Electronic Colloquium on Computational Complexity, Report No. 173 (2012)

A Cookbook for Black-Box Separations and a Recipe for UOWHFs*

Kfir Barhum Thomas Holenstein

Department of Computer Science
ETH Zurich, 8092 Zurich, Switzerland

December 2012

Abstract

We present a new framework for proving fully black-box separations and lower bounds. We prove a general theorem that facilitates the proofs of fully black-box lower bounds from a one-way function (OWF).

Loosely speaking, our theorem says that in order to prove that a fully black-box construction does not securely construct a cryptographic primitive **Q** (e.g., a pseudo-random generator or a universal one-way hash function) from a OWF, it is enough to come up with a large enough set of functions $\mathcal{F}$ and a parameterized oracle (i.e., an oracle that is defined for every $f \in \{0,1\}^n \to \{0,1\}^n$) such that $\mathcal{O}_f$ breaks the security of the construction when instantiated with f and the oracle satisfies two local properties.

Our main application of the theorem is a lower bound of $\Omega(n/\log(n))$ on the number of calls made by any fully black-box construction of a universal one-way hash function (UOWHF) from a general one-way function. The bound holds even when the OWF is regular, in which case it matches to a recent construction of Barhum and Maurer [BM12].

Keywords: Complexity-Based Cryptography, One-Way Functions, Universal One-Way Hash Functions, Black-Box Constructions, Lower-Bound on Efficiency

1 Introduction

1.1 Cryptographic Primitives and Black-Box Constructions

An important question in complexity-based cryptography is understanding which cryptographic primitives (e.g., one-way functions, pseudo-random generators) are implied by others. In principle, an implication between two primitives can be proved as a logical statement (e.g., the existence of one-way functions implies the existence of pseudo-random generators). However, most proofs of such implications (with very few exceptions, e.g., [Bar01]) are in fact so-called fully black-box constructions.

Informally, a black-box construction of a primitive **Q** from a primitive **P** is a pair of algorithms, called *construction* and *reduction*, such that the construction, using only the functionality of **P**, implements **Q** and the reduction, using only the functionality of **P** and the one of a potential breaker algorithm, breaks **P** whenever the breaker algorithm breaks **Q**. As a corollary, such a black-box construction establishes that the existence of **P** implies the existence of **Q**.

*This is the full version of a paper due to appear at the 10th Theory of Cryptography Conference (TCC 2013).

ISSN 1433-8092

One of many such examples is the construction of a one-way function from a weak one-way function [Yao82].

After futile attempts to prove that the existence of one-way functions implies that of key agreement, Impagliazzo and Rudich [IR89] proved the first black-box separation result: They showed that there is no fully black-box construction of key agreement from one-way functions. Their seminal work inspired a plethora of similar results and nowadays one identifies two main types of black-box separation results: black-box separations of a primitive **Q** from a primitive **P** and lower bounds on some complexity parameter (e.g., seed length, number of calls to the underlying primitive, etc.) in the construction of **Q** from **P**. Besides [IR89], the work of Simon [Sim98], where he shows that there is no fully black-box construction of a collision-resistant hash function from a one-way function, is an example of the former. As an example of the latter, Kim *et. al.* [KST99] established a lower bound of $\Omega(\sqrt{k/\log(n)})$ on the number of queries of any construction of a universal one-way hash function that compresses k bits from a one-way permutation on n bits. This was later improved by Gennaro *et. al.* [GGKT05] to $\Omega(k/\log(n))$.

Reingold *et. al.* [RTV04] were the first to formalize a model for and study the relations between different notions of "black-boxness" of cryptographic constructions.

A key property of a fully black-box construction of **Q** from **P** is the requirement that it constructs **Q** efficiently even when given black-box access to a non-efficient implementation of **P**. A proof technique utilizing this property, which is implicit in many black-box separations, involves an (inefficient) oracle instantiation of the primitive **P** and an appropriate (inefficient) breaker oracle B. The separation is usually proved by showing that B breaks the security of the candidate construction for **Q**, but at the same time no efficient oracle algorithm that has black-box oracle access to both the breaker and the primitive (in particular, the potential reduction) breaks the security property of the underlying instantiation of **P**.

In [HR04], Hsiao and Reyzin introduce the "two-oracle" paradigm, referring to the oracle implementations of **P** and the breaker B. The separation in [MM11] also makes explicit use of this paradigm.

1.2 Our Contribution

In constructions based on one-way functions (or permutations), i.e., when $\mathbf{P} = \mathbf{OWF}$, the oracle that implements **OWF** is usually set to be a random permutation, which is one-way with very high probability even in the presence of a non-uniform algorithm. On the other hand, the proof that the breaker algorithm for the constructed primitive **Q** does not help invert the permutation is repeated in an "ad-hoc" manner in many separation proofs, e.g., in [Sim98, HHRS07] and also in a recent result on lower bounds on the number of calls made by any construction of a pseudo-random generator from a one-way function [HS12].

Thus, while in many separation proofs the task of finding the right breaker oracle is different (this is inherent, as each time it is required to break the security of a different primitive), we observe that the proof that it does not help in inverting the underlying one-way function can be facilitated and unified to a large extent. To that end, we prove a general theorem that facilitates the proof of black-box separations (Theorem 18). In particular, we show that any circuit with access to an oracle that satisfies two local properties, does not help to invert many functions.

Our framework allows proving separation results that exclude the existence of reductions with very weak security requirements. In this work we focus on the important case where the black-box construction is so-called fixed-parameter. That is, for a security parameter ρ, both the construction algorithm and the reduction access the primitive and breaker of security ρ only. All black-box constructions found in the literature are in fact fixed-parameter constructions. We believe that adapting the approach of [HS12], it is possible to extend our results to the most

general case.

Our proof uses the encoding technique from [GGKT05], which was already adapted to the special cases in [HHRS07] and [HS12]. We also use the bending technique that originated in [Sim98] and was subsequently used in [HH09] and [HS12].

As an application, in Section 4 we prove a lower bound of $\Omega(n/\log^3(n))$ on the number of calls made by any fully black-box construction of a universal one-way hash function (UOWHF) from a one-way function $f : \{0,1\}^n \to \{0,1\}^n$. The bound is improved in Section 5 to $\Omega(n/\log(n))$, which matches the construction from [BM12].

UOWHFs are a fundamental cryptographic primitive, most notably used for obtaining digital signatures. They were studied extensively since their introduction by Naor and Yung [NY89], who showed a simple construction that makes only one call to the underlying one-way function whenever, additionally, the function is a permutation. Rompel [Rom90] showed a construction based on any one-way function, and the most efficient construction based on general one-way functions is due to Haitner *et. al.* [HHR$^+$10]. Their construction makes $\tilde{O}(n^6)$ calls to a one-way function $f : \{0,1\}^n \to \{0,1\}^n$. Note that the bound given in [GGKT05] does not say anything for the mere construction of a UOWHF (e.g., for a function which compresses one bit), and prior to our work it would have been possible to conjecture that there exists a construction of a UOWHF from a general one-way function that makes only one call to the underlying one-way function. Our bound matches exactly and up to a log-factor the number of calls made by the constructions of [BM12] and [**?**], respectively.

Our result can be understood as an analog to that of Holenstein and Sinha, who show a bound of $\Omega(n/\log(n))$ on the number of calls to a one-way function that are made by a construction of a pseudo-random generator. We observe (details are omitted) that the recent result of [HS12] can be explained in out framework. Our characterization of UOWHFs (presented in Section 4.1) is inspired by their characterization of pseudo-random generators. For some candidate constructions, our proof also utilizes their BreakOW oracle. Our main technical contribution in Section 4.2 is the oracle BreakPI and the proof that it satisfies the conditions of our theorem from Section 3.

2 Preliminaries

2.1 The Computational Model

A function $p = p(\rho)$ is polynomial if there exists a value c such that $p(\rho) = \rho^c$. A machine M is efficient if there exists a polynomial p such that on every input $x \in \{0,1\}^*$, $M(x)$ halts after at most $p(|x|)$ steps. A function $s : \mathbb{N}^+ \to \mathbb{N}^+$ is a security function if for every $\rho \in \mathbb{N}^+$ it holds that $s(\rho+1) \geq s(\rho)$, and s is efficiently computable (i.e., there exists an efficient machine M that on input 1^ρ outputs $s(\rho)$). For a security function s we define $\frac{1}{s} : \mathbb{N}^+ \to \mathbb{R}^+$ as $\frac{1}{s}(\rho) \stackrel{\text{def}}{=} \frac{1}{s(\rho)}$. A function $f : \mathbb{N}^+ \to \mathbb{R}^+$ is negligible if for all polynomial security functions p it holds that $f(\rho) < \frac{1}{p(\rho)}$ for all large enough ρ.

A boolean circuit $A : \{0,1\}^m \to \{0,1\}^{m'}$ is a directed acyclic graph in which every node (called gate) is either an input node of in-degree 0 labeled as one of the m input bits, an output node labeled by one of the m' output bits, an AND gate, an OR or a NOT gate. A circuit A implements the function $f : \{0,1\}^m \to \{0,1\}^{m'}$ that corresponds to its evaluation on its inputs. The converse also holds: For every function $f : \{0,1\}^m \to \{0,1\}^{m'}$ it is always possible to define canonically[1] a circuit A that implements f.

[1] E.g., the circuit that implements the DNF of f.

Let $n, n' \in \mathbb{N}^+$. An (n, n')-oracle circuit $C^{(?)}$ is a circuit that additionally has special "oracle" gates, each having in-degree n and out-degree n'. A circuit $C^{(?)}$ is an oracle circuit if it is an (l, l')-oracle circuit for some $l, l' \in \mathbb{N}^+$. Let $C^{(?)}$ be an (n, n')-oracle circuit and let A be a circuit with n input gates and n' output gates (in this case we say that A is compatible with $C^{(?)}$). The circuit $C^{(A)}$ is defined as the circuit $C^{(?)}$ where each oracle gate is substituted by a copy A. For a function $f : \{0,1\}^n \to \{0,1\}^{n'}$ define $C^{(f)}$ as the circuit $C^{(?)}$ where each oracle gate is substituted by a copy of the canonical circuit that evaluates f. Similarly, let $n_1, n_1', n_2, n_2' \in \mathbb{N}^+$. An (n_1, n_1', n_2, n_2')-two oracle circuit $C^{(?,?)}$ is a circuit that has two types of oracle gates, where a gate of the first type has n_1 inputs and n_1' outputs and a gate of the second type has n_2 inputs and n_2' outputs. As before, for compatible functions and circuits that evaluate compatible functions $f_1 : \{0,1\}^{n_1} \to \{0,1\}^{n_1'}$ and $f_2 : \{0,1\}^{n_2} \to \{0,1\}^{n_2'}$ the circuit $C^{(f_1,f_2)}$ is defined similarly.

A non-uniform algorithm $A = \{A_\rho\}_{\rho\in\mathbb{N}^+}$ is a parameterized family of circuits A_ρ. A non-uniform algorithm A implements the parametrized functions family $f = \{f_\rho\}_{\rho\in\mathbb{N}^+}$, if each A_ρ implements f_ρ.

A non-uniform oracle algorithm $A^{(?)} = \{A^{(?)}{}_\rho\}_{\rho\in\mathbb{N}^+}$ is a parameterized family of oracle circuits. For an oracle algorithm $A^{(?)}$ and a family of functions f we define $A^{(f)} \stackrel{\text{def}}{=} \{A_\rho^{f_\rho}\}$ (resp., $A^{(B)} \stackrel{\text{def}}{=} \{A_\rho^{B_\rho}\}$) whenever for every ρ the function is compatible with the oracle-circuit.

2.1.1 Uniform generation of oracle algorithms.

The construction and reduction algorithms in fully black-box constructions are assumed to work for any[2] input/output lengths of the primitive and breaker functionalities, and therefore are modeled in the following way: In addition to the security parameter ρ, both the construction and the reduction algorithms take as input information about the input/output lengths of the underlying primitive f_ρ and the breaker algorithm B_ρ.

A uniform oracle algorithm is a machine M that on input $M(1^\rho, n(\rho), n'(\rho))$ outputs an $(n(\rho), n'(\rho))$-oracle circuit $A_\rho^{(?)}$. For a uniform oracle algorithm M and a parameterized family of functions $f = \{f_\rho : \{0,1\}^{n(\rho)} \to \{0,1\}^{n'(\rho)}\}_{\rho\in\mathbb{N}^+}$, define $M^{(f)} \stackrel{\text{def}}{=} \{A_\rho^{(f_\rho)}\}_{\rho\in\mathbb{N}^+}$, where $A_\rho^{(?)} \stackrel{\text{def}}{=} M(1^\rho, n(\rho), n'(\rho))$. For a non-uniform algorithm A, the family $M^{(A)}$ is defined analogously.

Let $s = s(\rho)$ be a security function. An s-non-uniform two oracle algorithm is a machine M such that for every $\rho, n_1, n_1', n_2, n_2' \in \mathbb{N}^+$ and every $a \in \{0,1\}^{s(\rho)}$, it holds that $M(1^\rho, n_1, n_1', n_2, n_2', a)$ outputs an (n_1, n_1', n_2, n_2')-two oracle circuit $A_{\rho,a}^{(?,?)}$ with at most $s(\rho)$ oracle gates. Note that the last requirement is essential and is implicit in the case of an efficient uniform oracle algorithm, where the number of oracle gates is bounded by the polynomial that bounds the running time of the algorithm. For an s-non-uniform two oracle algorithm M, a non-uniform algorithm B and a family of functions f, we formally define $M^{[B,f]} \stackrel{\text{def}}{=} (M, B, f)$.

2.2 Modeling Cryptographic Primitives

In order to state our results in their full generality, and in particular to exclude reductions that are allowed to use non-uniformity and are considered successful in inverting the one-way function even if they invert only a negligible fraction of the inputs of the function, the following two definitions are very general, and extend Definitions 2.1 and 2.3 from [RTV04]. The example of modeling a one-way function follows the definition.

[2] A-priori, for a fixed security parameter ρ there is no bound on the input length the construction is expected to work, as long as the series of the input-output lengths is bounded by *some* polynomial.

Definition 1 (Cryptographic Primitive)**.** *A primitive* $\mathbf{Q}$ *is a pair* $\langle F_{\mathbf{Q}}, R_{\mathbf{Q}} \rangle$*, where* $F_{\mathbf{Q}}$ *is a set of parametrized families of functions* $f = \{f_\rho\}_{\rho \in \mathbb{N}^+}$ *and* $R_{\mathbf{Q}}$ *is a relation over triplets* $\langle f_\rho, C, \epsilon \rangle$ *of a function* $f_\rho \in f$ *(for some* $f \in F_{\mathbf{Q}}$*), a circuit* C *and a number* $\epsilon > 0$*. We define that* C $(\mathbf{Q}, \epsilon)$-breaks f_ρ *if and only if* $\langle f_\rho, C, \epsilon \rangle \in R_{\mathbf{Q}}$*.*

The set $F_{\mathbf{Q}}$ specifies all the correct implementations (not necessarily efficient) of $\mathbf{Q}$ and the relation $R_{\mathbf{Q}}$ captures the security property of $\mathbf{Q}$, that is, it specifies for every concrete security parameter implementation, how well a breaker algorithm performs with respect to the security property of the primitive.

