# Quantum preimage, 2$^{\text{nd}}$-preimage, and collision resistance of SHA3

Jan Czajkowski[1], Leon Groot Bruinderink[2], Andreas Hülsing[2], and Christian Schaffner[1]

[1] University of Amsterdam, CWI, QuSoft
Institute for Logic, Language and Computaion (ILLC)
P.O. Box 94242, 1090 GE Amsterdam, The Netherlands
`j.czajkowski@uva.nl , c.schaffner@uva.nl`

[2] Department of Mathematics and Computer Science
Technische Universiteit Eindhoven
P.O. Box 513, 5600 MB Eindhoven, The Netherlands
`l.groot.bruinderink@tue.nl,andreas@huelsing.net`

**Abstract.** SHA3 and its extendable output variant SHAKE belong to the family of sponge functions. In this work, we present formal security arguments for the quantum preimage, 2$^{\text{nd}}$-preimage, and collision resistance of any sponge function. We just assume that the internally used transformation behaves like a random transformation. These are the first formal arguments that sponge functions (incl. SHA3 and SHAKE) are secure in the post-quantum setting.
We even go one step further and prove that sponges are collapsing (Unruh, EUROCRYPT'16). Thereby, we can also derive the applicability of sponge functions for collapse-binding commitments.
In addition to the security arguments, we also present a quantum collision attack against sponges. The complexity of our attack asymptotically matches the proven lower bound up to a square root.

**Keywords:** Post-quantum cryptography; SHA3, SHAKE, sponges, keccak, hash function, quantum security, quantum collision resistance, quantum second-preimage resistance, quantum preimage resistance.

## 1 Introduction

(Cryptographic) hash functions are one of the most basic building blocks in cryptography. They are virtually used everywhere: As cryptographically secure checksums to verify integrity of software or data packages, as building block in security protocols, including TLS, SSH, IPSEC, as part of any efficient variable-input-length signature scheme, to build full-fledged hash-based signature schemes, in transformations for CCA-secure encryption, and many more.

This work was supported in part by the Commission of the European Communities through the Horizon 2020 program under project number 645622 PQCRYPTO. CS and JC are supported by a NWO VIDI grant. Permanent ID of this document: `287718adca85c4a117f2d334d94d1b08`. Date: April 4, 2017

While all widely deployed public-key cryptography is threatened by the rise of quantum computers, hash functions are widely believed to only be mildly effected. The reason for this is twofold. On the one hand, generic quantum attacks achieve at most a square-root speed up compared to their pre-quantum counterparts and can be proven asymptotically optimal [4,14,9]. On the other hand, there do not exist any dedicated quantum attacks on any specific hash function (excluding of course those based on number theory like, e.g., VSH [6]) that perform better than the generic quantum attacks.

However, in the pre-quantum setting we can give stronger security arguments than just the absence of dedicated attacks. For modern hash functions like SHA2 or SHA3 there exist proofs that show security properties of the hash function, assuming that an internal building block of the construction has a certain property. For example, the SHA3 hash function is an instantiation of the sponge construction [3]. For sponge functions it was shown [3] that if the internally used permutation (or function) behaves like a random permutation (or function), then the sponge function achieves one of the strongest security properties possible: indifferentiability from a random oracle. This property guarantees that there do not exist any attacks that perform better than generic attacks against SHA3 as long as the permutation used in SHA3 behaves like a random permutation.

Sadly, the security reduction for sponge functions does not carry over to the quantum setting. The key issue here is that the core of the security argument is query-based. Informally, omitting a lot of details, the core-argument goes in two steps:

1. The only source of non-random behavior are collisions in the internal state of the sponge.
2. Any adversary making a polynomially bounded number of (classical) queries has only negligible chance to see such an inner collision.

For this discussion it is irrelevant what inner collisions exactly are as these arguments do not work against a quantum adversary $\mathcal{A}$. Such an adversary can use a quantum circuit implementing SHA3 and can thereby query the function in superposition. In particular, $\mathcal{A}$ could execute SHA3 on the uniform superposition over all messages of a certain length. For suitable parameters, $\mathcal{A}$ is actually guaranteed that for several branches of the superposition inner collisions occur, possibly helping $\mathcal{A}$ to distinguish SHA3 from a random oracle. This leaves us with no security argument for SHA3 besides the absence of attacks which is an unfortunate situation.

**Our contribution.** In this paper, we solve this issue for the most fundamental security properties of a hash function. We provide formal security arguments for the quantum preimage, $2^{\text{nd}}$-preimage, and collision resistance of any sponge function under the assumption that the internally used permutation (or function) behaves like a random permutation (or function). Actually, we even go one step further and prove that sponges are collapsing under the same assumption. Collapsing is a property of (hash) functions, defined by Unruh [12] in the context of post-quantum secure commitment schemes – so called collapse-binding commitments. As a side-product we thereby also show the applicability of sponge functions for collapse-binding commitments. As collapsing tightly implies collision resistance, which in turn tightly implies second-preimage and (for strongly compressing functions) preimage resistance, we immediately derive the claimed result from this proof. As our main result, we prove the following theorem (using terminology from Section 2) :

**Theorem 1.** *Let* $\mathrm{Sponge}[f, \mathrm{pad}, r]$ *be a sponge construction with capacity* $c$ *and bitrate* $r$*. Moreover, assume that a minimum output length* $\ell$ *is given. The success probability of any quantum collision finder* $\mathcal{A}$*, making at most* $q$ *queries, is upper bounded by* $\mathcal{O}(\sqrt{q^3 \cdot \max(2^{-\ell}, 2^{-r}, 2^{-c})})$*.*

In addition to our constructive result, we also present a quantum attack for the same setting, providing an upper bound for security. We describe a quantum attack against any sponge function that internally uses a random function (or random permutation). Our attack has a success probability of $\mathcal{O}(q^3 \cdot \max(2^{-\ell}, 2^{-c}))$ when making $q$ quantum queries to an oracle, implementing the internal function. This matches our bound from Theorem 1 up to a square root, showing our bound is almost optimal.

**Related work.** In [12], Unruh introduces collapsing hash functions as technical tool to construct collapse-binding commitments. In the same work he shows that random oracles are collapsing. In a follow-up work [11], Unruh presents the first standard-model instantiation of collapsing hash functions and thereby collapse-binding commitments. As a step in this proof, he shows that hash functions using the Merkle-Damgård construction are collapsing if the internal compression function is. Thereby he implicitly gave a formal argument for the preimage, 2nd-preimage, and collision resistance of the SHA family (assuming the compression function behaves like a random oracle).

Brassard, Høyer, and Tapp [4] presented the first generic quantum algorithm for collision search in any $r$-to-1 function. Zhandry [14] extends this result, presenting upper and lower bounds for quantum collision search for random functions. In [9], Hülsing, Song, and Rijneveld present upper and lower bounds for quantum preimage and second-preimage search in the case of random functions. In [1], the authors give detailed cost estimates for Grover's algorithm when used to find preimages in SHA2 and SHA3.

**Organization.** In Section 2, we revisit some preliminaries. In Section 3, we begin with showing that the internal mechanisms used in the sponge are collapsing. Afterwards, we prove our main technical result, Lemma 6, from which we derive our main theorem and its implications for the security of SHA3. In Section 4 we describe a quantum algorithm for finding collisions in sponge function.

## 2 Preliminaries

In this section we briefly introduce quantum computing as needed for this work. We revisit terminology and notations of sponge constructions, which is used through out this paper. Next we recall the definition of collapsing and two related results, we end the section with description of algorithm by [4] and discuss its performance.

**Basic notations.** We say that $\varepsilon = \varepsilon(n)$ is negligible if, for all polynomials $p(n)$, $\varepsilon(n) < 1/p(n)$ for large enough $n$. With $x \leftarrow_{\$} X$, we denote sampling an element $x$ at random from set $X$. With $\mathbf{0}$ we denote the all zero bitstring.

**Quantum computing.** We assume the reader is familiar with the usual notations in quantum computation, but we give a very short introduction here. A quantum system $A$ is a complex Hilbert space $\mathcal{H}$, together with an inner product $\langle\cdot|\cdot\rangle$.The state of a quantum system is given by a vector $|\psi\rangle$ of unit norm ($\langle\psi|\psi\rangle = 1$). A joint system of $\mathcal{H}_1$ and $\mathcal{H}_2$ is denoted by $\mathcal{H} = \mathcal{H}_1 \otimes \mathcal{H}_2$, with elements $|\psi\rangle = |\psi_1\rangle |\psi_2\rangle$ for $|\psi_1\rangle \in \mathcal{H}_1, |\psi_2\rangle \in \mathcal{H}_2$. A unitary transformation $\mathbf{U}$ over a $d$-dimensional Hilbert space $\mathcal{H}$ is a $d \times d$ matrix $\mathbf{U}$ such that $\mathbf{U}\mathbf{U}^\dagger = \mathbf{I}_d$, where $\mathbf{U}^\dagger$ represents the conjugate transpose. In this paper, we assume quantum-accessible oracles, i.e., we will implement an oracle $\mathcal{O} : X \to Y$ by a unitary transformation $\mathbf{O}$ such that: $\mathbf{O}|x, y\rangle = |x, y + \mathcal{O}(x)\rangle$, where $+ : Y \times Y \to Y$ is some group operation on $Y$. Any quantum algorithm making $q$ queries can then be written as a final transformation $\mathbf{U}_q\mathbf{O}_q \ldots \mathbf{U}_1\mathbf{O}_1\mathbf{U}_0$ for unitaries $\mathbf{U}_i$ applied between queries and oracle queries $\mathbf{O}_i$.

For a quantum register $M$ we denote by $m \leftarrow \mathcal{M}_{\text{comp}}$ the measurement of $M$ in the computational basis with outcome $m$. For a function $f$, we denote by $\mathcal{M}_f(M)$ the projective measurement of register $M$ with projectors $P_y = \sum_{\mathbf{m}:f(\mathbf{m})=y} |\mathbf{m}\rangle\langle\mathbf{m}|$. In other words, applying $y \leftarrow \mathcal{M}_f(M)$ causes $M$ to collapse to superpositions of values $\mathbf{m}$ which all map to the same image $y$ under $f$. We will recall some notations and a definition from [13]. Given two sets $X$ and $Y$, define $Y^X$ as the set of functions $f : X \to Y$. If a function $f$ maps $X$ to $Y \times Z$, we can think of $f$ as two functions: one that maps $X$ to $Y$ and one that maps $X$ to $Z$. We will define quantum indistinguishability from random in the following sense for function families $\mathcal{F}$.

**Definition 1 (Quantum-Indistinguishability).** *We call a family of functions $\mathcal{F} \subseteq Y^X$ quantum-indistinguishable from random if no efficient quantum adversary $\mathcal{A}$ making quantum queries can distinguish between a function drawn at random from $\mathcal{F}$ and a truly random function. That is, for every security parameter $n$ and quantum adversary $\mathcal{A}$, there exists a negligible function $\varepsilon = \varepsilon(n)$ such that:*

$$\mathrm{Adv}^{QI}_{Y^X}(\mathcal{F}; \mathcal{A}) := |\underset{f \leftarrow_{\$} \mathcal{F}}{\mathbb{P}}[A^f(1^n) = 1] - \underset{\mathcal{O} \leftarrow_{\$} Y^X}{\mathbb{P}}[A^{\mathcal{O}}(1^n) = 1]| < \varepsilon$$

*We call* $\mathrm{Adv}^{QI}$ *the* oracle-distinguishing advantage.

**The sponge construction.** The concept of cryptographic sponges was introduced in [3]. A sponge is a function $\mathrm{Sponge}[f, \mathrm{pad}, r] : \{0,1\}^* \to \{0,1\}^\infty$ that uses a fixed-length transformation (a function or a permutation) $f$ (also called internal function in this paper), a sponge-compliant padding rule pad and a parameter bitrate $r$. Padding is a procedure used to prepare input in a form suitable for the sponge construction. It increases the length of the input message $\mathbf{M}$ so that the length of $\mathbf{M}\|\text{pad}[r](|\mathbf{M}|)$ is a multiple of $r$. We will use specific padding functions called sponge-compliant,

**Definition 2.** *A padding rule is* sponge-compliant *if it never results in the empty string and if it satisfies the following criterion:*

$$\forall n \geq 0\ \forall \boldsymbol{M}, \boldsymbol{M'} \in \{0,1\}^* : \boldsymbol{M} \neq \boldsymbol{M'} \Rightarrow \boldsymbol{M}\|\text{pad}[r](|\boldsymbol{M}|) \neq \boldsymbol{M'}\|\text{pad}[r](|\boldsymbol{M'}|)\|0^{nr}.$$

An example of such padding rule is pad10*, which appends a single bit 1 followed by the minimum number of bits 0 in order to get a multiple of $r$. A finite length output can be obtained by truncating the theoretically infinite output to its first $\ell$ bits. For any message $\mathbf{M} \in \{0,1\}^*$, we denote the $\ell$ bit truncated output by $Z = \text{Sponge}[f, \text{pad}, r](\mathbf{M}, \ell)$. A graphical description is shown in Figure 1.

The transformation $f$ operates on $r + c$ bits, which we call the state. We call the first $r$ bits of the state the *outer part* and the last $c$ bits the *inner part*. One of the security parameters of the sponge construction is this inner part $c$ of the state, which is hidden from the user. The size of this inner part $c$ is called the capacity. For a state $s \in \{0,1\}^{r+c}$, we denote its outer part with an overline and the inner part with a hat: $s = (\overline{s}, \widehat{s})$ with $\overline{s} \in \{0,1\}^r$ and $\widehat{s} \in \{0,1\}^c$.

First, the state is initialized to zero. The input message is padded using pad and cut into $r$-bit blocks. Then the sponge proceeds in two phases. In the *absorbing phase*, the first $r$ bits of the state (the outer part) are XORed with the $r$-bit message blocks, interleaved with applications of transformation $f$. When all message blocks are processed, the sponge switches to the *squeezing phase*. In this phase, the outer part of the state is iteratively returned as output blocks, again, interleaved with applications of $f$. The number of iterations is determined by the requested number of output bits $\ell$. In particular, this means for $\ell \leq r$ that no additional applications of $f$ are needed after the absorbing phase.

In the following we will assume oracle access to the sponge. It is important to specify the cost of each query. Different queries can not always be equally costly, since there might be both varying length input and varying length output. The cost of applying Sponge is the number of evaluations of the internal function $f$. So evaluating $\text{Sponge}[f, \text{pad10}^*, r](\mathbf{M}, \ell)$ costs $\left\lceil \frac{|\mathbf{M}|+1}{r} \right\rceil + \left\lceil \frac{\ell}{r} \right\rceil$ queries. This could vary slightly if a different padding rule is used.

