.: How to construct random functions. In: FOCS (1984)
15. Goldreich, O., Goldwasser, S., Micali, S.: On the cryptographic applications of random functions. In: Advances in Cryptology: Proceedings of CRYPTO 84 4, pp. 276–288. Springer (1985)
16. Goldreich, O., Nisan, N., Wigderson, A.: On yao's xor lemma. Technical Report TR95–050, Electronic Colloquium on Computational Complexity (1995)
17. Goldreich, O., Ostrovsky, R.: Software protection and simulation on oblivious rams. J. ACM **43**(3), 431–473 (1996)
18. Goldwasser, S., Micali, S.: Probabilistic encryption. J. Comput. Syst. Sci. **28**(2), 270–299 (1984)
19. Haitner, I., Harnik, D., Reingold, O.: On the power of the randomized iterate. In: CRYPTO, pp. 22–40 (2006)
20. Haitner, I., Reingold, O., Vadhan, S.: Efficiency improvements in constructing pseudorandom generators from one-way functions. In: Proceedings of the Forty-Second ACM Symposium on Theory of Computing, pp. 437–446 (2010)
21. Hartman, T., Raz, R.: On the distribution of the number of roots of polynomials and explicit weak designs. Random Struct. Algorithms **23**(3), 235–263 (2003)
22. Hartmanis, J.: Generalized kolmogorov complexity and the structure of feasible computations. In: 24th Annual Symposium on Foundations of Computer Science (sfcs 1983). pp. 439–445, November 1983. https://doi.org/10.1109/SFCS.1983.21
23. Håstad, J., Impagliazzo, R., Levin, L.A., Luby, M.: A pseudorandom generator from any one-way function. SIAM J. Comput. **28**(4), 1364–1396 (1999)
24. Hirahara, S.: Non-black-box worst-case to average-case reductions within NP. In: 59th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2018, pp. 247–258 (2018)
25. Holenstein, T.: Pseudorandom generators from one-way functions: a simple construction for any hardness. In: TCC, pp. 443–461 (2006)
26. Impagliazzo, R., Wigderson, A.: $P = BPP$ if e requires exponential circuits: Derandomizing the xor lemma. In: STOC 1997, pp. 220–229 (1997)

27. Kabanets, V., Cai, J.: Circuit minimization problem. In: Proceedings of the Thirty-Second Annual ACM Symposium on Theory of Computing, May 21-23, 2000, Portland, OR, USA, pp. 73–79 (2000)
28. Ko, K.: On the notion of infinite pseudorandom sequences. Theor. Comput. Sci. **48**(3), 9–33 (1986)
29. Kolmogorov, A.N.: Three approaches to the quantitative definition of information. Int. J. Comput. Math. **2**(1–4), 157–168 (1968)
30. Liu, Y., Pass, R.: On one-way functions and Kolmogorov complexity. In: 61st IEEE Annual Symposium on Foundations of Computer Science, FOCS 2020, Durham, NC, USA, November 16-19, 2020, pp. 1243–1254. IEEE (2020)
31. Liu, Y., Pass, R.: Cryptography from sublinear time hardness of time-bounded kolmogorov complexity. In: STOC (2021)
32. Liu, Y., Pass, R.: Characterizing derandomization through hardness of levin-kolmogorov complexity. In CCC (2022)
33. Liu, Y., Pass, R.: On one-way functions and the worst-case hardness of time-bounded kolmogorov complexity. Cryptology ePrint Archive p. 1086 (2023)
34. Luby, M.G.: Pseudorandomness and cryptographic applications, vol. 1. Princeton University Press (1996)
35. Maurer, U., Tessaro, S.: Computational indistinguishability amplification: tight product theorems for system composition. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 355–373. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-03356-8_21
36. Mazor, N., Pass, R.: Counting unpredictable bits: A simple prg from one-way functions. Cryptology ePrint Archive (2023)
37. McKay, D.M., Murray, C.D., Williams, R.R.: Weak lower bounds on resource-bounded compression imply strong separations of complexity classes. In: Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing, pp. 1215–1225 (2019)
38. 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)
39. Naor, M., Reingold, O.: Number-theoretic constructions of efficient pseudo-random functions. J. ACM (JACM) **51**(2), 231–262 (2004)
40. 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)
41. Nisan, N., Wigderson, A.: Hardness vs randomness. J. Comput. Syst. Sci. **49**(2), 149–167 (1994)
42. Oliveira, I., Pich, J., Santhanam, R.: Hardness magnification near state-of-the-art lower bounds (2019)
43. Oliveira, I.C.: Randomness and intractability in kolmogorov complexity. In: 46th International Colloquium on Automata, Languages, and Programming (ICALP 2019). Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik (2019)
44. Oliveira, I.C., Santhanam, R.: Hardness magnification for natural problems. In: 2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS), pp. 65–76. IEEE (2018)
45. Razborov, A.A., Rudich, S.: Natural proofs. J. Comput. Syst. Sci. **55**(1), 24–35 (1997)
46. Sipser, M.: A complexity theoretic approach to randomness. In: Proceedings of the 15th Annual ACM Symposium on Theory of Computing, 25–27 April, 1983, Boston, Massachusetts, USA, pp. 330–335. ACM (1983)

47. Sudan, M., Trevisan, L., Vadhan, S.: Pseudorandom generators without the xor lemma. J. Comput. Syst. Sci. **62**(2), 236–266 (2001)
48. Trakhtenbrot, B.A.: A survey of Russian approaches to perebor (brute-force searches) algorithms. Annal. History Comput. **6**(4), 384–400 (1984)
49. Vadhan, S.P.: Pseudorandomness. Foundations and Trends® in Theoretical Comput. Sci. **7**(1–3), 1–336 (2012)
50. Vadhan, S.P., Zheng, C.J.: Characterizing pseudoentropy and simplifying pseudorandom generator constructions. In: STOC, pp. 817–836 (2012)
51. Valiant, L.G.: A theory of the learnable. Commun. ACM **27**(11), 1134–1142 (1984)
52. Yablonski, S.: The algorithmic difficulties of synthesizing minimal switching circuits. Problemy Kibernetiki **2**(1), 75–121 (1959)
53. Yablonski, S.V.: On the impossibility of eliminating perebor in solving some problems of circuit theory. Dokl. Akad. Nauk SSSR **124**(1), 44–47 (1959)
54. Yao, A.C.: Theory and applications of trapdoor functions (extended abstract). In: 23rd Annual Symposium on Foundations of Computer Science, Chicago, Illinois, USA, 3–5 November 1982, pp. 80–91 (1982)