# Quantum-Secure Message Authentication Codes

Dan Boneh Mark Zhandry

Stanford University
{dabo,zhandry}@cs.stanford.edu

**Abstract**

We construct the first Message Authentication Codes (MACs) that are existentially unforgeable against a *quantum* chosen message attack. These chosen message attacks model a quantum adversary's ability to obtain the MAC on a superposition of messages of its choice. We begin by showing that a quantum secure PRF is sufficient for constructing a quantum secure MAC, a fact that is considerably harder to prove than its classical analogue. Next, we show that a variant of Carter-Wegman MACs can be proven to be quantum secure. Unlike the classical settings, we present an attack showing that a pair-wise independent hash family is insufficient to construct a quantum secure *one-time* MAC, but we prove that a four-wise independent family is sufficient for one-time security.

*Keywords*: Quantum computing, MAC,chosen message attacks, post-quantum security

## 1 Introduction

Message Authentication Codes (MACs) are an important building block in cryptography used to ensure data integrity. A MAC system is said to be secure if an efficient attacker capable of mounting a chosen message attack cannot produce an existential MAC forgery (see Section 2.2).

With the advent of quantum computing there is a strong interest in post-quantum cryptography, that is systems that remain secure even when the adversary has access to a quantum computer. There are two natural approaches to defining security of a MAC system against a quantum adversary. One approach is to restrict the adversary to issue classical chosen message queries, but then allow the adversary to perform quantum computations between queries. Security in this model can be achieved by basing the MAC construction on a quantum intractable problem.

The other more conservative approach to defining quantum MAC security is to model the entire security game as a quantum experiment and allow the adversary to issue *quantum* chosen message queries. That is, the adversary can submit a superposition of messages from the message space and in response receive a superposition of MAC tags on those messages. Informally, a quantum chosen message query performs the following transformation on a given superposition of messages:

$$\sum_m \psi_m \left|m\right\rangle \quad \longrightarrow \quad \sum_m \psi_m \left|m, S(k,m)\right\rangle$$

where $S(k,m)$ is a tag on the message $m$ with secret key $k$.

To define security, let $q$ be the number of queries that the adversary issues by the end of the game. Clearly it can now produce $q$ classical message-tag pairs by sampling the $q$ superpositions it received from the MAC signing oracle. We say that the MAC system is *quantum secure* if the adversary cannot produce $q+1$ valid message-tag pairs. This captures the fact that the adversary cannot do any better than trivially sampling the responses from its MAC signing oracle and is the quantum analogue of a classical existential forgery.

### 1.1 Our results

In this paper we construct the first quantum secure MAC systems. We begin with a definition of quantum secure MACs and give an example of a MAC system that is secure against quantum adversaries capable of *classical* chosen message queries, but is insecure when the adversary can issue *quantum* chosen message queries. We then present a number of quantum secure MAC systems.

**Quantum secure MACs.** In the classical settings many MAC systems are based on the observation that a secure pseudorandom function gives rise to a secure MAC [BKR00, BCK96]. We begin by studying the same question in the quantum settings. Very recently Zhandry [Zha12b] defined the concept of a quantum secure pseudorandom function (PRF) which is a PRF that remains indistinguishable from a random function even when the adversary can issue *quantum* queries to the PRF. He showed that the classic GGM construction [GGM86] remains secure under quantum queries assuming the underlying pseudorandom generator is quantum secure.

The first question we study is whether a quantum secure PRF gives rise to a quantum secure MAC, as in the classical settings. To the MAC adversary a quantum secure PRF is indistinguishable from a random function. Therefore proving that the MAC is secure amounts to proving that with $q$ quantum queries to a random oracle $H$ no adversary can produce $q+1$ input-output pairs of $H$ with non-negligible probability. In the classical settings where the adversary can only issue *classical* queries to $H$ this is trivial: given $q$ evaluations of a random function, the adversary learns nothing about the value of the function at other points. Unfortunately, this argument fails under quantum queries because the response to a *single* quantum query to $H : \mathcal{X} \to \mathcal{Y}$ contains information about all of $H$. In fact, with a single quantum query the adversary can produce two input-output pairs of $H$ with probability about $2/|\mathcal{Y}|$ (with classical queries the best possible is $1/|\mathcal{Y}|$). As a result, proving that $q$ quantum queries are insufficient to produce $q+1$ input-output pairs is quite challenging. We prove tight upper and lower bounds on this question by proving the following theorem:

**Theorem 1.1** (informal). *Let $H : \mathcal{X} \to \mathcal{Y}$ be a random oracle. Then an adversary making at most $q < |\mathcal{X}|$ quantum queries to $H$ will produce $q+1$ input-output pairs of $H$ with probability at most $(q+1)/|\mathcal{Y}|$. Furthermore, when $q \ll |\mathcal{Y}|$ there is an algorithm that with $q$ quantum queries to $H$ will produce $q+1$ input-output pairs of $H$ with probability $1-(1-1/|\mathcal{Y}|)^{q+1} \approx (q+1)/|\mathcal{Y}|$.*

The first part of the theorem is the crucial fact needed to build quantum secure MACs and is the harder part to prove. It shows that when $|\mathcal{Y}|$ is large any algorithm has only a negligible chance in producing $q+1$ input-output pairs of $H$ from $q$ quantum queries. To prove this bound we introduce a new lower-bound technique we call the *rank method* for bounding the success probability of algorithms that succeed with only small probability. Existing quantum lower bound techniques such as the polynomial method [BBC+01] and the adversary method [Amb00, Aar02, Amb06, ASdW09] do not give the result we need. One difficulty with existing lower bound techniques is that they generally prove asymptotic bounds on the number of queries required to solve a problem with high probability, whereas we need a bound on the success probability of an algorithm making a limited number of queries. Attempting to apply existing techniques to our problem at best only bounds the success probability away from 1 by an inverse polynomial factor, which is insufficient for our purposes. The rank method for proving quantum lower bounds overcomes these difficulties and is a general tool that can be used in other post-quantum security proofs.

The second part of Theorem 1.1 shows that the lower bound presented in the first part of the theorem is tight. A related algorithm was previously presented by van Dam [vD98], but only for oracles outputting one bit, namely when $\mathcal{Y} = \{0,1\}$. For such a small range only about $|\mathcal{X}|/2$ quantum queries are needed to learn the oracle at all $|\mathcal{X}|$ points. A special case where $\mathcal{Y} = \mathcal{X} = \{0,1\}$ and $q = 1$ was developed independently by Kerenidis and de Wolf [KdW03]. Our algorithm is a generalization of van Dam's result to multi-bit oracles.

**Quantum secure Carter-Wegman MACs.** A Carter-Wegman MAC [WC81] signs a message $m$ by computing $(r,\ h(m) \oplus F(k,r))$ where $h$ is a secret hash function chosen from an xor-universal hash family, $F$ is a secure PRF with secret key $k$, and $r$ is a short random nonce. The attractive feature of Carter-Wegman MACs is that the long message $m$ is hashed by a fast xor-universal hash $h$. We show that a slightly modified Carter-Wegman MAC is quantum secure assuming the underlying PRF is quantum secure in the sense of Zhandry [Zha12b].

**One-time quantum secure MACs.** A one-time MAC is existentially unforgeable when the adversary can make only a *single* chosen message query. Classically, one-time MACs are constructed from pair-wise independent hash functions [WC81]. These MACs are one-time secure since the value of a pair-wise independent hash at one point gives no information about its value at another point. Therefore, a single classical chosen-message query tells the adversary nothing about the MAC tag of another message.

In the quantum settings things are more complicated. Unlike the classical settings, we show that pair-wise independence does not imply existential unforgeability under a one-time quantum chosen message attack. For example, consider the simple pair-wise independent hash family $\mathcal{H} = \{h(x) = ax+b\}_{a,b\in\mathbb{F}_p}$ with domain and range $\mathbb{F}_p$. We show that a quantum adversary presented with an oracle for a random function $h \in \mathcal{H}$ can find both $a$ and $b$ with a *single* quantum query to $h$. Consequently, the classical one-time MAC constructed from $\mathcal{H}$ is completely insecure in the quantum settings. More generally we prove the following theorem:

**Theorem 1.2** (informal)**.** *There is a polynomial time quantum algorithm that when presented with an oracle for $h(x) = a_0 + a_1 x + \ldots + a_d x^d$ for random $a_0, \ldots, a_d$ in $\mathbb{F}_p$ can recover $a_0, \ldots, a_d$ using only $d$ quantum queries to the oracle with probability $1 - O(d/n)$.*

The $h(x) = ax + b$ attack discussed above is a special case of this theorem with $d = 1$. With classical queries finding $a_0, \ldots, a_d$ requires $d+1$ queries, but with quantum queries the theorem shows that $d$ queries are sufficient.

Theorem 1.2 is a quantum polynomial interpolation algorithm: given oracle access to the polynomial, the algorithm reconstructs its coefficients. This problem was studied previously by Kane and Kutin [KK11] who prove that $d/2$ quantum queries are insufficient to interpolate the polynomial. Interestingly, they conjecture that quantum interpolation requires $d+1$ quantum queries as in the classical case, but Theorem 1.2 refutes that conjecture. Theorem 1.2 also applies to a quantum version of secret sharing where the shares themselves are superpositions. It shows that the classical Shamir secret sharing scheme [Sha79] is insecure if the shares are allowed to be quantum states obtained by evaluating the secret sharing polynomial on quantum superpositions.

More generally, the security of secret sharing schemes in the quantum settings was analyzed by Damård et al. [DFNS11].

As for one-time secure MACs, while pair-wise independence is insufficient for quantum one-time security, we show that four-wise independence is sufficient. That is, a four-way independent hash family gives rise to an existentially unforgeable MAC under a one-time quantum chosen message attack. It is still an open problem whether three-way independence is sufficient. More generally, we show that $(q+1)$-way independence is insufficient for a $q$-time quantum secure MAC, but $(3q+1)$-way independence is sufficient.

**Motivation.** Allowing the adversary to issue quantum chosen message queries is a natural and conservative security model and is therefore an interesting one to study. Showing that classical MAC constructions remain secure in this model gives confidence in case end-user computing devices eventually become quantum. Nevertheless, one might imagine that even in a future where computers are quantum, the last step in a MAC signing procedure is to sample the resulting quantum state so that the generated MAC is always classical. The quantum chosen message query model ensures that even if the attacker can bypass this last "classicalization" step, the MAC remains secure.

As further motivation we note that the results in this paper are the tip of a large emerging area of research with many open questions. Consider for example signature schemes. Can one design schemes that remain secure when the adversary can issue quantum chosen message queries? Similarly, can one design encryption systems that remain secure when the the adversary can issue quantum chosen ciphertext queries? More generally, for any cryptographic primitive modeled as an interactive game, one can ask how to design primitives that remain secure when the interaction between the adversary and its given oracles is quantum.

**Other related work.** Several recent works study the security of cryptographic primitives when the adversary can issue quantum queries [BDF+11, Zha12a, Zha12b]. So far these have focused on proving security of signatures, encryption, and identity-based encryption in the *quantum* random oracle model where the adversary can query the random oracle on superpositions of inputs. These works show that many, but not all, random oracle constructions remain secure in the quantum random oracle model. The quantum random oracle model has also been used to prove security of Merkle's Puzzles in the quantum settings [BS08, BHK+11]. Meanwhile, Damård et al. [DFNS11] examine secret sharing and multiparty computation in a model where an adversary may corrupt a superposition of subsets of players, and build zero knowledge protocols that are secure, even when a dishonest verifier can issue challenges on superpositions.

Some progress toward identifying sufficient conditions under which classical protocols are also quantum immune has been made by Unruh [Unr10] and Hallgren et al. [HSS11]. Unruh shows that any scheme that is statistically secure in Cannetti's universal composition (UC) framework [Can01] against classical adversaries is also statistically secure against quantum adversaries. Hallgren et al. show that for many schemes this is also true in the computational setting. These results, however, do not apply to MACs.

## 2 Preliminaries: Definitions and Notation

Let $[n]$ be the set $\{1, ..., n\}$. For a prime power $n$, let $\mathbb{F}_n$ be the finite field on $n$ elements. For any positive integer $n$, let $\mathbb{Z}_n$ be the ring of integers modulo $n$.

Functions will be denoted by capitol letters (such as $F$), and sets by capitol script letters (such as $\mathcal{X}$). We denote vectors with bold lower-case letters (such as $\mathbf{v}$), and the components of a vector $\mathbf{v} \in \mathcal{A}^n$ by $v_i$, $i \in [n]$. We denote matrices with bold capital letters (such as $\mathbf{M}$), and the components of a matrix $\mathbf{M} \in \mathcal{A}^{m \times n}$ by $M_{i,j}$, $i \in [m], j \in [n]$. Given a function $F : \mathcal{X} \to \mathcal{Y}$ and a vector $\mathbf{v} \in \mathcal{X}^n$, let $F(\mathbf{v})$ denote the vector $(F(v_1), F(v_2), ..., F(v_k))$. Let $F([n])$ denote the vector $(F(1), F(2), ..., F(n))$.

