From Non-adaptive to Adaptive Pseudorandom Functions

Itay Berman and Iftach Haitner[⋆]

School of Computer Science, Tel Aviv University
itayberm@post.tau.ac.il, iftachh@cs.tau.ac.il

Abstract. Unlike the standard notion of pseudorandom functions (PRF), a *non-adaptive* PRF is only required to be indistinguishable from random in the eyes of a *non-adaptive* distinguisher (i.e., one that prepares its oracle calls in advance). A recent line of research has studied the possibility of a *direct* construction of adaptive PRFs from non-adaptive ones, where direct means that the constructed adaptive PRF uses only few (ideally, constant number of) calls to the underlying non-adaptive PRF. Unfortunately, this study has only yielded negative results, showing that "natural" such constructions are unlikely to exist (e.g., Myers [EUROCRYPT '04], Pietrzak [CRYPTO '05, EUROCRYPT '06]).

We give an affirmative answer to the above question, presenting a direct construction of adaptive PRFs from non-adaptive ones. Our construction is extremely simple, a composition of the non-adaptive PRF with an appropriate pairwise independent hash function.

1 Introduction

A pseudorandom function family (PRF), introduced by Goldreich, Goldwasser, and Micali [11], cannot be distinguished from a family of *truly* random functions by an efficient distinguisher who is given an oracle access to a random member of the family. PRFs have an extremely important role in cryptography, allowing parties, which share a common secret key, to send secure messages, identify themselves and to authenticate messages [10, 13]. In addition, they have many other applications, essentially in any setting that requires random function provided as black-box [2, 3, 6, 7, 14, 18]. Different PRF constructions are known in the literature, whose security is based on different hardness assumption. Constructions relevant to this work are those based on the existence of pseudorandom generators [11] (and thus on the existence of one-way functions [12]), and on, the so called, synthesizers [17].

In this work we study the question of constructing (adaptive) PRFs from *non-adaptive* PRFs. The latter primitive is a (weaker) variant of the standard PRF we mentioned above, whose security is only guaranteed to hold against non-adaptive distinguishers (i.e., ones that "write" all their queries before the

[⋆] Research supported by Check Point Institute for Information Security and BSF grant 2010196.

first oracle call). Since a non-adaptive PRF can be easily cast as a pseudorandom generator or as a synthesizer, [11, 17] tell us how to construct (adaptive) PRF from a non-adaptive one. In both of these constructions, however, the resulting (adaptive) PRF makes $\Theta(n)$ calls to the underlying non-adaptive PRF (where n being the input length of the functions).[1]

A recent line of work has tried to figure out whether more efficient reductions from adaptive to non-adaptive PRF's are likely to exist. In a sequence of works [16, 19, 20, 5], it was shown that several "natural" approaches (e.g., composition or XORing members of the non-adaptive family with itself) are unlikely to work. See more in Section 1.3.

1.1 Our Result

We show that a simple composition of a non-adaptive PRF with an appropriate pairwise independent hash function, yields an adaptive PRF. To state our result more formally, we use the following definitions: a function family $\mathcal{F}$ is $T = T(n)$-adaptive PRF, if no distinguisher of running time at most T, can tell a random member of $\mathcal{F}$ from a random function with advantage larger than $1/T$. The family $\mathcal{F}$ is T-*non*-adaptive PRF, if the above is only guarantee to hold against non-adaptive distinguishers. Given two function families $\mathcal{F}_1$ and $\mathcal{F}_2$, we let $\mathcal{F}_1 \circ \mathcal{F}_2$ [resp., $\mathcal{F}_1 \bigoplus \mathcal{F}_2$] be the function family whose members are all pairs $(f, g) \in \mathcal{F}_1 \times \mathcal{F}_2$, and the action $(f, g)(x)$ is defined as $f(g(x))$ [resp., $f(x) \oplus g(x)$]. We prove the following statements (see Section 3 for the formal statements).

Theorem 1 (Informal). *Let $\mathcal{F}$ be a $(p(n) \cdot T(n))$-non-adaptive PRF, where $p \in \text{poly}$ is function of the evaluating time of $\mathcal{F}$, and let $\mathcal{H}$ be an efficient pairwise-independent function family mapping strings of length n to $[T(n)]_{\{0,1\}^n}$, where $[T]_{\{0,1\}^n}$ is the first T elements (in lexicographic order) of $\{0,1\}^n$. Then $\mathcal{F} \circ \mathcal{H}$ is a $\left(\sqrt[3]{T(n)}/2\right)$-adaptive PRF.*

For instance, assuming that $\mathcal{F}$ is a $(p(n) \cdot 2^{cn})$-non-adaptive PRF and that $\mathcal{H}$ maps strings of length n to $[2^{cn}]_{\{0,1\}^n}$, Theorem 1 yields that $\mathcal{F} \circ \mathcal{H}$ is a $\left(2^{\frac{cn}{3}-1}\right)$-adaptive PRF.

Theorem 1 is only useful, however, for polynomial-time computable T's (in this case, the family $\mathcal{H}$ assumed by the theorem exists, see Section 2.2). Unfortunately, in the important case where $\mathcal{F}$ is only assumed to be polynomially secure non-adaptive PRF, no useful polynomial-time computable T is guaranteed to exists.[2]

We suggest two different solutions for handling polynomially secure PRFs. In Section 4 we observe (following Bellare [1]) that a polynomially secure non-adaptive PRF is a T-non-adaptive PRF for some $T \in n^{\omega(1)}$. Since this T can

[1] We remark that if one is only interested in *polynomial security* (i.e., no adaptive PPT distinguishes with more than negligible probability), then $w(\log n)$ calls are sufficient (cf., [8, Sec. 3.8.4, Exe. 30]).

