The Collision Intractability of MDC-2 in the Ideal-Cipher Model

John P. Steinberger*

August 29, 2006

Abstract

We provide the first proof of security for MDC-2, the most well-known construction for turning an n-bit blockcipher into a $2n$-bit cryptographic hash function. Our result, which is in the ideal-cipher model, shows that MDC-2, when built from a blockcipher having blocklength and keylength n, has security much better than that delivered by any hash function that has an n-bit output. When the blocklength and keylength are $n = 128$ bits, as with MDC-2 based on AES-128, an adversary that asks fewer than $2^{74.9}$ queries usually cannot find a collision.

Keywords: Blockcipher modes, cryptographic hash functions, ideal-cipher model, MDC-2.

1 Introduction

OVERVIEW. A double-length hash-function uses an n-bit blockcipher as the building block by which it maps (possibly long) strings to $2n$-bit ones. The classical double-length hash-function is MDC-2, illustrated in Figure 1. This nearly 20-year-old technique [5, 22] is specified in the ANSI X9.31 and ISO/IEC 10118-2 standards [1, 13], and it is implemented in popular libraries and toolkits, such as OpenSSL.

This paper gives the first proof of security for MDC-2. Our result establishes that when MDC-2 is based on an ideal blockcipher with keylength and blocklength of n bits, the adversary must ask well over $2^{n/2}$ queries to find a collision. In partiuclar, for $n = 128$, no adversry can find a collision with so much as a 50% chance if it asks fewer than $2^{74.9}$ forward-or-backward queries of a 128-bit blackbox-modeled blockcipher.

Getting a collision-resistance bound of $2^{74.9}$ queries when $n = 128$ is still far from the optimum one might hope for, which is a bound of 2^{128} queries for an output of $2n = 256$ bits (the birthday bound). But obtaining *any* bound above 2^{64} (a trivial lower bound) has proved elusive to researchers thus far, given the combinatorial complexity of the problem. We point out further down some of the characteristics of MDC-2 which make its security harder to tackle than that of other hash functions.

WHAT IS MDC-2? Traditionally, MDC-2 was instantiated using DES, and some people may understand MDC-2 to *mean* MDC-2 based on DES. This is not our meaning. Indeed this paper assumes a common keylength and blocklength n bits, and so our results don't directly apply to MDC-2 based on DES. (We assume that, with signficant work, one could extend our analysis to handle the DES parameters of 56-bit keys and 64-bit blocks, but we haven't done this.) In this paper we consider MDC-2 using a blockcipher E: $\{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$ with equal-length blocks and keys. We make this assumption both for simplicity while preserving contemporary applicability: eliminating "bit-dropping" makes the algorithm cleaner, while the usage of MDC-2 that people nowadays envisage is with the blockcipher AES-128 [28]. All future mention of MDC-2 in this paper assumes equal blocklength and keylength.

* Department of Mathematics, University of California, Davis, California 95616 USA. E-mail: jpsteinb@math.ucdavis.edu WWW: www.math.ucdavis.edu/~jpsteinb

Algorithm MDC2$^E(X)$
$X_1 \cdots X_m \leftarrow X$ where $|X_i| = n$
for $i \leftarrow 1$ **to** m **do**
    $V_i \leftarrow X_i \oplus E(A_i, X_i)$
    $W_i \leftarrow X_i \oplus E(B_i, X_i)$
    $(A_{i+1}, B_{i+1}) \leftarrow (V_i^L W_i^R, W_i^L V_i^R)$
return $V_m W_m$

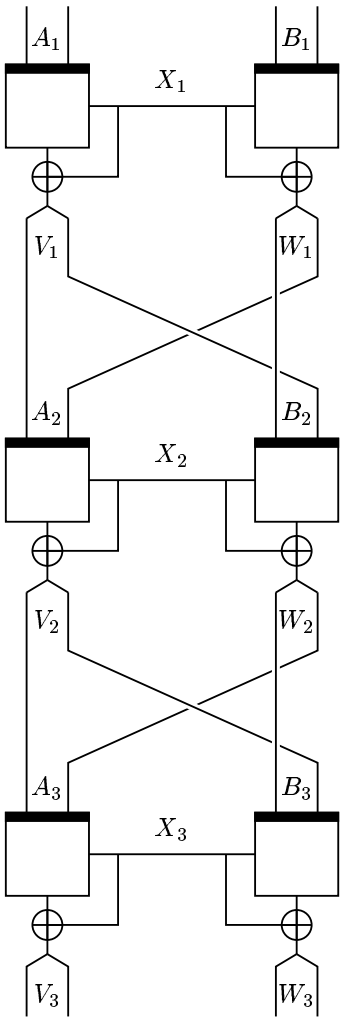


Figure 1: **Left:** Definition of the MDC-2 algorithm based on a blockcipher E with key length, word length, and output length n. The message being acted on is $X = X_1 \cdots X_m$ where $m \geq 1$ and $|X_i| = n$. Strings A_1 and B_1 are distinct n-bit constants. For an even-length sting S we let S^L and S^R be its left and right half. **Right:** Illustration of the algorithm acting on a three-block messsage $X = X_1 X_2 X_3$. The resulting hash is $H(X) = V_3\, W_3$. The darkened edge of the box representing the blockcipher indicates the input that is the key.

The MDC-2 algorithm is simple and elegant: building on the usual Merkle-Damgård approach [6, 21], the compression function uses two parallel invocations of the Matyas-Meyer-Oseas compression function [20] and then swaps the right halves of the outputs. It is defined and illustrated in Figure 1. It is easy to see that the algorithm doesn't work (that is, it admits efficient attacks) if it is "over-simplified" by dropping the left/right swapping, the feed-forward XOR, or both.

The version of MDC-2 that we consider does not incorporate a "bit fixing" step like replacing the leftmost bit of each left-column blockcipher key in Figure 1 with a 0-bit and replacing the leftmost bit of each right-column blockcipher key with a 1-bit. Such bit-fixing was employed in MDC2-DES to overcome the key-complementation property of the primitive and also, conceivably, as a security measure.

We also comment that in the version of MDC-2 that we consider, no length-annotation or padding is used, and the domain is correspondingly restricted to $(\{0,1\}^n)^+$. It is easy and customary to use padding and length-annotation to extend MDC-2 to handle a domain of any string of less than 2^n bits. Provable-security

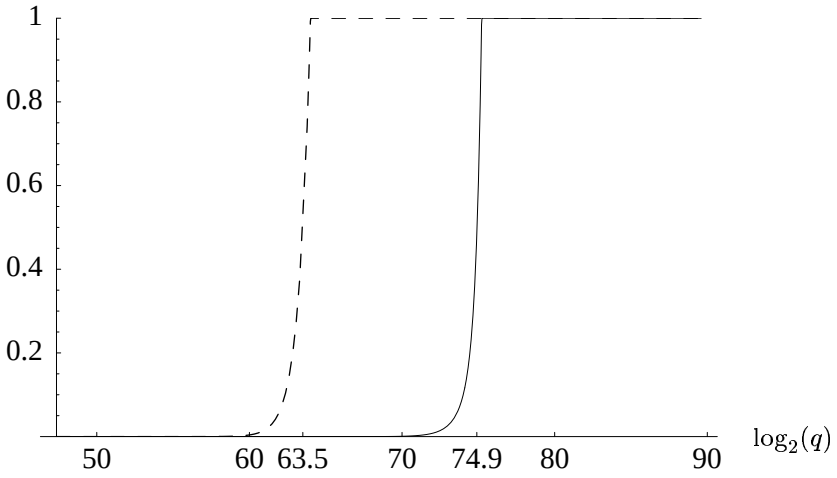


Figure 2: Our upper bound on $\mathsf{Adv}_n^{\mathrm{MDC2}}(q)$ as a function of q for $n = 128$ (solid line) compared to the previous best upper bound of $q(q+1)/2^n$ (dotted line).

results immediately extend: a collision-intractability result for the $(\{0,1\}^n)^+$ version of a hash function will always lift to give essentially the same bound for the $\{0,1\}^{<n}$ domain version one gets after padding and length-annotation.

OUR RESULTS. We work in the ideal-cipher model, as in [4,8,15]. This is the customary model for proving the security of a blockcipher-based hash function. In the ideal-cipher model the underlying primitive, a blockcipher E, is modeled as a family of random permutations with one random permutation chosen independently for each key K. The adversary may make a query $E(K, X)$ to discover the corresponding value $Y = E(K, X)$, or the adversary may make a query $E^{-1}(K, Y)$ so as to learn the corresponding value $X = E^{-1}(K, Y)$ for which $E(K, X) = Y$. We are interested in the chance that an adversary can find a collision, namely a pair of distinct messages that collide under $\mathrm{MDC2}^E$, by asking q queries to such a primitive (more formal definitions are given below).

It is easy to show that finding a collision for MDC2 implies finding X, K, X', K' with $(X, K) \neq (X', K')$ such that $E(K, X) \oplus X = E(K', X') \oplus X'$. From this it follows (see e.g. [4]) that an adversary's chance of finding a collision in q queries is less than $q(q+1)/2^n \approx q^2/2^n$ where $n = |X| = |K|$ is the block size. This is a trivial upper bound, equal to the best possible security of a hash function with output length n [4] (also known as the "birthday bound"). Ideally one would like to prove a similar birthday bound of $q^2/2^{2n}$ for MDC-2, since MDC-2 has output length $2n$. However, despite the lack of attacks on MDC-2, no one has even been able to exhibit an improvement on the bound of $q^2/2^n$. In this paper we give the first improvement by showing that the adversary has chance $O(q^5/2^{3n})$ of finding an attack for large enough n, and that an adversary needs at least $q \approx 2^{3n/5}$ queries to have an even chance of finding a collision for large enough n. For example when $n = 128$ (the main case of interest) we show that an adversary needs at least $q = 2^{74.9}$ queries to have an even chance of obtaining a collision, which is over 2^{10} greater than the trivial bound of $2^{63.5}$. Figure 2 shows our upper bound as function of q for the case $n = 128$.

We should mention the fact that there are well-known limitiations on the ideal-cipher model as a way to evidence security of a blockcipher-based construction, these limitiations stemming from that fact that the approach models, not defines, the security of the underlying blockcipher. In particular, we are not claiming any particular cryptographic property that, say, AES must have in order for $\mathrm{MDC2}^{\mathrm{AES128}}$ to be secure. A blackbox-model result comes closer to saying that if one can find collisions in $\mathrm{MDC2}^{\mathrm{AES128}}$ then there is some particular and damaging sense in which AES128 is not behaving like an idealized blockcipher.

OTHER TYPES OF SECURITY. There are other notions of security for hash functions besides collision intractability such as *preimage resistance* and *second preimage resistance*. Roughly speaking a hash function f is preimage resistant if given y in the range of f it is difficult to find x such that $f(x) = y$ and is second preimage resistant if given x and y such that $f(x) = y$ it is difficult to find $x' \neq x$ such that $f(x') = y$.

Preimage resistance and second preimage resistance are generally considered weaker than collision resistance.

In the case of MDC-2 it is easy to see that breaking preimage resistance implies the ability to find, given some word Y of length n, a pair (X, K) such that $E(K, X) \oplus X = Y$, which is hard to do (more precisely, the adversary's chance of achieving this in q queries is upper bounded by $q/(2^n - q)$ in the ideal-cipher model, which is a secure bound already for $n = 128$). Similarly breaking second preimage resistance[1] implies the ability to find, given some triple (X, K, Y) such that $E(X, K) \oplus X = Y$, a pair $(X', K') \neq (X, K)$ such that $E(X', K') \oplus X' = Y$, which is equally hard to do. The questions of preimage resistance and second preimage resistance are therefore not so interesting[2] for MDC-2, and our focus in this paper is on collision resistance.

THE RESURGENCE OF INTEREST IN BLOCKCIPHER-BASED HASHING. Since their initial design by Rivest [25], MD4-family hash functions (eg, MD4, MD5, and SHA-1) have dominated cryptographic practice. But in recent years a sequence of attacks on MD4-family hash-functions by Dobbertin, Wang, and others [2,7,29,30], has led to a generalized sense of concern about the MD4-approach. The most natural place to look for an alternative is in blockcipher-based constructions, which in fact predate the MD4-apprach [26]. Blockcipher-based constructions are based on principles unrelated to MD4-style functions. Moreover recent attacks by Joux [14] on concatenated hash functions have led to renewed interest in sound design principles for double block length hash functions [19,24]. As a result many authors [9–11,19,23,24,27] have recently proposed new double block-length constructions based either on an ideal blockcipher primitive or on an ideal compression function primitive. Other authors have shown new attacks or limitations on the efficiency or security of such constructions [3, 12, 16, 27].

Some of the newly proposed blockcipher-based double block hash functions have birthday-type collision resistance [10, 11] but suffer from drawbacks such as long key lengths. The original blockcipher-based constructions of Merkle [21] were also double block length (in essence) and with best possible collision resistance but had very low rates (the "rate" of a blockcipher-based hash function measures the number of blocks hashed per blockcipher invocation, and constitutes a rough measure of its efficiency). Overall, MDC-2 remains one of the more attractive candidates for a double length blockcipher-based hash function given its high rate, small keys, and lack of known attacks.

