UNIVERSITY OF CHICAGO

Time-Space Trade-offs in Cryptographic Primitives

by
AKSHIMA

A dissertation
submitted in thesis defense
for the degree of PhD
to the Department of Computer Science
of Physical Sciences Division
at the University of Chicago

Chicago, USA
September 2022

UNIVERSITY OF CHICAGO

Time-Space Trade-offs in Cryptographic Primitives

by

AKSHIMA

Approved by:

Dr. David Cash, Professor — Adviser

Computer Science

Dr. Aloni Cohen, Assistant Professor — Member of Committee

Computer Science

Dr. Hoeteck Wee, Senior Scientist — Member of Committee

NTT Research

Date of Dissertation Defense: September 15, 2022

Acknowledgements

Abstract of the Thesis of

Akshima for PhD
in Computer Science

Title: Time-Space Trade-offs in Cryptographic Primitives

The research in complexity theory, for a long time now, has been conscious of memory as a resource in building algorithms with improved asymptotic complexity. There is an understanding to compare time-memory trade-offs as opposed to only running times while choosing between algorithms to solve any problem. While cryptographers have recognized memory to be a resource, there has been little effort to analyze cryptographic primitives in a memory-conscious manner until recently.

This work contributes towards the recent efforts of understanding the role of memory in the security of cryptographic primitives. Our study is two-fold:

1. How much better can any adversary that is capable of performing pre-computation and storing a bounded amount of information about the cryptographic primitive (under attack) do?

2. Are there cryptographic applications which are provably more secure against adversaries with lesser memory?

This work focuses on cryptographic hash functions for the first part of the study. The study analyzes properties of collision resistance and multi-way collision resistance for these functions.

For the second part of the study, the aim is to analyze double encryption against the memory-bounded non-adaptive adversaries. It is known that *meet-in-the-middle* (MITM) adversary against double encryption runs in about the time required to brute force a single key, leading to the common-knowledge that double encryption is no more secure than the original block cipher. However, this is when the adversary is allowed to use as much memory as it would like. However, this is when the adversary is allowed to use as much memory as it would like.

Table of Contents

Terminology

Definition 1. ***Adversary*** *is a malicious entity that attempts to prevent any cryptosystem from achieving its goal.*

Definition 2. ***Cryptographic Hash Functions****, or simply referred to as Hash Functions, are functions that map inputs of arbitrary size to fixed size outputs, i.e.,* $\mathcal{H} : \{0,1\}^* \to \{0,1\}^\ell$ *for some fixed positive integer* ℓ*. They are one-way functions, which means they are hard to invert. Brute force search or rainbow tables are used for inverting these functions. Often times salted hash functions are used in applications to make them harder to invert by brute force search or using rainbow tables. Salted hash functions take an additional fixed size input called salt.*

Definition 3. *A* ***Collision*** *in a hash function* H *is defined as finding two distinct messages in* $\{0,1\}^*$ *that have the same output under* H*.* **m-*way Collision*** *in a hash function* H *is defined as finding* m *distinct messages* $\{0,1\}^*$ *that have the same output under* H *where* m *is a positive integer.*

Definition 4. ***Merkle Damgård*** *is a popular construction scheme for building a collision resistant Hash function for arbitrary input sizes from collision resistant compression functions on fixed input size. The scheme essentially breaks arbitrary sized inputs into blocks of fixed size and applies the compression function in sequence on these blocks.*

Definition 5. ***Random Oracle*** $\mathcal{O}$ *can be thought of as a machine implementing a function* H*. Its internal working are assumed unknown. An input to it is called* ***query****.*

Definition 6. ***Pseudo Random Functions*** *is a family of deterministic functions that are efficiently computable and indistinguishable from a truly random function by any efficient adversary.*

Definition 7. ***Pseudo Random Permutations*** *is a family of deterministic permutations that are efficiently computable and indistinguishable from a truly random permutation by any efficient adversary.*

Definition 8. *To prove security of an application using H in the* ***Random Oracle Model****, analysis assumes H to be a truly random function.*

To prove security of an application using H in the ***Auxiliary-Input Random Oracle Model****, analysis assumes H to be a truly random function and 2-stage adversaries $\mathcal{A} = (\mathcal{A}_0, \mathcal{A}_1)$ such that:*

1. *$\mathcal{A}_0$ gets computationally unbounded access to H and outputs a bounded size information about H, say of S-bits, and called advice.*
2. *$\mathcal{A}_1$ takes the advice output by $\mathcal{A}_0$ and the challenge as input and gets to make a bounded number of queries, say T to H to solve the challenge.*

Definition 9. *To prove security of an application using H in the* ***Bit-Fixing Random Oracle Model****, analysis assumes H to be a truly random function and 2-stage adversaries $\mathcal{A} = (\mathcal{A}_0, \mathcal{A}_1)$ such that:*

1. *$\mathcal{A}_0$ fixes some bounded number of bits of H, say P bits, to obtain say H'*
2. *$\mathcal{A}_1$ takes the challenge as input and gets to make a bounded number of queries, say T queries, to H' to solve the challenge for H'.*

Definition 10. *A* ***streaming adversary*** *is characterized by two parameters m, q such that the adversary gets to make q queries and receives the responses in a stream, i.e., the adversary cannot access the response to a particular query unless it stores that in its memory, which is bounded to m-bits.*

Definition 11. ***Block ciphers*** *are all functions $E : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$ such that for every $K \in \{0,1\}^k$, $E_K := E(K, \cdot)$ is a permutation on $\{0,1\}^n$. For a block cipher E, we write E^{-1} to denote the inverse cipher, i.e. for every $K \in \{0,1\}^k$, $E_K^{-1}(\cdot)$ is inverse permutation of $E_K(\cdot)$.*

Definition 12. *For a block cipher $E : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$, we define* ***Double Encryption*** *with E as a function in $\{0,1\}^{2k} \times \{0,1\}^n \to \{0,1\}^n$ that on inputs $\hat{K}_1 || \hat{K}_2, x$ returns $E_{\hat{K}_2}(E_{\hat{K}_1}(x))$.*

Abbreviations

PRP	Pseudo Random Permutation
PRF	Pseudo Random Function
RO	Random Oracle
AI-RO	Auxiliary Input Random Oracle
BF-RO	Bit Fixing Random Oracle
CR	Collision Resistance
MCR	Multi-way Collision Resistance
MD	Merkle Damgård
DE	Double Encryption

Thesis Outline

Chapter 1 gives an introduction to the works presented in the thesis. Chapter 2 gives a summary of the related work for the following chapters. Chapters 3 and 4 give an overview of the works already published in conferences ([1], [2]).

Chapter 3 talks about our results for 2-block collisions for MD based hash functions in the AI-RO model. Chapter 4 talks about our follow-up results for any B-block collisions, again for MD based hash functions in the AI-RO model.

Chapters 5 and 6 present new unpublished works. First of the two is a work in progress. Chapter 5 talks about m-way collisions for MD based hash functions in the AI-RO model and our results so far.

Chapter 6 is a proposed work contributing towards the second part of this study. The chapter focuses on analyzing double encryption scheme as a PRP.

Chapter 1

Introduction

1.1 Part 1

The works presented in this thesis consider the security of random-oracle-based hash functions against *preprocessing* adversaries which have a bounded amount of arbitrary auxiliary information on the random oracle to help them. Attacks in this model were first considered by Hellman [3], who gave a heuristic time-space tradeoff for inverting cryptographic functions. We would like to understand the power of these attacks in the context of finding collisions in hash functions, and in particular, salted hash functions based on the widely used Merkle-Damgård paradigm (MD).

Finding short collisions. In the works, we focus on understanding the best attacks for finding *short* collisions, as motivated by real-world applications. Concretely, across two works we put forth and study the following conjecture:

> *STB conjecture*: The best attack with time T and space S for finding collisions of length B in salted MD hash functions built from hash functions with n-bit outputs achieves success probability $\Theta((STB+T^2)/2^n)$.

The birthday attack achieves $O(T^2/2^n)$, and we will describe an attack from [1] that achieves $\Omega(STB/2^n)$. Short of proving circuit lower bounds, we cannot hope to rule out better attacks, except in idealized models, where we treat the underlying hash function as a random oracle.

The AI-RO model. We use the *auxiliary-input random oracle* (AI-RO) model introduced by Unruh [4], which was originally motivated by dealing with the non-uniformity of adversaries that is necessary for some applications of the random-oracle model [5]. In the AI-RO model, two parameters S, T are fixed, and adversaries are divided into two stages $(\mathcal{A}_1, \mathcal{A}_2)$: The first has unbounded access to a random function h, and computes an *auxiliary input* (or advice string) for $\mathcal{A}_2$, σ of length S bits. Then the second stage accepts σ as input, and gets T queries to an oracle computing h, and attempts to accomplish some goal involving the function h. We typically think of the adversaries as information-theoretic and ignore runtime. Adversaries in the AI-RO correspond roughly to circuits of size $O(S+T)$.

Salted-collision resistance of MD hash functions at the AI-RO model was first studied by Coretti, Dodis, Guo and Steinberger (CDGS) [6]. They proved the STB

conjecture in the setting $B = T$, showing an attack with success probability $ST^2/2^n$ and proving its optimality.

Collision-resistance in the AI-RO. We consider salted collision resistance following Dodis, Guo and Katz [7], in order to rule out trivial attacks where the adversary hardwires a collision on h. Assume, as we shall for the rest of the paper, that the function has the form $h : [N] \times [M] \to [N]$, where $[N] = [2^n]$ and $[M] = [2^m]$, which we identify with ^{n} and ^{m} respectively. In salted collision-resistance in the AI-RO model, the second stage adversary gets in addition as input a random "salt" $a \in [N]$ and must find $\alpha \neq \alpha' \in [M]$ such that $h(a, \alpha) = h(a, \alpha')$. The prior work obtained a bound of $O(S/N + T^2/N)$ on the success probability of any adversary, which is optimal (their result actually covers a wider parameter range and different forms of h that are not relevant for our results here). These results were interestingly proven via *compression arguments* [8], [9], where it is shown that an adversary that is successful too often can be used to compress uniformly random strings, which is impossible (cf. [10] for other applications of encoding arguments in computer science and combinatorics).

In order to better model in-use hash functions, the afore-mentioned work of Coretti, Dodis, Guo and Steinberger examined the salted-collision resistance of an MD hash function built from h in the AI-RO model [6]. Here, the first stage adversary works as before, but the second adversary only needs to find a collision in the iterated MD function built from h, starting at a random salt; We give precise definitions in the next section. That work showed that finding these collisions is substantially easier, giving an attack and matching lower bound of $O(ST^2/N)$. This was surprising in a sense, as it shows there exists an $S = T \approx 2^{60}$ attack against a hash function with 180-bit output, well below the birthday attack with $T \approx 2^{90}$.

A closer look at this attack reveals that the collisions it finds are very long (on the order of T blocks), so in our example the colliding messages each consist of 2^{60} blocks. While technically violating collision-resistance, this adversary is not damaging in any widely-used application we are aware of, as the colliding messages are several petabytes long. Addressing whether or not this attack, or the lower bound, can be improved to find shorter collisions is the starting point for our work.

The results of [6] did not use compression. Instead they applied a tightening of the remarkable and powerful *bit-fixing* (or *presampling*) method of Unruh [4], which we briefly recall here. In the bit-fixing random oracle (BF-RO) model, the adversary no longer receives an advice string. Instead the first stage adversary can fix, a priori, some bits P of the table of h. Then the rest of h is sampled, and the second stage attempts to find a salted collision as before. Building on Unruh's results, Coretti et al. showed (very roughly) that a bound of $O((T + P)T/N)$ on the advantage of any adversary in the BF-RO implies a bound of $O(ST^2/N)$ in the AI-RO. Moreover, the BF-RO bound was very easily proved, resulting in a simple and short proof.

1.1.1 *Chapter 3[1]*

Motivated by real-world hash functions like SHA-256, where $N = 2^n = 2^{256}$, $M = 2^{512}$, we are interested in parameter settings with $B \ll T$, such as $S = 2^{70}$, $T = 2^{95}$

and $B = 2^{18}$, which corresponds to computing a 256-bit digest of a 16MB message using SHA-256. Here, (ignoring constants) the CDG bound is meaningless, since $ST^2/N > 1$, whereas the corresponding attack achieves constant advantage when $T = B \approx 2^{93}$, collisions which are several yottabytes ($= 10^{24}$ bytes) long.

The starting point of this work is the observation that Hellman's attack (or an easy modification of the attack in [6]) can find length-B collisions with success probability roughly STB/N. We make this formal in Section 3.2.

While the attack was easy to modify for short collisions, proving that it is optimal is an entirely different matter with significant technical challenges. In order to explain them, we recall the approach of [6] used to prove the $O(ST^2/N)$ bound for salted MD. They used a technical approach (with tighter parameters) first developed by Unruh [4], which connects the AI-RO model to the *bit-fixing random oracle (BF-RO)* model (we defer the definition to the next section). Their work transfers lower bounds in the BF-RO model to lower bounds in the AI-RO model.

We show that the BF-to-AI template inherently cannot give a lower bound for finding short collisions, *because finding short collisions in the BF-RO model is relatively easy.* That is, the lower bound of the form we would need for BF-RO model simply does not hold. In the notation introduced above, we would need to show that no adversary finding length-2 collisions can do better than $O((P + T)/N)$ advantage, but we give a simple attack in BF-RO model that finds length-2 collisions with advantage $\Omega(PT/N)$. Thus another approach is required.

Our lower bound technique. Given that the BF-to-AI technique cannot distinguish between short and long collision finding, we must find another approach. There are two options from the literature: The previously-mentioned compression arguments, and another lesser-known but elegant method of Impagliazzo using concentration inequalities.

Compression arguments which were previously observed [6] to be difficult (or "intractable") to apply to the setting of salted MD collision finding despite working in the original non-MD setting [7]. Given that compression was already difficult in this setting, it does not seem promising to extend it to the harder problem of short collisions.

To address these difficulties, we introduce a new technique that first applies a variant of the "constructive" Chernoff bound of Impagliazzo and Kabanets [11] to prove time-space tradeoff lower bounds.

The concentration-based approach to time-space tradeoff lower bounds was, to our knowledge, first introduced by Impagliazzo in an unpublished work, and then later elucidated in an appendix [12] (there an older work of Zimand [13] is also credited). The high-level idea is to first prove that any adversary (with no advice) can succeed on any fixed $U \in [N]$ of $\Omega(S)$ of inputs with probability $\varepsilon^{\Omega(S)}$. (In some sense bounding every sufficiently large "moment of the adversary"). The argument continues by applying a concentration bound to the random variable that counts the number of winning inputs for this adversary, showing that it wins on a $O(\varepsilon)$-fraction of inputs except with probability $2^{-\Omega(S)}$. In a final elegant step, one shows that every advice string is likely to be bad via a union bound over all possible 2^S advice strings, to get a final bound of ε.

The technique of Impagliazzo gives a direct and simple proof for the optimal bound on inverting a random permutation. There are two issues in applying it to short MD collisions however. First, as we formally show later, it provable fails for salted MD hashing. The issue is that the adversary may simply succeed with probability greater than ε^S on some subsets U, so the first step cannot be carried out.

We salvage the technique by showing it is sufficient to bound the adversary's average advantage for *random* subsets U rather than *all* subsets. In the language of probability, we use a concentration bound that only needs *average* of the moments to be bounded by $\varepsilon^{\Omega(S)}$, rather than all of the moments; see Theorem 2.

So far we have been able to reduce the problem of proving a lower bound in AI-RO model to the problem of bounding the probability that an adversary with no advice can succeed on every element of a random subset of inputs. For the problems we considered, even this appeared to be complicated. To tame the complexity of these bounds, we apply compression arguments; Note that we are only proving the simpler bound needed for the Impagliazzo technique, but using compression, when previously compression was used for the problem directly. Our variation has the interesting twist that we can not only compress the random function (as other work did), but also the random subset U on which the adversary is being run. This turns out to vastly simplify such arguments.

Applications of our technique. We first apply our technique to reprove the $O(ST^2/N)$ bound for (non-short) collision finding against salted MD hash functions. We then turn to the question of short collisions. Proving a general bound (perhaps $O(STB/N)$) for finding length-B collisions appears to be very difficult, so we start by examining the first new case of $B = 2$.

We show that there are qualitative gaps between finding length 1 collisions, length 2 collisions, and arbitrary-length collisions. Specifically, while for length 1 collisions we have $\varepsilon = O((S + T^2)/N)$, we show that length 2 collisions are easier when $S > T$, as the optimal bound is $O((ST + T^2)/N)$. For arbitrary-length collisions there is another gap, where the optimal bound is $O(ST^2/N)$. Our bound for length 2 collisions uses our new compression approach used above.

It appears that we could, in principle, obtain similar bounds for other small length bounds like 3 and 4, but these proofs appear to be too long and complex for us to write down; Going to arbitrary length bounds seems to be out of reach, but there is no inherent obstruction in applying our technique to the general case with new ideas.

Bound for a restricted class of attacks. Given the difficulty of proving the general case, we instead consider ruling out the class of attacks that gives optimal attacks in the known cases. Roughly speaking, these attacks use auxiliary information consisting of S collisions at well-chosen points in the function graph. In the online phase, the attack repeatedly tries to "walk" to these points by taking one "randomizing" step followed by several steps with zero-blocks.

For this class of attacks, we show that the best choice of collision points will result in $\varepsilon = O(STB/N)$. This result requires carefully analyzing the size of large, low-depth trees in random functional graphs, a result that may be of independent

interest.

1.1.2 *Chapter 4 [2]*

Following up on the previous work that combined with known results for $B = 1$ and $B = T$, demonstrated qualitative jumps in the optimal attacks for finding length 1, length 2, and unbounded-length collisions, very recently, Ghoshal and Komargodski [14] confirmed STB conjecture for all constant B. However, for other choices of B, there is still a significant gap between the best-known attack [1] and known security upper bound $\tilde{O}(S^4TB^2/N + T^2/N)$ by [14] or $\tilde{O}(ST^2/N)$ by [6]. That motivates us to study the following question in [2]:

Can we further bridge the gap between the security upper and lower bounds, and prove STB conjecture for more choices of parameters?

Since prior techniques are limited or laborious even for $B = 2$, we start by asking:

Can we prove STB conjecture for $B = 2$ in a simpler way?

Looking ahead, we answer both questions affirmatively.
Our main contribution is the following theorem.

Theorem 1 (Informal)**.** *For any $2 < B < T$, the advantage of the best adversary with S-bit advice and T queries for finding B-block collisions in Merkle-Damgård hash functions in the auxiliary-input random oracle model, is*

$$\tilde{O}\left((STB/N) \cdot \max\{1, ST^2/N\} + T^2/N\right).$$

Our bound confirms the STB conjecture for any $2 < B < T$ for the range of S, T such that $ST^2 \leq N$. For the other range of S, T, as $T^2 \leq N$ (otherwise, finding a collision is trivial by the birthday attack), Our bound is at most $\tilde{O}(S^2TB/N + T^2/N)$, which is optimal up to a factor of S.

Comparing to the $\tilde{O}(STB^2(\log^2 S)^{B-2}/N + T^2/N)$ bound by [14], our bound works for any $2 < B < T$, while their bound becomes vacuous when $B > \log N$. However, for $B \leq \log N$, unlike our bound, their bound could be tight even when $ST^2 > N$. In particular, their bound confirms STB conjecture for $B = O(1)$.

Our bound strictly improves the $\tilde{O}(S^4TB^2/N + T^2/N)$ bound by [14], and the $\tilde{O}(S^2T/N)$ bound by [6] for any $2 < B < T$ and non-trivial choices of S, T (specifically, when STB attack succeeds with at most a constant probability, i.e., $STB = O(N)$). The two bounds by [14] only beat [6] for $B \ll \sqrt{T}$.

A comparison of all the relevant results is summarized in 4.1. Overall, our results subsume all previous upper bounds except for the range of S, T, B such that $B \leq \log N$ and $ST^2 > N$.

Our techniques In this section, we describe our techniques, how to use them to prove our main results, and what makes our techniques different from prior approaches used in [1], [6], [14].

	Best known attacks	Security bounds	Ref.	Proof techniques
$B = 1$	$\frac{S}{N} + \frac{T^2}{N}$	$\frac{S}{N} + \frac{T^2}{N}$	[7]	Compression
$B = 2$	$\frac{ST}{N} + \frac{T^2}{N}$	$\frac{ST}{N} + \frac{T^2}{N}$	Theorem 25[1]	Multi-instance problems
$B = 2$	$\frac{ST}{N} + \frac{T^2}{N}$	$\frac{ST}{N} + \frac{T^2}{N}$	Theorem 23[2]	Multi-instance games
$2 < B < T$	$\frac{STB}{N} + \frac{T^2}{N}$	$\frac{STB^2(\log^2 S)^{B-2}}{N} + \frac{T^2}{N}$	[14]	Multi-instance problems
$2 < B < T$	$\frac{STB}{N} + \frac{T^2}{N}$	$\frac{S^4TB^2}{N} + \frac{T^2}{N}$	[14]	Multi-instance problems
$2 < B < T$	$\frac{STB}{N} + \frac{T^2}{N}$	$\frac{STB}{N} \cdot \max\{1, \frac{ST^2}{N}\} + \frac{T^2}{N}$	Theorem 22[2]	Multi-instance games
Unbounded	$\frac{ST^2}{N}$	$\frac{ST^2}{N}$	[6]	Presampling

Table 1.1: Asymptotic security bounds on the security of finding B-block-long collisions in Merkle-Damgård Hash Functions constructed from a random function $H : [N] \times [M] \mapsto [N]$ against (S, T)-algorithms. For simplicity, logarithmic terms and constant factors are omitted.

Existing reduction to sequential multi-instance games. Our initial inspiration is the recent framework of Chung, Guo, Liu, Qian [15] for establishing tight time-space tradeoffs in the quantum random oracle model. Generally speaking, they reduce proving the security of a problem with S-bit advice to proving the security of multiple random instances of the problem, presented one at a time, *without* advice. Specifically, they observe that[1], if any adversary (with no advice) can solve S instances of the problem "sequentially" with success probability at most δ^S, then any adversary with S-bit advice can solve one instance of the problem with success probability at most 2δ.

[1]The framework of Chung, Guo, Liu, Qian [15] reduces to analyzing sequential multi-instance security for $S + \log N + 1$ instances instead of S-instances. We slightly improve their parameters and obtain a considerably cleaner version in Theorem 24.

This idea of reducing the security of a problem with advice to the security of a multi-instance problem without advice was first introduced by Impagliazzo and Kabanets in [11]. The idea was also used by later works [1], [14]. The difference between [11] and the later works, including this work, is that we reduce to a "sequential" multi-instance game as opposed to a "parallel" multi-instance problem. More concretely, in the parallel multi-instance problem, the adversary is presented with all the randomly chosen instances of the challenge problems to solve once at the start. Whereas in the multi-instance game, the adversary gets a new randomly chosen instance of challenge problem one at a time and only after solving all the previous challenges.

Chung et al. [15] recently demonstrated a separation between "sequential" multi-instance games and "parallel" multi-instance problems in the context of function inversion in the quantum setting[2]. Guo, Li, Liu and Zhang [16] pointed out a connection between "sequential" multi-instance game and the presampling technique (first introduced by Unruh [4], and further optimized by Coretti et al. [6]) —— the main technique used by Coretti et al. [6] for proving the $O(ST^2/N)$ bound. Roughly speaking, all results relying on presampling technique can be reproved using "sequential" multi-instance games. That suggested that "sequential" multi-instance games have the potential to prove stronger results. Therefore we are motivated to adapt and take full advantage of "sequential" multi-instance games in the context of collision finding.

To better illustrate the connection between "sequential" multi-instance games and the presampling technique, we show how to recover the $O(ST^2/N)$ bound by Coretti et al. [6]. Recall that presampling technique by Coretti et al. [6] generically reduces security proofs of unpredictability applications (including collision finding) in the AI-ROM to a much simpler P-bit-fixing random-oracle model (BF-ROM), where the attacker can arbitrarily fix the values of the random oracle on some $P := O(ST)$ coordinates, but then the remaining coordinates are chosen at random. Coretti et al. [6] showed that the security of finding collisions in Merkle-Damgård Hash Functions in the BF-ROM is $O(ST/N)$.

Using "sequential" multi-instance games, it suffices to bound the advantage of any adversary (with no advice) winning a new game, conditioning on winning all previous (up to at most S) ones, by $O(ST^2/N)$. The adversary wins all games with advantage $O(ST^2/N)^S$, which implies the desired security against S-bit advice. The key point is that the adversary (with no advice) made at most ST queries in previous games. Therefore, conditioning on any possible events of earlier games, from the view of the adversary, the random oracle is essentially a (convex combination of) bit-fixing random oracles (BF-ROM) [6], where at most ST-positions are known, and the rest remains independent and random. Hence, it suffices to prove the security of a single game in BF-ROM by $O(ST^2/N)$, which has been shown by Coretti et al. [6] as a necessary step to use the presampling technique.

[2]In particular, they showed that "sequentially" inverting S random images (with T quantum queries per round to a given random function $f : [N] \to [N]$) admits security $O(ST/N + T^2/N)^S$, and the corresponding "parallel" multi-instance problems admits an attack with advantage $\Omega(ST^2/N)^S$.

Barriers of the above idea. Akshima et al. [1] pointed out a barrier to using the vanilla presampling technique towards proving $B = 2$. In particular, one can only hope to achieve $\Omega(ST^2/N)$ in the BF-ROM even for $B = 2$. Recall that, to prove the sequential multi-instance security, it is sufficient to bound the advantage of any adversary that finds a 2-block collision for a fresh salt a, conditioned on it finds 2-block collisions for all the previous random challenge salts $a_1, \cdots, a_S$.

We will call these ST queries made during the first S rounds as offline queries. Among the T queries made for a, we will call the queries that were not made during the first S rounds as online queries. Throughout the discussion, we will focus on the case that the new salt a has never been queried before in offline queries, because the other case happens with probability at most ST/N (so won't affect our conclusion). As a result, all queries starting with the challenge salt a have to be online queries.

It is clear that the adversary learns about the function not only using the online queries but also from the offline queries. The information this algorithm can take advantage of from the offline queries varies by a lot. The followings are two extreme cases:

1. The offline queries consist of exactly one single query for each of ST distinct salts.

2. The offline queries consist of one collision for each of $ST/2$ distinct salts

For the first case, the offline queries can barely help[3]. Whereas, in the second case, as long as an adversary can find a pre-image (starting with the challenge salt a) of any of these $ST/2$ salts, it finds a 2-block collision (4.1). Since there are T online queries, the algorithm achieves advantage at least $ST^2/(2N)$ in the second case.

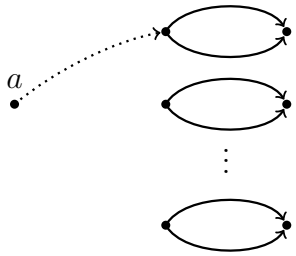


Figure 1.1: Nodes indicate salts in $[N]$. An arrow connected two salts means there is a query on the starting salt and a message in $[M]$ such that the output is the other salt. An online query hits an existing collision. Solid lines denote offline queries. The dotted line denotes the online query that forms a 2-block collision.

The vanilla presampling approach works for worst-case offline queries. Given the above example, the best security bound one can hope to achieve in the BF-ROM for $B = 2$ is $\Omega(ST^2/N)$.

[3]We do not prove it rigorously here. Instead, we focus on the more interesting case – offline queries do provide advantages.

Our main technical novelty. Our main insight is that, unlike the presampling technique in which offline queries can be arbitrary, the worst offline queries are not typical and can be tolerated by refining the technique. In the above example, the chance that offline queries form $ST/2$ pairs of collisions is quite unlikely. We define the following "high knowledge gaining" event $\mathbf{E}_1$:

> $\mathbf{E}_1$: By making ST queries, there are more than S distinct salts with 1-block collision.

The name "high knowledge gaining" suggests that whenever this event happens, the online algorithm can behave significantly better than average (following the attack in 4.1). If this event $\mathbf{E}_1$ does not happen, the probability that an online algorithm finds a query hitting an existing offline collision is bounded by $O((S/N) \cdot T)$; it is much better compared to the worst case – which is $O(ST^2/N)$. Remember that we have not shown how to prove that $\mathbf{E}_1$ happens with a tiny probability. We will not do that in this section since this is not our main technical novelty.

We then show two more "high knowledge gaining" events, which are all the events we consider. Conditioned on none of them happens, no online algorithms can find 2-block collisions with advantage better than $O(ST/N + T^2/N)$. The second event $\mathbf{E}_2$ is defined as:

> $\mathbf{E}_2$: By making ST queries, there are more than S^2 pairs of queries forming collisions.

In 1.2a, we denote a multi-collision by a claw. $\mathbf{E}_2$ says that many pair-wise collisions are found among all the offline queries. $\mathbf{E}_1$ only cares about collisions starting with the same salt, whereas $\mathbf{E}_2$ counts every pair of collisions (even starting with distinct salts). If there are many pairs of collisions, as long as an online adversary can hit two queries that form a collision, it finds a 2-block collision. The probability that an online algorithm having two queries hitting one particular existing collision is at most $O(T^2/N^2)$; if $\mathbf{E}_2$ does not happen, by union bound, the advantage of this type of attack is bounded by $O(S^2 \cdot (T^2/N^2))$, again smaller than $O(ST/N)$.

(a) $\mathbf{E}_2$ (b) $\mathbf{E}_3$

Figure 1.2: Other two "high knowledge gaining" events and their corresponding attacks.

The final event $\mathbf{E}_3$ is very similar to $\mathbf{E}_1$:

> $\mathbf{E}_3$: By making ST queries, there are more than S distinct salts with self-loops.

If an online algorithm hits an offline self-loop, it forms a 2-block collision. Following the same reasoning as $\mathbf{E}_1$, if $\mathbf{E}_3$ does not happen, the probability that an online algorithm finds a query hitting an existing self-loop is bounded by $O((S/N) \cdot T)$.

By identifying the "high knowledge gaining" events and managing to show that they are all unlikely (which is intuitive but non-trivial to prove), we obtain a considerably simpler proof for the $B = 2$ result from [1] using our approach in 4.2 for illustration. More precisely, with all these "high knowledge gaining" events, we show that[4]: (1). these events happen with probability at most $O(N^{-S})$, even conditioned on the adversary winning all the previous rounds;(2). when none of them happens, an online algorithm making T queries can find a 2-block collision with advantage $O(ST/N+T^2/N)$: such a 2-block collision will consist of either hybrid queries (both online and offline queries) or solely online queries; but for both cases, the probability is small.

It is an upside of our technique that it modularises and separates the bad events, making the overall proof more straightforward and intuitive. Following the same structure, we then extend our proof to larger B by identifying a few events, and obtaining our main result.

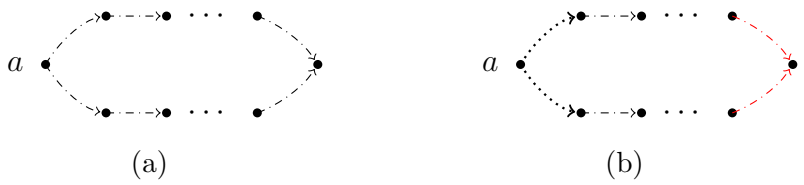
Applying our new techniques to larger B. As for $B = 2$, we present results for the sequential multi-instance model and use the reduction to prove results in the auxiliary input model. We simplify the sequential multi-instance model into the offline phase and online phase as in the $B = 2$ result and again use our insight that worst offline queries are unlikely and better bounds than $O(ST^2/N)$ can be achieved using a more refined analysis. However, unlike for $B = 2$ analysis, our larger B analysis is not as straightforward and requires some creative case analysis in terms of collision types.

We call offline queries that share an image under H with other offline query/ queries as marked queries. We define the following "high knowledge gaining" event:

> **E**: By making ST queries, there are more than κ marked queries where $\kappa = S \cdot \max\{1, ST^2/N\}$.

We can show that this event happens with probability at most $O(N^{-S})$, even conditioned on the adversary finding B-length collisions in all the previous rounds. When event E does not happen, there are two possibilities: 1) The B-length collisions found 'use' at least one of these (at most) κ marked queries 2) The B-length collisions found 'use' none of those κ marked queries. For case (1), we will show that some online query should hit one of (at most) $\kappa \cdot B$ offline queries en route to one of κ queries within B steps to succeed, and this happens with probability at most $O(\kappa TB/N)$. For case (2), note that it implies at least one of the two 'colliding' queries among the B-length collisions is a 'new' online query. Then, using this fact along with the structural knowledge of the type of B-length collision, we can show

[4]This is not a formal argument but captures the intuition behind our technique. For the formal proofs, please refer to 4.2.



(a) (b)

Figure 1.3: Dotted lines denote online queries. Solid lines denote offline queries. Dash-dotted lines can be either offline or online queries. Red lines denote 'colliding' queries.

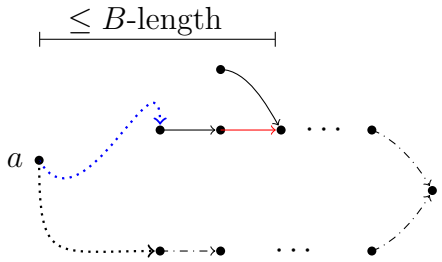


Figure 1.4: The B-length collision uses some marked query. The solid red line denotes the *first* marked query along the B-length collisions. The dotted blue line denote the *closest* online query to the red line along the B-length collisions.

that probability of finding any of these types of B-length collisions is bounded by $O(STB/N + T^2/N)$.

Here, we focus on one type of B-length collisions to reiterate our strategy with more details. Refer to Section 4.3 for the complete proof. Consider the type of B-length collision depicted in Figure 1.3a on input salt a.

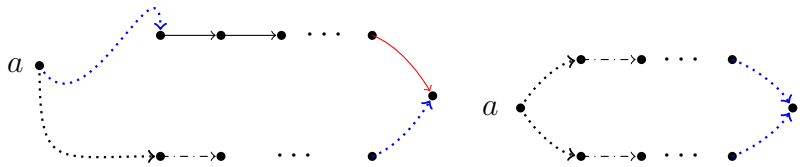
First, as we have discussed at the beginning of the section, note that the probability that the input salt a has been queried in the offline queries is at most ST/N (as a is randomly and independently sampled). So, it suffices to focus on the case that a has not being queried during offline queries depicted in Figure 1.3b. For this case, there should exist some queries (including the queries on a) along with the outputted B-length collisions that are online queries (i.e., made for the first time during the online phase).

In addition, we can also condition on event E not happening as we can show that the probability of event E is at most $O(N^{-S})$, even conditioned on the adversary winning all the previous rounds. Now observe that the queries in any found this type of B-length collisions would satisfy one of the two following possibilities:

1. The B-length collision uses some marked query.

2. None of the offline queries used by B-length collision is a marked query.

We first analyze B-length collisions with queries satisfying (1) above. Refer to Figure 1.4 for a pictorial depiction of such collisions. Conditioned on event E not happening, there will be at most κ marked queries. Consider the first such query



(a) (b)

Figure 1.5: The B-length collision uses no marked queries. The solid red line (if any) denotes the colliding query made in the offline phase. The dotted blue lines denote the two closest online queries to the colliding queries along the B-length collisions (they can also be colliding queries themselves).

along the B-length collisions. There is a unique 'chain' consisting of at most B offline queries connecting some online query to this marked query. Thus, the probability of finding B-length collisions satisfying (1) conditioned on event $\overline{E}$ is at most the probability of some online query whose output is one of (the salts of) these κB offline queries, which is at most $O(\kappa TB/N)$.

Note that when queries in the B-length collision satisfy (2) above, it implies at least one of the 'colliding queries' (two queries denoted by red arrows in Figure 1.3b) is made for the first time in the online phase.

The probability of both the colliding queries happening for the first time in the online phase (see Figure 1.5b) is bounded by $O(T^2/N)$.

In the case exactly one of the colliding queries happens in the offline phase, there are at most ST possibilities for this offline colliding query. There is a unique 'chain' of at most B offline queries from some online query to this query and the output of another online query should be the output of this query (see Figure 1.5a). Thus, the probability of finding such B-length collisions is bounded by $O(STB \cdot T/N \cdot T/N) = O(STB/N + T^2/N)$.

For other types of B-length collisions, we can analyze each type in a similar way. Instead of analyzing each type of B-length collisions, we further abstract out 5 conditions such that any type of B-length collisions must satisfy one of them. By considering one more "high knowledge gaining" event, and upper bounding the probability for every condition, we show that the probability of finding B-length collisions is bounded by $O(\kappa TB/N + T^2/N)$. Please see Section 4.3 for the details. It is worth noting that the S^2T^2/N term in κ cannot be further improved, because it is expected to have $\Omega(S^2T^2/N)$ marked queries among ST random oracle queries. Thus, it seems unlikely to obtain a better bound by just improving event E and its analysis.

