Electronic Colloquium on Computational Complexity, Report No. 46 (2026)

Black-Box Separation Between Multi-Collision Resistance and Collision Resistance

Xinyu Mao * Jiapeng Zhang *

March 29, 2026

Abstract

A K-multi-collision-resistant hash function (K-MCRH) is a shrinking keyed function for which it is computationally infeasible to find K distinct inputs that map to the same output under a randomly chosen hash key; the case $K = 2$ coincides with the standard definition of collision-resistant hash function (CRH). A natural question is whether K-MCRH implies CRH for $K \geq 3$, as noted by Komargodski, Naor, and Yogev (EUROCRYPT 2018) and also by Jain, Li, Robere, and Xun (FOCS 2024).

We resolve this question for all constant K, showing that there is no black-box construction of K-MCRH from $(K+1)$-MCRH for all constant $K \geq 2$. We also show that there is no black-box construction of distributional CRH (which is another relaxation of CRH) from 3-MCRH, answering an open question posed by Komargodski and Yogev (CRYPTO 2018) and also by Berman, Degwekar, Rothblum, and Vasudevan (EUROCRYPT 2018). Besides applications in cryptography, our separation also implies black-box separations between TFNP search problems, which are related to problems in proof complexity and other areas.

1 Introduction

Hash functions are among the most fundamental primitives in cryptography, providing a wide range of security properties from weak to strong. Among these properties, *collision resistance* is one of the most widely used. Specifically, a collision-resistant hash function (CRH) is a keyed function such that, for a randomly chosen hash key, it is computationally infeasible to find two distinct inputs that map to the same output.

While the definition of CRH is straightforward and intuitive, with numerous applications including vector commitment, digital signatures, and blockchain, its complexity remains elusive. CRH is often deemed as a symmetric primitive; however, we do not know how to construct it from one-way functions (OWFs). A seminal work by Simon [Sim98] showed that CRH cannot be built from OWFs in a black-box manner. More surprisingly, even when indistinguishable obfuscation (iO) is added to the picture, OWF plus iO still cannot give a black-box construction of CRH [AS16]. This is in stark contrast to the fact that CRH can be instantiated under a wide range of concrete assumptions, such as the Diffie–Hellman assumption, the Learning with Errors assumption, and the Quadratic Residuosity assumption.

*Research supported by NSF CAREER award 2141536.
Thomas Lord Department of Computer Science, University of Southern California.
Email: {xinyumao, jiapengz}@usc.edu

ISSN 1433-8092

Multi-collision resistance. In light of the difficulty of constructing CRHs from generic assumptions, a series of works considered a natural relaxation of collision resistance called *multi-collision resistance* [BKP18, BDRV18, KNY18]. Informally, a K-multi-collision-resistant hash function (K-MCRH) is a hash function such that no efficient adversary can find a *K-collision* (i.e., K distinct inputs that are mapped to the same hash value) given a random hash key. We note that a CRH is simply the special case where $K = 2$. Applications of MCRH include three-message zero-knowledge arguments [BKP18] and constant-round statistically hiding commitment schemes [BDRV18].

Complexity of MCRH. Komargodski, Naor, and Yogev [KNY18] proved that MCRH is also black-box separated from OWF. Intuitively, K-MCRH becomes weaker as K increases, and hence a natural yet more challenging question is posed:

Is there a black-box separation between K-MCRH and CRH (for $K \geq 3$)?

[KNY18] claimed that there is no black-box construction of CRHs from 3-MCRHs; however, a flaw was later found in their proof, and it is unlikely to be fixed in a simple way [JLRX24, Yog25]. Subsequently, weaker black-box separations were established using techniques from proof complexity [JLRX24, BGSD25]. Roughly speaking, these results only rule out black-box deterministic many-one reductions, whereas cryptographic reductions are typically randomized Turing reductions. On the other hand, there are *white-box* constructions of *infinitely-often secure* CRH from K-MCRH for any constant K [RV24, BT24].

Distributional collision resistance. Another relaxation of CRH, introduced by Dubrov and Ishai [DI06], is *distributional CRH*. Distributional CRH only guarantees that no efficient adversary can generate a *random* collision when given a randomly chosen hash key. More precisely, for a random hash key hk, it is computationally hard for any adversary to sample a pair (x, x') such that x is uniformly distributed and x' is uniformly distributed over the set of preimages $h_{\mathsf{hk}}^{-1}(h_{\mathsf{hk}}(x)) = \{z : h_{\mathsf{hk}}(x) = h_{\mathsf{hk}}(z)\}$. Distributional CRH is also sufficient for constructing constant-round statistically hiding commitment scheme [BHKY19]. Then, a natural question is the relation between MCRH and distributional CRH:

Does MCRH and distributional CRH imply each other?

Komargodski and Yogev [KY18] showed that distributional CRH can be constructed under the assumption of average-case hardness of the class *Statistical Zero-Knowledge* (SZK). In contrast, no analogous result is currently known for MCRH.

TFNP search problems. TFNP is the class of NP search problems that are guaranteed to have solutions. The guarantee usually stems from non-constructive combinatorial proofs. For example, by the pigeonhole principle, a function $f : [M] \to [N]$ must have a K-collision whenever $M > (K-1)\cdot N$, which defines a search problem: given a circuit computing f, find a K-collision of f. This problem, denoted K-$\textsc{Pigeon}_N^M$, is closely related to various problems in cryptography, proof complexity, and other areas. The class of TFNP problems reducible to 2-$\textsc{Pigeon}_N^{2N}$, known as PPP, is one of the "original five" subclasses of TFNP [Pap94]. The generalized K-$\textsc{Pigeon}_N^M$ problem is systematically studied by Jain, Li, Robere, and Xun [JLRX24]. K-$\textsc{Pigeon}_N^M$ is closely related to K-MCRH: A K-MCRH can be view as a distribution of K-$\textsc{Pigeon}_N^M$ instances for which the search problem

is intractable. It is proven in [JLRX24] that there is no black-box *many-one* reduction from $(K+1)$-$\text{PIGEON}_{N'}^{M'}$ to K-PIGEON_{N}^{M} where N, M, N', N' are polynomially related. Proving the non-existence of black-box Turing reduction is left as an open question, which is also posed in [FGPR24].

1.1 Our Results

In this paper, we show that (1) CRH cannot be constructed from 3-MCRH in a black-box way, and (2) distributional CRH cannot be constructed from 3-MCRH in a black-box way. We in fact prove a stronger separation result:

Theorem 1 (Informal version of theorem 15)**.** *For all constant $K \geq 2$, there is no black-box construction of K-MCRH from $(K+1)$-MCRH.*

The proof proceeds by presenting a pair of oracles (f, Ψ) and showing that Ψ breaks all K-MCRH construction and f remains $(K+1)$-collision-resistant in the presence of Ψ. In the case of $K = 2$, our oracle Ψ returns a random collision (x, x') with the same distribution as in the definition of distributional CRH; that is, Ψ also breaks any distributional CRH construction. Therefore, we conclude that

Theorem 2. *There is no black-box construction of distributional CRH from 3-MCRH.*

Connections to TFNP search problems. Two TFNP search problems are studied in [JLRX24]: one is based on pigeonhole principle (K-PIGEON_{N}^{M}), which is closely related to MCRH; the other (RAMSEY_{λ}) is related to Ramsey number.

- K-PIGEON_{N}^{M}: Let $K(\lambda), M(\lambda), N(\lambda)$ be integer parameters satisfying $M > (K-1)N$. The K-PIGEON_{N}^{M} problem is defined as follows:
  - Input: $(1^\lambda, h)$, where h a map $h : [M(\lambda)] \to [N(\lambda)]$ of M pigeons to N holes represented by a $\text{poly}(\lambda)$-size circuit.
  - Solutions: a K-collision of h.
- RAMSEY_{λ}: For this problem, the input specifies a graph on 2^λ vertices.
  - Input: a circuit $C : \{0,1\}^\lambda \times \{0,1\}^\lambda \to \{0,1\}$ of $\text{poly}(\lambda)$-size.
  - Solutions: There are two kinds of solutions. One is a certificate that C is not a valid encoding of a simple undirected graph, i.e., u for which $C(u,u) = 1$ (self-loops) or u, v for which $C(u,v) \neq C(v,u)$. Otherwise, we want to find $\lambda/2$ indices $V = \{v_1, v_2, \ldots, v_{\lambda/2}\}$ which form a clique or independent set. The existence of solution is guaranteed by an upper bound on Ramsey number: $R(\lambda/2, \lambda/2) \leq 2^\lambda$ [1].

A typical parameter setting is that $N = 2^n, M = 2^m$ for integer parameters m, n where m is polynomial in n. The correspondence between K-MCRH and K-Pigeon problem is obvious, and hence theorem 1 implies that

[1] That is, a graph with 2^λ vertices must contain either a clique of size $\lambda/2$, or an independent set of size $\lambda/2$.

Theorem 3. *For all constant $K \geq 2$ and parameters $n, m, n', m' \in \mathsf{poly}(\lambda)$ such that $m \geq n + \lceil \log K \rceil + 1$ and $m' \geq n' + \lceil \log(K+1) \rceil + 1$, there exists no black-box Turing reduction from $(K+1)$-$\textsc{Pigeon}^{2^{m'(\lambda)}}_{2^{n'(\lambda)}}$ to K-$\textsc{Pigeon}^{2^{m(\lambda)}}_{2^{n(\lambda)}}$.*

Goldberg and Papadimitriou [GP17] conjectured that there is no Turing reduction from $\textsc{Ramsey}_\lambda$ to 2-$\textsc{Pigeon}^{2^{m(\lambda)}}_{2^{n(\lambda)}}$. The result in [JLRX24] showed that there is no black-box many-one reduction. Our result strengthens the black-box separation by ruling out black-box Turing reductions.

Specifically, it is proven in [JLRX24] that for all constant $K \geq 1$, there is a Turing reduction from $(K+1)$-$\textsc{Pigeon}^{2^{m'(\lambda)}}_{2^{n'(\lambda)}}$ to $\textsc{Ramsey}_\lambda$ where $n'(\lambda) = \frac{\lambda}{4(K+1)} + \frac{1}{2}$ and $m'(\lambda) = \lambda$. Therefore, combining theorem 3, we conclude that

Corollary 4. *For all constant $K \geq 2$ and parameters $n, m \in \mathsf{poly}(\lambda)$ such that $m \geq n + \lceil \log K \rceil + 1$, there exists no black-box Turing reduction from $\textsc{Ramsey}_\lambda$ to K-$\textsc{Pigeon}^{2^{m(\lambda)}}_{2^{n(\lambda)}}$.*

1.2 Technical Overview

We use the separation of CRH from 3-MCRH (i.e., the case $K = 2$) to illustrate our proof techniques.

Two-oracle methodology. A fully black-box construction of primitive **P** from another primitive **Q** is a construction $\mathcal{P}$ that uses **Q** as black-box in following sense:

- syntactically, $\mathcal{P}^{\mathcal{Q}}$ is an instantiation of **P** as long as $\mathcal{Q}$ is an instantiation of **Q**; and
- there exists a reduction $\mathcal{R}$ such that for any instantiation $\mathcal{Q}$ (of **Q**) and adversary Ψ, if Ψ breaks $\mathcal{P}^{\mathcal{Q}}$ then $\mathcal{R}^{\mathcal{Q},\Psi}$ breaks $\mathcal{Q}$.

Our proof follows the two-oracle methodology, which is commonly used in previous separation results [Sim98, AS16, BD19, DM24]. That is, we present a pair of oracles (f, CF) such that f instantiates **Q**, which is 3-MCRH in our case, and the following holds:

- CF^f breaks any black-box construction of CRH from f.
- f remains 3-collision-resistant in the presence of the oracle CF^f, i.e., it is still hard to find a 3-collision for f given the oracle CF^f.

This is sufficient to rule out the existence of fully-black-box constructions. Moreover, it suffices to show a random function $f : \{0,1\}^\ell \to \{0,1\}^\lambda$ remains 3-collision-resistant given the oracle CF^f. The oracle CF, where CF stands for *collision finder*, is precisely the same oracle used in [Sim98] to separate CRH from OWF:

> $\mathsf{CF}^f(C)$: on input an oracle-aided circuit $C^{(\cdot)} : \{0,1\}^m \to \{0,1\}^n$, it picks $w^1_C \leftarrow \{0,1\}^m$ and $w^2_C \leftarrow \left\{w : C^f(w) = C^f(w^1_C)\right\}$ uniformly at random, and outputs (w^1_C, w^2_C).

Any black-box CRH construction from f can be viewed as an oracle-aided circuit, and hence CF^f indeed breaks all black-box CRH constructions from f.

Then, our goal is to show that

> for any efficient adversary $\mathcal{A}$, $\mathcal{A}^{f,\mathsf{CF}^f}$ outputs a 3-collision for f with probability negligible in λ, where $f : \{0,1\}^\ell \to \{0,1\}^\lambda$ is a random function.

Let CollFound denote the event that $\mathcal{A}^{f,\mathsf{CF}^f}$ outputs a 3-collision for f. Our first key insight is to split the event CollFound into two cases, depending on where the collision finally happens. Without loss of generality, we make the following assumptions on $\mathcal{A}$:

- $\mathcal{A}$ never makes a repeated query.
- After each f-query, $\mathcal{A}$ outputs a 3-collision immediately if there is a 3-collision in f-queries it has made.
- After each CF-query C, say the oracle answer is (w_C^1, w_C^2), $\mathcal{A}$ evaluates $C^f(w_C^1)$ and $C^f(w_C^2)$. We refer to the new f-query involved in the evaluation of $C^f(w_C^1)$ and $C^f(w_C^2)$ as ***confirmation queries***.

Let CFHit denote the event that $\mathcal{A}$ outputs a 3-collision while evaluating confirmation queries.

Case I ($\mathsf{CollFound} \wedge \neg\mathsf{CFHit}$**): Final collision does not happen in confirmation queries.** We show that the event $\mathsf{CollFound} \wedge \neg\mathsf{CFHit}$ can be utilized to construct a randomized compression scheme of f. Since f is a random function, it cannot be compressed on average, and hence the probability of this case is negligible.

Case II (CFHit**): Final collision happens in confirmation queries.** This case is more involved, and it is where the previous proof techniques fail. Let CFHit_i denote the event that a $(K+1)$-collision is found when $\mathcal{S}^{f,\mathsf{CF}^{f,\mathrm{r}}}(1^\lambda)$ is evaluating the confirmation queries right after the i-th CF-query C. Clearly,

$$\mathbf{Pr}\left[\mathsf{CFHit}\right] = \sum_{i\in[q]} \mathbf{Pr}\left[\mathsf{CFHit}_i\right],$$

where q is an upper bound of the running time of $\mathcal{A}$.

Fix $i \in [q]$. Let (w_C^1, w_C^2) denote the oracle answer of $\mathsf{CF}^f(C)$ and Q_i is the set of f-queries issued by $\mathcal{A}$ before the i-th CF-query; and let $\mathtt{Query}^f(C, w_C^j)$ denote the set of f-queries made for evaluating $C^f(w^j)$. Previous technique fail to exclude the possibility that the final 3-collision consists of one f-query from Q_i, one from $\mathtt{Query}^f(C, w_C^1)$, and one from $\mathtt{Query}^f(C, w_C^2)$.

We decompose the event CFHit_i as

$$\mathsf{CFHit}_i \subseteq \bigcup_{j\in[2]} \mathsf{FindSib}_{i,j} \cup \bigcup_{j\in[2]} \mathsf{FreshColl}_{i,j}, \tag{1}$$

where the events $\mathsf{FindSib}_{i,j}, \mathsf{FreshColl}_{i,j}$ are defined as follows.

- $\mathsf{FindSib}_{i,j}$: $\mathtt{Query}^f(C, w_C^j) \cap \mathtt{Sib}(f, Q_i) \neq \emptyset$, where $\mathtt{Sib}(f, Q_i) = \bigcup_{z\in Q_i} f^{-1}(f(z)) \setminus Q_i$ denotes the f-siblings of Q_i. That is, confirmation queries of C collide with previous queries.
- $\mathsf{FreshColl}_{i,j}$: $\exists x, x' \in \mathtt{Query}^f(C, w_C^j) \setminus Q_i$ s.t.(x, x') is a 2-collision for f. That is, there is a 2-collision for f in the new f-queries induced by evaluating $C^f(w_C^j)$.

The inclusion in eq. (1) holds because, if CFHit_i happens but $\bigcup_{j\in[2]} \mathsf{FindSib}_{i,j}$ does not happen, then the 3-collision found by $\mathcal{A}$ must come from

$$\bigcup_{j\in[2]} \mathtt{Query}^f(C, w_C^j) \setminus Q_i.$$

By the pigeonhole principle, there exists some $j \in [2]$ such that $\mathtt{Query}_\lambda(C, w_C^j) \setminus Q_i$ contains a 2-collision for f, meaning that $\bigcup_{j\in[2]} \mathsf{FreshColl}_{i,j}$ happens.