A common template for the constructions with proven best possible collision resistance in either the ideal cipher or ideal compression function model is the iteration of a compression function $F : \{0,1\}^{2n+m} \to \{0,1\}^{2n}$ where m is the message block length and where F is defined by $F(V, W, X) = f_1(V, W, X) \| f_2(V, W, X)$ where $f_1 : \{0,1\}^{2n+m} \to \{0,1\}^n$, $f_2 : \{0,1\}^{2n+m} \to \{0,1\}^n$ are two component functions made from the ideal primitive (note MDC-2 can be defined this way this way with $m = n$ and $f_1(V, W, X) = E(V^L W^R, X) \oplus X$, $f_2(V, W, X) = E(V^R W^L, X) \oplus X$). All the constructions that we are aware of for which it is possible to prove birthday-type collision security have the common trait that the inputs necessary to compute f_1 can be inferred from the inputs necessary to compute f_2 and vice-versa. The reader may contrast this with MDC-2, for which the inputs necessary to compute f_2, namely V^R, W^L and X, cannot be deduced from the inputs necessary to compute f_1, which are V^L, W^R and X. This independence between the left and right-hand sides of MDC-2 seems to account in part for the difficulty of proving its security.

2 Preliminaries

Let $\mathsf{Bloc}(n)$ be the set of functions E: $\{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$ such that $E(K, \cdot) = E_K(\cdot)$ is a permutation on $\{0,1\}^n$. Given a function $E \in \mathsf{Bloc}(n)$ we define $\mathrm{MDC2}^E$: $(\{0,1\}^n)^+ \to \{0,1\}^{2n}$ by the algorithm of Fig. 1. The hash of a word X where $|X|$ is a multiple of n by $\mathrm{MDC2}^E$ is denoted by $\mathrm{MDC2}^E(X)$.

An adversary is an (always-halting) algorithm with access to an oracle. If A is a deterministic adversary and $E \in \mathsf{Bloc}(n)$ is a keyed permutation then the *query history* of A^E, denoted $\mathcal{Q} = \mathcal{Q}A^E$, is the tuple $(Q_1, \ldots, Q_q)$ where $Q_i = (X_i, K_i, Y_i)$ and A asks total of q queries and the i^{th} query is either a "forward" query $(K_i, X_i)_{\text{fwd}}$, this query being answered by $Y_i = E_{K_i}(X_i)$, or a "backward" query $(K_i, Y_i)_{\text{bwd}}$, this query being answered by $X_i = E_{K_i}^{-1}(Y_i)$. We can assume the adversary never asks a query for which it already knows the answer, such as by asking the same query twice or by asking $(K_i, X_i)_{\text{fwd}}$ and then asking

[1] A proper definition of second preimage resistance includes the definition of a sampling mechanism for x; we could assume that x is chosen at random from all strings of length $\leq cn$ for some small constant c.

[2] Bounds of the order $q/2^n$ are indeed considered secure for $n = 128$, but are probably not the best achievable; more work might be done to obtain matching upper and lower bounds on preimage resistance and second preimage resistance.

$(K_i, Y_i)_{\text{bwd}}$ (thus the answer to any query is always a randomly chosen from a pool of size at least $2^n - q$ where q is the number of queries previously made by the adversary). We will often resort to the notational convention that (K_i, X_i) denotes a forward query and that (K_i, Y_i) denotes a backward query, without possible confusion.

If (X_i, K_i, Y_i) is an element of the query history then we refer to X_i as the "word input" of the query, to K_i as the "key" of the query, and to Y_i as the "output" of the query. The quantity $X_i \oplus Y_i$ is called the "XOR output" of the query.

The adversary's goal is to output a pair of nonempty strings X, X' such that $X \neq X'$ and $\text{MDC2}^E(X) = \text{MDC2}^E(X')$. Moreover we impose the condition that the adversary must have made all queries necessary to compute $\text{MDC2}^E(X)$, $\text{MDC2}^E(X')$. This restriction is reasonable since otherwise the adversary can output very long words X, X' where $\text{MDC2}^E(X) = \text{MDC2}^E(X')$ with good probability but where computing $\text{MDC2}^E(X)$, $\text{MDC2}^E(X')$ is infeasible in practicality. (For example, the adversary could simply output 0^{Kn}, 0^{2Kn} where K is the lcm of all numbers between 1 and 2^{2n} and have probability 1 of obtaining a collision without making any queries, but this isn't a reasonable type of attack.)

Since we may tell simply from the adversary's query history $\mathcal{Q}$ whether it is possible for the adversary to output words $X \neq X'$ such that $\text{MDC2}^E(X) = \text{MDC2}^E(X')$ and such that $\mathcal{Q}$ contains all the queries necessary for the computation of $\text{MDC2}^E(X)$, $\text{MDC2}(X')$, we will in fact dispense the adversary from having to output X, X' and simply determine whether the adversary has been successful or not by examining its query history $\mathcal{Q}$. Formally, we say that $\mathsf{Coll}^{\text{MDC2}^E}(\mathcal{Q})$ holds if there are two distinct nonempty words X, X' of lengths divisible by n such that $\text{MDC2}^E(X) = \text{MDC2}^E(X')$ and such that $\mathcal{Q}$ contains all the queries necessary to compute $\text{MDC2}^E(X)$, $\text{MDC2}^E(X')$ as defined by the algorithm of Fig. 1. The goal of the adversary A is thus to make some sequence $\mathcal{Q} = \mathcal{Q}A^E$ of q or fewer queries such that $\mathsf{Coll}^{\text{MDC2}^E}(\mathcal{Q})$. The adversary's ability to break MDC2 in q or fewer queries is measured by

$$\mathsf{Adv}_n^{\text{MDC2}}(A) = \Pr[E \xleftarrow{\$} \mathsf{Bloc}(n), \mathcal{Q} \leftarrow A^E; \mathsf{Coll}^{\text{MDC2}^E}(\mathcal{Q})].$$

We let $\mathsf{Adv}_n^{\text{MDC2}}(q)$ be the max over all adversaries A making at most q queries of $\mathsf{Adv}_n^{\text{MDC2}}(A)$. Our goal is thus to upper bound $\mathsf{Adv}_n^{\text{MDC2}}(q)$. Since q is always less than the total possible number of queries which can be made to E, we can assume without loss of generality that A asks exactly q queries and thus that $|\mathcal{Q}A^E| = q$.

Say that numbers n and q have been fixed as well as an adversary A such that $|\mathcal{Q}A^E| = q$ for all $E \in \mathsf{Bloc}(n)$. If P is any predicate which may be true or false for a sequence of queries $\mathcal{Q}$ (such as for example $\mathsf{Coll}^{\text{MDC2}^E}(\mathcal{Q})$) then we write $\Pr[\mathsf{P}(\mathcal{Q})]$ as a shorthand for $\Pr[E \xleftarrow{\$} \mathsf{Bloc}(n), \mathcal{Q} \leftarrow A^E; \mathsf{P}(\mathcal{Q})]$. With this notation we have $\mathsf{Adv}_n^{\text{MDC2}}(A) = \Pr[\mathsf{Coll}^{\text{MDC2}^E}(\mathcal{Q})]$. We will often use this simpler notation to avoid over-complicating our formulas.

By our definition $\mathsf{Coll}^{\text{MDC2}^E}(\mathcal{Q})$ holds if there are two nonempty words $X \neq X'$ such that $\text{MDC2}^E(X) = \text{MDC2}^E(X')$ and $\mathcal{Q}$ contains all the queries necessary to compute $\text{MDC2}^E(X)$, $\text{MDC2}^E(X')$. We say the pair X, X' forms an *earliest possible collision* if there do no exist two nonempty exist words $Y \neq Y'$ of length 0 mod n such that Y is a prefix of X, Y' is a prefix of X' and $\text{MDC2}^E(Y) = \text{MDC2}^E(Y')$. Obviously if $\mathsf{Coll}^{\text{MDC2}^E}(\mathcal{Q})$ holds then it holds for some X, X' that are an earliest possible collision. We will use this fact later to upper bound $\mathsf{Adv}_n^{\text{MDC2}}(q)$.

Our upper bound can be stated in varying degrees of generality and comprehensibility. The most general and least comprehensible statement of our upper bound is the following:

Theorem 1. *Let n, q be natural numbers with $n \geq q$. Let $N = 2^n$, $N' = n - q$ and let m_a, m_b, m_c be any positive numbers with $eqN^{\frac{1}{2}}/N' \leq m_b$, $eq/N' \leq m_c$. Finally let $M_b = m_b N'/qN^{\frac{1}{2}}$, $M_c = m_c N'/q$ and $N'' = N'(N^{\frac{1}{2}} - m_b)/N^{\frac{1}{2}}$. Then*

$$\begin{aligned}
\mathsf{Adv}_n^{\text{MDC2}}(q) \leq\ & \frac{q^2}{m_a N'} + 2qN^{\frac{1}{2}} e^{qN^{\frac{1}{2}} M_b(1-\ln(M_b))/N'} + qNe^{qM_c(1-\ln(M_c))/N'} + && (1)\\
& q(m_a^2 + m_a m_b^2 + m_b^4)/N' + && (2)\\
& q(4m_a m_b)/N' + q(2m_a m_b)/N'' +\\
& q(m_b^2 m_c + 5m_b^2 + m_a m_c + 6m_a)/N' + q(4m_a + 8m_b^2)/N'' +\\
& q(4 + 10m_b + 2m_b m_c)/N'' + 3q/N' + 4q/N'' + q^2/N'^2
\end{aligned}$$

q	$\mathsf{Adv}^{\mathrm{MDC2}}_{128}(q) \leq$	m_a	m_b	m_c
2^{64}	7.57×10^{-7}	2.64×10^6	44.01	3.7147
$2^{68.22}$	10^{-4}	7.01×10^6	128.09	3.9448
$2^{72.19}$	$1/100$	1.75×10^7	898.95	4.1899
$2^{74.00}$	$1/10$	2.66×10^7	2902.32	4.3082
$2^{74.72}$	$1/3$	3.14×10^7	4687.89	4.3523
$2^{74.91}$	$1/2$	3.29×10^7	5355.49	4.3640
$2^{75.21}$	1	–	–	–

Table 1: Upper bounds on $\mathsf{Adv}^{\mathrm{MDC2}}_{128}(q)$ given by Theorem 1.

What Theorem 1 concretely means for $n = 128$ is shown in Table 1 and Fig. 2. Table 1 shows specific numerical upper bounds for $\mathsf{Adv}^{\mathrm{MDC2}}_{128}(q)$ for various values of q. The threshold value where Theorem 1 gives an upper bound of $1/2$ is $q = 2^{74.91}$ (to be compared with the previous best threshold of $q = 2^{63.5}$). For each value of q we also show the values of m_a, m_b, m_c which yield the stated upper bound. Fig. 2 plots our upper bounds on $\mathsf{Adv}^{\mathrm{MDC2}}_{128}(q)$ as a function of q, compared to the previous upper bound of $q(q+1)/N$. The method for optimizing m_a, m_b, m_c for given values of n, q in order to obtain the best bound on $\mathsf{Adv}^{\mathrm{MDC2}}_{n}(q)$ is explained in the last section. In that appendix we also show that Theorem 1 implies the following:

Theorem 2. *Let $q = N^{\frac{3}{5}-\epsilon}$ where $\epsilon > 0$ and $N = 2^n$. Then $\mathsf{Adv}^{\mathrm{MDC2}}_{n}(q) \to 0$ as $n \to \infty$.*

Asymptotically as $n \to \infty$, thus, our bound for $\mathsf{Adv}^{\mathrm{MDC2}}_{n}(q)$ behaves like the function $\min(1, q^5/2^{3n})$, though the two functions still look significantly different for $n = 128$ (e.g. $q^5/2^{3n}$ has a threshold of 76.6 for $n = 128$ whereas our bound on $\mathsf{Adv}^{\mathrm{MDC2}}_{n}(q)$ has a threshold of 74.9). Though the two functions behave the same asymptotically there does not seem to be any good closed form relating our bound on $\mathsf{Adv}^{\mathrm{MDC2}}_{n}(q)$ to the function $q^5/2^{3n}$.

3 Analysis

Fix numbers n, q and an adversary A. We upper bound $\Pr[\mathsf{Adv}^{\mathrm{MDC2}}_{n}(\mathcal{Q})]$ by exhibiting predicates $\mathsf{Win0}(\mathcal{Q})$, ..., $\mathsf{Win8}(\mathcal{Q})$ such that $\mathsf{Adv}^{\mathrm{MDC2}}_{n}(\mathcal{Q}) \implies \mathsf{Win0}(\mathcal{Q}) \vee \ldots \vee \mathsf{Win8}(\mathcal{Q})$ and then by upper bounding separately the probabilities $\Pr[\mathsf{Win0}(\mathcal{Q})]$, ..., $\Pr[\mathsf{Win8}(\mathcal{Q})]$. Then obviously $\Pr[\mathsf{Adv}^{\mathrm{MDC2}}_{n}(\mathcal{Q})] \leq \Pr[\mathsf{Win0}(\mathcal{Q})] + \cdots + \Pr[\mathsf{Win8}(\mathcal{Q})]$.