Given a vector space $\mathcal{V}$, let $\dim \mathcal{V}$ be the dimension of $\mathcal{V}$, or the number of vectors in any basis for $\mathcal{V}$. Given a set of vectors $\{\mathbf{v}_1, ..., \mathbf{v}_k\}$, let $\text{span}\{\mathbf{v}_1, ..., \mathbf{v}_k\}$ denote the space of all linear combinations of vectors in $\{\mathbf{v}_1, ..., \mathbf{v}_k\}$. Given a subspace $S$ of an inner-product space $V$, and a vector $\mathbf{v} \in V$, define $\text{proj}_S \mathbf{v}$ as the orthogonal projection of $\mathbf{v}$ onto $S$, that is, the vector $\mathbf{w} \in S$ such that $|\mathbf{v} - \mathbf{w}|$ is minimized.

Given a matrix $\mathbf{M}$, we define the rank, denoted $\text{rank}(\mathbf{M})$, to be the size of the largest subset of rows (equivalently, columns) of $\mathbf{M}$ that are linearly independent.

Given a function $F : \mathcal{X} \to \mathcal{Y}$ and a subset $\mathcal{S} \subseteq \mathcal{X}$, the restriction of $F$ to $\mathcal{S}$ is the function $F_{\mathcal{S}} : \mathcal{S} \to \mathcal{Y}$ where $F_{\mathcal{S}}(x) = F(x)$ for all $x \in \mathcal{S}$. A distribution $D$ on the set of functions $F$ from $\mathcal{X}$ to $\mathcal{Y}$ induces a distribution $D_{\mathcal{S}}$ on the set of functions from $\mathcal{S}$ to $\mathcal{Y}$, where we sample from $D_{\mathcal{S}}$ by first sampling a function $F$ from $D$, and outputting $F_{\mathcal{S}}$. We say that $D$ is $k$-wise independent if, for each set $\mathcal{S}$ of size at most $k$, each of the distributions $D_{\mathcal{S}}$ are truly random distributions on functions from $\mathcal{S}$ to $\mathcal{Y}$. A set $\mathcal{F}$ of functions from $\mathcal{X}$ to $\mathcal{Y}$ is $k$-wise independent if the uniform distribution on $\mathcal{F}$ is $k$-wise independent.

## 2.1 Quantum Computation

The quantum system $A$ is a complex Hilbert space $\mathcal{H}$ with inner product $\langle \cdot | \cdot \rangle$. The state of a quantum system is given by a vector $|\psi\rangle$ of unit norm ($\langle \psi | \psi \rangle = 1$). Given quantum systems $\mathcal{H}_1$ and $\mathcal{H}_2$, the joint quantum system is given by the tensor product $\mathcal{H}_1 \otimes \mathcal{H}_2$. Given $|\psi_1\rangle \in \mathcal{H}_1$ and $|\psi_2\rangle \in \mathcal{H}_2$, the product state is given by $|\psi_1\rangle|\psi_2\rangle \in \mathcal{H}_1 \otimes \mathcal{H}_2$. Given a quantum state $|\psi\rangle$ and an orthonormal basis $B = \{|b_0\rangle, ..., |b_{d-1}\rangle\}$ for $\mathcal{H}$, a measurement of $|\psi\rangle$ in the basis $B$ results in a value $b_i$ with probability $|\langle b_i | \psi \rangle|^2$, and the state $|\psi\rangle$ is collapsed to the state $|b_i\rangle$. We let $b_i \leftarrow |\psi\rangle$ denote the distribution on $b_i$ obtained by sampling $|\psi\rangle$.

A unitary transformation over a $d$-dimensional Hilbert space $\mathcal{H}$ is a $d \times d$ matrix $\mathbf{U}$ such that $\mathbf{U}\mathbf{U}^\dagger = \mathbf{I}_d$, where $\mathbf{U}^\dagger$ represents the conjugate transpose. A quantum algorithm operates on a product space $\mathcal{H}_{in} \otimes \mathcal{H}_{out} \otimes \mathcal{H}_{work}$ and consists of $n$ unitary transformations $\mathbf{U}_1, ..., \mathbf{U}_n$ in this space. $\mathcal{H}_{in}$ represents the input to the algorithm, $\mathcal{H}_{out}$ the output, and $\mathcal{H}_{work}$ the work space. A classical input $x$ to the quantum algorithm is converted to the quantum state $|x, 0, 0\rangle$. Then, the unitary transformations are applied one-by-one, resulting in the final state

$$|\psi_x\rangle = \mathbf{U}_n ... \mathbf{U}_1 |x, 0, 0\rangle \ .$$

The final state is measured, obtaining $(a, b, c)$ with probability $|\langle a, b, c | \psi_x \rangle|^2$. The output of the algorithm is $b$.

**Quantum-accessible Oracles.** We will implement an oracle $O : \mathcal{X} \to \mathcal{Y}$ by a unitary transformation $\mathbf{O}$ where

$$\mathbf{O}|x, y, z\rangle = |x, y + O(x), z\rangle$$

where $+ : \mathcal{X} \times \mathcal{X} \to \mathcal{X}$ is some group operation on $\mathcal{X}$. Suppose we have a quantum algorithm that makes quantum queries to oracles $O_1, ..., O_q$. Let $|\psi_0\rangle$ be the state of the algorithm before any queries, and let $\mathbf{U}_1, ..., \mathbf{U}_q$ be the unitary transformations applied between queries. The final state of the algorithm will be

$$\mathbf{U}_q \mathbf{O}_q ... \mathbf{U}_1 \mathbf{O}_1 |\psi_0\rangle$$

We can also have an algorithm make classical queries to $O_i$. In this case, the input to the oracle is measured before applying the transformation $\mathbf{O}_i$.

Fix an oracle $O : \mathcal{X} \to \mathcal{Y}$. Let $O^{(q)} : \mathcal{X}^q \to \mathcal{Y}^q$ be the oracle that maps $\mathbf{x}$ into $O(\mathbf{x}) = (O(x_1), O(x_2), ..., O(x_q))$. Observe that any quantum query to $O^{(q)}$ can be implemented using $q$ quantum queries to $O$, where the unitary transformations between queries just permute the registers. We say that an algorithm that makes a single query to $O^{(q)}$ makes $q$ non-adaptive queries to $O$.

**The Density Matrix.** Suppose the state of a quantum system depends on some hidden random variable $z \in \mathcal{Z}$, which is distributed according to a distribution $D$. That is, if the hidden variable is $z$, the state of the system is $|\psi_z\rangle$. We can then define the density matrix of the quantum system as

$$\boldsymbol{\rho} = \sum_{z \in \mathcal{Z}} \Pr_D[z] |\psi_z\rangle\langle\psi_z|$$

Applying a unitary matrix $\mathbf{U}$ to the quantum state corresponds to the transformation

$$\boldsymbol{\rho} \to \mathbf{U}\boldsymbol{\rho}\mathbf{U}^\dagger$$

A partial measurement on some registers has the effect of zeroing out the terms in $\boldsymbol{\rho}$ where those registers are not equal. For example, if we have two registers $x$ and $y$, and we measure the $x$ register, then the new density matrix is

$$\boldsymbol{\rho}'_{x,y,x',y'} = \begin{cases} \boldsymbol{\rho}_{x,y,x',y'} & \text{if } x = x' \\ 0 & \text{otherwise} \end{cases}$$

## 2.2 Quantum secure MACs

A MAC system comprises two algorithms: a (possibly) randomized MAC signing algorithm $S(k, m)$ and a MAC verification algorithm $V(k, m, t)$. Here $k$ denotes the secret key chosen at random from the key space, $m$ denotes a message in the message space, and $t$ denotes the MAC tag in the tag space on the message $m$. These algorithms and spaces are parameterized by a security parameter $\lambda$.

Classically, a MAC system is said to be secure if no attacker can win the following game: a random key $k$ is chosen from the key space and the attacker is presented with a signing oracle $S(k, \cdot)$. Queries to the signing oracle are called chosen message queries. Let $\{(m_i, t_i)\}_{i=1}^q$ be the set of message-tag pairs that the attacker obtains by interacting with the signing oracle. The attacker wins the game if it can produce an existential forgery, namely a valid message-tag pair $(m^*, t^*)$ satisfying $(m^*, t^*) \notin \{(m_i, t_i)\}_{i=1}^q$. The MAC system is said to be secure if no "efficient" adversary can win this game with non-negligible probability in $\lambda$.

**Quantum chosen message queries.** In the quantum settings we allow the adversary to maintain its own quantum state and issue quantum queries to the signing oracle. Let $\sum_{m,x,y} \psi_{m,x,y} |m, x, y\rangle$ be the adversary's state just prior to issuing a signing query. The MAC signing oracle transforms this state as follows:

1. it chooses a random string $r$ that will be used by the MAC signing algorithm,
2. it signs each "slot" in the given superposition by running $S(k, m; r)$, that is running algorithm $S$ with randomness $r$. More precisely, the signing oracle performs the following transformation:

$$\sum_{m,x,y} \psi_{m,x,y} |m, x, y\rangle \quad \longrightarrow \quad \sum_{m,x,y} \psi_{m,x,y} |m,\ x \oplus S(k, m; r),\ y\rangle$$

When the signing algorithm is deterministic there is no need to choose an $r$. However, for randomized signing algorithms the same randomness is used to compute the tag for all slots in the superposition. Alternatively, we could have required fresh randomness in every slot, but this would make it harder to implement the MAC system on a quantum device. Allowing the same randomness in every slot is more conservative and frees the signer from this concern. At any rate, the two models are very close — if need be, the random string $r$ can be used as a key for a quantum-secure PRF [Zha12b] which is used to generate a fresh pseudorandom value for every slot.

**Existential forgery.** After issuing $q$ quantum chosen message queries the adversary wins the game if it can generate $q + 1$ valid classical message-tag pairs.

**Definition 2.1.** *A MAC system is existentially unforgeable under a quantum chosen message attack (EUF-qCMA) if no adversary can with the quantum MAC game with non-negligible advantage in* $\lambda$.

Zhandry [Zha12b] gives an example of a classically secure PRF that is insecure under quantum queries. This PRF gives an example MAC that is classically secure, but insecure under quantum queries. Our goal for the remainder of the paper is to construct EUF-qCMA secure MACs.

## 3 The Rank Method

In this section we introduce the rank method which is a general approach to proving lower bounds on quantum algorithms. The setup is as follows: we give a quantum algorithm $A$ access to some quantity $H \in \mathcal{H}$. By access, we mean that the final state of the algorithm is some fixed function of $H$. In this paper, $\mathcal{H}$ will be a set of functions, and $A$ will be given oracle access to $H \in \mathcal{H}$ by allowing $A$ to make $q$ quantum oracle queries to $H$, for some $q$. For now, we will treat $\mathcal{H}$ abstractly, and return to the specific case where $\mathcal{H}$ is a set of functions later.

The idea behind the rank method is that, if we treat the final states of the algorithm on different $H$ as vectors, the space spanned by these vectors will be some subspace of the overall Hilbert space. If the dimension of this subspace is small enough, the subspace (and hence all of the vectors in it) must be reasonably far from most of the vectors in the measurement basis. This allows us to bound the ability of such an algorithm to achieve some goal.

For $H \in \mathcal{H}$, let $|\psi_H\rangle$ be the final state of the quantum algorithm $A$, before measurement, when given access to $H$. Suppose the different $|\psi_H\rangle$ vectors all lie in a space of dimension $d$. Let $\mathbf{\Psi}_{A,\mathcal{H}}$ be the the $|\mathcal{H}| \times d$ matrix whose rows are the various vectors $|\psi_H\rangle$.

**Definition 3.1.** *For a quantum algorithm $A$ given access to some value $H \in \mathcal{H}$, we define the* rank, *denoted* $\operatorname{rank}(A, \mathcal{H})$, *as the rank of the matrix $\mathbf{\Psi}_{A,\mathcal{H}}$.*

The rank of an algorithm $A$ seemingly contains very little information: it gives the dimension of the subspace spanned by the $|\psi_H\rangle$ vectors, but gives no indication of the orientation of this subspace or the positions of the $|\psi_H\rangle$ vectors in the subspace. Nonetheless, we demonstrate how the success probability of an algorithm can be bounded from above knowing only the rank of $\mathbf{\Psi}_{A,\mathcal{H}}$.

