(Almost) Optimal Constructions of UOWHFs from 1-to-1 and Known-Regular One-way Functions

Yu Yu* Dawu Gu† Xiangxue Li‡ Jian Weng§

Abstract

A universal one-way hash function (UOWHF) is a compressing function for which finding a second preimage is infeasible. The seminal work of Rompel (STOC 1990) that one-way functions (OWFs) imply UOWHFs is one of the most important founding results of modern cryptography. The current best known UOWHF construction from any one-way function (on n-bit input) by Haitner et al. (Eurocrypt 2010) requires output and key length $\tilde{O}(n^7)$, which is far from practical.

On the other hand, special structured OWFs typically give rise to much more efficient (and almost practical) UOWHFs. Naor and Yung (STOC 1989) gave an optimal construction of UOWHFs of key and output lengths both linear in n by making a single call to any one-way permutation. De Santis and Yung (Eurocrypt 1990), Barhum and Maurer (Latincrypt 2012), and Ames, Gennaro, and Venkitasubramaniam (Asiacrypt 2012) further extended the work to more generalized settings, namely, 1-to-1 and regular one-way functions. However, the best known constructions still require key length $O(n \cdot \log n)$ even for 1-to-1 one-way functions, and need to make $O(\omega(1)\cdot \log n)$ calls to any known regular one-way functions, or even $\tilde{O}(n)$ adaptive calls if one wants linear output length at the same time.

In this paper, we first introduce a technical lemma about universal hashing with nice symmetry to the leftover hash lemma, which might be of independent interest. That is, if one applies universal hash function $h : \{0,1\}^n \to \{0,1\}^{a+d}$ to any random variable X of min-entropy a, then h will be 1-to-1 on X except for a 2^{-d} fraction. We also generalize the construction of Naor and Yung (that was optimal only for one-way permutations) to 1-to-1 and almost regular one-way functions, and significantly extend their analysis. The above yields the following results.

- For any 1-to-1 one-way function, we give an optimal construction of UOWHFs with key and output length $\Theta(n)$ by making a single call to the underlying OWF.
- For any known-(almost-)regular one-way function with known hardness, we give another optimal construction of UOWHFs with key and output length $\Theta(n)$ and a single call to the one-way function.
- For any known-(almost-)regular one-way function, we give a construction of UOWHFs with key and output length $O(\omega(1)\cdot n)$ and by making $\omega(1)$ non-adaptive calls to the one-way function.

where the first two constructions enjoy optimal parameters simultaneously and the third one is nearly optimal up to any (efficiently computable) super-constant factor $\omega(1)$, e.g., $\log\log\log n$ or even less. Furthermore, the constructions enjoy optimal shrinkages by matching the upper bound of Gennaro et al. (SICOMP 2005).

Keywords: Foundations, One-way Functions, Universal One-way Hash Functions, Target Collision Resistance.

*Shanghai Jiao Tong University. Email: `yuyuathk@gmail.com`.
†Shanghai Jiaotong University.
‡East China Normal University.
§Jinan University.

1 Introduction

Informally, a family of compressing hash functions (i.e., outputs are shorter than inputs), denoted by $\mathcal{G}$, is called *universal one-way*, if given a random function $g \in \mathcal{G}$ and a random (or equivalently, any pre-fixed) input x, it is infeasible for any efficient algorithm to find any $x' \neq x$ satisfying $g(x) = g(x')$. The seminal result that one-way functions (OWFs) imply universal one-way hash functions (UOWHFs) [14] is one of the central pieces upon which modern cryptography is successfully founded. It further implies that digital signature (as defined in [7]) can be based on any one-way function [13]. We mention other important applications of UOWHFs such as constructions of Cramer-Shoup encryption scheme [3] and statistically hiding commitment scheme [9, 10].

UOWHFs from any OWFs. The principle possibility result that UOWHFs can be based on any OWF was established by Rompel [14] (with some corrections given in [15, 11]). However, Rompel's construction was quite complicated and extremely unpractical. In particular, for any one-way function on n-bit inputs it requires key length $\tilde{O}(n^{12})$ and output length $\tilde{O}(n^8)$. Haitner et al. [8] improved the construction via the notion of inaccessible entropy [10], and reduced key and output length to $\tilde{O}(n^8)$. We mention also recent development by Gennaro and Venkitasubramaniam [5] that further reduces the key and output lengths. Despite of all these improvements, the constructions are still too inefficient to be of any practical use.

UOWHFs from special OWFs. Another line of research focuses on more efficient (and nearly practical) constructions of UOWHFs from special structured OWFs. Naor and Yung gave an elegant construction of UOWHFs with key and output length $\Theta(n)$ which does a single call to any one-way permutation. More specifically, let $\tilde{f} : \{0,1\}^n \to \{0,1\}^n$ be any one-way permutation, let h be a random permutation (over n bits) from a pairwise-independent hash permutation family $\mathcal{H}$, and let $\mathsf{trunc} : \{0,1\}^n \to \{0,1\}^{n-1}$ be a truncating function that outputs the first $n-1$ bits of input, then the following

$$\mathcal{G}_{owp} \stackrel{\mathsf{def}}{=} \{\ (\mathsf{trunc}\ \circ\ h\ \circ\ \tilde{f}) : \{0,1\}^n \to \{0,1\}^{n-1} \mid h \in \mathcal{H}\ \}$$

is a family of UOWHFs with 1 bit of shrinkage (i.e., compress by 1 bit), where "$\circ$" denotes function composition. However, for a slightly weaker primitive, namely, 1-to-1 one-way functions, the authors of [13] only gave a rather complicated construction. De Santis and Yung [16] gave an improved construction (from any 1-to-1 OWF $f : \{0,1\}^n \to \{0,1\}^l$) as below:

$$\mathcal{G}_{1-to-1} \stackrel{\mathsf{def}}{=} \{\ (h^n_{n-1}\ \circ\ \ldots\ \circ\ h^{l-1}_{l-2}\ \circ\ h^l_{l-1}\ \circ\ f) : \{0,1\}^n \to \{0,1\}^{n-1} \mid h^n_{n-1} \in \mathcal{H}^n_{n-1}, \ldots, h^l_{l-1} \in \mathcal{H}^l_{l-1}\ \}\ ,$$

where each $\mathcal{H}^i_{i-1}$ denotes a family of pairwise-independent hash functions that compress i-bit strings into $(i-1)$ bits. Although $\mathcal{G}_{1-to-1}$ enjoys linear output length and a single function call, it requires[1] key length $O(\omega(\log n){\cdot}n)$ for describing all the hash functions. In addition, the work of [16] also introduced a construction from any known-regular[2] one-way function with key and output length $O(\omega(\log^2 n) \cdot n)$ and $O(\omega(1) \cdot \log n)$ adaptive calls. Recently, Barhum and Maurer [2] improved the parameters (based on any known regular OWFs) to key and output length $O(\omega(\log n) \cdot n)$ and $O(\omega(1) \cdot \log n)$ non-adaptive calls. Ames et al. [1] presented a construction from any regular one-way function with output length $\Theta(n)$, but it requires key length $O(\mathrm{log}n \cdot n)$ and $\tilde{O}(n)$ adaptive calls. In summary, as tabulated in Table 1, the best known construction requires key length $O(n \cdot \log n)$ even for a 1-to-1 one-way function, and needs to make $O(\omega(1){\cdot} \log n)$ calls (or $\tilde{O}(n)$ adaptive calls if one wants linear output length at the same time) to a regular one-way function.

[1] A back-of-the-envelope calculation suggests that $\mathcal{G}_{1-to-1}$ needs key length $O(l{\cdot}(l-n))$, and we know (see Fact 3.1) that every 1-to-1 one-way function implies another one-way function $f' : \{0,1\}^{n' \in \Theta(n)} \to \{0,1\}^{n' + \omega(\log n)}$ that is 1-to-1 except on a negligible fraction of inputs, which implies that the key length of [13, 16] can be pushed to $O(\omega(\log n){\cdot}n)$.

[2] A function f is regular if every image has the same number (say α) of preimages, and it is known- (resp., unknown-) regular if α is efficiently computable (resp., inefficient to approximate) from the security parameter.

Table 1: A summary of the parameters from existing constructions [13, 16, 2, 1] and our work, where KR-OWF and UR-OWF are the shorthands for known-regular and unknown-regular one-way functions respectively, and ε-hard KR-OWF refers to known-regular one-way function with known hardness parameter ε.

	Assumption	Output Length	Key Length	# of Calls	Type
[13]	OWP	$\Theta(n)$	$\Theta(n)$	1	non-adaptive
[16, 13]	1-to-1 OWF	$\Theta(n)$	$O(\omega(\log n)\cdot n)$	1	non-adaptive
[16]	KR-OWF	$O(\omega(\log^2 n)\cdot n)$	$O(\omega(\log^2 n)\cdot n)$	$O(\omega(\log^2 n))$	adaptive
[2]	KR-OWF	$O(\omega(\log n)\cdot n)$	$O(\omega(\log n)\cdot n)$	$O(\omega(\log n))$	non-adaptive
[1]	UR-OWF	$\Theta(n)$	$O(\log n\cdot n)$	$\tilde{O}(n)$	adaptive
ours	1-to-1 OWF	$\Theta(n)$	$\Theta(n)$	1	non-adaptive
ours	ε-hard KR-OWF	$\Theta(n)$	$\Theta(n)$	1	non-adaptive
ours	KR-OWF	$O(\omega(1)\cdot n)$	$O(\omega(1)\cdot n)$	$O(\omega(1))$	non-adaptive

Summary of our constructions. In the paper, we give the following constructions from any 1-to-1 and known-regular one-way function. The first two constructions achieve optimal parameters simultaneously, and the third is almost optimal up to an arbitrarily small super-constant factor. Our constructions have optimal shrinkages (per invocation of OWF) by matching the upper bound of Gennaro et al. [4].