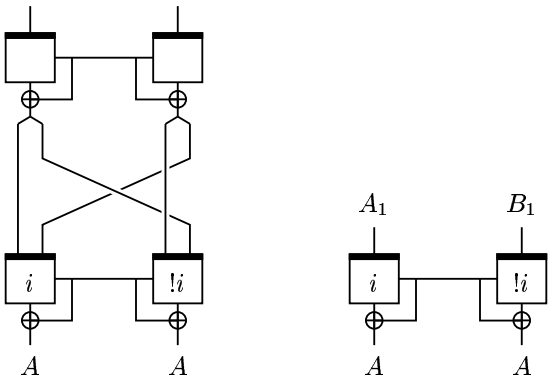
To state the predicates $\mathsf{Win0}(\mathcal{Q})$, ..., $\mathsf{Win8}(\mathcal{Q})$ we need some definitions; a, b, b^L, b^R, c are functions defined on query sequences of length q, as follows:

$$
\begin{aligned}
a(\mathcal{Q}) &= |\{(i,j) \in [1 \ldots q]^2 : i \neq j, X_i \oplus Y_i = X_j \oplus Y_j\}| \text{ is the number of ordered pairs of distinct queries in } \mathcal{Q} \text{ with same XOR outputs} \\
b^L(\mathcal{Q}) &= \max_{Y \in \{0,1\}^{n/2}} |\{i : (X_i \oplus Y_i)^L = Y\}| \text{ is the maximum size of a set of queries in } \mathcal{Q} \text{ whose XOR outputs all have the same left } n/2 \text{ bits} \\
b^R(\mathcal{Q}) &= \max_{Y \in \{0,1\}^{n/2}} |\{i : (X_i \oplus Y_i)^R = Y\}| \text{ is the maximum size of a set of queries in } \mathcal{Q} \text{ whose XOR outputs all have the same right } n/2 \text{ bits} \\
b(\mathcal{Q}) &= \max(b^L(\mathcal{Q}), b^R(\mathcal{Q})) \\
c(\mathcal{Q}) &= \max_{Y \in \{0,1\}^{n}} |\{i : X_i \oplus Y_i = Y\}| \text{ is the maximum size of a set of queries in } \mathcal{Q} \text{ whose XOR outputs are all the same}
\end{aligned}
$$

The event $\mathsf{Win0}(\mathcal{Q})$ is simply defined by

$$\mathsf{Win0}(\mathcal{Q}) = (a(\mathcal{Q}) > m_a) \vee (b(\mathcal{Q}) > m_b) \vee (c(\mathcal{Q}) > m_c)$$

where m_a, m_b, m_c are the constants from Theorem 1 (and that are chosen later depending on n and q—see Appendix B for how best to choose m_a, m_b, m_c). Thus as m_a, m_b, m_c are chosen larger $\Pr[\mathsf{Win0}(\mathcal{Q})]$ diminishes.



Configuration $1a$ Configuration $1b$

Figure 3: The configurations defining $\mathsf{Win1}(\mathcal{Q})$.

The events $\mathsf{Win1}(\mathcal{Q})$, ..., $\mathsf{Win8}(\mathcal{Q})$ are slightly different in nature from the event $\mathsf{Win0}(\mathcal{Q})$; they concern the feasibility of fitting certain subconfigurations of MDC-2 using queries from $\mathcal{Q} = (X_1, K_1, Y_1)$, ..., (X_q, K_q, Y_q). Take for example the configuration $1a$ of Fig. 3. In this configuration, the two strings marked 'A' are equal and the queries marked i, $!i$ are different. We now define what it means for four (or possibly fewer) queries from $\mathcal{Q}$ to "fit" configuration $1a$. We say that $\mathsf{Fit}_{1a}(i, i', j, k)$ holds if

$$(i \neq i') \wedge (X_i = X_{i'}) \wedge (X_i \oplus Y_i = X_{i'} \oplus Y_{i'}) \wedge$$
$$(X_j = X_k) \wedge ((X_j \oplus Y_j)^L = K_i^L) \wedge ((X_j \oplus Y_j)^R = K_{i'}^R) \wedge$$
$$((X_k \oplus Y_k)^L = K_{i'}^L) \wedge ((X_k \oplus Y_k)^R = K_i^R)$$

and we say that $\mathsf{ExistsFit}_{1a}(\mathcal{Q})$ holds if there exist $i, i', j, k \in [1\,..\,q]$ such that $\mathsf{Fit}_{1a}(i, i', j, k)$. The predicates $\mathsf{ExistsFit}_{1b}$, $\mathsf{ExistsFit}_2$, $\mathsf{ExistsFit}_3$, $\mathsf{ExistsFit}_{4a}$, $\mathsf{ExistsFit}_{4b}$, $\mathsf{ExistsFit}_{6a}$, $\mathsf{ExistsFit}_{6b}$, $\mathsf{ExistsFit}_{6c}$, $\mathsf{ExistsFit}_{6d}$, $\mathsf{ExistsFit}_{7a}$, $\mathsf{ExistsFit}_{7b}$, whose configurations are shown in Appendix A, are likewise defined (in these configurations strings marked by the same letter are equal but strings marked with different letters may or may not be equal; likewise queries marked i, $!i$ are different but queries marked i, j may be the same). Note that $\mathsf{ExistsFit}_{6a} = \mathsf{ExistsFit}_{6b}$ and that $\mathsf{ExistsFit}_{6c} = \mathsf{ExistsFit}_{6d}$; configurations $6b$, $6d$ are provided to facilitate referencing. An additional notation is required to indicate inequality between queries in configurations 5 and 8 (Figs. 8, 11). In these configurations, pairs of queries from the bottom row that do not both contain a '1' or both contain a '0' (namely, queries with different labels) are presumed different; there are no constraints relating top row to bottom row queries, and queries with the same label are not presumed equal (see Fig. 4 for an explanation of "top row", "bottom row"). The predicates $\mathsf{ExistsFit}_5(\mathcal{Q})$, $\mathsf{ExistsFit}_8(\mathcal{Q})$ then denote the existence of a set of queries in $\mathcal{Q}$ fitting respectively configurations 5 and 8 under these constraints.

We also let $\mathsf{NotWin}\,j = \overline{\mathsf{Win0}(\mathcal{Q})} \vee \cdots \vee \overline{\mathsf{Win}\,j(\mathcal{Q})}$ for $0 \leq j < 8$. We now define:

$$\mathsf{Win1}(\mathcal{Q}) = \mathsf{NotWin0}(\mathcal{Q}) \wedge \big(\mathsf{ExistsFit}_{1a}(\mathcal{Q}) \vee \mathsf{ExistsFit}_{1b}(\mathcal{Q})\big)$$

$$\mathsf{Win2}(\mathcal{Q}) = \mathsf{NotWin1}(\mathcal{Q}) \wedge \mathsf{ExistsFit}_2(\mathcal{Q})$$

$$\mathsf{Win3}(\mathcal{Q}) = \mathsf{NotWin2}(\mathcal{Q}) \wedge \mathsf{ExistsFit}_3(\mathcal{Q})$$

$$\mathsf{Win4}(\mathcal{Q}) = \mathsf{NotWin3}(\mathcal{Q}) \wedge \big(\mathsf{ExistsFit}_{4a}(\mathcal{Q}) \vee \mathsf{ExistsFit}_{4b}(\mathcal{Q})\big)$$

$$\mathsf{Win5}(\mathcal{Q}) = \mathsf{NotWin4}(\mathcal{Q}) \wedge \mathsf{ExistsFit}_5(\mathcal{Q})$$

$$\mathsf{Win6}(\mathcal{Q}) = \mathsf{NotWin5}(\mathcal{Q}) \wedge \big(\mathsf{ExistsFit}_{6a}(\mathcal{Q}) \vee \mathsf{ExistsFit}_{6c}(\mathcal{Q})\big)$$

$$\mathsf{Win7}(\mathcal{Q}) = \mathsf{NotWin6}(\mathcal{Q}) \wedge \big(\mathsf{ExistsFit}_{7a}(\mathcal{Q}) \vee \mathsf{ExistsFit}_{7b}(\mathcal{Q})\big)$$

$$\mathsf{Win8}(\mathcal{Q}) = \mathsf{NotWin7}(\mathcal{Q}) \wedge \mathsf{ExistsFit}_8(\mathcal{Q})$$

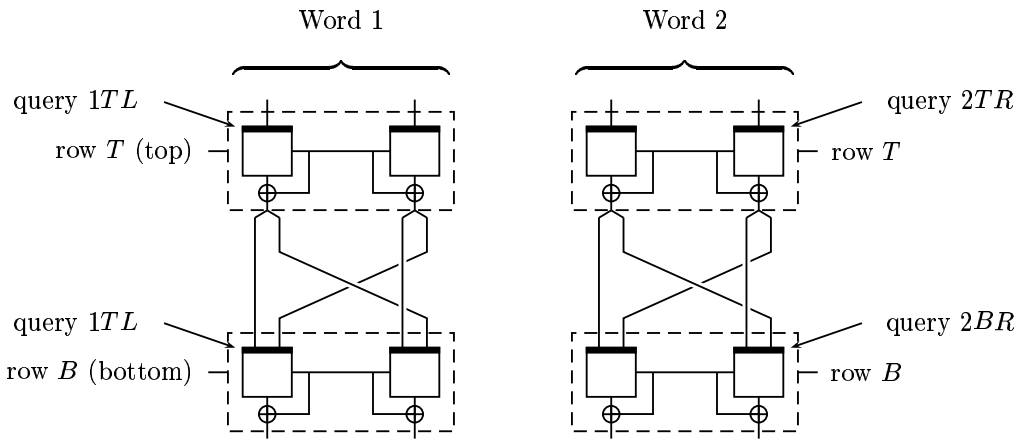


Figure 4: The query labels.

The reader will note that all configurations have at most 2 pieces and each piece is a subportion of two rounds of MDC2. If the configuration has two pieces (such as configurations 2, $4a$, $4b$, 5, $6a$, $6b$, $6c$, $6d$, $7a$, $7b$, 8 as opposed to configurations $1a$, $1b$, 3) then the left portion of the configuration is called "Word 1" and the right portion of the configuration is called "Word 2" (Fig. 4). Queries in the right-hand column of a two-round piece are called "right column" queries and queries in the left-hand column of a two-block portion are called "left column" queries. "Top row" and "bottom row" queries are defined the exptected way. A query in the configuration is given coordinates $1TR$ for "Word 1, Top row, Right column" or $2BL$ for "Word 2, Bottom row, Left column", etc. If the configuration has only one piece then we drop the prefix "1" or "2" and simply give coordinates TL, TR, etc. for the queries. The reader should refer to Fig. 4.

We now show that $\mathsf{Coll}^{\mathrm{MDC2}^E}(\mathcal{Q}) \implies \mathsf{Win0}(\mathcal{Q}) \vee \cdots \vee \mathsf{Win8}(\mathcal{Q})$:

Theorem 3. $\mathsf{Coll}^{\mathrm{MDC2}^E}(\mathcal{Q}) \implies \mathsf{Win0}(\mathcal{Q}) \vee \cdots \vee \mathsf{Win8}(\mathcal{Q})$

Proof. First we define new predicates $\mathsf{Win1}', \ldots, \mathsf{Win8}'$ such that $\mathsf{Win1}'(\mathcal{Q}) \vee \cdots \vee \mathsf{Win8}'(\mathcal{Q}) \implies \mathsf{Win0}(\mathcal{Q}) \vee \cdots \vee \mathsf{Win8}(\mathcal{Q})$. Then it is sufficient to show $\mathsf{Coll}^{\mathrm{MDC2}^E}(\mathcal{Q}) \implies \mathsf{Win1}'(\mathcal{Q}) \vee \cdots \vee \mathsf{Win8}'(\mathcal{Q})$. The new predicates are:

$$\mathsf{Win1}'(\mathcal{Q}) = \mathsf{ExistsFit}_{1a}(\mathcal{Q}) \vee \mathsf{ExistsFit}_{1b}(\mathcal{Q})$$

$$\mathsf{Win2}'(\mathcal{Q}) = \mathsf{ExistsFit}_{2}(\mathcal{Q})$$

$$\mathsf{Win3}'(\mathcal{Q}) = \mathsf{ExistsFit}_{3}(\mathcal{Q})$$

$$\mathsf{Win4}'(\mathcal{Q}) = \mathsf{ExistsFit}_{4a}(\mathcal{Q}) \vee \mathsf{ExistsFit}_{4b}(\mathcal{Q})$$

$$\mathsf{Win5}'(\mathcal{Q}) = \mathsf{ExistsFit}_{5}(\mathcal{Q})$$

$$\mathsf{Win6}'(\mathcal{Q}) = \mathsf{ExistsFit}_{6a}(\mathcal{Q}) \vee \mathsf{ExistsFit}_{6c}(\mathcal{Q})$$

$$\mathsf{Win7}'(\mathcal{Q}) = \mathsf{ExistsFit}_{7a}(\mathcal{Q}) \vee \mathsf{ExistsFit}_{7b}(\mathcal{Q})$$

$$\mathsf{Win8}'(\mathcal{Q}) = \mathsf{ExistsFit}_{8}(\mathcal{Q})$$

It is clear that $\mathsf{Win1}'(\mathcal{Q}) \vee \cdots \vee \mathsf{Win8}'(\mathcal{Q}) \implies \mathsf{Win0}(\mathcal{Q}) \vee \cdots \vee \mathsf{Win8}(\mathcal{Q})$.

Say $\mathsf{Coll}^{\mathrm{MDC2}^E}(\mathcal{Q})$. Then a collision can be constructed from the queries $\mathcal{Q}$. We can assume that the collision is the earliest possible, as explained in the introduction. By definition collisions involve words with at least 1 block, so the collision must either (i) use two words that are 1 block long each (ii) use one word of ≥ 2 blocks and one word of 1 block or (iii) use two words of ≥ 2 blocks. If the collision uses two words that are 1 block length each then obviously $\mathsf{ExistsFit}_2(\mathcal{Q})$ (if query i where equal to $!i$, the two words would be the same), so we can assume either (ii) or (iii).

Say first the collision is of type (ii), namely that the collision has one word with $m \geq 2$ blocks, which is WLOG word 1, and one word of with 1 block, which is word 2. Note first that when word 1 is hashed via MDC-2 there can never be a round where the same query appears both on the left and right-had sides unless $\mathsf{Win1}(\mathcal{Q})$ holds (to see this, take the earliest such round; then the two queries from the round before are different but have the same output, so $\mathsf{Win1}(\mathcal{Q})$). Therefore we can assume that at every round in the hashing of word 1, different queries appear on the left and right-hand sides. Naturally the same query may appear both in the left and right columns in different rounds.

