Electronic Colloquium on Computational Complexity, Report No. 97 (2017)

Multi Collision Resistant Hash Functions and their Applications*

Itay Berman Akshay Degwekar Ron D. Rothblum
Prashant Nalini Vasudevan

May 30, 2017

Abstract

Collision resistant hash functions are functions that shrink their input, but for which it is computationally infeasible to find a collision, namely two strings that hash to the same value (although collisions are abundant).

In this work we study *multi-collision resistant hash functions* (MCRH) a natural relaxation of collision resistant hash functions in which it is difficult to find a t-way collision (i.e., t strings that hash to the same value) although finding $(t-1)$-way collisions could be easy. We show the following:

- The existence of MCRH follows from the average case hardness of a variant of *Entropy Approximation*, a problem known to be complete for the class NISZK.
- MCRH imply the existence of *constant-round* statistically hiding (and computationally binding) commitment schemes.

In addition, we show a blackbox separation of MCRH from any one-way permutation.

*MIT. Emails: {itayberm, akshayd, ronr, prashvas}@mit.edu. Research supported in part by NSF Grants CNS-1413920 and CNS-1350619, and by the Defense Advanced Research Projects Agency (DARPA) and the U.S. Army Research Office under contracts W911NF-15-C-0226 and W911NF-15-C-0236.

ISSN 1433-8092

Contents

1 Introduction

Hash functions are efficiently computable functions that shrink their input and mimic 'random functions' in various aspects. They are prevalent in cryptography: both in theory and in practice. A central goal in the study of the foundations of cryptography has been to distill the precise, and minimal, security requirements necessary from hash functions for different applications.

One widely studied notion of hashing is that of *collision resistant hash functions* (CRHF). Namely, hash functions for which it is computationally infeasible to find two strings that hash to the same value, even when such collisions are abundant. CRHF have been extremely fruitful and have notable applications in cryptography such as digital signatures[1] [GMR88], efficient argument systems for NP [Kil92, Mic00] and (constant-round) statistically hiding commitment schemes [NY89, DPP93, HM96].

In this work we study a natural relaxation of collision resistance. Specifically, we consider hash functions for which it is infeasible to find a t-way collision: i.e., t strings that all have the same hash value. Here t is a parameter, where the standard notion of collision resistance corresponds to the special case of $t = 2$. Loosely speaking, we refer to such functions as multi-collision resistant hash functions (MCRH) and emphasize that, for $t > 2$, it is a *weaker* requirement than that of standard collision resistance. To the best of our knowledge, MCRH were first considered by Joux [Jou04], who showed that for specific classes of hash functions called *iterated* hash functions, finding a large number of collisions is no harder than finding just two colliding inputs.[2]

As in the case of CRH, to obtain a meaningful definition, we must consider keyed functions (since for non keyed functions there are trivial non-uniform attacks). Thus, we define MCRH as follows.

Definition 1.1 ((s,t)-MCRH)**.** *Let $s = s(n) \in \mathbb{N}$ and $t = t(n) \in \mathbb{N}$ be functions computable in time* $\mathsf{poly}(n)$.[3] *An (s,t)-*Multi-Collision Resistant Hash Function Family ((s,t)-MCRH) *consists of a probabilistic polynomial-time algorithm* Gen *that on input 1^n outputs a circuit h such that:*

- **s-Shrinkage:** *The circuit $h : \{0,1\}^n \to \{0,1\}^{n-s}$ maps inputs of length n to outputs of length $n - s$.*
- **t-Collision Resistance:** *For every polynomial size family of circuits $\mathsf{A} = (\mathsf{A}_n)_{n\in\mathbb{N}}$,*

$$\Pr_{\substack{h\leftarrow\mathsf{Gen}(1^n),\\ (x_1,x_2,\ldots,x_t)\leftarrow\mathsf{A}_n(h)}}\left[\begin{array}{c}\text{For all } i\neq j,\\ h(x_i)=h(x_j) \text{ and } x_i\neq x_j\end{array}\right] < \mathsf{negl}(n).$$

Note that the standard notion of CRH simply corresponds to $(1,2)$-MCRH (which is easily shown to be equivalent to $(s,2)$-CRH for any $s = n - \omega(\log n)$). We also remark that Definition 1.1 gives a *non-uniform* security guarantee, which is natural, especially in the context of collision resistance. Note though that all of our results are obtained by *uniform* reductions.

[1] We remark that the weaker notion of universal one-way hash functions (UOWHF) (which is known to be implied by standard one-way functions) suffices for this application [NY89, Rom90].

[2] We emphasize that Joux's result only applies to *iterated* hash functions and in the general case (i.e., arbitrary hash functions) it seems that MCRH is a weaker property than CRH.

[3] Here and throughout this work, we use n to denote the security parameter.

Remark 1.2 (Shrinkage vs. Collision Resistance)**.** *Observe that* (s,t)-MCRH *are meaningful only when* $s \geq \log t$*, as otherwise* t*-way collisions might not even exist (e.g., consider a* $(t-1)$ *regular function mapping inputs of length* n *to outputs of length* $n - \log(t-1)$*).*

Moreover, we note that in contrast to standard CRH*, it is unclear whether the shrinkage factor* s *can be trivially improved (e.g., by composition) while preserving the value of* t*. Specifically, constructions such as Tree Hashing (aka Merkle Tree) inherently rely on the fact that it is computationally infeasible to find* any *collision. It is possible to get some trade-offs between the number of collisions and shrinkage. For example, given an* $(s=2,t=4)$-MCRH*, we can compose it with itself to get an* $(s=4,t=10)$-MCRH*. But it is unclear if transformations that increase the shrinkage* s *while not increasing* t *exist.*

1.1 Our Results

The focus of this work is providing a systematic study of MCRH. We consider both the question of constructing MCRH and what applications can we derive from them.

Constructions. Since any CRH is in particular also an MCRH, candidate constructions are abundant (based on a variety of concrete computational assumptions). The actual question that we ask, which has a more foundational flavor, is whether we can construct MCRH from assumptions that are not known to imply CRH.

Our first main result is that the existence of MCRH follows from the average-case hardness of a variant of the *Entropy Approximation* problem studied by Goldreich, Sahai and Vadhan [GSV99]. Entropy Approximation, denoted EA, is a promise problem, where YES inputs are circuits whose output distribution[4] has entropy at least k, whereas NO inputs are circuits whose output distribution has entropy less than $k-1$ (where k is a parameter that is unimportant for the current discussion). Here by entropy we specifically refer to *Shannon* entropy.[5] Goldreich et al. showed that EA is complete for the class of languages that have non-interactive statistical zero-knowledge proofs (NISZK).

In this work we consider a variant of EA with respect to different notions of entropy. Specifically, consider the promise problem $\mathsf{EA}_{\text{min,max}}$, where the goal now is to distinguish between circuits whose output distribution has *min*-entropy[6] at least k from those with *max*-entropy at most $k-1$. It is easy to verify that $\mathsf{EA}_{\text{min,max}}$ is a strictly easier problem than EA.

The problem $\mathsf{EA}_{\text{min,max}}$ was previously studied by Dvir et al. [DGRV11], who showed that its average-case hardness follows from either quadratic residuocity (QR) or decisional Diffie Hellman assumptions (DDH).[7] Moreover, it is easy to see that the hardness of $\mathsf{EA}_{\text{min,max}}$ follows from the average-case hardness of Shortest/Closest Vector Problem with approximation factor $\sqrt{n}$ [GG98]. Assuming such average-case hardness of $\mathsf{EA}_{\text{min,max}}$ we construct MCRH.

Theorem 1 (Informal, see Theorem 3.6)**.** *If* $\mathsf{EA}_{\text{min,max}}$ *is average-case hard, then there exist* (s,t)-MCRH*, where* $s=\sqrt{n}$ *and* $t=6n^2$*.*

[4] By the output distribution of a circuit, we mean the distribution generated over its outputs when its inputs are sampled uniformly at random.

[5] Recall that the Shannon Entropy of a random variable X is defined as $\mathrm{H}_{\text{Shannon}}(X) = \mathrm{E}_{x \leftarrow X}\left[\log\left(\frac{1}{\Pr[X=x]}\right)\right]$.

[6] For a random variable X, the min-entropy is defined as $\mathrm{H}_{\min}(X) = \min_{x \in \mathrm{Supp}(X)} \log\left(\frac{1}{\Pr[X=x]}\right)$ whereas the max-entropy is $\mathrm{H}_{\max}(X) = \log\left(|\mathrm{Supp}(X)|\right)$.

[7] In fact, [DGRV11] show that the same conclusion holds even if we restrcit to NC_0 circuits.

(Note that in the MCRH that we construct there exist $2^{\sqrt{n}}$-way collisions, but it is computationally hard to find even a $6n^2$-way collision.)

Since we do not know whether $\mathsf{EA}_{\mathrm{min,max}}$ is also complete for NISZK, we remark that establishing the existence of MCRH based solely on the average-case hardness of NISZK (or SZK) is a fascinating open problem. Indeed such a result (which seems quite plausible) would be an interesting extension of Ostrovsky's [Ost91] proof that average-case hardness of SZK implies the existence of one-way functions.

Applications. The main application that we derive from MCRH is a *constant-round* statistically-hiding commitment scheme.

Theorem 2 (Informally stated, see Theorem 4.4)**.** *Assume that there exists a* $(\log(t), t)$-MCRH*. Then, there exists a* 3*-round statistically-hiding and computationally-binding commitment scheme.*

We note that Theorem 2 is optimal in the sense of holding for MCRH that are minimally shrinking. Indeed, as noted in Remark 1.2, (s,t)-MCRH with $s \leq \log(t-1)$ exist trivially and unconditionally.

It is also worthwhile to point out that by a result of Haitner et al. [HNO+09], statistically-hiding commitment schemes can be based on the existence of any one-way function. However, the commitment scheme of [HNO+09] uses a polynomial number of rounds of interaction and the main point in Theorem 2 is that it only uses only a *constant* number of rounds.

Moreover, by a result of [HHRS07], any *fully black-box* construction of a statistically hiding commitment schemes from one-way permutations (let alone one-way functions) must use a polynomial number of rounds. Loosely speaking, a construction is 'fully black-box' if the construction only requires an input-output access to the underlying primitive and the security proof also relies on the adversary in a black-box way. Most constructions in cryptography are fully black-box. Since our proof of Theorem 2 is via a fully black-box construction, we obtain the following immediate corollary:

Corollary 3 (Informally stated, see Theorem A.2)**.** *There is no fully blackbox construction of* MCRH *from one-way permutations.*

For self containment, we give a direct proof of Corollary 3 (i.e., without relying on the results [HHRS07]) in Appendix A.

1.2 Related Works

The main result of the work of Dvir et al. [DGRV11] (that was mentioned above) was showing that the problem EA for degree-3 polynomial mappings (i.e., where the entropies are measured by Shannon entropy) is complete for $\mathsf{SZK}_{\mathsf{L}}$, a sub-class of SZK in which the verifier and the simulator run in logarithmic space. They also construct algorithms to approximate different notions of entropy in certain restricted settings (and specifically, their algorithms do not violate the assumption that $\mathsf{EA}_{\mathrm{min,max}}$ is average-case hard).

Peikert and Waters [PW11] construct CRH from lossy trapdoor functions. Their construction can be viewed as a construction of CRH from $\mathsf{EA}_{\mathrm{min,max}}$ with a huge gap. (Specifically, the lossy trapdoor function h is either injective (i.e., $\mathrm{H}_{\min}(h) \geq n$) or very shrinking (i.e., $\mathrm{H}_{\max}(h) <$

$0.5n$).[8] One possible approach to constructing CRH from lossy functions with small 'lossiness' ($\mathrm{H}_{\max}(h)/\mathrm{H}_{\min}(h)$) is to first amplify the lossiness and then apply the [PW11] construction. Pietrzak et al. [PRS12] rule out this approach by showing that it is impossible to improve the 'lossiness' in a black-box way.[9] We show that even with distributions where the gap is tiny, we can achieve weaker yet very meaningful notions of collision-resistance.

Applebaum and Raykov [AR16] construct CRH from any average-case hard language with a *perfect randomized encoding*. Perfect Randomized Encodings are a way to encode the computation of a function f on input x such that information-theoretically, the *only* information revealed about x is the value $f(x)$.[10] The class of languages with such randomized encodings PRE is contained in PZK. Their assumption of an average-case hard language with a perfect randomized encoding implies $\mathsf{EA}_{\min,\max}$ as well.

Finally, Ong and Vadhan [OV08] construct constant-round statistically-hiding commitment schemes from average-case hardness of SZK.[11] Our construction of statistically-hiding commitments via MCRH is arguably simpler, although it relies on a stronger assumption ($\mathsf{EA}_{\min,\max}$) instead of average-case hardness of SZK.

Independent and Concurrent Work. MCRH have been recently considered in an independent and concurrent work by Komargodski et al. [KNY17]. Komargodski et al. study the problem, arising from Ramsey theory, of finding either a clique or an independent set (of roughly logarithmic size) in a graph, when such objects are guaranteed to exist. Interestingly, [KNY17] relate MCRH to the hardness of a bipartite variant of the foregoing Ramsey problem.

Beyond the work of [KNY17], we have very recently also been informed of two other concurrent works that study MCRH [KNY, BPK]. A comparison of these works with ours will be posted in an upcoming revision.

1.3 Our Techniques

We provide a detailed overview of our two main results: Constructing MCRH from $\mathsf{EA}_{\min,\max}$ and constructing constant-round statistically-hiding commitment scheme from MCRH.

1.3.1 Constructing MCRH from $\mathsf{EA}_{\min,\max}$

Assume that we are given a distribution on circuits $\left\{C\colon \{0,1\}^n \to \{0,1\}^{2n}\right\}$ such that that it is hard to distinguish between the cases $\mathrm{H}_{\min}(C) \geq k$ or $\mathrm{H}_{\max}(C) \leq k-1$, where we overload notation and let C also denote the output distribution of the circuit when given uniformly random inputs. Note that we have set the output length of the circuit C to $2n$ but this is mainly for concreteness (and to emphasize that the circuit need not be shrinking).

[8] The trapdoor to the lossy function is not used in the construction of CRH.

[9] It is easy to see that repetition amplifies the additive gap between the min-entropy and the max-entropy. In fact, we use this in our construction.

[10] Applebaum and Raykov [AR16] need the randomized encoding to satisfy some additional structural properties e.g. the encoding algorithm is one-to-one as a function of the randomness.

[11] Actually, Ong and Vadhan [OV08] only construct instance-dependent commitments. Dvir et al. [DGRV11] attribute the construction of constant-round statistically hiding commitments to an unpublished manuscript of Rothblum and Vadhan [RV09].

Our goal is to construct an MCRH using C. We will present our construction in steps, where in the first case we start off by assuming a very large entropy gap. Specifically, for the first (over-simplified) case, we assume that it is hard to distinguish between min-entropy $\geq n$ vs. max-entropy $\leq n/2$.[12] Note that having min-entropy n means that C is *injective*.

Warmup: The case of $\mathrm{H}_{\min}(C) \geq n$ vs. $\mathrm{H}_{\max}(C) \ll n/2$. In this case, it is already difficult to find even a 2-way collision in C: if $\mathrm{H}_{\min}(C) \geq n$, then C is injective and no collisions exist. Thus, if one can find a collision, it must be the case that $\mathrm{H}_{\max}(C) \leq n/2$ and so any collision finder distinguishes the two cases.

The problem though is that C by itself is not shrinking, and thus is not an MCRH. To resolve this issue, a natural idea that comes to mind is to hash the output of C, using a pairwise independent hash function.[13] Thus, the first idea is to choose $f : \{0,1\}^{2n} \to \{0,1\}^{n-s}$, for some $s \geq 1$, from a family of pairwise independent hash functions and consider the hash function $h(x) = f(C(x))$.

If $\mathrm{H}_{\min}(C) \geq n$ (i.e., C is injective), then every collision in h is a collision on the hash function f. On the other hand, if $\mathrm{H}_{\max}(C) \leq n/2$, then C itself has many collisions. To be able to distinguish between the two cases, we would like that in the latter case there will be no collisions that originate from f. The image size of C, if $\mathrm{H}_{\max}(C) \ll n/2$, is smaller than $2^{n/2}$. If we set s to be sufficiently small (say constant) than the range of f has size roughly 2^n. Thus, we are hashing a set into a range that is more than quadratic in its size. In such case, we are "below the birthday paradox regime" and a *random function* on this set will be injective. A similar statement can be easily shown also for functions that are merely pairwise independent (rather than being entirely random).

Thus, in case C is injective, all the collisions appear in the second part of the hash function (i.e., the application of f). On the other hand, if C has max-entropy smaller than $n/2$, then all the collisions happen in the first part of the hash function (i.e., in C). Thus, any adversary that finds a collision distinguishes between the two cases and we actually obtain a full-fledged CRH (rather than merely an MCRH) at the cost of making a much stronger assumption.

The next case that we consider is still restricted to circuits that are injective (i.e., have min entropy n) in one case but assumes that it is hard to distinguish injective circuits from circuits having max-entropy $n - \sqrt{n}$ (rather than $n/2$ that we already handled).

The case of $\mathrm{H}_{\min}(C) \geq n$ vs. $\mathrm{H}_{\max}(C) \leq n - \sqrt{n}$. The problem that we encounter now is that in the low max entropy case, the output of C has max-entropy $n - \sqrt{n}$. To apply the above birthday paradox argument we would need the range of f to be of size roughly $(2^{n-\sqrt{n}})^2 \gg 2^n$ and so our hash function would not be shrinking. Note that if the range of f were smaller, than even if f were chosen entirely at random (let alone from a pairwise independent family) we would see collisions in this case (again, by the birthday paradox).

The key observation that we make at this point is that although we will see collisions, there will not be too many of them. Specifically, suppose we set $s \approx \sqrt{n}$. Then, we are now hashing a set of size $2^{n-\sqrt{n}}$ into a range of size $2^{n-\sqrt{n}}$. If we were to choose f entirely at random, this process would correspond to throwing $N = 2^{n-\sqrt{n}}$ balls (i.e., the elements in the range of C) into N bins

[12] This setting (and construction) is similar to that of Peikert and Waters's construction of CRH from lossy functions [PW11].

[13] Recall that a collection of functions $\mathcal{F}$ is k-wise independent if for every distinct $x_1, \ldots, x_k$, the distribution of $(f(x_1), \ldots, f(x_k))$ (over the choice of $f \leftarrow \mathcal{F}$) is uniform.

(i.e., elements in the range of f). It is well-known that in such case, with high probability, the maximal load for any bin will be at most $\frac{\log(N)}{\log\log(N)} < n$. Thus, we are guaranteed that there will at most n collisions.

