Amplifying Collision Resistance: A Complexity-Theoretic Treatment

Ran Canetti[1]*, Ron Rivest[2], Madhu Sudan[2],
Luca Trevisan[3]**, Salil Vadhan[4]***, and Hoeteck Wee[3]†

[1] IBM Research, canetti@us.ibm.com
[2] MIT CSAIL, {rivest,madhu}@mit.edu
[3] UC Berkeley, {luca,hoeteck}@cs.berkeley.edu
[4] Harvard University, salil@eecs.harvard.edu

Abstract. We initiate a complexity-theoretic treatment of hardness amplification for collision-resistant hash functions, namely the transformation of weakly collision-resistant hash functions into strongly collision-resistant ones in the standard model of computation. We measure the level of collision resistance by the maximum probability, over the choice of the key, for which an efficient adversary can find a collision. The goal is to obtain constructions with short output, short keys, small loss in adversarial complexity tolerated, and a good trade-off between compression ratio and computational complexity. We provide an analysis of several simple constructions, and show that many of the parameters achieved by our constructions are almost optimal in some sense.

Keywords. collision resistance, hash functions, hardness amplification, combiners

1 Introduction

Constructing collision-resistant hash functions is a central problem in cryptography, both from the foundational and the practical points of view. The goal is to construct length-decreasing functions for which it is infeasible to find two distinct inputs with the same output. This problem has received much attention over the past two decades. Still, coming up with constructions that are efficient enough to be of use in practice and at the same time enjoy rigorous security guarantees (say, based on the hardness of some well-studied problem) has turned out to be elusive. We also seem unable to construct collision-resistant functions from potentially simpler primitives, c.f. [25]. The problem is highlighted by the repeated attacks on the popular MD4, MD5 and SHA1 hash functions (refer to [20] and references therein).

* Supported by NSF grants CFF-0635297 and Cybertrust 0430450.

** Supported by NSF grant CCF-0515231.

*** Supported by NSF grants CNS-0430336 and CCF-0133096, and ONR grant N00014-04-1-0478.

† Work done while visiting IBM Research, IPAM and Columbia University, the latter supported by NSF grants CCF-0515231 and CCF-0347839.

Given this state of affairs, it is natural to ask whether one can "bootstrap" collision resistance by constructing "full-fledged" collision-resistant hash functions (CRHF) from "weak" ones. That is, are there general mechanisms for transforming hash functions, for which it is "somewhat easy" (but not completely trivial) to find collisions, into one for which it is infeasible to find collisions? In addition to providing rigorous ways to improve the collision resistance of hash functions, such mechanisms could in themselves suggest methodologies for constructing hash functions "from scratch".

Several works propose design principles for hash functions, e.g. [17, 4, 14, 3]. These mechanisms can indeed be regarded as "hardness amplification" mechanisms for collision-resistant hash functions. However, with the exception of [4], which concentrates on increasing the domain size of the hash function, all the analyses provided for these mechanisms use idealized models of computation, such as modeling the underlying building blocks as random functions. Consequently, we do not currently have constructions that are guaranteed to provide some level of collision resistance in the standard model of computation, under the sole assumption that the underlying building blocks have some weaker collision resistance properties. (Recently, the closely related problem of constructing "combiners" for hash functions has been studied in the standard model [2, 19]; we discuss this problem in more detail below.)

This state of the art should be contrasted with the "sister problem" of constructing one-way functions. Here we have a well-established theory of hardness amplification [27] (see also [11]). That is, we have concrete notions of "strength" of one-way functions, and constructions that are guaranteed to provide "strong" one-way functions based on the sole assumption that the underlying building block is a "weak" one-way function. Several lower bounds for "black-box" hardness amplification are also known, e.g. [23, 15].

We note that collision resistance often exhibits very different properties than one-wayness. For one, constructing collision-resistant hash functions calls for different design principles (e.g. the proposed expander-based one-way function of [10] is very bad as a collision-resistant function). Furthermore, both practice and theory indicate that collision resistance is considerably harder to achieve than one-wayness, e.g. [6, 26, 25]. Still, except for some specific points highlighted within, we show that it is possible to translate much of the analysis used in the study of amplification of one-wayness to the setting of collision resistance.

1.1 This work.

We initiate a study of amplification of collision resistance, in a standard reductionist complexity-theoretic framework. That is, we first provide a measure for the "level" of collision resistance of hash functions. We then consider some constructions and quantitatively analyze the amount in which they amplify the collision resistance, along with a number of efficiency parameters (discussed below).

Model for hash functions. Following [4], we model hash functions as a *family* of functions, where a function in the family is specified via a *key*. Security is analyzed for the case where the key is chosen at random (from the space of keys) and made public. We point out several advantages of this approach. Refer to [21] for a more detailed discussion. First, it allows for a natural modeling of the adversary as an algorithm (a circuit) that takes for input a key κ identifying a function h_κ in the family and tries to output a collision $x_0 \neq x_1$ such that $h_\kappa(x_0) = h_\kappa(x_1)$. (Such modeling is not possible for single functions since for any length-reducing function there always exists an adversary that outputs a collision for that function in constant time.) Second, this approach supports a simple and natural quantitative measure for the level of collision resistance: the level of collision resistance is the maximum probability, over the choice of the key, with which an efficient adversary can find a collision. Third, current constructions of hash functions can be naturally regarded as keyed function families. For instance, we may interpret the initialization vector (IV) in SHA0 and SHA1 as a key. Finally, several collision-finding attacks seems to depend on specific values or properties of the key in use and work for some keys but not others. Specific examples include Dobbertin's attack on MD5 [6], time-memory trade-off attacks, and attacks on Gibson's hash function [8]. In particular, it may well be possible that even "broken" functions still have a significant fraction of keys for which attacks are less successful. On the other hand, it may not be sufficient to simply view an IV as a key, because the IV may not be incorporated into the computation in a sufficiently strong way; see the discussion at the end of the introduction.

Parameters. We consider the following parameters for hash functions and hardness amplification. First and foremost is the level of collision resistance. The goal in hardness amplification is to reduce the maximum probability that an efficient adversary can find collisions from $1-\delta$ to ϵ, where ϵ and δ are typically $o(1)$. Another salient parameter is the output length. Other parameters include the key size, the number of applications of the underlying hash function, the the running time (or, complexity) of the adversaries considered and the "compression ratio" (i.e the ratio of input length to output length). By itself, the compression ratio is less interesting since we may apply a transformation due to Merkle and Damgård [17, 4] to increase the compression ratio arbitrarily; this increases the number of applications of the underlying function but maintains the same key size and output length. Our goal is to construct hash functions with a high level of collision resistance, while maintaining short outputs, short keys, and a good trade-off between compression ratio and number of operations.

Constructions. We analyze two construction for hardness amplification. The first is based on simple concatenation (possibly folklore) and the second uses error-correcting codes and was suggested by Knudsen and Preneel [14]. Then, we analyze two additional constructions for reducing the key size and the output length respectively.

Amplification via concatenation. The first construction is simple concatenation: we hash the input using several independently chosen functions and concatenate the hash values. Formally, given a family $\mathcal{H} = \{h_\kappa\}$ of hash functions, and a parameter q, define the family $\mathcal{H}' = \{h'_{\kappa_1,\ldots,\kappa_q}\}$ so that $h'_{\kappa_1,\ldots,\kappa_q}(x) = h_{\kappa_1}(x) \circ \ldots \circ h_{\kappa_q}(x)$, where $\kappa_1, \ldots, \kappa_q$ are independently chosen keys in the family $\mathcal{H}$. The analysis is essentially the same as that for classic hardness amplification for one-way functions [27]. The underlying intuition is that finding collisions in $h'_{\kappa_1,\ldots,\kappa_q}$ is hard as long as finding collisions in one of $h_{\kappa_1},\ldots,h_{\kappa_q}$ is hard. If the initial maximal probability of finding collisions is δ, the maximal probability of finding collisions in the new hash family is $(1-\delta)^{\Omega(q)} = \mathrm{e}^{-\Omega(\delta q)}$. This means that the improvement in the level of collision resistance is exponential in q whereas the output length is linear in q.

Amplification via codes. In the second construction, we first encode the input with an error-correcting code wherein the codeword has length q over some large alphabet. Next, we hash the encoded input using q independently chosen functions (one for each of the q symbols in the codeword) and concatenate the hash values as before. In order to find a collision for this construction, one has to find collisions in *many* of the q underlying hash functions (as opposed to *all* q functions as in the previous construction). This construction was previously analyzed in an idealized setting in [14].

The analysis relies on the idea that finding collisions in the new hash function is hard as long as finding collisions in *several* of the q functions is hard (as opposed to finding collision in just a single function). Indeed, if the initial maximal probability of finding collisions is δ, then we expect that it is hard to find collisions in δq functions. To exploit this, we use a code with minimum distance $(1-O(\delta))q$, and for such codes, we may achieve a rate of $\Omega(\delta)$. Consequently this construction allows us to hash an input that is longer by a factor of $\Theta(\delta q)$ (compared to the first construction) while still using only q invocations of hash functions from the given family. When compared to amplifying the domain size via the Merkle-Damgård transformation and then applying the first construction, the second construction offers a $\Theta(\delta q)$ factor improvement in the number of hashing operations. The price we pay for this improvement is that for the same δ, ϵ (i.e., for fixed levels of collision resistance in the underlying and target hash functions), the choice of q for the second construction is a constant multiplicative factor larger than that for the first construction.