A detailed comparison with prior techniques. The similarity between [1], [14] and us is that we all adopt the idea of reducing the problem of interest to a multi-instance variant, in which an adversary has to solve multiple copies of the given problem.

Both [1] and [14] directly analyze the probability of solving all instances using the compression paradigm, which typically requires a non-trivial case analysis of the more complicated *multi-instance* problem. These case analyses may be quite laborious and detached from the single-instance problem (thus may not give many insights for the single-instance problem).

Our approach differs significantly from [1] and [14] in two places. First, we focus on analyzing a simple variant of the *single-instance* problem (corresponding to a single round of the sequential multi-instance game conditioning on winning previous games), which is sufficient to establish desired results in multi-instance security. This variant is more similar to the original problem, and may be easier to analyze than the multi-instance problems. The first step (reducing to a variant of the single-instance problem) is somewhat used and captured in the presampling technique (via a different route [6]). We do think this step is more modular than [1] and [14], but don't consider this as our main technical novelty.

The second place, also our main technical novelty, is that we further introduce "knowledge gaining events" for analyzing the variant of the single-instance problem. These events can be isolated and analyzed on their own, and precisely highlight the correlation in finding collisions given "typical" presampled random oracles. Before this work, all the presampling techniques for time-space tradeoffs considered worst-case presampled random oracles. The worst-case presampling may make the existing analyses sub-optimal. Our approach analyzes the "average-case" presampling random oracles and shows that those "worst-case" ones can never happen except with a tiny probability. To our best knowledge, this is the first work that takes advantage of "average-case" presampling and achieves tight bounds.

Overall, we consider our proofs more modular, because we utilize sequential games to focus on variants of the single-instance game (rather than directly compressing multi-instance games used by [1] and [14]). We further introduce "knowledge gaining events" to take advantage of "average-case" presampling (rather than working with worst-case ones used by [6]).

1.2 Part 2

Diffie and Hellman suggested key extension to increase security guarantees of the systems, without raising compatibility issues, as the computational capabilities of the attackers improve over time. Diffie and Hellman suggested using Cascade encryption to perform key extension for the popular block cipher constructions like DES.

Given possible keys $k_1, k_2, k_3, ..., k_t$ for a block cipher , Cascade encryption, as the name suggests, performs encryption as follows: ${}_{k_t} \circ \cdots \circ_{k_3} \circ_{k_2} \circ_{k_1}$. The smallest cascade possible is ${}_{k_2} \circ_{k_1}$ which requires 2 keys and is referred to as double encryption. However, it was quickly concluded that double encryption does not asymptotically increase security over single encryption. In order to increase security off the DES block cipher, which was famously deployed with key length too short (56 bits) to provide meaningful security against modern adversaries, without dealing with the difficulty of redeploying with replacement block ciphers, Triple-DES **X9.52**, which

runs DES three times under some keying strategy, has become the standard in some domains.

This conclusion about double encryption was based on an attack strategy called meet-in-the-middle attack, proposed by Diffie and Hellman. The attack roughly works as follows: adversary queries the double encrypting oracle on some plaintext, say X, to obtain ciphertext, say Z. Next, adversary queries on the set of all possible keys with X and $^{-1}$ on the set of all possible keys with Z. Then adversary looks for a collision between the responses from and $^{-1}$. The keys corresponding to the colliding responses are potential keys used in double encryption.

An adversary performing meet-in-the-middle attack on double encryption with 2 keys, each of size k-bits, can recover the keys with constant probability after making $2 \cdot 2^k = O(2^k)$ queries which is an order of 2^k better than a brute-force attack. For an adversary making q queries to each of and succeeds in recovering the correct keys with probability $O(q^2/2^{2k})$ which is better than brute force by a factor of $O(q)$.

However, one caveat of the meet-in-the-middle attack, that has not received much attention, is that it requires $O(2^k \cdot n)$ space to succeed with constant probability, where k is the key size and n is the block size of the underlying block cipher in bits. To mount meet-in-the-middle attack on DES, which has key size of 56 bits, over 2^{61} bits $\sim$ 250 petabytes of space is required. Not every attacker can afford to have this much memory.

Understandably, memory is a valuable and limited resource, sometimes even more than time, and thus should be accounted for in the abilities of the adversary while analyzing their advantage against any primitive. Security bounds ignoring memory as a parameter may be too pessimistic and their employment to decide optimal security parameter maybe an overkill of resources, in terms of computation cycles and bits of randomness used.

Oorschot and Wiener studied algorithms with time-memory trade-offs for attacking double encryption adaptively. They gave an attack for a more general problem of collision search which can be applied for cryptanalysis of double encryption. The collision search problem is as follows: given a function $f : [N] \to [N]$ and C, the adversary needs to find C pairs of distinct inputs that collide under f using m-bits of bounded memory and making at most q queries to f. Their attack achieves the best known trade-off of $mq^2 = \tilde{O}(C^2N)$. The algorithm can be interpreted to achieve the advantage of $mq^2/2^{3k}$ for key-recovery in double encryption, which is faster for $q > 2^k$ and currently the best known.

This trade-off is optimal for $m \approx C$ trivially. However, for $C >> m$, it is hard to prove the optimality of the bound. Proving optimality for $C >> m$ is relevant for cryptanalysis of double encryption.

Dinur in proved that the bound is optimal in the Post Filtering Model. It is worth to note that this does not prove that a better trade-off for recovering keys in Double encryption with adaptive queries is impossible. It just says that no attack that reduces the problem to collision search can do any better than the attack given by Oorschot and Wiener.

Overall, recovering keys for Double encryption with bounded time and memory using adaptive queries is a well-studied problem. The motivation for this work comes

from the lack of similar treatment given to what we refer to as the semi-adaptive setting. This work investigates the role that *adaptivity* plays in the gap between these algorithms. Both are known-plaintext attacks, but MITM is non-adaptive while the better tradeoffs use adaptive queries (akin to cycle finding techniques). We show that this is in some sense inherent: For a large class of non-adaptive attacks against double encryption (that includes MITM), we prove in the ideal-cipher model (ICM) that $mq/2^{2k}$ is the optimal advantage. This work studies the security of double encryption under this setting, and questions the need for triple encryption where the potential adversaries do not have hundreds of petabytes of data space at their disposal.

Chapter 2

Related Work

2.1 Part 1

Hellman [3] was the first to study pre-processing attacks for inverting functions, which was followed up by the famous work Fiat and Naor[17]. Recently, several works [7], [18] set out to understand the power of such attacks for finding collisions. All of them have studied this question in the auxiliary-input random oracle (AI-RO) model proposed by Unruh [4], for dealing with non-uniform and preprocessing attackers.

Dodis, Guo, and Katz [7] studied the collision resistance of a salted random function (which also corresponds to the $B = 1$ case for Merkle-Damgård). They prove an $\tilde{O}(S/N + T^2/N)$ security upper bound (with respect to a random salt). This bound shows the naive attack which precomputes collisions for S distinct salts as the advice (the $\tilde{O}(T^2/N)$ term is tight due to birthday attack) to be optimal.

Coretti, Dodis, Guo and Steinberger [6] further studied collision finding for salted Merkle-Damgård hash functions (corresponds to arbitrary B). Interestingly, unlike the $B = 1$ case, they show an attack achieving advantage $\tilde{\Omega}(ST^2/N)$, improving the birthday attack by a factor of S. They also prove this attack is optimal.

We mention that time-space lower bounds of attacks (or non-uniform security) against other fundamental cryptographic primitives, such as one-way functions, pseudorandom random generators, discrete log, have been investigated in various idealized models [6], [7], [18]–[23].

Motivated by analyzing post-quantum non-uniform security, several recent works [15], [16] studied the same question in the quantum setting, in which the adversary is given S-(qu)bit of advice and T quantum oracle queries. However, Unlike the classical setting, no matching bounds are known, even for $B = 2$ and $B = T$. The $\Omega(ST^3/N)$ security bound by Guo et al., suggests that the optimal attack may speed up the trivial quantum collision finding by a factor of S. However, the best known attack achieves $O(ST^2/N + T^3/N)$ for every $2 \leq B \leq T$.

2.2 Part 2

Relevant to the second part of this study are works that have studied streaming adversaries. Some of the most noteworthy recent works have revisited the switching lemma, results on popular symmetric encryption designs and authenticated encryption designs and many other results by considering the role of the adversary's memory capability. Memory had traditionally been considered by cryptanalysts, as a large-memory algorithm may be infeasible to implement even if its runtime is within reach. However, from the direction of proving impossibility results, memory was not considered until recently, when it was pointed out [24], [25] that computational reductions in cryptography could be made "memory tight" and sometimes yield a wider range of effective bounds. In particular, when a problem becomes harder for small-memory algorithms (such as LPN), a memory-tight reduction may be important. On the other hand, some problems (like finding a collision in a generic hash function) are solved optimally with small memory and do not benefit from memory-tightness. Subsequent work [26]–[30] has explored possibility and impossibility results on memory tightness. Further work [31], [32] extended the consideration of memory to information-theoretic steps as well.

Jaegar and Tessaro in [32] were the first to study the switching lemma with bounded memory. They realized that giving time-memory lower bounds for adversaries that can repeat queries would be difficult. This is because such an adversary can make use of Pollard-rho type memoryless algorithms to find collisions. Hence, they focused on adversaries that cannot repeat queries. The authors, however, found proving an unconditional bound out of reach. Their proof relied on a combinatorial conjecture on hypergraphs being true to give a conditional loose upper boundon the advantage. Dinur [33] succeeded in proving a bound which has a matching attack up to log factors, by giving a reduction from the Disjointness problem in the communication model to streaming and using the existing results from communication complexity. Another recent work [34] improved the result from [32].

Tessaro and Thiruvengadam [31] showed equivalence between Double Encryption and a special case of Element distinctness, which is list disjointness, and used that to give a conjectured time-space lower bound on Double Encryption. This bound was later proved unconditionally in a restricted model, namely the post filtering model, by Dinur [35]. It must be noted that our aim is to study Double Encryption for a restricted class of adversary, namely the non-adaptive and non-repeating in the streaming model. Follow-up works [26], [27], [31], [32], [36] have studied more primitives, namely authenticated encryption and symmetric encryption.

Chapter 3

Short Collisions

We study collision-finding against Merkle-Damgård hashing in the random-oracle model by adversaries with an arbitrary S-bit auxiliary advice input about the random oracle and T queries. Recent work showed that such adversaries can find collisions (with respect to a random IV) with advantage $\Omega(ST^2/2^n)$, where n is the output length, beating the birthday bound by a factor of S. These attacks were shown to be optimal.

We observe that the collisions produced are very long, on the order of T blocks, which would limit their practical relevance. We prove several results related to improving these attacks to find short collisions. We first exhibit a simple attack for finding B-block-long collisions achieving advantage $\tilde{\Omega}(STB/2^n)$. We then study if this attack is optimal. We show that the prior technique based on the bit-fixing model (used for the $ST^2/2^n$ bound) provably cannot reach this bound, and towards a general result we prove there are qualitative jumps in the optimal attacks for finding length 1, length 2, and unbounded-length collisions. Namely, the optimal attacks achieve (up to logarithmic factors) order of $(S+T)/2^n$, $ST/2^n$ and $ST^2/2^n$ advantage. We also give an upper bound on the advantage of a restricted class of short-collision finding attacks via a new analysis on the growth of trees in random functional graphs that may be of independent interest.

> *STB conjecture*: The best attack with time T and space S for finding collisions of length B in salted MD hash functions built from hash functions with n-bit outputs achieves success probability $\Theta((STB+T^2)/2^n)$.

Our results in a nutshell. We study the STB conjecture in the AI-RO model, studying both upper bounds (better attacks) and lower bounds (ruling out better attacks). Our contributions are as follows:

- *Upper bounds.* We present an attack that achieves success probability $O(STB/2^n)$. The attack exploits the existence of expanding depth-B trees of size $O(B)$ in random functional graphs defined by h.

- *Limitations of prior lower bounds.* We show that the CDGS [6] techniques cannot rule out attacks with success probability ST^2/N, even for $B = 2$. In particular, the crux of the CDGS technique is a $O(ST/N)$ bound in an intermediate idealized model (that translates to an AI-RO bound with a multiplicative loss of T), and we provide a matching attack with $B = 2$ in this intermediate model.

- *A lower bound for $B = 2$.* We present new techniques to prove the STB conjecture for $B = 2$ in the AI-RO model. That is, the optimal attack achieves success probability $\Theta((ST + T^2)/N)$ for $B = 2$. This is the main technical contribution of this work. Interestingly, this means that for $B = 2$, if the space $S \leq T$, then there is no better attack than the birthday attack!

- *Bounding low-depth trees.* We rule out the existence of expanding depth-B trees of size $\tilde{O}(B^2)$ in random functional graphs, which shows that simple extensions of our attack cannot achieve success probability better than STB/N.

3.1 Notations and Definitions

Notation. For non-negative integers N, k we write $[N]$ for $\{1, 2, \ldots, N\}$ and $\binom{[N]}{k}$ for collection of subsets of $[N]$ of size k. When X is a set, we write X^+ for tuples of 1 or more elements of X. Random variables will be written in bold, and we write $\mathbf{x} \xleftarrow{\$} X$ to indicate that $\mathbf{x}$ is a uniform random variable on X.

Merkle-Damgård (MD) hashing. We consider an abstraction of plain MD hashing, where a variable length hash function is constructed from a fixed-length compression function that is modeled as a random oracle. For integers N, M and a function $h : [N] \times [M] \to [N]$, Merkle-Damgård hashing is defined $\mathrm{MD}_h : [N] \times [M]^+ \to [N]$ recursively by $\mathrm{MD}_h(a, \alpha) = h(a, \alpha)$ for $\alpha \in [M]$, and

$$\mathrm{MD}_h(a, (\alpha_1, \ldots, \alpha_B)) = h(MD_h(a, (\alpha_1, \ldots, \alpha_{B-1})), \alpha_B)$$

for $\alpha_1, \ldots, \alpha_B \in [M]$. We refer to elements of $[M]$ as *blocks*.

3.1.1 *Collision Resistance Definitions*

We recall definitions for collision resistance against preprocessing and against bit-fixing.

Auxiliary-input security. We formalize auxiliary input security [4] for salted MD hashing as follows.

Definition 13. *For a pair of algorithms $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, a function $h : [N] \times [M] \to [N]$, and $a \in [N]$ we define game $\mathsf{AI\text{-}CR}_{h,a}(\mathcal{A})$ in Figure 3.1. We define the* auxiliary-input collision-resistance advantage of $\mathcal{A}$ against Merkle-Damgård *as*

$$\mathbf{Adv}^{\text{ai-cr}}_{\text{MD}}(\mathcal{A}) = \Pr[\mathsf{AI\text{-}CR}_{\mathbf{h},\mathbf{a}}(\mathcal{A}) = 1],$$

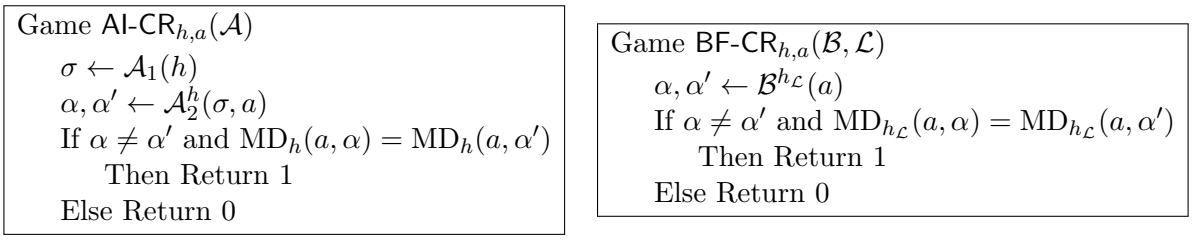


Figure 3.1: Games $\mathsf{AI\text{-}CR}_{h,a}(\mathcal{A})$ and $\mathsf{BF\text{-}CR}_{h,a}(\mathcal{B},\mathcal{L})$.

where $\mathbf{h} \xleftarrow{\$} \mathsf{Func}([N]\times[M],[N])$, $\mathbf{a} \xleftarrow{\$} [N]$ *are independent.*

We say $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ *is an* (S,T)-AI adversary *if* $\mathcal{A}_1$ *outputs* S *bits and* $\mathcal{A}_2$ *issues* T *queries to its oracle (for any inputs and oracles). We define the* (S,T)-auxiliary-input collision resistance of Merkle-Damgård, *denoted* $\mathbf{Adv}^{\text{ai-cr}}_{\text{MD}}(S,T)$, *as the maximum of* $\mathbf{Adv}^{\text{ai-cr}}_{\text{MD}}(\mathcal{A})$ *taken over all* (S,T)*-AI adversaries* $\mathcal{A}$.

We note that in our formalization, the games in the figures are not randomized, but just defined for any h and a. In the definition we use the games to define random variables by applying the game as a function to random variables $\mathbf{h}$ and $\mathbf{a}$. This has the advantage of being explicit about sample spaces when applying compression arguments.

We also consider bounded-length collisions as follows.

Definition 14. *We say a pair of algorithms* $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ *is an* (S,T,B)-AI adversary *if* $\mathcal{A}_1$ *outputs* S *bits,* $\mathcal{A}_2$ *issues* T *queries to its oracle, and the outputs of* $\mathcal{A}_2$ *each consist of* B *or fewer blocks.*

We define the (S,T,B)-auxiliary-input collision resistance of MD, *denoted* $\mathbf{Adv}^{\text{ai-cr}}_{\text{MD}}(S,T,B)$, *as the maximum of* $\mathbf{Adv}^{\text{ai-cr}}_{\text{MD}}(\mathcal{A})$ *taken over all* (S,T,B)*-AI adversaries* $\mathcal{A}$.

Bit-fixing security. We recall the bit-fixing model of Unruh [4]. When $f : X \to Y$ is a function on some domain and range, and $\mathcal{L}$ is an independent list $(x_i, y_i)_{i=1}^{|\mathcal{L}|}$ where $x_i \in X$ and $y_i \in Y$ for all $i \in 1,\ldots,|\mathcal{L}|$ and all the x_i are distinct, we define $f_{\mathcal{L}}$ as follows:

$$f_{\mathcal{L}}(x) = \begin{cases} y_i & \text{if } \exists (x_i, y_i) \in \mathcal{L} \text{ such that } x = x_i \\ f(x) & \text{otherwise.} \end{cases}$$

In other words, $\mathcal{L}$ is a list of input/output pairs, and $f_{\mathcal{L}}$ is just f, but with outputs overwritten by the tuples in $\mathcal{L}$.

Definition 15. *Let* $h : [N]\times[M] \to [N]$, *and* $a \in [N]$. *For an adversary* $\mathcal{B}$ *and a list* $\mathcal{L}$ *of input/output pairs for* h *we define* $\mathsf{BF\text{-}CR}_{h,a}(\mathcal{B},\mathcal{L})$ *in Figure 3.1. We define the* bit-fixing collision-resistance advantage of $(\mathcal{B},\mathcal{L})$ against Merkle-Damgård *as*

$$\mathbf{Adv}^{\text{bf-cr}}_{\text{MD}}(\mathcal{B},\mathcal{L}) = \Pr[\mathsf{BF\text{-}CR}_{\mathbf{h},\mathbf{a}}(\mathcal{B},\mathcal{L}) = 1],$$

where $\mathbf{h} \xleftarrow{\$} \mathsf{Func}([N]\times[M],[N])$, $\mathbf{a} \xleftarrow{\$} [N]$ *are independent.*

We say $(\mathcal{B},\mathcal{L})$ *is an* (P,T)-BF adversary *if* $\mathcal{L}$ *has at most* P *entries and* $\mathcal{B}$ *issues* T *queries to its oracle (for any inputs and oracles). We define the* (P,T)-bit-fixing collision resistance of MD, *denoted* $\mathbf{Adv}^{\text{bf-cr}}_{\text{MD}}(P,T)$, *as the maximum of* $\mathbf{Adv}^{\text{bf-cr}}_{\text{MD}}(\mathcal{B},\mathcal{L})$ *taken over all* (P,T)*-BF adversaries* $(\mathcal{B},\mathcal{L})$.

As with AI security, we also consider bounded-length collision resistance against BF adversaries.

Definition 16. *We say* $(\mathcal{B}, \mathcal{L})$ *is an* (P, T, B)-BF adversary *if* $\mathcal{L}$ *has at most* P *entries,* $\mathcal{B}$ *issues* T *queries to its oracle (for any inputs and oracles). and the outputs of* $\mathcal{B}$ *each consist of* B *or fewer blocks.*

We define the (P, T, B)-bit-fixing collision resistance of MD, *denoted* $\mathbf{Adv}^{\text{bf-cr}}_{\text{MD}}(P, T, B)$, *as the maximum of* $\mathbf{Adv}^{\text{bf-cr}}_{\text{MD}}(\mathcal{B}, \mathcal{L})$ *taken over all* (P, T, B)*-BF adversaries* $(\mathcal{B}, \mathcal{L})$.

A "CONSTRUCTIVE" CHERNOFF BOUND. We will use the following variant of the Chernoff-type bound proved by Impagliazzo and Kabanets [11]. It essentially says that if the u^{th} moments of a sum are bounded average, then we can conclude the sum is tightly concentrated, up to some dependence on u.

Theorem 2. *Let* $0 < \delta < 1$ *and let* $\mathbf{X}_1, \cdots, \mathbf{X}_N$ *be* 0/1 *random variables,* $\mathbf{X} = \mathbf{X}_1 + \cdots \mathbf{X}_N$*, and let* $\mathbf{U}$ *be an independent random subset of* $[N]$ *of size* u*. Assume*

$$\Pr[\bigwedge_{i \in \mathbf{U}} \mathbf{X}_i] \leq \delta^u.$$

Then

$$\Pr[\mathbf{X} \geq \max\{6\delta N, u\}] \leq 2^{-u}.$$

In their original version, instead of a random set $\mathbf{U}$ it was required that the first inequality hold for all sets U of size u (so all u^{th} moments must be bounded). It is easy to show that our weaker condition is still sufficient, and the proof of this version is almost identical and given in Supplementary material (Section **??**) only for completeness. As we show in Section **??**, the first inequality, in fact, does not hold true for every fixed set U of size u. Also, this weakening is crucial for our application with compression arguments.

We will also apply the following standard multiplicative Chernoff bound in Section 3.6.

Theorem 3. *Let* $0 < \delta < 1$ *and let* $\mathbf{X}_1, \cdots, \mathbf{X}_N$ *be independent* 0/1 *random variables and put* $\mathbf{X} = \mathbf{X}_1 + \cdots \mathbf{X}_N$*. Then*

$$\Pr[\mathbf{X} \geq (1 + \delta) \cdot E[\mathbf{X}]] \leq \exp\left(\frac{-\delta^2 \cdot E[\mathbf{X}]}{2 + \delta}\right).$$

3.2 Bounded-Length Auxiliary-Input Attack

Coretti et al. in [37] gave an $\Omega(ST^2)$ bound attack where adversary gets S-bit advice and T oracle queries to find (unbounded length) collisions against MD^h. It is easy to adapt the attack to B-bounded length collision finding attack for adversary with S-bit advice and T oracle queries with advantage $\Omega(STB)$. We describe this attack and its analysis.

Theorem 4. *For any positive integers S, T, B such that $B \leq T < N/4$, $STB \leq N/2$ and $M \geq N$,*

$$\mathbf{Adv}^{\text{ai-cr}}_{\text{MD}}(S,T,B) \geq \frac{STB - 96S}{48N \log N}$$

Proof. We construct an (S,T,B)-AI adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ with the required advantage for the case $M = N$. This attack easily extends to the case where $M > N$, since given $h : [N] \times [M] \to [N]$, we can run the attack on for the function $h' : [N] \times [N] \to [N]$ that coincides with h on all its inputs.

Below we write $h_0(\cdot) = h(\cdot, 0)$. The adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ works as follows:

- $\mathcal{A}_1(h)$: For $i = 1, \ldots, S/(3\lceil \log N \rceil)$, do the following:

  1. Pick $a_i \xleftarrow{\$} [N]$.
  2. Compute $y_i \leftarrow h_0^{B'}(a_i)$, where $B' = \lfloor B/2 \rfloor - 1$.
  3. If there exist $\alpha, \alpha' \in [M]$ such that $\alpha \neq \alpha'$ and $h(y_i, \alpha) = h(y_i, \alpha')$, then set $(\alpha_i, \alpha_i') = (\alpha, \alpha')$. If not, then set $(\alpha_i, \alpha_i') = (\perp, \perp)$.

  Output the triples $(y_i, \alpha_i, \alpha_i')$ for $i = 1, \ldots, S/3\lceil \log N \rceil$.

- $\mathcal{A}_2^h(\sigma, a)$: If $a = y_i$ for some i, return (α_i, α_i'). Otherwise, for $j = 1, \ldots, \lfloor T/B \rfloor$ do the following:

  1. Compute $\hat{a}_0 = h(a, j)$.
  2. For $k = 1, \ldots, B-1$:
     (a) If $\hat{a}_{k-1} = y_i$ for some i, return $j||0^{k-1}||\alpha_i$ and $j||0^{k-1}||\alpha_i'$.
     (b) If $\hat{a}_{k-1} = \hat{a}_j$ for some $j < k-1$, return $j||0^j$ and $j||0^{k-1}$.
     (c) Compute $\hat{a}_k = h_0(\hat{a}_{k-1})$.

  If no collision is found, output $\perp$ to indicate failure.

Next, we analyze the advantage of $\mathcal{A}$. This proof is very similar to the proof of Lemma 5 in [37], but we include it for completeness.

We first bound the probability that $\mathcal{A}_1$ ever fails to find a collision in step 3 (i.e. that some (α_i, α_i') is set to $(\perp, \perp)$) by

$$\frac{S}{3\log N} \frac{N!}{N^N} \leq \frac{S}{3\log N} \cdot e^{-(N-1)/2} \leq \frac{2S}{N \log N}.$$

The attack succeeds if $\hat{a} \in \mathcal{G}$ during an iteration with $k \leq B/2$, because after that there will be enough steps left to reach the end of the chain. Let R be the event that $\hat{a}$ repeats at some point during the algorithm. Let F_t be the event that $\hat{a} \in \mathcal{G}$

for the first time on the $(t+1)^{\text{th}}$ iteration of the next loop in $\mathcal{A}_2$ ($1 \le t \le T$). Then for any $A \subseteq [N]$,

$$\begin{aligned}\Pr[F_t | \neg R, \mathcal{G} = A] &= \frac{|A|}{N-t+1} \prod_{i=1}^{t} \left(1 - \frac{|A|}{N-i+1}\right) \\ &\ge \frac{|A|}{N} \left(1 - \frac{2|A|}{N}\right)^t \\ &\ge \frac{|A|}{N} \left(1 - \frac{2t|A|}{N}\right) \\ &\ge \frac{|A|}{2N}.\end{aligned}$$

The F_t are disjoint, so the probability that F_t occurs for some t within $B/2$ steps of a start point, conditioned on not-R and $\mathcal{G} = A$, is at least

$$\frac{T|A|}{4N}.$$

Putting this together we have

$$\mathbf{Adv}^{\text{ai-cr}}_{\text{MD}}(\mathcal{A}) \ge \mathbb{E}\left[\frac{T|\mathcal{G}|}{4N}\right] - \Pr[R] + \Pr[R] - \frac{2S}{N\lceil \log N \rceil}.$$

Finally we bound $\mathbb{E}[|\mathcal{G}|] \ge SB/12 \log N$ in exactly the same manner as[37]. □

3.3 Length 2 Collisions are Relatively Easy in the BF Model

Unruh in [4] proved a remarkable general relationship between the AI-RO and BF-RO models that was sharpened by Coretti et al. [37]. We recall their theorem now, and then show that this method, when applied to salted MD hashing, is insensitive to the length of collisions found and hence cannot give the improved bound we seek in the AI-RO model. We note that the second part of this theorem was not stated there, but follows exactly from their proof.

Theorem 5. *For any positive integers S, T, P and $\gamma > 0$ such that $P \ge (S + \log \gamma^{-1})T$,*

$$\mathbf{Adv}^{\text{ai-cr}}_{\text{MD}}(S, T) \le 2\mathbf{Adv}^{\text{bf-cr}}_{\text{MD}}(P, T) + \gamma.$$

Moreover, for any positive integer B,

$$\mathbf{Adv}^{\text{ai-cr}}_{\text{MD}}(S, T, B) \le 2\mathbf{Adv}^{\text{bf-cr}}_{\text{MD}}(P, T, B) + \gamma.$$

The following is a simple extension of an attack of [6], which shows that finding even just length-2 collisions in the BF model is much easier than finding length-1, and in fact as easy as finding general length collisions.

Theorem 6. *For all positive integers P, T such that $PT \leq 2N$, there exists a $(P, T, 2)$-BF adversary $(\mathcal{B}, \mathcal{L})$ such that*

$$\mathbf{Adv}^{\text{bf-cr}}_{\text{MD}}(\mathcal{B}, \mathcal{L}) = \left(1 - \frac{1}{e}\right) \frac{PT}{2N}.$$

Proof. We construct a $(P, T, 2)$-BF adversary $(\mathcal{B}, \mathcal{L})$ to prove the theorem. We assume P is even for simplicity of notation. To define the list $\mathcal{L}$, let $\alpha, \alpha' \in [M], a_1, \ldots, a_{P/2}, y \in [N]$ be some arbitrary points, and let $\mathcal{L}$ consist of the P entries

$$((a_1, \alpha), y),\ ((a_1, \alpha'), y),\ \ldots\ ((a_{P/2}, \alpha), y),\ ((a_{P/2}, \alpha'), y).$$

Adversary $\mathcal{B}$ does the following on input a and oracle $h_{\mathcal{L}}$: Let $\alpha_1, \cdots, \alpha_T \in [M]$ be arbitrary distinct points. For $j = 1, \ldots, T$, query (a, α_j) to the oracle $h_{\mathcal{L}}$. If the output is an element of $\{a_1, \ldots, a_{P/2}\}$, then output colliding messages $\alpha_j \| \alpha$ and $\alpha_j \| \alpha'$.

We lower bound $\mathbf{Adv}^{\text{bf-cr}}_{\text{MD}}(\mathcal{B}, \mathcal{L})$. Let E be the event that the output of some query made by $\mathcal{B}$ is in the set $\{a_1, \ldots, a_{P/2}\}$. Clearly, when E happens, our adversary wins the game. Thus

$$\mathbf{Adv}^{\text{bf-cr}}_{\text{MD}}(\mathcal{B}, \mathcal{L}) \geq 1 - \Pr[\neg E] \ \geq 1 - \left(1 - \frac{P}{2N}\right)^T \geq 1 - e^{-PT/2N} \geq \left(1 - \frac{1}{e}\right) \frac{PT}{2N}$$

where the last inequality holds because $0 \leq \frac{PT}{2N} \leq 1$. □

This adversary shows that applying Theorem 5 can at best give $\mathbf{Adv}^{\text{ai-cr}}_{\text{MD}}(S, T, 2) = O(ST^2/N)$, which we show to be suboptimal in Section 4.2, where using compression techniques we prove a bound of $\tilde{\text{O}}\,(ST/N)$.

3.4 Unbounded Length Collision AI Bound

In this section we give a different proof of the $O(ST^2)$ bound of Coretti et al.[37], formalized as follows.

Theorem 7. *For any positive integers S, T,*

$$\mathbf{Adv}^{\text{ai-cr}}_{\text{MD}}(S, T) \leq \frac{192e(S + \log N)T^2 + 1}{N}.$$

Following Impagliazzo [11], we proceed by analyzing an adversary without auxilliary input "locally."

Definition 17. *For a pair of algorithms $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ and positive integer u we define*

$$\mathbf{Adv}^{u\text{-ai-cr}}_{\text{MD}}(\mathcal{A}) = \Pr[\forall a \in \mathbf{U} \ :\ \mathsf{AI\text{-}CR}_{\mathbf{h},a}(\mathcal{A}) = 1],$$

where $\mathbf{h} \xleftarrow{\$} \mathsf{Func}([N] \times [M], [N])$, and $\mathbf{U} \xleftarrow{\$} \binom{[N]}{u}$ are independent.

Lemma 8. *For any positive integers S, T, u, and $\hat{\sigma} \in \{0,1\}^S$, let $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ be an adversary where $\mathcal{A}_1$ always outputs $\hat{\sigma}$, and $\mathcal{A}_2$ issues T queries to its oracle. Then*

$$\mathbf{Adv}^{u\text{-ai-cr}}_{\mathrm{MD}}(\mathcal{A}) \le \left(\frac{32euT^2}{N}\right)^u.$$

This lemma is proved below. We first prove the theorem using the lemma.

Proof of Theorem 7. Let $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ be an (S,T)-AI adversary. We need to bound $\mathbf{Adv}^{\text{ai-cr}}_{\mathrm{MD}}(\mathcal{A})$ as in the theorem. Call h *easy for adversary* $\mathcal{A}$ if

$$\Pr[\mathsf{AI\text{-}CR}_{h,\mathbf{a}}(\mathcal{A}) = 1] \ge \frac{192e(S+\log N)T^2}{N}.$$

We will first show that h is unlikely to be easy for $\mathcal{A}$, no matter how $\mathcal{A}_1$ computes its S advice bits. Below, for each $\hat{\sigma} \in \{0,1\}^S$ let $\mathcal{A}_{\hat{\sigma}}$ be $\mathcal{A}$, except with $\mathcal{A}_1$ replaced by an algorithm that always outputs $\hat{\sigma}$.

Fix some $\hat{\sigma} \in \{0,1\}^S$. For each $a \in [N]$ let $\mathbf{X}_a$ be an indicator random variable for the event that $\mathsf{AI\text{-}CR}_{\mathbf{h},a}(\mathcal{A}_{\hat{\sigma}}) = 1$. By Lemma 8 we have $\Pr[\bigwedge_{a\in\mathbf{U}} \mathbf{X}_a] \le \delta^u$ for any u and $\delta = 32euT^2/N$. Then by Theorem 2 with $u = S + \log N$, we have

$$\Pr[\mathbf{h} \text{ is easy for } \mathcal{A}_{\hat{\sigma}}] = \Pr[\sum_{a=1}^{N} \mathbf{X}_a \ge 192euT^2] \le 2^{-(S+\log N)}.$$

Let E be the event that there exists $\hat{\sigma} \in \{0,1\}^S$ such that $\mathbf{h}$ is easy for $\mathcal{A}_{\hat{\sigma}}$. By a union bound over $\hat{\sigma} \in \{0,1\}^S$,

$$\Pr[E] \le 2^S 2^{-(S+\log N)} = 1/N.$$

Finally we have

$$\begin{aligned}\Pr[\mathsf{AI\text{-}CR}_{\mathbf{h},\mathbf{a}}(\mathcal{A}) = 1] &\le \Pr[\mathsf{AI\text{-}CR}_{\mathbf{h},\mathbf{a}}(\mathcal{A}) = 1 | \neg E] + \Pr[E] \\ &\le \frac{192e(S+\log N)T^2}{N} + \frac{1}{N}.\end{aligned}$$

The last inequality holds because for each $h \notin E$,

$$\Pr[\mathsf{AI\text{-}CR}_{h,\mathbf{a}}(\mathcal{A}) = 1] \le \max_{\hat{\sigma}\in\{0,1\}^S} \Pr[\mathsf{AI\text{-}CR}_{h,\mathbf{a}}(\mathcal{A}_{\hat{\sigma}}) = 1] \le \frac{192e(S+\log N)T^2}{N}.$$

Since this holds for any (S,T)-AI adversary $\mathcal{A}$, we obtain the lemma. □

3.4.1 *Proof of Lemma 8*

Colliding chains. We start with some useful definitions and a simplifying lemma.

Definition 18. *Let $h : [N]\times[M] \to [N]$. A list of elements $(x_0, \alpha_0), \ldots, (x_\ell, \alpha_\ell)$ from $[N] \times [M]$ is called an* MD-chain (for h) *when $h(x_i, \alpha_i) = x_{i+1}$ for $i = 0, \ldots, \ell - 1$. We say two chains $(x_0, \alpha_0), \ldots (x_\ell, \alpha_\ell)$ and $(x'_0, \alpha'_0), \ldots (x'_{\ell'}, \alpha'_{\ell'})$* collide (for h) *if $h(x_\ell, \alpha_\ell) = h(x'_{\ell'}, \alpha'_{\ell'})$.*

We will treat chains as strings (over $[N] \times [M]$) and speak of prefixes and suffixes with their usual meaning.