**Collapsing hash functions.** At EUROCRYPT 2016 [12], Unruh introduced the notion of collapsing. This is a purely quantum notion, which is defined for a function $H$ as follows:

**Definition 3.** *[12, Definition 23] For a function $H$ and algorithms $\mathcal{A}, \mathcal{B}$, consider the following games:*

$$\begin{aligned} &\mathsf{Game}_1 : (S, M, c) \leftarrow \mathcal{A}(1^n), && m \leftarrow \mathcal{M}_{comp}(M), b \leftarrow \mathcal{B}(1^n, S, M) \\ &\mathsf{Game}_2 : (S, M, c) \leftarrow \mathcal{A}(1^n), && b \leftarrow \mathcal{B}(1^n, S, M) \end{aligned}$$

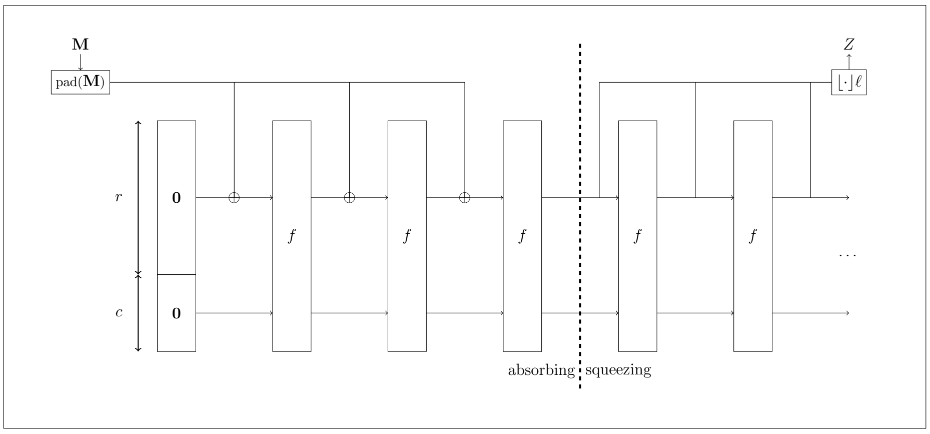


Fig. 1: The sponge construction $Z = \text{Sponge}[f, \text{pad}, r](\mathbf{M}, \ell)$

*Here, $S, M$ are quantum registers and $\mathcal{M}_{comp}(M)$ is a measurement of $M$ in the computational basis. We call an adversary $(\mathcal{A}, \mathcal{B})$ valid if $\mathbb{P}[H(m) = c] = 1$ when we run $(S, M, c) \leftarrow \mathcal{A}(1^n)$ and measure $M$ in the computational basis as $m$.*

*A function $H$ is collapsing iff for any valid quantum-polynomial-time adversary $(\mathcal{A}, \mathcal{B})$, the difference* $\text{Adv}_H^{coll}(A, B) := |\mathbb{P}[b = 1 : \mathsf{Game}_1] - \mathbb{P}[b = 1 : \mathsf{Game}_2]|$ *is negligible. We call* $\text{Adv}_H^{coll}$ *the* collapsing advantage.

Two related results from [12], which we use are:

**Theorem 2.** *[12, Theorem 38] Let $Y$ be finite, and $X \subseteq \{0,1\}^*$ (finite or infinite). Then $H \leftarrow_\$ Y^X$ is collapsing with advantage $\mathcal{O}(\sqrt{q^3/|Y|})$.*

**Lemma 1.** *[12, Lemma 25] A collapsing function is collision resistance.*

*Proof sketch*: A tight reduction is given in [12]. The basic idea is to use two colliding messages $(m, m')$ with $H(m) = H(m') = c$, and initialize register $M$ with $|\psi_{m,m'}\rangle = \frac{1}{\sqrt{2}}(|m\rangle + |m'\rangle)$. Algorithm $B$ is able to detect a measurement of register $M$ with non-negligible probability, using the projective measurement $|\psi_{m,m'}\rangle\langle\psi_{m,m'}|$. Details are given in the original paper.

In another recent paper by Unruh [11], it is proven that the Merkle-Dåmgard construction is collapsing when the compression function is. From this result and Lemma 1, it follows that the Merkle-Dåmgard construction is collision resistant.

**BHT algorithm** For the attack on the sponge function, We base our results on the quantum collision-finding algorithm developed by Brassard, Høyer and Tapp [4]. This is proved optimal by Zhandry [14]. Consider a function $f : X \to Y$, with $X = \{0,1\}^n$. The BHT algorithm for finding a collision in $f$ that makes $q$ queries, first creates a classical table $L$ of pairs $(x, f(x))$ of size $\frac{q}{3}$. Then the algorithm uses Grover search (see Appendix B) to find an element of the domain that was not previously queried, but has the same image as some $x \in L$.

That way, BHT outputs a pair of colliding inputs. The performance of BHT was investigated in [13,14], the results are as follows:

**Theorem 3.** *For $|Y| = N$ and $|X| \in \Omega(N^{2/3})$ the success probability of* BHT*, making $q$ queries to $f : X \to Y$ is bounded by $\mathrm{Succ}^q_N(\text{BHT}) \leq 27\frac{(q+2)^3}{N}$, where the probability is taken over the random choice of $f$.*

## 3 The security of Sponge hashes

At the end of this section, we will proof Theorem 1, which shows the collision resistance of the sponge construction. We do this by showing that the sponge construction is collapsing. This requires the internal mechanisms used in the sponge to be collapsing, which we will show first.

**Collapsing internal functions.** The sponge construction uses an internal function $f : \{0,1\}^{r+c} \to \{0,1\}^{r+c}$, which is either a random function $f^R$ or a random permutation $f^\pi$. Since the function maps values in $X = \{0,1\}^{r+c}$ into two spaces $Y_0 = \{0,1\}^r$ and $Y_1 = \{0,1\}^c$ with $|X| = |Y_0| \cdot |Y_1|$, we can think of this function as two functions $f_0$ and $f_1$, mapping $X$ to $Y_0$ and $Y_1$, respectively. In the following, let $P_i(f(x))$ be the projection of $f(x)$ onto $Y_i$ (this means $P_i(f(x)) = f_i(x)$) for $i \in \{0,1\}$.

**Lemma 2.** *Let $\mathcal{F} \subseteq (Y_0 \times Y_1)^X$ be a family of quantum-secure functions, i.e., for any efficient quantum adversary $\mathcal{A}$ and security parameter $n$, there exists a negligible function $\varepsilon = \varepsilon(n)$ with*

$$\mathrm{Adv}^{QI}_{(Y_0\times Y_1)^X}(\mathcal{F}; A) := |\underset{\mathcal{O}\leftarrow_\$\mathcal{F}}{\mathbb{P}}[\mathcal{A}^{\mathcal{O}}(1^n) = 1] - \underset{\mathcal{O}\leftarrow_\$(Y_0\times Y_1)^X}{\mathbb{P}}[\mathcal{A}^{\mathcal{O}}(1^n) = 1]| < \varepsilon$$

*Let $\mathcal{F}_i \overset{def}{=} \{P_i(f) | f \in \mathcal{F}\}$, i.e., the family of the projections of all elements of $\mathcal{F}$ onto $Y_i$, for $i \in \{0,1\}$. Then for all efficient quantum adversaries $\mathcal{A}_i$ and all $i \in \{0,1\}$:*

$$\mathrm{Adv}^{QI}_{Y_i^X}(\mathcal{F}; \mathcal{A}_i) := |\underset{\mathcal{O}\leftarrow_\$\mathcal{F}_i}{\mathbb{P}}[\mathcal{A}_i^{\mathcal{O}}(1^n) = 1] - \underset{\mathcal{O}\leftarrow_\$Y_i^X}{\mathbb{P}}[\mathcal{A}_i^{\mathcal{O}}(1^n) = 1]| \leq \varepsilon$$

*Proof.* Suppose there exists an adversary $\mathcal{A}_i$ such that $\mathrm{Adv}^{QI}_{Y_i^X}(\mathcal{F}; \mathcal{A}_i) = \mu > \varepsilon$ for arbitrary but fixed $i \in \{0,1\}$. We will now construct an oracle machine $\mathcal{M}^{\mathcal{A}_i}$ to distinguish $f \leftarrow_\$ \mathcal{F}_i$ from a random function with $\mathrm{Adv}^{QI}_{(Y_0\times Y_1)^X}(\mathcal{F}; \mathcal{M}^{\mathcal{A}_i}) > \varepsilon$.

The oracle machine $\mathcal{M}^{\mathcal{A}_i}$ is given black box access to $\mathcal{O}$ which is either randomly drawn from $\mathcal{F}$ or $Y^X$, and distinguishes the two cases as follows:

1. Construct function $g : X \to Y_i$ by $g(x) = P_i(\mathcal{O}(x))$.
2. Simulate $\mathcal{A}_i$ with function $g$, and output whatever $\mathcal{A}_i$ outputs.

It is easy to see that $\mathrm{Adv}^{QI}_{(Y_0\times Y_1)^X}(\mathcal{F}; \mathcal{M}^{\mathcal{A}_i}) = \mu > \varepsilon$. In particular, when $\mathcal{M}^{\mathcal{A}_i}$ is given access to an element of $\mathcal{F}$, $\mathcal{A}_i$ will be given access to $g \in \mathcal{F}_i$. When $\mathcal{M}^{\mathcal{A}_i}$ is given a random $\mathcal{O} \leftarrow_\$ Y^X$, $\mathcal{A}_i$ will be given $g = P_i(\mathcal{O})$ which is randomly distributed in $Y_i^X$. Hence, the advantage of $\mathcal{M}^{\mathcal{A}_i}$ will be exactly $\mu$. □

Let $\mathcal{F}^R \subset (Y_0 \times Y_1)^X$ denote the family of random functions and let $\mathcal{F}^\pi \subset (Y_0 \times Y_1)^X$ denote the case of a family of random permutations. When a random function $f^R$ is used as the internal function, we have that $\mathrm{Adv}^{QI}_{(Y_0\times Y_1)^X}(\mathcal{F}^R; \mathcal{A}) = 0$ for any quantum adversary $\mathcal{A}$, since the output distributions for these functions are the same. This means that also for $f_0^R$ and $f_1^R$ as defined above we have $\mathrm{Adv}^{QI}_{Y_i^X}(\mathcal{F}^R; \mathcal{A}_i) = 0$ for any quantum adversary $\mathcal{A}_i$.

For the case the internal function is a random permutation $f^\pi$, it is shown in [14, Section 3.1] that the distinguishing advantage between a random permutation and a random function is $\mathrm{Adv}^{QI}_{(Y_0\times Y_1)^X}(\mathcal{F}^\pi; \mathcal{A}) \leq \varepsilon \in \mathcal{O}(q^3/|X|)$ for any $\mathcal{A}$ making $q$ quantum queries. Therefore, using Lemma 2 we also have that the parts $f_0^\pi, f_1^\pi$ have $\mathrm{Adv}^{QI}_{Y_i^X}(\mathcal{F}^\pi; \mathcal{A}_i) \leq \varepsilon \in \mathcal{O}(q^3/|X|)$ for any $\mathcal{A}_i$ making $q$ quantum queries and $i \in \{0, 1\}$.

In [12], it was shown that a random oracle $\mathcal{O} : \{0,1\}^* \to Y$ is collapsing with advantage $\mathrm{Adv}^{coll}_{\mathcal{O}}(A, B) = |\mathbb{P}[\mathsf{Game}_1 : b = 1] - \mathbb{P}[\mathsf{Game}_2 : b = 1]| \leq \delta \in O(\sqrt{q^3/|Y|})$, where the adversary $(\mathcal{A}, \mathcal{B})$ makes at most $q$ queries. From this result, the following lemma follows immediately:

**Lemma 3.** *Let $f : X \to Y_0 \times Y_1$ either be a random function or a random permutation with $|X| = |Y_0| \cdot |Y_1|$. Let $f_i : X \to Y_i$ be given by $P_i(f(x))$, where $P_i$ is the projection onto $Y_i$, for $i \in \{0, 1\}$. Then $f_i$ is collapsing for $i \in \{0, 1\}$ with advantage $O(\sqrt{q^3/|Y_i|})$ for any adversary $(\mathcal{A}, \mathcal{B})$ making at most q queries.*

*Proof.* In the following, let $(\mathcal{A}, \mathcal{B})$ be any adversary making at most $q$ queries. From the fact that random oracles are collapsing (Theorem 2) and Lemma 2, it follows straightforward that random functions $f_0^R, f_1^R$ are collapsing with advantage $\mathrm{Adv}^{coll}_{f_i^R}(\mathcal{A}, \mathcal{B}) = |\mathbb{P}[b = 1 : \mathsf{Game}_1] - \mathbb{P}[b = 1 : \mathsf{Game}_2]| \leq \delta_i^R \in O(\sqrt{q^3/|Y_i|})$ for $i \in \{0, 1\}$. The random permutations $f_0^\pi, f_1^\pi$ are indistinguishable from random functions with advantage $\varepsilon \in O(q^3/|X|)$, so we have collapsing advantage $\mathrm{Adv}^{coll}_{f^\pi}(\mathcal{A}, \mathcal{B}) \leq \delta_i^R + \varepsilon =: \delta_i^\pi \in O(\sqrt{q^3/|Y_i|})$ as well. □

**Collapsing step function.** For the proof of Theorem 1, we show that the sponge construction is collapsing by using an hybrid argument. Each step in the hybrid argument exploits the fact that the internal function used in the sponge is collapsing. However, for the flow of the proof it is easier to look at a different function, which we call $\mathrm{step}_f$. To make this more formal: for $f(x) = (\overline{f(x)}, \widehat{f(x)})$, we define $\mathrm{step}_f(x, y) := f(x) \oplus (y||\mathbf{0}) = (\overline{f(x)} \oplus y, \widehat{f(x)})$. In the proof below, we only use the second way of writing $\mathrm{step}_f(x, y)$, since this underlines the fact that we have two functions $\overline{f(x)}$ and $\widehat{f(x)}$. The next lemma shows that $\mathrm{step}_f(x, y)$ is collapsing if $\widehat{f(x)}$ is collapsing.

**Lemma 4.** *Let $X = \overline{Y} \times \widehat{Y}$, $f : X \to X$ be given by two functions $f(x) = (\overline{f(x)}, \widehat{f(x)})$, with $\overline{f(x)} : X \to \overline{Y}$, and $\widehat{f(x)} : X \to \widehat{Y}$, for $x \in X$. Define $\mathrm{step}_f : X \times \overline{Y} \to X$ as $\mathrm{step}_f(x, y) := f(x) \oplus (y||\mathbf{0}) = (\overline{f(x)} \oplus y, \widehat{f(x)})$. Then if $\widehat{f}$ is collapsing with advantage $\varepsilon$, then also $\mathrm{step}_f$ is collapsing with advantage $\varepsilon$.*

The proof is given in Appendix D.

Note that the above Lemma does not hold for $\overline{f(x)}$ instead of $\widehat{f(x)}$. To be a valid adversary, $\mathcal{M}^{\mathcal{A}}$ has to construct a register $M$ and a register $c$, where register $M$ contains messages $x$ such that $\overline{f(x)} = c$ with probability 1. However, from adversary $\mathcal{A}$ it receives messages $|x, y\rangle$ from register $M^* = (X^*, Y^*)$, such that $\overline{f(x)} \oplus y = \overline{c}$. In particular, this could mean that $|x, y\rangle$ is in a superposition of entangled states, satisfying $\overline{f(x)} = \overline{c} \oplus y$. If $|x, y\rangle$ is in a superposition, then possibly $\overline{f(x)}$ is also in a superposition of states. But $\mathcal{M}^{\mathcal{A}}$ should give a fixed outcome $c$ (with probability 1) for $\overline{f(x)}$. The only possible way is to measure register $Y^*$, but this measurement (possibly) violates any collapsing advantage of $(\mathcal{A}, \mathcal{B})$ for $\text{step}_f$. Hence, it is not possible to construct a valid adversary $(\mathcal{M}^{\mathcal{A}}, \mathcal{M}^{\mathcal{B}})$ for $\overline{f(x)}$, that communicates with $(\mathcal{A}, \mathcal{B})$.