Finally, let $s = s(\rho)$ be a security function, $B = \{B_\rho\}_{\rho \in \mathbb{N}^+}$ be a non-uniform algorithm, and $f \in F_{\mathbf{Q}}$. We say that B $(\mathbf{Q}, \frac{1}{s})$-breaks f if $\langle f_\rho, B_\rho, \frac{1}{s(\rho)} \rangle \in R_{\mathbf{Q}}$ for infinitely many values ρ. Let us fix an s-non-uniform two oracle algorithm R. We say that $R^{[B,f]}$ $(\mathbf{Q}, \frac{1}{s})$-breaks f if for infinitely many values ρ there exists an $a \in \{0,1\}^{s(\rho)}$ (called advice) such that $\langle f_\rho, R_{\rho,a}^{(B_\rho, f_\rho)}, \frac{1}{s(\rho)} \rangle \in R_{\mathbf{Q}}$, where $R_{\rho,a}^{(?,?)} = R(1^\rho, n, n', b, b', a)$.

The usual notion of polynomial security of a primitive is captured by the following definition: B $\mathbf{Q}$-breaks f if there exists a polynomial $p = p(\rho)$ such that B $(\mathbf{Q}, \frac{1}{p})$-breaks f.

A primitive $\mathbf{Q}$ exists if there exists an efficient uniform algorithm M that implements an $f \in F_{\mathbf{Q}}$, and for every efficient uniform algorithm M' that, on input 1^ρ outputs a circuit, it holds that $\{M'(1^\rho)\}_{\rho \in \mathbb{N}^+}$ does not $\mathbf{Q}$-break f.

Observe that the requirement that M' outputs a circuit is made without loss of generality and captures the standard definition of an efficient randomized machine M' that breaks a primitive. Given such an M' that tosses at most $r = r(\rho)$ random coins, there exists[3] a (now deterministic) efficient uniform machine M'' that on input 1^ρ outputs a circuit C_ρ with $m(\rho) + r(\rho)$ input gates and $n(\rho)$ output gates that computes the output of M for all strings of length $m(\rho)$, and therefore $\mathbf{Q}$-breaks the primitive.

2.3 One-Way Functions

Our model for describing a primitive is very general and captures the security properties of many cryptographic primitives. As an example, we bring a standard definition of a one-way function and then explain how it can be described in our model.

Definition 2 (One-Way Function)**.** *A* one-way function $f = \{f_\rho\}_{\rho \in \mathbb{N}^+}$ *is an efficiently uniformly computable family of functions* $f_\rho : \{0,1\}^{n(\rho)} \to \{0,1\}^{m(\rho)}$*, such that for every efficient randomized machine* A*, the function that maps* ρ *to*

$$\Pr_{x \xleftarrow{r} \{0,1\}^{m(\rho)}} \left[A(1^\rho, f_\rho(x)) \in f_\rho^{-1}(f_\rho(v)) \right]$$

is negligible.

In order to model a one-way function (OWF), we set $f = \{f_\rho\}_{\rho \in \mathbb{N}^+} \in F_{\mathbf{OWF}}$, where $f_\rho : \{0,1\}^{n(\rho)} \to \{0,1\}^{m(\rho)}$, if and only if $n = n(\rho)$ and $m = m(\rho)$ are polynomial security functions. We say that $F_{\mathbf{OWF}}$ contains a collection of sets of functions $\mathcal{F} = \{\mathcal{F}_\rho\}_{\rho \in \mathbb{N}^+}$, if for every family $f' = \{f'_\rho\}_{\rho \in \mathbb{N}^+}$, where $f'_\rho \in \mathcal{F}_\rho$ for every ρ, it holds that $f' \in F_{\mathbf{OWF}}$.

In this case, for a function $f_\rho \in f \in F_{\mathbf{OWF}}$, a circuit C that inverts f_ρ on an ϵ-fraction of its inputs, and $\epsilon' > 0$, set $\langle f, C, \epsilon' \rangle \in R_{\mathbf{OWF}}$ if and only if $\epsilon \geq \epsilon'$. The definition is general, and allows for the circuit C to implicitly use randomness. In such a case, for f_ρ as before, a circuit with C with $m(\rho) + r(\rho)$ input bits that computes an output $x \in \{0,1\}^{n(\rho)}$, and a value $\epsilon' > 0$, define $\langle f_\rho, C, \epsilon' \rangle \in R_{\mathbf{OWF}}$ if and only if $\epsilon \geq \epsilon'$, where ϵ is the probability over uniform $z \in \{0,1\}^{r(\rho)}$ and $x \in \{0,1\}^{n(\rho)}$ that $C(f_\rho(x), z)$ outputs an $x' \in f_\rho^{-1}(f_\rho(x))$.

[3] For example, by the canonical encoding of an efficient machine as in the Cook-Levin Theorem.

2.4 Fully Black-Box Cryptographic Constructions

Finally, we bring the standard definition of a fixed-parameter fully black-box construction of a primitive $\mathbf{Q}$ from a primitive $\mathbf{P}$, which is usually implicit in the literature. The construction algorithm G is an efficient uniform oracle algorithm and the security reduction R is an efficient uniform two-oracle algorithm. For every security parameter ρ and a function $f_\rho : \{0,1\}^{n(\rho)} \to \{0,1\}^{n'(\rho)}$, G's output on $(1^\rho, n, n')$ is an (n, n')-oracle circuit $g_\rho^{(?)}$ such that $\{g_\rho^{(f_\rho)}\}_{\rho\in\mathbb{N}^+}$ implements $\mathbf{Q}$. The reduction algorithm works as follows: For a security parameter ρ and f as before, and additionally a breaker circuit $B : \{0,1\}^{b(\rho)} \to \{0,1\}^{b'(\rho)}$, the reduction R on input $(1^\rho, n, n', b, b')$ outputs an (n, n', b, b')-two-oracle circuit $R_\rho^{(?,?)}$. The security property property requires that indeed the series of circuits $\{R_\rho^{(B_\rho,f_\rho)}\}_{\rho\in\mathbb{N}^+}$ $\mathbf{P}$-breaks f. We emphasize that the vast majority (if not all) of the constructions of primitives from a one-way function found in the literature are in fact fixed-parameter fully black-box constructions. Formally:

Definition 3 (fixed-parameter fully black-box construction of $\mathbf{Q}$ from $\mathbf{P}$)**.** *An efficient uniform oracle algorithm G and an efficient uniform two oracle algorithm R are a* fixed-parameter fully-BB construction *of a primitive $\mathbf{Q} = \langle F_\mathbf{Q}, R_\mathbf{Q}\rangle$ from a primitive $\mathbf{P} = \langle F_\mathbf{P}, R_\mathbf{P}\rangle$ if for every $f \in F_\mathbf{P}$:*

1. **(correctness)** *$G^{(f)}$ implements $f' \in F_\mathbf{Q}$.*
2. **(security)** *For every algorithm B: If B $\mathbf{Q}$-breaks $G^{(f)}$ then $R^{(B,f)}$ $\mathbf{P}$-breaks f.*

For a super-polynomial security function $s = s(\rho)$ (e.g., $s(\rho) = 2^{\sqrt{\rho}}$), the following definition of a fully black-box construction is significantly weaker than the standard one in the following three aspects: First, it requires that reduction only mildly breaks the one-way property of the function f (whenever the breaker breaks the constructed primitive in the standard polynomial sense). Second, the reduction algorithm does not have to be efficient or uniform (but the non-uniformity is limited to an advice of length s). Lastly, it allows the reduction to make s calls to its oracles[4].

Definition 4 (s-weak fixed-parameter fully black-box construction of $\mathbf{Q}$ from $\mathbf{P}$)**.** *A uniform oracle algorithm G and an s-non-uniform two oracle algorithm R are an s-*weak fixed-parameter fully-BB construction *of a primitive $\mathbf{Q} = \langle F_\mathbf{Q}, R_\mathbf{Q}\rangle$ from a primitive $\mathbf{P} = \langle F_\mathbf{P}, R_\mathbf{P}\rangle$ if for every $f \in F_\mathbf{P}$:*

1. **(correctness)** *$G^{(f)}$ implements an $f' \in F_\mathbf{Q}$.*
2. **(security)** *For every non-uniform algorithm B: If B $\mathbf{Q}$-breaks $G^{(f)}$ then $R^{[B,f]}$ $(\mathbf{P}, 1/s)$-breaks f.*

2.5 Random Permutations and Regular Functions

Let n and i be two integers such that $0 \le i \le n$. We denote the set of all permutations on $\{0,1\}^n$ by $\mathcal{P}_n$. Let $\mathcal{X}, \mathcal{Y}$ be sets. We denote by $(\mathcal{X} \to \mathcal{Y})$ the set of all functions from $\mathcal{X}$ to $\mathcal{Y}$. A function $f : \mathcal{X} \to \mathcal{Y}$ is regular if $|\{x' : f(x) = f(x')\}|$ is constant for all $x \in \mathcal{X}$. A family of functions $f = \{f_\rho\}_{\rho\in\mathbb{N}^+}$ is a regular function if for every ρ the function f_ρ is regular. We denote by $\mathcal{R}_{n,i}$ the set of all regular functions from $\{0,1\}^n$ to itself such that the image of f contains 2^i values. E.g., $\mathcal{R}_{n,n} = \mathcal{P}_n$ is the set of all permutations, and $\mathcal{R}_{n,0}$ is the set of all constant functions.

[4]In Definition 3 the limitation on the number of queries made to the oracles is implicit as R is an efficient algorithm, and so its output circuit has at most a polynomially number of oracle gates.

2.6 Bending a Function and Image Adaptation

It will be useful for us to compare the run of a circuit with oracle access to a function f to a run that is identical except that the output of one specific value is altered.

For a fixed function $f : \{0,1\}^n \to \{0,1\}^n$ and $y', y'' \in \{0,1\}^n$, set

$$f_{(y',y'')}(x) \stackrel{\text{def}}{=} \begin{cases} y'' & \text{if } f(x) = y' \\ f(x) & \text{otherwise.} \end{cases}$$

Similarly, for two fixed functions $f, f' : \{0,1\}^n \to \{0,1\}^n$ and a set $S \subset \{0,1\}^n$, we define the image adaptation [5] of f to f' on S to be the function

$$f_{(S,f')}(x) \stackrel{\text{def}}{=} \begin{cases} f'(x) & \text{if } x \in f^{-1}(f(S)) \\ f(x) & \text{otherwise.} \end{cases}$$

3 A General Theorem for Proving Strong Black-Box Separations

3.1 Deterministic Parametrized Oracles and Local Sets

The following definition allows to model general parameterized oracles, that is, oracles that, for any function f from some set of functions and any q from some query domain, return a value a from some answer set. We observe that many of the oracles used for black-box separations found in the literature could be described in such a way.

Let $\mathcal{X}, \mathcal{Y}, \mathcal{D}$ and $\mathcal{R}$ be sets. A deterministic parametrized oracle for a class of functions $(\mathcal{X} \to \mathcal{Y})$ is an indexed collection $\mathcal{O} = \{\mathcal{O}_f\}_{f \in \mathcal{X} \to \mathcal{Y}}$, where $\mathcal{O}_f : \mathcal{D} \to \mathcal{R}$. We call f, $\mathcal{D}$, and $\mathcal{R}$ the function parameter, the domain, and the range of the oracle, respectively.

Our first example of a deterministic parametrized oracle is the evaluation oracle $\mathcal{E}$ for functions on $\{0,1\}^n$, which on a query q returns the evaluation of f on q. In this case we have that $\mathcal{X} = \mathcal{Y} = \mathcal{D} = \mathcal{R} = \{0,1\}^n$ and $\mathcal{E}_f(q) \stackrel{\text{def}}{=} f(q)$.

The next two definitions capture two important local properties of parametrized oracles. We believe that they are natural and observe that many of the oracles devised for separation results satisfy them.

Intuitively, a determining set is an indexed collection of sets that determine the output of the oracle for every function f and query q in the following sense: If for two functions f and f' it holds that their corresponding oracle outputs differ for some q, then for one of them (f or f') it holds that the local change of an image adaptation of one of the functions to agree with that of the other on its determining set changes the output of the oracle. Formally:

Definition 5. *Let $\mathcal{O}$ be a deterministic parametrized oracle. A* determining set *$\mathcal{I}^{\mathcal{O}}$ for a class of functions $\mathcal{F} \subset (\mathcal{X} \to \mathcal{Y})$ is an indexed collection $\{\mathcal{I}^{\mathcal{O}}_{f,q}\}_{f \in \mathcal{F}, q \in \mathcal{D}}$ of subsets of $\mathcal{X}$, such that for every $f, f' \in \mathcal{F}$ and every query $q \in \mathcal{D}$: If $\mathcal{O}_f(q) \neq \mathcal{O}_{f'}(q)$, then it holds that either the image adaptation of f to f' on $\mathcal{I}^{\mathcal{O}}_{f',q}$ changes $\mathcal{O}_f(q)$ (i.e., $\mathcal{O}_{f_{\left(\mathcal{I}^{\mathcal{O}}_{f',q}, f'\right)}}(q) \neq \mathcal{O}_f(q)$), or the image adaptation of f' to f on $\mathcal{I}^{\mathcal{O}}_{f,q}$ changes $\mathcal{O}_{f'}(q)$. $\mathcal{I}^{\mathcal{O}}$ is a t-*determining set *if for every function $f \in \mathcal{F}$ and query $q \in \mathcal{D}$ it holds that $\left|\mathcal{I}^{\mathcal{O}}_{f,q}\right| \leq t$.*

[5] We mention that if f is a permutation, the condition $f(x) = y$ can be replaced by $x = f^{-1}(y)$, and similarly for $f_{(S,f')}$ check whether $x \in S$, which is what one may expect initially from such a definition.

In the example of the evaluation oracle, we observe that it has a 1-determining set. Indeed, setting $\mathcal{I}^{\mathcal{E}}_{f,q} \stackrel{\text{def}}{=} \{q\}$ satisfies the required definition, since if for any $f, f' \in \{0,1\}^n \to \{0,1\}^n$ and $x \in \{0,1\}^n$ for which $f(x) \neq f'(x)$ it holds that $f_{(\{x\},f')}(x) = f'(x) \neq f(x)$.

