Finding Collisions in a Quantum World: Quantum Black-Box Separation of Collision-Resistance and One-Wayness

Akinori Hosoyamada[1,2] and Takashi Yamakawa[1]

[1] NTT Secure Platform Laboratories, Tokyo, Japan.
{akinori.hosoyamada.bh, takashi.yamakawa.ga}@hco.ntt.co.jp
[2] Nagoya University, Nagoya, Japan. hosoyamada.akinori@nagoya-u.jp

Abstract. Since the celebrated work of Impagliazzo and Rudich (STOC 1989), a number of black-box impossibility results have been established. However, these works only ruled out classical black-box reductions among cryptographic primitives. Therefore it may be possible to overcome these impossibility results by using quantum reductions. To exclude such a possibility, we have to extend these impossibility results to the quantum setting.
In this paper, we study black-box impossibility in the quantum setting. We first formalize a quantum counterpart of fully-black-box reduction following the formalization by Reingold, Trevisan and Vadhan (TCC 2004). Then we prove that there is no quantum fully-black-box reduction from collision-resistant hash functions to one-way permutations (or even trapdoor permutations). We take both of classical and quantum implementations of primitives into account. This is an extension to the quantum setting of the work of Simon (Eurocrypt 1998) who showed a similar result in the classical setting.

keywords post-quantum cryptography, one-way permutation, one-way trapdoor permutation, collision resistant hash function, fully black-box reduction, quantum reduction, impossibility

1 Introduction

1.1 Background

Black-box impossibility. Reductions among cryptographic primitives are fundamental in cryptography. For example, we know reductions from pseudorandom generators, pseudorandom functions, symmetric key encryptions, and digital signatures to one-way functions (OWF). On the other hand, there are some important cryptographic primitives including collision-resistant hash functions (CRH), key-exchanges, public key encryption (PKE), oblivious transfer, and non-interactive zero-knowledge proofs, for which there are no known reductions to OWF. Given this situation, we want to ask if it is impossible to reduce these primitives to OWF. We remark that under the widely believed assumption that

these primitives exist, OWF "imply" these primitives (i.e., these primitives are "reduced" to OWF) in a trivial sense. Therefore to make the question meaningful, we have to somehow restrict types of reductions.

For this purpose, Impagliazzo and Rudich [IR89] introduced the notion of *black-box reductions*. Roughly speaking, a black-box reduction is a reduction that uses an underlying primitive and an adversary in a black-box manner (i.e., use them just as oracles).[3] They proved that there does not exist a black-box reduction from key-exchange protocols (and especially PKE) to one-way permutations (OWP). They also observed that most existing reductions between cryptographic primitives are black-box. Thus their result can be interpreted as an evidence that we cannot construct key-exchange protocols based on OWP with commonly used techniques. After their seminal work, there have been numerous impossibility results of black-box reductions (See Section 1.4 for details).

Post-quantum and quantum cryptography. In 1994. Shor [Sho94] showed that we can efficiently compute integer factorization and discrete logarithm, whose hardness are the basis of widely used cryptographic systems, by using a quantum computer. After that, post-quantum cryptography, which treats classically computable cryptographic schemes that resist quantum attacks, has been intensively studied (e.g., [McE78,Ajt96,Reg05,JF11]). Indeed, NIST has recently started a standardization of post-quantum cryptography [NIS16]. We refer more detailed survey of post-quantum cryptography to [BL17].

As another direction to use quantum computer in cryptography, there have been study of quantum cryptography, in which even honest algorithms also use quantum computers. They include quantum key distribution [BB84], quantum encryption [ABF$^+$16,AGM18], quantum (fully) homomorphic encryption [BJ15,Mah18,Bra18], quantum copy-protection [Aar09], quantum digital signatures [GC01], quantum money [Wie83,AC12,Zha19], etc. We refer more detailed survey of quantum cryptography to [BS16].

Our motivation: black-box impossibility in a quantum world. In this paper, we consider black-box impossibility in a quantum setting where primitives and adversaries are quantum, and a reduction has quantum access to them.