1. For any 1-to-1 one-way function, we give an optimal construction of UOWHFs with key and output length $\Theta(n)$ and a single OWF call.
2. For any known-regular one-way function with known hardness, we give another optimal construction of UOWHFs with key and output length $\Theta(n)$ and a single call.
3. For any known-regular one-way function, we give a construction of UOWHFs with key and output length $O(\omega(1)\cdot n)$ and $\omega(1)$ non-adaptive calls, where all parameters are optimal up to any efficiently computable super-constant $\omega(1)$ (e.g., $\log\log\log n$ or even less).

On the (a)symmetry to PRGs. Our results further improve the understanding about the inherent "black-box duality" between one-way functions and pseudorandom generators, which was highlighted in [10, 8]. Firstly, we introduce a technical lemma (see Lemma 3.1) which is dual to the leftover hash lemma and might be of independent interest. Informally, it says that when applying a universal hash function h to any random variable X of min-entropy (**no more than**) $\mathbf{a}$ to produce an $(\mathbf{a}+\mathbf{d})$-bit output, h will be injective on X except for a $\mathbf{2^{-d}}$ fraction. In contrast, the leftover hash lemma states that when hashing any X of min-entropy (**no less than**) $\mathbf{a}$ into $(\mathbf{a}-\mathbf{d})$-bit strings, the resulting output distribution will be $\mathbf{2^{-d/2}}$-close (in terms of statistical distance) to uniform, where the symmetry is highlighted in bold. Secondly, the #2 and #3 constructions above match the best known results about constructions of PRGs from known-regular OWFs (see [18]), namely, seed length $O(\omega(1)\cdot n)$ or even $\Theta(n)$ if the hardness of the underlying OWF is known. Finally (and perhaps more interestingly), our #1 construction is asymmetric to the case of PRGs, where we do not know how to construct a linear seed length PRG from an arbitrary 1-to-1 one-way function in general[3].

[3]Given a 1-to-1 one-way function $f:\{0,1\}^n\rightarrow\{0,1\}^l$, one might think of getting a PRG by hashing $f(U_n)$ into $n-s$ bits concatenated with $s+1$ hard-core bits of f, where $s\in\omega(\log n)$ is the necessary entropy loss due to the leftover hash lemma. This is in general not possible without knowing the hardness of the underlying f. See more discussions and the relaxed solutions to this problem by Goldreich [6, Section 3.5.1.3]. For example, we get a linear seed-length PRG of the following weaker form, i.e., for every $\varepsilon=1/\mathsf{poly}(n)$ there exists a weak PRG whose output distribution is ε-indistinguishable from uniform to all PPT distinguishers. Alternatively, we use parallel repetition to obtain a standard PRG with seed length $O(\omega(n))$ [18].

The roadmap. We outline below the steps to build UOWHFs based on any 1-to-1 and regular function $f : \{0,1\}^n \to \{0,1\}^l$. In the latter case, we only assume the one-way function is almost regular [6], namely, the preimage size $f^{-1}(y)$ lies in the interval $[\alpha,\ \alpha{\cdot}n^c]$ for some constant c, where regular functions fall into the special case for $c = 0$. We also assume without loss of generality that $l \in O(n)$ for 1-to-1 one-way functions and even $l = n$ for (almost) regular one-way functions, which seems to be folklore but we also provide a full proof (see Fact 3.1) for completeness.

- Based on 1-to-1 OWFs. We adapt the classic Naor-Yung construction (for one-way permutation) to any 1-to-1 one-way function as follows:

  $$\mathcal{G}_1 \stackrel{\mathsf{def}}{=} \{\ (\mathsf{trunc}\ \circ\ h\ \circ\ f) : \{0,1\}^n \to \{0,1\}^{n-s} \mid h \in \mathcal{H}\ \}\ ,$$

  where $\mathcal{H}$ is a family of universal[4] hash permutations on l bits, and $\mathsf{trunc} : \{0,1\}^l \to \{0,1\}^{n-s}$ is a truncating function that outputs the first $n - s$ bits of input. We give a proof that if f is a (t,ε)- 1-to-1 OWF f then the resulting $\mathcal{G}_1$ is a $(t - n^{O(1)},\ 2^s \cdot \varepsilon)$-UOWHF family with key and output length $\Theta(n)$ and shrinkage s (see Definition 2.2 and Definition 2.4 for formal definitions). The construction enjoys optimal parameters and somewhat counter-intuitively the security bound drops only by factor 2^s (which is optimal by [4]) rather than by 2^{l-n+s} (i.e., exponential in the number of bits truncated). Loosely speaking, this is due to that for $l > n$ the range $f(\{0,1\}^n)$ is only a proper subsect of $\{0,1\}^l$. We refer to the proof of Theorem 3.1 and Remark 3.1 for more technical details and further discussions.

- Based on almost-regular ε-hard OWFs. Given an almost-regular f (see Definition 2.3) which is known to be (t,ε)-one-way, i.e., ε is efficiently computable, we define the following function family

  $$\mathcal{G}_2 \stackrel{\mathsf{def}}{=} \{\ g : \{0,1\}^n \to \{0,1\}^{n-s} \mid g(x) = (g_1(x), h_1(x)),\ g_1 = \mathsf{trunc} \circ h \circ f,\ h \in \mathcal{H}\ , h_1 \in \mathcal{H}_1\ \}$$

  where $\mathcal{H}$ is a family of universal hash permutations, and let $\mathcal{H}_1$ and trunc be a family of universal hash functions and the truncating function (both with appropriate output sizes) respectively. We show that $\mathcal{G}_2$ is a UOWHF family with key and output length $\Theta(n)$ and shrinkage s. The rationale is that for any[5] $x \neq x'$ colliding on $g \in \mathcal{G}_2$ it either satisfies "$f(x) = f(x') \wedge h_1(x) = h_1(x')$" or "$f(x){\neq}f(x') \wedge \mathsf{trunc}(h(f(x))) = \mathsf{trunc}(h(f(x')))$". The former is bounded information-theoretically by our hashing lemma, and the latter is computationally bounded (and reducible to the one-wayness of f). We refer to Theorem 4.1 and Lemma 4.1 for the details.

- Based on any known almost-regular OWFs. Finally, we consider any known (almost) regular OWF f whose hardness parameter is ε unknown (i.e., ε is negligible but may not be efficiently computable). In this case, we run q independent copies of f, and we get a construction by making q non-adaptive calls with shrinkage $q \log n$, key and output length $O(q \cdot n)$, where $q \in \omega(1)$ can be any efficiently computable super-constant. The parallel repetition technique was also used in similar contexts (e.g., the construction of PRG from any known regular OWF [18]). We refer to Theorem 4.2 for the detailed construction and proof.

[4]Most existing UOWHF constructions use pairwise (or even 3-wise) independent hashing to facilitate the analysis, but in fact universal hashing suffices here. Concretely, $\{h : h(x) = h \cdot x,\ \text{where}\ h, x{\in}GF(2^n)\}$ and $\{h_{a,b} : h_{a,b}(x) = a \cdot x + b,\ \text{where}\ x, a, b{\in}GF(2^n)\}$ are universal and pairwise-independent hash function families respectively and it is easy to see that the additional b adds no security to the Naor-Yung UOWHF construction.

[5]More precisely, x is sampled at random and x' can be any efficient function of x such that $x \neq x'$.

2 Preliminaries

Notations and definitions. We use $[n]$ to denote set $\{1, \ldots, n\}$. We use capital letters (e.g., X, Y) for random variables, standard letters (e.g., x, y) for values, and calligraphic letters (e.g. $\mathcal{X}$, $\mathcal{Y}$) for sets. The support of a random variable X, denoted by $\mathsf{Supp}(X)$, refers to the set of values on which X takes with non-zero probability, i.e.,

$$\mathsf{Supp}(X) \stackrel{\text{def}}{=} \{x : \Pr[X = x] > 0\}$$

For a binary string $x = x_1 \ldots x_n$, denote by $x_{[t]}$ the first t bits of x, i.e., $x_1 \ldots x_t$. We denote by $\mathsf{trunc} : \{0,1\}^n \to \{0,1\}^t$ a truncating function that outputs the first t bits of input, i.e., $\mathsf{trunc}(x) = x_{[t]}$. $|\mathcal{S}|$ denotes the cardinality of set $\mathcal{S}$. For function $f : \{0,1\}^n \to \{0,1\}^{l(n)}$, we use shorthand $f(\{0,1\}^n) \stackrel{\mathsf{def}}{=} \{f(x) : x \in \{0,1\}^n\}$, and denote by $f^{-1}(y)$ the set of y's preimages under f, i.e., $f^{-1}(y) \stackrel{\mathsf{def}}{=} \{x : f(x) = y\}$. We say f is length-preserving (resp., linear-preserving) if $l(n) = n$ (resp., $l(n) \in O(n)$). We use $s \leftarrow S$ to denote sampling an element s according to distribution S, and let $s \xleftarrow{\$} \mathcal{S}$ denote sampling s uniformly from set $\mathcal{S}$, and $y := f(x)$ denote value assignment. We use U_n and $U_{\mathcal{X}}$ to denote uniform distributions over $\{0,1\}^n$ and $\mathcal{X}$ respectively, and let $f(U_n)$ be the distribution induced by applying function f to U_n. We use $\mathsf{CP}(X)$ to denote the collision probability of X, and denote by $\mathsf{CP}(X|Z)$ the average collision entropy of X conditioned on another (possibly correlated) random variable Z by

$$\mathsf{CP}(X|Z) \stackrel{\text{def}}{=} \mathbb{E}_{z \leftarrow Z}\left[\ \sum_x \Pr[X = x|\ Z = z]^2\ \right] \quad .$$

Simplifying Notations. To simplify the presentation, we use the following simplified notations. Throughout, most parameters are functions of the security parameter n (e.g., $t(n)$, $\varepsilon(n)$, $r(n)$) and we often omit n when clear from the context (e.g., t, ε, r). Parameters (e.g., ε, r) are said to be known if they are polynomial-time computable from n. By notation $f : \{0,1\}^n \to \{0,1\}^l$ we refer to the ensemble of functions $\{f_n : \{0,1\}^n \to \{0,1\}^{l(n)}\}_{n\in\mathbb{N}}$. As slight abuse of notion, poly might be referring to the set of all polynomials or a certain polynomial, and h might be either a function or its description which will be clear from context. For example, in $h(y) \stackrel{\mathsf{def}}{=} h \cdot y$ the first h denotes a function, the second h refers to a string (a finite field element) that describes the function, and '$\cdot$' denotes multiplication between elements of a finite field.