We remark that this analysis yields also hardness amplification for one-way functions with a logarithmic factor improvement in the security reduction.

Reducing the key size. Next, we demonstrate how to modify both constructions so that the key size increases only by an *additive* logarithmic term (at the price of increasing the output length by a constant multiplicative factor). This is done by choosing the q keys via randomness-efficient sampling using expander graphs. The sampler we require for the concatenation construction is fairly standard (e.g. randomness-efficient samplers were exploited in a similar manner in [5]), whereas the coding-theoretic construction requires a modified analysis of a previous sampler [9].

Reducing the output length. Starting with a family $\mathcal{H}$ of hash functions with output length ℓ_{out} and parameter q, the first two constructions yield a family with output length $q\ell_{\mathsf{out}}$. We show that for any $\varDelta$, we may in fact reduce the output length to $q\cdot(\ell_{\mathsf{out}}-\varDelta)$. More generally, we show how to transform any family $\mathcal{H}$ with output length ℓ_{out} into one with output length $\ell_{\mathsf{out}}-\varDelta$ with a negligible loss in the level of collision resistance. However, the complexity of computing the function increases by a multiplicative factor of $2^{\varDelta}$, so the construction is only useful for logarithmic values of $\varDelta$.

Limitations. We point out some of the limitations of our constructions and try to justify them. A first limitation is that, given a guarantee on the resilience of $\mathcal{H}$ against adversaries of a given size, we can only guarantee resilience of the new hash family $\mathcal{H}'$ against adversaries of much smaller size. A similar limitation is shared by existing hardness amplification results for one-way functions. This may be expected, given that all our constructions, as well as all existing constructions for hardness amplification of one-way functions are "black box". Indeed, evidence that such limitation may be inherent in "black-box constructions" is given in [11, Chapter 2, Ex 16, p. 96]. In addition, our constructions increase both the complexity of the hashing and the output length. To explain the blow-up in these parameters, we provide lower bounds on the number of hashing operations and output length:

- We establish a matching lower bound (up to multiplicative constants) on the number of hashing operations used in our first two constructions. The bound holds for black-box constructions that do not use the input as keys for the underlying hash functions. In particular, the number of hashing operations must have an inverse dependency on δ, the initial maximal probability of finding collisions. The bound is derived from that for hardness amplification for one-way functions in [15].
- Assuming in addition some natural restrictions on the reduction used in the proof of security, we show that the output length of the new hash function is at least $\Omega(\frac{1}{\delta} \cdot \ell_{\mathsf{out}})$. Our constructions achieve output length $O(\frac{1}{\delta} \cdot \ell_{\mathsf{out}} \cdot \log \frac{1}{\epsilon})$.

While the guarantees provided by our constructions may be too weak to be of real practical significance, this is unfortunately the state of the art for general constructions. Providing better guarantees remains a fascinating open problem.

Combiners. Our results pertaining to the output length (namely the fourth construction and lower bounds thereof) build on the recent work on black-box combiners for collision resistance [2, 19, 12]. We briefly recall the notion and results and explain the connection to hardness amplification.

Black-box combiners for collision resistance. A black-box combiner for collision resistance is a procedure that given t functions $h^1, \ldots, h^t$ with output length ℓ_{out}, computes a single function $\tilde{h}$ with the following property: there is an efficient transformation that given a collision for $\tilde{h}$, outputs collision for each of $h^1, \ldots, h^t$. This guarantees that finding collisions on $\tilde{h}$ is hard as long as finding collisions on *one* of $h^1, \ldots, h^t$ is hard. Concatenating the outputs of $h^1, \ldots, h^t$ on the same input yields a combiner with output length $t \cdot \ell_{\mathsf{out}}$. Boneh, Boyen and Pietrzak [2, 19] showed that this trivial combiner is essentially optimal by giving a $t \cdot (\ell_{\mathsf{out}} - O(\log n))$ lower bound for *deterministic* black-box combiners.

Black-box combiners for collision resistance arise naturally in the context of our work. Indeed, our first hardness amplification construction may be viewed as choosing $\kappa_1, \ldots, \kappa_q$ at random and applying the trivial (deterministic) combiner to $h_{\kappa_1}, \ldots, h_{\kappa_q}$. In addition, since we deal with *families* of functions rather than with single functions, it makes sense in our model to consider also *randomized* combiners (still, for single functions). We can then incorporate any randomness used by the combiner in the key of the new hash family. Two natural questions arise here: Can we beat the [2, 19] bound by using randomized combiners? Alternatively, can the bound be improved by removing the additive logarithmic factor?

We answer both questions negatively. We first extend the lower bound of [2, 19] to derive a $t\cdot(\ell_{\mathsf{out}} - O(\log n)$ lower bound on the output length of randomized black-box combiners. Our lower bound for the output length for hardness amplification builds on this lower bound. We then construct a randomized black-box combiner with output length $t \cdot (\ell_{\mathsf{out}} - \Omega(\log n))$. This result is interesting in itself, since it is the first non-trivial combiner that beats concatenation. Furthermore, this combiner underlies our fourth construction mentioned above, which reduces the output length of hash functions. Putting these two results together, we deduce that the optimal randomized black-box combiner has output length $t\cdot(\ell_{\mathsf{out}} - \Theta(\log n))$.

Combiners for families of hash functions. So far, we've discussed the relationship between combiners for single functions and hardness amplification for function families. In addition, one may directly study combiners for*families* of functions: Given t families of hash functions with output length ℓ_{out}, construct a single family of hash functions that is collision-resistant as long as one of the t families is collision-resistant. We note that it is possible to construct a combiner having output length $t \cdot (\ell_{\mathsf{out}} - O(\log n))$ using our randomized black-box combiner. The concurrent work of Fischlin and Lehmann [7] studies a very similar problem, albeit in an idealized model that only admits generic attacks on the hash functions.

Extensions. Our positive results for hardness amplification of collision resistance may be extended to several other variants of collision resistance. Details of these extensions are deferred to the final version of the paper.

Resistance to correlations. As noted in previous work (e.g. [1]), collision resistance can be regarded as a special case of "resistance to finding correlations." That is, for a given k-ary relation R, say that a family of functions $\mathcal{H}$ is R-$\mathsf{resistant}$ if it is hard given a random $h \in \mathcal{H}$ to find $x_1, ...x_k$ such that $R(h(x_1), ..., h(x_k))$ holds. In this terminology, collision resistance is simply R_{eq}-resistance where $R_{\mathsf{eq}}(y_1, y_2)$ iff $y_1 = y_2$. Can R-resistance be amplified for other relations? Can collision resistance be derived from (or imply) R-resistance for other relations R? These are interesting questions.

As a small step in this direction, we consider amplification for the "near collision" relation R_{near}, where $R_{\mathsf{near}}(y_1, y_2)$ iff the Hamming distance between y_1 and y_2 is small (see e.g. [16, Sec 9.2.6]). We observe that by encoding the hash value with an error-correcting code, we may transform a standard collision-resistant hash family to a near-collision-resistant hash family. Conversely, given a near-collision-resistant hash family, one can construct a standard collision-resistant hash family with shorter output by "decoding" the hash value to the nearest codeword of a covering code. This yields an amplification theorem for resistance to near-collisions, as a corollary of our amplification theorems for collision resistance.

Target collision resistance. Our results extend also to the related notion of target collision resistance (namely, universal one-way hash functions [18]). Here we may use the same constructions as for collision resistance, except to replace the Merkle-Damgård domain expansion with that of Shoup [24], and the same analysis goes through. We stress that the extension should not be taken for granted, because techniques for collision resistance do not always extend readily to target collision resistance; domain expansion is a good example.

Discussion. We discuss some additional aspects of the analysis in this work. First, we address only collision resistance, which is one out of many desired properties of "cryptographic hash functions". In fact, we do not even address properties such as resistance to finding additional collisions, once a collision is found. Concentrating on plain collision resistance allows for clearer understanding. In fact, constructing hash functions achieving even this specific property seems to be challenging enough, as evidenced by the attacks on MD5 and SHA1.

Another point worth highlighting is that our analysis can be viewed as a demonstration of the benefits in having *families* of hash functions, where there is some assurance that finding collisions in one function in the family does not render other functions in the family completely insecure. This may suggest a methodology for *constructing* practical collision-resistant functions: Design such functions as keyed functions, where the key is intimately incorporated in the evaluation of the function. This might give some hope that finding collisions for one value of the key might not help much in finding collisions for other values of the key. Then, apply a generic amplification mechanism such as the ones studied here to guarantee strong collision resistance *even when a significant fraction of the keys result in weak functions.* We stress that, in order to be of value, the key has to be incorporated in the computation of the function in a strong way. This fact is exemplified (in the negative) by the MD/SHA line of functions: Although these functions are often modeled as families of functions that are keyed via the IV, the actual constructions do not incorporate the IV in the computation in a strong way. And, indeed, the very recent attacks against such functions (e.g. [26]) seem to work equally well for all values of the IV. Similarly suspect are related methods for creating a hash function family from a fixed hash function by treating a portion of the input as key.