Lemma 9. *Let C, C' be distinct non-empty colliding chains for h. Then C, C' contain prefixes $\tilde{C}, \tilde{C}'$ respectively, $\tilde{C} = (x_0, \alpha_0), \ldots (x_\ell, \alpha_\ell)$ and $\tilde{C}' = (x'_0, \alpha'_0), \ldots (x'_{\ell'}, \alpha'_{\ell'})$ where either $(x_\ell, \alpha_\ell) \neq (x'_{\ell'}, \alpha'_{\ell'})$ or one of $\tilde{C}, \tilde{C}'$ is a strict suffix of the other.*

Proof. Induction on the maximum length of C, C'. For length 1 this is obvious. For the inductive step, suppose neither is a strict suffix of the other, and that the final entries are equal. Then we can remove the final entries from both chains to get two non-empty shorter chains and apply the inductive hypothesis. □

Proof of Lemma 8. We prove the lemma by compression. Let $\mathbf{U} \xleftarrow{\$} \binom{[N]}{u}$ and $\mathbf{h} \xleftarrow{\$} \mathsf{Func}([N] \times [M], [N])$ be independent. Let $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ be (S, T)-AI adversary where $\mathcal{A}_1$ always outputs some fixed string $\hat{\sigma}$. Observe that if, for all $a \in \mathbf{U}$, $\mathcal{A}_2$ never queries a point of $\mathbf{h}$ with input salt $a' \in \mathbf{U}, a \neq a'$ among the T queries it makes for input a, and also that the chains produced for each a are disjoint from the chains for a', then it is relatively simple to carry out a compression argument, by storing two pointers $[T]$ to compress an entry in $[N]$ (which would translate to a bound $(T^2/N)^u$). But of course $\mathcal{A}$ might "cross up" queries for the different salts. If we tried to prove a version of the lemma for *all* $U \subset \binom{[N]}{u}$ (as was done in the original context of the appendix in [12]) rather than a random $\mathbf{U}$, then the adversary could be specialized for the set U; In section **??** we give an attack that finds collisions of $B = 2$ for a fixed subset with greater probability than the upper bound on the advantage of attacking a random subset of same size.

Also, when we choose $\mathbf{U}$ at random, a fixed $\mathcal{A}$ can't be specialized for the set, and the "crossed up" queries between salts are unlikely. Formally, if $\mathcal{A}$ queries a salt $a' \in \mathbf{U}$ while attacking $a \in \mathbf{U}$, we take advantage of this by *compressing an entry of the random set* $\mathbf{U}$ when a crossed-up query occurs. Very roughly, this requires a pointer in $[T]$ and saves a factor N/S (because we are omitting one entry from an (unordered) set of size about S). The net compression is then about ST/N per crossed-up query. The details are a bit more complicated, as this compression actually experiences a smooth trade-off as more such queries are compressed and the set shrinks. We handle the case when the chains for $a \neq a'$ intersect via a simpler strategy that also results in ST/N factors.

Once the crossed-up queries are handled, the proof effectively reduces to the simpler case without crossed-up queries.

Proof. For the rest of the proof we fix some $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, where $\mathcal{A}_1$ always outputs some fixed $\hat{\sigma}$. Let

$$\mathcal{G} = \{(U, h) \mid \forall a \in U : \mathsf{AI\text{-}CR}_{h,a}(\mathcal{A}) = 1\} \subseteq \binom{[N]}{u} \times \mathcal{H}.$$

Let

$$\varepsilon = \mathbf{Adv}^{u\text{-ai-cr}}_{\mathrm{MD}}(\mathcal{A}).$$

So $|\mathcal{G}| = \varepsilon\binom{N}{u}N^{MN}$. We define an injection

$$f : \mathcal{G} \to \{0,1\}^L$$

with L satisfying

$$\frac{2^L}{\binom{N}{u}N^{MN}} \le \left(\frac{32euT^2}{N}\right)^u.$$

Pigeonhole then immediately gives the bound on ε.

For $(U,h) \in \mathcal{G}$, $f(U,h)$ outputs an L-bit encoding, where L will be determined below, of

$$(F, U_{\mathsf{Fresh}}, \mathsf{Pred}, \mathsf{Cases}, \mathsf{Coll}, \tilde{h}),$$

where the first output F is an integer between 1 and u, U_{Fresh} is a subset of U of size F, Pred is a set of pointers, Cases is a list of elements in $\{\mathsf{1a}, \mathsf{1b}, \mathsf{2}\}$, and Coll is a list of pairs of pointers, and the last output $\tilde{h}$ is h but rearranged and with some entries deleted. We now define these outputs in order.

Fresh salts and prediction queries. Fix some $(U,h) \in \mathcal{G}$, and let $U = \{a_1, \ldots, a_u\}$, where the a_i are in lexicographic order. Let $\mathsf{Qrs}(a_i) \in ([N] \times [M])^T$ denote the queries $\mathcal{A}_2$ makes to its oracle when run on input $(\hat{\sigma}, a_i)$. Let us abuse notation by writing $a_i \notin \mathsf{Qrs}(a_j)$ to mean that a_i is not the first component of any entry in $\mathsf{Qrs}(a_j)$ (in other words, the salt a_i is not queried when $\mathcal{A}_2$ runs on a_j).

We define $U_{\mathsf{Fresh}} \subseteq U$ inductively by

$$a_i \in U_{\mathsf{Fresh}} \iff \forall 1 \le j < i : a_j \in U_{\mathsf{Fresh}} \implies a_i \notin \mathsf{Qrs}(a_j).$$

The set U_{Fresh} trivially contains a_1. For $i > 1$, $a_i \in U_{\mathsf{Fresh}}$ if no prior salt in U_{Fresh} causes $\mathcal{A}_2$ to query a_i. Conversely, for any $a_i \in U \setminus U_{\mathsf{Fresh}}$, there is a prior salt $a_j \in U_{\mathsf{Fresh}}$ such that $a_i \in \mathsf{Qrs}(a_j)$.

Let us denote the size of U_{Fresh} by F, and let us write $U_{\mathsf{Fresh}} = \{a'_1, \ldots, a'_F\}$ where the a'_j are in lexicographic order. From now on we will only need to deal with the queries issued when the adversary is run on fresh salts. Let $\mathsf{Q}_j = \mathsf{Qrs}(a'_j)$ for $j = 1, \ldots, F$ and

$$\mathsf{Q}_{\mathsf{Fresh}} = \mathsf{Q}_1 \| \cdots \| \mathsf{Q}_F \in ([N] \times [M])^{FT}.$$

Going forward, we will sometimes use indices from $[FT]$ to point to queries in $\mathsf{Q}_{\mathsf{Fresh}}$ and sometimes indices from $[T]$ to point to queries in Q_j for some j.

For each $a \in U \setminus U_{\mathsf{Fresh}}$, there exists a minimum $t_a \in [FT]$ such that $\mathsf{Q}_{\mathsf{Fresh}}[t_a]$ is a query with input salt a. We define $\mathsf{Pred} \subseteq [FT]$, the *prediction queries*, to be

$$\mathsf{Pred} = \{t_a \mid a \in U \setminus U_{\mathsf{Fresh}}\} \subseteq [FT].$$

We have $|\mathsf{Pred}| = u - F$.

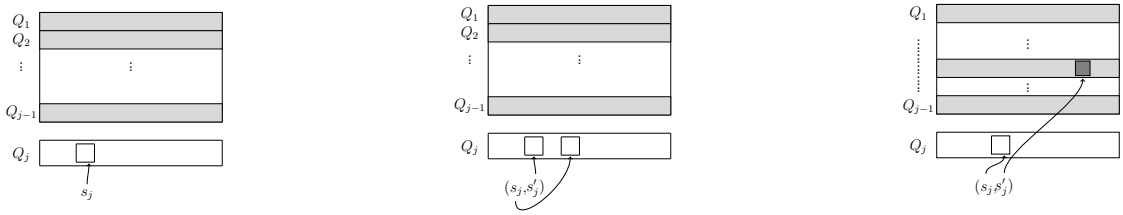
New and old queries. Call an index $r \in [FT]$ *new* if $\mathsf{Q}_{\mathsf{Fresh}}[r]$ does not appear earlier in $\mathsf{Q}_{\mathsf{Fresh}}$ (more precisely, if $s < r$ implies $\mathsf{Q}_{\mathsf{Fresh}}[s] \neq \mathsf{Q}_{\mathsf{Fresh}}[r]$). For $j \in [F]$ we

will speak of an index $t \in [T]$ being new in Q_j, technically meaning that $(j-1)T+t \in [FT]$ is new. Since we assume that the queries in Q_j are distinct, $\mathsf{Q}_j[t]$ being new is equivalent to $\mathsf{Q}_j[t]$ not appearing in $\mathsf{Q}_1 \| \cdots \| \mathsf{Q}_{j-1}$. When a query is not new, we say it is *old*.

Claim 10. *Let $\mathsf{Q}_{\mathsf{Fresh}} = \mathsf{Q}_1 \| \cdots \| \mathsf{Q}_F \in ([N] \times [M])^{FT}$ be defined as above. Then for each j, at least one of the following cases holds:*

1. *(a) There exists $s_j \in [T]$ such that s_j is new in Q_j and $h(\mathsf{Q}_j[s_j]) = a'_j$,*
   (b) There exists $s'_j < s_j \in [T]$ such that s_j and s'_j are new in Q_j and $h(\mathsf{Q}_j[s_j]) = h(\mathsf{Q}_j[s'_j])$
2. *There exists $s_j \in [T]$ and $s'_j \in [FT]$ such that s_j is new in Q_j, s'_j points to query in $\mathsf{Q}_1 || \ldots || \mathsf{Q}_{j-1}$, and $h(\mathsf{Q}_j[s_j])$ equals the input salt of $\mathsf{Q}_{\mathsf{Fresh}}[s'_j]$.*

These cases are depicted in Figure 3.2.



(i) Case 1(a) (ii) Case 1(b) (iii) Case 2

Figure 3.2: Cases from claim 10. Box denotes an index. White box denotes query at the index is new.

Proof of claim. For each j, Q_j contains a pair of colliding chains that both start from a'_j. Either some query in the chains is old, or else all the queries in both chains are new.

Suppose first that some query is old, and focus on the chain containing the old query. Since a'_j is fresh, this old query cannot be the first query of the chain. Thus, starting from the beginning of this chain, we eventually reach a query that is new but the next query is old. Because these queries form a chain, this new query will output the old query's input salt. So we take s_j to point to this new query in Q_j, and s'_j to point to the earlier query in $\mathsf{Q}_1 \| \cdots \| \mathsf{Q}_{j-1}$, and Case 2 of the claim holds.

Now suppose that all queries in the chains are new. By Lemma 9, we can assume without loss of generality that either the last queries of the chains are distinct, or that one chain is a strict suffix of the other. If the chains have distinct final queries, then we can take s_j, s'_j to point to these distinct (new) queries in Q_j and Case 1b of the claim holds. If one chain is a strict suffix of the other, then the longer chain must contain a (new) query that outputs a'_j, since both chains start with salt a'_j. Then Case 1a of the claim holds. □

Definition of f. On input (U, h), $f(U, h)$ first computes U_{Fresh} and Pred as defined above. It then computes $\mathsf{Q}_{\mathsf{Fresh}}$, and Q_j for each $j = 1, \ldots, F$. It initializes: (1) Array

Cases and Coll, each of size F, the latter of which will hold entries from domains depending on cases, (2) $\tilde{h}$ to be the table of h, but sorted to contain the responses for $\mathsf{Q}_{\mathsf{Fresh}}$, followed by the rest of the table in lexicographic order.

For $j = 1, \ldots, F$, f examines Q_j and determines which of the cases in the claim occurs. It sets $\mathsf{Cases}[j] \in \{\mathsf{1a}, \mathsf{1b}, \mathsf{2}\}$ and performs one of the following:

Case 1a: Set $\mathsf{Coll}[j] \leftarrow s_j \in [T]$ and delete the entry corresponding to $\mathsf{Q}_j[s_j]$ from $\tilde{h}$.

Case 1b: Set $\mathsf{Coll}[j] \leftarrow (s_j, s'_j) \in [T] \times [T]$ and delete the entry corresponding to query $\mathsf{Q}_j[s_j]$ from $\tilde{h}$.

Case 2: Set $\mathsf{Coll}[j] \leftarrow (s_j, s'_j) \in [T] \times [FT]$ and delete the entry corresponding to query $\mathsf{Q}_j[s_j]$ from $\tilde{h}$.

This completes the description of f. After this process, $\tilde{h}$ consists of the responses to the queries of $\mathcal{A}_2$ when it is run on the salts in U_{Fresh}, except for the deleted queries, followed by the remaining outputs of h in lexicographic order. Since at least one case always holds by the claim, and we delete exactly one new query from each Q_j, we have $\tilde{h} \in [N]^{MN-F}$.

Analysis of f. We first argue that the output length of f is not too long, and later that is it injective. Let the number of salts in U_{Fresh} having compression type 1a, 1b and 2 be δ_1, δ'_1 and δ_2, respectively. Then $F = \delta_1 + \delta'_1 + \delta_2$. We set the output length L to the maximum of the following expression, over $1 \le F \le u$, and $\delta_1 + \delta'_1 + \delta_2 = F$, rounded to the next integer:

$$\log \left(u \cdot 2^{2F} \binom{N}{F} \binom{FT}{u-F} T^{\delta_1} T^{2\delta'_1} (FT^2)^{\delta_2} N^{MN-F} \right).$$

This formula is explained by considering the outputs of f in turn:

- F and U_{Fresh} account for $\log u + \log \binom{N}{F}$ bits together,
- Pred needs $\log \binom{FT}{u-F}$ bits,
- Cases is an array of size F, storing a ternary value in each entry, and thus less than $2F$ bits total,
- Coll stores $\delta_1 \log T + \delta'_1 \log T^2 + \delta_2 \log FT^2$ bits,
- $\tilde{h}$ stores $(MN - F) \log N$ bits.

We have

$$\frac{2^L}{\binom{N}{u} N^{MN}} \le \max_{F=\delta_1+\delta'_1+\delta_2} u \cdot 2^{2F} \cdot \frac{\binom{N}{F}\binom{FT}{u-F}}{\binom{N}{u}} \cdot \left(\frac{T^{\delta_1} \cdot (T^2)^{\delta'_1} \cdot (FT^2)^{\delta_2}}{N^F} \right).$$

We bound the middle term by

$$\frac{\binom{N}{F}\binom{FT}{u-F}}{\binom{N}{u}} \le \left(\frac{eN}{F}\right)^F \left(\frac{u}{N}\right)^u \left(\frac{eFT}{u-F}\right)^{u-F} = \left(\frac{eu}{F}\right)^F \left(\frac{euFT}{N(u-F)}\right)^{u-F}$$
$$\le \left(e2^{u/F}\right)^F \left(\frac{e2^{u/(u-F)}FT}{N}\right)^{u-F} \le (4e)^u \left(\frac{FT}{N}\right)^{u-F}.$$

Assuming f is injective, by pigeonhole we have $2^L \ge \varepsilon\binom{N}{u}N^{MN}$. Plugging this in and using $F \le u \le 2^u$, we obtain

$$\varepsilon \le \max_{F=\delta_1+\delta_1'+\delta_2} (32e)^u \left(\frac{FT}{N}\right)^{u-F} \left(\frac{T}{N}\right)^{\delta_1} \left(\frac{T^2}{N}\right)^{\delta_1'} \left(\frac{FT^2}{N}\right)^{\delta_2} \le \left(\frac{32euT^2}{N}\right)^u.$$

It remains to show that f is injective, i.e. that (U, h) is determined by $f(U, h) = (F, U_{\mathsf{Fresh}}, \mathsf{Pred}, \mathsf{Cases}, \mathsf{Coll}, \tilde{h})$. The inversion algorithm works as follows:

1. Decode the binary input by reading off F from the first $\log u$ bits. The size of U_{Fresh}, Pred, and Cases are determiend by F. Then Coll can be parsed out using Cases, and finally $\tilde{h}$ can be parsed out easily.

2. Initialize h and $\mathsf{Q}_{\mathsf{Fresh}}$ to be empty tables.

3. For each $a'_j \in U_{\mathsf{Fresh}}$ (in lexicographic order) the j-th entry of Cases indicates the type of tuple of pointers stored in the j-th entry of Coll. Run $\mathcal{A}_2$ on a'_j. Respond to queries using the entries of $\tilde{h}$ in order (except if the query is a repeat) and populating the entires of h and $\mathsf{Q}_{\mathsf{Fresh}}$ until except when one of the following happens:

   (a) For $\mathsf{Cases}[j] = 1a$, when $\mathsf{Q}_j[s_j]$ is queried, respond to the query with a'_j.

   (b) For $\mathsf{Cases}[j] = 1b$, when $\mathsf{Q}_j[s_j]$ is queried, respond with $h(\mathsf{Q}_j[s'_j])$ (using the partial table for h).

   (c) For $\mathsf{Cases}[j] = 2$, when $\mathsf{Q}_j[s_j]$ is queried, respond with the input salt of query $\mathsf{Q}_{\mathsf{Fresh}}[s'_j]$.

   After responding, continue running $\mathcal{A}_2$ on a'_j and populating the tables.

4. After running $\mathcal{A}_2$ on all of the salts in U_{Fresh}, populate the rest of h using the remaining entries of $\tilde{h}$ in order.

5. Finally, examine the queries $\mathsf{Q}_{\mathsf{Fresh}}$, and form U by adding the salts pointed to by the indices of Pred to U_{Fresh}. (More formally, output $U = U_{\mathsf{Fresh}} \cup \{\text{input salt of } \mathsf{Q}_{\mathsf{Fresh}}[t] : t \in \mathsf{Pred}\}$.)

We first argue inversion replies to the queries issued by $\mathcal{A}_2$ on a'_j correctly, for each $a'_j \in U_{\mathsf{Fresh}}$. For queries that are not deleted from $\tilde{h}$, these are simply copied from $\tilde{h}$. By construction and Claim 10, the queries that were deleted will be copied correctly. Finally, once $\mathsf{Q}_{\mathsf{Fresh}}$ is correctly computed, we have that U is correctly recovered. □

3.5 Length 2 Collision AI Bound

We next prove an upper bound on the advantage of an adversary producing collisions of length at most 2.

Theorem 11. *For any positive integers S, T,*

$$\mathbf{Adv}^{\text{ai-cr}}_{\text{MD}}(S,T,2) \le 6\cdot(2^9e^3)\max\left\{\left(\frac{(S+\log N)T}{N}\right),\left(\frac{T^2}{N}\right)\right\}+\frac{1}{N}.$$

We prove this theorem in exactly the same fashion to Theorem 7, where an adversary is analyzed without auxiliary input "locally," as in the lemma below. We present it again for the sake of completeness.

Lemma 12. *For any positive integers S, T, u, and $\hat{\sigma}\in\{0,1\}^S$, let $\mathcal{A}=(\mathcal{A}_1,\mathcal{A}_2)$ be an adversary where $\mathcal{A}_1$ always outputs $\hat{\sigma}$, and $\mathcal{A}_2$ issues T queries to its oracle and always outputs collision of length at most 2. Then*

$$\mathbf{Adv}^{u\text{-ai-cr}}_{\text{MD}}(\mathcal{A}) \le (2^9e^3)^u\max\left\{\left(\frac{uT}{N}\right)^u,\left(\frac{T^2}{N}\right)^u\right\}.$$

Proof for theorem 11. Let $\mathcal{A}=(\mathcal{A}_1,\mathcal{A}_2)$ be an $(S,T,2)$-AI adversary. We need to bound $\mathbf{Adv}^{\text{ai-cr}}_{\text{MD}}(\mathcal{A})$ as in the theorem. Call h *easy for adversary* $\mathcal{A}$ if

$$\Pr[\mathsf{AI\text{-}CR}_{h,\mathbf{a}}(\mathcal{A})=1] \ge \frac{6(2^9e^3)\max\{(S+\log N)T,T^2\}}{N}.$$

We will first show that h is unlikely to be easy for $\mathcal{A}$, no matter how $\mathcal{A}_1$ computes its S advice bits. Below, for each $\hat{\sigma}\in\{0,1\}^S$ let $\mathcal{A}_{\hat{\sigma}}$ be $\mathcal{A}$, except with $\mathcal{A}_1$ replaced by an algorithm that always outputs $\hat{\sigma}$.

Fix some $\hat{\sigma}\in\{0,1\}^S$. For each $a\in[N]$ let $\mathbf{X}_a$ be an indicator random variable for the event that $\mathsf{AI\text{-}CR}_{\mathbf{h},a}(\mathcal{A}_{\hat{\sigma}})=1$. By Lemma 12 we have $\Pr[\bigwedge_{a\in\mathbf{U}}\mathbf{X}_a]\le\delta^u$ for any u and $\delta=(2^9e^3)\max\left\{\left(\frac{uT}{N}\right),\left(\frac{T^2}{N}\right)\right\}$. Then by Lemma 2 with $u=S+\log N$, we have

$$\Pr[\mathbf{h}\text{ is easy for }\mathcal{A}_{\hat{\sigma}}]=\Pr[\sum_{a=1}^{N}\mathbf{X}_a\ge 6(2^9e^3)\max\{(S+\log N)T,T^2\}]\le 2^{-(S+\log N)}.$$

Let E be the event that there exists $\hat{\sigma}\in\{0,1\}^S$ such that $\mathbf{h}$ is easy for $\mathcal{A}_{\hat{\sigma}}$. By a union bound over $\hat{\sigma}\in\{0,1\}^S$,

$$\Pr[E]\le 2^S2^{-(S+\log N)}=1/N.$$

Finally we have

$$\begin{aligned}\Pr[\mathsf{AI\text{-}CR}_{\mathbf{h},\mathbf{a}}(\mathcal{A})=1] &\le \Pr[\mathsf{AI\text{-}CR}_{\mathbf{h},\mathbf{a}}(\mathcal{A})=1|\neg E]+\Pr[E]\\ &\le \frac{6\cdot(2^9e^3)\max\{(S+\log N)T,T^2\}}{N}+\frac{1}{N}.\end{aligned}$$

The last inequality holds because for each $h \notin E$,

$$\begin{aligned}\Pr[\mathsf{AI\text{-}CR}_{h,\mathbf{a}}(\mathcal{A}) = 1] &\le \max_{\hat{\sigma}\in\{0,1\}^S} \Pr[\mathsf{AI\text{-}CR}_{h,\mathbf{a}}(\mathcal{A}_{\hat{\sigma}}) = 1] \\ &\le \frac{6 \cdot (2^9 e^3) \max\{(S + \log N)T, T^2\}}{N}.\end{aligned}$$

Since this holds for any $(S, T, 2)$-AI adversary $\mathcal{A}$, we obtain the lemma. □

3.5.1 *Proof for Lemma 12*

Intuition. At a high level this proof is similar to that of Lemma 8. The primary difference is that we must avoid the ST^2 factors that come from chains hitting old edges. This turns out to be quite subtle, as the adversary may have generated some structures involving collisions in early queries and later hit them. But if we try to compress these structures preemptively, we find they are not profitable (i.e. the required pointers are bigger than the savings). In our proof, however, this strategy is actually a gambit: We make some losing moves up front, and then later are able to compress multiple edges and eventually profit. Looking forward, this happens for either version of Case 4 in Figure 3.4, where the early edges are blue. There it is not profitable to compress the second blue edge on its own, but later get a super-profit by compressing one or two black edges, resulting in a net compression.

We now proceed with the formal compression proof. Let $\mathbf{U} \xleftarrow{\$} \binom{[N]}{u}$ and $\mathbf{h} \xleftarrow{\$} \mathsf{Func}([N] \times [M], [N])$ be independent. Let $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ be an adversary as specified in the lemma. Let

$$\mathcal{G} = \{(U, h) \mid \forall a \in U : \mathsf{AI\text{-}CR}_{h,a}(\mathcal{A}) = 1\} \subseteq \binom{[N]}{u} \times \mathcal{H}.$$

and $\varepsilon = \mathbf{Adv}^{u\text{-ai-cr}}_{\mathrm{MD}}(\mathcal{A})$, so $|\mathcal{G}| = \varepsilon\binom{N}{u}N^{MN}$. We define an injection

$$f : \mathcal{G} \to \{0,1\}^L$$

with L satisfying

$$\frac{|\{0,1\}^L|}{\binom{N}{u}N^{MN}} \le (2^9 e^3)^u \left(\frac{\max\{T^2, uT\}}{N}\right)^u.$$

Pigeonhole then immediately gives the bound on ε.

For $(U, h) \in \mathcal{G}$, $f(U, h)$ outputs an L-bit encoding, where L will be determined below, of

$$(F, U_{\mathsf{Fresh}}, \mathsf{Pred}, \mathsf{Cases}, \mathsf{Coll}, \mathsf{Loops}, \mathsf{Bulbs}, \mathsf{Diamonds}, \tilde{h}),$$

which we define below. The first, second and third outputs F, U_{Fresh} and Pred are computed *exactly* as in the proof of Lemma 8, and in particular $U_{\mathsf{Fresh}} \subseteq U$ and $\mathsf{Pred} \subseteq [FT]$, $|\mathsf{Pred}| = u - F$, where $|U_{\mathsf{Fresh}}| = F$.

In order to describe the remaining outputs of f, we need some definitions. Write $U_{\mathsf{Fresh}} = \{a'_1, \cdots, a'_F\}$ and let $\mathsf{Q}_{\mathsf{Fresh}}$ and Q_j be defined exactly as in the proof of Lemma 8. Without loss of generality, for each $j \in [F]$, we assume that an adversary

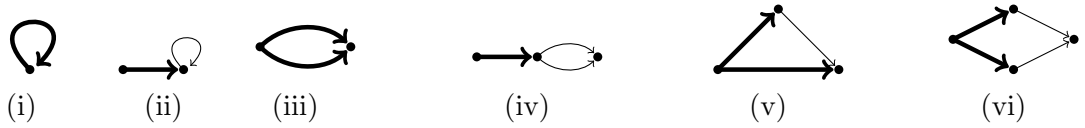


Figure 3.3: Types of queries *used* to obtain colliding chains for $B = 2$. Bold arrows indicate that the query is necessarily new.

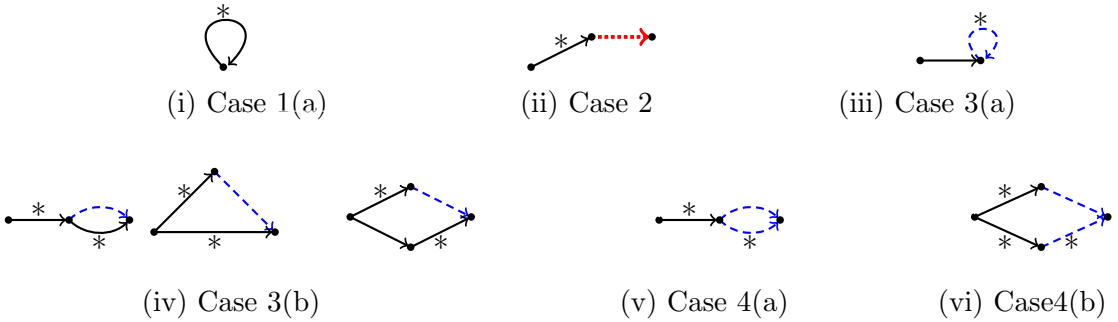


Figure 3.4: Cases in Claim 13. Red dotted arrow represents a *reused* old query. Blue dashed arrow represents an *unused-old* query. '*' marks the queries that will be compressed by f.

makes distinct queries in Q_j. In other words, for each fresh salt a'_j, $\mathsf{Qrs}(a'_j)$ contains distinct queries.

USED QUERIES. For $j = 1, \ldots, F$ Q_j contains a pair of colliding chains of length at most 2, and that when the adversary outputs its collision, the chains are formed from amongst at most 4 entries of Q_j. Note that in some cases one entry of Q_j could appear multiple times in the colliding chains (for instance, if there is a self loop, or both chains start with the same edge). By examining the colliding chains, we can always find a subset of queries corresponding to one of the case in Figure 3.3. This subset will be strict sometimes (for instance, in case (iii), the adversary may have opted to add the same query to the end of both chains, or even to add another pair of colliding queries; In these cases we only consider the queries shown in the diagram). We define *used* queries in Q_j, denoted Used_j, to be the subset of $[T]$ that are indices of the queries corresponding to the appropriate case in the Figure. We have that $1 \leq |\mathsf{Used}_j| \leq 4$.

NEW, REUSED, AND UNUSED-OLD QUERIES. For any $j \in [F]$, a query in Q_j is said to be *new* if it does not appear in $\mathsf{Q}_1||\cdots||\mathsf{Q}_{j-1}$. If query is not new, it is said to be *old*. If a query in Q_j is old, and that query appears amongst the queries pointed to by Used_k for some $k < j$ (i.e. the query equals $\mathsf{Q}_k[s]$ for some $s \in \mathsf{Used}_k$), then we say the query is *reused* and otherwise it is *unused-old*.

Claim 13. *Let* $\mathsf{Q}_{\mathsf{Fresh}}$, Q_j, *and* Used_j *be* $j = 1, \ldots, F$ *be defined as above. Then for each* j, *at least one of the following cases (which are depicted in Figure 3.4) holds:*

1. *[*Used_j ***contains only new queries.]***

(a) *There exists* $s_j \in [T]$ *such that* $\mathsf{Q}_j[s_j]$ *is new and* $h(\mathsf{Q}_j[s_j]) = a'_j$,

(b) *There exists* $s_j^1 \neq s_j^2 \in [T]$ *such that* $\mathsf{Q}_j[s_j^1]$ *and* $\mathsf{Q}_j[s_j^2]$ *are new, and* $h(\mathsf{Q}_j[s_j^1]) = h(\mathsf{Q}_j[s_j^2])$

2. *[*Used_j ***contains at least one reused query.]*** *There exists* $s_j \in [T]$ *and* $t_j \in [F]$ *such that* $\mathsf{Q}_j[s_j]$ *is new and* $h(\mathsf{Q}_j[s_j])$ *equals input salt of some query in* Q_{t_j} *pointed to by* Used_{t_j}.

3. *[*Used_j ***contains exactly*** 1 ***unused old query.]***

(a) *There exists* $s_j \in [T]$ *and* $u_j \in [FT]$ *such that* $\mathsf{Q}_j[s_j]$ *is new,* u_j *is new (in its respective* Q_k, $k < j$*) and* $h(\mathsf{Q}_j[s_j])$ *and* $h(\mathsf{Q}_{\mathsf{Fresh}}[u_j])$ *both equal input salt of query* $\mathsf{Q}_{\mathsf{Fresh}}[u_j]$,

(b) *There exists* $s_j^1 \neq s_j^2 \in [T]$ *and* $u_j \in [FT]$ *such that* $\mathsf{Q}_j[s_j^1], \mathsf{Q}_j[s_j^2]$ *are new, and* $h(\mathsf{Q}_j[s_j^1])$ *equals input salt of* $\mathsf{Q}_{\mathsf{Fresh}}[u_j]$*, and* $h(\mathsf{Q}_j[s_j^2]) = h(\mathsf{Q}_{\mathsf{Fresh}}[u_j])$.

4. *[*Used_j ***contains exactly*** 2 ***unused old queries.]***

(a) *There exists* $s_j \in [T]$ *and* $u_j^1 < u_j^2 \in [FT]$ *such that* $\mathsf{Q}_j[s_j]$ *is new,* u_j^2 *is new (in its respective* $\mathsf{Q}_k, k < j$*)*[1]*, the input salt of queries, the input salt of queries* $\mathsf{Q}_{\mathsf{Fresh}}[u_j^1]$ *and* $\mathsf{Q}_{\mathsf{Fresh}}[u_j^2]$ *are equal,* $h(\mathsf{Q}_j[s_j])$ *equals their common input salt, and* $h(\mathsf{Q}_{\mathsf{Fresh}}[u_j^2]) = h(\mathsf{Q}_{\mathsf{Fresh}}[u_j^1])$,

(b) *There exists* $s_j^1 \neq s_j^1 \in [T]$, $u_j^1 < u_j^2 \in [FT]$ *such that* $\mathsf{Q}_j[s_j^1], \mathsf{Q}_j[s_j^2]$ *are new,* u_j^2 *is new (in its respective* $\mathsf{Q}_k, k < j$*),* $h(\mathsf{Q}_j[s_j^1])$ *equals the input salt of query* $\mathsf{Q}_{\mathsf{Fresh}}[u_j^1]$*,* $h(\mathsf{Q}_j[s_j^2])$ *equals the input salt of query* $\mathsf{Q}_{\mathsf{Fresh}}[u_j^2]$*, and* $h(\mathsf{Q}_{\mathsf{Fresh}}[u_j^2]) = h(\mathsf{Q}_{\mathsf{Fresh}}[u_j^1])$.

Proof of claim. Fix $j \in [F]$ and consider Used_j. The queries in Q_j corresponding to Used_j fall into one of the cases in Figure 3.3. Since a'_j is fresh, the queries with input salt a'_j must be new (otherwise a'_j would be predicted). Thus the bold edges in the figure must be new queries. The other queries can be *new*, *reused* or *unused-old*.

Suppose all of the queries of Q_j in Used_j are new. For cases (ii)–(vi) in Fig. 3.3 there are 2 distinct queries that have the same output, so we take s_j^1, s_j^2 to point to these queries in Q_j and case 1(b) holds. In the remaining case (i), the output of the query is a'_j, so we take s_j to point at the relevant query in Q_j and case 1(a) of the claim holds.

From now on we assume that not all queries are new. Next, suppose an edge in Used_j is reused, say appearing in the used queries for Q_k, $k < j$. Since a'_j is fresh, the reused query cannot be the first query in the chain, so it is the second edge of one of the chains. As these queries form a chain, the output of the new query will be the input of the reused query. So we take $t_j = k$ (i.e. to point to the k^{th}-salt a'_k in U_{Fresh} from which the query is being reused), and $s_j \in [T]$ to point to the new query in Q_j that outputs the input salt of some query in Used_k, and thus case 2 of the claim holds.

[1]Query at u_j^1 is not compressed, so it does not matter whether it is new or not.

We have dealt with the case where the old query is a reused query. What remains is if the old query is unused-old. There are either 1 or 2 unused-old queries and we handle these separately.

Suppose there exists exactly one query in the colliding chains that is unused-old. Again this query has to be the last edge of the chain. Since the chain has length 2, we are in case (ii) or (iv)-(vi) of Figure 3.3. In case (ii), we have that case 3(a) of the claim holds as we can take $u_j \in [FT]$ to point to the loop. Otherwise we can find queries pointed to by s_j, s'_j in Q_j and $u_j \in [FT]$ such that case 3(b) holds.

Finally suppose are exactly two unused-old queries in the colliding chains. Then we must be in case (iv) or (vi) of Figure 3.3. By inspection we can find the required pointers, and either case 4(a) or (b) holds. □

Definition of f. On input (U, h), $f(U, h)$ first computes U_{Fresh} and Pred (and F) as before. It then computes $\mathsf{Q}_{\mathsf{Fresh}}$, and Q_j and Used_j for each $j = 1, \ldots, F$. It initializes: (1) Array Cases and Coll, each of size F, which will hold entries from domains depending on cases, (2) A list Loops which will hold elements of $[FT]$ (i.e. "large pointers") (3) A set Bulbs which will hold elements of $[FT]$ (i.e. "large pointers") (4) A list $\mathsf{Diamonds}$ which will hold elements of $[FT] \times [FT]$, (i.e. pairs of "large pointers") (5) $\tilde{h}$ to be the table of h, but sorted to contain the responses for $\mathsf{Q}_{\mathsf{Fresh}}$, followed by the rest of the table in lexicographic order.

The computation of f next populates the sets Loops, Bulbs, and $\mathsf{Diamonds}$. Specifically, for $j = 1, \ldots, F$, it checks which case holds; If case 3a, 4a, or 4b holds, then it does the following:

Case 3a: Add u_j to Loops and delete the entry corresponding to $\mathsf{Q}_{\mathsf{Fresh}}[u_j]$ from $\tilde{h}$,

Case 4a: Add u_j^1 and u_j^2 to Bulbs, and delete the entry corresponding to $\mathsf{Q}_{\mathsf{Fresh}}[u_j^2]$ from $\tilde{h}$,

Case 4b: Add the pair (u_j^1, u_j^2) to $\mathsf{Diamonds}$ and delete the entry corresponding to $\mathsf{Q}_{\mathsf{Fresh}}[u_j^2]$ from $\tilde{h}$.

There is a subtlety in Case 4a: It may be that two bulbs are hanging off of the same vertex, when the adversary produces two Case-(iv) (from Figure 3.3) collisions with the same intermediate node. In this case our algorithm will put the second collision into Case 2 and not 4a, even though strictly speaking there was no reused edge - only a reused node (which Case 2 allows). This ensures that the queries in Bulbs will be partitioned into pairs with the same input salts, which our inversion algorithm will leverage.