Definition 2.1 (ρ-almost universal hashing) *A family of functions $\mathcal{H} = \{h : \{0,1\}^l \to \{0,1\}^t\}$ is ρ-almost universal if for any distinct $x_1, x_2 \in \{0,1\}^l$, it holds that*

$$\Pr_{h \xleftarrow{\$} \mathcal{H}} [h(x_1) = h(x_2)] \leq \rho \quad .$$

In the special case $\rho = 2^{-t}$, we say that $\mathcal{H}$ is universal.

It is folklore that almost universal families of hash functions can be efficiently constructed.

Fact 2.1 (efficient constructions of almost universal hashing) *For any integers $t \leq l$, there exists a family of $O(l/t) \cdot 2^{-t}$-almost universal hash functions $\mathcal{H} = \{h : \{0,1\}^l \to \{0,1\}^t\}$ such that $\mathcal{H}$ has description length $O(t)$ and every $h \in \mathcal{H}$ is computable in time* $\mathsf{poly}(l)$.

A concrete example. Assume without loss of generality that t divides l, i.e., $l = k \cdot t$ for some $k \in \mathbb{N}$ (otherwise use $l' = \lceil (l/t) \rceil \cdot t$ instead of l), and parse x as a sequence of t-bit strings $(x_1, \ldots, x_k)$. Then, we have that $\mathcal{H} = \{h_a : h_a(x) \stackrel{\text{def}}{=} \sum_{i=1}^{k} a^i \cdot x_i,\ a, x_i \in GF(2^t)\}$ is a family of $k \cdot 2^{-t}$-almost universal hash functions of description length t. In fact, we could use explicit almost pairwise independent hash functions [12, 17] to achieve even smaller ρ (e.g., $\rho = O(2^{-t})$ for any $l \in \mathsf{poly}(t)$), but the above construction already suffices for our applications.

Definition 2.2 (one-way functions) *A function $f : \{0,1\}^n \rightarrow \{0,1\}^{l(n)}$ is $(t(n)$,$\varepsilon(n))$-one-way if f is polynomial-time computable and for any probabilistic algorithm* A *of running time $t(n)$*

$$\Pr_{y \leftarrow f(U_n)}[\mathsf{A}(1^n, y) \in f^{-1}(y)] \ \leq \ \varepsilon(n).$$

f is a one-way function if $t(\cdot)$ and $1/\varepsilon(\cdot)$ are super-polynomial.

Definition 2.3 ((almost) regular functions) *A function f is $\alpha(n)$-regular if there exists an integer function $\alpha(n)$, called the regularity function, such that for every $n \in \mathbb{N}$ and $x \in \{0,1\}^n$ we have*

$$|f^{-1}(f(x))| = \alpha(n).$$

For any constant c, f is $(\alpha(n),\ \alpha(n){\cdot}n^c)$-almost regular if for every $n \in \mathbb{N}$ and $x \in \{0,1\}^n$ we have

$$\alpha(n) \ \leq \ |f^{-1}(f(x))| \ \leq \ \alpha(n) \cdot n^c.$$

In particular, f is known-(almost)-regular if α is polynomial-time computable, or otherwise it is called unknown-(almost)-regular.

Definition 2.4 (UOWHFs [13]) *An ensemble of families of functions $\{\mathcal{G}_n\}_{n\in\mathbb{N}}$, where $\mathcal{G}_n = \{g : \{0,1\}^{\ell(n)} \rightarrow \{0,1\}^{\ell(n)-s(n)}\}$, is a $(t(n)$,$\varepsilon(n))$-universal one-way hash function (UOWHF) family if:*

- *Efficient: Function $\ell(\cdot)$ is a polynomial (i.e., $\ell \in$ poly). Further, for any $n \in \mathbb{N}$, $g \in \mathcal{G}$ and $x \in \{0,1\}^{\ell(n)}$, the value $g(x)$ can be computed in time* poly(n).
- *Shrinking: The difference between input and output lengths (i.e. $s(n)$) is called* ***shrinkage****.*
- *Target Collision Resistant: For any probabilistic algorithm* A *of running time $t(n)$, we have that*

$$\Pr[x \xleftarrow{\$} \{0,1\}^{\ell(n)};\ g \xleftarrow{\$} \mathcal{G}_n;\ x' \leftarrow \mathsf{A}(1^n, x, g) :\ x \neq x' \wedge g(x) = g(x')\ ] \ \leq \ \varepsilon(n) \quad .$$

Standard asymptotic security requires $t(\cdot)$ and $1/\varepsilon(\cdot)$ to be super-polynomial. For succinctness, hereafter we will use shorthand $\mathcal{G} = \{g : \{0,1\}^{\ell(n)} \rightarrow \{0,1\}^{\ell(n)-s(n)}\}$ for $\{\mathcal{G}_n\}_{n\in\mathbb{N}}$ defined above.

3 UOWHFs from 1-to-1 One-way Functions

3.1 A Technical Lemma

We introduce Lemma 3.1 below with nice symmetry to the leftover hash lemma which will be useful in our constructions and might be of independent interest.

Lemma 3.1 (The injective hash lemma) *For any integers a, d, k and l satisfying $a{\leq}l$, let Y be any random variable with $\mathsf{Supp}(Y) \subseteq \{0,1\}^l$ and $\mathbf{H}_\infty(Y) \leq a$, and let $\mathcal{H} \stackrel{\mathsf{def}}{=} \{h\ :\ \{0,1\}^l \rightarrow \{0,1\}^{a+d}\}$ be a family of $(k{\cdot}2^{-(a+d)})$-almost universal functions. Then, we have that*

$$\Pr_{y \leftarrow Y,\ h \xleftarrow{\$} \mathcal{H}}[\ \exists \tilde{y} \in \mathsf{Supp}(Y) :\ \tilde{y} \neq y \wedge h(\tilde{y}) = h(y)\ ] \leq k{\cdot}2^{-d} \ \ .$$

Recall that $k = 1$ corresponds to the special case that $\mathcal{H}$ is universal.

Proof. It is well-known that any Y of min-entropy a is a convex combination of random variables Y_1, $\ldots$, Y_ℓ where every Y_i ($1\le i \le \ell$) is uniformly distributed over a set of size 2^a. We thus assume without loss of generality (see Claim A.1 in Appendix A) that Y is uniform over some set $\mathcal{Y}$ of size 2^a. The (almost) universality of $\mathcal{H}$ implies an upper bound on $\mathsf{CP}(H(Y)|H)$, i.e.,

$$\mathsf{CP}(\ H(Y)\ |\ H\ ) \le \mathsf{CP}(Y) + \max_{y_1\neq y_2}\{\ \Pr_{h\xleftarrow{\$}\mathcal{H}}[\ h(y_1)=h(y_2)\ ]\ \}\ =\ 2^{-a}(1+k\cdot 2^{-d})\quad .$$

where we consider the random experiment of sampling y_1 and y_2 i.i.d. to Y and thus the collision probability of $H(Y)$ given H is bounded by the sum of $\Pr[Y_1=Y_2]$ and $\Pr[H(y_1)=H(y_2)]$ for any $y_1\neq y_2$.

Further, denote $\mathcal{S}_1 \overset{\mathsf{def}}{=} \{(z,h) : |\{\tilde{y}\in\mathcal{Y} : h(\tilde{y})=z\}|=1\}$ and $\mathcal{S}_2 \overset{\mathsf{def}}{=} \{(z,h) : |\{\tilde{y}\in\mathcal{Y} : h(\tilde{y})=z\}|\ge 2\}$. We then have the following lower bound

$$\begin{aligned}
\mathsf{CP}\ &\ (\ H(Y)\ |\ H\ )\\
=\ &\ \sum_h \Pr[H=h]\left(\sum_{z:(z,h)\in\mathcal{S}_1}\Pr[H(Y)=z|H=h]^2+\sum_{z:(z,h)\in\mathcal{S}_2}\Pr[H(Y)=z|H=h]^2\right)\\
\ge\ &\ 2^{-a}\cdot\sum_{(z,h)\in\mathcal{S}_1}\Pr[H=h,H(Y)=z]\ +\ \min_{(z,h)\in\mathcal{S}_2}\{\ \Pr[h(Y)=z]\ \}\cdot\sum_{(z,h)\in\mathcal{S}_2}\Pr[H=h,H(Y)=z]\\
=\ &\ 2^{-a}\cdot\Pr[\ (H(Y),H)\in\mathcal{S}_1\ ]\ +\ 2^{-a+1}\cdot\Pr[\ (H(Y),H)\in\mathcal{S}_2\ ]\\
=\ &\ 2^{-a}(1+\Pr[\ (H(Y),H)\in\mathcal{S}_2])\quad ,
\end{aligned}$$

where the inequality is due to that any $(z,h)\in\mathcal{S}_1$ satisfies $\Pr[h(Y)=z]=2^{-a}$, and for any $(z,h)\in\mathcal{S}_2$ we have $\Pr[h(Y)=z]\ge 2^{-a+1}$ (recall that Y is uniform over $\mathcal{Y}$ by assumption). Taking into account both the lower and upper bounds on $\mathsf{CP}(H(Y)|H)$, we get $\Pr[\ (H(Y),H)\in\mathcal{S}_2\ ]\le k\cdot 2^{-d}$ and thus complete the proof. □

3.2 Simplifying Assumption about Output Length

We argue that the input and output lengths of a 1-to-1 one-way function $f:\{0,1\}^n\to\{0,1\}^{l(n)}$ can be assumed to be linearly related (i.e., $l(n)\in O(n)$) without loss of generality. In case of almost-regular one-way functions, we can even assume that they are length-preserving (i.e., $l(n)=n$). We state it as Fact 3.1 below, which in turns refers to Lemma 3.2 whose proof is deferred to Appendix B.