It suffices to bound $\mathbf{Pr}\left[\mathsf{FindSib}_{i,j}\right]$, $\mathbf{Pr}\left[\mathsf{FreshColl}_{i,j}\right]$ for all $i \in [q]$, $j \in [2]$.

Random oracle games with adaptive leakage. For now, let us fix all other randomness (including that of CF) and only consider the random choice of f. When $\mathcal{A}$ issues the i-th CF-query C, it has gathered information on the random function f from two sources: (i) direct queries to f, i.e., $(Q_i, Y = f(Q_i))$, and (ii) oracle answers return by CF^f, which can be viewed as *global leakage on* f. That is, we are in a setting where the adversary has both leakage and query access to the random function, which we call *random oracle games with adaptive leakage*. We then analyze random oracle games with adaptive leakage using two techniques: *density-restoring partition* and *smoothening*.

Density-restoring partition. Using density-restoring partition, the distribution of f conditioned on the view of $\mathcal{A}$ (until the i-th CF-query C is issued) can be decomposed as a convex combination of flat distributions $\mathcal{D}_1, \ldots, \mathcal{D}_r$:

$$f|_{\text{view of } \mathcal{A}} = \sum_{\sigma=1}^{r} p_\sigma \mathcal{D}_\sigma \text{ where } \sum_{\sigma} p_\sigma = 1, 0 < p_\sigma < 1. \tag{2}$$

Here, each $\mathcal{D}_\sigma$ is a more structured flat distribution with the following properties.

- There exists a set $I_\sigma \subseteq \{0,1\}^\ell \setminus Q_i$ and $Y' \in (\{0,1\}^\lambda)^{I_\sigma}$ such that $\forall f \in \mathrm{supp}(\mathcal{D}_\sigma)$, $f(I_\sigma) = Y'$ (and $f(Q_i) = Y$). Moreover, $\mathcal{D}_\sigma$ is η-*dense* on the remaining coordinates, where $\eta \in (0,1)$ is parameter; that is, for all $J \subseteq \{0,1\}^\ell \setminus (Q_i \cup I_\sigma)$, the projection of $\mathcal{D}_\sigma$ on coordinates in J has min-entropy at least $\eta|J|\lambda$. See fig. 1 for an illustration.
- The expected size of I_σ, namely, $\sum_\sigma p_\sigma |I_\sigma|$, is $O(\frac{B}{1-\eta})$, where B is the amount of leakage. In our case, B is the total output length of CF^f.

This technique is used in proving communication complexity lower bounds [GPW17, CFK+19, YZ24, HMW+25, MYZ24], as well as in studies of the random oracle model with auxiliary input [CDGS18, DFMT20].

Let us analyze $\mathbf{Pr}\left[\mathsf{FreshColl}_{i,j}\right]$ as an example. Instead of choosing f at random in the beginning, it is equivalent to proceed as follows: We first sample the view of $\mathcal{A}$ (until the i-th CF query C is issued); then, using eq. (2), conditioned on the view of $\mathcal{A}$, we first pick $\sigma \leftarrow [r]$ with probability p_σ and then sample $f \leftarrow \mathcal{D}_\sigma$. Intuitively, if $\mathtt{Query}^f(C, w_C^j)$ is contained in the 'dense part' of $\mathcal{D}_\sigma$, it is unlikely to contain a 2-collision. In other words, if

$$\left(\bigcup_{j\in[2]} \mathtt{Query}^f(C, w_C^j) \setminus Q_i\right) \cap I_\sigma = \emptyset, \tag{3}$$

then $\mathsf{FreshColl}_{i,j}$ happens with tiny probability over the choice of $f \leftarrow \mathcal{D}_\sigma$.

Smoothening. It remains to show that the event in eq. (3) happens with overwhelming probability. To this end, we modify $\mathcal{A}$ as follows: When issuing a CF-query C, before querying CF, it first samples $w_1, \ldots, w_\beta$ and evaluates $C^f(w_1), \ldots, C^f(w_\beta)$. This procedure is called smoothening, which

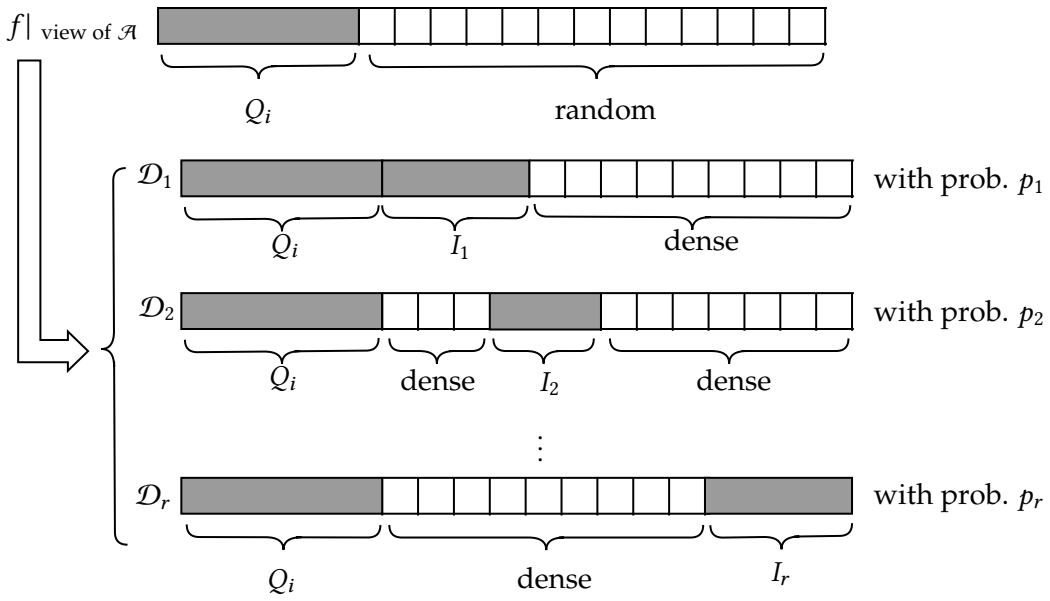

Figure 1: Shaded coordinates have fixed values. The distribution of f conditioned on the view of $\mathcal{A}$ assigns fixed values to coordinates in Q_i, since $\mathcal{A}$ queried Q_i and its view contains $Y = f(Q_i)$. It can be decomposed as $f|_{\text{view of } \mathcal{A}} = \sum_{\sigma=1}^{r} p_\sigma \mathcal{D}_\sigma$, where each distribution $\mathcal{D}_\sigma$ additionally fixes values on coordinates in I_σ, and is dense on other coordinates $\{0,1\}^\ell \setminus (Q_i \cup I_\sigma)$.

is used in the proofs of several separation results [AS16, BD19, DM24]. Call $x \in \{0,1\}^\ell$ a *γ-heavy query* if $\mathbf{Pr}_w \left[x \in \mathtt{Query}^f(C, w)\right] > \gamma$ where γ is parameter. For all γ-heavy query x, it holds that

$$\Pr_{w_1,\ldots,w_\beta}[x \notin Q_i] < (1-\gamma)^\beta < 2^{-\gamma\beta}.$$

That is, after smoothening, all γ-heavy queries will end up in Q_i with overwhelming probability (over the choice of $w_1, \ldots, w_\beta$). Note that $I_\sigma \subseteq \{0,1\}^\ell \setminus Q_i$, thus all $x \in I_\sigma$ are not heavy. Finally, since the marginal distribution of w_C^j is uniform, we conclude that for $j \in [2]$,

$$(\mathtt{Query}(C, w_C^j) \setminus Q_i) \cap I_\sigma \neq \emptyset$$

happens with a tiny probability (with proper setting of parameters). This implies that the event in eq. (3) happens with overwhelming probability.

1.3 Discussion and Related Work

The case where $K = \omega(1)$. If K is a superconstant, the event inclusion in eq. (1) no longer holds: When $\mathcal{A}^{f,\mathsf{CF}^f}$ is trying to find a $K(\lambda)$-collision for f, it could issue a CF-query C with input length $m \gg \lambda$, and $\mathsf{CF}^f(C)$ will return a $K(m)$-collision for C^f where $K(m) > K(\lambda)$. In this case, it is possible that CFHit_i happens but none of $\{\mathsf{FindSib}_{i,j}, \mathsf{FreshColl}_{i,j}\}_{j\in[K(m)]}$ happens. Moreover, when $K = \omega(1)$, it seems hard to generalize Simon's oracle to return a K-collision with non-negligible probability while retaining the property that the marginal distribution of each component of the K-collision returned is uniform—this is a crucial property that allows us to use smoothening. We note

that the weaker black-box separation in [JLRX24], which rules out black-box many-one reductions, applies to $K = \omega(1)$. It remains an open question whether there is still a black-box separation when $K = \omega(1)$.

Constructions of MCRH under generic assumptions. Another aspect is to explore under which assumptions MCRHs can be constructed. Along this line, Berman, Degwekar, and Rothblum [BDRV18] showed that for large K's ($K = \Omega(n^2)$ where n is the input length), K-MCRH can be based on the average-case hardness of a problem in NISZK; the problem is similar to *entropy approximation*, the complete problem for the class NISZK, but their exact relation is unclear.

2 Preliminary

Notations. $[N] \stackrel{\text{def}}{=} \{1, 2, \ldots, N\}$. We use $\leftarrow$ to denote sampling from a distribution, choosing an element from a set uniformly at random, or collecting the output of a randomized algorithm. Particularly, for a set X, we may use $\boldsymbol{X}$ to denote the random variable uniformly distributed over the set X.

For a function $\nu : \mathbb{N} \to [0, 1]$, we write $\nu = \mathsf{negl}(\lambda)$ if for every $c \in \mathbb{N}$, $\nu(\lambda) \leq 1/(c\lambda^c)$ for sufficiently large λ. $\mathtt{Perm}(U)$ denotes the set of all permutations over set U. We often equate the set $\{0,1\}^\lambda$ with $[2^\lambda]$.

Partial functions. We view a partial function as a map of the form

$$f : \{0,1\}^* \to \{0,1\}^* \cup \{\bot\},$$

where the symbol $\bot$ indicates that the value of f is undefined on a given input. Equivalently, a partial function can be represented as a set of pairs (x, y) with distinct first components, meaning that the function value at input x is y. Define

$$\mathrm{supp}(f) \stackrel{\text{def}}{=} \{\, x : \exists y \ (x, y) \in f \,\}.$$

The update of f with an assignment (x, y) is expressed as

$$f \coloneqq f \cup \{(x, y)\},$$

which extends f by setting $f(x) \coloneqq y$ (provided that $f(x) = \bot$ before the update). For two partial functions f, f', we write $f \subseteq f'$ to denote that f is consistent with f', i.e., $\mathrm{supp}(f) \subseteq \mathrm{supp}(f')$ and $f(x) = f'(x)$ for all $x \in \mathrm{supp}(f)$.

2.1 Multi-Collision-Resistant Hash Function (MCRH)

Definition 5 (Collisions and siblings)**.** Let $f : \mathcal{X} \to \mathcal{Y}$ be a function. For integer $K \geq 2$, define the set of K-collisions for f as

$$\mathtt{Coll}_K(f) \stackrel{\text{def}}{=} \left\{(x_1, \ldots, x_K) \in \mathcal{X}^K : x_1, \ldots, x_K \text{ are distinct and } f(x_1) = \cdots = f(x_K)\right\}.$$

For a set $Q \subseteq \mathcal{X}$, define the set of siblings as

$$\mathtt{Sib}(f, Q) \stackrel{\text{def}}{=} \bigcup_{x \in Q} f^{-1}(f(x)) \setminus Q.$$

Definition 6 (K-MCRH)**.** Let $\ell_{\mathsf{in}} : \mathbb{N} \to \mathbb{N}$ be a length function and $K = K(n) \geq 2$ be an integer parameter. A ***keyed function*** with input length ℓ_{in} is a collection $\mathcal{H} = \{\mathcal{H}_n\}_{n\in\mathbb{N}}$ where $\mathcal{H}_n = \left\{h_{\mathsf{hk}} : \{0,1\}^{\ell_{\mathsf{in}}(n)} \to \{0,1\}^n\right\}_{\mathsf{hk}\in\mathcal{K}_n}$ is a family of functions with key space $\mathcal{K}_n$. We use the shorthand $h \leftarrow \mathcal{H}_n$ to denote sampling $\mathsf{hk} \leftarrow \mathcal{K}_n$ and set $h := h_{\mathsf{hk}}$. $\mathcal{H}$ is said to be a ***K-multi-collision resistant hash function*** if it enjoys the following properties.

- Shrinkage. $\ell_{\mathsf{in}}(n) \geq n + \lceil \log(K(n) - 1) \rceil + 1$.
- K-collision resistance. For every PPT adversary $\mathcal{A}$, there exists a negligible function ν such that
$$\Pr_{\substack{h\leftarrow\mathcal{H}_n \\ (x_1,\ldots,x_K)\leftarrow\mathcal{A}(h)}} [(x_1,\ldots,x_K) \in \mathtt{Coll}_K(h)] \leq \nu(n).$$

Definition 7 (Fully black-box construction of K-MCRH from $(K+1)$-MCRH)**.** Let $m = m(n)$ and $\ell = \ell(\lambda)$ be length functions. A (q,ε)-fully black-box construction of K-MCRH with input length m from $(K+1)$-MCRH is an oracle-aided keyed function $\mathcal{H} = (\mathcal{H}_n)_{n\in\mathbb{N}}$ and a oracle-aided reduction $\mathcal{R}$ with the following properties.

- Syntax. For every $n \in \mathbb{N}$, $\mathcal{H}_n$ is a family of oracle-aided circuits [2] $h^{(\cdot)}$ with input length $m(n)$ and output length n.
- Black-box reduction. For any oracle $f = \left\{f_\lambda : \{0,1\}^{\ell(\lambda)} \to \{0,1\}^\lambda\right\}_{\lambda\in\mathbb{N}}$ and any oracle-aided (probabilistic) algorithm Ψ, if Ψ breaks the K-collision-resistance of $\mathcal{H}$, i.e.,
$$\Pr_{h\leftarrow\mathcal{H}_n,(w_1,\ldots,w_K)\leftarrow\Psi^f(h)} \left[(w_1,\ldots,w_K) \in \mathtt{Coll}_K(h^f)\right] \geq \frac{1}{n} \text{ for infinitely many } n \in \mathbb{N},$$
then the reduction $\mathcal{R}^{f,\Psi}$ breaks the $(K+1)$-collision-resistance of f, namely,
$$\mathbf{CollAdv}^{f,K+1}_{\mathcal{R}^{f,\Psi^f}}(\lambda) \stackrel{\mathsf{def}}{=} \Pr_{(x_1,\ldots,x_{K+1})\leftarrow\mathcal{R}^{f,\Psi^f}(1^\lambda)} [(x_1,\ldots,x_{K+1}) \in \mathtt{Coll}_{K+1}(f)] \geq \varepsilon(\lambda)$$
for infinitely many $\lambda \in \mathbb{N}$.
- Reduction efficiency. For all oracle f and $\lambda \in \mathbb{N}$, $\mathcal{R}^{f,\Psi}$ makes at most $q(\lambda)$ queries to f and Ψ. For every Ψ-query h made by $\mathcal{R}^{f,\Psi}(1^\lambda)$, the size of the oracle circuit $h^{(\cdot)}$ is most $q(\lambda)$; in particular, the input length of h is at most $q(\lambda)$ and h^f makes at most $q(\lambda)$ queries to f on any input.

2.2 Density-Restoring Partition

Min-entropy and dense distribution. For a random variable $\boldsymbol{X}$, we use $\mathrm{supp}(\boldsymbol{X})$ to denote the support of $\boldsymbol{X}$.

[2] We equate the hash key and the circuit computing the function.

Definition 8 (Min-entropy and deficiency)**.** The min-entropy of a random variable $\boldsymbol{X}$ is defined by

$$\mathbf{H}_\infty(\boldsymbol{X}) := \min_{x \in \mathrm{supp}(\boldsymbol{X})} \log\left(\frac{1}{\mathbf{Pr}[\boldsymbol{X} = x]}\right).$$

Suppose that $\boldsymbol{X}$ is supported on $[n]^J$. We define the *deficiency* of $\boldsymbol{X}$ as

$$\mathbf{D}(\boldsymbol{X}) := |J| \log n - \mathbf{H}_\infty(\boldsymbol{X}).$$

For $I \subseteq J$, $x \in [n]^J$, let $x(I) \stackrel{\mathsf{def}}{=} (x(i))_{i \in I} \in [n]^I$ be the projection of x on coordinates in I.