**Collapsing sponges.** Recall that a sponge is denoted by $\text{Sponge}[f, \text{pad}, r](\mathbf{M}, \ell)$ with capacity $c$, where $f : \{0,1\}^{r+c} \to \{0,1\}^{r+c}$ is the internal permutation (or function), pad is the sponge-compliant padding rule, $r$ is the bitrate, $\mathbf{M} \in \{0,1\}^*$ is the message and $\ell$ is the fixed output length. For simplicity we take $\ell = r$ in the following Lemma. We handle cases $\ell < r$ and $\ell > r$ separately, afterwards. This simplified sponge construction will be denoted by $\mathcal{S}_f$ and is defined as follows:

**Definition 4.** *We define the internal iteration of the sponge as* $\mathcal{I}_f : (\{0,1\}^r)^* \to (\{0,1\}^{r+c})$*, with message length divisible by* $r$*,* $\mathcal{I}_f(\lambda) := \boldsymbol{0}$ *for the empty word* $\lambda$*, and* $\mathcal{I}_f(\boldsymbol{m}||m') := f(\mathcal{I}_f(\boldsymbol{m}) \oplus (m'||\boldsymbol{0}))$*, for* $m' \in \{0,1\}^r$*. The simplified sponge function is then defined by* $\mathcal{S}_f : (\{0,1\}^r)^* \to (\{0,1\}^r)$*, with* $\mathcal{S}_f(\boldsymbol{m}) = \overline{\mathcal{I}_f(\boldsymbol{m})}$*, where* $\overline{\mathcal{I}_f(\boldsymbol{m})}$ *denotes the outer part of* $\mathcal{I}_f(\boldsymbol{m})$*. We assume* $\boldsymbol{m} = \text{pad}(\boldsymbol{M})$ *is the output of a sponge-compliant padding function.*

We omit the term "simplified" below. Our notation and proof technique follows Unruh's proof for the collapsing property of the Merkle-Damgård (M-D) construction in [11, Section 4]. However, the proof for the sponge construction turns out to be more complicated. The main reason being the necessity to consider the two parts of $f$: both $\overline{f(x)}$ and $\widehat{f(x)}$ need to be collapsing. There are two reasons for this. First, the output does not allow to compute the full final state of the function: only the outer part of the final state is given as output of the sponge. The second reason is that in contrast to M-D, in a sponge, message blocks are not used as direct input to the internal function $f$. These message blocks are XORed with the outer part, which makes it more difficult to analyze. This even requires a second property of $f$, which is naturally fulfilled by random functions (or permutations):

**Definition 5.** *A family of functions* $\mathcal{F} = \{f : X_n \to \{0,1\}^n\}$ *is zero-preimage resistant if for any efficient quantum adversary* $\mathcal{A}$ *there exists a negligible* $\varepsilon(n)$ *such that*

$$\mathsf{Succ}_{\mathcal{A}}^{zpre}(\mathcal{F}) := \mathbb{P}\left[f \leftarrow_{\$} \mathcal{F}, x \leftarrow \mathcal{A}^f(1^n) : f(x) = 0\right] < \varepsilon(n)$$

In the definition, we are giving the adversary oracle access to $f$ as we are concerned with zero-preimage resistance of perfectly random functions. In a definition for efficient function families (i.e., with polynomial-size descriptions), $\mathcal{A}$ would just be given a description of $f$.

**Lemma 5.** *Given a security parameter $n \in \mathbb{N}$, $r, c \in \mathrm{poly}(n)$, the success probability of any $q$-query quantum adversary against the zero-preimage resistance of the family of random functions $\mathcal{F} = \{f : \{0,1\}^{r+c} \to \{0,1\}^c\}$ is upper bounded by*

$$\mathsf{Succ}^{zpre}_{\mathcal{A}}(\mathcal{F}) \leq \mathcal{O}(q^2/2^c)\,.$$

We omit a detailed proof of Lemma 5 as it can be obtained by following the proof for quantum preimage resistance of random functions in [9].

We need this property later in the proof, because using a $\widehat{f}$-pre-image of zero, one can easily construct a collision for the sponge: given an input $x \in \{0,1\}^r$ such that $\widehat{f(x||\mathbf{0})} = \mathbf{0}$, a collision of the sponge is given by two messages $m_1 = (x||y \oplus x')$ and $m_2 = x'$ for any random $x' \in \{0,1\}^r$ and $y = \overline{f(x)}$. Note that the above construction does not give a collision for the internal function $f$, but it does give a collision for $\mathcal{S}_f$.

**Lemma 6.** *Let $\mathcal{P} \subset (\{0,1\}^r)^*$ be the set of padded messages for some sponge-compliant padding with padded messages $\boldsymbol{m} \in \mathcal{P}$ such that $T \geq |\boldsymbol{m}| \geq 2$ for some upper bound $T$. If the internal function of the sponge $f : \{0,1\}^{r+c} \to \{0,1\}^{r+c}$ is $f(x) = (\overline{f(x)}, \widehat{f(x)})$ where both outer part $\overline{f(x)} : \{0,1\}^{r+c} \to \{0,1\}^r$ and inner part $\widehat{f(x)} : \{0,1\}^{r+c} \to \{0,1\}^c$ are collapsing with advantages $\overline{\varepsilon}, \widehat{\varepsilon}$ respectively, and furthermore $\widehat{f}$ is zero-pre-image resistant with advantage $\varepsilon_z$. Then $\mathcal{S}_f$ is collapsing on $\mathcal{P}$ with advantage* $\max_{p\in[0,1]}(\overline{\varepsilon} + (T-1)(p2\sqrt{\varepsilon_z} + (1-p)\widehat{\varepsilon}))$ .

*Proof sketch*: We have to show that if an adversary $(\mathcal{A}, \mathcal{B})$ outputs classical $c = \mathcal{S}_f(\mathbf{m})$, we can measure register $M$ without the adversary noticing. We show this developing hybrids that successively measure more and more of the message register $M$. Afterwards we upper bound the advantage of $(\mathcal{A}, \mathcal{B})$ in the collapsing game by upper bounding the success probability of $(\mathcal{A}, \mathcal{B})$ in distinguishing any two consecutive hybrids.

The output of the sponge is simply the outer part of $\mathcal{I}_f(\mathbf{m})$. Hence, we can upper bound the advantage of detecting the measurement of the last state of the sponge using the collapsing property of $\overline{f}$. For all the remaining hybrids, we can upper bound the distinguishing advantage using the collapsing property of $\mathrm{step}_f$ and the zero-preimage resistance of $\widehat{f}$. The intuition is that a successful distinguisher either uses a superposition of messages where the $i$-th blocks of some of the messages are different (which then would allow to break the collapsing property of $\mathrm{step}_f$) or of some messages of length $i$ and some other messages that are longer (which allows to extract a zero-preimage).

The full proof is given in Appendix A.

In Appendix C we give a step-by-step example with $|\mathbf{m}| = 3$, with Figures to explain the meanings of $\mathrm{partial}_i(\mathbf{m})$ and $\mathrm{input}_i(\mathbf{m})$. We hope this will help the reader to understand the formal proof given in Appendix A.

### 3.1 The collision resistance of SHA3

In the following we use Lemma 6 to derive our main results. But first we still have to show that $\mathrm{Sponge}[f, \mathrm{pad}, r](\mathbf{M}, \ell)$ is also collapsing for $r \neq \ell$.

**Corollary 1.** *Let* $\mathrm{Sponge}[f, \mathrm{pad}, r](\boldsymbol{M}, \ell)$ *be a sponge construction with capacity $c$. Then, the collapsing advantage of any valid, efficient quantum adversary $(\mathcal{A}, \mathcal{B})$ against* Sponge*, making no more than $q$ queries, is upper bounded by* $\mathrm{Adv}^{coll}_{\mathrm{Sponge}}(\mathcal{A}, \mathcal{B}) \in \mathcal{O}(\sqrt{q^3 \cdot \max(2^{-\ell}, 2^{-r}, 2^{-c})})$.

*Proof.* In the following, let $(\mathcal{A}, \mathcal{B})$ by any adversary making at most $q$ queries. We have handled the case where $\ell = r$ in Lemma 6. By plugging values $\overline{\epsilon} \in \mathcal{O}(\sqrt{q^3/2^r})$ and $\widehat{\epsilon} \in \mathcal{O}(\sqrt{q^3/2^c})$ for the collapsing of $\overline{f}, \widehat{f}$ respectively (using Lemma 3), and $\epsilon_z \in \mathcal{O}(q^2/2^c)$ (using Lemma 5) into the bound of Lemma 6, we derive at a collapsing advantage of $\mathcal{O}(\sqrt{q^3/2^r} + \sqrt{q^3/2^c} + \sqrt{q^2/2^c}) = \mathcal{O}(\sqrt{q^3/2^r} + \sqrt{q^3/2^c})$ for a sponge construction $\mathrm{Sponge}[f, \mathrm{pad}, r](\mathbf{M}, r)$ with capacity $c$.

We denote $Z = \mathrm{Sponge}[f, \mathrm{pad}, r](\mathbf{M}, \ell)$ to be the output of the sponge construction with $\ell > r$. Since the $\ell > r$ output bits are truncated after the squeezing phase, we have that the first $r$ output bits of $Z$ are equal to $\mathcal{S}_f(\mathbf{m})$ where $\mathbf{m} = \mathrm{pad}(\mathbf{M})$ (see Figure 1). Thus, we can simply ignore the remaining $\ell - r$ output bits and use the same arguments as in Lemma 6 to obtain the bound $\mathcal{O}(\sqrt{q^3/2^r} + \sqrt{q^3/2^c})$.

When $\ell < r$, we only have a difference with the proof of Lemma 6 in the first step of the hybrid argument. Defining $\overline{f_\ell} : \{0,1\}^{r+c} \to \{0,1\}^\ell$ by $\overline{f_\ell}(x) = P_\ell(\overline{f(x)})$, where $P_\ell$ is the projection onto the first $\ell$ bits, we can apply (a slightly modified version of) Lemma 3 to get a collapsing advantage of $\overline{\varepsilon_\ell} := \mathrm{Adv}^{coll}_{\overline{f_\ell}}(\mathcal{A}, \mathcal{B}) = \mathcal{O}(\sqrt{q^3/2^\ell})$ for $\overline{f_\ell}$. The remaining part of the proof of Lemma 6 remains the same, but in the final result changing $\overline{\varepsilon}$ to $\overline{\varepsilon_\ell}$. This means the collapsing advantage in this case will be upper-bounded by $\mathrm{Adv}^{coll}_{\mathrm{Sponge}}(\mathcal{A}, \mathcal{B}) \in \mathcal{O}(\sqrt{q^3/2^\ell} + \sqrt{q^3/2^c})$.

Lastly, note that in Lemma 4 we required $|\mathbf{m}| \geq 2$, but we drop this restriction in here. The collapsing case for $|\mathbf{m}| = 1$ is trivial, as it follows from the collapsing of $\overline{f}$ and $\mathcal{S}_f(\mathbf{m}) = \overline{f}(\mathbf{m}||\mathbf{0})$ where $\mathcal{S}_f$ is defined as in Lemma 6.

Depending on the sizes of $\ell, r$ and $c$, we get the result in the corollary. □

We are now ready to prove Theorem 1:

**Theorem 1.** *Let Sponge$[f, \mathrm{pad}, r](\boldsymbol{M}, \ell)$ be a sponge construction with capacity $c$. Any quantum collision finder $\mathcal{A}$ making at most $q$ queries, has a success probability of at most $\mathcal{O}(\sqrt{q^3 \cdot \max(2^{-\ell}, 2^{-r}, 2^{-c})})$.*

*Proof.* The proof follows immediately from Corollary 1 and the tight reduction of collision resistance to collapsing(Lemma 1).

Given Theorem 1, we immediately obtain the following corollary on quantum second-preimage resistance.

**Corollary 2.** *Let* $\mathrm{Sponge}[f, \mathrm{pad}, r](\boldsymbol{M}, \ell)$ *be a sponge construction with capacity $c$. Any quantum second-preimage finder $\mathcal{A}$ making at most $q$ queries, has a success probability of at most $\mathcal{O}(\sqrt{q^3 \cdot \max(2^{-\ell}, 2^{-r}, 2^{-c})})$.*

The corollary follows from Theorem 1 as any second-preimage finder $\mathcal{A}$ can be used as collision finder. The reduction just runs $\mathcal{A}$ on a random domain element $x$ and returns $x$ together with $\mathcal{A}$'s output.

With slightly more effort, we also obtain the following corollary on quantum preimage resistance.

**Corollary 3.** *Let* $\mathrm{Sponge}[f, \mathrm{pad}, r](\boldsymbol{M}, \ell)$ *be a sponge construction with capacity* $c$*. Any quantum preimage finder* $\mathcal{A}$ *making at most* $q$ *queries, has a success probability of at most* $\mathcal{O}(\sqrt{q^3 \cdot \max(2^{-\ell}, 2^{-r}, 2^{-c})})$.

The corollary follows from Theorem 1 as any preimage finder $\mathcal{A}$ can be used as collision finder. The reduction takes a random domain element $x$, runs $\mathcal{A}$ on $\mathcal{S}_f(x)$ and returns $x$ together with $\mathcal{A}$'s output. A sponge function compresses inputs of length up to $Tr$ bits to an intermediate state of $r + c$ bits. For all practical parameters, $Tr \gg (r + c)$ and a sponge is classically indistinguishable from a random function. Hence, every image has an exponential number of preimages. Therefore, the probability that $\mathcal{A}$ returns $x$ is negligible.

## 4 Quantum collision attack

In this section we present our quantum collision attack against random sponges, i.e. sponges where the internal function is a truly random function or permutation. Our attack makes heavy use of the BHT algorithm. There are two possible approaches to find collisions in a sponge. One is a generic search that treats the Sponge as a black box. The other exploits the internal structure of sponges. In the former case one can just apply the plain BHT algorithm, which will succeed with probability $\mathcal{O}(\frac{q^3}{2^\ell})$, making $q$ queries, as stated in Theorem 3, where $\ell$ is the output length of the Sponge.

When taking the internal structure into account, it was already observed in [3] that an inner-collision can be transformed into a state collision. This, in turn, yields a full collision for Sponge s of any length. Hence, the second way of attacking Sponge s is via an inner-collision attack. Asymptotically, an inner-collision finder succeeds with probability $\mathcal{O}(\frac{q^3}{2^c})$, as there are $2^c$ possible inner-states. A challenge is to bypass the fact that the adversary has no access to the inner-state. In Appendix B we present a subroutine DETECT that allows to check if two messages lead to an inner-collision. In the same section, we present an algorithm that achieves the success probability of $\mathcal{O}(\frac{q^3}{2^c})$ without knowledge of the value of the inner-state.

**Theorem 4.** *There exists a quantum collision finder against random Sponges with output length* $\ell$ *that, making* $q$ *queries to the Sponge, succeeds with probability* $\mathcal{O}(q^3 \cdot \max(2^{-\ell}, 2^{-c}))$.

The Theorem follows immediately from the existence of the two different attack strategies, choosing the more efficient strategy depending on the output length.