Fact 3.1 (two folklore facts) *For any constant c and any efficiently computable $\kappa=\kappa(n)\in O(n)$, we have*

1. *Any 1-to-1 (t,ε)-one-way function $f:\{0,1\}^n\to\{0,1\}^l$ implies a $(t-n^{O(1)},\varepsilon+\mathsf{poly}(n)\cdot 2^{-\kappa})$-one-way function $f':\{0,1\}^{n'\in\Theta(n)}\to\{0,1\}^{(n'+\kappa)\in\Theta(n)}$ which is 1-to-1 except on a $(\mathsf{poly}(n)\cdot 2^{-\kappa})$-fraction of inputs, i.e.,*

$$\Pr_{x\xleftarrow{\$}\{0,1\}^{n'}}[\ \exists x'\in\{0,1\}^{n'} : x'\neq x\ \wedge\ f'(x)=f'(x')\ ]\ \le\ \mathsf{poly}(n)\cdot 2^{-\kappa}$$

2. *Any $(2^r,2^r\cdot n^c)$-almost regular (t,ε)-one-way function $f:\{0,1\}^n\to\{0,1\}^l$ implies a length-preserving $(t-n^{O(1)},\varepsilon+\mathsf{poly}(n)\cdot 2^{-\kappa-r})$-one-way function $\bar{f}:\{0,1\}^{n'\in\Theta(n)}\to\{0,1\}^{n'}$ which is $(2^{r+\kappa},2^{r+\kappa}\cdot n^c)$-almost regular except on a $(\mathsf{poly}(n)\cdot 2^{-\kappa-r})$-fraction of inputs, i.e.,*

$$\Pr_{x\xleftarrow{\$}\{0,1\}^{n'}}[\ 2^{r+\kappa}\ \le\ |\bar{f}^{-1}(\bar{f}(x))|\ \le\ 2^{r+\kappa}\cdot n^c\ ]\ \ge\ 1-\mathsf{poly}(n)\cdot 2^{-\kappa-r}\quad .$$

Note that it suffices to set $\kappa = \omega(\log n)$ to have a negligible error bound, and in case $\kappa = \Theta(n)$ the bound will be exponentially small.

Proof. The first statement immediately follows from Lemma 3.2 by setting regularity $c = r = 0$. As for the second statement, let f' be as defined in (1) from Lemma 3.2, we further define a padded function $\bar{f} : \{0,1\}^{n+\kappa} \times \mathcal{H} \to \{0,1\}^{n+\kappa} \times \mathcal{H}$ as

$$\bar{f}(x, \mathsf{dummy}, h) \stackrel{\text{def}}{=} f'(x, h) \ ,$$

where $x \in \{0,1\}^n$, $\mathsf{dummy} \in \{0,1\}^\kappa$, and $h \in \mathcal{H}$ (which is of size $O(n)$). Note that the preimage-size of $\bar{f}$ is multiplied by a factor of 2^κ than that of f' due to the κ-bit padding dummy, which concludes the statement. $\square$

Lemma 3.2 (regularity-preserving OWF) *For any constant c and any efficiently computable $\kappa = \kappa(n) \in O(n)$, let $f : \{0,1\}^n \to \{0,1\}^l$ be any $(2^r, 2^r{\cdot}n^c)$-almost regular (t,ε)-one-way function, let $\mathcal{H} = \{h : \{0,1\}^l \to \{0,1\}^{n+\kappa}\}$ be a family of $(\mathsf{poly}(n){\cdot}2^{-(n+\kappa)})$-almost universal hash functions with description length[6] $O(n)$, define function $f' : \{0,1\}^n \times \mathcal{H} \to \{0,1\}^{n+\kappa} \times \mathcal{H}$ as*

$$f'(x, h) = (h(f(x)), h) \quad . \tag{1}$$

Then, we have

1. Regularity-preserving. *f' is $(2^r, 2^r{\cdot}n^c)$-regular except on a $\mathsf{poly}(n){\cdot}2^{-\kappa-r}$-fraction of inputs, i.e.,*

$$\Pr_{x \xleftarrow{\$} \{0,1\}^n,\ h \xleftarrow{\$} \mathcal{H}} [\ 2^{r(n)} \ \le\ |f'^{-1}(f'(x,h))| \le 2^r{\cdot}n^c\ ]\ \ge\ 1 - \mathsf{poly}(n){\cdot}2^{-\kappa-r} \quad .$$

2. Hardness-preserving. *f' is a $(t - n^{O(1)},\ \varepsilon + \mathsf{poly}(n){\cdot}2^{-(\kappa+r)})$-one-way function.*

3.3 UOWHFs from 1-to-1 OWFs

We will assume in the remainder of the paper that the underlying 1-to-1 one-way function has linear output length (i.e., $l(n){\in}O(n)$) and that the almost-regular one-way function in consideration is length-preserving (i.e., $l(n) = n$). Our first result below adapts the Naor-Yung construction to any 1-to-1 one-way functions.

Theorem 3.1 (UOWHFs from 1-to-1 OWFs) *Let $f : \{0,1\}^n \to \{0,1\}^{l(n) \in O(n)}$ be any 1-to-1 $(t(n), \varepsilon(n))$-one-way function, let $\mathcal{H}$ be a family of universal hash permutations over $\{0,1\}^{l(n)}$, i.e.,*

$$\mathcal{H} = \{h : \{0,1\}^{l(n)} \to \{0,1\}^{l(n)} \mid h(y) \stackrel{\mathsf{def}}{=} h \cdot y, \quad \textit{where}\ \ y \in GF(2^{l(n)}),\ \vec{0}{\neq}h \in GF(2^{l(n)})\ \} \ ,$$

let $\mathsf{trunc} : \{0,1\}^{l(n)} \to \{0,1\}^{n-s(n)}$ be a truncating function, where $s(n)$ is efficiently computable. Then, we have that

$$\mathcal{G}_1 \stackrel{\mathsf{def}}{=} \{\ (\mathsf{trunc} \circ h \circ f\ ) : \{0,1\}^n \to \{0,1\}^{n-s(n)} \mid h \in \mathcal{H}\ \}$$

is a family of $(t - n^{O(1)},\ 2^s \cdot \varepsilon)$-universal one-way hash functions with shrinkage $s(n)$, key and output length $\Theta(n)$.

Proof. Suppose for contradiction that there exists a $\mathcal{G}_1$-collision finder A of running time t' that on input (x, h), breaks the target collision resistance with some non-negligible probability ε', i.e.,

$$\Pr_{x \xleftarrow{\$} \{0,1\}^n, h \xleftarrow{\$} \mathcal{H}} [\ x' {\leftarrow} \mathsf{A}(x, h) :\ x \neq x' \wedge h(f(x))_{[n-s]} = h(f(x'))_{[n-s]}\ ]\ \ge\ \varepsilon'$$

We define algorithm $\mathsf{Inv}^{\mathsf{A}}$ (that inverts f on input[7] $y^* \in \{0,1\}^l$ by invoking A) as in Algorithm 1. By Claim 3.1, conditioned on $f(x){\neq}y^*$ it is equivalent to consider that $\mathsf{Inv}^{\mathsf{A}}$ samples (x,h,v) from

[6] Such efficient $\mathcal{H}$ exists for any efficiently computable $l = l(n) \in \mathsf{poly}(n)$ and $\kappa = \kappa(n) \in O(n)$ by Fact 2.1.

[7] Notice that y^* is uniformly sampled from $\{0,1\}^l$ rather than from $f(U_n)$, which is a crucial step of our reduction.

Algorithm 1 $\mathsf{Inv}^{\mathsf{A}}$ that inverts f on input y^* using random coins (x,v).

```
Input: $y^* \xleftarrow{\$} \{0,1\}^l$
  Sample $x \xleftarrow{\$} \{0,1\}^n$
  if $f(x) = y^*$ then
    Output $x$ and terminate.
  end if
  sample $h := (f(x)-y^*)^{-1}\cdot v$, where $v \xleftarrow{\$} \mathcal{V} = \{v \in \{0,1\}^l \setminus \{\vec{0}\} : v_{[n-s]} = \overbrace{0\ldots0}^{n-s}\}$
  {note: The above implies $h \xleftarrow{\$} \{h \in \mathcal{H} :\ h(f(x))_{[n-s]} = h(y^*)_{[n-s]}\}$ by the $GF(2^l)$ arithmetics. }
  $x' \leftarrow \mathsf{A}(x,h)$
  if $f(x')$=$y^*$ then
    Output $x'$
  else
    Output $\perp$
  end if
  Terminate
```