[2] Clearly $\mathcal{F}$ is p-non-adaptive PRF for any $p \in \text{poly}$, but applying Theorem 1 with $T \in \text{poly}$, does not yield a polynomially secure adaptive PRF.

be assumed without loss of generality to be a power of two, Theorem 1 yields a non-uniform (uses n-bit advice) polynomially secure adaptive PRF, that makes a single call to the underlying non-adaptive PRF. Our second solution is to use the following "combiner", to construct a (uniform) adaptively secure PRF, which makes $\omega(1)$ parallel calls to the underlying non-adaptive PRF.

Corollary 1 (Informal). *Let $\mathcal{F}$ be a polynomially secure non-adaptive PRF, let $\mathcal{H} = \{\mathcal{H}_n\}_{n\in\mathbb{N}}$ be an efficient pairwise-independent length-preserving function family and let $k(n) \in \omega(1)$ be polynomial-time computable function.*

For $n \in \mathbb{N}$ and $i \in [n]$, let $\widehat{\mathcal{H}_n}^i$ be the function family $\widehat{\mathcal{H}_n}^i = \{\widehat{h}\colon h \in \mathcal{H}\}$, where $\widehat{h}(x) = 0^{n-i}||h(x)_{1,\ldots,i}$ ('$||$' stands for string concatenation). Then the ensemble $\{\bigoplus_{i\in[k(n)]}\left(\mathcal{F}_n \circ \widehat{\mathcal{H}_n}^{\lfloor i\cdot\log n\rfloor}\right)\}_{n\in\mathbb{N}}$ is a polynomially secure adaptive PRF.

1.2 Proof Idea

To prove Theorem 1 we first show that $\mathcal{F}\circ\mathcal{H}$ is indistinguishable from $\Pi\circ\mathcal{H}$, where Π being the set of *all* functions from $\{0,1\}^n$ to $\{0,1\}^{\ell(n)}$ (letting $\ell(n)$ be $\mathcal{F}$'s output length), and then conclude the proof by showing that $\Pi\circ\mathcal{H}$ is indistinguishable from Π.

$\mathcal{F}\circ\mathcal{H}$ Is indistinguishable from $\Pi\circ\mathcal{H}$. Let D be (a possibly adaptive) algorithm of running time $T(n)$, which distinguishes $\mathcal{F}\circ\mathcal{H}$ from $\Pi\circ\mathcal{H}$ with advantage $\varepsilon(n)$. We use D to build a *non-adaptive* distinguisher $\widehat{\mathsf{D}}$ of running time $p(n)\cdot T(n)$, which distinguishes $\mathcal{F}$ from Π with advantage $\varepsilon(n)$. Given an oracle access to a function ϕ, the distinguisher $\widehat{\mathsf{D}}^{\phi}(1^n)$ first queries ϕ on *all* the elements of $[T(n)]_{\{0,1\}^n}$. Next it chooses at uniform $h\in\mathcal{H}$, and uses the stored answers to its queries, to emulate $\mathsf{D}^{\phi\circ h}(1^n)$.

Since $\widehat{\mathsf{D}}$ runs in time $p(n)\cdot T(n)$, for some large enough $p\in$ poly, makes *non-adaptive* queries, and distinguishes $\mathcal{F}$ from Π with advantage $\varepsilon(n)$, the assumed security of $\mathcal{F}$ yields that $\varepsilon(n) < \frac{1}{p(n)\cdot T(n)}$.

$\Pi\circ\mathcal{H}$ Is indistinguishable from Π. We prove that $\Pi\circ\mathcal{H}$ is *statistically* indistinguishable from Π. Namely, even an unbounded distinguisher (that makes bounded number of calls) cannot distinguish between the families. The idea of the proof is fairly simple. Let D be an s-query algorithm trying to distinguish between $\Pi\circ\mathcal{H}$ and Π. We first note that the distinguishing advantage of D is bounded by its probability of finding a collision in a random $\phi\in\Pi\circ\mathcal{H}$ (in case no collision occurs, ϕ's output is uniform). We next argue that in order to find a collision in ϕ, the distinguisher D gains nothing from being adaptive. Indeed, assuming that D found no collision until the i'th call, then it has only learned that h does not collide on these first i queries. Therefore, a random (or even a constant) query as the $(i+1)$ call, has the same chance to yield a collision, as any other query has. Hence, we assume without loss of generality that D is non-adaptive, and use the pairwise independence of $\mathcal{H}$ to conclude that D's probability in finding a collision, and thus its distinguishing advantage, is bounded by $s(n)^2/T(n)$.

Combining the above two observations, we conclude that an adaptive distinguisher whose running time is bounded by $\frac{1}{2}\sqrt[3]{T(n)}$, cannot distinguish $\mathcal{F} \circ \mathcal{H}$ from Π (i.e., from a random function) with an advantage better than $\frac{T(n)^{\frac{2}{3}}/4}{T(n)} + \frac{1}{p(n)T(n)} \leq 2/\sqrt[3]{T(n)}$. Namely, $\mathcal{F} \circ \mathcal{H}$ is a $\left(\sqrt[3]{T(n)}/2\right)$-adaptive PRF.

1.3 Related Work

Maurer and Pietrzak [15] were the first to consider the question of building adaptive PRFs from non-adaptive ones. They showed that in the *information theoretic* model, a self composition of a non-adaptive PRF *does* yield an adaptive PRF.[3]

In contrast, the situation in the *computational model* (which we consider here) seems very different: Myers [16] proved that it is impossible to reprove the result of [15] via fully-black-box reductions. Pietrzak [19] showed that under the Decisional Diffie-Hellman (DDH) assumption, composition does not imply adaptive security. Where in [20] he showed that the existence of non-adaptive PRFs whose composition is not adaptively secure, yields that key-agreement protocol exists. Finally, Cho et al. [5] generalized [20] by proving that composition of two non-adaptive PRFs is not adaptively secure, iff (uniform transcript) key agreement protocol exists. We mention that [16, 19, 5], and in a sense also [15], hold also with respect to XORing of the non-adaptive families.