Definition 9 (Dense distribution)**.** Let $\gamma \in (0,1)$. A random variable $\boldsymbol{X}$ supported on $[n]^J$ is said to be η-*dense* if for all nonempty $I \subseteq J$, $\mathbf{H}_\infty(\boldsymbol{X}(I)) \geq \eta |I| \log n$.

Definition 10. For a subset $X \subseteq [N]^M$, we say X is fixed on $I \subset [M]$ if there exists an $\beta \in [N]^I$ such that $\forall x \in X, x(I) = \beta$.

The following lemma is the crux of the structure-vs-pseudorandomness method in [GPW17] used for proving communication lower bounds. It essentially says that a flat random variable could be decomposed into a convex combination of flat random variables with disjoint support and dense properties.

Lemma 11 (Density-restoring partition)**.** *Let $\eta \in (0,1)$. Let X be a subset of $[N]^M$ and $J \subseteq [M]$. Suppose that X is fixed on $\overline{J}$. Then, there exists a partition $X = X^1 \cup X^2 \cup \cdots \cup X^r$ and every X^i is associated with a set $I_i \subseteq J$ and a value $\alpha_i \in [N]^{I_i}$ that satisfy the following properties.*

1. $\forall x \in X^i, x(I_i) = \alpha_i$;
2. $\boldsymbol{X}^i(J \setminus I_i)$ *is η-dense;*
3. $\mathbf{D}\left(\boldsymbol{X}^i(J \setminus I_i)\right) \leq \mathbf{D}\left(\boldsymbol{X}(J)\right) - (1-\eta)|I_i| \cdot \log N + \delta_i$, *where* $\delta_i \stackrel{\mathsf{def}}{=} \log(|X|/|\cup_{j \geq i} X^j|)$.

Remark 12. Jumping ahead, in our use cases, X is a set of functions from $\{0,1\}^\ell$ to $\{0,1\}^\lambda$; that is, we equate the set $[2^\lambda]$ with $\{0,1\}^\lambda$ and apply the above lemma with $N = 2^\lambda, M = 2^\ell$, and $J \subseteq [2^\ell]$.

2.3 Randomized Compression Scheme

Let $\mathcal{X}$ be a finite set. A randomized compression scheme for $\mathcal{X}$ consists of two functions, namely, an encoding algorithm $\mathsf{Enc} : \mathcal{R} \times \mathcal{X} \to \mathcal{Y} \cup \{\perp\}$, and decoding algorithm $\mathsf{Dec} : \mathcal{Y} \to \mathcal{X}$, where $\mathcal{R}$ is the randomness space. It is required that for all $r \in \mathcal{R}, x \in \mathcal{X}$ such that $\mathsf{Enc}(r,x) \neq \perp$, we have $\mathsf{Dec}(r, \mathsf{Enc}(r,x)) = x$.

Lemma 13. *For all randomized compression schemes for $\mathcal{X}$, it holds that*

$$|\mathcal{Y}| \geq \delta |\mathcal{X}|,$$

where $\delta := \mathbf{Pr}_{r \leftarrow \mathcal{R}, x \leftarrow \mathcal{X}}[\mathsf{Enc}(r,x) \neq \perp]$.

Proof. For $r \in \mathcal{R}$, define

$$\delta_r \stackrel{\text{def}}{=} \Pr_{x \leftarrow \mathcal{X}}[\mathsf{Enc}(r, x) \neq \perp].$$

Since $\delta = \mathbf{E}_{r \leftarrow \mathcal{R}}[\delta_r]$, there exists some r^* such that $\delta_{r^*} \geq \delta$. Let

$$\mathcal{X}^* \stackrel{\text{def}}{=} \{x \in \mathcal{X} : \mathsf{Enc}(r^*, x) \neq \perp\}.$$

Note that $\mathsf{Dec}(r^*, \mathsf{Enc}(r^*, x)) = x$ for all $x \in \mathcal{X}^*$, meaning that $x \mapsto \mathsf{Enc}(r^*, x)$ is an injection from $\mathcal{X}^*$ to $\mathcal{Y}$. Consequently,

$$|\mathcal{Y}| \geq |\mathcal{X}^*| = \delta_{r^*}|\mathcal{X}| \geq \delta|\mathcal{X}|.$$

□

2.4 Technical Tools

Lemma 14 (Borel-Cantelli Lemma)**.** *Let $\mathsf{E}_1, \mathsf{E}_2, \ldots$ be a sequence of events on the same probability space. Then, if the sum of probabilities of events E_λ converges, then the probability that infinitely many of the events occur is 0:*

$$\sum_{\lambda=1}^{\infty} \mathbf{Pr}[\mathsf{E}_\lambda] < \infty \implies \mathbf{Pr}\left[\bigcap_{k=1}^{\infty} \bigcup_{\lambda=k}^{\infty} \mathsf{E}_\lambda\right] = 0.$$

3 Separating $(K+1)$-MCRH from K-MCRH

This section presents the main result: For every constant $K \geq 2$, there is no black-box construction of K-MCRH from $(K+1)$-MCRH. We start by stating the main result formally:

Theorem 15. *Let $K \geq 2$ be an integer constant, and let $m = m(n), \ell = \ell(\lambda)$ be length functions such that $m(n) \geq n + \lceil \log K \rceil + 1$ and $\ell(\lambda) \geq \lambda + \lceil \log(K+1) \rceil + 1$. Let $(\mathcal{H}, \mathcal{R})$ be a (q, ε)-fully black-box construction of K-MCRH with input length m from $(K+1)$-MCRH with input length ℓ. Then, either*

- *(large reduction time) $q(\lambda) \geq 2^{0.01\lambda}$ for infinitely many λ, or*
- *(large security loss) there exists a constant c such that $\varepsilon(\lambda) \leq \mathsf{c} \cdot 2^{-0.01\lambda}$ for infinitely many λ.*

We prove the theorem by presenting two oracles f and Ψ such that (1) Ψ^f breaks any K-MCRH construction $\mathcal{H}$ that uses f as oracle, and (2) f is a $(K+1)$-MCRH relative to Ψ.

Notations. We start with fixing notations used throughout our proof.

- Fix length functions $\ell = \ell(\lambda), m = m(n)$ and a constant K.
- $\mathcal{F} = \{\mathcal{F}_\lambda\}_{\lambda \in \mathbb{N}}$, where $\mathcal{F}_\lambda$ is the set of functions mapping $\ell(\lambda)$-bit inputs to λ-bit outputs. We use $f \leftarrow \mathcal{F}$ to indicate that $f = (f_\lambda)_{\lambda \in \mathbb{N}}$ is obtained by sampling $f_\lambda \leftarrow \mathcal{F}_\lambda$ for each λ.
- $\mathfrak{C} = \{\mathfrak{C}_n\}_{n \in \mathbb{N}}$, where $\mathfrak{C}_n$ is the set of all oracle-aided circuits mapping $m(n)$-bit inputs to n-bit outputs.
- For $C \in \mathfrak{C}_n, f \in \mathcal{F}$, let $\mathtt{Query}^f(C, w)$ be the set of f-queries when evaluating C^f on input w, and write
$$\mathtt{Query}^f_\lambda(C, w) \stackrel{\text{def}}{=} \left\{x \in \mathtt{Query}^f(C, w) : x \text{ is a query to } f_\lambda\right\}$$
for $\lambda \in \mathbb{N}$.

3.1 Simon's Oracle

Next, we describe a generalization of Simon's oracle, which takes as input an oracle-aided circuit C and returns a K-collision for the function C^f with constant probability.

Generalized Simon's collision-finding oracle CF^f

- Internal randomness: For every $n \in \mathbb{N}$ and circuit $C \in \mathfrak{C}_n$, choose $w_C^1 \leftarrow \{0,1\}^{m(n)}$, $\pi_C^2, \pi_C^3, \ldots, \pi_C^{K(n)} \leftarrow \mathtt{Perm}(\{0,1\}^m)$ uniformly at random.
- Input: oracle-aided circuit $C \in \mathfrak{C}_n$ for some $n \in \mathbb{N}$.
- Operations:
  1. For $j = 2, 3, \ldots, K$,
     (a) find the smallest index $i_j \in [2^{m(n)}]$ such that $C^f(\pi_C^j(i_j)) = C^f(w_C^1)$;
     (b) set $w_C^j := \pi_C^j(i_j)$.
  2. Return $(w_C^1, \ldots, w_C^K)$.

Lemma 16 (CF breaks K-CRH)**.** *Let f be any fixed oracle and $C^{(\cdot)} : \{0,1\}^m \to \{0,1\}^n$ be any candidate construction of K-CRH. If $m \geq n + \lceil \log K \rceil + 1$. Then,*

$$\Pr_{(w_C^1,\ldots,w_C^K)\leftarrow \mathsf{CF}^f(C)} \left[(w_C^1, \ldots, w_C^K) \in \mathtt{Coll}_K(C^f)\right] \geq \frac{1}{2K^{K-1}}.$$

Proof. Omit f from notations as it is fixed. The choice of $w_C^1, \cdots, w_C^K$ guarantees that they evaluate to the same value under C^f, and thus they form a K-collision as long as they are distinct. Write $a_y := |C^{-1}(y)|$ for $y \in \mathtt{Im}(C)$ and $A \stackrel{\mathsf{def}}{=} \{y \in \mathtt{Im}(C) : a_y \geq K\}$. Then,

$$\begin{aligned}
&\Pr_{(w_C^1,\ldots,w_C^K)\leftarrow \mathsf{CF}(C)} \left[w_C^1, \cdots, w_C^K \text{ are distinct}\right] \\
&\geq \sum_{y\in A} \Pr_{w\leftarrow\{0,1\}^m}[C(w) = y] \Pr_{w_C^1,\ldots,w_C^K \leftarrow C^{-1}(y)} \left[w_C^1, \cdots, w_C^K \text{ are distinct}\right] \\
&= \sum_{y\in A} \frac{a_y}{2^m} \cdot \frac{a_y(a_y-1)\cdots(a_y-K+1)}{a_y^K}. \qquad (4)
\end{aligned}$$

Observe that for $y \in A$,

$$\frac{a_y(a_y-1)\cdots(a_y-K+1)}{a_y^K} \geq \frac{(a_y-K+1)^{K-1}}{a_y^{K-1}} = \left(1 - \frac{K-1}{a_y}\right)^{K-1} \geq \frac{1}{K^{K-1}}.$$

where the second inequality holds since $a_y \geq K$. Meanwhile,

$$\sum_{y\in A} \frac{a_y}{2^m} = 1 - \sum_{y\in \mathtt{Im}(C)\setminus A} \frac{a_y}{2^m} \geq 1 - 2^n \cdot \frac{K}{2^m} \geq \frac{1}{2},$$

where the last inequality follows from $m \geq n + \lceil \log K \rceil + 1$. Plugging the above two equations into eq. (4), we conclude that

$$\Pr_{(w_C^1,\ldots,w_C^K)\leftarrow \mathsf{CF}(C)} \left[w_C^1, \cdots, w_C^K \text{ are distinct} \right] \geq \frac{1}{2K^{K-1}}.$$

□

We also define a variant of Simon's oracle as shown in fig. 2, which is used in later proofs. It uses a partial function f' as oracle, and whenever it encounters an undefined input of f' during execution, it outputs $\perp$.

Simon's oracle $\widetilde{\mathsf{CF}}^{f'}$ with a partial function f'

- Internal randomness: For every $n \in \mathbb{N}$ and circuit $C \in \mathfrak{C}_n$, choose $w_C^1 \leftarrow \{0,1\}^{m(n)}$, $\pi_C^2, \pi_C^3 \ldots, \pi_C^K \leftarrow \mathsf{Perm}(\{0,1\}^m)$ uniformly at random.
- Input: oracle-aided circuit $C \in \mathfrak{C}_n$.
- Operations:
  1. If $\mathsf{EvalC}^{f'}(C, w_C^1) = \perp$, return $\perp$.
  2. For $j = 2, 3, \ldots, K$,
     (a) find the smallest index $i_j \in [2^{m(n)}]$ such that $\mathsf{EvalC}^{f'}(C, \pi_C^j(i_j)) = \mathsf{EvalC}^{f'}(C, w_C^1)$; if no such index exists, return $\perp$.
     (b) set $w_C^j := \pi_C^j(i_j)$.
  3. Return $(w_C^1, \ldots, w_C^K)$.

Subroutine $\mathsf{EvalC}^{f'}(C, w)$:

- Evaluate $C^{(\cdot)}$ on input w by answering the oracle gates with f'. If some oracle gate queries an undefined point, return $\perp$; otherwise, return the evaluation result.

Figure 2: Simon's oracle with a partial function f'

A few observations about CF and $\widetilde{\mathsf{CF}}$ are in order.

Observation 17. CF and $\widetilde{\mathsf{CF}}$ have the same randomness space, denoted $\mathtt{Coin}^\star$. Given oracle f and $\mathbf{r} \in \mathtt{Coin}^\star$, we write $\mathsf{CF}^{f,\mathbf{r}}$ and $\widetilde{\mathsf{CF}}^{f,\mathbf{r}}$ to denote the CF-oracles with fixed randomness $\mathbf{r}$.

Lemma 18. *Fix $f \in \mathcal{F}, \mathrm{r} \in \mathtt{Coin}^\star, C \in \mathfrak{C}$ and let $(w_C^1, \ldots, w_C^K) := \mathsf{CF}^{f,\mathrm{r}}(C)$. Let f' be a partial function. If $f' \subseteq f$ and $\mathtt{Query}^f(C, w_C^j) \subseteq \mathrm{supp}(f')$ for all $j \in [K]$, then $\widetilde{\mathsf{CF}}^{f',\mathrm{r}}(C) = (w_C^1, \ldots, w_C^K)$.*

Proof. Since $f' \subseteq f$ and $\mathtt{Query}^f(C, w_C^j) \subseteq \mathrm{supp}(f')$, we have $\mathsf{EvalC}^{f'}(C, w_C^j) = C^f(w_C^j)$ for all $j \in [K]$.

Fix $j \in [K]$. By the choice of w_C^j in $\mathsf{CF}^{f,\mathrm{r}}(C)$, we know that $w_C^j = \pi_C^j(i_j)$, where i_j be the smallest index such that $C^f(w_C^1) = C^f(\pi_C^j(i_j))$. For $i' < i_j$, it must be that $\mathsf{EvalC}^{f'}(C, \pi_C^j(i')) \neq \mathsf{EvalC}^{f'}(C, w_C^1)$, because otherwise we will have

$$C^f(\pi(i')) = \mathsf{EvalC}^{f'}(C, \pi_C^j(i')) = \mathsf{EvalC}^{f'}(C, w_C^1) = C^f(w_1),$$

contradicting the minimality of i_j. Therefore, $\widetilde{\mathsf{CF}}^{f,\mathrm{r}}(C)$ will also choose w_C^j as the j-th component of its output. □

3.2 Proof of Main Result

Our goal is to show that CF^f does not help to find a $(K+1)$-collision for some f. We shall prove this for a random $f \leftarrow \mathcal{F}$ and then use a standard argument to show the existence of a fixed f with the desired property.

Definition 19 ((q, q', q'')-bounded adversary)**.** We say an oracle-aided adversary $\mathcal{A}$ is (q, q', q'')-bounded if for any fixed oracle f, $\mathcal{A}^{f,\mathsf{CF}^f}(1^\lambda)$ issues at most $q(\lambda)$ queries to f, $q'(\lambda)$ queries to CF, and each CF-query C has size at most $q''(\lambda)$.

Theorem 20. *For all (q, q, q)-bounded adversary $\mathcal{A}$ with $q(\lambda) = O(2^{0.01\lambda})$,*

$$\mathop{\mathbf{E}}_{f \leftarrow \mathcal{F}}\left[\mathbf{CollAdv}^{f,K+1}_{\mathcal{A}^{f,\mathsf{CF}^f}}(\lambda)\right] = O(2^{-0.01\lambda}).$$

Now we prove the main result (theorem 15) using theorem 20. The proof of theorem 20 is given in the rest of the paper.