**Theorem 3.2.** *Let $A$ be a quantum algorithm that has access to some value $H \in \mathcal{H}$ drawn from some distribution $D$ and produces some output $w \in \mathcal{W}$. Let $R : \mathcal{H} \times \mathcal{W} \to \{\texttt{True}, \texttt{False}\}$ be a binary relation. Then the probability that $A$ outputs some $w$ such that $R(H, w) = \texttt{True}$ is at most*

$$\left(\max_{w \in \mathcal{W}} \Pr_{H \leftarrow D}[R(H, w)]\right) \times \operatorname{rank}(A, \mathcal{H}) \ .$$

*In other words, the probability that $A$ succeeds in producing $w \in \mathcal{W}$ for which $R(H, w)$ is true is at most* $\operatorname{rank}(A, \mathcal{H})$ *times the best probability of success of any algorithm that ignores $H$ and just outputs some fixed $w$.*

**Proof**. The probability that $A$ outputs a $w$ such that $R(H, w) = \texttt{True}$ is

$$\Pr_{\substack{H \leftarrow D \\ w \leftarrow |\psi_H\rangle}}[R(H, w)] = \sum_H \Pr_D[H] \sum_{w : R(H,w)} |\langle w|\psi_H\rangle|^2 = \sum_w \sum_{H : R(H,w)} \Pr_D[H] |\langle w|\psi_H\rangle|^2$$

Now, $|\langle w|\psi_H\rangle|$ is just the magnitude of the projection of $|w\rangle$ onto the space spanned by the vector $|\psi_H\rangle$, that is, $\operatorname{proj}_{\operatorname{span}|\psi_H\rangle}(|w\rangle)$. This is at most the magnitude of the projection of $|w\rangle$ onto the space spanned by all of the $|\psi_{H'}\rangle$ for $H' \in \mathcal{H}$, or $\operatorname{proj}_{\operatorname{span}\{|\psi_{H'}\rangle\}}(|w\rangle)$. Thus,

$$\Pr_{\substack{H \leftarrow D \\ w \leftarrow |\psi_H\rangle}}[R(z, w)] \leq \sum_w \left(\sum_{H : R(H,w)} \Pr_D[H]\right) \left|\operatorname{proj}_{\operatorname{span}\{|\psi_{H'}\rangle\}}(|w\rangle)\right|^2$$

Now, we can perform the sum over $H$, which gives $\Pr_{H \leftarrow D}[R(H, w)]$. We can bound this by the maximum it attains over all $w$, giving us

$$\Pr_{\substack{H \leftarrow D \\ w \leftarrow |\psi_H\rangle}}[R(H, w)] \leq \left(\max_w \Pr_{H \leftarrow D}[R(H, w)]\right) \sum_w \left|\operatorname{proj}_{\operatorname{span}\{|\psi_{H'}\rangle\}}(|w\rangle)\right|^2$$

Now, let $|b_i\rangle$ be an orthonormal basis for $\operatorname{span}\{|\psi_{H'}\rangle\}$. Then

$$\left|\operatorname{proj}_{\operatorname{span}\{|\psi_{H'}\rangle\}}(|w\rangle)\right|^2 = \sum_i |\langle b_i|w\rangle|^2$$

Summing over all $w$ gives

$$\sum_w \left|\operatorname{proj}_{\operatorname{span}\{|\psi_{H'}\rangle\}}(|w\rangle)\right|^2 = \sum_w \sum_i |\langle b_i|w\rangle|^2 = \sum_i \sum_w |\langle b_i|w\rangle|^2$$

Since the $w$ are the possible results of measurement, the vectors $|w\rangle$ form an orthonormal basis for the whole space, meaning $\sum_w |\langle b_i|w\rangle|^2 = |\,|b_i\rangle\,|^2 = 1$. Hence, the sum just becomes the number of $|b_i\rangle$, which is just the dimension of the space spanned by the $|\psi_{H'}\rangle$. Thus,

$$\Pr_{\substack{H\leftarrow D\\ w\leftarrow|\psi_H\rangle}}[R(H,w)] \le \left(\max_{w\in\mathcal{W}} \Pr_{H\leftarrow D}[R(H,w)]\right)(\dim\operatorname{span}\{|\psi_{H'}\rangle\}) \ .$$

But $\dim\operatorname{span}\{|\psi_{H'}\rangle\}$ is exactly $\operatorname{rank}(\mathbf{\Psi}_{A,\mathcal{H}}) = \operatorname{rank}(A,\mathcal{H})$, which finishes the proof of the theorem. $\square$

We now move to the specific case of oracle access. $\mathcal{H}$ is now some set of functions from $\mathcal{X}$ to $\mathcal{Y}$, and our algorithm $A$ makes $q$ quantum oracle queries to a function $H \in \mathcal{H}$. Concretely, $A$ is specified by $q+1$ unitary matrices $\mathbf{U}_i$, and the final state of $A$ on input $H$ is the state

$$\mathbf{U}_q\mathbf{H}\mathbf{U}_{q-1}\cdots\mathbf{U}_1\mathbf{H}\mathbf{U}_0|0\rangle$$

where $\mathbf{H}$ is the unitary transformation mapping $|x,y,z\rangle$ into $|x,y+H(x),z\rangle$, representing an oracle query to the function $H$. To use the rank method (Theorem 3.2) for our purposes, we need to bound the rank of such an algorithm. First, we define the following quantity:

$$C_{k,q,n} \equiv \sum_{r=0}^{q}\binom{k}{r}(n-1)^r \ .$$

**Theorem 3.3.** *Let $\mathcal{X}$ and $\mathcal{Y}$ be sets of size $m$ and $n$ and let $H_0$ be some function from $\mathcal{X}$ to $\mathcal{Y}$. Let $\mathcal{S}$ be a subset of $\mathcal{X}$ of size $k$ and let $\mathcal{H}$ be some set of functions from $\mathcal{X}$ to $\mathcal{Y}$ that are equal to $H_0$ except possibly on points in $\mathcal{S}$. If $A$ is a quantum algorithm making $q$ queries to an oracle drawn from $\mathcal{H}$, then*

$$\operatorname{rank}(A,\mathcal{H}) \le C_{k,q,n} \ .$$

**Proof**. Let $|\psi_H^q\rangle$ be the final state of a quantum algorithm after $q$ quantum oracle calls to an oracle $H\in\mathcal{H}$. We wish to bound the dimension of the space spanned by the vectors $|\psi_H^q\rangle$ for all $H\in\mathcal{H}$. We accomplish this by exhibiting a spanning set for this space. Our basis consists of $|\psi_{H'}^q\rangle$ vectors where $H'$ only differs from $H_0$ at a maximum of $q$ points in $\mathcal{S}$. We need to show that two things: that our basis consists of $C_{k,q,n}$ vectors, and that our basis does in fact span the whole space.

We first count the number of basis vectors by counting the number of $H'$ oracles. For each $r$, there are $\binom{k}{r}$ ways of picking the subset $\mathcal{T}$ of size $r$ from $\mathcal{S}$ where $H'$ will differ from $H_0$. For each subset $\mathcal{T}$, there are $n^r$ possible functions $H'$. However, if any value $x\in\mathcal{T}$ satisfies $F(x) = H_0(x)$, then this is equivalent to a case where we remove $x$ from $\mathcal{T}$, and we would have already counted this case for a smaller value of $r$. Thus, we can assume $H'(x)\neq H_0(x)$ for all $x$ in $\mathcal{T}$. There are $(n-1)^r$ such functions. Summing over all $r$, we get that the number of distinct $H'$ oracles is

$$\sum_{r=0}^{q}\binom{k}{r}(n-1)^r = C_{k,q,n} \ .$$

Next, we need to show that the $|\psi_{H'}^q\rangle$ vectors span the entire space of $|\psi_H^q\rangle$ vectors. We first introduce some notation: let $|\psi^0\rangle$ be the state of a quantum algorithm before any quantum queries. Let $|\psi_H^q\rangle$ be the state after $q$ quantum oracle calls to the oracle $H$. Let

$$\mathbf{M}_H^q = \mathbf{U}_q\mathbf{H}\mathbf{U}_{q-1}\mathbf{H}\cdots\mathbf{U}_1\mathbf{H} \ .$$

Then $|\psi_H^q\rangle = \mathbf{M}_H^q|\psi^0\rangle$.

We note that since $|\psi^0\rangle$ is fixed for any algorithm, it is sufficient to prove that the $\mathbf{M}_H^q$ matrices are spanned by the $\mathbf{M}_{H'}^q$.

For any subset $\mathcal{T}$ of $\mathcal{S}$, and a function $F : \mathcal{T} \to \mathcal{Y}$, let $J_{\mathcal{T},F}$ be the oracle such that

$$J_{\mathcal{T},F}(x) = \begin{cases} F(x) & \text{if } x \in \mathcal{T} \\ H_0(x) & \text{otherwise} \end{cases} .$$

Let $\mathbf{M}_{\mathcal{T},H}^q$ denote $\mathbf{M}_{J_{\mathcal{T},H}}^q$. In other words, $\mathbf{M}_{\mathcal{T},H}$ is the transformation matrix corresponding to the oracle that is equal to $H$ on the set $\mathcal{T}$, and equal to $H_0$ elsewhere. We claim that any $\mathbf{M}_H^q$ for $H \in \mathcal{H}_{\mathcal{S}}$ is a linear combination of the matrices $\mathbf{M}_{\mathcal{T},H}^q$ for subsets $\mathcal{T}$ of $\mathcal{S}$ of size at most $q$. We will fix a particular $H$, and for convenience of notation, we will let $J_{\mathcal{T}} = J_{\mathcal{T},H}$. That is, $J_{\mathcal{T}}$ is the oracle that is equal to $H$ on the set $\mathcal{T}$ and $H_0$ otherwise. We will also let $\mathbf{M}_{\mathcal{T}}^q = \mathbf{M}_{\mathcal{T},H}^q$ and $\mathbf{M}^q = \mathbf{M}_H^q$. That is, $\mathbf{M}^q$ is the transition matrix corresponding to the oracle $H$, and $\mathbf{M}_{\mathcal{T}}$ is the transition matrix corresponding to using the oracle $J_{\mathcal{T}}$. For the singleton set $\{x\}$, we will also let $J_x = J_{\{x\}}$.

We make the following observations:

$$\mathbf{H} = \left(\sum_{x\in\mathcal{S}} \mathbf{J}_x\right) - (k-1)\mathbf{H}_0 \tag{3.1}$$

$$\mathbf{J}_{\mathcal{T}} = \left(\sum_{x\in\mathcal{T}} \mathbf{J}_x\right) - (|\mathcal{T}|-1)\mathbf{H}_0 \tag{3.2}$$

These identities can be seen by applying each side to the different inputs. Next, we take $\mathbf{M}_H^q$ and $\mathbf{M}_{\mathcal{T}}^q$ and expand out the $\mathbf{H}$ and $\mathbf{J}_{\mathcal{T}}$ terms using Equations 3.1 and 3.2:

$$\mathbf{M}^q = \mathbf{U}_q\left(\left(\sum_{x\in\mathcal{S}} \mathbf{J}_x\right) - (k-1)\mathbf{H}_0\right)\mathbf{U}_{q-1}\cdots\mathbf{U}_1\left(\left(\sum_{x\in\mathcal{S}} \mathbf{J}_x\right) - (k-1)\mathbf{H}_0\right) \tag{3.3}$$

$$\mathbf{M}_{\mathcal{T}}^q = \mathbf{U}_q\left(\left(\sum_{x\in\mathcal{T}} \mathbf{J}_x\right) - (|\mathcal{T}|-1)\mathbf{H}_0\right)\mathbf{U}_{q-1}\cdots\mathbf{U}_1\left(\left(\sum_{x\in\mathcal{T}} \mathbf{J}_x\right) - (|\mathcal{T}|-1)\mathbf{H}_0\right) \tag{3.4}$$

Let $\mathbf{J}_\perp = \mathbf{H}_0$. For a vector $\mathbf{r} \in (\mathcal{S} \cup \{\perp\})^q$, let

$$\mathbf{P}_{\mathbf{r}} = \mathbf{U}_q\mathbf{J}_{r_q}\mathbf{U}_{q-1}\cdots\mathbf{J}_{r_2}\mathbf{U}_1\mathbf{J}_{r_1}$$

For a particular $\mathbf{r}$, we wish to expand the $\mathbf{M}^q$ and $\mathbf{M}_{\mathcal{T}}^q$ matrices in terms of the $\mathbf{P}_{\mathbf{r}}$ matrices. If $d$ of the components of $\mathbf{r}$ are $\perp$, then the coefficient of $\mathbf{P}_{\mathbf{r}}$ in the expansion of $\mathbf{M}^q$ is $(-1)^d(k-1)^d$. If, in addition, all of the other components of $\mathbf{r}$ lie in $\mathcal{T}$, then the coefficient in the expansion of $\mathbf{M}_{\mathcal{T}}^q$ is $(-1)^d(|\mathcal{T}|-1)^d$ (if any of the components of $\mathbf{r}$ lie outside of $\mathcal{T}$, the coefficient is 0).