Finally, we stress that even though we use asymptotic notation to make our results more readable, they actually provide concrete bounds on the parameters achieved. Moreover, we provide uniform reductions in all of our proofs of security, so even though the positive results are stated for nonuniform adversaries, it is easy to derive an analogue of those results for uniform adversaries.

Organization. We begin with by reviewing quantitative definitions of collision resistance for CRHFs in Section 2. We present all of our constructions for hardness amplification, key size reduction and output length reduction in Section 3, and our lower bounds in Section 4. Given that randomized black-box combiners are a recurring tool in this paper, we define them in Section 2 and present the construction in Section 3 and the matching lower bound in Section 4.

2 Preliminaries

2.1 Quantitative definitions of collision resistance

A family of hash functions is a collection of polynomial-time computable functions $\mathcal{H} = \{\mathcal{H}_n : \{0,1\}^{\ell_{\mathsf{key}}(n)} \times \{0,1\}^{\ell_{\mathsf{in}}(n)} \rightarrow \{0,1\}^{\ell_{\mathsf{out}}(n)}\}$, where n is the security parameter, satisfying $\ell_{\mathsf{out}}(n) < \ell_{\mathsf{in}}(n)$. We refer to $\ell_{\mathsf{in}}, \ell_{\mathsf{out}}, \ell_{\mathsf{key}}$ as the input length, output length and key size of the hash function. We use $h_\kappa : \{0,1\}^{\ell_{\mathsf{in}}(n)} \rightarrow \{0,1\}^{\ell_{\mathsf{out}}(n)}$ to denote the function $\mathcal{H}_n(\kappa, \cdot)$ associated with the key $\kappa \in \{0,1\}^{\ell_{\mathsf{key}}(n)}$. We call a pair (x_0, x_1) satisfying $x_0 \neq x_1$ and $h_\kappa(x_0) = h_\kappa(x_1)$ a collision for h_κ.

For any n, we say that $\mathcal{H}_n$ is an (s, ϵ)-CRHF (collision-resistant hash function) if for every nonuniform A of size s,

$$\Pr[\kappa \leftarrow \{0,1\}^{\ell_{\mathsf{key}}(n)}; A(\kappa) \text{ outputs a collision for } h_\kappa] < \epsilon$$

(The quantity ϵ is what we refer to in the introduction as the level of collision resistance.) For notational simplicity, we omit references to n whenever the context is clear (e.g. $\mathcal{H} : \{0,1\}^{\ell_{\mathsf{key}}} \times \{0,1\}^{\ell_{\mathsf{in}}} \rightarrow \{0,1\}^{\ell_{\mathsf{out}}}$).

We will also refer to asymptotic notions of CRHFs. As with one-way functions, we want to consider the entire class of nonuniform polynomial-time adversaries (although we do provide uniform reductions in our proofs of security). Formally, we say that $\mathcal{H}$ is a strong CRHF if for every polynomial $p(\cdot)$ and every sufficiently large n, $\mathcal{H}$ is a $(p(n), \frac{1}{p(n)})$-CRHF. Similarly, we say that $\mathcal{H}$ is a weak CRHF if there exists a constant c such that for every polynomial $p(\cdot)$ and every sufficiently large n, $\mathcal{H}$ is a $(p(n), 1 - \frac{1}{n^c})$-CRHF. Standard cryptographic applications of hash functions actually require strong CRHFs, so whenever the strength of the CRHF is not qualified, we will refer to strong CRHFs.

Public-coin vs. secret-coin hash functions. As noted in [13], a distinction needs to be made between public-coin and secret-coin hash functions. In a public-coin hash function, the key corresponds to a uniformly generated random string and the key generation algorithm computes the identity function. In a secret-coin hash function, the distribution of the key may be any samplable distribution. For simplicity and clarity, our definition and exposition refer to public-coin hash functions. It is easy to see that all of our constructions (Constructions 1, 2 and 4) apart from the reduction in key size using randomness-efficient sampling extend to secret-coin hash functions.

2.2 Black-box combiners for collision resistance

We generalize the notion of black-box combiners from [2, 19] so as allow randomized constructions.

Definition 1. *We say that (C, R) is a* randomized black-box (t', t)-combiner for collision resistance *if C, R are deterministic poly-time oracle TMs, and there exists some negligible function $\nu(\cdot)$ such that for all $h^1, \ldots, h^t : \{0,1\}^{\ell_{\mathsf{in}}} \rightarrow \{0,1\}^{\ell_{\mathsf{out}}}$:*

CONSTRUCTION. *For every r, $C^{h^1,\dots,h^t}(r,\cdot)$ computes a function $\tilde{h}_r : \{0,1\}^{\ell_{\mathsf{in}}'} \rightarrow \{0,1\}^{\ell_{\mathsf{out}}'}$, where $\ell_{\mathsf{in}}' > \ell_{\mathsf{out}}'$.*

REDUCTION. *With probability $1-\nu(n)$ over r: if $(\tilde{x}_0,\tilde{x}_1)$ is a collision for $\tilde{h}_r$, then $R^{h^1,\dots,h^t}(r,\tilde{x}_0,\tilde{x}_1)$ outputs t pairs $(x_0^1,x_1^1),\dots,(x_0^t,x_1^t)$ such that for at least $t-t'+1$ values $i \in \{1,\dots,t\}$, (x_0^i,x_1^i) is a collision for h^i.*

Intuitively, the guarantee is that if it is hard to find collisions on some t' of the functions $h^1,\dots,h^t$, then with overwhelming probability over r, it is hard to find collisions on $\tilde{h}_r$. Our definition generalizes that in [2, 19] in that we provide both C and R with additional "randomness" r, which is interpreted as a key. Specifically, in the previous definitions, C computes a single function, whereas in our definition C computes a family of functions $\{\tilde{h}_r\}$. In our construction, R is deterministic, whereas our lower bound (as with previous work) extends to randomized reductions R.

3 Constructions

The goal of hardness amplification is to deduce the existence of strong CRHFs from weak CRHFs. Fix a security parameter n. The parameters for the new CRHF $\mathcal{H}'$ will be different from those for the starting CRHF $\mathcal{H}$: we use $\ell_{\mathsf{in}}, \ell_{\mathsf{out}}, \ell_{\mathsf{key}}$ to denote the parameters for a $(s, 1-\delta)$-CRHF that we start with, and $\ell_{\mathsf{in}}', \ell_{\mathsf{out}}', \ell_{\mathsf{key}}'$ to denote the parameters for the (s',ϵ)-CRHF that we are about to construct. Typical values of the parameters are $\delta = \frac{1}{\mathrm{poly}(n)}$ and $\epsilon = \mathrm{neg}(n)$. As outlined in the introduction, we begin two basic constructions for hardness amplification (Sections 3.1 and 3.2) and then show how to reduce the key size (Section 3.3) and output length (Section 3.4). A summary of the parameters is given in Fig 1.

Domain expansion. We compensate the loss in compression ratio in our constructions by first applying Merkle-Damgård domain expansion [4, 17], noting that domain expansion for collision resistance preserves the hardness parameter.

Proposition 0 ([4,17]). Fix some security parameter n. Suppose there exists a (s,ϵ)-CRHF $\mathcal{H}_n$ from $\{0,1\}^{\ell_{\mathsf{key}}} \times \{0,1\}^{\ell_{\mathsf{in}}}$ to $\{0,1\}^{\ell_{\mathsf{out}}}$ computable in time T. Then, Construction 0 yields an (s',ϵ)-CRHF $\mathcal{H}_n'$ from $\{0,1\}^{\ell_{\mathsf{key}}'} \times \{0,1\}^{\ell_{\mathsf{in}}'}$ to $\{0,1\}^{\ell_{\mathsf{out}}'}$ with the following parameters:

- $\ell_{\mathsf{out}}' = \ell_{\mathsf{out}}$ and $\ell_{\mathsf{key}}' = \ell_{\mathsf{key}}$
- # hash calls = $\frac{\ell_{\mathsf{in}}' - \ell_{\mathsf{in}}}{\ell_{\mathsf{out}} - \ell_{\mathsf{in}}}$
- security reduction : $s' = s - \ell_{\mathsf{in}}' \cdot T$

3.1 Amplification via concatenation

We begin with a description and the analysis of the basic concatenation construction. The analysis we provide is very similar to that for hardness amplification for one-way functions via direct product [27, 11]. The presentations is somewhat simpler. We also make a small modification to the analysis that facilitates the analysis of the coding-theoretic construction, discussed in the next section.

Construction 1 (basic). *Pick $q = \lceil \frac{2}{\delta} \ln \frac{2}{\epsilon} \rceil$ independent keys $\kappa_1, \ldots, \kappa_q$. On input $x \in \{0,1\}^{\ell_{\mathsf{in}}}$, output $h_{\kappa_1}(x) \circ h_{\kappa_2}(x) \circ \cdots \circ h_{\kappa_q}(x)$*

In using the same input to the hash functions under all of the q keys $\kappa_1, \ldots, \kappa_q$, we ensure that a collision x_0, x_1 for the key $(\kappa_1, \ldots, \kappa_q)$ is also a collision for the underlying hash function on each of the keys $\kappa_1, \ldots, \kappa_q$.