Unfortunately, the work of Alon et al. [ADM+99] shows that the same argument does not apply to functions that are merely pairwise independent (rather than entirely random). Thankfully though, suitable derandomizations are known. Specifically, it is not too difficult to show that if we take f from a family of *n-wise independent hash functions*, then the maximal load will also be at most n.[14]

Similarly to before, in case C is injective, there are no collisions in the first part. On the other hand, in case C has max-entropy at most $n - \sqrt{n}$, we have just argued that there will be less than n collisions in the second part. Thus, an adversary that finds at least n collisions distinguishes between the two cases and we have obtained an (s,t)-MCRH, with $s = \sqrt{n}$ and $t = n$.

The case of $\mathrm{H}_{\min}(C) \geq k$ vs. $\mathrm{H}_{\max}(C) \leq k - \sqrt{n}$. We want to remove the assumption that when the min-entropy of C is high, then it is in fact injective. Specifically, we consider the case that either C's min-entropy is at least k (for some parameter $k \leq n$) or its max entropy is at most $k - \sqrt{n}$. Note that in the high min-entropy case, C — although not injective — maps at most 2^{n-k} inputs to every output (this is essentially the definition of min-entropy). Our approach is to apply hashing a second time (in a different way), to effectively make C injective, and then apply the construction from the previous case.

Consider the mapping $h'(x) = (C(x), f_1(x))$, where f_1 will be defined ahead. For h' to be injective, f_1 must be injective over all sets of size 2^{n-k}. Taking f_1 to be pairwise-independent will force to set its output length to be too large, in a way that will ruin the entropy gap between the cases.

As in the previous case, we use many-wise independent hashing. Let $f_1 \colon \{0,1\}^n \to \{0,1\}^{n-k}$ be a $3n$-wise independent hash function. If $\mathrm{H}_{\min}(C) \geq k$, then the same load-balancing property of f that we used in the previous case, along with a union bound, implies that with high probability (over the choice of f_1) there will be no $3n$-way collisions in h'. Our final construction applies the previous construction on h'. Namely,

$$h_{C,f_1,f_2}(x) = f_2(C(x), f_1(x)),$$

for $f_1 \colon \{0,1\}^n \to \{0,1\}^{n-k}$ and $f_2 \colon \{0,1\}^{3n-k} \to \{0,1\}^{n-\sqrt{n}}$ being $3n$-wise and $2n$-wise independent hash functions, respectively. We can now show that

- If $\mathrm{H}_{\min}(C) \geq k$, then there do not exist $3n$ distinct inputs $x_1, \ldots, x_{3n}$ such that they all have the same value of $(C(x_i), f_1(x_i))$; and
- If $\mathrm{H}_{\max}(C) \leq k - \sqrt{n}$, then there do not exist $2n$ distinct inputs $x_1, \ldots, x_{2n}$ such that they all have distinct values of $(C(x_i), f_1(x_i))$, but all have the same value $f_2(C(x_i), f_1(x_i))$.

We claim that h_{C,f_1,f_2} is (s,t)-MCRH for $s = \sqrt{n}$ and $t = 6n^2$: First, note that in any set of $6n^2$ collisions for h_{C,f_1,f_2}, there has to be either a set of $3n$ collisions for (C, f_1) or a set of $2n$ collisions for f_2, and so at least one of the conditions in the above two statements is violated. Now, assume that A finds $6n^2$-way collision in h_{C,f_1,f_2} with high probability. Then, an algorithm

[14] We remark that more efficient constructions are known, see Remark 2.5.

D that distinguishes between $\mathrm{H}_{\min}(C) \geq k$ to $\mathrm{H}_{\max}(C) \leq k - \sqrt{n}$ chooses f_1 and f_2 uniformly at random and runs A on the input $h = h_{C,f_1,f_2}$ to get $x_1, \ldots, x_{6n^2}$ with $h(x_1) = \cdots = h(x_{6n^2})$. The distinguisher D now checks which of the two conditions above is violated, and thus can distinguish if it was given C with $\mathrm{H}_{\min}(C) \geq k$ or $\mathrm{H}_{\max}(C) \leq k - \sqrt{n}$.

We proceed to the case that the entropy gap is 1 (rather than $\sqrt{n}$).

The case of $\mathrm{H}_{\min}(C) \geq k$ vs. $\mathrm{H}_{\max}(C) \leq k - 1$. This case is handled by reduction to the previous case. The main observation is that if C has min-entropy at least k, and we take ℓ copies of C, then we get a new circuit with min-entropy at least $\ell \cdot k$. In contrast, if C had max-entropy at most $k - 1$, then C' has max-entropy at most $\ell \cdot k - \ell$. Setting $\ell = k$, we obtain that in the second case the max-entropy is $n' - \sqrt{n'}$, where $n' = \ell \cdot k$ is the new input length. Thus, we have obtained a reduction to the $\sqrt{n}$ gap case that we already handled.

1.3.2 Constructing Constant-Round Statisticall-Hinidng Commitment from MCRH

As a warm-up, we start with the construction of statistically-hiding commitment scheme from standard collision-resistance hash functions (i.e., 2-MCRH).

Warmup: Commitment from (Standard) CRH. Given a family of CRHFs $\mathcal{H} = \left\{h \colon \{0,1\}^n \to \{0,1\}^{n-1}\right\}$, a natural first attempt is to have the receiver sample the hash function $h \leftarrow \mathcal{H}$ and send it to the sender. The sender, trying to commit to a bit b, chooses $x \leftarrow \{0,1\}^n$ and $r \leftarrow \{0,1\}^n$, and sends $(y = h(x), r, \sigma = \langle r, x\rangle \oplus b)$ to the receiver. The commitment is defined as $c = (h, y, r, \sigma)$. To reveal, the sender sends (x, b) to the receiver, which verifies that $h(x) = y$ and $\sigma = \langle r, x\rangle \oplus b$. Pictorially, the commit stage is as follows:

$$
\begin{array}{lcl}
\mathsf{S}(b) & & \mathsf{R} \\
 & \xleftarrow{\qquad h \qquad} & h \leftarrow \mathsf{Gen}(1^n) \\
x, r \leftarrow \{0,1\}^n & \xrightarrow{\; c = (h(x), r, \langle r, x\rangle \oplus b) \;} &
\end{array}
$$

The fact that the scheme is computational-binding follows immediately from the collision resistance of h: if the sender can find $(x, 0)$ and $(x', 1)$ that pass the receiver's verification, then $x \neq x'$ and $h(x) = h(x')$.

Arguing that the scheme is statistically-hiding is trickier. The reason is that $h(x)$ might reveal a lot of information on x. What helps us is that h is *shrinking*, and thus some information about x is hidden from the receiver. In particular, this means that x has positive min-entropy given $h(x)$. At this point we would like to apply the Leftover Hash Lemma (LHL) to show that for any b, the statistical distance between $(h(x), r, \langle r, x\rangle \oplus b)$ to $(h(x), r, u)$ is small. Unfortunately, the min-entropy is insufficient for the LHL and indeed the distance between these two distributions is a constant (rather than negligible as required).

To reduce the statistical distance, we increase the min-entropy via repetition. We modify the protocol so that the sender selects k values $\mathbf{x} = (x_1, \ldots, x_k) \leftarrow \{0,1\}^{nk}$ and $r \leftarrow \{0,1\}^{nk}$, and sends $(h(x_1), \ldots, h(x_k), r, \langle r, \mathbf{x}\rangle \oplus b)$ to the receiver. The min-entropy of $\mathbf{x}$, even given $h(x_1), \ldots, h(x_k)$ is now $\Omega(k)$, and the LHL now yields that the statistical distance between the two distributions

$(h, h(x_1), \ldots, h(x_k), r, \langle r, \mathbf{x}\rangle \oplus 0)$ and $(h, h(x_1), \ldots, h(x_k), r, \langle r, \mathbf{x}\rangle \oplus 1)$ is roughly 2^{-k}. Setting k to be sufficiently large (e.g., $k = \mathsf{poly}(n)$ or even $k = \mathsf{poly}\log(n)$) we obtain that the scheme is statistically-hiding. Note that repetition also does not hurt binding: if the sender can find valid decommitments $(\mathbf{x} = (x_1 \ldots, x_k), 0)$ and $(\mathbf{x}' = (x'_1, \ldots, x'_k), 1)$ that pass the receiver's verification, then there must exists $i \in [k]$ with $x_i \neq x'_i$ and $h(x_i) = h(x'_i)$ (i.e., a collision).

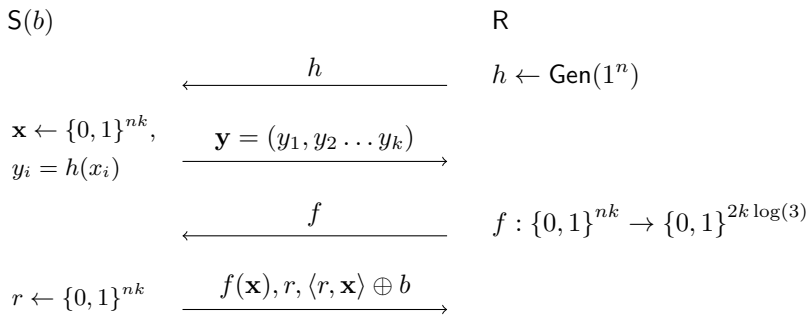
Handling MCRHs. For simplicity, let us focus on the case $t = 4$ (since it basically incorporates all the difficulty encountered when dealing the larger values of t). That is, we assume that $\mathcal{H} = \left\{h\colon \{0,1\}^n \to \{0,1\}^{n-2}\right\}$ is a (s,t)-MCRH where the shrinkage $s = 2$ and $t = 4$. Namely, it is hard to find 4 inputs that maps to the same hash value for a random function from $\mathcal{H}$, even though such 4-way collisions exist. Note however that it might very well be easy to find 3 such colliding inputs. And indeed, the binding argument that we had before breaks: finding $x \neq x'$ with $h(x) = h(x)$ is no longer (necessarily) a difficult task.

The problem comes up because even after the sender 'commits' to $y_1 = h(x_1), \ldots, y_k = h(x_k)$, it is no longer forced to reveal $x_1, \ldots, x_k$. Intuitively, for every y_i, the sender might know 3 inputs that map to y_i, so, the sender is free to reveal any value in the Cartesian product of these triples. Concretely, let $\mathcal{S}_{y_i}$ be the set of inputs that h maps to y_i that the sender can find efficiently, and let $\mathcal{S}_{\mathbf{y}} = \mathcal{S}_{y_1} \times \cdots \times \mathcal{S}_{y_k}$. Since the sender can find at most 3 colliding inputs, it holds that $|\mathcal{S}_{y_i}| \leq 3$ for every i, and thus $|\mathcal{S}_{\mathbf{y}}| \leq 3^k$. To fix the binding argument, we want to force every efficient sender to reveal a unique $\mathbf{x} = x_1, \ldots, x_k \in \mathcal{S}_{\mathbf{y}}$.

A first attempt toward achieving the above goal is to use a pairwise-independent hash function f that is injective over $\mathcal{S}_{\mathbf{y}}$ with high probability. At a high level, the sender will also specify to the receiver a random function f from the pairwise independent hash function family. The receiver in turn sends $f(\mathbf{x})$ as well as $(h(x_1), \ldots, h(x_k))$. The receiver adds a check to the verification step to ensure that f maps the input sequence $x'_1, \ldots, x'_k$ to the value that was pre-specified.

In order for the function f to be injective on the set $\mathcal{S}_{\mathbf{y}}$, the birthday paradox tells us that the range of f must have size at at least (roughly) $|\mathcal{S}_{\mathbf{y}}|^2$, which means at least 3^{2k}. Thus, to ensure that f is injective on $\mathcal{S}_{\mathbf{y}}$, we can use a pairwise-independent function $f\colon \{0,1\}^{nk} \to \{0,1\}^{2k\log(3)}$.

Unfortunately, this scheme is still not binding: f is promised (with high probability) to be injective for *fixed* sets of size 3^k, but the sender can choose $\mathbf{y}$ based on the value of f. Specifically, to choose $\mathbf{y}$ so that f is not injective over $\mathcal{S}_{\mathbf{y}}$. To fix the latter issue, we split the messages that the receiver send into two rounds. In the first round the receiver sends h and receives $\mathbf{y} = (h(x_1), \ldots, h(x_k))$ from the sender. Only then the receiver sends f and receive $z_1 = f(\mathbf{x})$. Now, the scheme is binding: since f is chosen *after* $\mathbf{y}$ is set, the pairwise-independence property guarantee that it will be injective over $\mathcal{S}_{\mathbf{y}}$ with high probability. Pictorially, the commit stage of the new scheme is as follows:

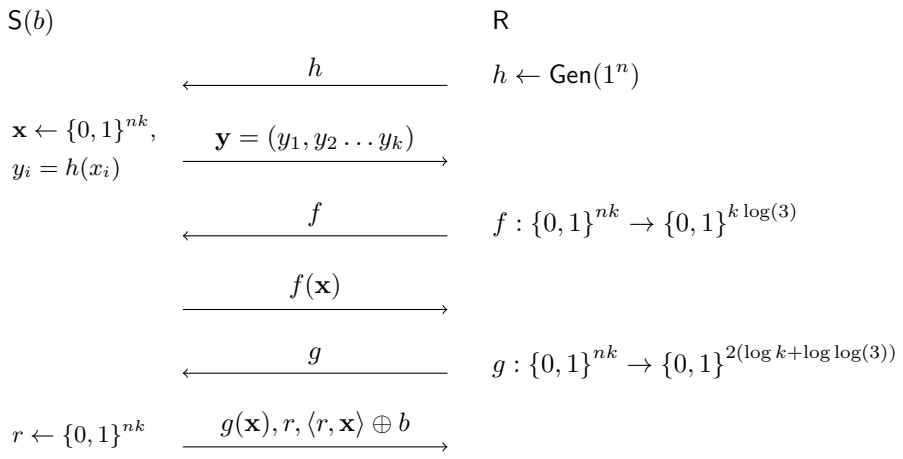


But is this scheme statistically-hiding? Recall that to argue hiding, we used the fact that the mapping $(x_1, \ldots, x_k) \mapsto (h(x_1), \ldots, h(x_k))$ is shrinking. Analogously here, we need the mapping $(x_1, \ldots, x_k) \mapsto (h(x_1), \ldots, h(x_k), f(\mathbf{x}))$ to be shrinking. However, the latter mapping maps $n \cdot k$ bits to $(n-2) \cdot k + 2\log(3) \cdot k$ bit, which is obviously not shrinking. One work-around is to simply assume that the given MCRH shrinks much more than we assumed so far. For example, to assume that $\mathcal{H}$ is $(4,4)$-MCRH (or more generally (s,t)-MCRH for $s \gg \log(t)$).[15] However, we can actually fix the protocol so that it gives statistically-hiding commitments even with tight shrinkage of $\log(t)$ by using hash functions that guarantee good load-balancing and adding one more round of interaction.

Overcoming the Birthday Paradox. To guarantee hiding, it seems that we cannot afford the domain of f to be as large as $(3^k)^2$. Instead, we set its domain size to 3^k (i.e., $f\colon \{0,1\}^{nk} \to \{0,1\}^{k\log(3)}$). Moreover, rather than choosing it from a pairwise independent hash function family, we shall use one that is *many-wise-independent*. Since such functions are load-balanced (see **??**) it holds that with high probability, z_1 — the value the sender sends in the second round — has at most $\log(3^k) = k \cdot \log(3)$ pre-images from $\mathcal{S}_{\mathbf{y}}$ under f (i.e., $|\{\mathbf{x} \in \mathcal{S}_{\mathbf{y}}\colon f(\mathbf{x}) = z_1\}| \leq k \cdot \log(3)$). We once more face the problem that the sender can reveal any of these inputs, but now their number is exponentially smaller — it is only $k\log(3)$ (as opposed to 3^k before). We can now choose a pairwise-independent $g\colon \{0,1\}^{nk} \to \{0,1\}^{2(\log(k)+\log\log(3))}$ that is injective over sets of size $k \cdot \log(3)$ (with high probability). For the same reasons that f was sent after h, the receiver sends g only after receiving $f(\mathbf{x})$.

Thus, our final protocol has three rounds (where each round is composed of one message for each of the two parties) and is as follows: In the first round, the receiver selects $h \leftarrow \mathcal{H}$ and sends it to the sender. The sender, trying to commit to a bit b, chooses $\mathbf{x} = (x_1, \ldots, x_k) \leftarrow \{0,1\}^{nk}$ and sends $\mathbf{y} = (y_1 = h(x_1), \ldots, y_k = h(x_k))$. In the second round, the receiver selects a many-wise-independent hash function $f\colon \{0,1\}^{nk} \to \{0,1\}^{k\log(3)}$ and sends it to the sender. The sender sends $z_1 = f(\mathbf{x})$ to the receiver. In the third and final round, the receiver selects a pairwise-independent hash function $g\colon \{0,1\}^{n\cdot k} \to \{0,1\}^{2(\log(k)+\log\log(3))}$ and sends it to the sender. The sender selects $r \leftarrow \{0,1\}^{nk}$, and sends $(z_2 = g(\mathbf{x}), r, \sigma = \langle r, \mathbf{x}\rangle \oplus b)$ to the receiver. The commitment is defined as $c = (h, \mathbf{y}, f, z_1, g, z_2, \sigma)$. To reveal, the sender sends $(\mathbf{x}, b)$ to the receiver, which verifies that $h(x_i) = y_i$ for every i, that $f(\mathbf{x}) = z_1$, $g(\mathbf{x}) = z_2$ and $\sigma = \langle r, \mathbf{x}\rangle \oplus b$. Pictorially, the commit stage is as follows:

[15] We remark that our construction of MCRH based on $\mathsf{EA}_{\min,\max}$ (see Section 3) actually supports such large shrinkage.



Intuitively, the scheme is computationally-binding since for any computationally-bounded sender that committed to c, there is a unique $\mathbf{x}$ that pass the receiver's verification. As for the hiding, we need the mapping $(x_1, \ldots, x_k) \mapsto (h(x_1), \ldots, h(x_k), f(\mathbf{x}), g(\mathbf{x}))$ to be shrinking. Observe that we are mapping $n \cdot k$ bits to $(n-2)k + \log(3)k + 2(\log(k) + \log\log(3))$ bits. Choosing k to be sufficiently large (e.g., $k = \mathsf{poly}(n)$ certainly suffices) yields that the mapping is shrinking.

Organization

In Section 2 we define the notion of many-wise independent hashing and prove that it has some load-balancing properties. In Section 3 we formally state the entropy approximation assumption and present our construction of MCRH. In Section 4 we describe the construction of constant-round statistically-hiding commitments from MCRH. Lastly, in Appendix A we provide a self-contained proof of the blackbox separation of MCRH from one-way functions (see Corollary 3).

2 Preliminaries

We use lowercase letters for values, uppercase for random variables, uppercase calligraphic letters (e.g., $\mathcal{U}$) to denote sets, boldface for vectors (e.g., $\mathbf{x}$), and uppercase sans-serif (e.g., A) for algorithms (i.e., Turing Machines). All logarithms considered here are in base two. Given a probabilistic polynomial-time algorithm A, we let $\mathsf{A}(x; r)$ be an execution of A on input x given randomness r. We let poly denote the set all polynomials. A function $\nu \colon \mathbb{N} \to [0,1]$ is *negligible*, denoted $\nu(n) = \mathsf{negl}(n)$, if $\nu(n) < 1/p(n)$ for every $p \in \mathsf{poly}$ and large enough n.