Quantum reductions are sometimes more powerful than classical reductions. For example, Regev [Reg05] gave a quantum reduction from the learning with errors (LWE) problem to the decision version of the shortest vector problem (GapSVP) or the shortest independent vectors problem (SIVP). We note that there are some follow-up works that give classical reduction between these problems in some parameter settings [Pei09,BLP$^+$13], but we still do not know any classical reduction that works in the same parameter setting as the quantum

[3] This is an explanation for *fully-black-box reduction* using the terminology of Reingold, Trevisan, and Vadhan [RTV04]. Since we only consider fully-black-box reductions in this paper, in this introduction, we just say black-box reduction to mean fully-black-box reduction.

one by Regev. This example illustrates that quantum reductions are sometimes more powerful than classical reductions even if all problem instances (e.g., implementations of primitives, adversaries, and reduction algorithms) are classical. Therefore it may be possible to overcome black-box impossibility results shown in the classical setting by using quantum reductions.

Since quantum computers may also be used to implement cryptographic primitives in the near future, it is of much interest to study how the classical impossibility results change in the quantum setting. In particular, it is theoretically very important to study whether the impossibility of black-box reductions from CRH to OWP shown by Simon [Sim98], which is one of the most fundamental results on impossibility and revisited in many follow-up works [HR04,HHRS07,AS15], can be overcome in the quantum setting. Despite the importance of the problem, the (im)possibility of the quantum reductions has not been studied.

1.2 Our Results

This paper shows that the impossibility of black-box reductions from CRH to OWP cannot be overcome in the quantum setting. First, we formally define the notion of quantum black-box reduction based on the work by Reingold, Trevisan and Vadhan [RTV04], which gave a formal framework for the notion of black-box reductions in the classical setting. Then we prove the following theorem.

Theorem 1 (informal). *There does not exist a quantum black-box reduction from CRH to OWP.*

We note that though we do not know any candidate of OWP that resists quantum attacks, the above theorem is still meaningful since it also rules out quantum black-box reductions from CRH to OWF (since OWP is also OWF) and there exist many candidates of post-quantum OWF. This theorem is stated with OWP instead of OWF just because this makes the theorem stronger.

We also extend the result to obtain the following theorem.

Theorem 2 (informal). *There does not exist a quantum black-box reduction from CRH to trapdoor permutations (TDP).*

Note that our results do not require any unproven assumptions nor the existence of any oracles. Some oracles are introduced in our proofs, but they are just technical tools.

Remark 1. In this paper, by quantum black-box reduction we denote reductions that have quantum black-box oracle accesses to primitives. We always consider security of primitives against quantum adversaries, and do not discuss primitives that are only secure against classical adversaries. In addition, since our main goal is to show the impossibility of reductions from CRH to OWP and CRH to TDP, and when we consider primitives with interactions in the quantum setting we have some subtle issues that do not matter in the classical setting (e.g., rewinding is sometimes hard in the quantum setting [ARU14]), we treat only primitives such that both of the primitives themselves and security games are non-interactive.

1.3 Technical Overview

Here, we give a brief technical overview of our results. We focus on the proof of Theorem 1 since Theorem 2 can be proven by a natural (yet non-trivial) extension of that of Theorem 1. We remark that we omit many details and often rely on non-rigorous arguments for intuitive explanations in this subsection.

First, we recall the *two-oracle technique*, which is a technique to rule out black-box reductions among cryptographic primitives in the classical setting introduced by Hsiao and Reyzin [HR04]. Roughly speaking, they showed that a black-box reduction from a primitive $\mathcal{P}$ to another primitive $\mathcal{Q}$ does not exist if there exist oracles Φ and Ψ^{Φ} such that $\mathcal{Q}$ exists and $\mathcal{P}$ does not exist relative to these oracles. As our first contribution, we show that a similar argument carries over to the quantum setting if we appropriately define primitives and black-box reductions in the quantum setting.

