The Sponge is Quantum Indifferentiable

Gorjan Alagic[1,2], Joseph Carolan[1], Christian Majenz[3], and Saliha Tokat[3]

[1]University of Maryland
[2]National Institute of Standards and Technology
[3]Technical University of Denmark

Abstract

The sponge is a cryptographic construction that turns a public permutation into a hash function. When instantiated with the Keccak permutation, the sponge forms the NIST SHA-3 standard. SHA-3 is a core component of most post-quantum public-key cryptography schemes slated for worldwide adoption.

While one can consider many security properties for the sponge, the ultimate one is *indifferentiability from a random oracle*, or simply *indifferentiability*. The sponge was proved indifferentiable against classical adversaries by Bertoni et al. in 2008. Despite significant efforts in the years since, little is known about sponge security against quantum adversaries, even for simple properties like preimage or collision resistance beyond a single round. This is primarily due to the lack of a satisfactory quantum analog of the lazy sampling technique for permutations.

In this work, we develop a specialized technique that overcomes this barrier in the case of the sponge. We prove that the sponge is in fact indifferentiable from a random oracle against quantum adversaries. Our result establishes that the domain extension technique behind SHA-3 is secure in the post-quantum setting. Our indifferentiability bound for the sponge is a loose $O(\mathsf{poly}(q)2^{-\min(r,c)/4})$, but we also give bounds on preimage and collision resistance that are tighter.

Contents

1 Introduction

1.1 The sponge construction

The sponge construction [Ber+07; Ber+11a] is a domain extension scheme that uses a public cryptographic permutation to construct a hash function or extendable-output function (XOF). The sponge is most prominently used in the SHA-3 standard [SND15]. The SHA-3 hash functions and XOFs, in turn, are core ingredients in all post-quantum cryptographic schemes recently standardized by NIST [Ala+24b; Dan+24; Coo+24]. Ascon, a suite of lightweight symmetric-key schemes selected by NIST for standardization is also based on the sponge [Sön+24].

To instantiate the sponge, one chooses a permutation $\varphi : \{0,1\}^n \to \{0,1\}^n$ as well as positive "rate" r and "capacity" c such that $r + c = n$. Given this data, the sponge Sp^φ is a XOF defined as follows. Parse the input $x = x_1 \| x_2 \| \cdots x_\ell$ into length-r blocks. Initialize the computation with the state $0^r \| 0^c$. XOR x_1 into the rate, and then apply φ to the entire state; repeat this for each block of the input, i.e., ℓ-many times. This completes the *absorbing phase*. Next, output the r bits contained in the rate, and then apply φ to the entire state; repeat this process until the desired number of output bits is produced. This completes the *squeezing phase*. Turning this XOF into a hash function amounts to selecting a fixed output length. For example, an r-bit output on a $2r$-bit input is simply the first r bits of $\varphi((x_2 \| 0^c) \oplus \varphi(x_1 \| 0^c))$, as depicted in Figure 1.

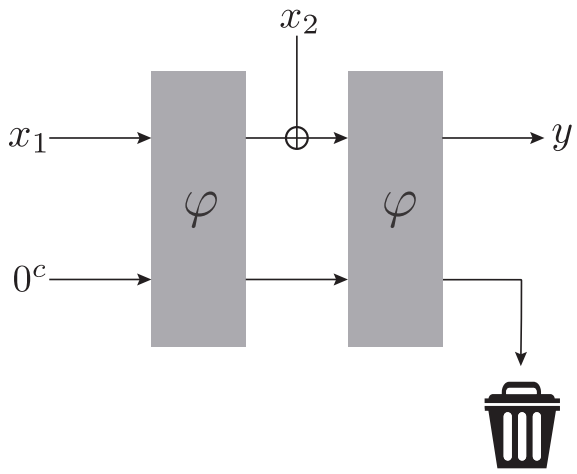

Figure 1: Sponge construction on input $x_1 \| x_2$, with output y.

Classical security. The security of the sponge construction is usually analyzed in the ideal permutation model, where the cryptographic permutation φ is modeled as a uniformly random permutation given as an oracle. As with all hash functions, the minimum security requirements include one-wayness and collision resistance. The "gold standard" for domain extension scheme security is indifferentiability from a random oracle [MRH04]. Roughly speaking, indifferentiability in this context means that the pair (construction Sp^φ, primitive φ) is query-indistinguishable from the pair (truly random function f, simulated primitive $\mathcal{S}^f$). The sponge construction where

φ is a random permutation is indifferentiable from a random oracle, up to advantage $O(q^2/2^c)$ for a classical q-query adversary [Ber+08]. This result is tight, and it implies tight collision resistance, as well as tight preimage resistance for some parameters. For the XOF setting, tight preimage resistance was proven separately [LM22].

The proofs for these classical results follow a general blueprint. In order to evaluate φ, the adversary has to query it. The simulator can examine the queries of the adversary and use its access to f to try to respond in a consistent manner. When an adversary attempts to break a certain property (e.g., find a collision, or distinguish the real and ideal worlds of the indifferentiability experiment), the list of input-output pairs of the permutation that the adversary has learned is analyzed. It is then shown that a query list allowing the adversary to achieve its goal is unlikely.

Quantum security. Rapid technological advances have made it increasingly necessary to ensure that cryptographic mechanisms remain secure against adversaries equipped with a quantum computer. In the case of SHA-3, such adversaries are free to evaluate both the Keccak permutation and SHA-3 itself in coherent superposition. It is thus appropriate to analyze the sponge construction in the quantum-accessible ideal permutation model (QIPM). In this model, the adversary can make quantum queries to both the random permutation φ and the sponge Sp^{φ}. Proving non-trivial security results in the QIPM has turned out to be challenging. One obvious obstacle is that there is no quantum analog of the generic proof blueprint outlined above. In particular, quantum queries to a random permutation cannot be recorded into a list due to the fact that quantum information cannot be copied (*no-cloning principle*). This obstruction can be circumvented, but only in the related setting of quantum-queryable *random functions*, where Zhandry's compressed oracle technique [Zha19] has enabled generalizing a multitude of proof techniques to the post-quantum setting. Indifferentiability appears particularly challenging in the quantum-query context. For domain extension schemes, it is known to require stateful simulators [RSS11], and the only plausible quantum technique for that is the compressed oracle.

As a consequence of these challenges, prior results on the quantum security of the sponge either instantiate it with a random function instead of a random permutation (as in [CHS19; Cza+18]), or limit the sponge to one round of absorption only (as in [Zha21; CP24; CPZ24; MMW24]). There have also been attempts to generalize the compressed oracle technique to the random permutation setting. The fundamental idea of *purifying* the random primitive is still sound [Ala+22; Ala+24a; MMW24]. The statistical dependence of the outputs of the random permutation, however, complicate the task of *oracle forensics*, the task of analyzing an adversary's behavior by inspecting the internal state of the oracle simulation. As a result, attempts at constructing a compressed permutation oracle that offers this feature have been unsuccessful [Cza+19; Unr21][1] or are currently limited to the analysis of simple problems [MMW24].

The state of affairs is thus still very unsatisfactory: even basic security properties like quantum preimage or collision resistance, let alone indifferentiability, are not known to hold for the sponge, and seem out of range for existing techniques. This is a concerning gap, as SHA-3 is in widespread use and cryptographically-relevant quantum computers might appear in the not-too-distant future.

[1] The difficulty of this problem has inspired very nice works that present approaches and characterizing results [Ros22; Unr23]

1.2 Our approach

In this work, we establish the quantum security properties of the sponge discussed above. Rather than constructing a fully general permutation analogue of the compressed oracle, we develop a tailored approach for our important special case.

We begin by observing that there are subgroups of the full permutation group where random elements have nice compressed oracles. For example, one round of the Feistel construction [LR88] is a (highly structured) permutation, and the set of these forms a subgroup (with composition corresponding to XORing the underlying round functions). A random element from this set is constructed from a random function, and is thus amenable to the compressed oracle technique[2]. For us it is natural to consider an unbalanced Feistel setting, split between the c bits of the capacity and the r bits of the rate. We express the sponge permutation φ as

$$\varphi = \omega_h \circ \tau_{k'} \circ \pi \circ \sigma_k \,, \tag{1}$$

where $\sigma_k(x\|y) = x\|y \oplus k(x), \tau_{k'}(x\|y) = x\|y \oplus k'(x)$ and $\omega_h(x\|y) = x \oplus h(y)\|y$, where k, k', h are random functions, and π is a random permutation. In other words, first apply a one-round Feistel cipher, then apply a random permutation π on $\{0,1\}^{r+c}$, and end with a two-round Feistel cipher. We are able to show that the three Feistel rounds combined with some global statistical properties of π are sufficient to account for the intractability of φ necessary for sponge security. Our focus here is on applying this idea to the sponge, but this technique may be useful for proving quantum security of other constructions based on ciphers and permutations, such as the Luby-Rackoff or Feistel construction itself.

The reason this decomposition is useful is that we have "offloaded" the hardness of φ onto the h, k, k' functions, which we have quantum tools to analyze. The sponge becomes insecure when an adversary can find two inputs which lead to the same state after absorbing, and it turns out that finding two such inputs will require many queries to the random functions h, k, k'. This holds even in the case where an adversary gets to see the whole truth table of π.

When directly proving query bounds, such as for finding a sponge collision, we can now relax the problem: we provide oracle access to h, k and k', and hand the query algorithm the entire truth-table description of π. We can then formulate the problem as a search problem relative to a quantum-accessible random oracle which is entirely amenable to the compressed oracle technique. We apply that technique using the query bound framework of [Chu+21].

For indifferentiability, we begin by showing that constructing φ as in Equation (1) is perfectly indifferentiable from sampling it uniformly at random. We then make a simplifying (but also perfectly indifferentiable) modification to the absorbing phase of the sponge: in each round, instead of XORing in the next message block, we simply replace the contents of the rate by that block. With these preparation steps out of the way, our high-level approach can then be seen as analogous to that of the quantum indifferentiability proof of the Merkle-Damgård construction [Zha19].

However, analyzing our construction proves more complicated. One reason is that, in Merkle-Damgård, one can assume the round function has no state collisions.

[2]Feistel itself is unfortunately still resistant to quantum security analysis, partly because there's no compressed oracle for the overall permutation.

For the sponge, it is straightforward to find such collisions using inverse permutation calls; however, it is hard to find such collisions where both preimages are "rooted" at the initial state. In this way, one has to consider global properties that depend on many rounds, whereas single-round properties were sufficient for Merkle-Damgård. Along the way, we uncover a gap in the Merkle-Damgård proof. Our proof for the sponge avoids this problem, and we expect similar ideas to apply to Merkle-Damgård, though they likely result in a looser bound.

1.3 Summary of results

Recall that, given a permutation $\varphi : \{0,1\}^{r+c} \to \{0,1\}^{r+c}$, the sponge construction using φ is denoted by Sp^{φ}.

Preimage and collision resistance. For the case of a single round of squeezing (but arbitrary rounds of absorbing), we prove the following (non-tight) bounds for preimage and collision resistance. See Section 2.2 for a technical overview.

Theorem 1.1 (Informal summary of Theorem 6.11 and Corollary 6.12)**.** *The probability that a quantum algorithm $\mathcal{A}$ making no more than q quantum queries to a random permutation $\varphi \in S_{\{0,1\}^{r+c}}$ and its inverse φ^{-1} outputs a sponge collision is upper bounded as*

$$\Pr_{\varphi \leftarrow S_{\{0,1\}^n}; m,m' \leftarrow \mathcal{A}}[m \neq m' \wedge \mathsf{Sp}^{\varphi}(m) = \mathsf{Sp}^{\varphi}(m')] \leq O(q^5 n 2^{-\min(r,c)}).$$

The same bound also holds for preimage-resistance. For constant success probability, $\tilde{\Omega}\left(2^{\min(r,c)/5}\right)$ quantum queries are thus necessary to find a collision or a preimage.

In the preimage resistance notion considered above, the adversary is given a uniformly random $y \in \{0,1\}^r$. This should be distinguished from one-wayness, where y is instead be the image of a uniformly random input.

Indifferentiability. The following formalizes our main result, i.e., that the sponge is indifferentiable against adversaries making quantum queries to φ, φ^{-1} and Sp^{φ}. Here Sp^{φ} denotes the full sponge, i.e., both the input and the output are of arbitrary length. For a given quantum query to Sp^{φ}, the length of that query is simply the length of the longest message in the superposition.

Theorem 1.2 (Informal summary of Theorem 7.22)**.** *There exists an efficient simulator $\mathcal{S}$ such that for all adversaries $\mathcal{A}$ making q quantum queries each of length at most $r \cdot \ell$,*

$$\left|\Pr[\mathcal{A}^{\varphi,\varphi^{-1},\mathsf{Sp}^{\varphi}}() = 1] - \Pr[\mathcal{A}^{\mathcal{S}^f,f}() = 1]\right| = O\left(\ell^3 \sqrt[4]{q^9 2^{-\min(r,c)}}\right).$$

In the above, φ is a uniformly random permutation while f is a uniformly random function with the same domain and range as Sp^{φ}.

1.4 Additional related work

The only previous quantum indifferentiability result with a stateful simulator is for the Merkle-Damgård construction [Zha19]. Prior to this, some barrier results for quantum indifferentiability were shown [Car+18]. Our approach to the direct query bounds for collision and preimage finding uses the query bound framework from [Chu+21] and draws inspiration from the techniques built on top of it [Don+22; Agu+23; Hül+24; Bau+25]. Some query bounds have been proven using the compressed oracle without additional generic tools [LZ19; HM23], and an alternative generic framework via oracle games presented in [CMS19] is similarly versatile (see, e.g., [RT24]).

1.5 Acknowledgements

Gorjan Alagic was supported in part by NSF award CNS-21547. Joseph Carolan acknowledges support from the U.S. Department of Energy (grant DE-SC0020264).

2 Technical summary

2.1 A View of permutation oracles

We avoid the aforementioned barriers to quantum lazy-sampling permutations by developing an alternative view of permutation oracles that enables us to apply existing techniques. We now describe this approach in somewhat more detail. The full construction with proofs is given in Section 5.

Stateful oracles inspired by Feistel. Recall that we may write a uniform random permutation φ as the product $\varphi = \pi \circ \sigma$ of a uniform random permutation π with a potentially structured permutation σ. We observe that, for certain distributions on structured permutations σ, we can use existing compressed oracle techniques. For example, suppose we define our distribution by sampling a random function k, and defining

$$\sigma(x\|y) = x\|y \oplus k(x). \tag{2}$$

Clearly, answering a query to σ can be done with one query to k. We also have $\sigma = \sigma^{-1}$, evading the problem of inverse queries. To build intuition for why this is useful, consider a toy problem introduced by Unruh [Unr21; Unr23].

Problem 1 (Double-Sided Zero Search)**.** *Given query access to a permutation φ on $\{0,1\}^{2r}$ and its inverse, find a "zero pair" $(x,y) \in \{0,1\}^r$ s.t. $\varphi(x\|0^r) = y\|0^r$.*

Lower bounds for this problem are known [CP24; MMW24]. Still, thinking about how to prove such bounds provides a simple jumping-off point for our technique. We will select independent random functions $k, k' : \{0,1\}^r \to \{0,1\}^r$, and define permutations τ and σ as

$$\sigma(x\|y) = x\|y \oplus k(x), \qquad \tau(x\|y) = x\|y \oplus k'(x). \tag{3}$$

Consider randomly selecting a π, and defining $\varphi = \tau \circ \pi \circ \sigma$. It follows that φ is random, so it suffices to prove our lower bound against an adversary querying φ and φ^{-1}. We can also strengthen the access model, so the adversary can query k and k' (which suffices to implement σ and τ), and also receives the *entire truth table* for π. It turns out that, for almost all π, such an adversary must still make $\tilde{\Omega}(2^{r/2})$ quantum queries to k and k' to find a zero pair in φ.

To see why such a bound is likely to hold, consider how lazy-sampling classical queries to k, k' might fail. When querying σ on input $x\|0^r$, a randomly selected suffix $w \in \{0,1\}^r$ is selected. This gives a candidate zero pair $y\|z = \pi(x\|w)$. In the case where y is not yet queried to k', this will be completed to a zero pair only when $k'(y) = z$, i.e., with probability 2^{-r}. In the case where y is already queried to k' with image $z' = k'(y)$, observe that for most permutations π the value of z will have nearly r bits of entropy from the entropy in w, making the event $z = z'$ have probability $\approx 2^{-r}$ (one can make this precise with a tail bound for π). To handle the case of quantum queries, one can use compressed oracles for k, k' in place of lazy sampling and obtain the appropriate quadratically weaker bound.

This simple example conveys the key observation underlying our approach. By writing a random permutation as a product involving a random permutation and well-chosen structured permutations, we can reduce (certain) problems about two-way-accessible random permutations to problems about random functions.

The case of the sponge. The example of Double-Sided Zero Search is somewhat analogous to the single round sponge with $c = r$. To analyze the many-round sponge, it turns out that we will need one more factor in our decomposition of the sponge permutation φ. Recall that $\varphi : \{0,1\}^n \to \{0,1\}^n$ where $n = r + c$, and that this determines the sponge construction Sp^φ. We will construct φ out of a random permutation $\pi : \{0,1\}^n \to \{0,1\}^n$ and permutations corresponding to functions $k, k' : \{0,1\}^r \to \{0,1\}^c$ and $h : \{0,1\}^c \to \{0,1\}^r$. We define the structured permutations by

$$\sigma_k(x\|y) = x\|y \oplus k(x), \qquad \tau_{k'}(x\|y) = x\|y \oplus k'(x), \qquad \omega_h(x\|y) = x \oplus h(y)\|y. \quad (4)$$

Note that the application of h in ω_h is "upside down" w.r.t. the applications of k, k' in $\sigma_k, \tau_{k'}$, as in the Feistel or Luby-Rackoff construction. We then write

$$\varphi := \omega_h \circ \tau_{k'} \circ \pi \circ \sigma_k. \quad (5)$$

We require a particular property from π: for any fixed $x_1, x_2 \in \{0,1\}^r$, the number of suffixes $z \in \{0,1\}^c$ such that $\pi(x_1\|z)$ begins with x_2 is not much larger than its average value of 2^{c-r} (or not larger than $O(n)$ if $r \geq c$). A standard argument shows that almost all permutations satisfy this property, described in Appendix B.

With this in place, we will henceforth discuss the sponge construction in terms of a fixed permutation π together with random functions h, k, k' that are implemented using compressed oracles (or lazy sampling in the classical case). We can then talk about query transcripts or databases for these functions, which we will refer to as $D_h, D_k, D_{k'}$. Roughly speaking, these databases contain the points that the adversary knows, and do not contain unqueried points. Central to our argument is the notion of a *tail* for a given capacity value z; this is essentially an input which reaches state z.

Definition (Tail)**.** *Let $z \in \{0,1\}^c$, and fix $D_k, D_{k'}$ and π. We say that z has a "tail" under the following recursive conditions.*

(1) $z = 0^c$ has a tail.

(2) z has a tail if it can be reached as an internal state by a single round of absorption from a state z_p which has a tail. That is, there exist $x_p \in D_k$, $x_i \in D_{k'}$, a $z_p \in \{0,1\}^c$ with a tail, and a $z_i \in \{0,1\}^c$ such that

$$\begin{aligned} x_i\|z_i &= \pi(x_p\|z_p \oplus k(x_p)) \\ z &= z_i \oplus k'(x_i), \end{aligned}$$

A tail of z is a string of inputs required to reach z according to the above conditions. Specifically, in case (1) the empty string is the unique tail of $z = 0^c$, and in case (2), any tail of z_p concatenated with x_p is a tail of z. We denote by $\mathsf{tail}(z)$ the set of tails of z.

A key property is that the tail does *not* depend on D_h. This is because h affects the top wire of the sponge immediately before another input is absorbed, and the input may be arbitrarily chosen. The adversary has full control over this wire during the

absorption phase, and can simply undo the application of h if it wishes. In this sense, the function h does not prevent the adversary from reaching any given capacity state of the sponge—but, as it turns out, h will be critical when we discuss obtaining final outputs from the sponge. On the other hand, the functions k and k' serve to ensure that the adversary can reach only a small number of states with a small number of queries, and furthermore that this set of reachable states is sufficiently random. This is necessary because an adversary that knows two tails that reach the same state can easily construct many collisions in the sponge, breaking security.

Towards making this precise, we first define the notion of an intermediate pair. This will help us to treat queries to k and k' somewhat symmetrically.

Definition (Intermediate pair)**.** *Fix a database pair $(D_k, D_{k'})$. We say a pair (x, z) is an intermediate pair if there exists $x_p \in D_k$ and a $z_p \in \{0,1\}^c$ with a tail such that*

$$x \| z = \pi(x_p \| z_p \oplus k(x_p)) \,.$$

We let $\mathsf{IP}(D_k, D_{k'})$ denote the set of all intermediate pairs.

Note that because π is a permutation, there is no multiplicity for intermediate pairs. We are now ready to introduce the concept of a good database.

Definition (Good)**.** *A database pair $(D_k, D_{k'})$ is **good** if every z has at most one tail, and $\mathsf{IP}(D_k, D_{k'})$ has unique x-values. In other words, we require both of the following conditions:*

$$\forall z \in \{0,1\}^c,\ |\mathsf{tail}(z)| \leq 1 \tag{6}$$

$$\forall (x_1, z_1), (x_2, z_2) \in \mathsf{IP}(D_k, D_{k'}),\ (x_1, z_1) \neq (x_2, z_2) \Leftrightarrow x_1 \neq x_2 \tag{7}$$

We show in Section 5 that any randomly selected output of k or k' is exponentially unlikely to create either an intermediate pair prefix collisions or a state collision. The intuition is that these are both collision type events, and so the number of possible "bad outputs" is bounded by $\mathsf{poly}(t)$ after t queries. Using the framework of compressed oracles [Zha19] and quantum transition capacities [Chu+21], we can use this to show that the quantum databases for D_k and $D_{k'}$ remain almost entirely in the good subspace.

2.2 Query lower bounds

We now briefly describe our approach to proving quantum-query bounds for finding collisions or preimages. The details are given in Section 6. The first step to establishing these bounds is to devise a simple reduction from a well-chosen search task. For the preimage finding case, for example, this task is roughly as follows: given a $y \in \{0,1\}^r$, output a "path" through the sponge that results in output y. Our reduction works for both classical and quantum oracle access, and is based on the indifferentiability of a random permutation oracle from the stateful oracle described in Section 2.1. We also derive the probability of an adversary with classical oracle access having a colliding input-output pair in its database. Taken together, these results are sufficient for deriving classical-query lower bounds.

As it turns out, the tools discussed above can also be used to establish quantum-query lower bounds. This can be done using the framework developed by [Chu+21].

In this framework, the square root of the probability that a database satisfies a predicate P after q queries, denoted $[\![\emptyset \xrightarrow{q} \mathsf{P}]\!]$, can be bounded using

$$[\![\emptyset \xrightarrow{q} \mathsf{P}]\!] \leq \sum_{t=1}^{q} [\![\neg \mathsf{P}_{t-1} \rightarrow \mathsf{P}_t]\!] .$$

Here $[\![\neg \mathsf{P}_{t-1} \rightarrow \mathsf{P}_t]\!]$ is the maximum probability of transitioning, during the t-th query, from not satisfying predicate P to satisfying it. Using the framework of [Chu+21], a bound on the probability of this transition in the classical setting also yields a bound in the quantum case. At a high level, this allows us to translate the classical tools described above into quantum-query lower bounds. The actual proof is more involved; for instance, it requires avoiding certain "bad" databases during the simulation. We do this by separating good and bad database at the beginning, and bounding the bad database case separately by treating them as a predicate.

Theorem 2.1 (quantum collision resistance)**.** *The probability that a quantum algorithm $\mathcal{A}$ with quantum query access to a random permutation $\varphi \in S_{\{0,1\}^{r+c}}$ and its inverse, making at most a total of q queries, returns $m, m' \in (\{0,1\}^r)^{\leq l}$ for $l \leq q$ such that $m \neq m'$ and $\mathsf{Sp}^{\varphi}(m) = \mathsf{Sp}^{\varphi}(m')$, can be upper bounded as*

$$\Pr_{\substack{\varphi \sim S_{\{0,1\}^{r+c}} \\ m,m' \leftarrow \mathcal{A}^{\varphi,\varphi^{-1}}}} [\mathsf{Sp}^{\varphi}(m) = \mathsf{Sp}^{\varphi}(m')] \leq O\left(q^5 n 2^{-\min(r,c)}\right) .$$

The derivation for the bound of preimage finding is very similar to the collision finding case. The bound is dominated by terms contributed as a result of the bad database predicate.

Theorem 2.2 (quantum preimage resistance)**.** *Given $y \sim \{0,1\}^r$, the probability that a quantum algorithm $\mathcal{A}$ with quantum query access to a random permutation $\varphi \in S_{\{0,1\}^{r+c}}$ and its inverse, making at most a total of q queries, returns $m \in (\{0,1\}^r)^{\leq l}$ for $l \leq q$ such that $y = \mathsf{Sp}^{\varphi}(m)$, can be upper bounded as*

$$\Pr_{\substack{y \sim \{0,1\}^r \\ m' \leftarrow \mathcal{A}^{\varphi,\varphi^{-1}}}} [y = \mathsf{Sp}^{\varphi}(m)] \leq O\left(q^5 n 2^{-\min(r,c)}\right) .$$

These results are not tight. A sponge collision can be found in $O\left(2^{\min(r,c)/3}\right)$ quantum queries using, e.g., the BHT algorithm for collision finding in case $r < c$, and the BHT algorithm for claw finding in case $c \leq r$ ("meet in the middle"). A preimage can be found using $O\left(\min\left(2^{c/3}, 2^{r/2}\right)\right)$ queries using Grover or claw-finding BHT.

2.3 Sponge indifferentiability

Showing indifferentiability amounts to constructing a simulator $\mathcal{S}$ which answers permutation queries in a way consistent with the ideal (i.e., random-oracle) functionality f corresponding to the sponge. We show the following bound.

Theorem 2.3. *There exists an efficient simulator $\mathcal{S}$ for the sponge construction such that all adversaries $\mathcal{A}$ making q queries of block length at most l have distinguishing advantage*

$$\left\|\Pr[\mathcal{A}^{\varphi,\varphi^{-1},\mathsf{Sp}^{\varphi}}() = 1 - \mathcal{A}^{\mathcal{S}^{f},f}() = 1]\right\| = O\left(l^2\sqrt{q^9 2^{-\min(r,c)}} + l^3\sqrt[4]{q^5 2^{-\min(r,c)}}\right).$$

We now briefly discuss our approach for constructing and analyzing $\mathcal{S}$. For this discussion only, we restrict to a single round of squeezing. Our simulator is somewhat similar to Zhandry's quantum simulator for Merkle-Damgård [Zha19], in that it will analyze the stored databases to determine whether the adversary is computing on an input to the sponge (in which case it answers using the ideal functionality), or on an arbitrary "disconnected" input (in which case it answers using a compressed oracle).

The simulator $\mathcal{S}^f$ will simulate π with a pseudorandom permutation on $\{0,1\}^{r+c}$ and use it only as a black box. It must also provide oracles h, k, k' which are, together with π, consistent with the ideal functionality f, so that it appears to the adversary as if f is the sponge built from h, k, k' and π. For k and k', the simulator simply answers queries using a compressed oracle database. By our bounds on the probability of a bad event, these databases will remain almost entirely on the subspace where each state value z has a unique tail.

Our procedure for answering queries to h is analyzed with a slightly modified sponge construction, the "Msponge". Whenever the sponge XORs an input block into the top wire, the Msponge will instead replace the content of the top wire with that input block. In Section 5, we show that indifferentiability of the Msponge implies indifferentiability of the sponge. The intuition is that one can compute the output of an l-block input to the standard sponge using l queries to the Msponge by simply querying on each prefix, and then adapting the following block by XORing in the output before calling on the next prefix. This reduction runs both ways, so we simply work with the Msponge.

Conceptually, this makes it clear that only the final h query is relevant, as all others are applied to a wire that is then immediately discarded. We show that the databases for k and k' will contain a complete record of the absorbed input whenever the adversary computes the sponge. Then, on the final h query to z, the simulator can reconstruct $\mathsf{tail}(z)$, which is precisely the input used to reach said z value. At this stage, the simulator knows to answer using the ideal functionality f instead. Of course, this analysis only applies on good databases.

While our analysis is conceptually similar to Zhandry's Merkle-Damgård indifferentiability proof [Zha19], the technical details are somewhat more involved due to the complexity of our construction. Further, we address the gap in the original work mentioned at the end of Section 1.2. This gap stems from the fact that projections onto "valid" databases (which always contain an output after decompressing) and "good" databases (which project out certain unwanted databases) do not commute. We elaborate in Section 7.6, but it is worth noting that repairing this gap is one of the main reasons our indifferentiability bound is looser than our query bounds[3]. We expect that a similar idea could be applied to Merkle-Damgård as well, and would result in a similarly looser bound.

[3]Essentially, because we cannot project onto both good and valid simultaneously, we pay for the component on each subspace on each query.

3 Preliminaries

3.1 Quantum

We consider quantum states as unit vectors of a Hilbert space $\mathbb{C}^D$. We define the Euclidean norm of a vector in such a Hilbert space as $\| |\psi\rangle \|^2 = \langle\psi|\psi\rangle$.

Definition 3.1. *The Euclidean distance between quantum states $|\psi\rangle, |\phi\rangle \in \mathbb{C}^D$ is*

$$d(|\psi\rangle, |\phi\rangle) := \min_{\theta} \left\| |\psi\rangle - e^{i\theta} |\phi\rangle \right\| = \sqrt{2(1 - |\langle\psi|\phi\rangle|)}.$$

We will need the following standard result that close quantum states cannot be distingushed.

Claim 3.2. *Let $|\psi\rangle, |\phi\rangle \in \mathbb{C}^D$ be quantum states that satisfy*

$$d(|\psi\rangle, |\phi\rangle) = \epsilon.$$

Then, no measurement can distinguish (a single copy of) $|\psi\rangle$ from $|\phi\rangle$ with advantage exceeding ϵ.

An operator $O : \mathbb{C}^{D_1} \to \mathbb{C}^{D_2}$ is a linear map between Hilbert spaces. We say that O is an isometry if inner products are preserved up to global phase. We say that O is a unitary if it is an isometry, and $D_1 = D_2$. We can define both a norm and a Euclidean distance measure on operators.

Definition 3.3. *The norm of an operator $O : \mathbb{C}^{D_1} \to \mathbb{C}^{D_2}$ is*

$$\|O\| := \max_{|\psi\rangle} \frac{\|O|\psi\rangle\|}{\||\psi\rangle\|}.$$

Definition 3.4. *The distance of two operators $O, O' : \mathbb{C}^{D_1} \to \mathbb{C}^{D_2}$ is*

$$d(O, O') := \max_{|\psi\rangle, \||\psi\rangle\|=1} d(O|\psi\rangle, O'|\psi\rangle).$$

Note that $d(O, O') \leq \|O - O'\|$, and similarly for states. We will occasionally say that an isometry "appends a blank register" or a similar statement, for instance an isometry A which acts as

$$A_X |x\rangle_X = |x\rangle_X |0\rangle_Y.$$

We use subscripts to denote the (domain) Hillbert space on which an operator acts. We may define the commutator on operators whose domain and range are equal in the usual way, though by an abuse of notation we also sometimes define a commutator between operators where one has an expanded range. This is defined as follows.

Definition 3.5. *Let $V : \mathcal{H}_A \to \mathcal{H}_{AB}$ and $U : \mathcal{H}_A \to \mathcal{H}_A$ be operators. We define their commutator as an operator from $\mathcal{H}_A \to \mathcal{H}_{AB}$ as*

$$[U_A, V_A] = U_A V_A - V_A U_A,$$

where the operator $U_A V_A$ has U acting on the subspace $\mathcal{H}_A$ of $\mathcal{H}_{AB}$.

We will generally use consistent labeling within sections for subsystems, which allows one to track the range of an operator with distinct domain and range.

Finally, the following lemmas will prove useful in our analysis.

Lemma 3.6. *Let A, B, B' be Hilbert spaces where B is of a smaller or equal dimension to B'. Let $|\psi\rangle_{AB} \in \mathcal{H}_{AB}$ and $|\phi\rangle_{AB'} \in \mathcal{H}_{AB'}$ be quantum states. Suppose that there exists an isometry $V : \mathcal{H}_B \to \mathcal{H}_{B'}$ such that*

$$d(|\phi\rangle_{AB'}, I_A \otimes V_B |\psi\rangle_{AB}) \leq \epsilon. \tag{8}$$

Then no measurement of subsystem A can distinguish $|\psi\rangle_{AB}$ from $|\phi\rangle_{AB'}$ with advantage exceeding ϵ.