Given a random variable X, we write $x \leftarrow X$ to indicate that x is selected according to X. Similarly, given a finite set $\mathcal{S}$, we let $s \leftarrow \mathcal{S}$ denote that s is selected according to the uniform distribution on $\mathcal{S}$. We adopt the convention that when the same random variable occurs several times in an expression, all occurrences refer to a single sample. For example, $\Pr[f(X) = X]$ is defined to be the probability that when $x \leftarrow X$, we have $f(x) = x$. We write U_n to denote the random variable distributed uniformly over $\{0,1\}^n$. The support of a distribution D over a finite set $\mathcal{U}$, denoted $\mathrm{Supp}(D)$, is defined as $\{u \in \mathcal{U} : D(u) > 0\}$. The *statistical distance* of two distributions P and Q over a finite set $\mathcal{U}$, denoted as $\mathrm{SD}(P, Q)$, is defined as $\max_{\mathcal{S} \subseteq \mathcal{U}} |P(\mathcal{S}) - Q(\mathcal{S})| = \frac{1}{2} \sum_{u \in \mathcal{U}} |P(u) - Q(u)|$.

We make use of the following simple fact, that show that a $[0,1]$ random variable with high-enough expected value, cannot be very small with high probability.

Fact 2.1. *Assume that X is a random variable taking values in $[0,1]$ with $\mathrm{E}[X] \geq \mu$. Then, $\Pr[X \geq \mu^2] \geq \mu/(1+\mu)$.*

Proof. Let $p = \Pr[X \geq \mu^2]$. It holds that

$$\mu \leq \mathrm{E}[X] \leq p \cdot 1 + (1-p) \cdot \mu^2.$$

Rearranging, we have that

$$p \geq \frac{\mu - \mu^2}{1 - \mu^2} = \frac{\mu(1-\mu)}{(1+\mu)(1-\mu)} = \frac{\mu}{1+\mu},$$

as required. □

2.1 Many-wise Independent Hashing

Many-wise independent hash functions are used extensively in complexity theory and cryptography.

Definition 2.2 (ℓ-wise Independent Hash Functions)**.** *For $\ell \in \mathbb{N}$, a family of functions $\mathcal{F} = \{f : \{0,1\}^n \to \{0,1\}^m\}$ is ℓ-wise independent if for every distinct $x_1, x_2, \ldots, x_\ell \in \{0,1\}^n$ and every $y_1, y_2, \ldots, y_\ell \in \{0,1\}^m$, it holds that*

$$\Pr_{f \leftarrow \mathcal{F}}[f(x_1) = y_1 \wedge f(x_2) = y_2 \wedge \cdots \wedge f(x_\ell) = y_\ell] = \frac{1}{M^\ell}.$$

Note that if $\mathcal{H}$ is k-wise independent for $k \geq 2$, it is also universal. The existence of efficient many-wise hash function families is well known.

Fact 2.3 (c.f. [Vad12, Corollary 3.34])**.** *For every $n, m, \ell \in \mathbb{N}$, there exists a family of ℓ-wise independent hash functions $\mathcal{F}^{(\ell)}_{n,m} = \{f \colon \{0,1\}^n \to \{0,1\}^m\}$ where a random function from $\mathcal{F}_{n,m}$ can be selected using $\ell \cdot \max(m,n)$ bits, and given a description of $f \in \mathcal{F}^{(\ell)}_{n.m}$ and $x \in \{0,1\}^n$, the value $f(x)$ can be evaluated in time $\mathsf{poly}(n,m,\ell)$.*

Whenever we only need pairwise independent hash function $\mathcal{F}^{(2)}_{n,m}$, we remove the two from the superscript and simply write $\mathcal{F}_{n,m}$.

2.2 Load Balancing

The theory of load balancing deals with allocating elements into bins, such that no bin has too many elements. If the allocation is done at random, it can be shown that with high probability the max load (i.e., the number of elements in the largest bin) is not large. In fact, allocating via many-wise independent hash function also suffices.

Fact 2.4 (Folklore (see, e.g., [CRSW13]))**.** *Let $n, m, \ell \in \mathbb{N}$ with $\ell \geq 2e$ (where e is the base of the natural logarithm) and let $\mathcal{F}^{\ell}_{n,m}$ be an ℓ-wise independent hash function family. Then, for every set $\mathcal{S} \subseteq \{0,1\}^n$ with $|\mathcal{S}| \leq 2^m$ it holds that:*

$$\Pr_{f \leftarrow \mathcal{F}^{\ell}_{n,m}}\left[\exists y \in \{0,1\}^m \text{ such that } \left|f^{-1}(y) \cap \mathcal{S}\right| \geq \ell\right] \leq 2^{m-\ell},$$

where $f^{-1}(y) = \{x \in \{0,1\}^n \colon f(x) = y\}$.

Proof. Fix $y \in \{0,1\}^m$. It holds that

$$\begin{aligned}
\Pr_{f \leftarrow \mathcal{F}^\ell_{n,m}}\left[\left|f^{-1}(y) \cap \mathcal{S}\right| \geq \ell\right] &\leq \Pr_{f \leftarrow \mathcal{F}^\ell_{n,m}}[\exists \text{ distinct } x_1, \ldots, x_\ell \in \mathcal{S} \colon f(x_1) = y \wedge \cdots \wedge f(x_\ell) = y] \\
&\leq \sum_{\text{distinct } x_1,\ldots,x_\ell \in \mathcal{S}} \Pr_{f \leftarrow \mathcal{F}^\ell_{n,m}}[f(x_1) = y \wedge \cdots \wedge f(x_\ell) = y] \\
&\leq \binom{2^m}{\ell} \cdot \left(\frac{1}{2^m}\right)^\ell \\
&\leq \left(\frac{e \cdot 2^m}{\ell}\right)^\ell \cdot \left(\frac{1}{2^m}\right)^\ell \\
&\leq 2^{-\ell},
\end{aligned}$$

where the second inequality is by a union bound, the third inequality follows from the ℓ-wise independence of $\mathcal{F}^\ell_{n,m}$, the fourth inequality is by a standard bound on binomial coefficients, and the last inequality follows by our assumption that $\ell \geq 2e$.

Fact 2.4 follows from a union bound over all values of $y \in \{0,1\}^m$. □

Remark 2.5 (More Efficient Hash Functions)**.** *We remark that more efficient constructions of hash functions guaranteeing the same load balancing performance as in Fact 2.4 are known in the literature.*

Specifically, focusing on the setting of $\ell = O(m)$, Fact 2.4 gives a load balancing guarantee for functions whose description size (i.e., key length) is $\Omega(m^2)$ bits. In contrast, a recent result of Celis et al. [CRSW13] constructs such functions that require only $\tilde{O}(m)$ key size. Furthermore, a follow up work of Meka et al. [MRRR14] improves the evaluation time *of the [CRSW13] hash function to be only poly-logarithmic in m (in the word RAM model).*

However, since our focus is not on concrete efficiency, we ignore these optimizations throughout this work.

3 Constructing MCRH Families

In this section, we present a construction of a Multi-Collision Resistant Hash family based on the hardness of estimating certain notions of entropy of a distribution given a circuit that samples it. We define and discuss this problem in Section 3.1, and present the construction in Section 3.2.

3.1 Entropy Approximation

In order to discuss the problem central to our construction, we first recall some standard notions of entropy.

Definition 3.1. *For a random variable X, we define the following notions of entropy:*

- **Min-entropy:** $\mathrm{H}_{\min}(X) = \min_{x \in \mathrm{Supp}(X)} \log\left(\frac{1}{\Pr[X=x]}\right)$.
- **Max-entropy:** $\mathrm{H}_{\max}(X) = \log\left(|\mathrm{Supp}(X)|\right)$.
- **Shannon entropy:** $\mathrm{H}_{\mathrm{Shannon}}(X) = \mathrm{E}_{x \leftarrow X}\left[\log\left(\frac{1}{\Pr[X=x]}\right)\right]$.

For any random variable, these entropies are related as described below. These relations ensure that the problems we describe later are well-defined.

Fact 3.2. *For a random variable X supported over $\{0,1\}^m$,*

$$0 \le \mathrm{H}_{\min}(X) \le \mathrm{H}_{\mathrm{Shannon}}(X) \le \mathrm{H}_{\max}(X) \le m.$$

Given a circuit $C : \{0,1\}^n \to \{0,1\}^m$, we overload C to also denote the random variable induced by evaluating C on a uniformly random input from $\{0,1\}^n$. With this notation, the Entropy Approximation problem is defined as below.

Definition 3.3. *Let $g = g(n) \in \mathbb{R}$ be a function. The* min-max Entropy Approximation *problem with* gap g*, denoted $\mathsf{EA}^{(g)}_{\min,\max}$, is a promise problem* $(\mathrm{YES}, \mathrm{NO})$ *for* $\mathrm{YES} = \{\mathrm{YES}_n\}_{n\in\mathbb{N}}$ *and* $\mathrm{NO} = \{\mathrm{NO}_n\}_{n\in\mathbb{N}}$*, where we define*

$$\begin{aligned} \mathrm{YES}_n &= \{(1^n, C_n, k) \ : \ \mathrm{H}_{\min}(C_n) \ge k\}, \text{ and} \\ \mathrm{NO}_n &= \{(1^n, C_n, k) \ : \ \mathrm{H}_{\max}(C_n) \le k - g(n)\}, \end{aligned}$$

and where in both cases C_n is a circuit that takes n bits of input, and $k \in \{0, \ldots, n\}$.

We also define $\mathsf{EA}_{\min,\max} = \mathsf{EA}^{(1)}_{\min,\max}$. That is, when we omit g we simply mean that $g = 1$.

The Shannon Entropy Approximation problem (where $\mathrm{H}_{\min}$ and $\mathrm{H}_{\max}$ above are replaced with $\mathrm{H}_{\mathrm{Shannon}}$) with constant gap was shown by Goldreich et al. [GSV99] to be complete for the class NISZK (problems with non-interactive statistical zero knowledge proof systems). For a discussion of generalizations of Entropy Approximation to other notions of entropy, and other related problems, see [DGRV11].

3.1.1 The Assumption: Average-Case Hardness of Entropy Approximation

Our construction of MCRH is based on the average-case hardness of the Entropy Approximation problem $\mathsf{EA}_{\min,\max}$ defined above (i.e., with gap 1). We use the following definition of average-case hardness of promise problems.

Definition 3.4 (Average-case Hardness)**.** *We say that a promise problem $\Pi = (\mathrm{YES}, \mathrm{NO})$, where $\mathrm{YES} = \{\mathrm{YES}_n\}_{n\in\mathbb{N}}$ and $\mathrm{NO} = \{\mathrm{NO}_n\}_{n\in\mathbb{N}}$, is* $\mathsf{average\text{-}case\ hard}$ *if there is a probabilistic algorithm S such that $\mathsf{S}(1^n)$ outputs samples from $(\mathrm{YES}_n \cup \mathrm{NO}_n)$, and for every family of polynomial-sized circuits $\mathsf{A} = (\mathsf{A}_n)_{n\in\mathbb{N}}$,*

$$\Pr_{x \leftarrow \mathsf{S}(1^n)}[\mathsf{A}_n(x) = \Pi(x)] \le \frac{1}{2} + \mathsf{negl}(n),$$

where $\Pi(x) = 1$ if $x \in \mathrm{YES}$ and $\Pi(x) = 0$ if $x \in \mathrm{NO}$. We call S a hard-instance sampler *for Π. The quantity $(\Pr_{x\leftarrow\mathsf{S}(1^n)}[\mathsf{A}_n(x) = \Pi(x)] - 1/2)$ is referred to as the* advantage *the algorithm A has in deciding Π with respect to the sampler S.*

In our construction and proofs, it will be convenient for us to work with the problem $\mathsf{EA}^{(\lfloor\sqrt{n}\rfloor)}_{\min,\max}$ rather than $\mathsf{EA}_{\min,\max} = \mathsf{EA}^{(1)}_{\min,\max}$. At first glance $\mathsf{EA}^{(\lfloor\sqrt{n}\rfloor)}_{\min,\max}$ seems to be an easier problem because the gap here is $\lfloor\sqrt{n}\rfloor$, which is much larger than a constant. The following simple proposition shows that these two problems are in fact equivalent (even in their average-case complexity). The key idea here is repetition: given a circuit C, we can construct a new circuit C' that outputs C evaluated on independent inputs with a larger gap.

The Construction of MCRH

Let S be a hard-instance sampler for $\mathsf{EA}^{(\lfloor\sqrt{n}\rfloor)}_{\min,\max}$.

$\mathsf{Gen}(1^n)$:

1. Sample $(1^n, C_n, k) \leftarrow \mathsf{S}(1^n)$, where C_n maps $\{0,1\}^n \to \{0,1\}^{n'}$.
2. Sample[a] $f_1 \leftarrow \mathcal{F}^{3n}_{n,(n-k)}$ and $f_2 \leftarrow \mathcal{F}^{2n}_{(n'+n-k),(n-\lfloor\sqrt{n}\rfloor)}$.
3. Output the circuit that computes the function $h_{C_n,f_1,f_2} : \{0,1\}^n \to \{0,1\}^{n-\lfloor\sqrt{n}\rfloor}$ that is defined as follows:

$$h_{C_n,f_1,f_2}(x) \coloneqq f_2(C_n(x), f_1(x)).$$

[a] Recall that $\mathcal{F}^{\ell}_{n,m} = \{f : \{0,1\}^n \to \{0,1\}^m\}$ is a family of ℓ-wise independent hash functions.

Figure 1: Construction of MCRH from Entropy Approximation.

Proposition 3.5. *$\mathsf{EA}^{(\lfloor\sqrt{n}\rfloor)}_{\min,\max}$ is average-case hard if and only if $\mathsf{EA}_{\min,\max}$ is average-case hard.*

Proof Sketch. Any YES instance of $\mathsf{EA}^{(\lfloor\sqrt{n}\rfloor)}_{\min,\max}$ is itself a YES instance of $\mathsf{EA}_{\min,\max}$, and the same holds for NO instances. So the average-case hardness of $\mathsf{EA}^{(\lfloor\sqrt{n}\rfloor)}_{\min,\max}$ immediately implies that of $\mathsf{EA}_{\min,\max}$, with the same hard-instance sampler. In order to show the implication in the other direction, we show how to use a hard-instance sampler for $\mathsf{EA}_{\min,\max}$ to construct a hard-instance sampler S' for $\mathsf{EA}^{(\lfloor\sqrt{n}\rfloor)}_{\min,\max}$.

S' on input (1^n):

1. Let $\ell = \lfloor\sqrt{n}\rfloor$. S' samples $(1^\ell, C_\ell, k) \leftarrow \mathsf{S}(1^\ell)$.
2. Let $\widehat{C}_n$ be the following circuit that takes an n-bit input x. It breaks x into $\ell+1$ disjoint blocks $x_1, \ldots, x_{\ell+1}$, where $x_1, \ldots, x_\ell$ are of size ℓ, and $x_{\ell+1}$ is whatever remains. It ignores $x_{\ell+1}$, runs a copy of C_ℓ on each of the other x_i's, and outputs a concatenation of all the outputs.
3. S' outputs $(1^n, \widehat{C}_n, k \cdot \ell)$.

As $\widehat{C}_n$ is the ℓ-fold repetition of C_ℓ, all its entropies are ℓ times the respective entropies of C_ℓ. So if C_ℓ had min-entropy at least k, then $\widehat{C}_n$ has min-entropy at least $k \cdot \ell$, and if C_ℓ had max-entropy at most $(k-1)$, then $\widehat{C}_n$ has max-entropy at most $(k-1)\cdot\ell = k\cdot\ell - \ell$, where $\ell = \lfloor\sqrt{n}\rfloor$. The proposition follows. □

3.2 The Construction

Our construction of a Multi-Collision Resistant Hash (MCRH) family is presented in Figure 1. For most of this section, to avoid cluttering up notations, we will denote the problem $\mathsf{EA}^{(\lfloor\sqrt{n}\rfloor)}_{\min,\max}$ by just EA. We now prove that the construction is secure under our average-case hardness assumption.

Theorem 3.6. *If* $\mathsf{EA}_{\text{min,max}}^{(\lfloor\sqrt{n}\rfloor)}$ *is average-case hard, then the construction in Figure 1 is an* (s,t)-MCRH*, where* $s = 6n^2$ *and* $t = \lfloor\sqrt{n}\rfloor$*.*

The above theorem, along with Proposition 3.5, now implies the following.

Corollary 3.7. *If* $\mathsf{EA}_{\text{min,max}}$ *is average-case hard, then there exists an* (s,t)-MCRH*, where* $s = 6n^2$ *and* $t = \lfloor\sqrt{n}\rfloor$*.*

Note that above, the shrinkage being $\lfloor\sqrt{n}\rfloor$ guarantees that there exist $2^{\lfloor\sqrt{n}\rfloor}$-way collisions. But the construction is such that it is not possible to find even a $6n^2$-way collision, (which is sub-exponentially smaller). This is significant because, unlike in the case of standard collision-resistant hash functions (i.e., in which it is hard to find a pair of collisions), shrinkage in MCRHs cannot be easily amplified by composition while maintaining the same amount of collision-resistance (see Remark 1.2).

The rest of this section is dedicated to proving Theorem 3.6.

Proof of Theorem 3.6. Let Gen denote the algorithm described in Figure 1, and S be the hard-instance sampler used there. Fact 2.3, along with the fact that S runs in polynomial-time ensures that Gen runs in polynomial-time as well. The shrinkage requirement of an MCRH is satisfied because here the shrinkage is $s(n) = \lfloor\sqrt{n}\rfloor$. To demonstrate multi-collision resistance, we show how to use an adversary that finds $6n^2$ collisions in hash functions sampled by Gen to break the average-case hardness of $\mathsf{EA}_{\text{min,max}}^{(\lfloor\sqrt{n}\rfloor)}$.

We begin with an informal discussion of the proof. We first prove that large sets of collisions that exist in a hash function output by Gen have different properties depending on whether the instance that was sampled in step 1 of Gen was a YES or NO instance of EA. Specifically, notice that the hash functions that are output by Gen have the form $h_{C_n,f_1,f_2}(x) = f_2(C_n(x), f_1(x))$; we show that, except with negligible probability:

- In functions h_{C_n,f_1,f_2} generated from $(1^n, C_n, k) \in \mathsf{EA}$, there do not exist $3n$ distinct inputs $x_1, \ldots, x_{3n}$ such that they all have the same value of $(C_n(x_i), f_1(x_i))$.
- In functions h_{C_n,f_1,f_2} generated from $(1^n, C_n, k) \notin \mathsf{EA}$, there do not exist $2n$ distinct inputs $x_1, \ldots, x_{2n}$ such that they all have distinct values of $(C_n(x_i), f_1(x_i))$, but all have the same value $f_2(C_n(x_i), f_1(x_i))$.