## References

1. Matthew Amy, Olivia Di Matteo, Vlad Gheorghiu, Michele Mosca, Alex Parent, and John Schanck. Estimating the cost of generic quantum pre-image attacks on sha-2 and sha-3. Cryptology ePrint Archive, Report 2016/992, 2016. `http://eprint.iacr.org/2016/992`.
2. Charles H Bennett, Ethan Bernstein, Gilles Brassard, and Umesh Vazirani. Strengths and weaknesses of quantum computing. *SIAM journal on Computing*, 26(5):1510–1523, 1997.
3. Guido Bertoni, Joan Daemen, Michaël Peeters, and Gilles Van Assche. Sponge functions. ECRYPT Hash Workshop, 2007.
4. Gilles Brassard, Peter Hoyer, and Alain Tapp. Quantum algorithm for the collision problem. *arXiv preprint quant-ph/9705002*, 1997.
5. Anthony Chefles. Quantum state discrimination. *Contemporary Physics*, 41(6):401–424, 2000.
6. Scott Contini, Arjen K. Lenstra, and Ron Steinfeld. *VSH, an Efficient and Provable Collision-Resistant Hash Function*, pages 165–182. Springer Berlin Heidelberg, Berlin, Heidelberg, 2006.
7. Lov K Grover. A fast quantum mechanical algorithm for database search. In *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, pages 212–219. ACM, 1996.
8. Carl W Helstrom. Quantum detection and estimation theory. *Journal of Statistical Physics*, 1(2):231–252, 1969.
9. Andreas Hülsing, Joost Rijneveld, and Fang Song. *Mitigating Multi-target Attacks in Hash-Based Signatures*, pages 387–416. Springer Berlin Heidelberg, Berlin, Heidelberg, 2016.
10. Frédéric Magniez, Ashwin Nayak, Jérémie Roland, and Miklos Santha. Search via quantum walk. *SIAM Journal on Computing*, 40(1):142–164, 2011.
11. Dominique Unruh. Collapse-binding quantum commitments without random oracles. In Jung Hee Cheon and Tsuyoshi Takagi, editors, *Advances in Cryptology - ASIACRYPT 2016 - 22nd International Conference on the Theory and Application of Cryptology and Information Security, Hanoi, Vietnam, December 4-8, 2016, Proceedings, Part II*, volume 10032 of *Lecture Notes in Computer Science*, pages 166–195, 2016.
12. Dominique Unruh. Computationally binding quantum commitments. In Marc Fischlin and Jean-Sébastien Coron, editors, *Advances in Cryptology - EUROCRYPT 2016 - 35th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Vienna, Austria, May 8-12, 2016, Proceedings, Part II*, volume 9666 of *Lecture Notes in Computer Science*, pages 497–527. Springer, 2016.
13. Mark Zhandry. How to construct quantum random functions. In *53rd Annual IEEE Symposium on Foundations of Computer Science, FOCS 2012, New Brunswick, NJ, USA, October 20-23, 2012*, pages 679–687. IEEE Computer Society, 2012.
14. Mark Zhandry. A note on the quantum collision and set equality problems. *Quantum Information and Computation*, 15(7&8), 2015.
15. Mark Zhandry. Secure identity-based encryption in the quantum random oracle model. *International Journal of Quantum Information*, 13(4), 2015.

## A Proof of Lemma 6

Here we will formally prove Lemma 6, using a hybrid argument.

*Proof.* Assume a quantum-polynomial-time adversary $(\mathcal{A}, \mathcal{B})$ that is valid on $\mathcal{P}$ for the sponge function $\mathcal{S}_f$.Let $\mathsf{Game}_1$ and $\mathsf{Game}_2$ be the collapsing games from Definition 3 for adversary $(\mathcal{A}, \mathcal{B})$. Let

$$\varepsilon := \mathrm{Adv}^{coll}_{\mathcal{S}_f}(\mathcal{A}, \mathcal{B}) = \big|\mathbb{P}[b = 1 : \mathsf{Game}_1] - \mathbb{P}[b = 1 : \mathsf{Game}_2]\big|.$$

In the following we upper bound $\varepsilon$.

As the padded message space $\mathcal{P}$ is the output of a sponge-compliant padding, it cannot contain the empty word. For a multi-block message $\mathbf{m} \in (\{0,1\}^r)^*$, let from now on $|\mathbf{m}|$ denote the number of $r$-bit blocks in $\mathbf{m}$ (so the bit-length of $\mathbf{m}$ is $r|\mathbf{m}|$), overloading notation. Let $\mathbf{m}_i$ be the $i$-th block of $\mathbf{m}$. Let $\mathbf{m}_{-i}$ denote the $i$-th block from the end (so $\mathbf{m}_{-1}$ is the last message block). Let $\mathbf{m}_{\geq -i}$ denote all the blocks in $\mathbf{m}$ starting from $\mathbf{m}_{-i}$ (so $\mathbf{m}_{\geq -i}$ consists of the last $i$ blocks of $\mathbf{m}$). Let $\mathbf{m}_{<-i}$ denote the blocks before $\mathbf{m}_{-i}$ (so $\mathbf{m} = \mathbf{m}_{<-i}||\mathbf{m}_{\geq -i}$ for $i \leq |\mathbf{m}|$). See Figure 2 for an example, showing the meaning of these terms.

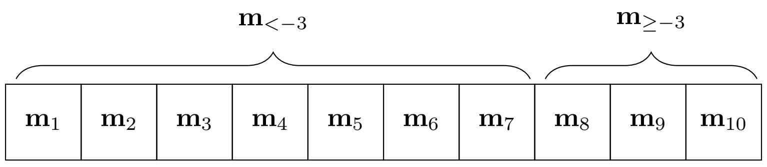


Fig. 2: Example of 10-block message $\mathbf{m}$, showing the meaning of $\mathbf{m}_{<-3}$ and $\mathbf{m}_{\geq -3}$

Let $T$ be a (polynomial) upper bound on the number of blocks for $|\mathbf{m}|$ (so $|\mathbf{m}| \leq T$). For a message $\mathbf{m}$ and $-1 \leq i \leq T$, we define:

$$\mathrm{partial}_i(\mathbf{m}) := \begin{cases} (\bot, \bot, \mathbf{m}) & \text{(if } |\mathbf{m}| \leq i) \\ (\mathcal{I}_f(\mathbf{m}_{<-(i+1)}) \oplus (\mathbf{m}_{-(i+1)}||0^c), \mathbf{m}_{-i}, \mathbf{m}_{>-i}) & \text{(if } 0 < i < |\mathbf{m}|) \\ (\mathcal{I}_f(\mathbf{m}_{<-1}) \oplus (\mathbf{m}_{-1}||0^c), \bot, \bot) & \text{(if } i = 0) \\ (\mathcal{S}_f(\mathbf{m}), \bot, \bot) & \text{(if } i = -1) \end{cases}$$

We use $\mathrm{partial}_i$ to define partial measurements of the message register output by $\mathcal{A}$. The intuition behind $\mathrm{partial}_i(\mathbf{m})$ is the following. The last two elements show the message blocks that are not yet processed by the sponge (none in the case of $\mathrm{partial}_{-1}(\mathbf{m})$ and $\mathrm{partial}_0(\mathbf{m})$ and all in the case of $\mathrm{partial}_{|\mathbf{m}|}(\mathbf{m})$). If the second element is non-empty, this element would be the next block to be processed by the sponge. The first element in $\mathrm{partial}_i(\mathbf{m})$ shows the (intermediate) state of the sponge, which will be input to internal function $f$ for the next iteration.

Note that $\text{partial}_i(\mathbf{m})$ always contains enough information to compute $\mathcal{I}_f(\mathbf{m})$ and therefore also $\mathcal{S}_f(\mathbf{m})$.

We need one more function $\text{input}_i(\mathbf{m})$, which is basically the necessary information to compute the first element in $\text{partial}_{i-1}$ for $0 \leq i \leq |\mathbf{m}|$:

$$\text{input}_i(\mathbf{m}) := \begin{cases} (\perp, \perp) & (\text{if } |\mathbf{m}| \leq i) \\ (\mathcal{I}_f(\mathbf{m}_{<-(i+1)}) \oplus (\mathbf{m}_{-(i+1)}||0^c), \mathbf{m}_{-i}) & (\text{if } 0 < i < |\mathbf{m}|) \\ (\mathcal{I}_f(\mathbf{m}_{<-1}) \oplus (\mathbf{m}_{-1}||0^c), \perp) & (\text{if } i = 0) \end{cases}$$

Note that for $0 \leq i < |\mathbf{m}|$, $\text{input}_i(\mathbf{m})$ is equal to the first two elements of $\text{partial}_i(\mathbf{m})$. The idea is that $\text{input}_i(\mathbf{m})$ will be used as input to collapsing functions, which will then allow us to (basically) map the elements in $\text{partial}_i(\mathbf{m})$ to the elements in $\text{partial}_{i+1}(\mathbf{m})$. We will now make this formal by deriving the following facts for $\text{input}_i$ and $\text{partial}_i$ from their definition:

**Fact 1** $\text{input}_{|\boldsymbol{m}|-1}(\boldsymbol{m}) = (\boldsymbol{m}_1||0^c, \boldsymbol{m}_2)$

From $\mathcal{I}_f(\mathbf{m}_{<-|\mathbf{m}|}) = \mathcal{I}_f(\lambda) = 0^c$, the fact follows straightforward. What this Fact will mean to our proof, is that in the $(|\mathbf{m}|-1)$'th hybrid argument (defined below), we will measure the last two message blocks at the same time.

**Fact 2** *If* $\text{partial}_{-1}(\boldsymbol{m}) = (h'_{-1}, \perp, \perp)$ *and* $\text{partial}_0(\boldsymbol{m}) = (h'_0, \perp, \perp)$ *then* $\overline{f}(\text{input}_0(\boldsymbol{m})) = \overline{f}(h'_0) = \mathcal{S}_f(\boldsymbol{m}) = h'_{-1}$

Note that we slightly abused notation here, as we ignored the second element $\perp$ of $\text{input}_0(\mathbf{m})$, i.e., $\overline{f}$ (and below also $f$) only acts on the first element of $\text{input}_0(\mathbf{m})$. It is easy to see that $\mathbf{m}$ must be a message such that the output of the sponge is $\mathcal{S}_f(\mathbf{m}) = h'_{-1}$. Also, $\text{input}_0(\mathbf{m}) \in \{0,1\}^{r+c}$ must be such that $f(\text{input}_0(\mathbf{m})) = \mathcal{I}_f(\mathbf{m})$. But this means that if we would apply $\overline{f}$ to $\text{input}_0(\mathbf{m})$, we get the sponge outcome: $\overline{f}(\text{input}_0(\mathbf{m})) = h'_{-1} = \mathcal{S}_f(\mathbf{m})$. Lastly, as also mentioned above, we defined $\text{input}_0(\mathbf{m})$ in such a way that $\text{input}_0(\mathbf{m}) = h'_0$.

Using $\text{step}_f(x, y) := f(x) \oplus (y||0^c) = (\overline{f(x)} \oplus y, \widehat{f(x)})$ as defined in Lemma 4, we now have the following facts for $\text{partial}_i$.

**Fact 3** *For* $0 \leq i < |\boldsymbol{m}| - 1$, *let* $\text{partial}_i(\boldsymbol{m}) = (h'_i, s_i, s'_i)$. *Then* $\text{step}_f(\text{input}_{i+1}(\boldsymbol{m})) = \text{step}_f(h_{i+1}, s_{i+1}) = h'_i$

Since $1 \leq i + 1$, we have

$$\begin{aligned} \text{step}_f(\text{input}_{i+1}(\mathbf{m})) &= \text{step}_f(\mathcal{I}_f(\mathbf{m}_{<-(i+2)}) \oplus (\mathbf{m}_{-(i+2)}||0^c), \mathbf{m}_{-(i+1)}) \\ &= f(\mathcal{I}_f(\mathbf{m}_{<-(i+2)}) \oplus (\mathbf{m}_{-(i+2)}||0^c)) \oplus (\mathbf{m}_{-(i+1)}||0^c) \\ &= \mathcal{I}_f(\mathbf{m}_{<-(i+1)}) \oplus (\mathbf{m}_{-(i+1)}||0^c) = h'_i \end{aligned}$$

**Fact 4** *From* $(\text{partial}_i(\boldsymbol{m}), \text{input}_{i+1}(\boldsymbol{m}))$, *one can compute* $\text{partial}_{i+1}(\boldsymbol{m})$ *and vice versa.*

Note that the last element of $\mathrm{partial}_i(\mathbf{m})$ is also contained in the last element of $\mathrm{partial}_{i+1}(\mathbf{m})$. The missing elements are contained in $\mathrm{input}_{i+1}(\mathbf{m})$. So basically, $\mathrm{partial}_i(\mathbf{m})$ interpolates between knowledge of only $\mathcal{S}_f(\mathbf{m})$ (case $i=-1$), and full knowledge of $\mathbf{m}$ (case $i=T-1$ ). We will make this formal by defining the following hybrid games, for $i=-1,\ldots,T-1$:

$$\begin{aligned} \mathsf{Hyb}_i :&(S,M,c) \leftarrow \mathcal{A}(1^n) && (1)\\ &(h',s,s') \leftarrow \mathcal{M}_{\mathrm{partial}_i}(M) && (2)\\ &b \leftarrow \mathcal{B}(S,M) && (3)\end{aligned}$$

Here, $\mathcal{M}_{\mathrm{partial}_i}$ is $M_f$ with $f(\mathbf{m}) = \mathrm{partial}_i(\mathbf{m})$, as defined in Section 2. By construction of function $\mathrm{partial}_i$, we have

$$\mathbb{P}[b=1 : \mathsf{Hyb}_{-1}] = \mathbb{P}[b=1 : \mathsf{Game}_2]$$

since in $\mathsf{Hyb}_{-1}$, the measurement $(h',\perp,\perp) \leftarrow \mathcal{M}_{\mathrm{partial}_{-1}}(M)$ does not have any influence: in this case, $h'$ has a determined outcome, namely $h'=h$.

We also have

$$\mathbb{P}[b=1 : \mathsf{Hyb}_{T-1}] = \mathbb{P}[b=1 : \mathsf{Game}_1]$$

since $(\mathbf{m}_1||0^c, \mathbf{m}_2, \mathbf{m}_{>2}) \leftarrow \mathcal{M}_{\mathrm{partial}_{T-1}}(M)$ fully measures register $M$ in the computational basis[3], which would also be the case in $\mathsf{Game}_1$. So this means

$$\varepsilon = \left|\mathbb{P}[b=1 : \mathsf{Hyb}_{T-1}] - \mathbb{P}[b=1 : \mathsf{Hyb}_{-1}]\right| \qquad (4)$$

A standard hybrid argument shows that we can now bound $\varepsilon$ by bounding the success probability of $(\mathcal{A},\mathcal{B})$ in distinguishing any two consecutive hybrids. Towards bounding these distinguishing advantages, we now define oracle machines $(\mathcal{M}_i^{\mathcal{A}}, \mathcal{M}^{\mathcal{B}})$ for $i=-1,\ldots,T-1$ that use $(\mathcal{A},\mathcal{B})$'s distinguishing advantage to either win some collapsing game or find a preimage of zero.

Let $U_{\mathrm{input}_i}$ refer to the unitary transformation $|\mathbf{x}\rangle|\mathbf{y}\rangle \to |\mathbf{x}\rangle|\mathbf{y}\oplus \mathrm{input}_i(\mathbf{x})\rangle$. We define the oracle machines $(\mathcal{M}_i^{\mathcal{A}}, \mathcal{M}^{\mathcal{B}})$ for $i=-1,\ldots,T-1$ in Algorithm 1 and Algorithm 2.

[3] This is the measurement outcome for messages of length $T$, for shorter messages the whole measured message is already in the last register.

**Algorithm 1** Algorithm $\mathcal{M}_i^{\mathcal{A}}$ with $-1 \leq i \leq T-1$

```
Input: Security parameter $n$, index $-1 \leq i \leq T-1$.
Output: Valid quantum registers $(S, M, c)$ or zero-preimage $x$.
 1: $(S^*, M^*, h^*) \leftarrow \mathcal{A}(1^n)$
 2: $(h', s, s') \leftarrow \mathcal{M}_{\text{partial}_i}(M^*)$
 3: If $h' = \perp$, abort
 4: If $\widehat{h'} = 0^c$:
 5:     $(h'_{i+1}, s_{i+1}) \leftarrow \mathcal{M}_{\text{input}_{i+1}}(M^*)$
 6:     Output $h'_{i+1}$
 7: If $0 \leq i \leq T-1$:
 8:     Initialize $M$ with $|0^{r+c}\rangle|0^r\rangle$
 9: Else if $i = -1$:
10:     Initialize $M$ with $|0^{r+c}\rangle$
11: Apply $U_{\text{input}_{i+1}}$ to $M^*, M$.
12: Set $c := h'$
13: Let $S = S^*, M^*, h', i$
14: Return $(S, M, c)$
```

**Algorithm 2** Algorithm $\mathcal{M}^{\mathcal{B}}$

```
Input: Security parameter $n$, quantum registers $(S, M, c)$
Output: Bit $b$.
 1: Unpack $S$, giving $S^*, M^*, h', i$
 2: Apply $U_{\text{input}_{i+1}}$ to $M^*, M$
 3: Run $b \leftarrow \mathcal{B}(S^*, M^*)$
 4: Return $b$.
```

We now have the following claims for adversary $(\mathcal{M}_i^{\mathcal{A}}, \mathcal{M}^{\mathcal{B}})$. We begin with showing that the advantage of $(\mathcal{A}, \mathcal{B})$ in distinguishing $\mathsf{Hyb}_{-1}$ from $\mathsf{Hyb}_0$ is bounded by the collapsing advantage of any efficient quantum adversary against $\overline{f}$. This is done in the first four claims using the properties of $(\mathcal{M}_{-1}^{\mathcal{A}}, \mathcal{M}^{\mathcal{B}})$.