Proof. The mixed state of subsystem A is invariant under any quantum channel applied to the other subspace, so $|\psi\rangle_{AB}$ cannot be distinguished from $I \otimes V |\psi\rangle_{AB}$ by a measurement on subsystem A. However, by Claim 3.2, no measurement can distinguish $I \otimes V |\psi\rangle_{AB}$ from $|\phi\rangle_{AB'}$ with advantage exceeding ϵ. □

Lemma 3.7. *Let $O : \mathcal{H}_N \to \mathcal{H}_N$ be a quantum operator. Consider orthogonal subspaces $S_1, \ldots, S_m \in \mathcal{H}_N$ of dimension $\dim(S_i)$, where the S_i span all of $\mathcal{H}_N$. Suppose that*

$$\forall |x_i\rangle \in S_i, |x_j\rangle \in S_j, i \neq j, \langle x_i | O^\dagger O | x_j \rangle = 0, \tag{9}$$

and we further have

$$\forall |x_i\rangle \in S_i, \|O |x_i\rangle\| \leq \epsilon_i.$$

Then we have

$$\|O\| \leq \max_{i \in [m]} \epsilon_i.$$

Proof. [Don+21], Lemma 2.1. □

Lemma 3.8. *Let $\Pi : \mathcal{H} \to \mathcal{H}$ be a projector, and $|\psi\rangle \in \mathcal{H}$ be a unit norm quantum state. Suppose that*

$$\|\Pi |\psi\rangle\| \geq 1 - \epsilon.$$

Then we have

$$\left\| \Pi^\perp |\psi\rangle \right\| \leq \sqrt{2\epsilon}.$$

Proof. We have

$$\begin{aligned}\||\psi\rangle\|^2 = \|\Pi|\psi\rangle\|^2 + \left\|\Pi^\perp|\psi\rangle\right\|^2 \\ \geq (1-\epsilon)^2 + \left\|\Pi^\perp|\psi\rangle\right\|^2,\end{aligned}$$

and $\||\psi\rangle\|^2 = 1$. Rearranging these inequalities, we obtain

$$\begin{aligned}\left\|\Pi^\perp|\psi\rangle\right\|^2 \leq 2\epsilon - \epsilon^2 \\ \leq 2\epsilon.\end{aligned}$$

□

3.2 Indifferentiability

Let us consider proving that oracle O and construction C^O is indifferentiable from oracle Q. Often, Q will be a random oracle.

Definition 3.9. *Consider a construction C^O with access to an oracle O, which syntactically matches ideal primitive Q. A simulator $\mathcal{S}$ is* (q,ϵ)-indifferentiable *for construction C if, for all q-query quantum algorithms $\mathcal{D}$,*

$$\left|\Pr[\mathcal{D}^{\mathcal{S}^Q,Q}() = 1] - \Pr[\mathcal{D}^{O,C^O}() = 1]\right| \leq \epsilon.$$

The construction C^O is indifferentiable from Q if there exists an efficient simulator that is $(q(n), \mathrm{negl}(n))$-indifferentiable for all polynomial $q(n)$.

The following notions and helper lemma will be useful.

Definition 3.10. *A simulator $\mathcal{S}$ is* (q,α)-indistinguishable *if, for all q-query quantum algorithms $\mathcal{D}$,*

$$\left|\Pr[\mathcal{D}^{\mathcal{S}^Q}() = 1] - \Pr[\mathcal{D}^{O}() = 1]\right| \leq \alpha.$$

Definition 3.11. *A simulator $\mathcal{S}$ is* (q,β)-consistent *if, for all q-query quantum algorithms $\mathcal{D}$,*

$$\left|\Pr[\mathcal{D}^{\mathcal{S}^Q,Q}() = 1] - \Pr[\mathcal{D}^{\mathcal{S}^Q,C^{\mathcal{S}^Q}}() = 1]\right| \leq \beta.$$

Lemma 3.12 ([Zha19])**.** *A simulator which is (q,α)-indistinguishable and (q,β)-consistent is $(q,\alpha+\beta)$-indifferentiable.*

It is worth observing that indifferentiability is not in general symmetric, because the simulator is allowed to be stateful. However, it is transitive in the following sense, which will be used to justify our modifications of the sponge.

Lemma 3.13. *Suppose that oracle A and construction C_1^A is indifferentiable from oracle B. Suppose that oracle B and construction C_2^B is indifferentiable from oracle Q. Then oracle A and construction $C_2^{C_1^A}$ is indifferentiable from oracle Q.*

Proof. We prove this via a simple sequence of hybrids, where $\mathcal{D}$ is a q-query distinguisher taking oracles which syntactically match A and Q. Let $\mathcal{S}_1^B$ be a simulator in the indifferentiability experiment between A and B, and $\mathcal{S}_2^Q$ be the simulator in the indifferentiability experiment between B and Q. Let $\lambda \in \mathbb{N}$ be our security parameter.

(Hybrid 1) The provided oracles are $A, C_2^{C_1^A}$. Let $p_0 = \Pr[\mathcal{D}^{A, C_2^{C_1^A}}() = 1]$.

(Hybrid 2) The provided oracles are $\mathcal{S}_1^B, C_2^B$. Let $p_1 = \Pr[\mathcal{D}^{\mathcal{S}_1^B, C_2^B}() = 1]$.

(Hybrid 3) The provided oracles are $\mathcal{S}_1^{\mathcal{S}_2^Q}, Q$. Let $p_2 = \Pr[\mathcal{D}^{\mathcal{S}_1^{\mathcal{S}_2^Q}, Q}() = 1]$.

The first to the second hybrid replaces the A oracle with S_1^B, and the C_1^A oracle with B. We know that $\mathcal{D}$ and C_2 are both efficient and therefore make a polynomial number of queries to these oracles, and so by the indifferentiability of A, C_1^A from B we have $|p_1 - p_0| = \mathrm{negl}(\lambda)$. The second to the third hybrid similarly replaces B and C_2^B with $\mathcal{S}_2^Q$ and Q. Once again, the distinguisher, simulators, and constructions are all efficient, so by indifferentiability of B, C_2^B from Q we have $|p_2 - p_1| = \mathrm{negl}(\lambda)$. The union bound now implies the result. □

3.3 The sponge construction

The sponge is a construction that uses a fixed-length permutation to produce a function whose domain and codomain are arbitrarily large strings. Classically, the sponge is known to be indifferentiable from a random oracle [Ber+08]. The international hash standard SHA-3 is a sponge construction instantiated with the public Keccak permutation family [Ber+11b; Ber+11a].

Let r and c be positive integers, and let $\varphi : \{0,1\}^{r+c} \to \{0,1\}^{r+c}$ be a permutation. This data defines a sponge function Sp^φ, described below. The natural security parameters are the "capacity" c and the "rate" r. The rate and capacity of a state $x \in \{0,1\}^n$ are denoted with $x|_0^r$ and $x|_r^{r+c}$, respectively. In general, $x|_a^b = x_a x_{a+1} \dots x_{b-1}$ for $x = x_0 x_1 \dots x_{n-1}$.

Definition 3.14. *A string $x \in \{0,1\}^*$ is a valid input to the sponge if it is of the form $x = x_1 \| \dots \| x_p$ for $x_i \in \{0,1\}^r$, where $p \geq 1$ and $x_p \neq 0^r$.*

To obtain a proper function $\{0,1\}^* \to \{0,1\}^*$, we can use an injective pad function from $\{0,1\}^*$ to the set of valid sponge inputs, and then apply the sponge. A simple example of such a function is

$$\mathsf{PAD}(x) = x \| 10^{r-(|x|+1 \mod r)}.$$

The sponge function Sp^φ is defined as follows. To process an input, we initialize the state $0^r \| 0^c$. We then alternate (i.) XORing in the next block of the input to the top r bits, beginning with x_1, with (ii.) applying the permutation φ to the state, until the entire input is "absorbed". To produce the output, we then alternate (i.) outputting the top r bits with (ii.) applying φ, until as many bits as needed are output. The

process of producing the output is called squeezing, and is what gives the sponge unbounded range.

As an alternative to a function with unbounded range, we can consider a random function which maps inputs of the form $x = x_1 \| \dots \| x_p$ for $x_i \in \{0,1\}^r$ *but* which possibly end in 0^r, to random r bit strings. While such a construction has only finite output length, using a PAD function satisfying the above it naturally corresponds to functions with unbounded range in the following way. Let $f : (\{0,1\}^r)^* \to \{0,1\}^r$ be such a function, and define function $g : \{0,1\}^* \to \{0,1\}^*$ as

$$g(x) = f(\mathsf{PAD}(x)) \| f(\mathsf{PAD}(x) \| 0^r) \| f(\mathsf{PAD}(x) \| 0^{2r}) \dots .$$

In this way, we have the following remark.

Claim 3.15. *Suppose that the sponge construction with a single round of squeezing and no constraint on the inputs,* $\mathsf{Sp}^\varphi : (\{0,1\}^r)^* \to \{0,1\}^r$*, is indifferentiable from a random oracle* $f : (\{0,1\}^r)^* \to \{0,1\}^r$*. Then, the full sponge construction with arbitrary squeezing and a valid pad function is indifferentiable from a random oracle* $f : \{0,1\}^* \to \{0,1\}^*$*.*

We will prove the former, i.e., that $\mathsf{Sp}^\varphi : (\{0,1\}^r)^* \to \{0,1\}^r$, is indifferentiable from a random oracle.

4 Compressed oracle framework

The compressed oracle technique, introduced by Zhandry [Zha19], can be viewed as a quantum analog of lazy sampling. Since it is the main technical workhorse of our result, we give a complete and self-contained introduction to our formulation of the technique. The exposition is most similar to that of [Don+21; Chu+21].

By analyzing a suitable purification of a random oracle[4] in a certain "compressed" basis, one obtains a query transcript corresponding to points the adversary has queried. This technique is essentially the only known tool for stateful simulation of a quantum-accessible randomized oracle, as is required for quantum indifferentiability proofs of domain extenders. Further, the technique allows for simple and direct proofs of query lower bounds for many natural random oracle search problems.

4.1 Purified oracles

Let $f : [M] \to [N]$ be a uniform random function. We are interested in analyzing quantum algorithms that query such functions, and which know nothing about f other than the distribution from which it was picked. For this introductory section we will neglect the algorithms workspace. We will assume here that $M = 2^m$ and $N = 2^n$ for integers m and n.

Going forward, we will use subscripts to indicate which quantum register(s) a state is supported on: here X will denote a register holding an input, Y a register holding an output. A subscript of an operator or a state is always a label of a register; we use superscripts for additional information. We will sometimes omit labels on states or operators if it is clear from context. We define the quantum oracle for f as O^f, which acts on an input and output register as

$$O^f \left|x\right\rangle_X \left|y\right\rangle_Y := \left|x\right\rangle_X \left|y \oplus f(x)\right\rangle_Y .$$

We could instead write a purified oracle $\mathcal{P}$ that acts on an input, output, and a third function register (subscript F) as

$$\mathcal{P} \left|x\right\rangle_X \left|y\right\rangle_Y \left|f\right\rangle_F := \left|x\right\rangle_X \left|y \oplus f(x)\right\rangle_Y \left|f\right\rangle_F .$$

The third register has the set $\mathfrak{F}$ of all functions $[M] \to [N]$ as computational basis. Such a representation allows us to model a distribution on functions as the purified superposition over functions. As f is drawn uniformly at random, we can instead prepare an initial function oracle

$$\left|\mathfrak{F}\right\rangle_F := \frac{1}{\sqrt{|\mathfrak{F}|}} \sum_{f \in \mathfrak{F}} \left|f\right\rangle_F , \tag{10}$$

and replace each application of O^f with the purified oracle $\mathcal{P}$. Measuring the F register to obtain f after an algorithm has finished, and drawing f uniformly at random and then answering queries with O^f, yield identical joint distributions of f and the algorithms (possibly quantum) output, showing that the purified oracle is an exactly faithful simulation of a random oracle.

[4] More precisely: a suitable Stinespring dilation (in fact, a generalization thereof) of the random oracle viewed as a quantum channel with memory

4.2 The compressed basis

We will now introduce the "compressed basis", which is a convenient basis in which to analyze the function register. First, let us consider enlarging the Hilbert space of the function register to the span of the set $\mathfrak{D}$ of functions from $[M]$ to $[N] \cup \{\perp\}$. The resulting function register can be viewed as an $(M+1)$-dimensional register for every output. In particular, for $g \in \mathfrak{D}$,

$$|g\rangle_F = |g(0)\rangle_0 |g(1)\rangle_1 \dots |g(M-1)\rangle_{M-1} .$$

We will use $|\mu\rangle$ to denote the uniform superposition over (non-$\perp$) outputs, i.e.,

$$|\mu\rangle = \frac{1}{\sqrt{N}} \sum_{y \in [N]} |y\rangle .$$

The initial superposition over all total functions (i.e., functions with no $\perp$ outputs) can then be written as

$$|\mathfrak{F}\rangle_F = \bigotimes_{x \in [M]} |\mu\rangle_x .$$

The transformation from the standard to the compressed basis is the unitary which, for each input x, exchanges the uniform superposition $|\mu\rangle_x$ with $|\perp\rangle_x$.

Definition 4.1. *Define the register compression function $C^x \in \mathrm{SU}(N+1)$, acting on register $x \in [M]$ of a function, as*

$$C^x := I_x - |\perp\rangle\langle\perp|_x - |\mu\rangle\langle\mu|_x + |\perp\rangle\langle\mu|_x + |\mu\rangle\langle\perp|_x.$$

We can use this to define the full compression function $C_F \in \mathrm{SU}(\mathfrak{D})$ as

$$C_F := \bigotimes_{x \in [M]} C^x.$$

It is easy to verify that $C^2 = I$, and the initial purification compresses to the constant $\perp$ function, meaning $C |\mathfrak{F}\rangle_F = \bigotimes_{x \in [M]} |\perp\rangle_x$. Going forward, it will be helpful to denote basis vectors in the compressed basis by the set of non-$\perp$ input-output pairs (i.e. the input values where the partial function is defined, and the corresponding outputs). We call such a set D as a "database". We write $x \in D$ to mean that there exists a pair $(x, y) \in D$ (with $y \neq \perp$), and let $|D|$ denote the total number of such pairs in D.

In this notation, the compression of the uniform superposition over all functions compresses to the empty database, $C |\mathfrak{F}\rangle_F = |\emptyset\rangle_F$. The action of querying can be understood in the compressed basis by undoing the compression operator (which is self-inverse), applying the purified oracle $\mathcal{P}$, and then re-compressing. It turns out that one only needs to (de)compress the register being queried. To define this, we use the notation

$$L_{XF} |x\rangle_X |f\rangle_F := |x\rangle_X C_F^x |f\rangle_F , \tag{11}$$

in other words L is the "local compression", compressing the x-th register of of the F register conditioned on the value x in the first register. Note that this is formally distinct from C^x, which compresses a fixed register; the operators are related by the identity $L_{XF} |x\rangle = C_F^x$.

Lemma 4.2. *We have*

$$C_F \mathcal{P}_{XYF} C_F^\dagger = L_{XF} \mathcal{P}_{XYF} L_{XF}.$$

Proof. Observe that $[C^x, C^{x'}] = 0$, and for $x' \neq x$ we have $\mathcal{P} C^{x'} |x\rangle_X |y\rangle_Y |f\rangle_F = C^{x'} \mathcal{P} |x\rangle_X |y\rangle_Y |f\rangle_F$. Working out the action on an arbitrary basis state, have

$$\begin{aligned} C_F \mathcal{P}_{XYF} C_F^\dagger |x\rangle_X |y\rangle_Y |f\rangle_F &= C_F \mathcal{P}_{XYF} C_F |x\rangle_X |y\rangle_Y |f\rangle_F \\ &= \left(\bigotimes_{x' \in [M]} C^{x'} \right) \mathcal{P}_{XYF} \left(\bigotimes_{x'' \in [M]} C^{x''} \right) |x\rangle_X |y\rangle_Y |f\rangle_F \\ &= \left(\bigotimes_{x' \in [M]} C^{x'} \right) \left(\bigotimes_{x'' \in [M], x'' \neq x} C^{x''} \right) \mathcal{P}_{XYF} C_x |x\rangle_X |y\rangle_Y |f\rangle_F \\ &= L_{XF} \mathcal{P}_{XYF} L_{XF} |x\rangle_X |y\rangle_Y |f\rangle_F \end{aligned}$$

□

Definition 4.3. *We will sometimes use the notation $\mathsf{cO}_{XYF} := L_{XF} \mathcal{P}_{XYF} L_{XF}$ to denote compressed oracle calls.*

Consider an experiment where a t query adversary has workspace A, query register X and output register Y, interacting with a random function oracle initialized as the empty database in register F. Parameterizing the adversary by unitaries $U^0, U^1, \ldots, U^t$, we can write the final state

$$|\psi^t\rangle_{AXYF} = U^t_{AXY} \ldots U^1_{AXY} \mathsf{cO}_{XYF} U^0_{AXY} |0\rangle_A |0\rangle_X |0\rangle_Y |\emptyset\rangle_F .$$

Corollary 4.4 (of Lemma 4.2)**.** *In the state $|\psi^t\rangle$, the function register is supported fully on databases having at most t non-$\perp$ output values.*

Note that the function register needn't have exactly t non-$\perp$ values; indeed, quantum algorithms can forget already learned outputs by uncomputing them, in which case they will not appear in the database.

Another important Corollary is that there is always an image of x after decompressing the x-th register, so long as the databases are queried using cO. We call the span of such states valid. To define them, we use the notation $\Pi_F^{x \in \mathsf{db}} := I_x - |\perp\rangle\langle\perp|_x$.

Definition 4.5. *We say that a database state $|\psi\rangle_F$ is "valid" if for all $x \in [M]$, it satisfies*

$$\Pi^{x \in \mathsf{db}} C^x |\psi\rangle = C^x |\psi\rangle .$$

Observe that such states form a subspace, and we use Π^v to denote the projector onto this subspace.

Corollary 4.6 (of Lemma 4.2)**.** *The operation cO preserves membership in the valid subspace.*

We will sometimes also need a projector onto databases containing a certain value, controlled on an external register. We use the notation $\Pi_{XF}^{\in \mathsf{db}}$ for this, such that

$$\Pi_{XF}^{\in \mathsf{db}} |x\rangle_X = \Pi_F^{x \in \mathsf{db}}.$$

We have seen that the initial state $|\emptyset\rangle_F$ in the compressed basis evolves to a superposition of input-output pairs as queries are made. These states are sometimes called compressed database states. We will now analyze how these database states evolve under the query operator.

4.3 Transition capacities

Here we recall the framework of [Chu+21] for proving query bounds using the compressed oracle. Our exposition of the compressed oracle is slightly different, and so in some places we include alternative proofs. Note also that certain aspects of the framework described here are simplified from that of [Chu+21], as we are not concerned with parallel queries.

Recall that $\mathfrak{D}$ denotes the set of functions from $[M]$ to $[N] \cup \{\perp\}$. We let $\mathfrak{D}_t$ denote the subset of $\mathfrak{D}$ to functions with at most t non-$\perp$ values.

Let us define a predicate P as a subset of databases. A simple example of a predicate is the collision predicate C over $\mathfrak{D}$, which is the set of all databases which contain points $x, x' \in D$ with $x \neq x'$ and $D(x) = D(x')$. Recall that in our notation $x \in D \Leftrightarrow D(x) \neq \perp$, so collisions on $\perp$ do not count.

Definition 4.7. *A predicate P is a subset of databases $P \subset \mathfrak{D}$. We sometimes also refer to a database $D \in \mathfrak{D}$ as "satisfying" P if $D \in \mathsf{P}$, or say $\mathsf{P}(D) = \top$ in this case (and otherwise $\mathsf{P}(D) = \perp$).*

A predicate has a negation, and a restriction to databases of size at most t, which we denote as follows.

$$\neg\mathsf{P} = \mathfrak{D} \setminus \mathsf{P}, \qquad \mathsf{P}_t = \mathfrak{D}_t \cap \mathsf{P}, \qquad \neg\mathsf{P}_t = \mathfrak{D}_t \setminus \mathsf{P}_t.$$

The quantum transition capacity between predicate $\neg\mathsf{P}$ and predicate Q measures the amplitude transfer from databases in $\neg\mathsf{P}$ to ones in Q under a single query.

Definition 4.8. *The single-query quantum transition capacity from predicate $\neg\mathsf{P}$ to predicate Q is defined as follows.*

$$[\![\neg\mathsf{P} \to \mathsf{Q}]\!] = \left\| \Pi_F^{\mathsf{Q}} C_F \mathcal{P}_{XYF} C_F^\dagger \Pi_F^{\neg\mathsf{P}} \right\|.$$

Similarly, the quantum transition capacity for q sequential queries is defined as

$$[\![\neg\mathsf{P} \xrightarrow{q} \mathsf{Q}]\!] = \sup_{U_1,\ldots,U_{q-1}} \|\Pi_F^{\mathsf{Q}} C_F \mathcal{P}_{XYF}\, U_{q-1}\, \mathcal{P}_{X,Y,F} \cdots \mathcal{P}_{XYF}\, U_1\, \mathcal{P}_{XYF}\, C_F^\dagger \Pi_F^{\neg\mathsf{P}}\|$$

where the supremum is over all positive $d \in \mathbb{Z}$ and all unitaries $U_1, \ldots, U_{q-1}$ acting on $\mathbb{C}[\mathcal{X}] \otimes \mathbb{C}[\mathcal{Y}] \otimes \mathbb{C}^d$.

Note that the definition of the quantum transition capacity is the same as in [Chu+21], Definition 5.5. We will state some of its properties that we will use while deriving the query lower bounds in the quantum adversary setting. We list them under the same lemma.

Lemma 4.9 ([Chu+21], Lemma 5.6). *For any sequence of predicates* $\mathsf{P}_0, \mathsf{P}_1, \ldots, \mathsf{P}_q$,

$$[\![\neg \mathsf{P}_0 \xrightarrow{q} \mathsf{P}_q]\!] \leq \sum_{s=1}^{q} [\![\neg \mathsf{P}_{s-1} \to \mathsf{P}_s]\!] .$$

Lemma 4.10 ([Chu+21], Lemma 5.31). *For any predicates* P, P' *and* Q, *we have*

$$[\![\mathsf{Q} \to \mathsf{P}]\!] \leq [\![\mathsf{Q} \to \mathsf{P} \cup \mathsf{P}']\!],$$

$$[\![\mathsf{Q} \to \mathsf{P} \cup \mathsf{P}']\!] \leq [\![\mathsf{Q} \to \mathsf{P}]\!] + [\![\mathsf{Q} \to \mathsf{P}']\!], \text{ and}$$

$$[\![\mathsf{P} \cap \mathsf{P}' \to \mathsf{Q}]\!] \leq [\![\mathsf{P} \to \mathsf{Q}]\!].$$

Let us now introduce the concept of a local property. Informally, a local property is a predicate which is determined only by the value of a database on a single special input point. Further, local properties are required to be monotone w.r.t. the special input. We will here use the notation $D|_x$ to denote the set of databases which agree with D on all points $x' \neq x$. The following definition is adapted from [Chu+21], removing an ambiguity over which domain the local property is to be considered.

Definition 4.11. *Let* $\mathsf{L}^{x,D} \subseteq D|_x$ *be a property parameterized by an* $x \in \mathcal{X}$ *and* $D \in \mathfrak{D}$. *Consider* $\mathsf{L}^{x,D} : D|_x \to \{0,1\}$ *as a function to map* D' *to* 1 *if* $D' \in \mathsf{L}^{x,D}$ *and* 0 *otherwise. As a database property* $\mathsf{L}^{x,D}$ *is local if*

1. $\exists \hat{\mathsf{L}}^x : \bar{\mathcal{Y}} \to \{0,1\}$ *with* $\hat{\mathsf{L}}^x(D'(x)) = \mathsf{L}^{x,D}(D')$ *for all* $D' \in D|_x$,
2. $D' \in \mathsf{L}^{x,D} \wedge D'(x) = \perp$, *then* $D'[x \mapsto r] \in \mathsf{L}^{x,D}$ *for all* $r \in \mathcal{Y}$.

We will usually use the symbol L to refer to a collection $\mathsf{L} := \{\mathsf{L}^{x,D} : x \in \mathcal{X}, D \in \mathfrak{D}\}$ of local properties. By an abuse of notation, we will also sometimes use L to denote the union of these sets $\mathsf{L} := \bigcup_{x,D} \{\mathsf{L}^{x,D} : x \in \mathcal{X}, D \in \mathfrak{D}\}$, depending on context. We say that L interpolates the transition $\neg \mathsf{P} \to \mathsf{Q}$ if it satisfies $\mathsf{Q} \subset \mathsf{L} \subset \mathsf{P}$ (here using the latter interpretation). We can define the distance Δ of a local property as the maximum number of images of the special point which change the truth value of the property.

Definition 4.12. *The distance* Δ *of a local property* $\mathsf{L}^{x,D}$ *is defined as*

$$\Delta(\mathsf{L}^{x,D}) := |\{y : \mathsf{L}^{x,D}(D[x \to y]) \neq \mathsf{L}^{x,D}(D[x \to \perp])\}|.$$

The distance Δ *of a collection of local properties* L *is defined as*

$$\Delta(\mathsf{L}) := \max_{\mathsf{L}^{x,D} \in \mathsf{L}} \Delta(\mathsf{L}^{x,D}).$$

We can bound quantum transition capacities using local properties. In particular, if we can construct a family of local properties which interpolate the transition, the distance of this local property bounds the transition capacity.

Lemma 4.13. *Let* $\neg \mathsf{P}$ *and* Q *be predicates, and suppose the local property family* L *interpolates the transition* $\neg \mathsf{P} \to \mathsf{Q}$ *(meaning we have* $\mathsf{Q} \subseteq \mathsf{L} \subseteq \mathsf{P}$*). The quantum transition capacity then satisfies*

$$[\![\neg \mathsf{P} \to \mathsf{Q}]\!] \leq 4\sqrt{\Delta(\mathsf{L})/N}.$$

Proof. We have $\mathsf{Q} \subseteq \mathsf{L}$ and $\neg\mathsf{P} \subseteq \neg\mathsf{L}$ by assumption, which implies that $\Pi^{\mathsf{Q}} \leq \Pi^{\mathsf{L}}$ and $\Pi^{\neg\mathsf{P}} \leq \Pi^{\neg\mathsf{L}}$ in the semi-definite order. It follows that

$$\begin{aligned}\llbracket \neg\mathsf{P} \to \mathsf{Q} \rrbracket &= \left\| \Pi_F^{\mathsf{Q}} C_F \mathcal{P}_{XYF} C_F^\dagger \Pi_F^{\neg\mathsf{P}} \right\| \\ &\leq \left\| \Pi_F^{\mathsf{L}} C_F \mathcal{P}_{XYF} C_F^\dagger \Pi_F^{\neg\mathsf{L}} \right\| \\ &= \llbracket \neg\mathsf{L} \to \mathsf{L} \rrbracket.\end{aligned}$$

We will bound the transition capacity $\neg\mathsf{L} \to \mathsf{L}$. Let us define $\mathcal{Y}^{x,D} = \{y : \mathsf{L}^{x,D}(D[x \to y]) \neq \mathsf{L}^{x,D}(D[x \to \bot])\}$ as the set of assignments to x that cause a transition in $\mathsf{L}^{x,D}$. Now we have

$$\begin{aligned}\left\| \Pi_F^{\mathsf{L}} C_F \mathcal{P}_{XYF} C_F^\dagger \Pi_F^{\neg\mathsf{L}} \right\| &= \left\| \Pi_F^{\mathsf{L}} L_{XF} \mathcal{P}_{XYF} L_{XF} \Pi_F^{\neg\mathsf{L}} \right\| \\ &\leq 2 \left\| \Pi_F^{\mathsf{L}} L_{XF} \Pi_F^{\neg\mathsf{L}} \right\| + \left\| \Pi_F^{\mathsf{L}} \mathcal{P}_{XYF} \Pi_F^{\neg\mathsf{L}} \right\| \quad \text{(Orthogonality of } \Pi^{\neg\mathsf{L}}, \Pi^{\mathsf{L}}\text{)}\end{aligned}$$

Observe that $\mathcal{P}$ preserves the computational basis, so the second term is 0. We will focus on bounding the first term, and drop the Y register as it is not acted on. Let us define $\mathcal{Y}^{x,D} = \{y : \mathsf{L}^{x,D}(D[x \to y]) \neq \mathsf{L}^{x,D}(D[x \to \bot])\}$ as the set of assignments to x that cause a transition in $\mathsf{L}^{x,D}$. We have $\Pi_{XF}^{\in\mathsf{db}} + \Pi_{XF}^{\notin\mathsf{db}} = I$, so we can analyze two subspaces separately. Noting that $\Pi^{\mathsf{L}}, \Pi^{\neg\mathsf{L}}$ are diagonal in the computational basis and L_{XF} preserves the X register and the database values outside of X. By Lemma 3.7 we can WLOG consider a state of the form

$$|\psi_{x,D}\rangle = \sum_{z \in [N] \cup \{\bot\} \setminus \mathcal{Y}^{x,D}} \alpha_z |x\rangle_X |D[x \to z]\rangle_F,$$

where $D \notin \mathsf{L}$.

(1) Case $x \notin D$. We have

$$\begin{aligned}\left\| \Pi_F^{\mathsf{L}} L_{XF} \Pi_F^{\neg\mathsf{L}} \Pi_{XF}^{\notin\mathsf{db}} |\psi\rangle \right\| &= |\alpha_\bot| \left\| \Pi_F^{\mathsf{L}} \frac{1}{\sqrt{N}} \sum_{u \in [N]} |x\rangle_X |D[x \to u]\rangle_F \right\| \\ &= |\alpha_\bot| \sqrt{\frac{\Delta(\mathsf{L}^{x,D})}{N}} \\ &\leq \sqrt{\frac{\Delta(\mathsf{L})}{N}}.\end{aligned}$$

(2) Case $x \in D$. We have

$$\begin{aligned}
\left\|\Pi_F^{\mathsf{L}} L_{XF} \Pi_F^{\neg\mathsf{L}} \Pi_{XF}^{\in\mathsf{db}} |\psi\rangle\right\| =& \Bigg\| \Pi_F^{\mathsf{L}} \sum_{z\in[N]\setminus\mathcal{Y}^{x,D}} \alpha_z |x\rangle_X \otimes \\
& \left(|D[x\to z]\rangle_F - \frac{1}{N}\sum_{u\in[N]} |D[x\to u]\rangle + \frac{1}{\sqrt{N}} |D[x\to\perp]\rangle \right) \Bigg\| \\
=& \left\| \frac{1}{N} \sum_{z\in[N]\setminus\mathcal{Y}^{x,D}, u\in\mathcal{Y}^{x,D}} \alpha_z |x\rangle_X |D[x\to u]\rangle \right\| \\
\leq& \frac{1}{N} \sqrt{\sum_{u\in\mathcal{Y}^{x,D}} \left(\sum_{z\in[N]\setminus\mathcal{Y}^{x,D}} |\alpha_z| \right)^2} \\
\leq& \sqrt{\frac{\Delta(\mathsf{L}^{x,D})}{N}} \\
\leq& \sqrt{\frac{\Delta(\mathsf{L})}{N}}
\end{aligned}$$

Putting the cases together, we obtain the bound

$$[\![\neg\mathsf{P}\to\mathsf{Q}]\!] \leq 4\sqrt{\frac{\Delta(\mathsf{L})}{N}}$$

□

We will now see how to connect the recorded set of input-output pairs to the adversary's knowledge, and how to use this to prove query lower bounds.

4.4 Query lower bounds

A key fact about compressed oracles is that the input-output pairs in the compressed database almost entirely capture the adversary's knowledge about the function. This is formally captured by the fundamental lemma, which says that any set of input-output pairs which is not in the compressed database has exponentially small probability to be in the uncompressed database.