Note that in any set of $6n^2$ collisions for h_{C_n,f_1,f_2}, there has to be either a set of $3n$ collisions for (C_n, f_1) or a set of $2n$ collisions for f_2, and so at least one of the conclusions in the above two statements is violated.

A candidate average-case solver for EA, when given an instance $(1^n, C_n, k)$, runs steps 2 and 3 of the algorithm Gen from Figure 1 with this C_n and k. It then runs the collision-finding adversary on the hash function h_{C_n,f_1,f_2} that is thus produced. If the adversary does not return $6n^2$ collisions, it outputs a random answer. But if these many collisions are returned, it checks which of the conclusions above is violated, and thus knows whether it started with a YES or NO instance. So whenever the adversary succeeds in finding collisions, the distinguisher can decide EA correctly with overwhelming probability. So if the collision-finding adversary works with non-negligible probability, then the distinguisher also has non-negligible advantage, contradicting the average-case hardness of EA, which was assumed.

We now state and prove the above claims about the properties of sets of collisions, then formally write down the adversary outlined above and prove that it works. The first claim is that for hash functions h_{C_n,f_1,f_2} generated according to Gen using a YES instance, there is no set of $3n$ distinct x_i's that all have the same value for $C_n(x_i)$ and $f_1(x_i)$, except with negligible probability.

Claim 3.7.1. *Let* $(1^n, C_n, k)$ *be a* YES *instance of* EA*. Then,*

$$\Pr_{f_1 \leftarrow \mathcal{F}^{3n}_{n,(n-k)}}\left[\exists y, y_1 : \left|C_n^{-1}(y) \cap f_1^{-1}(y_1)\right| \geq 3n\right] \leq \mathsf{negl}(n)$$

Intuitively, the reason this should be true is that when C_n comes from a YES instance, it has high min-entropy. This means that for any y, $C_n^{-1}(y)$ will be quite small. f_1 can now be thought of as partitioning each set $C_n^{-1}(y)$ into several parts, none of which will be too large because of the load-balancing properties of many-wise independent hash functions.

Proof. The above probability can be bounded using the union bound as follows:

$$\Pr_{f_1}\left[\exists y, y_1 : \left|C_n^{-1}(y) \cap f_1^{-1}(y_1)\right| \geq 3n\right] \leq \sum_{y \in \mathrm{Im}(C_n)} \Pr_{f_1}\left[\exists y_1 : \left|C_n^{-1}(y) \cap f_1^{-1}(y_1)\right| \geq 3n\right].$$

The fact that $(1^n, C_n, k)$ is a YES instance of EA means that $\mathrm{H}_{\min}(C_n) \geq k$. The definition of min-entropy now implies that for any $y \in \mathrm{Im}(C_n)$:

$$\log\left(\frac{1}{\Pr_{x \leftarrow \{0,1\}^n}[C_n(x) = y]}\right) \geq k.$$

This implies that for any y, $\left|C_n^{-1}(y)\right| \leq 2^{n-k}$. Fact 2.4 (about the load-balancing properties of $\mathcal{F}^{3n}_{n,(n-k)}$) now implies that for any $y \in \mathrm{Im}(C_n)$:

$$\Pr_{f_1}\left[\exists y_1 : \left|C_n^{-1}(y) \cap f_1^{-1}(y_1)\right| \geq 3n\right] \leq \frac{2^{n-k}}{2^{3n}} \leq \frac{1}{2^{2n}}.$$

Putting this back into the union bound and noting that the image of C_n has at most 2^n elements, we get the bound we want:

$$\Pr_{f_1}\left[\exists y, y_1 : \left|C_n^{-1}(y) \cap f_1^{-1}(y_1)\right| \geq 3n\right] \leq 2^n \cdot \frac{1}{2^{2n}} \leq \mathsf{negl}(n).$$

□

The next claim is that for hash functions h_{C_n,f_1,f_2} generated according to Gen using a NO instance, there is no set of $2n$ values of x_i that all have distinct values of $(C_n(x_i), f_1(x_i))$, but the same value $f_2(C_n(x_i), f_1(x_i))$, except with negligible probability.

Claim 3.7.2. *Let* $(1^n, C_n, k)$ *be a* NO *instance of* EA*. Then,*

$$\Pr_{\substack{f_1 \leftarrow \mathcal{F}^{3n}_{n,(n-k)} \\ f_2 \leftarrow \mathcal{F}^{2n}_{(n'+n-k),(n-\lfloor\sqrt{n}\rfloor)}}}\left[\exists x_1, \ldots, x_{2n} : \begin{array}{c} \textit{For all } i \neq j,\ (C_n(x_i), f_1(x_i)) \neq (C_n(x_j), f_1(x_j)),\ \textit{and} \\ f_2(C_n(x_i), f_1(x_i)) = f_2(C_n(x_j), f_1(x_j)) \end{array}\right] \leq \mathsf{negl}(n)$$

Proof. The fact that $(1^n, C_n, k)$ is a NO instance of EA means that $\mathrm{H}_{\max}(C_n) \leq k - \lfloor\sqrt{n}\rfloor$; that is, C_n has a small range:

$$|\mathrm{Im}(C_n)| \leq 2^{k-\lfloor\sqrt{n}\rfloor}.$$

For any $f_1 \in \mathcal{F}^{3n}_{n,(n-k)}$, which is what is sampled by Gen when this instance is used, the range of f_1 is a subset of $\{0,1\}^{n-k}$. This implies that even together, C_n and f_1 have a range whose size is bounded as:

$$|\mathrm{Im}(C_n, f_1)| \leq 2^{k-\lfloor\sqrt{n}\rfloor} \cdot 2^{n-k} = 2^{n-\lfloor\sqrt{n}\rfloor}.$$

For there to exist a set of $2n$ inputs x_i that all have distinct values for $(C_n(x_i), f_1(x_i))$ but the same value for $f_2(C_n(x_i), f_1(x_i))$, there has to be a y that has more than $2n$ inverses under f_2 that are all in the image of (C_n, f_1). As f_2 comes from $\mathcal{F}^{2n}_{(n'+n-k),(n-\lfloor\sqrt{n}\rfloor)}$, we can use Fact 2.4 along with the above bound on the size of the image of (C_n, f_1) to bound the probability that such a y exists as follows:

$$\Pr_{f_2}\Big[\exists y : \Big|f_2^{-1}(y) \cap \mathrm{Im}(C_n, f_1)\Big| \geq 2n\Big] \leq \frac{2^{n-\lfloor\sqrt{n}\rfloor}}{2^{2n}} \leq \mathsf{negl}(n).$$

□

Let $\mathsf{A} = (\mathsf{A}_n)_{n\in\mathbb{N}}$ be a polynomial-size family of circuits that given a hash function output by $\mathsf{Gen}(1^n)$ finds $6n^2$ collisions in it with non-negligible probability. The candidate circuit family $\mathsf{A}' = (\mathsf{A}'_n)_{n\in\mathbb{N}}$ for solving EA on average is described below.

A'_n on input $(1^n, C_n, k)$:

1. Run steps 2 and 3 of the algorithm Gen in Figure 1 with $(1^n, C_n, k)$ in place of the instance sampled from S there. This results in the description of a hash function h_{C_n,f_1,f_2}.
2. Run $\mathsf{A}_n(h_{C_n,f_1,f_2})$ to get a set of purported collisions $\mathcal{S}$.
3. If $\mathcal{S}$ does not actually contain $6n^2$ collisions under h_{C_n,f_1,f_2}, output a random bit.
4. If $\mathcal{S}$ contains $3n$ distinct x_i's such that they all have the same value of $(C_n(x_i), f_1(x_i))$, output 0.
5. If $\mathcal{S}$ contains $2n$ distinct x_i's such that they all have distinct values of $(C_n(x_i), f_1(x_i))$ but the same value $f_2(C_n(x_i), f_1(x_i))$, output 1.

The following claim now states that any collision-finding adversary for the MCRH constructed can be used to break the average-case hardness of EA, thus completing the proof.

Claim 3.7.3. *If* A *finds* $6n^2$ *collisions in hash functions output by* $\mathsf{Gen}(1^n)$ *with non-negligible probability, then* A' *has non-negligible advantage in deciding* EA *with respect to the hard-instance sampler* S *used in* Gen.

Proof. On input $(1^n, C_n, k)$, say A'_n computes h_{C_n,f_1,f_2} and runs A_n on it. If A_n does not find $6n^2$ collisions for h_{C_n,f_1,f_2}, then A'_n guesses at random and is correct in its output with probability $1/2$. If A_n does find $6n^2$ collisions, then A'_n is correct whenever one of the following is true:

1. $(1^n, C_n, k)$ is a YES instance and there is no set of $3n$ collisions for (C_n, f_1).
2. $(1^n, C_n, k)$ is a NO instance and there is no set of $2n$ collisions for f_2 in the image of (C_n, f_1).

Note that inputs to A'_n are drawn from $\mathsf{S}(1^n)$, and so the distribution over h_{C_n,f_1,f_2} produced by A'_n is the same as that produced by $\mathsf{Gen}(1^n)$ itself. With such samples, let E_1 denote the event of (C_n, f_1) having a set of $3n$ collisions from $\mathcal{S}$ (the set output by A_n), and let E_2 denote the event of f_2 having a set of $2n$ collisions in the image of (C_n, f_1) from $\mathcal{S}$. Also, let E_Y denote the event of the input to A'_n being a YES instance, E_N that of it being a NO instance, and E_{A} the event that $\mathcal{S}$ contains at least $6n^2$ collisions.

Following the statements above, the probability that A'_n is *wrong* in deciding EA with respect to $(1^n, C_n, k) \leftarrow \mathsf{S}(1^n)$ can be upper-bounded as:

$$\begin{aligned}\Pr\left[\mathsf{A}'_n(1^n, C_n, k) \text{ is wrong}\right] &= \Pr\left[(\neg E_{\mathsf{A}}) \wedge (\mathsf{A}'_n \text{ is wrong})\right] + \Pr\left[E_{\mathsf{A}} \wedge (\mathsf{A}'_n \text{ is wrong})\right] \\ &\leq \Pr[\neg E_{\mathsf{A}}] \cdot \frac{1}{2} + \Pr[(E_Y \wedge E_1) \vee (E_N \wedge E_2)].\end{aligned}$$

The first term comes from the fact that if A_n doesn't find enough collisions, A'_n guesses at random. The second term comes from the fact that if both $(E_Y \wedge E_1)$ and $(E_N \wedge E_2)$ are false and E_{A} is true, then since at least one of E_Y and E_N is always true, one of $(E_Y \wedge \neg E_1)$ and $(E_N \wedge \neg E_2)$ will also be true, either of which would ensure that A'_n is correct, as noted earlier.

We now bound the second term above, starting as follows:

$$\begin{aligned}\Pr[(E_Y \wedge E_1) \vee (E_N \wedge E_2)] &\leq \Pr[(E_Y \wedge E_1)] + \Pr[(E_N \wedge E_2)] \\ &= \Pr[E_Y] \Pr[E_1 | E_Y] + \Pr[E_N] \Pr[E_2 | E_N] \\ &\leq \Pr[E_Y] \cdot \mathsf{negl}(n) + \Pr[E_N] \cdot \mathsf{negl}(n) = \mathsf{negl}(n),\end{aligned}$$

where the first inequality follows from the union bound and the last inequality follows from Claims 3.7.1 and 3.7.2.

Putting this back in the earlier expression,

$$\begin{aligned}\Pr\left[\mathsf{A}'_n(1^n, C_n, k) \text{ is wrong}\right] &\leq \Pr[\neg E_{\mathsf{A}}] \cdot \frac{1}{2} + \mathsf{negl}(n) \\ &= \frac{1}{2} - \frac{\Pr[E_{\mathsf{A}}]}{2} + \mathsf{negl}(n).\end{aligned}$$

In other words,

$$\Pr\left[\mathsf{A}'_n(1^n, C_n, k) \text{ is correct}\right] \geq \frac{1}{2} + \frac{\Pr[E_{\mathsf{A}}]}{2} - \mathsf{negl}(n).$$

So if A succeeds with non-negligible probability in finding $6n^2$ collisions, then A' had non-negligible advantage in deciding EA over S. □

□

4 Constant-Round Statistically-Hiding Commitments

In this section we show that multi-collision-resistance hash functions imply the existence of *constant-round* statistically-hiding commitments.

Definition 4.1 (Commitment Scheme)**.** *A* commitment scheme *is an interactive protocol between two polynomial-time parties — the* sender S *and the* receiver R *— that satisfies the following properties.*

1. *The protocol proceeds in two stages: the* commit *stage and the* reveal *stage.*
2. *At the start of the commit stage both parties get a security parameter 1^n as a common input and the sender S also gets a private input $b \in \{0,1\}$. At the end of the commit stage the parties have a shared output c, which is called the* commitment*, and the sender S has an additional private output d, which is called the* decommitment*.*
3. *In the reveal stage, the sender S sends (b,d) to the receiver R. The receiver R accepts or rejects based on c, d and b. If both parties follow the protocol, then the receiver R always accepts.*

In this section we focus on commitment schemes that are *statistically*-hiding and *computationally*-binding.

Definition 4.2 (Statistically Hiding Commitment)**.** *A commitment scheme (S,R) is* statistically-hiding *if for every cheating receiver R^* it holds that*

$$\mathrm{SD}((\mathsf{S}(0),\mathsf{R}^*)(1^n),(\mathsf{S}(1),\mathsf{R}^*)(1^n)) = \mathsf{negl}(n),$$

where $(\mathsf{S}(b),\mathsf{R}^*)(1^n)$ denotes the transcript of the interaction between R^* and $\mathsf{S}(b)$ in the commit stage.

Definition 4.3 (Computationally Binding Commitment)**.** *A commitment scheme (S,R) is* computationally-binding *if for every family of polynomial-size circuits sender $\mathsf{S}^* = (\mathsf{S}^*_n)_{n\in\mathbb{N}}$ it holds that S^* wins in the following game with only with $\mathsf{negl}(n)$ probability:*

1. *The cheating sender S^*_n interacts with the honest receiver $\mathsf{R}(1^n)$ in the commit stage obtaining a commitment c.*
2. *Then, S^*_n outputs two pairs $(0,d_0)$ and $(1,d_1)$. The cheating sender S^* wins if the honest receiver R accepts both $(c,0,d_0)$ and $(c,1,d_1)$.*

We are now ready to state the main result of this section. A *round* of a commitment scheme is a pair of messages, the first sent from the receiver to the sender, and the second the other way.

Theorem 4.4 ($\mathsf{MCRH} \implies$ Constant-Round Statistically-Hiding Commitments)**.** *Let $t = t(n) \in \mathbb{N}$ be a polynomial computable in $\mathsf{poly}(n)$ time. Assume that there exists a (s,t)-MCRH for $s \geq \log(t)$, then there exists a* three-round *statistically-hiding computationally-binding commitment scheme.*

As we already mentioned in Section 1, constructions of statistically-hiding computationally-binding commitment schemes are known assuming only the minimal assumption that one-way functions exist. Those constructions, however, have polynomial number of rounds (and this in inherit for black-box constructions). Our construction, on the other hand, has only three rounds and is simple to describe.

The rest of this section is dedicated to proving Theorem 4.4.

The Commitment Scheme (S,R)

S's Input: security parameter 1^n and a bit $b \in \{0,1\}$.
R's Input: security parameter 1^n.

Algorithm Gen: polynomial-time algorithm that on input 1^n returns a circuit computing a $(\lceil \log(t(n)) \rceil, t(n))$-$\mathsf{MCRH}$ $h\colon \{0,1\}^n \to \{0,1\}^{n-\lceil \log(t(n)) \rceil}$

The commit stage:

1. Both parties set $t = t(n)$ and $k = n \cdot t$.
2. R samples $h \leftarrow \mathsf{Gen}(1^n)$ and sends h to S.
3. S samples $\mathbf{x} = (x_1, \ldots, x_k) \leftarrow \{0,1\}^{n \cdot k}$, computes $\mathbf{y} = (y_1, \ldots, y_k)$ for $y_i = h(x_i)$, and sends $\mathbf{y}$ to R.
4. R samples[a] $f \leftarrow \mathcal{F}_{n \cdot k, \lceil k \cdot \log(t-1) \rceil}^{(\lceil k \cdot \log(t-1) \rceil)}$ and sends f to S.
5. S sends $z_1 = f(\mathbf{x})$ to R.
6. R samples $g \leftarrow \mathcal{F}_{n \cdot k, \lceil 2\log(k) + 2\log\log(t-1) + \log^2(n) \rceil}$ and sends g to S.
7. S sends $z_2 = g(\mathbf{x})$ to R.
8. S samples $r \leftarrow \{0,1\}^{n \cdot k}$ and computes $\sigma = \langle r, \mathbf{x} \rangle \oplus b$ and sends (r, σ) to R.
9. The commitment is defined as $c = (h, \mathbf{y}, f, z_1, g, z_2, r, \sigma)$ and the decommitment is defined as $d = \mathbf{x}$.

The reveal stage:

1. S sends $(b, \mathbf{x})$ to R.
2. R accepts if $h(x_i) = y_i$ for every $i \in [k]$, $f(\mathbf{x}) = z_1$, $g(\mathbf{x}) = z_2$ and $\langle r, \mathbf{x} \rangle \oplus b = \sigma$.

[a] Recall that $\mathcal{F}_{n,m}^{(k)}$ is a family of k-wise-independent hash functions from $\{0,1\}^n$ to $\{0,1\}^m$. See Fact 2.3.

Figure 2: Statistically Hiding Commitment

4.1 Proving Theorem 4.4

The proof follows the outline detailed in Section 1.3.2.

Let Gen be the generating algorithm that defines a (s,t)-MCRH for $s \geq \log(t)$, assumed to exists in the theorem's statement. Since s must be an integer, we can assume without the loss of generality that the function defined by Gen is $(\lceil \log(t) \rceil, t)$-$\mathsf{MCRH}$ (we can always pad the output of the function without making it easier to find collisions). Consider the protocol in Fig. 2. The proof now follows from the next two lemmas.

Lemma 4.5 (Computationally Binding)**.** *The commitment scheme* (S,R) *in Fig. 2 is computationally binding.*

Lemma 4.6 (Statistically Hiding)**.** *The commitment scheme* (S,R) *in Fig. 2 is statistically hiding.*

The proof of Lemma 4.5 is given in Section 4.1.1 and the proof of Lemma 4.6 is given in Section 4.1.2.