We have now defined all of the outputs of f except for Cases, Coll and $\tilde{h}$, which we define now. For $j = 1, \ldots, F$, f examines Q_j and determines which of the cases above occurs for Used_j. It sets $\mathsf{Cases}[j] \in \{\mathsf{1a}, \mathsf{1b}, \mathsf{2}, \mathsf{3a}, \mathsf{3b}, \mathsf{4a}, \mathsf{4b}\}$ and performs one of the following (see Figure 3.5):

Case 1a: Set $\mathsf{Coll}[j] \leftarrow s_j \in [T]$ and delete the entry corresponding to $\mathsf{Q}_j[s_j]$ from $\tilde{h}$.

Case 1b: Set $\mathsf{Coll}[j] \leftarrow (s_j^1, s_j^2) \in [T] \times [T]$ and delete the entry corresponding to query $\mathsf{Q}_j[s_j^2]$ from $\tilde{h}$.

Case 2: Compute $v_j \in [4]$ to point to which of the (at most) four used queries in Q_{t_j} is reused, and then set $\mathsf{Coll}[j] \leftarrow (s_j, t_j, v_j) \in [T] \times [F] \times [4]$ and delete the entry corresponding to $\mathsf{Q}_j[s_j]$ from $\tilde{h}$.

Case 3b: Set $\mathsf{Coll}[j] \leftarrow (s_j^1, s_j^2, u_j)$ and delete entries corresponding to queries $\mathsf{Q}_j[s_j^1]$ and $\mathsf{Q}_j[s_j^2]$ from $\tilde{h}$.

Case 4a: Compute v_j, index of u_j^1 in Bulbs. Set $\mathsf{Coll}[j] \leftarrow (s_j, v_j) \in [T] \times [|\mathsf{Bulbs}|]$ and delete the entry corresponding to query $\mathsf{Q}_j[s_j]$ from $\tilde{h}$.

Case 4b: Set $\mathsf{Coll}[j] \leftarrow (s_j^1, s_j^2) \in [T] \times [T]$, and delete the entries corresponding to queries $\mathsf{Q}_j[s_j^1]$ and $\mathsf{Q}_j[s_j^2]$ from $\tilde{h}$.

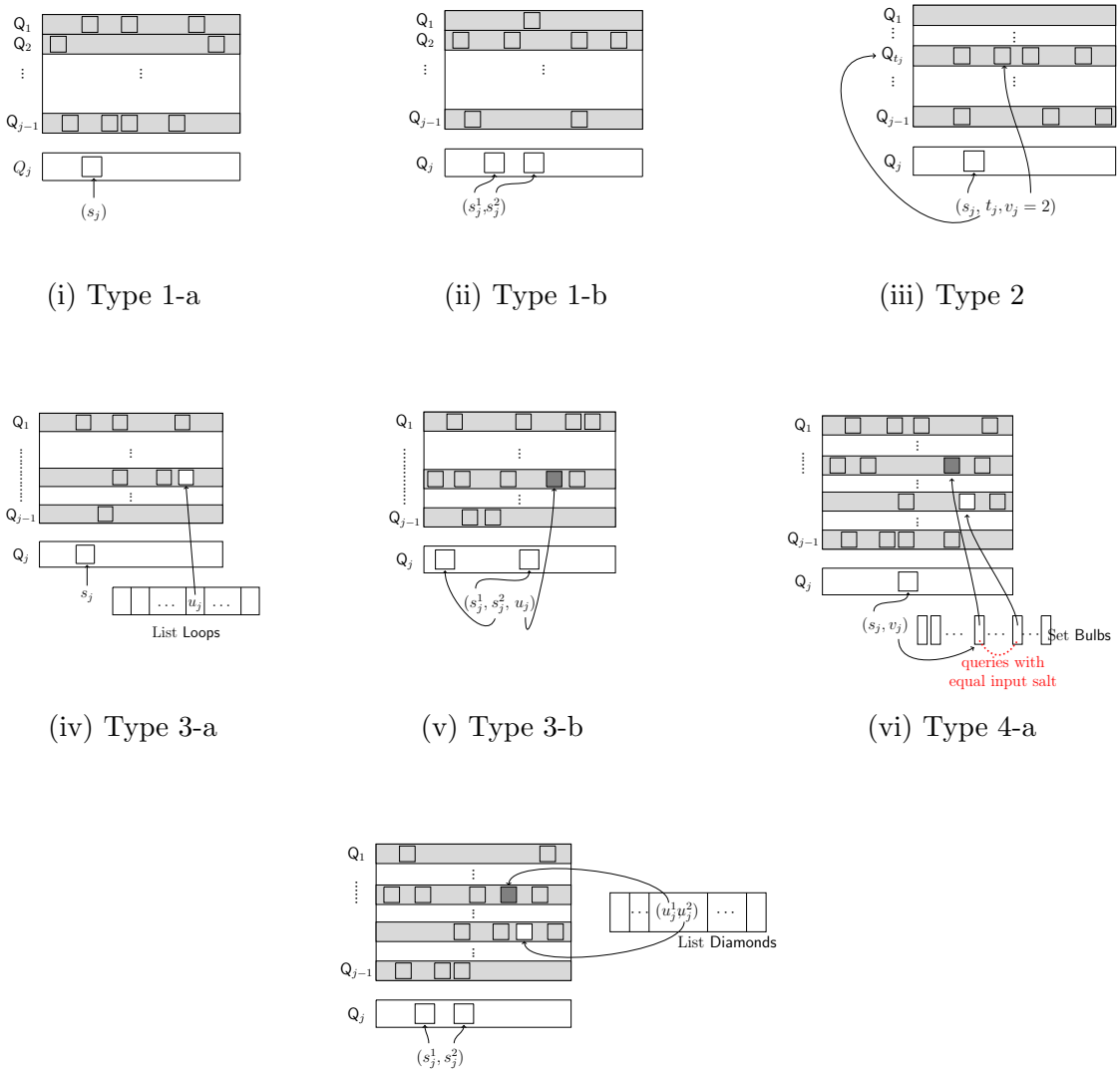
Thus, $\tilde{h}$ consists of the query responses for $\mathcal{A}_2$ when run on the salts in U_{Fresh}, except for the queries indicated to be deleted by compressor, followed by the remaining outputs of h in lexicographic order. This completes the description of f.

Analysis of f. We first argue the output length of f is not too long, and later that it is injective. Let the number of salts in U_{Fresh} having compression type 1(a) and 1(b) be δ_1 and δ_1' respectively, compression type 2 be δ_2, compression type 3(b) to be δ_3 and compression type 4(a) be $\delta_4 = |\mathsf{Bulbs}|/2$. Let $|\mathsf{Bulbs}| = n_b$, $|\mathsf{Loops}| = n_\ell$ and $|\mathsf{Diamonds}| = n_d$. Then $F = \delta_1 + \delta_1' + \delta_2 + n_\ell + \delta_3 + \delta_4 + n_d$.

Claim 14. *The number of entries deleted from $\tilde{h}$ by f is equal to $\delta_1 + \delta_1' + \delta_2 + n_\ell + 2\delta_3 + \frac{n_b}{2} + \delta_4 + 3n_d$.*

Proof. Observe that f does the following:

- deletes 1 entry from $\tilde{h}$ for each index added to Loops. So, n_ℓ entries deleted from $\tilde{h}$ when f populates Loops.
- deletes 1 entry from $\tilde{h}$ for each pair of indices added to Bulbs. So, $n_b/2$ entries deleted from $\tilde{h}$ when f populates Bulbs.
- deletes 1 entry from $\tilde{h}$ for each pair of indices added to $\mathsf{Diamonds}$. So, n_d entries deleted from $\tilde{h}$ when f populates $\mathsf{Diamonds}$.
- for every fresh salt of type 1a, 1b, 2 and 4a one entry corresponding to a new query among the queries of the salt is deleted from $\tilde{h}$. Thus, δ_1, δ_1', δ_2 and δ_4 entries will be deleted by f from $\tilde{h}$ due to fresh salts belonging to Case 1a, 1b, 2 and 4a, respectively.
- for every salt of type 3b and 4b, entries corresponding to two queries that are new in Qrs of salt are deleted from $\tilde{h}$, thus $2\delta_3$ and $2n_d$ entries are deleted by f from $\tilde{h}$ for fresh salts undergoing compression of type 3b and 4b, respectively.



(i) Type 1-a (ii) Type 1-b (iii) Type 2

(iv) Type 3-a (v) Type 3-b (vi) Type 4-a

(vii) Type 4-b

Figure 3.5: Compression of different cases for $B = 2$ by f. Boxes denote an index. Boxes with blue stripes denote used queries in Q_j. White box denotes query at the index is new (and gets compressed).

This totals to $\delta_1 + \delta_1' + \delta_2 + n_\ell + 2\delta_3 + \frac{n_b}{2} + \delta_4 + 3n_d$ entries being deleted. Therefore, to prove the claim, we need to show that f deletes the entry of any query exactly once from $\tilde{h}$. To this end, we proceed in 3 steps as follows:

Claim 15. *Any query belongs to at most one of* Loops, Bulbs *or* Diamonds.

Proof. We prove the above claim by contradiction. Lets assume there exists a query q that belongs in both Loops and Bulbs. Without loss of generality, we can assume that the query's first occurrence is at index $i \in [FT]$, it is added in Loops for some fresh salt a_j' of type 3a and added to Bulbs for some fresh salt a_k' of type 4a such that $j, k \in [F]$ and $j < k$. However, as index i is added to Loops, query q becomes a used

query in Q_j. When q is used in Q_k, it will be a reused query and hence a'_k should be of type 2 instead of type 4a and q would not be added to Bulbs, contradicting our assumption.

We can similarly show that no query can simultaneously be in Loops and $\mathsf{Diamonds}$ or Bulbs and $\mathsf{Diamonds}$, thus proving the claim. □

For every $j \in [F]$ and for any value of $\mathsf{Cases}[j]$, f deletes an entry corresponding to a new query in Q_j from $\tilde{h}$ while populating $\mathsf{Cases}[j]$ and $\mathsf{Coll}[j]$. Lets assume there exists $j, k \in [F]$ such that $j < k$ and f deletes some query q while populating Cases and Coll for both a'_j and a'_k. Then q should be a new query both in queries of a'_j and queries of a'_k for it to be deleted. However, this is not possible by the definition of new queries.

Claim 16. *If f adds a query to one of* Loops, Bulbs *and* $\mathsf{Diamonds}$, *then it never deletes its entry from $\tilde{h}$ while populating* Cases *and* Coll.

Proof. Lets assume otherwise that f adds some query q to Loops for some salt a'_j of type 3a and for some $k \in [F]$, it deletes the entry corresponding to q from $\tilde{h}$ while populating $\mathsf{Cases}[k]$ and $\mathsf{Coll}[k]$. This means, q should be a used query in both Q_j and Q_k. However, it can be a new query in at most one of Q_j and Q_k and has to be an old query in the queries of the other salt depending on whether $j < k$ or $j > k$. Thus, q would be a reused query in one of Q_j or Q_k. If $j < k$, then q is a reused query among the used queries of a'_k. This means a'_k would be of type 2 (as it is not a new query and not the first query in the chain). This means, q would not be deleted while processing $\mathsf{Cases}[k]$ and $\mathsf{Coll}[k]$. This contradicts our assumption. When $j > k$, then q would be a reused query among the queries for a'_j and then a'_j should not be of type 3a, again contradicting our assumption. Similarly, we can prove when a'_j is of type 4a or 4b. This proves the above claim. □

We set the output length of L via the following equation, where the maximum is taken over $1 \leq F \leq u$ and the $\delta_i, \delta'_1, n_\ell, n_b, n_d$ summing to F:

$$2^L = \max \binom{N}{u} N^{MN} \cdot u \cdot 2^{3F} \cdot \frac{\binom{N}{F}\binom{FT}{u-F}}{\binom{N}{u}} \cdot \left(\frac{T}{N}\right)^{\delta_1} \cdot \left(\frac{T^2}{N}\right)^{\delta'_1} \cdot \left(\frac{4FT}{N}\right)^{\delta_2}$$
$$\cdot \left(\frac{FT}{N}\right)^{n_\ell} \cdot \left(\frac{FT \cdot T^2}{N^2}\right)^{\delta_3} \cdot \frac{\binom{FT}{n_b}(T \cdot n_b)^{\delta_4}}{N^{n_b/2+\delta_4}} \cdot \left(\frac{(FT)^2 T^2}{N^3}\right)^{n_d}$$

Then using Stirling's approximation, we bound $\frac{\binom{N}{F}\binom{FT}{u-F}}{\binom{N}{u}} \leq (4e)^u \left(\frac{FT}{N}\right)^{u-F}$ exactly

as before. Next, using $n_b/2 = \delta_4 \leq F$ and $T^2 \leq N$ we simplify,

$$\begin{aligned}\frac{\binom{FT}{n_b} \cdot (T \cdot n_b)^{\delta_4}}{N^{n_b/2+\delta_4}} &\leq \left(\frac{eFT}{n_b \cdot N}\right)^{n_b} \cdot (T \cdot n_b)^{n_b/2} = \left(\frac{2e^2F^2T^2 \cdot T \cdot n_b}{2n_b^2N^2}\right)^{n_b/2} \\ &\leq \left(\frac{e^2 2^{2F/n_b} FT^2 \cdot T}{2N^2}\right)^{n_b/2} \leq (2e^2)^F \left(\frac{FT}{N}\right)^{n_b/2} \left(\frac{T^2}{N}\right)^{n_b/2} \\ &\leq (2e^2)^u \left(\frac{FT}{N}\right)^{\delta_4} \cdot \left(\frac{T^2}{N}\right)^{n_b/2}\end{aligned}$$

Thus, putting everything together and simplifying using $F \leq u \leq 2^u$, and assuming f is injective, we obtain:

$$\begin{aligned}\frac{2^L}{\binom{N}{u}N^{MN}} &\leq 2^{4u} \cdot (4e)^u \cdot (2e^2)^u \cdot \left(\frac{FT}{N}\right)^{u-F} \cdot \left(\frac{T^2}{N}\right)^{\delta_1+\delta_1'} \cdot \left(\frac{4FT}{N}\right)^{\delta_2} \cdot \\ &\quad \left(\frac{FT}{N}\right)^{n_\ell} \cdot \left(\frac{FT \cdot T^2}{N^2}\right)^{\delta_3} \cdot \left(\frac{FT}{N}\right)^{\delta_4} \cdot \left(\frac{T^2}{N}\right)^{n_b/2} \cdot \left(\frac{(FT)^2T^2}{N^3}\right)^{n_d} \\ &\leq (2^9e^3)^u \cdot \left(\frac{uT}{N}\right)^{u-F+\delta_2+n_\ell+\delta_3+\delta_4+2n_d} \cdot \left(\frac{T^2}{N}\right)^{\delta_1+\delta_1'+\delta_3+n_b/2+n_d} \\ &\leq (2^9e^3)^u \cdot \max\left\{\left(\frac{T^2}{N}\right)^u, \left(\frac{uT}{N}\right)^u\right\}\end{aligned}$$

Next, we need to prove the assumption that f is injective. In other words, it needs to be shown that given $f(U,h) = (F, U_{\mathsf{Fresh}}, \mathsf{Pred}, \mathsf{Cases}, \mathsf{Coll}, \mathsf{Loops}, \mathsf{Bulbs}, \mathsf{Diamonds}, \tilde{h})$, (U,h) can be uniquely determined. The inversion algorithm works as follows:

1. Parse the inputs, starting with F, as with the previous proof.
2. Initialize h and $\mathsf{Q}_{\mathsf{Fresh}}$ to be empty tables and array $U = U_{\mathsf{Fresh}}$.
3. For each $a'_j \in U_{\mathsf{Fresh}}$ (in lexicographic order), the j-th entry of Cases indicates the type of tuple of pointers stored in the j-th entry of Coll. Run $\mathcal{A}_2$ on a'_j and respond to queries using the entries of $\tilde{h}$ in order (except if the query is a repeat) and populating the entries of h and $\mathsf{Q}_{\mathsf{Fresh}}$ except when one of the following happens:
   (a) For $\mathsf{Cases}[j] = 1a$, when $\mathsf{Q}_j[s_j]$ is queried, respond to the query with a'_j. If $\mathsf{Cases}[j] = 1b$ continue until it reaches query at index s_j^2 in Q_j. To respond to this query, return the output of the query $\mathsf{Q}_j[s_j^1]$.
   (b) For $\mathsf{Cases}[j] = 2$, when $\mathsf{Q}_j[s_j]$ is queried, return the input salt of query $\mathsf{Q}_{t_j}[\mathsf{Used}_{t_j}[v_j]]$.
   (c) For $\mathsf{Cases}[j] = 3b$, if query pointed by s_j^1 in Q_j is made, return the input salt of the query pointed by u_j. If query pointed by s_j^2 in Q_j is made, return the output of the query pointed by u_j.

(d) For $\mathsf{Cases}[j] = 4a$, when $\mathsf{Q}_j[s_j]$ is queried, return the input salt of the query at index v_j in the set Bulbs.

(e) For $\mathsf{Cases}[j] = 4b$, when $\mathsf{Q}_j[s_j^1]$ or $\mathsf{Q}_j[s_j^2]$ is queried, return the input salt of first and second query in the first unused element of list $\mathsf{Diamonds}$, respectively.

(f) The query is in Loops, then respond to the query with the input salt of the query.

(g) The query is in Bulbs and there is a prior query with the same input salt in Bulbs, then respond to the query with the output of this prior query.

(h) The second query of an element in $\mathsf{Diamonds}$, then respond to the query with the output of the first query of the same element in $\mathsf{Diamonds}$.

After responding, continue running $\mathcal{A}_2$ on a'_j and populating the tables.

4. After running $\mathcal{A}_2$ on all of the salts in U_{Fresh}, populate the rest of h using the remaining entries of $\tilde{h}$ in order.

5. Finally, examine the queries $\mathsf{Q}_{\mathsf{Fresh}}$, and form U by adding the salts of queries in $\mathsf{Q}_{\mathsf{Fresh}}$ indexed by the elements of Pred to U_{Fresh}.

We first argue inversion replies to the queries issued by $\mathcal{A}_2$ on a'_j correctly, for each $a'_j \in U_{\mathsf{Fresh}}$. For queries that are not deleted from $\tilde{h}$, these are simply copied from $\tilde{h}$. By construction, the queries that were deleted will be copied correctly. Finally, once $\mathsf{Q}_{\mathsf{Fresh}}$ and Q are correctly computed, we have that U is correctly recovered. □

3.6 Impossibility of Improving Zero-Walk AI Attacks

The attack in Section 3.2 and the attack of Corretti et al. follow the same template: The first unbounded phase can find collisions for some salts $a_1, \ldots, a_s$, and then the second phase tries to "walk" to these salts by querying a fixed message repeatedly. The bounded-length version needs to restart the walk to obey the length bound.

A obvious improvement to these attacks would be to examine the functional graph[2] for the function $h_0(\cdot) := h(\cdot, 0)$ and select $a_1, \ldots, a_s$ that are especially likely to be reached by the random walking stage. It is tempting to conjecture such an attack is optimal for the bounded case, as it was for the unbounded case, and we are not aware of a better attack.

In this section we formalize the approach in these attacks and show that these "zero walk" attacks cannot do much better than the basic attack in Section 3.2. Concretely, we will show that these attacks can do no better (up to logarithmic factors) than $O(STB/N)$ advantage. The bound of known attacks and our bound matches up to logarithmic factors.

At the heart of this bound is a delicate concentration inequality for the size of bounded-depth trees in random functional graphs, which may be of independent

[2]That is, the graph on vertex set $[N]$ with edges directed from a to $f(a)$.

interest: Essentially, we show that with high probability, in the functional graph for a random $f : [N] \to [N]$, all of the directed depth-D trees will have at most $\tilde{O}(D^2)$ nodes. Typical results in this area (cf. **EC:FlaOdl89**) only give asymptotic expectations.

Below we formalize the notion of zero-walk adversaries and then state and prove our bound.

Definition 19. *An (S, T, B)-AI adversary $A = (A_1, A_2)$ is said to be a* zero-walk adversary *if it has the following form:*

1. *The first stage $\mathcal{A}_1$ always produces a bit-encoded output of the form $\sigma = \{(a_1, \alpha_1, \alpha'_1), \ldots, (a_s, \alpha_s, \alpha'_s)\}$ where $s = S/\lceil(\log N + 2\log M)\rceil$, $a_i \in [N]$, $\alpha_i, \alpha'_i \in [M]$.*

2. *The second stage $\mathcal{A}_2$, on input a and $\sigma = \{(a_1, \alpha_1, \alpha'_1), \ldots, (a_s, \alpha_s, \alpha'_s)\}$ and given oracle h, does the following:*

   If $a \in \{a_1, \ldots, a_s\}$, say $a = a_k$, then output α_k and α'_k.

   Else: For i in $1, \ldots, \lfloor T/B \rfloor$:

   (a) *Choose $\hat{\alpha}_i \xleftarrow{\$} [M]$. Query $c_0 \leftarrow h(a, \hat{\alpha}_i)$.*

   (b) *For j in $1, \ldots, B-1$:*
   If $c_{j-1} \in \{a_1, \ldots, a_s\}$, say $c_j = a_k$, then output $\hat{\alpha}_i \| 0^{j-1} \| \alpha_k$ and $\hat{\alpha}_i \| 0^{j-1} \| \alpha'_k$.
   Else: Query $c_j \leftarrow h(c_{j-1}, 0)$.

Theorem 17. *For any positive integers S, T, B such that $B \leq T$, $SB \geq T$ and any zero-walk (S, T, B)-AI adversary $\mathcal{A}$,*

$$\mathbf{Adv}^{\text{ai-cr}}_{\text{MD}}(\mathcal{A}) = O\left(\frac{STB\ln(NB)}{N}\right).$$

This theorem follows easily from Lemma 18 which we state now, but the proof of that lemma is technical. For a function $f \in \mathsf{Func}([N], [N])$, an element $a \in [N]$, and non-negative integer g, we define $f^{-g}(a) = \bigcup_{i=0}^{g} \{a' \in [N] : f^i(a') = a\}$, where we define f^0 to be the identity function. Note that $f^{-g}(a)$ includes all of the elements that iterate to a in g *or fewer* steps, so we have in particular that $a \in f^{-g}(a)$ for any $g \geq 0$. We say that an element $a \in [N]$ is (r, B')-*rich for* f if $|f^{-B'}(a)| \geq r$.

Lemma 18. *Let $\mathbf{f} \xleftarrow{\$} \mathsf{Func}([N], [N])$. Define $r = \lceil 1000(B')^2 \ln(NB')\rceil + 1$ and let ε be the probability that there exists $a \in [N]$ that is (r, B')-rich for $\mathbf{f}$. Then*

$$\varepsilon < 1/N.$$

We remark that lemma is much easier to prove if we settle for a weaker result with r proportional to $\tilde{\Omega}(B'^3)$.

Proof of Theorem 17. Let C be a constant to be fixed later. Let $\mathcal{A}$ be zero-walk adversary, and write $\mathbf{h}_0(\cdot)$ for $\mathbf{h}(0,\cdot)$. Let E be the event that there exist $a_1,\ldots,a_s$ such that $\bigcup_{i=1}^{s}\mathbf{h}_0^{-(B-2)}(a_i)$ has size at least $CSB^2\ln(NB)$. Then

$$\begin{aligned}\Pr[\mathsf{AI\text{-}CR}_{\mathbf{h},\mathbf{a}}(\mathcal{A})=1] &\le \Pr[\mathsf{AI\text{-}CR}_{\mathbf{h},\mathbf{a}}(\mathcal{A})=1|\neg E]+\Pr[E]\\ &\le \left(\sum_{i\in[\lfloor T/B\rfloor]}\frac{CSB^2\ln(NB)}{N}\right)+\frac{1}{N}=O\left(\frac{STB\ln(NB)}{N}\right).\end{aligned}$$

The first probability bound holds because for a fixed h, we have that $\mathcal{A}$ wins only if one of its T chosen c_0 values lies in $\bigcup_{i=0}^{s}\mathbf{h}_0^{-(B-2)}(a_i)$, which has size at most $CSB^2\ln(NB)$ when E does not hold; We simply apply a union bound over the $\lfloor T/B\rfloor$ choices of c_0. The second probability bound is by Lemma 18, with $B'=B-2$ and the constant C set appropriately. □

3.6.1 *Proof of Lemma 18*

Proof. In this section we dispense with writing random variables in bold.

Let $f\in\mathsf{Func}([N],[N])$ be uniformly random, and fix $a\in[N]$. Define *generations* G_j (random variables), $j=0,\ldots,B'$, by $G_0=\{a\}$ and for $j>0$

$$G_j=f^{-1}(G_{j-1})\setminus(\bigcup_{l<j}G_l).$$

Thus, G_j consists of elements $a'\in[N]$ such that $f^j(a')=a$ but for all $l<j$, $f^l(a')\neq a$. Define the j^{th} *population* as $P_j=|G_j|$. Then a being (r,B')-rich is equivalent to $\sum_{j=0}^{B'}P_j\ge r$.

We are going to analyze the probability that any individual P_j is exceptionally large, but this is complicated by the fact that they *will* typically grow to a reasonable size.

We define P_j to be:

"small" if $P_j<100B'\ln(NB')$,

"big" if $P_j\ge 100B'\ln(NB')$,

"really big" if $P_j\ge 200B'\ln(NB')$,

"huge" if $P_j\ge 1000B'\ln(NB')$.

We will show that with probability at least $1-1/N$, there are no huge values P_j. Then Theorem 17 follows by summing P_j over $j=0,\ldots,B'$.

Claim 19. *For any subsets $g_0,\ldots,g_j\subseteq[N]$ such that $\Pr[G_0=g_0,\ldots,G_j=g_j]\neq 0$, the distribution of P_{j+1}, conditioned on $G_0=g_0,\ldots,G_j=g_j$, is stochastically dominated by a binomial sum of $n_{j+1}=N-\sum_{i=0}^{j}|g_i|$ i.i.d. 0/1 random variables with expectation $p_{j+1}\cdot n_{j+1}\le|g_j|$.*

Proof of claim. Condition on some $g_0, \ldots, g_j$. We know that none of the elements of these previous generations can appear in G_{j+1}, so for any $a' \notin \bigcup_{i=0}^{j} g_i$, the conditional distribution of $f(a')$ is uniform over $[N] \setminus \bigcup_{i'=0}^{j-1} g_{i'}$. Moreover, for these a', $f(a')$ is independent of any other value of $f(a'')$ under our conditioning. Thus

$$\Pr[a' \in G_{j+1} | G_0 = g_0, \ldots, G_j = g_j] = p_{j+1} \quad \text{where} \quad p_{j+1} = \frac{|g_j|}{N - \sum_{i'=0}^{j-1} |g_{i'}|}.$$

The random variable P_{j+1}, conditioned upon $g_0, \ldots, g_j$, is equal to the sum over a' of indicator random variables for the event $a' \in G_{j+1}$, which gives the claim. □

We first bound the probability that either there is ever a jump from small directly to really big (or huge). Once that possibility has been dealt with, the only other way a huge generation can come about is if there is a consecutive series of generations that are all big or really big that immediately precede a huge generation. This plan is formalized in the next two claims.

Claim 20. *The probability that there exists $j = 0, \ldots, B' - 1$ such that P_j is small and P_{j+1} is really big is at most N^{-30}.*

Proof of claim. Fix some j and condition on any $g_0, \ldots, g_j$ that occur with nonzero probability and assume $|g_j| < 100B' \ln(NB')$. Then by the first claim P_{j+1} is stochastically dominated by a binomial random variable X with expectation at most $100B' \ln(NB')$. By the multiplicative Chernoff bound we have that $X \leq 200B' \ln(NB')$ except with probability

$$\exp\left(\frac{-1 \cdot 100B' \ln(NB')}{2+1}\right) < (NB')^{-33B'}$$

Since this holds conditioned on any $g_0, \ldots, g_j$, it holds without the conditioning. Finally taking a union bound over $j = 0, \ldots, B' - 1$ gives the claim (with a lot of slack to spare). □

Obtaining a cubic bound. A variant of the previous claim is already enough to prove the aforementioned weaker version of the lemma, where r is cubic in B'. We sketch how now. Instead of focusing on a jump from small to really big, we simply bound the probability that, say, $P_j \geq 100jB' \ln(NB')$. Conditioned on the previous generations not violating this bound, a simple application of Chernoff shows that P_j is very unlikely to violate this bound. Taking a union bound, we can show that for all j, $P_j < 100jB' \ln(NB')$ with high probability. Finally summing these bounds up to $j = B'$ gives the cubic bound.

We now return to obtaining the tighter quadratic bound. This will depend on the next claim.

Claim 21. *The probability that for some ℓ' and ℓ, $P_{\ell'}$ is big but not really big, $P_{\ell'}, \ldots, P_{\ell-1}$ are big or really big, and P_ℓ is huge, is at most $N^{-4.2}$.*

The proof of this claim is much longer and more delicate. We observe that the proof of the lemma will be complete using these two claims and a union bound over $a \in [N]$; The key is that both claims sum to a bound below N^{-2}. □

Proof of claim. Define random variables δ_j by

$$P_j = (1 + \delta_j)P_{j-1}.$$

We have $\delta_j \geq -1$. If we condition on some $(g_0, \ldots, g_j)$, we have the expected value of $\delta_j \leq 0$ by the first claim. Moreover, we have for any $0 < \delta < 1$

$$\Pr[\delta_{j+1} \geq \delta \mid g_0, \ldots, g_j] = \Pr[P_{j+1} \geq |g_j|(1 + \delta) \mid g_0, \ldots, g_j].$$

By the first claim and the multiplicative Chernoff bound,

$$\Pr[\delta_{j+1} \geq \delta \mid g_0, \ldots, g_j] \leq \exp\left(\frac{-\delta^2 \cdot 100B' \ln(NB')}{3}\right) < (NB')^{-33\delta^2 B'}. \tag{3.1}$$

On the other hand, if P_ℓ is huge and $P_{\ell'}$ is not really big, we have $P_\ell \geq 5P_{\ell'}$. In terms of the δ_j this means $\prod_{j=\ell'+1}^{\ell}(1 + \delta_j) \geq 5$. By $1 + x \leq e^x$, if this inequality holds then $\sum_{j=\ell'+1}^{\ell} \delta_j \geq 1.6$. We let $Q = \sum_{j=\ell'+1}^{\ell} \delta_j$ and now aim to bound $\Pr[Q \geq 1.6] \leq N^{-4.2}$, which is small enough to allow us to take a final union bound over all $a \in [N]$.

To streamline what follows, we actually consider random variables

$$\delta'_j := \begin{cases} \delta_j & \text{if } P_{j-1} \text{ is big,} \\ -N^{100} & \text{otherwise} \end{cases}.$$

Note that $\delta'_j \leq \delta_j$ and for the event $\{Q \geq 1.6\} \cap \{P_{l'}, \ldots, P_l \text{ big}\}$ to hold it is necessary and sufficient that $Q' := \sum_{l' < j \leq l} \delta'_j > 1.6$.

Bounding this probability is a delicate operation as the summands of Q' are only quasi-independent, and yet we still want to apply a Chernoff bound to this sum. Our plan is to apply a version of the "exponential method" usually used for Chernoff-like bounds. Namely, let $u(x) = x^3$, so by Markov's inequality

$$\Pr[Q' \geq 1.6] \leq \Pr[u(Q') \geq N^{4.5}] \leq \frac{E[u(Q')]}{N^{4.5}} = \frac{E\left[\prod_{j=\ell'+1}^{\ell} N^{3\delta'_j}\right]}{N^{4.5}}. \tag{3.2}$$

Thus we need to bound this expectation. We will do this by showing, for any of the relevant j, and any $g_0, \ldots, g_{j-1}$, that $E[N^{\delta'_j} | g_0, \ldots, g_{j-1}]$ is not too large. Since this conditioning is arbitrary, we will get that the expectation of the product of $N^{\delta'_j}$ is bounded by the product of the bounds.

We now bound $E[N^{\delta'_j} | g_0, \ldots, g_{j-1}]$. If the condition puts g_{j-1} as small, then $\delta'_j = -N^{100}$ and we easily have $E[N^{\delta'_j} | g_0, \ldots, g_{j-1}] < 2^{1/B'}$.

Next suppose the condition puts g_{j-1} as big. Recall that for any non-negative random variable X, $E[X] = \int_0^\infty P[X \geq x]$. Apply this to $X = N^{3\delta'_j}$ and splitting the sum, we have

$$E[N^{3\delta'_j}|g_0,\ldots,g_{j-1}] \leq \Pr[\delta'_j \leq 0.2/B'] \cdot N^{0.2/B'} + \int_{N^{0.2/B'}}^{\leq N^3} \Pr[N^{3\delta'_j} \geq c]\ dc \qquad (3.3)$$
$$+ \int_{c>N^3} \Pr[N^{3\delta'_j} \geq c]\ dc \leq N^{0.2/B'} + I_1 + I_2,$$

using I_1, I_2 to denote the two integrals above.

The probability appearing in I_1 is $\Pr[\delta'_j \geq \delta]$ where $\delta := \frac{\ln c}{3\ln N}$, a value in $\left(\frac{0.06}{B'}, 1\right)$. By Eq. (3.1),

$$\Pr[\delta'_j \geq \delta] \leq (NB')^{-33\delta^2 \cdot B'} \leq (NB')^{-33(0.06/B')\delta \cdot B'} < (NB')^{-1.8\delta}$$
$$= \exp\left(\frac{-1.8\ln(NB')\ln(c)}{3\ln N}\right) \leq c^{-0.6\ln(NB')}.$$

Then,

$$I_1 \leq \int_{N^{0.2/B'} \leq c \leq N^3} c^{-0.6\ln(NB')}\ dc < \frac{N^{-0.12\ln(NB')/B' + 0.2/B'}}{0.6\ln(NB') - 1}$$

Bounding I_2 is similar. Applying Chernoff with $\delta = \frac{\ln c}{3\ln N} \geq 1$, we have

$$\Pr[\delta'_j \geq \delta] \leq (NB')^{-33\delta^2 B'/(1+\delta)} \leq (NB')^{-15B'\delta} \leq \exp\left(\frac{-15B'\ln(NB')\ln(c)}{3\ln N}\right) \leq c^{-5B'},$$

and then

$$I_2 \leq \int_{c>N^3} c^{-5B'}\ dc < N^{-B'}.$$

Combining our three terms in (3.3), and the easy case where P_{j-1} was small, we have under any conditioning that

$$E[N^{3\delta'_j}|g_0,\ldots,g_{j-1}] \leq N^{.3/B'}$$

since the first of the three segments in the split integral dominates. We can now apply this bound and get

$$E[u(Q')] \leq (N^{.3/B'})^{\ell-\ell'} \leq N^{.3}$$

since $\ell - \ell' \leq B'$. By (3.2) this proves $\Pr[Q' \geq 1.6] < N^{.3-4.5} = N^{-4.2}$ and establishes the third claim. □

Chapter 4

Bounded Length Collisions

We revisit the problem of finding B-block-long collisions in Merkle-Damgård Hash Functions in the auxiliary-input random oracle model, in which an attacker gets a piece of S-bit advice about the random oracle and makes T oracle queries.

Akshima, Cash, Drucker and Wee (CRYPTO 2020), based on the work of Coretti, Dodis, Guo and Steinberger (EUROCRYPT 2018), showed a simple attack for $2 \leq B \leq T$ (with respect to a random salt). The attack achieves advantage $\widetilde{\Omega}(STB/2^n + T^2/2^n)$ where n is the output length of the random oracle. They conjectured that this attack is optimal. However, this so-called STB conjecture was only proved for $B \approx T$ and $B = 2$. Very recently, Ghoshal and Komargodski (CRYPTO 22) confirmed STB conjecture for all constant values of B, and provided an $\widetilde{O}(S^4TB^2/2^n+T^2/2^n)$ bound for all choices of B.

In this work, we prove an $\widetilde{O}((STB/2^n) \cdot \max\{1, ST^2/2^n\} + T^2/2^n)$ bound for every $2 < B < T$. Our bound confirms the STB conjecture for $ST^2 \leq 2^n$, and is optimal up to a factor of S for $ST^2 > 2^n$ (note as T^2 is always at most 2^n, otherwise finding a collision is trivial by the birthday attack). Our result subsumes all previous upper bounds for all ranges of parameters except for $B = \widetilde{O}(1)$ and $ST^2 > 2^n$.

We obtain our results by adopting and refining the technique of Chung, Guo, Liu, and Qian (FOCS 2020). Our approach yields more modular proofs and sheds light on how to bypass the limitations of prior techniques. Along the way, we obtain a considerably simpler and illuminating proof for $B = 2$, recovering the main result of Akshima, Cash, Drucker and Wee.

4.0.1 *Our results*

Our main contribution is the following theorem.