Proposition 1 (Construction 1). *Fix some security parameter n. Suppose there exists a $(s, 1-\delta)$-CRHF $\mathcal{H}_n$ from $\{0,1\}^{\ell_{\mathsf{key}}} \times \{0,1\}^{\ell_{\mathsf{in}}}$ to $\{0,1\}^{\ell_{\mathsf{out}}}$. Then, Construction 1 yields an (s', ϵ)-CRHF $\mathcal{H}'_n$ from $\{0,1\}^{\ell_{\mathsf{key}}'} \times \{0,1\}^{\ell_{\mathsf{in}}'}$ to $\{0,1\}^{\ell_{\mathsf{out}}'}$ with the following parameters:*

- *$\ell_{\mathsf{in}}' = \ell_{\mathsf{in}}$ and $\ell_{\mathsf{out}}' = \Theta(\frac{\ell_{\mathsf{out}}}{\delta} \log \frac{1}{\epsilon})$ and $\ell_{\mathsf{key}} = \Theta(\frac{\ell_{\mathsf{key}}}{\delta} \log \frac{1}{\epsilon})$*
- *# hash calls = $\Theta(\frac{\ell_{\mathsf{in}}'}{\delta \ell_{\mathsf{in}}} \log \frac{1}{\epsilon})$*
- *security reduction : $s' = s \cdot \Theta(\frac{1}{\epsilon} \log \frac{1}{\epsilon} \log \frac{1}{\delta})^{-1}$*

Proof. Suppose A finds collisions on $\mathcal{H}'_n$ with probability at least ϵ, and consider the following algorithm A' for finding collisions on $\mathcal{H}_n$: on input κ,

1. chooses $\kappa_1, \ldots, \kappa_q$ at random, $i \in [q]$ at random, and sets $\kappa_i = \kappa$.
2. runs $A(\kappa_1, \ldots, \kappa_q)$ to obtain x_0, x_1, and outputs x_0, x_1.

To analyze the success probability for A', first fix any set S of keys κ of density $\frac{\delta}{2}$. Intuitively, S represents the set of keys for which it is hard for A' to find a collision.

$$\Pr_{\kappa_1, \ldots, \kappa_q}[A(\kappa_1, \ldots, \kappa_q) \text{ outputs a collision} \bigwedge \text{at least one of the } \kappa_j\text{'s lies in } S] \\ \geq \epsilon - (1 - \tfrac{\delta}{2})^q \geq \tfrac{\epsilon}{2}$$

Hence,

$$\Pr_{\kappa_1, \ldots, \kappa_q, i}[A(\kappa_1, \ldots, \kappa_q) \text{ outputs a collision} \bigwedge \kappa_i \in S] \geq \tfrac{\epsilon}{2q}$$

On the other hand,

$$\begin{aligned} &\Pr_{\kappa_1, \ldots, \kappa_q, i}[A(\kappa_1, \ldots, \kappa_q) \text{ outputs a collision} \bigwedge \kappa_i \in S] \\ &= \tfrac{\delta}{2} \cdot \Pr_{\kappa \in S} \Pr[A'(\kappa) \text{ outputs a collision for } h_\kappa] \\ &\leq \tfrac{\delta}{2} \cdot \max_{\kappa \in S} \Pr[A'(\kappa) \text{ outputs a collision for } h_\kappa] \end{aligned}$$

This implies that for any set S of density $\frac{\delta}{2}$,

$$\max_{\kappa \in S} \Pr[A'(\kappa) \text{ outputs a collision for } h_\kappa] \geq \tfrac{\epsilon}{\delta q}$$

Hence,

$$\Pr_{\kappa}\Big[\Pr[A'(\kappa) \text{ outputs a collision for } h_\kappa] \geq \tfrac{\epsilon}{\delta q}\Big] \geq 1 - \tfrac{\delta}{2}$$

By running A' a total of $\frac{\delta q}{\epsilon} \log \frac{1}{\delta} = O(\frac{1}{\epsilon} \log \frac{1}{\epsilon} \log \frac{1}{\delta})$ times, we find collisions on $\mathcal{H}_n$ for a $1 - \frac{\delta}{2}$ fraction of keys with probability $1 - \frac{\delta}{2}$. This means we find collisions on $\mathcal{H}_n$ for a random key with probability at least $1 - \delta$. □

3.2 Amplification via codes

Note how the basic construction loses an $O(q)$ factor in the compression ratio because we repeat the same input for each of the q keys. The following work-around was suggested in [14]. We first encode the input x using an error-correcting code C to obtain q symbols $C(x)_1, \dots, C(x)_q \in \{0,1\}^{\ell_{\mathsf{in}}}$, and then we hash each of the q blocks with independently chosen hash functions $h_{\kappa_1}, \dots, h_{\kappa_q}$ and output the concatenation. Note that the adversary may upon receiving the q keys only produce collisions wherein the codewords disagree only on the "easy" keys. For the analysis to go through, we argue that w.h.p., a $\frac{\delta}{4}$ fraction of the keys (and not just one key) must be "hard". If we pick C to be a code with relative distance $1 - \frac{\delta}{8}$, we are guaranteed there is a $\frac{\delta}{8}$ fraction of positions wherein the codewords disagree and the corresponding keys are "hard".

Construction 2 (coding-theoretic). *Pick $q = \lceil \frac{16}{\delta} \ln \frac{2}{\epsilon} \rceil$ independent keys $\kappa_1, \dots, \kappa_q$. Let $C : \{0,1\}^{\ell_{\mathsf{in}}'} \to (\{0,1\}^{\ell_{\mathsf{in}}})^q$ be an error-correcting code with minimum relative distance $1 - \frac{\delta}{8}$ (e.g., the Reed-Solomon code), where $\ell_{\mathsf{in}}' = \Theta(\delta q \ell_{\mathsf{in}})$. On input $x \in \{0,1\}^{\ell_{\mathsf{in}}'}$, output $h_{\kappa_1}(C(x)_1) \circ h_{\kappa_2}(C(x)_2) \circ \cdots \circ h_{\kappa_q}(C(x)_q)$.*

Proposition 2 (Construction 2). *Fix some security parameter n. Suppose there exists a $(s, 1-\delta)$-CRHF $\mathcal{H}_n$ from $\{0,1\}^{\ell_{\mathsf{key}}} \times \{0,1\}^{\ell_{\mathsf{in}}}$ to $\{0,1\}^{\ell_{\mathsf{out}}}$. Then, Construction 2 yields an (s', ϵ)-CRHF $\mathcal{H}_n'$ from $\{0,1\}^{\ell_{\mathsf{key}}'} \times \{0,1\}^{\ell_{\mathsf{in}}'}$ to $\{0,1\}^{\ell_{\mathsf{out}}'}$ with the following parameters:*

- *$\ell_{\mathsf{in}}' = \Theta(\ell_{\mathsf{in}} \log \frac{1}{\epsilon})$ and $\ell_{\mathsf{out}}' = \Theta(\frac{\ell_{\mathsf{out}}}{\delta} \log \frac{1}{\epsilon})$ and $\ell_{\mathsf{key}}' = \Theta(\frac{\ell_{\mathsf{key}}}{\delta} \log \frac{1}{\epsilon})$*
- *# hash calls = $\Theta(\frac{1}{\delta} \log \frac{1}{\epsilon})$*
- *security reduction : $s' = s \cdot \Theta(\frac{1}{\epsilon} \log \frac{1}{\delta})^{-1}$*

Proof. Suppose A finds collisions on $\mathcal{H}_n'$ with probability at least ϵ, and consider the following algorithm A' for finding collisions on $\mathcal{H}_n$: on input κ,

1. chooses $\kappa_1, \dots, \kappa_q$ at random, $i \in [q]$ at random, and sets $\kappa_i = \kappa$.
2. runs $A(\kappa_1, \dots, \kappa_q)$ to obtain x_0, x_1, and outputs $C(x_0)_i, C(x_1)_i$.