Proof of theorem 15. Fix a (q, ε)-fully black-box construction $(\mathcal{H}, \mathcal{R})$. If $q(\lambda) \geq 2^{0.01\lambda}$ for infinitely many λ, we are done. Suppose that $q(\lambda) \leq 2^{0.01\lambda}$ for sufficiently large λ. Since $\mathcal{R}$ is a (q, q, q)-bounded adversary, by theorem 20, we have there exists a function $\delta(\lambda) = O(2^{-0.01\lambda})$ such that

$$\mathop{\mathbf{E}}_{f}\left[\mathbf{CollAdv}^{f,K+1}_{\mathcal{R}^{f,\mathsf{CF}^f}}(\lambda)\right] \leq \delta(\lambda)$$

for sufficiently large λ. Let $\varepsilon'(\lambda) \stackrel{\mathsf{def}}{=} \lambda^2 \cdot \delta(\lambda)$, and let E_λ denote the event that

$$\mathbf{CollAdv}^{f,K+1}_{\mathcal{R}^{f,\mathsf{CF}^f}}(\lambda) \geq \varepsilon'(\lambda).$$

By Markov inequality, $\mathbf{Pr}_f[\mathsf{E}_\lambda] < \frac{1}{\lambda^2}$ for sufficiently large λ. Since $\sum_{\lambda=1}^{\infty} \frac{1}{\lambda^2} < \infty$, we have $\sum_{\lambda=1}^{\infty} \mathbf{Pr}_f[\mathsf{E}_\lambda] < \infty$. Then by Borel-Cantelli lemma (lemma 14), we have

$$\mathop{\mathbf{Pr}}_{f}\left[\mathbf{CollAdv}^{f,K+1}_{\mathcal{R}^{f,\mathsf{CF}^f}}(\lambda) \geq \varepsilon'(\lambda) \text{ for infinitely many } \lambda\right] = 0.$$

Therefore, there exists a some $f \in \mathcal{F}$ such that $\mathbf{CollAdv}^{f,K+1}_{\mathcal{R}^{f,\mathsf{CF}^f}}(\lambda) < \varepsilon'(\lambda)$ for sufficiently large λ. Fix such an f in the following argument.

By lemma 16, CF^f breaks the K-collision resistance of $\mathcal{H}$ with probability $\frac{1}{2K^{K-1}}$, which is a constant as K is a constant. By the property of black-box reduction,

$$\mathbf{CollAdv}^{f,K+1}_{\mathcal{R}^{f,\mathsf{CF}^f}}(\lambda) > \varepsilon(\lambda)$$

for infinitely many λ, and thus $\varepsilon(\lambda) < \varepsilon'(\lambda)$ for infinitely many λ. Note that $\varepsilon'(\lambda) = O(2^{0.01\lambda})$, meaning that there exists a constant c such that $\varepsilon'(\lambda) < \mathsf{c} \cdot 2^{0.01\lambda}$ for sufficiently large λ, and hence $\varepsilon(\lambda) < \mathsf{c} \cdot 2^{0.01\lambda}$ for infinitely many λ. □

3.3 Normalizations

We proceed to prove theorem 20. For a start, we introduce the notion of normalized versions of an adversary. Intuitively, the normalized adversary issues some fixed f-queries in the very beginning and then simulates $\mathcal{A}$. In the simulation of CF-queries of $\mathcal{A}$, we incorporate a technique called *smoothening*, which is used in several separation results [AS16, BD19, DM24].

Definition 21 ((β, β')-normalized adversary)**.** Let $\mathcal{A}$ be an oracle-aided adversary that, on input 1^λ, aims to find a $(K+1)$-collision for f_λ with oracle access to (f, CF^f). For $\beta, \beta' \in \mathbb{N}$, we construct another oracle-aided adversary $\mathcal{S}$, called the (β, β')-***normalized version of*** $\mathcal{A}$: $\mathcal{S}(1^\lambda)$ simulates $\mathcal{A}(1^\lambda)$ with the following modifications.

- In the beginning, $\mathcal{S}$ first make β' f-queries $\zeta_1, \ldots, \zeta_{\beta'}$, which are the lexicographically first β' strings in $\{0,1\}^{\ell(\lambda)}$.
- Rush winning rule. After each f-query, $\mathcal{S}$ immediately checks whether a $(K+1)$-collision for f_λ is found; if found, it immediately returns the $(K+1)$-collision and halts.
- No repeating queries. If $\mathcal{A}$ makes a repeating query, $\mathcal{S}$ answer with the previously-known answer and does not make that query.
- When $\mathcal{A}$ issues a CF-query $C : \{0,1\}^{m(n)} \to \{0,1\}^n$, $\mathcal{S}$ proceeds as follows:
  1. Choose $w_1, \ldots, w_\beta \leftarrow \{0,1\}^{m(n)}$ uniformly at random and evaluates $C^f(w_1), \ldots, C^f(w_\beta)$; we refer to $w_1, \ldots, w_\beta$ as ***smoothening points***.
  2. Invoke $\mathsf{CF}^f(C)$ and receive oracle answer $(w_C^1, \ldots, w_C^K)$.
  3. Evaluate $C^f(w_C^1), \ldots, C^f(w_C^K)$. We call f-queries involved in these evaluations ***confirmation queries***.
  4. Forwards the answer $(w_C^1, \ldots, w_C^K)$ to $\mathcal{A}$.
- If $\mathcal{A}$ outputs $(x_1, \ldots, x_{K+1})$ and halts, then $\mathcal{S}$ evaluate $f(x_1), \ldots, f(x_{K+1})$.

It is straightforward to see that $\mathcal{S}$ outputs a $(K+1)$-collision for f whenever $\mathcal{A}$ does; thus we have

Observation 22. For all fixed oracle f, randomness $\mathtt{r} \in \mathtt{Coin}^\star$, and $\beta, \beta' \in \mathbb{N}$,

$$\mathbf{CollAdv}^{f,K+1}_{\mathcal{S}^{f,\mathsf{CF}^{f,\mathtt{r}}}}(\lambda) \geq \mathbf{CollAdv}^{f,K+1}_{\mathcal{A}^{f,\mathsf{CF}^{f,\mathtt{r}}}}(\lambda),$$

where $\mathcal{S}$ is the (β, β')-normalized version of $\mathcal{A}$.

In terms of efficiency, $\mathcal{S}$ only makes more f-queries than $\mathcal{A}$:

Observation 23. If $\mathcal{A}$ is (q, q', q'')-bounded, then its (β, β')-normalized version is $(\widetilde{q}, q', q'')$-bounded where $\widetilde{q} = \beta' + q + q' \cdot (\beta q'' + K q'') + K + 1$.

3.4 Proof of Theorem 20

Theorem (Theorem 20 restated). *For all (q, q, q)-bounded adversary $\mathcal{A}$ with $q(\lambda) = O(2^{0.01\lambda})$,*

$$\mathop{\mathbf{E}}_{f \leftarrow \mathcal{F}} \left[\mathbf{CollAdv}^{f,K+1}_{\mathcal{A}^{f,\mathsf{CF}^f}}(\lambda) \right] = O(2^{-0.01\lambda}).$$

Proof. Fix $\lambda \in \mathbb{N}$ and the oracle $f_{-\lambda} = \{f_{\lambda'}\}_{\lambda' \neq \lambda}$; we only consider the random choice of f_λ. Let $\beta = \lambda \cdot 2^{0.1\lambda}, \beta' = 2^{0.2\lambda}$. and let $\mathcal{S}$ be the (β, β')-normalized version of $\mathcal{A}$. By observation 23, $\mathcal{S}$ is $(\widetilde{q}, q, q)$-bounded where $\widetilde{q} = \beta' + q + q^2 \cdot (\beta + K) + K + 1 = O(2^{0.2\lambda})$. We use $\mathtt{Coin}_{\mathcal{S}}$ to denote the randomness space for $\mathcal{S}$.

The probability space is $\Omega \stackrel{\text{def}}{=} \mathtt{Coin}^\star \times \mathtt{Coin}_{\mathcal{S}} \times \mathcal{F}_\lambda$. Given $(\mathrm{r}, \mathsf{coin}, f_\lambda) \in \Omega$, the execution of $\mathcal{S}^{f,\mathsf{CF}^{f,\mathrm{r}}}(1^\lambda; \mathsf{coin})$ is completely determined. Let CollFound denote the event that a $(K+1)$-collision is output by $\mathcal{S}^{f,\mathsf{CF}^{f,\mathrm{r}}}(1^\lambda; \mathsf{coin})$. By observation 22 we have

$$\mathop{\mathbf{E}}_{f_\lambda \leftarrow \mathcal{F}_\lambda, \mathrm{r} \leftarrow \mathtt{Coin}^\star} \left[\mathbf{CollAdv}^{f,K+1}_{\mathcal{A}^{f,\mathsf{CF}^{f,\mathrm{r}}}}(\lambda) \right] \leq \mathop{\mathbf{E}}_{f_\lambda \leftarrow \mathcal{F}_\lambda, \mathrm{r} \leftarrow \mathtt{Coin}^\star} \left[\mathbf{CollAdv}^{f,K+1}_{\mathcal{S}^{f,\mathsf{CF}^{f,\mathrm{r}}}}(\lambda) \right] = \Pr_{(\mathrm{r},\mathsf{coin},f_\lambda) \leftarrow \Omega} [\mathsf{CollFound}] .$$

Let CFHit be the event that a $(K+1)$-collision is found when $\mathcal{S}$ is evaluating the confirmation queries. Then we have

$$\Pr_{(\mathrm{r},\mathsf{coin},f_\lambda) \leftarrow \Omega} [\mathsf{CollFound}] \leq \Pr_{(\mathrm{r},\mathsf{coin},f_\lambda) \leftarrow \Omega} [\mathsf{CollFound} \wedge \neg \mathsf{CFHit}] + \Pr_{(\mathrm{r},\mathsf{coin},f_\lambda) \leftarrow \Omega} [\mathsf{CFHit}] .$$

Therefore, the theorem follows from the following two lemmas.

Lemma 24. $\Pr_{(\mathrm{r},\mathsf{coin},f_\lambda) \leftarrow \Omega} [\mathsf{CFHit}] = O(2^{-0.01\lambda})$.

Lemma 25. $\Pr_{(\mathrm{r},\mathsf{coin},f_\lambda) \leftarrow \Omega} [\mathsf{CollFound} \wedge \neg \mathsf{CFHit}] = O(2^{-0.4\lambda})$.

□

It remains to prove the above two lemmas. The proof of lemma 24 is deferred to the next section, as new techniques are required. Next, we prove lemma 25 by a compression argument.

Proof of lemma 25. Fix $\mathrm{r} \in \mathtt{Coin}^\star$, $\mathsf{coin} \in \mathtt{Coin}_{\mathcal{S}}$, and define

$$\mathtt{GoodF}(\mathrm{r}, \mathsf{coin}) \stackrel{\text{def}}{=} \{f_\lambda \in \mathcal{F}_\lambda : (\mathrm{r}, \mathsf{coin}, f_\lambda) \in \mathsf{CollFound} \setminus \mathsf{CFHit}\} .$$

Let S be the deterministic adversary obtained by fixing the random coin of $\mathcal{S}$ to be coin. We shall prove that $|\mathtt{GoodF}(\mathrm{r}, \mathsf{coin})|$ is small for every fixed $(\mathrm{r}, \mathsf{coin})$. To this end, we devise a randomized encoding scheme for the set $\mathtt{GoodF}(\mathrm{r}, \mathsf{coin})$. The randomness used by the encoding scheme is a random permutation $\phi : [2^\ell] \to \{0,1\}^\ell$. The encoding will be successful if the permutation is good in the following sense:

Definition 26 (Good permutation). Let $V = \{\zeta_1, \ldots, \zeta_{\beta'}\}$ be the set of the lexicographically first β' strings in $\{0,1\}^{\ell(\lambda)}$. We say $\phi : [2^\ell] \to \{0,1\}^\ell$ is a good permutation w.r.t. $(\mathrm{r}, \mathsf{coin}, f)$ if the following two conditions hold:

1. Let $U = \{u_1, \cdots u_{|U|}\}$ is the set of all confirmation queries of $\mathsf{S}^{f,\mathsf{CF}^{f,\mathrm{r}}}(1^\lambda)$. The last element among $V \cup U$, in the order given by ϕ, is from V. Formally, $\mathrm{argmax}_{z \in V \cup U} \phi^{-1}(z) \in V$.

$\mathsf{Enc}(\phi, f_\lambda)$

- Input: A permutation $\phi : [2^\ell] \to \{0,1\}^\ell$ and $f_\lambda \in \mathtt{GoodF}(\mathrm{r}, \mathsf{coin})$.
- Output: $(i^*, d, L) \in [\widetilde{q}] \times [\widetilde{q}] \times \{0,1\}^{(2^\ell - 1)\lambda}$ or $\perp$.
- Operations:
  1. Return $\perp$ if ϕ is not a good permutation w.r.t. $(\mathrm{r}, \mathsf{coin}, f)$.
  2. Let $z_1, \ldots, z_d$ $(d \le q)$ be the f_λ-queries issued by $\mathsf{S}^{f, \mathsf{CF}^{f,r}}(1^\lambda)$.
  3. Let $i^* \in [d]$ be the smallest index such that $f_\lambda(z_{i^*}) = f_\lambda(z_d)$. `// Since a` $(K+1)$`-collision is found and S abides by the rush winning rule,` z_d `is a component of the` $(K+1)$`-collision.`
  4. Initialize L to be an empty list.
  5. For j from 1 to 2^ℓ, if $\phi(j) \neq z_d$, append $f_\lambda(\phi(j))$ to L.
  6. Return (i^*, d, L).

Figure 3: The encoding procedure.

2. z_d ranks last among $z_1, \ldots, z_d$ in the order given by ϕ, where $z_1, \ldots, z_d$ $(d \le \widetilde{q})$ are the f_λ-queries issued by $\mathsf{S}^{f, \mathsf{CF}^{f,r}}(1^\lambda)$. Formally, $z_d = \mathrm{argmax}_{z \in \{z_1, \ldots, z_d\}} \phi^{-1}(z)$.

The encoding procedure and decoding procedure are described in fig. 3 and fig. 4, respectively.

We have two claims regarding the compression scheme, showing that $(\mathsf{Enc}, \mathsf{Dec})$ is a randomized compression scheme for the set $\mathtt{GoodF}(\mathrm{r}, \mathsf{coin})$ with correctness $\delta = \Omega(2^{-0.2\lambda})$:

Claim 27. *For all* $f_\lambda \in \mathtt{GoodF}(\mathrm{r}, \mathsf{coin})$,

$$\Pr_{\phi}\left[\phi \textit{ is a good permutation w.r.t. } (\mathrm{r}, \mathsf{coin}, f)\right] \ge \Omega(2^{-0.2\lambda}).$$

Claim 28. *Let* $f_\lambda \in \mathtt{GoodF}(\mathrm{r}, \mathsf{coin})$. *If* ϕ *is a good permutation w.r.t.* $(\mathrm{r}, \mathsf{coin}, f_\lambda)$, *then*

$$|L| = 2^{\ell(\lambda)} - 1 \textit{ and } \mathsf{Dec}(\phi, (i^*, d, L)) = f_\lambda,$$

where $(i^*, d, L) := \mathsf{Enc}(\phi, f_\lambda)$.

We continue the proof of lemma 25 and prove the two claims later. Note that the size of the encoding space is $\widetilde{q}^2 \cdot 2^{(2^\ell - 1)\lambda}$. By lemma 13, it holds that

$$\widetilde{q}^2 \cdot 2^{(2^\ell - 1)\lambda} \ge \delta \cdot |\mathtt{GoodF}(\mathrm{r}, \mathsf{coin})|,$$

meaning that

$$|\mathtt{GoodF}(\mathrm{r}, \mathsf{coin})| \le \widetilde{q}^2 \cdot 2^{(2^\ell - 1)\lambda} \cdot O(2^{0.2\lambda}) = 2^{2^\ell \cdot \lambda} \cdot O(2^{-0.4\lambda}). \tag{5}$$

$\mathsf{Dec}(\phi, f_\lambda)$

- Input: A permutation $\phi : [2^\ell] \to \{0,1\}^\ell$ and $(i^*, d, L) \in [\widetilde{q}] \times [\widetilde{q}] \times \{0,1\}^{(2^\ell - 1)\lambda}$.
- Output: $f_\lambda \in \mathcal{F}_\lambda$.
- Operations:
  1. Initialize a partial function $Z : \{0,1\}^{\ell(\lambda)} \to \{0,1\}^\lambda \cup \{\bot\}$ undefined everywhere, and set $j := 1, x' := \bot$.
  2. For i from 1 to $d-1$:
     (a) $(\mathsf{flag}, x) := \mathsf{GetQuery}(Z, i)$;
     (b) if $\mathsf{flag} = \textsc{Fail}$, return $\bot$;
     (c) if $i = i^*$, set $x' := x$;
     (d) if $\mathsf{flag} = \textsc{UnFilled}$, then
        – while $j \le \phi^{-1}(x)$: set $Z := Z \cup \{(\phi(j), L(j))\}$, $j := j + 1$;
  3. $(\mathsf{flag}, x) := \mathsf{GetQuery}(Z, d)$.
  4. If $\mathsf{flag} = \textsc{Fail}$, return $\bot$.
  5. While $j < \phi^{-1}(x)$: set $Z := Z \cup \{(\phi(j), L(j))\}$, $j := j + 1$.
  6. $Z(x) := Z(x')$. `// Filling the d-th query.`
  7. While $j \le 2^\ell - 1$: set $Z := Z \cup \{(\phi(j+1), L(j))\}$, $j := j + 1$.
  8. Return Z. `// Z is all filled and thus is total.`

Subprodecure $\mathsf{GetQuery}(Z, i)$

1. Simulate S until it issues the i-th f-query while answering its oracle queries as follows:
   (a) Directly answer $f_{-\lambda}$-queries with $f_{-\lambda}$.
   (b) For the v-th f_λ-query x where $v < i$:
      - if $Z(x) = \bot$; return $(\textsc{Fail}, \bot)$; otherwise, set the oracle answer to be $Z(x)$.
   (c) For the i-th f_λ-query x:
      - if $Z(x) = \bot$, return $(\textsc{Unfilled}, x)$; otherwise, return $(\textsc{Filled}, x)$.
   (d) For a CF-query $C : \{0,1\}^m \to \{0,1\}^n$:
      - let $f' := f_{-\lambda} \cup Z$.
      - if $\widetilde{\mathsf{CF}}^{f',\mathrm{r}}(C) \neq \bot$, set the oracle answer to be $\widetilde{\mathsf{CF}}^{f',\mathrm{r}}(C)$; otherwise, return $(\textsc{Fail}, \bot)$.
2. Return $(\textsc{Fail}, \bot)$. `//The simulation is completed with fewer than` i `queries.`

Figure 4: The decoding procedure.