$\mathsf{CollFinder}_n$ on input (1^n):

1. Set $t = t(n)$, $k = n \cdot t^2$ and $q = q(n)$
2. Sample at random coins[a] for S^*_n, denoted by τ, and $h \leftarrow \mathsf{Gen}(1^n)$
3. Emulate[b] $\mathsf{S}^*_n(h;\tau)$ to obtain $\mathbf{y} = (y_1, \ldots, y_k)$
4. For $i \in [k]$ set $\mathcal{S}_i = \emptyset$
5. Repeat for $2 \cdot q^3 \cdot n \cdot (t-1) \cdot k$ times:
   (a) Sample $f \leftarrow \mathcal{F}^{(2\cdot\lceil k\cdot\log(t-1)\rceil)}_{n\cdot k,\lceil k\cdot\log(t-1)\rceil}$
   (b) Emulate $\mathsf{S}^*_n(f;\tau)$ to obtain z_1
   (c) Sample $g \leftarrow \mathcal{F}_{n\cdot k,\lceil 2\log(k)+2\log\log(t-1)+\log^2(n)\rceil}$
   (d) Emulate $\mathsf{S}^*_n(g;\tau)$ to obtain r, σ, $\mathbf{x} = (x_1, \ldots, x_k)$ and $\mathbf{x}' = (x'_1, \ldots, x'_k)$
   (e) If S^* *wins*,[c] then for every i update $\mathcal{S}_i = \mathcal{S}_i \cup \{x_i, x'_i\}$
6. Output $\mathcal{S}_{\text{out}} = \mathcal{S}_j$ such that $j = \arg\max_i\{|\mathcal{S}_i|\}$

[a] Since we consider a non-uniform adversary S^*, we could have simply fixed its random coins. However, we avoid doing so to highlight that the reduction is uniform.

[b] Recall that $\mathsf{A}(\cdot;\tau)$ mean that A is run when its coins are set to τ.

[c] Namely, if the conditions in Eq. (1) are satisfied, or equvalently if $(f,g) \in \mathcal{W}_{h,\tau}$.

Figure 3: Algorithm to find t-collision in $\mathcal{H}$

4.1.1 Analyzing Binding — Proving Lemma 4.5

Assume toward a contradiction that the scheme is not computationally binding. That is, there exists a polynomial-size family of circuits $\mathsf{S}^* = (\mathsf{S}^*_n)_{n\in\mathbb{N}}$ of size $q \in \mathsf{poly}(n)$, and an infinite index set $\mathcal{I} \subseteq \mathbb{N}$ such that for every $n \in \mathcal{I}$ the following events occur with probability at least $1/q(n)$: (1) the cheating sender S^*_n interacts with the honest receiver $\mathsf{R}(1^n)$ in the commit stage of the protocol and the parties obtain a commitment $c = (h, \mathbf{y} = (y_1 \ldots, y_k), f, z_1, g, z_2, r, \sigma)$; then (2) S^*_n outputs valid decommitments to two distinct values $(0, \mathbf{x} = (x_1, \ldots, x_k))$ and $(1, \mathbf{x}' = (x'_1, \ldots, x'_k))$ such that

$$\begin{aligned} \forall i \in [k]\colon \quad & h(x_i) = h(x'_i) = y_i \\ f(\mathbf{x}) = f(\mathbf{x}') = z_1 \quad \text{and} \quad & g(\mathbf{x}) = g(\mathbf{x}') = z_2 \\ \langle r, \mathbf{x}\rangle \oplus 0 = \sigma \quad \text{and} \quad & \langle r, \mathbf{x}'\rangle \oplus 1 = \sigma. \end{aligned} \tag{1}$$

Whenever the above conditions are met, we say that S^* *wins*. We use S^* to find t-collisions in $\mathcal{H}$ and thus deriving a contradiction.

Let τ be some random coins used by S^* during its interaction with R. Observe that $\mathbf{y}$, the first message sent by S^*, is a deterministic function of τ and h, where the latter is the first message sent by R. Similarly, z_1, the second message sent by S^*, is a deterministic function of τ, h and f, where the latter is the second message sent by R. Finally, r, σ, $\mathbf{x}$ and $\mathbf{x}'$, the values sent in the third message of S^*, are all deterministic functions of τ, h, f and g, where the latter is the third message sent by R. Hence, for τ and h we can define

$$\mathcal{W}_{\tau,h} = \{(f,g)\colon (h, \mathbf{y}, f, g, r, \sigma, \mathbf{x}, \mathbf{x}') \text{ satisfy the condition in Eq. (1)}\}.$$

We can now describe an algorithm to find t-way collision in $\mathcal{H}$. Consider the (non-uniform) algorithm $\mathsf{CollFinder} = (\mathsf{CollFinder}_n)_{n\in\mathbb{N}}$ given in Fig. 3. It is easy to verify that $\mathsf{CollFinder}_n$ is of

polynomial size.[16] In the rest of the proof we show that CollFinder finds t-way collision in $\mathcal{H}$ with probability roughly $1/q(n)$.

Intuitively, the sets $\mathcal{S}_i$ store collisions of h. The choice of f and g guarantee that, with probability at least $1/\mathsf{poly}(n)$ and as long as $|\mathcal{S}_i| < t$ for every i, in every iteration the main loop of CollFinder in which S^* wins, at least one of the $\mathcal{S}_i$ increases. Iterating the loop for sufficiently many times guarantee that with probability at least $1/\mathsf{poly}(n)$, one of the $\mathcal{S}_i$'s contain at least t values at the end of the loop.

Formally, fix some large enough $n \in \mathcal{I}$ and remove it from notation. Observe that $\mathcal{S}_{\text{out}}$, the set of alleged collisions returned by CollFinder, is updated only when S^* wins. Thus, it holds that $h(x) = h(x')$ for every $x, x' \in \mathcal{S}_{\text{out}}$. Let L be a random variable induced by cardinality of $\mathcal{S}_{\text{out}}$ in a random execution of CollFinder. Hence, out goal is to show that

$$\Pr[L \geq t] \geq \Omega\left(\frac{1}{q}\right). \tag{2}$$

Our first step is to analyze how the choice of (τ, h) affects the success probability of CollFinder. Let (T, H, F, G) be (jointly distributed) random variables induced by the values of (τ, h, f, g) in a random execution of $(\mathsf{S}^*(b), \mathsf{R})$ where τ are the random coins used by S^* and $b \in \{0,1\}$.[17] Note that (T, H) are both independent of (F, G) and distributed as the values of (τ, h) in Step 2 in a random execution of CollFinder. Let

$$\mathcal{W} = \left\{(\tau, h) \in \text{Supp}(T, H) \colon \Pr[(F, G) \in \mathcal{W}_{\tau,h}] \geq \frac{1}{q^2}\right\}.$$

We make use of the next claim.

Claim 4.6.1. *It holds that* $\Pr[(T, H) \in \mathcal{W}] \geq 1/(q+1)$.

Proof. Let $P_{\tau,h} = \Pr[(F, G) \in \mathcal{W}_{\tau,h}]$. It follows that,

$$\frac{1}{q} \leq \Pr[\mathsf{S}^* \textit{ wins}] = \Pr[(F, G) \in \mathcal{W}_{T,H}] = \mathrm{E}[P_{T,H}].$$

Fact 2.1 now yields that,

$$\Pr[(T, H) \in \mathcal{W}] = \Pr\left[P_{T,H} \geq \frac{1}{q^2}\right] \geq \frac{1}{q+1}.$$

□

For τ, h, let $L_{\tau,h}$ denote the random variable distributed as L conditioned on $(T, H) = (\tau, h)$. Using Claim 4.6.1, we have that

$$\begin{aligned} \Pr[L \geq t] &= \mathop{\mathrm{E}}_{(\tau,h)\leftarrow(T,H)}\Big[\Pr[L_{\tau,h} \geq t]\Big] \\ &\geq \Pr[(T, H) \in \mathcal{W}] \cdot \mathop{\mathrm{E}}_{(\tau,h)\leftarrow(T,H)}\Big[\Pr[L_{\tau,h} \geq t \mid (\tau, h) \in \mathcal{W}]\Big] \\ &\geq \frac{1}{q+1} \cdot \mathop{\mathrm{E}}_{(\tau,h)\leftarrow(T,H)}\Big[\Pr[L_{\tau,h} \geq t \mid (\tau, h) \in \mathcal{W}]\Big]. \end{aligned} \tag{3}$$

[16] We assume, without the loss of generality, that $q(n)$ is efficiently computed (otherwise, take $q' > q$ that is efficiently computed)

[17] Those random variables are identically distributed if $b = 0$ or $b = 1$.

In the rest of the proof we show that for every $(\tau, h) \in \mathcal{W}$, it holds that

$$\Pr[L_{\tau,h} \geq t] \geq 1 - \mathsf{negl}(n). \tag{4}$$

Fix $(\tau, h) \in \mathcal{W}$ and let $S_1, \ldots, S_i$ be random variables induced by the values of $\mathcal{S}_1, \ldots, \mathcal{S}_k$ at the end of a random execution of $\mathsf{CollFinder}$, conditioned on $(T, H) = (\tau, h)$. It holds that

$$\begin{aligned}\Pr[L_{\tau,h} < t] &= \Pr[|S_1| \leq t-1 \wedge \cdots \wedge |S_k| \leq t-1] \\ &= \Pr[\exists \mathcal{S}_1, \ldots, \mathcal{S}_k \colon \forall i \ |\mathcal{S}_i| \leq t-1 \wedge S_i = \mathcal{S}_i] \\ &\leq \sum_{\substack{\mathcal{S}_1,\ldots,\mathcal{S}_k \\ \forall i \colon |\mathcal{S}_i| \leq t-1}} \Pr[S_1 = \mathcal{S}_1 \wedge \cdots \wedge S_k = \mathcal{S}_k],\end{aligned} \tag{5}$$

where the last inequality follows from the union bound.

Fix $\mathcal{S}_1, \ldots, \mathcal{S}_k$ with $|\mathcal{S}_i| \leq t-1$ for every $i \in [k]$. For f, z_1, let $\mathcal{S}_{f,z_1} = \{\mathbf{x} \in \mathcal{S}_1 \times \cdots \times \mathcal{S}_k \colon f(\mathbf{x}) = z_1\}$. The crux of the proof is the next two claims.

Claim 4.6.2. *If $(h, \mathbf{y}, f, g, r, \sigma, \mathbf{x}, \mathbf{x}')$ satisfy the condition in Eq. (1) and g is injective over $\mathcal{S}_{f,z_1}$, then there exists $i \in [k]$ such that $x_i \notin \mathcal{S}_i$ or $x'_i \notin \mathcal{S}_i$.*

Proof. First, note that since the conditions in Eq. (1) are satisfied, it holds that $\langle r, r \rangle \oplus 0 = \sigma$ and $\langle r, \mathbf{x}' \rangle \oplus 1 = \sigma$, and thus $\mathbf{x} \neq \mathbf{x}'$.

Assume toward a contradiction that $x_i \in \mathcal{S}_i$ and $x'_i \in \mathcal{S}_i$ for every i. It follows that $\mathbf{x}, \mathbf{x}' \in \mathcal{S}_1 \times \cdots \times \mathcal{S}_k$. Since the conditions in Eq. (1) are satisfied it holds that $f(\mathbf{x}) = f(\mathbf{x}') = z_1$, and thus $\mathbf{x}, \mathbf{x}' \in \mathcal{S}_{f,z_1}$. Using once more that the conditions in Eq. (1) are satisfied it holds that $g(\mathbf{x}) = g(\mathbf{x}')$, a contradiction to the assumption that g is injective over $\mathcal{S}_{f,z_1}$. □

Recall that z_1 is a deterministic function of τ, h and f, and since τ and h are fixed, z_1 is a deterministic function of f, i.e., $z_1 = z_1(f)$. For (f, g), let $\mathsf{inj}(f, g) = 1$ if g is injective over $\mathcal{S}_{f,z_1}$.

Claim 4.6.3. *It holds that*

$$\Pr[\mathsf{inj}(F, G) = 1] \geq 1 - \mathsf{negl}(n).$$

Proof. For $f \in \mathrm{Supp}(F)$, let $B_f = \left|S_{f,z_1(f)}\right|$. First, we show that with high probability B_F is small. Indeed, since $|\mathcal{S}_i| \leq (t-1)$ for every $i \in [k]$, it follows that $|\mathcal{S}_1 \times \cdots \times \mathcal{S}_k| \leq (t-1)^k$. Moreover, since F is chosen from the family $\mathcal{F}^{(2 \cdot \lceil k \cdot \log(t-1) \rceil)}_{n \cdot k, \lceil k \cdot \log(t-1) \rceil}$, Fact 2.4 (about the load balancing of many-wise independent functions) yields that

$$\begin{aligned}\Pr[B_F \geq 2 \cdot \lceil k \cdot \log(t-1) \rceil] &\leq \Pr[\exists z \colon |\mathcal{S}_{F,z}| \geq 2 \cdot \lceil k \cdot \log(t-1) \rceil] \\ &\leq 2^{-2 \cdot \lceil k \cdot \log(t-1) \rceil + \lceil k \cdot \log(t-1) \rceil} = 2^{-\lceil k \cdot \log(t-1) \rceil} = \mathsf{negl}(n),\end{aligned}$$

where the last equality follows since $k = n \cdot t$.

Second, we show that if B_F is small, then $\mathsf{inj}(F, G) = 1$ with high probability. Recall that in the protocol $(\mathsf{S}^*, \mathsf{R})$, the function G is chosen (independently from everything else) from the family $\mathcal{F}_{n \cdot k, \lceil 2\log(k) + 2\log\log(t-1) + \log^2(n) \rceil}$ *after* $z_1(F)$ (and $\mathcal{S}_{F,z_1(F)}$) is determined. Thus, we can use the

pairwise independence of G to complete the proof. Fix $f \in \mathrm{Supp}(F)$ with $B_f < 2 \cdot \lceil k \cdot \log(t-1) \rceil$. It holds that

$$\begin{aligned}\Pr[\mathsf{inj}(F,G) = 0 \mid F = f] &= \Pr\Big[\exists x \neq x' \in \mathcal{S}_{f,z_1(f)} \colon G(x) = G(x')\Big] \\ &\leq \sum_{x \neq x' \in \mathcal{S}_{f,z_1(f)}} \Pr[G(x) = G(x')] \\ &\leq \frac{(2(k \cdot \log(t-1)+1))^2}{(k^2 \cdot \log^2(t-1) \cdot 2^{\log^2(n)})} = \mathsf{negl}(n).\end{aligned}$$

Finally, combining the above we have that

$$\begin{aligned}&\Pr[\mathsf{inj}(F,G) = 1] \\ &\geq \Pr[B_F < 2 \cdot \lceil k \cdot \log(t-1) \rceil] \cdot \Pr[\mathsf{inj}(F,G) = 1 | B_F < 2 \cdot \lceil k \cdot \log(t-1) \rceil] \\ &\geq (1 - \mathsf{negl}(n)) \cdot (1 - \mathsf{negl}(n)) = 1 - \mathsf{negl}(n).\end{aligned}$$

□

Using the above claims, we can proceed to complete the proof. For $j \in [2 \cdot q^3 \cdot n \cdot (t-1) \cdot k]$ let $F^{(j)}$ and $G^{(j)}$ be random variables induced by the values of f and g in j'th iteration of the loop in Step 5 in a random execution of $\mathsf{CollFinder}$, conditioned that $(T, H) = (\tau, h)$. Claim 4.6.2 implies that if $\mathsf{inj}(F^{(j^*)}, G^{(j^*)}) = 1$ and $(F^{(j^*)}, G^{(j^*)}) \in \mathcal{W}_{\tau,h}$ for some j^*, then the random variables $S_1, \ldots, S_k$ cannot take the values of the sets $\mathcal{S}_1, \ldots, \mathcal{S}_k$. It follows that

$$\Pr[S_1 = \mathcal{S}_1 \wedge \cdots \wedge S_k = \mathcal{S}_k] \leq \Pr\Big[\forall j \in [2 \cdot q^3 \cdot n \cdot (t-1) \cdot k],\ \mathsf{inj}(F^{(j)}, G^{(j)}) = 0 \vee (F^{(j)}, G^{(j)}) \notin \mathcal{W}_{\tau,h}\Big] \tag{6}$$

$$\begin{aligned}&= \prod_{j=1}^{2 \cdot q^3 \cdot n \cdot (t-1) \cdot k} \Pr\Big[\mathsf{inj}(F^{(j)}, G^{(j)}) = 0 \vee (F^{(j)}, G^{(j)}) \notin \mathcal{W}_{\tau,h}\Big] \\ &\leq \prod_{j=1}^{2 \cdot q^3 \cdot n \cdot (t-1) \cdot k} \Big(\Pr\Big[\mathsf{inj}(F^{(j)}, G^{(j)}) = 0\Big] + \Pr\Big[(F^{(j)}, G^{(j)}) \notin \mathcal{W}_{\tau,h}\Big]\Big) \\ &\leq \prod_{j=1}^{2 \cdot q^3 \cdot n \cdot (t-1) \cdot k} \left(\mathsf{negl}(n) + 1 - \frac{1}{q^2}\right) \\ &\leq \left(1 - \frac{1}{q^3}\right)^{2 \cdot q^3 \cdot n \cdot (t-1) \cdot k} \\ &\leq e^{-2 \cdot n \cdot (t-1) \cdot k},\end{aligned}$$

where the first equality follows since $(F^{(j)}, G^{(j)})$'s are independent, the third inequality follows from Claim 4.6.3, since $(\tau, h) \in \mathcal{W}$ and since $(F^{(j)}, G^{(j)})$ are identically distributed as (F, G), and the last inequality follows since $\ln(1-x) \leq -x$ for any x.

Finally, we bound the number of different sets $\mathcal{S}_1, \ldots, \mathcal{S}_k$ with $|\mathcal{S}_i| \leq t-1$. For every $\mathcal{S}_i$, there are

$$\sum_{s=1}^{t-1} \binom{2^{n \cdot k}}{s} \leq (2^{n \cdot k} + 1)^{t-1} \leq 2^{2n \cdot (t-1) \cdot k},$$

different possibilities. Hence, there are at most $\left(2^{2n\cdot(t-1)\cdot k}\right)^k = 2^{2n\cdot(t-1)\cdot k^2}$ different possibilities for $\mathcal{S}_1, \ldots, \mathcal{S}_k$.

Plugging Eq. (6) into Eq. (5) yields that

$$\Pr[L_{\tau,h} < t] \leq 2^{2n\cdot(t-1)\cdot k^2} \cdot e^{-2n\cdot(t-1)\cdot k^2} = \mathsf{negl}(n).$$

Hence, Eq. (4) holds, and the proof of Lemma 4.5 is complete.

4.1.2 Analyzing Hiding — Proving Lemma 4.6

The crux of proving that the scheme is statistically-hiding is the following observation: If a function f is shrinking and X is a random input for it, then the random variable $(X \mid f(X))$ has (some form of) conditional min-entropy. Assuming this observation hold, the receiver — who sees only $f(X)$ — cannot completely recover X. The actual notion of entropy we use is that of *average min-entropy*.

Average Min-Entropy. The notion of average min-entropy was defined by Dodis et al. [DORS08] as follows.