Theorem 22 (Informal)**.** *For any $2 < B < T$, the advantage of the best adversary with S-bit advice and T queries for finding B-block collisions in Merkle-Damgård hash functions in the auxiliary-input random oracle model, is*

$$\tilde{O}\left((STB/N) \cdot \max\{1, ST^2/N\} + T^2/N\right).$$

Our bound confirms the STB conjecture for any $2 < B < T$ for the range of S, T such that $ST^2 \leq N$. For the other range of S, T, as $T^2 \leq N$ (otherwise, finding a

collision is trivial by the birthday attack), Our bound is at most $\widetilde{O}(S^2TB/N+T^2/N)$, which is optimal up to a factor of S.

Comparing to the $\widetilde{O}(STB^2(\log^2 S)^{B-2}/N + T^2/N)$ bound by [14], our bound works for any $2 < B < T$, while their bound becomes vacuous when $B > \log N$. However, for $B \leq \log N$, unlike our bound, their bound could be tight even when $ST^2 > N$. In particular, their bound confirms STB conjecture for $B = O(1)$.

Our bound strictly improves the $\widetilde{O}(S^4TB^2/N + T^2/N)$ bound by [14], and the $\widetilde{O}(S^2T/N)$ bound by [6] for any $2 < B < T$ and non-trivial choices of S, T (specifically, when STB attack succeeds with at most a constant probability, i.e., $STB = O(N)$). The two bounds by [14] only beat [6] for $B \ll \sqrt{T}$.

As an additional contribution, we give a considerably simpler proof for proving the tight bound for $B = 2$, recovering the main result of [1].

Theorem 23 (Informal)**.** *The advantage of the best adversary with S-bit advice and T queries for finding 2-block collisions in Merkle-Damgård hash functions in the auxiliary-input random oracle model, is $\tilde{O}\left(ST/N + T^2/N\right)$.*

A comparison of our results with the prior works is summarized in 4.1. Overall, our results subsume all previous upper bounds except for the range of S, T, B such that $B \leq \log N$ and $ST^2 > N$.

4.0.2 *Our techniques*

In this section, we describe our techniques, how to use them to prove our main results, and what makes our techniques different from prior approaches used in [1], [6], [14].

Existing reduction to sequential multi-instance games. Our initial inspiration is the recent framework of Chung, Guo, Liu, Qian [15] for establishing tight time-space tradeoffs in the quantum random oracle model. Generally speaking, they reduce proving the security of a problem with S-bit advice to proving the security of multiple random instances of the problem, presented one at a time, *without* advice. Specifically, they observe that[1], if any adversary (with no advice) can solve S instances of the problem "sequentially" with success probability at most δ^S, then any adversary with S-bit advice can solve one instance of the problem with success probability at most 2δ.

This idea of reducing the security of a problem with advice to the security of a multi-instance problem without advice was first introduced by Impagliazzo and Kabanets in [11]. The idea was also used by later works [1], [14]. The difference between [11] and the later works, including this work, is that we reduce to a "sequential" multi-instance game as opposed to a "parallel" multi-instance problem. More concretely, in the parallel multi-instance problem, the adversary is presented with all the randomly chosen instances of the challenge problems to solve once at

[1]The framework of Chung, Guo, Liu, Qian [15] reduces to analyzing sequential multi-instance security for $S + \log N + 1$ instances instead of S-instances. We slightly improve their parameters and obtain a considerably cleaner version in Theorem 24.

	Best attacks	Security bounds	Ref.	Proof techniques
$B = 1$	$\frac{S}{N} + \frac{T^2}{N}$	$\frac{S}{N} + \frac{T^2}{N}$	[7]	Compression
$B = 2$	$\frac{ST}{N} + \frac{T^2}{N}$	$\frac{ST}{N} + \frac{T^2}{N}$	[1]	Multi-instance problems
$B = 2$	$\frac{ST}{N} + \frac{T^2}{N}$	$\frac{ST}{N} + \frac{T^2}{N}$	Theorem 23	Multi-instance games
$2 < B < T$	$\frac{STB}{N} + \frac{T^2}{N}$	$\frac{STB^2(\log^2 S)^{B-2}}{N} + \frac{T^2}{N}$	[14]	Multi-instance problems
$2 < B < T$	$\frac{STB}{N} + \frac{T^2}{N}$	$\frac{S^4TB^2}{N} + \frac{T^2}{N}$	[14]	Multi-instance problems
$2 < B < T$	$\frac{STB}{N} + \frac{T^2}{N}$	$\frac{STB}{N} \cdot \max\{1, \frac{ST^2}{N}\} + \frac{T^2}{N}$	Theorem 22	Multi-instance games
Unbounded	$\frac{ST^2}{N}$	$\frac{ST^2}{N}$	[6]	Presampling

Table 4.1: Asymptotic security bounds on the security of finding B-block-long collisions in Merkle-Damgård Hash Functions constructed from a random function $H : [N] \times [M] \mapsto [N]$ against (S, T)-algorithms. For simplicity, logarithmic terms and constant factors are omitted.

the start. Whereas in the multi-instance game, the adversary gets a new randomly chosen instance of challenge problem one at a time and only after solving all the previous challenges.

Chung et al. [15] recently demonstrated a separation between "sequential" multi-instance games and "parallel" multi-instance problems in the context of function inversion in the quantum setting[2]. Guo, Li, Liu and Zhang [16] pointed out a connection between "sequential" multi-instance game and the presampling technique

[2]In particular, they showed that "sequentially" inverting S random images (with T quantum queries per round to a given random function $f : [N] \to [N]$) admits security $O(ST/N + T^2/N)^S$, and the corresponding "parallel" multi-instance problems admits an attack with advantage $\Omega(ST^2/N)^S$.

(first introduced by Unruh [4], and further optimized by Coretti et al. [6]) —— the main technique used by Coretti et al. [6] for proving the $O(ST^2/N)$ bound. Roughly speaking, all results relying on presampling technique can be reproved using "sequential" multi-instance games. That suggested that "sequential" multi-instance games have the potential to prove stronger results. Therefore we are motivated to adapt and take full advantage of "sequential" multi-instance games in the context of collision finding.

To better illustrate the connection between "sequential" multi-instance games and the presampling technique, we show how to recover the $O(ST^2/N)$ bound by Coretti et al. [6]. Recall that presampling technique by Coretti et al. [6] generically reduces security proofs of unpredictability applications (including collision finding) in the AI-ROM to a much simpler P-bit-fixing random-oracle model (BF-ROM), where the attacker can arbitrarily fix the values of the random oracle on some $P := O(ST)$ coordinates, but then the remaining coordinates are chosen at random. Coretti et al. [6] showed that the security of finding collisions in Merkle-Damgård Hash Functions in the BF-ROM is $O(ST/N)$.

Using "sequential" multi-instance games, it suffices to bound the advantage of any adversary (with no advice) winning a new game, conditioning on winning all previous (up to at most S) ones, by $O(ST^2/N)$. The adversary wins all games with advantage $O(ST^2/N)^S$, which implies the desired security against S-bit advice. The key point is that the adversary (with no advice) made at most ST queries in previous games. Therefore, conditioning on any possible events of earlier games, from the view of the adversary, the random oracle is essentially a (convex combination of) bit-fixing random oracles (BF-ROM) [6], where at most ST-positions are known, and the rest remains independent and random. Hence, it suffices to prove the security of a single game in BF-ROM by $O(ST^2/N)$, which has been shown by Coretti et al. [6] as a necessary step to use the presampling technique.

Barriers of the above idea. Akshima et al. [1] pointed out a barrier to using the vanilla presampling technique towards proving $B = 2$. In particular, one can only hope to achieve $\Omega(ST^2/N)$ in the BF-ROM even for $B = 2$. Recall that, to prove the sequential multi-instance security, it is sufficient to bound the advantage of any adversary that finds a 2-block collision for a fresh salt a, conditioned on it finds 2-block collisions for all the previous random challenge salts $a_1, \cdots, a_S$.

We will call these ST queries made during the first S rounds as offline queries. Among the T queries made for a, we will call the queries that were not made during the first S rounds as online queries. Throughout the discussion, we will focus on the case that the new salt a has never been queried before in offline queries, because the other case happens with probability at most ST/N (so won't affect our conclusion). As a result, all queries starting with the challenge salt a have to be online queries.

It is clear that the adversary learns about the function not only using the online queries but also from the offline queries. The information this algorithm can take advantage of from the offline queries varies by a lot. The followings are two extreme cases:

1. The offline queries consist of exactly one single query for each of ST distinct salts.

2. The offline queries consist of one collision for each of $ST/2$ distinct salts

For the first case, the offline queries can barely help[3]. Whereas, in the second case, as long as an adversary can find a pre-image (starting with the challenge salt a) of any of these $ST/2$ salts, it finds a 2-block collision (Refer fig. 4.1). Since there are T online queries, the algorithm achieves advantage at least $ST^2/(2N)$ in the second case.

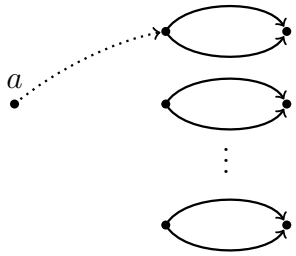


Figure 4.1: Nodes indicate salts in $[N]$. An arrow connected two salts means there is a query on the starting salt and a message in $[M]$ such that the output is the other salt. An online query hits an existing collision. Solid lines denote offline queries. The dotted line denotes the online query that forms a 2-block collision.

The vanilla presampling approach works for worst-case offline queries. Given the above example, the best security bound one can hope to achieve in the BF-ROM for $B = 2$ is $\Omega(ST^2/N)$.

A detailed comparison with prior techniques. The similarity between [1], [14] and us is that we all adopt the idea of reducing the problem of interest to a multi-instance variant, in which an adversary has to solve multiple copies of the given problem.

Both [1] and [14] directly analyze the probability of solving all instances using the compression paradigm, which typically requires a non-trivial case analysis of the more complicated *multi-instance* problem. These case analyses may be quite laborious and detached from the single-instance problem (thus may not give many insights for the single-instance problem).

Our approach differs significantly from [1] and [14] in two places. First, we focus on analyzing a simple variant of the *single-instance* problem (corresponding to a single round of the sequential multi-instance game conditioning on winning previous games), which is sufficient to establish desired results in multi-instance security. This variant is more similar to the original problem, and may be easier to analyze than the multi-instance problems. The first step (reducing to a variant of the single-instance problem) is somewhat used and captured in the presampling

[3]We do not prove it rigorously here. Instead, we focus on the more interesting case – offline queries do provide advantages.

technique (via a different route [6]). We do think this step is more modular than [1] and [14], but don't consider this as our main technical novelty.

The second place, also our main technical novelty, is that we further introduce "knowledge gaining events" for analyzing the variant of the single-instance problem. These events can be isolated and analyzed on their own, and precisely highlight the correlation in finding collisions given "typical" presampled random oracles. Before this work, all the presampling techniques for time-space tradeoffs considered worst-case presampled random oracles. The worst-case presampling may make the existing analyses sub-optimal. Our approach analyzes the "average-case" presampling random oracles and shows that those "worst-case" ones can never happen except with a tiny probability. To our best knowledge, this is the first work that takes advantage of "average-case" presampling and achieves tight bounds.

Overall, we consider our proofs more modular, because we utilize sequential games to focus on variants of the single-instance game (rather than directly compressing multi-instance games used by [1] and [14]). We further introduce "knowledge gaining events" to take advantage of "average-case" presampling (rather than working with worst-case ones used by [6]).

4.1 Notations and Definitions

Notation. For non-negative integers N, k, we write $[N]$ for $\{1, 2, \cdots, N\}$ and $\binom{[N]}{k}$ for the collection of all size-k subsets of $[N]$. For a finite set X, we write X^+ for the set of tuples of 1 or more elements of X. Random variables will be written in bold, and we write $\mathbf{x} \leftarrow_\$ X$ to indicate that $\mathbf{x}$ is a uniform random variable in X.

Chernoff Bound. Suppose $\mathbf{X}_1, \cdots, \mathbf{X}_t$ are independent binary random variables. Let $\mathbf{X}$ denote their sum and $\mu = \mathbb{E}[\mathbf{X}]$. For any $\delta \geq 0$,

$$\Pr[\mathbf{X} \geq (1+\delta)\mu] \leq \exp\left(-\frac{\delta^2 \mu}{2+\delta}\right).$$

Random Oracle bellare1993random. In random oracle model, we model a hash function as a random function H that is sampled uniformly at random from all functions at the beginning. H is publicly accessible to every entity.

A useful property about random oracle model is that, instead of sampling H uniformly at random, one can assume H is initialized as a function that always outputs $\perp$; which indicates the response has not been sampled. Whenever an input x is queried and $H(x)$ has not been sampled (i.e. $H(x) = \perp$), the random oracle samples y uniformly from the range and $H(x) := y$.

Definition 20 (Lazy Sampling and Databases)**.** *We refer to the table of sampled queries (for those $H(x) \neq \perp$) on H and their responses as the database or the partially sampled random oracle.*

*The set of **offline queries** is the set of distinct queries made in the offline stage. The set of **online queries** is the set of distinct queries made in the online stage and had not been made in the offline stage.*

While dealing with algorithms with both offline and online stages, the table of only the offline queries on H and their responses is referred to as the offline database.

Note that the outputs of the offline and online queries are independent and uniformly distributed.

4.1.1 *Merkle-Damgård Hash Functions (*MD*)*

A hash function usually is required to function over inputs with different lengths. Many practical hash functions are based on the Merkle-Damgård construction (MD). It takes a hash function with fixed length input to a new hash function with arbitrary input lengths.

We treat the underlying hash function as a random oracle $H : [N] \times [M] \to [N]$. We call a message $\mathbf{m}$ is a B-block message if $\mathbf{m}$ can be written as $\mathbf{m} = (m_1, \cdots, m_B)$ where each $m_i \in [M]$. The function $\mathsf{MD}_H(a, \mathbf{m})$ evaluates on a salt $a \in [N]$ and a message $\mathbf{m}$ as the follows:

$$\mathsf{MD}_H(a, \mathbf{m}) = \mathsf{MD}_H^{\ell}(a, (m_1, \cdots, m_\ell)) = \begin{cases} H(\mathsf{MD}_H^{\ell-1}(a, (m_1, \cdots, m_{\ell-1})), m_\ell) & \ell > 1 \\ H(a, m_1) & \ell = 1 \end{cases}$$

It applies the fixed-length hash function H on the salt a and the first block m_1 to get a new salt a_2; it then applies H again on a_2 and m_2 until finally it outputs a single string in $[N]$.

4.1.2 *Collision-Resistance against Auxiliary Input (AI).*

We start by defining the security game of collision-resistance against auxiliary input adversaries. The adversary is unbounded in the preprocessing stage and leave nothing but a piece of bounded-length advice for the online stage.

Definition 21 ((S, T)-AI algorithm)**.** *A pair of algorithms $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ is an (S, T)-AI adversary for* MD *if*

- $\mathcal{A}_1^H$ *is unbounded (making unbounded number of oracle queries to H) and outputs S bits of advice σ;*
- $\mathcal{A}_2^H$ *takes σ and a salt $a \in [N]$, issues T queries to H and outputs $\mathbf{m}_1, \mathbf{m}_2$.*

We are ready to define the security game of collision-resistance against an (S, T)-AI adversary.

Definition 22 (Auxiliary-Input Collision-Resistance)**.** *We define the following game B-*AICR *for a fixed random oracle H and a salt $a \in [N]$ in fig:AICRB, where B is a function of N (the range size of the random oracle). The game outputs* 1 *(indicating that the adversary wins) if and only if $\mathcal{A}$ outputs a pair of* MD *collision with at most $B(N)$ blocks.*

For an (S, T)-AI adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, we define the advantage of $\mathcal{A}$ as its winning probability in the B-$\mathsf{AICR}_{H,a}$ with uniformly random $H \leftarrow \{f : [N] \times$

Game B-$\mathsf{AICR}_{H,a}(\mathcal{A})$
$\sigma \leftarrow \mathcal{A}_1^H$
$\mathbf{m}_1, \mathbf{m}_2 \leftarrow \mathcal{A}_2^H(\sigma, a)$
If $\mathbf{m}_1$ or $\mathbf{m}_2$ consists of more than $B(N)$ blocks
Then Return 0
If $\mathbf{m}_1 \neq \mathbf{m}_2$ and $\mathsf{MD}_H(a, \mathbf{m}_1) = \mathsf{MD}_H(a, \mathbf{m}_2)$
Then Return 1
Else Return 0

Figure 4.2: B-$\mathsf{AICR}_{H,a}(\mathcal{A})$

Game 2-$\mathsf{AICR}_{H,a}(\mathcal{A})$
$\sigma \leftarrow \mathcal{A}_1^H$
$\mathbf{m}_1, \mathbf{m}_2 \leftarrow \mathcal{A}_2^H(\sigma, a)$
If $\mathbf{m}_1$ or $\mathbf{m}_2$ consists of more than 2 blocks
Then Return 0
If $\mathbf{m}_1 \neq \mathbf{m}_2$ and $\mathsf{MD}_H(a, \mathbf{m}_1) = \mathsf{MD}_H(a, \mathbf{m}_2)$
Then Return 1
Else Return 0

Figure 4.3: 2-$\mathsf{AICR}_{H,a}(\mathcal{A})$

Game B-$\mathsf{AICR}_{H,a}(\mathcal{A})$
$\sigma \leftarrow \mathcal{A}_1^H$
$\mathbf{m}_1, \mathbf{m}_2 \leftarrow \mathcal{A}_2^H(\sigma, a)$
If $\mathbf{m}_1$ or $\mathbf{m}_2$ consists of more than $B(N)$ blocks
Then Return 0
If $\mathbf{m}_1 \neq \mathbf{m}_2$ and $\mathsf{MD}_H(a, \mathbf{m}_1) = \mathsf{MD}_H(a, \mathbf{m}_2)$
Then Return 1
Else Return 0

Figure 4.4: B-$\mathsf{AICR}_{H,a}(\mathcal{A})$

Game 2-$\mathsf{AICR}_{H,a}(\mathcal{A})$
$\sigma \leftarrow \mathcal{A}_1^H$
$\mathbf{m}_1, \mathbf{m}_2 \leftarrow \mathcal{A}_2^H(\sigma, a)$
If $\mathbf{m}_1$ or $\mathbf{m}_2$ consists of more than 2 blocks
Then Return 0
If $\mathbf{m}_1 \neq \mathbf{m}_2$ and $\mathsf{MD}_H(a, \mathbf{m}_1) = \mathsf{MD}_H(a, \mathbf{m}_2)$
Then Return 1
Else Return 0

Figure 4.5: 2-$\mathsf{AICR}_{H,a}(\mathcal{A})$

$[M] \to [N]\}$ *and random* $a \leftarrow [N]$. *We define the* (S, T, B)*-auxiliary-input collision-resistance of Merkle-Damgård, denoted by* $\underline{\mathsf{Adv}^{\text{AI-CR}}_{\text{B-MD}}(S, T)}$*, as the maximum of advantage taken over all* (S, T)*-AI adversaries* $\mathcal{A}$.

For convenience, we similarly define $\underline{\mathsf{Adv}^{\text{AI-CR}}_{\text{2-MD}}(S, T)}$ *as the maximum of advantage of winning the game* 2-AICR *(see fig:AICRTWO) taken over all* (S, T)*-AI adversaries* $\mathcal{A}$.

Multi-Instance Collision-Resistance (MI). We then define the sequential multi-instance collision-resistance of Merkle-Damgård. As shown by [15], the AI-security is closely related to the (sequential) MI-security. Note that in the MI security, an adversary does not take any advice but tries to solve independent instances sequentially.

Definition 23 (Multi-Instance Collision-Resistance)**.** *Fixing functions* B *and* S*, and a random oracle* H*, we define the following game* B-MICR^S *in fig:MICRB. In this game,* $\mathcal{A}$ *will receive* S *freshly independent and uniform salts and it needs to find a* MD *collision with respect to each salt* a_i *of at most* B *blocks,* in a sequential order*. In other words,* $\mathcal{A}$ *will never see the next challenge salt until it solves the current one.*

Game B-$\mathsf{MICR}^S_{H,a}(\mathcal{A})$
  For $i \in \{1, 2, \cdots, S\}$:
    Sample $a_i \leftarrow [N]$
    $\mathbf{m}_1, \mathbf{m}_2 \leftarrow \mathcal{A}^H(a_i)$
    If $\mathbf{m}_1$ or $\mathbf{m}_2$ consists of more than B blocks,
    or $\mathsf{MD}_H(a_i, \mathbf{m}_1) \neq \mathsf{MD}_H(a_i, \mathbf{m}_2)$
      Return 0
  Return 1

Figure 4.6: Games B-$\mathsf{MICR}^S_{H,a}(\mathcal{A})$.

In this security game, $\mathcal{A}$ *is a statefulalgorithm that maintains its internal state between each stage. We usually consider an* (S, T)*-MI adversary* $\mathcal{A}$ *which makes at most* T *queries in each of these* S *stages. We similarly define* 2-MICR *by setting* $B = 2$ *in* B-MICR.

For an (S, T)*-MI adversary* $\mathcal{A}$*, we define the advantage of* $\mathcal{A}$ *as its winning probability in the* B-$\mathsf{MICR}^S_{H,a}$ *with uniformly random* H *and* $a \leftarrow [N]$.

We define the (S, T, B)*-multi-instance collision-resistance of Merkle-Damgård, denoted by* $\underline{\mathsf{Adv}^{\text{MI-CR}}_{\text{B-MD}}(S, T)}$*, as the maximum of advantage taken over all* (S, T)*-MI adversaries* $\mathcal{A}$.

For convenience, we similarly define $\underline{\mathsf{Adv}^{\text{MI-CR}}_{\text{2-MD}}(S, T)}$ *as the maximum of advantage of winning the game* 2-$\mathsf{MICR}^S_{H,a}$ *(for random* H, a*) taken over all* (S, T)*-MI adversaries* $\mathcal{A}$.

The following theorem will be useful for proving the AI collision-resistance of Merkle-Damgård. It says a lower bound for the MI collision-resistance implies

a lower bound for the AI security. Therefore, in the rest of the paper, we will focus on the MI collision-resistance of Merkle-Damgård with different lengths B. The theorem is based on the idea of Theorem 4.1 in [15], which implies that if $\mathsf{Adv}^{\text{MI-CR}}_{\text{B-MD}}(S + \log N + 1, T) \leq \delta^{S+\log N+1}$, then $\mathsf{Adv}^{\text{AI-CR}}_{\text{B-MD}}(S, T) \leq 4\delta$. We slightly improve their parameter, and obtain a considerably cleaner statement.

Theorem 24. *For any S, T, B and $0 \leq \delta \leq 1$, if $\mathsf{Adv}^{\text{MI-CR}}_{\text{B-MD}}(S, T) \leq \delta^S$, then $\mathsf{Adv}^{\text{AI-CR}}_{\text{B-MD}}(S, T) \leq 2\delta$.*

Proof of thm:mi$_t$o$_a$i We prove by contradiction. Assume there is an (S, T)-AI adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ such that

$$\Pr_{H,a}\left[B\text{-}\mathsf{AICR}_{H,a}(\mathcal{A}) = 1\right] > 2\delta,$$

Consider the following (S, T)-MI adversary $\mathcal{B}$:

1. $\mathcal{B}$ samples a uniformly random σ of S bits.
2. For each stage $i \in [S]$:
   - $\mathcal{B}$ receives a_i from the challenger.
   - $\mathcal{B}$ runs $\mathcal{A}_2^H(\sigma, a_i)$ to obtain and output $\mathbf{m}_1, \mathbf{m}_2$.

We will show that $\Pr_{H,a_1,\ldots,a_S}\left[B\text{-}\mathsf{MICR}^S_H(\mathcal{B}) = 1\right] > \delta^S$. For every fixed choice of H, we define

$$\delta_H := \Pr_a\left[B\text{-}\mathsf{AICR}_{H,a}(\mathcal{A}) = 1\right].$$

Observe that $\mathbf{E}_H[\delta_H] = \Pr_{H,a}\left[B\text{-}\mathsf{AICR}_{H,a}(\mathcal{A}) = 1\right] > 2\delta$. For every fixed choice of H, conditioning on that $\mathcal{B}$ guesses the output of $\mathcal{A}_1^H$ correctly, then $\mathcal{B}$ perfectly simulates $\mathcal{A}$. Therefore,

$$\Pr_{a_1,\ldots,a_S}\left[B\text{-}\mathsf{MICR}_H(\mathcal{B}) = 1\right] \geq \Pr_{a_1,\ldots,a_S}\left[B\text{-}\mathsf{MICR}_H(\mathcal{B}) = 1 \,|\, \sigma = \mathcal{A}_1^H\right] \cdot \Pr[\sigma = \mathcal{A}_1^H] = \delta_H^S / 2^S\,.$$

By averaging over the randomness of H,

$$\Pr_{H,a_1,\ldots,a_S}\left[B\text{-}\mathsf{MICR}_{H,a}(\mathcal{B}) = 1\right] \geq \mathbf{E}_H[\delta_H^S]/2^S \geq \mathbf{E}[\delta_H]^S/2^S > \delta^S\,,$$

where the second inequality is by Jensen's inequality, and the last inequality is by $\mathbf{E}_H[\delta_H] > 2\delta$. □

4.2 Auxiliary Input Collision Resistance for $B = 2$ Merkle-Damgård

In this section we prove the following theorem, which recovers Theorem 7 in [1].

Theorem 25. *For any S, T and $N \geq 64$,*

$$\mathsf{Adv}^{\text{AI-CR}}_{\text{2-MD}}(S, T) \leq (200 \log^2 N) \cdot \frac{ST + T^2}{N}\,.$$

By thm:mi$_t o_a i, it suffices to prove the following lemma.$

Lemma 26. *For any S, T and $N \geq 64$, $\mathsf{Adv}^{\text{MI-CR}}_{\text{2-MD}}(S,T) \leq \frac{100(ST+T^2)\log^2 N}{N}$.*

The purpose of this section is to show the simplicity of our new framework. The proof will also serve as a stepping stone for a better understanding of our proof for larger B cases.

Proof of lemma:micr$_t$wo Let H be a random oracle in the game 2-MICR^S and $\mathcal{A}$ be an arbitrary (S,T)-MI adversary. We show that its advantage of succeeding in 2-MICR^S is at most $(100(ST+T^2)\log^2 N/N)^S$. In this proof, we will also assume the random oracle H is lazily sampled by the challenger, which is equivalent to being sampled at the very beginning.

Let $\mathbf{X}_i$ be the indicator variable that $\mathcal{A}$ wins the i-th stage on a uniformly random salt a_i. The advantage of $\mathcal{A}$ can be then written as $\Pr[\mathbf{X}_1 \wedge \cdots \wedge \mathbf{X}_S]$. We additionally define the indicator variable $\mathbf{X}_{<i} = \mathbf{X}_1 \wedge \cdots \wedge \mathbf{X}_{i-1}$, meaning whether $\mathcal{A}$ wins the first $(i-1)$ stages of the sequential game. Then

$$\Pr[\mathbf{X}_1 \wedge \ldots \wedge \mathbf{X}_S] = \prod_{i=1}^{S} \Pr[\mathbf{X}_i | \mathbf{X}_{<i}]. \tag{4.1}$$

We will bound $\Pr[\mathbf{X}_{<i+1}] < (\delta_S)^i$ for each $i \in \{1, \cdots, S\}$ by induction, where $\delta_S = 100 \cdot \frac{(ST+T^2)\log^2 N}{N}$.

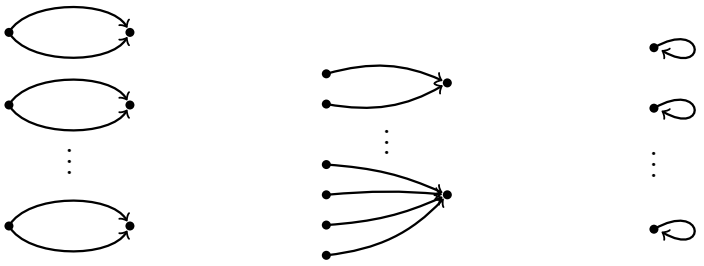
If $\Pr[\mathbf{X}_{<i}]$ is already bounded by $(\delta_S)^i$, then it trivially holds for $\Pr[\mathbf{X}_{<i+1}]$. Otherwise, we assume $\Pr[\mathbf{X}_{<i}] \geq (\delta_S)^i$.

We want to bound $\Pr[\mathbf{X}_i | \mathbf{X}_{<i}] \leq \delta_S$ for any arbitrary $i \in [S]$. In the following proof, we will carefully deal with the conditioning on $\mathbf{X}_{<i}$, since $\mathcal{A}$ learns about the function H not only using the T queries in the i-th stage, but also from these $(i-1)T$ queries in the early stages. We will call all the queries made in the previous $(i-1)$ stages as "offline" queries and those made in the i-th stage as "online" queries. We also recall the definition for "databases" in def:database.

As mention in the introduction, one bad example is that the previous $(i-1)T$ queries consist of $(i-1)T/2$ distinct salts, each has a pair of 1-block collision. An online adversary can use T queries to hit any of these salts and form a 2-block collision with probability roughly iT^2/N. Below, we will show that this event (and other events that give non-trivial advantage to the online adversary) happens with very small probability.

Defining Knowledge-Gaining Events. To bound the knowledge that $\mathcal{A}$ learns in the previous stages, we define the following events: all events are defined for the lazily sampled random oracle right after the first $(i-1)$ stages. We are going to show that these events are the "only events" that $\mathcal{A}$ can learn take advantage of the previous queries but they happen with very small probability.

- Let $\mathbf{E}_1^i$ be the event that 1-block collisions can be found for at least $10i \log N$ distinct salts within $(i-1)T$ queries.



(i) $\mathbf{E}_1^i$ (ii) $\mathbf{E}_2^i$ (iii) $\mathbf{E}_3^i$

Figure 4.7: All events $\mathbf{E}_1^i, \mathbf{E}_2^i, \mathbf{E}_i^3$. Nodes indicate salts in $[N]$. An arrow connected two salts means there is a query on the starting salt and a message in $[M]$, and the output is the other salt.

Formally, in the database, there exist $10i \log N$ salts: for each such salt a, there exists $m \neq m' \in [N]$ satisfying $H(a, m) = H(a, m')$. See fig:B2event1.

- Let $\mathbf{E}_2^i$ be the event that at least $10i^2 \log^3 N$ pairs of block collisions can be found within $(i-1)T$ queries.

  Formally, in the database, there exist $10i^2 \log^3 N$ pairs of inputs $(a, m) \neq (a', m')$ satisfying $H(a, m) = H(a', m')$. We emphasize that we do not ask a pair of collision to start with distinct salts. See fig:B2event2.

- Let $\mathbf{E}_3^i$ be the event that self loops can be found for at least $10i \log N$ distinct salts within $(i-1)T$ queries.

  Formally, in the database, there exist $10i \log N$ distinct salts: for each such salt a, there exists some $m \in [N]$ satisfying $H(a, m) = a$. See fig:B2event3.

Then

$$\begin{aligned}\Pr[\mathbf{X}_i|\mathbf{X}_{<i}] &\leq \Pr[\mathbf{X}_i|\mathbf{X}_{<i} \wedge \overline{\mathbf{E}_1^i} \wedge \overline{\mathbf{E}_2^i} \wedge \overline{\mathbf{E}_3^i}] + \Pr[\mathbf{E}_1^i \vee \mathbf{E}_2^i \vee \mathbf{E}_3^i|\mathbf{X}_{<i}] \\ &\leq \Pr[\mathbf{X}_i|\mathbf{X}_{<i} \wedge \overline{\mathbf{E}_1^i} \wedge \overline{\mathbf{E}_2^i} \wedge \overline{\mathbf{E}_3^i}] + \frac{\Pr[\mathbf{E}_1^i]}{\Pr[\mathbf{X}_{<i}]} + \frac{\Pr[\mathbf{E}_2^i]}{\Pr[\mathbf{X}_{<i}]} + \frac{\Pr[\mathbf{E}_3^i]}{\Pr[\mathbf{X}_{<i}]}.\end{aligned}$$

Here we use the fact that $\Pr[\mathbf{A}|\mathbf{B}] \leq \Pr[\mathbf{A}]/\Pr[\mathbf{B}]$ for $\Pr[\mathbf{B}] > 0$.

Next, we will show that assuming none of $\mathbf{E}_1^i, \mathbf{E}_2^i, \mathbf{E}_3^i$ happens, an adversary can not take too much advantage of the information from the previous stages. We show that its advantage $\Pr[\mathbf{X}_i|\mathbf{X}_{<i} \wedge \overline{\mathbf{E}_1^i} \wedge \overline{\mathbf{E}_2^i} \wedge \overline{\mathbf{E}_3^i}]$ is bounded by $98 \cdot (ST + T^2) \log^2 N/N$. Secondly, any of these event happens with very small probability. We can safely "assume" these events never happen. In total, the conditional probability is at most $100 \cdot (ST + T^2) \log^2 N/N = \delta_S$.

Claim 27. *For any $i \in [S]$ and $T^2 \leq N/2$, $\Pr[\mathbf{E}_1^i] \leq N^{-10i}$.*

Claim 28. *For any $i \in [S]$, $iT + T^2 < N/2$ and $N \geq 64$, $\Pr[\mathbf{E}_2^i] \leq 4N^{-2i}$.*

Claim 29. *For any $i \in [S]$, $N \geq 4$ and $T \leq N/2$, $\Pr[\mathbf{E}_3^i] \leq N^{-4i}$.*

The proofs for these lemma are deferred to the end of this section (sec:bound$_e$$vents_b$2).$Fornow$,

Recall that we assume $\Pr[X_{<i}] \geq (\delta_S)^i$, otherwise $\Pr[\mathbf{X}_1 \wedge \ldots \wedge \mathbf{X}_i] \leq (\delta_S)^i$ holds trivially for the first i stages. Therefore,

$$\Pr[\mathbf{X}_i|\mathbf{X}_{<i}] \leq \Pr[\mathbf{X}_i|\mathbf{X}_{<i} \wedge \overline{\mathbf{E}_1^i} \wedge \overline{\mathbf{E}_2^i} \wedge \overline{\mathbf{E}_3^i}] + \frac{\Pr[\mathbf{E}_1^i]}{\Pr[\mathbf{X}_{<i}]} + \frac{\Pr[\mathbf{E}_2^i]}{\Pr[\mathbf{X}_{<i}]} + \frac{\Pr[\mathbf{E}_3^i]}{\Pr[\mathbf{X}_{<i}]} \quad (4.2)$$

$$\leq \Pr[\mathbf{X}_i|\mathbf{X}_{<i} \wedge \overline{\mathbf{E}_1^i} \wedge \overline{\mathbf{E}_2^i} \wedge \overline{\mathbf{E}_3^i}] + \frac{1}{N}, \quad (4.3)$$

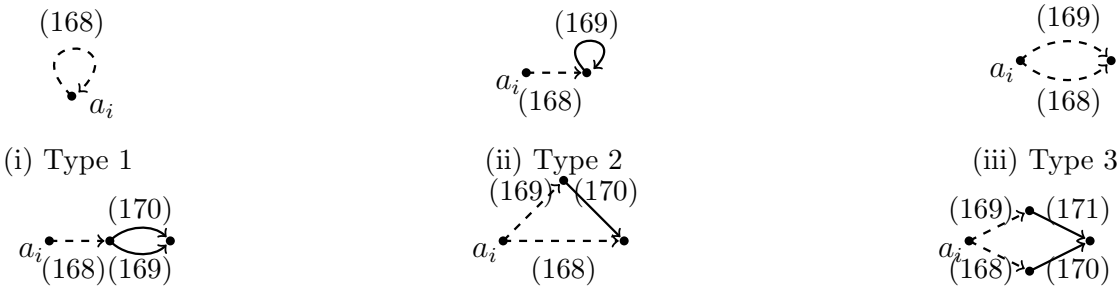
where the last inequality comes from the fact that $1/\Pr[\mathbf{X}_{<i}] \leq N^i$ but $(\Pr[\mathbf{E}_1^i] + \Pr[\mathbf{E}_2^i] + \Pr[\mathbf{E}_3^i]) \leq 6N^{-2i}$.