To analyze the success probability for A', first fix any set S of keys κ of density $\frac{\delta}{2}$. By a Chernoff bound (the multiplicative variant), we have

$$\Pr_{\kappa_1,\dots,\kappa_q}[A(\kappa_1,\dots,\kappa_q) \text{ outputs a collision } (x_0,x_1) \bigwedge \text{at least } \tfrac{\delta}{4} \text{ fraction of } \kappa_j\text{'s lies in } S]$$
$$\geq \epsilon - \mathrm{e}^{-\delta q/16} \geq \frac{\epsilon}{2}$$

Conditioned on the above event, for a $\frac{\delta}{8}$ fraction of j's in $\{1,2,\dots,q\}$, we have $C(x_0)_j \neq C(x_1)_j$ and $\kappa_j \in S$ (since the former occurs for a $1 - \frac{\delta}{8}$ fraction of j's and the latter occurs for a $\frac{\delta}{4}$ fraction of j's). Hence,

$$\Pr_{\kappa_1,\dots,\kappa_q,i}[A(\kappa_1,\dots,\kappa_q) \text{ outputs a collision } (x_0,x_1) \wedge \kappa_i \in S \wedge C(x_0)_i \neq C(x_1)_i] \geq \frac{\delta\epsilon}{16}$$

On the other hand,

$$\begin{aligned}&\Pr_{\kappa_1,\ldots,\kappa_q,i}[A(\kappa_1,\ldots,\kappa_q) \text{ outputs a collision } (x_0,x_1) \wedge \kappa_i \in S \wedge C(x_0)_i \neq C(x_1)_i] \\ &= \tfrac{\delta}{2} \cdot \Pr_{\kappa \in S} \Pr[A'(\kappa) \text{ outputs a collision for } h_\kappa] \\ &\leq \tfrac{\delta}{2} \cdot \max_{\kappa \in S} \Pr[A'(\kappa) \text{ outputs a collision for } h_\kappa]\end{aligned}$$

This implies that for any set S of density $\frac{\delta}{2}$,

$$\max_{\kappa \in S} \Pr[A'(\kappa) \text{ outputs a collision for } h_\kappa] \geq \tfrac{\epsilon}{8}$$

Hence,

$$\Pr_{\kappa}\Big[\Pr[A'(\kappa) \text{ outputs a collision for } h_\kappa] \geq \tfrac{\epsilon}{8}\Big] \geq 1 - \tfrac{\delta}{2}$$

Again by running A' a total of $O(\frac{1}{\epsilon}\log\frac{1}{\delta})$ times, we can find collisions on $\mathcal{H}_n$ with probability $1-\delta$. □

3.3 Reducing the key size

From a theoretical point of view, it is useful to have hash functions with short descriptions (i.e. short keys). Short keys may also be of interest from a practical point of view, although for the most common application of collision-resistant hash functions (digital signatures) the key would be standardized and only distributed once. Starting with a 160-bit key, the above transformations could yield a key that is much longer. Fortunately, there is no inherent cause for this blow-up: we may reduce the key size in each of the above constructions using randomness-efficient sampling [9], namely, we want to sample q keys in $\{0,1\}^{\ell_{\mathsf{key}}}$ using r bits of randomness, where $r \ll q\ell_{\mathsf{key}}$.

To accomplish this, we will use the randomness-efficient hitter in [9, Appendix C], with a slightly different analysis showing that for the parameters we are interested in, the construction satisfies a stronger sampler-like property. The weaker hitter guarantee is sufficient to reduce the key size for Construction 1, whereas the stronger sampler-like property is necessary for Construction 2. For our application, we will also require that that the hitter satisfy a certain reconstructibility property, previously used in [5]. This is used in the security reduction to generate challenges for the adversary breaking $\mathcal{H}'$ given a key for $\mathcal{H}$.

We stress here that for specific concrete parameters, we may use different choices of hitters and samplers for ease of implementation and optimality for those specific parameters.

Lemma 1. *There exists a constant c such that for every $\delta, \epsilon > 0$, there is an efficient randomized procedure $G : \{0,1\}^r \to (\{0,1\}^{\ell_{\mathsf{key}}})^q$ with the following properties:*

— *(sampler) for every subset $S \subseteq \{0,1\}^{\ell_{\mathsf{key}}}$ of density δ, with probability at least $1-\epsilon$, at least $\frac{\delta q}{2c}$ of the strings output by G lie in S.*

— *(complexity) the randomness complexity r is $\ell_{\mathsf{key}} + O(\log\frac{1}{\epsilon})$ and the sample complexity q is $O(\frac{1}{\delta}\log\frac{1}{\epsilon})$.*

— *(reconstructible) there exists an efficient algorithm that on input (i, x), outputs a uniformly random element from the set $\{\sigma \mid G(\sigma)_i = x\}$.*

Proof (sketch). The construction (based on that in [9]) proceeds in three stages:

- First, we construct a hitter that generates $\frac{c}{\delta}$ samples in $\{0,1\}^{\ell_{\mathsf{key}}}$ using ℓ_{key} random bits with the following property: for every subset S of $\{0,1\}^{\ell_{\mathsf{key}}}$ with density δ, with probability at least $\frac{2}{3}$, at least one sample lies in S. We may obtain such a hitter using Ramanujan graphs of degree $\frac{c}{\delta}$ and vertex set $\{0,1\}^{\ell_{\mathsf{key}}}$, wherein we pick a random vertex v, and the samples are the indices of the neighbors of v [9].
- Next, we construct a sampler that generates $d = O(\log \frac{1}{\epsilon})$ samples in $\{0,1\}^{\ell_{\mathsf{key}}}$ using $\ell_{\mathsf{key}} + O(d)$ random bits with the following property: for every subset S' of $\{0,1\}^{\ell_{\mathsf{key}}}$ with density $\frac{2}{3}$, with probability at least $1 - \epsilon$, at least $\frac{1}{2}$ of the samples lie in S'. We may obtain such a sampler by taking a random walk of length $d - 1$ on a constant-degree expander with vertex set $\{0,1\}^{\ell_{\mathsf{key}}}$ [9].
- Finally, we compose the sampler and the hitter as follows: we consider a random walk of length $d-1$ on the expander, and use each of the d vertices along the path as random coins for the hitter. Overall, we will run the hitter d times, which generate a total of $q = d \cdot \frac{c}{\delta}$ samples using a total of $\ell_{\mathsf{key}} + O(d)$ random bits. This yields the desired query and randomness complexity.

The sampler guarantee follows fairly readily. Fix S of density δ. Let S' be the set of random coins for the hitter such that at least one sample lies in S, so S' has density at least $\frac{2}{3}$. We know that with probability at least $1 - \epsilon$ (over the random walk), we generate at least $\frac{d}{2}$ samples in S', which in turn yields $\frac{d}{2} = \frac{\delta q}{2c}$ samples that lie in S.

Finally, we check each of the two components in our construction is reconstructible, from which it follows that the combined construction is also reconstructible. For the expander-based hitter, this means that given i, x, we need to compute the vertex v whose i'th neighbor is labeled x. For the expander-based sampler, we need to given i, x, sample a start vertex and a path such that the i'th vertex on the path is labeled x. Indeed, both properties are readily satisfied for standard explicit constructions of constant-degree expanders. □

The next construction is obtained from Construction 2 by replacing independent sampling of the q keys with randomness-efficient sampling using G, and using a code with slightly different parameters:

Construction 3 (reduced key size). *Run G to obtain q keys $\kappa_1, \ldots, \kappa_q \in \{0,1\}^{\ell_{\mathsf{key}}}$. Let $C : \{0,1\}^{\ell_{\mathsf{in}}'} \to (\{0,1\}^{\ell_{\mathsf{in}}})^q$ be an error-correcting code with minimum relative distance $1 - \frac{\delta}{4c}$ (e.g., the Reed-Solomon code), where $\ell_{\mathsf{in}}' = \Theta(\delta q \ell_{\mathsf{in}})$. On input $x \in \{0,1\}^{\ell_{\mathsf{in}}'}$, output $h_{\kappa_1}(C(x)_1) \circ h_{\kappa_2}(C(x)_2) \circ \cdots \circ h_{\kappa_q}(C(x)_q)$.*

It is straight-forward to verify that an analogue of Proposition 2 holds for Construction 3 if the CRHF is public-coin, and with essentially the same parameters except that the key size is now reduced to $\ell_{\mathsf{key}} + O(\log \frac{1}{\epsilon})$ (i.e., the randomness complexity of G). We now state our main result for hardness amplification of collision-resistance, which is essentially a restatement of Proposition 2 for independent sampling and for randomness-efficient sampling:

Parameters	Construction 0	Construction 1	Construction 2	Construction 4
input length	ℓ_{in}'	ℓ_{in}	$\Theta(\ell_{\mathsf{in}} \log \frac{1}{\epsilon})$	$\ell_{\mathsf{in}} - \Delta - \log \ell_{\mathsf{in}}$
output length	ℓ_{out}	$\Theta(\frac{\ell_{\mathsf{out}}}{\delta} \log \frac{1}{\epsilon})$	$\Theta(\frac{\ell_{\mathsf{out}}}{\delta} \log \frac{1}{\epsilon})$	$\ell_{\mathsf{out}} - \Delta$
# hash calls	$\frac{\ell_{\mathsf{in}}' - \ell_{\mathsf{in}}}{\ell_{\mathsf{out}} - \ell_{\mathsf{in}}}$	$\Theta(\frac{1}{\delta} \log \frac{1}{\epsilon})$	$\Theta(\frac{1}{\delta} \log \frac{1}{\epsilon})$	$\Theta(2^{\Delta} \ell_{\mathsf{in}})$
key size	ℓ_{key}	$\Theta(\frac{\ell_{\mathsf{key}}}{\delta} \log \frac{1}{\epsilon})$	$\Theta(\frac{\ell_{\mathsf{key}}}{\delta} \log \frac{1}{\epsilon})$	$\Theta({\ell_{\mathsf{in}}}^2 + \Delta)$
(public-coin)	ℓ_{key}	$\ell_{\mathsf{key}} + \Theta(\log \frac{1}{\epsilon})$	$\ell_{\mathsf{key}} + \Theta(\log \frac{1}{\epsilon})$	$\Theta({\ell_{\mathsf{in}}}^2 + \Delta)$

Fig. 1. Summary of parameters for Constructions 0, 1, 2, & 4. In order to compare constructions 1 and 2 on inputs of the same length, we could apply the Merkle-Damgård transformation first, in which case the latter offers a $\Theta(\log \frac{1}{\epsilon})$ factor improvement in the number of hashing operations. For the key size, the second line refers that achieved using Construction 3 for public-coin hash functions.