We now examine the last two rounds of the hashing of word 1. The four (not necessarily distinct) queries comprising these two rounds are labeled $1TL$, $1TR$, etc. as in Fig. 4 and as per our convention described above. The two queries making up the unique round for the hashing of word 2 are simply labeled $2L$ and $2R$, where $2L$ is the query with key input A_1 and $2R$ is the query with key input B_1. By our previous remark, queries $1TL$ and $1TR$ are distinct as well as queries $1BL$ and $1BR$. If query $1BL$ equals query $2L$ and query $1BR$ equals query $2R$ then $\mathsf{ExistsFit}_3(\mathcal{Q})$. On the other hand if query $1BL$ is not equal to query $2L$ and query $2BR$ is not equal to query $2R$ then $\mathsf{ExistsFit}_5(\mathcal{Q})$. Therefore we can assume (by symmetry) that query $1BL$ is not equal to query $2L$ but that query $1BR$ equals query $2R$. But then $\mathsf{ExistsFit}_{4a}(\mathcal{Q})$. This concludes the case when the adversary's collision is of type (ii).

We now assume that both of the words involved in the collision have ≥ 2 rounds. We examine the last two rounds of the hashing of each word; the queries for these last two rounds are labeled as in Fig. 4. By the same remark as above, the same query cannot appear in both left and right positions at the same round of the same word, so the top row constraints of configuration 8 are satisfied. If query $1BL$ equals $2BL$ and query $1BR$ equals query $2BR$ then the collision is not earliest possible, a contradiction, so we can assume (by symmetry) that query $1BL$ is not equal to query $2BL$. If queries $1BR$ and $2BR$ are equal then $\mathsf{ExistsFit}_{7a}$ so they too must be unequal. But then $\mathsf{ExistsFit}_8$ so we are done. □

The reader may have noted that $\mathsf{Win6}'(\mathcal{Q})$ does not actually appear in the proof of Theorem 3. However $\mathsf{Win6}(\mathcal{Q})$ will be used in the upper bounding of $\Pr[\mathsf{Win7}(\mathcal{Q})]$ (recall that $\mathsf{Win7}(\mathcal{Q})$ is false if $\mathsf{Win6}(\mathcal{Q})$).

Let $\mathsf{WinFit}(\mathcal{Q}) = \mathsf{Win1}(\mathcal{Q}) \vee \ldots \vee \mathsf{Win8}(\mathcal{Q})$, so $\Pr[\mathsf{Coll}^{\mathrm{MDC2}^E}(\mathcal{Q})] \leq \Pr[\mathsf{Win0}(\mathcal{Q})] + \Pr[\mathsf{WinFit}(\mathcal{Q})]$. We will show:

Theorem 4. *Let N, N', N'', m_a, m_b, M_b, m_c, M_c be as in Theorem 1. Then:*

$$\Pr[\mathsf{Win0}(\mathcal{Q})] \leq \frac{q^2}{m_a N'} + 2qN^{\frac{1}{2}} e^{qN^{\frac{1}{2}} M_b(1-\ln(M_b))/N'} + qNe^{qM_c(1-\ln(M_c))/N'}.$$

and:

Theorem 5. *Let N, N', N'', m_a, m_b, m_c be as in Theorem 1. Then:*

$$\begin{aligned}\Pr[\mathsf{WinFit}(\mathcal{Q})] \quad \leq \quad & q(m_a^2 + m_a m_b^2 + m_b^4)/N' + \\ & q(4m_a m_b)/N' + q(2m_a m_b)/N'' + \\ & q(m_b^2 m_c + 5m_b^2 + m_a m_c + 6m_a)/N' + q(4m_a + 8m_b^2)/N'' + \\ & q(4 + 10m_b + 2m_b m_c)/N'' + 3q/N' + 4q/N'' + q^2/N'^2\end{aligned}$$

Theorems 4 and 5 imply Theorem 1 (by Theorem 3). The proof of Theorem 4 is given in Appendix B, which also contains the proof of Theorem 2. The proof of Theorem 5 (which constitutes the "meat" of the paper) is given in Appendix A. To give a feel for the type of argument involved in proving Theorem 5 we show here how to upper bound $\Pr[\mathsf{NotWin0}(\mathcal{Q}) \wedge \mathsf{ExistsFit}_{1a}(\mathcal{Q})]$, which establishes "half" of the upper bound for $\Pr[\mathsf{Win1}(\mathcal{Q}) = \mathsf{NotWin0}(\mathcal{Q}) \wedge (\mathsf{ExistsFit}_{1a}(\mathcal{Q}) \vee \mathsf{ExistsFit}_{1b}(\mathcal{Q}))]$. The constants N, N', N'' will remain throughout as defined in Theorem 1, namely $N = 2^n$, $N' = N - q$, $N'' = N'(N^{\frac{1}{2}} - m_b)/N^{\frac{1}{2}}$.

Proposition 6. $\Pr[\mathsf{NotWin0}(\mathcal{Q}) \wedge \mathsf{ExistsFit}_{1a}(\mathcal{Q})] \leq q(m_a + m_b^2)/N' + 2qm_b/N''$.

Proof. Let $\mathcal{Q}_i$ denote the first i queries made by the adversary. The term "last query" means the latest query made by the adversary (we examine the adversary's queries (K_i, X_i) or (K_i, Y_i) one at a time, in succession as they come in). The last query is always given index i. We say the last query is "successful" if the output Y_i or X_i for the last query is such that $a(\mathcal{Q}_i) < m_a$, $b(\mathcal{Q}_i) < m_b$, $c(\mathcal{Q}_i) < m_c$ and such that

the adversary can use the query (X_i, K_i, Y_i) to fit configuration 1a using only queries in $\mathcal{Q}_i$ (in particular, the last query *must* be used in the fitting for that query to count as successful). The goal is thus to upper bound the adversary's chance of ever making a successful last query.

The strategy for upper bounding the probability of the last query being successful is to consider separately the different ways in which the last query can be used to fit the configuration and to upper bound the probability of success in each case, and finally to sum the various upper bounds. For example, the adversary may use the last query only once in the configuration or otherwise in several different positions of the configuration (such as, say, TL and BL). The basic setup for upper bounding the probability of success in a given case is to upper bound the maximum number of different outputs Y_i or X_i (depending on whether the last query is a forward or backward query) that would allow the query (X_i, K_i, Y_i) to be used to fit the configuration, and then to divide this number by $N' = N - q$ (since either Y_i or X_i, depending, is chosen randomly among a set of at least N' different values). That ratio is then multiplied by q, since the adversary makes q queries in all, each of which could become a successful last query.

Case 1: The last query is used exactly once in the configuration. We can assume WLOG that it is used in the left column.

Subcase 1.1: The last query is used in position BL. Say first that the last query is a forward query (K_i, X_i). Since the last query cannot be successful if $b(\mathcal{Q}_{i-1}) \geq m_b$ (by definition) we can assume that $b(\mathcal{Q}_{i-1}) < m_b$. Then since the left half of the XOR output of the query used in position TL must be equal to the left half of K_i there are at most m_b different queries in $\mathcal{Q}_{i-1}$ that could be used in position TL, for the given inputs (K_i, X_i) of the last query. Likewise because the right half of the XOR output of the query used in position TR must be equal to the right half of K_i there are at most m_b different queries in $\mathcal{Q}_{i-1}$ that could be used in position TR. Since X_i together with the outputs of the queries used in positions TL, TR completely determine the query used in position BR, there are therefore at most m_b^2 different queries in $\mathcal{Q}_{i-1}$ which can be used in position BR for the given inputs (K_i, X_i). Therefore there are at most m_b^2 outputs Y_i which would enable the last query be used to fit the configuration at position BL (namely which would enable the XOR output $X_i \oplus Y_i$ of query BL to be equal to the XOR output of query BR), so the chance of success of the last query if it is forward is $\leq m_b^2/N'$.

Now say the last query is a backward query (K_i, Y_i). We cannot reason like for the forward query case that there are only m_b^2 queries in $\mathcal{Q}_{i-1}$ that that can appear in position BR since we do not know the word input X_i anymore. However because the query used in position BR has same XOR output and same word input as the query in position BL it must also have the same output as the query in position BL, which means the output of the query in position BR is actually Y_i. Now because E is a blockcipher, there is exactly at most one possible query for position BR in $\mathcal{Q}_{i-1}$ for any given value of the key of the query in position BR, and since the key can take at most m_b^2 different values (as in the forward case) there are again at most m_b^2 different queries that can be used in position BR. Therefore there are at most m_b^2 different values for X_i which would make the backwards query (K_i, Y_i) successful, so the last query again has chance of success $\leq m_b^2/N'$.

Thus the last query has chance of success $\leq m_b^2/N'$ whether it is a forward or backward query. Multiplying by q, we obtain that the chance of ever making a successful last query of this type is $\leq qm_b^2/N'$. This concludes the analysis of Subcase 1.1.

Note: we will not always give as many details as in Subcase 1.1. In particular, we will not continue to remind that one can assume $a(\mathcal{Q}_{i-1}) < m_a$, $b(\mathcal{Q}_{i-1}) < m_b$, $c(\mathcal{Q})_{i-1} < m_c$ (or else the last query is by definition not successful) and we will often shorten phrases of the type "query used in position TL" to simply "query TL".

Subcase 1.2: The last query is used in position TL. Because the queries use in positions BL, BR are distinct but have the same XOR output there are at most m_a different ordered pairs of queries in $\mathcal{Q}_{i-1}$ that can be used for the pair BL, BR. But the pair of queries for BL, BR completely determines what the XOR output $X_i \oplus Y_i$ of the last query should be. Therefore the last query has chance at most m_a/N' of success and the total probability of making this type of successful last query is $\leq qm_a/N'$.

Note: Subcase 1.2 does not require a separate analysis for the forward and backward case because we can upper bound the maximum number of successful XOR outputs for the last query *without* looking at the inputs for the last query; by contrast, in Subcase 1.1 we inspected X_i in the forward case and Y_i in the backward case in order to determine the maximum possible number of successful XOR outputs. In general, whenever an upper bound on the total number of successful XOR outputs for the last query can be found without inspecting any inputs for the last query besides the key, the same analysis will work both for the forward and backward cases.

Case 2: The last query is used twice or more in the configuration. Because queries BL and BR are distinct the queries TL and TR are also distinct and so the last query must in fact appear exactly twice in the configuration. We can assume WLOG that it is used in position TL.

The type of analysis we use for this case is slightly different than the analysis for Subcases 1.1, 1.2. To estimate the probability of the last query succeeding we will first look at the left $n/2$ bits of XOR output, estimate their probability P_l of success (the left bits are "successful" if they do not preclude the last query from being successful) and then we estimate the probability of success $P_{r|l}$ of the right $n/2$ bits of XOR output being successful, conditioned on the fact that the left $n/2$ bits are successful (the right $n/2$ bits are "successful" if the last query is successful). The probability of success of the last query is then $P_l P_{r|l}$. Note that if the set of left half of XOR outputs which are successful has size T then $P_l \leq TN^{\frac{1}{2}}/N'$ since the return to any query has chance $\leq N^{\frac{1}{2}}/N'$ of having its left half of XOR output equal to any particular value (there are at most $N^{\frac{1}{2}}$ strings that have that left half, each of which is returned with chance at most $1/N'$). Then if the left half is successful and there are U different possible ways of completing the left half into a successful string, namely U different successful right halfs, the chance of the right half being successful given $\mathsf{NotWin0}(\mathcal{Q}_{i-1})$ is $\leq U/(N^{\frac{1}{2}} - m_b)$ since the XOR output could be any of at least $N^{\frac{1}{2}} - m_b$ values with equal probability (there are at most m_b values which we know will not appear because they have already appeared for this left half). So the total chance of success of the last query in this case (assuming U was independent of the left half, as it will be in our analysis) is $\leq TUN^{\frac{1}{2}}/N'(N^{\frac{1}{2}} - m_b)$ or $\leq TU/N''$.

Subcase 2.1: The last query is used in positions TL, BL. Since the last query appears in positions TL, BL the left half of the last query's XOR output must be equal to the left half of its key input, so the left half of output has chance $P_l \leq N^{\frac{1}{2}}/N'$ chances of succeeding. If it succeeds, there are at most m_b queries for BR in $\mathcal{Q}_{i-1}$ with that left half of XOR output (which must be shared with query BL), so the right half of XOR output has chance $P_{r|l} \leq m_b/(N^{\frac{1}{2}} - m_b)$ of succeeding if the the left half succeeds. Therefore the last query has chance $P_l P_{r|l} \leq m_b N^{\frac{1}{2}}/N'(N^{\frac{1}{2}} - m_b) = m_b/N''$ of succeeding and the adversary's total chance of making this kind of successful last query is $\leq qm_b/N''$.

Subcase 2.2: The last query is used in position TL and in position BR. The same type of analysis as for Subcase 2.1 applies, showing that the total chance of a successful last query of this type is $\leq qm_b/N''$.

Subcase 2.2 concludes Case 2 and thus all possible cases of making a successful query for configuration $1a$. Summing up the probabilities we get that $\Pr[\mathsf{NotWin0}(\mathcal{Q}) \wedge \mathsf{ExistsFit}_{1a}(\mathcal{Q})] \leq q(m_a + m_b^2)/N' + 2qm_b/N''$. □

References

[1] ANSI X9.31. Public key cryptography using reversible algorithms for the financial services industry. American National Standards Institute, 1998.