Lemma 4.14 (Fundamental lemma)**.** *Let* $\mathbf{x} = \{(x_1, y_1), \ldots (x_l, y_l)\}$ *be a set of* l *input-output pairs with distinct inputs and no* $\perp$ *output. Let* $\Pi^{\mathbf{x}}$ *act on a database state* $|D_f\rangle$ *as the projection onto databases that are consistent with* $\mathbf{x}$*. Suppose further that the database state is valid,* $\Pi^v |D_f\rangle = |D_f\rangle$*. Then we have*

$$\left\|\Pi^{\mathbf{x}} \mathsf{C} |D_f\rangle\right\| \leq \left\|\Pi^{\mathbf{x}} |D_f\rangle\right\| + \sqrt{\frac{l}{N}}.$$

Proof, adapted from [Chu+21]. We have

$$\begin{aligned}
\left\|\Pi^{\mathbf{x}} \mathsf{C} |D_f\rangle\right\| &\leq \left\|\Pi^{\mathbf{x}} |D_f\rangle\right\| + \left\|(\Pi^{\mathbf{x}} - \Pi^{\mathbf{x}}\mathsf{C}) |D_f\rangle\right\| \qquad \text{(Triangle Inequality)} \\
&\leq \left\|\Pi^{\mathbf{x}} |D_f\rangle\right\| + \left\|(\Pi^{\mathbf{x}} - \Pi^{\mathbf{x}}\mathsf{C})\Pi^v\right\|.
\end{aligned}$$

To bound the second term, we can bound the Fourier matrix elements[5], restricted to the l database entries which are acted on.

$$|\langle\tilde{\mathbf{p}}|\left(\Pi^{\mathbf{x}}-\Pi^{\mathbf{x}}C\right)|\tilde{\mathbf{q}}\rangle| \leq \begin{cases} 0 & (\text{If } \tilde{\mathbf{p}} \text{ contains } \perp) \\ 0 & (\text{If } \tilde{\mathbf{q}} \text{ contains no } \perp \text{ or } \tilde{0}) \\ N^{-l} & (\text{If } \tilde{\mathbf{q}} \text{ contains a } \perp \text{ or } \tilde{0}) \end{cases}$$

Finally observe that

$$\Pi^{v}|\tilde{q}\rangle = \begin{cases} 0 & (\text{If } \tilde{q} \text{ contains } \tilde{0}) \\ |\tilde{q}\rangle & (\text{Otherwise}) \end{cases}$$

We can then bound the operator norm by the Frobenius norm to obtain

$$\begin{aligned} \|(\Pi^{\mathbf{x}}-\Pi^{\mathbf{x}}C)\Pi^{v}\|^2 &\leq \sum_{\tilde{\mathbf{p}} \text{ has no } \perp,\, \tilde{\mathbf{q}} \text{ contains } \perp \text{ and no } \tilde{0}} |\langle\tilde{\mathbf{p}}|\left(\Pi^{\mathbf{x}}-\Pi^{\mathbf{x}}C\right)|\tilde{\mathbf{q}}\rangle|^2 \\ &\leq \underbrace{N^l}_{(\tilde{p} \text{ choices})} \cdot \underbrace{lN^{l-1}}_{(\geq \tilde{q} \text{ choices})} \cdot N^{-2l} \\ &\leq \frac{l}{N} \end{aligned}$$

□

This translates to the following algorithmic statement.

Corollary 4.15 ([Chu+21], Corollary 4.2). *Let $R \subseteq \mathcal{X}^l \times \mathcal{Y}^l$ be a relation. Let $\mathcal{A}$ be an oracle quantum algorithm that outputs $(x_1,\ldots,x_l) \in \mathcal{X}^l$ and $(y_1,\ldots,y_l) \in \mathcal{Y}^l$. Let p be the probability that $y_i = H(x_i)$ for all $i = 1,\ldots,l$ and $((x_1,\ldots,x_l),(y_1,\ldots,y_l)) \in R$ when $\mathcal{A}$ has interacted with the standard random oracle, initialized with a uniformly random function H. Similarly, let p' be the probability that $y_i = D(x_i)$ and $((x_1,\ldots,x_l),(y_1,\ldots,y_l)) \in R$ when $\mathcal{A}$ has interacted with the compressed oracle instead and D is obtained by measuring its internal state (in the computational basis). Then*

$$\sqrt{p} \leq \sqrt{p'} + \sqrt{\frac{l}{N}}.$$

4.5 Efficient representation

Observe that a compressed database with at most t input-output pairs can straightforwardly be stored using $O(tn+tm)$ space, by storing (say) a list of input-output pairs with non-$\perp$ outputs sorted by input, potentially with padding if there are fewer than t such pairs. It turns out that one can also efficiently implement the query operator in this representation. In this section, we will describe a canonical simulator which efficiently simulates a random oracle for a quantum algorithm, based on the description of [Zha19].

By Lemma 4.2, it suffices to be able to implement $\mathsf{cO} = L\mathcal{P}L$ as a circuit, controlled on the first register, on the efficient database representation. Note that this operation adds at most a single input/output pair to the database. In this representation, the databases are "padded", such that the list of defined input output pairs appear at the

Algorithm 1: Compression operator circuit

Input: $|\psi\rangle = \sum \alpha_{x,y,D} |x\rangle_X |y\rangle_Y \underbrace{|(x_1, y_1)\rangle_{D_1} \dots |(x_t, y_t)\rangle_{D_t}}_{D}$

Require: $x_i \in [M] \cup \{\infty\}$, $y_i \in [N] \cup \{\perp\}$, D sorted by x_i with no duplicate non-∞ inputs, $y_i = \perp$ if and only if $x_i = \infty$.

Output: $|\psi'\rangle = \sum \alpha_{x,y,D'} |x\rangle |y\rangle \underbrace{|(x'_1, y'_1)\rangle_{D'_1} \dots |(x'_{t+1}, y'_{t+1})\rangle_{D'_{t+1}}}_{D'}$ s.t. $|\psi'\rangle = C^x |\psi\rangle$

(1) Append $(\infty, \perp)$ to the end of register D ▷ Upper case for registers

(2) For every j from 1 up to t:

(1′) Compute flag $f \leftarrow (D_j[0] = X \text{ and } D_{j+1}[0] > X)$ ▷ Zero based indexing

(2′) If f, swap registers $D_j \leftrightarrow D_{j+1}$

(3′) Uncompute $f \leftarrow f \oplus (D_{j+1}[0] = X \text{ and } D_j[0] > X)$ ▷ $\oplus$ is bitwise addition

(3) If $D_{t+1}[1] = \perp$, perform $D_{t+1}[0] \leftarrow D_{t+1}[0] \oplus X \oplus \infty$ ▷ Exchanges values $x \leftrightarrow \infty$

(4) Perform $C^{D_{t+1}[0]}$ on register $D_{t+1}[1]$

(5) If $D_{t+1}[1] = \perp$, perform $D_{t+1}[0] \leftarrow D_{t+1}[0] \oplus X \oplus \infty$

(6) For every j from t down to 1:

(1′) Compute flag $f \leftarrow (D_j[0] = X \text{ and } D_{j+1}[0] > X)$

(2′) If f, swap registers $D_j \leftrightarrow D_{j+1}$

(3′) Uncompute $f \leftarrow f \oplus (D_{j+1}[0] = X \text{ and } D_j[0] > X)$

beginning of the list, followed by dummy $(\infty, \perp)$ pairs such that the stored state is always the same size.

To simulate the full query operator, we can:

(1) Uncompress the database on input x using Algorithm 1

(2) Answer the query using this database

(3) Run Algorithm 1 once again, omitting step (1), to recompress the database.

Note that we may omit Step (1) on re-compression because the $\mathcal{P}$ operator preserves the computational basis on the input register, and we already created space for the x input pair in the first step.

[5] we take the Fourier transform register-wise over $\mathbb{Z}_N$, ignoring the $\perp$ symbol

5 Developing the model

Here we formally develop the view of permutation oracles which we apply to the sponge.

5.1 An alternative picture of permutation oracles

Let $\varphi : \{0,1\}^n \to \{0,1\}^n$ be a random injective function, i.e. a permutation on n bit strings. Let r, c be integers such that $r + c = n$, and let $s|_a^b$ denote the substring of s starting from the a-th (inclusive) through the b-th (exclusive) bit.

Let $\mathfrak{H}$ and $\mathfrak{K}$ be subgroups of the symmetric group S_{2^n} defined by the following sets.

$$\mathfrak{H} := \{\pi \,:\, \pi(x\|y) = (x \oplus h(y))\|y \text{ for some } h : \{0,1\}^c \to \{0,1\}^r\}$$
$$\mathfrak{K} := \{\pi \,:\, \pi(x\|y) = x\|(y \oplus k(x)) \text{ for some } k : \{0,1\}^r \to \{0,1\}^c\}$$

Note that a random $\varphi \sim S_{2^n}$ can equivalently be sampled by choosing a random $\pi \sim S_{2^n}$, and sampling random $\omega_h \sim \mathfrak{H}$, and $\sigma_k, \tau_{k'} \sim \mathfrak{K}$ (defined by functions h, k, k' respectively), and defining

$$\varphi := \omega_h \circ \tau_{k'} \circ \pi \circ \sigma_k. \tag{12}$$

When there is no risk of confusion, we will sometimes write the above permutation simply as $\varphi = hk'\pi k$. We will now consider two access models for φ.

World 1. *An adversary $\mathcal{A}$ receives access to quantum oracles for both φ and φ^{-1}.*

World 2. *An adversary $\mathcal{A}$ receives access to quantum oracles for random functions $h : \{0,1\}^c \to \{0,1\}^r$ and $k, k' : \{0,1\}^r \to \{0,1\}^c$ (each independently uniform random), which correspond to permutations $\omega_h \in \mathfrak{H}$ and $\sigma_k, \tau_{k'} \in \mathfrak{K}$ as described above. The adversary is also given quantum query access to a permutation $\pi : \{0,1\}^n \to \{0,1\}^n$ and its inverse. Define*

$$\varphi := \omega_h \circ \tau_{k'} \circ \pi \circ \sigma_k. \tag{13}$$

It is intuitive that the interface for φ in World 2 is stronger than the interface for φ in World 1. This is because the adversary can use h, k, k' and π to implement φ and φ^{-1}, but is also free to use the oracles in whatever ways it wishes. Indeed, we can formalize this with an indifferentiability proof. Consider the trivial construction $C^\varphi = \varphi$.

Lemma 5.1. *Let World 1′ be like World 1, but $\mathcal{A}$ additionally has oracles for h, k, k' as in World 2, and for*

$$\pi := \tau_{k'} \circ \omega_h \circ \varphi \circ \sigma_k. \tag{14}$$

Then

1. *For all $\mathcal{A}$ in World 1 playing game G^φ there exists $\mathcal{A}'$ in World 1′ such that*

$$\Pr[\mathcal{A} \text{ wins } G^\varphi] = \Pr[\mathcal{A}' \text{ wins } G[\varphi]].$$

Here the notation G^φ means that G depends on φ but not on h, k, k' or π.

2. *World 1′ is indistinguishable from World 2.*

Proof. Statement 1 is clear, A' just ignores its additional oracles. For statement 2, it suffices to observe that the oracles in the two worlds have the same joint distribution. □

In particular, the content of Lemma 5.1 is that for any security game G^{φ}, the maximum winning probability in World 2 upper-bounds the maximum winning probability in World 1. Therefore, it suffices for us to show all lower bounds in the latter model. By Lemma 3.13, and the fact that indistinguishability is a special case of indifferentiability, it suffices to show indifferentiability in this model as well. In certain cases we will consider adversaries which have the full truth table of π, which is clearly a stronger model still.

5.2 An alternative picture of the sponge: the Msponge

We now define an alternative construction that is more suited to our indifferentiability proof. We call this construction "the Msponge". The Msponge is identical to the sponge as defined in Section 3.3 in all aspects except how each block of the input is absorbed. Specifically, in each absorbing round, instead of XORing the next block $x_i \in \{0,1\}^r$ of the input into the rate wire, we discard the contents of the rate wire and replace them with x_i; we then apply the permutation φ to the rate and capacity wires as normal. We emphasize that all other aspects of the Msponge are identical to the sponge. In this section, we show that the sponge and the Msponge are equivalent in an appropriate sense, up to an overhead of the maximum block length.

Let $f, g : (\{0,1\}^r)^* \to \{0,1\}^r$ be random functions. The function f will represent the standard sponge construction, and the function g will represent the Msponge. To define the two constructions $C_1^f, C_2^g : (\{0,1\}^r)^* \to \{0,1\}^r$, we will need to introduce the notion of "fixing" an input. Intuitively, this map describes the correspondence between inputs to f and inputs to g, which are in bijection with one another.

Definition 5.2. *Let $f : (\{0,1\}^r)^* \to \{0,1\}^r$ be a function. For an input of the form $x = x_1 \| \dots \| x_m \in (\{0,1\}^r)^*$, we define $\mathsf{fix}(x)$ recursively as follows.*

(1) The fix of the empty string ϵ is

$$\mathsf{fix}(\epsilon) = \epsilon$$

(2) The fix of a one-block string x_1 is

$$\mathsf{fix}(x_1) = x_1$$

(3) The fix of a multi-block string $t\|x_m$, where t is a non-empty string and $x_m \in \{0,1\}^r$, is

$$\mathsf{fix}(t\|x_m) = \mathsf{fix}(t)\|x_m \oplus f(\mathsf{fix}(t)).$$

It is worth observing that fix is a bijective function, for any function f. Computing the inverse is straightforward, by reversing the above recursion. We can now define our constructions.

Definition 5.3. *Let us define C_1^f as $f \circ \mathsf{fix}$. Let us define C_2^g as $g \circ \mathsf{fix}^{-1}$.*

The key point is that these constructions are indifferentiable from g and f respectively, as formalized in the following lemma. This justifies our choice to prove that the Msponge, where we replace the top wire with the next input rather than XORing it in, is indifferentiable will indeed be sufficient to prove that the standard sponge is indifferentiable from a random oracle.

Lemma 5.4. *The construction C_1^f is perfectly indifferentiable from g, and the construction C_2^g is perfectly indifferentiable from f.*

Proof. In the first case, we set the simulator as C_2^g. In the second case, we set the simulator as C_1^f. By Lemma 5.5 (below), both worlds are perfectly indistinguishable, even given unlimited queries. Moreover, the simulator can be implemented such that it answers a query in $O(l)$ time, where l is an upper bound on the block length of a query, by Lemma 5.6. □

Lemma 5.5. *Let $f : (\{0,1\}^r)^* \to \{0,1\}^r$ be a random function, which defines a corresponding* fix *function. Then, the truth table of $f \circ \mathsf{fix}$ cannot be distinguished from f with any advantage. In other words, $f \circ \mathsf{fix}$ is also a uniform random function, though one dependent on f.*

Proof. To see this, consider sampling the output values of f in order of increasing input length. The fix of the empty string and r-length strings is the identity, so we clearly can sample i.i.d. strings from $\{0,1\}^r$ for images of such strings. Now inductively consider selecting the images of $\{0,1\}^{kr}$, where f has been defined for all $\{0,1\}^{<kr}$. The value of f on such strings suffices to define the fix of strings $\{0,1\}^{kr}$. Further, fix is a bijective function. The composition of a random function with a bijection on it's domain is still a random function, so we may sample each image of $\{0,1\}^{kr}$ as an i.i.d. string in $\{0,1\}^r$. □

Lemma 5.6. *Let $f : (\{0,1\}^r)^* \to \{0,1\}^r$ be a random function, which defines a corresponding* fix *function. Then, given query access to $g = f \circ \mathsf{fix}$, one can implement an l block query to f using $O(l)$ queries to g.*

Proof. First note that, for single block inputs the fix operator is the identity, so a call to f is a call to g. Furthermore, the input to g is then fix^{-1} on the original input (trivially, because the fix is the identity). This forms our base case. We will also need the following characterization of fix^{-1}.

Claim 5.7. *We can characterize fix^{-1} recursively as follows.*

(1) The inverse fix of the empty string ϵ is

$$\mathsf{fix}^{-1}(\epsilon) = \epsilon$$

(2) The inverse fix of a one-block string x_1 is

$$\mathsf{fix}^{-1}(x_1) = x_1$$

(3) *The inverse fix of a multi-block string* $t\|x_m$*, where* t *is a non-empty string and* $x_m \in \{0,1\}^r$*, is*

$$\mathsf{fix}^{-1}(t\|x_m) = \mathsf{fix}^{-1}(t)\|x_m \oplus f(t).$$

Inductively, let $u_1\|\ldots\|u_l = \mathsf{fix}^{-1}(x_1\|\ldots\|x_l)$, and consider an input $x_1\|\ldots\|x_l$. Let y be the output register. By assumption, we can use $l-1$ queries to g to compute the value of $f \circ \mathsf{fix}$ on input $x_1\|\ldots\|x_{l-1}$ into the register x_l, while simultaneously computing the fix of $x_1\|\ldots\|x_{l-1}$. We are then left with the strings $u_1\|\ldots\|u_{l-1}\|x_l \oplus f(x_1\|\ldots\|x_{l-1})\|y = \mathsf{fix}^{-1}(x_1\|\ldots\|x_l)\|y$. We can then call the oracle for $g = f \circ \mathsf{fix}$ a single time on the whole input register to obtain the string $\mathsf{fix}^{-1}(x_1\|\ldots\|x_l)\|y \oplus f(x)$. This procedure uses l calls exactly, and we can uncompute the change to the input with $2l$ calls. □

With this in place, by Lemma 3.13 it will suffice to prove that the Msponge is indifferentiable from a random oracle.

5.3 Msponge terminology and combinatorics

Consider the Msponge construction MSp^φ, where we view the underlying permutation φ in the alternative form described in Section 5.1. In this view, φ is constructed from three random functions h, k, k' and a random permutation π as given in Equation (12).

We will now define various properties of compressed oracle databases $D_k, D_{k'}$ corresponding to the functions k, k' above. These properties will depend on the permutation π, and will implicitly refer to the structure of the Msponge construction MSp^φ. The properties we define will not depend on h.

We will always assume that π is satisfies the following desirable property.

Definition 5.8. *We say that permutation* π *is "good" if for any* $x_1, x_2 \in \{0,1\}^r$*, the number of suffixes* $z \in \{0,1\}^c$ *such that* $\pi(x_1\|z)$ *begins with* x_2 *is at most* $O(n + 2^{c-r})$*.*

It follows from Lemma B.3 that a fraction $1 - O(2^{-n})$ of permutations are good, validating this assumption.

Definition 5.9. *Let* $z \in \{0,1\}^c$*, and fix* $D_k, D_{k'}$ *and* π*. We say that* z *has a "tail" under the following recursive conditions.*

(1) $z = 0^c$ *has a tail.*

(2) z *has a tail if it can be reached as an internal state by a single round of absorption from a state* z_p *which has a tail. That is, there exist* $x_p \in D_k$*,* $x_i \in D_{k'}$*, a* $z_p \in \{0,1\}^c$ *with a tail, and a* $z_i \in \{0,1\}^c$ *such that*

$$\begin{aligned} x_i\|z_i &= \pi(x_p\|z_p \oplus D_k(x_p)) \\ z &= z_i \oplus D_{k'}(x_i). \end{aligned}$$

A tail of z *is a string of inputs required to reach* z *according to the above conditions. Specifically, in case (1) the empty string is the unique tail of* $z = 0^c$*, and in case (2), any tail of* z_p *concatenated with* x_p *is a tail of* z*. We denote by* $\mathsf{tail}(z)$ *the set of tails of* z*.*

One can think of the set of internal states z with a tail as those that an adversary "knows how to reach". We define $\mathsf{tail}_1(z)$ to be the lexicographically first tail of z, and will often deal with databases where tails are unique. We define the "head" of a tail of z to be empty in case (1) above, and x_i in case (2) above. We set $\mathsf{head}(z) = \mathsf{head}(\mathsf{tail}_1(z))$. Observe that both of these notions have an implicit dependence on $D_k, D_{k'}$ which we suppress here for notational convenience.

The definition here does not depend on the database for h. The reason for this is clearer to see in the Msponge: there, every call to h during an internal round is immediately discarded, as the next input takes its place, and the final call to h does not affect the state wire. In this way, the reachable states are decoupled from h.

We will also sometimes use this definition in the standard sponge, e.g. in Section 6. This definition still makes sense in that context, because an adversary that knows an input reaching a z value could easily recover the tail of z, by evaluating the fix^{-1} function using the sponge, and in the other direction an adversary knowing a tail of z could evaluate fix to recover an input reaching z.

We also require the notion of an intermediate pair. These correspond to the states an adversary "knows how to reach" in the intermediate stage of applying the block function, after applying π and before applying k'.

Definition 5.10. *Fix a database pair $(D_k, D_{k'})$. We say a pair (x, z) is an intermediate pair if there exists $x_p \in D_k$ and a $z_p \in \{0,1\}^c$ with a tail such that*

$$x \| z = \pi(x_p \| z_p \oplus k(x_p)) \,.$$

We let $\mathsf{IP}(D_k, D_{k'})$ denote the set of all intermediate pairs.

Note that because π is a permutation, there is no multiplicity for intermediate pairs. We can now define the set of good databases.

Definition 5.11. *A database pair $(D_k, D_{k'})$ is **good** if every $z \in \{0,1\}^c$ has at most one tail, and $\mathsf{IP}(D_k, D_{k'})$ has unique x-values. Formally, we require both of the following conditions:*

$$\forall z \in \{0,1\}^c,\ |\mathsf{tail}(z)| \leq 1$$
$$\forall (x_1, z_1), (x_2, z_2) \in \mathsf{IP}(D_k, D_{k'}),\ (x_1, z_1) \neq (x_2, z_2) \Leftrightarrow x_1 \neq x_2$$

Note that the empty databases are clearly good, because the only state value with a tail is 0^c and the set of intermediate states is empty. A useful fact about good databases is that the number of state values with a tail is bounded by the number of queries, and the number of intermediate pairs is bounded by the number of queries squared.

Lemma 5.12. *Fix databases $(D_k, D_{k'})$ which are good and have at most t non-$\perp$ inputs each. Then there are $O(t)$ values with a tail and $O(t^2)$ intermediate pairs. Formally, we have $|\{z : |\mathsf{tail}(z)| \geq 1\}| \leq O(t)$, and $|\mathsf{IP}(D_k, D_{k'})| = O(t^2)$.*

Proof. Each $z \neq 0^c$ value with a tail has a non-empty one, which means there exists a unique $(x_i, z_i) \in \mathsf{IP}(D_k^t, D_{k'}^t)$ such that $z = z_i \oplus k'(x_i)$. In this way, each $z \neq 0^c$ with a tail corresponds to a unique intermediate pair, for which $x_i \in D_{k'}$. In a good database the x_i's are distinct for all distinct intermediate pairs, and we have $|D_{k'}| \leq t$, so there are at most t such pairs. Note that we excluded 0^c from this argument, so we have at most $t + 1$ values of z with a tail.

The set of intermediate pairs is constructed directly from z values with a tail, and values of x such that $x \in D_k$. There are at most $O(t)$ of each, so there are $O(t^2)$ pairs of each, and hence $|\mathsf{IP}(D_k^t, D_{k'}^t)| = O(t^2)$. □

Towards bounding the transitions from good to bad, we will need the notion of a bad attach. This is the "bad event" for a query to k.

Definition 5.13. *Fix good databases $(D_k, D_{k'})$, and fix input $x \in \{0,1\}^r$ s.t. $x \notin D_k$. We say that image y causes a bad attach if either there is a collision in intermediate pairs, or if the newly created intermediate pairs attach to a value in $D_{k'}$. Formally, for an output value y, define the intermediate pairs of (x, y) as*

$$\mathsf{IP}_{(x,y)}(D_k, D_{k'}) = \{(x_i, z_i) \,:\, \exists z \in \{0,1\}^c \text{ s.t. } |\mathsf{tail}(z)| \geq 1, x_i \| z_i = \pi(x \| z \oplus y)\}.$$

We say that y causes a bad attach (on preimage x) if either of the following hold:

(1) There exists $(x_1, z_1) \in \mathsf{IP}_{(x,y)}(D_k, D_{k'})$ and $(x_2, z_2) \in \mathsf{IP}(D_k, D_{k'})$ such that $x_1 = x_2$.

(2) There exists $(x_1, z_1) \in \mathsf{IP}_{(x,y)}(D_k, D_{k'})$ such that $x_1 \in D_{k'}$.

Lemma 5.14. *Fix databases $(D_k, D_{k'})$ which are good and have at most t non-$\perp$ inputs each, and fix input $x \in \{0,1\}^r$ s.t. $x \notin D_k$. Let B be the set of images which causes a bad attach on preimage x. Then we have $|B| \leq O(t^3 n + t^3 2^{c-r})$ when π is good.*

Proof. Let us denote by $X_{bad} \subset \{0,1\}^r$ the set of $x \in \{0,1\}^r$ such that either x is the first half of an intermediate pair in $\mathsf{IP}(D_k, D_{k'})$, or $x \in D_{k'}$, or both. Note that $|X_{bad}| = O(t^2)$ by Lemma 5.12. Let us consider some $z \in \{0,1\}^c$ which has a tail, and $x_i \in X_{bad}$. We will use $N_{x_i,z}$ to count the number of images y such that the value

$$\pi(x \| y \oplus z)|_0^r = x_i,$$

i.e. we obtain a bad attach corresponding to (x_i, z). If y causes a bad attach, then it causes it for some (x_i, z). We then have

$$\begin{aligned} |B| \leq & \sum_{z, \mathsf{tail}(z) \neq \emptyset} \sum_{x_i \in X_{bad}} N_{x_i,z} \\ \leq & O(t^3 n + t^3 2^{c-r}) \qquad (\pi \text{ good}) \end{aligned}$$

□

The other case we must consider is a query to k'. Here, we will refer to a "bad completion" event.

Definition 5.15. *Fix good databases $(D_k, D_{k'})$ which have at most t non-$\perp$ inputs each, and fix input $x' \in \{0,1\}^r$ s.t. $x' \notin D_{k'}$. We say that image y' causes a bad completion if x' appears in an intermediate pair (x', y'_i), and further there is either a created collision in intermediate pairs or state values with a tail, or if the newly created intermediate pairs attach to a value in $D_{k'}$. Formally, for an output value y', define the completed pairs of (x', y') as*

$$\mathsf{CP}_{(x',y')}(D_k, D_{k'}) = \begin{cases} \{(x_i, z_i) \,:\, \exists x \in D_k \text{ s.t. } x_i \| z_i = \pi(x \| y'_i \oplus y' \oplus D_k(x))\} & \textit{(If } \exists (x', y'_i) \in \mathsf{IP}(D_k, D_{k'})\textit{)} \\ \emptyset & \textit{(Otherwise)} \end{cases}$$

We say that y causes a bad attach (on preimage x) if either of the following hold:

(1) *There exists* $(x_1, z_1) \in \mathsf{CP}_{(x',y')}(D_k, D_{k'})$ *and* $(x_2, z_2) \in \mathsf{IP}(D_k, D_{k'})$ *such that* $x_1 = x_2$.

(2) *There exists* $(x_1, z_1) \in \mathsf{CP}_{(x',y')}(D_k, D_{k'})$ *such that* $x_1 \in D_{k'}$.

(3) *There exists* $(x', y'_i) \in \mathsf{IP}(D_k, D_{k'})$ *s.t.* $\mathsf{tail}(y'_i \oplus y') \neq \emptyset$ *(where here the tail is computed before assigning* $[x' \to y']$*).*

Lemma 5.16. *Fix databases* $(D_k, D_{k'})$ *which are good and have at most* t *non-*$\perp$ *inputs each, and fix input* $x' \in \{0,1\}^r$ *s.t.* $x' \notin D_{k'}$*. Let* B *be the set of images which causes a bad completion on preimage* x'*. Then we have* $|B| \leq O(t^3 n + t^3 2^{c-r})$ *when* π *is good.*

Proof. If there is not a $z'_i \in \{0,1\}^c$ such that $(x', z'_i) \in \mathsf{IP}(D^t_k, D^t_{k'})$, then the claim trivially holds as the completed pairs set is empty and no new value has a tail. If there is such a $z'_i \in \{0,1\}^c$, then the value $z'_i \oplus y'$ will now have a tail on image y'. This may create a bad completion if this value already has a tail, of which there are $O(t)$ possible values by Lemma 5.12. This gives $O(t)$ values of y' that are in B. For the values of y' where this does not happen, we then analyze the completed set.

As in the proof of Lemma 5.14, let us denote by $X_{bad} \subset \{0,1\}^r$ the set of $x \in \{0,1\}^r$ such that either x is the first half of an intermediate pair in $\mathsf{IP}(D_k, D_{k'})$, or $x \in D_{k'}$, or both. Note that $|X_{bad}| = O(t^2)$ by Lemma 5.12. Let us consider some $x \in D_k$ and $x_i \in X_{bad}$. We will use $N_{x_i,x}$ to count the number of images y' such that the value

$$\pi(x \| z'_i \oplus y' \oplus D_k(x))|_0^r = x_i,$$

i.e. we obtain a bad completion. If y' causes a bad completion, then it causes it for some (x_i, x). We then have

$$\begin{aligned} |B| \leq & \sum_{x \in D_k} \sum_{x_i \in X_{bad}} N_{x_i,x} \\ \leq & O(t^3 n + t^3 2^{c-r}) \qquad (\pi \text{ good}) \end{aligned}$$

□

Putting these results together, we can construct a local property which will enable us to bound the transition to a bad database from a good one.

Definition 5.17. *Fix databases* $D = (D_k, D_{k'})$*, and let* x *be an input to either* k *s.t.* $x \notin D_k$*, or an input to* k' *s.t.* $x \notin D_{k'}$*. Define the local property*

$$\mathsf{L}_g^{x,D} = \begin{cases} \{(D'_k[x \to y], D'_{k'}) : (D'_k, D'_{k'}) \in (D_k|_x, D_{k'}), & y \in \{0,1\}^c \cup \{\perp\} \text{ and } D'_k[x \to y], D'_{k'} \text{ is good}\} \\ & \text{(if } x \text{ input to } k) \\ \{(D'_k, D'_{k'}[x \to y]) : (D'_k, D'_{k'}) \in (D_k|_x, D_{k'}), & y \in \{0,1\}^c \cup \{\perp\} \text{ and } D'_k, D'_{k'}[x \to y] \text{ is good}\} \\ & \text{(if } x \text{ input to } k') \end{cases}$$

and let L_g *be the family of such local properties; in other words* L_g *is the family of local properties which recognize good databases. We use the notation* L_g^t *to denote the restriction of* L_g *to databases with at most* t *inputs each.*

Lemma 5.18. *The local property* L_g^t *satisfies* $\Delta(\mathsf{L}_g^t) \leq O(nt^3 + 2^{c-r}t^3)$.

Proof. It suffices to bound the distance for any of the constituent $\mathsf{L}_g^{x,D,t}$. Observe that if $D[x \to \perp]$ (where x is either an input to k or k') is bad, then the property becomes trivial as the bad predicate is monotone, and $\Delta(\mathsf{L}_g^{x,D,t}) = 0$. We therefore assume $D[x \to \perp]$ is good. Let us split into cases.

(1) Suppose that x is an input to D_k. In order for the database to become bad by assigning a value to x, the image of x must cause a collision in the intermediate pair prefixes, or result in a new intermediate pair whose prefix already lies in $D_{k'}$. In other words, the image y must cause a bad attach. By Lemma 5.14, there are $O(nt^3 + 2^{c-r}t^3)$, which implies

$$\Delta(\mathsf{L}_g^{x,D,t}) \leq O(nt^3 + 2^{c-r}t^3).$$

(2) Suppose that x is an input to $D_{k'}$. In order for the database to become bad by assigning a value to x, the image of x must cause a collision in the intermediate pair prefixes or tail values, or result in a new intermediate pair whose prefix already lies in $D_{k'}$. In other words, the image y must cause a bad completion. By Lemma 5.16, there are $O(nt^3 + 2^{c-r}t^3)$, which implies

$$\Delta(\mathsf{L}_g^{x,D,t}) \leq O(nt^3 + 2^{c-r}t^3).$$

□

Finally, the following lemma will be helpful in our indifferentiability proof, and is based on a similar analysis. It essentially states that, for a fixed input x and state value z, the number of images of x which cause the tail of z to be different from when the image of x is $\perp$ is small.