2 Preliminaries

2.1 Notations

All logarithms considered here are in base two. We let '||' denote string concatenation. We use calligraphic letters to denote sets, uppercase for random variables, and lowercase for values. For an integer t, we let $[t] = \{1, \ldots, t\}$, and for a set $\mathcal{S} \subseteq \{0,1\}^*$ with $|\mathcal{S}| \geq t$, we let $[t]_{\mathcal{S}}$ be the first t elements (in increasing lexicographic order) of $\mathcal{S}$. A function $\mu \colon \mathbb{N} \to [0,1]$ is *negligible*, denoted $\mu(n) = \operatorname{neg}(n)$, if $\mu(n) = n^{-\omega(1)}$. We let poly denote the set all polynomials, and let PPT denote the set of probabilistic algorithms (i.e., Turing machines) that run in *strictly* polynomial time.

Given a random variable X, we write $X(x)$ to denote $\Pr[X = x]$, and write $x \leftarrow X$ to indicate that x is selected according to X. Similarly, given a finite set $\mathcal{S}$, we let $s \leftarrow \mathcal{S}$ denote that s is selected according to the uniform distribution on $\mathcal{S}$. The *statistical distance* of two distributions P and Q over a finite set $\mathcal{U}$, denoted as $\operatorname{SD}(P,Q)$, is defined as $\max_{\mathcal{S}\subseteq\mathcal{U}} |P(\mathcal{S}) - Q(\mathcal{S})| = \frac{1}{2}\sum_{u\in\mathcal{U}} |P(u) - Q(u)|$.

[3] Specifically, assuming that the non-adaptive PRF is (Q,ε)-non-adaptively secure, no Q-query non-adaptive algorithm distinguishes it from random with advantage larger than ε, then the resulting PRF is $(Q, \varepsilon(1 + \ln\frac{1}{\varepsilon}))$-adaptively secure.

2.2 Ensemble of Function Families

Let $\mathcal{F} = \{\mathcal{F}_n \colon \mathcal{D}_n \mapsto \mathcal{R}_n\}_{n\in\mathbb{N}}$ stands for an ensemble of function families, where each $f \in \mathcal{F}_n$ has domain $\mathcal{D}_n$ and its range contained in $\mathcal{R}_n$. Such ensemble is *length preserving*, if $\mathcal{D}_n = \mathcal{R}_n = \{0,1\}^n$ for every n.

Definition 1 (efficient function family ensembles). *A function family ensemble $\mathcal{F} = \{\mathcal{F}_n\}_{n\in\mathbb{N}}$ is* efficient*, if the following hold:*

Samplable. *$\mathcal{F}$ is samplable in polynomial-time: there exists a* PPT *that given 1^n, outputs (the description of) a uniform element in $\mathcal{F}_n$.*

Efficient. *There exists a polynomial-time algorithm that given $x \in \{0,1\}^n$ and (a description of) $f \in \mathcal{F}_n$, outputs $f(x)$.*

Operating on Function Families

Definition 2 (composition of function families). *Let $\mathcal{F}^1 = \{\mathcal{F}^1_n \colon \mathcal{D}^1_n \mapsto \mathcal{R}^1_n\}_{n\in\mathbb{N}}$ and $\mathcal{F}^2 = \{\mathcal{F}^2_n \colon \mathcal{D}^2_n \mapsto \mathcal{R}^2_n\}_{n\in\mathbb{N}}$ be two ensembles of function families with $\mathcal{R}^1_n \subseteq \mathcal{D}^2_n$ for every n. We define the* composition *of $\mathcal{F}^1$ with $\mathcal{F}^2$ as $\mathcal{F}^2 \circ \mathcal{F}^1 = \{\mathcal{F}^2_n \circ \mathcal{F}^1_n \colon \mathcal{D}^1_n \mapsto \mathcal{R}^2_n\}_{n\in\mathbb{N}}$, where $\mathcal{F}^2_n \circ \mathcal{F}^1_n = \{(f_2, f_1) \in \mathcal{F}^2_n \times \mathcal{F}^1_n\}$, and $(f_2, f_1)(x) := f_2(f_1(x))$.*

Definition 3 (XOR of function families). *Let $\mathcal{F}^1 = \{\mathcal{F}^1_n \colon \mathcal{D}^1_n \mapsto \mathcal{R}^1_n\}_{n\in\mathbb{N}}$ and $\mathcal{F}^2 = \{\mathcal{F}^2_n \colon \mathcal{D}^2_n \mapsto \mathcal{R}^2_n\}_{n\in\mathbb{N}}$ be two ensembles of function families with $\mathcal{R}^1_n, \mathcal{R}^2_n \subseteq \{0,1\}^{\ell(n)}$ for every n. We define the* XOR *of $\mathcal{F}^1$ with $\mathcal{F}^2$ as $\mathcal{F}^2 \bigoplus \mathcal{F}^1 = \{\mathcal{F}^2_n \bigoplus \mathcal{F}^1_n \colon \mathcal{D}^1_n \cap \mathcal{D}^2_n \mapsto \{0,1\}^{\ell(n)}\}_{n\in\mathbb{N}}$, where $\mathcal{F}^2_n \bigoplus \mathcal{F}^1_n = \{(f_2, f_1) \in \mathcal{F}^2_n \times \mathcal{F}^1_n\}$, and $(f_2, f_1)(x) := f_2(x) \oplus f_1(x)$.*

Pairwise Independent Hashing

Definition 4 (pairwise independent families). *A function family $\mathcal{H} = \{h \colon \mathcal{D} \mapsto \mathcal{R}\}$ is* pairwise independent *(with respect to $\mathcal{D}$ and $\mathcal{R}$), if*

$$\Pr_{h\leftarrow\mathcal{H}}[h(x_1) = y_1 \wedge h(x_2) = y_2] = \frac{1}{|\mathcal{R}|^2},$$

for every distinct $x_1, x_2 \in \mathcal{D}$ and every $y_1, y_2 \in \mathcal{R}$.