Now, we claim that, for some values $a_\ell$, we have

$$\mathbf{M}^q = \sum_{\ell=0}^{q} a_\ell \sum_{\mathcal{T}\subseteq\mathcal{S}:|\mathcal{T}|=\ell} \mathbf{M}_{\mathcal{T}}^q$$

To accomplish this, we look for the coefficient of $\mathbf{P}_{\mathbf{r}}$ in the expansion of the right hand side of this equation. Fix an $\ell$. Let $d$ be the number of components of $\mathbf{r}$ equal to $\perp$, and let $p$ be the number of distinct component values other than $\perp$. Notice that $p+d \leq q$. Then there are $\binom{k-p}{\ell-p}$ different sets $\mathcal{T}$ of size $\ell$ for which all of the values of the components lie in $\mathcal{T}$. Thus, the coefficient of $\mathbf{P_r}$ is

$$\sum_{\ell=p}^{q} a_\ell \binom{k-p}{\ell-p} (-1)^i (\ell-1)^d$$

Therefore, we need values $a_\ell$ such that

$$\sum_{\ell=p}^{q} a_\ell \binom{k-p}{\ell-p} (\ell-1)^d = (k-1)^d \tag{3.5}$$

for all $d, p$. Notice that we can instead phrase this problem as a polynomial interpolation problem. The right hand side of Equation 3.5 is a polynomial $P$ of degree $d \leq q-p$, evaluated at $k-1$. We can interpolate this polynomial using the points $\ell = p, ..., q$, obtaining

$$P(k-1) = \sum_{\ell=p}^{q} P(\ell-1) \prod_{j=p, j\neq\ell}^{q} \frac{k-j}{\ell-j} \quad .$$

The numerator of the product evaluates to

$$\frac{(k-p)!}{(k-\ell)(k-q-1)!}$$

while to evaluate the bottom, we split it into two parts: $j = p, ..., \ell-1$ and $j - \ell+1, ..., q$. The first part evaluates to $(\ell-p)!$, and the second part evaluates to $(-1)^{q-\ell}(q-\ell)!$. With a little algebraic manipulation, we have that

$$P(k-1) = \sum_{\ell=p}^{q} P(\ell-1) \left( \binom{k-\ell-1}{k-q-1} (-1)^{q-\ell} \right) \binom{k-p}{\ell-p}$$

for all polynomials $P(x)$ of degree at most $q-p$. Setting $P(x) = x^d$ for $d = 0, ..., q-\ell$, we see that Equation 3.5 is satisfied if

$$a_\ell = \binom{k-1-\ell}{k-1-q} (-1)^{q-\ell} \quad .$$

$\square$

## 3.1 An Example

Suppose our task is to, given one quantum query to an oracle $H : \mathcal{X} \to \mathcal{Y}$, produce two distinct pairs $(x_0, y_0)$ and $(x_1, y_1)$ such that $H(x_0) = y_0$ and $H(x_1) = y_1$. Suppose further that $H$ is drawn from a pairwise independent set $\mathcal{H}$. We will now see that the rank method leads to a bound on the success probability of any quantum algorithm $A$.

**Corollary 3.4.** *No quantum algorithm $A$, making a single query to a function $H : \mathcal{X} \to \mathcal{Y}$ drawn from a pairwise independent set $\mathcal{H}$, can produce two distinct input/output pairs of $H$, except with probability at most $|\mathcal{X}|/|\mathcal{Y}|$.*

**Proof**. Let $m = |\mathcal{X}|$ and $n = |\mathcal{Y}|$. Since no outputs of $H$ are fixed, we will set $\mathcal{S} = \mathcal{X}$ in Theorem 3.3, showing that the rank of the algorithm $A$ is bounded by $C_{m,1,n} = 1 + m(n-1) < mn$. If an algorithm makes no queries to $H$, the best it can do at outputting two distinct input/output pairs is to just pick two arbitrary distinct pairs, and output those. The probability that this zero-query algorithm succeeds is at most $1/n^2$. Then Theorem 3.2 tells us that $A$ succeeds with probability at most $\text{rank}(A, \mathcal{H})$ times this amount, which equates to $\frac{m}{n}$. □

For $m > n$, this bound is trivial. However, for $m$ smaller than $n$, this gives a non-trivial bound, and for $m$ exponentially smaller than $n$, this bound is negligible.

## 4 Outputting Values of a Random Oracle

In this section, we will prove Theorem 1.1. We consider the following problem: given $q$ quantum queries to a random oracle $H : \mathcal{X} \to \mathcal{Y}$, produce $k > q$ distinct pairs $(x_i, y_i)$ such that $y_i = H(x_i)$. Let $n = |\mathcal{Y}|$ be the size of the range. Motivated by our application to quantum-accessible MACs, we are interested in the case where the range $\mathcal{Y}$ of the oracle is large, and we want to show that to produce even one extra input/output pair ($k = q + 1$) is impossible, except with negligible probability. We are also interested in the case where the range of the oracle, though large, is far smaller than the domain. Thus, the bound we obtained in the previous section (Corollary 3.4) is not sufficient for our purposes, since it is only non-trivial if the range is larger than the domain.

In the classical setting, when $k \leq q$, this problem is easy, since we can just pick an arbitrary set of $k$ different $x_i$ values, and query the oracle on each value. For $k > q$, no adversary of even unbounded complexity can solve this problem, except with probability $1/n^{k-q}$, since for any set of $k$ inputs, at least $k - q$ of the corresponding outputs are completely unknown to the adversary. Therefore, for large $n$, we have have a sharp threshold: for $k \leq q$, this problem can be solved efficiently with probability 1, and for even $k = q+1$, this problem cannot be solved, even inefficiently, except with negligible probability.

In the quantum setting, the $k \leq q$ case is the same as before, since we can still query the oracle classically. However, for $k > q$, the quantum setting is more challenging. The adversary can potentially query the random oracle on a superposition of all inputs, so he "sees" the output of the oracle on all points. Proving that it is still impossible to produce $k$ input/output pairs is thus more complicated, and existing methods fail to prove that this problem is difficult. Therefore, it is not immediately clear that we have the same sharp threshold as before.

In Section 4.1 we use the rank method to bound the probability that any (even computationally unbounded) quantum adversary succeeds. Then in Section 4.2 we show that our bound is tight by giving an efficient algorithm for this problem that achieves the lower bound. In particular, for an oracle $H : \mathcal{X} \to \mathcal{Y}$ we consider two cases:

- Exponentially-large range $\mathcal{Y}$ and polynomial $k, q$. In this case, we will see that the success probability even when $k = q + 1$ is negligible. That is, to produce even one additional input/output pair is hard. Thus, we get the same sharp threshold as in the classical case
- Constant size range $\mathcal{Y}$ and polynomial $k, q$. We show that even when $q$ is a constant fraction of $k$ we can still produce $k$ input/output pairs with overwhelming probability using only $q$ quantum queries. This is in contrast to the classical case, where the success probability for $q = ck$, $c < 1$, is negligible in $k$.

## 4.1 A Tight Upper Bound

**Theorem 4.1.** *Let $A$ be a quantum algorithm making $q$ queries to a random oracle $H : \mathcal{X} \to \mathcal{Y}$ whose range has size $n$, and produces $k > q$ pairs $(x_i, y_i) \in \mathcal{X} \times \mathcal{Y}$. The probability that the $x_i$ values are distinct and $y_i = H(x_i)$ for all $i \in [k]$ is at most $\frac{1}{n^k} C_{k,q,n}$.*

**Proof**. Before giving the complete proof, we sketch the special case where $k$ is equal to the size of the domain. In this case, any quantum algorithm that outputs $k$ distinct input/output pairs must output *all* input/output pairs. Similar to the proof of Corollary 3.4, we will set $\mathcal{S} = \mathcal{X}$, and use Theorem 3.3 to bound the rank of $A$ at $C_{k,q,n}$. Now, any algorithm making *zero* queries succeeds with probability at most $1/n^k$. Theorem 3.2 then bounds the success probability of any $q$ query algorithm as

$$\frac{1}{n^k} C_{k,q,n} \quad .$$

Now for the general proof: first, we will assume that the probability $A$ outputs any particular sequence of $x_i$ values is independent of the oracle $H$. We will show how to remove this assumption later. We can thus write

$$|\psi_H^q\rangle = \sum_{\mathbf{x}} \alpha_{\mathbf{x}} |\mathbf{x}\rangle |\phi_{H,\mathbf{x}}\rangle$$

where $\alpha_{\mathcal{X}}$ are complex numbers whose square magnitudes sum to one, and $|\mathbf{x}\rangle|\phi_{H,\mathbf{x}}\rangle$ is the normalized projection of $|\psi_H^q\rangle$ onto the space spanned by $|\mathbf{x}, w\rangle$ for all $w$. The probability that $A$ succeeds is equal to

$$\sum_H \Pr[H] \sum_{\mathbf{x}} |\langle \mathbf{x}, H(\mathbf{x}) | \psi_H^q \rangle|^2 = \sum_H \Pr[H] \sum_{\mathbf{x}} |\alpha_{\mathbf{x}}|^2 |\langle H(\mathbf{x}) | \phi_{H,\mathbf{x}} \rangle|^2 \quad .$$

First, we reorder the sums so the outer sum is the sum over $\mathbf{x}$. Now, we write $H = (H_0, H_1)$ where $H_0$ is the oracle restricted to the components of $\mathbf{x}$, and $H_1$ is the oracle restricted to all other inputs. Thus, our probability is:

$$\frac{1}{n^m} \sum_{\mathbf{x}} |\alpha_{\mathbf{x}}|^2 \sum_{H_0, H_1} \left| \langle H_0(\mathbf{x}) | \phi_{(H_0,H_1),\mathbf{x}} \rangle \right|^2 \quad .$$

Using the same trick as we did before, we can replace $|\langle H(\mathbf{x}) | \phi_{H,\mathbf{x}} \rangle|$ with the quantity

$$\left| \mathrm{proj}_{\mathrm{span}|\phi_{(H_0,H_1),\mathbf{x}}\rangle} |H_0(\mathcal{X})\rangle \right| \quad ,$$

which is bounded by $\left| \mathrm{proj}_{\mathrm{span}\{|\phi_{(H_0',H_1),\mathbf{x}}\rangle\}} |H_0(\mathbf{x})\rangle \right|$ as we vary $H_0'$ over oracles whose domain is the components of $\mathbf{x}$. The probability of success is then bounded by

$$\frac{1}{n^m} \sum_{\mathbf{x}} |\alpha_{\mathbf{x}}|^2 \sum_{H_0, H_1} \left| \mathrm{proj}_{\mathrm{span}\{|\phi_{(H_0',H_1),\mathbf{x}}\rangle\}} |H_0(\mathbf{x})\rangle \right|^2 \quad .$$

We now perform the sum over $H_0$. Like in the proof of Corollary 3.4, the sum evaluates to $\dim \mathrm{span}\{|\phi_{(H_0',H_1),\mathbf{x}}\rangle\}$. Since the $|\phi_{(H_0',H_1),\mathbf{x}}\rangle$ vectors are projections of $|\psi_H^q\rangle$, this dimension is bounded by $\dim \mathrm{span}\{\left|\psi_{(H_0',H_1)}^q\right\rangle\}$. Let $\mathcal{H}$ be the set of oracles $(H_0', H_1)$ as we vary $H_0'$, and consider $A$ acting on oracles in $\mathcal{H}$. Fix some oracle $H_0^*$ from among the $H_0'$ oracles, and let $\mathcal{S}$ be the set of components of $\mathbf{x}$. Then $(H_0', H_1)$ differs from $(H_0^*, H_1)$ only on the elements of $\mathcal{S}$. Since $|\mathcal{S}| \leq k$, Theorem 3.2 tells us that $\text{rank}(A, \mathcal{H}) \leq C_{k,q,n}$. But

$$\text{rank}(A, \mathcal{H}) = \dim \text{span}\{\left|\psi^q_{(H_0', H_1)}\right\rangle\}$$

Therefore, we can bound the success probability by

$$\frac{1}{n^m} \sum_{\mathbf{x}} |\alpha_{\mathbf{x}}|^2 \sum_{H_1} C_{k,q,n} \quad .$$

Summing over all $n^{m-k}$ different $H_1$ values and all $\mathbf{x}$ values gives a bound of

$$\frac{1}{n^k} C_{k,q,n}$$

as desired.