Lemma 5.19. *Fix databases $D = (D_k, D_{k'})$ which are good, let x be an input to either k s.t. $x \notin D_k$, or an input to k' s.t. $x \notin D_{k'}$, and let $z \in \{0,1\}^c$ be some possible state. Denote by $\mathsf{tail}_1(z)$ the lexicographically first tail of z in D (or a failure symbol $\perp$ if z has no tail), and $\mathsf{tail}_{1,[x\to y]}(z)$ as the lexicographically first tail of z in $D[x \to y]$ (or again a failure symbol $\perp$ if z has no tail). Then we have*

$$|\{y : \mathsf{tail}_{1,[x\to y]}(z) \neq \mathsf{tail}_1(z)\}| \leq O(nt^3 + 2^{c-r}t^3).$$

Proof. The case where $\mathsf{tail}_1(z) \neq \emptyset$ is trivial, as it means that z has a tail in D_k and $D_{k'}$. This tail will remain valid if a new tail of z involving x is created, which must happen if $\mathsf{tail}_{1,[x\to y]}(z) \neq \mathsf{tail}_1(z)$. There is then a z value with multiple tails, and so the database is bad; the claim now follows from Lemma 5.18. Let us instead focus on the case where $\mathsf{tail}_1(z) = t \neq \emptyset$. We have two cases.

(Case 1) x is an input to k. The only image assignments to x which could possibly involve x in a tail are ones where the prefix of one of the resultant intermediate pairs appears in $D_{k'}$, which is a bad attach. From Lemma 5.14 there are $O(nt^3 + t^3 2^{c-r})$ values.

(Case 2) x is an input to k'. In the case where there is no z' such that $(x, z') \in \mathsf{IP}(D_k, D_{k'})$, no new tail is created for any image assignment, so the claim holds trivially. Suppose the opposite, and let us exclude the values of y which lead to a bad completion; by Lemma 5.16 there are $O(nt^3 + t^3 2^{c-r})$ values. There is then at most one new value of z with a tail (recalling that intermediate pair prefixes are unique in a good database), which in particular has value $z' \oplus y$. This can only change the tail of z if $y = z' \oplus z$, which is a single value.

□

6 Query lower bound proofs

In this section, we derive bounds for collision resistance and preimage resistance of the sponge construction. We start by re-deriving classical lower bounds for these problems within our framework, using lazy sampling arguments. We then use the tools developed along the way to give quantum query lower bounds (in Section 6.2) via the framework of [Zha19; Chu+21]. We will show lower bounds for preimage and collision resistance in the Msponge, which implies via black-box reduction preimage and collision resistance of the standard sponge.

6.1 Classical lower bounds

Consider a classical adversary with query access to φ and φ^{-1}. The probability that this adversary creates a bad database with fewer than q queries is upper bounded by Lemma 5.18. Now consider the probability that this same adversary outputs a collision in Sp^{φ}, and separate this event into two disjoint events: one where the database is good, and one where it is bad. We derive a bound for the probability of outputting a collision conditioned on the database being good, as this considerably simplifies the analysis. We take a similar approach to preimage finding. One downside of this strategy is that it does not yield a tight bound for either of the considered problems.

We begin by defining a notion of reachable outputs given some database $D = \{D_k, D_{k'}, D_h\}$ and permutation π. Under this notion, an output $y \in \{0,1\}^r$ is reachable if there's a collection of input-output pairs in the databases D that, when put together with π in the manner defined by the Msponge, would yield y as an output. In particular, this implies that D determines an input m such that $\mathsf{MSp}(m) = y$. Note that, if an output y is reachable in this sense, then one can use D and block-length-many queries to h to construct another input m' such that $y = \mathsf{Sp}^{\varphi}(m')$.

Definition 6.1 (reachable output)**.** *Let $y \in \{0,1\}^r$. We say that y is a reachable output of Sp^{φ} in D if there exists $z \in \{0,1\}^c$ such that*

(1) $(z, h(z)) \in D_h$,

(2) There exists a tail x of z such that $y = \mathsf{head}(x) \oplus h(z)$.

We refer to the tail-capacity pair (x,z) as a path. We say that the path (x,z) reaches y.

Condition (2) implicitly means that every k and k' query appearing when we feed Sp^{φ} with input $\mathsf{fix}(x)$ in the framework of World 2 must be in the databases D_k and $D_{k'}$, respectively. A path is an analog of input for Sp^{φ}. Also, note that the tail x can appear in the path of one output at most, but an output might be reachable via several paths each having a distinct tail x.

Lemma 6.2. *Let $\mathcal{A}^{\varphi,\varphi^{-1}}$ be a (classical or quantum) q-query algorithm outputting colliding inputs $m, m' \in (\{0,1\}^r)^{\leq l}$ for Sp^{φ} with probability ϵ. Then there exists a $(3q+6l)$-query algorithm $\mathcal{B}^{h,k,k'}(\pi)$ outputting colliding paths s, s' for $\mathsf{Sp}^{hk'\pi k}$ with the same probability, i.e.,*

$$\Pr_{m,m' \leftarrow \mathcal{A}^{\varphi,\varphi^{-1}}}[\mathsf{Sp}^{\varphi}(m) = \mathsf{Sp}^{\varphi}(m')] = \Pr_{s,s' \leftarrow \mathcal{B}^{h,k,k'}(\pi)}[s, s' \text{ reach the same output}]. \tag{15}$$

The probabilities in Equation (15) are taken over the appropriate sampling of the oracles, i.e., uniformly random φ for the left-hand side and uniformly random h, k, k', π for the right-hand side.

Proof. The algorithm $\mathcal{B}$ will begin by running $\mathcal{A}$ as a subroutine, simulating the oracles φ, φ^{-1} using h, k', k, π until $\mathcal{A}$ terminates and outputs m, m'. Let $y := \mathsf{Sp}^{hk'\pi k}(m)$ and $y' := \mathsf{Sp}^{hk'\pi k}(m')$. By the indifferentiability result in Lemma 5.1, the probability that $y = y'$ after this simulated run of $\mathcal{A}$ is exactly equal to

$$\Pr_{m,m' \leftarrow \mathcal{A}^{\varphi,\varphi^{-1}}}[\mathsf{Sp}^{\varphi}(m) = \mathsf{Sp}^{\varphi}(m')] \tag{16}$$

where φ is sampled uniformly at random.

The above simulation of $\mathcal{A}$ by $\mathcal{B}$ works as follows. When $\mathcal{A}$ queries on x, $\mathcal{B}$ queries on

1. $x|_0^r$ to k,
2. $x' = \pi(x|_0^r||(x|_r^n \oplus k(x|_0^r)))|_0^r$ to k', and
3. $z = \pi(x|_0^r||(x|_r^n \oplus k(x|_0^r)))_r^n \oplus k'(x')$ to h,

then returns $(x' \oplus h(z))||z$. Each query of $\mathcal{A}$ costs 3 queries for $\mathcal{B}$. After q queries, $\mathcal{A}$ terminates and outputs m, m'.

Next, algorithm $\mathcal{B}$ makes the necessary queries to form the paths s, s' that reach y, y' (respectively) with the following procedure. We will WLOG write the set of necessary queries for the case of m explicitly, assuming the block size of m is bounded by l. The relevant query input-output set is then

$$\{(x_i, k(x_i)), (x_i', k'(x_i')), (z_i, h(z_i))\}_{i \in \{1,\ldots,l\}} \tag{17}$$

such that

$$\begin{array}{lll}
x_1 := m|_0^r & x_1' := \pi(x_1||k(x_1))|_0^r & z_1 := \pi(x_1||k(x_1))|_r^n \oplus k'(x_1') \\
x_2 := m|_r^{2r} \oplus x_1' \oplus h(z_1) & x_2' := \pi(x_2||k(x_2) \oplus z_1)|_0^r & z_2 := \pi(x_2||k(x_2))|_r^n \oplus k'(x_2') \\
\vdots & \vdots & \vdots \\
x_l := m|_{(l-1)r}^{lr} \oplus x_{l-1}' \oplus h(z_{l-1}') & x_l' := \pi(x_l||k(x_l) \oplus z_{l-1})|_0^r & z_l := \pi(x_l||k(x_l)|_r^n \oplus k'(x_l').
\end{array}$$

This procedure costs at most $6l$ many queries for $\mathcal{B}$. Note that these $6l$ queries made by $\mathcal{B}$ are classical even if the queries of $\mathcal{A}$ to its oracle are quantum. The path that reaches y becomes $s = (x_1|| \cdots ||x_l, z_l)$. The path s' that reaches y' is defined similarly. $\mathcal{B}$ completes by outputting s, s' and terminating.

By construction, s and s' reach the same output if and only if $y = y'$. As observed above, the latter occurs with probability exactly equal to Equation (16). □

Remark 6.3. *The proof of Lemma 6.2 shows that the success probability of any classical q-query adversary $\mathcal{A}^{\varphi,\varphi^{-1}}$ outputting a collision is equal to the probability that, after the classical $3q + 6l$-query adversary $\mathcal{B}^{h,k,k'}(\pi)$ finishes, the database contains a collision.*

It's straightforward to adapt the construction of $\mathcal{B}$ in the proof of Lemma 6.2 so that, if $\mathcal{A}$ is some algorithm that outputs only a single m, then the result is a single path s that reaches the output of the sponge evaluated on m. This yields the following.

Lemma 6.4. *Let $\mathcal{A}^{\varphi,\varphi^{-1}}$ be a (classical or quantum) q-query algorithm that, given $y \sim \{0,1\}^r$, outputs $m \in (\{0,1\}^r)^{\leq l}$ such that $y = \mathsf{Sp}^{\varphi}(m)$ with probability ϵ. Then there*

exists a $(3q+3l)$*-query algorithm* $\mathcal{B}^{h,k,k'}(\pi)$ *that, given* $y \sim \{0,1\}^r$*, outputs a path* s *that reaches* y *for* $\mathsf{Sp}^{hk'\pi k}$ *with the same probability, i.e.,*

$$\Pr_{\substack{y\sim\{0,1\}^r \\ m'\leftarrow\mathcal{A}^{\varphi,\varphi^{-1}}}}[y=\mathsf{Sp}^{\varphi}(m')] = \Pr_{\substack{y\sim\{0,1\}^r \\ s\leftarrow\mathcal{B}^{h,k,k'}(\pi)}}[s \text{ reaches } y]$$

When we consider the framework of World 2, we will refer to the existence of paths reaching the same output as a collision. We will derive the collision resistance bound for Sp^{φ} in Theorem 6.8 by combining Lemma 6.2 and the bound for the database containing a collision as in Lemma 6.6.

Definition 6.5. *We say that there is a collision in the database (in the framework of World 2) if at least two paths reach the same output.*

Lemma 6.6 (collision in the database)**.** *Given classical query access to* h, k, k'*, the probability of the database containing a collision with bounded block length* l *after* q *queries is bounded by* $O(q^4 n 2^{-\min(r,c)})$*. Consequently, for constant success probability, the number of queries required is* $\Omega(\sqrt[4]{2^{\min(r,c)}/n})$*.*

Proof. Let $D_k, D_{k'}, D_h$, denoted by D, be a set of databases recording queries to functions k, k', h, respectively. Define the event

$$\mathsf{C}_t := \exists \text{ a collision in } D^t$$

where D^t is a database such that the total number of defined input/output points is at most t. Then, $\Pr[\mathsf{C}_q]$ denotes the probability of a database containing a collision after q queries. We separate the event into the cases that database is *good*, as in Definition 5.11, and *bad* as follows

$$\Pr[\mathsf{C}_q] = \Pr[\mathsf{C}_q \cap D^{q-1} \text{ good}] + \Pr[\mathsf{C}_q \cap D^{q-1} \text{ bad}]. \tag{18}$$

We will work with good databases after bounding the second term in Equation (18) by

$$\begin{aligned}\Pr[\mathsf{C}_q \cap D^{q-1} \text{ bad}] &\leq \Pr[D^q \text{ bad}] \\ &\leq \sum_{t=1}^{q} \Pr[D^t \text{ bad} | D^{t-1} \text{ good}] + \Pr[D^t \text{ bad} | D^{t-1} \text{ good}] \\ &\leq O(nq^4 2^{-\min(r,c)})\end{aligned}$$

where the last inequality follows from Lemma 5.18. Observe that D^t good implies D^{t-1} good and C_t implies C_{t-1}. Using these, we can bound the first term in Equation (18) by

$$\begin{aligned}\Pr[\mathsf{C}_q \cap D^{q-1} \text{ good}] &= \Pr[\mathsf{C}_{q-1} \cap \mathsf{C}_q \cap D^{q-1} \text{ good}] + \Pr[\neg\mathsf{C}_{q-1} \cap \mathsf{C}_q \cap D^{q-1} \text{ good}] \\ &\leq \Pr[\mathsf{C}_{q-1} \cap D^{q-2} \text{ good}] + \Pr[\mathsf{C}_q | \neg\mathsf{C}_{q-1}, D^{q-1} \text{ good}] \\ &\leq \sum_{t=1}^{q} \Pr[\mathsf{C}_t | \neg\mathsf{C}_{t-1}, D^{t-1} \text{ good}]\end{aligned}$$

where the second inequality followed from the observation and the third from repeatedly applying the procedure in the first two lines.

Notice that adding a collision to the database with a single query, assuming there is none, requires adding a new path. This can be done by either

- querying to k or k' to create a new tail $x \in (\{0,1\}^r)^*$ for some $z \in \{0,1\}^c$, or
- querying to h on $z \in \{0,1\}^c$ with $\mathsf{tail}(z) \neq \emptyset$.

These are just necessary, but not sufficient, conditions. In the t-th query, we have three options: the adversary can query to the function k, k', or h. We consider these events separately; the event "f query" means the queried function is $f \in \{k, k', h\}$.

(1) A query to k on input x. If a collision occurs after this query, a new path, in particular a new tail, must have been created in a round starting with the state $x||z$ where $z \in \{0,1\}^c$ and $\mathsf{tail}(z) \neq \emptyset$. This means the k' query in this round must be in the database, i.e., $\pi(x||k(x) \oplus z)|_0^r \in D_{k'}^{t-1}$. Then, with the union bound and the fact that $|\mathsf{tail}(z)| \leq O(t)$ in a good database, we get

$$\begin{aligned}\Pr[\mathsf{C}_t | k \text{ query}, \neg E_{t-1}, D^{t-1} \text{ good}] &\leq \sum_{z:\mathsf{tail}(z)\neq\emptyset} \sum_{x' \in D_{k'}^{t-1}} \Pr_{k(x)}[\pi(x||k(x) \oplus z)|_0^r = x'] \\ &\leq \sum_{z:\mathsf{tail}(z)\neq\emptyset} \sum_{x' \in D_{k'}^{t-1}} N_{x_i,z} 2^{-c} \\ &\leq O(t^2 2^{-r} + t^2 n 2^{-c})\end{aligned}$$

where $N_{x_i,z}$ is the number of values $k(x)$ such that $\pi(x||k(x) \oplus z)|_0^r = x'$.

(2) A query to k' on input x'. Similar to the previous one, creating a collision requires creating a new tail, which must have been created by querying the first entry of an element in the set of intermediate pairs IP. Since all the first entries in IP are distinct, we can extend only one tail. Say the input to be queried is $\pi(x||k(x) \oplus z)|_0^r = x'$ for some $z \in \{0,1\}^c$ with $\mathsf{tail}(z) \neq \emptyset$ and $(x, k(x)) \in D_k^{t-1}$. After this query, the value $\bar{z} = \pi(x||k(x) \oplus z)|_r^n \oplus k'(x')$ will have a tail. We can separate the collision event into disjoint events while skipping writing the conditions on the database for simplicity:

$$\Pr[\mathsf{C}_t] = \Pr[\mathsf{C}_t \cap \mathsf{tail}(\bar{z}) \neq \emptyset \text{ in } D^{t-1}] + \Pr[\mathsf{C}_t \cap \mathsf{tail}(\bar{z}) = \emptyset \text{ in } D^{t-1}]$$

If $\bar{z}$ has a tail before this query, then collision is inevitable. So,

$$\Pr[\mathsf{C}_t \cap \mathsf{tail}(\bar{z}) \neq \emptyset \text{ in } D^{t-1}] = \Pr[\mathsf{tail}(\bar{z}) \neq \emptyset \text{ in } D^{t-1}] \leq O(t 2^{-c})$$

since $\bar{z}$ is uniformly random in $\{0,1\}^c$ and number of z with a tail is $O(t)$ in D^{t-1}. If $\bar{z}$ does not have a tail before this query, then having a collision implies

1. $\bar{z} \in D_h^{t-1}$ and $x' \oplus h(\bar{z})$ collides with a reachable output via D^{t-1}, or
2. k'-query following a round ending with $\bar{z}$ is in $D_{k'}^{t-1}$.

Case 1 is upper bounded by $O(t2^{-c})$ due to the uniformity of $\bar{z}$. Case 2 can be upper bounded as

$$\begin{aligned}\sum_{\bar{x} \in D_k^{t-1}} \sum_{\bar{x}' \in D_{k'}^{t-1}} \Pr_{k'(x')}[\pi(\bar{x}||k(\bar{x}) \oplus \bar{z})|_0^r = \bar{x}'] &\leq \sum_{\bar{x} \in D_k^{t-1}} \sum_{\bar{x}' \in D_{k'}^{t-1}} N_{\bar{x}',\bar{z}} 2^{-c} \\ &\leq O(t^2 2^{-r} + t^2 n 2^{-c})\end{aligned}$$

where $N_{\bar{x}',\bar{z}}$ is the number of values $k'(x')$ such that $\pi(\bar{x}||k(\bar{x}) \oplus \bar{z})|_0^r = \bar{x}'$.

(3) A query to h on input z. Let R denote the set of reachable outputs via D^{t-1}, then $|R| \leq O(t)$ since D^{t-1} is good. A collision can be formed if the query z has a tail x in D^{t-1} and x, z constructs a path for an element in R. So,

$$\begin{aligned}\Pr[\mathsf{C}_t | h \text{ query}, \neg E_{t-1}, D^{t-1} \text{ good}] &\leq \sum_{y \in R} \Pr_{h(z)}[y = x'_z \oplus h(z)] \\ &\leq O(t 2^{-c})\end{aligned}$$

where x'_z head of the tail of z.

□

This is our main theorem for the classical collision resistance of the sponge construction. This will be the main ingredient while proving the quantum collision resistance using quantum transition capacities.

The following result is obtained immediately from the proof of Lemma 6.6 since the bound for k, k' query cases are argued by the bound for creating a new tail, which is also required to create a path reaching some output conditioned on there is no path for this image. Also, the bound for the h query case is argued by querying a capacity value with a tail, which is again necessary to create a path reaching some output. The success probability of the h query case is less than the one in Lemma 6.6.

Corollary 6.7. *Given $y \sim \{0,1\}^r$ and classical query access to h, k, k', the probability of the database containing m with bounded block length l such that $y = \mathsf{Sp}^{\varphi}(m)$ after q queries is bounded by $O(q^4 n 2^{-\min(r,c)})$. Consequently, for constant success probability, the number of queries required is $\Omega(\sqrt[4]{2^{\min(r,c)}/n})$.*

Theorem 6.8 (classical collision resistance)**.** *Given classical query access to φ and φ^{-1}, the probability of q-query adversary outputting a collision, i.e., an $m, m' \in (\{0,1\}^r)^{\leq l}$ for some $l \leq q$ such that $m \neq m'$ and $\mathsf{Sp}^{\varphi}(m) = \mathsf{Sp}^{\varphi}(m')$ is bounded by*

$$\Pr_{m, m' \leftarrow \mathcal{A}^{\varphi, \varphi^{-1}}}[\mathsf{Sp}^{\varphi}(m) = \mathsf{Sp}^{\varphi}(m')] \leq O(q^4 n 2^{-\min(r,c)}).$$

Consequently, for constant success probability, the number of queries required is $\Omega(\sqrt[4]{2^{\min(r,c)}/n})$.

Proof. We will consider adversaries in the framework of World 2, which by Lemma 5.1 suffices. By Lemma 6.2, for each adversary $\mathcal{A}^{\varphi,\varphi^{-1}}$, there exists an adversary $\mathcal{B}^{h,k,k'}$ such that the probabilities of $\mathcal{A}$ outputting a colliding inputs with q queries and $\mathcal{B}$ outputting colliding paths with $3q + 6l$ queries are the same. Moreover, as $l \leq q$ by assumption, and using Lemma 6.6, we get the desired bound. □

Corollary 6.9. *Given $y \sim \{0,1\}^r$ and classical query access to φ and φ^{-1}, the probability of q-query adversary outputting $m \in (\{0,1\}^r)^{\leq l}$ such that $y = \mathsf{Sp}^{\varphi}(m)$ and some $l \leq q$, namely*

$$\Pr_{\substack{y \sim \{0,1\}^r \\ m' \leftarrow \mathcal{A}^{\varphi,\varphi^{-1}}}}[y = Sp^{\varphi}(m')]$$

is bounded by $O(q^4 n 2^{-\min(r,c)})$. Consequently, for constant success probability, the number of queries required is $\Omega(\sqrt[4]{2^{\min(r,c)}/n})$.

6.2 Quantum lower bounds

With the result in Theorem 6.8 and the notion of quantum transition capacities, we can prove the quantum collision resistance of the sponge construction.

By definition, the value $[\![\neg \mathsf{P} \xrightarrow{q} \mathsf{Q}]\!]$ is equal to the square-root of the maximal probability that the internal state of the compressed oracle, when supported only on databases $D \in \neg\mathsf{P}$, is measured to be in a database $D' \in \mathsf{Q}$ after a quantum query algorithm performs q sequential queries. The special case $[\![\emptyset \xrightarrow{q} \mathsf{Q}]\!]$ is the square-root of the maximal probability of D satisfying Q when D is obtained by measuring the internal state of the compressed oracle after the interaction with $\mathcal{A}$, maximized over all q-query quantum algorithms $\mathcal{A}$, i.e.,

$$[\![\emptyset \xrightarrow{q} \mathsf{Q}]\!] := \max_{\mathcal{A}} \sqrt{\Pr[D \in \mathsf{Q}]}. \tag{19}$$

Before deriving the quantum collision resistance proof, we need a bound for having a bad database after the oracle calls since we always separate the good and bad databases.

Theorem 6.10. *Define the following predicate*

$$\begin{aligned}\mathrm{BAD} = \Big\{ &\Big(D \in \mathfrak{D} | \exists z \in \{0,1\}^c : |\mathsf{tail}^D(z)| \geq 2\Big) \\ &\vee \Big(\exists (x_1, z_1), (x_2, z_2) \in \mathsf{IP}^D : x_1 = x_2, z_1 \neq z_2\Big) \Big\}\end{aligned}$$

where D in the superscripts indicates the database the notion is defined for. Then, the probability p' that algorithm $\mathcal{A}$ creates a bad database (i.e., applying the binary measurement defined by Π^{BAD} returns BAD*) after q queries to the compressed oracle is bounded as*

$$p' \leq [\![\emptyset \xrightarrow{q} \mathrm{BAD}]\!]^2 \leq O(q^5 n 2^{-\min(r,c)}).$$

Proof. By definition of the quantum transition capacity, we have $\sqrt{p'} \leq [\![\emptyset \xrightarrow{q} \mathrm{BAD}]\!]$. Then, using its properties, we get

$$\begin{aligned}[\![\emptyset \xrightarrow{q} \mathrm{BAD}]\!] &\leq [\![\emptyset \xrightarrow{q} \mathrm{BAD} \cup \neg D^q]\!] && \text{(Lemma 4.10)} \\ &\leq \sum_{t=1}^{q} [\![\mathrm{GOOD} \cap \mathsf{D}^{t-1} \to \mathrm{BAD} \cup \neg D^t]\!] && \text{(Lemma 4.9)} \\ &\leq \sum_{t=1}^{q} [\![\mathrm{GOOD} \cap \mathsf{D}^{t-1} \to \mathrm{BAD}]\!]\end{aligned}$$

where the last inequality follows from the fact that a single query cannot increase the size of the database by more than one.

We can bound $[\![\mathrm{GOOD} \cap D^{t-1} \to \mathrm{BAD}]\!]$ using the classical reasoning. By setting $\mathsf{L}^{x,D} := \mathrm{BAD}|_{D^{|x}}$ and applying Lemma 4.13, we get

$$[\![\mathrm{GOOD} \cap D^{t-1} \to \mathrm{BAD}]\!] \leq 4\sqrt{\Delta(\mathsf{L})/N}$$

where $\mathsf{L} := \bigcup_{x,D} \{\mathsf{L}^{x,D} : x \in \mathcal{X}, D \in \mathfrak{D}\}$.

Notice that L corresponds to L^t_g in Lemma 5.18, hence $\Delta(\mathsf{L}) \leq O(t^3 n + t^3 2^{c-r})$ and

$$\Delta(\mathsf{L})/N \leq O((t^3 n + t^3 2^{c-r})/2^c) \leq O\left(t^3 n 2^{-\min(r,c)}\right)$$

where 2^c in denominator appears because the non-trivial queries x for L are either k or k' queries and their output length is c. □

Theorem 6.11. *Define the following predicate*

$$\mathsf{C} = \{D \in \mathfrak{D} | \exists s, s' \in (\{0,1\}^{\leq rl}, \{0,1\}^c) : s \neq s' \text{ and } s, s' \text{ reaches the same output}\},$$

i.e., it is the collection of databases that contains the necessary queries that can construct at least two distinct paths reaching the same output. Then, the probability p' that algorithm $\mathcal{B}$ creates a database in C (i.e., applying the binary measurement defined by Π^{C} returns C) after q queries to the compressed oracle is bounded as

$$p' \leq [\![\emptyset \xrightarrow{q} \mathsf{C}]\!]^2 \leq O\left(q^5 n 2^{-\min(r,c)}\right).$$

Proof. The maximality in the definition of the quantum transition capacity implies

$$\sqrt{p'} \leq [\![\emptyset \xrightarrow{q} \mathsf{C}]\!].$$

Next, we will use the properties of the quantum transition capacity to upper bound this as follows

$$\begin{aligned}
[\![\emptyset \xrightarrow{q} \mathsf{C}]\!] &\leq [\![\emptyset \xrightarrow{q} \mathsf{C} \cup \mathrm{BAD} \cup \neg D^q]\!] && \text{(Lemma 4.10)} \\
&\leq \sum_{t=1}^{q} [\![\neg \mathsf{P}_{t-1} \to \mathsf{P}_t]\!] && \text{(Lemma 4.9)}
\end{aligned}$$

where $\mathsf{P}_0 = \neg\bot$ and $\mathsf{P}_t = \neg D^t \cup \mathsf{C} \cup \mathrm{BAD}$ for $t \in \{1, \ldots, q\}$. Moreover, to make it more similar to the classical proofs for finding a collision in the sponge and having a bad database, we will do

$$\begin{aligned}
[\![\neg \mathsf{P}_{t-1} \to \mathsf{P}_t]\!] &= [\![\neg \mathsf{P}_{t-1} \to \mathsf{C} \cup \mathrm{BAD}]\!] \\
&\leq [\![\neg \mathsf{P}_{t-1} \to \mathsf{C}]\!] + [\![\neg \mathsf{P}_{t-1} \to \mathrm{BAD}]\!] \\
&\leq [\![\neg \mathsf{P}_{t-1} \to \mathsf{C}]\!] + [\![D^{t-1} \cap \mathrm{GOOD} \to \mathrm{BAD}]\!]
\end{aligned}$$

where the first line follows from the fact that a single query cannot increase the size of the database by more than one and the rest follows from Lemma 4.10.

The latter term is already bounded as in Theorem 6.10. The former one, namely $[\![D^{t-1} \cap \neg\mathsf{C} \cap \mathrm{GOOD} \to \mathsf{C}]\!]$, will be bounded using the classical reasoning. Setting $\mathsf{L}^{x,D} := \mathsf{C}|_{D^{|x}}$, applying Lemma 4.13 and Theorem 6.8, we get

$$[\![D^{t-1} \cap \neg\mathsf{C} \cap \mathrm{GOOD} \to \mathsf{C}]\!] \leq \sqrt{O\left(t^3 n 2^{-\min(r,c)}\right)}.$$

Combining all these gives

$$\begin{aligned}\sqrt{p'} &\leq [\![\emptyset \xrightarrow{q} \mathrm{C}]\!] \\ &\leq \sum_{t=1}^{q} [\![D^{t-1} \cap \neg\mathrm{C} \cap \mathrm{GOOD} \to \mathrm{C}]\!] + [\![D^{t-1} \cap \mathrm{GOOD} \to \mathrm{BAD}]\!] \\ &\leq \sum_{t=1}^{q} \sqrt{O\left(t^3 n 2^{-\min(r,c)}\right)} \\ &\leq q\sqrt{O\left(q^3 n 2^{-\min(r,c)}\right)}.\end{aligned}$$

□

Corollary 6.12. *For $y \sim \{0,1\}^r$, define the following predicate*

$$\mathsf{P}_y = \{D \in \mathfrak{D} | \exists s \in (\{0,1\}^{\leq rl}, \{0,1\}^c) : s \text{ reaches the output } y\},$$

i.e., the set of databases that contains the necessary queries that can construct a path that reaches y. Then, the probability p' that algorithm $\mathcal{B}$ creates a database in P_y (i.e., applying the binary measurement defined by Π_y^{P} returns P_y) after q queries to the compressed oracle is bounded as

$$p' \leq [\![\emptyset \xrightarrow{q} \mathsf{P}_y]\!]^2 \leq O\left(q^5 n 2^{-\min(r,c)}\right).$$

Now, we can give the corollary that bounds the success probability of an actual sponge collision finding adversary, i.e., one that outputs $m, m' \in (\{0,1\}^r)^{\leq l}$ such that $\mathsf{Sp}^\varphi(m) = \mathsf{Sp}^\varphi(m')$.

Corollary 6.13 (quantum collision resistance)**.** *The probability that a quantum algorithm $\mathcal{A}$ with quantum query access to a random permutation $\varphi \in S_{\{0,1\}^{r+c}}$ and its inverse, making at most a total of q queries, returns $m, m' \in (\{0,1\}^r)^{\leq l}$ for $l \leq q$ such that $m \neq m'$ and $\mathsf{Sp}^\varphi(m) = \mathsf{Sp}^\varphi(m')$, can be upper bounded as*

$$\Pr_{\substack{\varphi \sim S_{\{0,1\}^{r+c}} \\ m,m' \leftarrow \mathcal{A}^{\varphi,\varphi^{-1}}}} [\mathsf{Sp}^\varphi(m) = \mathsf{Sp}^\varphi(m')] \leq O\left(q^5 n 2^{-\min(r,c)}\right).$$

Proof. For each such algorithm $\mathcal{A}$ construct an algorithm $\mathcal{B}$ with query complexity $O(q)$ that simulates φ, φ^{-1} to $\mathcal{A}$ as in Lemma 6.2 and outputs paths s, s' corresponding to the outputs m, m' of $\mathcal{A}$, respectively. Suppose m and m' are $\tilde{l} \leq l$ and $\tilde{l}' \leq l$ blocks, respectively. More explicitly, a path s is derived from the set of queries corresponding to m, namely

$$\{(x_i, k(x_i)), (x_i', k'(x_i')), (z_i, h(z_i))\}_{i \in \{1,\ldots,\tilde{l}\}}$$

as described in Lemma 6.2. Similarly, one can form another set of queries

$$\{(x_i, k(x_i)), (x_i', k'(x_i')), (z_i, h(z_i))\}_{i \in \{\tilde{l}+1,\ldots,\tilde{l}+\tilde{l}'\}}$$

that defines a path s' corresponding to m'. By Lemma 6.2, we get

$$\Pr_{m,m' \leftarrow \mathcal{A}^{\varphi,\varphi^{-1}}} [Sp^\varphi(m) = Sp^\varphi(m')] = \Pr_{s,s' \leftarrow \mathcal{B}^{h,k,k'}} [s, s' \text{ reaches the same output}]. \tag{20}$$

By taking the union of the two sets of queries forming the paths s and s', we form

$$\mathbf{x} = \{(x_i, k(x_i)), (x'_i, k'(x'_i)), (z_i, h(z_i))\}_{i\in\{1,\ldots,\tilde{l}+\tilde{l}'\}}.$$

If the set $\mathbf{x}$ has a size smaller than $6l$ we can complete it set to have size $6l$ by adding random input-output pairs. Let $\mathcal{B}'$ be the slightly modified algorithm that outputs that $\mathbf{x}$. We define the following relation[6]

$$R = \{\mathbf{x} \in \mathcal{X}^{6l} \times \mathcal{Y}^{6l} : \mathbf{x} \text{ contains the queries constructing at least two colliding paths}\}.$$

Note that

$$\Pr_{\mathbf{x}\leftarrow\mathcal{B}'^{h,k,k'}}[\mathbf{x} \in R] = \Pr_{s,s'\leftarrow\mathcal{B}^{h,k,k'}}[s, s' \text{ reaches the same output}] =: p.$$

Applying Corollary 4.15, we get

$$\sqrt{p} \leq \sqrt{p'} + \sqrt{\frac{6l}{N}}$$

with p' as defined in Corollary 4.15. We have the bound for p' as $O(q\sqrt{q^3 n 2^{-\min(r,c)}})$ by Theorem 6.11. Hence,

$$\Pr_{m,m'\leftarrow\mathcal{A}^{\varphi,\varphi^{-1}}}[\mathsf{Sp}^{\varphi}(m) = \mathsf{Sp}^{\varphi}(m')] \leq O(q^5 n 2^{-\min(r,c)}).$$

□

Corollary 6.14. *Given $y \sim \{0,1\}^r$, the probability that a quantum algorithm $\mathcal{A}$ with quantum query access to a random permutation $\varphi \in S_{\{0,1\}^{r+c}}$ and its inverse, making at most a total of q queries, returns $m \in (\{0,1\}^r)^{\leq l}$ for $l \leq q$ such that $y = \mathsf{Sp}^{\varphi}(m)$, can be upper bounded as*

$$\Pr_{\substack{y\sim\{0,1\}^r \\ m'\leftarrow\mathcal{A}^{\varphi,\varphi^{-1}}}}[y = \mathsf{Sp}^{\varphi}(m)] \leq O\left(q^5 n 2^{-\min(r,c)}\right).$$

We now briefly sketch how to adapt the above approach to the case of preimage finding. The detailed derivation is omitted since it is very similar to collision finding, and the actual bound for both is dominated by the bad database predicate. The proof of Lemma 6.4 is only technical and almost the same with the one derived for the collision resistance, namely Lemma 6.2, except that in the former one fewer output is queried. Also, the proof of Corollary 6.7 uses the same reasoning with its analogous statement for collision resistance, which is Lemma 6.6. If we are in a good database, the proof argues that—conditioned on no collisions in the database—creating a collision requires creating a new path. This requires creating a new tail via a k-query or

[6]We slightly abused the notation here. In Corollary 4.15 elements of the relation are pairs of input and output tuples of size $6l$. However, in the relation described here, for each element of R, the element obtained by applying the same permutation on the indices of both input and output tuples is also in R. Moreover, any element obtained this way satisfies the conditions in the description of the probabilities of p and p', we consider elements of R modulo the permutation action on the indices.

k'-query or reaching one of the already reachable outputs in the database via an h-query. In a good database, the same argument works for finding a preimage of some value, with a slight difference in the h query case (now, one has to reach a specific output, which is slightly more difficult). If we are in a bad database, without considering the preimage or collision events, we can use the bound for database being bad. Eventually, the bound for having a database dominates the ones for having collision or preimage in the good database. Hence, the preimage and collision finding bounds become equal.