[2] B. den. Boer, A. Bosselaers. Collisions for the compression function of MD5. *Advances in Cryptology – EUROCRYPT 93*, Lecture Notes in Compuetr Science, vol. 765, Springer, pp. 293–304, 1993.

[3] J. Black, M. Cochran, and T. Shrimpton. On the impossibility of highly efficient blockcipher-based hash functions. *Advances in Cryptology – EUROCRYPT '05*. Lecture Notes in Computer Science, vol. 3494, pp.–546-541, 2005.

[4] J. Black, P. Rogaway, and T. Shrimpton. Black-box analysis of the block-cipher-based hash-function constructions from PGV. *Advances in Cryptology – Crypto 02*. Lecture Notes in Compuer Science, vol. 2442, pp. 320–355, Springer, 2002.

[5] B. Brachtl, D. Coppersmith, M. Hyden, S. Matyas, C. Meyer, J. Oseas, S. Pilpel M. Schilling. Data authentication using modification detection codes based on a public one-way encryption function. US Patent #4,908,861. Awarded March 13, 1990 (filed Auguest 28, 1987).

[6] I. Damgård. A design principle for hash functions. *Advances in Cryptology –CRYPTO 89*. Lecture Notes in Computer Science, vol. 435, pp. 416–427, 1990.

[7] H. Dobbertin. The status of MD5 after a recent attack. *CryptoBytes* 2 (2), 1996.

[8] S. Even and Y. Mansour. A construction of a cipher from a single pseudorandom permutation. *Advances in Cryptology – ASIACRYPT '91*. Lecture Notes in Computer Science, vol. 739, Springer, pp. 210–224, 1991.

[9] M. Hattori, S. Hirose and S. Yoshida. Analysis of double block lengh hash functions. *Cryptography and Coding, 9th IMA International Conference*, Lecture Notes in Computer Science, vol. 2898, Springer, pp. 290–302, 2003.

[10] S. Hirose. Provably secure double-block-length hash functions in a black box model. *Information Security and Cryptology—ICISC 2004*. Lecture Notes in Computer Science, Springer.

[11] S. Hirose. Some plausible constructions of double-block-length hash functions. *Fast Software Encryption (FSE 2006)*. Lecture Notes in Computer Science, Springer. To appear.

[12] W. Hohl, X. Lai, T. Meier C. Waldvogel. Security of iterated hash functions based on block ciphers. *Advances in Cryptology – CRYPTO 1993*. Lecture Notes in Comptuer Science, vol. 773, pp. 303–311, 1993.

[13] ISO/IEC 10118-2:2000. Information technology – Security techniques – Hash functions – Hash functions using an n-bit block cipher. International Organization for Standardization, Geneva, Switzerland, 2000. First released in 1992.

[14] A. Joux, Multicollisions in iterated hash functions, application to cascaded constructions. Crypto 04, LNCS 3152, pp. 306–316.

[15] J. Kilian, P. Rogaway. How to Protect DES Against Exhaustive Key Search. *Journal of Cryptology*, vol. 14, no. 1, pp. 17–35, 2001.

[16] L. Knudsen, X. Lai and B. Preneel. Attacks on fast double block length hash functions. *Journal of Cryptology*, vol. 11, no. 1, pp. 59-72, 1998.

[17] X. Lai and J. Massey. Hash functions based on block ciphers. *Advances in Cryptology – Eurocrypt 92*. Lecture Notes in Computer Science, vol. 658, pp. 55–70,

[18] W. Lee, M. Nandi, P. Sarkar, D. Chang, S. Lee, and K. Sakurai. PGV-style block-cipher-based hash families and black-box analysis. *IEICE Transactions 88-A(1)*, pp. 39–48, 2005.

[19] S. Lucks. Design principles for iterated hash functions. Cryptology ePrint Archive, Report 2004/253, 2004. http://eprint.iacr.org/.

[20] S. Matyas, C. Meyer, and J. Oseas. Generating strong one-way functions with cryptographic algorithm. IBM Technical Disclosure Bulletin, 27, pp. 5658–5659, 1985.

[21] R. Merkle. One way hash functions and DES. *Advances in Cryptology – CRYPTO 89*. Lecture Notes in Computer Science, vol. 435, Springer, pp. 428–446, 1990.

[22] C. Meyer and S. Matyas. Secure program load with manipulation detection code. *Proceedings of the 6th Worldwide Congress on Computer and Communications Security and Protection (SECURICOM '88)*, pp. 111–130, 1988.

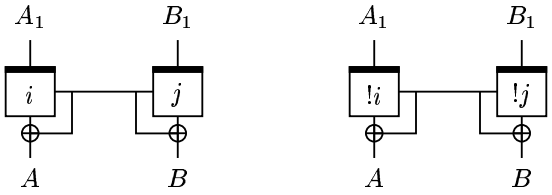
[23] M.Nandi, W. Lee, K. Sakurai, and S. Lee. Security analysis of a 2/3-rate double length compression function in the black-box model. *Fast Software Encryption (FSE 2005)*, Lecture Notes in Computer Science, vol. 3557, pp. 243–254, 2005.

[24] M. Nandi. Towards optimal double-length hash functions. *Progress in Cryptography – INDOCRYPT 2005*, Lecture Notes in Computer Science, vol. 3797, Springer, pp. 77–89, 2005.

[25] R. Rivest. The MD4 message digest algorithm. *Advances in Cryptology – CRYPTO 1990*. Lecture Notes in Comptuer Science, vol. 537, pp. 303–311, 1991.

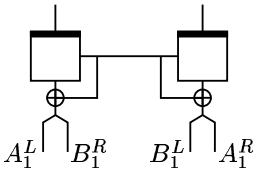
[26] M. Rabin. Digitalized signatures. In R. DeMillo, D. Dobkin, A. Jones, and R. Lipton, editors, *Foundations of Secure Computation*, Academic Press, pp. 155–168, 1978.

[27] T. Satoh, M. Haga, K. Kurosawa. Towards secure and fast hash functions. *IEICE Transactions on*



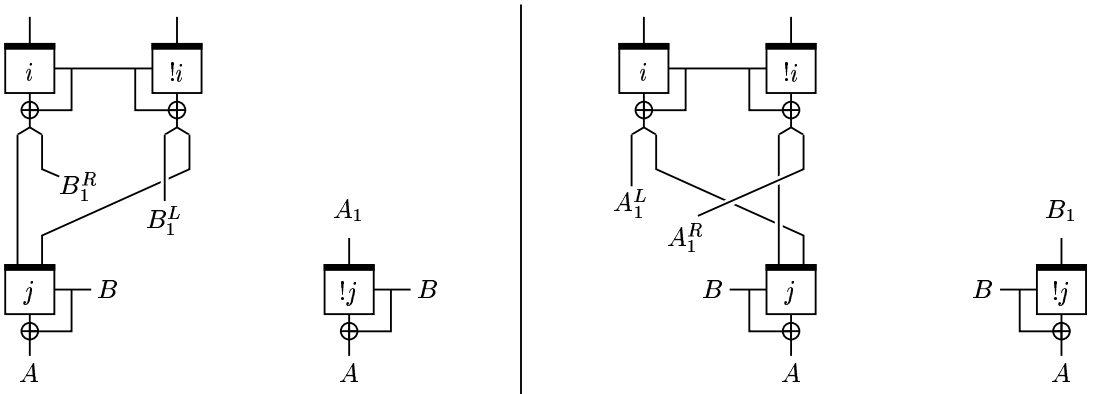
Configuration 2

Figure 5: The configurations defining $\mathsf{Win2}(\mathcal{Q})$



Configuration 3

Figure 6 The configuration defining $\mathsf{Win3}(\mathcal{Q})$.



Configuration 4*a* Configuration 4*b*

Figure 7: The configurations defining $\mathsf{Win4}(\mathcal{Q})$.

Fundamentals of Electronics, Communications and Computer Sciences, Vol. E82–A No. 1, pp. 55-62

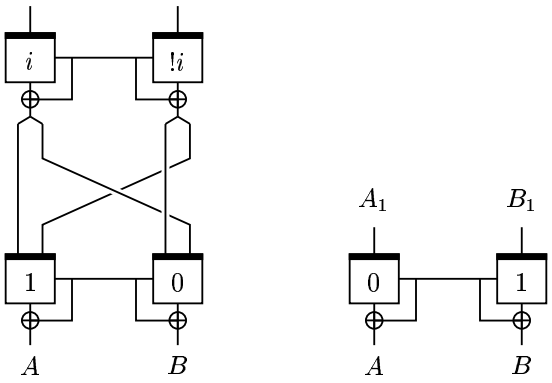
[28] J. Viega. The AHASH mode of operation. Manuscript, 2004. Available from www.cryptobarn.com

[29] X. Wang, X. Lai, D. Feng, H. Chen, and X. Yu. Cryptanalysis of the Hash Functions MD4 and RIPEMD. *Advances in Cryptology – EUROCRYPT '05*, Lecture Notes in Computer Science, vol. 3494, Springer, pp. 1–18. 2005.

[30] X. Wang, Y. Yin, and H. Yu. Finding collisions in the full SHA-1. *Advances in Cryptology – CRYPTO 05*, Lecture Notes in Computer Science, vol. 3621, Springer, pp. 17–36, 2005.

A Proof of Theorem 5

To upper bound $\Pr[\mathsf{WinFit}(\mathcal{Q})]$ we give separate upper bounds for $\Pr[\mathsf{Win1}(\mathcal{Q})]$, ..., $\Pr[\mathsf{Win8}(\mathcal{Q})]$ and sum the result. The proofs for the upper bounds of $\Pr[\mathsf{Win1}(\mathcal{Q})]$, ..., $\Pr[\mathsf{Win8}(\mathcal{Q})]$ all function similarly to the proof of Proposition 6 by upper bounding the probability of making a successful last query (namely a last



Configuration 5

Figure 8: The configuration defining $\mathsf{Win5}(\mathcal{Q})$.

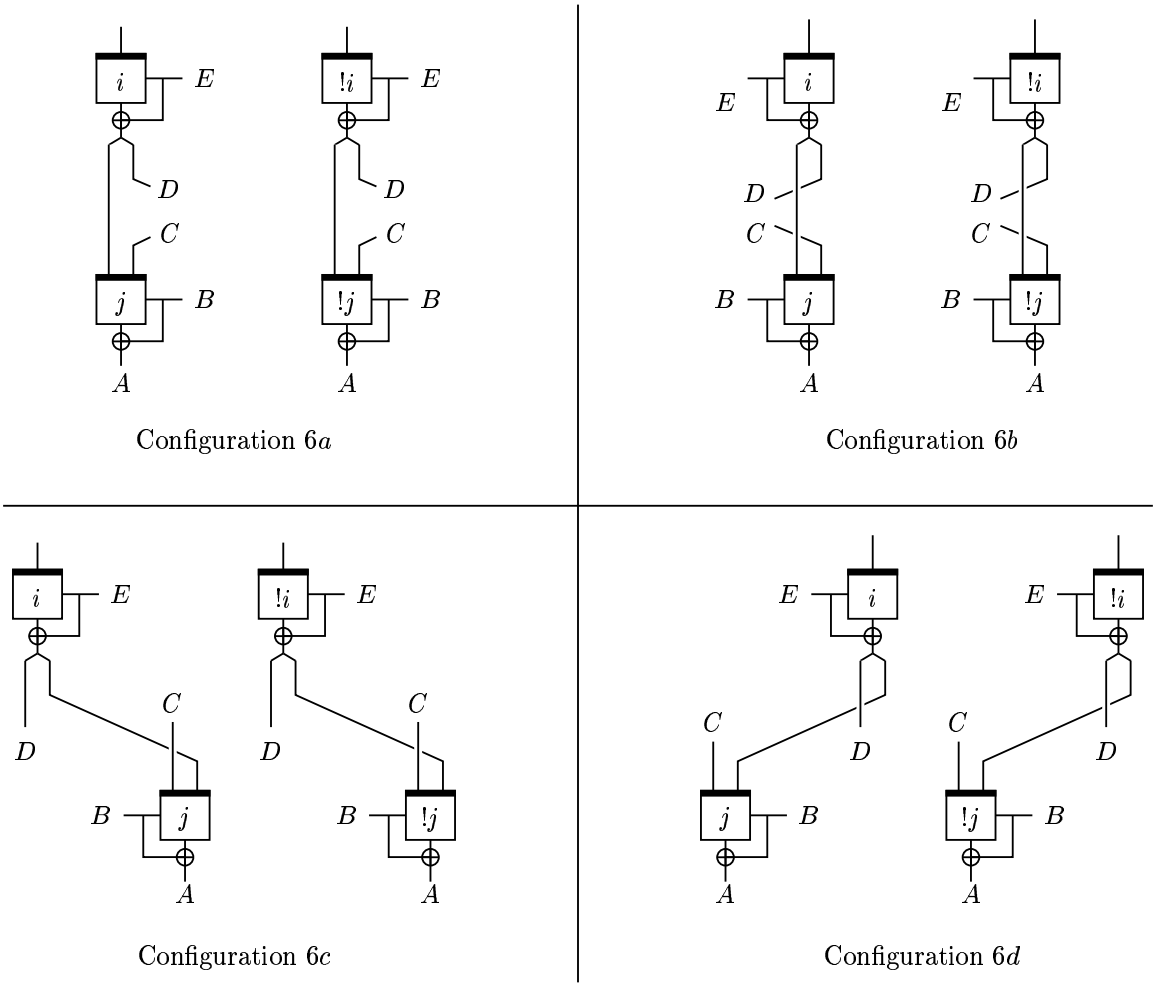


Figure 9: The configurations defining $\mathsf{Win6}(\mathcal{Q})$.

query completing the configuration in question, subject to the "NotWin" condition). We therefore keep the same terminology and notation as for the proof of Proposition 7 We keep all the terminology of the proof

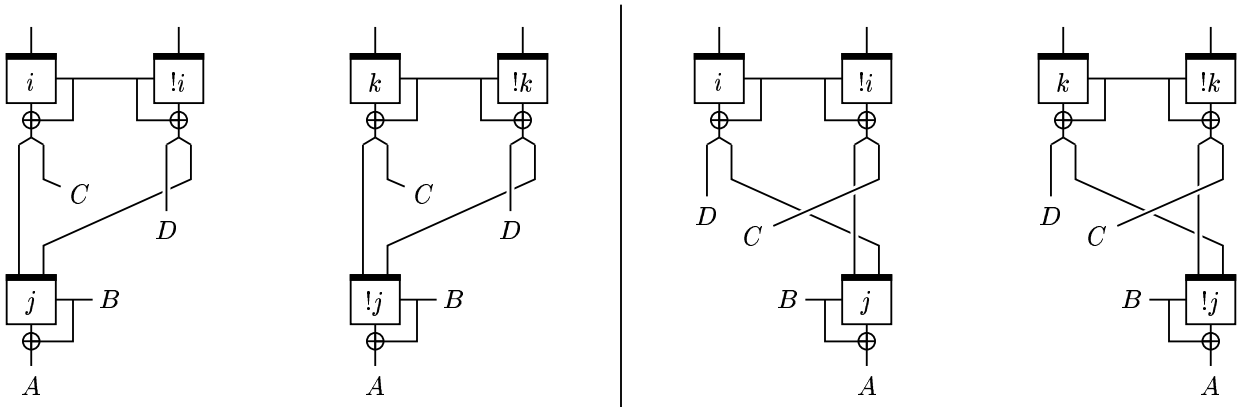


Figure 10: The configurations defining $\mathsf{Win7}(\mathcal{Q})$.