Theorem 1. *Fix some security parameter n. Suppose there exists a $(s, 1-\delta)$-CRHF $\mathcal{H}_n$ from $\{0,1\}^{\ell_{\mathsf{key}}} \times \{0,1\}^{\ell_{\mathsf{in}}}$ to $\{0,1\}^{\ell_{\mathsf{out}}}$. Then, there exists an (s', ϵ)-CRHF $\mathcal{H}_n'$ from $\{0,1\}^{\ell_{\mathsf{key}}'} \times \{0,1\}^{\ell_{\mathsf{in}}'}$ to $\{0,1\}^{\ell_{\mathsf{out}}'}$ with the following parameters:*

- *${\ell_{\mathsf{in}}}' = \Theta(\ell_{\mathsf{in}} \log \frac{1}{\epsilon})$ and ${\ell_{\mathsf{out}}}' = \Theta(\frac{\ell_{\mathsf{out}}}{\delta} \log \frac{1}{\epsilon})$ and ${\ell_{\mathsf{key}}}' = \Theta(\frac{\ell_{\mathsf{key}}}{\delta} \log \frac{1}{\epsilon})$*
- *# hash calls = $\Theta(\frac{1}{\delta} \log \frac{1}{\epsilon})$*
- *security reduction : $s' = s \cdot \Theta(\frac{1}{\epsilon} \log \frac{1}{\delta})^{-1}$*

Moreover, if the CRHF is public-coin, then we may reduce ${\ell_{\mathsf{key}}}'$ to $\ell_{\mathsf{key}} + \Theta(\log \frac{1}{\epsilon})$.

3.4 Reducing the output length

We show that it is possible to reduce the output size of any CRHF by an additive factor of Δ, with a negligible loss in the the probability of finding collisions, but at the price of an exponential (in Δ) multiplicative increase in the complexity of the function, along with a similar decrease in the size of adversaries tolerated. This imposes a limitation of $\Delta = O(\log n)$ for all reasonable settings.

Proposition 3. *Suppose there exists a (s, ϵ)-CRHF $\mathcal{H}$ from $\{0,1\}^{\ell_{\mathsf{in}}}$ to $\{0,1\}^{\ell_{\mathsf{out}}}$. Let $\Delta = O(\log n)$. Then, there exists a $(s - \mathrm{poly}(2^{\Delta}, n), \epsilon + 2^{-\Omega(\ell_{\mathsf{in}})})$-CRHF from $\{0,1\}^{\ell_{\mathsf{in}} - \Delta - \log \ell_{\mathsf{in}} - 2}$ to $\{0,1\}^{\ell_{\mathsf{out}} - \Delta}$. The complexity of the new CRHF is increases by a factor $\mathrm{poly}(2^{\Delta}, \ell_{\mathsf{in}})$.*

This result follows the randomized black-box combiner in the following theorem, setting $t' = t = 1$.

Theorem 2. *There is a randomized black-box (t', t)-combiner (C, R) achieving parameters ${\ell_{\mathsf{in}}}' = \ell_{\mathsf{in}} - \Delta - \log \ell_{\mathsf{in}} - 2$ and ${\ell_{\mathsf{out}}}' = (t - t' + 1) \cdot (\ell_{\mathsf{out}} - \Delta)$ for any positive Δ such that ${\ell_{\mathsf{in}}}' > {\ell_{\mathsf{out}}}' > 0$. The running times of C and R are polynomial in n and 2^{Δ} and the randomness complexity of C is $O({\ell_{\mathsf{in}}}^2 + \Delta)$.*

We may in fact use this combiner instead of the trivial combiner for our hardness amplification constructions. However, since we do not optimize on the output length of our hardness amplification within constant multiplicative factors, it does not make sense to try to cut down on the additive terms.

Overview of combiner. We begin with the case $t' = t = 1$ and suppose $h = h^1$ is "highly regular", and we have a partition of $\{0,1\}^{\ell_{\mathsf{in}}}$ into $2^{\ell_{\mathsf{in}}-\Delta}$ sets $\{S_{\tilde{x}} \mid y \in \{0,1\}^{\ell_{\mathsf{in}}-\Delta}\}$ each of size 2^t with the following property: for every $\tilde{x}$, $S_{\tilde{x}}$ contains a unique string x such that $h(x)$ has prefix 0^{Δ}. Then, we define $\tilde{h}(\tilde{x})$ to be the $(\ell_{\mathsf{out}} - \Delta)$-bit suffix of $h(x)$. It is easy to see how every collision $(\tilde{x}, \tilde{x}')$ for $\tilde{h}$ yields a collision (x_0, x_1) for h. To arrive at the general construction (which is where randomness plays a role),

- We replace 0^{Δ} with a string $z \in \{0,1\}^{\Delta}$ that is relatively popular in the sense that it occurs in at least an $\Omega(1/2^{\Delta})$ fraction of the images of h. Such a z can be identified by evaluating h on $O(\ell_{\mathsf{in}} \cdot 2^{2\Delta})$ random inputs. To bring the randomness complexity down to $O(\ell_{\mathsf{in}} + \Delta)$, we choose these inputs using the randomness-efficient Boolean sampler for approximating the mean within an additive error of $\frac{1}{2} \cdot 2^{-\Delta}$ with probability $1 - 2^{-2\ell_{\mathsf{in}}}$ in [9].
- We replace the fixed partitioning with a random partitioning induced by a family $\mathcal{G}$ of ℓ_{in}-wise independent functions from $\{0,1\}^{\ell_{\mathsf{in}}}$ to $\{0,1\}^{\ell_{\mathsf{in}}-\Delta-\log \ell_{\mathsf{in}}-2}$. Given $g \in \mathcal{G}$, we take $S_{\tilde{x}} = g^{-1}(\tilde{x})$. This gives us a partition of $\{0,1\}^{\ell_{\mathsf{in}}}$ into sets each of size $\tilde{O}(2^{\Delta}\ell_{\mathsf{in}})$. With overwhelming probability over g, for every $\tilde{x}$, there exists $x \in S_{\tilde{x}}$ such that $h(x)$ has prefix z (we set x to be the lexicographically first string with this property).

Construction and analysis. We formally state the construction for $t' = t = 1$. For simplicity, we present the construction using independent samples u_i and defer the randomness-efficient version to the full version.

Construction 4. *Let $\mathcal{G} = \{g : \{0,1\}^{\ell_{\mathsf{in}}} \to \{0,1\}^{\ell_{\mathsf{in}}-\Delta-\log \ell_{\mathsf{in}}-2}\}$ be a family of $6\ell_{\mathsf{in}}$-wise independent hash functions that such that given y, the set $g^{-1}(y)$ is computable in time $\mathrm{poly}(2^{\Delta}, n)$. (This can be achieved using univariate polynomials of degree $6\ell_{\mathsf{in}}$). On input $\tilde{x} \in \{0,1\}^{\ell_{\mathsf{in}}-\Delta-\log \ell_{\mathsf{in}}-2}$ and randomness $r \in \{0,1\}^{O(\Delta+\ell_{\mathsf{in}}{}^2)}$, we compute $\tilde{h}_r(\tilde{x}) \in \{0,1\}^{\ell_{\mathsf{out}}-\Delta}$ as follows:*

1. *Parse r as $g \in \mathcal{G}$ and $u_1, \ldots, u_m \in \{0,1\}^{\ell_{\mathsf{in}}}$, where $m = \Theta(2^{2\Delta}\ell_{\mathsf{in}})$.*
2. *Let $z \in \{0,1\}^{\Delta}$ be the lexicographically first string that occurs at least a $1/2^{\Delta}$ fraction of times as a prefix among $h(u_1), \ldots, h(u_m)$ (where $h = h^1$);*
3. *Compute $S_{\tilde{x}} = g^{-1}(\tilde{x})$ in order to find a string x in $S_{\tilde{x}}$ such that $h(x)$ has prefix z. Choose the lexicographically first string if there are more than 1; output $0^{\ell_{\mathsf{out}}-\Delta}$ if no such string exists or if $|S_{\tilde{x}}| > 8\ell_{\mathsf{in}} \cdot 2^{\Delta}$.*
4. *Output the $(\ell_{\mathsf{out}} - \Delta)$-bit suffix of $h(x)$.*

For general t', t, we may simply apply the above construction to each of $h^1, \ldots, h^{t-t'+1}$ and concatenate the output; it will be clear from the analysis that we may use the same randomness r for all t functions. Theorem 2 follows readily once we establish the following technical claim for $t' = t = 1$.