Definition 4.7 (Average Min-Entropy [DORS08])**.** *Let X, Y be jointly distributed random variables. The* average min-entropy *of X given Y is defined by*

$$\tilde{\mathrm{H}}_{\min}(X|Y) := -\log\left(\mathop{\mathrm{E}}_{y\leftarrow Y}\left[\max_x \Pr[X = x \mid Y = y]\right]\right).$$

This notion is useful since the Leftover Hash Lemma can be generalized to sources having high average min-entropy.

Definition 4.8 (Universal Hash Function)**.** *A family of functions $\mathcal{F} = \{f : \{0,1\}^n \to \{0,1\}^m\}$ is* Universal *if for every $x_1 \neq x_2 \in \{0,1\}^n$, it holds that*

$$\Pr_{f\leftarrow\mathcal{F}}[f(x_1) = f(x_2)] = \frac{1}{2^m}.$$

Lemma 4.9 (Generalized Leftover Hash Lemma [DORS08, Lemma 2.4])**.** *Let $\mathcal{F} = \{h\colon \{0,1\}^n \to \{0,1\}^m\}$ be a family of universal hash functions. Then, for any random variables X and Y and the random variable $F \leftarrow \mathcal{F}$, it holds that*

$$\mathrm{SD}\Big((F(X), F, Y), (U_m, F, Y)\Big) \leq \frac{1}{2} \cdot \sqrt{2^{-\tilde{\mathrm{H}}_{\min}(X|Y)} \cdot 2^m},$$

where U_m is distributed uniformly over $\{0,1\}^m$.

Finally, we can show that if f is shrinking, then the average min-entropy of $(X \mid f(X))$ is high.

Claim 4.9.1. *Let $f\colon \{0,1\}^n \to \{0,1\}^{n-m}$, let X be a random variable uniformly distributed over $\{0,1\}^n$, and let $Y = f(X)$. Then, $\tilde{\mathrm{H}}_{\min}(X|Y) \geq m$.*

Proof. For $y \in \{0,1\}^{n-m}$, let $f^{-1}(y) = \{x \in \{0,1\}^n\colon f(x) = y\}$. Fix $y \in \mathrm{Im}(f)$. For $x \in f^{-1}(y)$, it holds that $\Pr[X = x|Y = y] = 1/|f^{-1}(y)|$, while for $x \notin f^{-1}(y)$, it holds that $\Pr[X = x|Y = y] = 0$.

Thus, $\max_x \Pr[X = x|Y = y] = 1/|f^{-1}(y)|$. Moreover, it holds that $\Pr[Y = y] = |f^{-1}(y)|/2^n$. Finally, for every $y \notin \mathrm{Im}(f)$, it holds that $\Pr[Y = y] = 0$. Hence,

$$\begin{aligned}
\tilde{\mathrm{H}}_{\min}(X|Y) &= -\log\left(\mathop{\mathrm{E}}_{y\leftarrow Y}\left[\max_x \Pr[X = x|Y = y]\right]\right) \\
&= -\log\left(\sum_{y\in\mathrm{Im}(f)} \frac{|f^{-1}(y)|}{2^n} \cdot \frac{1}{|f^{-1}(y)|}\right) \\
&= \log\left(\frac{2^n}{|\mathrm{Im}(f)|}\right) \\
&\geq \log(2^m) = m.
\end{aligned}$$

□

We are now finally ready to prove that the scheme is statistically-hiding.

Proof of Lemma 4.6. Let R^* be any (possibly unbounded) algorithm. Fix large enough $n \in \mathbb{N}$ and remove it from notation when convenient. Let $(H^*, \mathbf{X} = (X_1, \ldots, X_k), F^*, G^*, r)$ be (jointly distributed) random variables induced by the values of $(h^*, \mathbf{x} = (x_1, \ldots, x_k), f^*, g^*, r)$ in a random execution of $(\mathsf{S}(b), \mathsf{R}^*)$, for an arbitrary $b \in \{0, 1\}$.[18] The transcript of the interaction between $\mathsf{S}(b)$ and R^* for any $b \in \{0, 1\}$ is thus

$$(\mathsf{S}(b), \mathsf{R}^*) \equiv (H^*, H^*(X_1), \ldots, H^*(X_k), F^*, F^*(\mathbf{X}), G^*, G^*(\mathbf{X}), r, \langle r, \mathbf{X}\rangle \oplus b).$$

Note that (H^*, F^*, G^*) can be viewed as a description of a function Q^* mapping $n \cdot k$ bits to $n \cdot k - m$ bits for $m := k \cdot \lceil \log t\rceil - \lceil k \cdot \log(t-1)\rceil - \left\lceil 2\log(k) + 2\log\log(t-1) + \log^2(n)\right\rceil$ bits. Namely, $Q^*(\mathbf{X}) = (H^*(X_1), \ldots, H^*(X_k), F^*(\mathbf{X}), G^*(\mathbf{X}))$. We can thus write

$$(\mathsf{S}(b), \mathsf{R}^*) \equiv (Q^*, Q^*(\mathbf{X}), r, \langle r, \mathbf{X}\rangle \oplus b).$$

Fix $b \in \{0, 1\}$ and let $U \leftarrow \{0, 1\}$ be a uniform bit. It holds that

$$\begin{aligned}
&\mathrm{SD}\Big((Q^*, Q^*(\mathbf{X}), r, \langle r, \mathbf{X}\rangle \oplus b), (Q^*, Q^*(\mathbf{X}), r, U)\Big) \qquad (7)\\
&= \mathop{\mathrm{E}}_{q^*\leftarrow Q^*}\Big[\mathrm{SD}\Big((q^*(\mathbf{X}), r, \langle r, \mathbf{X}\rangle \oplus b), (q^*(\mathbf{X}), r, U \oplus b)\Big)\Big] \\
&\leq \mathop{\mathrm{E}}_{q^*\leftarrow Q^*}\left[\frac{1}{2} \cdot \sqrt{2^{-\tilde{\mathrm{H}}_{\min}(\mathbf{X}|q^*(\mathbf{X}))} \cdot 2}\right] \\
&\leq \frac{1}{2} \cdot 2^{-(m-1)/2},
\end{aligned}$$

where the equality follows since r and $\mathbf{X}$ are independent of Q^*, the first inequality follows from Lemma 4.9 (Generalized Leftover Hash Lemma) and since inner product is a universal hash function, and the second inequality follows from Claim 4.9.1.

[18] Note that these random variables are identically distributed if $b = 0$ or $b = 1$.

Finally, by the setting of parameters and for large enough n it holds that

$$\begin{aligned} m &= k \cdot \lceil \log t \rceil - \lceil k \cdot \log(t-1) \rceil - \left\lceil 2\log(k) + 2\log\log(t-1) + \log^2(n) \right\rceil \qquad (8) \\ &\geq k \cdot \log(t) - k \cdot \log(t-1) - 1 - 2\log(k) - 2\log\log(t-1) - \log^2(n) - 1 \\ &= k \cdot \log\left(1 + \frac{1}{t-1}\right) - 2\log(k) - 2\log\log(t-1) - \log^2(n) - 2 \\ &\geq k \cdot \frac{1}{t-1} - 2\log(k) - 2\log\log(t-1) - \log^2(n) - 2 \\ &\geq (n \cdot t^2) \cdot \frac{1}{t-1} - 2\log(n \cdot t^2) - 2\log\log(t-1) - \log^2(n) - 2 \\ &\geq \log^3(n). \end{aligned}$$

Putting it all together, it holds that

$$\begin{aligned} \mathrm{SD}((\mathsf{S}(0), \mathsf{R}^*), (\mathsf{S}(1), \mathsf{R}^*)) &= \mathrm{SD}\Big((Q^*, Q^*(\mathbf{X}), R, \langle R, \mathbf{X}\rangle \oplus 0), (Q^*, Q^*(\mathbf{X}), R, \langle R, \mathbf{X}\rangle \oplus 1)\Big) \\ &\leq \sum_{b \in \{0,1\}} \mathrm{SD}\Big((Q^*, Q^*(\mathbf{X}), R, \langle R, \mathbf{X}\rangle \oplus b), (Q^*, Q^*(\mathbf{X}), R, U)\Big) \\ &\leq 2^{-(m-1)/2} \\ &\leq 2^{-(\log^3(n)-1)/2} = \mathsf{negl}(n), \end{aligned}$$

where the first inequality follows from the triangle inequality for statistical distance, the second inequality follows from Eq. (7) and the last inequality follows from Eq. (8). □

Acknowledgments

We thank Vinod Vaikuntanathan for helpful discussions and for his support. We also thank Nir Bitansky, Yael Kalai, Ilan Komargodski, Moni Naor, Omer Paneth and Eylon Yogev for helping us provide a good example of a t-way collision. .

References

[ADM+99] Noga Alon, Martin Dietzfelbinger, Peter Bro Miltersen, Erez Petrank, and Gábor Tardos. Linear hash functions. *J. ACM*, 46(5):667–683, 1999.

[AR16] Benny Applebaum and Pavel Raykov. On the relationship between statistical zero-knowledge and statistical randomized encodings. In *Annual Cryptology Conference*, pages 449–477. Springer, 2016.

[AS15] Gilad Asharov and Gil Segev. Limits on the power of indistinguishability obfuscation and functional encryption. In *Symposium on the Foundations of Computer Science*, 2015.

[BPK] Nir Bitansky, Omer Paneth, and Yael Kalai. Private Communication.

[CRSW13] L. Elisa Celis, Omer Reingold, Gil Segev, and Udi Wieder. Balls and bins: Smaller hash families and faster evaluation. *SIAM J. Comput.*, 42(3):1030–1050, 2013.

[DGRV11] Zeev Dvir, Dan Gutfreund, Guy N. Rothblum, and Salil P. Vadhan. On approximating the entropy of polynomial mappings. In Bernard Chazelle, editor, *Innovations in Computer Science - ICS 2010, Tsinghua University, Beijing, China, January 7-9, 2011. Proceedings*, pages 460–475. Tsinghua University Press, 2011.

[DORS08] Yevgeniy Dodis, Rafail Ostrovsky, Leonid Reyzin, and Adam D. Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. *SIAM J. Comput.*, 38(1):97–139, 2008.

[DPP93] Ivan Damgård, Torben P. Pedersen, and Birgit Pfitzmann. On the existence of statistically hiding bit commitment schemes and fail-stop signatures. In Douglas R. Stinson, editor, *Advances in Cryptology - CRYPTO '93, 13th Annual International Cryptology Conference, Santa Barbara, California, USA, August 22-26, 1993, Proceedings*, volume 773 of *Lecture Notes in Computer Science*, pages 250–265. Springer, 1993.

[GG98] Oded Goldreich and Shafi Goldwasser. On the limits of non-approximability of lattice problems. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, pages 1–9. ACM, 1998.

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

[GMR88] Shafi Goldwasser, Silvio Micali, and Ronald L. Rivest. A digital signature scheme secure against adaptive chosen-message attacks. *SIAM J. Comput.*, 17(2):281–308, 1988.

[GSV99] Oded Goldreich, Amit Sahai, and Salil P. Vadhan. Can statistical zero knowledge be made non-interactive? or on the relationship of SZK and NISZK. In Michael J. Wiener, editor, *Advances in Cryptology - CRYPTO '99, 19th Annual International Cryptology Conference, Santa Barbara, California, USA, August 15-19, 1999, Proceedings*, volume 1666 of *Lecture Notes in Computer Science*, pages 467–484. Springer, 1999.

[GT00] Rosario Gennaro and Luca Trevisan. Lower bounds on the efficiency of generic cryptographic constructions. In *41st Annual Symposium on Foundations of Computer Science, FOCS 2000, 12-14 November 2000, Redondo Beach, California, USA*, pages 305–313. IEEE Computer Society, 2000.

[HHRS07] Iftach Haitner, Jonathan J Hoch, Omer Reingold, and Gil Segev. Finding collisions in interactive protocols-a tight lower bound on the round complexity of statistically-hiding commitments. In *Foundations of Computer Science, 2007. FOCS'07. 48th Annual IEEE Symposium on*, pages 669–679. IEEE, 2007.

[HM96] Shai Halevi and Silvio Micali. Practical and provably-secure commitment schemes from collision-free hashing. In Neal Koblitz, editor, *Advances in Cryptology - CRYPTO '96, 16th Annual International Cryptology Conference, Santa Barbara, California, USA, August 18-22, 1996, Proceedings*, volume 1109 of *Lecture Notes in Computer Science*, pages 201–215. Springer, 1996.

[HNO+09] 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 J. Comput.*, 39(3):1153–1218, 2009.

[Jou04] Antoine Joux. Multicollisions in iterated hash functions. application to cascaded constructions. In Matthew K. Franklin, editor, *Advances in Cryptology - CRYPTO 2004, 24th Annual International CryptologyConference, Santa Barbara, California, USA, August 15-19, 2004, Proceedings*, volume 3152 of *Lecture Notes in Computer Science*, pages 306–316. Springer, 2004.

[Kil92] Joe Kilian. A note on efficient zero-knowledge proofs and arguments (extended abstract). In *Proceedings of the 24th Annual ACM Symposium on Theory of Computing, May 4-6, 1992, Victoria, British Columbia, Canada*, pages 723–732, 1992.

[KNY] Ilan Komargodski, Moni Naor, and Eylon Yogev. Private Communication.

[KNY17] Ilan Komargodski, Moni Naor, and Eylon Yogev. White-box vs. black-box complexity of search problems: Ramsey and graph property testing. *Electronic Colloquium on Computational Complexity (ECCC)*, 24:15, 2017.

[Mic00] Silvio Micali. Computationally sound proofs. *SIAM J. Comput.*, 30(4):1253–1298, 2000.

[MRRR14] Raghu Meka, Omer Reingold, Guy N. Rothblum, and Ron D. Rothblum. Fast pseudorandomness for independence and load balancing - (extended abstract). In *Automata, Languages, and Programming - 41st International Colloquium, ICALP 2014, Copenhagen, Denmark, July 8-11, 2014, Proceedings, Part I*, pages 859–870, 2014.

[NY89] Moni Naor and Moti Yung. Universal one-way hash functions and their cryptographic applications. In *Proceedings of the 21st Annual ACM Symposium on Theory of Computing, May 14-17, 1989, Seattle, Washigton, USA*, pages 33–43, 1989.

[Ost91] Rafail Ostrovsky. One-way functions, hard on average problems, and statistical zero-knowledge proofs. In *Proceedings of the Sixth Annual Structure in Complexity Theory Conference, Chicago, Illinois, USA, June 30 - July 3, 1991*, pages 133–138, 1991.

[OV08] Shien Jin Ong and Salil P. Vadhan. An equivalence between zero knowledge and commitments. In *Theory of Cryptography, Fifth Theory of Cryptography Conference, TCC 2008, New York, USA, March 19-21, 2008.*, pages 482–500, 2008.

[PRS12] Krzysztof Pietrzak, Alon Rosen, and Gil Segev. Lossy functions do not amplify well. In Ronald Cramer, editor, *Theory of Cryptography - 9th Theory of Cryptography Conference, TCC 2012, Taormina, Sicily, Italy, March 19-21, 2012. Proceedings*, volume 7194 of *Lecture Notes in Computer Science*, pages 458–475. Springer, 2012.

[PW11] Chris Peikert and Brent Waters. Lossy trapdoor functions and their applications. *SIAM J. Comput.*, 40(6):1803–1844, 2011.

[Rom90] John Rompel. One-way functions are necessary and sufficient for secure signatures. In *Proceedings of the 22nd Annual ACM Symposium on Theory of Computing, May 13-17, 1990, Baltimore, Maryland, USA*, pages 387–394, 1990.

[RV09] Guy N. Rothblum and Salil P. Vadhan. Unpublished manuscript. 2009.

[Sim98] Daniel R. Simon. Finding collisions on a one-way street: Can secure hash functions be based on general assumptions? In Kaisa Nyberg, editor, *Advances in Cryptology - EUROCRYPT '98, International Conference on the Theory and Application of Cryptographic Techniques, Espoo, Finland, May 31 - June 4, 1998, Proceeding*, volume 1403 of *Lecture Notes in Computer Science*, pages 334–345. Springer, 1998.

[Vad12] Salil P. Vadhan. Pseudorandomness. *Foundations and Trends in Theoretical Computer Science*, 7(1-3):1–336, 2012.

A Black-Box Separation

This separation builds on Simon's oracle [Sim98], and the Reconstruction Paradigm of Gennaro and Trevisan [GT00] and extensions [GGKT05, HHRS07]. The presentation here closely follows that of Asharov and Segev [AS15]. We start by defining the notion of a fully-black-box construction.

Definition A.1. *A* fully black-box construction *of $(s(n), t(n))$-multi-collision resistant function family from a one-way permutation family, where $s(n) \geq \log t(n)$, consists of a polynomial-time oracle-aided algorithm* Gen *and an oracle-aided security reduction $\mathcal{M}$ that runs in time $T_{\mathcal{M}}(\cdot)$, with loss functions $\epsilon_{\mathcal{M},1}(\cdot)$ and $\epsilon_{\mathcal{M},2}(\cdot)$, satisfying:*

- **Correctness:** *Given any permutation $f = \{f_n : \{0,1\}^n \to \{0,1\}^n\}_{n\in\mathbb{N}}$, the algorithm $\mathsf{Gen}^f(1^n)$ outputs the description of an oracle aided circuit $C^{(\cdot)} : \{0,1\}^n \to \{0,1\}^{n-s(n)}$.*
- **Black-Box Proof of Security:** *Let $f = \{f_n : \{0,1\}^n \to \{0,1\}^n\}_{n\in\mathbb{N}}$ be any permutation. There exists a reduction $\mathcal{M}$ that runs in time $T_{\mathcal{M}}(\cdot)$, that given any (possibly non-uniform) adversary A, that runs in time $T_{\mathsf{A}}(\cdot)$, such that,*

$$\Pr_{\substack{C^{(\cdot)}\leftarrow\mathsf{Gen}^f(1^n)\\(x_1,x_2\ldots x_t)\leftarrow\mathsf{A}^f(1^n,C)}}\left[\begin{array}{c}\text{For all } i\neq j,\ x_i\neq x_j \text{ and}\\ C^f(x_i)=C^f(x_j)\end{array}\right] \geq \epsilon_{\mathsf{A}}(n)$$

for infinitely many $n \in \mathbb{N}$, the reduction $\mathcal{M}$ inverts the one-way permutation for infinitely many $n \in \mathbb{N}$. That is,

$$\Pr_{y\leftarrow\{0,1\}^n}\left[\mathcal{M}^{f,\mathsf{A}}(y) = f^{-1}(y)\right] \geq \epsilon_{\mathcal{M},1}\left(\epsilon_{\mathsf{A}}^{-1}(n)\cdot T_{\mathsf{A}}(n)\right)\cdot\epsilon_{\mathcal{R},2}(n)$$

where the probability is over the randomness of $\mathcal{M}$.

We will rule out fully-black-box constructions of (s,t)-MCRHs from one-way permutations for all $s \geq \log t$. In contrast, note that $(\log(t-1), t)$-MCRHs exist trivially (and unconditionally) for all values of t.[19]

[19] Consider a $(t-1)$-regular function $f : \{0,1\}^n \to \{0,1\}^{n-\log(t-1)}$. For such a function t-way collisions simply do not exist.