**Claim 1** *$(\mathcal{M}_{-1}^{\mathcal{A}}, \mathcal{M}^{\mathcal{B}})$ is a valid adversary for $\overline{f}$*

We show this claim: after the measurement $(h', \perp, \perp) \leftarrow \mathcal{M}_{\text{partial}_{-1}}(M^*)$, we have that $M^*$ contains a superposition of messages $|\mathbf{m}\rangle$ with $\text{partial}_{-1}(\mathbf{m}) = (h', \perp, \perp)$. So by Fact 2, $M^*$ contains a superposition of messages $|\mathbf{m}\rangle$ such that $\overline{f}(\text{input}_0(\mathbf{m})) = h' = c$. Now algorithm $\mathcal{M}_{-1}^{\mathcal{A}}$ initializes $M$ with $|0^{r+c}\rangle$ and applies $U_{\text{input}_0}$ to $M^*, M$. Thus after that, $M$ is in a superposition of messages $|m\rangle$ such that $\overline{f}(m) = c$. Concluding, $(\mathcal{M}_{-1}^{\mathcal{A}}, \mathcal{M}^{\mathcal{B}})$ is a valid adversary for $\overline{f}$.

Let $\mathsf{Game}_1^{\mathcal{M}_{-1}}$ denote $\mathsf{Game}_1$ from Definition 3, but with adversary $(\mathcal{M}_{-1}^{\mathcal{A}}, \mathcal{M}^{\mathcal{B}})$ and function $\overline{f}$, analogous for $\mathsf{Game}_2^{\mathcal{M}_{-1}}$.

**Claim 2** $\mathbb{P}[b = 1 : \mathsf{Game}_2^{\mathcal{M}_{-1}}] = \mathbb{P}[b = 1 : \mathsf{Hyb}_{-1}]$.

We show this claim: in $\mathsf{Game}_2^{\mathcal{M}_{-1}}$ no measurement occurs between $U_{\mathrm{input}_0}$ by $\mathcal{M}_{-1}^{\mathcal{A}}$ and the invocation of $U_{\mathrm{input}_0}$ by $\mathcal{M}^{\mathcal{B}}$.Thus, these two invocations cancel each other out. So only the invocations of $\mathcal{A}, \mathcal{M}_{\mathrm{partial}_{-1}}(M^*)$ and $\mathcal{B}$ remain. This is exactly $\mathsf{Hyb}_{-1}$.

**Claim 3** $\mathbb{P}[b = 1 : \mathsf{Game}_1^{\mathcal{M}_{-1}}] = \mathbb{P}[b = 1 : \mathsf{Hyb}_0]$.

In $\mathsf{Game}_1^{\mathcal{M}_{-1}}$, $M$ is initialized with $|0^{r+c}\rangle$. $U_{\mathrm{input}_0}$ is applied to $M^*, M$. $M$ is measured in the computational basis (with outcome $m$). $U_{\mathrm{input}_0}$ is applied to $M^*, M$. Then $M$ is discarded. This is equivalent to executing $m \leftarrow \mathcal{M}_{\mathrm{input}_0}(M^*)$. Thus, in $\mathsf{Game}_1^{\mathcal{M}_{-1}}$, both $(h', s, s') \leftarrow \mathcal{M}_{\mathrm{partial}_{-1}}(M^*)$ and $m \leftarrow \mathcal{M}_{\mathrm{input}_0}(M^*)$ were executed. By Fact 4, this is equivalent to executing $(h', s, s') \leftarrow \mathcal{M}_{\mathrm{partial}_0}(M^*)$. This is exactly $\mathsf{Hyb}_0$.

From Claim 2 and Claim 3 and by assumption of this Lemma, we get:

**Claim 4** $\left|\mathbb{P}[b = 1 : \mathsf{Hyb}_0] - \mathbb{P}[b = 1 : \mathsf{Hyb}_{-1}]\right| \leq \bar{\varepsilon}$.

Let $\mu_i := \left|\mathbb{P}[b = 1 : \mathsf{Hyb}_{i+1}] - \mathbb{P}[b = 1 : \mathsf{Hyb}_i]\right|$ for $i \geq 0$ be the advantage of adversary $(\mathcal{A}, \mathcal{B})$ in distinguishing games $\mathsf{Hyb}_i$ and $\mathsf{Hyb}_{i+1}$. Next, we are upper bounding this advantage for $0 \leq i < T$. This is done analyzing the success probability of $(\mathcal{M}_i^{\mathcal{A}}, \mathcal{M}^{\mathcal{B}})$ for $0 \leq i < T$. Note that whenever the case $h' = \bot$ in line 3 of $\mathcal{M}_i^{\mathcal{A}}$ occurs, the two hybrids $\mathsf{Hyb}_i$ and $\mathsf{Hyb}_{i+1}$ are perfectly indistinguishable as applying $\mathcal{M}_{\mathrm{partial}_{i+1}}(M^*)$ has no effect at all. Hence, these cases cannot contribute to $(\mathcal{A}, \mathcal{B})$'s success probability and we can abort. Therefore, the distinguishing advantage $\mu_i$ must come from cases where $h' \neq \bot$. We can split these cases into two depending on the value of $\widehat{h'}$. In the following, we denote by $\mu_i'$ the advantage of $(\mathcal{A}, \mathcal{B})$ in distinguishing $\mathsf{Hyb}_i$ and $\mathsf{Hyb}_{i+1}$ conditioned on $\widehat{h'} = 0^c$ and by $\mu_i''$ the distinguishing advantage conditioned on $\widehat{h'} \neq 0^c$. These two probabilities are related by the following claim.

**Claim 5** *There exists a $p \in [0, 1]$ such that*

$$\mu_i = p\mu_i' + (1 - p)\mu_i''.$$

As the conditioning is on a binary event such a $p$ must exist.

Now, we first develop a bound on $\mu_i''$ in the next four claims.

**Claim 6** $(\mathcal{M}_i^{\mathcal{A}}, \mathcal{M}^{\mathcal{B}})$ *is a valid adversary for function* $\mathrm{step}_f$ *for* $0 \leq i \leq T - 1$, *conditioned on* $\widehat{h'} \neq 0^c$.

After measurement $(h', s, s') \leftarrow \mathcal{M}_{\mathrm{partial}_i}(M^*)$, we have that $M^*$ contains a superposition of messages $|\mathbf{m}\rangle$ with $\mathrm{partial}_i(\mathbf{m}) = (h', s, s')$. Per assumption we got $\widehat{h'} \neq 0^c$. So by Fact 4, $M^*$ contains a superposition of messages $|\mathbf{m}\rangle$ such that $\mathrm{step}_f(\mathrm{input}_{i+1}(\mathbf{m})) = h' = c$. Now $\mathcal{M}_i^{\mathcal{A}}$ initializes $M$ with $|0^{r+c}\rangle|0^r\rangle$ and applies $\mathrm{input}_{i+1}$ to $M^*, M$. Thus after that, $M$ is in a superposition of $|x, y\rangle$ such that $\mathrm{step}_f(x, y) = c$. This means $(\mathcal{M}_i^{\mathcal{A}}, \mathcal{M}^{\mathcal{B}})$ is a valid adversary for $\mathrm{step}_f$.

For $0 \leq i \leq T - 1$, let $\mathsf{Game}_1^{\mathcal{M}_i}$ denote $\mathsf{Game}_1$ of Definition 3, but with adversary $(\mathcal{M}_i^{\mathcal{A}}, \mathcal{M}^{\mathcal{B}})$ and function $\mathrm{step}_f$. Analogous for $\mathsf{Game}_2^{\mathcal{M}_i}$.

**Claim 7** *$\mathbb{P}[b = 1 : \mathsf{Game}_2^{\mathcal{M}_i}] = \mathbb{P}[b = 1 : \mathsf{Hyb}_i]$, conditioned on $\widehat{h'} \neq 0^c$.*

In $\mathsf{Game}_2^{\mathcal{M}_i}$, no measurement occurs between the two invocations of $U_{\text{input}_{i+1}}$ by $\mathcal{M}_i^{\mathcal{A}}$ and $\mathcal{M}^{\mathcal{B}}$, so these two invocations cancel out. Thus only the invocations of $\mathcal{A}, \mathcal{M}_{\text{partial}_i}$ and $\mathcal{B}$ remain. This is exactly $\mathsf{Hyb}_i$.

**Claim 8** *$\mathbb{P}[b = 1 : \mathsf{Game}_1^{\mathcal{M}_i}] = \mathbb{P}[b = 1 : \mathsf{Hyb}_{i+1}]$, conditioned on $\widehat{h'} \neq 0^c$.*

We show this claim: note that in $\mathsf{Game}_1^{\mathcal{M}_i}$, after the measurement $\mathcal{M}_{\text{partial}_i}$, on the registers $M^*, M$ we have the following sequence of operations. As $\widehat{h'} \neq 0^c$ per assumption, $M$ is initialized with $|0^{r+c}\rangle|0^r\rangle$. $U_{\text{input}_{i+1}}$ is applied to $M^*, M$. $M$ is measured in the computational basis (with outcome $m$). $U_{\text{input}_{i+1}}$ is applied to $M^*, M$. $M$ is discarded. This is equivalent to executing $m \leftarrow \mathcal{M}_{\text{input}_{i+1}}(M^*)$.

So $\mathcal{M}_{\text{partial}_i}(M^*)$ and $\mathcal{M}_{\text{input}_{i+1}}(M^*)$ were executed. By Fact 4, this is the same as executing $\mathcal{M}_{\text{partial}_{i+1}}(M^*)$. This means $\mathsf{Game}_1^{\mathcal{M}_i}$ is equivalent to $\mathsf{Hyb}_{i+1}$, which is the claim.

From Claims 6, 7, and 8, Lemma 4 and the assumptions of this Lemma, we get:

**Claim 9** *$\mu''_i \leq \mathrm{Adv}^{coll}_{\text{step}_f}(\mathcal{M}_i^{\mathcal{A}}, \mathcal{M}^{\mathcal{B}}) \leq \widehat{\varepsilon}$.*

Before we can put things together, we still have to upper bound $\mu'_i$. This is done in the next claim.

To upper bound $\mu'_i$ we have to analyze the case where $\widehat{h'} = 0^c$. From the definition of $\text{input}_{i+1}$ we know that applying $U_{\text{input}_{i+1}}$ to $M^*$, $|0^{r+c}\rangle|0^r\rangle$ leads to a superposition of messages in the second register of the following form: Every message in the superposition is either $\perp$ (if the message only consisted of $i$ blocks) or some $m_j$ such that $\widehat{f}(m_j) = 0^c$, i.e., a preimage of zero. It remains to show that executing $\mathcal{M}_{\text{input}_{i+1}}(M^*)$ outputs one of the $m_j$ and not $\perp$ with a probability close to $\mu'_i$.

**Claim 10** *$\mu'_i \leq 2\sqrt{\varepsilon_z}$.*

We can see the success probability $\mu'_i$ of the adversary $(\mathcal{M}_i^{\mathcal{A}}, \mathcal{M}^{\mathcal{B}})$ in distinguishing $\mathsf{Game}_2$ from $\mathsf{Game}_1$ as quantum state-discrimanation problem between the two states $\rho_0$ and $\rho_1$, where $\rho_0$ is the superposition of messages $|\mathbf{m}\rangle$ in $M^*$ after measuring $\text{partial}_i$ and $\rho_1$ is the superposition in $M^*$ after measuring $\text{partial}_{i+1}$. Notice that $\rho_0 = |\mathbf{m}\rangle\langle\mathbf{m}|$ is a pure state, and $\rho_1 = \sum_{\mathbf{m}_j} |\alpha_{\mathbf{m}_j}|^2 |\mathbf{m}_j\rangle\langle\mathbf{m}_j|$ is a mixture where some additional measurement in the computational basis has been performed on $\rho_0$. The success probability $\mu'_i$ of the adversary is upper bounded by the maximum success probability to distinguish $\rho_0$ from $\rho_1$ which by the Helström bound [8] is at most $\frac{1}{2}(1 + \frac{1}{2}\|\rho_0 - \rho_1\|_1)$, where $\frac{1}{2}\|\rho_0 - \rho_1\|_1$ is the trace distance between the two states. We use the triangle inequality to derive

$$\mu'_i \leq \frac{1}{2}\|\rho_0 - \rho_1\|_1 \leq \sum_{\mathbf{m}_j} |\alpha_{\mathbf{m}_j}|^2 \frac{1}{2}\|\rho_0 - |\mathbf{m}_j\rangle\langle\mathbf{m}_j|\,\|_1 = \sum_{\mathbf{m}_j} |\alpha_{\mathbf{m}_j}|^2 \sqrt{1 - |\alpha_{\mathbf{m}_j}|^2}, \tag{5}$$

where in the last equality, we used that the trace distance of two pure states $|\mathbf{m}_1\rangle$ and $\alpha|\mathbf{m}_1\rangle + \beta|\mathbf{m}_2\rangle$ equals $\sqrt{1-|\alpha|^2}$ [5].

Let $\mathbf{m}_s$ be the message (of length $i$) for which applying $\mathrm{input}_{i+1}$ will result in $\perp$. Note that there can be at most a single such message, as the adversary has already applied $\mathrm{partial}_i$ and thereby measured the whole message $\mathbf{m}_s$. For all other messages $\mathbf{m}_j \neq \mathbf{m}_s$, our adversary will find a preimage of zero by applying $\mathrm{input}_{i+1}$, and hence the sum of their squared amplitudes $\sum_{\mathbf{m}_j\neq\mathbf{m}_s}|\alpha_{\mathbf{m}_j}|^2$ is at most $\varepsilon_z$. For the same reason, we have that $\sqrt{1-|\alpha_{\mathbf{m}_s}|^2} = \sqrt{\sum_{\mathbf{m}_j\neq\mathbf{m}_s}|\alpha_{\mathbf{m}_j}|^2} \leq \sqrt{\varepsilon_z}$. Continuing (5), we get

$$\begin{aligned}\mu_i' &\leq \sum_{\mathbf{m}_j}|\alpha_{\mathbf{m}_j}|^2\sqrt{1-|\alpha_{\mathbf{m}_j}|^2}\\ &= |\alpha_{\mathbf{m}_s}|^2\sqrt{1-|\alpha_{\mathbf{m}_s}|^2} + \sum_{\mathbf{m}_j\neq\mathbf{m}_s}|\alpha_{\mathbf{m}_j}|^2\sqrt{1-|\alpha_{\mathbf{m}_j}|^2}\\ &\leq |\alpha_{\mathbf{m}_s}|^2\sqrt{\varepsilon_z} + \sum_{\mathbf{m}_j\neq\mathbf{m}_s}|\alpha_{\mathbf{m}_j}|^2\\ &\leq \sqrt{\varepsilon_z} + \varepsilon_z \leq 2\sqrt{\varepsilon_z}\,.\end{aligned}$$

Putting Claims 9 and 10 together with Claim 5 we obtain

**Claim 11** *There exists a $p \in [0,1]$ such that*

$$\mu_i \leq p2\sqrt{\varepsilon_z} + (1-p)\widehat{\varepsilon}.$$

Thus, using Claims 4 and 11, we get:

$$\begin{aligned}\varepsilon &\overset{(4)}{=} \left|\mathbb{P}[b=1:\mathsf{Hyb}_{-1}] - \mathbb{P}[b=1:\mathsf{Hyb}_{T-1}]\right|\\ &= \left|\sum_{i=-1}^{T-2}\mathbb{P}[b=1:\mathsf{Hyb}_i] - \sum_{i=-1}^{T-2}\mathbb{P}[b=1:\mathsf{Hyb}_{i+1}]\right|\\ &= \left|\mathbb{P}[b=1:\mathsf{Hyb}_{-1}] - \mathbb{P}[b=1:\mathsf{Hyb}_0] + \sum_{i=0}^{T-2}(\mathbb{P}[b=1:\mathsf{Hyb}_i] - \mathbb{P}[b=1:\mathsf{Hyb}_{i+1}])\right|\\ &= \left|\overline{\varepsilon} + \sum_{i=0}^{T-2}(\mathbb{P}[b=1:\mathsf{Game}_2^{\mathcal{M}_i}] - \mathbb{P}[b=1:\mathsf{Game}_1^{\mathcal{M}_i}])\right|\\ &\leq |\overline{\varepsilon} + (T-1)(2p\sqrt{\varepsilon_z} + (1-p)\widehat{\varepsilon})| = \overline{\varepsilon} + (T-1)(2p\sqrt{\varepsilon_z} + (1-p)\widehat{\varepsilon})\end{aligned}$$

This is exactly the claimed bound and thereby concludes the proof. □

## B Quantum Attack

In what follows we present several observations that will help us better understand the structure of the sponge construction. Our goal is to define an algorithm that would output a collision in a sponge. First we give a proper definition of BHT algorithm.