$\{0,1\}^n \times \mathcal{H} \times \mathcal{V}$ uniformly and independently, and then determines the value of y^*. We argue that $\mathsf{Inv}^{\mathsf{A}}$ inverts f with the following probability (see the rationale below)

$$\begin{aligned}
&\Pr_{y^*\xleftarrow{\$}\{0,1\}^l,\ x\xleftarrow{\$}\{0,1\}^n,\ v\xleftarrow{\$}\mathcal{V}}[\ f(\mathsf{Inv}^{\mathsf{A}}(y^*)) = y^*\ ]\\
\geq\ & \Pr_{x\xleftarrow{\$}\{0,1\}^n, y^*\xleftarrow{\$}\{0,1\}^l}[\ f(x)=y^*\ ]\ +\ \Pr_{x\xleftarrow{\$}\{0,1\}^n, y^*\xleftarrow{\$}\{0,1\}^l}[\ f(x)\neq y^*\ ]\\
&\times \Pr_{x\xleftarrow{\$}\{0,1\}^n, h\xleftarrow{\$}\mathcal{H}, x\neq(x'\leftarrow\mathsf{A}(x,h)), v\xleftarrow{\$}\mathcal{V}}[\ h(f(x))_{[n-s]} = h(f(x'))_{[n-s]}\ \wedge\ y^* = f(x')\ |\ f(x)\neq y^*\ ]\\
\geq\ & 2^{-l}\ +\ (1-2^{-l})\cdot\varepsilon\cdot \Pr_{v\xleftarrow{\$}\mathcal{V}}[y^* = f(x')\ |\ f(x)\neq y^*\ \wedge\ f(x)\neq f(x')\wedge\ h(f(x))_{[n-s]} = h(f(x'))_{[n-s]}\ ]\\
=\ & 2^{-l}\ +\ (1-2^{-l})\cdot\varepsilon'\cdot\frac{1}{|\mathcal{V}|}\ =\ 2^{-l}\ +\ (1-2^{-l})\cdot\varepsilon'\cdot\frac{1}{2^{l-n+s}-1}\quad,
\end{aligned}$$

where A takes only x and h as input (i.e., independent of v), and thus conditioned on that A produces a valid $x' \neq x$ satisfying $h(f(x'))_{[n-s]} = h(f(x))_{[n-s]}$, we have by Claim 3.1 that string y^* is uniformly distributed over set $\mathcal{Y}^* \overset{\mathsf{def}}{=} \{y^* : y^* = f(x) - v{\cdot}h^{-1}, v \in \mathcal{V}\}$. Note that the already fixed $f(x')$ is also an element of $\mathcal{Y}^*$ and thus y^* hits $f(x')$ with probability $1/|\mathcal{Y}^*|$=$1/|\mathcal{V}|$= $1/(2^{l-n+s}-1)$. On the other hand, we have that inverting f on random y^* is upper bounded by:

$$\begin{aligned}
&\Pr_{y^*\xleftarrow{\$}\{0,1\}^l}[\ f(\mathsf{Inv}^{\mathsf{A}}(y^*)) = y^*\ ]\\
\leq\ & \Pr_{y^*\xleftarrow{\$}\{0,1\}^l}[y^* \in f(\{0,1\}^n)]\cdot \Pr_{y^*\xleftarrow{\$}f(\{0,1\}^n)}[\ f(\mathsf{Inv}^{\mathsf{A}}(y^*)) = y^*\ ]\\
\leq\ & 2^{-(l-n)}\cdot\varepsilon\quad.
\end{aligned}$$

Therefore, it must hold that $2^{-l}\ +\ (1-2^{-l})\varepsilon'\cdot\frac{1}{2^{l-n+s}-1}\ \leq\ 2^{-(l-n)}\cdot\varepsilon$, i.e.,

$$\varepsilon'\ \leq\ (2^{-(l-n)}\cdot\varepsilon - 2^{-l})\cdot(2^{l-n+s}-1)/(1-2^{-l})\ \leq\ 2^{-(l-n)}\cdot\varepsilon{\cdot}2^{l-n+s}\ =\ \varepsilon{\cdot}2^s$$

as otherwise it leads to a contradiction to the upper and lower bounds we show above. ☐

Claim 3.1 (equivalent sampling) *Let the values h, v, x, y^* be sampled as in Algorithm 1 (or as in Algorithm 3), and conditioned on the event $y^* \neq f(x)$, it is equivalent to sample $(x,\ h,\ v) \xleftarrow{\$} \{0,1\}^n \times \mathcal{H} \times \mathcal{V}$ uniformly and independently and then determine $y^* := f(x) - v \cdot h^{-1}$.*

Proof of Claim 3.1. We known that (x, v) is uniformly sampled from $\{0,1\}^n \times \mathcal{V}$ by definition, and thus it suffices to show that "fix any (x, v), and conditioned on $y^* \neq f(x)$ (i.e., Y^* is uniform distributed over $\{0,1\}^l \setminus \{f(x)\}$), it holds that h is uniform over $\mathcal{H}$". As $v \neq \vec{0}$ ($\mathcal{V}$ excludes $\vec{0}$ by definition), it follows that $h = (f(x) - Y^*)^{-1} \cdot v$ is uniform over $\{0,1\}^l \setminus \{\vec{0}\}$. Finally, for any given $(x,\ h,\ v)$, one efficiently determines the value $y^* = f(x) - v \cdot h^{-1}$ due to the arithmetics over the finite field. □

Remark 3.1 (On the optimal security bounds.) *Theorem 3.1 enjoys optimal security degradations, in particular, the collision resistance deteriorates exponentially only with respect to shrinkage s (which is optimal by [4]), i.e., not to the number of bits truncated (i.e. $l-n+s$). This is due to the fact that we reduce the collision-finding problem to that of inverting a random y^* over $\{0,1\}^l$, where the probability that y^* is valid image (i.e., over $f(\{0,1\}^n)$) is $2^{-(l-n)}$ and thus cancel the factor $(l-n)$.*

4 UOWHFs from Known Regular OWFs

We proceed to the more general case that f is a known almost-regular function. Recall that by Fact 3.1 we can assume that the underlying almost regular one-way function is length-preserving without loss of generality.

4.1 Compressing the Output is Necessary but Not Sufficient

We may generalize the Naor-Yung approach for one-way permutations (and 1-to-1 one-way functions) to almost regular one-way functions by compressing (using $\mathsf{trunc} \circ h$) the output $Y = f(X)$ into $\mathbf{H}_\infty(Y) - s$ bits, where $s \in O(\log(1/\varepsilon))$. However, this only gives a weak form of guarantee, as stated in Lemma 4.1 below, that given a random x it is infeasible for efficient algorithms to find any $f(x') \neq f(x)$ such that $\mathsf{trunc}(h(f(x'))) = \mathsf{trunc}(h(f(x)))$. Otherwise said, it does not rule out the possibility that one may easily find $x' \neq x$ satisfying $f(x') = f(x)$. Hence, compressing the output is only a useful intermediate step to obtain UOWHFs. Lemma 4.1 below further generalizes Theorem 3.1 to regular functions, whose proof is similar to that of Theorem 3.1 and thus we defer it to Appendix B to avoid redundancy.

Lemma 4.1 *For any constant c, and any efficiently computable $r = r(n)$ and $s' = s'(n)$, let $f : \{0,1\}^n \to \{0,1\}^n$ be any $(2^r, 2^r n^c)$-almost regular (length-preserving) $(t,\ \varepsilon)$-one-way function, let $\mathcal{H}$ be a family of universal hash permutations over $\{0,1\}^n$, i.e.,*

$$\mathcal{H} = \{h : \{0,1\}^n \to \{0,1\}^n \mid h(y) \overset{\mathsf{def}}{=} h \cdot y, \quad \textit{where} \quad y \in GF(2^n),\ \vec{0} \neq h \in GF(2^n)\ \} \ ,$$

let $\mathsf{trunc} : \{0,1\}^n \to \{0,1\}^{n-r-c\cdot\log n - s'}$ be a truncating function. Then, for any $\tilde{\mathsf{A}}$ of running time $t - n^{O(1)}$ (for some universal constant $O(1)$) we have that

$$\Pr_{x \xleftarrow{\$} \{0,1\}^n,\ h \xleftarrow{\$} \mathcal{H}} [\ x' \leftarrow \tilde{\mathsf{A}}(x,h)\ \wedge\ f(x) \neq f(x')\ \wedge\ \mathsf{trunc}(h(f(x))) = \mathsf{trunc}(h(f(x')))\ ]\ \leq\ n^c \cdot 2^{s'} \cdot \varepsilon\ .$$

4.2 UOWHFs from Known (Almost-)Regular OWFs with Known Hardness

We first give an optimal construction assuming that the inversion probability upper bound ε is known.

Theorem 4.1 (UOWHFs from known almost-regular OWFs with known ε) *Let $f : \{0,1\}^n \to \{0,1\}^n$ be any $(2^r, 2^r n^c)$-almost regular (length-preserving) (t,ε)-one-way function, where c is any constant, and $r = r(n)$ and $\varepsilon = \varepsilon(n)$ are any efficiently computable functions. Let shrinkage $s = s(n)$ be any efficiently computable function, and let $\mathcal{H}$ and* trunc *be as defined in Lemma 4.1 with $s' = (s + \log(1/\varepsilon) - c \log n)/2$, and let $\mathcal{H}_1 = \{h_1 : \{0,1\}^n \to \{0,1\}^{r+c\log n+s'-s}\}$ be a family of universal hash functions. Then, we have that*

$$\mathcal{G}_2 \stackrel{\mathsf{def}}{=} \{\ g : \{0,1\}^n \to \{0,1\}^{n-s} \mid g(x) \stackrel{\mathsf{def}}{=} (g_1(x), h_1(x)), g_1 \stackrel{\mathsf{def}}{=} (\mathsf{trunc} \circ h \circ f),\ h \in \mathcal{H}\ , h_1 \in \mathcal{H}_1\ \}$$

is a $(t - n^{O(1)}, O(\sqrt{2^s \cdot n^c \cdot \varepsilon}))$-universal one-way hash function family with key and output length $\Theta(n)$.