For every $\ell \in$ poly, the existence of efficient pairwise-independent family ensembles mapping strings of length n to strings of length $\ell(n)$ is well known ([4]). In this paper we use efficient pairwise-independent function family ensembles mapping strings of length n to the set $[T(n)]_{\{0,1\}^n}$, where $T(n) \leq 2^n$ and is without loss of generality a power of two.[4] Let $\mathcal{H}$ be an efficient length-preserving, pairwise-independent function family ensemble and assume that $t(n) := \log T(n)$ is polynomial-time computable. Then the function family $\widehat{\mathcal{H}} = \{\widehat{\mathcal{H}_n} = \{h' \colon h \in \mathcal{H}_n, h'(x) = 0^{n-t(n)}||h(x)_{1,\ldots,t(n)}\}\}$, is an efficient pairwise-independent function family ensemble, mapping strings of length n to the set $[T(n)]_{\{0,1\}^n}$.

[4] For our applications, see Section 3, we can always consider $T'(n) = 2^{\lfloor \log(T(n)) \rfloor}$, which only causes us a factor of two loss in the resulting security.

Pseudorandom Functions

Definition 5 (pseudorandom functions). *An efficient function family ensemble $\mathcal{F} = \{\mathcal{F}_n \colon \{0,1\}^n \mapsto \{0,1\}^{\ell(n)}\}_{n \in \mathbb{N}}$ is a $(T(n), \varepsilon(n))$*-adaptive PRF*, if for every oracle-aided algorithm (distinguisher)* D *of running time $T(n)$ and large enough n, it holds that*

$$\left|\Pr_{f \leftarrow \mathcal{F}_n}[\mathsf{D}^f(1^n) = 1] - \Pr_{\pi \leftarrow \Pi_n}[\mathsf{D}^\pi(1^n) = 1]\right| \leq \varepsilon(n),$$

where Π_n is the set of all functions from $\{0,1\}^n$ to $\{0,1\}^{\ell(n)}$. If we limit D *above to be non-adaptive (i.e., it has to write all his oracle calls before making the first call), then $\mathcal{F}$ is called $(T(n), \varepsilon(n))$*-non-adaptive PRF.

The ensemble $\mathcal{F}$ is a t-adaptive PRF, if it is a $(t, 1/t)$-adaptive PRF according to the above definition. It is polynomially secure *adaptive PRF (for short, adaptive PRF), if it is a p-adaptive PRF for every $p \in$ poly. Finally, it is* super-polynomial *secure adaptive PRF, if it T-adaptive PRF for some $T(n) \in n^{\omega(1)}$. The same conventions are also used for non-adaptive PRFs.*

Clearly, a super-polynomial secure PRF is also polynomially secure. In Section 4 we prove that the converse is also true: a polynomially secure PRF is also super-polynomial secure PRF.

3 Our Construction

In this section we present the main contribution of this paper — a direct construction of an adaptive pseudorandom function family from a non-adaptive one.

Theorem 2 (restatement of Theorem 1). *Let T be a polynomial-time computable integer function, let $\mathcal{H} = \{\mathcal{H}_n \colon \{0,1\}^n \mapsto [T(n)]_{\{0,1\}^n}\}$ be an efficient pairwise independent function family ensemble, and let $\mathcal{F} = \{\mathcal{F}_n \colon \{0,1\}^n \mapsto \{0,1\}^{\ell(n)}\}$ be a $(p(n) \cdot T(n), \varepsilon(n))$-non-adaptive PRF, where $p \in$ poly is determined by the computation time of T, $\mathcal{F}$ and $\mathcal{H}$. Then $\mathcal{F} \circ \mathcal{H}$ is a $\left(s(n), \varepsilon(n) + \frac{s(n)^2}{T(n)}\right)$-adaptive PRF for every $s(n) < T(n)$.*

Theorem 2 yields the following simpler statement.

Corollary 2. *Let T, p and $\mathcal{H}$ be as in Theorem 2. Assuming $\mathcal{F}$ is a $(p(n)T(n))$-non-adaptive PRF, then $\mathcal{F} \circ \mathcal{H}$ is a $\left(\sqrt[3]{T(n)}/2\right)$-adaptive PRF.*

Proof. Applying Theorem 2 with respect to $s(n) = \sqrt[3]{T(n)}/2$ and $\varepsilon(n) = \frac{1}{p(n)T(n)}$, yields that $\mathcal{F} \circ \mathcal{H}$ is a $\left(s(n), \frac{1}{p(n)T(n)} + \frac{s(n)^2}{T(n)}\right)$-adaptive PRF. Since $\frac{1}{p(n)T(n)} < \frac{1}{2s(n)}$ and $\frac{s(n)^2}{T(n)} \leq \frac{1}{2s(n)}$, it follows that $\mathcal{F} \circ \mathcal{H}$ is a $(s, 1/s)$-adaptive PRF. □

To prove Theorem 2, we use the (non efficient) function family ensemble $\Pi \circ \mathcal{H}$, where $\Pi = \Pi_\ell$ (i.e., the ensemble of all functions from $\{0,1\}^n$ to $\{0,1\}^\ell$), and $\ell = \ell(n)$ is the output length of $\mathcal{F}$. We first show that $\mathcal{F} \circ \mathcal{H}$ is *computationally* indistinguishable from $\Pi \circ \mathcal{H}$, and complete the proof showing that $\Pi \circ \mathcal{H}$ is *statistically* indistinguishable from Π.