Consider an oracle $\mathcal{O}$ with a determining set $\mathcal{I}^{\mathcal{O}}$ for some class of permutations $\mathcal{F}$. Fix $f, f' \in \mathcal{F}$ and $q \in \mathcal{D}$. The following two propositions are immediate from the definition of determining sets:

Proposition 6. *If $\mathcal{O}_f(q) \neq \mathcal{O}_{f'}(q)$ and $f(x) = f'(x)$ for all $x \in \mathcal{I}^{\mathcal{O}}_{f',q}$ (in this case we say that f agrees with f' on $\mathcal{I}^{\mathcal{O}}_{f',q}$), then adapting f' to agree with f on $\mathcal{I}^{\mathcal{O}}_{f,q}$ changes $\mathcal{O}_{f'}(q)$.*

Proposition 7. *If for all $x \in \mathcal{I}^{\mathcal{O}}_{f,q} \cup \mathcal{I}^{\mathcal{O}}_{f',q}$ it holds that $f(x) = f'(x)$ (in this case we say that the functions* agree *on their determining sets), then $\mathcal{O}_f(q) = \mathcal{O}_{f'}(q)$.*

Proposition 7 establishes that determining sets indeed determine the output of the oracle in the following sense: If we know the value $\mathcal{O}_f(q)$ for a query q and a function f, and, moreover, we know that functions f', f agree on their determining sets for q, then this information already determines for us the value $\mathcal{O}_{f'}(q)$.

The next local property of an oracle captures the fact that it is in some sense "stable". For a function f and query q as before, and a value y in the image set of f, a bending set for f, q, and y is a set of all potentially "sensitive" y' values: For any value y which is not in the image of f on its determining set, and for any value y' which is not in the bending set, the oracle's answer to query q does not change for the local adaptations of f from y' to y. That is, it holds that $\mathcal{O}_{f_{(y',y)}}(q) = \mathcal{O}_f(q)$. Formally:

Definition 8. *Let $\mathcal{O}$ be a deterministic parametrized oracle. A* bending set *$\mathcal{B}^{\mathcal{O}}$ for $\mathcal{F}$ is an indexed collection $\{\mathcal{B}^{\mathcal{O}}_{f,q,y}\}_{f\in\mathcal{F},q\in\mathcal{D},y\in\mathcal{Y}}$ of subsets of $\mathcal{Y}$, such that for every function $f \in \mathcal{F}$, query $q \in \mathcal{D}$, for every target image $y \in \mathcal{Y}$, and for every source image $y' \notin \mathcal{B}^{\mathcal{O}}_{f,q,y}$, it holds that $\mathcal{O}_f(q) = \mathcal{O}_{f_{(y',y)}}(q)$. We say that $\mathcal{B}^{\mathcal{O}}$ is a t-*bending set *if for every function $f \in \mathcal{F}$, query $q \in \mathcal{D}$ and $y \in \mathcal{Y}$ it holds that $|\mathcal{B}^{\mathcal{O}}_{f,q,y}| \leq t$.*

For the example of the evaluation oracle, we observe that it also has a 1-bending set. Setting $\mathcal{B}^{\mathcal{E}}_{f,q,y} \stackrel{\text{def}}{=} \{f(q)\}$ (for the relevant f, q and y) satisfies the required definition. Indeed, for any $y' \neq f(q)$ and $y'' \in \mathcal{Y}$, it holds that $\mathcal{E}_{f_{(y',y'')}}(q) = f_{(y',y'')}(q) = f(q) = \mathcal{E}_f(q)$.

Finally, a deterministic parametrized algorithm $\mathcal{O}$ is t-stable for a class of functions $\mathcal{F}$ if there exist $(\mathcal{I}^{\mathcal{O}}, \mathcal{B}^{\mathcal{O}})$ that are a t-determining set and a t-bending set for $\mathcal{F}$, respectively, and at least one of them is not empty.

We note that determining and bending sets always exist unconditionally (just choose the entire domain and range of f, for every determining and bending set, respectively). The challange is finding an oracle that allows to break a primitive and at the same time is t-stable.

3.1.1 Exhaustive-search oracles.

We identify that all the parametrized oracles in the literature are in fact of a special type, which we call exhaustive-search oracles. We say that an oracle is an exhaustive-search oracle if there is an oracle circuit $\Phi^{(?)} : \mathcal{D} \times \mathcal{R} \to \{0,1\}$ (called a predicate) such that for every function f and query q the computation of $\mathcal{O}_f(q)$ can be computed by a loop (according to an understood enumeration) over the values $v \in \mathcal{R}$, where in each step the current value v is checked to satisfy a predicate $\Phi^{(f)}(q, v)$. If the predicate is satisfied, i.e., $\Phi^{(f)}(q, v) = 1$, the algorithm outputs v, and if no such v exists it returns a special bottom value $\perp$.

Formally, a deterministic parameterized oracle $\mathcal{O}$ is an exhaustive-search oracle if there exists Φ (as before) such that following algorithm outputs $\mathcal{O}_f(q)$ for every function parameter f and every query $q \in \mathcal{D}$:

Exhaustive Search Algorithm $\mathcal{O}_f(q)$ (on function $f \in \mathcal{X} \to \mathcal{Y}$ and input $q \in \mathcal{D}$)

---

for all $v \in \mathcal{R}$ **do**
. **if** $\Phi^{(f)}(q, v) = 1$ **then**
. **return** v
return $\perp$

Observe that the range of an exhaustive-search algorithm is the set $\mathcal{R} \cup \{\perp\}$. Of special interest are exhaustive-search oracles that make relatively few queries to their oracles (e.g., a polynomial number of queries). We note that most parameterized oracles in the literature are in fact of this type. The following lemma shows that an exhaustive-search oracle $\mathcal{O} : \mathcal{D} \to \mathcal{R} \cup \{\perp\}$ for which Φ makes t queries has a t-determining set.

Lemma 9. *Let $\mathcal{O}$ be an exhaustive-search oracle with predicate $\Phi^{(?)} : \mathcal{D} \times \mathcal{R} \to \{0,1\}$ that makes t queries to its oracle. Denote by $X_\Phi^{(f)}(q, v)$ the list of queries issued by the predicate during the evaluation of $\Phi^{(f)}(q, v)$. Then*

$$\mathcal{I}_{f,q}^{\mathcal{O}} \stackrel{\text{def}}{=} \begin{cases} X_\Phi^{(f)}(q, v) & v \text{ is the first value for which } \Phi^{(f)}(q, v) = 1 \\ X_\Phi^{(f)}(q, v_{\text{last}}) & \text{if } \mathcal{O}_f(q) = \perp, \text{ where } v_{\text{last}} \text{ is the last value in the enumeration of } \mathcal{R} \end{cases}$$

is a t-determining set for $\mathcal{O}$.

The proof follows by inspection of the algorithm and the definition of determining sets.

Proof: The bound on the size of the set is immediate and so we check that $\mathcal{I}_{f,q}^{\mathcal{O}}$ is indeed a determining set. Suppose that for two functions f, f' and a query q it holds that $\mathcal{O}_f(q) \neq \mathcal{O}_{f'}(q)$. It follows that for at least one of the functions, there is an iteration v for which the predicate $\Phi^{(f)}(q, v)$ holds (otherwise both return $\perp$). W.l.o.g. assume that this happens for f, i.e., that $\mathcal{O}_f(q) = v$, and that v is the minimal of the returned values, i.e., that $\mathcal{O}_{f'}(q) = v'$, where either $v' > v$ or $v' = \perp$. Now, observe that after adapting f' to agree with f on $\mathcal{I}_{f,q}^{\mathcal{O}} = X_\Phi(f, q)$ it holds that with the adapted function oracle the predicate circuit returns v (or some value that appears in the enumeration before v), as the answer of the predicate depends only on the answers of the function to the query set. ■

The choice of v_{last} (for the case $\mathcal{O}_f(q) = \perp$) is somewhat arbitrary, but it will be useful for us to have the queries involved with q on some arbitrary evaluation of the predicate as part of the bending set. We note that this is not needed for the proof of the lemma.

3.2 A t-Stable Oracle $\mathcal{O}_f$ Inverts Only a Few Functions

The next lemma, which first appeared in [GGKT05] and was subsequently adapted to many other separation results, e.g., [HHRS07, RS10, HS12], establishes an information-theoretic bound on the number of functions an oracle-aided algorithm can invert from a set $\mathcal{F}$ if the oracle is t-stable for $\mathcal{F}$. Essentially, it shows that given an oracle circuit $A^{(?)}$ with access to such an oracle $\mathcal{O}$, it is possible to encode a function $f \in \mathcal{F}$ that A inverts well using significantly fewer bits than $\log(|\mathcal{F}|)$, such that f can still be fully reconstructed, or equivallently, that the encoding is injective.

Lemma 10 (Encoding Lemma)**.** *Let $A^{(?)}$ be an oracle circuit making at most c calls to its oracle, and let $\mathcal{O} = \{\mathcal{O}_f\}_{f\in\{0,1\}^n\to\{0,1\}^n}$ be a deterministic parameterized oracle such that for a class of permutations $\mathcal{F} \subseteq \mathcal{P}_n$ it is t-stable with sets $(\mathcal{I}^{\mathcal{O}}, \mathcal{B}^{\mathcal{O}})$. Then, for at most $d_n = d_n(c,t) = \left(\binom{2^n}{b}\right)^2 \cdot ((2^n - b)!)$, where $b \stackrel{\text{def}}{=} \frac{2^n}{3\cdot c^2\cdot t}$, of the permutations f in $\mathcal{F}$, it holds that $\Pr_{x\xleftarrow{r}\{0,1\}^n}\left[A^{\mathcal{O}_f}(f(x)) = x\right] > \frac{1}{c}$.*

The proof is a generalized version of the encoding technique of [GGKT05].

Proof: We describe an injective canonical mapping from the set of all functions $f \in \mathcal{F}$ which $A^{\mathcal{O}_f}$ inverts well (i.e., inverts more than a $\frac{1}{c}$-fraction of the inputs to f) into a small set.

Denote by Queries$(A^{\mathcal{O}_f}(y))$ the list of at most c of queries made by A to $\mathcal{O}_f$ during its computation on input y. Consider the following algorithm:

Algorithm Buildimage(f):

1. $S := \emptyset$
2. $T := \{y : A^{\mathcal{O}_f}(y) \in f^{-1}(y)\}$
3. **while** $T \neq \emptyset$ **do** :
4. $\quad y' := \min_{y\in T}$
5. $\quad S := S \cup \{y'\}$
6. $\quad R := f\left(\bigcup_{q\in\text{Queries}\left(A^{\mathcal{O}_f}(y')\right)} \mathcal{I}^{\mathcal{O}}_{f,q}\right)$
7. $\quad R := R \cup \bigcup_{q\in\text{Queries}\left(A^{\mathcal{O}_f}(y')\right)} \mathcal{B}^{\mathcal{O}}_{f,q,y'}$
8. $\quad R := R \cup \{y'\}$
9. $\quad T := T \setminus R$
10. **end**

Define the mapping $\mathcal{M}$ that maps every function f which $A^{\mathcal{O}_f}$ inverts well to a list of its action on $f^{-1}(\mathcal{Y}\setminus S)$ as a list of pairs, i.e., $f \mapsto \big(x_1, f(x_1), x_2, f(x_2), \ldots, x_{|\mathcal{Y}\setminus S|}, f(x_{|\mathcal{Y}\setminus S|})\big)$, where $x_1 < x_2 < \cdots < x_{|\mathcal{Y}\setminus S|}$. We show that $\mathcal{M}$ is injective. Suppose that for $f_1, f_2 \in \mathcal{F}$, $A^{\mathcal{O}_{f_1}}$ and $A^{\mathcal{O}_{f_2}}$ invert well f_1 and f_2, respectively.

Claim 11. *If $\mathcal{M}(f_1) = \mathcal{M}(f_2)$ then $f_1 = f_2$.*

Proof: Assume towards contradiction that $f_1 \neq f_2$. We first observe that both functions agree on the sets $\{y : A^{\mathcal{O}_{f_1}}(y) \notin f_1^{-1}(y)\}$ and on $\{y : A^{\mathcal{O}_{f_2}}(y) \notin f_2^{-1}(y)\ \}$. This holds as these images are explicitly given by $\mathcal{M}(\cdot)$ since they are in $\mathcal{Y} \setminus S$. Therefore, there must be a value y, which they both invert, such that

$$A^{\mathcal{O}_{f_1}}(y) = {f_1}^{-1}(y) \neq {f_2}^{-1}(y) = A^{\mathcal{O}_{f_2}}(y)\ . \tag{1}$$

Let y be the lexicographically first element for which (1) holds. Define $x_1 \stackrel{\text{def}}{=} A^{\mathcal{O}_{f_1}}(y)$ and $x_2 \stackrel{\text{def}}{=} A^{\mathcal{O}_{f_2}}(y)$.

Proposition 12. *For both f_1 and f_2 there is an iteration during the run of the algorithm* Buildimage*(f_1) (resp.,* Buildimage*(f_2)) during which y is added to S.*

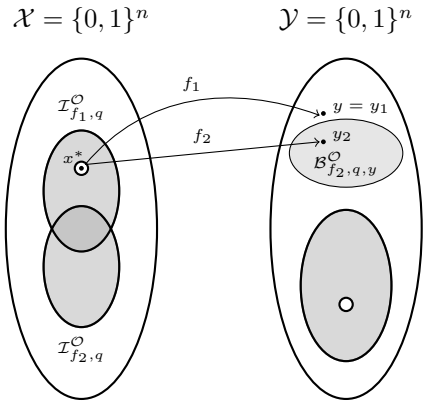

Figure 1: An illustration of the proof of Lemma 10. The functions f_1 and f_2 agree on $\mathcal{I}^{\mathcal{O}}_{f_1,q} \setminus \{x^*\}$, $f_1(x^*) = y = y_1 \neq y_2 = f_2(x^*)$ and adapting f_2 to f_1 on $\mathcal{I}^{\mathcal{O}}_{f_1,q}$ changes $\mathcal{O}_{f_2}(q)$.

Proof: Otherwise, we reach a contradiction to the assumption that $\mathcal{M}(f_1) = \mathcal{M}(f_2)$ as the value of y is explicitly given by $\mathcal{M}$. ■

Any difference in the computation of $A^{\mathcal{O}_{f_1}}(y)$ and $A^{\mathcal{O}_{f_2}}(y)$ may only stem from different answers to some oracle query. Let q be an oracle query on which the computations differ, that is, for a query q made in both computations and $a_1 \neq a_2 \in \mathcal{R}$ we have:

$$a_1 = \mathcal{O}_{f_1}(q) \neq \mathcal{O}_{f_2}(q) = a_2 \ . \tag{2}$$

By Proposition 7 (for f_1, f_2 and their corresponding determining sets $\mathcal{I}^{\mathcal{O}}$), we have that there exists a value $x^* \in \mathcal{I}^{\mathcal{O}}_{f_1,q} \cup \mathcal{I}^{\mathcal{O}}_{f_2,q}$ for which

$$y_1 = f_1(x^*) \neq f_2(x^*) = y_2 \ , \tag{3}$$

as otherwise, (2) cannot hold.