So far, we have assume that $A$ produces $\mathbf{x}$ with probability independent of $H$. Now, suppose our algorithm $A$ does not produce $\mathbf{x}$ with probability independent of the oracle. We construct a new algorithm $B$ with access to $H$ that does the following: pick a random oracle $O$ with the same domain and range as $H$, and give $A$ the oracle $H + O$ that maps $x$ into $H(x) + O(x)$. When $A$ produces $k$ input/output pairs $(x_i, y_i)$, output the pairs $(x_i, y_i - O(x_i))$. $(x_i, y_i)$ are input/output pairs of $H + O$ if and only if $(x_i, y_i - O(x_i))$ are input/output pairs of $H$. Further, $A$ still sees a random oracle, so it succeeds with the same probability as before. Moreover, the oracle $A$ sees is now independent of $H$, so $B$ outputs $\mathbf{x}$ with probability independent of $H$. Thus, applying the above analysis to $B$ shows that $B$, and hence $A$, produce $k$ input/output pairs with probability at most

$$\frac{1}{n^k} C_{k,q,n}$$

□

For this paper, we are interested in the case where $n = |\mathcal{Y}|$ is exponentially large, and we are only allowed a polynomial number of queries. Suppose $k = q + 1$, the easiest non-trivial case for the adversary. Then, the probability of success is

$$\frac{1}{n^{q+1}} \sum_{r=0}^{q} \binom{q+1}{r} (n-1)^r = 1 - \left(1 - \frac{1}{n}\right)^{q+1} \leq \frac{q+1}{n} \quad . \tag{4.1}$$

Therefore, to produce even one extra input/output pair is impossible, except with exponentially small probability, just like in the classical case. This proves the first part of Theorem 1.1.

## 4.2 The Optimal Attack

In this section, we present a quantum algorithm for the problem of computing $H(x_i)$ for $k$ different $x_i$ values, given only $q < k$ queries:

**Theorem 4.2.** *Let $\mathcal{X}$ and $\mathcal{Y}$ be sets, and fix integers $q < k$, and $k$ distinct values $x_1, ..., x_k \in \mathcal{X}$. There exists a quantum algorithm $A$ that makes $q$ non-adaptive quantum queries to any function $H : \mathcal{X} \to \mathcal{Y}$, and produces $H(x_1), ..., H(x_k)$ with probability $C_{k,q,n}/n^k$, where $n = |\mathcal{Y}|$.*

The algorithm is similar to the algorithm of [vD98], though generalized to handle arbitrary range sizes. This algorithm has the same success probability as in Theorem 4.1, showing that both our attack and lower bound of Theorem 4.1 are optimal. This proves the second part of Theorem 1.1.

**Proof**. Assume that $\mathcal{Y} = \{0, ..., n-1\}$. For a vector $\mathbf{y} \in \mathcal{Y}^k$, let $\Delta(\mathbf{y})$ be the number of coordinates of $\mathbf{y}$ that do not equal 0. Also, assume that $x_i = i$.

Initially, prepare the state that is a uniform superposition of all vectors $\mathbf{y} \in \mathcal{Y}^k$ such that $\Delta(\mathbf{y}) \le q$:

$$|\psi_1\rangle = \frac{1}{\sqrt{V}} \sum_{\mathbf{y}:\Delta(\mathbf{y})\le q} |\mathbf{y}\rangle$$

Notice that the number of vectors of length $k$ with at most $q$ non-zero coordinates is exactly

$$\sum_{r=0}^{q} \binom{k}{r} (n-1)^r = C_{k,q,n} \quad .$$

We can prepare the state efficiently as follows: Let $\mathsf{Setup}_{k,q,n} : [C_{k,q,n}] \to [n]^k$ be the following function: on input $\ell \in [C_{k,q,n}]$,

- Check if $\ell \le C_{k-1,q,n}$. If so, compute the vector $\mathbf{y}' = \mathsf{Setup}_{k-1,q,n}(n)$, and output the vector $\mathbf{y} = (0, \mathbf{y}')$.
- Otherwise, let $\ell' = \ell - C_{k-1,q,n}$. It is easy to verify that $\ell' \in [(n-1)C_{k-1,q-1,n}]$.
- Let $\ell'' \in C_{k-1,q-1,n}$ and $y_0 \in [n]\backslash\{0\}$ be the unique such integers such that $\ell' = (n-1)\ell'' + y_0 - n$.
- Let $\mathbf{y}' = \mathsf{Setup}_{k-1,q-1,n}(\ell'')$, and output the vector $\mathbf{y} = (y_0, \mathbf{y}')$.

The algorithm relies on the observation that a vector $\mathbf{y}$ of length $k$ with at most $q$ non-zero coordinates falls into one of either two categories:

- The first coordinate is 0, and the remaining $k-1$ coordinates form a vector with at most $q$ non-zero coordinates
- The first coordinate is non-zero, and the remaining $k-1$ coordinates form a vector with at most $q-1$ non-zero coordinates.

There are $C_{k-1,q,n}$ vectors of the first type, and $C_{k-1,q-1,n}$ vectors of the second type for each possible setting of the first coordinate to something other than 0. Therefore, we divide $[A_{k,q,n}]$ into two parts: the first $C_{k-1,q,n}$ integers map to the first type, and the remaining $(n-1)C_{k-1,q-1,n}$ integers map to vectors of the second type.

We note that $\mathsf{Setup}$ is efficiently computable, invertible, and its inverse is also efficiently computable. Therefore, we can prepare $|\psi_1\rangle$ by first preparing the state

$$\frac{1}{\sqrt{C_{k,q,n}}} \sum_{\ell \in [C_{k,q,n}]} |\ell\rangle$$

and reversibly converting this state into $|\phi_1\rangle$ using $\mathsf{Setup}_{k,q,n}$.

Next, let $F : \mathcal{Y}^k \to [k]^q$ be the function that outputs the indexes $i$ such that $\mathbf{y}_i \neq 0$, in order of increasing $i$. If there are fewer than $q$ such indexes, the function fills in the remaining spaces the first indexes such that $\mathbf{y}_i = 0$ If there are more than $q$ indexes, the function truncates to the first $q$. $F$ is realizable by a simple classical algorithm, so it can be implemented as a quantum algorithm. Apply this algorithm to $|\psi_1\rangle$, obtaining the state

$$|\psi_2\rangle = \frac{1}{\sqrt{C_{k,q,n}}} \sum_{\mathbf{y}:\Delta(\mathbf{y})\leq q} |\mathbf{y}, F(\mathbf{y})\rangle$$

Next, let $G : \mathcal{Y}^k \to \mathcal{Y}_q$ be the function that takes in vector $\mathbf{y}$, computes $\mathbf{x} = F(\mathbf{y})$, and outputs the vector $(y_{x_1}, y_{x_2}, ..., y_{x_q})$. In other words, it outputs the vector of the non-zero components of $\mathbf{y}$, padding with zeros if needed. This function is also efficiently computable by a classical algorithm, so we can apply if to each part of the superposition:

$$|\psi_3\rangle = \frac{1}{\sqrt{C_{k,q,n}}} \sum_{\mathbf{y}:\Delta(\mathbf{y})\leq q} |\mathbf{y}, F(\mathbf{y}), G(\mathbf{y})\rangle$$

Now we apply the Fourier transform to the $G(\mathbf{y})$ part, obtaining

$$|\psi_4\rangle = \frac{1}{\sqrt{C_{k,q,n}}} \sum_{\mathbf{y}:\Delta(\mathbf{y})\leq q} |\mathbf{y}, F(\mathbf{y})\rangle \sum_{\mathbf{z}} e^{-i\frac{2\pi}{n}\langle \mathbf{z}, G(\mathbf{y})\rangle} |\mathbf{z}\rangle$$

Now we can apply $H$ to the $F(\mathbf{y})$ part using $q$ non-adaptive queries, adding the answer to the $\mathbf{z}$ part. The result is the state

$$|\psi_5\rangle = \frac{1}{\sqrt{C_{k,q,n}}} \sum_{\mathbf{y}:\Delta(\mathbf{y})\leq q} |\mathbf{y}, F(\mathbf{y})\rangle \sum_{\mathbf{z}} e^{-i\frac{2\pi}{n}\langle \mathbf{z}, G(\mathbf{y})\rangle} |\mathbf{z} + H(F(\mathbf{y}))\rangle$$

We can rewrite this last state as follows:

$$|\psi_5\rangle = \frac{1}{\sqrt{C_{k,q,n}}} \sum_{\mathbf{y}:\Delta(\mathbf{y})\leq q} e^{i\frac{2\pi}{n}\langle H(F(\mathbf{y})), G(\mathbf{y})\rangle} |\mathbf{y}, F(\mathbf{y})\rangle \sum_{\mathbf{z}} e^{-i\frac{2\pi}{n}\langle \mathbf{z}, G(\mathbf{y})\rangle} |\mathbf{z}\rangle$$

Now, notice that $H(F(\mathbf{y}))$ is the vector of $H$ applied to the indexes where $\mathbf{y}$ is non-zero, and that $G(\mathbf{y})$ is the vector of values of $\mathbf{y}$ that those points. Thus the inner product is

$$\langle H(F(\mathbf{y}), G(\mathbf{y})\rangle = \sum_{i:y_i\neq 0} H(i) \times y_i = \sum_{i=0}^{k} H(i) y_i = \langle H([k]), \mathbf{y}\rangle \quad .$$

The next step is to uncompute the $\mathbf{z}$ and $F(\mathbf{y})$ registers, obtaining

$$|\psi_6\rangle = \frac{1}{\sqrt{C_{k,q,n}}} \sum_{\mathbf{y}:\Delta(\mathbf{y})\leq q} e^{i\frac{2\pi}{n}\langle H([k]), \mathbf{y}\rangle} |\mathbf{y}\rangle$$

Lastly, we perform a Fourier transform the remaining space, obtaining

$$|\psi_7\rangle = \frac{1}{\sqrt{C_{k,q,n} n^k}} \sum_{\mathbf{z}} \left( \sum_{\mathbf{y}:\Delta(\mathbf{y})\leq q} e^{i\frac{2\pi}{n}\langle H([k]) - \mathbf{z}, \mathbf{y}\rangle} \right) |\mathbf{z}\rangle$$

Now measure. The probability we obtain $H([k])$ is

$$\frac{1}{C_{k,q,n} n^k} \left| \sum_{\mathbf{y}:\Delta(\mathbf{y})\leq q} 1 \right|^2 = \frac{C_{k,q,n}}{n^k}$$

as desired. $\square$

As we have already seen, for exponentially-large $\mathcal{Y}$, this attack has negligible advantage for any $k > q$. However, if $n = |\mathcal{Y}|$ is constant, we can do better. The error probability is

$$\sum_{r=q+1}^{k} \binom{k}{r} \left(1 - \frac{1}{n}\right)^r \left(\frac{1}{n}\right)^{k-r} = \sum_{s=0}^{k-q-1} \binom{k}{s} \left(\frac{1}{n}\right)^s \left(1 - \frac{1}{n}\right)^{k-s} \quad .$$

This is the probability that $k$ consecutive coin flips, where each coin is heads with probability $1/n$, yields fewer than $k - q$ heads. Using the Chernoff bound, if $q > k(1 - 1/n)$, this probability is at most

$$e^{-\frac{n}{2k}(q-k(1-1/n))^2} \quad .$$

For a constant $n$, let $c$ be any constant with $1 - 1/n < c < 1$. If we use $q = ck$ queries, the error probability is less than

$$e^{-\frac{n}{2k}(k(c+1/n-1))^2} = e^{-\frac{nk}{2}(c+1/n-1)^2},$$

which is exponentially small in $k$. Thus, for constant $n$, and any constant $c$ with $1 - 1/n < c < 1$, using $q = ck$ quantum queries, we can determine $k$ input/output pairs with overwhelming probability. This is in contrast to the classical case, where with any constant fraction of $k$ queries, we can only produce $k$ input/output pairs with negligible probability. As an example, if $H$ outputs two bits, it is possible to produce $k$ input/output pairs of of $H$ using only $q = 0.8k$ quantum queries. However, with $0.8k$ classical queries, we can output $k$ input/output pairs with probability at most $4^{-0.2k} < 0.76^k$.

## 5 Quantum-Accessible MACs

Using Theorem 4.1 we can now show that a quantum secure pseudorandom function [Zha12b] gives rise to the quantum-secure MAC, namely $S(k, m) = \mathsf{PRF}(k, m)$. We prove that this mac is secure.

**Theorem 5.1.** *If* $\mathsf{PRF} : \mathcal{K} \times \mathcal{X} \to \mathcal{Y}$ *is a quantum-secure pseudorandom function and* $1/|\mathcal{Y}|$ *is negligible, then* $S(k, m) = \mathsf{PRF}(k, m)$ *is a EUF-qCMA-secure MAC.*

**Proof**. Let $A$ be a polynomial time adversary that makes $q$ quantum queries to $S(k, \cdot)$ and produces $q + 1$ valid input/output pairs with probability $\epsilon$. Let Game 0 be the standard quantum MAC attack game, where $A$ makes $q$ quantum queries to $MAC_k$. By definition, $A$'s success probability in this game is $\epsilon$.