For proving the separation between CRH and OWP, we consider oracles $\Phi = f$, which is a random permutation over $\{0,1\}^n$, and $\Psi^{\Phi} = \mathsf{ColFinder}^f$, which is an oracle that finds a collision of any function described by an oracle-aided quantum circuit C that accesses f as an oracle by brute-force similarly to the previous works in the classical setting [Sim98,HHRS07,AS15]. CRH does not exist relative to f and $\mathsf{ColFinder}^f$ since we can compute a collision for any (efficiently computable length-decreasing) function C^f by querying C to $\mathsf{ColFinder}^f$. Thus, what is left is to prove that a random permutation f is hard to invert even if an adversary is given an additional oracle access to $\mathsf{ColFinder}^f$.

We first recall how this was done in the classical setting based on the proof in [AS15]. [4] The underlying idea behind the proof is a very simple information theoretic fact often referred to as the "compression argument," which dates back to the work of Gennaro and Trevisan [GT00]: if we can encode a truth table of a random permutation into an encoding that can be decoded to the original truth table with high probability, then the size of the encoding should be almost as large as that of the truth table. Based on this, the strategy of the proof is to encode a truth table of f into an encoding that consists of a "partial truth table" of f that specifies values of $f(x)$ for all $x \in \{0,1\}^n \setminus G$ for an appropriately chosen subset G so that one can decode the encoding to the original truth table by recovering "forgotten values" of $f(x)$ on $x \in G$ by using the power of an adversary $\mathcal{A}$ that inverts the permutation f with oracle accesses to f and $\mathsf{ColFinder}^f$. What is non-trivial in the proof is that the decoding procedure has to simulate oracles f and $\mathsf{ColFinder}^f$ for $\mathcal{A}$ whereas the encoding only contains a partial truth table of f. To overcome this issue, they demonstrated a very clever way of choosing the subset G such that the simulation of oracles f and $\mathsf{ColFinder}^f$ does not require values of f on G. Especially, they showed that the larger $\mathcal{A}$'s success probability is, the larger the subset G is, i.e., the smaller the encoding size is. By using the lower bound of the encoding size obtained by the

[4] Though the basic idea is similar to the proof of Simon [Sim98], we explain the description in [AS15] since this is more suitable for explaining how we extend the proof to the quantum setting.

compression argument, they upper bound $\mathcal{A}$'s success probability by a negligible function in n.

Unfortunately, their proof cannot be directly extended to the quantum setting since the choice of the subset G crucially relies on the fact that queries by $\mathcal{A}$ are classical. Indeed, $\mathcal{A}$ may query a uniform superposition of all inputs to the oracle f, in which case it is impossible to perfectly simulate the oracle f with a partial truth table. Thus, instead of directly generalizing their proof to the quantum setting, we start from another work by Nayebi et al. [NABT15], which showed that it is hard to invert a random permutation f with a quantum oracle access to f.[5] The proof strategy of their work is similar to the above, and they also rely on the compression argument, but a crucial difference is that they choose the subset G in a randomized way.[6] Specifically, they first choose a random subset $R \subset \{0,1\}^n$ of a certain size, and define G as the set of x such that (1): $x \in R$, (2): $\mathcal{A}$ succeeds in inverting $f(x)$ with high probability, and (3): query magnitudes of $\mathcal{A}$ on any element in $R \setminus \{x\}$ is sufficiently small. The condition (3) implies that $\mathcal{A}$ is still likely to succeed in inverting $f(x)$ even if the function (oracle) f is replaced with any function f' that agrees with f on $\{0,1\}^n \setminus (R \setminus \{x\})$.[7] Especially, a decoder can use the function h_y that agrees with f on $\{0,1\}^n \setminus G$ and returns y on G instead of the original oracle f when it runs $\mathcal{A}$ on an input $y \in f(G)$. Since the function h_y can be implemented by the partial truth table of f on $\{0,1\}^n \setminus G$, the decoder can simulate the oracle for $\mathcal{A}$ to correctly invert y in f for each $y \in f(G)$, which implies that the decoder can recover the original truth table of f from the partial truth table. Finally, they showed that an appropriate choice of parameters gives a lower bound of the size of G, which in turn gives an upper bound of $\mathcal{A}$'s success probability based on the compression argument.