Bounding the Last Term. Finally, we are going to bound $\Pr[\mathbf{X}_i|\mathbf{X}_{<i} \wedge \overline{\mathbf{E}_1^i} \wedge \overline{\mathbf{E}_2^i} \wedge \overline{\mathbf{E}_3^i}]$. In order to do that, we define another event $\mathbf{G}$ as the event that the input salt a_i has been queried among the queries in the previous $(i-1)$ iterations; i.e., for some $m \in [N]$, (a_i, m) is in the lazily sampled hash function. Then it holds that:

$$\begin{aligned}
&\Pr\left[\mathbf{X}_i \middle| \mathbf{X}_{<i} \wedge \overline{\mathbf{E}_1^i} \wedge \overline{\mathbf{E}_2^i} \wedge \overline{\mathbf{E}_3^i}\right] \\
\leq &\Pr\left[\mathbf{G} \middle| \mathbf{X}_{<i} \wedge \overline{\mathbf{E}_1^i} \wedge \overline{\mathbf{E}_2^i} \wedge \overline{\mathbf{E}_3^i}\right] + \Pr\left[\mathbf{X}_i \middle| \mathbf{X}_{<i} \wedge \overline{\mathbf{E}_1^i} \wedge \overline{\mathbf{E}_2^i} \wedge \overline{\mathbf{E}_3^i} \wedge \overline{\mathbf{G}}\right] \\
\leq &\frac{(i-1)T}{N} + \Pr\left[\mathbf{X}_i \middle| \mathbf{X}_{<i} \wedge \overline{\mathbf{E}_1^i} \wedge \overline{\mathbf{E}_2^i} \wedge \overline{\mathbf{E}_3^i} \wedge \overline{\mathbf{G}}\right].
\end{aligned}$$

Now all that remains to bound is $\Pr\left[\mathbf{X}_i \middle| \mathbf{X}_{<i} \wedge \overline{\mathbf{E}_1^i} \wedge \overline{\mathbf{E}_2^i} \wedge \overline{\mathbf{E}_3^i} \wedge \overline{\mathbf{G}}\right]$, which requires collision type-wise analysis. By enumeration, there are total 6 types of 2-block collisions (fig:B2alltype).

A dashed line origins from a_i. It indicates that the query should be made online, conditioned on $\overline{\mathbf{G}}$. Other queries can be either made online or offline in the previous iterations. The label 168, 169, 170 and 171 will be used later for a better presentation of our proof. By enumerating each solid edge being an online query or a offline query, we show that it is sufficient to consider the cases in Claim 30.



(i) Type 1 (ii) Type 2 (iii) Type 3

(iv) Type 4 (v) Type 5 (vi) Type 6

Figure 4.8: All types of 2-block collisions.

Claim 30. *For any $i \in [S]$, to find a 2-block collision on a_i conditioned on $\overline{\mathbf{G}}$, the queries should satisfy at least one of the following conditions:*

1. *There exists an online query (i.e., a query among the T queries in the i-th iteration after receiving the challenge input a_i), denoted (a, m) such that $H(a, m) = a$.*

   In other words, a self loop is found among the online queries. This covers the case when (168) edge in type 1 collisions and the (169) edge in type 2 collisions are online queries. See fig:B2type1.

2. *There exists two online queries, denoted (a, m) and (a', m'), such that $(a, m) \neq (a', m')$ and $H(a, m) = H(a', m')$.*

   A collision is found among the online queries. This covers the case when the (168) and (169) edges in Type 3 collisions, the (169) and (170) edges in Type 4 collisions, the (168) and (170) edges in Type 5 collisions, the (170) and (171) edges in Type 6 collisions are online queries. See fig:B2type2.

3. *There exists an online query, denoted by (a, m), and one offline query, denoted by (a', m'), such that $a \neq a'$, $H(a, m) = a'$ and $H(a', m') = a'$.*

   This denotes an online query hits an existing self loop. This covers the case when the (168) edge in type 2 collisions is an online query. See fig:B2type3.

4. *There exists an online query, denoted by (a, m), and two offline queries, denoted by (a', m') and (a', m''), such that $a \neq a'$, $H(a, m) = a'$ and $H(a', m') = H(a', m'')$.*

   This denotes an online query hits an existing collision (starting with the same salt a'). This covers the case when (168) edge in type 4 collisions is an online query. See fig:B2type4.

5. *There exists two online queries, denoted by (a, m) and (a', m'), and an offline query, denoted by (a', m'') such that $a \neq a'$, $H(a, m) = a'$ and $H(a', m') = H(a', m'')$.*

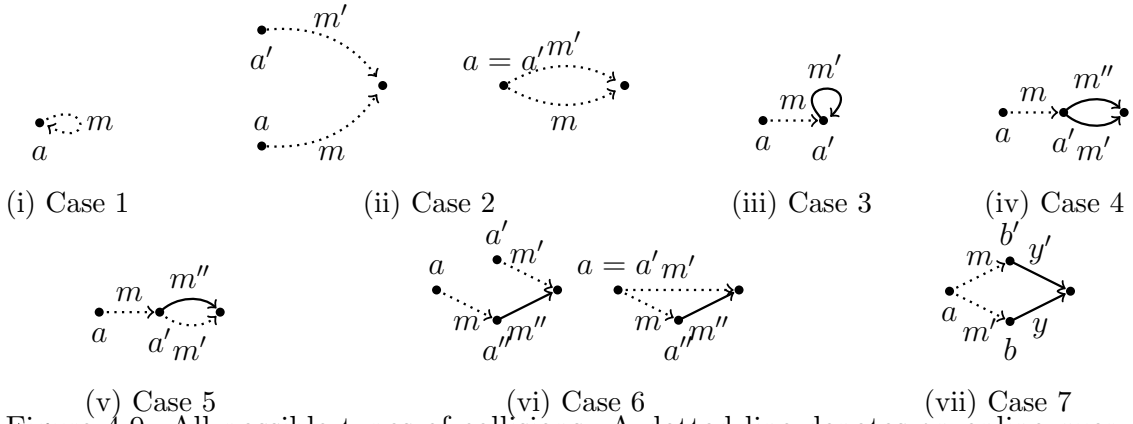
   This covers the case when the (168) and (169) edges in type 4 collisions are online queries. See fig:B2type5.

6. *There exists two online queries, denoted by (a, m) and (a', m'), and an offline query, denoted by (a'', m'') such that $H(a, m) = a'$ and $H(a', m') = H(a'', m'')$.*

   This denotes two online queries hit two ends of an existing queries. This covers the case when the (168) and (169) edges in type 5 collisions, the (168) and (171) edges in type 6 collisions are online queries. See fig:B2type6.

7. *There exists two online queries, denoted by (a, m) and (a, m'), and two offline queries, denoted by $(b, y), (b', y')$ such that $b \neq b'$, $H(a, m) = b, H(a, m') = b'$ and $H(b, y) = H(b', y')$.*

   This covers the case when the (168) and (169) edges in type 6 collisions are online queries. See fig:B2type7.



(i) Case 1 (ii) Case 2 (iii) Case 3 (iv) Case 4

(v) Case 5 (vi) Case 6 (vii) Case 7

Figure 4.9: All possible types of collisions. A dotted line denotes an online query. A solid line denotes a offline query.

Proof for Claim 30. We only prove for type 6 collisions. Other five cases are easier and similar.

When both (170) and (171) are offline queries, it is Case 7. If only one of the two edges is offline, it is Case 6. If they are all online queries, we can reduce it to Case 2. □

Finally, we show that for each case in Claim 30, the advantage is bounded by $(98(ST + T^2)\log^2 N)/N$.

Case 1. By making T new queries, each query (a, m) has $1/N$ chance to satisfy $H(a, m) = a$. Therefore, the probability is bounded by T/N.

Case 2. The probability of finding a collision among these T new queries is smaller than T^2/N, by birthday bound.

Case 3. Recall $\overline{\mathbf{E}_3^i}$: there are at most $10i \log N$ salts that has a self loop in the offline queries. By making T new queries, each query (a, m) has $(10i \log N)/N$ chance to hit any of these salts. Therefore, the probability is bounded by $(10iT \log N)/N$.

Case 4. Recall $\overline{\mathbf{E}_1^i}$: there are at most $10i \log N$ salts that has a collision starting from it in the offline queries. By making T new queries, each query (a, m) has $(10i \log N)/N$ chance to hit any of these salts. Therefore, the probability is bounded by $(10iT \log N)/N$.

Case 5. and **Case 6**. The proofs are identical. Fixing any offline query (a'', m''), by making T queries, the chance of hitting both ends is T^2/N^2. This is because we can enumerate which are the first queries that hit the starting salt a'' and the end $H(a'', m'')$. Each case happens w.p. at most $1/N^2$.

Since there are total $(i-1)T$ offline queries, by union bound, the advantage is at most $(i-1)T \cdot T^3/N^2 \le \frac{iT}{N} \cdot \frac{T^2}{N}$ for both cases.

Case 7. Recall $\overline{\mathbf{E}_2^i}$: there are at most $10i^2 \log^3 N$ pair-wise collisions. For every such collision that start with different salts, the probability of hitting both salts within T queries is T^2/N^2. This is due to the same counting argument in the analysis of Case 5 and Case 6.

By union bound, the advantage is at most $(10i^2T^2 \log^3 N)/N^2$.

We have shown all the cases in Claim 30. Therefore,

$$\Pr[\mathbf{X}_i|\mathbf{X}_{<i} \wedge \overline{\mathbf{E}_1^i} \wedge \overline{\mathbf{E}_2^i} \wedge \overline{\mathbf{E}_3^i}] \leq \frac{98(iT + T^2) \log^2 N}{N}.$$

Combining with eq:mis$_p$*roductandeq* : *mis$_c$ondition*, *weconcludelemma* : *micr$_t$wo* :$\Pr[\mathbf{X}_1 \wedge \ldots \wedge \mathbf{X}_S] \leq (\delta_S)^S$. □

4.2.1 *Bounding* $\mathbf{E}_1^i, \mathbf{E}_2^i, \mathbf{E}_3^i$

Without loss of generality, we assume the algorithm does not make duplicate queries since it can record every query it makes. We also assume an algorithm makes iT queries instead of $(i-1)T$ queries, for the convenience of presentation.

We first show Claim 29 for $\mathbf{E}_3^i$, which is the easiest one.

Proof of Claim 29. Let $\mathbf{B}_j$ be the indicator random variable, denote that the j-th query gives a self loop. Since each output of the random oracle is freshly sampled, it is clearly to see that $\{\mathbf{B}_j\}$ are independent. For every $j \in [iT]$, $\mathbb{E}[\mathbf{B}_j] = 1/N$. By Chernoff bound (see Preliminary), setting $\delta = (9N \log N)/T, \mu = iT/N$,

$$\Pr\left[\mathbf{B}_1 + \mathbf{B}_2 + \cdots + \mathbf{B}_{iT} \geq 10i \log N\right] \leq \exp(-\delta^2\mu/(2+\delta)) \leq \exp(-4i \log N).$$

□

Then we show Claim 27 for $\mathbf{E}_1^i$.

Proof for Claim 27. Let $\mathbf{a}_j$ be the j-th distinct salt where an algorithm finds a collision on. If the algorithm only finds collisions for fewer than j salts, $\mathbf{a}_j$ is defined as $\perp$. Then the probability of finding collisions for at least $t = 10i \log N$ salts is $\Pr\left[\forall j \in [t], \mathbf{a}_j \neq \perp\right]$.

Let $\mathbf{Z}_j$ be the number of queries that are already made towards salt $\mathbf{a}_j$; if $\mathbf{a}_j = \perp$, we define $\mathbf{Z}_j = 0$. We know that $\mathbf{Z}_1 + \cdots + \mathbf{Z}_t \leq iT$, since $\mathbf{a}_j$ are pairwise different.

For every $z_1, \cdots, z_t > 0$ and $z_1 + \cdots + z_t \leq iT$, the following probability denotes the event that collisions are found for at least t salts, and for the j-th collision, it happens at the z_j-th queries for the salt $\mathbf{a}_j$:

$$\Pr\left[\forall j \in [t], \mathbf{a}_j \neq \perp \wedge \mathbf{Z}_j = z_j\right] \leq \prod_{i=j}^{t} \frac{z_j}{N} \leq \left(\frac{iT}{tN}\right)^t. \tag{4.4}$$

The first inequality is due to the fact that for every $j \in [t]$, the image of the z_j-th query should match the one of the images among the first $z_j - 1$ queries made towards

a_j. For each $j \in [t]$, the probability is at most $(z_j - 1)/N$. The last inequality follows from the fact $z_1 + \cdots + z_t \le iT$.

By union bound, we have:

$$\Pr[\forall j \in [t], \mathbf{a}_j \neq \perp] \le \sum_{\substack{z_1,\cdots,z_t>0 \\ z_1+\cdots+z_t \le iT}} \Pr[\forall j \in [t], \mathbf{a}_j \neq \perp \wedge \mathbf{Z}_j = z_j]$$

$$\le \sum_{\substack{z_1,\cdots,z_t>0 \\ z_1+\cdots+z_t \le iT}} \left(\frac{iT}{tN}\right)^t.$$

The last inequality follows eq:bound$_e ach_q ueryassign$.

Because $\sum_{\substack{z_1,\cdots,z_t>0 \\ z_1+\cdots+z_t \le iT}} 1 \le \binom{2iT}{t}$, assuming $T^2 < N/2$, the above probability is then bounded by

$$\binom{2iT}{t}\left(\frac{iT}{tN}\right)^t \le \left(\frac{2ei^2T^2}{100i^2\log^2 N \cdot N}\right)^{10i\log N} < 2^{-10i\log N}.$$

□

Finally, we prove Claim 28 for $\mathbf{E}_2^i$.

Proof for Claim 28. We first notice that adaptive queries will not be more useful than non-adaptive queries. This is simply because when every query is a new query (never queried before), its image is uniform at random (assuming the random oracle is lazily sampled). Thus, let $\mathbf{Y}_j$ be the random variable for the image of the j-th query, $j \in [iT]$. We know that: (1). $\mathbf{Y}_j$ is a uniform random variable in $[N]$; (2). $\{\mathbf{Y}_j\}$ are independent.

To prove the claim, it is equivalent to show:

$$\Pr\left[\sum_{j<k} \mathbf{1}_{\mathbf{Y}_j=\mathbf{Y}_k} \ge 10i^2\log^3 N\right] \le 2\exp(-2i\log N).$$

For every image $w \in [N]$, let $\mathbf{Z}_w$ denote the number of images among all queries that are equal to w. Then we have $\sum_{j<k} \mathbf{1}_{\mathbf{Y}_j=\mathbf{Y}_k} = \sum_{w\in[N]} \binom{\mathbf{Z}_w}{2}$. This is because if there are $\mathbf{Z}_w$ queries that have image w, every pair of the queries will contribute one to the sum $\sum_{j<k} \mathbf{1}_{\mathbf{Y}_j=\mathbf{Y}_k}$. For the sake of convenience, we say a pair of collision belong to a claw of size ℓ if their image w satisfies that $\mathbf{Z}_w = \ell$, similar to def:claw.

We define the following 3 events:

- Event $\mathbf{F}_1^i$: at least $2i^2\log^3 N$ pairs of collisions belong to claws of size in $[2, \log N)$.
- Event $\mathbf{F}_2^i$: at least $2i^2\log^2 N$ pairs of collisions belong to claws of size in $[\log N, i\log N)$.
- Event $\mathbf{F}_3^i$: at least $2i^2\log^2 N$ pairs of collisions belong to claws of size at least $i\log N$.

Note that the only event we have a $\log^3 N$ factor in the number of pairs of collisions is $\mathbf{F}_1^i$.

Claim 31.

$$\Pr[\mathbf{E}_2^i] \leq \Pr[\mathbf{F}_1^i] + \Pr[\mathbf{F}_2^i] + \Pr[\mathbf{F}_3^i].$$

Proof. For the event $\mathbf{E}_2^i$ to occur, at least one of the events $\mathbf{F}_1^i, \mathbf{F}_2^i, \mathbf{F}_3^i$ has to happen. Therefore,

$$\Pr[\mathbf{E}_2^i] \leq \Pr[\mathbf{E}_2^i \cap \mathbf{F}_1^i] + \Pr[\mathbf{E}_2^i \cap \mathbf{F}_2^i] + \Pr[\mathbf{E}_2^i \cap \mathbf{F}_3^i].$$

It implies the claim as for any $j \in \{1, 2, 3\}$, $\Pr[\mathbf{E}_2^i \cap \mathbf{F}_j^i] \leq \Pr[\mathbf{F}_j^i]$. □

Thus, in order to bound $\Pr[\mathbf{E}_2^i]$, it is sufficient to bound the probability of events $\Pr[\mathbf{F}_1^i], \Pr[\mathbf{F}_2^i], \Pr[\mathbf{F}_3^i]$.

$\mathbf{F}_1^i$. We then apply counting arguments for bounding all the probabilities. If $2i^2 \log^3 N$ pairs of collisions have to be obtained from claws of size at most $\log N$, it implies that at least $t = 2i^2 \log N$ such claws have to be found. Therefore,

$$\begin{aligned}\Pr[\mathbf{F}_1^i] &\leq \Pr[\text{finding } t \text{ claws of size } \leq \log N \text{ in } iT \text{ queries}] \\ &\leq \frac{\binom{iT}{t} \cdot \binom{iT}{t} \cdot (t!)}{N^t} < \left(\frac{T^2}{N}\right)^t.\end{aligned}$$

The counting argument works in the following way: we enumerate which pairs of $\mathbf{Y}_j, \mathbf{Y}_k$ will collide, and they pairwise collide with probability $1/N^t$. When $T^2 \leq N/2$, it is at most N^{-2i}.

$\mathbf{F}_3^i$. Before bounding the probability of event $\mathbf{F}_2^i$, we will bound the probability of event $\mathbf{F}_3^i$ first.

$$\begin{aligned}\Pr[\mathbf{F}_3^i] &\leq \Pr[\text{finding 1 claw of size } i \log N \text{ in } iT \text{ queries}] \\ &= \frac{\binom{iT}{i \log N}}{N^{i \log N - 1}} \leq \frac{(\frac{eiT}{i \log N})^{i \log N}}{N^{i \log N}} \cdot N \\ &\leq \left(\frac{T}{N}\right)^{i \log N} \cdot \left(\frac{1}{N^i}\right) \cdot N \leq \left(\frac{T}{N}\right)^{i \log N},\end{aligned}$$

where the second last inequality is obtained using $\log N \geq 2$. In the counting argument, we enumerate which $i \log N$ queries have the same image and they collide with probability $N^{i \log N - 1}$. This is at most $2^{-2i \log N} = N^{-2i}$ when $T \leq \sqrt{N}$ and $\log N \geq 2e$.

$\mathbf{F}_2^i$. Finally, we look at event $\mathbf{F}_2^i$. Assume for some $k \in [\log N, i \log N)$ there exists j claws of size *exact* k such that they make $2i^2 \log^2 N$ pairs of collisions. Then

$$j \cdot \binom{k}{2} \geq 2i^2 \log^2 N \quad \Rightarrow \quad j \geq \frac{2i^2 \log^2 N}{\binom{k}{2}} \geq \frac{2i \log N}{k}.$$

For any $k \in [\log N, i \log N)$ the probability of finding $\frac{2i \log N}{k}$ claws each of size k in iT queries is

$$\left[\frac{\binom{iT}{k}}{N^{k-1}}\right]^{2i \log N/k} \leq \left[\left(\frac{\frac{eiT}{k}}{N}\right)^k \cdot N\right]^{2i \log N/k} \leq 2\left(\frac{iT}{2N}\right)^{2i \log N},$$

where the last inequality holds using $k \geq \log N \geq 2e$.

Then following union bound, the probability that there exists some $k \in [\log N, i \log N)$ such that $\frac{2i \log N}{k}$ claws each of size k can be found in iT queries is at most

$$2i \log N \cdot \left(\frac{iT}{2N}\right)^{2i \log N} \leq 2\left(\frac{iT}{N}\right)^{2i \log N}, \tag{4.5}$$

using $x \leq 2^x$ for all x.

Let S_k denote the number of claws of size k found in iT queries. Then the number of pairs of collisions found for $k \in [\log N, i \log N)$ is

$$\sum_{k=\log N}^{i \log N} S_k \cdot \binom{k}{2} = \sum_{k=\log N}^{i \log N} S_k \cdot \left(\sum_{\ell=1}^{k} \ell\right) = \sum_{k=\log N}^{i \log N} S_k \cdot \left(\sum_{\ell=1}^{\log N} \ell\right) + \sum_{k=\log N}^{i \log N} S_k \cdot \left(\sum_{\ell=\log N}^{k} \ell\right)$$

$$\leq \log^2 N \sum_{k=\log N}^{i \log N} S_k + \sum_{\ell=\log N}^{i \log N} \ell \cdot \left(\sum_{k=\ell}^{i \log N} S_k\right),$$

where $\sum_{k=\ell}^{i \log N} S_k$ is the number of claws of size at least ℓ. Note that any claw of size $(\ell + x)$ for $x \geq 0$ contains a claw of size ℓ. Thus, $\sum_{k=\ell}^{i \log N} S_k$ can be bounded by $2i \log N/\ell$ with probability at least $1 - 2\left(\frac{iT}{N}\right)^{2i \log N}$, by eq:exist$_e nough_c law$.

Then, with probability at least $1 - 2\left(\frac{iT}{N}\right)^{2i \log N}$, the number of pairs of collisions found from claws of size $k \in [\log N, i \log N)$ in iT queries is

$$\leq \log^2 N \sum_{k=\log N}^{i \log N} S_k + \sum_{\ell=\log N}^{i \log N} \ell \cdot \left(\sum_{k=\ell}^{i \log N} S_k\right)$$

$$\leq \log^2 N \cdot \frac{2i \log N}{\log N} + \sum_{\ell=\log N}^{i \log N} \ell \cdot \frac{2i \log N}{\ell} \leq 2i \log^2 N + 2i^2 \log^2 N \leq 4i^2 \log^2 N.$$

Thus assuming $iT < N/2$,

$$\Pr[\mathbf{F}_3^i] \leq 2\left(\frac{iT}{N}\right)^{2i \log N} < 2N^{-2i}.$$

Putting together the above results we obtain $\Pr[\mathbf{E}_2^i] < 4N^{-2i}$. □

4.3 Auxiliary Input Collision Resistance for B Merkle-Damgård

In this section we prove the following theorem.

Theorem 32. *For any functions S, T, B, and $N \geq 64$*

$$\mathsf{Adv}^{\text{AI-CR}}_{\text{B-MD}}(S,T) \leq (34\log^2 N) \cdot \frac{STB}{N} \cdot \max\left\{1, \frac{ST^2}{N}\right\} + 2 \cdot \frac{T^2}{N} .$$

Lemma 33. *For any functions S, T, B, and $N \geq 64$,*

$$\mathsf{Adv}^{\text{MI-CR}}_{\text{B-MD}}(S,T) \leq \left(\frac{17\kappa TB\log^2 N + T^2}{N}\right)^S$$

where $\kappa = S \cdot \max\{1, ST^2/N\}$.

As for the case of $B = 2$, we prove an upper bound on the advantage of B-block collision finding adversary in the MI-CR model, which implies an upper bound in the AI-CR model via thm:mi$_t o_a i$.

Proof of thm:micr$_b$ We prove this lemma in similar fashion as lemma:micr$_t$*wo.Let*H*bearandomora* and $\mathcal{A}$ be any (S,T)-MI adversary.

We analogously define $\mathbf{X}_i$ to be the indicator variable that $\mathcal{A}$ finds at most B-length collisions on uniformly random salt a_i given as input in the i-th stage of the game. We also define $\mathbf{X}_{<i} = \mathbf{X}_1 \wedge \cdots \wedge \mathbf{X}_{i-1}$. So, the advantage of $\mathcal{A}$ is

$$\Pr[\mathbf{X}_1 \wedge \ldots \wedge \mathbf{X}_S] = \prod_{i=1}^{S} \Pr[\mathbf{X}_i | \mathbf{X}_{<i}].$$

As in the proof for $B = 2$ case, we will inductively bound $\Pr[\mathbf{X}_{<i+1}]$ for each $i \in [S]$. Here we will bound $\Pr[\mathbf{X}_{<i+1}]$ to $((17\kappa_i TB\log^2 N + T^2)/N)^i$ where $\kappa_i = i \cdot \max\{1, iT^2/N\}$. Recall that we will analogously assume $\Pr[\mathbf{X}_{<i}] \geq ((17\kappa_i TB\log^2 N + T^2)/N)^i$. Otherwise $\Pr[\mathbf{X}_{<i+1}] \leq ((17\kappa_i TB\log^2 N + T^2)/N)^i$ holds trivially.

In order to prove the lemma, it suffices to upper bound $\Pr[\mathbf{X}_i|\mathbf{X}_{<i}]$ by $17\kappa_i TB\log^2 N/N + T^2/N$ for any arbitrary $i \in [S]$. That is because $\Pr[\mathbf{X}_{<i+1}] = \Pr[\mathbf{X}_i|\mathbf{X}_{<i}] \cdot \Pr[\mathbf{X}_{<i}]$ where $\Pr[\mathbf{X}_{<i}] \leq ((17\kappa_i TB\log^2 N + T^2)/N)^{i-1}$ by the inductive hypothesis. In the proof, we will handle the conditioning on $\mathbf{X}_{<i}$ in a similar fashion to our proof for $B = 2$ case.

First we state some useful definitions.

Definition 24. *A list of elements $(a_1, m_1), \ldots, (a_\ell, m_\ell)$ in $[N] \times [M]$ are said to form a chain for H when for every $j \in [\ell - 1]$, $H(a_j, m_j) = a_{j+1}$.*
A chain $(a_1, m_1), \ldots, (a_\ell, m_\ell)$ for H is called a cycle when $H(a_\ell, m_\ell) = a_1$. The length of a cycle is the number of elements in it, ℓ here.

Definition 25. *Two distinct chains $(a_1, m_1), \ldots, (a_\ell, m_\ell)$ and $(a'_1, m'_1), \ldots, (a'_{\ell'}, m'_{\ell'})$ are called colliding chains for H if $H(a_\ell, m_\ell) = H(a'_{\ell'}, m'_{\ell'})$.*

Definition 26. *For any $a \in [N]$, a set of elements $(a_1, m_1), \ldots, (a_\ell, m_\ell)$ in $[N] \times [M]$ are said to form a claw at a under H if $\ell > 1$, $a_1, \ldots, a_\ell$ are distinct and $H(a_1, m_1) = \ldots = H(a_\ell, m_\ell) = a$. We refer to $a_1, \ldots, a_\ell$ as the pre-images of a.*

Next, we define events to illustrate the bound on 'useful' information gained by $\mathcal{A}$ from the prior iterations in the B-MICR game. Each of these events are defined over responses from the random oracle in the first $(i-1)$ iterations.

- Let Y be the set of salts with more than one pre-image on it in the offline database. Then we define $\mathbf{E}_2^i$ to be the event that $\sum_{a \in Y}$ (# pre-images on a) $\geq 16\kappa_i \log^2 N$ after $(i-1)T$ queries where $\kappa_i = \max\left\{i, \frac{i^2T^2}{N}\right\}$.
- Let $\mathbf{E}_3^i$ be the event that there exists at least $i \log N$ 'special' cycles of length in $[B-1]$ among the $(i-1)T$ offline queries. A cycle $(a_1, m_1), \ldots, (a_\ell, m_\ell)$ is called 'special' if the number of pre-images on a_i is exactly 1 for every $i \in [\ell]$.

Next, we can write

$$\begin{aligned}
\Pr[\mathbf{X}_i|\mathbf{X}_{<i}] &= \Pr[\mathbf{X}_i|\mathbf{X}_{<i} \wedge \overline{\mathbf{E}_2^i} \wedge \overline{\mathbf{E}_3^i}] + \Pr[\mathbf{E}_2^i \vee \mathbf{E}_3^i|\mathbf{X}_{<i}] \\
&\leq \Pr[\mathbf{X}_i|\mathbf{X}_{<i} \wedge \overline{\mathbf{E}_2^i} \wedge \overline{\mathbf{E}_3^i}] + \frac{\Pr[\mathbf{E}_2^i]}{\Pr[\mathbf{X}_{<i}]} + \frac{\Pr[\mathbf{E}_3^i]}{\Pr[\mathbf{X}_{<i}]} \\
&\leq \Pr[\mathbf{X}_i|\mathbf{X}_{<i} \wedge \overline{\mathbf{E}_2^i} \wedge \overline{\mathbf{E}_3^i}] + \frac{1}{N}
\end{aligned}$$

where the last inequality holds via Claim 34, Claim 35 (which are stated next) and our assumption that $\Pr[\mathbf{X}_{<i}] \geq ((17\kappa_i TB \log^2 N + T^2)/N)^i$.

Claim 34. *For any $i \in [S]$, $iT + T^2 < N/2$, $2i \log N + 1 \leq N/2$ and $N \geq 64$, $\Pr[\mathbf{E}_2^i] \leq \frac{5}{N^{2i}}$.*

Claim 35. *For any $i \in [S]$, $\Pr[\mathbf{E}_3^i] \leq \left(\frac{T}{N}\right)^{i \log N}$.*

We will prove Claim 34 and 35 later. As before, we will prove Claim 34 and 35 in the full version of the paper. Readers may safely skip the proofs and assume these "knowledge-gaining events" happen with exponentially small probability.

Next, we want to study $\Pr[\mathbf{X}_i|\mathbf{X}_{<i} \wedge \overline{\mathbf{E}_2^i} \wedge \overline{\mathbf{E}_3^i}]$. We define $\mathbf{G}$ to be the event that input salt a_i has been queried among the previous $(i-1)$ iterations or that input salt a_i is the output of some query among the previous $(i-1)$ iterations. So, we can rewrite $\Pr[\mathbf{X}_i|\mathbf{X}_{<i} \wedge \overline{\mathbf{E}_2^i} \wedge \overline{\mathbf{E}_3^i}]$ as follows:

$$\begin{aligned}
\Pr[\mathbf{X}_i|\mathbf{X}_{<i} \wedge \overline{\mathbf{E}_2^i} \wedge \overline{\mathbf{E}_3^i}] &\leq \Pr[\mathbf{X}_i|\mathbf{X}_{<i} \wedge \overline{\mathbf{E}_2^i} \wedge \overline{\mathbf{E}_3^i} \wedge \overline{\mathbf{G}}] + \Pr\left[\mathbf{G}\middle|\mathbf{X}_{<i} \wedge \overline{\mathbf{E}_2^i} \wedge \overline{\mathbf{E}_3^i}\right] \\
&\leq \Pr[\mathbf{X}_i|\mathbf{X}_{<i} \wedge \overline{\mathbf{E}_2^i} \wedge \overline{\mathbf{E}_3^i} \wedge \overline{\mathbf{G}}] + \frac{2(i-1)T}{N}.
\end{aligned}$$

Note that a_i is chosen uniformly and independently and as queries in the previous iterations could be made on at most $(i-1)T$ distinct salts and can output at most $(i-1)T$ distinct salts in the previous $(i-1)$ iterations, it is easy to bound

$$\Pr\left[\mathbf{G}\middle|\mathbf{X}_{<i} \wedge \overline{\mathbf{E}_2^i} \wedge \overline{\mathbf{E}_3^i}\right] \leq \frac{2(i-1)T}{N}.$$

Finally, we analyze $\Pr[\mathbf{X}_i|\mathbf{X}_{<i} \wedge \overline{\mathbf{E}_2^i} \wedge \overline{\mathbf{E}_3^i} \wedge \overline{\mathbf{G}}]$.

Claim 36. *For any any $i \in [S]$,*

$$\Pr[\mathbf{X}_i|\mathbf{X}_{<i} \wedge \overline{\mathbf{E}_2^i} \wedge \overline{\mathbf{E}_3^i} \wedge \overline{\mathbf{G}}] \leq \frac{16\kappa_i TB\log^2 N + T^2}{N}.$$

Proof of claim 36 requires different analysis for different types of colliding chains which we show in subsection 4.3.1. Before we move onto that subsection, we first show how we obtain the lemma by putting together all the claims.

$$\begin{aligned}
\Pr[\mathbf{X}_i|\mathbf{X}_{<i}] &\leq \Pr[\mathbf{X}_i|\mathbf{X}_{<i} \wedge \overline{\mathbf{E}_2^i} \wedge \overline{\mathbf{E}_3^i} \wedge \overline{\mathbf{G}}] + \Pr\left[\mathbf{G}\middle|\mathbf{X}_{<i} \wedge \overline{\mathbf{E}_2^i} \wedge \overline{\mathbf{E}_3^i}\right] + \Pr[\mathbf{E}_2^i \vee \mathbf{E}_3^i|\mathbf{X}_{<i}] \\
&\leq \frac{16\kappa_i TB\log^2 N + T^2}{N} + \frac{2(i-1)T}{N} + \frac{1}{N} \\
&\leq \frac{17\kappa_i TB\log^2 N}{N} + \frac{T^2}{N}
\end{aligned}$$

where the last inequality holds from that $\kappa_i = \max\{i, i^2T^2/N\}$ and $N \geq 4$.

4.3.1 *Proof of Claim 36*

To this end, we state the following claim.

Claim 37. *For any $i \in [S]$, to find a B-length collision on a_i, the queries in the database should satisfy at least one of the following conditions given there exists no query in the offline database that takes a_i as input or outputs a_i:*

1. *There exists an online query (i.e., a query among at most T queries that were made for the first time in the i-th iteration after receiving the challenge input a_i), denoted (a, m) such that $H(a, m) = a_i$.*
2. *There exists two distinct online queries, denoted (a, m) and (a', m') such that $H(a, m) = H(a', m')$.*

   This includes both of the following possibilities: the online queries are such (1) $a = a'$ (and thus m and m' will be distinct); (2) $a \neq a'$.

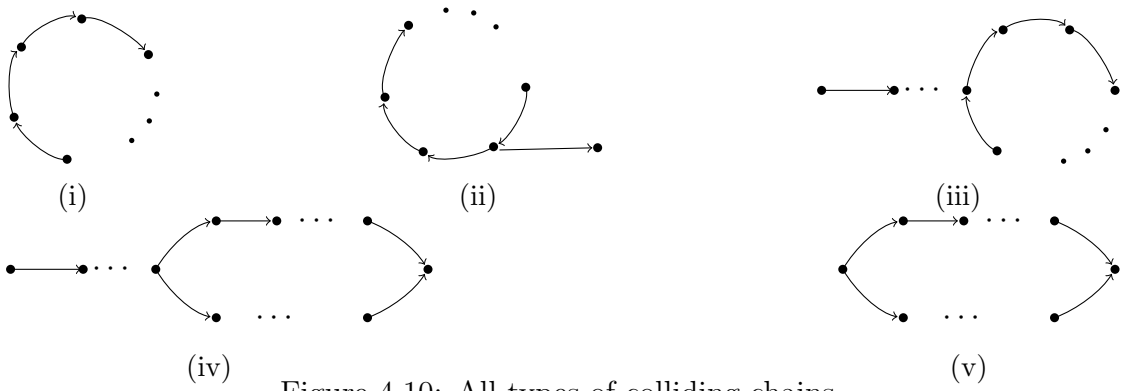


Figure 4.10: All types of colliding chains

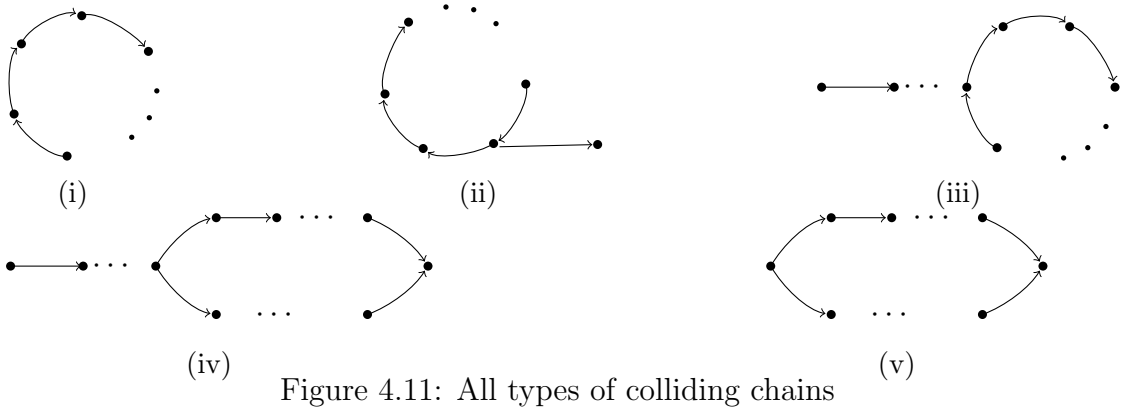


Figure 4.11: All types of colliding chains

3. *There exists an online query, denoted* (a, m)*, a chain (recall definition 24) of offline queries*[1]*, denoted* $(b_1, m_1), \ldots, (b_\ell, m_\ell)$ *for some* $0 < \ell < B$ *, and an offline query* $(b, m') \neq (b_\ell, m_\ell)$ *such that* $H(a, m) = b_1$*,* $H(b, m') = H(b_\ell, m_\ell)$ *and the number of pre-images for every salt in* $\{b_2, \ldots, b_\ell\}$ *in the offline database is exactly* 1*.*

4. *There exists two online queries, denoted* (a, m) *and* (a', m')*, and a chain of offline queries, denoted* $(b_1, m_1), \ldots, (b_\ell, m_\ell)$ *for some* $\ell < B$*, such that* $H(a, m) = b_1$*,* $H(a', m') = H(b_\ell, m_\ell)$ *and the number of pre-images on every salt in* $\{b_2, \ldots, b_\ell\}$ *in the offline database is exactly* 1*.*

5. *There exists an online query, denoted* (a, m)*, and a cycle in the offline database, denoted* $(b_1, m_1), \ldots, (b_\ell, m_\ell)$ *for some* $\ell < B$*, such that* $H(a, m) = b_1$ *and the number of pre-images on every salt in* $\{b_1, b_2, \ldots, b_\ell\}$ *in the offline database is exactly* 1*.*

Proof for Claim 37. Fig. 4.11 enumerates all the possible types of colliding chains. Depending on where the queries in the chains are first made for each of the types,

[1]The set of **Offline queries** is the set of distinct queries made in the previous $(i-1)$ iterations. So there are at most $(i-1)T$ of these queries and their outputs are independent and uniformly distributed. The set of **Online queries** is the set of distinct queries made in the i-th iteration after receiving the challenge input a_i that had not been made in any of the previous $(i-1)$ iterations. Note that the outputs of online queries are also independent and uniformly distributed.