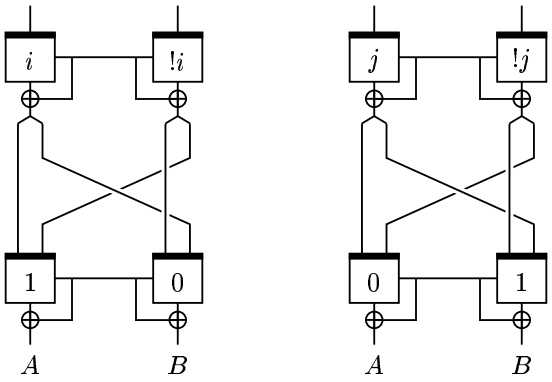


Figure 11: The configuration defining $\mathsf{Win8}(\mathcal{Q})$.

Proposition 6. We start by completing our upper bound for $\Pr[\mathsf{Win1}(\mathcal{Q})]$, already started in Proposition 6.

Proposition 7. $\Pr[\mathsf{Win1}(\mathcal{Q})] \leq q(1 + m_a + m_b^2)/N' + 2qm_b/N''$.

Proof. By Proposition 6 it suffices to show $\Pr[\mathsf{NotWin0}(\mathcal{Q}) \wedge \mathsf{ExistsFit}_{1b}(\mathcal{Q})] \leq q/N'$. We use the same technique as in Proposition 6, namely we upper bound the probability of the adversary ever making a successful last query, where a last query is now "successful" if it can be used to complete configuration $1b$ and if $\mathsf{NotWin0}(\mathcal{Q}_i)$ (with i being the index of the last query). The two positions of configuration $1b$ will simply be labeled L and R for "left" and "right".

Case 1: The last query has key A_1. Then the last query, if it is successful, will be used in position L. If the last query is forward then we know the word input for query R so there is at most 1 choice in $\mathcal{Q}_{i-1}$ for query R (whose key is constant) and therefore at most 1 successful value for the XOR output of the last query. The last query then has chance $\leq 1/N'$ of succeeding. If the last query is backward then we know its output Y_i and therefore the output of query R, whose output (as well as its XOR output) must be equal to the output of query L, so there is again only at most 1 choice in $\mathcal{Q}_{i-1}$ for query R and the last query has chance at most $1/N'$ of succeeding. Therefore either way the last query has chance $\leq 1/N'$ of succeeding.

Case 2: The last query has key B_1. Same analysis, same bound as above.

Since the last query cannot be successful if its key is neither A_1 or B_1 and since its key is either A_1 or

B_1 but not both, the chance of success of a last query is $\leq 1/N'$ and the total chance of success is $\leq q/N'$, so $\Pr[\mathsf{NotWin0}(\mathcal{Q}) \wedge \mathsf{ExistsFit}_{1b}(\mathcal{Q})] \leq q/N'$. □

Proposition 8. $\Pr[\mathsf{Win2}(\mathcal{Q})] \leq q^2/N'^2$.

Proof. For every forward query of the form (A_1, X_i) made by the adversary we give it for free the answer to the forward query (B_1, X_i) and vice-versa if the adversary makes a forward query (B_1, X_i) we give it for free the answer to the forward query (A_1, X_i). Also for every backward query (A_1, Y_i) made by the adversary we give it for free the answer to the forward query (B_1, X_i) where X_i is the answer to (A_1, Y_i), and similarly for every backward query (B_1, Y_i) made by the adversary we give it for free the answer to the forward query (A_1, X_i) where X_i is the answer to (B_1, Y_i). Thus at any point of its computation and for any word X the adversary either knows both the answers to the queries (A_1, X), (B_1, X) or else knows neither. The reader can also check that the adversary is never given for free a query that it already knows the answer for.

Say the last query made by the adversary is a forward query (K_i, X_i). Then that query has 0 chance of being successful if $K_1 \neq A_1, B_1$ and otherwise is equivalent to the pair of forward queries (A_1, X_i), (B_1, X_i) neither of which the adversary knows the answers to. The pair (A_1, X_i), (B_1, X_i) is successful if the adversary's query history $\mathcal{Q}_{i-1}$ contains a pair $(A_1, X_j, E(A_1, X_j))$, $(B_1, X_j, E(B_1, X_j))$ such that $X_i \oplus E(A_1, X_i) = X_j \oplus E(A_1, X_j)$, $X_i \oplus E(B_1, X_i) = X_j \oplus E(B_1, X_j)$. Take any such pair $(A_1, X_j, E(A_1, X_j))$, $(B_1, X_j, E(B_1, X_j))$ in $\mathcal{Q}_{i-1}$. Then the output of $E(A_1, X_i)$ of the query (A_1, X_i) has chance $\leq 1/N'$ of satisfying $X_i \oplus E(A_1, X_i) = X_j \oplus E(A_1, X_j)$, since that equation has a unique solution. If the output does satisfy that equation, the output of (B_1, X_i) still has chance $\leq 1/N'$ of satisfying its equation $X_i \oplus E(B_1, X_i) = X_j \oplus E(B_1, X_j)$. Therefore there is $\leq 1/N'^2$ chance that the last query will be successful with respect to the pair of queries $(A_1, X_j, E(A_1, X_j))$, $(B_1, X_j, E(B_1, X_j))$. Since there are at most q such pairs in $\mathcal{Q}_{i-1}$ the last query has chance $\leq q/N'^2$ of success.

If the last query is a backward query (K_i, Y_i) then the last query has 0 chance of success if $K_1 \neq A_1, B_1$, otherwise we can assume WLOG that $K_i = A_1$. We return to the adversary both $X_i = E^{-1}(A_1, Y_i)$ and $E(B_1, X_i)$. The query is successful if there is some pair $(A_1, X_j, E(A_1, X_j))$, $(B_1, X_j, E(B_1, X_j))$ in the adversary's query history $\mathcal{Q}_{i-1}$ such that $X_i \oplus Y_i = X_j \oplus E(A_1, X_j)$, $X_i \oplus E(B_1, X_i) = X_j \oplus E(B_1, X_j)$. Fix such a pair $(A_1, X_j, E(A_1, X_j))$, $(B_1, X_j, E(B_1, X_j))$ before looking at the returned values X_i, $E(B_1, X_i)$. The returned value X_i has chance $\leq 1/N'$ of satisfying $X_i \oplus Y_i = X_j \oplus E(A_1, X_j)$ and, if it does, then $E(B_1, X_i)$ also has chance $\leq 1/N'$ of satisfying $X_i \oplus E(B_1, X_i) = X_j \oplus E(B_1, X_j)$. Therefore there is $\leq 1/N'^2$ chance that the last query will be successful with respect to the pair of queries $(A_1, X_j, E(A_1, X_j))$, $(B_1, X_j, E(B_1, X_j))$. Since there are at most q such pairs in $\mathcal{Q}_{i-1}$ the last query has chance $\leq q/N'^2$ of success.

Thus whether the last query is forward or backward it has chance $\leq q/N'^2$ of being successful, and so the total chance of success is $\leq q^2/N'^2$. □

Proposition 9. $\Pr[\mathsf{Win3}(\mathcal{Q})] \leq q/N'$.

Proof. Obviously $\mathsf{Win3}(\mathcal{Q})$ implies the adversary has made at least one query with XOR output $A_1^L B_1^R$, which happens with probability $\leq 1/N'$ for each of the q queries, so $\Pr[\mathsf{Win3}(\mathcal{Q})] \leq q/N'$. □

Proposition 10. $\Pr[\mathsf{Win4}(\mathcal{Q})] \leq q(2 + 2m_b^2)/N' + q(4 + 4m_a + 4m_b)/N''$.

Proof. We upper bound the probability that $\mathsf{NotWin3}(\mathcal{Q}) \wedge \mathsf{ExistsFit}_{4a}(\mathcal{Q})$ since the proof for configuration $4b$ is equivalent. We refer to the single query in Word 2 of configuration $4a$ as query "$2L$" of the configuration.

Case 1: The last query appears only once in the configuration.

Subcase 1.1: The last query appears at position $1BL$. If the last query is a forward query then given its word input X_i there is only one query possible in position $2L$ so the last query has chance $\leq 1/N'$ of success. If the last query is backward then because the queries $1BL$, $2L$ must have same output as well as same XOR output there is still only one possible query for query $2L$ so the last query has chance $\leq 1/N'$ of being returned the correct input word X_i (which needs to be equal to the input word of query $2L$). Therefore the last query has chance $\leq 1/N'$ of success and the total chance of success of making a successful last query

of this type is $\leq q/N$.

Subcase 1.2: The last query appears at position $1TL$. First say that the last query is forward. We first look at the left $n/2$ bits of output and then at the right $n/2$ bits. Since queries $1BL$, $2L$ are distinct but have the same output there are at most m_a possibilities for the pair $1BL$, $2L$ in $\mathcal{Q}_{i-1}$ so the left half of the output of the last query has chance $\leq m_a N^{\frac{1}{2}}/N'$ of succeeding. Then since there is only one right answer for the right output half, the right output has chance $\leq 1/(N^{\frac{1}{2}} - m_b)$ of succeeding if the left half succeeds. Therefore the last query has chance $\leq m_a/N''$ of succeeding. Since the analysis did not require knowledge of the input word the same analysis applies if the query is backward, so the total chance of making a successful query of this type is $\leq qm_a/N''$.

Subcase 1.3: The last query appears at position $1TR$. Same analysis as above, total chance of success $\leq qm_a/N''$.

Subcase 1.4: The last query appears at position $2L$. There are at most m_b queries $1TL$ and m_b queries $1TR$ with the given right-half XOR outputs so at most m_b^2 different keys for query $1BR$. If the last query is forward then we also know the word input for query $1BR$ making at most m_b^2 queries $1BR$, whereas if the last query is backward then we know the output for query $1BR$ also making at most m_b^2 queries $1BR$. Therefore the chance of success of the last query is either way $\leq m_b^2/N'$ and the total chance of success is $\leq qm_b^2/N'$.

Case 2: The last query appears twice or more in the configuration. Then it appears exactly twice.

Subcase 2.1: The last query appears at positions $1TL$, $1BL$. Whether the query is backward of forward the left half of the output has chance $\leq N^{\frac{1}{2}}/N'$ of success because the left half of the XOR output is equal to the left half of the key. The right half of output then obviously has chance $\leq 1/(N^{\frac{1}{2}} - m_b)$ of success (whether the query is forward or backward). The chance of success of the last query is therefore $\leq 1/N''$ for a total chance of success of $\leq q/N''$.

Subcase 2.2: The last query appears at positions $1TR$, $1BL$. Same analysis, same bound.

Subcase 2.3: The last query appears at positions $1TL$, $2L$. Whether it is backwards or forwards, the right output bits have chance $\leq N^{\frac{1}{2}}/N'$ of success. If the right output bits are successful then we know the right half of the XOR output of query $2L$ and hence of query $1BL$ and there are at most m_b queries in $\mathcal{Q}_{i-1}$ with those right XOR output bits, so the left output bits have chance $\leq m_b/(N^{\frac{1}{2}} - m_b)$ of being successful (whether the last query is backwards or forwards). The total probability of success is $\leq qm_b/N''$.

Subcase 2.4: The last query appears at positions $1TL$, $2R$. Same analysis as above, same bound.

Subcase 2.4 concludes Case 2, and thus all cases. Tallying the probabilities and multiplying by 2 to account for configuration $4b$ yields a bound of $q(2 + 2m_b^2)/N' + q(4 + 4m_a + 4m_b)/N''$. □

Proposition 11. $\Pr[\mathsf{Win5}(\mathcal{Q})] \leq q(4m_a + m_a m_c + m_b^2 m_c)/N' + q(2m_b + 2m_b m_c)/N''$.

Proof. The two queries of Word 2 are simply labeled "$2L$, $2R$" since that word has only one row. We proceed with the same type of case analysis as usual:

Case 1: The last query is used only once in the configuration, in Word 1. We can assume WLOG that it is used in the left column of Word 1.