These lemmas are sufficient to derive the bound for classical collision resistance in our framework. Consequently, they are sufficient to derive the same bound for classical preimage resistance. Moreover, the quantum bound for collision resistance is a direct consequence of the classical bound due to quantum transition capacity and employment of Corollary 4.15, which are technical tools and apply similarly to preimage predicate.

7 Sponge indifferentiability proofs

Let us consider here the Msponge construction with functions h, k, k' and permutation π determining the sponge permutation φ, as described in Section 5.2. When using compressed oracles here, we will consider the efficient representation. We consider two worlds.

Real world: The experiment proceeds as:

(1) A permutation $\pi : \{0,1\}^n \to \{0,1\}^n$ is selected at uniform random.

(2) A database D_h and $D_k, D_{k'}$ are initialized as empty

(3) A q query quantum algorithm $\mathcal{A}$ receives oracle access to π, π^{-1} and (compressed) oracle access to h, k, k' using cO on databases $D_h, D_k, D_{k'}$, and an oracle for $\mathsf{Sp}^{hk'\pi k}$ implemented using the prior oracles.

(4) The algorithm $\mathcal{A}$ outputs a bit b.

We can write the real world via a set of quantum registers:

$$\underbrace{|A\rangle_A}_{\text{distinguisher}} \quad |D_h\rangle_H |D_k\rangle_K |D_{k'}\rangle_{K'} . \tag{21}$$

Ideal world: The experiment proceeds as:

(1) A permutation $\pi : \{0,1\}^n \to \{0,1\}^n$ is selected at uniform random.

(2) A database D_f is initialized as empty

(3) A simulator $\mathcal{S}$ is given query access to π, π^{-1} and f using cO on D_f.

(4) A q query quantum algorithm $\mathcal{A}$ receives oracle access to π, π^{-1}, access to simulated oracles for h, k, k' by querying the simulator, and an oracle for f implemented using the compressed oracle on D_f.

(5) The algorithm $\mathcal{A}$ outputs a bit b.

We can write the ideal world via a set of quantum registers:

$$\underbrace{|A\rangle_A}_{\text{distinguisher}} \quad \underbrace{|D_h\rangle_H |D_k\rangle_K |D_{k'}\rangle_{K'}}_{\text{simulator}} |D_f\rangle_F , \tag{22}$$

where the underbraces denotes the quantum registers maintained by the distinguisher and the simulator. We will in fact consider indistinguishability of the two worlds given a fixed permutation π, for any good π (as in Definition 5.8). We know $1 - O(2^{-n})$ permutations are good, so this incurs a negligible loss which we will ignore. Note that our simulator will in fact be secure even against adversaries that see the whole truth table of π, so long as π is good.

7.1 Defining the simulator

To define the action of the simulator, we will need to define a "find tail" operation, fT. This operation takes a $z \in \{0,1\}^c$, and examines $D_k, D_{k'}$ to find the $\mathsf{tail}(z)$, finding the first such tail if many exist, or a fail flag if one cannot be found. If a tail is found, then this operation also returns the corresponding head x_i. Note that the tail uniquely determines the corresponding head.

Definition 7.1. *The operation* find-tail*, which we write as* fT*, is a unitary acting on a tail input register* Z*, databases* K, K' *of functions* $\{0,1\}^r \to \{0,1\}^c$ *with at most* t *input points, and output registers* T *for the tail and associated head and* S *for the success flag. We let* $\mathsf{tail}(z)_1$ *denote the lexicographically first tail of* z *in* K, K'*. The operation then acts like*

$$\mathsf{fT}\left|z\right\rangle_Z\left|tl\right\rangle_T\left|s\right\rangle_S\left|D_k\right\rangle_K\left|D_{k'}\right\rangle_{K'} := \begin{cases} \left|z\right\rangle_Z\left|tl\right\rangle_T\left|s\right\rangle_S\left|D_k\right\rangle_K\left|D_{k'}\right\rangle_{K'} & (\mathsf{tail}(z)=\emptyset) \\ \left|z\right\rangle_Z\left|tl\oplus\mathsf{tail}_1(z)\right\rangle_T\left|s\oplus 1\right\rangle_S\left|D_k\right\rangle_K\left|D_{k'}\right\rangle_{K'} & \textit{(Otherwise)} \end{cases}$$

We can now define the action of the simulator. Queries to k, k' are answered using the compressed oracle on $D_k, D_{k'}$. Queries to h on input z are answered using the following procedure.

(1) First, compute fT, to find the tail tl of z if one exists

(2) If a tail does not exist, answer using the compressed oracle and D_h.

(3) If a tail does exist, then determine it's corresponding head x_i.

(4) Then, query $f(tl)$, and return $x_i \oplus f(tl)$.

(5) Finally, uncompute intermediate variables used above.

7.2 Good databases

For this section, we will require a refinement of good databases. In the real world, good databases satisfy the definition we have been using all along. In the ideal world, we additionally stipulate that no z value with a tail appears in the H database. Intuitively, this will be satisfied because the simulator will never query a z value to h which has a tail, and it is unlikely a query to k' attaches to a z value already queried. We will follow the strategy outlined by Lemma 3.12, prooving separately indistinguishability and consistency of our simulator. Our indisitinguishability proof requires both notions, and our consistency proof requires only the ideal notion.

Definition 7.2. *We say that databases* $D_h, D_k, D_{k'}$ *are "ideal good" if they satisfy Definition 5.11 and* D_h *is undefined on every* z *value that has a tail. We use* $\Pi^{\mathsf{ig}}_{KK'H}$ *as a projector onto ideal good databases, and* IG *as the predicate for ideal good databases.*

Definition 7.3. *We say that database* $D_k, D_{k'}$ *are "real good" if they satisfy Definition 5.11. We use* $\Pi^{\mathsf{rg}}_{KK'}$ *as a projector onto real good databases, and* RG *as the predicate for real good databases.*

It will be an important fact that the states in our experiment are close to ideal good and real good, in the corresponding worlds. After q queries to the above simulator, the norm of the state on bad databases satisfies $\left\|\Pi^{\mathsf{ig}\perp}_{HKK'}\left|\psi^q_I\right\rangle\right\| = \tilde{O}(\sqrt{q^5 2^{-\min(r,c)}})$. After q queries to any oracle in the real world, the norm of the state on bad databases satisfies $\left\|\Pi^{\mathsf{rg}\perp}_{KK'}\left|\psi^q_R\right\rangle\right\| = \tilde{O}(\sqrt{q^5 2^{-\min(r,c)}})$. This can be expressed formally as follows.

Remark 7.4. *We have transition capacities under queries to k, k', and h[7] given by*

$$[\![\mathsf{IG}^t \to \neg\mathsf{IG}]\!] \leq \tilde{O}(\sqrt{t^3 2^{-\min(r,c)}}) \qquad [\![\mathsf{RG}^t \to \neg\mathsf{RG}]\!] \leq \tilde{O}(\sqrt{t^3 2^{-\min(r,c)}}).$$

These further imply

$$[\![\emptyset \xrightarrow{q} \neg\mathsf{IG}]\!] \leq \tilde{O}(\sqrt{q^5 2^{-\min(r,c)}}) \qquad [\![\emptyset \xrightarrow{q} \neg\mathsf{RG}]\!] \leq \tilde{O}(\sqrt{q^5 2^{-\min(r,c)}}).$$

Proof. The statements concerning RG follow from combining Lemma 5.18 and Lemma 4.13.

For the statements involving IG, two observations are in order. First, observe that the simulator's action $\mathcal{S}^h$ does not affect the database register corresponding to any input $z \in D_h$ where $\mathsf{tail}(z) \neq \emptyset$; this is by definition, since in this case the simulator answers using a compressed query to database register F. Thus, a query to h cannot cause a bad event in our simulator, and $\mathcal{S}^h$ preserves the ideal good subspace.

It now only remains to analyze queries to k and k'. A transition from ideal good to ideal bad can happen on such queries in the case of a bad attach or a bad completion, whose distances are bounded by Lemmas 5.14 and 5.16. It may also happen in the case where a query causes a value z to have a tail where $(z, y) \in D_h$; note that there are at most t such values by assumption, so using an argument similar to Lemmas 5.14 and 5.16, one can construct a local property of similarly bounded distance governing this translition. The statements then follow from Lemma 5.18 and Lemma 4.13. □

7.3 Indistingushability

We first define an isometry $V : \mathcal{H}_{HKK'} \to \mathcal{H}_{HKK'F}$ which will map three function databases to four function databases. V will map computational basis vectors to computational basis vectors, so it is easiest to define V in terms of an injective function V_c on such databases (we consider π here to be fixed between both experiments):

$V_c(D_h^R, D_k^R, D_{k'}^R) = D_h^I, D_k^I, D_{k'}^I, D_f^I$ such that:

$$D_k^I(x) = D_k^R(x)$$
$$D_{k'}^I(x) = D_{k'}^R(x)$$
$$D_h^I(z) = \begin{cases} D_h^R(z) & \text{(If } z \text{ has no tail in } D_k^R, D_{k'}^R\text{)} \\ \perp & \text{(Otherwise)} \end{cases}$$
$$D_f^I(x_1 \| \ldots \| x_m) = \begin{cases} D_h^R(z) \oplus \mathsf{head}(z) & \text{(If } z\text{'s first tail is } x_1 \| \ldots \| x_m\text{)} \\ \perp & \text{(Otherwise)} \end{cases}$$

When we say the first tail of z, we mean the first tail under lexicographic ordering. Observe that V_c is a bijection from real good databases to ideal good databases. We define the full isometry V as the linear continuation of V_c, noting that V is an isometry because V_c is an injective mapping on basis states.

[7] Observe that in the ideal world queries to h are not implemented using just cO, and we technically did not define the transition capacity in this case. It will turn out that the simulators action on h queries perfectly preserves ideal goodness, so this point is moot.

Commutation relations of V. It turns out that the isometry V nearly commutes with the compression operator on both K and K', notated as L in Equation (11), at least on the good subspace. Here and going forward, we use the projector Π^t to denote the projection onto databases with at most t non-$\perp$ outputs. Definition 3.5 defines the commutator notion of a unitary and an isometry used here.

Lemma 7.5. *The commutator between the isometry V and the local compression operators on K almost commute on the good subspace:*

$$\left\|[V_{KK'H}, L_{XK}]\Pi^t_{KK'H}\Pi^{\mathsf{rg}}_{KK'}\right\| \leq \tilde{O}(\left(\sqrt{t^3 2^{-\min(r,c)}}\right).$$

Proof. The key observation is essentially that V is a bijective function on good databases which leaves the k, k' databases invariant, and a query to either k or k' remains mostly within the set of good databases. In more detail, consider projectors $\Pi^{\in\mathsf{db}}_{XK}, \Pi^{\notin\mathsf{db}}_{XK}$ which sum to the identity. Using the triangle inequality, we split into two cases.

(Case 1) $x \notin D_k$, or $\left\|[V_{KK'H}, L_{XK}]\Pi^t_{KK'H}\Pi^{\mathsf{rg}}_{KK'}\Pi^{\notin\mathsf{db}}_{XK}\right\|$. Note that L_{XK} preserves computational basis states, i.e. is a controlled operator, on everything except the x-th register of D_k, and on distinct databases the images of V are orthogonal. Additionally, $\Pi^{\notin\mathsf{db}}_{XK}|x\rangle_X = |\perp\rangle\langle\perp|_{D_x}$. By Lemma 3.7, it suffices to fix $x, D_k, D_{k'}, D_h$ as computational basis states with good databases of size at most t and $x \notin D_k$[8], and consider the action on the state

$$|\psi\rangle = |x\rangle_X |D_k\rangle_K |D_{k'}\rangle_{K'} |D^R_h\rangle_H .$$

Let us define D^I_h (in the ideal world) as the database of input/output pairs in D^R_h (in the real world) without a tail under $D_k, D_{k'}$, and D^I_f (in the ideal world) as the database for f constructed from the set of input/output pairs in D^R_h with a tail. These are such that

$$V|D_k\rangle_K |D_{k'}\rangle_{K'} |D^R_h\rangle_H = |D_k\rangle_K |D_{k'}\rangle_{K'} |D^I_h\rangle_H |D^I_f\rangle_F .$$

Recall that databases without a superscript do not change between the ideal and real world. We may now compute

$$\begin{aligned}|\psi^I\rangle =& L_{XK}V|\psi\rangle \\ =& |A\rangle_A |x\rangle_X |D^I_h\rangle_H |D_{k'}\rangle_{K'} |D^I_f\rangle_F \left(\sum_{y\in\{0,1\}^c} 2^{-c/2}|D_k[x\to y]\rangle_K\right)\end{aligned}$$

as well as

$$\begin{aligned}|\psi^R\rangle =& L_{XK}|\psi\rangle \\ =& |A\rangle_A |x\rangle_X |D^R_h\rangle_H |D_{k'}\rangle_{K'} \left(\sum_{y\in\{0,1\}^c} 2^{-c/2}|D_k[x\to y]\rangle_K\right).\end{aligned}$$

[8] the cleanest argument is obtained by applying Lemma 3.7 first wrt. register X, and then for the remaining registers

Let B be the set of images y of x such that assigning x to y will cause a bad attach. By Lemma 5.14, we have $|B| \leq O(t^3 n + t^3 2^{c-r})$. For any value $y \notin B$, we have the identity

$$V \left|D_k[x \to y]\right\rangle_K \left|D_{k'}\right\rangle_{K'} \left|D_h^I\right\rangle_H = \left|D_k[x \to y]\right\rangle_K \left|D_{k'}\right\rangle_{K'} \left|D_h^R\right\rangle_H \left|D_f^R\right\rangle_F,$$

because in such cases no new z values will have a tail under the assignment $[x \to y]$. It follows that

$$V\Pi^{B\perp} \left|\psi^R\right\rangle = \Pi^{B\perp} \left|\psi^I\right\rangle. \tag{23}$$

We can write

$$\begin{aligned}
\left\| \left|\psi^R\right\rangle - \Pi^{B\perp} \left|\psi^R\right\rangle \right\| &= \left\| \left|A\right\rangle_A \left|x\right\rangle_X \left|D_h^R\right\rangle_H \left|D_{k'}\right\rangle_{K'} \left(\sum_{y \in B} 2^{-c/2} \left|D_k[x \to y]\right\rangle_K \right) \right\| \\
&= \sqrt{|B| 2^{-c}} \qquad (24) \\
&\leq \tilde{O}(\sqrt{t^3 2^{-\min(r,c)}}), \\
\left\| \left|\psi^I\right\rangle - \Pi^{B\perp} \left|\psi^I\right\rangle \right\| &= \left\| \left|A\right\rangle_A \left|x\right\rangle_X \left|D_h^I\right\rangle_H \left|D_{k'}\right\rangle_{K'} \left|D_f^I\right\rangle_F \left(\sum_{y \in B} 2^{-c/2} \left|D_k[x \to y]\right\rangle_K \right) \right\| \\
&= \sqrt{|B| 2^{-c}} \\
&\leq \tilde{O}(\sqrt{t^3 2^{-\min(r,c)}}). \qquad (25)
\end{aligned}$$

Putting everything together, we have

$$\begin{aligned}
\left\| \left|\psi^I\right\rangle - V \left|\psi^R\right\rangle \right\| &\leq \left\| \left|\psi^I\right\rangle - \Pi^{B\perp} \left|\psi^I\right\rangle \right\| + \left\| \Pi^{B\perp} \left|\psi^I\right\rangle - V\Pi^{B\perp} \left|\psi^R\right\rangle \right\| \\
&\quad + \left\| V \left|\psi^R\right\rangle - V\Pi^{B\perp} \left|\psi^R\right\rangle \right\| && \text{(Triangle inequality)} \\
&= \left\| \left|\psi^I\right\rangle - \Pi^{B\perp} \left|\psi^I\right\rangle \right\| + \left\| V \left|\psi^R\right\rangle - V\Pi^{B\perp} \left|\psi^R\right\rangle \right\| && \text{(Equation (23))} \\
&= \left\| \left|\psi^I\right\rangle - \Pi^{B\perp} \left|\psi^I\right\rangle \right\| + \left\| \left|\psi^R\right\rangle - \Pi^{B\perp} \left|\psi^R\right\rangle \right\| && (V \text{ an isometry}) \\
&\leq \tilde{O}(\sqrt{t^3 2^{-\min(r,c)}}) && \text{(Equations (24) and (25))}
\end{aligned}$$

(Case 2) $x \in D_k$, or $\left\| [V_{KK'H}, L_{XK}] \Pi^t_{KK'} \Pi^{\mathsf{rg}}_{KK'} \Pi^{\in \mathsf{db}}_{XK} \right\|$. Once again, note that L_{XK} preserves the computational basis everywhere except the x-th register of D_k, and on distinct databases the images of V are orthogonal. By Lemma 3.7, it suffices to fix $x, D_k, D_{k'}, D_h$ as computational basis states with good databases of size at most t and where $x \notin D_k$, and consider the action on a state of the form

$$\left|\psi\right\rangle = \sum_{z \text{ s.t. } D_k[x \to z], D_{k'} \text{ is good}} \alpha_z \left|x\right\rangle_X \left|D_k[x \to z]\right\rangle_K \left|D_{k'}\right\rangle_{K'} \left|D_h\right\rangle_H.$$

Let us similarly define $D_h^{I,y}$ (in the ideal world) as the database of input/output pairs in D_h^R (in the real world) without a tail under $D_k[x \to y], D_{k'}$, and $D_f^{I,y}$ (in

the ideal world) as the database for f constructed from the set of input/output pairs in D_h^R with a tail in $D_k[x \to y], D_{k'}$. These are such that

$$V \left|D_k[x \to y]\right\rangle_K \left|D_{k'}\right\rangle_{K'} \left|D_h^R\right\rangle_H = \left|D_k[x \to y]\right\rangle_K \left|D_{k'}\right\rangle_{K'} \left|D_h^{I,y}\right\rangle_H \left|D_f^{I,y}\right\rangle_F .$$

Recall that databases without a superscript do not change between the ideal and real world. We may now compute

$$\begin{aligned}
\left|\psi^I\right\rangle =& L_{XK} V \left|\psi\right\rangle \\
=& \sum_{z \text{ s.t. } D_k[x \to z], D_{k'} \text{ is good}} \alpha_z \left|A\right\rangle_A \left|x\right\rangle_X \left|D_h^{I,z}\right\rangle_H \left|D_{k'}\right\rangle_{K'} \left|D_f^{I,z}\right\rangle_F \otimes \\
& \left(\left|D_k[x \to z]\right\rangle_K - 2^{-c} \sum_{u \in \{0,1\}^c} \left|D_k[x \to u]\right\rangle_K + 2^{-c/2} \left|D_k\right\rangle_K \right)
\end{aligned}$$

as well as

$$\begin{aligned}
\left|\psi^R\right\rangle =& L_{XK} \left|\psi\right\rangle \\
=& \sum_{z \text{ s.t. } D_k[x \to z], D_{k'} \text{ is good}} \alpha_z \left|A\right\rangle_A \left|x\right\rangle_X \left|D_h^{R}\right\rangle_H \left|D_{k'}\right\rangle_{K'} \otimes \\
& \left(\left|D_k[x \to z]\right\rangle_K - 2^{-c} \sum_{u \in \{0,1\}^c} \left|D_k[x \to u]\right\rangle_K + 2^{-c/2} \left|D_k\right\rangle_K \right)
\end{aligned}$$

Let B be the set of images y of x such that assigning x to y will cause a bad attach. Observe that, from the analysis of Lemma 5.14, we have $|B| \leq O(t^3 n + t^3 2^{c-r})$. For any value $y \notin B$, we have the identity

$$V \left|D_k[x \to y]\right\rangle_X \left|D_{k'}\right\rangle_{K'} \left|D_h^I\right\rangle_H = \left|D_k[x \to y]\right\rangle_X \left|D_{k'}\right\rangle_{K'} \left|D_h^R\right\rangle_H \left|D_f^R\right\rangle ,$$

because in such cases no new state values will have a tail under the assignment $[x \to y]$. Observe here that the initial state $|\psi\rangle$ may be entirely supported on images in B that lead to a "bad attach", for instance if x is part of a tail in $D_k[x \to z], D_{k'}$. This prevents applying the strategy from the previous case where we just project out those values, so instead we have to explicitly write down the difference. Let us analyze the difference

$$\begin{aligned}
\left|\psi^I\right\rangle - V \left|\psi^R\right\rangle = & \sum_{z \text{ s.t. } D_k[x \to z], D_{k'} \text{ is good}} \alpha_z \left|A\right\rangle_A \left|x\right\rangle_X \left|D_h^{I,z}\right\rangle_H \left|D_{k'}\right\rangle_{K'} \left|D_f^{I,z}\right\rangle_F \otimes \\
& \left(\left|D_k[x \to z]\right\rangle_K - 2^{-c} \sum_{u \in \{0,1\}^c} \left|D_k[x \to u]\right\rangle_K + 2^{-c/2} \left|D_k\right\rangle_K \right) - \\
& \sum_{z \text{ s.t. } D_k[x \to z], D_{k'} \text{ is good}} \alpha_z \left|A\right\rangle_A \left|x\right\rangle_X \left|D_{k'}\right\rangle_{K'} \otimes \\
& \left(\left|D_k[x \to z]\right\rangle_K \left|D_h^{I,z}\right\rangle_H \left|D_f^{I,z}\right\rangle_F - 2^{-c} \sum_{u \in \{0,1\}^c} \left|D_k[x \to u]\right\rangle_K \left|D_h^{I,u}\right\rangle_H \left|D_f^{I,u}\right\rangle_F + \right. \\
& \left. 2^{-c/2} \left|D_k\right\rangle_K \left|D_h^{I,\perp}\right\rangle_H \left|D_f^{I,\perp}\right\rangle_F \right)
\end{aligned}$$

Which, after collapsing terms, can be written as

$$|\psi^I\rangle - V|\psi^R\rangle = \sum_{z \text{ s.t. } D_k[x\to z], D_{k'} \text{ is good}} \alpha_z |A\rangle_A |x\rangle_X |D_{k'}\rangle_{K'} \otimes \left(2^{-c} \sum_{u\in\{0,1\}^c} |D_k[x\to u]\rangle_K (|D_h^{I,u}\rangle_H |D_f^{I,u}\rangle_F - |D_h^{I,z}\rangle_H |D_f^{I,z}\rangle_F)\right) + \sum_{z \text{ s.t. } D_k[x\to z], D_{k'} \text{ is good}} \alpha_z |A\rangle_A |x\rangle_X |D_{k'}\rangle_{K'} \otimes \left(|D_k\rangle_K 2^{-c/2}(|D_h^{I,z}\rangle_H |D_f^{I,z}\rangle_F - |D_h^{I,\perp}\rangle_H |D_f^{I,\perp}\rangle_F)\right).$$

We can then write

$$\begin{aligned}\|VL|\psi\rangle - LV|\psi\rangle\| &= \left\||V|\psi^R\rangle - |\psi^I\rangle\right\| \\ &\leq 2\underbrace{\left\|\sum_{z\in\{0,1\}^c} \alpha_z \sum_{u\in\{0,1\}^c, (D_h^{I,u}D_f^{I,u})\neq(D_h^{I,z}D_f^{I,z})} 2^{-c}|D_k[x\to u]\rangle\right\|}_{T_1} + \\ &\quad \underbrace{2\left\|\sum_{z\in\{0,1\}^c, (D_h^{I,z}D_f^{I,z})\neq(D_h^{I,\perp}D_f^{I,\perp})} \alpha_z 2^{-c/2}\right\|}_{T_2} \qquad \text{(Triangle Inequality)}\end{aligned}$$

Let us focus on bounding each term individually. We begin with

$$\begin{aligned}T_1 &= \left\|\sum_{z\in\{0,1\}^c} \sum_{u\in\{0,1\}^c, (D_h^{I,u}D_f^{I,u})\neq(D_h^{I,z}D_f^{I,z})} \alpha_z 2^{-c}|D_k[x\to u]\rangle\right\| \\ &\leq \underbrace{\left\|\sum_{z\in\{0,1\}^c, (D_h^{I,z}D_f^{I,z})\neq(D_h^{I,\perp}D_f^{I,\perp})} \sum_{u\in\{0,1\}^c, (D_h^{I,u}D_f^{I,u})\neq(D_h^{I,z}D_f^{I,z})} \alpha_z 2^{-c}|D_k[x\to u]\rangle\right\|}_{T_{11}} + \\ &\quad \underbrace{\left\|\sum_{z\in\{0,1\}^c, (D_h^{I,z}D_f^{I,z})=(D_h^{I,\perp}D_f^{I,\perp})} \sum_{u\in\{0,1\}^c, (D_h^{I,u}D_f^{I,u})\neq(D_h^{I,z}D_f^{I,z})} \alpha_z 2^{-c}|D_k[x\to u]\rangle\right\|}_{T_{12}} \qquad \text{(Triangle inequality)}\end{aligned}$$

Observe that the second sum in T_{11} is over at most 2^c terms, which gives an upper bound of $|\alpha_z|\,2^{-c/2}$ for it's norm. The first sum in T_{11} is over a set of size $O(t^3n + t^3 2^{c-r})$ by Lemmas 5.16 and 5.19, so by a standard inequality between L_1 and L_2 norm we have

$$\sum_{z\in\{0,1\}^c, (D_h^{I,z}D_f^{I,z})\neq(D_h^{I,\perp}D_f^{I,\perp})} |\alpha_z| \leq O(\sqrt{t^3n + t^3 2^{c-r}}). \tag{26}$$

It follows that $T_{11} \leq \tilde{O}(\sqrt{t^3 2^{-\min(r,c)}})$.

Observe that the second sum in T_{12} is over a set of size $O(t^3 n + t^3 2^{c-r})$ by Lemmas 5.16 and 5.19, which gives an upper bound of $|\alpha_z| \sqrt{O(t^3 n + t^3 2^{c-r})} \cdot 2^{-c}$ for it's norm. The first sum in T_{12} is over a set of size at most 2^{-c}, so by the relation between L_1 and L_2 norm we have

$$\sum_{z \in \{0,1\}^c, (D_h^{I,z} D_f^{I,z}) = (D_h^{I,\perp} D_f^{I,\perp})} |\alpha_z| \leq 2^{c/2}.$$

It follows that $T_{12} \leq \tilde{O}(\sqrt{t^3 2^{-\min(r,c)}})$.

Finally, it follows from Equation (26) that $T_2 \leq \tilde{O}(\sqrt{t^3 2^{-\min(r,c)}})$.

□

Lemma 7.6. *The commutator between the isometry V and the local compression operators on K' almost commutes on the good subspace:*

$$\left\| [V_{KK'H}, L_{XK'}] \Pi^t_{KK'H} \Pi^{\mathsf{rg}}_{KK'} \right\| \leq \tilde{O}(\left(\sqrt{t^3 2^{-\min(r,c)}} \right).$$

The proof for the k' case is similar to the one for k. For completeness, we give it in Appendix A.

Query closeness. We can apply the previous commutator bounds to give a bound on how much the distinguisher's view can diverge as a result of queries to k, k', and h. We do this by showing that V acts as an approximate *intertwiner* between real world and ideal world queries. Consider, e.g., the query unitary cO_{AK} and the operator $\mathcal{S}^k$ the simulator applies upon a k query. Then $S^k V \approx V \mathsf{cO}_{AK}$ for an appropriate notion of $\approx$. Note that in this case $\mathcal{S}^k_{AHKK'F} = \mathsf{cO}_{AK}$, as the simulator simply answers by a compressed database call on database K, so here we have, in fact, a certain approximate commutation relation.

Lemma 7.7. *Let $\mathcal{S}^k$ denote the action of the simulator on a k query. Consider the following two operators,*

$$\begin{aligned} O^I &= \mathcal{S}^k_{AHKK'F} V_{HKK'} \Pi^{\mathsf{rg}}_{HKK'} \\ O^R &= V_{HKK'} \Pi^{\mathsf{rg}}_{HKK'} \mathsf{cO}_{AK} \Pi^{\mathsf{rg}}_{KK'} \end{aligned}$$

and let $\Pi^t_{HKK'}$ be the projector onto databases with at most t query points in each. Then we have

$$\left\| (O^I - O^R) \Pi^t \right\| \leq O(\sqrt{t^3 2^{-\min(r,c)}})$$

Proof. Since the simulator replies to k-queries simply by using the corresponding compressed oracle, we can write

$$\begin{aligned} O^I &= \mathcal{S}^k_{AHKK'F} V_{HKK'} \Pi^{\mathsf{rg}}_{HKK'} && (27) \\ &= L_{XK} \mathcal{P}_{XYK} L_{XK} V_{HKK'} \Pi^{\mathsf{rg}}_{HKK'}, && (28) \end{aligned}$$

observing that

$$V_{HKK'}\Pi^{\mathsf{rg}}_{KK'} = \Pi^{\mathsf{ig}}_{HKK'}V_{HKK'} \tag{29}$$

we further have

$$O^R = \Pi^{\mathsf{ig}}_{HKK'}V_{HKK'}L_{XK}\mathcal{P}_{XYK}L_{XK}\Pi^{\mathsf{rg}}_{KK'}.$$

Making heavy use of Equation (29), and the fact that $\mathcal{P}_{XYK}$ commutes with good projectors and the V operation, we have

$$\begin{aligned}
\left\|(O^I - O^R)\Pi^t\right\| &= \left\|(L\mathcal{P}LV\Pi^{\mathsf{rg}} - \Pi^{\mathsf{ig}}VL\mathcal{P}L\Pi^{\mathsf{rg}})\Pi^t\right\| \\
&\leq \left\|(L\mathcal{P}LV\Pi^{\mathsf{rg}} - L\mathcal{P}\Pi^{\mathsf{ig}}VL\Pi^{\mathsf{rg}})\Pi^t\right\| + \underbrace{\left\|(L\mathcal{P}V\Pi^{\mathsf{rg}}L\Pi^{\mathsf{rg}} - LV\Pi^{\mathsf{rg}}\mathcal{P}L\Pi^{\mathsf{rg}})\Pi^t\right\|}_{=0} + \\
&\quad \left\|(LV\Pi^{\mathsf{rg}}\mathcal{P}L\Pi^{\mathsf{rg}} - \Pi^{\mathsf{ig}}VL\Pi^{\mathsf{rg}}\mathcal{P}L\Pi^{\mathsf{rg}})\Pi^t\right\| + \\
&\quad \left\|VL\Pi^{\mathsf{rg}}\mathcal{P}L\Pi^{\mathsf{rg}})\Pi^t - \Pi^{\mathsf{ig}}VL\Pi^{\mathsf{rg}}\mathcal{P}L\Pi^{\mathsf{rg}})\Pi^t\right\| \quad \text{(Triangle inequality)} \\
&\leq \left\|[L_{XK}, V]\Pi^{\mathsf{rg}}\Pi^t\right\| + \left\|[L_{XK}, V]\Pi^{\mathsf{rg}}\Pi^{t+1}\right\| + \\
&\quad \tilde{O}(\sqrt{t^3 2^{-\min(r,c)}}) \quad \text{(Remark 7.4)} \\
&\leq \tilde{O}(\sqrt{t^3 2^{-\min(r,c)}}) \quad \text{(Lemma 7.5)}
\end{aligned}$$

□

Once again, note that $\mathcal{S}^{k'}_{AHKK'F} = \mathsf{cO}_{AK'}$, as the simulator simply answers by a compressed database call on database K.