For our purpose, we have to prove that a random permutation is hard to invert for a quantum adversary $\mathcal{A}$ even if it is given a quantum access to the additional oracle $\mathsf{ColFinder}^f$. Here, we make a simplifying assumption that the oracle $\mathsf{ColFinder}^f$ is only classically accessible since this case conveys our essential idea and can be readily generalized to the quantumly accessible case. For generalizing the proof of [NABT15] to our case, we have to find a way to simulate $\mathsf{ColFinder}^f$ by using the partial truth table of f on $\{0,1\}^n \setminus G$.

Before describing our strategy about how to simulate $\mathsf{ColFinder}^f$, here we give its more detailed definition: At the beginning of each game before $\mathcal{A}$ runs relative to $\mathsf{ColFinder}^f$, two permutations $\pi_C^{(1)}, \pi_C^{(2)} \in \mathsf{Perm}(\{0,1\}^m)$ are chosen uniformly at random for each circuit C ($\{0,1\}^m$ is the domain of the function C^f). On each input C, $\mathsf{ColFinder}^f$ runs the following procedures:

1. Set $w^{(1)} \leftarrow \pi_C^{(1)}(0^m)$.

[5] Actually, they showed that a random permutation is hard to invert even given a classical advice string.

[6] Such a randomized encoder was also used in some works in the classical setting, e.g., [DTT10].

[7] Formally, this is proven by using the swapping lemma shown by Vazirani [Vaz98, Lem. 3.1]

2. Compute $u = C^f(w^{(1)})$ by running the circuit C relative to f on $w^{(1)}$.
3. Find the minimum t such that $C^f(\pi_C^{(2)}(t)) = u$ by running the circuit C relative to f on the input $\pi_C^{(2)}(i)$ and checking whether $C^f(\pi_C^{(2)}(i)) = u$ holds for $i = 0, 1, 2 \ldots$, sequentially. Set $w^{(2)} \leftarrow \pi_C^{(2)}(t)$.
4. Return $(w^{(1)}, w^{(2)}, u)$.

Next, we explain our strategy to simulate $\mathsf{ColFinder}^f$. Given a query (circuit) C and an (appropriately produced) partial truth table of f, the simulator works similarly to $\mathsf{ColFinder}$ except that it uses the partial truth table instead of f to simulate outputs of C. For making sure that this results in a correct simulation of $\mathsf{ColFinder}^f$, we require the following two properties:

P1. Given $w^{(1)}$ and $w^{(2)} = \pi_C^{(2)}(t)$, the simulator computes the value $C^f(w^{(1)}) = C^f(w^{(2)}) = u$ correctly.
P2. For $i < t$, the simulator does not misjudges that "the value $C^f(\pi_C^{(2)}(i))$ is equal to u".

The first property P1 is obviously necessary to simulate $\mathsf{ColFinder}^f$. The second property P2 is also indispensable since, if it is not satisfied, there is a possibility that the simulator responds with a wrong answer $(w^{(1)}, \pi_C(i), u)$. We have to make sure that the properties P1 and P2 will hold as well when we design our encoder (or, equivalently, how to choose $G \subset \{0,1\}^n$).

Let us explain how to encode the truth table of each permutation f into its partial table. We choose another random subset $R' \subset \{0,1\}^n$ of a certain size and require two additional conditions for x to be in G: (4): $x \in R'$ and (5): All oracle-aided quantum circuits C queried by $\mathcal{A}$ when it runs on input $f(x)$ are "good" w.r.t. (R', x) in the following sense.[8] We say that C is good w.r.t. (R', x) if query magnitudes of C on any element of $R' \setminus \{x\}$ is "small" when C runs on input $w^{(1)}$ or $w^{(2)}$ relative to f, where $(w^{(1)}, w^{(2)})$ is the collision found by $\mathsf{ColFinder}^f$. Finally, we encode f into the partial truth table that specifies the value of $f(x)$ if and only if $x \in \{0,1\}^n \setminus G$.