A crucial subroutine of this algorithm is Grover's search algorithm [7] GROVER, proven optimal in [2]. We use the notation introduced by [10] to clarify the calculation of complexity of introduced algorithms. Given a state space $\Omega$ and a *marked set* $M$ we define a function $H : \Omega \to \{0, 1\}$,

$$H(\mathbf{M}) = \begin{cases} 1 & \text{if } \mathbf{M} \in M \\ 0 & \text{otherwise} \end{cases} . \tag{6}$$

The algorithm additionally needs a sampling distribution $\pi$ on $\Omega$ and a lower bound on probability that a random element is marked $\varepsilon \leq \sum_{\mathbf{M} \in M} \pi(\mathbf{M})$. For a uniform distribution $\varepsilon = \frac{|M|}{|\Omega|}$. The algorithm searches the domain of $H$ for elements that map to 1. Doing so it runs *sampling* subroutine to sample $\mathbf{M}$ from $\Omega$ and *checking* subroutine to check if $\mathbf{M} \in M$. Cost of those subroutines are $\mathsf{S}$ and $\mathsf{C}$ respectively. Number of runs necessary for the algorithm to succeed is $\frac{1}{\varepsilon}$ and so complexity is $\mathcal{O}\left(\frac{1}{\sqrt{\varepsilon}}(\mathsf{S} + \mathsf{C})\right)$, where the square root comes from the "quantumness" of the algorithm.

We will focus on the average case instead of the worst-case scenario, meaning that performance of an algorithm making $q$ queries will be measured in terms of the success probability $\mathrm{Succ}^q$ of the algorithm after $q$ queries to $H$. Nevertheless we will keep track of cost of subroutines by using $q^C := q(\mathsf{S} + \mathsf{C})$. In case of GROVER, the probability is taken over the random choice of a function $H$ that marks an element with probability $\frac{1}{N}$.

**Definition 6.** *Let $\mathcal{F} := \{H : \{0,1\}^n \to \{0,1\}\}$ be a set of Boolean functions. $D_{1/N}$ is a family of distributions on $\mathcal{F}$ such that $H \leftarrow D_{1/N}$ outputs 1 with probability $\frac{1}{N}$. Then*

$$\mathrm{Succ}^q_{1/N}(\text{GROVER}) := \mathbb{P}_{H \leftarrow D_{1/N}}[H(\boldsymbol{M}) = 1 : \boldsymbol{M} \leftarrow \text{GROVER}^H(.)]. \tag{7}$$

Grover's algorithm, searching for a marked element in a set of size $N$ has success probability given by [9,15]:

**Theorem 5.** *[9, Theorem 1]* $\mathrm{Succ}^q_{1/N}(\text{GROVER}) \leq 8\frac{(q+1)^2}{N}$ *holds for any quantum algorithm* GROVER *with $q$ queries.*

Now we are ready to describe the BHT algorithm.

**Algorithm 3** BHT$^F$

```
Require: $F : X \to Y$, $q \geq 0$.
Output: $(x_0, x_1)$
1: Create a random subset $\mathfrak{x} \subseteq X$ of cardinality $\frac{q}{3}$ and query all its elements to $F$,
   write the pairs $(x, F(x))$ in a table $L$.
2: Sort $L$ according to the second entry in each item of $L$.
3: Check if $L$ contains a collision, that is, check if there exist two entries in $L$
   $(x_0, F(x_0))$ and $(x_1, F(x_1))$ such that $x_0 \neq x_1$ and $F(x_0) = F(x_1)$. If so go to
   step 6.
4: Run GROVER$^H$ making $\frac{2q}{3}$ queries, where $H(x) = 1$ if there exists $x_0 \in \mathfrak{x}$ such that
   $(x_0, F(x)) \in L$ but $x_1 \neq x_0$. GROVER outputs some $x_1$.
5: Find $(x_0, F(x_1)) \in L$.
6: Return the collision $(x_0, x_1)$.
```

Complexity of the classical part, i.e. from Step 1 to Step 3 is $\mathcal{O}\left(\frac{q}{3}(\mathsf{S} + \mathsf{C})\right)$, cost $\mathsf{S}$ is constant, each sample is just a single query to $F$. Note however that for some cases, like in Sponge , the cost of checking $\mathsf{C}$ might not be, we will investigate that in the next section. For now though lets say that $\mathsf{C}$ is essentially constant as it corresponds to looking through a sorted database. Complexity of the quantum part comes from GROVER. It depends on the size of the created database $L$. Probability that $x$ is marked $\varepsilon = \frac{q/3}{|Y|}$, as $F$ is random each element has the same probability to yield a collision with a member of $\mathfrak{x}$. Total complexity is

$$\mathcal{O}\left( (\frac{q}{3} + \frac{1}{\sqrt{\varepsilon}})(\mathsf{S} + \mathsf{C}) \right). \tag{8}$$

Optimization over $q$ gives us complexity of order $\mathcal{O}(|Y|^{1/3})$ for $q \in \mathcal{O}(|Y|^{1/3})$. Now that we know what exact steps we want to do we will provide the reader with formal construction of the subroutine allowing us to look for an inner-collision.

## B.1 Detecting an inner collision

Formal definition of the construction is provided as Algorithm 4. $|P|_r$ signifies number of blocks of length $r$ in $P$, $P_i$ is the $i$-th block of $P$ and $\lfloor Z \rfloor_\ell$ are the first $\ell$ bits of $Z$.

**Algorithm 4** Sponge $[f, \text{pad}, r]$

```
Input: $\mathbf{M} \in \{0,1\}^*$, $\ell \geq 0$.
Output: $Z \in \{0,1\}^l$
1: $P := \mathbf{M} \| \text{pad}[r](|\mathbf{M}|)$, and $s := 0^{r+c}$.
2: For $i = 0$ to $|P|_r - 1$ do:
3:     $s = s \oplus (P_i \| 0^c)$
4:     $s = f(s)$
5: $Z := \lfloor s \rfloor_r$
6: While $|Z|_r r < \ell$ do
7:     $s = f(s)$
8:     $Z = Z \| \lfloor s \rfloor_r$
9: Return $\lfloor Z \rfloor_\ell$
```

The Algorithm 5 is the absorbing part of a sponge that outputs the whole state of $f$, but does not apply the padding rule of the sponge.

**Algorithm 5** ABSORB$[f, r]$

```
Input: $P \in \{0,1\}^*$.
Output: $s \in \{0,1\}^{r+c}$
1: $s := 0^{r+c}$.
2: For $i = 0$ to $|P|_r - 1$ do:
3:     $s = s \oplus (P_i \| 0^c)$
4:     $s = f(s)$
5: Return $s$.
```

With above definitions at hand we will focus on the problem of detecting inner-collisions without having access to the value of $\hat{s}$. A pair of messages $\mathbf{M}_1 \neq \mathbf{M}_2$ are inner-colliding if

$$\widehat{\text{ABSORB}}(\mathbf{M}_1 \| \text{pad}(|\mathbf{M}_1|)) = \widehat{\text{ABSORB}}(\mathbf{M}_2 \| \text{pad}(|\mathbf{M}_2|)). \tag{9}$$

A first problem we encounter is not knowing what is the actual value of the inner state. But as shown in [3] inner collision can easily be transformed into a state collision which is visible to the user. A way to check whether or not two messages are inner-colliding without access to the inner part of the state is described in Algorithm 6. In the following we will abbreviate Sponge$[f, \text{pad}, r](\mathbf{M}, \ell)$ with

Sponge($\mathbf{M}, \frac{\ell}{r}$), assuming without loss of generality that user always asks for output of length which is a multiple of $r$.

**Algorithm 6** DETECT$^{\text{Sponge}}$

```
Input: $\mathbf{M}_1, \mathbf{M}_2 \in \{0,1\}^*$, $t > 0$
Output: $b \in \{0,1\}$
1: Query Sponge($\mathbf{M}_1, 1$) and Sponge($\mathbf{M}_2, 1$).
2: Calculate $a_1, a_2 \in \{0,1\}^r$, such that
       $\mathrm{Sponge}(\mathbf{M}_1, 1) \oplus a_1 = \mathrm{Sponge}(\mathbf{M}_2, 1) \oplus a_2.$   (10)
3: Query
       $Z_1 := \mathrm{Sponge}(\mathbf{M}_1 \| \mathrm{pad}[r](|\mathbf{M}_1|) \| a_1, t),$
       $Z_2 := \mathrm{Sponge}(\mathbf{M}_2 \| \mathrm{pad}[r](|\mathbf{M}_2|) \| a_2, t).$
4: If $Z_1 = Z_2$ set $b = 1$, else set $b = 0$.
5: Return $b$.
```

If $\mathbf{M}_1$ and $\mathbf{M}_2$ are inner-colliding then DETECT outputs 1, if not it outputs 1 with probability $2^{-tr}$ in case there is no state collision during Step 3. We will call $t$ the confidence level. The probability that a state collision occurs is $2^{-(c+r)+1}$, although this is an error in DETECT, it is not an error if we look for inner-collisions. We will include the possibility of this event at the end of the collision finding algorithm. The cost in the number of queries to the internal function $f$ is at most $2k_1 + 2k_2 + 8 + 2t$, where $k_i := \left\lceil \frac{|\mathbf{M}_i|}{r} \right\rceil$.

We will also need a slightly modified version of the above algorithm $\widetilde{\text{DETECT}}$. Input to $\widetilde{\text{DETECT}}$ is $(\mathbf{M}_1, \mathbf{M}_2, t, a_1, \mathrm{Sponge}(\mathbf{M}_1, 1), \mathrm{Sponge}(\mathbf{M}_1 \| \mathrm{pad}[r](|\mathbf{M}_1|) \| a_1, t))$ and the output is $(b, a_2, \mathrm{Sponge}(\mathbf{M}_2, 1), \mathrm{Sponge}(\mathbf{M}_2 \| \mathrm{pad}[r](|\mathbf{M}_2|) \| a_2, t))$. This modified algorithm makes only $2k_2 + 4 + t$ queries to $f$.

## B.2 Algorithm for finding inner collision

Below we give a detailed description of algorithm that outputs an inner-collision.

**Algorithm 7** INN-COLL$^{\text{Sponge}}$

```
Input: $t > 0$, $q \geq 0$, $\kappa \geq 0$.
Output: $\tilde{\mathbf{M}}_0, \tilde{\mathbf{M}}_1$
1: Create a random subset $\mathfrak{k} \subseteq \mathcal{K} := \{\mathbf{M} : \mathbf{M} \in \{0,1\}^*, |\mathbf{M}| \leq \kappa r\}$ of cardinality $q_1 := \frac{q}{3}$. Sort $\mathfrak{k}$ in lexicographical order.
2: Query Sponge to create the first entry to $L$:
     $$\text{entry}_1 := (\mathbf{M}_1, \text{Sponge}(\mathbf{M}_1, 1), 0^r, \text{Sponge}(\mathbf{M}_1 \| \text{pad}[r](|\mathbf{M}_1|) \| 0^r, t)). \quad (11)$$
3: Set $i := 1$
4: while $i < q_1$ do
5:     Run $\widetilde{\text{DETECT}}(\text{entry}_i(0), \mathbf{M}_{i+1}, t, \text{entry}_i(2), \text{entry}_i(1), \text{entry}_i(3))$ and save outputs to $L$.
6:     If $\widetilde{\text{DETECT}}$ outputs $b = 1$ set
         $$(\tilde{\mathbf{M}}_0, \tilde{\mathbf{M}}_1) := (\text{entry}_i(0) \| \text{pad}_i \| \text{entry}_i(2), \text{entry}_{i+1}(0) \| \text{pad}_{i+1} \| \text{entry}_{i+1}(2)) \quad (12)$$
       and go to Step 11. $\text{pad}_i$ is padding appropriate for $\mathbf{M}_i$.
7:     $i = i + 1$.
8: Sort database $L$ according to the fourth entry and check if there are two distinct entries for which $\text{entry}_{i_0}(3) = \text{entry}_{i_1 \neq i_0}(3)$. If there is a collision set $(\tilde{\mathbf{M}}_0, \tilde{\mathbf{M}}_1) := (\text{entry}_{i_0}(0) \| \text{pad}_{i_0} \| \text{entry}_{i_0}(2), \text{entry}_{i_1}(0) \| \text{pad}_{i_1} \| \text{entry}_{i_1}(2))$ and go to Step 11.
9: Set $q_2 := \frac{2q}{3}$ and run GROVER$^H$ making $q_2$ queries to $H : \mathcal{K} \to \{0,1\}$, defined as
     $$H(\mathbf{M}) = \begin{cases} 1 \text{ if } \mathbf{M} \notin \mathfrak{k} \text{ and DETECT}(\mathbf{M}, \mathbf{M}_0, t) = 1 \text{ for some } \mathbf{M}_0 \in \mathfrak{k}, \\ 0 \quad \text{otherwise.} \end{cases} \quad (13)$$
10: Run $\widetilde{\text{DETECT}}$ on output of GROVER$^H$ and set $(\tilde{\mathbf{M}}_0, \tilde{\mathbf{M}}_1)$ to messages prolonged as stated in Equation (12).
11: Return $(\tilde{\mathbf{M}}_0, \tilde{\mathbf{M}}_1)$ .
```

Fact that we can only check if two elements are colliding makes a difference when calculating complexity. In Equation 8 we assumed that $\mathsf{C}$ is constant, which is also the case here but there we compared the checked element with all members of $\mathfrak{x}$. In fact comparing with all $L$ would cost $\mathcal{O}(q)$ queries and a simple BHT would not give good complexity. Actual checking cost is the cost of $\widetilde{\text{DETECT}}$, which is at most $\mathsf{C} = 2\kappa + t + 4 + \mathcal{O}(1)$. It is important to notice that we check only one pair at a time but it does not pose a problem. Sampling requires only constant time $\mathsf{S} \in \mathcal{O}(1)$, generating a random message can be done efficiently and does not require queries to Sponge .