The next two propositions yield that for such an $x^* \in \mathcal{I}^{\mathcal{O}}_{f_1,q} \cup \mathcal{I}^{\mathcal{O}}_{f_2,q}$ if $x^* \in \mathcal{I}^{\mathcal{O}}_{f_1,q}$ (resp., $x^* \in \mathcal{I}^{\mathcal{O}}_{f_2,q}$) it holds that $f_1(x^*) = y$ (resp., $f_2(x^*) = y$). In particular, combining this with Equation (3) and the fact that f_1 and f_2 are permutations asserts that in each of the sets $\mathcal{I}^{\mathcal{O}}_{f_1,q}$ and $\mathcal{I}^{\mathcal{O}}_{f_2,q}$ there is at most one such x^* . Let us assume that $x^* \in \mathcal{I}^{\mathcal{O}}_{f_1,q}$.

Proposition 13. *It holds that $y_1 \leq y$.*

Proof: Assuming otherwise (that $y_1 > y$), consider the execution of the algorithm Buildimage(f_1). In such a case, recall that by Proposition 12 y is added to the set S, and in line (6) of the algorithm y_1 is added to R (as it is in the image of the determining set for query q). Subsequently, in line (9) y_1 is removed from T. Since at this point of the execution of Bulidimage(f_1) y_1 is not in S, it is never added to S, hence $\mathcal{M}(f_1)$ contains (x^*, y_1), which contradicts $\mathcal{M}(f_1) = \mathcal{M}(f_2)$. ■

Proposition 14. *It holds that $y_1 \geq y$ and $y_2 \geq y$.*

Proof: We prove that $y_1 \geq y$ and observe that the proof for $y_2 \geq y$ follows from symmetry. Assuming otherwise (that $y_1 < y$), we consider two cases: If (x^*, y_1) appears in $\mathcal{M}(f_1)$ or $(f_2^{-1}(y_1), y_1)$ appears in $\mathcal{M}(f_2)$, then since both are permutations, combining with (3) we reach

a contradiction for $\mathcal{M}(f_1) = \mathcal{M}(f_2)$. Otherwise, both $A^{\mathcal{O}_{f_1}}(y_1) = x^*$ and $A^{\mathcal{O}_{f_2}}(y_1) = f_2^{-1}(y_1)$ must hold, and moreover y_1 is added to S at each of the corresponding Buildimage runs. Observe now that this contradicts the minimality of y. ∎

It follows that: For any element x in $\mathcal{I}^{\mathcal{O}}_{f_1,q} \cup \mathcal{I}^{\mathcal{O}}_{f_2,q}$ on which f_2 and f_1 do not agree it holds that $f_1(x) = y$ if $x \in \mathcal{I}^{\mathcal{O}}_{f_1,q}$ and that $f_2(x) = y$ if $x \in \mathcal{I}^{\mathcal{O}}_{f_2,q}$. Moreover, since f_1, f_2 are permutations, we know that there is at most one such element in each of the sets, and in at least one of them such an element exists.

Next, if there exists exactly one such element, then, as before, without loss of generality we assume that there exists $x^* \in \mathcal{I}^{\mathcal{O}}_{f_1,q}$ for which $f_1(x^*) = y$. If there are two elements, i.e., $x^* \in \mathcal{I}^{\mathcal{O}}_{f_1,q}$ and $x^{**} \in \mathcal{I}^{\mathcal{O}}_{f_2,q}$, then by the definition of determining sets we know that adapting one of the functions to agree with the other changes the value of the oracle on it.

In both cases we get that (without loss of generality) we may assume that adapting f_2 to agree with f_1 on $\mathcal{I}^{\mathcal{O}}_{f_1,q}$ changes $\mathcal{O}_{f_2}(q)$. Recall that if there is exactly one such element, this follows from Proposition 6.

Now, note that the image adaptation of f_2 to f_1 on $\mathcal{I}^{\mathcal{O}}_{f_1,q}$ is just $f_{2(y_2,y)}$ and therefore $\mathcal{O}_{f_{2(y_2,y)}}(q) \neq \mathcal{O}_{f_2}(q)$, and by defintion of the bending sets it must hold that $y_2 = f_2(x^*) \in \mathcal{B}^{\mathcal{O}}_{f_2,q,y}$. Observe that at step (4) of the execution of Buildimage(f_2), after y' is set to y from T, in line (7) y_2 is added to R (by the definition of the determining sets and in line (9) y_2 is removed from T. Propositions 13 and 14 establish that $y_2 > y = y_1$. Therefore, at this point of the execution y_2 is not in S and hence never added to S. Finally, this means that (x^*, y_2) appears in $\mathcal{M}(f_2)$ which contradicts $\mathcal{M}(f_1) = \mathcal{M}(f_2)$. ∎

Claim 15. *The set S generated by the algorithm Buildimage(f) contains at least b elements.*

Proof: We start by showing that for all the functions f which the circuit inverts well, the generated set S in algorithm Buildimage(f) is large. Let f be a function for which $\Pr_{x \xleftarrow{r} \mathcal{X}}[A^{O_f} \text{ inverts } f(x)] > \frac{1}{c}$. By the definition of the set T at the begining of the run of Buildimage(f), T is of size at least $\frac{|\mathcal{X}|}{c}$. Observe that during each iteration of the algorithm one element is added to S and at most $(c \cdot t + c \cdot t + 1)$ elements are removed from T, where the first two summands correspond the size of the relevant bending set and determining set for each of the c potential calls to the oracle, and the third to the element y (lines 6 - 9 in the algorithm). It follows that when the algorithm terminates, the set S contains at least $\frac{|\mathcal{X}|}{c^2 \cdot (2 \cdot t + 1)} > \frac{2^n}{3 \cdot c^2 \cdot t} = b$ elements. ∎

We note that it is possible to encode $\mathcal{M}(f)$ by describing the set S, the set $\mathcal{Y} \setminus f(S)$, and an ordering on the set $\mathcal{Y} \setminus f(S)$, which by what we have shown has size at most $2^n - b$. Therefore, the size of the range of the mapping $\mathcal{M}$ is at most $\left(\binom{2^n}{b}\right)^2 \cdot ((2^n - b)!)$. Combining this with the first claim, we infer that there are at most $\left(\binom{2^n}{b}\right)^2 \cdot ((2^n - b)!)$ such functions. The lemma is proved. ∎

Our next goal is to extend the encoding lemma to the case where $\mathcal{F} \subseteq \mathcal{R}_{n,i}$. We observe that the following process, which consists of 3 independent random choices, samples a uniform random f from from $\mathcal{R}_{n,i}$:

1. **Partition of the domain**: Sample a random uniform partition P of the domain into subsets of size 2^{n-i}. The partition P is chosen in an unordered manner, and then we use some order relation[6] ϕ on all the subsets of size 2^{n-i}. For $x \in P_i$, where $P_i \in P$, we define

[6] For example, the order relation induced by the minimal element of the sets. That is, $S_1 < S_2$ if and only if $\min_{x \in S_1} < \min_{x \in S_2}$.

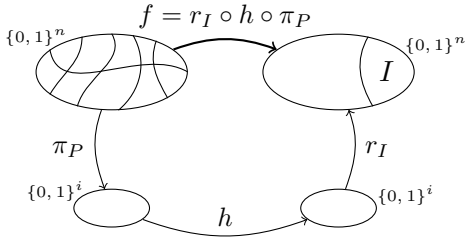

Figure 2: Decomposition of a regular function f: π_P is a regular function onto $\{0,1\}^i$ induced by a partition P of $\{0,1\}^n$, h is a permutation on $\{0,1\}^i$, and r_I is injective from $\{0,1\}^i$ into I.

$\pi_P : \{0,1\}^n \to \{0,1\}^i$ by $\pi_P(x)$ as the order function of P_i in P according to ϕ [7] .

2. **A permutation on** $\{0,1\}^i$: Sample a random permutation h from $\mathcal{P}_i$.
3. **Image determination**: Sample a set $I \subset \{0,1\}^n$ of size 2^i and set $r_I : \{0,1\}^i \to \{0,1\}^n$ as the inverse function of the lexicographical order function[8] of I.
4. Finally, set $f \stackrel{\text{def}}{=} r_I \circ h \circ \pi_P$.

Indeed, the reader can verify that the mapping $(P, h, I) \mapsto r_I \circ h \circ \pi_P$ is an injective function onto $\mathcal{R}_{n,i}$. In particular, it holds that $|\mathcal{R}_{n,i}| = \left(\frac{(2^n)!}{((2^{n-i})!)^{2^i} \cdot (2^i!))}\right) \cdot ((2^i)!) \cdot \binom{2^n}{2^i}$, where the factors in the product correspond to the number of partitions of $\{0,1\}^n$ to sets of size 2^i, the number of permutations on $\{0,1\}^i$ and the number of subsets of size 2^{n-i} from $\{0,1\}^n$, respectively.

For a fixed P and I (of the matching size), we denote by $\mathcal{R}_{n,i}(P, I)$ the subset of $R_{n,i}$ with partition P and image set I. The sampling process establishes a natural bijection $f : \mathcal{P}_i \to \mathcal{R}_{n,i}(P, I)$, where $f[h] \stackrel{\text{def}}{=} r_I \circ h \circ \pi_P$. Similarly, we denote the inverse transformation by $h[f]$. We extend the definition of $f[h]$ to the entire set $\{0,1\}^i \to \{0,1\}^i$ with $f[h] : (\{0,1\}^i \to \{0,1\}^i) \to \mathcal{F}_{n,i}(P, I)$ given by $f[h] \stackrel{\text{def}}{=} r_I \circ h \circ \pi_P$, where $\mathcal{F}_{n,i}(P, I) \stackrel{\text{def}}{=} \{\pi_P \circ h \circ r_I\}_{h \in \{0,1\}^i \to \{0,1\}^i}$. That is, $\mathcal{F}_{n,i}(P, I)$ is the set of all functions f on n bits with image set contained in I, such that if for $x, x' \in \{0,1\}^n$ it holds that $\pi_P(x) = \pi_P(x')$ then $f(x) = f(x')$.

Therefore, it is not surprising that any oracle circuit $A^{(?)}$ with access to a t-stable deterministic parameterized oracle for a class of functions $\mathcal{F} \subset \mathcal{R}_{n,i}(P, I)$ does not invert many functions from $\mathcal{F}$. This is formalized in the next lemma:

Lemma 16. *Let $n \geq i \geq 0$, P be a fixed partition of $\{0,1\}^n$ into subsets of size 2^{n-i}, and $I \subset \{0,1\}^n$ an image set of size 2^i. Let $A^{(?)}$ be an oracle circuit making at most c calls to its oracle, and let $\mathcal{O} = \{\mathcal{O}_f\}_{f \in \mathcal{F}_{n,i}(P,I)}$ be a deterministic parameterized oracle such that for a class of functions $\mathcal{F} \subseteq \mathcal{R}_{n,i}(P, I)$ it is t-stable with sets $(\mathcal{I}^{\mathcal{O}}, \mathcal{B}^{\mathcal{O}})$. Then for at most $d_i(c, t)$ of the functions $f \in \mathcal{F}$, where d_i is as[9] in Lemma 10, it holds that $\Pr_{x \xleftarrow{r} \{0,1\}^n}\left[A^{\mathcal{O}_f}(f(x)) \in f^{-1}(f(x))\right] > \frac{1}{c}$.*

The proof works by reduction to the setting of Lemma 10. To this end, we define a deterministic parameterized oracle $\tilde{\mathcal{O}}$, a set $\tilde{\mathcal{F}} \subset \mathcal{P}_i$ of permutations with corresponding determining and bending sets, and show how to use $A^{(?)}$ to derive an algorithm $\tilde{A}^{(?)}$ that inverts many permutations from $\tilde{\mathcal{F}}$ which contradicts Lemma 10.

[7] Continuing the example, for the relation described in Footnote 6, we always have that for all $x \in S$, where $S \in P$ contains the all zero string 0^n, that $\pi_P(x)$ is the all zero string 0^i.

[8] Of course, any order relation would work just as well.

[9] Note that here we substitute n for i.

Proof: Towards contradiction, let $\mathcal{O}$ be a deterministic parametrized oracle, where $\mathcal{O}_f : \mathcal{D} \to \mathcal{R}$ defined for all $f \in \{0,1\}^n \to I$, such that for $\mathcal{F} \subset \mathcal{R}_{n,i}(P,I)$ it is t-stable with sets $(\mathcal{I}^{\mathcal{O}}, \mathcal{B}^{\mathcal{O}})$, and $A^{(?)}$ an oracle circuit that inverts well (as before inverts well f means inverting at least a $\frac{1}{c}$-fraction of the inputs to f) more than d_i functions from $\mathcal{F}$.

We define $\tilde{\mathcal{O}} = \{\tilde{\mathcal{O}}_h\}_{h \in \{0,1\}^i \to \{0,1\}^i}$, a deterministic parameterized oracle, where $\tilde{\mathcal{O}}_h : \mathcal{D} \to \mathcal{R}$ is given by $\tilde{\mathcal{O}}_h(q) \stackrel{\text{def}}{=} \mathcal{O}_{f[h]}(q)$, $\tilde{\mathcal{F}} \stackrel{\text{def}}{=} \{h[f] : f \in \mathcal{F}\}$, and for every $h \in \tilde{\mathcal{F}}$, $q \in \mathcal{D}$ and every $y' \in \{0,1\}^i$ set $\mathcal{I}^{\tilde{\mathcal{O}}}_{h,q} \stackrel{\text{def}}{=} \pi_P\left(\mathcal{I}^{\mathcal{O}}_{f[h],q}\right)$ and $\mathcal{B}^{\tilde{\mathcal{O}}}_{h,q,y'} \stackrel{\text{def}}{=} {r_I}^{-1}\left(\mathcal{B}^{\mathcal{O}}_{f[h],q,r_I(y')}\right)$.

By the construction, the definition of determining sets, bending sets and the definition of image adapting a function to agree with another function on a set, one can check that the new constructed sets are indeed determining sets and bending sets for $\tilde{\mathcal{O}}$. Moreover, as r_I is an injective function it holds that $\mathcal{B}^{\tilde{\mathcal{O}}}$ is a t-bending set for $\tilde{\mathcal{F}}$. For every $h \in \{0,1\}^i$ and $q \in \mathcal{D}$ it holds that

$$\left|\pi_P(\mathcal{I}^{\mathcal{O}}_{f,q})\right| \leq \left|\mathcal{I}^{\mathcal{O}}_{f,q}\right| \leq t,$$

and so $\mathcal{I}^{\tilde{\mathcal{O}}}$ is a t-determining set[10]. Therefore, the oracle is t-stable for $\tilde{\mathcal{F}}$.