Let Game 1 be the same as Game 0, except that $S(k, \cdot)$ is replaced with a truly random function $O : \mathcal{X} \to \mathcal{Y}$, and define $A$'s success probability as the probability that $A$ outputs $q + 1$ input/output pairs of $O$. Since $\mathsf{PRF}$ is a quantum-secure PRF, $A$'s advantage in distinguishing Game 0 from Game 1 is negligible.

Now, in Game 1, $A$ makes $q$ quantum queries to a random oracle, and tries to produce $q + 1$ input/output pairs. However, by Theorem 4.1 and Eq. (4.1) we know that $A$'s success probability is bounded by $(q + 1)/|\mathcal{Y}|$ which is negligible. It now follows that $\epsilon$ is negligible and therefore, $S$ is a EUF-qCMA-secure MAC. $\square$

## 5.1 Carter-Wegman MACs

In this section, we show how to modify the Carter-Wegman MAC so that it is secure in the quantum setting presented in Section 2.2. Recall that $H$ is an XOR-universal family of hash functions from $\mathcal{X}$ into $\mathcal{Y}$ if for any two distinct points $x$ and $y$, and any constant $c \in \mathcal{Y}$,

$$\Pr_{h \leftarrow H}[H(x) - H(y) = c] = 1/|\mathcal{Y}|$$

The Carter-Wegman construction uses a pseudorandom function family $\mathsf{PRF}$ with domain $\mathcal{X}$ and range $\mathcal{Y}$, and an XOR-universal family of hash functions $\mathcal{H}$ from $\mathcal{M}$ to $\mathcal{Y}$. The key is a pair $(k, H)$, where $k$ is a key for $\mathsf{PRF}$ and $H$ is a function drawn from $\mathcal{H}$. To sign a message, pick a random $r \in \mathcal{X}$, and return $(r,\ \mathsf{PRF}(k,r) + H(m))$.

This MAC is not, in general, secure in the quantum setting presented in Section 2.2. The reason is that the same randomness is used in all slots of a quantum chosen message query, that is the signing oracle computes:

$$\sum_m \alpha_m |m\rangle \longrightarrow \sum_m \alpha_m |m, r, \mathsf{PRF}(k,r) + H(m)\rangle$$

where the same $r$ is used for all classical states of the superposition. For example, suppose $\mathcal{H}$ is the set of functions $H(x) = ax + b$ for random $a$ and $b$. With even a single quantum query, the adversary will be able to obtain $a$ and $\mathsf{PRF}(k,r) + b$ with high probability, using the algorithm from Theorem 6.2 in Section 6. Knowing both of these will allow the adversary to forge any message.

We show how to modify the standard Carter-Wegman MAC to make it secure in the quantum setting.

**Construction 1** (Quantum Carter-Wegman)**.** *The Quantum Carter-Wegman MAC (QCW-MAC) is built from a pseudorandom function* $\mathsf{PRF}$*, an XOR-universal set of functions* $\mathcal{H}$*, and a pairwise independent set of functions* $\mathcal{R}$*.*

*Keys: The secret key for QCW-MAC is a pair* $(k, H)$*, where* $k$ *is a key for* $\mathsf{PRF}$ *and* $H : \mathcal{M} \to \mathcal{Y}$ *is drawn from* $\mathcal{H}$

*Signing: To sign a message* $m$ *choose a random* $R \in \mathcal{R}$ *and output the pair* $\big(R(m),\ \mathsf{PRF}(k, R(m)) + H(m)\ \big)$ *as the tag. When responding to a quantum chosen message query, the same* $R$ *is used in all classical states of the superposition.*

*Verification: To verify that* $(r, s)$ *is a valid tag for* $m$*, accept iff* $\mathsf{PRF}(k,r) + H(m) = s$*.*

**Theorem 5.2.** *The Quantum Carter-Wegman MAC is a EUF-qCMA secure MAC.*

**Proof**. We start with an adversary $A$ that makes $q$ tag queries, and then produces $q+1$ valid message/tag pairs with probability $\epsilon$. We now adapt the classical Carter-Wegman security proof to our MAC in the quantum setting.

When the adversary makes query $i$ on the superposition

$$\sum_{m,y,z} \alpha^{(i)}_{m,y,z} |m, y, z\rangle \ ,$$

the challenger responds with the superposition

$$\sum_{m,y,z} \alpha^{(i)}_{m,y,z} |m, y + S_i(m), z\rangle$$

where $S_i(m) = (R_i(m), \mathsf{PRF}(k, (R_i(m)) + H(m))$ for a randomly chosen $R_i \in \mathcal{R}$, where $\mathcal{R}$ is a pairwise independent set of functions.

The adversary then creates $q{+}1$ triples $(m_j, r_j, s_j)$ which, with probability $\epsilon$, are valid message/tag tuples. That means $H(m_j) + \mathsf{PRF}(k, r_j) = s_j$ for all $j$.

We now prove that $\epsilon$ must be small using a sequence of games:

Game 0: Run the standard MAC game, responding to query $i$ with the oracle that maps $m$ to $(R_i(m), \mathsf{PRF}(k, R_i(m)) + H(m))$, where $R_i$ is a random function from $\mathcal{R}$. The advantage of $A$ in this game is the probability is produces $q+1$ forgeries. Denote this advantage as $\epsilon_0$, which is equal to $\epsilon$.

Game 1: Replace $\mathsf{PRF}(k, \cdot)$ with a truly random function $F$, and denote the advantage in this game as $\epsilon_1$. Since $\mathsf{PRF}$ is a quantum-secure PRF, $\epsilon_1$ is negligibly close to $\epsilon_0$.

Game 2: Next we change the goal of the adversary. The adversary is now asked to produce a triple $(m_0, m_1, s)$ where $H(m_0) - H(m_1) = s$. Given an adversary $A$ for Game 1, we construct an adversary $B$ for Game 2 as follows: run $A$, obtaining $q+1$ forgeries $(m_j, r_j, s_j)$ such that $H(m_j) + F(r_j) = s_j$ with probability $\epsilon_1$. If all $r_j$ are distinct, abort. Otherwise, assume without loss of generality that $r_0 = r_1$. Then

$$H(m_0) - H(m_1) = (s_0 - F(r_0)) - (s_1 - F(r_1)) = s_0 - s_1$$

so output $(m_0, m_1, s_0 - s_1)$. Let $\epsilon_2$ be the advantage of $B$ in this game. Let $p$ be the probability that all $r_j$ are distinct and $A$ succeeds. Then $\epsilon_2 \geq \epsilon_1 - p$.

We wish to bound $p$. Define a new algorithm $C$, with oracle access to $F$, that first generates $H$, and then runs $A$, playing the role of challenger to $A$. When $A$ outputs $q+1$ triples $(m_j, r_j, s_j)$, $B$ outputs $q{+}1$ pairs $(r_j, s_j{-}H(m_j))$. If $A$ succeeded, then $H(m_j){+}F(r_j) = s_j$, so $F(r_j) = s_j{-}H(m_j)$, meaning the pairs $C$ outputs are all input/output pairs of $F$. If all the $r_j$ are distinct, then $C$ will output $q+1$ input/output pairs, which is impossible except with probability at most $(q+1)/|\mathcal{Y}|$. Therefore, $p \leq (q+1)/|\mathcal{Y}|$. Therefore, as long as $|\mathcal{Y}|$ is super-polynomial in size, $p$ is negligible, meaning $\epsilon_2$ is negligibly close to $\epsilon_1$.

Game 3: Now modify the game so that we draw $R_i$ uniformly at random from the set of all oracles. Notice that each $R_i$ is queried only once, meaning pairwise-independent $R_i$ look exactly like truly random $R_i$, so Game 3 looks exactly like Game 2 from the point of view of the adversary. Thus the success probability $\epsilon_3$ is equal to $\epsilon_2$.

Game 4: For this game, we answer query $i$ with the oracle that maps $m$ to $(R_i(m), F(R_i(m))$. That is, we ignore $H$ for answering MAC queries. Let $\epsilon_3$ be the success probability in this game.

To prove that $\epsilon_4$ is negligibly close to $\epsilon_3$, we need the following lemma:

**Lemma 5.3.** *Consider two distributions $D_1$ and $D_2$ on oracles from $\mathcal{M}$ into $\mathcal{X} \times \mathcal{Y}$:*

- *$D_1$: generate a random oracle $R : \mathcal{M} \to \mathcal{X}$ and a random oracle $P : \mathcal{M} \to \mathcal{Y}$, and output the oracle that maps $m$ to $(R(m), P(m))$.*
- *$D_2$: generate a random oracle $R : \mathcal{M} \to \mathcal{X}$ and a random oracle $F : \mathcal{X} \to \mathcal{Y}$, and output the oracle that maps $m$ to $(R(m), F(R(m)))$.*

*Then the probability that any $q$-quantum query algorithm distinguishes $D_1$ from $D_2$ is at most $O(q^2/|\mathcal{X}|^{1/3})$.*

**Proof**. Let $B$ be a quantum algorithm making quantum queries that distinguishes with probability $\lambda$. We will now define a quantum algorithm $C$ that is given $r$ samples $(s_i, t_i) \in \mathcal{X} \times \mathcal{Y}$, where $s_i$ are chosen randomly, and $t_i$ are either chosen randomly, or are equal to $T(s_i)$ for a randomly chosen function $T : \mathcal{X} \to \mathcal{Y}$. $C$'s goal is to distinguish these two cases. Notice that as long as the $s_i$ are distinct, these two distributions are identical. Therefore, $C$'s distinguishing probability is at most the probability of a collision, which is at most $O(r^2/|\mathcal{X}|)$.

$C$ works as follows: generate a random oracle $A : \mathcal{M} \to [r]$. Let $R(m) = s_{A(m)}$ and $P(m) = t_{A(m)}$, and give $B$ the oracle $(R(m), P(m))$. If $t_i$ are random, then we have the oracle that maps $m$ to $(s_{A(m)}, t_{A(m)})$. This is exactly the small-range distribution of Zhandry [Zha12b], and is indistinguishable from $D_1$ except with probability $O(q^3/r)$.

Similarly, if $t_i = T(s_i)$, then the oracle maps $m$ to $(s_{A(m)}, T(s_{A(m)}))$. The oracle that maps $m$ to $s_{A(m)}$ is also a small-range distribution, so it is indistinguishable from a random oracle except with probability $O(q^3/r)$. If we replace $s_{A(m)}$ with a random oracle, we get exactly the distribution $D_2$. Thus, $D_2$ is indistinguishable from $(s_{A(m)}, T(s_{A(m)}))$ except with probability $O(q^3/r)$.

Therefore, $C$'s success probability at distinguishing $D_1$ from $D_2$ is at least $\lambda - O(q^3/r)$, and is at most $O(r^2/|\mathcal{X}|)$. This means the distinguishing probability of $B$ is at most

$$O\left(\frac{r^2}{\mathcal{X}} + \frac{q^3}{r}\right)$$

This is minimized by choosing $r = O(q|\mathcal{X}|^{1/3})$, which gives a distinguishing probability of at most $O(q^2/|\mathcal{X}|^{1/3})$.

□

We show that $\epsilon_4$ is negligibly-close to $\epsilon_3$ using a sequence of sub-games. Game 3a is the game where we answer query $i$ with the oracle that maps $m$ to $(R_i(m), P_i(m) + H(m))$ where $P_i$ is another random oracle. Notice that we can define oracles $R(i, m) = R_i(m)$ and $P(i, m) = P_i(m)$. Then $R$ and $P$ are random oracles, and using the above lemma, the success probability of $B$ in Game 3a is negligibly close to that of Game 3. Notice that since $P_i$ is random, $P_i'(m) = P_i(m) + H(m)$ is also random, so Game 3a is equivalent to the game were we answer query $i$ with the oracle that maps $m$ to $(R_i(m), P_i(m))$. Using the above lemma again, the success probability of $B$ in this game is negligibly close to that of Game 4.

Now, we claim that $\epsilon_4$, the success probability in Game 4 is negligible. Indeed, the view of $B$ is independent of $H$, so the probability that $H(m_0) - H(m_1) = s$ is $1/|\mathcal{Y}|$. Since $\epsilon_4$ is negligibly close to $\epsilon = \epsilon_0$, the advantage of $A$, $A$'s advantage is also negligible. □

## 6 q-time MACs

In this section, we develop quantum one-time MACs, MACs that are secure when the adversary can issue only *one* quantum chosen message query. More generally, we will study quantum $q$-time MACs.