Lemma 7.8. *Let $\mathcal{S}^{k'}$ denote the action of the simulator on a k' query. Consider the following two operators,*

$$\begin{aligned}
O^I &= \mathcal{S}^{k'}_{AHKK'F}V_{HKK'}\Pi^{\mathsf{rg}}_{HKK'} \\
O^R &= V_{HKK'}\Pi^{\mathsf{rg}}_{KK'}\mathsf{cO}_{AK'}\Pi^{\mathsf{rg}}_{KK'},
\end{aligned}$$

and let $\Pi^t_{HKK'}$ be the projector onto databases with at most t query points. Then we have

$$\left\|(O^I - O^R)\Pi^t\right\| \leq O(\sqrt{t^3 2^{-\min(r,c)}})$$

Proof. The proof follows similarly to that of Lemma 7.7, except with Lemma 7.6 taking the place of Lemma 7.5. □

Lemma 7.9. *Let $\mathcal{S}^h$ denote the action of the simulator on a h query. Consider the following two operators,*

$$\begin{aligned}
O^I &= \mathcal{S}^h_{AHKK'F}V_{HKK'}\Pi^{\mathsf{rg}}_{HKK'} \\
O^R &= V_{HKK'}\Pi^{\mathsf{rg}}_{KK'}\mathsf{cO}_{AH}\Pi^{\mathsf{rg}}_{KK'}.
\end{aligned}$$

Then we have

$$O^I = O^R.$$

Proof. Recall that V is a bijection from good real databases to good ideal databases, and an injection from real databases to ideal. Observe that the construction of D_f^I and D_h^I is simply a re-labeling of the input/output pairs contained in D_h^R, depending on which have a tail. Further, the simulator answers according to a random function which depends only on the input value of such pairs. Let us work out the action of O^I and O^R on good databases. Let

$$|\psi^R\rangle = |A\rangle_A |x\rangle_X |y\rangle_Y |D_k\rangle_K |D_{k'}\rangle_{K'} |D_h\rangle_h$$

denote a computational basis state in the real world which is good.

(O^I). We will consider the action on $|\psi^R\rangle$. We have $\mathsf{tail}(z)$ (with respect to D_k, $D_{k'}$) is either an empty or a singleton set, and so is $\mathsf{head}(z)$. Let us define the sets

$$Z_t = \{z | z \in D_h, \mathsf{tail}(z) \neq \emptyset\}$$
$$Z_{\neg t} = \{z | z \in D_h, \mathsf{tail}(z) = \emptyset\}.$$

We can then work out

$$V\Pi^{\mathsf{rg}} |\psi_R\rangle = |A\rangle_A |x\rangle_x |y\rangle_y |D_k\rangle_k |D_{k'}\rangle_{k'} |\{(z, h(z)) | z \in Z_{\neg t}\}\rangle_h |\{(\mathsf{tail}(z), h(z) \oplus \mathsf{head}(z)) | z \in Z_t\rangle_f$$

The action of the simulator can then be split into two cases.

(1) $x \in Z_t$. In this case, the query is answered using a compressed oracle call on input $\mathsf{tail}_1(x)$ to D_f^I, and XORed with $\mathsf{head}(x)$.

(2) $x \in Z_{\neg t}$. In this case, the query is answered using a compressed oracle call on input x to D_h.

(O^R). We will consider the action on $|\psi^R\rangle$. We have $\mathsf{tail}(z)$ (with respect to D_k, $D_{k'}$) is of size at most one. Let us again consider the sets

$$Z_t = \{z | z \in D_h, \mathsf{tail}(z) \neq \emptyset\}$$
$$Z_{\neg t} = \{z | z \in D_h, \mathsf{tail}(z) = \emptyset\}.$$

The query to h will be answered using cO_{XYH}, which is a controlled operation on x that targets the x-th output in the database for h. Note that (because D_h is good) we have that V will simply move and permute the labels of input output pairs. It follows that

$$VL_{XH} |x\rangle_X = \begin{cases} C_f^{\mathsf{tail}(x)} V |x\rangle_X & (\text{If } \mathsf{tail}(x) \neq \emptyset) \\ C_H^x V |x\rangle_X & (\text{Otherwise}) \end{cases}$$

Using the notation $\mathcal{P}_{x,y,H}$ to denote $\mathcal{P}$ with input value x, output value y, and database register H, we then have for any x, y that

$$V\mathcal{P}_{XYH} |x\rangle_X |y\rangle_Y = \begin{cases} \mathcal{P}_{\mathsf{tail}(x), y \oplus \mathsf{head}(x), F} V |x\rangle_X |y\rangle_Y & (\text{If } \mathsf{tail}(x) \neq \emptyset) \\ \mathcal{P}_{x,y,H} V |x\rangle_X |y\rangle_Y & (\text{Otherwise}) \end{cases}$$

It follows that queries are answered using the same procedure as by the simulator.

The action of O^I and O^R is the same on all good computational basis states, so it is the same on the subspace spanned by good computational basis states. Note that Π^{rg} does not depend on the h database, and so clearly commutes with cO_{AH}; this justifies the final Π^{rg} in O^R. □

Putting the pieces together.

Theorem 7.10. *The simulator defined in Section 7.1 is indistinguishable. In particular, for a q-query distinguisher we have*

$$|\Pr[\mathcal{A}^{\pi,\pi^{-1},h,k,k'}() = 1] - \Pr[\mathcal{A}^{\pi,\pi^{-1},\mathcal{S}^f}() = 1]| = \tilde{O}\left(\sqrt{q^5 2^{-\min(r,c)}}\right)$$

Proof. Let us consider some initial state for the real experiment,

$$|\psi^0\rangle = |\alpha^0\rangle_A |\emptyset\rangle_H |\emptyset\rangle_K |\emptyset\rangle_{K'},$$

where $|\alpha^0\rangle$ is an arbitrary initial state of the adversary. Let us consider in fact a strengthened adversary which is time unbounded, and which holds the entire truth table of π. The only constraint we require is that π is good, which happens with probability $1 - O(2^{-n})$. Our proof shows that for any fixed, good π, the real and ideal worlds are indistinguishable. We will do this by showing that the reduced density matrix of an adversary which queries compressed random oracles for h, k, k', and one which queries the simulator are close, which we do by showing an isometry mapping the purification of the real state to an approximate purification of the ideal state.

We can WLOG take our adversary to alternate queries to the various oracles (at a loss of an $O(1)$ factor), suppose in the order k, k', h. We can parameterize a q-query adversary of this form by unitaries $U^1, \ldots, U^q$ acting on the internal register A. The final state in the real experiment is given by

$$|\psi^R\rangle = U^q_A \ldots U^3_A \mathsf{cO}_{AH} U^2_A \mathsf{cO}_{AK'} U^1_A \mathsf{cO}_{AK} |\psi^0\rangle.$$

We can write the state at the end of the ideal experiment as

$$|\psi^I\rangle = U^q_A \ldots U^3_A \mathcal{S}^h_{AHKK'F} U^2_A \mathcal{S}^{k'}_{AHKK'F} U^1_A \mathcal{S}^k_{AHKK'F} V_{KK'H} |\psi^0\rangle.$$

Recall that we would like show that $V_{KK'H}|\psi^R\rangle$ is close to $|\psi^I\rangle$. Let us define $|\psi^{R,g}\rangle$ similar to $|\psi^R\rangle$, but with projectors onto good placed after each query:

$$|\psi^{R,g}\rangle := U^q_A \Pi^{\mathsf{rg}} \ldots U^3_A \Pi^{\mathsf{rg}} \mathsf{cO}_{AH} U^2_A \Pi^{\mathsf{rg}} \mathsf{cO}_{AK'} U^1_A \Pi^{\mathsf{rg}} \mathsf{cO}_{AK} \Pi^{\mathsf{rg}} |\psi^0\rangle.$$

Note that in the real world, we have

$$\begin{aligned} \left\| V_{KK'H}|\psi^R\rangle - V_{KK'H}|\psi^{R,g}\rangle \right\| &= \left\| |\psi^R\rangle - |\psi^{R,g}\rangle \right\| && (V \text{ an isometry}) \\ &\leq \tilde{O}(\sqrt{q^5 2^{-\min(r,c)}}) && (\text{Remark } 7.4) \end{aligned}$$

and in the ideal,

$$\begin{aligned} |\psi^I\rangle =& U^q_A \ldots U^3_A \mathcal{S}^h_{AHKK'F} U^2_A \mathcal{S}^{k'}_{AHKK'F} U^1_A \mathcal{S}^k_{AHKK'F} V_{KK'H} |\psi^0\rangle \\ =& U^q_A \ldots U^3_A \mathcal{S}^h_{AHKK'F} U^2_A \mathcal{S}^{k'}_{AHKK'F} U^1_A \mathcal{S}^k_{AHKK'F} V_{KK'H} \Pi^{\mathsf{rg}} |\psi^0\rangle. \end{aligned}$$

Let us now define

$$\begin{aligned}
|\psi_0^I\rangle :=& U_A^q \dots U_A^3 \mathcal{S}^h_{AHKK'F} U_A^2 \mathcal{S}^{k'}_{AHKK'F} U_A^1 \mathcal{S}^k_{AHKK'F} V_{KK'H} \Pi^{\mathsf{rg}} |\psi^0\rangle \\
|\psi_1^I\rangle :=& U_A^q \dots U_A^3 \mathcal{S}^h_{AHKK'F} U_A^2 \mathcal{S}^{k'}_{AHKK'F} V_{KK'H} \Pi^{\mathsf{rg}} U_A^1 \mathsf{cO}_{AK} \Pi^{\mathsf{rg}} |\psi^0\rangle \\
|\psi_2^I\rangle :=& U_A^q \dots U_A^3 \mathcal{S}^h_{AHKK'F} V_{KK'H} \Pi^{\mathsf{rg}} U_A^2 \mathsf{cO}_{AK'} \Pi^{\mathsf{rg}} U_A^1 \mathsf{cO}_{AK} \Pi^{\mathsf{rg}} |\psi^0\rangle \\
&\vdots \\
|\psi_q^I\rangle :=& V_{KK'H} \Pi^{\mathsf{rg}} U_A^q \dots \Pi^{\mathsf{rg}} U_A^3 \mathsf{cO}_{AH} \Pi^{\mathsf{rg}} U_A^2 \Pi^{\mathsf{rg}} \mathsf{cO}_{AK'} U_A^1 \mathsf{cO}_{AK} \Pi^{\mathsf{rg}} |\psi^0\rangle ,
\end{aligned}$$

where we have $|\psi_0^I\rangle = |\psi^I\rangle$ and $|\psi_q^I\rangle = V |\psi^{R,g}\rangle$ (since the U's acting on A commute with $\Pi^{\mathsf{rg}}_{KK'}$), meaning $\left\| |\psi_q^I\rangle - V |\psi^R\rangle \right\| \leq \tilde{O}(\sqrt{q^5 2^{-\min(r,c)}})$ by the argument prior. To complete the proof, observe that we have

$$\begin{aligned}
\left\| |\psi_1^I\rangle - |\psi_0^I\rangle \right\| \leq& \tilde{O}(\sqrt{1^3 2^{-\min(r,c)}}) && \text{(By Lemma 7.7)} \\
\left\| |\psi_2^I\rangle - |\psi_1^I\rangle \right\| \leq& \tilde{O}(\sqrt{2^3 2^{-\min(r,c)}}) && \text{(By Lemma 7.8)} \\
&\vdots \\
\left\| |\psi_{t+1}^I\rangle - |\psi_t^I\rangle \right\| \leq& \tilde{O}(\sqrt{t^3 2^{-\min(r,c)}}) && \text{(By Lemmas 7.7 to 7.9)}
\end{aligned}$$

and by triangle inequality,

$$\begin{aligned}
\left\| |\psi^I\rangle - V_{KK'H} |\psi^R\rangle \right\| &\leq \left\| |\psi_q^I\rangle - V_{KK'H} |\psi^R\rangle \right\| + \sum_{t=1}^{q} \left\| |\psi_t^I\rangle - |\psi_{t-1}^I\rangle \right\| \\
&\leq \tilde{O}(\sqrt{q^5 2^{-\min(r,c)}}).
\end{aligned}$$

The claim now follows from Lemma 3.6. □

7.4 Consistency

We would like to argue that the procedure for answering Msponge queries using the simulator are close to the ideal functionality. We will compute the Msponge using an out-of-place circuit in each round, as depicted in Figures 2 to 4. Let l be an upper bound on the block length of an Msponge input, and we will use t to represent the number of queries made so far to the oracles. Let us first define some operations which can be seen as building blocks of the full Msponge.

Throughout this section, let $|x\rangle_X |D_k\rangle_K |D_{k'}\rangle_{K'} |D_h\rangle_H$ denote input registers and function databases respectively. We split $x = x_1 \| \dots \| x_l$ into r bit blocks.

Definition 7.11. *Define isometry A as the isometry which simply appends registers $|0\rangle_Z |0\rangle_{Hd} |0\rangle_W$ for the state, head, and intermediate outputs registers, as well as $|0\rangle_T |0\rangle_S$ for the tail, and success flag.*

Definition 7.12. *Define unitary U as the unitary which out-of-place computes the sponge state in input x, depicted in Figures 2 and 3. In particular, compute the sponge state from X into Z and Hd registers, as in Figure 3, using the W register for intermediate computations.*

Definition 7.13. *We define isometries O^R, O^I as follows. Each operation will make use of A and U, and we refer to the Z, Hd, W registers created by A using the same labels. We will refer to the tail T as both a tail portion and a recovered head (potentially distinct from the actual head Hd) portion. The operators both begin by applying A to create the commensurate registers, and continue as follows.*

O^R. *Continue with the following operations:*

(1) *Use U to compute the sponge state from X into Z and Hd registers, as in Figure 3, using the W register for intermediate computations*

(2) *Call* find-tail *with input register Z, on databases K, K', with target register T (both portions) and success flag target S.*

O^I. *Continue with the following operations:*

(1) *Flip on the success flag S*

(2) *XOR the X register into the tail portion of T*

(3) *Use U to compute the sponge state from X into Z and Hd, as in Figure 3, using the W register for intermediate computations*

(4) *XOR the head Hd into the recovered head portion of T*

To start, we will show that the databases for K, K' almost always contain all of the input-output points computed by O^R, so long as the input state is valid. Noting that this is true for the uncompressed databases, this essentially follows from the fundamental lemma. Let us begin by defining a projector onto states where K, K' indeed contain all of the input-output points computed by O^R. Here g denotes a placeholder data for workspace values which do not correspond to input-output pairs.

Definition 7.14. *Define the recorded projector $\Pi^r_{X,W,K,K'}$ that checks whether every $(x, k(x))$ and $(x', k(x'))$ pair in the X, W registers appears in the K, K' database. In particular, we have*

$$|\psi\rangle = |\mathbf{x}\rangle_X |\mathbf{w}\|\mathbf{x}'\|\mathbf{w}'\|\mathbf{g}\rangle_W |D_k\rangle_K |D_{k'}\rangle_K$$

$$\Pi^r |\psi\rangle := \begin{cases} 0 & (\text{If } \exists i \text{ s.t. } (x_i, w_i) \notin D_k) \\ 0 & (\text{If } \exists i \text{ s.t. } (x'_i, w'_i) \notin D_{k'}) \\ |\psi\rangle & (\text{Otherwise}) \end{cases}$$

Lemma 7.15. *On a valid initial state $|\psi\rangle$, the input to the* MSp *is almost always recorded in the databases after the out-of-place circuit. Formally,*

$$\left\| \Pi^r O^R |\psi\rangle \right\| \geq 1 - O(\sqrt{l 2^{-c}}),$$

Proof. Let $|\psi\rangle_{XKK'H}$ be a state such that $\Pi^v |\psi\rangle = |\psi\rangle$. Observe that the fully uncompressed databases of K, K' would be guaranteed to satisfy Π^r at the end; this is because the X and W registers contain an exact record of the queries made to k, k'. In other words, we can write

$$\Pi^r C_K C_{K'} O^R |\psi\rangle = C_K C_{K'} O^R |\psi\rangle , \tag{30}$$

where here the C operator denotes full decompression. Let us define

$$|\psi'\rangle = O^R |\psi\rangle = \sum_{\mathbf{x},\mathbf{w},\mathbf{x}',\mathbf{w}'} \alpha_{\mathbf{x},\mathbf{w},\mathbf{x}',\mathbf{w}'} |\mathbf{x}\rangle_X |\mathbf{w}\|\mathbf{x}'\|\mathbf{w}'\|\mathbf{g}\rangle_W |D_{k,\mathbf{x},\mathbf{w},\mathbf{x}',\mathbf{w}'}\rangle_K |D_{k',\mathbf{x},\mathbf{w},\mathbf{x}',\mathbf{w}'}\rangle_{K'}$$

where $\mathbf{x}, \mathbf{w}, \mathbf{x}', \mathbf{w}'$ are the recorded input-output pairs at the end of O^R, and $\mathbf{g}$ captures all other non-database wires. We know that $|\psi'\rangle$ satisfies Equation (30), which implies

$$\forall \mathbf{x}, \mathbf{w}, \mathbf{x}', \mathbf{w}' :$$
$$\Pi^{(\mathbf{x},\mathbf{w})} C_K |D_{k,\mathbf{x},\mathbf{w},\mathbf{x}',\mathbf{w}'}\rangle_K = C_K |D_{k,\mathbf{x},\mathbf{w},\mathbf{x}',\mathbf{w}'}\rangle_K$$
$$\Pi^{(\mathbf{x}',\mathbf{w}')} C_{K'} |D_{k',\mathbf{x},\mathbf{w},\mathbf{x}',\mathbf{w}'}\rangle_{K'} = C_{K'} |D_{k',\mathbf{x},\mathbf{w},\mathbf{x}',\mathbf{w}'}\rangle_{K'} .$$

We can now write

$$\begin{aligned}
\left\|\Pi^r O^R |\psi\rangle\right\| =& \Bigg\| \Pi^r \sum_{\mathbf{x},\mathbf{w},\mathbf{x}',\mathbf{w}'} \alpha_{\mathbf{x},\mathbf{w},\mathbf{x}',\mathbf{w}'} |\mathbf{x}\rangle_X |\mathbf{w}\|\mathbf{x}\|\mathbf{w}'\|\mathbf{g}\rangle_W \otimes \\
& |D_{k,\mathbf{x},\mathbf{w},\mathbf{x}',\mathbf{w}'}\rangle_K |D_{k',\mathbf{x},\mathbf{w},\mathbf{x}',\mathbf{w}'}\rangle_{K'} \Bigg\| \\
=& \Bigg\| \sum_{\mathbf{x},\mathbf{w},\mathbf{x}',\mathbf{w}'} \alpha_{\mathbf{x},\mathbf{w},\mathbf{x}',\mathbf{w}'} |\mathbf{x}\rangle_X |\mathbf{w}\|\mathbf{x}\|\mathbf{w}'\|\mathbf{g}\rangle_W \otimes \\
& \left(\Pi^{(\mathbf{x},\mathbf{w})} |D_{k,\mathbf{x},\mathbf{w},\mathbf{x}',\mathbf{w}'}\rangle_K\right) \otimes \left(\Pi^{(\mathbf{x}',\mathbf{w}')} |D_{k',\mathbf{x},\mathbf{w},\mathbf{x}',\mathbf{w}'}\rangle_{K'}\right) \Bigg\| \\
=& \left(\sum_{\mathbf{x},\mathbf{w},\mathbf{x}',\mathbf{w}'} |\alpha_{\mathbf{x},\mathbf{w},\mathbf{x}',\mathbf{w}'}|^2 \cdot \right. \\
& \left. \left\| \left(\Pi^{(\mathbf{x},\mathbf{w})} |D_{k,\mathbf{x},\mathbf{w},\mathbf{x}',\mathbf{w}'}\rangle_K\right) \otimes \left(\Pi^{(\mathbf{x}',\mathbf{w}')} |D_{k',\mathbf{x},\mathbf{w},\mathbf{x}',\mathbf{w}'}\rangle_{K'}\right) \right\|^2 \right)^{-1/2} && \text{(Orthogonal states)} \\
\geq & \min_{\mathbf{x},\mathbf{w},\mathbf{x}',\mathbf{w}'} \left\| \left(\Pi^{(\mathbf{x},\mathbf{w})} |D_{k,\mathbf{x},\mathbf{w},\mathbf{x}',\mathbf{w}'}\rangle_K\right) \otimes \left(\Pi^{(\mathbf{x}',\mathbf{w}')} |D_{k',\mathbf{x},\mathbf{w},\mathbf{x}',\mathbf{w}'}\rangle_{K'}\right) \right\| && \text{(Convex combination)} \\
\geq & \min_{\mathbf{x},\mathbf{w},\mathbf{x}',\mathbf{w}'} \left\| \left(\Pi^{(\mathbf{x},\mathbf{w})} C_K |D_{k,\mathbf{x},\mathbf{w},\mathbf{x}',\mathbf{w}'}\rangle_K\right) \otimes \left(\Pi^{(\mathbf{x}',\mathbf{w}')} C_{K'} |D_{k',\mathbf{x},\mathbf{w},\mathbf{x}',\mathbf{w}'}\rangle_{K'}\right) \right\| \\
& - \sqrt{l 2^{-c+1}} && \text{(Fundamental lemma)} \\
\geq & 1 - O(\sqrt{l 2^{-c}}) && \text{(Equation (30))}
\end{aligned}$$

□

Corollary 7.16. *From Lemma 7.15 and Lemma 3.8, we have*

$$\left\|\Pi^{\neg r} O^R \Pi^v\right\| \leq O(\sqrt[4]{l 2^{-c}}).$$

We can continue with the main lemma, which essentially states that feeding an input into the Msponge and then calling find-tail on the state will nearly always return the original input and the correct head. This will allow us to show that the action of

the simulator is essentially to compute f on the original input, and is the technical core. Note that intermediate wires which are later uncomputed are not mentioned in this statement, to prevent clutter.

Note that the formulation of this lemma is slightly different from the formulation in Section 7.3, in that we do not project onto good databases. This is because we require the input state to be valid, and the projector Π^{ig} does not commute with the projector Π^{v}. We instead explicitly write the form of the state on which this lemma will apply, which we will then show are the kinds of states that arise in our experiment.

Lemma 7.17. *Consider a (possibly sub-normalized) state $|\gamma\rangle$ over registers $XKK'H$, which is valid s.t. $\Pi^{v}|\gamma\rangle = |\gamma\rangle$, and whose databases have size at most t s.t. $\Pi^{t}|\gamma\rangle = |\gamma\rangle$. Let $\beta = \left\|\Pi^{\neg\mathsf{ig}}|\gamma\rangle\right\|$. Then we have*

$$\left\|(O^I - O^R)|\gamma\rangle\right\| \leq \tilde{O}\left(\beta + l\sqrt{t^3 2^{-\min(r,c)}} + \sqrt[4]{l 2^{-c}}\right)$$

Proof. First observe that steps (3) and (4) of O^R preserve the computational basis of the X and S registers. They therefore commute with steps (1) and (2), so we can move steps (1) and (2) to be at the end. Let us call this new procedure $O^{R'}$ with steps (1′), ..., (4′). Let us use the letter F to refer to the operator implementing the first step of O^I (which is the same first step as $(O^{R'})$.

Now observe that Π^{r} and Π^{ig} are diagonal in the computational basis, meaning they commute. This implies that $\Pi^{r\cap\mathsf{ig}} = \Pi^{r}\Pi^{\mathsf{ig}}$, i.e. the projector onto recorded and good databases is the product of the projector onto good with the projector onto recorded. We can write

$$\begin{aligned} F|\gamma\rangle &= \Pi^{r\cap\mathsf{ig}}F|\gamma\rangle + \Pi^{\neg r\cup\neg\mathsf{ig}}F|\gamma\rangle \\ &= \Pi^{r\cap\mathsf{ig}}F|\gamma\rangle + \underbrace{\Pi^{\neg r}F|\gamma\rangle}_{|\xi_1\rangle} + \underbrace{\Pi^{r\cap\neg\mathsf{ig}}F|\gamma\rangle}_{|\xi_2\rangle}. \end{aligned}$$

We have that $\||\xi_1\rangle\| \leq \tilde{O}(\sqrt[4]{l2^{-c}})$ from Lemma 7.15, and $\||\xi_2\rangle\| \leq \tilde{O}(\beta + l\sqrt{t^3 2^{-\min(r,c)}})$ from Lemma 5.18. It follows that there exists a state $|\xi\rangle$ of norm $O(\beta + l\sqrt{t^3 2^{-\min(r,c)}} + \sqrt[4]{l2^{-c}})$ s.t.

$$F|\gamma\rangle = \Pi^{r\cap\mathsf{ig}}F|\gamma\rangle + |\xi\rangle.$$

Observe that the action of steps (2′, 3′, 4′) of $O^{R'}$ and step (2) of O^I act identically on states within the $\Pi^{r\cap\mathsf{ig}}$ subspace; this is because such states have $\mathbf{x}$ as the unique recorded tail of the z value appearing as the final state. In other words, $O^I F^\dagger$ and $O^R F^\dagger$ act identically on such states.

From the fact that unitaries preserve distance, we then have

$$\begin{aligned} \left\|O^I|\gamma\rangle - O^R|\gamma\rangle\right\| &\leq \left\|O^I F^\dagger \Pi^{r\cap\mathsf{ig}}F|\gamma\rangle - O^R F^\dagger \Pi^{r\cap\mathsf{ig}}F|\gamma\rangle\right\| + \left\|O^I F^\dagger \Pi^{\neg r\cup\neg\mathsf{ig}}F|\gamma\rangle - O^R F^\dagger \Pi^{\neg r\cup\neg\mathsf{ig}}F|\gamma\rangle\right\| \\ &\leq \underbrace{\left\|\Pi^{r\cap\mathsf{ig}}F|\gamma\rangle - \Pi^{r\cap\mathsf{ig}}F|\gamma\rangle\right\|}_{=0} + 2\||\xi\rangle\| \\ &\leq \tilde{O}\left(\beta + \sqrt{t^3 2^{-\min(r,c)}} + \sqrt[4]{l2^{-c}}\right). \end{aligned}$$

□

Preserving the valid subspace. We will often use the fact that the state in the consistency experiment is close to valid. Here we justify that assumption. The first and easy case is interactions via compressed oracle calls cO. As was shown in Corollary 4.6, these preserve the valid subspace. The only other way in which the simulator and adversary interact with the compressed databases is via the $\mathsf{find\text{-}tail}$ operation. When we say a state is "valid", here we mean that all the compressed databases for K, K', H, F are valid.

Showing that the $\mathsf{find\text{-}tail}$ operation essentially preserves validity on good databases will be somewhat more involved. We will often refer back to the formal definition of $\mathsf{find\text{-}tail}$, Definition 7.1.

Lemma 7.18. *Let $|\psi\rangle_{ZTSKK'}$ be a valid state be such that $\left\|\Pi^{\neg \mathsf{ig}}_{KK'} |\psi\rangle_{ZTSKK'}\right\| = \beta$. Then we have*

$$\|\Pi^{\neg v}_{KK'} \mathsf{fT} |\psi\rangle\| \leq \tilde{O}(t\beta + \sqrt{t^5 2^{-\min(r,c)}})$$

Proof. Let us say that the input vector $\mathbf{x}$ of a database D_k is the set of $\mathbf{x}$ values on which D_k is defined, and similarly $\mathbf{x}'$ for $D_{k'}$. The output vector $\mathbf{y}$ and $\mathbf{y}'$ are similarly defined by the outputs of D_k and $D_{k'}$ respectively. Databases with different input vectors are clearly orthogonal. Now observe that $\Pi^{\neg v}$ and fT preserve the input vectors of D_k and $D_{k'}$, as well as the value of z. We can write

$$|\psi\rangle = \sum_{\mathbf{x},\mathbf{x}',z} \alpha_{\mathbf{x},z} |\psi_{\mathbf{x},\mathbf{x}',z}\rangle,$$

where $|\psi_{\mathbf{x},\mathbf{x}',z}\rangle$ is an arbitrary state with input vectors $\mathbf{x}, \mathbf{x}'$ for registers K, K' and fixed value z for register Z. From the previous observation, we have

$$\|\Pi^{\neg v}_{KK'} \mathsf{fT} |\psi\rangle\| = \sqrt{\sum_{\mathbf{x},\mathbf{x}',z} |\alpha_{\mathbf{x},z}|^2 \left\|\Pi^{\neg v}_{KK'} \mathsf{fT} |\psi_{\mathbf{x},\mathbf{x}',z}\rangle\right\|^2}.$$

Let us now suppose the following claim, which we defer the proof of.