Proof. Denote events $\mathcal{E}_1 \stackrel{\mathsf{def}}{=} \big(f(x) = f(x')\ \wedge h_1(x) = h_1(x')\big)$ and $\mathcal{E}_2 \stackrel{\mathsf{def}}{=} \big(f(x) \neq f(x')\ \wedge g_1(x) = g_1(x')\big)$. For any $\mathcal{G}_2$-collision finder A, we have (with explanations below)

$$\begin{aligned}
&\Pr_{x \xleftarrow{\$} \{0,1\}^n,\ (h,h_1) \xleftarrow{\$} (\mathcal{H},\mathcal{H}_1),\ x \neq (x' \leftarrow \mathsf{A}(x,h,h_1))} [\ g(x) = g(x')\ ] \\
\leq\ &\Pr_{x \xleftarrow{\$} \{0,1\}^n,\ (h,h_1) \xleftarrow{\$} (\mathcal{H},\mathcal{H}_1),\ x \neq (x' \leftarrow \mathsf{A}(x,h,h_1))} [\ \mathcal{E}_1\ \vee\ \mathcal{E}_2\ ] \\
\leq\ &\Pr_{x \xleftarrow{\$} \{0,1\}^n,\ h_1 \xleftarrow{\$} \mathcal{H}_1} [\ \exists\ x' \neq x\ \wedge\ f(x) = f(x')\ \wedge\ h_1(x) = h_1(x')\ ] \\
&+ \Pr_{x \xleftarrow{\$} \{0,1\}^n,\ (h,h_1) \xleftarrow{\$} (\mathcal{H},\mathcal{H}_1),\ x \neq (x' \leftarrow \mathsf{A}(x,h,h_1))} [\ f(x) \neq f(x')\ \wedge\ g_1(x) = g_1(x')\ ] \\
\leq\ & 2^{-(s'-s)}\ +\ n^c \cdot 2^{s'} \cdot \varepsilon\ =\ \sqrt{2^s \cdot n^c \cdot \varepsilon}\ +\ \sqrt{2^s \cdot n^c \cdot \varepsilon}\ =\ 2\sqrt{2^s \cdot n^c \cdot \varepsilon}\ ,
\end{aligned}$$

where the second inequality follows by a union bound, namely, for a random x, if there is some $x' \neq x$ colliding on $g \in \mathcal{G}_2$ then it must either satisfy $\mathcal{E}_1$ or $\mathcal{E}_2$. We already know by Lemma 4.1 that the second term is bounded by $n^c \cdot 2^{s'} \varepsilon$, and thus it remains to show that the first term is bounded by $2^{-(s'-s)}$. Conditioned on any $y = f(X)$ random variable X has min-entropy $\mathbf{H}_\infty(X | f(X) = y) \leq r + c \cdot \log n$, so we apply Lemma 3.1 (setting $a = r + c \cdot \log n$, $d = s' - s$ and $k = 1$) to get

$$\begin{aligned}
&\Pr_{x \xleftarrow{\$} \{0,1\}^n,\ h_1 \xleftarrow{\$} \mathcal{H}_1} [\ \exists\ x' \neq x\ \wedge\ f(x) = f(x')\ \wedge\ h_1(x) = h_1(x')\ ] \\
=\ &\mathbb{E}_{y \leftarrow f(U_n)}\Big[\Pr_{x \xleftarrow{\$} f^{-1}(y),\ h_1 \xleftarrow{\$} \mathcal{H}_1} [\ \exists\ x' \neq x\ \wedge\ f(x) = f(x')\ \wedge\ h_1(x) = h_1(x')\ ]\ \Big] \\
\leq\ &\mathbb{E}_{y \leftarrow f(U_n)}[\ 2^{-(s'-s)}\ ]\ = 2^{-(s'-s)}\ .
\end{aligned}$$

which completes the proof. $\square$

4.3 UOWHFs from any Known (Almost-)Regular OWFs

Generalizing to any known regular function. Unfortunately, Theorem 4.1 doesn't immediately apply to an arbitrary regular function because in general we cannot assume that the hardness parameter ε is known or efficiently computable. To see the difficulty, consider the parameters in Theorem 4.1, where we need to decide the values of s and s'. That is, to get a shrinkage of s bits, the hash function h_1 first expands $s' - s$ bits (than the actual entropy of X conditioned on $f(X)$) to get an error bound $2^{-(s'-s)}$ (by Lemma 3.1), and then trunc discards s' bits of entropy (of $f(X)$) to get another bound $n^c \cdot 2^{s'} \cdot \varepsilon$. Without the knowledge about ε, one may end up setting some super-polynomial $2^{s'}$ (to make the first term negligible) which kills the second term $n^c \cdot 2^{s'} \cdot \varepsilon$. Same problems arise in similar situations (e.g., construction of PRGs from regular OWFs [18]). A remedy for this is parallel repetition: for any efficiently computable $q \in \omega(1)$, run q copies of f, apply hashing and truncating functions (setting

$s' = 2\log n$) to every copy (to get a bound $O(\varepsilon \cdot n^{c+2})$), which shrinks the entropies by $2q\log n$ bits, and finally apply a single hashing (to the q inputs of f) that expands $q\cdot\log n$ bits (to yield another negligible bound n^{-q}). This gives a family of UOWHFs with shrinkage $2q\log n - q\log n = q\log n$, and key and output length $O(q\cdot n)$ for any (efficiently computable) super-constant q.

Definition 4.1 (parallel repetition) *For any function $g : \mathcal{X} \to \mathcal{Y}$, we define its q-fold parallel repetition $g^q : \mathcal{X}^q \to \mathcal{Y}^q$ as*

$$g^q(x_1, ..., x_q) = (\ g(x_1)\ , ...,\ g(x_q)\ ) \quad .$$

For simplicity, we will use shorthand $\vec{x} \overset{\text{def}}{=} (x_1, \ldots, x_q)$ and thus $g^q(\vec{x}) = g^q(x_1, \ldots, x_q)$.

Theorem 4.2 (UOWHFs from any known almost-regular OWFs) *Let $f : \{0,1\}^n \to \{0,1\}^n$ be any $(2^r, 2^r n^c)$-almost regular (length-preserving) (t,ε)-one-way function, where c is any constant, and $r = r(n)$ is any efficiently computable function. Then, for any efficiently computable $q = q(n) \in \omega(1)$, let $\mathcal{H}$ and* trunc *be as defined in Lemma 4.1 with $s' = 2\log n$, and let $\mathcal{H}_1 = \{h_1 : \{0,1\}^{q\cdot n} \to \{0,1\}^{q(r+(c+1)\log n)}\}$ be a family of universal hash functions, we have that*

$$\mathcal{G}_3 \overset{\mathsf{def}}{=} \{\ g : \{0,1\}^{qn} \to \{0,1\}^{qn - q\log n} \mid g(\vec{x}) \overset{\mathsf{def}}{=} (g_1(\vec{x}), h_1(\vec{x})), g_1 \overset{\mathsf{def}}{=} (\mathsf{trunc} \circ h \circ f)^q,\ h \in \mathcal{H}\ , h_1 \in \mathcal{H}_1\ \}$$

is a $(t - n^{O(1)}, n^{-q} + q\cdot n^{c+2}\cdot\varepsilon)$-universal one-way hash function family with key and output length $O(q\cdot n)$, and shrinkage $q\cdot\log n$.

Proof. Similar to the proof of Theorem 4.1, define $\mathcal{E}_1 \overset{\text{def}}{=} \big(f^q(\vec{x}) = f^q(\vec{x'})\ \wedge h_1(\vec{x}) = h_1(\vec{x'})\big)$ and $\mathcal{E}_2 \overset{\text{def}}{=} \big(f^q(\vec{x}) \neq f^q(\vec{x'})\ \wedge g_1(\vec{x}) = g_1(\vec{x'})\big)$, we have

$$\begin{aligned}
&\Pr_{\vec{x} \xleftarrow{\$} \{0,1\}^{qn},\ (h,h_1) \xleftarrow{\$} (\mathcal{H},\mathcal{H}_1),\ \vec{x} \neq (\vec{x'} \leftarrow \mathsf{A}(\vec{x},h,h_1))} [\ g(\vec{x}) = g(\vec{x'})] \\
\leq\ &\Pr_{\vec{x} \xleftarrow{\$} \{0,1\}^{qn},\ (h,h_1) \xleftarrow{\$} (\mathcal{H},\mathcal{H}_1),\ \vec{x} \neq (\vec{x'} \leftarrow \mathsf{A}(\vec{x},h,h_1))} [\ \mathcal{E}_1\ \vee \mathcal{E}_2\ ] \\
\leq\ &\Pr_{\vec{x} \xleftarrow{\$} \{0,1\}^{qn},\ h_1 \xleftarrow{\$} \mathcal{H}_1} [\ \exists\ \vec{x'} \neq \vec{x}\ \wedge\ f^q(\vec{x}) = f^q(\vec{x'})\ \wedge\ h_1(\vec{x}) = h_1(\vec{x'})\ ] \\
+\ &\Pr_{\vec{x} \xleftarrow{\$} \{0,1\}^{qn},\ (h,h_1) \xleftarrow{\$} (\mathcal{H},\mathcal{H}_1)} [\ \vec{x'} \leftarrow \mathsf{A}(\vec{x},h,h_1)\ \wedge\ f^q(\vec{x}) \neq f^q(\vec{x'})\ \wedge\ g_1(\vec{x}) = g_1(\vec{x'})\ ] \\
\leq\ & 2^{-q\log n}\ +\ q\cdot n^{c+2}\cdot\varepsilon\ =\ n^{-q}\ +\ q\cdot n^{c+2}\cdot\varepsilon\ ,
\end{aligned}$$

where the second inequality follows by a union bound, and the first term of the third inequality is due to that conditioned on any $\vec{y} = f^q(\vec{X})$ random variable $\vec{X}$ has min-entropy $\mathbf{H}_\infty(\vec{X}|f^q(\vec{X}) = \vec{y}) \leq q(r + c\cdot \log n)$, so we apply Lemma 3.1 (setting $a = q(r + c\cdot\log n)$, $d = q\log n$ and $k = 1$) to get

$$\begin{aligned}
&\Pr_{\vec{x} \xleftarrow{\$} \{0,1\}^{qn},\ h_1 \xleftarrow{\$} \mathcal{H}_1} [\ \exists\ \vec{x'} \neq \vec{x}\ \wedge\ f^q(\vec{x}) = f^q(\vec{x'})\ \wedge\ h_1(\vec{x}) = h_1(\vec{x'})\ ] \\
=\ & \mathbb{E}_{\vec{y} \leftarrow f^q(U_{qn})}[\ \Pr_{\vec{x} \xleftarrow{\$} (f^q)^{-1}(\vec{y}),\ h_1 \xleftarrow{\$} \mathcal{H}_1} [\ \exists\ \vec{x'} \neq \vec{x}\ \wedge\ f^q(\vec{x}) = f^q(\vec{x'})\ \wedge\ h_1(\vec{x}) = h_1(\vec{x'})\ ]\ ] \\
\leq\ & \mathbb{E}_{\vec{y} \leftarrow f^q(U_{qn})}[\ 2^{-q\log n}\ ]\ =\ n^{-q} \quad .
\end{aligned}$$