For the algorithm to work properly we need to specify the input parameters. Size of $\mathcal{K}$ is approximately $2^{(\kappa+1)r} - q$, we need to set $\kappa$ sufficiently large so that $|\mathcal{K}| \in \Omega(2^{2c/3})$, this is discussed is more detail in [14]. This means that $\kappa \geq \frac{1}{r}\log(2^{2c/3}+q)$. The parameter $q$ is not actually the cost of queries to Sponge the algorithm makes but sum of the number of runs of its subroutines. The total cost of this algorithm is $q^C := q_1^C + q_2^C = (\mathsf{C}+\mathsf{S})q_1 + (\mathsf{C}+\mathsf{S})(2q_2+1)$. Fortunately $\kappa$ and $t$ are relatively small and do not depend on the security parameter $c$. As we have chosen a sufficiently large set $\mathcal{K}$ the Grover algorithm certainly outputs a valid colliding pair. There is a possibility that the detection algorithm makes a mistake, but either this is because it finds another collision in the longer input, which we include by outputting the messages we are more confident about, or can be made arbitrarily small by increasing $t$, which increases the query complexity by a constant multiplicative factor.

## B.3 Unitaries for Grover algorithm

A detailed analysis of Grover's algorithm will justify calculations of number of queries our algorithm makes to the internal function $f$.

**Algorithm 8** GROVER$^H$

```
Require: $H : \{0,1\}^n \to \{0,1\}$, $q \geq 0$.
Output: $\mathbf{M} \in \{0,1\}^n$
1: Construct
       $\mathbf{G} := \mathbf{U}\,(2\,|0\rangle\,\langle 0| - \mathbb{1})\,\mathbf{U}^*\mathbf{O}.$   (14)
2: Prepare the initial state $|\Psi\rangle := \mathbf{U}\,|0\rangle$.
3: Apply the Grover iterate $\mathbf{G}$ $q$ times.
4: Measure the state and set $\mathbf{M}$ to the outcome of measurement.
5: Return $\mathbf{M}$.
```

Operator $\mathbf{U}$ prepares the initial state in an equal superposition of states ready to be evaluated by the oracle. This is where we want to formulate a quantum counterpart of the algorithm DETECT. In our case we want to have an equal superposition of messages that are already detected, that is a suitable query was done to Sponge, similarly to the steps of DETECT. First thing we need is a sponge unitary,

$$\mathbf{U}_{\mathrm{Sponge}^1} : |\mathbf{M}\rangle\,|z\rangle \mapsto |\mathbf{M}\rangle\,|z \oplus \mathrm{Sponge}(\mathbf{M},1)\rangle\,. \tag{15}$$

For the second part of the detection if two messages collide we define a function

$$\textsc{check}^t(\mathbf{M}, z_1, a_1, z_2) := \mathrm{Sponge}(\mathbf{M}\|\mathrm{pad}[r](|\mathbf{M}|)\|a_2, t), \tag{16}$$

where $a_2 := z_1 \oplus a_1 \oplus z_2$. Operator corresponding to it is

$$\mathbf{U}_{\textsc{check}^t} : |\mathbf{M}\rangle\,|z_1\rangle\,|a_1\rangle\,|z_2\rangle\,|z\rangle \mapsto |\mathbf{M}\rangle\,|z_1\rangle\,|a_1\rangle\,|z_2\rangle\,|z \oplus \textsc{check}^t(\mathbf{M}, z_1, a_1, z_2)\rangle\,. \tag{17}$$

For the full operator allowing us to prepare a suitable initial state we merge the above unitaries

$$\begin{aligned}
\mathbf{U}_{\text{DETECT}} :& \mathcal{H}_{ABCDE} \to \mathcal{H}_{ABCDE} && (18)\\
\mathbf{U}_{\text{DETECT}} :& = \mathbf{U}_{\text{CHECK}^t,CABDE}\mathbf{U}_{\text{Sponge}^1,CD} && (19)\\
\mathbf{U}_{\text{DETECT}} :& \left|\text{Sponge}(\mathbf{M},1)\right\rangle_A \left|a\right\rangle_B \left|\mathbf{M}^*\right\rangle_C \left|0^{(1+t)r}\right\rangle_{DE} \mapsto && (20)\\
& \left|\text{Sponge}(\mathbf{M},1)\right\rangle_A \left|a\right\rangle_B \left|\mathbf{M}^*\right\rangle_C \left|\text{Sponge}(\mathbf{M}^*,1)\right\rangle_D \left|0^{tr}\right\rangle_E \mapsto && (21)\\
& \left|\text{Sponge}(\mathbf{M},1)\right\rangle_A \left|a\right\rangle_B \left|\mathbf{M}^*\right\rangle_C \left|\text{Sponge}(\mathbf{M}^*,1)\right\rangle_D \otimes \\
& \left|\text{CHECK}^t(\mathbf{M}^*, \text{Sponge}(\mathbf{M},1), a, \text{Sponge}(\mathbf{M}^*,1))\right\rangle_E , && (22)
\end{aligned}$$

dummy entries (previously denoted as $z$) are set to 0, that is just for the convenience of the reader but in general they can differ, this also applies to the initial value of the entry in subspace $A$. The subscripts denote the Hilbert spaces on which the given operators act. The cost of applying $\mathbf{U}_{\text{DETECT}}$ to $|\mathbf{M}^*\rangle$ is at most $2\lceil \frac{|\mathbf{M}^*|+1}{r} \rceil + t + 2$.

Finally we can define a unitary that can be the input to GROVER

$$\begin{aligned}
\mathbf{U} :=& \mathbf{U}_{\text{DETECT},ABCDE}\mathbf{U}_{L,GABF}\mathbf{U}_{\tilde{\mathcal{K}},C} && (23)\\
\mathbf{U}\left|0\right\rangle :=& \mathbf{U}_{\text{DETECT}ABCDE} \sum_{\mathbf{M}^* \in \tilde{\mathcal{K}}} \sum_{i=1}^{q_1} \frac{1}{\sqrt{|\tilde{\mathcal{K}}| q_1}} \left|\text{entry}_i(0)\right\rangle_G \left|\text{entry}_i(1)\right\rangle_A \otimes \\
& \left|\text{entry}_i(2)\right\rangle_B \left|\mathbf{M}^*\right\rangle_C \left|0^{(1+t)r}\right\rangle_{DE} \left|\text{entry}_i(3)\right\rangle_F . && (24)
\end{aligned}$$

$\mathbf{U}_{L,GABF}$ denotes a unitary acting on subspaces $GABF$ by XORing the values encountered in database $L$ (of size $q_1$) in an equal superposition. $\mathbf{U}_{\tilde{\mathcal{K}},C}$ acts similarly on $C$ by preparing a superposition of states in $\tilde{\mathcal{K}}$. $\tilde{\mathcal{K}}$ being a set of messages not in the first column of $L$ and of length at most $\kappa r$. Using notation from Algorithm 7 $\tilde{\mathcal{K}} := \mathcal{K} \setminus \mathfrak{k}$.

Operator $\mathbf{O}$ is called the oracle as it shifts the phase of marked elements by $\pi$. In our case it is a simple operation,

$$\mathbf{O} : \left|z_1\right\rangle \left|z_2\right\rangle \mapsto (-1)^{\text{EQUAL?}(z_1,z_2)} \left|z_1\right\rangle \left|z_2\right\rangle , \quad (25)$$

where $\text{EQUAL?}(z_1, z_2) = 1$ if $z_1 = z_2$ and 0 otherwise. Considering the Hilbert spaces used by $\mathbf{U}$ we set $\mathbf{O} := \mathbf{O}_{EF}$.

## B.4 Algorithm for finding collisions in a random sponge

It actually could be simpler to find a collision in the sponge function without looking for an inner collision. Fortunately we can just apply the BHT algorithm without the consideration of the actual construction standing behind Sponge. Up till now we assumed that output length of Sponge is always a multiple of $r$, we no longer assume that. Sponge$(m,\ell)$ now means a sponge making $\lceil \frac{\ell}{r} \rceil$ evaluations of $f$ in the squeezing phase and outputting only the first $\ell$ bits of the total output.

**Algorithm 9** $\textsc{coll}^{\text{Sponge}}$

```
Input: $\ell, q, \kappa \geq 0$.
Output: $\tilde{\mathbf{M}}_0, \tilde{\mathbf{M}}_1$
1: Create a random subset $\mathfrak{l} \subseteq \mathcal{K} = \{\mathbf{M} : \mathbf{M} \in \{0,1\}^*, |\mathbf{M}| \leq \kappa r\}$ of cardinality $q_1 := \frac{q}{3}$. Query Sponge$(\mathbf{M},\ell)$ on elements of this set and create a database $L$ of pairs $(\mathbf{M}, \text{Sponge}(\mathbf{M},\ell))$.
2: Sort $L$ according to the second entry. If there exist two messages $\mathbf{M}_0$ and $\mathbf{M}_1 \neq \mathbf{M}_0$ such that Sponge$(\mathbf{M}_0,\ell)$ = Sponge$(\mathbf{M}_1,\ell)$ output $(\mathbf{M}_0,\mathbf{M}_1)$ and go to Step 5.
3: Set $q_2 := \frac{2q}{3}$ and run $\textsc{grover}^H$ making $q_2$ queries to $H : \mathcal{K} \to \{0,1\}$, defined as
       $$H(\mathbf{M}) = \begin{cases} 1 \text{ if } \mathbf{M} \notin \mathfrak{l} \text{ and } \text{Sponge}(\mathbf{M},\ell) = \text{Sponge}(\mathbf{M}_0,\ell) \text{ for some } \mathbf{M}_0 \in \mathfrak{l}, \\ 0 \qquad\qquad \text{otherwise.} \end{cases} \tag{26}$$
4: Set $\tilde{\mathbf{M}}_1$ to output of $\textsc{grover}$ and find $(\mathbf{M}_0, F(\mathbf{M}_1)) \in L$.
5: Return $(\tilde{\mathbf{M}}_0, \tilde{\mathbf{M}}_1)$ .
```

Similarly to $\textsc{inn-coll}$ sampling and checking subroutines have constant cost, $\mathsf{S} \in \mathcal{O}(1)$ and $\mathsf{C} = \kappa + \lceil \frac{\ell}{r} \rceil + \mathcal{O}(1)$. For that algorithm number of elements we are comparing to has size $2^l$, so analyzing $\mathcal{K}$ similarly as before we get that $\kappa \geq \frac{1}{r} \log(2^{2l/3} + q)$.

## C Step-by-step example of Lemma 6

In this section, we will show the meaning of the games described in Lemma 6 with a small example, to give the idea behind the proof. We take $|\mathbf{m}| = 3$ and use $\mathcal{M}$ to denote a measurement in the computational basis, on a specific part of the sponge. Note that this means that the output of the part is fixed after measurement. The example construction of the Sponge is drawn in Figure 3.

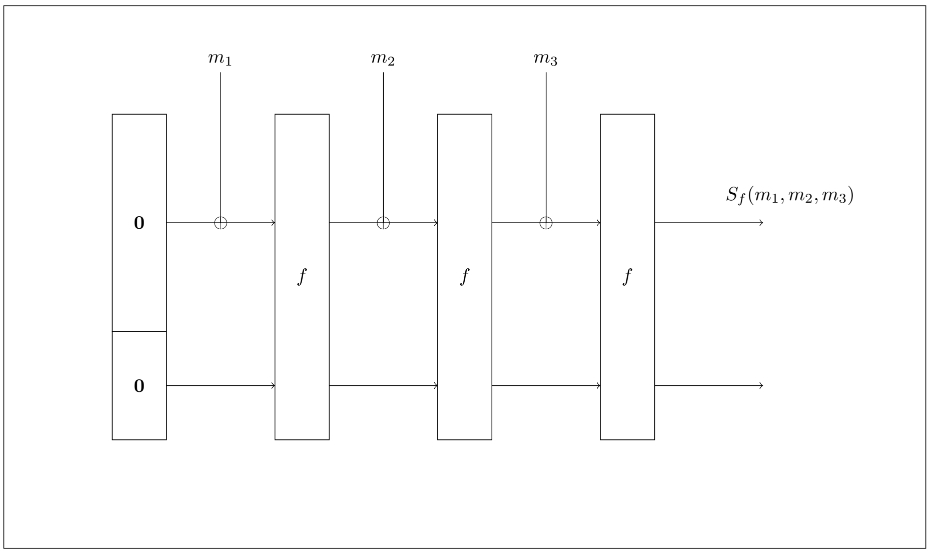
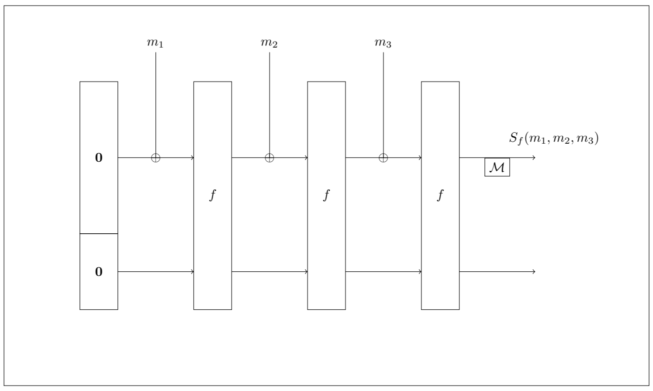
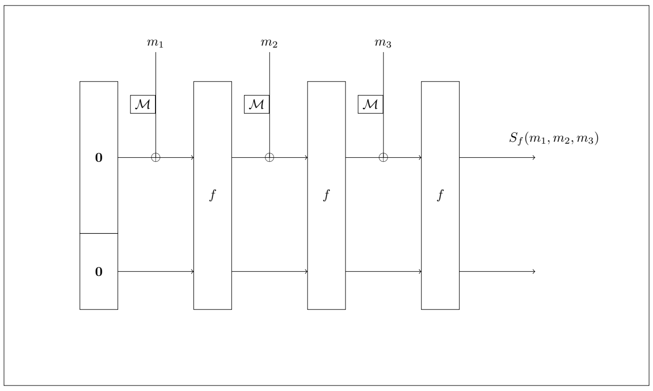


Fig. 3: A Sponge construction with $|\mathbf{m}| = 3$ and internal function $f$, that is either a random function or random permutation.

For the collapsing games (Definition 3) for the Sponge function $\mathcal{S}_f(\mathbf{m})$, one needs to distinguish the situations drawn in Figure 4.



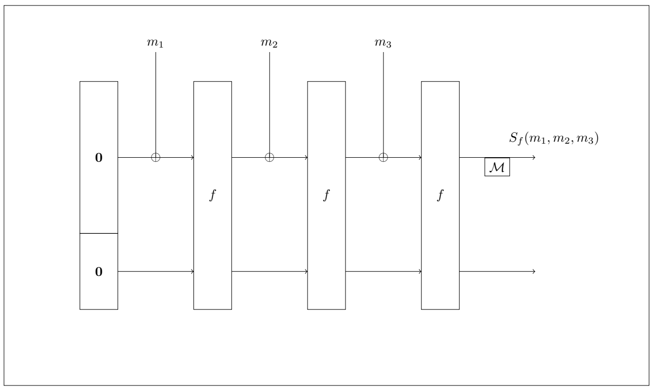
(a) A drawing of $\mathsf{Game}_2 = \mathsf{Hyb}_{-1}$



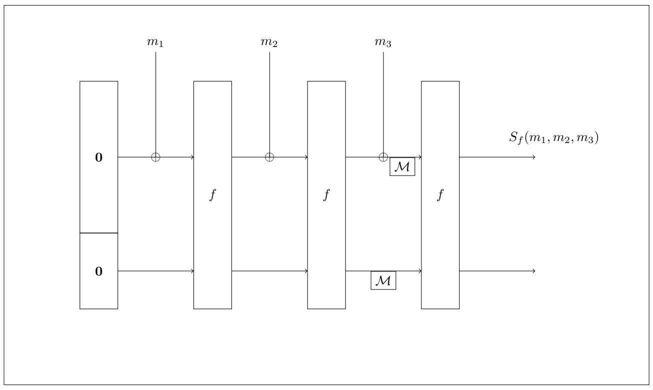
(b) A drawing of $\mathsf{Game}_1 = \mathsf{Hyb}_{T-1}$

Fig. 4: A drawing of $\mathsf{Game}_2 = \mathsf{Hyb}_{-1}$ and $\mathsf{Game}_1 = \mathsf{Hyb}_{T-1} = \mathsf{Hyb}_{|\mathbf{m}|-1}$ for sponge function $\mathcal{S}_f(\mathbf{m}_1, \mathbf{m}_2, \mathbf{m}_3)$. $\mathcal{M}$ denotes a measurement in the computational basis for the specified parts.