Intuitively, the condition (5) implies that a collision $(w^{(1)}, w^{(2)})$ found by $\mathsf{ColFinder}^f$ for any $\mathcal{A}$'s query C is not likely to change even if its oracle f is replaced with any function f' that just agrees with f on $\{0,1\}^n \setminus (R' \setminus \{x\})$, which implies that the property P1 is satisfied. In our proof, suitable permutations $\pi_C^{(1)}$ and $\pi_C^{(2)}$ are fixed and the decoder have the truth table of them. In particular, the decoder knows the correct $w^{(1)} = \pi_C^{(1)}(0^m)$ for each C, and can compute the correct $u = C^f(w^{(1)})$ since the outputs of $C^{f'}(w^{(1)})$ is likely to be the same value as $C^f(w^{(1)})$ if f' agrees with f on $\{0,1\}^n \setminus (R' \setminus \{x\})$ due to the definition of goodness of C.

Thus, in this case, the oracle $\mathsf{ColFinder}^f$ seems to be simulatable with the partial truth table of f on $\{0,1\}^n \setminus G$. However, there is an issue: It is not trivial how to ensure that the property P2 holds. Note that the property P2 holds and

[8] The definition of "good" given here corresponds to the negation of "bad" defined in the main body.

the issue is resolved if we can ensure that the simulator judges "I cannot compute the correct value $C^f(\pi_C^{(2)}(i))$" (instead of misjudging "the value $C^f(\pi_C^{(2)}(i))$ is u" for some $i < t$) when the given partial table of f does not contain enough information to compute the value $C^f(\pi_C^{(2)}(i))$. We can easily ensure it in the classical setting by measuring the queries made by C and judging that "the information is not enough" if the value $f(x)$ is not defined in the partial table for a query x made by C. However, it is highly non-trivial how to ensure it in the quantum setting since measuring queries may disturb C's computations significantly, and $\mathsf{ColFinder}^f$ runs C on $\pi_C^{(2)}(i)$ for (possibly exponentially) many i until it finds the minimum t such that $C^f(\pi_C^{(2)}(t)) = u$, in which case its total query magnitude on $R' \setminus \{x\}$ is not always small. [9]

We overcome the issue by introducing a new technique. Specifically, whenever the simulation algorithm picks i, it checks whether the partial truth table contains enough information to compute the correct value of $C^f(\pi_C^{(2)}(i))$ by running C on the input $\pi_C^{(2)}(i)$ relative to f' for *all possible permutations* f' that are consistent with the given partial truth table of f on $\{0,1\}^n \setminus (R' \setminus \{x\})$, and judges that "the partial truth table contains enough information to compute the correct value of $C^f(\pi_C^{(2)}(i))$" only if the outputs of $C^{f'}(\pi_C^{(2)}(i))$ are the same value for all possible oracles f'. (Otherwise, it judges that "The partial truth table does not contain enough information to compute the correct value of $C^f(\pi_C^{(2)}(i))$" and do the same again for the next index $(i+1)$.) This procedure prevents the simulation algorithm from outputting a "wrong" collision $(w^{(1)}, \pi_C^{(2)}(i))$ that is different from $(w^{(1)}, w^{(2)})$ and the property P2 is satisfied since the actual function f is one of the candidates of f' with which the validity of the collision is checked. On the other hand, the correct collision $(w^{(1)}, w^{(2)})$ cannot be judged to be a wrong one since the outputs of $C^{f'}(w^{(2)})$ are likely to be the same value for all f' due to the definition of goodness of C.

In this way, we can simulate both oracles f and $\mathsf{ColFinder}^f$ by using the partial truth table of f on $\{0,1\}^n \setminus G$. Similarly to the proof in [NABT15], an appropriate choice of parameters enables us to upper bound $\mathcal{A}$'s success probability by a negligible function in n. This implies that OWP exists relative to oracles f and $\mathsf{ColFinder}^f$, and thus there does not exist a black-box reduction from CRH to OWP.

We believe that our new technique can be used in more and more applications when we want to apply compression arguments with some complex oracles (such as $\mathsf{ColFinder}$) in the quantum setting.

1.4 Related Work

Rotem and Segev [RS18] showed a limitation of black-box impossibility by giving an example that overcomes the black-box impossibility result by Rudich [Rud88] by using a non-black-box reduction. Nonetheless, black-box impossibility results

[9] Note that we consider information theoretic encoder and decoder, and we do not care whether they run efficiently.

are still meaningful since we know very limited number of non-black-box techniques. Indeed, they left it as an open problem to overcome the black-box separation of CRH and OWP shown by Simon [Sim98].