Subcase 1.1: The last query is used in position $1BL$. There are at most m_a pairs of queries in $\mathcal{Q}_{i-1}$ that can be used for queries $1BR$, $2R$ so at most m_a possibilities for query $2L$. Therefore (whether it is backwards or forwards) the last query has chance $\leq m_a/N'$ of succeeding and the total chance of success is $\leq qm_a/N'$.

Subcase 1.2: The last query is used in position $1TL$. There are m_a choices at most for the pair of queries $1BL$, $2L$ and for each such choice exactly one possibility for query $2R$, and for every query $2R$ at most m_c queries $1BR$, so the last query (whether it is forwards or backwards) has chance $\leq m_a m_c/N'$ of success and the total chance of success is $\leq q m_a m_c/N'$.

Case 2: The last query is used only once in the configuration, in Word 2. We can assume WLOG it is used in position $2L$. If the last query is forward then given its input there is only one possible query $2R$ and at most m_c queries $1BR$ and for each of those at most m_b^2 queries $1BL$ (because at most m_b queries $1TL$ and m_b queries $1TR$, see e.g. Subcase 1.2 of Proposition 6) so that the probability of success of the last query becomes $\leq m_b^2 m_c/N'$. On the other hand if the last query is backwards then because the queries $2R$, $1BR$ are distinct there are at most m_a choices for the pair $2R$, $1BR$ in $\mathcal{Q}_{i-1}$ so the word output for the last query has chance $\leq m_a/N'$ of fitting the word input of query $2R$. Therefore the last query has chance $\leq (m_a + m_b^2 m_c)/N'$ of succeeding whether it is forward or backward and therefore the total chance of success is $\leq q(m_a + m_b^2 m_c)/N'$.

Case 3: The last query is used exactly twice in the configuration, both times in Word 1. Then it is used once in the top row of Word 1 and once in the bottom row of Word 1. We can assume WLOG that it is used in position $1TL$.

Subcase 3.1: The last query is also used in position $1BL$. Whether it is backwards or forwards the left half of output has chance $\leq N^{\frac{1}{2}}/N'$ of succeeding. If it succeeds, there are at most m_b choices for query $2L$ so the right half of output then has chance at most $m_b/(N^{\frac{1}{2}} - m_b)$ of succeeding. The probability of success of the last query is therefore $\leq m_b/N''$ and the total probability of success is $\leq q m_b/N''$.

Subcase 3.2: The last query is also used in position $1BR$. Same analysis, same bound as above.

Case 4: The last query is used exactly twice in the configuration, once in Word 1, once in Word 2. We can assume it is used in position $2L$. Then the last query cannot appear in position $1BR$ or else $\mathsf{ExistsFit}_{1a}(\mathcal{Q})$ (and $\mathsf{ExistsFit}_{1b}(\mathcal{Q})$), so must appear either in position $1TL$ or $1TR$. Note that if the last query is forward then query $2R$ is uniquely determined by the inputs of the last query.

Subcase 4.1: The last query is also used in position $1TL$. If the last query is forward then B is uniquely determined so there are only m_c possibilities for query $1BR$ and the right half of output has chance $\leq m_c N^{\frac{1}{2}}/N'$ of succeeding. If the right output is successful then there are at most m_b possibilities for query $1BL$ so the left bits of output have chance $\leq m_b/(N^{\frac{1}{2}} - m_b)$ of succeeding. So if the last query is forward then it has $\leq m_b m_c/N''$ chance of succeeding. If the last query is backward then it has $\leq m_a/N'$ chance of succeeding since there are only at most m_a choices in $\mathcal{Q}_{i-1}$ for the pair of queries $1BR$, $2R$ (since the word input of $2R$ has to match up with the word input of the last query). So the total chance of success is therefore $\leq q m_a/N' + q m_b m_c/N''$.

Subcase 4.2: The last query is also used in position $1TR$. Same analysis and same bound as above.

Case 5: The last query is used 3 or more times in the configuration. Then it must be used twice in the second row of the configuration so it will appear either both in positions $1BL$, $2R$ or both in positions $1BR$, $2L$. But either way implies $\mathsf{ExistsFit}_{1a}(\mathcal{Q})$ (and $\mathsf{ExistsFit}_{1b}(\mathcal{Q})$), so it is impossible for this case to yield a successful last query.

Case 5 concludes all possible cases. Tallying the probabilities we obtain the upper bound $q(4m_a + m_a m_c + m_b^2 m_c)/N' + q(2m_b + 2m_b m_c)/N''$. □

Proposition 12. $\Pr[\mathsf{Win6}(\mathcal{Q})] \leq q(2m_a m_b + 2m_b^2)/N' + 4q m_b^2/N''$.

Proof. We first upper bound $\Pr[\mathsf{NotWin5}(\mathcal{Q}) \wedge \mathsf{ExistsFit}_{6a}(\mathcal{Q})]$. The queries of configuration $6a$ are labeled $1T$, $1B$, $2T$, $2R$ since they are all left column queries.

Case 1: The last query is used exactly once in the configuration. We can assume it appears in Word 1.

Subcase 1.1: The last query is used at position $1T$. Since queries $1B$, $2B$ are distinct there are at most m_a different possibilities for that pair in $\mathcal{Q}_{i-1}$. Then for each query $2B$ there are at most m_b queries $2T$ so in total there are most $m_a m_b$ possibilities for the pair $1B$, $2T$. Since the outputs of the last query are determined by that pair (whether the last query is forwards or backwards) the last query has chance $\leq m_a m_b/N'$ of being successful and the total chance of success if $\leq q m_a m_b/N'$.

Subcase 1.2: The last query is used at position $1B$. Given the key input of the last query there are at most m_b possibilities for query $1T$ and for each of those at most m_b different possibilities for the query $2T$ (that shares its right XOR output with $1T$), each of which then uniquely determine the query $2B$ (together with the inputs of the last query—if the last query is forward we know the word input of $2B$ and if the last query is backward we know its output). Therefore the last query has chance $\leq m_b^2/N'$ of succeeding, for a total probability of success $\leq q m_b^2/N'$.

Case 2: The last query is used twice or more in the configuration. Then the last query must be used exactly twice and by symmetry we can assume it is used either in positions $1T$, $1B$ or else in positions $1T$, $2B$.

Subcase 2.1: The last query is used in positions $1T$, $1B$. The left output of the query has chance $\leq N^{1/2}/N'$ of succeeding since it must match the left half of the key. If the left half of output succeeds there are at most m_b queries $2B$ with that left half of XOR output and for each query $2B$ at most m_b queries $2T$ with the left half of XOR output to match the left half of key of query $2B$. Since the half of XOR output of the last query must match the right half of XOR output of query $2T$, the right half has chance $\leq m_b^2/(N^{\frac{1}{2}} - m_b)$ of succeeding. Therefore the total chance of success is $\leq q m_b^2/N''$.

Subcase 2.2: The last query is used in positions $1T$, $2B$. There are at most m_b different choices for query $2T$ given the key input of the last query and the right half of XOR output of query $2T$ needs to be equal to the right half of XOR output of the last query, so the right half of output of the last query has chance $\leq m_b N^{\frac{1}{2}}/N'$ of succeeding. If the right half is successful then there are most m_b different queries for position $1B$ (with that right half of XOR output) so the left half of output has chance $\leq m_b/(N^{\frac{1}{2}} - m_b)$ of being successful. The total probability of success is therefore $\leq q m_b^2/N''$.

Case 2 completes all cases. We have examined 3 cases in all, and there are 3 more identical cases for configuration $6c$ so the overall chance of success is $\leq q(2m_a m_b + 2m_b^2)/N' + 4q m_b^2/N''$. □

Proposition 13. $\Pr[\mathsf{Win7}(\mathcal{Q})] \leq q(m_b^4 + 2m_a m_b + 2m_a)/N' + 4q m_b^2/N''$.

Proof. It suffices to upper bound $\Pr[\mathsf{NotWin6}(\mathcal{Q}) \wedge \mathsf{ExistsFit}_{7a}]$ because configuration $7b$ is a different drawing of configuration $7a$ (included for convenience).

Case 1: The last query is used only once. We can assume it is used in Word 1.

Subcase 1.1: The last query is used in position $1BL$. There are at most m_b possible choices for query $1TL$ and m_b possible choices for query $1TR$. Then for every query $1TL$ there are at most m_b possible choices for query $2TL$ and for every query $1TR$ there are at most m_b possible choices for query $2TR$. This makes a total of m_b^4 possible choices for the pair $2TL$, $2TR$ and thus m_b^4 possible choices for query $2BR$ (if the last query is forward then we know the word input of $2BR$, otherwise we know its output). So the total probability of success is $\leq q m_b^4/N'$.

Subcase 1.2: The last query is used in position $1TL$. Since the queries $1BL$ and $2BL$ are distinct there are most m_a possible choices for that pair of queries. Then for any query $2BL$ there are at most m_b possible choices for query $2TL$. Since the output of the last query is uniquely determined by the queries $1BL$, $2TL$ (whether the last query is forward or backward) the total chance of success is $\leq q m_a m_b/N'$.

Subcase 1.3: The last query is used in position $1TR$. Like above, there are m_a possible pairs $1BL$, $2BL$ and for each query $2BL$ there are m_b possible queries $2TR$, so the total probability of success is $\leq q m_a m_b/N'$.

This concludes Case 1.

Case 2: The last query is used twice or more in the configuration. Then it is used at least once in the top row, and we can assume in the top row of Word 1. If the last query appears both in positions $1TL$, $2TL$ then queries $1TR$, $2TR$ must be different (or queries $1BL$, $2BL$ could not be different) so this would imply $\mathsf{ExistsFit}_{6d}(\mathcal{Q})$, meaning the last query could not be successful. Likewise the last query cannot appear both in positions $1TR$, $2TR$ or else $\mathsf{ExistsFit}_{6a}(\mathcal{Q})$. By symmetry we can assume that the last query appears in position $1TL$ but not in position $2TL$ (we can revert to this case from the other case by interchanging the role of the left and right outputs; we simply need to remember to tally twice the probabilities of success for all the following subcases).

Subcase 2.1: The last query also appears in position $1BL$. Then the left output half of the last query has chance $\leq N^{\frac{1}{2}}/N'$ of being successful. If the left output half is successful there are at most m_b different queries $2BL$ with that left half of XOR output and for each of them at most m_b queries $2TL$, so the right output has chance $\leq m_b^2/(N^{\frac{1}{2}} - m_b)$ of being successful. The overall chance of success is therefore $\leq qm_b^2/N''$.

Subcase 2.2: The last query also appears in position $2BL$. Then given the key input of the last query there are at most m_b different queries $2TL$ so the right output half has chance $\leq m_b N^{\frac{1}{2}}/N'$ of being successful. If the right output half is successful then there are at most m_b queries $1BL$ with that right XOR output half and so the left output half has chance $\leq m_b/(N^{\frac{1}{2}} - m_b)$ of being successful. The total chance of success is therefore $\leq qm_b^2/N''$.

Subcase 2.3: The last query also appears in position $2TR$, and does not appear either in position $1BL$ or $2BL$ (or else we can revert to Subcase 2.1 or Subcase 2.2). Then there are at most m_a possibilities for the pair $1BL$, $2BL$ so the last query has chance $\leq m_a/N'$ of success and the total chance of success is $\leq qm_a/N'$. This concludes Case 2.

Case 2 concludes all cases. Tallying the probabilities yields the bound $q(m_b^4 + 2m_a m_b + 2m_a)/N' + 4qm_b^2/N''$. □

Proposition 14. $\Pr[\mathsf{Win8}(\mathcal{Q})] \leq q(m_a m_b^2 + m_a^2)/N' + q(2m_b + 2m_a m_b)/N''$.

Proof. Dispensing with comforting initial remarks:

Case 1: The last query is used only once in the configuration. We can assume WLOG that it is used in the left column of Word 1.

Subcase 1.1: The last query is used in position $1BL$. Since query $1BR$ is distinct from query $2BR$ there are only m_a possible pairs of queries $1BR$, $2BR$ and in particular only m_a possible choices for query $2BR$. For any query $2BR$ there are at most m_b queries $2TR$ and at most m_b queries $2TL$, so in total there are at most $m_a m_b^2$ possible choices for the query $2BL$. Therefore the last query (whether it is backwards or forwards) has chance $\leq m_a m_b^2/N'$ of being successful and the total chance of success is $\leq qm_a m_b^2/N'$.

Subcase 1.2: The last query is used in position $1TL$. There are at most m_a possible choices for the pair of queries $1BL$, $2BL$ and also at most m_a possible choices for the pair of queries $1BR$, $2BR$. In particular, this implies there are at most m_a^2 possible choices for the pair of queries $1BL$, $1BR$ so the last query has probability of success $\leq m_a^2/N'$. The total chance of success is $\leq qm_a^2/N'$.

Case 2: The last query is only used in the bottom row. Then it is used exactly twice, WLOG in positions $1BL$, $2BR$. But then $A = B$ and $\mathsf{ExistsFit}_{1a}(\mathcal{Q})$.

Case 3: The last query is only used in the top row. Then the analysis of Subcase 1.2 applies and we do not need to tally a new probability.

Case 4: The last query is used at least once in the bottom row and at least once in the top row. We can assume WLOG that it is used in position $1TL$. By the same reasoning as for Case 2, the last query must appear exactly once in the bottom row.