Claim. With probability $1 - 2^{-\Omega(\ell_{\mathsf{in}})}$ over $r = (g, u_1, \ldots, u_m)$, the following statements hold simultaneously:

- $|\Gamma_z| \geq 2^{\ell_{\mathsf{in}} - \Delta - 1}$, where z is as in the construction and $\Gamma_z = \{x \in \{0,1\}^{\ell_{\mathsf{in}}} \mid h(x)$ has prefix $z\}$;
- for all $\tilde{x}$, we have $S_{\tilde{x}} \cap \Gamma_z \neq \emptyset$ (where $S_{\tilde{x}} = g^{-1}(\tilde{x})$);
- for all $\tilde{x}$, we have $|S_{\tilde{x}}| \leq 8\ell_{\mathsf{in}} \cdot 2^{\Delta}$.

Suppose we have a collision $(\tilde{x}_0, \tilde{x}_1)$ for $\tilde{h}_r$, where the conditions in the technical claim do hold for r. Then, we could in $\mathrm{poly}(2^{\Delta}, \ell_{\mathsf{in}})$ time compute $(x_0, x_1) \in S_{\tilde{x}_0} \times S_{\tilde{x}_1}$ such that $h(x_0) = z \circ h_r(\tilde{x}_0)$ and $h(x_1) = z \circ h_r(\tilde{x}_1)$. This implies (x_0, x_1) is a collision for h.

Proof (of claim). By a Chernoff bound, we have that for each Δ-bit prefix w, if w occurs in a p_w fraction of outputs of h as a prefix, then with probability at least $1 - 2^{-2\ell_{\mathsf{in}}}$ over the u_i's, w will occur at most a $p_w + \frac{1}{2} \cdot 2^{-\Delta}$ fraction of times (as a prefix) among the $h(u_i)$'s. Taking a union bound over all $2^{\Delta} < 2^{\ell_{\mathsf{in}}}$ prefixes, we see that with probability at least $1 - 2^{-\ell_{\mathsf{in}}}$, the prefix z must satisfy $p_z \geq \frac{1}{2} \cdot 2^{-\Delta}$ and thus $|\Gamma_z| \geq 2^{\ell_{\mathsf{in}} - \Delta - 1}$. We assume in the rest of the proof that this is the case. Then, for each $y \in \{0,1\}^{\ell_{\mathsf{in}} - \Delta - \log \ell_{\mathsf{in}} - 2}$: $\mathrm{E}[|S_{\tilde{x}} \cup \Gamma_z|] = |\Gamma_z| \cdot 2^{-\ell_{\mathsf{in}} + t + \log \ell_{\mathsf{in}} + 2} \geq 2\ell_{\mathsf{in}}$. Applying a tail bound for $6\ell_{\mathsf{in}}$-wise independence [22], we obtain:

$$\Pr_g[S_{\tilde{x}} \cap \Gamma_z = \emptyset] \leq 2^{-2\ell_{\mathsf{in}}}$$

Taking a union bound over all $y \in \{0,1\}^{\ell_{\mathsf{in}} - \Delta - \log \ell_{\mathsf{in}} - 2}$, we have:

$$\Pr_g[\exists y : S_{\tilde{x}} \cap \Gamma_z = \emptyset] \leq 2^{-2\ell_{\mathsf{in}}} \cdot 2^{\ell_{\mathsf{in}} - \Delta - \log \ell_{\mathsf{in}} - 2} = 2^{-\Omega(\ell_{\mathsf{in}})}$$

Finally, for each y, $\mathrm{E}[|S_{\tilde{x}}|] = 4\ell_{\mathsf{in}} \cdot 2^{\Delta}$. Again, by using the tail bound for $6\ell_{\mathsf{in}}$-wise independence and a union bound, we have $\Pr[\exists y : |S_{\tilde{x}}| > 8\ell_{\mathsf{in}} \cdot 2^{\Delta}] < 2^{-\Omega(\ell_{\mathsf{in}})}$. □

4 Limitations

We begin by presenting the class of constructions for which we prove lower bounds:

Definition 2. *We say that (C, R) is a* black-box $(1 - \delta, \epsilon)$-amplifier for collision resistance *if $C = (C_{\mathsf{key}}, C_{\mathsf{hash}})$ is a pair of deterministic (oracle) TMs, and $R = (R_{\mathsf{key}}, R_{\mathsf{coll}})$ is a pair of randomized (oracle) TMs, and both pairs of TMs run in time* $\mathrm{poly}(n, \frac{1}{\delta}, \frac{1}{\epsilon})$. *In addition, for all $\mathcal{H} = \{\{0,1\}^{\ell_{\mathsf{key}}} \times \{0,1\}^{\ell_{\mathsf{in}}} \to \{0,1\}^{\ell_{\mathsf{out}}}\}$:*

> CONSTRUCTION. *C compute $\mathcal{H}' = \{\{0,1\}^{\ell_{\mathsf{key}}'} \times \{0,1\}^{\ell_{\mathsf{in}}'} \to \{0,1\}^{\ell_{\mathsf{out}}'}\}$ where $\ell_{\mathsf{out}}' > \ell_{\mathsf{in}}'$ as follows: given a key κ' and a string x, we run $C_{\mathsf{key}}(\kappa')$ to obtain $\kappa_1, \ldots, \kappa_q$ and then set $h'_{\kappa'}(x)$ to be $C_{\mathsf{hash}}^{h_{\kappa_1}, \ldots, h_{\kappa_q}}(\kappa', x)$.*
>
> REDUCTION. *There exists a constant c such that for every TM A that outputs a collision on $h'_{\kappa'}$ with probability at least ϵ and any subset S of $\{0,1\}^{\ell_{\mathsf{key}}}$ of density at least $\delta/2$, there exists $\kappa \in S$ such that*
>
> $$\Pr_{\sigma, R_{\mathsf{coll}}}\left[R_{\mathsf{key}}(\kappa; \sigma) = \kappa'; R_{\mathsf{coll}}^{\mathcal{H}}(i, \sigma, A(\kappa')) \text{ outputs a collision on } h_\kappa\right] > \left(\frac{\delta\epsilon}{n}\right)^c$$

Note that a black-box amplifier should provide an efficient reduction that converts any adversary A that finds collisions in $h'_{\kappa'}$ with probability ϵ into an adversary A' that finds collisions in h_κ with probability $1-\delta$. Indeed, Definition 2 guarantees that for a $1-\frac{\delta}{2}$ fraction of keys κ, $R^{A,\mathcal{H}}(\kappa)$ outputs a collision for h_κ with probability $(\frac{\delta\epsilon}{n})^c$. Running R a total of $O((\frac{n}{\delta\epsilon})^c \log\frac{1}{\delta})$ yields the desired reduction. The above reduction is more restrictive than an arbitrary black-box reduction due to the following structural restrictions we place on the construction and the reduction, and this makes our result weaker.

Construction. We do not allow constructions that use the input as a key into the underlying family hash functions. We enforce this constraint by having a key generation algorithm C_{key} select the members $h_{\kappa_1},\ldots,h_{\kappa_q}$ of the underlying family given only the new key κ', and restrict the actual computation C_{hash} to only query $h_{\kappa_1},\ldots,h_{\kappa_q}$. We will refer to q as the query complexity of the construction, the idea being that C_{hash} will query each of the functions $h_{\kappa_1},\ldots,h_{\kappa_q}$ at least once by having C_{key} not generate extraneous keys.

Reduction. The restriction on the reduction states that the reduction only requires a single collision from A' to break $\mathcal{H}$ with noticeable probability. This is true of the reductions used in our constructions and of all known reductions used in hardness amplification for one-way functions (c.f. [15]): all these reductions generate multiple challenges to the adversary and if the adversary successfully answers any of the challenges, the reduction succeeds with high probability.

We present lower bounds for the query complexity of the construction q and the output length ${\ell_{\mathsf{out}}}'$.

Theorem 3. *Suppose (C,R) is a black-box $(1-\delta,\epsilon)$-amplifier for collision resistance with $\epsilon \le \frac{\delta}{2}$. Then,*

$$q \ge \Omega(\tfrac{1}{\delta}\log\tfrac{1}{\epsilon}) \quad \textit{and} \quad {\ell_{\mathsf{out}}}' \ge \tfrac{1}{\delta}\cdot\big(\ell_{\mathsf{out}} - O(\log n + \log\tfrac{1}{\epsilon} + \log\tfrac{1}{\delta})\big) - 2$$

The lower bound for q follows closely the lower bound in [15], by arguing that C_{key} must compute a randomness-efficient hitting sampler, and is omitted due to lack of space. To obtain a lower bound for ${\ell_{\mathsf{out}}}'$, we begin with an observation of a connection between black-box hardness amplification and randomized black-box combiners. Intuitively, a $(1-\delta)$-CRHF could comprise $\lfloor\frac{1}{\delta}\rfloor$ functions, of which it is hard to find collisions on just one of them. In this case, the black-box $(1-\delta,\epsilon)$-amplifier acts like a randomized black-box $(1,\lfloor\frac{1}{\delta}\rfloor)$-combiner. To derive a lower bound for the latter, we use the probabilistic argument in Pietrzak's work [19]. We also note that the probabilistic argument is already sufficient to obtain the lower bounds for deterministic black-box combiners, therefore simplifying the lower bounds in [2, 19] by eliminating an additional randomization argument therein.