We construct a circuit $\tilde{A}^{(?)}$ such that for every function $h \in \tilde{\mathcal{F}}$ and and every image $y' \in \{0,1\}^i$ it holds that $\tilde{A}^{(\tilde{\mathcal{O}}_h)}(y') = h^{-1}(y')$ whenever $A^{(\mathcal{O}_{f[h]})}(r_I(y')) \in (f[h])^{-1}(r_I(y'))$.

The circuit $\tilde{A}$ stores the complete descriptions of π_P and r_I. On input $y' \in \{0,1\}^i$ it computes $y \stackrel{\text{def}}{=} r_I(y')$ and simulates the run of $A^{(\mathcal{O}_f)}(y)$. Whenever in the simulation A issuses a query q to its oracle, $\tilde{A}$ queries its oracle and uses the answer as an answer for the simulation. When the simulation terminates with output $x = A(y)$, $\tilde{A}$ outputs $x' = \pi_P(x)$.

By the construction of the oracle $\tilde{\mathcal{O}}$ it follows that $\tilde{A}$ inverts well exactly the same number of functions as A, and so we reach a contradiction to Lemma 10. ∎

We next show that any two oracle circuit making few queries to two deterministic parametrized oracles, where both oracles are t-stable for some set of functions $\mathcal{F}$, does not invert well many functions from $\mathcal{F}$.

Lemma 17. *Let $n \geq i \geq 0$, P be a fixed partition of $\{0,1\}^n$ into subsets of size 2^{n-i}, and $I \subset \{0,1\}^n$ an image set of size 2^i. Let $A^{(?,?)}$ be an oracle circuit making a total number of at most c calls to its oracles, and let $\mathcal{O}^{(0)} = \{\mathcal{O}^{(0)}_f\}_{f \in \mathcal{F}_{n,i}(P,I)}$ and $\mathcal{O}^{(1)} = \{\mathcal{O}^{(1)}_f\}_{f \in \mathcal{F}_{n,i}(P,I)}$ be deterministic parameterized oracles such that for a class of functions $\mathcal{F} \subseteq \mathcal{R}_{n,i}(P,I)$ both oracles are t-stable with sets $(\mathcal{I}^{\mathcal{O}^{(0)}}, \mathcal{B}^{\mathcal{O}^{(0)}})$ and $(\mathcal{I}^{\mathcal{O}^{(1)}}, \mathcal{B}^{\mathcal{O}^{(1)}})$, respectively. Then for at most $d_i(c,t)$ of the functions $f \in \mathcal{F}$, where d_i is as in Lemma 10, it holds that $\Pr_{x \xleftarrow{r} \{0,1\}^n}\left[A^{\mathcal{O}^{(0)}_f, \mathcal{O}^{(1)}_f} \in f^{-1}(f(x))\right] > \frac{1}{c}$.*

The proof is straight forward and very similar to the proof of Lemma 16. We show how to reduce this case to the setting of Lemma 16. In this case we define a new parametrized oracle $\tilde{\mathcal{O}}_f$ that on query $q' = (b,q) \in (\{0\} \times \mathcal{D}^{(0)}) \cup (\{1\} \times \mathcal{D}^{(1)})$, where $b \in \{0,1\}$ and $\mathcal{O}^{(b)}_f : \mathcal{D}^{(b)} \to \mathcal{R}^{(b)}$, answers $\tilde{\mathcal{O}}_f(q') \stackrel{\text{def}}{=} \mathcal{O}^{(b)}_f(q)$. It follows readily that $\tilde{\mathcal{O}}$ is t-stable. As before, any oracle circuit $A^{(?,?)}$ that inverts well more that d_i functions from $\mathcal{F}$ when plugged with $\mathcal{O}^{(0)}$ and $\mathcal{O}^{(1)}$ can be used to construct a circuit $\tilde{A}^{(?)}$ that inverts well more than d_i functions when plugged with $\tilde{\mathcal{O}}$.

We are now ready to prove the main theorem of this section:

Theorem 18 (Black-Box Separation Factory)**.** *Let $s = s(\rho)$ be a security function, and $p = p(\rho)$ be a polynomial function. Let $(G,A) = (G^{(?)}, A^{(?,?)})$ be a uniform oracle algorithm and an s-non-uniform two-oracle algorithm, respectively. Let $\mathcal{F} = \{\mathcal{F}_\rho\}_{\rho>0}$, where $\mathcal{F}_\rho \subset \mathcal{R}_{n(\rho),i(\rho)}(P_\rho, I_\rho)$,*

[10]Note that in both inequalities an equality may hold.

be contained in $F_{\mathbf{OWF}}$*, and* $\mathcal{O} = \{\mathcal{O}_\rho\}_{\rho>0}$*, where* $\mathcal{O}_\rho = \{\mathcal{O}_{\rho,f}\}_{f\in\mathcal{F}_{n(\rho),i(\rho)}(P_\rho,I_\rho)}$*, such that for all large enough* ρ*:*

1. $\mathcal{O}_{\rho,f}$ $(\mathbf{Q}, \frac{1}{p(\rho)})$*-breaks* $g_\rho^{(f)}$ *for every* $f \in \mathcal{F}_\rho$*, where* $g_\rho^{(?)} \stackrel{\text{def}}{=} G(1^\rho, n, n')$*.*
2. $\mathcal{O}_\rho$ *is* t*-stable with sets* $(\mathcal{I}^{\mathcal{O}}, \mathcal{B}^{\mathcal{O}})$ *for* $\mathcal{F}_\rho$ *such that* $2^{s(\rho)} \cdot d_i(s(\rho), t) < |\mathcal{F}_\rho|$ *holds, where* d_i *is as in Lemma 10.*

Then (G, A) *is not an* s*-weak fixed-parameter fully black-box construction of* $\mathbf{Q}$ *from* $\mathbf{OWF}$*.*

Proof: Towards contradiction, assume that (G, R) is an s-non-uniform fixed-parameter fully black-box construction of $\mathbf{Q}$ from $\mathbf{OWF}$. We explain how to construct a family of functions f' that will be used to contradict the security assumption on our construction. As before, we set $\mathcal{E}_\rho$ is the evaluation oracle for $\mathcal{R}_{n(\rho),i(\rho)}$, and recall that we have shown that $\mathcal{E}_\rho$ is 1-stable for $\mathcal{R}_{n(\rho),i(\rho)}$. Combining this with the first condition and using Lemma 17 we have that for every fixed advice $a \in \{0,1\}^{s(\rho)}$, the two-oracle circuit $R_{\rho,a}^{(?,?)}$ inverts with probability greater than $\frac{1}{s(\rho)}$ at most $d_i(s(\rho), t)$ of the functions in $\mathcal{F}_\rho$. Therefore, by the union bound there are at most $2^{s(\rho)} \cdot d_i(s(\rho), t)$ functions $f \in \mathcal{F}_\rho$, for which there exists an advice a, such that the circuit inverts more than a $\frac{1}{s}$-fraction of the inputs to f. The condition on the size of $\mathcal{F}_\rho$ gives a a function $f'_\rho \in \mathcal{F}_\rho$, such that for every $a \in \{0,1\}^{s(\rho)}$ it holds that $R_{\rho,a_\rho}^{(\mathcal{O}_{f'_\rho}, \mathcal{E}_{f'_\rho})}$ does not $(\mathbf{OWF}, \frac{1}{s(\rho)})$-break f'_ρ. The first condition, which holds for all the functions in $\mathcal{F}_\rho$, gives that $\mathcal{O}_{\rho,f_\rho}$ $(\mathbf{Q}, \frac{1}{p(\rho)})$-breaks f'_ρ.

Now, set $f' = \{f'_\rho\}_{\rho\in\mathbb{N}^+}$ and $B = \{\mathcal{O}_{\rho,f'_\rho}\}_{\rho\in\mathbb{N}^+}$. By the fact that $\mathcal{F}$ is contained in $F_{\mathbf{OWF}}$, we have that $f' \in F_{\mathbf{OWF}}$. By our assumption on (R, G) it holds that $G^{(f')} \in \mathbf{Q}$, and by what we have shown, it also holds that B $\mathbf{Q}$-breaks $G^{(f')}$. Lastly, (again) by our construction of f', it holds that $R^{[B,f']}$ does not $(\mathbf{OWF}, \frac{1}{s})$-break $G^{(f')}$, which contradicts our assumption on (G, R). ■

4 A Lower Bound on the Number of Calls for a Fixed-Parameter Fully Black-Box Construction of UOWHF from OWF

In this section we prove our second main result, namely a lower bound on the number of calls made by the construction algorithm G in any fully black-box construction (G, R) of **UOWHF** from **OWF**. Our bound is achieved by showing a sequence of efficient fixed-parameter fully black-box constructions, where each primitive is constructed from the one that precedes it, and by proving the lower bound on the number of calls a construction makes on the last primitive. A diagram of the reduction sequence is depicted in Figure 3.

4.1 A Characterization of Universal One-Way Hash Functions

Loosely speaking, a universal one-way hash function is a keyed compressing function for which the probability that an adversary wins the following game is very small: First the adversary chooses a preimage v. Then a random key for the UOWHF is chosen. Finally, the adversary "wins" the game he finds a different preimage v' that maps to the same value under the chosen key. Formally:

Definition 19 (UOWHF)**.** *A* universal one-way hash function $h = \{h_\rho\}_{\rho\in\mathbb{N}^+}$ *is a family of uniformly efficiently computable keyed functions* $h_\rho : \{0,1\}^{\kappa(\rho)} \times \{0,1\}^{m(\rho)} \to \{0,1\}^{m'(\rho)}$ *with*

$m'(\rho) < m(\rho)$ *such that for any pair of efficient randomized algorithms* (B_1, B_2) *the function mapping* ρ *to*

$$\Pr_{\substack{(v,\sigma)\xleftarrow{r}B_1(\rho)\\ k\xleftarrow{r}\{0,1\}^{\kappa(\rho)}\\ v'\xleftarrow{r}B_2(k,v,\sigma)}}\left[h_\rho(k,v) = h_\rho(k,v') \wedge v \neq v'\right]$$

is negligible. The family h *is an* ℓ-bit compressing *UOWHF, where* $\ell = \ell(\rho)$*, if* $m(\rho) - m'(\rho) \geq \ell(\rho)$ *for all large enough* ρ*.*

The primitive $\mathbf{UOWHF} = (F_{\mathbf{UOWHF}}, R_{\mathbf{UOWHF}})$ is defined implicitly analogously to the way **OWF** was defined for one-way functions In this case, for every security parameter, the breaker (B_1, B_2) can be modeled as one combined circuit that corresponds to the Cook-Levin encoding of their relevant circuits, with an extra input bit that determines which one is actually computed. Similarly, for every $\ell = \ell(\rho)$ the primitive ℓ-**UOWHF** $= \langle F_{\ell\text{-}\mathbf{UOWHF}}, R_{\ell\text{-}\mathbf{UOWHF}}\rangle$ corresponds to ℓ-bit compressing universal one-way hash functions.

4.1.1 Domain extension of a UOWHF.

The definition of a UOWHF only guarantees that h_ρ is compressing (i.e., it is possible that $\ell(\rho) = 1$). The first reduction we use is a domain extension of a UOWHF, that allows to construct an ℓ-bit compressing UOWHF from a UOWHF. Shoup [Sho00] shows a fully-black box construction of a ℓ-bit compressing UOWHF from one that compresses only one bit, which is the minimal requirement from any UOWHF.

Lemma 20 (UOWHF domain extension)**.** *There exists a fixed-parameter fully black-box construction of an* ℓ*-bit compressing UOWHF* $h'_\rho : \{0,1\}^{\log(\ell)\cdot\kappa(\rho)} \times \{0,1\}^{m+\ell} \to \{0,1\}^m$ *from a one-bit compressing UOWHF* $h_\rho : \{0,1\}^{\kappa(\rho)} \times \{0,1\}^{m+1} \to \{0,1\}^m$*. In order to evaluate* h'_ρ *the construction makes exactly* $\ell(\rho)$ *calls to* h_ρ*. The security reduction* $R_\rho^{h_\rho,B}$ *makes* ℓ *calls to its* h_ρ *oracle, and exactly one call to the breaker* $B_\rho = (B_1, B_2)_\rho$ *oracle. Furthermore, if* B_ρ $(\ell\text{-}\mathbf{UOWHF}, \epsilon)$*-breaks* h'_ρ*, then the reduction* $(\mathbf{UOWHF}, \frac{\epsilon}{\ell})$*-breaks* h_ρ*.*

We observe that the security definition for UOWHFs involves an interaction, and allows the adversary to save its state using σ. It will be more convenient for us to work with an equivalent non-interactive version. The following definition of collision resistance is tightly related to that of a UOWHF by the lemma that follows it, where we denote by $a\|b$ the concatenation of a and b.

Definition 21 (RP-CRHF)**.** *A* random preimage collision resistance hash function *is an efficiently uniformly computable family of functions* $h_\rho : \{0,1\}^{m(\rho)} \to \{0,1\}^{m'(\rho)}$ *with* $m'(\rho) < m(\rho)$*, such that for every efficient randomized machine* B *the function mapping* ρ *to*

$$\Pr_{\substack{v\xleftarrow{r}\{0,1\}^{m(\rho)}\\ v'\xleftarrow{r}B(\rho,v)}}\left[h_\rho(v) = h_\rho(v') \wedge v \neq v'\right]$$

is negligible. The family h *is an* ℓ-bit compressing *RP-CRHF, where* $\ell = \ell(\rho)$*, if additionally it holds that* $m(\rho) - m'(\rho) \geq \ell(\rho)$ *for all large enough* ρ*.*

The primitives **RP-CRHF** and $\log^2(\rho)$-**RP-CRHF** are defined analogously.