3.1 $\mathcal{F} \circ \mathcal{H}$ Is Computationally Indistinguishable From $\Pi \circ \mathcal{H}$

Lemma 1. *Let T, $\mathcal{F}$ and $\mathcal{H}$ be as in Theorem 2. Then for every oracle-aided distinguisher D of running time T, there exists a* non-adaptive *oracle-aided distinguisher $\widehat{\mathsf{D}}$ of running time $p(n) \cdot T(n)$, for some $p \in$ poly (determined by the computation time of T, $\mathcal{F}$ and $\mathcal{H}$), with*

$$\left|\Pr_{g \leftarrow \mathcal{F}_n}[\widehat{\mathsf{D}}^g(1^n) = 1] - \Pr_{g \leftarrow \Pi_n}[\widehat{\mathsf{D}}^g(1^n) = 1]\right| = \left|\Pr_{g \leftarrow \mathcal{F}_n \circ \mathcal{H}_n}[\mathsf{D}^g(1^n) = 1] - \Pr_{g \leftarrow \Pi_n \circ \mathcal{H}_n}[\mathsf{D}^g(1^n) = 1]\right|$$

for every $n \in \mathbb{N}$, where Π_n is the set of all functions from $\{0,1\}^n$ to $\{0,1\}^{\ell(n)}$.

In particular, the pseudorandomness of $\mathcal{F}$ yields that $\mathcal{F} \circ \mathcal{H}$ is computationally indistinguishable from the ensemble $\{\Pi_n \circ \mathcal{H}_n\}_{n \in \mathbb{N}}$ by an adaptive distinguisher of running time T.

Proof. The distinguisher $\widehat{\mathsf{D}}$ is defined as follows:

Algorithm 3 ($\widehat{\mathsf{D}}$)

Input: 1^n.
Oracle: *a function ϕ over $\{0,1\}^n$.*

```
1. Compute $\phi(x)$ for every $x \in [T(n)]_{\{0,1\}^n}$.
2. Set $g = \phi \circ h$, where $h$ is uniformly chosen in $\mathcal{H}_n$.
3. Emulate $\mathsf{D}^g(1^n)$: answer a query $x$ to $\phi$ made by $\mathsf{D}$ with $g(x)$, using the
   information obtained in Step 1.
```

....

Note that $\widehat{\mathsf{D}}$ makes $T(n)$ *non-adaptive* queries to ϕ, and it can be implemented to run in time $p(n)T(n)$, for large enough $p \in$ poly. We conclude the proof by observing that in case ϕ is uniformly drawn from $\mathcal{F}_n$, the emulation of D done in $\widehat{\mathsf{D}}^\phi$ is identical to a random execution of D^g with $g \leftarrow \mathcal{F}_n \circ \mathcal{H}_n$. Similarly, in case ϕ is uniformly drawn from Π_n, the emulation is identical to a random execution of D^π with $\pi \leftarrow \Pi_n$. □

3.2 $\Pi \circ \mathcal{H}$ Is Statistically Indistinguishable From Π

The following lemma is commonly used for proving the security of hash based MACs (cf., [9, Proposition 6.3.6]), yet for completeness we give it a full proof below.

Lemma 2. *Let n, T be integers with $T \leq 2^n$, and let $\mathcal{H}$ be a pairwise-independent function family mapping string of length n to $[T]_{\{0,1\}^n}$. Let D be an (unbounded) s-query oracle-aided algorithm (i.e., making at most s queries), then*

$$\left|\Pr_{g \leftarrow \Pi \circ \mathcal{H}}[\mathsf{D}^g = 1] - \Pr_{\pi \leftarrow \Pi}[\mathsf{D}^\pi = 1]\right| \leq s^2/T,$$

where Π is the set of all functions from $\{0,1\}^n$ to $\{0,1\}^\ell$ (for some $\ell \in \mathbb{N}$).

Proof. We assume for simplicity that D is deterministic (the reduction to the randomized case is standard) and makes exactly s valid (i.e., inside $\{0,1\}^n$) distinct queries, and let $\Omega = (\{0,1\}^\ell)^s$. Consider the following random process:

Algorithm 4

```
1. Emulate $\mathsf{D}$, while answering the $i$'th query $q_i$ with a uniformly chosen $a_i \in \{0,1\}^\ell$.
   Set $\overline{q} = (q_1, \ldots, q_s)$ and $\overline{a} = (a_1, \ldots, a_s)$.
2. Choose $h \leftarrow \mathcal{H}$.
3. Emulate $\mathsf{D}$ again, while answering the $i$'th query $q'_i$ with $a'_i = a_i$ (the same $a_i$ from Step 1), if $h(q'_i) \notin \{h(q'_j)\}_{j \in [i-1]}$, and with $a'_i = a_j$, if $h(q'_i) = h(q'_j)$ for some $j \in [i-1]$.
   Set $\overline{q'} = (q'_1, \ldots, q'_s)$ and $\overline{a'} = (a'_1, \ldots, a'_s)$.
```

....

Let $\overline{A}$, $\overline{Q}$, $\overline{A'}$, $\overline{Q'}$ and H be the (jointly distributed) random variables induced by the values of $\overline{q}$, $\overline{a}$, $\overline{q'}$, $\overline{a'}$ and h respectively, in a random execution of the above process. It is not hard to verify that $\overline{A}$ is distributed the same as the oracle answers in a random execution of D^π with $\pi \leftarrow \Pi$, and that $\overline{A'}$ is distributed the same as the oracle answers in a random execution of D^g with $g \leftarrow \Pi \circ \mathcal{H}$. Hence, for proving Lemma 2, it suffices to bound the statistical distance between $\overline{A}$ and $\overline{A'}$.