We proceed to the proof of bounding the second term. Suppose for contradiction that there exists A_{g_1} of running time $t - n^{O(1)}$ such that

$$\Pr_{\vec{x} \xleftarrow{\$} \{0,1\}^{qn},\ (h,h_1) \xleftarrow{\$} (\mathcal{H},\mathcal{H}_1)} [\ \vec{x'} \leftarrow \mathsf{A}_{g_1}(\vec{x},h,h_1)\ \wedge\ f^q(\vec{x}) \neq f^q(\vec{x'})\ \wedge\ g_1(\vec{x}) = g_1(\vec{x'})\ ]\ >\ q\cdot n^{c+2}\cdot\varepsilon$$

Algorithm 2 $(\mathsf{trunc} \circ h)$-collision finder $\tilde{\mathsf{A}}$ on input (x, h).

```
Input: $(x,h) \xleftarrow{\$} \{0,1\}^n \times \mathcal{H}$
  Sample $\vec{x} = (x_1, \dots, x_q) \xleftarrow{\$} \{0,1\}^{qn}$,$h_1 \xleftarrow{\$} \mathcal{H}_1$, $i \xleftarrow{\$} [q]$
  $\vec{x'} = (x'_1, \dots, x'_q) \leftarrow \mathsf{A}_{g_1}(\ (x_1, \dots, x_{i-1}, x, x_{i+1}, \dots, x_q)\ , h, h_1)$ {i.e., replace $x_i$ with $x$ }
  return $x'_i$
```

Then, define $\tilde{\mathsf{A}}$ as in Algorithm 2. Conditioned on A_{g_1} finds a collision, i.e., $f^q(\vec{x}) \neq f^q(\vec{x'})$ and $g_1(\vec{x}) = g_1(\vec{x'})$, there exists at least one $i^* \in [q]$ satisfying $f(x_{i^*}) \neq f(x'_{i^*})$ and $\mathsf{trunc}(h(f(x_{i^*}))) = \mathsf{trunc}(h(f(x'_{i^*})))$. We have

$$\begin{aligned}
&\Pr_{x \xleftarrow{\$} \{0,1\}^n,\ h \xleftarrow{\$} \mathcal{H}} [\ x' \leftarrow \tilde{\mathsf{A}}(x,h)\ \wedge\ f(x) \neq f(x')\ \wedge\ \mathsf{trunc}(h(f(x))) = \mathsf{trunc}(h(f(x')))\ ] \\
\geq\ &\Pr_{\vec{x} \xleftarrow{\$} \{0,1\}^{qn},\ (h,h_1) \xleftarrow{\$} (\mathcal{H},\mathcal{H}_1)} [\ \vec{x'} \leftarrow \mathsf{A}_{g_1}(\vec{x}, h, h_1)\ \wedge\ f^q(\vec{x}) \neq f^q(\vec{x'})\ \wedge\ g_1(\vec{x}) = g_1(\vec{x'})\ \wedge\ i = i^*\ ] \\
>\ & q \cdot n^{c+2}\varepsilon \cdot (1/q)\ =\ n^{c+2}\varepsilon\ ,
\end{aligned}$$

which is a contradiction to Lemma 4.1 (recall that $s' = 2\log n$) and thus completes the proof. □

References

[1] Scott Ames, Rosario Gennaro, and Muthuramakrishnan Venkitasubramaniam. The generalized randomized iterate and its application to new efficient constructions of UOWHFs from regular one-way functions. In *ASIACRYPT*, pages 154–171, 2012.

[2] Kfir Barhum and Ueli Maurer. UOWHFs from OWFs: Trading regularity for efficiency. In *LATINCRYPT*, pages 234–253, 2012.

[3] Ronald Cramer and Victor Shoup. Design and analysis of practical public-key encryption schemes secure against adaptive chosen ciphertext attack. *SIAM Journal on Computing*, 33(1):167–226, 2003.

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

[5] Rosario Gennaro and Muthuramakrishnan Venkitasubramaniam. Can you compress where you expand? new and efficient hash functions. announced at the rump session of ASIACRYPT 2013, 2013. http://asiacrypt.2013.rump.cr.yp.to/.

[6] Oded Goldreich. *Foundations of Cryptography: Basic Tools*. Cambridge University Press, 2001.

[7] Shafi Goldwasser, Silvio Micali, and Ronald L. Rivest. A digital signature scheme secure against adaptive chosen-message attacks. *SIAM Journal on Computing*, 17(2):281–308, April 1988.

[8] Iftach Haitner, Thomas Holenstein, Omer Reingold, Salil P. Vadhan, and Hoeteck Wee. Universal one-way hash functions via inaccessible entropy. In *EUROCRYPT*, pages 616–637, 2010.

[9] Iftach Haitner, Minh-Huyen Nguyen, Shien Jin Ong, Omer Reingold, and Salil P. Vadhan. Statistically hiding commitments and statistical zero-knowledge arguments from any one-way function. *SIAM Journal on Computing*, 39(3):1153–1218, 2009.

[10] Iftach Haitner, Omer Reingold, Salil P. Vadhan, and Hoeteck Wee. Inaccessible entropy. In *Proceedings of the 41st ACM Symposium on the Theory of Computing*, pages 611–620, 2009.

[11] Jonathan Katz and Chiu-Yuen Koo. On constructing universal one-way hash functions from arbitrary one-way functions. *IACR Cryptology ePrint Archive*, 2005. http://eprint.iacr.org/2005/328.

[12] Joseph Naor and Moni Naor. Small-bias probability spaces: Efficient constructions and applications. *SIAM Journal on Computing*, 22(4):838–856, 1993.

[13] Moni Naor and Moti Yung. Universal one-way hash functions and their cryptographic applications. In D. S. Johnson, editor, *Proceedings of the Twenty First Annual ACM Symposium on Theory of Computing*, pages 33–43, Seattle, Washington, 15–17 May 1989.

[14] John Rompel. One-way functions are necessary and sufficient for secure signatures. In *Proceedings of the Twenty Second Annual ACM Symposium on Theory of Computing*, pages 387–394, Baltimore, Maryland, 14–16 May 1990.

[15] John Rompel. *Techniques for computing with low-independence randomness.* PhD thesis, Massachusetts Institute of Technology, 1990. http://dspace.mit.edu/handle/1721.1/7582.

[16] Alfredo De Santis and Moti Yung. On the design of provably secure cryptographic hash functions. In *EUROCRYPT*, pages 412–431, 1990.

[17] D. R. Stinson. Universal hashing and authentication codes. *Designs, Codes, and Cryptography*, 4(4):369–380, 1994.

[18] Yu Yu, Xiangxue Li, and Jian Weng. Pseudorandom generators from regular one-way functions: New constructions with improved parameters. In *ASIACRYPT*, pages 261–279, 2013.

A Technical Lemma

Claim A.1 *In Lemma 3.1 it suffices to consider 'flat' Y, i.e., Y is uniformly distributed over set $\mathcal{Y}$ of size 2^a.*

Proof. Let Y be any random variable of min-entropy a. We know Y is a convex combination of random variables $Y_1, \ldots, Y_\ell$, where every Y_i ($1 \le i \le \ell$) is uniformly distributed over a set of size 2^a. Let p_i be the possibility that Y takes values from Y_i, and the probabilities sum to unity (i.e., $\sum_{i=1}^{\ell} p_i = 1$). We have

$$\Pr_{y \leftarrow Y,\ h \xleftarrow{\$} \mathcal{H}} [\ \exists \tilde{y} \in \mathsf{Supp}(Y) :\ \tilde{y} \neq y \wedge h(\tilde{y}) = h(y)\ ]$$

$$= \sum_{i=1}^{\ell} p_i \cdot \Pr_{y \leftarrow Y_i,\ h \xleftarrow{\$} \mathcal{H}} [\ \exists \tilde{y} \in \mathsf{Supp}(Y) :\ \tilde{y} \neq y \wedge h(\tilde{y}) = h(y)\ ]$$

As long as we can bound the term $\Pr_{y \leftarrow Y_i,\ h \xleftarrow{\$} \mathcal{H}} [\ \exists \tilde{y} \in \mathsf{Supp}(Y) :\ \tilde{y} \neq y \wedge h(\tilde{y}) = h(y)\ ]$ by $k \cdot 2^{-d}$ for every flat Y_i of min-entropy a, the above will be bounded by $k \cdot 2^{-d} \cdot \sum_{i=1}^{\ell} p_i = k \cdot 2^{-d}$, which concludes the statement. □