Since eq. (5) holds for all r and coin, we get

$$\begin{aligned}\Pr_{(\mathrm{r},\mathsf{coin},f_\lambda)\leftarrow\Omega}[\mathsf{CollFound}\wedge\neg\mathsf{CFHit}] &= \mathop{\mathbf{E}}_{\mathrm{r},\mathsf{coin}}\left[\Pr_{f_\lambda\leftarrow\mathcal{F}_\lambda}[f_\lambda\in\mathtt{GoodF}(\mathrm{r},\mathsf{coin})]\right]\\ &= \mathop{\mathbf{E}}_{\mathrm{r},\mathsf{coin}}[|\mathtt{GoodF}(\mathrm{r},\mathsf{coin})|/|\mathcal{F}_\lambda|]\\ &\le O(2^{-0.4\lambda}),\end{aligned}$$

where we recall that $|\mathcal{F}_\lambda| = 2^{2^\ell\cdot\lambda}$. This finishes the proof. □

We conclude this section with the proofs of claim 27 and claim 28.

Proof of claim 27. If $d \le \beta'$, i.e., a $(K+1)$-collision is found in $V = \{\zeta_1,\ldots,\zeta_{\beta'}\}$, then first item is vacuous and the thus ϕ is good with probability $1/d \ge 1/\beta' \ge 2^{-0.2\lambda}$.

Now assume $d > \beta'$, which means $z_d \notin V$. Let $U = \{u_1,\ldots,u_{|U|}\}$ be the set of all confirmation queries where $|U| \le Kq^2$. Since $(\mathrm{r},\mathsf{coin},f_\lambda)\notin\mathsf{CFHit}$, we have $z_d\notin U$. Consider the order of $V\cup U\cup\{z_d\}$ in ϕ. Again, the second item holds with probability $1/d \ge 1/\widetilde{q}$. Conditioned on the event that z_d ranks last, the ordering of $V\cup U$ is still uniformly random. Consequently, the first item happens with probability

$$\frac{|V|}{|V|+|U|} = 1-\frac{|U|}{|V|+|U|} \ge 1-\frac{Kq^2}{\beta'} \ge 1-O(2^{-0.1\lambda}).$$

Therefore, the probability that both items hold is at least

$$\frac{1}{\widetilde{q}}\cdot(1-O(2^{-0.1\lambda})) = \Omega(2^{-0.2\lambda}).$$

□

Proof of claim 28. If ϕ is a good permutation w.r.t. $(\mathrm{r},\mathsf{coin},f)$, then $\mathsf{Enc}(\phi,f_\lambda)$ will not output $\perp$ in step 1, and hence will add $f_\lambda(\phi(j))$ to L for all j except for $j=\phi^{-1}(z_d)$, meaning that L contains $(2^\ell-1)$ strings, each of length λ.

Let $z_1,\ldots,z_d$ be f-queries and U be the set of confirmation queries in the execution of $\mathsf{S}^{f,\mathsf{CF}^{f,\mathrm{r}}}(1^\lambda)$. We prove by induction the following loop invariants for the FOR loop in step 2 of $\mathsf{Dec}(\phi,f_\lambda)$: At the end of the i-th iteration,

(L1) $j = \max_{v\in[i]}\phi^{-1}(z_v)+1$;

(L2) for all $j'<j$, $Z(\phi(j'))=f_\lambda(\phi(j'))$; for all $j'\ge j$, $Z(j)=\perp$.

For the basis step, the above invariants hold trivially for $i=0$ with $j=1$ as in initialization. Now, suppose that loop invariants hold for $i=k$. Let j_i be the value of j at the end of the i-th iteration. The induction hypothesis says that, at the beginning of the $(k+1)$-th iteration, we have

(IH1) $j_k = \max_{v\in[k]}\phi^{-1}(z_v)+1$;

(IH2) for all $j'<j_k$, $Z(\phi(j'))=f_\lambda(\phi(j'))$; for all $j'\ge j_k$, $Z(\phi(j))=\perp$.

We shall show that L1 and L2 hold for $i=k+1$. We start by showing that

($\diamond$) $\mathsf{GetQuery}(Z,k{+}1)$ successfully reaches step 1(c) with $x=z_{k+1}$, so it returns either $(\textsc{UnFilled},z_{k+1})$ or $(\textsc{Filled},z_{k+1})$.

It can be split into two cases.

- $1 \le k < \min(\beta', d-1)$. The first k f_λ-queries are $\zeta_1, \ldots, \zeta_k$, and there is no CF-queries until the $(k+1)$-th f_λ-query $z_{k+1} = \zeta_{k+1}$ is issued. IH1 and IH2 imply that $Z(z_t) = f_\lambda(z_t)$ for all $t \in [k]$; hence, $\mathsf{GetQuery}(Z, k+1)$ will not fail at step 1(b) and will successfully reach step 1(c) with $x = z_{k+1}$.
- $\beta' \le k < d-1$. IH1 implies that $j > \phi^{-1}(z_t) = \phi^{-1}(\zeta_t)$ for all $t \in [\beta']$. Since ϕ is good, this implies that $j > \phi^{-1}(u)$ for all $u \in U$. Then by IH2, we have $Z(u) = f_\lambda(u)$ for all $u \in U$, which, according to lemma 18, implies $\widetilde{\mathsf{CF}}^{f',\mathrm{r}}(C) = \mathsf{CF}^{f,\mathrm{r}}(C)$ for all CF-query C issued by S. Consequently, $\mathsf{GetQuery}(Z, k+1)$ will not fail at step 1(b) or 1(d), and will successfully reach step 1(c) with $x = z_{k+1}$

Next, given $(\diamond)$, we show the invariants continue to hold for $i = k+1$.

- Case 1: $\mathsf{GetQuery}(Z, k+1) = (\textsc{Filled}, z_{k+1})$. The while loop in step 2(d) will not be executed, and thus $j_{k+1} = j_k$. Meanwhile, IH2 implies that $\phi^{-1}(z_{k+1}) < j_k$ since $Z(z_{k+1}) \neq \perp$.
- Case 2: $\mathsf{GetQuery}(Z, k+1) = (\textsc{Unfilled}, z_{k+1})$. The while loop in step 2(d) will be executed and thus $j_{k+1} = \phi^{-1}(z_{k+1}) + 1$. IH2 implies that $j_k \le \phi^{-1}(z_{k+1})$ since $Z(z_{k+1}) = \perp$, meaning that $\max_{v \in [k+1]} \phi^{-1}(z_v) = \phi^{-1}(z_{k+1})$. Moreover, for the values filled in step 2(d), since $\phi^{-1}(z_{k+1}) < \phi^{-1}(z_d)$ (because ϕ is good), we have $L(\phi(j')) = f_\lambda(\phi(j'))$ for $j_k \le j' < j_{k+1}$. After step 2(d), we have $j_{k+1} = \phi^{-1}(z_{k+1}) + 1$.

After the FOR loop, we have L1 and L2 hold for $i = d-1$. Since z_d ranks last among $z_1, \ldots, z_d$, L1 implies that (i) $j \le z_d$. Moreover, $(\diamond)$ implies $x' = z_{i^*}$, and thus (ii) $f_\lambda(z_d) = f_\lambda(z_{i^*}) = Z(x')$. And by the same argument above, we have (iii) $\mathsf{GetQuery}(Z, d) = (\textsc{Unfilled}, z_d)$ at step 3. With (i)(ii)(iii), comparing steps 5-7 with the encoding algorithm, we conclude that $Z = f_\lambda$ when returned. □

4 No $(K+1)$-Collision in Confirmation Queries

In this section, we prove lemma 24, restated below:

Lemma (Lemma 24 restated with context)**.** *Let $q = O(2^{0.01\lambda}), \beta = \lambda \cdot 2^{0.1\lambda}, \beta' = 2^{0.2\lambda}$. Let $\mathcal{A}$ be a (q,q,q)-bound adversary and let $\mathcal{S}$ be the (β,β')-normalized version of $\mathcal{A}$, which is $(\widetilde{q},q,q)$-bounded for $\widetilde{q} = O(2^{0.2\lambda})$. Then, for all λ and fixed $f_{-\lambda}$, we have*

$$\Pr_{\mathsf{r}\leftarrow\mathsf{Coin}^\star,\mathsf{coin}\leftarrow\mathsf{Coin}_S,f_\lambda\leftarrow\mathcal{F}_\lambda}[\mathsf{CFHit}] = O(2^{-0.01\lambda}),$$

where CFHit *denotes the event that a $(K+1)$-collision is found when $\mathcal{S}^{f,\mathsf{CF}^{f,\mathrm{r}}}(1^\lambda;\mathsf{coin})$ is evaluating the confirmation queries.*

Fix $i \in [q]$ and consider $\mathcal{S}$'s i-th CF-query C. Let CFHit_i denote the event that a $(K+1)$-collision is found when $\mathcal{S}^{f,\mathsf{CF}^{f,\mathrm{r}}}(1^\lambda)$ is evaluating the confirmation queries right after the i-th CF-query C. Clearly,

$$\Pr[\mathsf{CFHit}] = \sum_{i\in[q]} \Pr[\mathsf{CFHit}_i].$$

For $j \in [K]$, define the following two classes of events, where $(w_C^1, \ldots w_C^K) := \mathsf{CF}^{f,\mathrm{r}}(C)$ and Q_i is the set of f_λ-queries issued by $\mathcal{S}$ before the i-th CF-query.

- $\mathsf{FindSib}_{i,j}$: $\mathtt{Query}_\lambda^f(C, w_C^j) \cap \mathtt{Sib}(f_\lambda, Q_i) \neq \emptyset$;
- $\mathsf{FreshColl}_{i,j}$: $\exists x, x' \in \mathtt{Query}_\lambda^f(C, w_C^j) \setminus Q_i$ s.t.$(x, x') \in \mathtt{Coll}_2(f_\lambda)$. That is, there is a 2-collision for f_λ in the new queries induced by evaluating $C^f(w_C^j)$.

We claim that

$$\mathsf{CFHit}_i \subseteq \bigcup_{j\in[K]} \mathsf{FindSib}_{i,j} \cup \bigcup_{j\in[K]} \mathsf{FreshColl}_{i,j}. \tag{6}$$

This is because, if CFHit_i happens but $\bigcup_{j\in[K]} \mathsf{FindSib}_{i,j}$ does not happen, then the $(K+1)$-collision found by $\mathcal{S}$ must come from

$$\bigcup_{j\in[K]} \mathtt{Query}_\lambda^f(C, w_C^j) \setminus Q_i.$$

By the pigeonhole principle, there exists some $j \in [K]$ such that $\mathtt{Query}_\lambda(C, w_C^j) \setminus Q_i$ contains a 2-collision for f_λ, meaning that $\bigcup_{j\in[K]} \mathsf{FreshColl}_{i,j}$ happens.

Therefore, it suffices to derive an upper bound for the probability of the events $\mathsf{FindSib}_{i,j}$, $\mathsf{FreshColl}_{i,j}$ for all i, j. Our proof proceeds in three steps:

1. Section 4.1 introduces two *random oracle games with adaptive leakage* such that the probability of these events is bounded from above by the advantage of an adversary in the two games. By adaptive leakage, we mean the adversary can get global information about the random oracle besides query access.
2. Next, section 4.2 shows that random oracle games with adaptive leakage can be simulated in an alternative way that reveals more structural information.
3. Finally, section 4.3 proves that in both games, adversary's advantage can be bounded in terms of the number of queries and the amount of leakage.

4.1 Random Oracle Games with Adaptive Leakage

When λ and $f_{-\lambda}$ are fixed, in the $(K+1)$-collision-finding experiment for f, the adversary learns information about f_λ either by direct queries to f_λ or by calling CF^f. The answers returned by CF can be viewed as a global leakage on the function f_λ. Therefore, access to the random function in the experiment can be simulated using the following two interfaces.

- $\mathsf{Eval}^{f_\lambda}(x)$: On input $x \in \{0,1\}^{\ell(\lambda)}$, return the value $f(x)$.
- $\mathsf{Leak}^{f_\lambda}(L)$: On input a leakage function $L : \mathcal{F}_\lambda \to \{0,1\}^a$, output $L(f_\lambda)$.

We also define an additional interface $\mathsf{Finalize}^{f_\lambda}$, which simply outputs the entire function f_λ. We assume that $\mathsf{Finalize}^{f_\lambda}$ is invoked at the end of the interaction, that is, after all queries to $(\mathsf{Eval}^{f_\lambda}, \mathsf{Leak}^{f_\lambda})$. Let Q_{final} be the set of all queries made to Eval^f at the moment when $\mathsf{Finalize}^{f_\lambda}$ is invoked.

Using these interfaces, we next define two random oracle games; in lemma 32 and lemma 33 we shall show that the two games capture the events $\mathsf{FindSib}_{i,j}$ and $\mathsf{FreshColl}_{i,j}$.

Definition 29 (Sibling-finding game)**.** For an adversary $\mathcal{B}$, we define an experiment, denoted $\mathsf{Expt}_{\mathcal{B}}^{\mathsf{find\text{-}sib}}(\lambda)$, as follows:

1. $f_\lambda \leftarrow \mathcal{F}_\lambda$ is chosen uniformly at random and $\mathcal{B}$ is given access to $\mathsf{Eval}^{f_\lambda}$ and $\mathsf{Leak}^{f_\lambda}$.
2. At some point, $\mathcal{B}$ submits a challenge circuit $C^{(\cdot)} : \{0,1\}^m \to \{0,1\}^n$, which is an oracle-aided circuit.
3. Sample $w_1, \ldots, w_\beta \leftarrow \{0,1\}^m$ and evaluate $C^{(\cdot)}$ on inputs $w_1, \ldots, w_\beta$ using $\mathsf{Eval}^{f_\lambda}$ to answer the oracle gates.
4. **Decide winning condition:** $f_\lambda \leftarrow \mathsf{Finalize}^{f_\lambda}$ and sample $w^* \leftarrow \{0,1\}^m$. The experiment outputs 1 if and only if
$$\mathtt{Query}^f_\lambda(C, w^*) \cap \mathtt{Sib}(f_\lambda, Q_{\mathsf{final}}) \neq \emptyset. \tag{7}$$

Definition 30 (Fresh-collision-finding game)**.** For an adversary $\mathcal{B}$, we define an experiment $\mathsf{Expt}^{\mathsf{fresh\text{-}coll}}_{\mathcal{B}}(\lambda)$ to be identical to $\mathsf{Expt}^{\mathsf{find\text{-}sib}}_{\mathcal{B}}(\lambda)$ except that the winning condition in eq. (7) is replaced with
$$\exists x, x' \in \mathtt{Query}^f_\lambda(C, w^*) \setminus Q_{\mathsf{final}} \text{ s.t. } (x, x') \in \mathtt{Coll}_2(f_\lambda). \tag{8}$$

Definition 31. We say an adversary $\mathcal{B}$ is $(B, \mathfrak{q}, \mathfrak{q}')$-***admissible*** if (1) it makes at most $\mathfrak{q}$ queries to $\mathsf{Eval}^{f_\lambda}$; (2) the sum of output length of its queries to $\mathsf{Leak}^{f_\lambda}$ never exceeds B; (3) its challenge circuit has at most $\mathfrak{q}'$ oracle gates.