Let Coll be the event that $H(\overline{Q}_i) = H(\overline{Q}_j)$ for some $i \neq j \in [s]$. Since the queries and answers in both emulations of Algorithm 4 are the same until a collision with respect to H occurs, it follows that

$$\Pr[\overline{A} \neq \overline{A'}] \leq \Pr[\text{Coll}] \tag{1}$$

On the other hand, since H is chosen *after* $\overline{Q}$ is set, the pairwise independent of $\mathcal{H}$ yields that

$$\Pr[\text{Coll}] \leq s^2/T, \tag{2}$$

and therefore $\Pr[\overline{A} \neq \overline{A'}] \leq s^2/T$. It follows that $\Pr[\overline{A} \in C] \leq \Pr[\overline{A'} \in C] + s^2/T$ for every $C \subseteq \Omega$, yielding that $\mathrm{SD}(\overline{A}, \overline{A'}) \leq s^2/T$. □

3.3 Putting It Together

We are now finally ready to prove Theorem 2.

Proof (of Theorem 2). Let D be an oracle-aided algorithm of running time s with $s(n) < T(n)$. Lemma 1 yields that $|\Pr_{g \leftarrow \mathcal{F}_n \circ \mathcal{H}_n}[\mathsf{D}^g(1^n) = 1] - \Pr_{g \leftarrow \Pi_n \circ \mathcal{H}_n}[\mathsf{D}^g(1^n) = 1]| \leq \varepsilon(n)$ for large enough n, where Lemma 2 yields that $|\Pr_{g \leftarrow \Pi_n \circ \mathcal{H}_n}[\mathsf{D}^g(1^n) = 1] - \Pr_{\pi \leftarrow \Pi_n}[\mathsf{D}^\pi(1^n) = 1]| \leq s(n)^2/T(n)$ for every $n \in \mathbb{N}$. Hence, the triangle inequality yields that $|\Pr_{g \leftarrow \mathcal{F}_n \circ \mathcal{H}_n}[\mathsf{D}^g(1^n) = 1] - \Pr_{\pi \leftarrow \Pi_n}[\mathsf{D}^\pi(1^n) = 1]| \leq \varepsilon(n) + s(n)^2/T(n)$ for large enough n, as requested. □

3.4 Handling Polynomial Security

Corollary 2 is only useful when the security of the underlying non-adaptive PRF (i.e., T) is efficiently computable (or when considering non-uniform PRF constructions, see Section 1.1). In this section we show how to handle the important case of polynomially secure non-adaptive PRF. We use the following "combiner".

Definition 6. *Let $\mathcal{H}$ be a function family into $\{0,1\}^n$. For $i \in [n]$, let $\widehat{\mathcal{H}}^i$ be the function family $\widehat{\mathcal{H}}^i = \{\widehat{h}\colon h \in \mathcal{H}\}$, where $\widehat{h}(x) = 0^{n-i}||h(x)_{1,\ldots,i}$.*

Corollary 3. *Let $\mathcal{F}$ be a $T(n)$-non-adaptive PRF, let $\mathcal{H}$ be an efficient length-preserving pairwise-independent function family ensemble, and let $\mathcal{I}(n) \subseteq [n]$ be polynomial-time computable (in n) index set. Define the function family ensemble $G = \{G_n\}_{n\in\mathbb{N}}$, where $G_n = \bigoplus_{i\in\mathcal{I}(n)} \left(\mathcal{F}_n \circ \widehat{\mathcal{H}_n}^{i}\right)$.*

There exists $q \in \mathrm{poly}$ such that G is a $\left(\sqrt[3]{2^{t(n)}}/2\right)$-adaptive PRF, for every polynomial-time computable integer function t, with $t(n) \in \mathcal{I}(n)$ and $2^{t(n)} \leq T(n)/q(n)$.

Before proving the corollary, let us first use it for constructing adaptive PRF from non-adaptive polynomially secure one.

Corollary 4 (restatement of Corollary 1). *Let $\mathcal{F}$ be a polynomially secure non-adaptive PRF, let $\mathcal{H}$ be an efficient pairwise-independent length-preserving function family ensemble and let $k(n) \in \omega(1)$ be polynomial-time computable function. Then $G := \{\bigoplus_{i\in[k(n)]} \left(\mathcal{F}_n \circ \widehat{\mathcal{H}_n}^{\lfloor i\cdot\log n\rfloor}\right)\}_{n\in\mathbb{N}}$ is polynomially secure adaptive PRF.*

Proof. Let $\mathcal{I}(n) := \{\lfloor\log n\rfloor, \lfloor 2\cdot\log n\rfloor \ldots, \lfloor k(n)\cdot\log n\rfloor\}$. Applying Corollary 3 with respect to $\mathcal{F}$, $\mathcal{H}$, $\mathcal{I}$ and $t(n) = \lfloor c\cdot\log n\rfloor$, where $c \in \mathbb{N}$, yields that G is a $O(\sqrt[3]{n^c})$-adaptive PRF. It follows that G is p-adaptive PRF for every $p \in \mathrm{poly}$. Namely, G is polynomially secure adaptive PRF. □

Remark 1 (unknown security). Corollary 3 is also useful when the security of $\mathcal{F}$ is "not known" in the construction time. Taking $\mathcal{I}(n) = \{1, 2, 4, \ldots, 2^{\lfloor\log n\rfloor}\}$ (resulting in $\log n$ calls to $\mathcal{F}$) and assuming that $\mathcal{F}$ is found to be $T(n)$-non-adaptive PRF for some polynomial-time computable T, the resulting PRF is guaranteed to be $O(\sqrt[6]{T(n)})$-adaptive PRF (neglecting polynomial factors).

Proof (of Corollary 3). It is easy to see that G is efficient, so it is left to argue for its security. Let $q(n) = q'(n)p(n)$, where p is as in the statement of Corollary 2, and $q' \in \mathrm{poly}$ to be determined later. Let t be a polynomial-time computable integer function with $t(n) \in \mathcal{I}(n)$ and $2^{t(n)} \leq T(n)/q(n)$. It follows that $\widehat{\mathcal{H}}^t = \{\widehat{\mathcal{H}_n}^{t(n)}\}_{n\in\mathbb{N}}$ is an efficient pairwise-independent function family ensemble, and Corollary 2 yields that $\mathcal{F} \circ \widehat{\mathcal{H}}^t$ is a $\left(\sqrt[3]{q'(n)2^{t(n)}}/2\right)$-adaptive PRF.