Classically, any pairwise independent function is a one-time MAC. In the quantum setting, Corollary 3.4 shows that when the range is much larger than the domain, this still holds. However, such MACs are not useful since we want the tag to be short. We first show that when the range is not larger than the domain, pairwise independence is not enough to ensure security:

**Theorem 6.1.** *For any set $\mathcal{Y}$ of prime-power size, and any set $\mathcal{X}$ with $|\mathcal{X}| \geq |\mathcal{Y}|$, there exist $(q + 1)$-wise independent functions from $\mathcal{X}$ to $\mathcal{Y}$ that are not $q$-time MACs.*

To prove this theorem, we treat $\mathcal{Y}$ as a finite field, and assume $\mathcal{X} = \mathcal{Y}$, as our results are easy to generalize to larger domains. We use random degree $q$ polynomials as our $(q+1)$-wise independent family, and show in Theorem 6.2 below that such polynomials can be completely recovered using only $q$ quantum queries. It follows that the derived MAC cannot be $q$-time secure since once the adversary has the polynomial it can easily forge tags on new messages.

**Theorem 6.2.** *For any prime power $n$, there is an efficient quantum algorithm that makes only $q$ quantum queries to an oracle implementing a degree-$q$ polynomial $F : \mathbb{F}_n \to \mathbb{F}_n$, and completely determines $F$ with probability $1 - O(qn^{-1})$.*

The theorem shows that a $(q+1)$-wise independence family is not necessarily a secure quantum $q$-time MAC since after $q$ quantum chosen message queries the adversary extracts the entire secret key. The case $q = 1$ is particularly interesting. The following lemma will be used to prove Theorem 6.2:

**Lemma 6.3.** *For any prime power $n$, and any subset $\mathcal{X} \subseteq \mathbb{F}_n$ of size $n - k$, there is an efficient quantum algorithm that makes a single quantum query to any degree-1 polynomial $F : \mathcal{X} \to \mathbb{F}_n$, and completely determines $F$ with probability $1 - O(kn^{-1})$.*

**Proof**. Write $F(x) = ax + b$ for values $a, b \in \mathbb{F}_n$, and write $n = p^t$ for some prime $p$ and integer $t$. We design an algorithm to recover $a$ and $b$.

Initialize the quantum registers to the state

$$|\psi_1\rangle = \frac{1}{\sqrt{n-k}} \sum_{x \in \mathcal{X}} |x, 0\rangle$$

Next, make a single oracle query to $F$, obtaining

$$|\psi_2\rangle = \frac{1}{\sqrt{n-k}} \sum_{x \in \mathcal{X}} |x, ax + b\rangle$$

Note that we can interpret elements $z \in \mathbb{F}_n$ as vectors $\mathbf{z} \in \mathbb{F}_p^t$. Let $\langle \mathbf{y}, \mathbf{z} \rangle$ be the inner product of vectors $\mathbf{y}, \mathbf{z} \in \mathbb{F}_p^t$. Multiplication by $a$ in $\mathbb{F}_n$ is a linear transformation over the vector space $\mathbb{F}_p^t$, and can therefore be represented by a matrix $\mathbf{M}_a \in \mathbb{F}_p^{t \times t}$. Thus, we can write

$$|\psi_2\rangle = \frac{1}{\sqrt{n-k}} \sum_{\mathbf{x} \in \mathcal{X}} |\mathbf{x}, \mathbf{M}_a \mathbf{x} + \mathbf{b}\rangle$$

Note that in the case $t = 1$, $a$ is a scalar in $\mathbb{F}_p$, so $\mathbf{M}_a$ is just the scalar $a$.

Now, the algorithm applies the Fourier transform to both registers, to obtain

$$|\psi_3\rangle = \frac{1}{n\sqrt{n-k}} \sum_{\mathbf{y}, \mathbf{z}} \left( \sum_{\mathbf{x} \in \mathcal{X}} \omega_p^{\langle \mathbf{x}, \mathbf{y} \rangle + \langle \mathbf{M}_a \mathbf{x} + \mathbf{b}, \mathbf{z} \rangle} \right) |\mathbf{y}, \mathbf{z}\rangle$$

where $\omega_p$ is a complex primitive $p$th root of unity.

The term in parenthesis can be written as

$$\left( \sum_{\mathbf{x} \in \mathcal{X}} \omega_p^{\langle \mathbf{x}, \mathbf{y} + \mathbf{M}_a^T \mathbf{z} \rangle} \right) \omega_p^{\langle \mathbf{b}, \mathbf{z} \rangle}$$

We will then do a change of variables, setting $\mathbf{y}' = \mathbf{y} + \mathbf{M}_a^T\mathbf{z}$.

Therefore, we can write the state as

$$|\psi_3\rangle = \frac{1}{n\sqrt{n-k}} \sum_{\mathbf{y}',\mathbf{z}} \left( \sum_{\mathbf{x}\in\mathcal{X}} \omega_p^{\langle \mathbf{x},\mathbf{y}'\rangle} \right) \omega_p^{\langle \mathbf{b},\mathbf{z}\rangle} |\mathbf{y}' - \mathbf{M}_a^T\mathbf{z}, \mathbf{z}\rangle$$

For $\mathbf{z} \neq 0$ and $\mathbf{y}' = 0$, we will now explain how to recover $a$ from $(-\mathbf{M}_a^T\mathbf{z}, \mathbf{z})$. Notice that the transformation that takes $\mathbf{a}$ and outputs $-\mathbf{M}_a^T\mathbf{z}$ is a linear transformation. Call this transformation $\mathbf{L}_z$. The coefficients of $\mathbf{L}_z$ are easily computable, given $\mathbf{z}$, by applying the transformation to each of the unit vectors. Notice that if $t = 1$, $\mathbf{L}_z$ is just the scalar $-z$. We claim that $\mathbf{L}_z$ is invertible if $\mathbf{z} \neq 0$. Suppose there is some $\mathbf{a}$ such that $\mathbf{L}_z\mathbf{a} = -\mathbf{M}_a^T\mathbf{z} = 0$. Since $\mathbf{z} \neq 0$, this means the linear operator $-\mathbf{M}_a^T$ is not invertible, so neither is $-\mathbf{M}_a$. But $-\mathbf{M}_a$ is just multiplication by $-a$ in the field $\mathbb{F}_n$. This multiplication is only non-invertible if $-a = 0$, meaning $\mathbf{a} = 0$, a contradiction. Therefore, the kernel of $\mathbf{L}_z$ is just 0, so the map is invertible.

Therefore, to compute $a$, compute the inverse operator $\mathbf{L}_z^{-1}$ and apply it to $-\mathbf{M}_a^T\mathbf{z}$, interpreting the result as a field element in $\mathbb{F}_n$. The result is $a$. More specifically, for $\mathbf{z} \neq 0$, apply the computation mapping $(\mathbf{y}, \mathbf{z})$ to $(\mathbf{L}_z^{-1}\mathbf{y}, \mathbf{z})$, which will take $(-\mathbf{M}_a^T\mathbf{z}, \mathbf{z})$ to $(a, \mathbf{z})$. For $\mathbf{z} = 0$, we will just apply the identity map, leaving both registers as is. This map is now reversible, meaning this computation can be implemented as a quantum computation. The result is the state

$$|\psi_4\rangle = \frac{1}{n\sqrt{n-k}} \sum_{\mathbf{y}'} \left( \sum_{\mathbf{x}\in\mathcal{X}} \omega_p^{\langle \mathbf{x},\mathbf{y}'\rangle} \right) \left( \sum_{\mathbf{z}\neq 0} \omega_p^{\langle \mathbf{b},\mathbf{z}\rangle} |\mathbf{L}_z^{-1}\mathbf{y}' + a, \mathbf{z}\rangle + |\mathbf{y}', 0\rangle \right)$$

We will now get rid of the $|\mathbf{y}', 0\rangle$ terms by measuring whether $\mathbf{z} = 0$. The probability that $\mathbf{z} = 0$ is $1/n$, and in this case, we abort. Otherwise, we are left if the state

$$|\psi_5\rangle = \frac{1}{\sqrt{n(n-1)(n-k)}} \sum_{\mathbf{z}\neq 0,\mathbf{y}'} \left( \sum_{\mathbf{x}\in\mathcal{X}} \omega_p^{\langle \mathbf{x},\mathbf{y}'\rangle} \right) \omega_p^{\langle \mathbf{b},\mathbf{z}\rangle} |\mathbf{L}_z^{-1}\mathbf{y}' + a, \mathbf{z}\rangle$$

The algorithm then measures the first register. Recall that $\mathcal{X}$ has size $n - k$. The probability the outcome of the measurement is $a$ is then $(1 - k/n)$. In this case, we are left in the state

$$|\psi_6\rangle = \frac{1}{\sqrt{n-1}} \sum_{\mathbf{z}\neq 0} \omega_p^{\langle \mathbf{b},\mathbf{z}\rangle} |\mathbf{z}\rangle$$

Next, the algorithm performs the inverse Fourier transform to the second register, arriving at the state

$$|\psi_7\rangle = \frac{1}{\sqrt{n(n-1)}} \sum_{\mathbf{w}} \left( \sum_{\mathbf{z}\neq 0} \omega_p^{\langle \mathbf{b}-\mathbf{w},\mathbf{z}\rangle} \right) |\mathbf{w}\rangle$$

Now the algorithm measures again, and interpret the resulting vector as a field element. The probability that the result is $b$ is $1-1/n$. Therefore, with probability $(1-k/n)(1-1/n)^2 = 1-O(k/n)$, the algorithm outputs both $a$ and $b$.

□

Now we use this attack to obtain an attack on degree-$d$ polynomials, for general $d$:

**Proof of Theorem 6.2.** We show how to recover the $q+1$ different coefficients of any degree-$q$ polynomial, using only $q-1$ classical queries and a single quantum query.

Let $a$ be the coefficient of $x^q$, and $b$ the coefficient of $x^{q-1}$ in $F(x)$. First, make $q-1$ *classical* queries to arbitrary distinct points $\{x_1, ..., x_{q-1}\}$. Let $Z(x)$ be the unique polynomial of degree $q-2$ such that $r(x_i) = F(x_i)$, using standard interpolation techniques. Let $G(x) = F(x) - Z(x)$. $G(x)$ is a polynomial of degree $q$ that is zero on the $x_i$, so it factors, allowing us to write

$$F(x) = Z(x) + (a'x + b') \prod_{i=1}^{q-1} (x - x_i)$$

By expanding the product, we see that $a = a'$ and $b = b' - a \sum x_i$. Therefore, we can implement an oracle mapping $x$ to $a(x + \sum x_i) + b$ as follows:

- Query $F$ on $x$, obtaining $F(x)$.
- Compute $Z(x)$, and let $G(x) = F(x) - Z(x)$.
- Output $G(x)/\prod(x - x_i) = a(x + \sum x_i) + b$.

This oracle works on all inputs except the $q-1$ different $x_i$ values. We run the algorithm from Lemma 6.3 on $\mathcal{X} = \mathbb{F}_n \setminus \{x_i\}$, we will recover with probability $1 - O(q/n)$ both $a$ and $b + a\sum x_i$ using a single quantum query, from which we can compute $a$ and $b$. Along with the $F(x_i)$ values, we can then reconstruct the entire polynomial. $\square$

## 6.1 Sufficient Conditions for a One-Time Mac

We show that, while pairwise independence is not enough for a one-time MAC, 4-wise independence is. We first generalize a theorem of Zhandry [Zha12a]:

**Lemma 6.4.** *Let $A$ be any quantum algorithm that makes $c$ classical queries and $q$ quantum queries to an oracle $H$. If $H$ is drawn from a $(c+2q)$-wise independent function, then the output distribution of $A$ is identical to the case where $H$ is truly random.*

**Proof**. If $q = 0$, then this theorem is trivial, since the $c$ outputs $A$ sees are distributed randomly. If $c = 0$, then the theorem reduces to that of Zhandry [Zha12a]. By adapting the proof of the $c = 0$ case to the general case, we get the lemma. Our approach is similar to the polynomial method, but needs to be adapted to handle classical queries correctly.

Our quantum algorithm makes $k = c + q$ queries. Let $Q \subseteq [k]$ be the set of queries that are quantum, and let $C \subseteq [k]$ be the set of queries that are classical.

Fix an oracle $H$. Let $\delta_{x,y}$ be 1 if $H(x) = y$ and 0 otherwise. Let $\boldsymbol{\rho}^{(i)}$ be the density matrix after the $i$th query, and $\boldsymbol{\rho}^{(i-1/2)}$ be the density matrix before the $i$th query. $\rho^{(q+1/2)}$ is the final state of the algorithm.