Lemma 32 (Reduction to oracle game I)**.** *For all $i \in [q]$, $j \in [K]$, there exists a $(q^2(K-1), \widetilde{q}, q)$-admissible adversary $\mathcal{B}$ such that* $\mathbf{Pr}\left[\mathsf{FindSib}_{i,j}\right] \leq \mathbf{Pr}\left[\mathsf{Expt}^{\mathit{find\text{-}sib}}_{\mathcal{B}}(\lambda) \Rightarrow 1\right]$.

Proof. Fix $i \in [q], j \in [K]$. Consider the adversary $\mathcal{B}^{\mathsf{Eval}^{f_\lambda}, \mathsf{Leak}^{f_\lambda}}$ that uses randomness $(\mathrm{r}, \mathsf{coin}) \in \mathtt{Coin}^\star \times \mathtt{Coin}_{\mathcal{S}}$ and operates by simulating $\mathcal{A}^{f, \mathsf{CF}^{f,\mathrm{r}}}(1^\lambda)$ the same way as $\mathcal{S}^{f, \mathsf{CF}^{f,\mathrm{r}}}(1^\lambda; \mathsf{coin})$ does, until $\mathcal{A}$ issues the i-th new CF-query (excluding repeating queries); $\mathcal{B}(\mathrm{r}, \mathsf{coin})$ simulates the oracle $(f, \mathsf{CF}^{f,\mathrm{r}})$ as follows.

- Directly answer $f_{-\lambda}$-queries with $f_{-\lambda}$.
- For an f_λ-query x, set the oracle answer to be $\mathsf{Eval}^{f_\lambda}(x)$.
- For the v-th CF-query $C : \{0,1\}^m \to \{0,1\}^n$ ($m \leq q$) where $v < i$:
  1. Read $w_C \in \{0,1\}^m, \pi^2_C, \ldots, \pi^K_C \in \mathtt{Perm}(\{0,1\}^m)$ from r.
  2. Let the leakage function $L_{w^1_C, \pi^2_C, \ldots, \pi^K_C} : \mathcal{F}_\lambda \to (\{0,1\}^m)^{K-1}$ be defined as
  $$L_{w^1_C, \pi^2_C, \ldots, \pi^K_C}(f_\lambda) \stackrel{\mathsf{def}}{=} (\pi^j_C(i_2), \ldots, \pi^j_C(i_K)),$$
  where i_j is the smallest index in $[2^m]$ such that $C^f(\pi^j_C(i_j)) = C^f(w^1_C)$. Invoke $\mathsf{Leak}^{f_\lambda}(L_{w^1_C, \pi^2_C, \ldots, \pi^K_C})$ and receive $(w^2_C, \ldots, w^K_C) \in (\{0,1\}^m)^{K-1}$ from the $\mathsf{Leak}^{f_\lambda}$.
  3. Set the oracle answer to be $(w^1_C, \ldots, w^K_C)$.
- When $\mathcal{A}$ issues the i-th (new) CF-query C, $\mathcal{B}$ submits C as the challenge.
- If $\mathcal{A}$ halts before submitting the i-th CF-query, $\mathcal{B}$ aborts.

Two observations regarding $\mathcal{B}$ are immediate:

- $\mathcal{B}$ is $(q^2(K-1), \widetilde{q}, q)$-admissible. This is becauase there are at most q CF-queries, and each CF-query induces a leakage of length $m(K-1) \le q(K-1)$.
- When $f_\lambda \leftarrow \mathcal{F}_\lambda$, $\mathcal{B}^{\mathsf{Eval}^{f_\lambda}, \mathsf{Leak}^{f_\lambda}}(\mathrm{r}, \mathsf{coin})$ perfectly simulates $(f, \mathsf{CF}^{f,\mathrm{r}})$ in the view of $\mathcal{A}$ and runs exactly the same as $\mathcal{S}^{f,\mathsf{CF}^{f,\mathrm{r}}}(1^\lambda; \mathsf{coin})$ until $\mathcal{A}$ issues the i-th CF-query $C : \{0,1\}^{m(n)} \to \{0,1\}^n$.

Let Q' be the set of Eval-queries made by $\mathcal{B}$, which is also the f_λ-queries made by $\mathcal{S}$ until $\mathcal{A}$ issues the i-th CF-query. After $\mathcal{B}$ submits the challenge, the experiment $\mathsf{Expt}^{\mathsf{find\text{-}sib}}_{\mathcal{B}}$ continues as follows:

1. Sample $w_1, \ldots, w_\beta, w^* \leftarrow \{0,1\}^m$.
2. The experiment output 1 if and only if

$$\mathtt{Query}^f_\lambda(C, w^*) \cap \mathtt{Sib}(f_\lambda, Q_{\mathsf{final}}) \neq \emptyset,$$

where

$$Q_{\mathsf{final}} = Q' \cup \bigcup_{v \in [\beta]} \mathtt{Query}^f_\lambda(C, w_v).$$

Consider the experiment $\mathsf{Expt}'_{\mathcal{B}}$ which is identical to $\mathsf{Expt}^{\mathsf{find\text{-}sib}}_{\mathcal{B}}$ except that $(w_1, \ldots, w_\beta, w^*)$ is sampled as follows:

- Sample $w_1, \ldots, w_\beta \leftarrow \{0,1\}^m$ and $(w^1_C, \ldots, w^K_C) := \mathsf{CF}^{f,\mathrm{r}}(C)$; set $w^* := w^j_C$.

Since the marginal distribution of w^j_C is uniform as $\mathrm{r} \leftarrow \mathtt{Coin}^\star$, the distribution of $\mathsf{Expt}'_{\mathcal{B}}$ is the same as $\mathsf{Expt}^{\mathsf{find\text{-}sib}}_{\mathcal{B}}$. We claim that

$$\mathbf{Pr}\left[\mathsf{FindSib}_{i,j}\right] \le \mathbf{Pr}\left[\mathsf{Expt}'_{\mathcal{B}} \Rightarrow 1\right].$$

To see this, we consider a natural coupling:

- Run $\mathcal{S}^{f,\mathsf{CF}^{f,\mathrm{r}}}(1^\lambda; \mathsf{coin})$ and $\mathsf{Expt}'_{\mathcal{B}}$ with the same randomness $(\mathrm{r}, \mathsf{coin}, f_\lambda)$. In particular, $\mathsf{Expt}'_{\mathcal{B}}$ reads $(w_1, \ldots, w_\beta)$ from coin, where $(w_1, \ldots, w_\beta)$ is used as the smoothening points for the i-th CF-query in the execution of $\mathcal{S}^{f,\mathsf{CF}^{f,\mathrm{r}}}(1^\lambda; \mathsf{coin})$.

In the above coupling process, we have

$$Q_i = Q' \cup \bigcup_{v \in [\beta]} \mathtt{Query}^f_\lambda(C, w_v) = Q_{\mathsf{final}},$$

where we recall that Q_i is the set of f_λ-queries issued by $\mathcal{S}$ before the i-th CF-query; hence, if $\mathsf{FindSib}_{i,j}$ happens, then the experiment $\mathsf{Expt}'_{\mathcal{B}}$ outputs 1. Therefore,

$$\mathbf{Pr}\left[\mathsf{FindSib}_{i,j}\right] \le \mathbf{Pr}\left[\mathsf{Expt}'_{\mathcal{B}} \Rightarrow 1\right] = \mathbf{Pr}\left[\mathsf{Expt}^{\mathsf{find\text{-}sib}}_{\mathcal{B}} \Rightarrow 1\right],$$

concluding the proof. □

Lemma 33 (Reduction to oracle game II)**.** *For all $i \in [q]$, $j \in [K]$, there exists a $(q^2(K-1), \widetilde{q}, q)$-admissible adversary $\mathcal{B}$ such that* $\mathbf{Pr}\left[\mathsf{FreshColl}_{i,j}\right] \le \mathbf{Pr}\left[\mathsf{Expt}^{\mathit{fresh\text{-}coll}}_{\mathcal{B}}(\lambda) \Rightarrow 1\right]$.

Proof. It follows from the same argument as lemma 32, since the winning condition and the event $\mathsf{FreshColl}_{i,j}$ have an analogous correspondence. □

The advantage of any $(B, \mathfrak{q}, \mathfrak{q}')$-admissible adversary in the random oracle experiments is captured by the next two lemmas, proven in the rest of this section.

Lemma 34. *For all $(B, \mathfrak{q}, \mathfrak{q}')$-admissible adversary $\mathcal{B}$ and $\gamma \in (0,1)$, it holds that*

$$\mathbf{Pr}\left[\mathsf{Expt}_{\mathcal{B}}^{\textit{find-sib}}(\lambda) \Rightarrow 1\right] \leq 2^{-0.05\lambda} \cdot \mathfrak{q}' \cdot (B + \log(\mathfrak{q} + \beta\mathfrak{q}' + 1) + 1) + \gamma \cdot 2^{0.05\lambda} + \frac{\mathfrak{q}'^2}{\gamma} \cdot 2^{-\gamma\beta} + 2\mathfrak{q}'^2 \cdot (\mathfrak{q} + \beta\mathfrak{q}') \cdot 2^{-\lambda}.$$

Lemma 35. *For all $(B, \mathfrak{q}, \mathfrak{q}')$-admissible adversary $\mathcal{B}$ and $\gamma \in (0,1)$, it holds that*

$$\mathbf{Pr}\left[\mathsf{Expt}_{\mathcal{B}}^{\textit{fresh-coll}}(\lambda) \Rightarrow 1\right] \leq 2^{-0.05\lambda} \cdot \mathfrak{q}' \cdot (B + \log(\mathfrak{q} + \beta\mathfrak{q}' + 1) + 1) + \gamma \cdot 2^{0.05\lambda} + \frac{\mathfrak{q}'^2}{\gamma} \cdot 2^{-\gamma\beta} + 2\mathfrak{q}'^3 \cdot 2^{-\lambda}.$$

Proving lemma 24. We first complete the proof of lemma 24 using the above lemmas and then proceed to prove the lemmas. Recall that $\beta = \lambda \cdot 2^{0.1\lambda}$. Let

$$B := q^2(K-1) = O(2^{0.02\lambda}), \quad \gamma := 2^{-0.1\lambda}.$$

By lemmas 34 and 35, for all $(B, \widetilde{q}, q)$-admissible adversaries $\mathcal{B}, \mathcal{B}'$, it holds that

$$\mathbf{Pr}\left[\mathsf{Expt}_{\mathcal{B}}^{\mathsf{find\text{-}sib}}(\lambda) \Rightarrow 1\right] \leq O(2^{-0.02\lambda}),$$
$$\text{and } \mathbf{Pr}\left[\mathsf{Expt}_{\mathcal{B}'}^{\mathsf{fresh\text{-}coll}}(\lambda) \Rightarrow 1\right] \leq O(2^{-0.02\lambda}).$$

Combining with lemmas 32 and 33, we have

$$\mathbf{Pr}\left[\mathsf{FindSib}_{i,j}\right] \leq O(2^{-0.02\lambda}) \text{ and } \mathbf{Pr}\left[\mathsf{FreshColl}_{i,j}\right] \leq O(2^{-0.02\lambda})$$

for all $i \in [q]$ and $j \in [K]$. It follows from eq. (6) that for all $i \in [q]$,

$$\mathbf{Pr}\left[\mathsf{CFHit}_i\right] \leq 2K \cdot O(2^{-0.02\lambda}) = O(2^{-0.02\lambda}).$$

Therefore,

$$\mathbf{Pr}\left[\mathsf{CFHit}\right] = \sum_{i \in [q]} \mathbf{Pr}\left[\mathsf{CFHit}_i\right] \leq q \cdot O(2^{-0.02\lambda}) = O(2^{-0.01\lambda}),$$

completing the proof.

4.2 Simulating a Random Oracle with Adaptive Leakage

From now on, we omit the subscript λ and write $f \in \mathcal{F}_\lambda$. For $f \leftarrow \mathcal{F}_\lambda$ chosen uniformly at random, the three interfaces $(\mathsf{Eval}^f, \mathsf{Leak}^f, \mathsf{Finalize}^f)$ can be perfectly simulated as follows: The simulator is parameterized by a real number $\eta \in (0,1)$. It samples the information required on the fly and maintains a set $G \subseteq \mathcal{F}_\lambda$, which keeps track of the set of all possible functions conditioned on the revealed partial information. Specifically, it sets $G := \mathcal{F}_\lambda$ in the beginning, and then provides three interfaces $(\mathsf{SEval}, \mathsf{SLeak}, \mathsf{SFinalize})$ and maintains G by operating as follows.

- $\mathsf{SEval}(x)$: sample $y \leftarrow G(x)$ and update $G := \{f \in G : f(x) = y\}$; return y.

- SLeak(L): sample $\zeta \leftarrow L(\boldsymbol{G})$ and update $G := \{f \in G : L(f) = \zeta\}$; return ζ.
- SFinalize :
  1. Let Q_{final} be the set of all queries made to SEval, then G is fixed on Q_{final}.
  2. Apply density restoring partition with density parameter η to obtain a partition $G = G^1 \cup \cdots \cup G^r$ and $I_1, \ldots, I_r \subseteq \{0,1\}^\ell$ such that each G^i is fixed on $Q_{\mathsf{final}} \cup I_i$ and η-dense on $\overline{Q_{\mathsf{final}} \cup I_i}$.
  3. Sample $\boldsymbol{\sigma}$ with probabilities $\mathbf{Pr}[\boldsymbol{\sigma} = \sigma] = |G^\sigma|/|G|$ for all $\sigma \in [r]$.
  4. Set $G_{\mathsf{final}} := G^{\boldsymbol{\sigma}}$ and $I_{\mathsf{final}} := I_{\boldsymbol{\sigma}}$. `//` G_{final} `is` η`-dense on` $\overline{Q_{\mathsf{final}} \cup I_{\mathsf{final}}}$.
  5. Sample $f \leftarrow G_{\mathsf{final}}$ and output f.

Lemma 36. (SEval, SLeak, SFinalize) *perfectly simulates* ($\mathsf{Eval}^f, \mathsf{Leak}^f, \mathsf{Finalize}^f$) *where f is uniformly chosen from $\mathcal{F}_\lambda$.*

Proof. It is clear that (SEval, SLeak) perfectly simulates ($\mathsf{Eval}^f, \mathsf{Leak}^f$). Observe that SFinalize essentially chooses $f \in G$ uniformly at random and outputs f. Indeed, it first partitions G into subsets, then chooses a subset with probabilities proportional to the size of each subset, and finally chooses f uniformly at random from that subset. □

The next lemma shows that the expected size of I_{final} is bounded by $O(\frac{B}{(1-\eta)\lambda})$ (modulo less significant terms), where B is the amount of leakage.

Lemma 37 (Average fixed size)**.** *Let T be an algorithm that has access to* (SEval, SLeak, SFinalize)*. Suppose that given any random coins and oracle answers, T makes at most $\mathfrak{q}$ queries to* SEval *and the total output length of its queries to* SLeak *is at most B. Then we have*

$$\mathbf{E}[|I_{\mathsf{final}}|] \le \frac{B + \log(\mathfrak{q}+1) + 1}{(1-\eta)\lambda}.$$

Proof. It suffices to prove the lemma for all deterministic algorithms, so we assume T is deterministic in what follows. WLOG, assume under any circumstance, the total output length of T's queries to SLeak is exactly B; this can be ensured by adding a dummy query (i.e., a constant function L) to SLeak in the end, which has no effect on G. Then the transcript between T and (SEval, SLeak) has the form

$$(\vec{y} = (y_1, \ldots, y_j), a) \in \bigcup_{j=0}^{q} (\{0,1\}^\lambda)^j \times \{0,1\}^B,$$

where $0 \le j \le \mathfrak{q}$, y_i is the answer to the i-th SEval query, and a is the (concatenation of) answers to the SLeak-queries.

Let $\widehat{G}$ denote the value of G at the time when SFinalize is invoked (at the end of the interaction). Note that Q_{final} and $\widehat{G}$ are both determined by the transcript. Let $G_{\vec{y},a}$ be the value of $\widehat{G}$ when the transcript is $(\vec{y}, a)$. Then $\{G_{\vec{y},a}\}_{\vec{y},a}$ are disjoint sets and form a partition of $\mathcal{F}_\lambda$. Let $\mathcal{T}$ be the distribution of the transcript; clearly, the probability that the transcript is $(\vec{y}, a)$ equals to $|G_{\vec{y},a}|/|\mathcal{F}_\lambda|$.

By the property of density-restoring partition, we have the following upper bound of $\mathbf{E}[|I_{\mathsf{final}}|]$ for every fixed transcript.