Proof. Set t to be a power of 2 in the interval $[\frac{1}{\delta},\frac{2}{\delta})$. Pick t random functions $f_1,\ldots,f_t : \{0,1\}^{\ell_{\mathsf{in}}} \to \{0,1\}^{\ell_{\mathsf{out}}}$ and identify $\{0,1\}^{\ell_{\mathsf{key}}}$ with $\{1,2,\ldots,t\}$ and $\mathcal{H}$ with $\{f_1,\ldots,f_t\}$. Consider the following procedure $\tilde{R}$ for finding collisions in $f_1,\ldots,f_t$ given oracle access to these functions:

— picks $x'_0, x'_1 \in \{0,1\}^{\ell_{\mathsf{in}}'}$ and $\kappa' \in \{0,1\}^{\ell_{\mathsf{key}}'}$ at random;

— for each $i = 1, 2, \ldots, t$, sample a random σ_i such that $R_{\mathsf{key}}(i; \sigma_i) = \kappa'$, and output $R_{\mathsf{coll}}^{f_1,\ldots,f_t}(i, \sigma_i, (x'_0, x'_1))$.

We note that for all $f_1, \ldots, f_t$ and for all κ', the function $h'_{\kappa'}$ maps $\{0,1\}^{\ell_{\mathsf{in}}'}$ to $\{0,1\}^{\ell_{\mathsf{out}}'}$. By the standard lower bound on collision probability or a simple application of Cauchy-Schwartz, we have

$$\Pr_{x'_0,x'_1}[(x'_0, x'_1) \text{ is a collision for } h'_{\kappa'}] \geq 2^{-\ell_{\mathsf{out}}'} - 2^{-\ell_{\mathsf{in}}'} \geq 2^{-\ell_{\mathsf{out}}'-1}$$

Consider a procedure A that outputs collisions on every $h'_{\kappa'}$ by repeatedly choosing (x'_0, x'_1) at random until it finds a collision. By our choice of t, each $\{i\}$ is a subset of $\{0,1\}^{\ell_{\mathsf{key}}}$ of density $\frac{1}{t} \geq \delta/2$, for $i = 1, 2, \ldots, t$. The reduction then guarantees that

$$\Pr_{\sigma, R_{\mathsf{coll}}}\left[R_{\mathsf{key}}(i;\sigma) = \kappa'; R_{\mathsf{coll}}^{\mathcal{H}}(i, \sigma, A(\kappa')) \text{ outputs a collision on } f_i\right] > \left(\tfrac{\delta\epsilon}{n}\right)^c$$

In fact, the above statement is true even if we restrict A to only output collisions for κ' lying in some subset S' of $\{0,1\}^{\ell_{\mathsf{key}}'}$ of density ϵ. By a probabilistic argument, this implies that for every subset S' of $\{0,1\}^{\ell_{\mathsf{key}}'}$ of density ϵ, there exists $\kappa' \in S'$ such that:

$$\Pr\left[\sigma \leftarrow R_{\mathsf{key}}(i;\cdot) = \kappa'; R_{\mathsf{coll}}^{\mathcal{H}}(i, \sigma, A(\kappa')) \text{ outputs a collision on } h_\kappa\right] > \left(\tfrac{\delta\epsilon}{n}\right)^c$$

Call such a κ' i-good. Then, for each i, a $1 - \epsilon$ fraction of κ' is i-good. By a union bound, there exists a $1 - t\epsilon$ fraction of κ' that are i-good, for all $i = 1, 2, \ldots, t$. Hence,

$$\begin{aligned}&\Pr_{\tilde{R}}\left[\tilde{R}^{f_1,\ldots,f_t} \text{ outputs collisions for each of } f_1, \ldots, f_t\right]\\ &\geq (1 - t\epsilon) \cdot 2^{-\ell_{\mathsf{out}}'-1} \cdot \left(\tfrac{\delta\epsilon}{n}\right)^{ct}\end{aligned}$$

Note that the preceding inequality holds for all functions $f_1, \ldots, f_t$ and thus also holds for random functions $f_1, \ldots, f_t$. On the other hand, by the birthday paradox and independence of the t functions, we know that the probability (over random functions) $\tilde{R}$ outputs collisions in each of $f_1, \ldots, f_t$ is at most $\left(\frac{Q^2}{2^{\ell_{\mathsf{out}}}}\right)^t$, where $Q = \mathrm{poly}(n, \frac{1}{\delta}, \frac{1}{\epsilon})$ is the query complexity of $\tilde{R}$. Comparing the two bounds and solving for ℓ_{out}' yields the desired bound. □

The above argument also yields a lower bound on the output length for (t', t)-combiners. The idea is to use R to find $t - t' + 1$ collisions amongst random functions $f_1, \ldots, f_t$ and observe that the probability is bounded by $\binom{t}{t-t'+1} \cdot (\frac{Q^2}{2^{\ell_{\mathsf{out}}}})^{t-t'+1}$. This establishes the optimality of our construction in Theorem 2 (up to constant factors in the $O(\log n)$ term):

Theorem 4. *Suppose (C, R) is a randomized black-box (t', t)-combiner for CRHFs. Let Q be an upper bound on the query complexity of R. Then,*

$$\ell_{\mathsf{out}}' \geq (t - t' + 1)(\ell_{\mathsf{out}} - 2\log Q) - t - 1$$

Acknowledgments. We would like to thank Krzysztof Pietrzak for helpful discussions on combiners.

References

1. Ross Anderson. The classification of hash functions. In *Cryptography and Coding '93*, 1993.
2. Dan Boneh and Xavier Boyen. On the impossibility of efficiently combining collision resistant hash functions. In *Proc. Crypto '06*, 2006.
3. Jean-Sébastien Coron, Yevgeniy Dodis, Cécile Malinaud, and Prashant Puniya. Merkle-Damgård revisited: How to construct a hash function. In *Proc. Crypto '06*, 2006.
4. Ivan Damgård. A design principle for hash functions. In *Proc. Crypto '89*, 1989.
5. Alfredo De Santis, Giovanni Di Crescenzo, and Giuseppe Persiano. Randomness-optimal characterization of two NP proof systems. In *Proc. Random '02*, 2002.
6. Hans Dobbertin. Cryptanalysis of MD4. In *Fast Software Encryption*, 1996.
7. Marc Fischlin and Anja Lehmann. Security-amplifying combiners for collision-resistant hash functions. In *these proceedings*, 2007.
8. J. K. Gibson. Discrete logarithm hash function that is collision free and one way. *IEE Proceedings - E*, 138(6):407–410, 1991.
9. Oded Goldreich. A sample of samplers - a computational perspective on sampling. ECCC TR97-020, 1997.
10. Oded Goldreich. Candidate one-way functions based on expander graphs. Cryptology ePrint Archive, Report 2000/063, 2000.
11. Oded Goldreich. *Foundations of Cryptography: Basic Tools*. Cambridge University Press, 2001.
12. Amir Herzberg. Tolerant combiners: Resilient cryptographic design. Cryptology ePrint Archive, Report 2002/135, 2002.
13. Chun-Yuan Hsiao and Leonid Reyzin. Finding collisions on a public road, or do secure hash functions need secret coins? In *Proc. Crypto '04*, 2004.
14. Lars R. Knudsen and Bart Preneel. Construction of secure and fast hash functions using nonbinary error-correcting codes. *IEEE Transactions on Information Theory*, 48(9):2524–2539, 2002.
15. Henry Lin, Luca Trevisan, and Hoeteck Wee. On hardness amplification of one-way functions. In *Proc. 2nd TCC*, 2005.
16. Alfred J. Menezes, Paul C. van Oorschot, and Scott A. Vanstone. *Handbook of Applied Cryptography*. CRC Press, 1996.
17. Ralph C. Merkle. One way hash functions and DES. In *Proc. Crypto '89*, 1989.
18. Moni Naor and Moti Yung. Universal one-way hash functions and their cryptographic applications. In *Proc. 20th STOC*, 1989.
19. Krzysztof Pietrzak. Non-trivial black-box combiners for collision-resistant hash-functions don't exist. In *Proc. Eurocrypt '07*, 2007. Cryptology ePrint Archive, Report 2006/348.
20. Bart Preneel. Hash functions - present state of art. ECrypt Conference on Hash Functions, 2005.
21. Phillip Rogaway. Formalizing human ignorance: Collision-resistant hashing without the keys. In *Proc. Vietcrypt '06*, 2006.
22. Jeanette P. Schmidt, Alan Siegel, and Aravind Srinivasan. Chernoff-Hoeffding bounds for applications with limited independence. *SIAM J. Discrete Math*, 8(2):223–250, 1995.
23. Ronen Shaltiel. Towards proving strong direct product theorems. *Computational Complexity*, 12(1–2):1–22, 2003.
24. Victor Shoup. A composition theorem for universal one-way hash functions. In *Proc. Eurocrypt '00*, 2000.
25. Daniel R. Simon. Finding collisions on a one-way street: Can secure hash functions be based on general assumptions? In *Proc. Eurocrypt '98*, 1998.
26. Xiaoyun Wang, Yiqun Lisa Yin, and Hongbo Yu. Finding collisions in the full SHA-1. In *Proc. Crypto '05*, 2005.
27. Andrew Yao. Theory and applications of trapdoor functions. In *Proc. 23rd FOCS*, 1982.