Subcase 4.1: The last query is also used in position $1BL$. Then the left output of the last query has chance $\leq N^{\frac{1}{2}}/N'$ of succeeding. There are $\leq m_b$ queries $2BL$ with that left output so the right output has probability $\leq m_b/(N^{\frac{1}{2}} - m_b)$ of succeeding. The total probability of success is $\leq qm_b/N''$.

Subcase 4.2: The last query is also used in position $1BR$. Same analysis as above but arguing first on the right output and then on the left, for a bound of qm_b/N''.

Subcase 4.3: The last query is also used in position $2BL$. There are then $\leq m_a$ different possibilities for the pair of queries $1BR$, $2BR$ so the right output of the last query has chance $\leq m_a N^{\frac{1}{2}}/N'$ of succeeding. Since the right XOR output of the last query is also the right XOR output of query $1BL$, if the right output is successful then there are at most m_b queries $1BL$ so the left output has chance $\leq m_b/(N^{\frac{1}{2}} - m_b)$ of succeeding. The total chance of success is therefore $\leq qm_a m_b/N''$.

Subcase 4.4: The last query is also used in position $2BR$. Same analysis as above but arguing first on the left output and then on the right, for a bound of $qm_a m_b/N''$. This completes Case 4.

Case 4 completes all possible cases. Tallying the probabilities yields $q(m_a m_b^2 + m_a^2)/N' + q(2m_b + 2m_a m_b)/N''$. □

Tallying the bounds of Propositions 7–14 we obtain that

$$\begin{aligned}
\Pr[\mathsf{WinFit}(\mathcal{Q})] \;\leq\;& q(1 + m_a + m_b^2)/N' + 2qm_b/N'' + \\
& q^2/N'^2 + \\
& q/N' + \\
& q(2 + 2m_b^2)/N' + q(4 + 4m_a + 4m_b)/N'' + \\
& q(4m_a + m_a m_c + m_b^2 m_c)/N' + q(2m_b + 2m_b m_c)/N'' + \\
& q(2m_a m_b + 2m_b^2)/N' + 4qm_b^2/N'' + \\
& q(m_b^4 + 2m_a m_b + 2m_a)/N' + 4qm_b^2/N'' \\
& q(m_a m_b^2 + m_a^2)/N' + q(2m_b + 2m_a m_b)/N'' \\
=\;& q(m_a^2 + m_a m_b^2 + m_b^4)/N' + \\
& q(4m_a m_b)/N' + q(2m_a m_b)/N'' + \\
& q(m_b^2 m_c + 5m_b^2 + m_a m_c + 6m_a)/N' + q(4m_a + 8m_b^2)/N'' + \\
& q(4 + 10m_b + 2m_b m_c)/N'' + 3q/N' + 4q/N'' + q^2/N'^2
\end{aligned}$$

which establishes Theorem 5.

B Appendix B: Proofs of Theorems 1, 2 and optimization of constants

To prove Theorem 1 it suffices to prove Theorem 4 since Theorem 5 is established in Appendix A. That is, we need to show

$$\Pr[\mathsf{Win0}(\mathcal{Q})] \leq \frac{q^2}{m_a N'} + 2qN^{\frac{1}{2}} e^{qN^{\frac{1}{2}} M_b(1-\ln(M_b))/N'} + qNe^{qM_c(1-\ln(M_c))/N'}$$

for all positive numbers m_a, m_b, m_c obeying the conditions of Theorem 1 and where $M_b = m_b N'/qN^{\frac{1}{2}}$, $M_c = m_c N'/q$ (the conditions of Theorem 1 being that $eqN^{\frac{1}{2}}/N' \leq m_b$, $eq/N' \leq m_c$).

To upper bound $\Pr[\mathsf{Win0}(\mathcal{Q})]$ it is sufficient to find separate bounds for $\Pr[a(\mathcal{Q}) > m_a]$, $\Pr[b(\mathcal{Q}) > m_b]$, $\Pr[c(\mathcal{Q}) > m_c]$ because $\Pr[\mathsf{Win0}(\mathcal{Q})] \leq \Pr[a(\mathcal{Q}) > m_a] + \Pr[b(\mathcal{Q}) > m_b] + \Pr[c(\mathcal{Q}) > m_c]$. We do each of these in a separate proposition.

Proposition 15. $\Pr[a(\mathcal{Q}) > m_a] \leq q^2/m_a N'$.

Proof. Let S_{ij} denote the event that queries i, j have the same XOR output. Then obviously $\Pr[S_{ij}] \leq 1/N'$ for all $i \neq j$ so $E[a(Q)] = \sum_{i \neq j} E[S_{ij}] \leq q^2/N'$. Using a Markov inequality we then get

$$\Pr[a(\mathcal{Q}) > m_a] \leq \frac{q^2}{m_a N'}$$

as desired. □

Proposition 16. $\Pr[b(\mathcal{Q}) > m_b] \leq 2qN^{\frac{1}{2}} e^{qN^{\frac{1}{2}} M_b(1-\ln(M_b))/N'}$ *if* $M_b \geq e$, *where* $M_b = m_b N'/qN^{\frac{1}{2}}$.

Proof. It is sufficient to show $\Pr[b^L(\mathcal{Q}) > m_b] \leq qN^{\frac{1}{2}} e^{qN^{\frac{1}{2}} M_b(1-\ln(M_b))/N'}$ since by symmetry $\Pr[b^L(\mathcal{Q}) > m_b] = \Pr[b^R(\mathcal{Q}) > m_b]$ and therefore $\Pr[b(\mathcal{Q}) > m_b] \leq 2\Pr[b^L(\mathcal{Q}) > m_b]$.

We can rephrase the problem of upper bounding $\Pr[b^L(\mathcal{Q}) > m_b]$ as a balls-in-bins question. Let $r = N^{\frac{1}{2}}$ be the number of bins and q be the number of balls to be thrown. The i-th ball falls into the j-th bin if the left half of the XOR output of the i-th query is equal to j. Our problem is to upper bound the probability that some bin contains more than m_b balls. The i-th ball always has probability $\leq p = N^{\frac{1}{2}}/N'$ of falling in the j-th bin (regardless of how previous balls were thrown) because when the XOR output of the i-th query is chosen uniformly at random from a set of size at least N', of which at most $N^{\frac{1}{2}}$ members have left half j. If we let B_k be the probability of having exactly k balls fall in a particular bin (say bin 1) then

$$B_k \leq p^k \binom{q}{k}$$

Let $\mu = qp$ (note μ is an upper bound for the expected number of balls in any bin, and that $m_b = M_b\mu$). By Stirling's approximation

$$\begin{aligned} B_k &\leq p^k \frac{1}{\sqrt{2\pi}} \sqrt{\frac{q}{k(q-k)}} \frac{q^q}{k^k (q-k)^{q-k}} \\ &\leq \frac{1}{k^k} \mu^k \left(\frac{q}{q-k}\right)^{q-k} \\ &\leq \mu^k \frac{e^k}{k^k} \end{aligned}$$

Since $m_b = M_b\mu$ we get

$$\begin{aligned} B_{m_b} &\leq \frac{e^{\mu M_b}}{{M_b}^{M_b}} \\ &= e^{\mu M_b(1-\ln(M_b))}. \end{aligned}$$

Now note that B_k is a decreasing function of k after $k = e\mu$, so if $M_b \geq e$ we have

$$\begin{aligned} \Pr[b^L(\mathcal{Q}) > m_b] &\leq N^{\frac{1}{2}} \sum_{j=m_b}^{q} B_j && (3) \\ &\leq qN^{\frac{1}{2}} B_{m_b} && (4) \\ &\leq qN^{\frac{1}{2}} e^{\mu M_b(1-\ln(M_b))} && (5) \end{aligned}$$

as desired. □

Proposition 17. $\Pr[c(\mathcal{Q}) > m_c] \leq qNe^{qM_c(1-\ln(M_c))/N'}$ *if* $M_c \geq e$, *where* $M_c = m_c N'/q$.

Proof. We skip most of the details since the computation is essentially the same as for Proposition 16, but this time with N bins instead of $N^{\frac{1}{2}}$ and with probability of a ball falling in a given bin $\leq p = 1/N'$ (instead of $\leq N^{\frac{1}{2}}/N'$). The corresponding value to μ is $\nu = qp = q/N'$ so $m_c = M_c\nu$ and if $M_c \geq e$ then we have

$$\begin{aligned}\Pr[c(\mathcal{Q}) > m_c] &\leq qNe^{\nu M_c(1-\ln(M_c))} \\ &= qNe^{qM_c(1-\ln(M_c))/N'}\end{aligned}$$

as desired. □

Proposition 17 finishes the proof of Theorem 4 and thus of Theorem 1.

Before proving Theorem 2 we discuss how to best optimize the constants m_a, m_b, m_c for given values of n and q. Note that we do not require the best possible values of m_a, m_b, m_c for the bound to be corect—any values of m_a, m_b, m_c which meet the conditions of Theorem 1 yield a valid upper bound.

The value of m_a is the easiest to optimize. Since the dominating term with m_a in $\Pr[\mathsf{WinFit}(\mathcal{Q})]$ is qm_a^2/N', optimizing m_a roughly amounts to minimizing the sum $q^2/m_aN' + qm_a^2/N'$. Taking the derivative and setting to 0 we find that a near-optimal value for m_a is $\sqrt[3]{q/2}$.

Since the dominating term involving m_b in $\Pr[\mathsf{WinFit}(\mathcal{Q})]$ is qm_b^4/N', an optimal M_b (and hence m_b) will be chosen such as to minimize the sum

$$2qN^{\frac{1}{2}}e^{\mu M_b(1-\ln(M_b))} + q\mu^4 M_b^4/N' \tag{6}$$

subject to the condition that $M_b \geq e$. Unfortunately one cannot take the derivative of (6) and solve directly for M_b since this leads to a transcendental equation. Instead one can obtain a numerical approximation by an iterating a procedure which attempts to make the two sides of the sum (6) as equal as possible. One can also do slightly better by aiming to make $q\mu^4 M_b^4/N'$ about 100 (say) times larger than $2qN^{\frac{1}{2}}e^{\mu M_b(1-\ln(M_b))}$, since a small change in M_b can greatly decrease $2qN^{\frac{1}{2}}e^{\mu M_b(1-\ln(M_b))}$ without increasing $q\mu^4 M_b^4/N'$ by nearly as much.

The biggest terms involving m_c in $\Pr[\mathsf{WinFit}(\mathcal{Q})]$ are $q(m_b^2 m_c + m_a m_c)/N'$ so an optimal value of m_c will be chosen such as to minimize the sum

$$qNe^{qM_c(1-\ln(M_c))/N'} + q^2(m_b^2 M_c + m_a M_c)/N'^2. \tag{7}$$

Like for M_b there is no way to find an exact solution for the optimal value of M_c, so M_c must be found numerically. Since the terms involving M_c represent much smaller terms than the largest terms involving m_a, m_b one may choose m_a, m_b first and then choose m_c such as to minimize (7). However the value of m_c is much less critical than the values of m_a, m_b, precisely because the terms involving m_c are so much smaller than the largest terms involving m_a, m_b. Naturally, m_b and m_c are always chosen within the allowed ranges (this doesn't turn out to be a relevant restriction since the optimal values are always in those ranges anyway).

Our last task is prove Theorem 2, which asserts that if $q = N^{\frac{3}{5}-\epsilon}$ then $\Pr[\mathsf{Coll}^{\mathrm{MDC2}^E}(\mathcal{Q})] \to 0$ as $n \to \infty$. So let $q = N^{\frac{3}{5}-\epsilon}$ where $\epsilon > 0$ is a constant. We can assume $\epsilon < 1/10$ because $\Pr[\mathsf{Coll}^{\mathrm{MDC2}^E}(\mathcal{Q})]$ is monotone increasing with q and that n is large enough that $N'/q > 1$. We set $m_a = q^{\frac{1}{3}}$, $M_b = e^2$, $M_c = (N'/q)^{1+\alpha}$ where $0 < \alpha < \frac{1}{5}$. It is sufficient to check that the terms

$$\frac{q^2}{m_aN'},\ \ 2qN^{\frac{1}{2}}e^{qN^{\frac{1}{2}}M_b(1-\ln(M_b))/N'},\ \ qNe^{qM_c(1-\ln(M_c))/N'}, qm_a^2/N',\ \ qm_b^4/N', qm_b^2m_c/N'',\ \ qm_am_c/N' \tag{8}$$

go to zero because other terms in the upper bound of Theorem 1 are bounded by these. Because $N/N' \to 1$ as $n \to \infty$ it is straightforward to check that terms q^2/m_aN' and qm_a^2/N' of (8) go to zero as $n \to \infty$. For example substituting $m_a = q^{\frac{1}{3}} = N^{\frac{1}{5}-\frac{1}{3}\epsilon}$ into q^2/m_aN' gives $N^{1-\frac{5}{3}\epsilon}/N'$, which goes to zero. When $M_b = e^2$ the term $2qN^{\frac{1}{2}}e^{qN^{\frac{1}{2}}M_b(1-\ln(M_b))/N'}$ becomes $2qN^{\frac{1}{2}}e^{-e^2qN^{\frac{1}{2}}/N'}$ which one easily checks goes to 0 since e^{N^r} grows faster than N^s for any $r, s > 0$. After substitution of $M_c = (N'/q)^{1+\alpha}$ the term $qNe^{qM_c(1-\ln(M_c))/N'}$ becomes $\leq qNe^{-(N'/q)^\alpha}$ which again goes to 0 for the same reason. The term qm_b^4/N' becomes $e^8q^5/N^2N' = e^8N^{3-5\epsilon}/N^2N'$ which goes to 0 because $N'/N \to 1$. The remaining terms $qm_b^2m_c/N''$, qm_am_c/N' are checked likewise. This establishes Theorem 2 and concludes our results.