Assume towards a contradiction that there exists an oracle-aided distinguisher D that runs in time $T'(n) = \sqrt[3]{2^{t(n)}}/2$ and

$$|\Pr_{g \leftarrow G_n}[\mathsf{D}^g(1^n) = 1] - \Pr_{\pi \leftarrow \Pi_n}[\mathsf{D}^\pi(1^n) = 1]| > 1/T'(n) \tag{3}$$

for infinitely many n's. We use the following distinguisher for breaking the pseudorandomness of $\mathcal{F} \circ \widehat{\mathcal{H}}^t$:

Algorithm 5 ($\widehat{\mathsf{D}}$)

Input: 1^n.
Oracle: *a function* ϕ *over* $\{0,1\}^n$.

1. *For every* $i \in \mathcal{I}(n) \setminus \{t(n)\}$, *choose* $g^i \leftarrow \mathcal{F}_n \circ \widehat{\mathcal{H}_n}^i$.
2. *Set* $g := \phi \oplus \bigoplus_{i \in \mathcal{I}(n) \setminus \{t(n)\}} g^i$.
3. *Emulate* $\mathsf{D}^g(1^n)$.

....

Note that $\widehat{\mathsf{D}}$ can be implemented to run in time $|\mathcal{I}(n)| \cdot r(n) \cdot T'(n)$ for some $r \in$ poly, which is smaller than $\sqrt[3]{q'(n)2^{t(n)}}/2$ for large enough q'. Also note that in case ϕ is uniformly distributed over Π_n, then g (selected by $\widehat{\mathsf{D}}^\phi(1^n)$) is uniformly distributed in Π_n, where in case ϕ is uniformly distributed in $\mathcal{F}_n \circ \widehat{\mathcal{H}_n}^{t(n)}$, then g is uniformly distributed in G_n. It follows that

$$\left|\Pr_{g \leftarrow (\mathcal{F} \circ \widehat{\mathcal{H}}^t)_n}[\widehat{\mathsf{D}}^g(1^n) = 1] - \Pr_{\pi \leftarrow \Pi_n}[\widehat{\mathsf{D}}^\pi(1^n) = 1]\right| = |\Pr_{g \leftarrow G_n}[\mathsf{D}^g(1^n) = 1] - \Pr_{\pi \leftarrow \Pi_n}[\mathsf{D}^\pi(1^n) = 1]| \tag{4}$$

for every $n \in \mathbb{N}$. In particular, Equation (3) yields that

$$\left|\Pr_{g \leftarrow (\mathcal{F} \circ \widehat{\mathcal{H}}^t)_n}[\widehat{\mathsf{D}}^g(1^n) = 1] - \Pr_{\pi \leftarrow \Pi_n}[\widehat{\mathsf{D}}^\pi(1^n) = 1]\right| > \frac{2}{\sqrt[3]{2^{t(n)}}} > \frac{2}{\sqrt[3]{q'(n)2^{t(n)}}}$$

for infinitely many n's, in contradiction to the pseudorandomness of $\mathcal{F} \circ \widehat{\mathcal{H}}^t$ we proved above. □

Acknowledgment. We are very grateful to Omer Reingold for very useful discussions, and for challenging the second author with this research question a long while ago.

References

1. Bellare, M.: A note on negligible functions. Journal of Cryptology, 271–284 (2002)
2. Bellare, M., Goldwasser, S.: New Paradigms for Digital Signatures and Message Authentication Based on Non-interactive Zero Knowledge Proofs. In: Brassard, G. (ed.) CRYPTO 1989. LNCS, vol. 435, pp. 194–211. Springer, Heidelberg (1990)

3. Blum, M., Evans, W.S., Gemmell, P., Kannan, S., Naor, M.: Checking the correctness of memories. Algorithmica 12(2/3), 225–244 (1994)
4. Carter, L.J., Wegman, M.N.: Universal classes of hash functions. Journal of Computer and System Sciences, 143–154 (1979)
5. Cho, C., Lee, C.-K., Ostrovsky, R.: Equivalence of Uniform Key Agreement and Composition Insecurity. In: Rabin, T. (ed.) CRYPTO 2010. LNCS, vol. 6223, pp. 447–464. Springer, Heidelberg (2010)
6. Chor, B., Fiat, A., Naor, M., Pinkas, B.: Tracing traitors. IEEE Transactions on Information Theory 46(3), 893–910 (2000)
7. Goldreich, O.: Towards a Theory of Software Protection. In: Odlyzko, A.M. (ed.) CRYPTO 1986. LNCS, vol. 263, pp. 426–439. Springer, Heidelberg (1987)
8. Goldreich, O.: Foundations of Cryptography: Basic Tools. Cambridge University Press (2001)
9. Goldreich, O.: Foundations of Cryptography. Basic Applications, vol. 2. Cambridge University Press (2004)
10. Goldreich, O., Goldwasser, S., Micali, S.: On the Cryptographic Applications of Random Functions. In: Blakely, G.R., Chaum, D. (eds.) CRYPTO 1984. LNCS, vol. 196, pp. 276–288. Springer, Heidelberg (1985)
11. Goldreich, O., Goldwasser, S., Micali, S.: How to construct random functions. Journal of the ACM, 792–807 (1986)
12. Håstad, J., Impagliazzo, R., Levin, L.A., Luby, M.: A pseudorandom generator from any one-way function. SIAM Journal on Computing, 1364–1396 (1999)
13. Luby, M.: Pseudorandomness and cryptographic applications. Princeton computer science notes. Princeton University Press (1996) ISBN 978-0-691-02546-9
14. Luby, M., Rackoff, C.: How to construct pseudorandom permutations from pseudorandom functions. SIAM Journal on Computing
15. Maurer, U.M., Pietrzak, K.: Composition of Random Systems: When Two Weak Make One Strong. In: Naor, M. (ed.) TCC 2004. LNCS, vol. 2951, pp. 410–427. Springer, Heidelberg (2004)
16. Myers, S.: Black-Box Composition Does Not Imply Adaptive Security. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 189–206. Springer, Heidelberg (2004)
17. Naor, M., Reingold, O.: Synthesizers and their application to the parallel construction of psuedo-random functions. In: Proceedings of the 36th Annual Symposium on Foundations of Computer Science (FOCS), pp. 170–181 (1995)
18. Ostrovsky, R.: An Efficient Software Protection Scheme. In: Brassard, G. (ed.) CRYPTO 1989. LNCS, vol. 435, pp. 610–611. Springer, Heidelberg (1990)
19. Pietrzak, K.: Composition Does Not Imply Adaptive Security. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, pp. 55–65. Springer, Heidelberg (2005)
20. Pietrzak, K.: Composition Implies Adaptive Security in Minicrypt. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 328–338. Springer, Heidelberg (2006)