Bitansky and Degwekar [BD19] gave a new proof for the black-box separation of CRH from OWP in the classical setting, which is conceptually different from previous ones [Sim98,HHRS07,AS15]. However, it is unclear if their proof extends to the quantum setting.

Holmgren and Lombardi [HL18] gave a construction of CRH based on a stronger variant of OWF which they call one-way product functions (OWPF). However, since they do not give a construction of OWPF from OWF (or OWP) even with exponential security, their result does not overcome the impossibility result by Simon [Sim98].

Chia, Hallgren, and Song [CHS18] considered the problem of separating OWP from NP hardness in the quantum setting. They ruled out a special type of quantum reductions called locally random reductions under a certain complexity theoretic assumption. We note that in our work, we do not put any restriction on a type of a reduction as long as it is quantum fully-black-box, and we do not assume any unproven assumption. Also, they focus on the separation of OWP from NP hardness, and do not give a general definition of black-box reduction in the quantum setting. Thus their work is incomparable to ours.

Hhan et al. [HXY19] also used the compression technique in the quantum setting to analyze the quantum random oracle model in the presence of auxiliary information. A crucial difference between their work and this work is that they consider a setting where an adversary is given an auxiliary information which is fixed at the beginning of a security game whereas we consider a setting where an adversary can adaptively make a query to the quantum oracle $\mathsf{ColFinder}$ during the game. Thus, our results are incomparable to theirs.

See Section 1.4 of this paper's full version [HY18] for more about related works.

1.5 Paper Organization

Section 2 describes notations, definitions, and fundamental technical lemmas that are used throughout the paper. Section 3 gives formalizations of quantum primitives and quantum fully-black-box reductions. Section 4 shows the impossibility of quantum fully-black-box reductions from CRH to OWP. Section 5 shows the impossibility of quantum fully-black-box reductions from CRH to TDP.

2 Preliminaries

A classical algorithm is a classical Turing machine, and an efficient classical algorithm is a probabilistic efficient Turing machine. We denote the set of positive integers by $\mathbb{N}$. We write A instead of $A \otimes I$ for short, for any linear operator A. For sets X and Y, let $\mathsf{Func}(X,Y)$ denote the set of functions from X to Y, and $\mathsf{Perm}(X)$ denote the set of permutations on X. Let $\Delta(f,g)$ denote the set

$\{x \in X | f(x) \neq g(x)\}$ for any functions $f, g \in \mathsf{Func}(X, Y)$. Let $\{0,1\}^*$ denote the set $\cup_{n \geq 1}\{0,1\}^n$, and by abuse of notation we let $\mathsf{Perm}(\{0,1\}^*)$ denote the set of permutations $\{P : \{0,1\}^* \to \{0,1\}^* | P(\{0,1\}^n) = \{0,1\}^n$ for each $n \geq 1\}$. When we say that $f : \{0,1\}^* \to \{0,1\}^*$ is a permutation, we assume that $f(\{0,1\}^n) = \{0,1\}^n$ holds for each n, and thus f is in $\mathsf{Perm}(\{0,1\}^*)$ (i.e., in this paper we do not treat permutations such that there exist $n \neq n'$ and $x \in \{0,1\}^n$ such that $f(x) \in \{0,1\}^{n'}$). We say that a function $f : \mathbb{N} \to \mathbb{R}$ is negligible if, for any positive integer c, $f(n) \leq n^{-c}$ holds for all sufficiently large n, and we write $f(n) \leq \mathsf{negl}(n)$.