Lemma 22 (**UOWHF** to **RP-CRHF**, folklore)**.** *Let $h = \{h_\rho\}_{\rho\in\mathbb{N}^+}$ be a UOWHF. Then the family $h'_\rho : \{0,1\}^{\kappa(\rho)+m(\rho)} \to \{0,1\}^{\kappa(\rho)+m'(\rho)}$ given by $h'_\rho(k\|v) \stackrel{\text{def}}{=} (k\|h_\rho(k,v))$ is an RP-CRHF.*

Proof: The security definition of an RP-CRHF is trivially implied by that of a UOWHF. Suppose that we have an breaker B_ρ that finds a collision for a random preimage of $h'_\rho(k\|v)$. The reduction outputs the algorithms $(B_1, B_2)_\rho$ as follows: B_1 is the algorithm that chooses a uniform random $v \in \{0,1\}^{m(\rho)}$. Upon receiving a random key k, the reduction B_2 returns $B_\rho(k\|v)$. It is immediate that $(B_1, B_2)_\rho$ break the security of h as a UOWHF whenever B_ρ breaks the security of h' as a RP-CRHF. ∎

In particular, we observe that the proof of the lemma implies the existence of a fixed-parameter fully black-box construction of **RP-CRHF** from **UOWHF** that for every security parameter makes exactly one call to the universal one-way hash function. Additionally, the construction is security preserving in the strongest possible sense: If B_ρ $(\mathbf{RP\text{-}CRHF}, \epsilon)$-breaks the constructed h'_ρ (i.e., returns a collision on a uniform input v' with probability ϵ) then it holds that the reduction $(\mathbf{UOWHF}, \epsilon)$-breaks the underlying h_ρ. We mention that both the construction and the security reduction are uniform and efficient, and both make exactly one query to their oracles.

4.1.2 Pseudo-injective Functions.

Our last reduction establishes that padding the output of a $\log^2(\rho)$-**RP-CRHF** yields a primitive that is both a one-way function, and behaves like an injective function. A pseudo-injective function is an efficiently uniformly computable family $g = \{g_\rho\}_{\rho\in\mathbb{N}^+}$ of length preserving functions $g_\rho : \{0,1\}^{m(\rho)} \to \{0,1\}^{m(\rho)}$ such that for a uniformly chosen input $v \in \{0,1\}^{m(\rho)}$ it is impossible to find another input $v' \neq v$ such that both map to the same value under g_ρ. We stress that pseudo-injective functions exists unconditionally: Any permutation is a pseudo-injective function. Formally:

Definition 23 (Pseudo-Injectivity)**.** *A* pseudo-injective function *$g = \{g_\rho\}_{\rho\in\mathbb{N}^+}$ is a uniformly efficiently computable family of functions $g_\rho : \{0,1\}^{m(\rho)} \to \{0,1\}^{m(\rho)}$, such that for all uniform efficient algorithms A the function mapping ρ to*

$$\Pr_{\substack{v\stackrel{r}{\leftarrow}\{0,1\}^{m(\rho)}\\ v'\stackrel{r}{\leftarrow}A(1^\rho,v)}}\left[g_\rho(v') = g_\rho(v) \wedge v' \neq v\right]$$

is negligible.

Similarly to before, the primitive $\mathbf{PI} = \langle F_{\mathbf{PI}}, R_{\mathbf{PI}}\rangle$ corresponds to a pseudo-injective function. Next, we consider the primitive **OWF**∧**PI** that corresponds to all functions which are both a one-way function and a pseudo-injective function. Formally, it holds that $f \in F_{\mathbf{OWF}\wedge\mathbf{PI}}$ if and only if $f \in F_{\mathbf{OWF}}$ *and* $f \in F_{\mathbf{PI}}$. For a breaker circuit C, a function $f_\rho \in f \in F_{\mathbf{OWF}\wedge\mathbf{PI}}$, and a number ϵ it holds that $\langle f_\rho, C, \epsilon\rangle \in R_{\mathbf{OWF}\wedge\mathbf{PI}}$ if and only if $\langle f_\rho, C, \epsilon\rangle \in R_{\mathbf{OWF}}$ *or* $\langle f_\rho, C, \epsilon\rangle \in R_{\mathbf{PI}}$.

It turns out that padding any $\log^2(n)$-**RP-CRHF** to a length-preserving function, yields a function which is both a one-way function and a pseudo-injective function.

Lemma 24 ($\log^2(\rho)$-**RP-CRHF** to **OWF**∧**PI**)**.** *Let $h = \{h_\rho\}_{\rho\in\mathbb{N}^+}$ be an RP-CRHF that compresses $\ell(\rho) \stackrel{\text{def}}{=} m(\rho) - m'(\rho)$ bits, where $\ell(\rho) \geq \log^2(\rho)$. Then the family $\{h'_\rho\}_{\rho\in\mathbb{N}^+}$, where $h'_\rho(v) \stackrel{\text{def}}{=} h_\rho(v)\|0^{\ell(\rho)}$, is a one-way function and a pseudo-injective function.*

Proof: By the construction h'_ρ is always length-preserving and so h' implements a pseudo-injective and a one-way function. It is also clear that the construction is uniformly efficiently computable.

As for the security reduction, **OWF∧PI**-breaking h' implies either inverting h' on a uniformly random input, or finding a collision for one, for infinitely many security parameters ρ. This implies that for infinitely many security parameters ρ it is possible to break at least one of the properties.

First observe that any breaker B that breaks the pseudo-injectivity of h' immediately leads to one that **RP-CRHF**-breaks h with the same probability: On input $v' \in \{0,1\}^m$ return the output of $B(h'(v')) = B(h(v)\|0^{\ell(\rho)})$.

On the other hand, if B **OWF**-breaks h', we have that B inverts h' for infinitely many ρ's. We show that in this case, the algorithm, that on input $v \in \{0,1\}^{m(\rho)}$ returns $B(h_\rho(v))$, is the required reduction.

By our assumption B inverts h' on a random input with probability at least $\frac{1}{p}$, where p is a polynomial, for infinitely many security parameters. As h'_ρ compresses at least $\log^2(\rho)$ bits, we have that for a random v, the probability that $h'_\rho(v)$ has only one preimage is at most $2^{-\log^2(\rho)}$. Therefore, with probability at least $\frac{1}{p(\rho)} - \frac{1}{\rho^{\log(\rho)}} > \frac{1}{2p(\rho)}$ (for sufficiently large ρ) we have that B inverts $h'_\rho(v)$ and that $h'_\rho(v)$ has at least two preimages, in which case v is still uniform among $h_\rho^{-1}(h_\rho(v))$. Conitioned on this event it holds that with probability at least $\frac{1}{2}$ B returns a $v' \neq v$ that collides with v. Therefore we conclude that the reduction finds a random collision with probability at least $\frac{1}{4 \cdot p(\rho)}$ for all sufficiently large ρ. ∎

The composition of the constructions depicted in Lemmas 20, 22 and 24 establishes a fixed-parameter fully black-box construction of an $h' \in F_{\mathbf{OWF}\wedge\mathbf{PI}}$ from any $h \in F_{\mathbf{UOWHF}}$ that makes $\log^2(\rho)$ calls to the underlying UOWHF. The security reduction makes $\log^2(\rho)$ calls (to both its oracles) in order to break the security of the underlying UOWHF, and if B $(\mathbf{OWF}\wedge\mathbf{PI}, \frac{1}{p})$-breaks the constructed h for some polynomial p and breaker B, then the reduction $(\mathbf{UOWHF}, \frac{1}{p'})$-breaks h, where p' is a different polynomial such that $p'(\rho) > 4 \cdot p(\rho) \cdot \log^2(\rho)$. Thus we obtain:

Corollary 25. *Suppose that (G, R) is an s'-weak fixed-parameter fully black-box construction of* **UOWHF** *from* **OWF** *that makes at most $r' = r'(\rho)$ queries to* **OWF**. *Then there exists an s-weak fixed-parameter fully black-box construction of* **OWF∧PI** *from* **OWF** *that makes $r'(\rho) \cdot \log^2(\rho)$ calls to the underlying one-way function, where $s(\rho) \stackrel{\text{def}}{=} s'(\rho) \cdot \log^2(\rho)$.*

Proof: The composition of G with the constructions described in Lemmas 20, 22 and 24 implements a fixed-parameter fully black-box construction of an $h' \in F_{\mathbf{OWF}\wedge\mathbf{PI}}$ from any $h \in F_{\mathbf{OWF}}$. The composition of the reduction algorithms described in the lemmas with the s'-non-uniform reduction makes at most s calls to both its oracles. It follows that for every B that **OWF∧PI**-breaks the constructed h', the reduction $R^{[B,h]}$ $(\mathbf{OWF}, \frac{1}{s(\rho)})$-breaks h.[11] ∎

Therefore, in order to show that there is no s'-weak fixed-parameter fully black-box construction of **UOWHF** from **OWF**, where the construction makes r' calls to the one-way functions, it is sufficient to show that there is no s-weak fixed-parameter fully black-box construction of **OWF∧PI** from **OWF** that makes r calls, where $s(\rho) \stackrel{\text{def}}{=} s'(\rho) \cdot \log^2(\rho)$ and $r(\rho) \stackrel{\text{def}}{=} r'(\rho) \cdot \log^2(\rho)$. This is the goal of the next section.

[11] Note that for a large enough security parameter it still holds that it inverts h with probability $\frac{1}{s'(\rho)}$, however, as each query in the reduction algorithm of the assumed s-weak construction results in up to $\log^2(\rho)$ queries to the composed one, we get a total number of at most s queries.

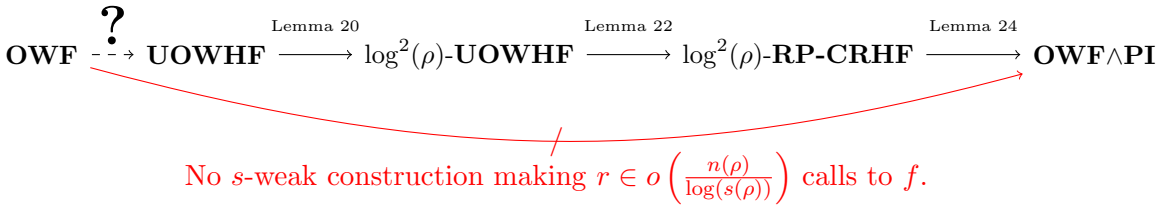

Figure 3: Fully Black-Box Constructions Diagram.

4.2 A Lower Bound on the Number of Calls for an s-Weak Fixed-Parameter Fully Black-Box Construction of OWF∧PI from OWF

As explained, a lower bound on a construction of **OWF∧PI** from **OWF** yields a very good (up to a $\log^2$-factor) bound on the construction of **UOWHF**. Our proof utilizes the machinery from Section 3. Let us introduce some notation. For an (n,n)-oracle circuit $g^{(?)}:\{0,1\}^m\to\{0,1\}^m$, a function $f:\{0,1\}^n\to\{0,1\}^n$ and a value $v\in\{0,1\}^m$, denote by $X_g(f,v)$ and $Y_g(f,v)$ the sets of queries and answers made to and received from f during the evaluation of $g^{(f)}(v)$, respectively.

For any potential construction (G,R) denote by $r=r(\rho)$ the number of queries g_ρ makes when instantiated for security parameter ρ with a one-way function $f_\rho:\{0,1\}^\rho\to\{0,1\}^\rho$, that is we set $n(\rho)\stackrel{\text{def}}{=}n'(\rho)\stackrel{\text{def}}{=}\rho$. Additionally, let $s=s(\rho)$ be a super-polynomial security function smaller than $2^{\frac{\rho}{10}}$. I.e., for all polynomials p and all large enough ρ it holds that $p(\rho)<s(\rho)<2^{\frac{\rho}{10}}$. We prove that if $r(\rho)<\frac{n(\rho)}{2000\cdot\log(s(\rho))}$ holds for all large enough ρ, then (G,R) is not an s-weak fixed-parameter fully black-box construction of **OWF∧PI** from **OWF**.

Theorem 26. *For all super-polynomial security functions $s=s(\rho)<2^{\frac{\rho}{10}}$ and $r=r(\rho)$ there is no s-weak fixed-parameter fully black-box construction of* **OWF∧PI** *from* **OWF** *such that $g_\rho^{(?)}:\{0,1\}^{m(\rho)}\to\{0,1\}^{m(\rho)}$ makes at most $r(\rho)$ calls to the underlying one-way function, where $n(\rho)\stackrel{\text{def}}{=}n'(\rho)\stackrel{\text{def}}{=}\rho$ and $g_\rho^{(?)}\stackrel{\text{def}}{=}G(1^\rho,n,n')$, and $r(\rho)\leq\frac{n(\rho)}{2000\cdot\log(s(\rho))}$ holds for all large enough ρ.*

Proof: Without loss of generality, we assume that the construction g makes exactly $r(\rho)\stackrel{\text{def}}{=}\frac{n(\rho)}{2000\cdot\log(s(\rho))}$ different queries. Whenever this is not the case, it is always possible to amend G so that it behaves exactly as before, but on input $(1^\rho,n,n')$ it outputs an (n,n')-oracle circuit with $r(\rho)$ oracle gates, and additionally, all queries are different.

Let s and r be a pair of security functions such that s is super-polynomial, that is, for every polynomial p and large enough ρ it holds that $s(\rho)>p(\rho)$, and that $r(\rho)=\frac{n(\rho)}{2000\cdot\log(s(\rho))}$ holds for all sufficiently large ρ.

We now explain how to construct the oracle $\mathcal{O}=\{\mathcal{O}_\rho\}_{\rho\in\mathbb{N}^+}$ and the collection of sets of functions $\mathcal{F}=\{\mathcal{F}_\rho\}_{\rho\in\mathbb{N}^+}$. For each security paramter ρ we define the oracle $\mathcal{O}_\rho$ and the set $\mathcal{F}_\rho$ *independently* of the oracles and function sets chosen for other security parameters. It will always hold that $\mathcal{F}_\rho\subset\{0,1\}^n\to\{0,1\}^n$, and so the constructed $\mathcal{F}$ is contained in $F_{\mathbf{OWF}}$.

Therefore, from now on we omit the security parameter in our notation, but formally all our parameters depend on the security parameter ρ. In particular, $g^{(?)}$ is the construction that the uniform construction algorithm G outputs for security parameter $\rho=n=n'$ with a function $f_\rho:\{0,1\}^\rho\to\{0,1\}^\rho$.

Analogously to [HS12], for every security parameter we break either the one-wayness property of the constructed function, or its pseudo-injectivity. For the oracle circuit $g^{(?)} : \{0,1\}^m \to \{0,1\}^m$, we check whether when g is evaluated with a random permutation $f \xleftarrow{r} \mathcal{P}_n$ and a random input $v \xleftarrow{r} \{0,1\}^m$, the output $g^f(v)$ is significantly correlated with any subset of the set of oracle answers returned by f on the calls made to it during the evaluation of $g^f(v)$ (recall that these are denoted by $Y_g(f,v)$). To this end, we bring the procedure STA (for safe to answer), which returns true if and only if there is no such correlation:

Procedure STA(w, Q) (on $w \in \{0,1\}^m$ and $Q \subset \{0,1\}^n$ of size r)

---

for all $B \subseteq Q$ **do**
. **if** $\Pr_{f' \xleftarrow{r} \mathcal{P}_n, v' \xleftarrow{r} \{0,1\}^m} \left[g^{(f')}(v') = w \,\middle|\, B \subseteq Y_g(f', v') \right] \geq 2^{-m+\frac{n}{30}}$
. **return** `false`
return `true`

We set $p(g)$, the probability that for a random permutation f and a random input v, the output $g^f(v)$ is correlated with some subset of the answers $Y_g(f,v)$. Define

$$p(g) \stackrel{\text{def}}{=} \Pr_{f \xleftarrow{r} \mathcal{P}_n, v \xleftarrow{r} \{0,1\}^m} \left[\text{STA}(g^{(f)}(v), Y_g(f,v)) \right] \quad . \tag{4}$$

We stress that both the output of STA (for any value y and a set Q), and the value $p(g)$ do not depend on any specific permutation, but rather on a combinatorial property of the construction as a whole, which averages over all permutations.