4 From Polynomial to Super-Polynomial Security

The standard security definition for cryptographic primitives is *polynomial security*: any PPT trying to break the primitive has only negligible success probability. Bellare [1] showed that for any polynomially secure primitive there exists a *single* negligible function μ, such that no PPT can break the primitive with probability larger than μ. Here we take his approach a step further, showing that for a polynomially secure primitive there exists a super-polynomial function T, such that no adversary of running time T breaks the primitive with probability larger than $1/T$.

In the following we identify algorithms with their string description. In particular, when considering algorithm A, we mean the algorithm defined by the string A (according to some canonical representation). We prove the following result.

Theorem 6. *Let $v\colon \{0,1\}^* \times \mathbb{N} \mapsto [0,1]$ be a function with the following properties: 1) $v(\mathsf{A}, n) \leq 1/p(n)$ for every oracle-aided* PPT A*, $p \in \text{poly}$ and large enough n; and 2) if the distributions induced by random executions of $\mathsf{A}^f(x)$ and $\mathsf{B}^f(x)$ are the same for any input $x \in \{0,1\}^n$ and function f (each distribution describes the algorithm's output and oracle queries), then $v(\mathsf{A}, n) = v(\mathsf{B}, n)$.*

Then there exists an integer function $T(n) \in n^{\omega(1)}$ such that following holds: for any algorithm A of running time at most $T(n)$, it holds that $v(\mathsf{A}, n) \leq 1/T(n)$ for large enough n.

Remark 2 (Applications). Let f be a polynomially secure OWF (i.e., $\Pr[\mathsf{A}(f(U_n)) \in f^{-1}(f(U_n))] = \text{neg}(n)$ for any PPT A). Applying Theorem 6 with $v(\mathsf{A}, n) := \Pr[\mathsf{A}(f(U_n)) \in f^{-1}(f(U_n))]$ (where if A expects to get an oracle, provide him with the constant function $\phi(x) = 1$), yields that f is super-polynomial secure OWF (i.e., exists $T(n) \in n^{\omega(1)}$ such that $\Pr[\mathsf{A}(f(U_n)) \in f^{-1}(f(U_n))] \leq 1/T(n)$ for any algorithm of running time T and large enough n).

Similarly, for a polynomially secure PRF $\mathcal{F} = \{\mathcal{F}_n\}_{n\in\mathbb{N}}$ (see Definition 5), applying Theorem 6 with $v(\mathsf{A}, n) := \left|\Pr_{f\leftarrow\mathcal{F}_n}[A^f(1^n) = 1] - \Pr_{\pi\leftarrow\Pi_n}[A^\pi(1^n) = 1]\right|$, where Π_n is the set of all functions with the same domain/range as $\mathcal{F}_n$, yields that $\mathcal{F}$ is super-polynomial secure PRF.

Proof (of Theorem 6). Given a probabilistic algorithm A and an integer i, let A_i denote the variant of A that on input of length n, halts after n^i steps (hence, A_i is a PPT). Let $\mathcal{S}_i$ be the first i strings in $\{0,1\}^*$, according to some canonical order, viewed as descriptions of i algorithms. Let $\mathcal{I}(n) = \{i \in [n]\colon \forall \mathsf{A} \in \mathcal{S}_i, k \geq n\colon v(\mathsf{A}_i, k) < 1/k^i\} \cup \{1\}$, let $t(n) = \max \mathcal{I}(n)$ and $T(n) = n^{t(n)}$.

Let A be an algorithm of running time $T(n)$, and let i_{A} be the first integer such that $\mathsf{A} \in \mathcal{S}_{i_{\mathsf{A}}}$. In Claim 7 we prove that $t(n) \in \omega(1)$, hence it follows that $t(n) > i_{\mathsf{A}}$ for any large enough n. For any such n, the definition of t guarantees that $v(\mathsf{A}_{t(n)}, n) < 1/n^{t(n)} = 1/T(n)$. Since A is of running time $T(n)$, the second property of v yields that $v(\mathsf{A}, n) = v(\mathsf{A}_{t(n)}, n)$, and therefore $v(\mathsf{A}, n) < 1/T(n)$. □

Claim 7. It holds that $t(n) \in \omega(1)$.

Proof. Fix $i \in \mathbb{N}$. For each $\mathsf{A} \in \mathcal{S}_i$, let n_{A} be the first integer such that $v(\mathsf{A}_i, n) \leq 1/n^i$ for every $n \geq n_{\mathsf{A}}$ (note that such n_{A} exists by the first property of v), and let $n_i = \max\{n_{\mathsf{A}}\colon \mathsf{A} \in \mathcal{S}_i\}$. It follows that $v(\mathsf{A}_i, n) \leq 1/n^i$ for every $n \geq n_i$ and $\mathsf{A} \in \mathcal{S}_i$, and therefore $t(n_i) \geq i$. □