B Proofs Omitted

Proof of Lemma 3.2. As f is $(2^r, 2^r\cdot n^c)$-almost regular it suffices to consider the injectiveness during the composition of f and h, i.e.,

$$\begin{aligned}
&\Pr_{x\xleftarrow{\$}\{0,1\}^n,\ h\xleftarrow{\$}\mathcal{H}}[\ 2^r\ \le\ |f'^{-1}(f'(x,h))|\ \le\ 2^r\cdot n^c\ ]\\
\ge&\ \Pr_{y\leftarrow f(U_n),\ h\xleftarrow{\$}\mathcal{H}}[\ |\{\ \tilde{y}\in f(\{0,1\}^n) : h(\tilde{y}) = h(y)\ \}| = 1\ ]\\
=&\ 1\ -\ \Pr_{y\leftarrow f(U_n),\ h\xleftarrow{\$}\mathcal{H}}[\ \exists\tilde{y}\in f(\{0,1\}^n) : h(\tilde{y}) = h(y)\ \wedge\ y\neq\tilde{y}\ ]\\
\ge&\ 1-\mathsf{poly}(n)\cdot 2^{-(\kappa+r+c\log n)}\ \ge\ 1-\mathsf{poly}(n)\cdot 2^{-(\kappa+r)}\ ,
\end{aligned}$$

where the second inequality is by Lemma 3.1 (setting $Y = f(U_n)$, $a = \mathbf{H}_\infty(Y) = n - r - c\log n$, and thus $d = \kappa + r + c\log n$). Further, it is not hard to see that any f'-inverting algorithm $\mathsf{A}_{f'}$ implies an f-inverting algorithm A_f. That is, on input $f(x)$, A_f applies random h to $f(x)$, and then invokes $\mathsf{A}_{f'}$ on $(h(f(x)), h)$ to recover x. The inversion probability of A_f is

$$\begin{aligned}
&\Pr_{y\leftarrow f(U_n)}[\mathsf{A}_f(y)\in f^{-1}(y)]\\
\ge&\ \Pr_{y\leftarrow f(U_n),h\xleftarrow{\$}\mathcal{H}}[\mathsf{A}_{f'}(h(y),h)\in f'^{-1}(h(y),h)\ \wedge\neg\Big(\exists\tilde{y}\in f(\{0,1\}^n) : h(\tilde{y}) = h(y)\ \wedge\ y\neq\tilde{y}\Big)\ ]\\
=&\ 1\ -\ \Pr_{y\leftarrow f(U_n),h\xleftarrow{\$}\mathcal{H}}[\mathsf{A}_{f'}(h(y),h)\notin f'^{-1}(h(y),h)\ \vee\Big(\exists\tilde{y}\in f(\{0,1\}^n) : h(\tilde{y}) = h(y)\ \wedge\ y\neq\tilde{y}\Big)\ ]\\
\ge&\ 1\ -\ \Pr_{y\leftarrow f(U_n),h\xleftarrow{\$}\mathcal{H}}[\mathsf{A}_{f'}(h(y),h)\notin f'^{-1}(h(y),h)]\ -\ \Pr_{y\leftarrow f(U_n),h\xleftarrow{\$}\mathcal{H}}[\exists\tilde{y}\in f(\{0,1\}^n) : h(\tilde{y}) = h(y)\ \wedge\ y\neq\tilde{y}\ ]\\
\ge&\ \Pr_{y\leftarrow f(U_n),h\xleftarrow{\$}\mathcal{H}}[\mathsf{A}_{f'}(h(y),h)\in f'^{-1}(h(y),h)]\ -\ \mathsf{poly}(n)\cdot 2^{-(\kappa+r)}\ ,
\end{aligned}$$

where the first inequality refers to that A_f inverts f if $\mathsf{A}_{f'}$ inverts f' on those $(h(y), h)$ for which there exists no distinct $\tilde{y}$ satisfying $h(\tilde{y}) = h(y)$, the second inequality is the union bound, and the third is due to the probability that h is not injective on Y as given above. This completes the proof. □

Proof of Lemma 4.1. Suppose for contradiction that there exists an efficient $\tilde{\mathsf{A}}$ of running time t' such that

$$\Pr_{x\xleftarrow{\$}\{0,1\}^n,h\xleftarrow{\$}\mathcal{H}}[\ x'\leftarrow\tilde{\mathsf{A}}(x,h) :\ f(x)\neq f(x')\wedge h(f(x))_{[u]} = h(f(x'))_{[u]}\ ]\ \ge\ \varepsilon'$$

where $u = n - r - c\log n - s'$. We proceed to the definition of algorithm $\mathsf{Inv}^{\tilde{\mathsf{A}}}$ (that inverts f by invoking $\tilde{\mathsf{A}}$) as in Algorithm 3. By Claim 3.1, conditioned on $f(x)\neq y^*$ it is equivalent to consider that $\mathsf{Inv}^{\tilde{\mathsf{A}}}$ samples (x,h,v) from $\{0,1\}^n\times\mathcal{H}\times\mathcal{V}$ uniformly and independently, from which y^* can be determined.

Algorithm 3 $\mathsf{Inv}^{\tilde{\mathsf{A}}}$ that inverts f on input y^* using random coins (x, v).

```
Input: $y^* \xleftarrow{\$} \{0,1\}^n$
  Sample $x \xleftarrow{\$} \{0,1\}^n$
  if $f(x) = y^*$ then
    Output $x$ and terminate.
  end if
  sample $h := (f(x) - y^*)^{-1} \cdot v$, where $v \xleftarrow{\$} \mathcal{V} = \{v \in \{0,1\}^n \setminus \{\vec{0}\} : v_{[u]} = \overbrace{0\ldots0}^{u}\}$
  {note: The above implies $h \xleftarrow{\$} \{h \in \mathcal{H} :\ h(f(x))_{[u]} = h(y^*)_{[u]}\}$ by the algebraic structure of $h$. }
  $x' \leftarrow \tilde{\mathsf{A}}(x, h)$
  if $f(x')$=$y^*$ then
    Output $x'$
  else
    Output $\perp$
  end if
  Terminate
```

Then, we argue that $\mathsf{Inv}^{\tilde{\mathsf{A}}}$ inverts f with the following probability (see the rationale below)

$$\begin{aligned}
&\Pr_{y^*\xleftarrow{\$}\{0,1\}^n,\ x\xleftarrow{\$}\{0,1\}^n,\ v\xleftarrow{\$}\mathcal{V}} [\ f(\mathsf{Inv}^{\tilde{\mathsf{A}}}(y^*)) = y^*\ ] \\
\geq\ & \Pr_{x\xleftarrow{\$}\{0,1\}^n, y^*\xleftarrow{\$}\{0,1\}^n} [\ f(x) = y^*\ ] \ +\ \Pr_{x\xleftarrow{\$}\{0,1\}^n, y^*\xleftarrow{\$}\{0,1\}^n} [\ f(x) \neq y^*\ ] \\
& \times \Pr_{x\xleftarrow{\$}\{0,1\}^n, h\xleftarrow{\$}\mathcal{H}, x\neq(x'\leftarrow\tilde{\mathsf{A}}(x,h)), v\xleftarrow{\$}\mathcal{V}} [\ h(f(x))_{[u]} = h(f(x'))_{[u]}\ \wedge\ y^* = f(x')\ |\ f(x)\neq y^*\ ] \\
\geq\ & 2^{-n}\ +\ (1-2^{-n})\cdot\varepsilon'\cdot \Pr_{v\xleftarrow{\$}\mathcal{V}}[\ y^* = f(x')\ |\ f(x)\neq y^*\ \wedge\ f(x)\neq f(x')\ \wedge\ h(f(x))_{[u]} = h(f(x'))_{[u]}\ ] \\
=\ & 2^{-n}\ +\ (1-2^{-n})\cdot\varepsilon'\cdot\frac{1}{|\mathcal{V}|}\ =\ 2^{-n}\ +\ (1-2^{-n})\cdot\varepsilon'\cdot\frac{1}{2^{r+c\log n+s'}-1}\quad ,
\end{aligned}$$

where $\tilde{\mathsf{A}}$ takes only x and h as input (i.e., independent of v), and thus conditioned on that $\tilde{\mathsf{A}}$ produces a valid $f(x') \neq f(x)$ satisfying $h(f(x'))_{[u]} = h(f(x))_{[u]}$, we have by Claim 3.1 that string y^* is uniformly distributed over set $\mathcal{Y}^* \overset{\mathsf{def}}{=} \{y^* : y^* = f(x) - v{\cdot}h^{-1}, v \in \mathcal{V}\}$. Note that the already fixed $f(x')$ is also an element of $\mathcal{Y}^*$ and thus y^* hits $f(x')$ with probability $1/|\mathcal{Y}^*|$=$1/|\mathcal{V}|$= $1/(2^{n-u}-1)$. On the other hand, we have that inverting f on input y^* is upper bounded by:

$$\begin{aligned}
&\Pr_{y^*\xleftarrow{\$}\{0,1\}^n} [\ f(\mathsf{Inv}^{\tilde{\mathsf{A}}}(y^*)) = y^*\ ] \\
=\ & \sum_{y\in f(\{0,1\}^n)} \Pr_{y^*\xleftarrow{\$}\{0,1\}^n} [y = y^*]{\cdot}\Pr[f(\mathsf{Inv}^{\tilde{\mathsf{A}}}(y)) = y] \\
\leq\ & 2^{-r}\cdot \sum_{y\in f(\{0,1\}^n)} \Pr[f(U_n) = y]{\cdot}\Pr[f(\mathsf{Inv}^{\tilde{\mathsf{A}}}(y)) = y] \\
\leq\ & 2^{-r}\cdot\varepsilon
\end{aligned}$$

where the first inequality is due to $\Pr[f(U_n) = y] \geq 2^{r-n}$ and the second is due to the one-wayness of f. Therefore, it must hold that $2^{-n}\ +\ (1-2^{-n})\cdot\varepsilon'\cdot\frac{1}{2^{r+c\log n+s'}-1}\ \leq\ 2^{-r}\cdot\varepsilon$

$$\varepsilon'\ \leq\ (2^{-r}\varepsilon - 2^{-n})\cdot(2^{r+c\log n+s'}-1)/(1-2^{-n})\ \leq\ 2^{-r}\varepsilon\cdot 2^{r+c\log n+s'}\ =\ n^c\cdot 2^{s'}\varepsilon$$

which completes the proof. $\square$