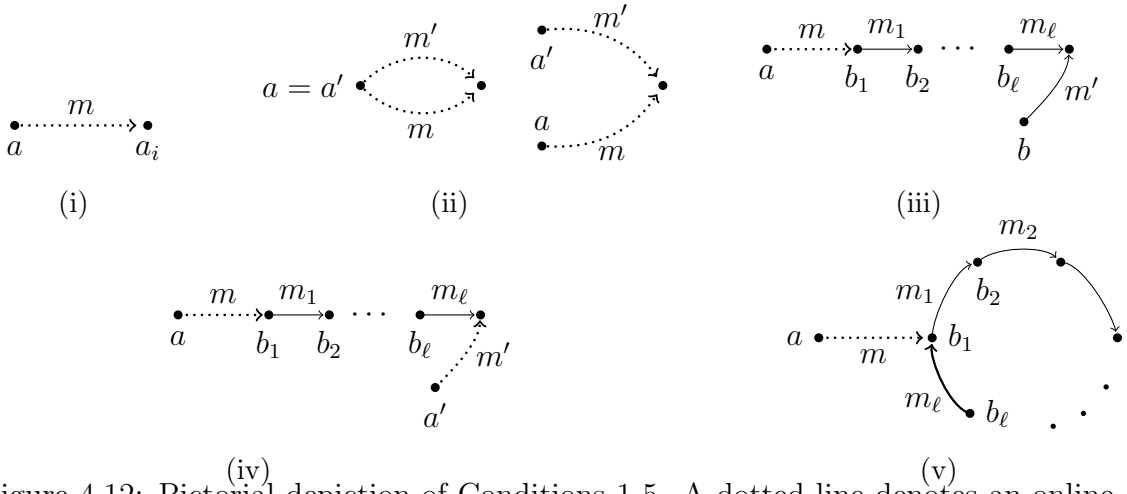


Figure 4.12: Pictorial depiction of Conditions 1-5. A dotted line denotes an online query. A solid line denotes an offline query.

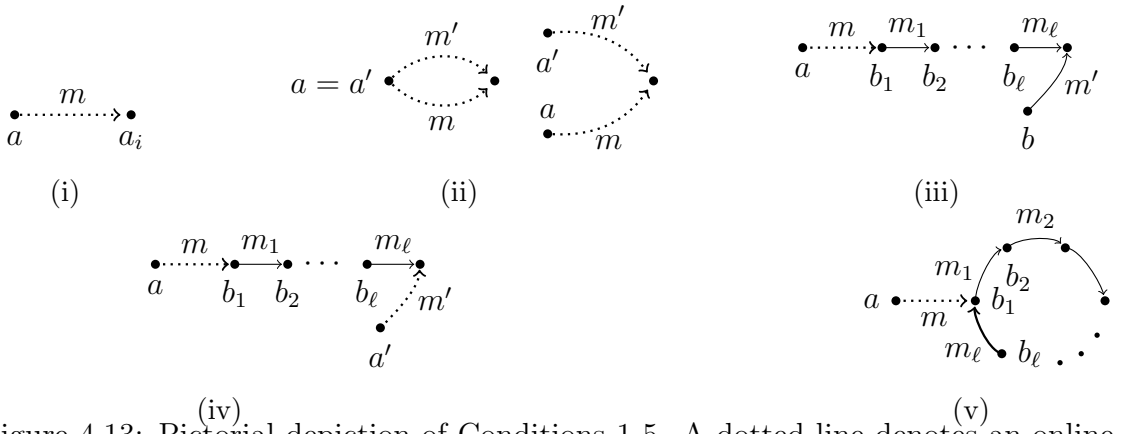


Figure 4.13: Pictorial depiction of Conditions 1-5. A dotted line denotes an online query. A solid line denotes an offline query.

we show that the list of conditions in the claim is complete. (Refer to fig. 4.13 for a visual representation of the conditions in the claim.)

We know that all the queries with output a_i or of the form $(a_i, \cdot)$ in the colliding chains are online queries. This implies if the colliding chains are of the types in fig. 4.11i or 4.11ii, the queries in the database will satisfy condition 1.

For the remaining types of colliding chains (ref fig. 4.11iii,4.11iv,4.11v), one of the following 3 cases can happen:

1. **Both the 'colliding' queries are online** In this case, the queries in the database will satisfy condition 2.

2. **Both the 'colliding' queries are offline** In this case, the queries in the database will satisfy condition 3. Note that b_ℓ can be thought of as the earliest query among the chains that has more than one pre-image in the offline database.

3. **One of the 'colliding' queries is offline and online each** For the colliding

(i) (ii)

Figure 4.14: A dotted line denotes an online query. A solid line denotes an offline query.

(i) (ii)

Figure 4.15: A dotted line denotes an online query. A solid line denotes an offline query.

chains of types in fig. 4.11iv and 4.11v), the queries in the database will satisfy condition 4. For the colliding chains of type in 4.11iii, there are two possibilities as shown in Fig. 4.15. For the possibility in fig. 4.15i, the queries in the database satisfy condition 4. On the other hand, for the possibility in fig. 4.15ii, the queries in the database satisfy condition 5.

□

Claim 38. *For $j \in [5]$, let ϵ_j be the advantage in achieving condition j from claim 37 when $\overline{\mathbf{E}_2^i}$, $\overline{\mathbf{E}_3^i}$ and $\overline{\mathbf{G}}$ hold. Then for any $i \in [S]$, the results summarized in Table 4.2 on the upper bounds of ϵ_j hold.*

We prove the bounds stated in Claim 38 next.

Condition 1. Recall that online queries are 'new' queries, as in they are made for the first time among the T queries in the i-th iteration after receiving a_i. Thus, the output of online queries is independent of output from offline queries and has $1/N$ chance to be a_i under H via lazy sampling. By taking a union bound over at most T online queries, we can bound the probability to T/N.

Condition 2. By birthday bound, it holds that the probability of finding 'colliding' queries among T online queries is at most T^2/N.

Condition 3. Given $\overline{\mathbf{E}_2^i}$ implies that there can be at most $16\kappa_i \log^2 N$ queries in the offline database that are part of some claw. As per the definition of condition 4, there will be a unique chain of length $< B$ in the offline database ending in each of these at most $16\kappa_i \log^2 N$ queries, such that an online query hits the start of this chain. The probability of hitting one of these at most $B \cdot 16\kappa_i \log^2 N$ salts within T queries is at most $16\kappa_i TB \log^2 N/N$.

Condition 4. As per the definition of condition 5, there can be at most iT such chains of length $< B$ in the offline database, such that an online query hits the start

of this chain and another online hits the end of this chain. The probability of hitting both the salts within at most T queries is bounded by T^2/N^2. By union bound the advantage is at most iT^3/N^2.

Condition 5. Given $\overline{\mathbf{E}_3^i}$ implies there are at most $i \log N$ 'special' cycles in the offline database, each with at most B queries in it. So, there are at most $iB \log N$ queries in these cycles and the probability of hitting one of the starting salts of these queries within T online queries is bounded by $iB \log N \cdot T/N$.

From Claim 38 it holds that the advantage of achieving any of the conditions in Claim 37 given $\overline{\mathbf{E}_2^i}$, $\overline{\mathbf{E}_3^i}$ and $\overline{\mathbf{G}}$ is bounded by $(16\kappa_i TB \log^2 N + T^2)/N$. Note that for $i \leq S$, when $ST^2 < N$ implies $iT^2 < N$. Hence $\kappa_i = i$ if $\kappa_S = S$.

Finally to complete this proof, we prove our Claim 34 and 35 next.

4.3.2 *Proof of Claim 34*

We first note that proof of claim 34 is similar to the proof of claim 28 in essence. We again use that adaptive queries will not be more useful than non-adaptive queries because output of every new query (never queried before) is uniform at random (assuming the random oracle is lazily sampled).

For every $a \in [N]$, let $\mathbf{Z}_a$ denote the number of pre-images of a. Then proving Claim 34 is equivalent to showing

$$\Pr\left[\sum_{a\in[N];\mathbf{Z}_a\neq 1} \mathbf{Z}_a \geq \max\{16i \log^2 N, 16i^2T^2 \log^2 N/N\}\right] \leq 2\exp(-2i \log N).$$

We will separate the salts into 3 buckets depending on the number of their pre-images (in the offline database) and analyze the sum of number of pre-images separately for each bucket. Let's define the buckets:

- $\mathsf{Bucket}_1 := \{a | \mathbf{Z}_a \in [2, \log N)\}$
- $\mathsf{Bucket}_2 := \{a | \mathbf{Z}_a \in [\log N, i \log N)\}$
- $\mathsf{Bucket}_3 := \{a | \mathbf{Z}_a \geq i \log N\}$

So for $\sum_{a\in[N];\mathbf{Z}_a\neq 1} \mathbf{Z}_a$ to exceed $\max\{16i \log^2 N, 16i^2T^2 \log^2 N/N\}$, the sum of number of pre-images of salts in at least one of the buckets has to exceed $\max\{4i \log^2 N, 4i^2T^2 \log^2 N$ We show that this happens with exponentially small chance.

In order to do that we define the following 3 events:

- Event $\mathbf{F}_1$: $\sum_{a\in\mathsf{Bucket}_1} \mathbf{Z}_a \geq 4i \log^2 N \cdot \max\{1, iT^2/N\}$.
- Event $\mathbf{F}_2$: $\sum_{a\in\mathsf{Bucket}_2} \mathbf{Z}_a \geq 4i \log^2 N \cdot \max\{1, iT^2/N\}$.
- Event $\mathbf{F}_3$: $\sum_{a\in\mathsf{Bucket}_3} \mathbf{Z}_a \geq 2i \log N \cdot \max\{1, iT^2/N\}$.

In order to prove the claim, it is sufficient to bound the probability of events $\mathbf{F}_1, \mathbf{F}_2, \mathbf{F}_3$.

We begin with the easiest to analyze events, which is $\mathbf{F}_3$.

Bounding $\Pr[\mathbf{F}_3]$

For $\mathbf{F}_3$, we can actually obtain the following stronger statement:

$$\Pr\left[\sum_{a\in\mathsf{Bucket}_3}\mathbf{Z}_a \geq 2i\log N\right] \leq 2\exp\left(-2i\log N\right).$$

That is because

$$\begin{aligned}\Pr\left[\sum_{a\in\mathsf{Bucket}_3}\mathbf{Z}_a \geq 2i\log N\right] &\leq \Pr[\text{finding 1 claw of size } i\log N \text{ in } iT \text{ queries}]\\ &= \frac{\binom{iT}{i\log N}}{N^{i\log N-1}} \leq \frac{(\frac{eiT}{i\log N})^{i\log N}}{N^{i\log N}}\cdot N\\ &\leq \left(\frac{T}{N}\right)^{i\log N}\cdot\left(\frac{1}{N}\right)\cdot N \leq \left(\frac{T}{N}\right)^{i\log N},\end{aligned}$$

where the second last inequality is obtained using $\log N \geq 2$. In the counting argument, we enumerate which $i\log N$ queries have the same image and they collide with probability $N^{i\log N-1}$. $\left(\frac{T}{N}\right)^{i\log N}$ is at most $2^{-2i\log N} = N^{-2i}$ when $T \leq \sqrt{N}$ and $\log N \geq 2e$.

Bounding $\Pr[\mathbf{F}_2]$

Next, we prove bound for $\Pr[\mathbf{F}_2]$. Again we can show the following stronger statement:

$$\Pr\left[\sum_{a\in\mathsf{Bucket}_2}\mathbf{Z}_a \geq 4i\log^2 N\right] \leq 2\exp\left(-2i\log N\right).$$

Assume for some $k \in [\log N, i\log N)$ there exists j claws of size *exact* k such that the sum of the number of their pre-images is $2i\log N$. Then

$$j\cdot k \geq 2i\log N \quad\Rightarrow\quad j \geq \frac{2i\log N}{k}.$$

For any $k \in [\log N, i\log N)$ the probability of finding $\frac{2i\log N}{k}$ claws each of size k in iT queries is

$$\left[\frac{\binom{iT}{k}}{N^{k-1}}\right]^{2i\log N/k} \leq \left[\left(\frac{\frac{eiT}{k}}{N}\right)^{k}\cdot N\right]^{2i\log N/k} \leq 2\left(\frac{iT}{2N}\right)^{2i\log N},$$

where the last inequality holds using $k \geq \log N \geq 2e$.

Then taking a union bound, the probability that there exists some $k \in [\log N, i\log N)$ such that $\frac{2i\log N}{k}$ claws each of size k can be found in iT queries is at most

$$2i\log N\cdot\left(\frac{iT}{2N}\right)^{2i\log N} \leq 2\left(\frac{iT}{N}\right)^{2i\log N}, \tag{4.6}$$

using $x \leq 2^x$ for all x.

Let S_k denote the number of claws of size k found in iT queries. Then the sum of number of pre-images of salts in Bucket_2 is

$$\sum_{k=\log N}^{i \log N} S_k \cdot k = \sum_{k=\log N}^{i \log N} S_k \cdot \left(\sum_{\ell=1}^{k} 1 \right) = \sum_{k=\log N}^{i \log N} S_k \cdot \left(\sum_{\ell=1}^{\log N} 1 \right) + \sum_{k=\log N}^{i \log N} S_k \cdot \left(\sum_{\ell=\log N}^{k} 1 \right)$$
$$\leq \log N \sum_{k=\log N}^{i \log N} S_k + \sum_{\ell=\log N}^{i \log N} \left(\sum_{k=\ell}^{i \log N} S_k \right),$$

where $\sum_{k=\ell}^{i \log N} S_k$ is the number of claws of size at least ℓ. Note that any claw of size $(\ell + x)$ for $x \geq 0$ contains a claw of size ℓ. Thus, $\sum_{k=\ell}^{i \log N} S_k$ can be bounded by $2i \log N/\ell$ with probability at least $1 - 2\left(\frac{iT}{N}\right)^{2i \log N}$, by eq:claw.

Then, with probability at least $1 - 2\left(\frac{iT}{N}\right)^{2i \log N}$, the sum of number of pre-images of salts in Bucket_2 in iT queries is

$$\leq \log N \sum_{k=\log N}^{i \log N} S_k + \sum_{\ell=\log N}^{i \log N} \left(\sum_{k=\ell}^{i \log N} S_k \right)$$
$$\leq \log N \cdot \frac{2i \log N}{\log N} + \sum_{\ell=\log N}^{i \log N} \frac{2i \log N}{\ell} \leq 2i \log N + 2i \log^2 N \leq 4i \log^2 N$$

where the second-to-last inequality holds from the fact that the upper bound on the m-th harmonic series is $\log(m+1)$ and $2i \log N + 1 \leq N/2$.

Again using the assumption $iT < N/2$,

$$\left[\sum_{a \in \mathsf{Bucket}_2} \mathbf{Z}_a \geq 4i \log^2 N \right] \leq 2 \left(\frac{iT}{N} \right)^{2i \log N} < 2N^{-2i}.$$

Bounding $\Pr[\mathbf{F}_1]$

Finally we analyze the event $\mathbf{F}_1$.

For analyzing $\mathbf{F}_1$, first let's consider the case when $iT^2 \geq N$, i.e., we have to prove

$$\Pr \left[\sum_{a \in \mathsf{Bucket}_1} \mathbf{Z}_a \geq 4i^2 T^2 \log^2 N/N \right] \leq 2 \exp(-2i \log N).$$

Note that there have to be at least $4i^2T^2 \log N/N$ salts in Bucket_1 to make $\sum_{a \in \mathsf{Bucket}_1} \mathbf{Z}_a \geq 4i^2T^2 \log^2 N/N$. This means there should be at least $4i^2T^2 \log N/N$ claws of size 2. Then,

$$
\begin{aligned}
&\Pr\left[\sum_{a\in\mathsf{Bucket}_1}\mathbf{Z}_a \geq 4i^2T^2\log^2 N/N\right]\\
&\leq \Pr[\text{finding } 4i^2T^2\log N/N \text{ distinct claws of size 2 in } iT \text{ queries}]\\
&\leq \frac{\binom{iT}{4i^2T^2\log N/N}\cdot\binom{iT}{4i^2T^2\log N/N}\cdot(4i^2T^2\log N/N)!}{N^{4i^2T^2\log N/N}}\\
&\leq \left(\frac{e^2i^2T^2}{\frac{4i^2T^2\cdot\log N}{N}\cdot N}\right)^{4i^2T^2\log N/N}\\
&\leq \left(\frac{e^2}{4\log N}\right)^{4i\log N}
\end{aligned}
$$

where the last inequality holds because $iT^2 \geq N$.

Next, consider the case when $iT^2 < N$. So we have to show

$$
\Pr\left[\sum_{a\in\mathsf{Bucket}_1}\mathbf{Z}_a \geq 4i\log^2 N\right] \leq 2\exp\left(-2i\log N\right).
$$

Proceeding in a similar fashion as above,

$$
\begin{aligned}
\Pr\left[\sum_{a\in\mathsf{Bucket}_1}\mathbf{Z}_a \geq 4i\log^2 N\right] &\leq \Pr[\text{finding } 4i\log N \text{ distinct claws of size 2 in } iT \text{ queries}]\\
&\leq \frac{\binom{iT}{4i\log N}\cdot\binom{iT}{4i\log N}\cdot(4i\log N)!}{N^{4i\log N}}\\
&\leq \left(\frac{e^2i^2T^2}{4i\cdot\log N\cdot N}\right)^{4i\log N}\\
&\leq \left(\frac{e^2}{4\log N}\right)^{4i\log N}
\end{aligned}
$$

where the last inequality holds using $iT^2 < N$.

4.3.3 *Proof of Claim 35*

We prove the claim via compression. To that end, we use the following lemma from [19].

Lemma 39 ([19], restated in [7])**.** *For any pair of encoding and decoding algorithms, $(\mathcal{E}, \mathsf{Dec})$, where $\mathcal{E} : \{0,1\}^x \to \{0,1\}^y$ and $\mathsf{Dec} : \{0,1\}^y \to \{0,1\}^x$ such that $\mathsf{Dec}(\mathcal{E}(z)) = z$ with probability at least ϵ where $z \leftarrow_\$ \{0,1\}^x$, then y is at least $x - \log 1/\epsilon$.*

Before we present the encoding algorithm, recall the definition of 'special' cycles. They are cycles where the input salt of each query has exactly one pre-image. This implies that no salt is part of more than 1 of the $i \log N$ 'special' cycles by definition. Thus, for each cycle there is a unique and distinct query, denoted (b, m), that is made after all the other queries in the cycle. Moreover, the input salt of this query, b, has a unique pre-image (among the offline queries), which itself has a unique pre-image and so on until $H(b, m)$ is the unique pre-image of another salt in the cycle. Our encoding compresses the output of the last query made on each of the $i \log N$ cycles.

We give a formal description of our encoding algorithm next.

- Store the $i \log N$ queries that are the last queries made in their respective 'special' cycle in an unordered set, say W. This would require $\log \binom{iT}{i \log N}$ bits.
- Delete the output of the queries, each $\log N$ bits long, in the unordered set W from the database (table of sampled queries on H).

The decoding algorithm is trivial. For every query (b, m) in the set W, it follows the chain backward using the uniqueness of pre-image, until it reaches some query whose input salt, denote b', has no pre-image and set $H(b, m) = b'$. For completeness we give a formal description of the decoding algorithm, which is as follows:

For every query in W, say (a, m):

- Set temp= a
- While true:
  - If there is no query with output temp: break.
  - Find the query in the table with output temp. Say the query is (a', m').
  - Set temp= a'.
- Output $H(a, m)$ =temp.

Let $\epsilon = \Pr[\mathbf{E}_3^i]$. Then,

$$\log\left(\frac{1}{\epsilon}\right) + \log\binom{iT}{i \log N} \geq i \log N \cdot \log N$$

$$\Rightarrow \log\left(\frac{1}{\epsilon}\right) + i \log N \cdot \log\left(\frac{eiT}{i \log N}\right) \geq i \log N \cdot \log N$$

$$\Rightarrow \epsilon \leq \left(\frac{T}{N}\right)^{i \log N}.$$

□

Condition j	1	2	3	4	5
ϵ_j	$\frac{T}{N}$	$\frac{T^2}{N}$	$\frac{16\kappa_i TB\log^2 N}{N}$	$\frac{iT}{N} \cdot \frac{T^2}{N}$	$\frac{iTB\log N}{N}$

Table 4.2: Summary of upper bounds on ϵ_j for $j \in [5]$ where $\kappa_i := \max\{i, i^2T^2/N\}$.

Chapter 5

Multi-Collisions

We study multi-collisions in MD hash functions in the AI-RO model. Formally, by multi-collisions we mean the adversary has to find m distinct messages, for some parameter m, that have the same output under a given cryptographic hash function. We focus on MD based hash functions.

Say the MD hash function uses a one-way compression function h in $[N]\times[M] \to [N]$, then an m-way multi-collision is m distinct messages in $[M]^+$ that have the same output under MD_h and a given salt $a \in [N]$. We consider the following question.

> *How efficiently can an adversary in AI-RO model running in time T and computing an S-bit advice, find m-way multi-collisions in MD based hash functions?*

We further study bounded length multi-collisions in MD hash functions in the AI-RO model. An m-way B-length multi-collision is m distinct messages in $[M]^{\leq B}$ that have the same output under MD_h and a given salt $a \in [N]$. We consider the following question.

> *How efficiently can an adversary in AI-RO model running in time T and computing an S-bit advice, find m-way B-**length** multi-collisions in MD based hash functions?*

Some of the existing digital signature constructions reduce their security to multi-collision resistance which motivates this study. Multi-collision resistance is considered a better assumption than collision resistance as finding multi-collisions is harder (than finding collisions) in the RO model and thus should be easier to achieve. We want to know if this the case in AI-ROM too. Our study in the AI-RO model shows otherwise for unbounded length and B-length multi-collisions for sufficiently large B.

5.1 Definitions

We start by defining m-way multi-collision resistance game for MD against auxiliary input adversaries. Such an adversary works in two stages: (1) In the *Offline/pre-processing stage*, the adversary is unbounded but outputs a bounded length string, which we will refer to as advice. (2) In the *online stage*, the adversary takes the advice and makes a bounded number of queries.

Definition 27 ((S, T, m)-AI adversary)**.** *A pair of algorithms $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ is an (S, T, m)-AI adversary against MD_h if*

- *$\mathcal{A}_1^h$ is unbounded (making unbounded number of oracle queries to h) and outputs S bits of advice σ;*
- *$\mathcal{A}_2^h$ takes σ and a salt $a \in [N]$, issues T queries to h and outputs $\mathsf{msg}_1, \ldots, \mathsf{msg}_m$*

where h is a function in $[N] \times [M] \to [N]$.

The m-way multi-collision resistance security game, namely m-AICR, is defined next.

Definition 28. *For a fixed random function h in $[N] \times [M] \to [N]$ and a random salt $a \in [N]$, the game* m-AICR *is defined in fig. 5.1.*

Game $\mathsf{m\text{-}AICR}_{h,a}(\mathcal{A})$
$\sigma \leftarrow \mathcal{A}_1^h$
$\mathsf{msg}_1, \ldots, \mathsf{msg}_m \leftarrow \mathcal{A}_2^h(\sigma, a)$
If $\mathsf{msg}_i \neq \mathsf{msg}_j$ and $\mathrm{MD}_h(a, \mathsf{msg}_i) = \mathrm{MD}_h(a, \mathsf{msg}_j) \quad \forall i \neq j \in [m]$
Then Return 1
Else Return 0

Figure 5.1: Security game $\mathsf{m\text{-}AICR}_{h,a}(\mathcal{A})$

For an (S, T, m)-AI adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, the advantage of $\mathcal{A}$, denoted as $\mathsf{Adv}^{\text{m-AICR}}(\mathcal{A})$, is defined as the probability of $\mathsf{m\text{-}AICR}_{h,a}(\mathcal{A})$ returning 1. We define the (S, T, m)-AI multi-collision resistance, denoted by $\mathsf{Adv}^{\text{m-AICR}}(S, T, m)$, as the maximum advantage taken over all (S, T, m)-AI adversaries.

Definition 29. *For a fixed random function h in $[N] \times [M] \to [N]$ and a random salt $a \in [N]$, the game $\mathsf{m\text{-}AICR}^B$ is defined in fig. 5.2.*

For an (S, T, m)-AI adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, the advantage of $\mathcal{A}$ in the game $\mathsf{m\text{-}AICR}^B$, denoted as $\mathsf{Adv}_{\mathsf{B}}^{\text{m-AICR}}(\mathcal{A})$, is defined as the probability of $\mathsf{m\text{-}AICR}_{h,a}^B(\mathcal{A})$ returning 1. We define the (S, T, m)-AI B-length multi-collision resistance, denoted by $\mathsf{Adv}_{\mathsf{B}}^{\text{m-AICR}}(S, T, m)$, as the maximum advantage taken over all (S, T, m)-AI adversaries.

Game $\mathsf{m\text{-}AICR}^{B}_{h,a}(\mathcal{A})$
$\sigma \leftarrow \mathcal{A}_1^h$
$\mathsf{msg}_1, \ldots, \mathsf{msg}_m \leftarrow \mathcal{A}_2^h(\sigma, a)$
If any of $\mathsf{msg}_1, \ldots, \mathsf{msg}_m$ have more than B blocks
Then Return 0
If $\mathsf{msg}_i \neq \mathsf{msg}_j$ and $\mathrm{MD}_h(a, \mathsf{msg}_i) = \mathrm{MD}_h(a, \mathsf{msg}_j) \quad \forall i \neq j \in [m]$
Then Return 1
Else Return 0

Figure 5.2: Security game $\mathsf{m\text{-}AICR}^{B}_{h,a}(\mathcal{A})$

5.2 Results

Theorem 40. *For any S, T, m and N,*

$$\mathsf{Adv}^{\text{m-AICR}}(S, T, m) = \widetilde{\theta}\left(\frac{ST^2}{N}\right).$$

Theorem 41. *For any S, T, m, N and $B = 1$,*

$$\mathsf{Adv}^{\text{m-AICR}}_{\mathsf{B}}(S, T, m) = \widetilde{\theta}\left(\frac{S}{mN} + \frac{(T/m)^m}{N^{m-1}}\right).$$

Theorem 42. *For any S, T, m, N and $B = 2$ such that $T \leq S$ and $m \leq ST$,*

$$\mathsf{Adv}^{\text{m-AICR}}_{\mathsf{B}}(S, T, m) = \widetilde{\theta}\left(\frac{ST}{mN}\right).$$

Theorem 43. *For any S, T, m, B, N and M such that $T \leq SB$*

$$\mathsf{Adv}^{\text{m-AICR}}_{\mathsf{B}}(S, T, m) = \widetilde{\Omega}\left(\frac{ST}{m^{1/(B-1)}N}\right).$$

and

$$\mathsf{Adv}^{\text{m-AICR}}_{\mathsf{B}}(S, T, m) = \widetilde{\Omega}\left(\frac{STB}{N}\right)$$

when $B \geq 2 \log m$.

Theorem 44. *For any S, T, m, N and M such that $T \leq SB$,*

$$\mathsf{Adv}^{\text{m-AICR}}_{\mathsf{B}}(S, T, m) = \widetilde{O}\left(\max\left\{1, \frac{ST^2}{N}\right\} \cdot \frac{STB}{N}\right).$$

5.3 Proof of Theorem 40

5.3.1 *Lower Bound*

We will first present an attack achieving the bound in the theorem. We assume $M > N$ and thus collisions exist for every salt in $[N]$ under h. We also assume $m \leq N$. Then our (S, T, m)- AI adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ is as follows:

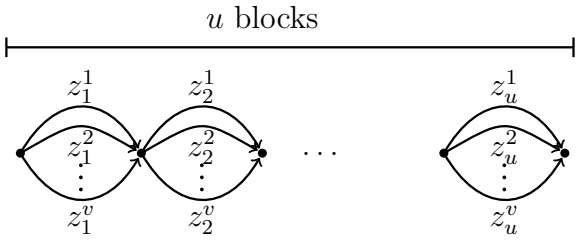


Figure 5.3: u-length chain of v-way collisions

1. In *Offline stage*, $\mathcal{A}_1$ picks $s = S/(3 \log m \cdot \log M)$ random salts, denoted $\{a_1, \ldots, a_s\}$ and zero-walks for $T/2$ times on each of them to obtain, say $\{a'_1, \ldots, a'_s\}$ set of salts. Then for each of the salt a'_i in this set, $\mathcal{A}_1$ finds a $\log m$-length chain of 2-ways collisions (refer fig. 5.3 for a pictorial depiction u-length chain of v-way collisions) and store these along with a'_i as advice.

2. In *Online stage*, $\mathcal{A}_2$ takes the advice output by $\mathcal{A}_1$ and random salt a and zero walks up to T times. If output of any of these queries is some $a' \in \{a'_1, \ldots, a'_s\}$, then $\mathcal{A}_2$ succeeds in outputting an m-way collision.

Then,

$$\mathsf{Adv}^{\text{m-AICR}}(\mathcal{A}) = \widetilde{\Omega}\left(\frac{ST^2}{N}\right).$$

5.3.2 *Upper Bound*

We will give a reduction from AICR to this end. Say there exists an (S, T, m)-AI adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ that finds an m-way multi-collision with advantage $\widetilde{\omega}(ST^2/N)$, then we present an $(S, T, 2)$-AI adversary $\mathcal{A}' = (\mathcal{A}'_1, \mathcal{A}'_2)$ that finds a collision with the same advantage. From [6], we know that is impossible. Hence, showing that there exists no (S, T, m)-AI adversary with advantage $\widetilde{\omega}(ST^2/N)$ in the m-AICR game.

Adversary $\mathcal{A}'$ is as follows:

$\underline{\mathcal{A}'_1}$:

- Run $\mathcal{A}_1$ and answer the queries of $\mathcal{A}_1$ using it's own oracle
- Return output of $\mathcal{A}_1$

$\underline{\mathcal{A}'_2}$:

- Forward the input salt to $\mathcal{A}_2$ and answer the queries of $\mathcal{A}_2$ using it's own oracle.
- Say the output of $\mathcal{A}_2$ is $\mathsf{msg}_1, \ldots, \mathsf{msg}_m$, return $\mathsf{msg}_1, \mathsf{msg}_2$.

5.4 Proof of Theorem 41

5.4.1 *Lower Bound*

The following trivial attack:

1. In pre-computation phase, stores m-way collisions for $\tilde{\Omega}(S/m)$ salts as part of the advice.
2. In online phase, if the challenge salt is among the salts whose m-way collision is contained in the advice, adversary returns the corresponding m-way collision. Else the adversary tries to find m-way collision using it's queries to the h.

achieves $\tilde{\Omega}\left(\frac{S}{mN} + \frac{(T/M)^m}{N^{m-1}}\right)$ advantage.

5.4.2 *Upper Bound*

We will follow the approach of global compression from the work of [7] to prove the security bound in theorem.

It is worth noting that the approach of reducing the problem to multi-instance m-way collision finding game from our work in [ACDW20] and work of [CGLQ20] does not seem to give a tight security bound. It is unclear whether this is due to lack of sophistication in our analysis or an indication of a gap between pre-computation and Multi-instance game models.

We present our proof next.

Fix an (S, T, m)-AI adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ having advantage ϵ. Let σ denote the S-bit advice output by $\mathcal{A}_1$ and let $\mathcal{G}$ be the set of salts in $[N]$ on which $\mathcal{A}_2$ succeeds in finding m-way collision when given σ and makes T queries to h.

We want to show that the set $\mathcal{G}$ cannot be large for any (S, T, m)-AI adversary. In other words, it is impossible for any (S, T, m)-AI adversary to succeed in finding m-way collisions on 'too' many salts, irrespective of what information is contained in the S-bit advice σ. The idea is to give an encoding algorithm that uses such an adversary that succeeds on a 'lot' of salts to compress h, which should be impossible as h is a random function.

Note that since $\mathcal{A}_2$ finds m-way collisions on each of the salts in $\mathcal{G}$, for every $a \in \mathcal{G}$ there should exist m queries of the type $(a, *) \in [N] \times [M]$ such that outputs of all the m queries are same under h. Our encoding algorithm uses this repetition in outputs to compress as follows:

1. Store the advice σ (requires S-bits)
2. Store the size of the set $\mathcal{G}$ (in other words, the number of elements in $\mathcal{G}$), denoted $|\mathcal{G}|$ (requires $\log N$ bits)

3. Store the set $\mathcal{G}$ (requires $\log \binom{N}{|\mathcal{G}|}$ bits)

4. Store the unordered set of $|\mathcal{G}| \cdot m$ queries corresponding to the m-way collisions for each salt in $\mathcal{G}$. Let's denote this set by $\mathcal{X}$. Note that $\mathcal{X}$ is a subset of the $|\mathcal{G}| \cdot T$ queries made by $\mathcal{A}_2$ using the assumption that $\mathcal{A}_2$ has to query it's outputs. Thus, storing $\mathcal{X}$ requires $\log \binom{|\mathcal{G}|T}{|\mathcal{G}|m}$ bits.

5. Delete the outputs of $\mathcal{G}|(m-1)$ queries in $\mathcal{X}$ from table of h. (Note that the table for h is stored as follows: table contains output of h on the queries made by $\mathcal{A}_2$ on the set $\mathcal{G}$ followed by the output of h on the remaining entries of queries in $[N] \times [M]$ in lexicographic order.) This saves $|\mathcal{G}|(m-1) \cdot \log N$ bits.