Claim 7.19. *Let $|\psi_{\mathbf{x},\mathbf{x}',z}\rangle \in \Pi^{v}_{KK'}$ be as above with fixed input vectors and z value, such that $\left\|\Pi^{\neg \mathsf{ig}}_{KK'} |\psi_{\mathbf{x},\mathbf{x}',z}\rangle\right\| = \beta$. Then we have*

$$\|\Pi^{\neg v}_{KK'} \mathsf{fT} \psi_{\mathbf{x},\mathbf{x}',z}\| \leq \tilde{O}(t\beta + \sqrt{t^5 2^{-\min(r,c)}}).$$

Assuming such a claim, we can write

$$\begin{aligned}
\|\Pi^{\neg v}_{KK'} \mathsf{fT} |\psi\rangle\| =& \sqrt{\sum_{\mathbf{x},\mathbf{x}',z} |\alpha_{\mathbf{x},z}|^2 \left\|\Pi^{\neg v}_{KK'} \mathsf{fT} |\psi_{\mathbf{x},\mathbf{x}',z}\rangle\right\|^2} \\
\leq& \tilde{O}\left(\sqrt{\sum_{\mathbf{x},\mathbf{x}',z} |\alpha_{\mathbf{x},z}|^2 (t \left\|\Pi^{\neg \mathsf{ig}}_{KK'} |\psi_{\mathbf{x},\mathbf{x}',z}\rangle\right\|)^2} + \sqrt{\sum_{\mathbf{x},\mathbf{x}',z} |\alpha_{\mathbf{x},z}|^2 t^5 2^{-\min(r,c)}}\right) \\
\leq& \tilde{O}\left(t \left\|\Pi^{\neg \mathsf{ig}}_{KK'} |\psi\rangle\right\| + \sqrt{t^5 2^{-\min(r,c)}}\right).
\end{aligned}$$

It simply remains to prove the claim. □

Proof of Claim 7.19. Consider a state of the form

$$|\psi_{\mathbf{x},\mathbf{x}',z}\rangle = \sum_{\mathbf{y},\mathbf{y}',tl,s} \alpha_{\mathbf{y},\mathbf{y}',tl,s} \, |D_k[\mathbf{x} \to \mathbf{y}]\rangle_K \, |D_{k'}[\mathbf{x}' \to \mathbf{y}']\rangle_{K'} \, |z\rangle_Z \, |tl\rangle_T \, |s\rangle_S \,, \tag{31}$$

where $\mathbf{x}, \mathbf{x}'$, and z are fixed strings. For ease of notation, for most of this proof we will drop the subscripts on $|\psi\rangle$. Recalling that database validity means every output has zero overlap with the uniform superposition, we can write the validity condition in this notation as

$$\forall tl, s, \mathbf{x}, \mathbf{x}', \mathbf{y}, \mathbf{y}' : \sum_{y_i} \alpha_{\mathbf{y}|_{y_i},\mathbf{y}',tl,s} = 0,$$
$$\sum_{y'_i} \alpha_{\mathbf{y},\mathbf{y}'|_{y'_i},tl,s} = 0,$$

where the notation $\mathbf{y}|_{y_i}$ means replacing the i-th index of $\mathbf{y}$ with y_i. Now let us define $\Pi^{\neg v,x}$ as the projector onto span of databases that are invalid on input x. For a general database, this projector can be written as

$$\Pi^{\neg v,x} = \sum_{D, x \notin D} \frac{1}{N} \sum_{u,v} |D[x \to u]\rangle \langle D[x \to v]| \,.$$

In our case, we will use $\Pi_K^{\neg v,x}$ to denote such a projector on the K register and $\Pi_{K'}^{\neg v,x'}$ to denote such a projector on the K' register, with the subscripts sometimes dropped. Observe that in general

$$\|\Pi_{KK'}^{\neg v} |\phi\rangle\| \leq \sum_x \|\Pi_K^{\neg v,x} |\phi\rangle\| + \sum_{x'} \left\|\Pi_{K'}^{\neg v,x'} |\phi\rangle\right\| .$$

For a state of the form in Equation (31), at most t of these terms will be non-zero, corresponding to the terms in the input vectors. Let us analyze the norm of one such term, say corresponding to $\Pi_K^{\neg v,x_1}$. Note that this operation (as well as fT) preserves all other input-output points, except the value of y_1. We will for convenience simply use D_k and $D_{k'}$ to refer to the databases, and x and y to refer to the selected input-output pair. We can write

$$|\psi^g\rangle = \sum_{\mathbf{y},\mathbf{y}',tl,s;[\mathbf{x}\to(\perp,y_2,\dots)]_K,[\mathbf{x}'\to\mathbf{y}']_{K'} \text{ is good}} \alpha_{\mathbf{y},\mathbf{y}',tl,s} \, |D_k[\mathbf{x} \to \mathbf{y}]\rangle_K \, |D_{k'}[\mathbf{x}' \to \mathbf{y}']\rangle_{K'} \, |z\rangle_Z \, |tl\rangle_T \, |s\rangle_S$$
$$|\psi^b\rangle = \sum_{\mathbf{y},\mathbf{y}',tl,s;[\mathbf{x}\to(\perp,y_2,\dots)]_K,[\mathbf{x}'\to\mathbf{y}']_{K'} \text{ is bad}} \alpha_{\mathbf{y},\mathbf{y}',tl,s} \, |D_k[\mathbf{x} \to \mathbf{y}]\rangle_K \, |D_{k'}[\mathbf{x}' \to \mathbf{y}']\rangle_{K'} \, |z\rangle_Z \, |tl\rangle_T \, |s\rangle_S \,,$$

or in words $|\psi^g\rangle$ are the terms in which the database with x assigned to $\perp$ is good, and $|\psi^b\rangle$ are the terms in which the database with x assigned to $\perp$ is bad. Observe that we have

$$\begin{aligned}
\|\Pi^{\neg v,x}\mathsf{fT} |\psi\rangle\| &= \left\|\Pi^{\neg v,x}\mathsf{fT}(|\psi^g\rangle + |\psi^b\rangle)\right\| \\
&\leq \|\Pi^{\neg v,x}\mathsf{fT} |\psi^g\rangle\| + \left\|\Pi^{\neg v,x}\mathsf{fT} |\psi^b\rangle\right\| && \text{(Triangle inequality)} \\
&\leq \|\Pi^{\neg v,x}\mathsf{fT} |\psi^g\rangle\| + \beta && \text{(Monotonicity of bad)}
\end{aligned}$$

Note that the last inequality follows because adding a new input-output pair cannot convert a bad database into a good one; hence every term with non-zero amplitude in $|\psi^b\rangle$ is bad. It therefore remains to bound the first term. Observe that $|\psi^g\rangle$ is valid on input x, meaning $\Pi^{\neg v,x}|\psi^g\rangle = 0$. To see this, we can write the amplitudes

$$|\psi^g\rangle = \sum_{\mathbf{y},\mathbf{y}',tl,s} \beta_{\mathbf{y},\mathbf{y}',tl,s} |D_k[\mathbf{x}\to\mathbf{y}]\rangle_K |D_{k'}[\mathbf{x}'\to\mathbf{y}']\rangle_{K'} |z\rangle_Z |tl\rangle_T |s\rangle_S$$

$$\beta_{\mathbf{y},\mathbf{y}',tl,s} := \begin{cases} \alpha_{\mathbf{y},\mathbf{y}',tl,s} & ([\mathbf{x}\to(\perp,y_2,\dots)]_K, [\mathbf{x}'\to\mathbf{y}']_{K'} \text{ is good}) \\ 0 & ([\mathbf{x}\to(\perp,y_2,\dots)]_K, [\mathbf{x}'\to\mathbf{y}']_{K'} \text{ is bad}) \end{cases}$$

and we can write the sums

$$\begin{aligned}\sum_{y_1} \beta_{\mathbf{y}|_{y_1},\mathbf{y}',tl,s} &= \begin{cases} \sum_{y_1} \alpha_{\mathbf{y}|_{y_1},\mathbf{y}',tl,s} & ([\mathbf{x}\to(\perp,y_2,\dots)]_K, [\mathbf{x}'\to\mathbf{y}']_{K'} \text{ is good}) \\ 0 & ([\mathbf{x}\to(\perp,y_2,\dots)]_K, [\mathbf{x}'\to\mathbf{y}']_{K'} \text{ is bad}) \end{cases} \\ &= 0.\end{aligned}$$

Continuing on, denote by t_y the output (first) tail on $D_k[x\to y], D_{k'}$. Let S denote the possible values of y such that the tail of z will be different on $D_k[x\to y], D_{k'}$ than on $D_k[x\to\perp], D_{k'}$. Note that by Lemma 5.19, we have $|S| = O(t^3(n+2^{c-r}))$. Finally, observe that $\Pi^{\neg v,x}$ annihilates databases not defined on x, so we may drop such terms. We now split into two cases.

(1) Suppose that z has a tail $t_\perp$ in $D_k[x\to\perp], D_{k'}$. It follows that, for any y, z will have a tail in $D_k[x\to y], D_{k'}$. We have

$$\begin{aligned}
&\left\|\Pi_K^{\neg v,x_1}\mathsf{fT}|\psi\rangle\right\| \\
=&\left\|\Pi^{\neg v,x_1}\sum_{y,tl,s}\alpha_{y,tl,s}|D_k[x\to y]\rangle_K|D_{k'}\rangle_{K'}|tl\oplus t_y\rangle_T|s\oplus 1\rangle_S\right\| \\
=&\left\|\sum_{y,u,tl,s}\alpha_{y,tl,s}2^{-c}|D_k[x\to u]\rangle_K|D_{k'}\rangle_{K'}|tl\oplus t_y\rangle_T|s\oplus 1\rangle_S\right\| \\
=&\left\|\sum_{u,tl,s}2^{-c}\left(\sum_{y\notin S}\alpha_{y,tl\oplus t_\perp,s\oplus 1}+\sum_{y\in S}\alpha_{y,tl\oplus t_y,s\oplus 1}\right)|u\rangle|tl\rangle|s\rangle\right\| \\
\le&\left\|\sum_{u,tl,s}2^{-c}\left(-\sum_{y\in S}\alpha_{y,tl\oplus t_\perp,s\oplus 1}+\sum_{y\in S}\alpha_{y,tl\oplus t_y,s\oplus 1}\right)|u\rangle|tl\rangle|s\rangle\right\| && \text{(Validity of } |\psi\rangle\text{)} \\
\le&\left\|\sum_{u,tl,s}2^{-c}\sqrt{2|S|\left(\sum_{y\in S}|\alpha_{y,tl\oplus t_\perp,s\oplus 1}|^2+\sum_{y\in S}|\alpha_{y,tl\oplus t_y,s\oplus 1}|^2\right)}|u\rangle|tl\rangle|s\rangle\right\| && (L_2 \text{ bound on } L_1) \\
=&2^{-c}\sqrt{2|S|\sum_{y\in S,u,tl,s}\left(|\alpha_{y,tl\oplus t_\perp,s\oplus 1}|^2+|\alpha_{y,tl\oplus t_y,s\oplus 1}|^2\right)} \\
\le&2^{-c}\sqrt{2|S|\sum_u 2} && \text{(Normalization of } |\psi\rangle\text{)} \\
\le&\sqrt{4|S|2^{-c}} && \text{(Cauchy-Schwarz)} \\
\le&\tilde{O}(\sqrt{t^3 2^{-\min(r,c)}}).
\end{aligned}$$

(2) Suppose that z has no tail in $D_k[x \to \bot], D_{k'}$. It follows by definition of S that z has a tail only when $y \in S$. We write

$$
\begin{aligned}
&\left\|\Pi_K^{\neg v,x_1}\mathsf{fT}\left|\psi\right\rangle\right\| \\
=&\left\|\Pi^{\neg v,x_1}\sum_{y,tl,s}\alpha_{y,tl,s}\left|D_k[x\to y]\right\rangle_K\left|D_{k'}\right\rangle_{K'}\left|tl\oplus t_y\right\rangle_T\left|s\oplus\mathbb{I}[y\in S]\right\rangle_S\right\| \\
=&\left\|\sum_{y,u,tl,s}\alpha_{y,tl,s}2^{-c}\left|D_k[x\to u]\right\rangle_K\left|D_{k'}\right\rangle_{K'}\left|tl\oplus t_y\right\rangle_T\left|s\oplus\mathbb{I}[y\in S]\right\rangle_S\right\| \\
=&\left\|\sum_{u,tl,s}2^{-c}\left(\sum_{y\notin S}\alpha_{y,tl,s\oplus 1}+\sum_{y\in S}\alpha_{y,tl\oplus t_y,s}\right)\left|u\right\rangle\left|tl\right\rangle\left|s\right\rangle\right\| \\
\le&\left\|\sum_{u,tl,s}2^{-c}\left(-\sum_{y\in S}\alpha_{y,tl,s\oplus 1}+\sum_{y\in S}\alpha_{y,tl\oplus t_y,s}\right)\left|u\right\rangle\left|tl\right\rangle\left|s\right\rangle\right\| && \text{(Validity of } |\psi\rangle\text{)} \\
\le&\left\|\sum_{u,tl,s}2^{-c}\sqrt{2|S|\left(\sum_{y\in S}|\alpha_{y,tl,s\oplus 1}|^2+\sum_{y\in S}|\alpha_{y,tl\oplus t_y,s}|^2\right)}\left|u\right\rangle\left|tl\right\rangle\left|s\right\rangle\right\| && (L_2 \text{ bound on } L_1) \\
=&2^{-c}\sqrt{2|S|\sum_{y\in S,u,tl,s}\left(|\alpha_{y,tl,s\oplus 1}|^2+|\alpha_{y,tl\oplus t_y,s}|^2\right)} \\
\le&2^{-c}\sqrt{2|S|\sum_u 1} && \text{(Normalization of } |\psi\rangle\text{)} \\
\le&\sqrt{2|S|2^{-c}} && \text{(Cauchy-Schwarz)} \\
\le&\tilde{O}(\sqrt{t^3 2^{-\min(r,c)}}).
\end{aligned}
$$

Putting everything together, we have

$$\Pi_K^{\neg v,x_1}\mathsf{fT}\left|\psi_{\mathbf{x},\mathbf{x}',z}\right\rangle \le \tilde{O}(\beta+\sqrt{t^3 2^{-\min(r,c)}}).$$

A similar argument implies the same bound for any $x_i \in \mathbf{x}$ and $x_i' \in \mathbf{x}'$, so we have

$$
\begin{aligned}
\left\|\Pi_{K,K'}^{\neg v}\mathsf{fT}\left|\psi_{\mathbf{x},\mathbf{x}',z}\right\rangle\right\| &\le \sum_{x_i\in\mathbf{x}}\left\|\Pi_K^{\neg v,x_i}\mathsf{fT}\left|\psi_{\mathbf{x},\mathbf{x}',z}\right\rangle\right\| + \sum_{x_i'\in\mathbf{x}'}\left\|\Pi_{K'}^{\neg v,x_i'}\mathsf{fT}\left|\psi_{\mathbf{x},\mathbf{x}',z}\right\rangle\right\| \\
&\le \tilde{O}(t\beta+\sqrt{t^5 2^{-\min(r,c)}}).
\end{aligned}
$$

□

Putting the pieces together. We are now ready to argue that answering a query using the simulated Msponge (i.e. $\mathsf{MSp}^{\mathcal{S}^f}$) and using the ideal functionality (i.e. f) are indistinguishable. We will show that this is the case for any state which has a bounded number of input output points in each database, and is close to both valid and ideal good.

Lemma 7.20. *Consider a state $|\psi\rangle$ on adversary register A and database registers K, K', H, F that satisfies*

$$\left\|\Pi^{\neg t}\left|\psi\right\rangle\right\| = 0 \qquad \left\|\Pi^{\neg v}\left|\psi\right\rangle\right\| = \gamma \qquad \left\|\Pi^{\neg \mathsf{ig}}\left|\psi\right\rangle\right\| = \beta.$$

If we define the ideal and real states after an l-block query to the sponge, we may write

$$\begin{aligned}|\psi^I\rangle :=& \mathsf{cO}_{AF} A\,|\psi\rangle\\ |\psi^R\rangle :=& \mathsf{Sp}^{\mathcal{S}^f}_{AHKK'F} A\,|\psi\rangle\,,\end{aligned}$$

where here the isometry A creates the workspace used in the out-of-place sponge circuit—which is untouched in the ideal $|\psi^I\rangle$, and (approximately) uncomputed in the real $|\psi^R\rangle$. Then we have

$$\left\| |\psi^I\rangle - |\psi^R\rangle \right\| \leq \tilde{O}(\beta + \gamma + l\sqrt{t^3 2^{-\min(r,c)}} + \sqrt[4]{l 2^{-c}}).$$

Proof. Recalling that we break the adversary state A into an X and Y component (plus leftover storage), and using X to denote a bitflip and $XOR_{A\to B}$ to denote bitwise xor from A to B, we have

$$\begin{aligned}|\psi^I\rangle =& U^\dagger X_S XOR_{Hd\to T} XOR_{X\to T} \mathsf{cO}_{XYF} O^I\,|\psi\rangle\\ |\psi^R\rangle =& U^\dagger \mathsf{fT}_{KK'TS} \mathcal{S}^h_{XYHTSF} O^R\,|\psi\rangle\,.\end{aligned}$$

Note that we are being somewhat sloppy with notation in the second equation; in fact, the calls to find-tail made by $\mathcal{S}^h$ are here being absorbed into O^R and the latter fT call.

We have

$$|\phi\rangle = \Pi^v\,|\psi\rangle\,, \qquad\qquad \||\psi\rangle - |\phi\rangle\| = \gamma, \tag{32}$$

so we will consider instead the states

$$\begin{aligned}|\phi^I\rangle =& U^\dagger X_S XOR_{Hd\to T} XOR_{X\to T} \mathsf{cO}_{XYF} O^I\,|\phi\rangle\\ |\phi^R\rangle =& U^\dagger \mathsf{fT}_{KK'TS} \mathcal{S}^h_{XYHTF} XOR_{H\to Y} O^R\,|\phi\rangle\,.\end{aligned}$$

From the fact that $|\phi\rangle$ is valid, we have

$$\left\| O^I\,|\phi\rangle - O^R\,|\phi\rangle \right\| \leq \tilde{O}\left(\beta + l\sqrt{t^3 2^{-\min(r,c)}} + \sqrt[4]{l 2^{-c}}\right) \quad \text{(From Lemma 7.17).} \tag{33}$$

Observe that we have

$$\mathcal{S}^h_{XYHTF} XOR_{Hd\to Y} O^I\,|\phi\rangle = \mathsf{cO}_{XYF} O^I\,|\phi\rangle\,. \tag{34}$$

To see this, recall that O^I simply places the input X into the tail register T, and sets the flag S to be success. On states of this form, the simulator will implement the call to h via a (compressed oracle) call to f, and then XOR in the recovered head. The head Hd is already XORed into the Y register, and so the two cancel out and the remaining operation is simply an oracle call to f on input X and output Y. Finally, we have

$$\left\| \mathsf{fT}_{KK'TS} \mathsf{cO}_{XYF} O^I\,|\phi\rangle - X_S XOR_{Hd\to T} XOR_{X\to T} \mathsf{cO}_{XYF} O^I\,|\phi\rangle \right\| \leq O(\sqrt[4]{l 2^{-c}}) \tag{35}$$

from Lemma 7.15. We can therefore write

$$\begin{aligned}
\left\| |\psi^I\rangle - |\psi^R\rangle \right\| \leq & \left\| |\phi^I\rangle - |\psi^I\rangle \right\| + \left\| |\phi^R\rangle - |\psi^R\rangle \right\| + \left\| |\phi^I\rangle - |\phi^R\rangle \right\| && \text{(Triangle Inequality)} \\
\leq & 2\gamma + \left\| \left(O^I - O^R\right) |\phi\rangle \right\| + \left\| \mathcal{S}^h XOR_{Hd\to Y} O^I |\phi\rangle - \mathsf{cO}_{XYF} O^I |\phi\rangle \right\| + \\
& \left\| X_S XOR_{Hd\to T} XOR_{X\to T} \mathsf{cO}_{XYF} O^I |\phi\rangle - \right. \\
& \left. \mathsf{fT}_{KK'TS} \mathsf{cO}_{XYF} O^I |\phi\rangle \right\| && \text{(Triangle Inequality)} \\
\leq & \tilde{O}\left(\gamma + \beta + l\sqrt{t^3 2^{-\min(r,c)}} + \sqrt[4]{l 2^{-c}}\right) && \text{(Equations (32) to (35))}
\end{aligned}$$

□

With this in place, we can argue that the simulator we describe is consistent.

Theorem 7.21. *The simulator described in Section 7.1 is consistent. In particular, for a q-query distinguisher which makes queries of block length at most l we have*

$$\left|\Pr[\mathcal{A}^{\pi,\pi^{-1},\mathcal{S}^f,f}() = 1] - \Pr[\mathcal{A}^{\pi,\pi^{-1},\mathcal{S}^f,\mathsf{Sp}^{\mathcal{S}^f}}() = 1]\right| = O\left(\sqrt{q^9 2^{-\min(r,c)}} + l\sqrt[4]{q^5 2^{-\min(r,c)}}\right)$$

Proof. We will refer to the left world as the ideal (i.e. the one with $\mathcal{A}^{\pi,\pi^{-1},\mathcal{S}^f,f}()$), and the right as real. Observe that the oracle calls to $\mathsf{Sp}^{\mathcal{S}^f}$, as depicted in Figure 4, use intermediate workspace initialized to a fixed 0 state. We will consider the experiment where this workspace is maintained in a separate garbage register G for the remainder of the real experiment. In the ideal experiment, we simply initialize a corresponding amount of auxiliary space in the fixed 0 state. The isometry which corresponds to this is A, as in Definition 7.11. In other words, we purify both experiments; we will show that the purified states remain close in between queries.

The initial state for each experiment is the same,

$$|\psi^0\rangle := |\alpha^0\rangle_A |\emptyset\rangle_H |\emptyset\rangle_K |\emptyset\rangle_{K'} |\emptyset\rangle_F |\emptyset\rangle_G ,$$

where $|\alpha^0\rangle$ is an arbitrary initial state of the adversary. As in the proof of Theorem 7.10, let us consider in fact a strengthened adversary which is time unbounded, and which holds the entire truth table of π. The only constraint we require is that π is good, which happens with probability $1 - O(2^{-n})$. Our proof will show that for any fixed, good π, the final real and ideal states are close. We will let l be an upper bound on the block length of any query to the sponge/ideal functionality.

We can WLOG take our adversary to alternate queries to the $\mathcal{S}^f$ oracle and the hash function (f in ideal, and $\mathsf{Sp}^{\mathcal{S}^f}$ in real), increasing the total number of queries by at most 2. We can parameterize a q-query adversary of this form by unitaries $W^1, \ldots, W^q$ acting on the internal register A. The final state in the real experiment is given by

$$|\psi^R\rangle := W^q_A \ldots W^2_A \mathsf{Sp}^{\mathcal{S}^f}_{AHKK'FG} A_G W^1_A \mathcal{S}^f_{AHKK'F} |\psi^0\rangle .$$

The final state in the ideal is given by

$$|\psi^I\rangle := W_A^q \dots W_A^2 \mathsf{cO}_{AF} A_G W_A^1 \mathcal{S}^f_{AHKK'F} |\psi^0\rangle .$$

We define the "hybrid" states $|\psi_t^H\rangle$, that begins like the ideal world but switches after the t-th query to answering using the procedure from the real world. We have

$$|\psi_t^H\rangle := W^q \dots \mathsf{Sp}^{\mathcal{S}^f}_{AHKK'FG} A_G W_A^{2t+1} \mathcal{S}^f_{AHKK'F} W_A^{2t} \mathsf{cO}_{AF} A_G W_A^{2t-1} \dots W_A^2 \mathsf{cO}_{AF} A_G W_A^1 \mathcal{S}^f_{AHKK'F} |\psi^0\rangle ,$$

where $|\psi_0^H\rangle = |\psi^I\rangle$ and $|\psi_{t/2}^H\rangle = |\psi^R\rangle$. Let us also define the intermediate states in the ideal world,

$$|\psi_t^I\rangle := \mathsf{cO}_{AF} A_G W_A^{2t-1} \dots W_A^2 \mathsf{cO}_{AF} A_G W_A^1 \mathcal{S}^f_{AHKK'F} |\psi^0\rangle ,$$

in other words the ideal states right after the $t+1$-th query to the hash function. Note that we have $|\psi_{q/2}^I\rangle = |\psi^I\rangle$, by definition. Observe that we have

$$\left\| \Pi^{\neg \mathsf{ig}} |\psi_t^I\rangle \right\| \leq \tilde{O}(\sqrt{t^5 2^{-\min(r,c)}}) \qquad \text{(By Remark 7.4)} \tag{36}$$

$$\left\| \Pi^{\neg v} |\psi_t^I\rangle \right\| \leq \tilde{O}(\sqrt{t^7 2^{-\min(r,c)}}) \qquad \text{(By Lemma 7.18)} \tag{37}$$

The first inequality comes from the fact that the find-tail operation preserves the computational basis for $D_k, D_{k'}, D_h$, and hence preserves the good subspace; the only other interaction with these databases is via compressed oracle calls. The second inequality comes from the fact that only $O(t)$ calls to find-tail are made to reach $|\psi_t^I\rangle$, and the bad component on each call can be bounded by the first inequality. This is the only operation which does not preserve the valid subspace.

Now observe that

$$\left\| |\psi_t^H\rangle - |\psi_{t-1}^H\rangle \right\| = \left\| |\psi_t^I\rangle - \mathsf{Sp}^{\mathcal{S}^f}_{AHKK'FG} A_G W_A^{2t+1} \mathcal{S}^f_{AHKK'F} W_A^{2t} |\psi_{t-1}^I\rangle \right\| ,$$

following from unitary and isometry invariance of norms. If we define

$$|\phi_t\rangle = W_A^{2t+1} \mathcal{S}^f_{AHKK'F} W_A^{2t} |\psi_{t-1}^I\rangle ,$$

this can be written as

$$\begin{aligned} \left\| |\psi_t^H\rangle - |\psi_{t-1}^H\rangle \right\| &= \left\| \mathsf{cO}_{AF} A_G |\phi\rangle - \mathsf{Sp}^{\mathcal{S}^f}_{AHKK'FG} A_G |\phi\rangle \right\| \\ &\leq \tilde{O}\left(\sqrt{t^7 2^{-\min(r,c)}} + l\sqrt{t^3 2^{-\min(r,c)}} + \sqrt[4]{l 2^{-c}} \right) \quad \text{(By Lemma 7.20).} \end{aligned} \tag{38}$$

where we observe that $|\phi_t\rangle$ satisfies Equations (36) and (37) for the same reason that $|\psi_t^I\rangle$ does. We can then write

$$\begin{aligned} \left\| |\psi^R\rangle - |\psi^I\rangle \right\| &\leq \sum_{t=1}^{q/2} \left\| |\psi_t^H\rangle - |\psi_{t-1}^H\rangle \right\| && \text{(Triangle inequality)} \\ &\leq O\left(\sqrt{q^9 2^{-\min(r,c)}} + l\sqrt{q^5 2^{-\min(r,c)}} + \sqrt[4]{l q^4 2^{-c}} \right) && \text{(By Equation (38))} \\ &\leq O\left(\sqrt{q^9 2^{-\min(r,c)}} + l\sqrt[4]{q^5 2^{-\min(r,c)}} \right) \end{aligned}$$

□

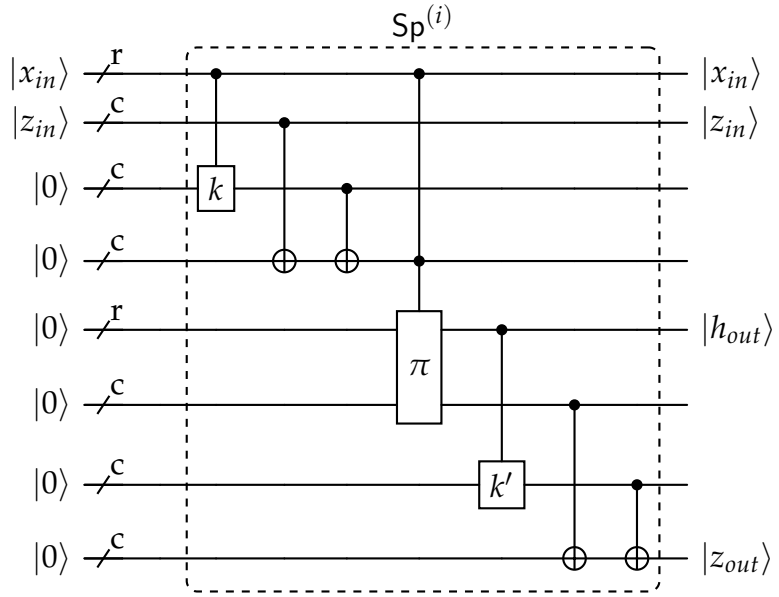

Figure 2: An out-of-place quantum circuit for intermediate sponge rounds.

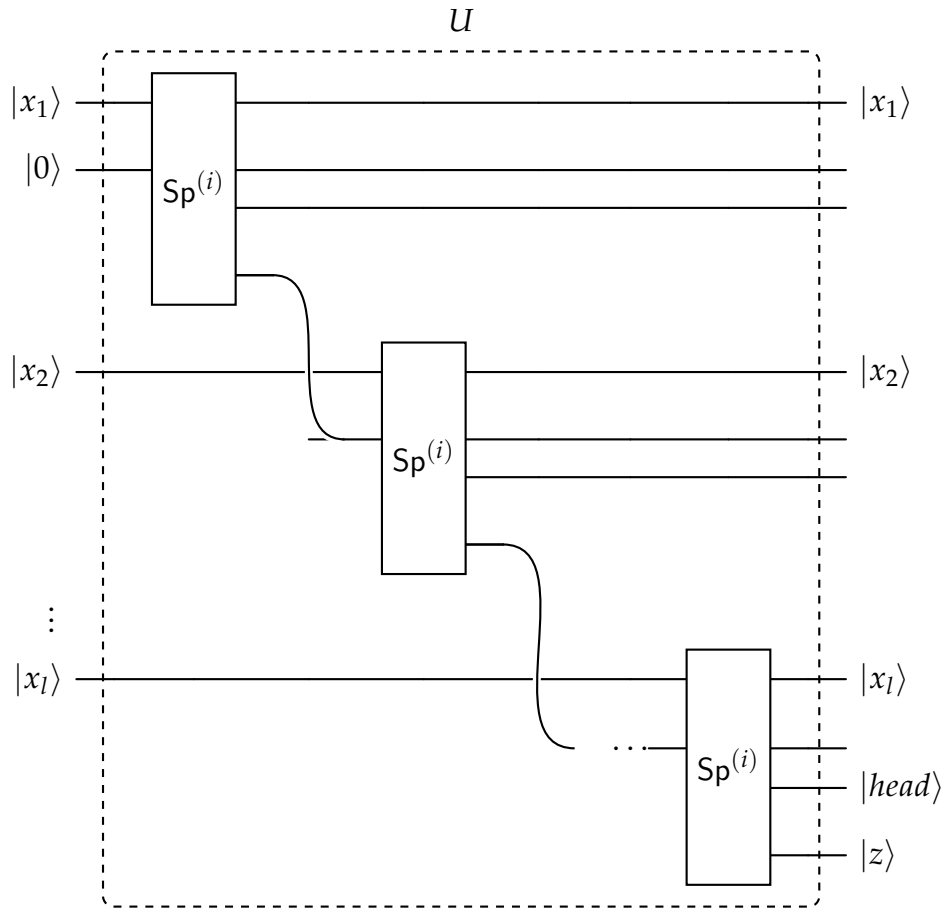

Figure 3: Circuit U, representing an out-of-place quantum circuit for computing the sponge state before the h call. Because we consider the purified experiment, uncomputed workspace is not discarded.

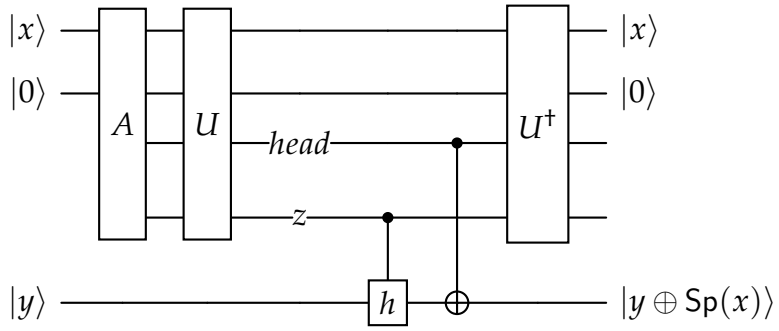

Figure 4: An out-of-place quantum circuit for computing the Msponge. Because we consider the purified experiment, uncomputed workspace is not discarded.

7.5 Main theorem

Combining the results of Sections 7.3 and 7.4, we obtain a bound as follows.

Theorem 7.22. *There exists an efficient simulator $\mathcal{S}$ for the sponge construction such that for all adversaries $\mathcal{A}$ making q queries of block length at most l, they can distinguish with advantage*

$$\left\|\Pr[\mathcal{A}^{\varphi,\varphi^{-1},\mathsf{Sp}^{\varphi}}() = 1 - \mathcal{A}^{\mathcal{S}^{f},f}() = 1]\right\| = O\left(l^2\sqrt{q^9 2^{-\min(r,c)}} + l^3\sqrt[4]{q^5 2^{-\min(r,c)}}\right).$$

Proof. It follows from Theorems 7.10 and 7.21 and Lemma 3.12 that the Msponge is indifferentiable up to a bound $O\left(\sqrt{q^9 2^{-\min(r,c)}} + l\sqrt[4]{q^5 2^{-\min(r,c)}}\right)$, in the model where both the simulator and the adversary query a truly random permutation π, and the simulator provides oracles for random functions h, k, k'. To obtain a computationally bounded simulator, the simulator can construct a quantum secure psuedorandom permutation [Zha16] in place of π, and we maintain security against computationally bounded adversaries. In the computationally unbounded setting, the simulator may use something sufficiently close to a q-wise independent permutation, such as a $\mathsf{poly}(q)$ round unbalanced Feistel cipher. Note also that our indifferentiability result does not directly consider superposition queries over different lengths. However, this can be handled by Lemma 3.7, noting that the query operators will be controlled on the length control register, and therefore have orthogonal ranges and domains.

Plugging into Lemma 5.4, we obtain an indifferentiability result for the proper sponge with a single round of squeezing and inputs from $(\{0,1\}^r)^*$, with an additional factor $O(l)$. In turn plugging this into Claim 3.15, we incur yet another factor $O(l)$, but now obtain an indifferentiability result of the full sponge construction with any valid PAD function. □

7.6 On the gap in Merkle-Damgård indifferentiability

In the original indifferentiability proof of Merkle-Damgård [Zha19], the validity of compressed databases throughout the experiment is not explicitly considered. Recall that validity is the property that decompressing on any input leads to a well defined output. By Corollary 4.6 (or analogous result in [Zha19]), interacting with compressed databases through compressed oracle calls perfectly preserves validity. However, subroutines which inspect the database in a more direct way, such as find-tail in this work and find-input in the Merkle-Damgård proof, do not perfectly preserve validity. In addition, projecting onto good databases is diagonal in the computational basis of the database register, and projecting onto valid databases is diagonal in the fourier basis; hence, these projectors also do not commute.