By definition of partial$_i$, we have the following figures for $\mathsf{Hyb}_{-1}$ (Figure 5a) and $\mathsf{Hyb}_0$ (Figure 5b):



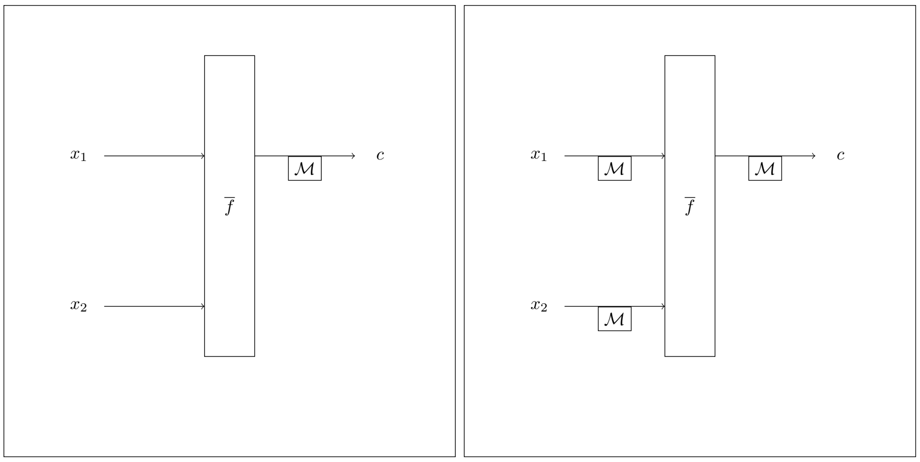
(a) A drawing of $\mathsf{Hyb}_{-1}$



(b) A drawing of $\mathsf{Hyb}_0$

Fig. 5: A drawing of $\mathsf{Hyb}_{-1}$ and $\mathsf{Hyb}_0$. $\mathcal{M}$ denotes a measurement in the computational basis for the specified parts.

Note that these two measurements occur on two sides of the function $\overline{f}$, where in $\mathsf{Hyb}_{-1}$ only the output of $\overline{f}$ is measured, and in $\mathsf{Hyb}_0$ the input to $\overline{f}$ is measured. So we have the drawing in Figure 6a for $\mathsf{Hyb}_{-1}$ (formally shown in Claim 2) and the drawing in Figure 6b for $\mathsf{Hyb}_0$ (formally shown in Claim 3). However, these parts are equal to $\mathsf{Game}_2^{\mathcal{M}_{-1}}$ and $\mathsf{Game}_1^{\mathcal{M}_{-1}}$ for function $\overline{f}$.

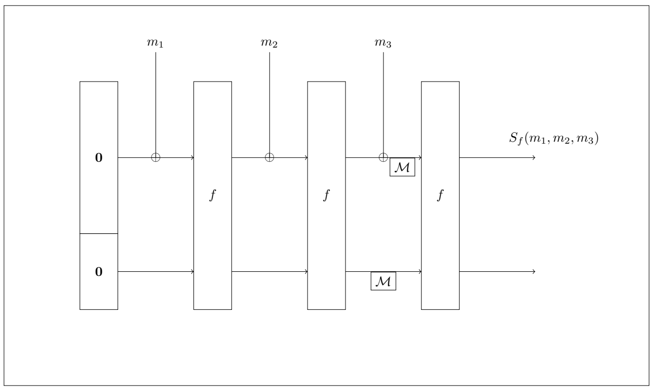


(a) A drawing of $\mathsf{Game}_2^{\mathcal{M}-1}$ (b) A drawing of $\mathsf{Game}_1^{\mathcal{M}-1}$

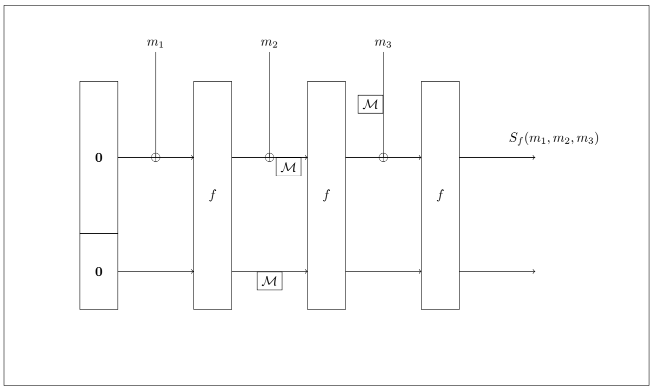
Fig. 6: A drawing of $\mathsf{Game}_2^{\mathcal{M}-1}$ and $\mathsf{Game}_1^{\mathcal{M}-1}$ for function $\overline{f}$. $\mathcal{M}$ denotes a measurement in the computational basis for the specified parts.

Since $\overline{f}$ is collapsing with advantage $\overline{\varepsilon}$, we have that the advantage of distinguishing the situation in Figure 6a from the situation in Figure 6b is $\overline{\varepsilon}$. However, these situations are simular (as in, the advantage in both cases is the same) to the situations in Figure 5. This means the advantage of distinguishing the situation in Figure 5a and the situation in Figure 5b is $\overline{\varepsilon}$. This is formally shown in Claim 4.

Now we are ready to continue with one step. Figure 7 shows the measured parts in $\mathsf{Hyb}_0$ and $\mathsf{Hyb}_1$.



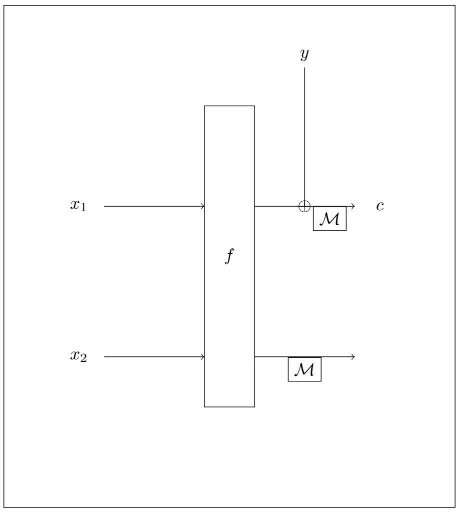
(a) A drawing of $\mathsf{Hyb}_0$



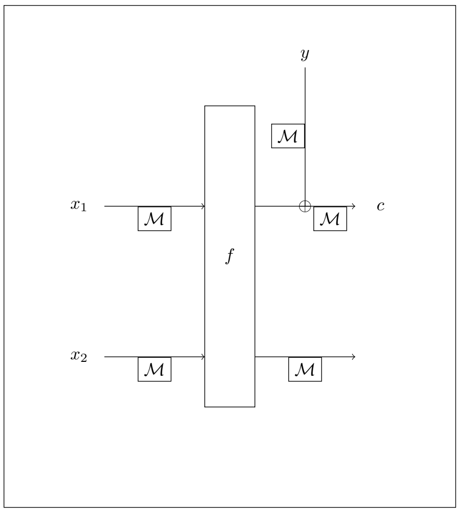
(b) A drawing of $\mathsf{Hyb}_1$

Fig. 7: A drawing of $\mathsf{Hyb}_0$ and $\mathsf{Hyb}_1$. $\mathcal{M}$ denotes a measurement in the computational basis for the specified parts.

Note that $\text{input}_1(\mathbf{m})$ are exactly those parts which will be measured in $\mathsf{Hyb}_1$ (Figure 7b). We now have a similar situation as is the case in Figure 5, but now the two games are the input/output of function $\text{step}_f(x, y)$. In particular, we have the drawing in Figure 8a for $\mathsf{Hyb}_0$ (formally shown in Claim 7) and the drawing in Figure 8b for $\mathsf{Hyb}_1$ (formally shown in Claim 8). However, these inputs are equal to $\mathsf{Game}_2^{\mathcal{M}_0}$ and $\mathsf{Game}_1^{\mathcal{M}_0}$ for function $\text{step}_f(x, y)$. In Figure 8 there is a drawing of these two games.



(a) A drawing of $\mathsf{Game}_2^{\mathcal{M}_0}$

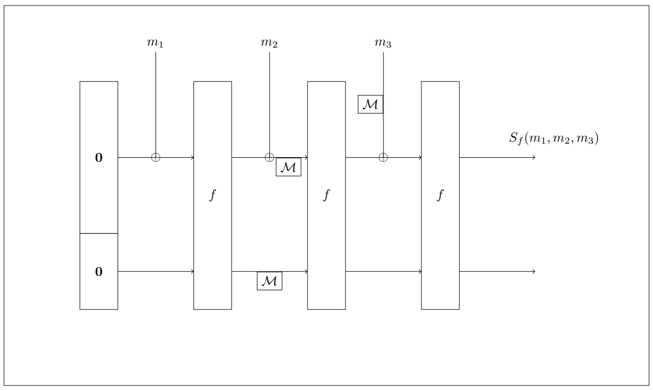


(b) A drawing of $\mathsf{Game}_1^{\mathcal{M}_0}$

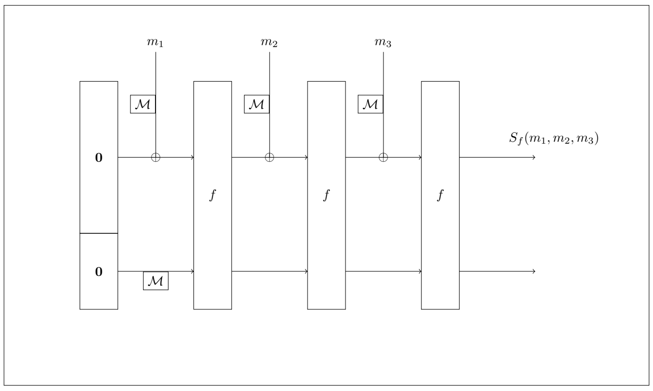
Fig. 8: A drawing of $\mathsf{Game}_2^{\mathcal{M}_0}$ and $\mathsf{Game}_1^{\mathcal{M}_0}$. $\mathcal{M}$ denotes a measurement in the computational basis for the specified parts.

Since $\mathrm{step}_f(x, y)$ is collapsing with advantage $\varepsilon^*$ (by assumption of Lemma 4), we have that the advantage of distinguishing the situation in Figure 8a from the situation in Figure 8b is $\varepsilon^*$. However, these situations are simular (as in, the advantage in both cases is the same) to the situations in Figure 7. This means the advantage of distinguishing the situation in Figure 7a and the situation in Figure 7b is $\varepsilon^*$.

For the last step, we have the same situations as in Figure 8, but by definition $\text{partial}_2(\mathbf{m})$ also contains message block $\mathbf{m}_3$.



(a) A drawing of $\mathsf{Hyb}_1$



(b) A drawing of $\mathsf{Hyb}_2$

Fig. 9: A drawing of $\mathsf{Hyb}_1$ and $\mathsf{Hyb}_2$. $\mathcal{M}$ denotes a measurement in the computational basis for the specified parts. Note that in $\mathsf{Hyb}_2$, we used Fact 1 in measuring $\mathbf{m}_1$.

Using the same arguments (in particular: $\text{step}_f(x, y)$ is collapsing with advantage $\varepsilon^*$ in Figure 8), we can conclude that distinguishing the situation in Figure 9a and the situation in Figure 9b is $\varepsilon^*$. Note that the additional measured parts in Figure 9b is equal to $\text{input}(\mathbf{m})$, that is $\text{input}_2(\mathbf{m}) = (\mathbf{m}_1 || 0, \mathbf{m}_2)$, which is exactly the input to $\text{step}_f$.

Putting all arguments together, we can conclude that distinguishing the situation in Figure 4a ($\mathsf{Game}_2$) from the situation in Figure 4b($\mathsf{Game}_1$) is $\overline{\varepsilon} + 2\varepsilon^*$. This is what is formally shown in Lemma 6.

## D Proof of Lemma 4

*Proof.* Let $(\mathcal{A}, \mathcal{B})$ be a valid quantum-polynomial-time adversary for the collapsing games for function $\text{step}_f$ with advantage $\text{Adv}^{coll}_{\text{step}_f}(\mathcal{A}, \mathcal{B}) = \mu > \varepsilon$. We now construct a quantum-polynomial-time oracle machine $(\mathcal{M}^{\mathcal{A}}, \mathcal{M}^{\mathcal{B}})$ for function $\widehat{f}$ with collapsing advantage $\text{Adv}^{coll}_{\widehat{f}}(\mathcal{M}^{\mathcal{A}}, \mathcal{M}^{\mathcal{B}}) = \mu$.

Let $\mathcal{M}^{\mathcal{A}}$ be the following quantum algorithm: it runs $(S^*, M^*, c^*) \leftarrow \mathcal{A}(1^n)$ to obtain quantum registers, such that $\mathcal{B}$ can distinguish between the collapsing games for $\text{step}_f$ with advantage $\mu > \varepsilon$. In particular, $M^*$ is a quantum register consisting of two registers $M^* = (X^*, Y^*)$ which contain superpositions of basis states $|x, y\rangle$ that fulfill $\text{step}_f(x, y) = c^*$. Note that the registers $X^*$ and $Y^*$ can be entangled. Denoting $c^* = \overline{c}||\widehat{c}$, $\mathcal{A}$ outputs $(S, M, c)$ with $S = (S^*, Y^*, c^*)$, $M = X^*$ and $c = \widehat{c}$. In other words, algorithm $\mathcal{M}^{\mathcal{A}}$ simply rearranges the registers from $\mathcal{A}$. The algorithm $\mathcal{M}^{\mathcal{B}}$ retrieves $(S, M, c)$ and simply reverses the rearrangement from $\mathcal{M}^{\mathcal{A}}$: it sets $M^*$ to be $(X^*, Y^*)$ and sends $(S^*, M^*)$ to $\mathcal{B}$. $\mathcal{M}^{\mathcal{B}}$ will output whatever $\mathcal{B}$ outputs.

We need to show two things now: $(\mathcal{M}^{\mathcal{A}}, \mathcal{M}^{\mathcal{B}})$ is a valid adversary for the collapsing games of $\widehat{f}$ and $\text{Adv}^{coll}_{\widehat{f}}(\mathcal{M}^{\mathcal{A}}, \mathcal{M}^{\mathcal{B}}) = \mu$. Validity of $(\mathcal{M}^{\mathcal{A}}, \mathcal{M}^{\mathcal{B}})$ follows from validity of $(\mathcal{A}, \mathcal{B})$, because register $M$ (which contains $X^*$) must yield messages $x$ for which $\widehat{f(x)} = \widehat{c} = c$. Otherwise, $(\mathcal{A}, \mathcal{B})$ was invalid.

For the collapsing advantage of $(\mathcal{M}^{\mathcal{A}}, \mathcal{M}^{\mathcal{B}})$, we note that in $\mathsf{Game}_1$, register $M$ will be measured and collapsed to a single $x$ such that $\widehat{f(x)} = \widehat{c}$. However, the crucial point is that also $y = \overline{f(x)} \oplus \overline{c}$ collapses, since by measuring $x$, the part $\overline{f(x)}$ cannot be in a superposition anymore and $\overline{c}$ was already classical. In $\mathsf{Game}_2$, no such measurement occurs on register $M$, so $x$ is still (possibly) in superposition. Therefore, the collapsing advantage of $(\mathcal{M}^{\mathcal{A}}, \mathcal{M}^{\mathcal{B}})$ for $\widehat{f}$ is equal to the one of $(\mathcal{A}, \mathcal{B})$ for $\text{step}_f$. □