As explained, we set the oracle $\mathcal{O}$ and the set $\mathcal{F}$ based on the value $p(g)$. In case that $p(g) > \frac{1}{2}$ we set the oracle $\mathcal{O} \stackrel{\text{def}}{=} \text{BreakOW}_g \stackrel{\text{def}}{=} \{\text{BreakOW}_{g,f}\}_{f \in \{0,1\}^n \to \{0,1\}^n}$, where we use the oracle BreakOW_g from [HS12], which is described next. In [HS12] it is implicitly proved that there exists a set $\mathcal{F} \subset \mathcal{P}_n$ of size $|\mathcal{F}| > \frac{|\mathcal{P}_n|}{5}$, such that $\text{BreakOW}_{g,f}$ $(\mathbf{OWF}, \frac{1}{4})$-breaks g^f for all $f \in \mathcal{F}$, and that BreakOW_g is $2^{\frac{n}{5}}$-stable for $\mathcal{F}$, in which case condition (2) in Theorem 18 is satisfied.

Algorithm $\text{BreakOW}_{g,f}(w)$ (on input $w \in \{0,1\}^m$)

---

for all $v \in \{0,1\}^m$ **do**
. **if** $g^{(f)}(v) = w$ **then**
. **if** $\text{STA}(w, Y_g(f,v))$ **then**
. **return** v
return $\perp$

In the case $p(g) \leq \frac{1}{2}$ we show that when f is chosen uniformly at random from a set of regular degenerate functions, it is often the case that the construction $g^{(f)}$ is not injective, and therefore there exists an oracle which breaks the pseudo-injectivity of $g^{(f)}$. The challenge is to find a breaker oracle that is t-stable. The next lemmas establish that the oracle BreakPI satisfies the required conditions in this case.

Formally, for a construction circuit g we define the oracle $\text{BreakPI}_g = \{\text{BreakPI}_{g,f}\}_{f \in \{0,1\}^n \to \{0,1\}^n}$ that for a function f is given by:

Algorithm $\text{BreakPI}_{g,f}(v)$ (on input $v \in \{0,1\}^m$)

---

for all $v' \in \{0,1\}^m$ **do**
. **if** $g^{(f)}(v) = g^{(f)}(v')$ **and** $v' \neq v$ **then**
. **if** $Y_g(f,v) = Y_g(f,v')$ **then**
. **return** v'
return $\perp$

Now, we fix $i \stackrel{\text{def}}{=} \frac{n}{200 \cdot r} = 10 \cdot \log(s)$. We show that for a $\frac{1}{6}$-fraction of the functions f in $\mathcal{R}_{n,i}$ it holds that $\text{BreakPI}_{g,f}$ breaks the pseudo-injectivity of $g^{(f)}$.

Lemma 27. *Let $g : \{0,1\}^m \to \{0,1\}^m$ be an r-query oracle construction with $p(g) \leq \frac{1}{2}$. Then for a $\frac{1}{6}$-fraction of the functions in $\mathcal{R}_{n,\frac{n}{200 \cdot r}}$ it holds that*

$$\Pr_{v \xleftarrow{r} \{0,1\}^m} \left[\text{BreakPI}_{g,f}(v) \text{ outputs } v' \text{ s.t. } v \neq v' \wedge g^{(f)}(v) = g^{(f)}(v') \right] \geq \frac{1}{24}. \tag{5}$$

Proof:

By inspection of the algorithm BreakPI it holds that whenever $\text{BreakPI}_{g,f}(v) \neq \perp$ it outputs a value $v' \neq v$ such that $g^{(f)}(v) = g^{(f)}(v')$, and so it breaks the pseudo-injectivity of $g^{(f)}(\cdot)$. By the assumption, it holds that a random evaluation $\text{STA}(g^{(f)}(v), Y_g(f,v))$ on a random permutation f returns `false` with probability $(1 - p(g)) \geq \frac{1}{2}$. Next, as for every (n,n)-oracle circuit $D^{(?)}$ that does not take an input and has one output bit (called distinguisher) making at most r queries, it holds that

$$\left| \Pr_{f \xleftarrow{r} \mathcal{P}_n} [D^{(f)} = 1] - \Pr_{f \xleftarrow{r} \mathcal{R}_{n,i}} [D^{(f)} = 1] \right| \leq \frac{r^2}{2^i} \ ,$$

we obtain

$$\Pr_{f \xleftarrow{r} \mathcal{R}_{n,\frac{n}{200 \cdot r}}, v \xleftarrow{r} \{0,1\}^m} \left[\neg \text{STA}(g^{(f)}(v), Y_g(f,v)) \right] \geq (1 - p(g)) \cdot \left(1 - \frac{r^2}{2^{\frac{n}{200 \cdot r}}} \right) \ . \tag{6}$$

The following counting argument shows that for every $f \in \mathcal{R}_{n,\frac{n}{200 \cdot r}}$ there exists a set $\mathcal{W}_f = \mathcal{W}_f(\text{Im}(f))$ of size $|W_f| \leq 2^{m - \frac{n}{100}}$ such that

$$\Pr_{f \xleftarrow{r} \mathcal{R}_{n,\frac{n}{200 \cdot r}}, v \xleftarrow{r} \{0,1\}^m} \left[g^{(f)}(v) \in \mathcal{W}_f \right] \geq (1 - p(g)) \cdot \left(1 - \frac{r^2}{2^{\frac{n}{200 \cdot r}}} \right) > \frac{1}{3} \ , \tag{7}$$

for sufficiently large n.

Define $\mathcal{W}_f \stackrel{\text{def}}{=} \left\{ g^{(f)}(v) : v \in \{0,1\}^m \wedge \neg \text{STA}(g^f(v), Y_g(f,v)) \right\}$. Equation (7) follows readily from the definition of $\mathcal{W}_f$ and (6). As for $|\mathcal{W}_f|$, we have that there are less than $|\text{Im}(f)|^r = 2^{\frac{n}{200}}$ different possibilities for $Y_g(f,v)$ sets. For each of these sets there are 2^r subsets, and for each such subset S there are at most $2^{m - \frac{n}{30}}$ values w that satisfy $\Pr_{f \xleftarrow{r} P_n, v \xleftarrow{r} \{0,1\}^m} \left[g^f(v) = w | S \subseteq Y_g(f,v) \right] > 2^{-m + \frac{n}{30}}$. In total, we have that $|\mathcal{W}_f| \leq 2^{m - \frac{n}{30}} \cdot 2^{\frac{n}{200}} \cdot 2^r < 2^{m - \frac{n}{100}}$. The existence of such a set $\mathcal{W}_f$ was observed in [HS12].

Now, By Lemma 32 (with $p = \frac{1}{3}, \alpha = \frac{1}{2}$ and $\beta = \frac{1}{2}$) it holds that a $\frac{1}{6}$-fraction of the functions $f \in \mathcal{R}_{n,\frac{n}{200 \cdot r}}$ satisfy:

$$\Pr_{v \xleftarrow{r} \{0,1\}^m} \left[g^{(f)}(v) \in \mathcal{W}_f \right] > \frac{1}{6} \ . \tag{8}$$

Fix a function $f \in \mathcal{R}_{n,\frac{n}{200 \cdot r}}$ for which (8) holds. We show that in this case $\text{BreakPI}_{g,f}$ returns a collision with probability at least $\frac{1}{30}$ for a randomly chosen v.

Define $\mathcal{W}'_f \stackrel{\text{def}}{=} \left\{ w \in \mathcal{W}_f \,:\, |\left(g^{(f)}\right)^{-1}(w)| > \frac{1}{12} \cdot 2^{\frac{n}{100}} \right\}$. We claim that

$$\Pr_{v \xleftarrow{r} \{0,1\}^m} \left[g^{(f)}(v) \in \mathcal{W}'_f \,\middle|\, g^{(f)}(v) \in \mathcal{W}_f \right] \geq \frac{1}{2} \ . \tag{9}$$

Equation (8) asserts that there are at least $\frac{1}{6} \cdot 2^m$ elements v that are mapped to fewer than $2^{m-\frac{n}{100}}$ images. It follows that there are at most $\frac{1}{12} \cdot 2^{\frac{n}{100}} \cdot 2^{m-\frac{n}{100}} = \frac{1}{12} \cdot 2^m$ elements in $\left(g^{(f)}\right)^{-1}(\mathcal{W}_f) \setminus \left(g^{(f)}\right)^{-1}(\mathcal{W}'_f)$, which asserts the claim. Finally, for any $w \in \mathcal{W}'_f$, we claim that

$$\Pr_{v \xleftarrow{r} \{0,1\}^n} \left[\text{BreakPI}_{g,f}(v) \neq \perp \,\middle|\, g^{(f)}(v) = w \right] > \frac{1}{2} \ . \tag{10}$$

Conditioned on $g^{(f)}(v) = w$, we have that v is still uniform among $\left(g^{(f)}\right)^{-1}(w)$. Now, recall that $\text{BreakPI}_{g,f}$ fails to return a collision only when v is such that $Y_g(f, v)$ differs from the sets $Y_g(f, v')$, for all $v' \in \left(g^{(f)}\right)^{-1}(w) \setminus \{v\}$. Since f is degenerate, there are at most $|\text{Im}\,(f)\,|^r = 2^{\frac{n}{200}}$ different $Y_g(f, v)$ sets[12]. Therefore, $\text{BreakPI}_{g,f}(v)$ fails for at most $2^{\frac{n}{200}}$ of the elements of $\left(g^{(f)}\right)^{-1}(w)$, as for at most $2^{\frac{n}{200}} - 1$ of them it can be the case that there is no other v' for which $Y_g(f, v) = Y_g(f, v')$. Since $w \in \mathcal{W}'_f$ we have that $|\left(g^{(f)}\right)^{-1}(w)| > 2^{\frac{n}{100}}$ and so $\text{BreakPI}_{g,f} \neq \perp$ with probability at least $\frac{1}{2}$. The claim follows. Combining our claims we obtain:

$$\begin{aligned}
&\Pr_{v \xleftarrow{r} \{0,1\}^m} [\text{BreakPI}_{g,f}(v) \neq \perp] \\
&\geq \Pr_v \left[\text{BreakPI}_{g,f}(v) \neq \perp \,\middle|\, g^{(f)}(v) \in \mathcal{W}'_f \right] \\
&\quad \cdot \Pr_v \left[g^{(f)}(v) \in \mathcal{W}'_f \,\middle|\, g^{(f)}(v) \in \mathcal{W}_f \right] \cdot \Pr_v \left[g^{(f)}(v) \in \mathcal{W}_f \right] \\
&\geq \frac{1}{12} \sum_{w \in \mathcal{W}'_f} \Big(\Pr_v \left[\text{BreakPI}_{g,f}(v) \neq \perp \,\middle|\, g^{(f)}(v) = w \right] \\
&\qquad\qquad \cdot \Pr_v \left[g^{(f)}(v) = w \,\middle|\, g^{(f)}(v) \in \mathcal{W}'_f \right] \Big) > \frac{1}{24} \ .
\end{aligned}$$

The lemma follows. ■

We conclude from Lemma 27 that if $p(g) \leq \frac{1}{2}$, there exists a partition P of $\{0,1\}^n$ to sets of size 2^{n-i} and an image-set I of size 2^i, such that (5) holds for at least a $\frac{1}{6}$-fraction of the functions $f \in \mathcal{R}_{n,i}(P, I)$. Set $\mathcal{F} \subset \mathcal{R}_{n,i}(P, I)$ to be the set of all functions for which (5) holds. It follows that $|\mathcal{F}| \geq \frac{1}{6} \cdot 2^i$, as $|\mathcal{R}_{n,i}(P, I)| = |\mathcal{P}_i|$.

We next show that that for the class of functions $\mathcal{R}_{n,i}(P, I)$ the oracle can be implemented such that it is stable.

Lemma 28. *Let $i \in \mathbb{N}^+$ and $I \subset \{0,1\}^n$ of size 2^i and P a partition of $\{0,1\}^n$ to sets of size 2^{n-i}. Then there exists an implementation of the oracle BreakPI_g that is n-stable for $\mathcal{R}_{n,i}(P, I)$.*

Proof: As we show next, when we limit the oracle to a fixed partition P and a fixed image I, it can be implemented such that it makes only few queries to f on every call to BreakPI and therefore enjoys stable sets.

[12] In fact, this shows that we could also require that BreakPI returns a value $v' \neq v$ such that the vector (now differentiating two y-answer sets according to the ordering of the answers) of query answers in the computation of $g^{(f)}(v)$ matches that of $g^{(f)}(v')$, but this is not needed in order to prove that BreakPI is t-stable.

We begin by describing the parsimonious implementation of the algorithm, i.e., an implementation that makes very few queries to f.

The parsimonious algorithm will have a description of the partition P and the image mapping I. On input v it simulates BreakPI_g as follows: In every iteration the BreakPI_g queries only $g^{(f)}(v)$. The algorithm records action of the f on the inputs of $X_g(f,v)$. That is, it records the values of $h[f]$ on $\pi_P(X_g(f,v))$.

It then tries to emulate the value $g^{(f)}(v')$ while assuming that f is in $\mathcal{R}_{n,i}(P,I)$. Whenever $g^f(v')$ issues a query x, the algorithm checks whether it is mapped under the partition function π_P to the same value as one of the x queries issued before. I.e., it checks whether $\pi_P(x) = \pi_P(x')$ for some $x' \in X_g(P,I)$. In such a case it uses the recorded value of $f(x')$ and the emulation continues. If the algorithm gets stuck it decides that $Y_g(f,v) \neq Y_g(f,v')$ and continues to the next iteration. We first note that the algorithm is well defined for all $f \in \mathcal{F}_{n,i}(P,I)$.

Next, we observe that for a function $f \in \mathcal{R}_{n,i}(P,I)$ the parsimonious algorithm simulates BreakPI_g perfectly: If for some value v' the evaluation of $g^{(f)}(v')$ gets stuck on some query x, we know that the output would not return the value v' since in this case the function $h[f]$ is a permutation and r_I is injective it holds that $f(x) \notin Y_g(f,v)$. On the other hand, when the simulation of an iteration succeeds it follows that for every x it holds that $f(x)$ is computed correctly (and accordingly $g^{(f)}(v)$ and $Y_g(f,v')$). Again, this follows from the decomposition $f = r_I \circ h \circ \pi_P$.