Theorem A.2. *Let* $(\mathsf{Gen}, \mathcal{R}, T_{\mathcal{R}}, \epsilon_{\mathcal{R},1}, \epsilon_{\mathcal{R},2})$ *be a fully black-box construction of* (s,t)*-MCRH from a one-way permutation family* $\mathcal{F}$*, where* $t \in \mathsf{poly}$ *and* $s(n) \geq \log t(n)$*, then at least one of the two holds:*

- $T_{\mathcal{M}}(n) \geq 2^{n/30}$ *(i.e., the reduction runs in exponential time).*
- $\epsilon_{\mathcal{M},1}(n^c) \cdot \epsilon_{\mathcal{M},2}(n) \leq 2^{-n/30}$ *for some constant* $c > 1$ *(i.e., the security loss is exponential).*

Because fully-black box constructions relativize, these constructions are correct and secure even when the underlying primitive (one-way permutations) and the adversary A (that breaks MCRHs) are given as oracles. So, as is the norm, we will prove the theorem by constructing an oracle Γ such that relative to Γ, secure one-way permutations exist while no construction of a multi-collision resistant hash function family is secure. In the next section, we describe the oracle used to prove this separation and give a proof outline.

A.1 The Oracle Γ_t Description

In this section, we define the oracle Γ_t relative to which we will show that one-way permutations exist while $(\log t, t)$-MCRH do not. We then give an overview of the proof. The oracle is an extension of Simon's oracle [Sim98] for breaking collision resistant hash functions. Instead of returning a pair of inputs that collide w.r.t. the circuit C, this oracle provides a vector of $\mathsf{poly}(t(n))$ inputs that collide w.r.t. C. Formally, we define the oracle below.

Definition A.3. *The oracle* Γ_t*'s interface is a pair of functions* (f, MCFind_t^f)*. Further, the oracle's state includes a randomness tape* $\mathcal{R}$ *which cannot be queried externally. We describe these below.*

- ***The function*** $f = \{f_n\}_{n\in\mathbb{N}} : \{0,1\}^n \to \{0,1\}^n$***.*** *For every* $n \in \mathbb{N}$*, the function* f_n *is a uniformly chosen permutation over* $\{0,1\}^n$*.*
- ***Randomness tape*** $\mathcal{R}$***.*** *The randomness tape* $\mathcal{R}$ *that has independent randomness for every possible circuit* C *queried to* MCFind_t^f*.*
- ***The multi-collision finder*** MCFind_t^f***.*** *Given the encoding of an oracle-aided circuit* C *that takes* n*-bit inputs and may access the oracle* f*, the* multi collision finder oracle *does the following:*

  $\underline{\mathsf{MCFind}_t^{f,\mathcal{R}}(C)}$*:*
  1. *Let* $t' = 3t^2 + t$*.*
  2. *From the random tape* $\mathcal{R}$*, interpret the random bits allocated for circuit* C *as* $(w_1, \pi_2, \pi_3, \ldots \pi_{t'})$ *where* $w_1 \in \{0,1\}^n$ *is a random* n*-bit string and each* $\pi_i : \{0,1\}^n \to \{0,1\}^n$ *is a random permutation.*
  3. *For all* $i \in \{2, 3, \ldots t'\}$*,*
     (a) *Find the first* j *such that* $C^f(w_1) = C^f(\pi_i(j))$*. Set* $w_j = \pi_i(j)$*.*
  4. *Output* $(w_1, w_2 \ldots w_{t'})$*.*

We need to show that relative to the oracle Γ_t, a random permutation f is still one-way and that no t-MCRH exists. We use a family of one-way permutations $f = \{f_n\}_{n\in\mathbb{N}}$ to allow the construction to invoke one-way permutation instances of various security parameters. We explicitly mention the randomness $\mathcal{R}$ because we want the oracle to return the same answer to any query when queried repeatedly.

Breaking MCRHs. This part is straightforward. The adversary given the encoding of the hash function C queries the MCFind oracle to find colliding inputs. We need to compute the success probability of this attack. A formal statement is given in Appendix A.2.

f is one-way relative to Γ. Showing that no adversary can invert f even given the MCFind-oracle is the challenging part. The proof here is based on the Reconstruction Paradigm due to Gennaro and Trevisan [GT00]. In this paradigm, f is proved to be one way by using any adversary who inverts the one-way permutation to find a compressed encoding of the oracle f. Since the oracle is a random permutation, it is incompressible. The formal proof is given in Appendix A.3.

This proof proceeds in two parts. In this first part, we show that the MCFind^f oracle is not very helpful in inverting f. The key challenge here is limiting the amount of information the adversary can learn using the MCFind^f oracle. The MCFind^f oracle makes exponentially many queries to f when it finds collisions. We want to show that even if that is the case, given that it does not output all the queries it makes, we can limit the amount of assistance it provides in inverting f. In particular we will crucially rely on the fact that, for a circuit C queried to MCFind^f, in the vector of collisions $\mathbf{w}$ the oracle returns, the marginal distribution of each element w_i is uniformly random.

In the second part, we show that given any adversary A that can invert f 'without much help from' the MCFind oracle, we can compress the random permutation f. This gives us a contradiction.

A.2 Breaking Multi-Collision-Resistant Hash Functions Relative to Γ_t

In this section, we prove that the $\mathsf{MCFind}_t^{f,\mathcal{R}}$ oracle can be used to break any candidate construction of a $t(n)$-MCRH. For brevity, we will refer to $t(n)$ as t.

Lemma A.4. *Relative to the oracle Γ_t, there exists a probabilistic polynomial-time algorithm A, such that for any function f, for any $n \in \mathbb{N}$, and for any oracle-aided circuit $C^f : \{0,1\}^n \to \{0,1\}^{n-s}$, where $s \geq \log t(n)$, it holds that:*

$$\Pr_{\mathbf{x} \leftarrow \mathsf{A}^{\Gamma_t}(1^n, C)} \left[\begin{array}{c} \textit{For all } i \neq j \in [t], \\ x_i \neq x_j \wedge C^f(x_i) = C^f(x_j) \end{array} \right] > \frac{1}{2t}.$$

where $\mathbf{x} = (x_1, x_2 \ldots x_t)$.

Proof. The input is a circuit $C : \{0,1\}^n \to \{0,1\}^{n-\lceil \log t \rceil}$. For any input $x \in \{0,1\}^n$, let $\mathcal{S}_{C,x}$ be the set of inputs x' such that $C(x) = C(x')$. Consider the following adversary A:

$\underline{\mathsf{A}^{\Gamma_t}(C)}$

1. Get $\mathbf{w} \leftarrow \mathsf{MCFind}_t^f(C)$.
2. If $\mathbf{w}$ has more than t distinct elements, output any t distinct elements.
3. Else output $\perp$.

We claim that A succeeds with the required probability. We show this in two steps. In the first step, we show that at least $1/t$ fraction of inputs have over t distinct collisions. In the second step, we bound the probability of the oracle finding t distinct collisions if they existed.

Claim A.4.1. *It holds that,*

$$\Pr_{x \leftarrow \{0,1\}^n}[|\mathcal{S}_{C,x}| \geq t] \geq 1/t.$$

Proof. The proof is by a counting argument. For any $y \in \{0,1\}^{n-\lceil \log t \rceil}$, let $C^{-1}(y) = \{x \in \{0,1\}^n : C(x) = y\}$. If $x \in C^{-1}(y)$, then the set $\mathcal{S}_{C,x}$ is exactly $C^{-1}(y)$. Let $\mathsf{Bad} = \{x \in \{0,1\}^n : |\mathcal{S}_{C,x}| \leq t-1\}$. As $\Pr_x[|\mathcal{S}_{C,x}| \geq t] = 1 - \Pr_x[x \in \mathsf{Bad}]$, we bound the size of the set Bad.

$$\begin{aligned}\mathsf{Bad} &= \{x : |\mathcal{S}_{C,x}| \leq t-1\} \\ &= \bigcup_{y \in \{0,1\}^{n-\lceil \log t \rceil}} C^{-1}(y) \text{ where } \left|C^{-1}(y)\right| \leq t-1\end{aligned}$$

Hence,

$$|\mathsf{Bad}| \leq 2^{n-\lceil \log t \rceil}(t-1) \leq 2^n \cdot \left(1 - \frac{1}{t}\right).$$

This implies that

$$\Pr_x[|\mathcal{S}_{C,x}| \geq t] = 1 - \Pr_x[x \in \mathsf{Bad}] \geq 1 - \frac{2^n(1-1/t)}{2^n} \geq \frac{1}{t},$$

as required. □

We will now show that for any $w_1 \in \{0,1\}^n$ with at least t-collisions w.r.t C, the MCFind oracle will find t distinct collisions with good probability.

Claim A.4.2. *Given any circuit C, consider any $w_1 \in \{0,1\}^n$ such that $S_{w_1} \geq t$, then*

$$\Pr_{\substack{\mathcal{R} \\ \mathbf{w} \leftarrow \mathsf{MCFind}(C)}} [\mathbf{w} \textit{ has } t \textit{ distinct elements}] \geq \frac{1}{2}$$

Proof. Let $\tau \geq t$ be the size of the set $\mathcal{S}_{C,w_1}$. Each subsequent w_i is an independent random sample from the set $\mathcal{S}_{C,w_1}$. Let Fail denote the event that the set of w_i's take at most $t-1$ values. We bound the probability of Fail as follows:

$$\begin{aligned}\Pr_{\mathcal{R}}[\mathsf{Fail}] &\leq \binom{\tau}{t-1}\left(\frac{t-1}{\tau}\right)^{t'} \\ &\leq \left(\frac{e\tau}{t-1}\right)^{t}\left(\frac{t-1}{\tau}\right)^{t'} && \text{using } \binom{N}{R} \leq \left(\frac{eN}{R}\right)^R \\ &\leq \left(\frac{t-1}{t}\right)^{3t^2} e^t && \text{substituting } t' = 3t^2 + t \\ &\leq \left(\frac{1}{2}\right)^{3t} e^t \leq \frac{1}{2} && \text{using } \left(1 - \frac{1}{t}\right)^t \leq 1/2\end{aligned}$$

□

Together, the two claims complete the proof. Since w_1 is chosen at random, with probability at least $1/t$, w_1 has at least t collisions w.r.t. C and then with probability $1/2$, the MCFind oracle will output at least t of them. □

A.3 f is a One-Way Permutation Relative to Γ_t

In this section, we will show that a random permutation $f = \{f_n : \{0,1\}^n \to \{0,1\}^n\}_n$ is one-way relative to the oracle Γ_t for any $t = \mathsf{poly}(n)$. We say that an adversary A is a (q, q')-adversary if A on any input makes at most $q(n)$ queries to the oracle Γ_t and for every circuit $C^f(\cdot)$ it queries to the MCFind oracle, C^f makes at most $q'(n)$ queries to f when evaluating any input.

We show that any (possibly computationally-unbounded) adversary that makes a bounded number of queries to the oracle Γ_t cannot invert a random one-way permutation.

Theorem A.5. *For any (q, q)-adversary A where $q = 2^{n/30}$, it holds that:*

$$\Pr_{\substack{f,\mathcal{R} \\ y \leftarrow \{0,1\}^n}} \left[\mathsf{A}^{\Gamma}(y) = f^{-1}(y)\right] < 2^{-n/30}$$

We will show that this result holds for any fixing of the oracle $f_{-n} = \{f_m\}_{m \neq n}$ while only f_n is picked at random. We want to formalize what it means for the adversary A to learn about an input y from the MCFind oracle. We define this notion below:

Definition A.6. *A query x to the oracle f produces a y-hit if $f(x) = y$. A query $C^f(\cdot)$ to the MCFind-oracle produces a $\mathsf{indirect}$-y-hit if MCFind outputs a vector $\mathbf{w} = (w_1, w_2 \dots w_{t'})$ such that there exists an index j such that the evaluation of $C^f(w_j)$ produces a y-hit.*

We denote by $\mathsf{CollHit}_y$ the event where one of the MCFind-queries made by the adversary $\mathsf{A}^{\Gamma}(y)$ produces an $\mathsf{indirect}$-y-hit.

If the event $\mathsf{CollHit}_y$ occurred, then the adversary A learns the pre-image of y from the MCFind oracle. Our proof proceeds in two steps. In the first step, we show that for every adversary A that succeeds in inverting y via a $\mathsf{CollHit}_y$, there is another adversary B that also inverts with good probability, but without triggering the $\mathsf{CollHit}_y$ event.

Lemma A.7. *Fix the values for the one-way permutation at other input lengths $f_{-n} = \{f_m\}_{m \neq n}$. For every (q, q)-query algorithm A such that,*

$$\Pr_{\substack{f_n,\mathcal{R}, \\ y \leftarrow \{0,1\}^n}} \left[\mathsf{A}^{\Gamma}(y) = f^{-1}(y)\right] \geq \frac{1}{q(n)},$$

then there exists a $(2q^2, q)$-query algorithm B such that,

$$\Pr_{\substack{f_n,\mathcal{R}, \\ y \leftarrow \{0,1\}^n}} \left[\mathsf{B}^{\Gamma}(y) = f^{-1}(y) \wedge \neg \mathsf{CollHit}_y\right] \geq \frac{1}{8t'q(n)^2}.$$

We prove Lemma A.7 in Appendix A.3.1. In the second step, using Gennaro and Trevisan's reconstruction paradigm, we show that no adversary can invert f without triggering the $\mathsf{CollHit}_y$ event.

Lemma A.8. *For every (q, q)-query algorithm A, where $q(n) = 2^{n/8}$,*

$$\Pr_{\substack{f_n,\mathcal{R}, \\ y \leftarrow \{0,1\}^n}} \left[\mathsf{A}^{\Gamma}(y) = f^{-1}(y) \wedge \neg \mathsf{CollHit}_y\right] \leq 2^{-n/7}.$$

Lemma A.8 is proved in Appendix A.3.2. Taken together, these two lemmas complete the proof.

Proof of Theorem A.5. Assume towards a contradiction that there exists an algorithm A with $q = 2^{n/30}$ such that,

$$\Pr_{f,y\leftarrow\{0,1\}^n}\left[\mathsf{A}^{\Gamma}(y) = f^{-1}(y)\right] > 2^{-n/30}$$

for infinitely many n, we can transform it to B, a $(2\cdot 2^{n/15}, 2^{n/30})$ algorithm such that,

$$\Pr_{\substack{f_n,\mathcal{R},\\ y\leftarrow\{0,1\}^n}}\left[\mathsf{B}^{\Gamma}(y) = f^{-1}(y) \wedge \neg\mathsf{CollHit}_y\right] \geq \frac{1}{8t'q(n)^2} \geq \frac{1}{8t'2^{2n/30}} > 2^{-n/7}$$

for infinitely many n's. This contradicts Lemma A.8. □

A.3.1 Hit Elimination

We will now prove Lemma A.7. We state it below for reference.

Lemma A.7. *Fix the values for the one-way permutation at other input lengths $f_{-n} = \{f_m\}_{m\neq n}$. For every (q,q)-query algorithm A such that,*

$$\Pr_{\substack{f_n,\mathcal{R},\\ y\leftarrow\{0,1\}^n}}\left[\mathsf{A}^{\Gamma}(y) = f^{-1}(y)\right] \geq \frac{1}{q(n)},$$

then there exists a $(2q^2,q)$-query algorithm B such that,

$$\Pr_{\substack{f_n,\mathcal{R},\\ y\leftarrow\{0,1\}^n}}\left[\mathsf{B}^{\Gamma}(y) = f^{-1}(y) \wedge \neg\mathsf{CollHit}_y\right] \geq \frac{1}{8t'q(n)^2}.$$

Proof. In this proof, we will show that given any adversary A that can invert the one-way permutation, we can construct another adversary B that does so without 'getting the inverse using the MCFind oracle'. We describe the adversary B below.

$\underline{\mathsf{B}^{\Gamma} \text{ on input } y\in\{0,1\}^n}$

Run $\mathsf{A}^{(\cdot)}(y)$ step-by-step and answer the queries to Γ as follows:

1. f *query*: Forward the query to Γ and return the answer.
2. MCFind^f *query*: Given a circuit $C^f : \{0,1\}^m \to \{0,1\}^{m'}$ as query,
   (a) *Hit Attempt:* Pick a random $z \leftarrow \{0,1\}^m$ and locally evaluate $C^f(z)$. While evaluating an f-gate in C, if a y-hit happens (i.e., the output of the gate is y), terminate and return the input to the gate as pre-image.
   (b) *Forward:* Query MCFind^f at C and return the answer.

Output. When A stops, return its output.

The queries B makes to MCFind oracle are subset of the ones A makes. So, B makes at most q queries to MCFind and $q^2+q \leq 2q^2$ queries to f oracle (A could make q queries to f and additionally every MCFind query that A makes, B incurrs q more f-queries). We need to show that B can invert

without the event $\mathsf{CollHit}_y$ happening. First, if A inverts with good probability without $\mathsf{CollHit}_y$, then we are done. That is, if

$$\Pr_{\substack{f_n,\mathcal{R},\\ y\leftarrow\{0,1\}^n}}\left[\mathsf{A}^{\Gamma}(y) = f^{-1}(y) \wedge \neg\mathsf{CollHit}_y\right] \geq \frac{1}{2q(n)}$$

for infinitely many n's, then the claim holds. So, let us assume that:

$$\Pr_{\substack{f_n,\mathcal{R},\\ y\leftarrow\{0,1\}^n}}\left[\mathsf{A}^{\Gamma}(y) = f^{-1}(y) \wedge \mathsf{CollHit}_y\right] \geq \frac{1}{2q(n)}$$

In this case, we will show that B can also invert f_n without triggering $\mathsf{CollHit}_y$. First we state a claim about particular 'good' y's and then use it to complete the proof using an averaging argument.