Via the compression argument of De et al. [19], the following holds:

$$S + \log N + \log \binom{N}{|\mathcal{G}|} + \log \binom{|\mathcal{G}|T}{|\mathcal{G}|m} + NM \log N - |\mathcal{G}|(m-1) \cdot \log N \geq NM \log N$$

$$\Rightarrow S + \log N + |\mathcal{G}| \log \left(\frac{N}{|\mathcal{G}|}\right) + |\mathcal{G}|m \log \left(\frac{|\mathcal{G}|T}{|\mathcal{G}|m}\right) \geq |\mathcal{G}|(m-1) \log N$$

$$\Rightarrow S + \log N \geq |\mathcal{G}| \log \left(\frac{N^{m-1}|\mathcal{G}|}{N(T/m)^m}\right)$$

We take expectation on both sides of the above equation and use convexity of the function $x \log x$ along with $\mathbb{E}[|\mathcal{G}|] = \epsilon N$ as follows:

$$\mathbb{E}[S + \log N] \geq \mathbb{E}\left[|\mathcal{G}| \log \left(\frac{N^{m-1}|\mathcal{G}|}{N(T/m)^m}\right)\right]$$

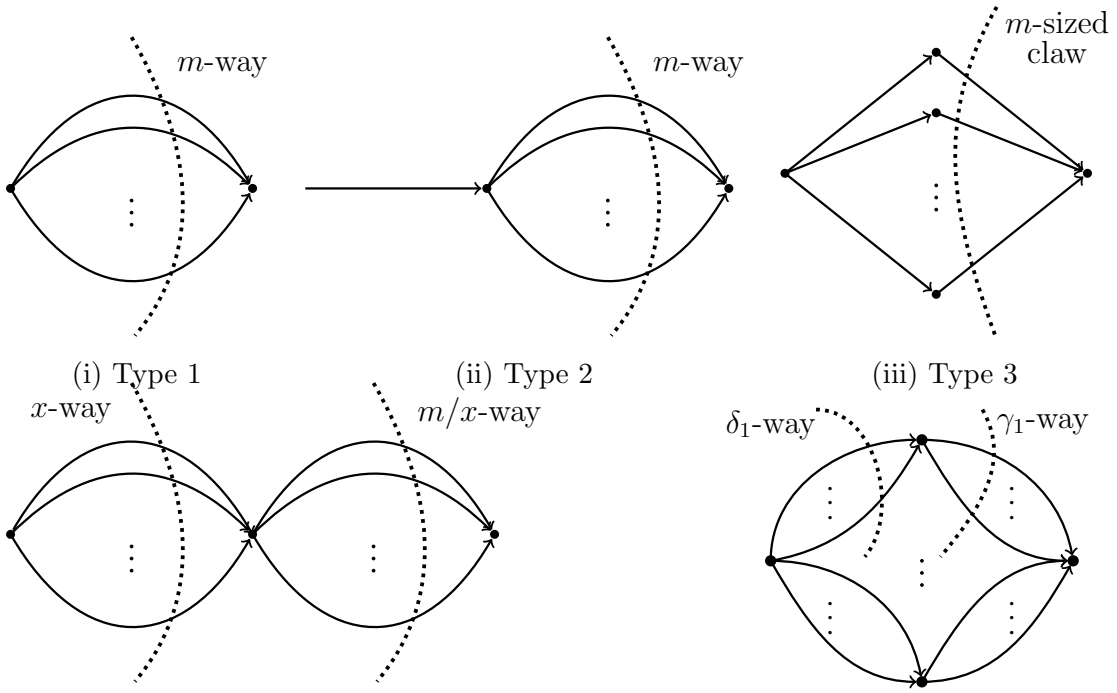
$$\Rightarrow S + \log N \geq \mathbb{E}[|\mathcal{G}|] \log \left(\frac{N^{m-1}\mathbb{E}[|\mathcal{G}|]}{N(T/m)^m}\right)$$

$$\geq \epsilon N \cdot \log \left(\frac{N^{m-1} \cdot \epsilon N}{N(T/m)^m}\right) \geq \epsilon N \cdot \log \left(\frac{N^{m-1} \cdot \epsilon}{(T/m)^m}\right)$$

Consider the following two cases:

1. If
$$\frac{N^{m-1}\epsilon}{(T/m)^m} \leq 2^m$$
This simply implies
$$\epsilon \leq \frac{(2T/m)^m}{N^{m-1}}$$

2. Otherwise
$$\frac{N^{m-1}\epsilon}{(T/m)^m} > 2^m$$
which implies
$$S + \log N \geq \epsilon N \log 2^m \Rightarrow \epsilon \leq \frac{S + \log N}{mN}$$



(i) Type 1 (ii) Type 2 (iii) Type 3

(iv) Type 4 (v) Type 5

Figure 5.4: All types of 2-block m-way colliding chains

This completes the proof for the security bound.

5.5 Proof of Theorem 42

5.5.1 *Security bound*

Following the approach of several prior works [IK10, ACDW20, CGLQ20, GK22, AGL22], we will use reduction of security in the auxiliary input RO model to the security in the Multi-instance game and then bound the security in the Multi-instance game to prove the theorem. It is worth-noting that unlike all these prior works (except [IK10]), for us it would suffice to analyze the parallel version of the Multi-instance game. Also our analysis is slightly different from that of all the other prior works.

To put it simply, we will define the parallel version of Multi-instance game for each type of m-way collision and analyze each of them separately. To this end, we first identify all the types of 2-block m-way collisions. See fig. 5.4 for a pictorial depiction of the types. It is worth noting that in our analysis we assume $m = \omega(\log^c n)$ for a constant c. And that is because for $m = O(\log^c n)$, the theorem holds trivially from the reduction presented in section 5.3.2 and the result of [ACDW20].

Let's fix an (S, T, m)-AI adversary $\mathcal{A}$. Let E_t be the event that $\mathcal{A}$ wins game $\mathsf{m\text{-}AICR}^2$ by outputting m-way collision of type t in fig. []. Then

$$\mathsf{Adv}_2^{\text{m-AICR}}(\mathcal{A}) \leq \sum_{t=1}^{6} \Pr[\mathsf{m\text{-}AICR}^2_{h,a}(\mathcal{A}) = 1 \wedge E_t]$$

Definition 30. *For a fixed random function h in $[N] \times [M] \to [N]$, fixed function S, a random salt $a \in [N]$ any $t \in [6]$, the game* $\mathsf{m\text{-}MICR}^{S,t}$ *is defined in fig. 5.5.*

Game $\mathsf{m\text{-}MICR}^{S,t}_{h,a}(\mathcal{A})$

Sample $\{a_1, \ldots, a_S\} \leftarrow_\$ \left[\binom{N}{S}\right]$
$(\mathsf{msg}^i_1, \ldots, \mathsf{msg}^i_m)^S_{i=1} \leftarrow \mathcal{A}^h(\{a_1, \ldots, a_S\})$
If for any $i \in [S]$ and any of $\mathsf{msg}^i_1, \ldots, \mathsf{msg}^i_m$ have more than 2 blocks
  Then Return 0
If $\forall i \in [S]$: $(\mathsf{msg}^i_1, \ldots, \mathsf{msg}^i_m)$ is m-way collision of type t on a_i
  Then Return 1
Else Return 0

Figure 5.5: Security game $\mathsf{m\text{-}MICR}^{S,t}_{h,a}(\mathcal{A})$

For an (S, T, m)-MI adversary $\mathcal{A}$, the advantage of $\mathcal{A}$ in the game $\mathsf{m\text{-}MICR}^{S,t}$*, denoted as* $\mathsf{Adv}^{\text{m-MICR}}_{\mathsf{t}}(\mathcal{A})$*, is defined as the probability of* $\mathsf{m\text{-}MICR}^{S,t}_{h,a}(\mathcal{A})$ *returning 1. We define the (S, T, m)-MI 2-length multi-instance multi-collision resistance, denoted by* $\mathsf{Adv}^{\text{m-MICR}}_{\mathsf{t}}(S, T, m)$*, as the maximum advantage taken over all (S, T, m)-MI adversaries.*

Lemma 45. *For any S, T, N, m and $t \in [6]$ such that $m = \omega(\log^2 N)$ and $m \leq ST$,*

$$\mathsf{Adv}^{\text{m-MICR}}_{\mathsf{t}}(S, T, m) = \left(\tilde{O}\left(\frac{ST}{mN} + \frac{(T/m)^m}{N^{m-1}}\right)\right)^S.$$

Proof of theorem 42 follows from this lemma and [ai-mi reduction] as follows:

$$\forall t \in [6] : \Pr[\mathsf{m\text{-}AICR}^2_{h,a}(\mathcal{A}) = 1 \wedge E_t] \leq \mathsf{Adv}^{\text{m-MICR}}_{\mathsf{t}}(S, T, m)$$

$$\Rightarrow \mathsf{Adv}^{\text{m-AICR}}_{2}(\mathcal{A}) \leq \sum_{t=1}^{6} \Pr[\mathsf{m\text{-}AICR}^2_{h,a}(\mathcal{A}) = 1 \wedge E_t] \leq \sum_{t=1}^{6} \mathsf{Adv}^{\text{m-MICR}}_{\mathsf{t}}(S, T, m)$$

$$= \left(\tilde{O}\left(\frac{ST}{mN} + \frac{(T/m)^m}{N^{m-1}}\right)\right)^S$$

To complete the proof of theorem 42, we prove this lemma next.

Proof. Let's fix a (S, T, m)-MI adversary $\mathcal{A}$ and let h be a random oracle. We denote the input of random S-sized set of salts by $\{a_1, \ldots, a_S\}$. Let $\mathbf{X}^t_i$ be the indicator variable that $\mathcal{A}$ wins on salt a_i, i.e., $\mathcal{A}$ finds m-way collision of type t on salt a_i for $t \in [6]$.

Next we will show for each $t \in [6]$, that $\mathsf{Adv}^{\text{m-MICR}}_{\mathsf{t}}(S, T, m) = \left(\tilde{O}\left(\frac{ST}{mN} + \frac{(T/m)^m}{N^{m-1}}\right)\right)^S$. Let's begin with $\underline{\mathbf{t = 1}}$.

$$\mathsf{Adv}^{\text{m-AICR}}_{1}(\mathsf{S}, \mathsf{T}, \mathsf{m}) = \Pr[\forall i \in [S]\ \exists \text{1-block } m\text{-way collision for } a_i \text{ in } ST \text{ queries}]$$

$$\leq \frac{\binom{ST}{Sm}}{N^{S(m-1)}} \leq \left[\frac{(eT/m)^m}{N^{m-1}}\right]^S.$$

Next, we analyze for $\underline{\mathbf{t = 2}}$. The analysis would be trivial and same as for $t = 1$ if m-way collision for each salt used distinct queries. However, queries can be reused among collisions for different salts. Refer to fig. [] for a visual depiction.

In the analysis, we consider the following cases:

1. $x \geq m$
   It would suffice to take into account the probability of x of the ST queries having the same output, which is:

$$\frac{\binom{ST}{x}}{N^{x-1}} \leq \left(\frac{eST}{xN}\right)^x \cdot N \leq \left(\frac{2eST}{xN}\right)^x \cdot \frac{1}{2^x} \cdot N \leq \left(\frac{2eST}{mN}\right)^x$$

   where the last inequality uses that $\frac{1}{2^x} \leq \frac{1}{2^m} \leq N$ as $x \geq m \geq \log N$.

2. $x < m$
   Then the probability of an m-way collision is:

$$\leq \frac{\binom{ST}{m}}{N^{m-1}} \leq \left(\frac{eST}{mN}\right)^m \cdot N \leq \left(\frac{2eST}{mN}\right)^m \cdot \frac{1}{2^m} \cdot N \leq \left(\frac{2eST}{mN}\right)^x$$

   where the last inequality holds for $m \geq \log N$ and $2eST/mN \leq 1$.

Now we give a proof via compression. Say the number of structures shown in fig. [] among ST queries be y with the in-degree of the input salt of the m-way collision being $\delta_1, \ldots, \delta_y$ respectively. Then it holds that

$$\delta_1 + \cdots + \delta_y \geq S$$

The encoding algorithm will be as follows:

1. Store the subset of ST queries forming m-way collisions in an unordered fashion. From the table of h, delete the entries corresponding to the queries in the set except the first query. There will be y such m-sized sets.

2. For $i \in [y]$: if $\delta_i > m$ do the following

   - Store δ_i
   - Store the δ_i-sized set of queries that have the same output. From the table of h, delete the entries corresponding to the queries in the set except the first query.

Say ϵ is the advantage. Then via the compression argument, following holds:

$$\log \epsilon \leq y \log \binom{ST}{m} - y \cdot (m-1) \log N + \sum_{i \in [y]: \delta_i > m} \left[\log S + \log \binom{ST}{\delta_i} - (\delta_i - 1) \log N\right]$$

$$\Rightarrow \epsilon \leq \left(\frac{4eST}{mN}\right)^S$$

using $\delta_1 + \cdots + \delta_y \geq S$ and for any i such that $\delta_i > m > \log N$ the following holds:

$$S \cdot \frac{\binom{ST}{\delta_i}}{N^{\delta_i - 1}} \leq \left(\frac{eST}{\delta_i N}\right)^{\delta_i} \cdot S \cdot N \leq \left(\frac{4eST}{xN}\right)^{\delta_i} \cdot \frac{1}{4^x} \cdot S \cdot N \leq \left(\frac{4eST}{mN}\right)^{\delta_i}.$$

Next, we analyze $\underline{\mathbf{t = 3}}$. Again analysis will be trivial if the collisions for each salt used distinct queries. So, we will focus when that is not the case. Say there are $x \geq m$ queries with the same output and y salts 'use' them to get m-way collision. Again in the analysis, there are two cases:

1. $y \leq x$
   Again it suffices to take into account that the probability of x of the ST queries having the same output is 'small' as follows:

   $$\frac{\binom{ST}{x}}{N^{x-1}} \leq \left(\frac{eST}{xN}\right)^{x} \cdot N \leq \left(\frac{2eST}{xN}\right)^{x} \cdot \frac{1}{2^x} \cdot N \leq \left(\frac{2eST}{mN}\right)^{y}$$

   where the last inequality uses that $y \leq x$ (along with that $2eST \leq mN$) and $\frac{1}{2^x} \leq \frac{1}{2^m} \leq N$ as $x \geq m \geq \log N$.

2. $y > x$
   Note that for this case, it should be that the there are x queries among ST queries that have the same output and that each of the y salts need to at least m-way 'hit' this x-sized claw. Let the in-degree of each node in the x-sized claw be denoted by $\delta_1, \ldots, \delta_x$. Then we will show it suffices to compress only for $i \in [x]$ where $\delta_i \geq m$.

   First, let's consider the number of queries(or edges) from the y salts hitting the claw. There should be at least $y \cdot m$ such queries(or edges). Then the average in-degree of each node on the claw is $\geq (y \cdot m)/x \geq m$ as $y > x$. This implies that there exists nodes in the claw with in-degree $\geq m$.

   For simplicity and WLOG, let's assume $\delta_1 \leq \delta_2 \leq \cdots \leq \delta_x$. Let's say for some $z < x$, $\delta_1, \ldots, \delta_z < m$ and $\delta_{z+1}, \ldots, \delta_x \geq m$. Then

   $$\begin{aligned}\delta_{z+1} + \cdots + \delta_x &\geq y \cdot m - (\delta_1 + \cdots + \delta_z) \\ &\geq y \cdot m - z \cdot m = m(y - z) \\ &\geq m(y - x) \geq y - x.\end{aligned}$$

   Then the probability of finding such a structure is:

   $$\frac{\binom{ST}{x}}{N^{x-1}} \cdot \prod_{i=z+1}^{x} \left[\frac{\binom{ST}{\delta_i}}{N^{\delta_i - 1}}\right] \leq \left(\frac{eST}{xN}\right)^{x} \cdot N \prod_{i=z+1}^{x} \left[\left(\frac{eST}{\delta_i N}\right)^{\delta_i} \cdot N\right]$$
   $$\leq \left(\frac{2eST}{mN}\right)^{x+\delta_{z+1}+\cdots+\delta_x} \leq \left(\frac{2eST}{mN}\right)^{y}$$

Say there are z structures as given in fig. []. We give a proof via compression.The encoding algorithm will be as follows.
For each of the z structures:

1. Store x and the subset of ST queries forming x-sized claw in an unordered fashion. From the table of h, delete the entries corresponding to the queries in the set except the first query.

2. For $i \in [x]$: if $\delta_i > m$ do the following

   - Store δ_i
   - Store the δ_i-sized set of queries that have the same output. From the table of h, delete the entries corresponding to the queries in the set except the first query.

It is worth noting that as the size of each of the sets in the encoding algorithm above is at most ST, storing their size requires at most $\log ST$ bits. And since each of these sets have a size of at least $m \geq \log N$ we can use the following:

$$\left(\frac{2eST}{mN}\right)^{\delta} \cdot ST \leq \left(\frac{4eST}{mN}\right)^{\delta} \cdot \frac{1}{2^{\delta}} \cdot ST \leq \left(\frac{4eST}{mN}\right)^{\delta}$$

when $\delta \geq \log N$. Thus, the advantage for $t = 3$ can be bounded to $\left(\frac{4eST}{mN}\right)^{S}$.

Next, we analyze for $\underline{\mathbf{t = 4}}$. Refer to fig. [] for a visual depiction of this case.

We would like to clarify at this point that we cheated a bit before. Actually event E_4 would have $\log N$ sub-events and each of them gets their own reduction to MI game and separate analysis but we clubbed them together for simplicity. Note that having $\log N$ sub-events only adds a multiplicative factor of $\log N$ in the final bound which is tolerable. We describe these sub-events next.

1. **Sub-event** $E_4^1 : x \geq \log N$
   It suffices to analyze this collision type in a similar fashion to collision-type 1. That is advantage of the adversary in MI game would be bounded by the probability of finding $\log N$-way collision on S distinct salts among ST queries, which is
   $$\frac{\binom{ST}{S \log N}}{(N^{\log N - 1})^S} \leq \left(\frac{eT}{\log N \cdot N}\right)^{S \log N} \cdot N^S \leq \left(\frac{1}{N}\right)^S$$
   where the last inequality holds using $\log N \geq 2$, $eT \leq 2N$ and $ST \geq m$.

2. $(\log N - 1)$ **Sub-events**, one for each $x \in [2, \log N)$:
   For any $x < \log N \Rightarrow m/x > m/\log N$
   Then the probability of finding each 1-block $(m/\log N)$-way collision is
   $$\frac{\binom{ST}{m/\log N}}{N^{m/\log N - 1}} \leq \left(\frac{eST \log N}{mN}\right)^{m/\log N} \cdot N \leq \left(\frac{2eST \log N}{mN}\right)^{m/\log N}$$

using $\frac{1}{2^{m/\log N}} \cdot N \leq 1$ when $m \geq \log^2 N$.

Say there exists y such structures of 1-block $(m/\log N)$-way collision in ST queries. If $y \geq S \log N/m$, then it suffices to only compress these structures to get the desired bound. When $y < S \log N/m$, it becomes necessary to take into account the probability of different salts sharing a structure.

So, we analyze the case $y < S \log N/m$ more closely. It is important to note S salts imply at least $S \cdot x$ queries hitting one of these y structures. Then the average number of queries hitting a structure is at least $S \cdot x \cdot \frac{m}{S \log N} \geq \frac{m}{\log N}$. This implies there exists structures hit by more than $m/\log N$ queries and these can be compressed in a similar fashion as 1-block $(m/\log N)$-way collisions.

Let δ_i denote the number of queries hitting the i^{th} structure. Then

$$\delta_1 + \cdots + \delta_y \geq S \cdot x$$

WLOG, we can assume $\delta_1 \leq \cdots \leq \delta_y$ and let $z < y$ be such that $\forall i \in [z]$: $\delta_i < m/\log N$ and $\forall i > z : \delta_i \geq m/\log N$. Then

$$\delta_{z+1} + \cdots + \delta_y \geq S \cdot x - (\delta_1 + \cdots + \delta_z) \geq S \cdot x - z \cdot \frac{m}{\log N} \geq \left(S - \frac{m}{\log N}\right) \cdot x \geq S - \frac{m}{\log N}$$

We give a proof via compression.The encoding algorithm will be as follows. For $i \in [y]$:

(a) Store the subset of ST queries forming i^{th} structure in an unordered fashion. From the table of h, delete the entries corresponding to the queries in the set except the first query.

(b) If $\delta_i > m/\log N$ do the following

- Store δ_i
- Store the δ_i-sized set of queries that hit the i^{th} structure in an unordered fashion. From the table of h, delete the entries corresponding to the queries in the set except the first query.

It is worth noting that as the size of each of the sets in the encoding algorithm above is at most S, storing their size requires at most $\log S$ bits. And since each of these sets have a size of at least $\log N$ we can use the following:

$$\left(\frac{2eST \log N}{mN}\right)^{\delta} \cdot S \leq \left(\frac{4eST \log N}{mN}\right)^{\delta} \cdot \frac{1}{2^{\delta}} \cdot S \leq \left(\frac{4eST \log N}{mN}\right)^{\delta}$$

when δ denotes the size of the set. Thus, the advantage for $t = 4$ can be bounded to $\left(\frac{4eST \log N}{mN}\right)^{S}$.

Finally, we analyze $\underline{\mathbf{t = 5}}$. To this end, we consider a structure with x branches, where the i^{th} branch is 1-block γ_i-way collision and all x branches have the same output. Let y be the number of salts that use this structure in collisions. Let δ_i denote the number of queries hitting the i^{th} branch of the structure. Let δ_i^j denote the number of queries from the j^{th} of y salts hitting the i^{th} branch of the structure. Then $\delta_i = \sum_{j=1}^{y} \delta_i^j$.

We would like to state that we had cheated earlier when we stated E_5 as a single event and defined the MI game corresponding to it. Actually event E_5 corresponds to 2 sub-events, each of which we need to reduce to MI game and analyze on its own. This affects the final bound only by a constant factor. We describe the sub-events next.

1. $\forall j \in [y], \exists i \in [x]$ such that $\delta_i^j \geq \log N$
   Analysis for this collision type is similar to that for event E_4^1. That is advantage of the adversary in MI game would be bounded by the probability of finding $\log N$-way collision on S distinct salts among ST queries, which is

   $$\frac{\binom{ST}{S \log N}}{(N^{\log N - 1})^S} \leq \left(\frac{eT}{\log N \cdot N}\right)^{S \log N} \cdot N^S \leq \left(\frac{1}{N}\right)^S \leq \left(\frac{ST}{mN}\right)^S$$

   where the second-to-last inequality holds using $\log N \geq 2$ and $eT \leq 2N$ and the last inequality holds using $ST \geq m$.

2. $\forall i \in [x], j \in [y] : \delta_i^j < \log N$

We inspect the second sub-event above. We need to analyze two cases:

1. $y \leq \gamma_1 + \cdots + \gamma_x$
   Given $\delta_i^j < \log N$ for all i, j, we can bound $\delta_i = \sum_{j=1}^{y} \delta_i^j \leq y \cdot \log N$ for every i. Using this together with the fact that

   $$\delta_1 \cdot \gamma_1 + \cdots + \delta_x \cdot \gamma_x \geq y \cdot m$$

   implies

   $$\gamma_1 + \cdots + \gamma_x \geq \frac{y \cdot m}{y \log N} = \frac{m}{\log N}$$

   and as $y \leq \gamma_1 + \cdots + \gamma_x$, it suffices to compress the structure. Encoding algorithm should compress for each such structure as follows:

   (a) Store the sum of size of the x branches, i.e., $\gamma_1 + \cdots + \gamma_x$ which is at most ST and hence requires at most $\log(ST)$-bits.

   (b) Store the subset of ST queries forming the x branches in an unordered fashion. From the table of h, delete the entries corresponding to the queries in the set except the first query.

Via the compression argument, we obtain the probability of $\mathcal{A}$ finding such a structure is at most

$$ST \cdot \left(\frac{\binom{ST}{z}}{N^{z-1}} \right) \leq \left(\frac{eST}{zN} \right)^z \cdot ST \cdot N \leq \left(\frac{4eST \log N}{mN} \right)^y$$

where $z = \gamma_1 + \cdots + \gamma_x \geq m/\log N$ and the last inequality uses $y \leq z$ and $m \geq \log^2 N$.

2. $y > \gamma_1 + \cdots + \gamma_x$
As in the previous case, it holds that

$$\gamma_1 + \cdots + \gamma_x \geq \frac{m}{\log N}$$

and we will compress the x branches. However, in this case that wouldn't suffice to get the desired bound and we will have to use that salts are sharing the branches. Since more salts are using fewer branches, this would lead to lots of collisions some of which can be compressed.

Let $\mathcal{Z}$ be the subset of x branches such that for every $i \in \mathcal{Z}$ the in-degree of the i^{th} branch, i.e., $\delta_i < m/\log N$. We will compress the queries hitting the rest of the branches in a manner similar to the previous case. What remains to be shown is we can compress sets such that the sum of their sizes is $y - (\gamma_1 + \cdots + \gamma_x)$, which we do next.

$$\begin{aligned}
&\delta_1 \cdot \gamma_1 + \cdots + \delta_x \cdot \gamma_x \geq y \cdot m \\
&\quad \Rightarrow \sum_{i \in [x] \setminus \mathcal{Z}} \delta_i \cdot \gamma_i \geq y \cdot m - \left(\sum_{i \in \mathcal{Z}} \delta_i \cdot \gamma_i \right) \geq y \cdot m - \frac{m}{\log N} \cdot \left(\sum_{i \in \mathcal{Z}} \gamma_i \right) \\
&\quad \geq m \cdot \left(y - \sum_{i \in \mathcal{Z}} \gamma_i \right)
\end{aligned}$$

Then using that $\gamma_i \leq m$ for every $i \in [x]$, we get

$$\sum_{i \in [x] \setminus \mathcal{Z}} \delta_i \cdot m \geq \sum_{i \in [x] \setminus \mathcal{Z}} \delta_i \cdot \gamma_i \geq m \cdot \left(y - \sum_{i \in \mathcal{Z}} \gamma_i \right) \Rightarrow \sum_{i \in [x] \setminus \mathcal{Z}} \delta_i \geq \left(y - \sum_{i \in \mathcal{Z}} \gamma_i \right).$$

□

5.6 Proof of Theorem 43

We present an attack that achieves the bound in the theorem. We assume $M > (m-1)N + 1$ and thus m-way collisions exist for every salt in $[N]$ under h. We also assume $m = O(N^c)$ for some constant c. Then our proposed (S, T, m)-AI adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ is a trivial variation of our attack proposed in the proof of theorem 40 and is as follows:

1. In *Offline stage*, $\mathcal{A}_1$ picks s random salts, denoted $\{a_1, \ldots, a_s\}$. For salt a_i (any $i \in [s]$), $\mathcal{A}_1$ zero-walks for $x := \max\{0, \lfloor (B - \log m)/2 \rfloor\}$ times to obtain a'_i and then finds $y := \min\{\log_2 m, B-1\}$-length chain of $m^{1/y}$-way collision. $\mathcal{A}_1$ stores a'_i and the corresponding chain as advice.

2. In *Online stage*, given advice from offline stage and random salt denoted a, $\mathcal{A}_2$ runs for $i \in [\lfloor T/(2x+1) \rfloor]$:

   - $\mathcal{A}_2$ queries $h(a, i)$ $\{a'_1, \ldots, a'_s\}$.
   - $\mathcal{A}_2$ zero-walks up to $2 \cdot x$ times

   After any step above, if the output is in $\{a'_1, \ldots, a'_s\}$, $\mathcal{A}_2$ can use the advice to output m-way collision on a consisting of i concatenated with appropriate number of 0s and the corresponding chain in the advice.

Analysis of the attack

Case 1: $B \leq \log m$

In this case, $x = 0$ and $\mathcal{A}_2$ stores $(B-1)$-length chain of $m^{1/(B-1)}$-way collision for each randomly chosen salt. Thus, S-bit advice can contain the salt and the corresponding chain for at least

$$s = \frac{S}{\log m(m^{1/(B-1)} \log M + \log N) + \log N} = \tilde{\Omega}\left(\frac{S}{m^{1/(B-1)}}\right)$$

salts.

This implies $\mathcal{A}_2$ can achieve an advantage of

$$\tilde{\Omega}\left(\frac{ST}{m^{1/(B-1)}N}\right).$$

Case 2: $\log m < B < 2 \log m$

In this case, $\mathcal{A}_1$ zero-walks for $\lfloor (B - \log m)/2 \rfloor$ times on each randomly chosen salt and then finds a $\log m$-length chain of 2-collisions on the output of zero-walking. Thus, the S-bit advice can contain the salt and the corresponding chain for at least

$$s = \frac{S}{\log m(2 \cdot \log M + \log N) + \log N} = \tilde{\Omega}\left(S\right)$$

salts.

This implies $\mathcal{A}_2$ can achieve an advantage of

$$\tilde{\Omega}\left(\frac{ST}{N}\right).$$

Case 3: $B \geq 2 \cdot \log m$

In this case, $\mathcal{A}_1$ zero-walks for $\lfloor (B - \log m)/2 \rfloor \geq B/4$ times on each randomly chosen salt and then finds a $\log m$-length chain of 2-collisions on the output of zero-walking. Thus, the S-bit advice, as in the previous case, can contain the salt and the corresponding chain for at least

$$s = \frac{S}{\log m(2 \cdot \log M + \log N) + \log N} = \tilde{\Omega}\left(S\right)$$

salts.

This implies $\mathcal{A}_2$ can achieve an advantage of

$$\tilde{\Omega}\left(\frac{STB}{N}\right).$$

5.7 Proof of Theorem 44

The proof for the theorem follows from the reduction presented in section 5.3.2 and the result from [2].

Chapter 6

Double Encryption

Bibliography

[1] Akshima, D. Cash, A. Drucker, and H. Wee, "Time-space tradeoffs and short collisions in merkle-damgård hash functions," in *Advances in Cryptology - CRYPTO 2020 - 40th Annual International Cryptology Conference, CRYPTO 2020, Santa Barbara, CA, USA, August 17-21, 2020, Proceedings, Part I*, D. Micciancio and T. Ristenpart, Eds., ser. Lecture Notes in Computer Science, vol. 12170, Springer, 2020, pp. 157–186. DOI: 10.1007/978-3-030-56784-2_6. [Online]. Available: https://doi.org/10.1007/978-3-030-56784-2%5C_6.

[2] Akshima, S. Guo, and Q. Liu, *Time-space lower bounds for finding collisions in merkle-damgård hash functions*, Cryptology ePrint Archive, Paper 2022/885, https://eprint.iacr.org/2022/885, 2022. [Online]. Available: https://eprint.iacr.org/2022/885.

[3] M. Hellman, "A cryptanalytic time-memory trade-off," *IEEE Trans. Inf. Theor.*, vol. 26, no. 4, pp. 401–406, Jul. 1980.

[4] D. Unruh, "Random oracles and auxiliary input," in *Annual International Cryptology Conference*, Springer, 2007, pp. 205–223.

[5] M. Bellare and P. Rogaway, "Random oracles are practical: A paradigm for designing efficient protocols," in *Proceedings of the 1st ACM Conference on Computer and Communications Security*, ser. CCS '93, Fairfax, Virginia, USA: Association for Computing Machinery, 1993, pp. 62–73, ISBN: 0897916298. DOI: 10.1145/168588.168596. [Online]. Available: https://doi.org/10.1145/168588.168596.

[6] S. Coretti, Y. Dodis, S. Guo, and J. Steinberger, "Random oracles and non-uniformity," in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, Springer, 2018, pp. 227–258.

[7] Y. Dodis, S. Guo, and J. Katz, "Fixing cracks in the concrete: Random oracles with auxiliary input, revisited," in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, Springer, 2017, pp. 473–495.

[8] R. Gennaro and L. Trevisan, *Lower bounds on the efficiency of generic cryptographic constructions*, Cryptology ePrint Archive, Paper 2000/017, https://eprint.iacr.org/2000/017, 2000. [Online]. Available: https://eprint.iacr.org/2000/017.

[9] R. Gennaro, Y. Gertner, J. Katz, and L. Trevisan, "Bounds on the efficiency of generic cryptographic constructions," *SIAM journal on Computing*, vol. 35, no. 1, pp. 217–246, 2005.

[10] P. Morin, W. Mulzer, and T. Reddad, "Encoding arguments," *ACM Computing Surveys (CSUR)*, vol. 50, no. 3, pp. 1–36, 2017.

[11] R. Impagliazzo and V. Kabanets, "Constructive proofs of concentration bounds," in *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, 13th International Workshop, APPROX 2010, and 14th International Workshop, RANDOM 2010, Barcelona, Spain, September 1-3, 2010. Proceedings*, 2010, pp. 617–631.

[12] R. Impagliazzo, "Relativized separations of worst-case and average-case complexities for np," in *Proceedings of the 2011 IEEE 26th Annual Conference on Computational Complexity*, ser. CCC '11, IEEE Computer Society, 2011, pp. 104–114, ISBN: 978-0-7695-4411-3. DOI: 10.1109/CCC.2011.34. [Online]. Available: https://doi.org/10.1109/CCC.2011.34.

[13] M. Zimand, "How to privatize random bits," University of Rochester, Tech. Rep., Apr. 1996.

[14] A. Ghoshal and I. Komargodski, *On time-space tradeoffs for bounded-length collisions in merkle-damgård hashing*, Cryptology ePrint Archive, Paper 2022/309, https://eprint.iacr.org/2022/309, 2022. [Online]. Available: https://eprint.iacr.org/2022/309.

[15] K.-M. Chung, S. Guo, Q. Liu, and L. Qian, "Tight quantum time-space tradeoffs for function inversion," in *2020 IEEE 61st Annual Symposium on Foundations of Computer Science (FOCS)*, IEEE, 2020, pp. 673–684.

[16] S. Guo, Q. Li, Q. Liu, and J. Zhang, "Unifying presampling via concentration bounds," in *Theory of Cryptography Conference*, Springer, 2021, pp. 177–208.

[17] A. Fiat and M. Naor, "Rigorous time/space tradeoffs for inverting functions," in *Proceedings of the twenty-third annual ACM symposium on Theory of computing*, 1991, pp. 534–541.

[18] S. Coretti, Y. Dodis, and S. Guo, "Non-uniform bounds in the random-permutation, ideal-cipher, and generic-group models," in *Annual International Cryptology Conference*, Springer, 2018, pp. 693–721.

[19] A. De, L. Trevisan, and M. Tulsiani, "Time space tradeoffs for attacks against one-way functions and prgs," in *Annual Cryptology Conference*, Springer, 2010, pp. 649–665.

[20] D. Chawin, I. Haitner, and N. Mazor, "Lower bounds on the time/memory tradeoff of function inversion," in *Theory of Cryptography - 18th International Conference, TCC 2020, Durham, NC, USA, November 16-19, 2020, Proceedings, Part III*, 2020, pp. 305–334. DOI: 10.1007/978-3-030-64381-2_11. [Online]. Available: https://doi.org/10.1007/978-3-030-64381-2%5C_11.

[21] H. Corrigan-Gibbs and D. Kogan, "The discrete-logarithm problem with preprocessing," in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, Springer, 2018, pp. 415–447.

[22] ——, "The function-inversion problem: Barriers and opportunities," in *Theory of Cryptography Conference*, Springer, 2019, pp. 393–421.

[23] N. Gravin, S. Guo, T. C. Kwok, and P. Lu, "Concentration bounds for almost k-wise independence with applications to non-uniform security," in *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms, SODA 2021, Virtual Conference, January 10 - 13, 2021*, 2021, pp. 2404–2423. DOI: 10.1137/1.9781611976465.143. [Online]. Available: https://doi.org/10.1137/1.9781611976465.143.

[24] B. Auerbach, D. Cash, M. Fersch, and E. Kiltz, "Memory-tight reductions," in *Annual International Cryptology Conference*, Springer, 2017, pp. 101–132.

[25] Y. Wang, T. Matsuda, G. Hanaoka, and K. Tanaka, "Memory lower bounds of reductions revisited," in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, Springer, 2018, pp. 61–90.

[26] A. Ghoshal, J. Jaeger, and S. Tessaro, "The memory-tightness of authenticated encryption," in *Annual International Cryptology Conference*, Springer, 2020, pp. 127–156.

[27] A. Ghoshal and S. Tessaro, "On the memory-tightness of hashed elgamal," in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, Springer, 2020, pp. 33–62.

[28] D. Diemert, K. Gellert, T. Jager, and L. Lyu, *Digital signatures with memory-tight security in the multi-challenge setting*, Cryptology ePrint Archive, Report 2021/1220, https://eprint.iacr.org/2021/1220, 2021.

[29] R. Bhattacharyya, "Memory-tight reductions for practical key encapsulation mechanisms," 2020, pp. 249–278. DOI: 10.1007/978-3-030-45374-9_9.

[30] A. Ghoshal, R. Ghosal, J. Jaeger, and S. Tessaro, *Hiding in plain sight: Memory-tight proofs via randomness programming*, Cryptology ePrint Archive, Report 2021/1409, https://eprint.iacr.org/2021/1409, 2021.

[31] S. Tessaro and A. Thiruvengadam, "Provable time-memory trade-offs: Symmetric cryptography against memory-bounded adversaries," in *Theory of Cryptography Conference*, Springer, 2018, pp. 3–32.

[32] J. Jaeger and S. Tessaro, "Tight time-memory trade-offs for symmetric encryption," in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, Springer, 2019, pp. 467–497.

[33] I. Dinur, "On the streaming indistinguishability of a random permutation and a random function," in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, Springer, 2020, pp. 433–460.

[34] I. Shahaf, O. Ordentlich, and G. Segev, "An information-theoretic proof of the streaming switching lemma for symmetric encryption," in *IEEE International Symposium on Information Theory, ISIT 2020, Los Angeles, CA, USA, June 21-26, 2020*, IEEE, 2020, pp. 858–863.

[35] I. Dinur, "Tight time-space lower bounds for finding multiple collision pairs and their applications," in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, Springer, 2020, pp. 405–434.

[36] W. Dai, S. Tessaro, and X. Zhang, "Super-linear time-memory trade-offs for symmetric encryption," in *Theory of Cryptography Conference*, Springer, 2020, pp. 335–365.

[37] S. Coretti, Y. Dodis, and S. Guo, "Non-uniform bounds in the random-permutation, ideal-cipher, and generic-group models," in *Advances in Cryptology - CRYPTO 2018 - 38th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2018, Proceedings, Part I*, H. Shacham and A. Boldyreva, Eds., ser. Lecture Notes in Computer Science, vol. 10991, Springer, 2018, pp. 693–721. DOI: 10.1007/978-3-319-96884-1_23. [Online]. Available: https://doi.org/10.1007/978-3-319-96884-1%5C_23.