Claim 38. *For every fixed transcript $(\vec{y}, a) \in \mathrm{supp}(\mathcal{T})$, it holds that*

$$\mathop{\mathbf{E}}_{\boldsymbol{\sigma}}\left[|I_{\mathsf{final}}| \mid \textit{ the transcript is } (\vec{y}, a)\right] \leq \frac{(2^\ell - |\vec{y}|)\lambda - \log|G_{\vec{y},a}| + 1}{(1-\eta)\lambda}.$$

Proof. Write $N \stackrel{\mathsf{def}}{=} 2^\ell$ and equate $\{0,1\}^\ell$ with $[N]$. Let $G^1, \ldots, G^r$ denote the density restoring partition of $G_{\vec{y},a}$ with associated fixed index $I_1, \ldots, I_r$. For $\sigma \in [r]$, With probability $p_\sigma \stackrel{\mathsf{def}}{=} |G^\sigma|/|G_{\vec{y},a}|$, we set $G_{\mathsf{final}} := G^\sigma$ and $I_{\mathsf{final}} := I_\sigma$. Therefore,

$$\mathop{\mathbf{E}}_{\boldsymbol{\sigma}}\left[|I_{\mathsf{final}}| \mid \text{ the transcript is } (\vec{y}, a)\right] = \mathop{\mathbf{E}}_{\sigma \leftarrow \boldsymbol{\sigma}}\left[|I_\sigma|\right].$$

Let Q be value of Q_{final} determined by $(\vec{y}, a)$. By lemma 11, for all $\sigma \in [r]$ we have

$$0 \leq \mathbf{D}(G^\sigma(\overline{Q \cup I^\sigma})) \leq \mathbf{D}(G_{\vec{y},a}(\overline{Q})) - (1-\eta)\lambda|I_\sigma| + \delta_\sigma,$$

where $\delta_\sigma = \log(|X|/|\cup_{v \geq \sigma} X^v|)$. Therefore, taking expectation over the random choice of $\boldsymbol{\sigma}$, it holds that

$$\begin{aligned}(1-\eta)\lambda \mathop{\mathbf{E}}_{\sigma \leftarrow \boldsymbol{\sigma}}\left[|I_\sigma|\right] &\leq \mathbf{D}(G_{\vec{y},a}(\overline{Q})) + \mathop{\mathbf{E}}_{\sigma \leftarrow \boldsymbol{\sigma}}\left[\delta_\sigma\right] \\ &= (N - |\vec{y}|)\lambda - \log|G_{\vec{y},a}| + \mathop{\mathbf{E}}_{\sigma \leftarrow \boldsymbol{\sigma}}\left[\delta_\sigma\right],\end{aligned} \tag{9}$$

where the second inequality uses the definition of deficiency and $|Q| = |\vec{y}|$. Note that $\delta_\sigma \stackrel{\mathsf{def}}{=} \log \frac{1}{\sum_{v \geq \sigma} p_v}$ and thus

$$\mathop{\mathbf{E}}_{\sigma \leftarrow \boldsymbol{\sigma}}\left[\delta_\sigma\right] = \sum_{\sigma=1}^{m} p_\sigma \log \frac{1}{\sum_{v \geq \sigma} p_\sigma} \leq \int_0^1 \log \frac{1}{1-x} \mathrm{d}x \leq 1.$$

Plugging into eq. (9) we get

$$\mathop{\mathbf{E}}_{\sigma \leftarrow \boldsymbol{\sigma}}\left[|I_\sigma|\right] \leq \frac{(N - |\vec{y}|)\lambda - \log|G_{\vec{y},a}| + 1}{(1-\eta)\lambda}.$$

□

By the above claim, we have

$$\mathbf{E}\left[|I_{\mathsf{final}}|\right] = \mathop{\mathbf{E}}_{(\vec{y},a) \leftarrow \mathcal{T}}\left[\mathop{\mathbf{E}}_{\boldsymbol{\sigma}}\left[|I_{\mathsf{final}}| \mid \text{ the transcript is } (\vec{y}, a)\right]\right] \leq \frac{2^\ell \lambda + 1 - \mathbf{E}_{(\vec{y},a) \leftarrow \mathcal{T}}\left[|\vec{y}|\lambda + \log|G_{\vec{y},a}|\right]}{(1-\eta)\lambda}. \tag{10}$$

Let $\mathcal{T}_j$ be the distribution of the transcript conditioned on the event that the number of SEval-queries is j (i.e., $|\vec{y}| = j$). Then we have

$$\mathop{\mathbf{E}}_{(\vec{y},a) \leftarrow \mathcal{T}}\left[|\vec{y}|\lambda + \log|G_{\vec{y},a}|\right] = \sum_{j=0}^{\mathfrak{q}} \theta_j \underbrace{\mathop{\mathbf{E}}_{(\vec{y},a) \leftarrow \mathcal{T}_j}\left[j\lambda + \log|G_{\vec{y},a}|\right]}_{\stackrel{\mathsf{def}}{=} W_j}, \tag{11}$$

where θ_j is the probability that the number of SEval-queries is j. For each $j \in [\mathfrak{q}]$, letting $G_j := \bigcup_{(\vec{y},a)\in\mathrm{supp}(\mathcal{T}_j)} G_{\vec{y},a}$, it holds that

$$\begin{aligned} W_j &= j\lambda + \sum_{(\vec{y},a)\in\mathrm{supp}(\mathcal{T}_j)} \frac{|G_{\vec{y},a}|}{|G_j|} \log |G_{\vec{y},a}| \\ &= j\lambda + \sum_{(\vec{y},a)\in\mathrm{supp}(\mathcal{T}_j)} \frac{|G_{\vec{y},a}|}{|G_j|} \left(\log \frac{|G_{\vec{y},a}|}{|G_j|} + \log |G_j| \right) \\ &= j\lambda - \mathbf{H}(\mathcal{T}_j) + \log |G_j| \\ &\geq -B + \log |G_j|, \end{aligned}$$

where the last inequality follows from $\mathbf{H}(\mathcal{T}_j) \leq \log \mathrm{supp}(\mathcal{T}_j) \leq j\lambda + B$. Continuing eq. (11), we get

$$\begin{aligned} \mathop{\mathbf{E}}_{(\vec{y},a)\leftarrow\mathcal{T}} \left[|\vec{y}|\lambda + \log |G_{\vec{y},a}| \right] &\geq -B + \sum_{j=0}^{\mathfrak{q}} \theta_j \log |G_j| = -B + \sum_{j=0}^{\mathfrak{q}} \theta_j (\log \theta_j + 2^\ell \lambda) \\ &\geq -B - \log(\mathfrak{q}+1) + 2^\ell \lambda, \end{aligned}$$

where the last inequality uses the fact that $\sum_{j=0}^{\mathfrak{q}} -\theta_j \log \theta_j \leq \log(\mathfrak{q}+1)$. Plugging this into eq. (10), we conclude that $\mathbf{E}\left[|I_{\mathsf{final}}|\right] \leq \frac{B+\log(\mathfrak{q}+1)+1}{(1-\eta)\lambda}$, as desired. □

4.3 Advantage in Random Oracle Games with Adaptive Leakage

This subsection is dedicated for proving lemma 34 and lemma 35.

Lemma (Lemma 34 restated)**.** *For all $(B,\mathfrak{q},\mathfrak{q}')$-admissible adversary $\mathcal{B}$ and $\gamma \in (0,1)$, it holds that*

$$\mathbf{Pr}\left[\mathsf{Expt}_{\mathcal{B}}^{\textit{find-sib}}(\lambda) \Rightarrow 1\right] \leq 2^{-0.05\lambda} \cdot \mathfrak{q}' \cdot (B + \log(\mathfrak{q} + \beta\mathfrak{q}' + 1) + 1) + \gamma \cdot 2^{0.05\lambda} + \frac{\mathfrak{q}'^2}{\gamma} \cdot 2^{-\gamma\beta} + 2\mathfrak{q}'^2 \cdot (\mathfrak{q} + \beta\mathfrak{q}') \cdot 2^{-\lambda}.$$

Proof of lemma 34. Let $\mathcal{B}$ be a $(B,\mathfrak{q},\mathfrak{q}')$-admissible adversary. By lemma 36, $\mathsf{Expt}_{\mathcal{B}}^{\mathsf{find\text{-}sib}}(\lambda)$ can be perfectly simulated with parameter $\eta = 1 - \frac{1}{\mathfrak{q}'\lambda}$ as follows:

1. $\mathcal{B}$ is given access to SEval and SLeak.
2. At some point, $\mathcal{B}$ submits a challenge circuit $C^{(\cdot)} : \{0,1\}^m \to \{0,1\}^n$, which is an oracle-aided circuit with at most $\mathfrak{q}'$ oracle gates.
3. Sample $w_1, \ldots, w_\beta \leftarrow \{0,1\}^m$. Evaluate $C^{(\cdot)}$ on inputs $w_1, \ldots, w_\beta$ using SEval to answer the oracle gates.
4. **Decide winning condition:** $f \leftarrow \mathsf{SFinalize}$ and sample $w^* \leftarrow \{0,1\}^m$. The experiment outputs 1 if and only if eq. (7) holds.

Claim 39. $\mathbf{Pr}\left[|I_{\mathsf{final}}| > 2^{0.05\lambda}\right] < 2^{-0.05\lambda} \cdot \mathfrak{q}' \cdot (B + \log(\mathfrak{q} + \beta\mathfrak{q}' + 1) + 1)$.

Proof. The number of SEval queries is bound by $\mathfrak{q} + \beta\mathfrak{q}'$. By lemma 37, it holds that $\mathbf{E}\left[|I_{\mathsf{final}}|\right] \leq \frac{B+\log(\mathfrak{q}+\beta\mathfrak{q}'+1)+1}{(1-\eta)\lambda}$ where $\eta = 1 - \frac{1}{\mathfrak{q}'\lambda}$; and the claim follows from Markov inequality. □

Define

$$\texttt{Heavy} \stackrel{\text{def}}{=} \left\{x \in \{0,1\}^m : \mathop{\mathbf{Pr}}_{w}\left[x \in \texttt{Query}^f(C,w)\right] > \gamma\right\}.$$

Note that

$$\texttt{Heavy} \subseteq \bigcup_{i\in[\mathfrak{q}']} \texttt{Heavy}_i$$

where $\texttt{Heavy}_i \stackrel{\text{def}}{=} \left\{x \in \{0,1\}^m : \mathbf{Pr}_w\left[\text{ the } i\text{-th query of } C^f(w) \text{ is } x\right] > \gamma/\mathfrak{q}'\right\}$. For each $i \in [\mathfrak{q}']$, we have $|\texttt{Heavy}_i| \le \mathfrak{q}'/\gamma$ since the events 'the i-th query of $C^f(w)$ is x' are disjoint for all x's. Consequently, $|\texttt{Heavy}| \le \mathfrak{q}'^2/\gamma$. For all $x \in \texttt{Heavy}$,

$$\mathop{\mathbf{Pr}}_{w_1,\ldots,w_\beta}\left[x \notin Q_{\mathsf{final}}\right] \le (1-\gamma)^\beta \le 2^{-\gamma\beta}.$$

Let Smooth denote the event that $\texttt{Heavy} \subseteq Q_{\mathsf{final}}$. By union bound, we have

$$\mathbf{Pr}\left[\neg\mathsf{Smooth}\right] \le |\texttt{Heavy}| \cdot 2^{-\gamma\beta} \le \frac{\mathfrak{q}'^2}{\gamma} \cdot 2^{-\gamma\beta}. \tag{12}$$

Let Hit denote the event that $\texttt{Query}^f(w^*) \cap I_{\mathsf{final}} \neq \emptyset$, and $\mathsf{E} \stackrel{\text{def}}{=} \mathsf{Smooth} \wedge (|I_{\mathsf{final}}| \le 2^{0.05\lambda})$. Note that $I_{\mathsf{final}} \cap Q_{\mathsf{final}} = \emptyset$ by construction and the event E does not depend on w^*. Hence, when fixing all randomness except w^* such that E happens and considering the random choice of w^*, it holds that

$$\mathop{\mathbf{Pr}}_{w^*}\left[\mathsf{Hit} \mid \mathsf{E}\right] \le |I_{\mathsf{final}}| \cdot \gamma \le 2^{0.05\lambda} \cdot \gamma. \tag{13}$$

Let Win denote the winning condition (i.e., eq. (7)). Then we have

$$\begin{aligned}
\mathbf{Pr}\left[\mathsf{Win}\right] &\le \mathbf{Pr}\left[\mathsf{Win} \wedge \neg\mathsf{Hit}\right] + \mathbf{Pr}\left[\mathsf{Hit}\right] \\
&\le \mathbf{Pr}\left[\mathsf{Win} \wedge \neg\mathsf{Hit}\right] + \mathbf{Pr}\left[\mathsf{Hit} \mid \mathsf{E}\right] + \mathbf{Pr}\left[\neg\mathsf{E}\right] \\
&\le \mathbf{Pr}\left[\mathsf{Win} \wedge \neg\mathsf{Hit}\right] + \mathbf{Pr}\left[\mathsf{Hit} \mid \mathsf{E}\right] + \mathbf{Pr}\left[\neg\mathsf{Smooth}\right] + \mathbf{Pr}\left[|I_{\mathsf{final}}| > 2^{0.05\lambda}\right],
\end{aligned}$$

and thus the lemma follows from claim 39, eq. (12), eq. (13), and the following claim.

Claim 40. $\mathbf{Pr}\left[\mathsf{Win} \wedge \neg\mathsf{Hit}\right] \le 2\mathfrak{q}'^2 \cdot (\mathfrak{q} + \beta\mathfrak{q}') \cdot 2^{-\lambda}$.

Proof. For $j \in [\mathfrak{q}']$, define two events:

- Let Win_j denote the event that the j-th query in $\texttt{Query}^f(C, w^*)$ is the first query that lies in $\texttt{Sib}(f, Q_{\mathsf{final}})$.
- Let Hit_j denote the event that the j-th query in $\texttt{Query}^f(C, w^*)$ lies in I''.

Clearly, $\mathsf{Win} = \cup_{j\in[\mathfrak{q}']}\mathsf{Win}_j$ and $\mathsf{Hit}_j \subseteq \mathsf{Hit}$. Consequently,

$$\mathbf{Pr}\left[\mathsf{Win} \wedge \neg\mathsf{Hit}\right] \le \sum_{j\in[\mathfrak{q}']} \mathbf{Pr}\left[\mathsf{Win}_j \wedge \neg\mathsf{Hit}\right] \le \sum_{j\in[\mathfrak{q}']} \mathbf{Pr}\left[\mathsf{Win}_j \wedge \neg\mathsf{Hit}_j\right].$$

It suffices to bound $\mathbf{Pr}\left[\mathsf{Win}_j \wedge \neg\mathsf{Hit}_j\right]$ for all j's.

Fix $j \in [\mathfrak{q}']$ and w^*. WLOG, assume C has exactly $\mathfrak{q}'$ oracle gates. Let the $\boldsymbol{y}_t^*$ be the oracle answer of the t-th oracle gate when evaluating $C^f(w^*)$, where the probability space is defined by $f \leftarrow G_{\mathsf{final}}$. Then the events Win_j and Hit_j are determined by the random variables $(\boldsymbol{y}_1^*, \ldots, \boldsymbol{y}_j^*)$.