We now claim that $\boldsymbol{\rho}^{(i)}$ and $\boldsymbol{\rho}^{(i+1/2)}$ are polynomials of the $\delta_{x,y}$ of degree $k_i$, where $k_i$ is twice the number of quantum queries made so far, plus the number of classical queries made so far.

$\boldsymbol{\rho}^{(0)}$ and $\boldsymbol{\rho}^{(0+1/2)}$ are independent of $H$, so they are not a function of the $\delta_{x,y}$ at all, meaning the degree is $0 = k_0$.

We now inductively assume our claim is true for $i-1$, and express $\boldsymbol{\rho}^{(i)}$ in terms of $\boldsymbol{\rho}^{(i-1/2)}$. There are two cases:

- $i$ is a quantum query. In this case, $k_i = k_{i-1} + 2$. We can write

$$\boldsymbol{\rho}^{(i)}_{x,y,z,x',y',z'} = \boldsymbol{\rho}^{(i-1/2)}_{x,y-H(x),z,x',y'-H(x'),z}$$

An alternative way to write this is as

$$\boldsymbol{\rho}^{(i)}_{x,y,z,x',y',z'} = \sum_{r,r'} \delta_{x,y-r}\delta_{x',y'-r'}\boldsymbol{\rho}^{(i-1/2)}_{x,r,z,x',r',z}$$

By induction, each of the $\boldsymbol{\rho}^{(i-1/2)}_{x,r,z,x',r',z}$ are polynomials of degree $k_{i-1}$ in the $d_{x,y}$ values, so $\boldsymbol{\rho}^{(i)}_{x,y,z,x',y',z'}$ is a polynomial of degree $k_{i-1} + 2 = k_i$.

- $i$ is a classical query. This means $l_i = k_{i-1} + 1$. Let $\boldsymbol{\rho}^{(i-1/4)}$ representing the state after measuring the $x$ register, but before making the actual query. This is identical to $\boldsymbol{\rho}^{(i-1/2)}$, except the entries where $x \neq x'$ are zeroed out. We can then write

$$\boldsymbol{\rho}^{(i)}_{x,y,z,x',y',z'} = \sum_{r,r'} \delta_{x,y-r}\delta_{x',y'-r'}\boldsymbol{\rho}^{(i-1/4)}_{x,r,z,x',r',z} = \sum_{r,r'} \delta_{x,y-r}\delta_{x,y'-r'}\boldsymbol{\rho}^{(i-1/2)}_{x,r,z,x,r',z}$$

Now, notice that $\delta_{x,y-r}\delta_{x,y'-r'}$ is zero unless $y - r = y' - r'$, in which case it just reduces to $\delta_{x,y-r}$. Therefore, we can simply further:

$$\boldsymbol{\rho}^{(i)}_{x,y,z,x',y',z'} = \sum_{r} \delta_{x,y-r}\boldsymbol{\rho}^{(i-1/2)}_{x,r,z,x,(y-y')+r,z}$$

By induction, each of the $\boldsymbol{\rho}^{(i-1/2)}_{x,r,z,x,(y-y')+r,z}$ values are polynomials of degree $k_{i-1}$ in the $d_{x,y}$ values, so $\boldsymbol{\rho}^{(i)}_{x,y,z,x',y',z'}$ is a polynomial of degree $k_{i-1} + 1 = k_i$

Therefore, after all $q$ queries, final matrix $\boldsymbol{\rho}^{(q+1/2)}$ is a polynomial in the $\delta_{x,y}$ of degree at most $k = 2q + c$. We can then write the density matrix as

$$\boldsymbol{\rho}^{(q+1/2)} = \sum_{\mathbf{x},\mathbf{y}} \mathbf{M}_{\mathbf{x},\mathbf{y}} \prod_{t=0}^{r_q} \delta_{x_i,y_i}$$

where $\mathbf{x}$ and $\mathbf{y}$ are vectors of length $k$, $\mathbf{M}_{\mathbf{x},\mathbf{y}}$ are matrices, and the sum is over all possible vectors.

Now, fix a distribution $D$ on oracles $H$. The density matrix for the final state of the algorithm, when the oracle is drawn from $H$, is given by

$$\sum_{\mathbf{x},\mathbf{y}} \mathbf{M}_{\mathbf{x},\mathbf{y}} \left( \sum_{H} \Pr[H \leftarrow D] \prod_{t=0}^{r_q} \delta_{x_i,y_i} \right)$$

The term in parenthesis evaluates to $\Pr_{H \leftarrow D}[H(\mathbf{x}) = \mathbf{y}]$. Therefore, the final density matrix can be expressed as

$$\sum_{\mathbf{x},\mathbf{y}} \mathbf{M}_{\mathbf{x},\mathbf{y}} \Pr_{H \leftarrow D}[H(\mathbf{x}) = \mathbf{y}]$$

Since $\mathbf{x}$ and $\mathbf{y}$ are vectors of length $k = 2q + c$, if $D$ is $k$-wise independent, $\Pr_{H \leftarrow D}[H(\mathbf{x}) = \mathbf{y}]$ evaluates to the same quantity as if $D$ was truly random. Thus the density matrices are the same. Since all of the statistical information about the final state of the algorithm is contained in the density matrix, the distributions of outputs are thus identical, completing the proof.

□

Using this lemma we show that $(3q+1)$-wise independence is sufficient for $q$-time MACs.

**Theorem 6.5.** *Any* $(3q+1)$*-wise independent family with domain* $\mathcal{X}$ *and range* $\mathcal{Y}$ *is a quantum* $q$*-time secure MAC provided* $(q+1)/|\mathcal{Y}|$ *is negligible.*

**Proof**. Let $D$ be some $(3q+1)$-wise independent function. Suppose we have an adversary $A$ that makes $q$ quantum queries to an oracle $H$, and attempts to produces $q+1$ input/output pairs. Let $\epsilon_R$ be the probability of success when $H$ is a random oracle, and let $\epsilon_D$ be the probability of success when $H$ is drawn from $D$. We construct an algorithm $B$ with access to $H$ as follows: simulate $A$ with oracle access to $H$. When $A$ outputs $q+1$ input/output pairs, simply make $q+1$ queries to $H$ to check that these are valid pairs. Output 1 if and only if all pairs are valid. Therefore, $B$ makes $q$ quantum queries and $c = q+1$ classical queries to $H$, and outputs 1 if and only if $A$ succeeds: if $H$ is random, $B$ outputs 1 with probability $\epsilon_R$, and if $H$ is drawn from $D$, $B$ outputs 1 with probability $\epsilon_D$. Now, since $D$ is $(3q+1)$-wise independent and $3q+1 = 2q+c$, Lemma 6.4 shows that the distributions of outputs when $H$ is drawn from $D$ is identical to that when $H$ is random, meaning $\epsilon_D = \epsilon_R$.

Thus, when $H$ is drawn from $D$, $A$'s succeeds with the same probability that it would if $H$ was random. But we already know that if $H$ is truly random, $A$'s success probability is less than $(q+1)/|\mathcal{Y}|$. Therefore, when $H$ is drawn from $D$, $A$ succeeds with probability less than $(q+1)/|\mathcal{Y}|$, which is negligible. Hence, if $H$ is drawn from $D$, $H$ is a $q$-time MAC. □ □

## 7 Conclusion

We introduced the rank method as a general technique for obtaining lower bounds on quantum oracle algorithms and used this method to bound the probability that a quantum algorithm can evaluate a random oracle $\mathcal{O} : \mathcal{X} \to \mathcal{Y}$ at $k$ points using $q < k$ queries. When the range of $\mathcal{Y}$ is small, say $|\mathcal{Y}| = 8$, a quantum algorithm can recover $k$ points of $\mathcal{O}$ from only $0.9k$ queries with high probability. However, we show that when the range $\mathcal{Y}$ is large, no algorithm can produce $k$ input-output pairs of $\mathcal{O}$ using only $k-1$ queries, with non-negligible probability. We use these bounds to construct the first MACs secure against quantum chosen message attacks. We consider both PRF and Carter-Wegman constructions. For one-time MACs we showed that pair-wise independence does not ensure security, but four-way independence does.

These results suggest many directions for future work. First, can these bounds be generalized to signatures to obtain signatures secure against quantum chosen message attacks? Similarly, can we construct encryption systems secure against quantum chosen ciphertext attacks where decryption queries are superpositions of ciphertexts?

## Acknowledgments

We thank Luca Trevisan and Amit Sahai for helpful conversations about this work. This work was supported by NSF, DARPA, IARPA, the Air Force Office of Scientific Research (AFO SR) under a MURI award, Samsung, and a Google Faculty Research Award. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of DARPA, IARPA, DoI/NBC, or the U.S. Government.

## References

[Aar02] Scott Aaronson. Quantum lower bound for the collision problem. In *STOC*, pages 635–642, 2002.

[Amb00] Andris Ambainis. Quantum lower bounds by quantum arguments. In *STOC*, pages 636–643, 2000.

[Amb06] Andris Ambainis. Polynomial degree vs. quantum query complexity. *J. Comput. Syst. Sci.*, 72(2):220–238, 2006.

[ASdW09] Andris Ambainis, Robert Spalek, and Ronald de Wolf. A new quantum lower bound method, with applications to direct product theorems and time-space tradeoffs. *Algorithmica*, 55(3):422–461, 2009.

[BBC$^+$01] Robert Beals, Harry Buhrman, Richard Cleve, Michele Mosca, and Ronald de Wolf. Quantum Lower Bounds by Polynomials. *Journal of the ACM (JACM)*, 48(4):778–797, July 2001.

[BCK96] Mihir Bellare, Ran Canetti, and Hugo Krawczyk. Pseudorandom functions revisited: The cascade construction and its concrete security. In *FOCS*, pages 514–523, 1996.

[BDF$^+$11] Dan Boneh, Özgür Dagdelen, Marc Fischlin, Anja Lehmann, Christian Schaffner, and Mark Zhandry. Random Oracles in a Quantum World. In *Advances in Cryptology — ASIACRYPT 2011*, 2011.

[BHK$^+$11] Gilles Brassard, Peter Høyer, Kassem Kalach, Marc Kaplan, Sophie Laplante, and Louis Salvail. Merkle Puzzles in a Quantum World. *Advances in Cryptology - CRYPTO 2011*, pages 391–410, 2011.

[BKR00] Mihir Bellare, Joe Kilian, and Phillip Rogaway. The security of the cipher block chaining message authentication code. *J. Comput. Syst. Sci.*, 61(3), 2000.

[BS08] Gilles Brassard and Louis Salvail. Quantum Merkle Puzzles. *Second International Conference on Quantum, Nano and Micro Technologies (ICQNM 2008)*, pages 76–79, February 2008.

[Can01] Ran Canetti. Universally composable security: A new paradigm for cryptographic protocols. In *Proc. of FOCS*. IEEE, 2001.

[DFNS11] Ivan Damgård, Jakob Funder, Jesper Buus Nielsen, and Louis Salvail. Superposition attacks on cryptographic protocols. *CoRR*, abs/1108.6313, 2011.

[GGM86] Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to Construct Random Functions. *Journal of the ACM (JACM)*, 33(4):792–807, 1986.

[HSS11] Sean Hallgren, Adam Smith, and Fang Song. Classical cryptographic protocols in a quantum world. In *Proc. of Crypto*, LNCS. Springer, 2011.

[KdW03] Iordanis Kerenidis and Ronald de Wolf. Exponential lower bound for 2-query locally decodable codes via a quantum argument. In *Proceedings of the 35th Annual ACM Symposium on Theory of Computing (STOC)*, pages 106–115, 2003.

[KK11] Daniel M. Kane and Samuel A. Kutin. Quantum interpolation of polynomials. *Quantum Information & Computation*, 11(1&2):95–103, 2011. First published in 2009.

[Sha79] Adi Shamir. How to share a secret. *Commun. ACM*, 22(11):612–613, 1979.

[Unr10] Dominique Unruh. Universally Composable Quantum Multi-Party Computation. *Advances in Cryptology — EUROCRYPT 2010*, pages 486–505, 2010.

[vD98] Wim van Dam. Quantum oracle interrogation: Getting all information for almost half the price. In *FOCS*, pages 362–367, 1998.

[WC81] Mark N. Wegman and Larry Carter. New hash functions and their use in authentication and set equality. *J. Comput. Syst. Sci.*, 22(3):265–279, 1981.

[Zha12a] Mark Zhandry. Secure Identity-Based Encryption in the Quantum Random Oracle Model. In *Advances in Cryptology — CRYPTO*, 2012. Full version available at the Cryptology ePrint Archives: `http://eprint.iacr.org/2012/076/`.

[Zha12b] Mark Zhandry. How to Construct Quantum Random Functions. In *Proceedings of FOCS*, 2012. Full version available at the Cryptology ePrint Archives: `http://eprint.iacr.org/2012/182/`.