Claim A.8.1. *For every choice of f and $y \in \{0,1\}^n$ that satisfy:*

$$\Pr_{\mathcal{R}}\left[\mathsf{A}^{\Gamma}(y) = f^{-1}(y) \wedge \mathsf{CollHit}_y\right] \geq \frac{1}{4q(n)},$$

B also inverts y with good probability without a $\mathsf{CollHit}_y$:

$$\Pr_{\mathcal{R}}\left[\mathsf{B}^{\Gamma}(y) = f^{-1}(y) \wedge \neg\mathsf{CollHit}_y\right] \geq \frac{1}{4q(n)t'}$$

We will prove the claim below. First, we finish the proof of Lemma A.7 by an averaging argument. Let $T = \left\{(y,f) \mid \Pr_{\mathcal{R}}\left[\mathsf{A}^{\Gamma}(y) = f^{-1}(y) \wedge \mathsf{CollHit}_y\right] \geq \frac{1}{4q(n)}\right\}$. Observe that

$$\Pr_{y,f}[(y,f) \in T] \geq 1/4q(n)$$

because $\frac{1}{2q(n)} \leq \Pr_{\mathcal{R},f,y}\left[\mathsf{A}^{\Gamma}(y) = f^{-1}(y) \wedge \mathsf{CollHit}_y\right] \leq \Pr_{y,f}[(y,f) \in T] + \frac{1}{4q(n)} \Pr_{y,f}[(y,f) \notin T]$. We lower bound the success probability of B below:

$$\begin{aligned}\Pr_{\substack{f_n,\mathcal{R},\\ y\leftarrow\{0,1\}^n}}\left[\mathsf{B}^{\Gamma}(y) = f^{-1}(y) \wedge \neg\mathsf{CollHit}_y\right] &\geq \Pr_{f,y}[(f,y) \in T] \cdot \Pr_{\mathcal{R}}\left[\mathsf{B}^{\Gamma}(y) = f^{-1}(y) \wedge \neg\mathsf{CollHit}_y\right] \\ &\geq \frac{1}{2q(n)} \cdot \frac{1}{4t'q(n)} \geq \frac{1}{8t'q(n)^2}.\end{aligned}$$

This concludes the proof.

□

Proof of Claim A.8.1. In this proof, we need to show that if A can invert f by triggering the $\mathsf{CollHit}_y$ event, B can do so without triggering the event. This proof crucially relies on the fact that in the output of the MCFind oracle $\mathbf{w}$, the marginal distribution of every individual coordinate of $\mathbf{w}$ is uniform. This allows B to 'mimic' a $\mathsf{MCFind}(C)$ query by simply picking a random z and evaluating $C^f(z)$. While this will not help in finding collisions, we show that it has an identical effect in terms of finding y's pre-image.

Let $q = q(n)$ and $C_1, C_2, \ldots, C_q$ be the random variables corresponding to A's queries to MCFind. Assume that they are distinct. In addition, let $\mathbf{w}_1, \mathbf{w}_2 \ldots \mathbf{w}_q$ denote the responses by the oracle. We define two notions of hits. While B is running A on input y, and A asks B to query C_i to the MCFind oracle:

- **Good Hit:** A $\mathsf{GoodHit}_i$ event happens when B runs $C_i^f(z_i)$ in Step 2a and a y-hit happens.
- **Bad Hit:** A BadHit_i event happens when B sends query C_i to MCFind-oracle and a $\mathsf{CollHit}_y$ happens (i.e., the oracle MCFind returns $\mathbf{w}_i$ where one of the elements in the vector incurs an indirect-y-hit.)

$\mathsf{GoodHit}$ is the event when B has inverted the one-way permutation without incurring a $\mathsf{CollHit}_y$ on any query to MCFind oracle. In the $\mathsf{GoodHit}$ event, B discovers the inverse itself while in the BadHit event, the MCFind oracle finds the inverse. We want to show that B can invert with good probability when a $\mathsf{GoodHit}_i$ event occurs before any BadHit events happen. Let $\mathsf{BadHit}_{<i}$ denote the event when $\mathsf{BadHit}_{i'}$ happens for some $i' < i$.

It holds that,

$$\begin{aligned}\frac{1}{4q} &\leq \Pr_{\mathcal{R}}\Big[\mathsf{A}^{\Gamma}(y) = f^{-1}(y) \wedge \mathsf{CollHit}_y\Big] \\ &\leq \sum_i \Pr_{\mathcal{R}}[\mathsf{BadHit}_i \mid \neg\mathsf{BadHit}_{<i}]\end{aligned}$$

We observe that, $\Pr_{z_i}[\mathsf{GoodHit}_i \mid \mathsf{BadHit}_{<i}] \geq \frac{1}{t'}\Pr_{\mathcal{R}}[\mathsf{BadHit}_i \mid \mathsf{BadHit}_{<i}]$. This follows from the fact that $\mathcal{R}$ has independent randomness for every circuit and that in the vector $\mathbf{w}_i = (w_{i,1}, w_{i,2} \ldots w_{i,t'})$ output by MCFind, the marginal distribution of each $w_{i,j}$ is uniformly random. Hence,

$$\leq t' \cdot \sum_i \Pr_{\mathcal{R},z_i}[\mathsf{GoodHit}_i \mid \neg\mathsf{BadHit}_{<i}]$$

This implies that,

$$\sum_i \Pr_{\mathcal{R},z_i}[\mathsf{GoodHit}_i \mid \neg\mathsf{BadHit}_{<i}] \geq \frac{1}{t'} \cdot \frac{1}{4q}.$$

This is precisely the success probability $\Pr_{\mathcal{R},\mathbf{z},y}\Big[\mathsf{B}^{\Gamma}(y) = f^{-1}(y) \wedge \neg\mathsf{CollHit}_y\Big]$. This completes the argument. □

A.3.2 From Inverting to Compressing

In this section we will use any adversary A that inverts f_n with good probability to construct a shorter encoding for the random permutation f_n. The canonical encoding for a random permutation on $\{0,1\}^n$ requires $\log(2^n!)$ bits to store, and this is necessary. If our new encoding is shorter, it is a contradiction.

Lemma A.8. *For every (q,q)-query algorithm A, where $q(n) = 2^{n/8}$,*

$$\Pr_{\substack{f_n,\mathcal{R},\\ y\leftarrow\{0,1\}^n}}\Big[\mathsf{A}^{\Gamma}(y) = f^{-1}(y) \wedge \neg\mathsf{CollHit}_y\Big] \leq 2^{-n/7}.$$

Proof. We will prove the statement for any fixed $\mathcal{R}$ and f_{-n}. Let $q = q(n)$. This argument is based on the Reconstruction Paradigm of Gennaro and Trevisan. The oracle Γ consists of the random permutation $f = \{f_i\}_{i\in\mathbb{N}}$ and the randomness tape $\mathcal{R}$. We will show that given such an adversary A, we can compress the oracle Γ and this yields a contradiction because random oracles are incompressible.

Encoding f_n. The permutation f_n is usually encoded as a truth table. The new encoding also has a partial truth table Z, and a set of pre-images X and images Y such that $f_n(X) = Y$. But the exact truth table mapping the relationship between X and Y is not stored, instead A is used to reconstruct this. This is the source of compression.

Let $I_f \subseteq \{0,1\}^n$ be the set of images A inverts without $\mathsf{CollHit}$ happening. That is,

$$I_f \triangleq \left\{ y \mid \mathsf{A}^{\Gamma}(y) = f^{-1}(y) \wedge \neg \mathsf{CollHit}_y \right\}.$$

We will now construct $Y_f \subseteq I_f$:

$\mathsf{EncodeY}$ on input I_f

1. Set $Y = \emptyset$, where $\emptyset$ denotes the empty set.
2. While $I \neq \emptyset$,
   (a) Pick y, the lexicographically smallest element in I.
   (b) Construct the set F_y as follows:
      i. For each f_n-query x_i made by $\mathsf{A}(y)$, add the output $y_i = f_n(x_i)$ to F_y.
      ii. For each MCFind-query C_i made by A, let the output be $\mathbf{w_i} = (w_{i,1}, w_{i,2} \dots w_{i,t'})$. Add to F_y the output of all queries made to f_n while evaluating $C_i^f(w_{i,j})$ for all $j \in [t']$.
   (c) $I = I \setminus F_y$ (Remove all the elements of F_y from I.)
   (d) $I = I \setminus \{y\}$
   (e) $Y = Y \cup \{y\}$
3. Output the set Y

Claim A.8.1. *The set $Y_f = \mathsf{EncodeY}(I_f)$ is not too small.*

$$|Y_f| \geq \frac{|I_f|}{t'q^2}$$

Proof. The claim follows from the fact that $I_f = \cup_{y \in Y_f} F_y$ and that for every y, $|F_y| \leq t'q^2$. This follows from the fact that A is a (q,q)-adversary and hence makes at most q queries to Γ. Every query to f adds one element to F_y while every query C_i to MCFind-oracle can add $t'q$ elements when the C_i is evaluated on every output from $\mathbf{w_i}$. Therefore, at most $t'q^2$ queries are added to F_y. □

Let $X_f = f^{-1}(Y_f)$. Let Z_f be the partial truth-table of f_n corresponding to all elements in $\{0,1\}^n \setminus Y_f$. The new encoding of f consists of f_{-n} and $\mathcal{R}$ as before along with X_f, Y_f, Z_f. The sets X_f and Y_f are stored but not the mapping of f_n between them. The sets X_f and Y_f need $\log \binom{2^n}{|Y_f|}$ bits each to describe while encoding Z_f needs $\log((2^n - |Y_f|)!)$ bits of space.

Claim A.8.2. *The oracle Γ_t can be encoded as $(f_{-n}, \mathcal{R}, X_f, Y_f, Z_f)$ along with a description of A.*

Proof. To prove the claim, given $\mathcal{R}, f_{-n}$ along with X_f, Y_f, Z_f and a description of A we will show how to reconstruct f_n. We drop the subscript f for convenience. This implies that the compressed encoding is a valid encoding of the oracle Γ_t. To reconstruct Γ_t, we need to show an algorithm that

can reconstruct the truth-table of f_n. We describe the algorithm below:

Reconstruct on input $(\mathcal{R}, f_{-n}, X_f, Y_f, Z_f)$

1. While $Y \neq \emptyset$ do:
   (a) Pick the lexicographically first element $y \in Y$.
   (b) Run A with input y. Answer every query as follows:
      i. For an f_n query x', if $(x', y') \in Z$ for some y', answer y'. Else, do the following:
         A. Answer y.
         B. Set $Z = Z \cup \{(x', y)\}$.
         C. Remove y from Y.
         D. Jump to Step 1.
      ii. For an MCFind-query C', run the modified MCFind algorithm described below:
         A. Set $\gamma = \lceil 2t \log t \rceil$.
         B. Interpret randomness for C' as $(w_1, \pi_2, \pi_3, \ldots \pi_{t'})$ where $w_1 \in \{0,1\}^n$ and each $\pi_i : \{0,1\}^n \to \{0,1\}^n$ is a permutation.
         C. Compute $C'^f(w_1)$ answer all f_n queries according to Z.
         D. For all $i \in \{2, 3, \ldots t'\}$:
            - For every j from 0 to 2^n, evaluate $C^f(\pi_i(j))$. If any f_n query is not in Z, pick the next j. If all f_n queries are in Z and $C^f(\pi_i(j)) = C^f(w_1)$, set $w_j = \pi_i(j)$.
         E. return $(w_1, w_2 \ldots w_{t'})$.
   (c) When A terminates and outputs x do the following:
      i. Set $Z = Z \cup \{(x, y)\}$.
      ii. Remove y from Y.
      iii. Repeat.
2. **Output.** The completed truth table Z.

We will now show that the Reconstruct algorithm does reconstruct the truth table of f_n.

Claim A.8.3. *The set Z output by* ReConstruct *contains the entire truth table of f_n*

Proof. We need to show that the reconstruction algorithm finds the pre-images for all $y \in Y$. We prove this by induction on y's sorted lexicographically. The base case and the induction are identical. Assume that the reconstruction algorithm trying to invert is y, the lexicographically smallest element in the set Y, and that all elements lexicographically smaller have been inverted and are now in Z.

The reconstruction algorithm runs A with input y. Because $y \in Y \subseteq I$, the algorithm A would invert y correctly if every query it makes to Γ_t is answered correctly. First we will show that every MCFind query that $\mathsf{A}(y)$ makes is answered identically.

MCFind returns the same response. Consider any query C' that $\mathsf{A}(y)$ made to MCFind oracle and $\mathbf{w}'$ be the response received. We want to show that the reconstruction algorithm would give the same response. First we show that all f_n-queries made while evaluating $C'(w'_i)$ are in Z. Then we use this fact to show that the reconstruction algorithm will return the same response.

Consider the set F_y from the $\mathsf{EncodeY}$ algorithm. The set F_y contains all f_n-queries made while evaluating $C'(w'_i)$. Furthermore, we also know that none of these queries returned the answer y because $\mathsf{CollHit}_y$ did not happen (because the set I_f is defined as elements which A inverts without $\mathsf{CollHit}_y$ event happening). Hence $F_y \setminus \{y\}$ contains all the queries made the f_n-queries made when evaluating $C'(w'_i)$ for all i. Finally, we claim that $F_y \setminus \{y\}$ is contained in Z. Observe that all elements in $F_y \setminus \{y\}$ not in I_f were originally in Z; by induction, all elements in $F_y \cap I_f$ lexicographically smaller than y are in Z, finally all the elements that were lexicographically larger were removed from I_f and added to Z while encoding Y.

To show that the reconstruction algorithm would return the same answer, first observe that the algorithm uses the same randomnesss $\mathcal{R}$ to find the same w'_1 and the permutations π_2 to $\pi_{t'}$. We know that each subsequent w'_i is the first element according to π_i that collides with w'_1 w.r.t. C'. We also know that the reconstruction algorithm can compute $C'(w'_i)$. The reconstruction algorithm picks the first j such that it can compute $C'(\pi_i(j))$ and that $C'(\pi_i(j)) = C'(w'_1)$. Hence for every coordinate, it would pick w'_i — the original answer. This implies that its response will be identical.

Answering f_n queries. We know that when A ran on input y one of the two happened:

1. *The event y-hit occurred*, i.e., A made a query x directly to f such that the response was y. We will show that in this case, the $\mathsf{ReConstruct}$ algorithm will find the pre-image of y in Step 1(b)i.
2. *The event y-hit did not occur.* In this case, we will show that all the queries to f are answered correctly and hence the output of the algorithm $\mathsf{A}(y)$ will be the pre-image of y.

Event y-hit happened. This implies that when A ran on input y, there was a query x such that $f_n(x) = y$. We claim that this would cause loop-termination in Item 1(b)i. We have already seen that MCFind oracle returns the same responses. Consider every query made to f_n by A: either the value was already in Z, in which case it was answered correctly or the value was in X. If the value was in X, we claim that the query must be x (the pre-image of y). Assume not, let the query be $x' \neq x$. Let $y' = f_n(x')$. We know that $y' \in Y$, because otherwise, it would be in Z and hence answered. We also know that y is the lexicographically smallest element in Y at the moment and hence smaller than y'. Then the $\mathsf{EncodeY}$ algorithm inserted y in to Y before y'. Also $y' \in F_y$ because it was queried in the execution of A on y. This is a contradiction because $y' \in F_y$ implies that y' would have been removed from I_f and hence not inserted in to Y_f. So, it must be the case that x was the query made and the reconstruction algorithm correctly answered y.

Event y-hit did not occur. We also know that the event $\mathsf{CollHit}_y$ did not happen. This implies that F_y does not contain the element y. Hence all the queries needed for executing A on y are present in Z. Hence A would get the same answers on all the queries it makes. Hence A would terminate returning the pre-image of y, $x = f^{-1}(y)$ as the output. The reconstruction algorithm would then add that pair (x, y) to Z.

This implies that the Reconstruction algorithm will populate Z with the pre-images of all elements in Y_f. □

This completes the proof. We have shown that $(f_{-n}, \mathcal{R}, X_f, Y_f, Z_f)$ is a valid encoding of the oracle. □

Now we will show that the success probability of A must be low. Otherwise, it can be used to compress the oracle 'too much.'

Computations. Set $\epsilon = 2^{-n/8}$ and $q = 2^{n/8}$. For any fixed values of $\mathcal{R}, f_{-n}$, consider the following set:

$$\mathcal{S}_{\mathcal{R},f_{-n}} = \left\{ f_n : \Pr_{y \leftarrow \{0,1\}^n} \left[\mathsf{A}^{\Gamma}(y) = f_n^{-1}(y) \wedge \neg \mathsf{CollHit}_y \right] \geq \epsilon \right\}.$$

Any $f_n \in \mathcal{S}_{\mathcal{R},f_{-n}}$ can be encoded in $2\log(\binom{2^n}{|Y_f|}) + \log((2^n - |Y_f|)!)$ bits. From Claim A.8.1, $|Y_f| \geq \frac{|I_f|}{q^2 t'} \geq 2^{5n/8}/t' \triangleq B$. Hence f_{-n} can be encoded in $2\log(\binom{2^n}{|B|}) + \log((2^n - B)!)$ bits. The total number of permutations that can be encoded by $2\log(\binom{2^n}{|B|}) + \log((2^n - B)!)$ bits is bounded by $2^{2\log(\binom{2^n}{|B|}) + \log((2^n - B)!)}$. This is an upper-bound on the size of $S_{\mathcal{R},f_{-n}}$. Hence, for any fixed $\mathcal{R}, f_{-n}$,

$$\Pr_{f_n}[f_n \in \mathcal{S}_{\mathcal{R},f_{-n}}] \leq \frac{\binom{2^n}{B}^2 \cdot (2^n - B)!}{(2^n)!} = \frac{\binom{2^n}{B}}{|Y|!} \leq \left(\frac{2^n e}{B}\right)^B \left(\frac{e}{B}\right)^B = \left(\frac{2^n e^2}{B^2}\right)^B,$$

where we use the inequalities $N! \geq (N/e)^N$ and $\binom{N}{R} \leq (Ne/R)^R$. Substituting $B = 2^{5n/8}/t'$, for large enough n, this implies that:

$$\left(\frac{2^n e^2}{B^2}\right)^B \leq \left(\frac{2^n e^2 t'^2}{2^{10n/8}}\right)^B \leq \left(2^{-n/4} e^2 t'^2\right)^B \leq 2^{-n}.$$

So, we can conclude that for any computationally-unbounded algorithm A that makes at most $2^{n/8}$ queries to Γ_t, and its queries to MCFind are circuits of size at most q cannot invert f without triggering the $\mathsf{CollHit}_y$ event: For every $f_{-n}, \mathcal{R}$,

$$\begin{aligned}
\Pr_{\substack{f_n \\ y \leftarrow \{0,1\}^n}} \left[\mathsf{A}^{\Gamma}(y) = f^{-1}(y) \wedge \neg \mathsf{CollHit}_y \right] &\leq \Pr_{f_n}[f_n \in \mathcal{S}_{\mathcal{R},f_{-n}}] \cdot \Pr_{y \leftarrow \{0,1\}^n} \left[\mathsf{A}^{\Gamma}(y) = f^{-1}(y) \wedge \neg \mathsf{CollHit}_y \mid f_n \in \mathcal{S}_{\mathcal{R},f_{-n}} \right] \\
&\quad + \Pr_{f_n}[f_n \notin \mathcal{S}_{\mathcal{R},f_{-n}}] \cdot \Pr_{y \leftarrow \{0,1\}^n} \left[\mathsf{A}^{\Gamma}(y) = f^{-1}(y) \wedge \neg \mathsf{CollHit}_y \mid f_n \notin \mathcal{S}_{\mathcal{R},f_{-n}} \right] \\
&\leq 2^{-n} \cdot 1 + 1 \cdot 2^{-n/8} \\
&\leq 2^{-n/7}.
\end{aligned}$$

This completes the argument. □

ECCC ISSN 1433-8092

https://eccc.weizmann.ac.il