To see why this is a problem, consider for instance the simpler two-round domain extender of Section 5, in particular Lemma 16, of [Zha19]. The setting of this lemma is a hybrid in which certain bad events, namely round function collisions, have been projected out, and further the simulator has potentially made many calls to find-input while answering previous queries. The proof of Lemma 16 in [Zha19] implicitly assumes that, after decompressing on input x_1, there is an output pair in the database. This is necessary because the simulator must later recover the input to answer consistently. The prior calls to find-input and projections onto good databases however break this assumption, making the database not fully valid.

However, we believe this gap is fixable. Using an argument similar to that in Section 7.4, one could likely bound how much find-input deviates from the valid subspace. Observe that the good projector also negligibly disturbs the state, so it cannot dramatically leave the valid subspace. Applying this fix directly would almost certainly result in a looser bound: because one cannot simultaneously project onto good and valid databases, it seems one has to pay for both the bad and the invalid component on each query. Indeed, this is what happens in our Theorem 7.21, and is one of the reasons why our indifferentiability bound is weaker than our direct bounds for collision and preimage resistance presented in Section 6.

A Deferred proofs

A.1 Indifferentiability

Restatement of Lemma 7.6:

Lemma A.1. *The commutator between the isometry V and the local compression operators on K' almost commute on the good subspace:*

$$\left\|[V_{KK'H}, L_{XK'}]\Pi^{t}_{KK'H}\Pi^{\mathsf{rg}}_{KK'}\right\| \leq \tilde{O}(\left(\sqrt{t^3 2^{-\min(r,c)}}\right).$$

Proof of Lemma 7.6. The key observation is essentially that V is a bijective function on good databases which leaves the k, k' databases invariant, and a query to k' remains mostly within the set of good databases. In more detail, consider projectors $\Pi^{\in\mathsf{db}}_{XK'}, \Pi^{\notin\mathsf{db}}_{XK'}$ which sum to the identity. We split into two cases.

(Case 1) $x \notin D_{k'}$, or $\left\|[V_{KK'H}, L_{XK'}]\Pi^{\mathsf{rg}}_{KK'}\Pi^{\notin\mathsf{db}}_{XK'}\right\|$. Note that $\mathsf{L}_{XK'}$ preserves the computational basis everywhere except the x-th register of D_k, and on distinct databases the images of V are orthogonal. By Lemma 3.7, it suffices to fix $x, D_k, D_{k'}, D_h$ as computational basis states with good databases of size at most t and $x \notin D_{k'}$, and consider the action on the state

$$|\psi\rangle = |x\rangle_X |D_k\rangle_K |D_{k'}\rangle_{K'} |D^R_h\rangle_H .$$

Let us define D^I_h (in the ideal world) as the database of input/output pairs in D^R_h (in the real world) without a tail under $D_k, D_{k'}$, and D^I_f (in the ideal world) as the database for f constructed from the set of input/output pairs in D_h with a tail. These are such that

$$V |D_k\rangle_K |D_{k'}\rangle_{K'} |D^R_h\rangle_H = |D_k\rangle_K |D_{k'}\rangle_{K'} |D^I_h\rangle_H |D^I_f\rangle_F .$$

Recall that databases without a superscript do not change between the ideal and real world. We may now compute

$$\begin{aligned}|\psi^I\rangle =& L_{XK'} V |\psi\rangle \\ =& |A\rangle_A |x\rangle_X |D^I_h\rangle_H |D_k\rangle_K |D^I_f\rangle_F \left(\sum_{y\in\{0,1\}^c} 2^{-c/2} |D_{k'}[x \to y]\rangle_{K'}\right)\end{aligned}$$

as well as

$$\begin{aligned}|\psi^R\rangle =& L_{XK} |\psi\rangle \\ =& |A\rangle_A |x\rangle_X |D^R_h\rangle_H |D_k\rangle_K \left(\sum_{y\in\{0,1\}^c} 2^{-c/2} |D_{k'}[x \to y]\rangle_{K'}\right).\end{aligned}$$

Let B be the set of images y of x such that assigning x to y will cause a bad completion. Observe that, from the analysis of Lemma 5.16, we have $|B| \leq O(t^3 n + t^3 2^{r-c})$. For any value $y \notin B$, we have the identity

$$V |D_k\rangle_X |D_{k'}[x \to y]\rangle_{K'} |D^I_h\rangle_H = |D_k\rangle_X |D_{k'}[x \to y]\rangle_{K'} |D^R_h\rangle_H |D^R_f\rangle ,$$

because in such cases no new values $z \in D_h$ will have a tail under the assignment $[x \to y]$. It follows that

$$V\Pi^{B\perp}\left|\psi^R\right\rangle = \Pi^{B\perp}\left|\psi^I\right\rangle. \tag{39}$$

We can write

$$\begin{aligned}
\left\|\left|\psi^R\right\rangle - \Pi^{B\perp}\left|\psi^R\right\rangle\right\| &= \left\|\left|A\right\rangle_A\left|x\right\rangle_X\left|D_h^R\right\rangle_H\left|D_k\right\rangle_K\left(\sum_{y\in B}2^{-c/2}\left|D_{k'}[x\to y]\right\rangle_{K'}\right)\right\| \\
&= \sqrt{|B|2^{-c}} \qquad (40)\\
&\leq \tilde{O}(\sqrt{t^3 2^{-\min(r,c)}}), \\
\left\|\left|\psi^I\right\rangle - \Pi^{B\perp}\left|\psi^I\right\rangle\right\| &= \left\|\left|A\right\rangle_A\left|x\right\rangle_X\left|D_h^I\right\rangle_H\left|D_k\right\rangle_K\left|D_f^I\right\rangle_F\left(\sum_{y\in B}2^{-c/2}\left|D_{k'}[x\to y]\right\rangle_{K'}\right)\right\| \\
&= \sqrt{|B|2^{-c}} \\
&\leq \tilde{O}(\sqrt{t^3 2^{-\min(r,c)}}). \qquad (41)
\end{aligned}$$

Putting everything together, we have

$$\begin{aligned}
\left\|\left|\psi^I\right\rangle - V\left|\psi^R\right\rangle\right\| &\leq \left\|\left|\psi^I\right\rangle - \Pi^{B\perp}\left|\psi^I\right\rangle\right\| + \left\|\Pi^{B\perp}\left|\psi^I\right\rangle - V\Pi^{B\perp}\left|\psi^R\right\rangle\right\| \\
&\quad + \left\|V\left|\psi^R\right\rangle - V\Pi^{B\perp}\left|\psi^R\right\rangle\right\| && \text{(Triangle inequality)} \\
&= \left\|\left|\psi^I\right\rangle - \Pi^{B\perp}\left|\psi^I\right\rangle\right\| + \left\|V\left|\psi^R\right\rangle - V\Pi^{B\perp}\left|\psi^R\right\rangle\right\| && \text{(Equation (39))} \\
&= \left\|\left|\psi^I\right\rangle - \Pi^{B\perp}\left|\psi^I\right\rangle\right\| + \left\|\left|\psi^R\right\rangle - \Pi^{B\perp}\left|\psi^R\right\rangle\right\| && (V \text{ an isometry}) \\
&\leq \tilde{O}(\sqrt{t^3 2^{-\min(r,c)}}) && \text{(Equations (40) and (41))}
\end{aligned}$$

(Case 2) $x \in D_{k'}$, or $\left\|[V_{KK'H}, L_{XK'}]\Pi^{\mathsf{rg}}_{KK'}\Pi^{\in\mathsf{db}}_{XK'}\right\|$. Once again, note that $L_{XK'}$ preserves the computational basis everywhere except the x-th register of D_k, and on distinct databases the images of V are orthogonal. By Lemma 3.7, it suffices to fix $x, D_k, D_{k'}, D_h$ as computational basis states with good databases of size at most t and where $x \notin D_{k'}$, and consider the action on a state of the form

$$\left|\psi\right\rangle = \sum_{z \text{ s.t. } D_k, D_{k'}[x\to z] \text{ is good}} \alpha_z \left|x\right\rangle_X \left|D_k\right\rangle_K \left|D_{k'}[x\to z]\right\rangle_{K'} \left|D_h\right\rangle_H.$$

Let us similarly define $D_h^{I,y}$ (in the ideal world) as the database of input/output pairs in D_h^R (in the real world) without a tail under $D_k, D_{k'}[x \to y]$, and $D_f^{I,y}$ (in the ideal world) as the database for f constructed from the set of input/output pairs in D_h with a tail in $D_k, D_{k'}[x \to y]$. These are such that

$$V\left|D_k\right\rangle_K\left|D_{k'}[x\to y]\right\rangle_{K'}\left|D_h^R\right\rangle_H = \left|D_k\right\rangle_K\left|D_{k'}[x\to y]\right\rangle_{K'}\left|D_h^{I,y}\right\rangle_H\left|D_f^{I,y}\right\rangle_F.$$

Recall that databases without a superscript do not change between the ideal and real world. We may now compute

$$\begin{aligned}|\psi^I\rangle =& L_{XK'} V |\psi\rangle \\ =& \sum_{z \text{ s.t. } D_k, D_{k'}[x\to z] \text{ is good}} \alpha_z |A\rangle_A |x\rangle_X |D_h^{I,z}\rangle_H |D_k\rangle_K |D_f^{I,z}\rangle_F \otimes \\ & \left(|D_{k'}[x\to z]\rangle_{K'} - 2^{-c} \sum_{u\in\{0,1\}^c} |D_{k'}[x\to u]\rangle_{K'} + 2^{-c/2} |D_{k'}\rangle_{K'} \right)\end{aligned}$$

as well as

$$\begin{aligned}|\psi^R\rangle =& L_{XK'} |\psi\rangle \\ =& \sum_{z \text{ s.t. } D_k, D_{k'}[x\to z] \text{ is good}} \alpha_z |A\rangle_A |x\rangle_X |D_h^{R}\rangle_H |D_k\rangle_K \otimes \\ & \left(|D_{k'}[x\to z]\rangle_{K'} - 2^{-c} \sum_{u\in\{0,1\}^c} |D_{k'}[x\to u]\rangle_{K'} + 2^{-c/2} |D_{k'}\rangle_{K'} \right)\end{aligned}$$

Let B be the set of images y of x such that assigning x to y in $D_{k'}$ will cause a bad completion. Observe that, from the analysis of Lemma 5.16, we have $|B| \leq O(t^3 n + t^3 2^{r-c})$. For any value $y \notin B$, we have the identity

$$V |D_k\rangle_X |D_{k'}[x\to y]\rangle_{K'} |D_h^I\rangle_H = |D_k\rangle_X |D_{k'}[x\to y]\rangle_{K'} |D_h^R\rangle_H |D_f^R\rangle ,$$

because in such cases no new state values which are in the database D_h will have a tail under the assignment $[x \to y]$. Observe here that the initial state $|\psi\rangle$ may be supported on images that lead to a "bad completion", i.e. if x is part of a tail. Let us analyze the difference

$$\begin{aligned}|\psi^I\rangle - V|\psi^R\rangle =& \sum_{z \text{ s.t. } D_k, D_{k'}[x\to z] \text{ is good}} \alpha_z |A\rangle_A |x\rangle_X |D_h^{I,z}\rangle_H |D_k\rangle_K |D_f^{I,z}\rangle_F \otimes \\ & \left(|D_{k'}[x\to z]\rangle_{K'} - 2^{-c} \sum_{u\in\{0,1\}^c} |D_{k'}[x\to u]\rangle_{K'} + 2^{-c/2} |D_{k'}\rangle_{K'} \right) - \\ & \sum_{z \text{ s.t. } D_k, D_{k'}[x\to z] \text{ is good}} \alpha_z |A\rangle_A |x\rangle_X |D_k\rangle_K \otimes \\ & \left(|D_{k'}[x\to z]\rangle_{K'} |D_h^{I,z}\rangle_H |D_f^{I,z}\rangle_F - 2^{-c} \sum_{u\in\{0,1\}^c} |D_{k'}[x\to u]\rangle_K |D_h^{I,u}\rangle_H |D_f^{I,u}\rangle_F + \right. \\ & \left. 2^{-c/2} |D_{k'}\rangle_{K'} |D_h^{I,\perp}\rangle_H |D_f^{I,\perp}\rangle_F \right)\end{aligned}$$

Which, after collapsing terms, can be written as

$$\begin{aligned}|\psi^I\rangle - V|\psi^R\rangle = &\sum_{z \text{ s.t. } D_k, D_{k'}[x\to z] \text{ is good}} \alpha_z |A\rangle_A |x\rangle_X |D_k\rangle_K \otimes \\ &\left(2^{-c} \sum_{u\in\{0,1\}^c} |D_{k'}[x\to u]\rangle_{K'} \left(|D_h^{I,u}\rangle_H |D_f^{I,u}\rangle_F - |D_h^{I,z}\rangle_H |D_f^{I,z}\rangle_F\right)\right) + \\ &\sum_{z \text{ s.t. } D_k, D_{k'}[x\to z] \text{ is good}} \alpha_z |A\rangle_A |x\rangle_X |D_k\rangle_K \otimes \\ &\left(|D_{k'}\rangle_{K'} 2^{-c/2} (|D_h^{I,z}\rangle_H |D_f^{I,z}\rangle_F - |D_h^{I,\perp}\rangle_H |D_f^{I,\perp}\rangle_F)\right).\end{aligned}$$

We can then write

$$\begin{aligned}\|VL|\psi\rangle - LV|\psi\rangle\| = &\left\| V|\psi^R\rangle - |\psi^I\rangle \right\| \\ \leq &2 \underbrace{\left\| \sum_{z\in\{0,1\}^c} \alpha_z \sum_{u\in\{0,1\}^c, (D_h^{I,u} D_f^{I,u}) \neq (D_h^{I,z} D_f^{I,z})} 2^{-c} |D_{k'}[x\to u]\rangle \right\|}_{T_1} + \\ &\underbrace{2 \left\| \sum_{z\in\{0,1\}^c, (D_h^{I,z} D_f^{I,z}) \neq (D_h^{I,\perp} D_f^{I,\perp})} \alpha_z 2^{-c/2} \right\|}_{T_2} \qquad \text{(Triangle Inequality)}\end{aligned}$$

Let us focus on bounding each term individually. We begin with

$$\begin{aligned}T_1 = &\left\| \sum_{z\in\{0,1\}^c} \sum_{u\in\{0,1\}^c, (D_h^{I,u} D_f^{I,u}) \neq (D_h^{I,z} D_f^{I,z})} \alpha_z 2^{-c} |D_{k'}[x\to u]\rangle \right\| \\ \leq &2 \underbrace{\left\| \sum_{z\in\{0,1\}^c, (D_h^{I,z} D_f^{I,z}) \neq (D_h^{I,\perp} D_f^{I,\perp})} \sum_{u\in\{0,1\}^c, (D_h^{I,u} D_f^{I,u}) \neq (D_h^{I,z} D_f^{I,z})} \alpha_z 2^{-c} |D_{k'}[x\to u]\rangle \right\|}_{T_{11}} + \\ &\underbrace{2 \left\| \sum_{z\in\{0,1\}^c, (D_h^{I,z} D_f^{I,z}) = (D_h^{I,\perp} D_f^{I,\perp})} \sum_{u\in\{0,1\}^c, (D_h^{I,u} D_f^{I,u}) \neq (D_h^{I,z} D_f^{I,z})} \alpha_z 2^{-c} |D_{k'}[x\to u]\rangle \right\|}_{T_{12}} \qquad \text{(Triangle inequality}\end{aligned}$$

Observe that the second sum in T_{11} is over at most 2^c terms, which gives an upper bound of $|\alpha_z| 2^{-c/2}$ for it's norm. The first sum in T_{11} is over a set of size $O(t^3 n + t^3 2^{c-r})$ by Lemmas 5.16 and 5.19, so by the relation between L_1 and L_2 norm we have

$$\sum_{z\in\{0,1\}^c, (D_h^{I,z} D_f^{I,z}) \neq (D_h^{I,\perp} D_f^{I,\perp})} |\alpha_z| \leq O(\sqrt{t^3 n + t^3 2^{c-r}}). \tag{42}$$

It follows that $T_{11} \leq \tilde{O}(\sqrt{t^3 2^{-\min(r,c)}})$.

Observe that the second sum in T_{12} is over a set of size $O(t^3 n + t^3 2^{c-r})$ by Lemmas 5.16 and 5.19, which gives an upper bound of $|\alpha_z| \sqrt{O(t^3 + t^3 2^{c-r}} \cdot 2^{-c}$ for it's norm. The first sum in T_{12} is over a set of size at most 2^{-c}, so by the relation between L_1 and L_2 norm we have

$$\sum_{z \in \{0,1\}^c, (D_h^{I,z} D_f^{I,z}) = (D_h^{I,\perp} D_f^{I,\perp})} |\alpha_z| \leq 2^{c/2}.$$

It follows that $T_{12} \leq \tilde{O}(\sqrt{t^3 2^{-\min(r,c)}})$.

Finally, it follows from Equation (42) that $T_2 \leq \tilde{O}(\sqrt{t^3 2^{-\min(r,c)}})$.

□

B Permutation tail bounds

Let X, Y be subsets of $[N]$, and consider choosing a random permutation $\pi \sim S_N$ acting on $[N]$. We are interested in the number of elements sent by π from X to Y, and potentially want to consider many different X and Y at the same time. The most relevant characterization is below, which depends on the lemmas which follow it.

Intuitively, suppose we have a partition of preimages into equal sized bins, and a partition of images into equal sized buckets (potentially of a different size than the bins). Theorem B.1 states that the maximum number of elements sent from any given bin to any given bucket will be within a constant factor of the average, if the average is appreciable. If the average number is small (say $O(1)$ or even less than 1), then the max will be a positive integer which scales like $n = \log N$.

Theorem B.1. *Let $N \in \mathbb{N}$, $X_1 \sqcup X_2 \sqcup ... \sqcup X_l$ be a partition of $[N]$ such that $|X_i| = x$ for all i, and similarly $Y_1 \sqcup Y_2 \sqcup ... \sqcup Y_k$ be a partition of $[N]$ such that $|Y_i| = y$ for all i. Define $m = xy/N$ and suppose $N = 2^n$. Then, over the uniform choice of $\pi \sim S_N$, we have the bound*

$$\Pr_{\pi \sim S_N}[\max_{i \in [l], j \in [k]} (|\pi(X_i) \cap Y_j|) \geq 7m + 3n] \leq 2^{-n}.$$

Proof. Let us first consider any fixed $X = X_i$ and $Y = Y_j$. From Lemma B.2 and Lemma B.3, we have the equation

$$\begin{aligned} \Pr_{\pi \sim S_N}[|\pi(X) \cap Y| \geq 7m + k] &\leq \exp(-3k/4) \\ &\leq 2^{-k}. \end{aligned}$$

Let us choose k to be $3n$, and union bound over all possible values of i, j. Note that there are at most $N = 2^n$ values for each i and j, as each element of the partition is of the same (non-empty) size. We then have

$$\begin{aligned} \Pr_{\pi \sim S_N}[\max_{i \in [l], j \in [k]} (|\pi(X_i) \cap Y_j|) \geq 7m + 3n] &\leq \sum_{i \in [l], j \in [k]} \Pr_{\pi \sim S_N}[(|\pi(X_i) \cap Y_j|) \geq 7m + 3n] \\ &\leq \sum_{i \in [l], j \in [k]} 2^{-3n} \\ &\leq 2^{-n}, \end{aligned}$$

where the first line follows from a union bound. This completes the proof. □

B.1 Helper lemmas

We first can compute the expectation by linearity of expectation.

Lemma B.2. *Let $N \in \mathbb{N}$ and let $X, Y \subseteq [N]$ be subsets. Then, on average over the uniform choice of $\pi \sim S_N$, the expected number of elements sent from X to Y by π equals*

$$\mathop{\mathbb{E}}_{\pi \sim S_N}[|\pi(X) \cap Y|] = \frac{|X||Y|}{N} = m.$$

Proof. [CP24], Theorem 3.13. □

We now state our tail bound, which will be sufficiently tight for the case where the expected number of subset pairs is small.

Lemma B.3. *Let $N \in \mathbb{N}$, $X, Y \subseteq [N]$ be subsets, and $N = 2^n$. Denote $x = |X|$, $y = |Y|$, and $m = \mathop{\mathbb{E}}_{\sigma \sim S_N}[|\sigma(X) \cap Y|]$. Then, for any real number $u \geq 6m$, it holds that*

$$\Pr_{\pi \sim S_N}\left[|\pi(X) \cap Y| \geq m + u\right] \leq \exp\left(-\frac{3}{4}u\right).$$

Proof. [CP24], Theorem 3.15. □

References

[LR88] Michael Luby and Charles Rackoff. "How to Construct Pseudorandom Permutations from Pseudorandom Functions". In: *SIAM Journal on Computing* 17.2 (1988), pp. 373–386. DOI: 10.1137/0217022. eprint: https://doi.org/10.1137/0217022. URL: https://doi.org/10.1137/0217022.

[MRH04] Ueli Maurer, Renato Renner, and Clemens Holenstein. "Indifferentiability, Impossibility Results on Reductions, and Applications to the Random Oracle Methodology". In: *Theory of Cryptography*. Ed. by Moni Naor. Vol. 2951. Lecture Notes in Computer Science. Springer Berlin, Heidelberg, 2004, pp. 21–39.

[Ber+07] Guido Bertoni, Joan Daemen, Michaël Peeters, and Gilles van Assche. "Sponge functions". In: *ECRYPT Hash Workhsop*. 2007.

[Ber+08] Guido Bertoni, Joan Daemen, Michaël Peeters, and Gilles Van Assche. "On the Indifferentiability of the Sponge Construction". In: *Advances in Cryptology – EUROCRYPT 2008*. Ed. by Nigel Smart. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 181–197. ISBN: 978-3-540-78967-3.

[Ber+11a] G. Bertoni, J. Daemen, M. Peeters, and G. Van Assche. *Cryptographic sponge functions*. Submission to NIST (Round 3). 2011. URL: http://sponge.noekeon.org/CSF-0.1.pdf.

[Ber+11b] G. Bertoni, J. Daemen, M. Peeters, and G. Van Assche. *The Keccak SHA-3 submission*. Submission to NIST (Round 3). 2011. URL: http://keccak.noekeon.org/Keccak-submission-3.pdf.

[RSS11] Thomas Ristenpart, Hovav Shacham, and Thomas Shrimpton. "Careful with Composition: Limitations of the Indifferentiability Framework". In: *Advances in Cryptology – EUROCRYPT 2011*. Ed. by Kenneth G. Paterson. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 487–506. ISBN: 978-3-642-20465-4.

[SND15] National Institute of Standards, Technology (NIST), and Morris J. Dworkin. *SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions*. en. Aug. 2015. DOI: https://doi.org/10.6028/NIST.FIPS.202. URL: https://tsapps.nist.gov/publication/get_pdf.cfm?pub_id=919061.

[Zha16] Mark Zhandry. *A Note on Quantum-Secure PRPs*. 2016. arXiv: 1611.05564 [cs.CR]. URL: https://arxiv.org/abs/1611.05564.

[Car+18] Tore Vincent Carstens, Ehsan Ebrahimi, Gelo Noel Tabia, and Dominique Unruh. *On Quantum Indifferentiability*. Cryptology ePrint Archive, Paper 2018/257. 2018. URL: https://eprint.iacr.org/2018/257.

[Cza+18] Jan Czajkowski, Leon Groot Bruinderink, Andreas Hülsing, Christian Schaffner, and Dominique Unruh. "Post-quantum Security of the Sponge Construction". In: *Post-Quantum Cryptography*. Ed. by Tanja Lange and Rainer Steinwandt. Cham: Springer International Publishing, 2018, pp. 185–204. ISBN: 978-3-319-79063-3.

[CMS19] Alessandro Chiesa, Peter Manohar, and Nicholas Spooner. "Succinct Arguments in the Quantum Random Oracle Model". In: *Theory of Cryptography*. Ed. by Dennis Hofheinz and Alon Rosen. Cham: Springer International Publishing, 2019, pp. 1–29. ISBN: 978-3-030-36033-7.

[CHS19] Jan Czajkowski, Andreas Hülsing, and Christian Schaffner. "Quantum Indistinguishability of Random Sponges". In: *Advances in Cryptology – CRYPTO 2019*. Ed. by Alexandra Boldyreva and Daniele Micciancio. Cham: Springer International Publishing, 2019, pp. 296–325. ISBN: 978-3-030-26951-7.

[Cza+19] Jan Czajkowski, Christian Majenz, Christian Schaffner, and Sebastian Zur. *Quantum Lazy Sampling and Game-Playing Proofs for Quantum Indifferentiability*. Cryptology ePrint Archive, Paper 2019/428. https://eprint.iacr.org/2019/428. 2019. URL: https://eprint.iacr.org/2019/428.

[LZ19] Qipeng Liu and Mark Zhandry. "On Finding Quantum Multi-collisions". In: *Advances in Cryptology – EUROCRYPT 2019*. Ed. by Yuval Ishai and Vincent Rijmen. Cham: Springer International Publishing, 2019, pp. 189–218. ISBN: 978-3-030-17659-4.

[Zha19] Mark Zhandry. "How to Record Quantum Queries, and Applications to Quantum Indifferentiability". In: *Advances in Cryptology – CRYPTO 2019*. Ed. by Alexandra Boldyreva and Daniele Micciancio. Cham: Springer International Publishing, 2019, pp. 239–268. ISBN: 978-3-030-26951-7.

[Chu+21] Kai-Min Chung, Serge Fehr, Yu-Hsuan Huang, and Tai-Ning Liao. "On the compressed-oracle technique, and post-quantum security of proofs of sequential work". In: *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer. 2021, pp. 598–629.

[Don+21] Jelle Don, Serge Fehr, Christian Majenz, and Christian Schaffner. *Online-Extractability in the Quantum Random-Oracle Model*. Cryptology ePrint Archive, Paper 2021/280. 2021. URL: https://eprint.iacr.org/2021/280.

[Unr21] Dominique Unruh. *Compressed Permutation Oracles (And the Collision-Resistance of Sponge/SHA3)*. Cryptology ePrint Archive, Paper 2021/062. https://eprint.iacr.org/2021/062. 2021. URL: https://eprint.iacr.org/2021/062.

[Zha21] Mark Zhandry. "Redeeming Reset Indifferentiability and Applications to Post-quantum Security". In: *Advances in Cryptology – ASIACRYPT 2021*. Ed. by Mehdi Tibouchi and Huaxiong Wang. Cham: Springer International Publishing, 2021, pp. 518–548. ISBN: 978-3-030-92062-3.

[Ala+22] Gorjan Alagic, Chen Bai, Jonathan Katz, and Christian Majenz. "Post-Quantum Security of the Even-Mansour Cipher". In: *Advances in Cryptology – EUROCRYPT 2022*. Ed. by Orr Dunkelman and Stefan Dziembowski. Cham: Springer International Publishing, 2022, pp. 458–487. ISBN: 978-3-031-07082-2.

[Don+22] Jelle Don, Serge Fehr, Christian Majenz, and Christian Schaffner. "Efficient NIZKs and Signatures from Commit-and-Open Protocols in the QROM". In: *Advances in Cryptology – CRYPTO 2022*. Ed. by Yevgeniy Dodis and Thomas Shrimpton. Cham: Springer Nature Switzerland, 2022, pp. 729–757. ISBN: 978-3-031-15979-4.

[LM22] Charlotte Lefevre and Bart Mennink. "Tight Preimage Resistance of the Sponge Construction". In: *Advances in Cryptology – CRYPTO 2022*. Ed. by Yevgeniy Dodis and Thomas Shrimpton. Cham: Springer Nature Switzerland, 2022, pp. 185–204. ISBN: 978-3-031-15985-5.

[Ros22] Ansis Rosmanis. *Tight Bounds for Inverting Permutations via Compressed Oracle Arguments*. 2022. arXiv: 2103.08975 [quant-ph].

[Agu+23] Carlos Aguilar-Melchor, Andreas Hülsing, David Joseph, Christian Majenz, Eyal Ronen, and Dongze Yue. "SDitH in the QROM". In: *Advances in Cryptology – ASIACRYPT 2023*. Ed. by Jian Guo and Ron Steinfeld. Singapore: Springer Nature Singapore, 2023, pp. 317–350. ISBN: 978-981-99-8739-9.

[HM23] Yassine Hamoudi and Frédéric Magniez. "Quantum Time–Space Tradeoff for Finding Multiple Collision Pairs". In: *ACM Transactions on Computation Theory* 15.1–2 (June 2023), pp. 1–22. ISSN: 1942-3462. DOI: 10.1145/3589986. URL: http://dx.doi.org/10.1145/3589986.

[Unr23] Dominique Unruh. "Towards Compressed Permutation Oracles". In: *Advances in Cryptology – ASIACRYPT 2023: 29th International Conference on the Theory and Application of Cryptology and Information Security, Guangzhou, China, December 4–8, 2023, Proceedings, Part IV*. Guangzhou, China: Springer-Verlag, 2023, pp. 369–400. ISBN: 978-981-99-8729-0. DOI: 10.1007/978-981-99-8730-6_12. URL: https://doi.org/10.1007/978-981-99-8730-6_12.

[Ala+24a] Gorjan Alagic, Chen Bai, Jonathan Katz, Christian Majenz, and Patrick Struck. "Post-quantum Security of Tweakable Even-Mansour, and Applications". In: *Advances in Cryptology – EUROCRYPT 2024*. Ed. by Marc Joye and Gregor Leander. Cham: Springer Nature Switzerland, 2024, pp. 310–338. ISBN: 978-3-031-58716-0.

[Ala+24b] Gorjan Alagic, Quynh Dang, Dustin Moody, Angela Robinson, Hamilton Silberg, Daniel Smith-Tone, et al. "Module-Lattice-Based Key-Encapsulation Mechanism Standard". In: (2024).

[CP24] Joseph Carolan and Alexander Poremba. "Quantum One-Wayness of the Single-Round Sponge with Invertible Permutations". In: *Advances in Cryptology – CRYPTO 2024*. Ed. by Leonid Reyzin and Douglas Stebila. Cham: Springer Nature Switzerland, 2024, pp. 218–252. ISBN: 978-3-031-68391-6.

[CPZ24] Joseph Carolan, Alexander Poremba, and Mark Zhandry. *(Quantum) Indifferentiability and Pre-Computation*. Cryptology ePrint Archive, Paper 2024/1727. 2024. URL: https://eprint.iacr.org/2024/1727.

[Coo+24] David Cooper et al. "Stateless Hash-Based Digital Signature Standard". In: (2024).

[Dan+24] Thinh Dang, Jacob Lichtinger, Yi-Kai Liu, Carl Miller, Dustin Moody, Rene Peralta, Ray Perlner, Angela Robinson, et al. "Module-Lattice-Based Digital Signature Standard". In: (2024).

[Hül+24] Andreas Hülsing, David Joseph, Christian Majenz, and Anand Kumar Narayanan. "On Round Elimination for Special-Sound Multi-round Identification and the Generality of the Hypercube for MPCitH". In: *Advances in Cryptology – CRYPTO 2024*. Ed. by Leonid Reyzin and Douglas Stebila. Cham: Springer Nature Switzerland, 2024, pp. 373–408. ISBN: 978-3-031-68376-3.

[MMW24] Christian Majenz, Giulio Malavolta, and Michael Walter. *Permutation Superposition Oracles for Quantum Query Lower Bounds*. Cryptology ePrint Archive, Paper 2024/1140. 2024. URL: https://eprint.iacr.org/2024/1140.

[RT24] Lior Rotem and Stefano Tessaro. *Straight-Line Knowledge Extraction for Multi-Round Protocols*. Cryptology ePrint Archive, Paper 2024/1724. 2024. URL: https://eprint.iacr.org/2024/1724.

[Sön+24] Meltem Sönmez Turan, Kerry McKay, Donghoon Chang, Jinkeon Kang, and John Kelsey. *Ascon-Based Lightweight Cryptography Standards for Constrained Devices: Authenticated Encryption, Hash, and Extendable Output Functions*. Tech. rep. National Institute of Standards and Technology, 2024.

[Bau+25] Carsten Baum, Ward Beullens, Lennart Braun, Cyprien Delpech de Saint Guilhem, Michael Klooß, Christian Majenz, Shibam Mukherjee, Emmanuela Orsini, Sebastian Ramacher, Christian Rechberger, Lawrence Roy, and Peter Scholl. "FAEST v2: Algorithm Specifications". In: (2025).