We note that the algorithm is not guranteed to behave the same as BreakPI_g for an arbitrary $f \in \mathcal{F}_{n,i}(P,I)$, but we only care about its behavior for a regular f. It is now immediate that the implementation is r-stable: In a similar manner to the case of the evaluation oracle we set $\mathcal{I}^{\mathcal{O}}_{f,v} \stackrel{\text{def}}{=} X_g(f,v)$ and $\mathcal{B}^{\mathcal{O}}_{f,q,y} \stackrel{\text{def}}{=} Y_g(f,v)$ as determining and bending sets for BreakPI, respectively. ∎

It is left to check that $2^s \cdot d_i(s,n) < |\mathcal{F}|$. We compute

$$\frac{d_i(s,n)}{|\mathcal{F}|} = \frac{\binom{2^i}{b}^2 \cdot (2^i - b)!}{(1/6) \cdot 2^i} = \binom{s^{10}}{b} \frac{1}{(1/6) \cdot b!} \leq \left(\frac{e^2 \cdot s^{10}}{b^2}\right)^b \quad , \tag{11}$$

where

$$b = \frac{2^i}{3 \cdot s^2 \cdot n^2} = \frac{2^{\frac{n}{200 \cdot r}}}{3 \cdot s^2 \cdot n^2} = \frac{2^{10 \cdot \log(s)}}{3 \cdot s^2 \cdot n^2} > s^7, \tag{12}$$

since $s > 3 \cdot n^2$ for sufficiently large n. Combining (11) with (12) we conclude that $\frac{2^s \cdot d_i(s,n)}{|\mathcal{F}|} < 1$.

We have shown that the conditions of Theorem 18 hold, and therefore we conclude that there is no s-weak fixed-parameter fully black-box construction of **OWF∧PI** from **OWF**. The theorem is proved. ∎

4.3 Deriving the Lower Bound

We are now ready to derive our lower bound for constructions of a universal one-way hash function from a one-way function:

Corollary 29. *Let s' be a security function such that $s(n) \stackrel{\text{def}}{=} s'(n) \cdot \log^2(n)$ is a super-polynomial security function for which $s(n) < 2^{\frac{n}{10}}$ holds. Then there is no s-weak fixed-parameter fully black-box construction of* **UOWHF** *from* **OWF**, *where the construction makes at most $r'(n) = \frac{n}{2000 \cdot \log(s(n)) \cdot \log^2(n)}$ calls to a one-way function $f = \{f_n : \{0,1\}^n \to \{0,1\}^n\}_{n \in \mathbb{N}^+}$.*

Proof: We apply Corollary 25 with Theorem 26. ■

Corollary 30. *There is no fixed-parameter fully black-box construction of* **UOWHF** *from* **OWF**, *where the construction makes at most $r = r(n)$ calls to a one-way function $f = \{f_n : \{0,1\}^n \to \{0,1\}^n\}_{n\in\mathbb{N}^+}$, where $r \in o\left(\frac{n}{\log^3(n)}\right)$.*

Proof: Let $r \in o\left(\frac{n}{\log^3(n)}\right)$. Then there exists a super-constant function $\alpha = \alpha(n)$, such that the function $r'(n)$ given by $r'(n) \stackrel{\text{def}}{=} r(n)\cdot\alpha(n)$ is still in $o\left(\frac{n}{\log^3(n)}\right)$. The bound follows immediately from Corollary 29 applied with $s(n) \stackrel{\text{def}}{=} 2^{\alpha(n)\cdot\log(n)}$. ■

5 A Tight Lower-Bound for Fully Black-Box Constructions of UOWHF from OWF.

Corollary 30 establishes a lower bound of $\Omega(n/\log^3(n))$ calls for any fully black-box construction of **UOWHF** from a **OWF**. We now explain how to improve the lower-bound by a $\log(n)$ factor.

Recall that the choice of $\log^2(\rho)$ in Lemma 24 was made such that any breaker for **OWF**$\wedge$**PI** can be translated to one that breaks the security of the **RP-CRHF**, i.e., finds a collision for a random preimage. We observe that the lemma is still correct if we start with a $\ell(\rho)$-**RP-CRHF**, for any super-logarithmic function $\ell(\rho) \in \omega(\log(\rho))$. That is, a compression by a super-logarithmic number of bits is sufficient for our proof to go through. In this case (as in the proof of Lemma 24) it still follows that at most a negligible fraction of the inputs have at most one preimage, and therefore the reduction converts a breaker with noticeable success probability for **OWF**$\wedge$**PI** is converted to a breaker with a noticeable success probability for $\ell(\rho)$-**RP-CRHF**. Next, this improves Corollary 25 by a $\log(\rho)$ factor, which consequently improves our bound to $\Omega(n/\log^2(n))$ calls.

5.1 A Tight Lower-Bound for the Construction of a $\log(\rho)$-UOWHF

Using the analysis of the construction presented in [BM12], we get a construction that makes $O(\log(n)/n)$ queries from a one-way function from this class of functions. That is, when considering constructions of **UOWHF** from **OWF** there is a gap of a $\log(n)$ factor.

The construction in [BM12] offers an approach to explain this gap. Namely, it yields a universal one-way hash function that compresses $\log(n)$ bits and not just one bit. Therefore, when considering the construction of a $\log(\rho)$-**UOWHF** from **OWF** and using the analysis presented in the previous section (Section 5) it follows that we need only $\omega(1)$ calls to a $\log(\rho)$-**UOWHF** to get a **PI**$\wedge$**OWF** (as explained before), in which case our bound improves to $\Omega(n/\log(n))$ calls. That is, when considering construction of UOWHFs that compress $O(\log(n))$ bits we already get a tight lower bound. Recall that even for constructions of $\log(n)$-**UOWHF** the lower-bound presented in [GGKT05] is trivial (i.e., instantiating their bound for $k = \log(n)$ we get a lower-bound of one call to the **OWF**.

5.2 A Tight Lower-Bound of $\Omega(n/\log(n))$ Calls

In this section we present the most general version of our lower-bound. We consider once more the series of reductions used to derive Corollary 25. We show that it is possible to get rid of the last $\log(n)$ factor used in Lemma 24, which in turn results in a lower-bound of $\Omega(n/\log(n))$ calls.

We do not know of a reduction that transforms a breaker with an inverse polynomial success probability of a **OWF∧PI** to a breaker with an inverse polynomial success probability for $O(1)$-**RP-CRHF** (i.e., a UOWHF that compresses a constant number of bits). Nevertheless, we can exploit the fact that both breakers presented in Section 4 perform better than merely breaking the primitive. That is, in order to derive the lower-bound using Corollary 25 it is sufficient that the breaker in Theorem 26 breaks (with some inverse-polynomial probability). However, both BreakPI and BreakOW enjoy a stronger property: They both break the security of the **OWF∧PI** with constant probability (both break the candidate construction with probability at least 1/100) rather than with inverse-polynomial probability.

We observe that in this case the reduction algorithm presented in Lemma 24 gives a breaker with a constant success probability for $\ell(\rho)$-**RP-CRHF**, where $\ell(\rho) = 10$. Following the analysis in Lemma 24 we have that at most a 2^{-10}-fraction of the images have only one preimage, and using the union-bound the reduction breaks the $\ell(\rho)$-**RP-CRHF** with probability at least $1/100 - 1/2^{10} > 1/200$. In a similar manner to Section 5 this gives now a version of Corollary 25, that asserts the existence of an s-weak fixed-parameter fully black-box construction of **OWF∧PI** from **OWF** for breakers that are considered successful only if they break the security property of the **OWF∧PI** with constant probability:

Corollary 31. *Suppose that (G, R) is an s'-weak fixed-parameter fully black-box construction of* **UOWHF** *from* **OWF** *that makes at most $r' = r'(\rho)$ queries to* **OWF**. *Then there exists an s-weak fixed-parameter fully black-box construction of* **OWF∧PI** *from* **OWF** *that makes $10 \cdot r'(\rho)$ calls to the underlying one-way function, where $s(\rho) \stackrel{\text{def}}{=} 10 \cdot s'(\rho)$ and additionally, a potential breaker for* **OWF∧PI** *is successful only if it breaks the security if property of the primitive with probability at least* 1/100.

We stress that the restriction with respect to the constant success probability of the potential breaker in the corollary does *not* propagate to the constructions we rule out, and our lower-bound holds with respect to the 'standard' (polynomial) security requirements. Applying Corollary 31 with Theorem 26 we derive a lower-bound of $\Omega(n/\log(n))$ calls for a fully black-box construction of a **UOWHF** from a **OWF** in the conditions of Corollaries 29 and 30.

Acknowledgments.

We thank the anonymous reviewers of TCC'13 for their helpful comments.

References

[Bar01] Boaz Barak. How to go beyond the black-box simulation barrier. In *FOCS*, pages 106–115. IEEE Computer Society, 2001.

[BM12] Kfir Barhum and Ueli Maurer. UOWHFs from OWFs: Trading regularity for efficiency. In Alejandro Hevia and Gregory Neven, editors, *LATINCRYPT*, volume 7533 of *Lecture Notes in Computer Science*, pages 234–253. Springer, 2012.

[GGKT05] Rosario Gennaro, Yael Gertner, Jonathan Katz, and Luca Trevisan. Bounds on the efficiency of generic cryptographic constructions. *SIAM J. Comput.*, 35(1):217–246, 2005.

[HH09] Iftach Haitner and Thomas Holenstein. On the (im)possibility of key dependent encryption. In Omer Reingold, editor, *TCC*, volume 5444 of *Lecture Notes in Computer Science*, pages 202–219. Springer, 2009.

[HHR+10] Iftach Haitner, Thomas Holenstein, Omer Reingold, Salil P. Vadhan, and Hoeteck Wee. Universal one-way hash functions via inaccessible entropy. In Henri Gilbert, editor, *EUROCRYPT*, volume 6110 of *Lecture Notes in Computer Science*, pages 616–637. Springer, 2010.

[HHRS07] Iftach Haitner, Jonathan J. Hoch, Omer Reingold, and Gil Segev. Finding collisions in interactive protocols - a tight lower bound on the round complexity of statistically-hiding commitments. In *FOCS*, pages 669–679. IEEE Computer Society, 2007.

[HR04] Chun-Yuan Hsiao and Leonid Reyzin. Finding collisions on a public road, or do secure hash functions need secret coins? In Matthew K. Franklin, editor, *CRYPTO*, volume 3152 of *Lecture Notes in Computer Science*, pages 92–105. Springer, 2004.

[HS12] Thomas Holenstein and Makrand Sinha. Constructing a pseudorandom generator requires an almost linear number of calls. *CoRR*, abs/1205.4576. Extended Abstract to appear in FOCS 2012, 2012.

[IR89] Russell Impagliazzo and Steven Rudich. Limits on the provable consequences of one-way permutations. In Johnson [Joh89], pages 44–61.

[Joh89] David S. Johnson, editor. *Proceedings of the 21st Annual ACM Symposium on Theory of Computing, May 14-17, 1989, Seattle, Washigton, USA*. ACM, 1989.

[KST99] Jeong Han Kim, Daniel R. Simon, and Prasad Tetali. Limits on the efficiency of one-way permutation-based hash functions. In *FOCS*, pages 535–542, 1999.

[MM11] Takahiro Matsuda and Kanta Matsuura. On black-box separations among injective one-way functions. In Yuval Ishai, editor, *TCC*, volume 6597 of *Lecture Notes in Computer Science*, pages 597–614. Springer, 2011.

[NY89] Moni Naor and Moti Yung. Universal one-way hash functions and their cryptographic applications. In Johnson [Joh89], pages 33–43.

[Rom90] John Rompel. One-way functions are necessary and sufficient for secure signatures. In *STOC*, pages 387–394. ACM, 1990.

[RS10] Alon Rosen and Gil Segev. Chosen-ciphertext security via correlated products. *SIAM J. Comput.*, 39(7):3058–3088, 2010.

[RTV04] Omer Reingold, Luca Trevisan, and Salil P. Vadhan. Notions of reducibility between cryptographic primitives. In Moni Naor, editor, *TCC*, volume 2951 of *Lecture Notes in Computer Science*, pages 1–20. Springer, 2004.

[Sho00] Victor Shoup. A composition theorem for universal one-way hash functions. In Bart Preneel, editor, *EUROCRYPT*, volume 1807 of *Lecture Notes in Computer Science*, pages 445–452. Springer, 2000.

[Sim98] Daniel R. Simon. Finding collisions on a one-way street: Can secure hash functions be based on general assumptions? In *EUROCRYPT*, pages 334–345, 1998.

[Yao82] Andrew Chi-Chih Yao. Theory and applications of trapdoor functions (extended abstract). In *FOCS*, pages 80–91. IEEE Computer Society, 1982.

A Appendix

A.1 A Lemma on Probability Spaces

The following is a well-known lemma on probability spaces:

Lemma 32. *Let (X,Y) be jointly distributed random variables taking values from some set $(\mathcal{X},\mathcal{Y})$, and let $A : \mathcal{X}\times\mathcal{Y}\to\{0,1\}$ such that $\Pr_{x\xleftarrow{r}X,y\xleftarrow{r}Y}[A(x,y)=1]\geq p>0$. Then for all $\alpha,\beta>0$ satisfying $\alpha+(1-\alpha\cdot p)\cdot\beta<1$ it holds that*

$$\Pr_{x\xleftarrow{r}X}\left[\Pr_{y\xleftarrow{r}Y}[A(x,y)=1]\geq\beta\cdot p\right]\geq\alpha\cdot p\ . \tag{13}$$

Proof: Assume otherwise, we have that

$$\begin{aligned}
&\Pr_{X,Y}[A(X,Y)]\\
=\ &\Pr_{X,Y}\left[A(X,Y)\mid\Pr_{Y}[A(X,Y)]<\beta\cdot p\right]\cdot\Pr_{X}\left[\Pr_{Y}[A(X,Y)]<\beta\cdot p\right]\\
+\ &\Pr_{X,Y}\left[A(X,Y)\mid\Pr_{Y}[A(X,Y)]\geq\beta\cdot p\right]\cdot\Pr_{X}\left[\Pr_{Y}[A(X,Y)]\geq\beta\cdot p\right]\\
<\ &(\beta\cdot p)\cdot(1-\alpha\cdot p)+1\cdot(\alpha\cdot p)<p\ ,
\end{aligned}$$

where we write $A(X,Y)$ for the event that $A(X,Y)=1$, which contradicts the assumption on (α,β). ∎

ECCC ISSN 1433-8092

http://eccc.hpi-web.de