Let $\mathtt{Good}$ be the set of all sequences $(y_1^*, \ldots, y_j^*) \in \mathrm{supp}(\boldsymbol{y}_1^*, \ldots, \boldsymbol{y}_j^*)$ such that $\mathsf{Win}_j \wedge \neg\mathsf{Hit}_j$ happens if $(\boldsymbol{y}_1^*, \ldots, \boldsymbol{y}_j^*) = (y_1^*, \ldots, y_j^*)$. For $y_{<j}^* = (y_1^*, \ldots, y_{j-1}^*) \in (\{0,1\}^\lambda)^{j-1}$, let $x_1^*(y_{<j}^*), \ldots, x_j^*(y_{<j}^*)$ be the first j queries made by $C^{(\cdot)}(w^*)$ when the first $(j-1)$ oracle answers are $y_{<j}^*$, and let $\mathtt{Good}(y_{<j}^*) \stackrel{\mathsf{def}}{=} \left\{y \in \{0,1\}^\lambda : (y_{<j}^*, y) \in \mathtt{Good}\right\}$ and

$$\mathtt{DenseQ}(y_{<j}^*) \stackrel{\mathsf{def}}{=} \left\{t \in [j-1] : x_t^*(y_{<j}) \notin I_{\mathsf{final}} \cup Q_{\mathsf{final}}\right\}.$$

Then we have

$$\begin{aligned}
\mathbf{Pr}\left[\mathsf{Win}_j \wedge \neg\mathsf{Hit}_j\right] &= \sum_{(y_{<j}^*, y_j^*) \in \mathtt{Good}} \mathop{\mathbf{Pr}}_{f \leftarrow G_{\mathsf{final}}}\left[\forall t \in [j]\ f(x_t^*) = y_t^*\right] \\
&= \sum_{y_{<j}^* \in \mathrm{supp}(\boldsymbol{y}_1^*, \ldots, \boldsymbol{y}_{j-1}^*)} \sum_{y \in \mathtt{Good}(y_{<j}^*)} \mathop{\mathbf{Pr}}_{f \leftarrow G_{\mathsf{final}}}\left[\forall t \in [j]\ f(x_t^*) = y_t^*\right] \\
&= \sum_{s=0}^{j-1} \underbrace{\sum_{y_{<j}^* \in Y_s} \sum_{y \in \mathtt{Good}(y_{<j}^*)} \mathop{\mathbf{Pr}}_{f \leftarrow G_{\mathsf{final}}}\left[\forall t \in [j]\ f(x_t^*) = y_t^*\right]}_{\stackrel{\mathsf{def}}{=} p_s},
\end{aligned} \tag{14}$$

where $x_t^* \stackrel{\mathsf{def}}{=} x_t^*(y_{<j}^*)$ for all $t \in [j]$ and

$$Y_s \stackrel{\mathsf{def}}{=} \left\{y_{<j}^* \in \mathrm{supp}(\boldsymbol{y}_1^*, \ldots, \boldsymbol{y}_{j-1}^*) : |\mathtt{DenseQ}(y_{<j}^*)| = s\right\}.$$

We claim that $|Y_s| \le 2^{s\lambda}$ for all s. This is because, given G_{final} and w^*, each $y_{<j}^* \in Y_s$ can be uniquely encoded by s oracle answers

$$y' = (y_{t_1}^*, \ldots, y_{t_s}^*) \in (\{0,1\}^\lambda)^s \text{ where } \{t_1, \ldots, t_s\} = \mathtt{DenseQ}(y_{<j}) \text{ and } t_1 < \cdots < t_s.$$

Indeed, given y', one can evaluate $C^{(\cdot)}$ on w^* (until the j-th query is issued) using the fact that $y_{<j}^* \in \mathrm{supp}(\boldsymbol{y}_1^*, \ldots, \boldsymbol{y}_{j-1}^*)$, since G_{final} has fixed values on $I_{\mathsf{final}} \cup Q_{\mathsf{final}}$; thus $y_{<j}^*$ is recovered.

Fix s and $y_{<j}^* \in Y_s$. For Win_j to happen, it must be that $x_j^* \notin Q_{\mathsf{final}}$ and $f(x_j^*) \in f(Q_{\mathsf{final}})$; hence,

$$\mathtt{Good}(y_{<j}^*) \subseteq f(Q_{\mathsf{final}}),$$

which implies $|\mathtt{Good}(y_{<j}^*)| \le |Q_{\mathsf{final}}| \le \mathfrak{q} + \beta\mathfrak{q}'$. For Hit_j not to happen, it must be that $x_j^* \notin I''$. In sum, for $\mathtt{Good}(y_{<j}^*)$ to be non-empty, it must be that $x_j^* \notin I_{\mathsf{final}} \cup Q_{\mathsf{final}}$. Moreover, for Win_j to happen, x_j^* must be the first query such that $x^* \in \mathtt{Sib}(f, Q_{\mathsf{final}})$, and thus $x_j^* \notin \left\{x_1^*, \ldots, x_{j-1}^*\right\}$. Since G_{final} is η-dense on $\overline{I_{\mathsf{final}} \cup Q_{\mathsf{final}}}$, in particular, on $\left\{x_j^*\right\} \cup \mathtt{DenseQ}(y_{<j}^*)$, we have

$$\mathop{\mathbf{Pr}}_{f \leftarrow G_{\mathsf{final}}}\left[\forall t \in [j]\ f(x_t^*) = y_t^*\right] \le (2^\lambda)^{-\eta(s+1)} \le 2 \cdot 2^{-\lambda(s+1)}$$

holds for all $y_{<j}^* \in Y_s$ and $y_j^* \in \{0,1\}^\lambda$, where the last step follows from $\eta = 1 - \frac{1}{\lambda\mathfrak{q}'}$ and $s \le \mathfrak{q}'$. Consequently,

$$p_s \le 2 \cdot 2^{-\lambda(s+1)} \cdot \sum_{y_{<j}^* \in Y_s} |\mathtt{Good}(y_{<j}^*)| \le 2 \cdot (\mathfrak{q} + \beta\mathfrak{q}') \cdot 2^{-\lambda}.$$

Hence, we conclude that

$$\mathbf{Pr}\left[\mathsf{Win}_j \wedge \neg \mathsf{Hit}_j\right] \leq \sum_{s=0}^{j-1} p_s \leq \mathfrak{q}' \cdot 2(\mathfrak{q} + \beta\mathfrak{q}') \cdot 2^{-\lambda}.$$

This finishes the proof. □

□

Proof of lemma 35. The proof is analogous to that of lemma 34. Let Hit and Hit_j be as defined in the proof of lemma 34, and let Win′ denote the winning condition in eq. (8). Then by the same argument, it suffices to show that

$$\mathbf{Pr}\left[\mathsf{Win}' \wedge \neg \mathsf{Hit}\right] \leq 2\mathfrak{q}'^3 \cdot 2^{-\lambda}. \tag{15}$$

For $j \in [\mathfrak{q}']$, let Win'_j denote the event that j is the smallest index such that

$$\exists j' \in [j-1] \text{ s.t. } \left(x^*_j, x^*_{j'} \notin Q_{\mathsf{final}}\right) \wedge \left((x^*_j, x^*_{j'}) \in \mathtt{Coll}_2(f)\right),$$

where $(x^*_1, \ldots, x^*_{\mathfrak{q}'}) = \mathtt{Query}(C^f, w^*)$. Since $\mathsf{Win}' = \cup_{j \in [\mathfrak{q}']} \mathsf{Win}'_j$ and $\mathsf{Hit}_j \subseteq \mathsf{Hit}$, we have

$$\mathbf{Pr}\left[\mathsf{Win}' \wedge \neg \mathsf{Hit}\right] \leq \sum_{j=1}^{\mathfrak{q}'} \mathbf{Pr}\left[\mathsf{Win}'_j \wedge \neg \mathsf{Hit}_j\right].$$

Hence, it suffices to show that

Claim 41. *For all* $j \in [\mathfrak{q}']$, $\mathbf{Pr}\left[\mathsf{Win}'_j \wedge \neg \mathsf{Hit}_j\right] \leq 2\mathfrak{q}'^2 \cdot 2^{-\lambda}$.

Proof. Fix $j \in [\mathfrak{q}']$ and w^*. Let $\boldsymbol{y}^*_1, \ldots, \boldsymbol{y}^*_j, \mathtt{DenseQ}, Y_0, \ldots, Y_{j-1}$ be the same as in the proof of lemma 34. Let $\mathtt{Good}'$ be the set of all sequences $(y^*_1, \ldots, y^*_j) \in \mathrm{supp}(\boldsymbol{y}^*_1, \ldots, \boldsymbol{y}^*_j)$ such that $\mathsf{Win}'_j \wedge \neg \mathsf{Hit}_j$ happens if $(\boldsymbol{y}^*_1, \ldots, \boldsymbol{y}^*_j) = (y^*_1, \ldots, y^*_j)$. For $y^*_{<j} = (y^*_1, \ldots, y^*_{j-1}) \in (\{0,1\}^\lambda)^{j-1}$, let $x^*_1(y^*_{<j}), \ldots, x^*_j(y^*_{<j})$ be the first j queries made by $C^{(\cdot)}(w^*)$ when the first $(j-1)$ oracle answers are $y^*_{<j}$, and let $\mathtt{Good}'(y^*_{<j}) \stackrel{\mathsf{def}}{=} \left\{y \in \{0,1\}^\lambda : (y^*_{<j}, y) \in \mathtt{Good}'\right\}$. By the same calculation as in eq. (14), we have

$$\mathbf{Pr}\left[\mathsf{Win}'_j \wedge \neg \mathsf{Hit}_j\right] \leq \sum_{s=0}^{j-1} p'_s$$

where

$$p'_s \stackrel{\mathsf{def}}{=} \sum_{y^*_{<j} \in Y_s} \sum_{y \in \mathtt{Good}'(y^*_{<j})} \mathop{\mathbf{Pr}}_{f \leftarrow G_{\mathsf{final}}}\left[\forall t \in [j]\ f(x^*_t) = y^*_t\right].$$

Fix $s \in [j-1]$ and $y^*_{<j} \in Y_s$. For Win'_j to happen, it must be that (i) $x^*_j \notin Q_{\mathsf{final}}$, (ii) $f(x^*_j) \in \left\{y^*_1, \ldots, y^*_{j-1}\right\}$, and (iii) $x^*_j \notin \left\{x^*_1, \ldots, x^*_{j-1}\right\}$. Observe that (ii) implies

$$\mathtt{Good}'(y^*_{<j}) \subseteq \left\{y^*_1, \ldots, y^*_{j-1}\right\},$$

meaning that $|\mathtt{Good}'(y^*_{<j})| \le j-1 \le \mathfrak{q}'$. For Hit_j not to happen, it must be that $x^*_j \notin I''$. In sum, for $\mathtt{Good}'(y^*_{<j})$ to be non-empty, it must be that $x^*_j \notin I_{\mathsf{final}} \cup Q_{\mathsf{final}}$. Since G_{final} is η-dense on $\overline{I_{\mathsf{final}} \cup Q_{\mathsf{final}}}$, in particular, on $\left\{x^*_j\right\} \cup \mathtt{DenseQ}(y^*_{<j})$, we have

$$\Pr_{f \leftarrow G_{\mathsf{final}}}\left[\forall t \in [j]\ f(x^*_t) = y^*_t\right] \le (2^\lambda)^{-\eta(s+1)} \le 2 \cdot 2^{-\lambda(s+1)}$$

holds for all $y^*_{<j} \in Y_s$ and $y^*_j \in \{0,1\}^\lambda$. Consequently,

$$p'_s \le 2 \cdot 2^{-\lambda(s+1)} \cdot \sum_{y^*_{<j} \in Y_s} |\mathtt{Good}'(y^*_{<j})| \le 2 \cdot \mathfrak{q}' \cdot 2^{-\lambda},$$

where we recall that $|Y_s| \le 2^{\lambda s}$. Hence, we conclude that

$$\Pr\left[\mathsf{Win}'_j \wedge \neg \mathsf{Hit}_j\right] \le \sum_{s=0}^{j-1} p_s \le 2\mathfrak{q}'^2 \cdot 2^{-\lambda}.$$

This finishes the proof. □

□

Acknowledgment

We thank Siddhartha Jain and Zhiyang Xun for clarifying our questions on related TFNP problems.

References

[AS16] Gilad Asharov and Gil Segev. Limits on the power of indistinguishability obfuscation and functional encryption. *SIAM Journal on Computing*, 45(6):2117–2176, 2016. 1, 4, 7, 15

[BD19] Nir Bitansky and Akshay Degwekar. On the complexity of collision resistant hash functions: New and old black-box separations. In *Theory of Cryptography: 17th International Conference, TCC 2019, Nuremberg, Germany, December 1–5, 2019, Proceedings, Part I 17*, pages 422–450. Springer, 2019. 4, 7, 15

[BDRV18] Itay Berman, Akshay Degwekar, Ron D Rothblum, and Prashant Nalini Vasudevan. Multi-collision resistant hash functions and their applications. In *Advances in Cryptology–EUROCRYPT 2018: 37th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tel Aviv, Israel, April 29-May 3, 2018 Proceedings, Part II 37*, pages 133–161. Springer, 2018. 2, 8

[BGSD25] Huck Bennett, Surendra Ghentiyala, and Noah Stephens-Davidowitz. The more the merrier! on total coding and lattice problems and the complexity of finding multicollisions. In *16th Innovations in Theoretical Computer Science Conference (ITCS 2025)*, pages 14–1. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2025. 2

[BHKY19] Nir Bitansky, Iftach Haitner, Ilan Komargodski, and Eylon Yogev. Distributional collision resistance beyond one-way functions. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 667–695. Springer, 2019. 2

[BKP18] Nir Bitansky, Yael Tauman Kalai, and Omer Paneth. Multi-collision resistance: a paradigm for keyless hash functions. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, pages 671–684, 2018. 2

[BT24] Jan Buzek and Stefano Tessaro. Collision resistance from multi-collision resistance for all constant parameters. In *Annual International Cryptology Conference*, pages 429–458. Springer, 2024. 2

[CDGS18] Sandro Coretti, Yevgeniy Dodis, Siyao Guo, and John Steinberger. Random oracles and non-uniformity. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 227–258. Springer, 2018. 6

[CFK+19] Arkadev Chattopadhyay, Yuval Filmus, Sajin Koroth, Or Meir, and Toniann Pitassi. Query-to-communication lifting for bpp using inner product. In *46th International Colloquium on Automata, Languages, and Programming (ICALP 2019)*, pages 35–1. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2019. 6

[DFMT20] Yevgeniy Dodis, Pooya Farshim, Sogol Mazaheri, and Stefano Tessaro. Towards defeating backdoored random oracles: indifferentiability with bounded adaptivity. In *Theory of Cryptography Conference*, pages 241–273. Springer, 2020. 6

[DI06] Bella Dubrov and Yuval Ishai. On the randomness complexity of efficient sampling. In *Proceedings of the thirty-eighth annual ACM symposium on Theory of computing*, pages 711–720, 2006. 2

[DM24] Nico Döttling and Tamer Mour. On the black-box complexity of correlation intractability. In Venkatesan Guruswami, editor, *15th Innovations in Theoretical Computer Science Conference (ITCS 2024)*, volume 287 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 40:1–40:24, Dagstuhl, Germany, 2024. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. 4, 7, 15

[FGPR24] Noah Fleming, Stefan Grosser, Toniann Pitassi, and Robert Robere. Black-box ppp is not turing-closed. In *Proceedings of the 56th Annual ACM Symposium on Theory of Computing*, pages 1405–1414, 2024. 3

[GP17] Paul W Goldberg and Christos H Papadimitriou. Tfnp: an update. In *International Conference on Algorithms and Complexity*, pages 3–9. Springer, 2017. 4

[GPW17] Mika Göös, Toniann Pitassi, and Thomas Watson. Query-to-communication lifting for bpp. In *2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 132–143. IEEE, 2017. 6, 10

[HMW+25] Mi-Ying Miryam Huang, Xinyu Mao, Shuo Wang, Guangxu Yang, and Jiapeng Zhang. A min-entropy approach to multi-party communication lower bounds. In *40th Computational Complexity Conference (CCC 2025)*, pages 33–1. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2025. 6

[JLRX24] Siddhartha Jain, Jiawei Li, Robert Robere, and Zhiyang Xun. On pigeonhole principles and ramsey in tfnp. In *2024 IEEE 65th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 406–428. IEEE, 2024. 2, 3, 4, 8

[KNY18] Ilan Komargodski, Moni Naor, and Eylon Yogev. Collision resistant hashing for paranoids: Dealing with multiple collisions. In *Advances in Cryptology–EUROCRYPT 2018: 37th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tel Aviv, Israel, April 29-May 3, 2018 Proceedings, Part II 37*, pages 162–194. Springer, 2018. 2

[KY18] Ilan Komargodski and Eylon Yogev. On distributional collision resistant hashing. In *Annual International Cryptology Conference*, pages 303–327. Springer, 2018. 2

[MYZ24] Xinyu Mao, Guangxu Yang, and Jiapeng Zhang. Gadgetless lifting beats round elimination: improved lower bounds for pointer chasing. *arXiv preprint arXiv:2411.10996*, 2024. 6

[Pap94] Christos H Papadimitriou. On the complexity of the parity argument and other inefficient proofs of existence. *Journal of Computer and system Sciences*, 48(3):498–532, 1994. 2

[RV24] Ron D Rothblum and Prashant Nalini Vasudevan. Collision resistance from multicollision resistance. *Journal of Cryptology*, 37(2):14, 2024. 2

[Sim98] Daniel R Simon. Finding collisions on a one-way street: Can secure hash functions be based on general assumptions? In *Advances in Cryptology—EUROCRYPT'98: International Conference on the Theory and Application of Cryptographic Techniques Espoo, Finland, May 31–June 4, 1998 Proceedings 17*, pages 334–345. Springer, 1998. 1, 4

[Yog25] Eylon Yogev. On the status of mcrh separation. Personal communication, 2025. 2

[YZ24] Guangxu Yang and Jiapeng Zhang. Communication lower bounds for collision problems via density increment arguments. In *Proceedings of the 56th Annual ACM Symposium on Theory of Computing*, pages 630–639, 2024. 6

ECCC ISSN 1433-8092

https://eccc.weizmann.ac.il