2.1 Quantum Algorithms

We refer basics of quantum computation to [NC10,KSVV02]. In this paper, we use the computational model of quantum circuits. Let $\mathcal{Q}$ be the standard basis of quantum circuits [KSVV02]. We assume that quantum circuits (without oracle) are constructed over the standard basis $\mathcal{Q}$, and define the size of a quantum circuit as the total number of elements in $\mathcal{Q}$ used to construct it. Let $|C|$ denote the size of each quantum circuit C. An oracle-aided quantum circuit is a quantum circuit with oracle gates. When an oracle-aided quantum circuit is implemented relative to an oracle O represented by a unitary operator, the oracle gates are replaced by the unitary operator. When there are multiple oracles, each oracle gate should specify an index of an oracle. In this paper, we assume that all oracles are stateless, that is, the behavior of the oracle is independent from a previous history and the same for all queries. For a stateless quantum oracle O, we often identify the oracle and a unitary operator that represents the oracle, and use the same notation O for both of them. Note that each classical algorithm can be regarded as a quantum algorithm. We fix an encoding $\mathcal{E}$ of (oracle-aided) quantum circuits to bit strings, and we identify $\mathcal{E}(C)$ with C. For a quantum circuit C, we will denote the event that we measure an output z when we run C on an input x and measure the final state by $C(x) = z$.

First, we define quantum algorithms. We note that we only consider classical-input-output quantum algorithms.

Definition 1 (Quantum algorithms). *A quantum algorithm $\mathcal{A}$ is a family of quantum circuits $\{\mathcal{A}_n\}_{n \in \mathbb{N}}$ that acts on a quantum system $\mathcal{H}_n = \mathcal{H}_{n,in} \otimes \mathcal{H}_{n,out} \otimes \mathcal{H}_{n,work}$ for each n. When we feed $\mathcal{A}$ with an input $x \in \{0,1\}^n$, $\mathcal{A}$ runs the circuit $\mathcal{A}_n$ on the initial state $|x\rangle |0\rangle |0\rangle$, measures the final state with the computational basis, and outputs the measurement result of the register which corresponds to $\mathcal{H}_{n,out}$. We say that $\mathcal{A}$ is an efficient quantum algorithm if it is a family of polynomial-size quantum circuits, i.e., there is a polynomial $\lambda(n)$ such that $|\mathcal{A}_n| \leq \lambda(n)$ for all sufficiently large n.*

Remark 2. Though we use a Turing machine for a computational model of classical computation, we use a quantum circuit for a computational model of quantum computation. This is just because quantum circuits are better studied than quantum Turing machines [Yao93], and are easier to treat. We remark that we

do not intend to rule out reductions with full non-uniform techniques as was done in [CLMP13].

Next, we define oracle-aided quantum algorithms, which are quantum algorithms that can access to oracles.

Definition 2 (Oracle-aided quantum algorithms). *An oracle-aided quantum algorithm $\mathcal{A}$ is a family of oracle aided quantum circuits $\{\mathcal{A}_n\}_{n\in\mathbb{N}}$ that acts on a quantum system $\mathcal{H}_n = \mathcal{H}_{n,in} \otimes \mathcal{H}_{n,out} \otimes \mathcal{H}_{n,work}$ for each n. Let $O_1 = \{O_{1,i}\}_{i\in\mathbb{N}}, ..., O_t = \{O_{t,i}\}_{i\in\mathbb{N}}$ be families of quantum oracle gates. When we feed $\mathcal{A}$ with an input $x \in \{0,1\}^n$ relative to oracles $(O_1, ..., O_t)$, $\mathcal{A}$ runs the circuit $\mathcal{A}_n^{O_{1,n},...,O_{t,n}}$ on the initial state $|x\rangle|0\rangle|0\rangle$, measures the final state with the computational basis, and outputs the measurement result of the register which corresponds to $\mathcal{H}_{n,out}$.* [10] *We note that an oracle-aided quantum circuit $\mathcal{A}_n^{O_{1,n},...,O_{t,n}}$ that makes q queries can be described by a unitary operator*

$$\mathcal{A}_n^{O_{1,n},...,O_{t,n}} = \left(\prod_{j=1}^{q(n)} (U_{j,t,n}O_{t,n}\dots U_{j,1,n}O_{1,n})\right) U_{0,n}, \tag{1}$$

where $(U_{0,n}, \{U_{j,1,n}, \dots, U_{j,t,n}\}_{j\in[q]})$ are some unitary operators.

Remark 3. We also often consider an oracle access to a quantum algorithm. This is interpreted as an oracle access to a unitary operator that represents $\mathcal{A}$.

Next, we define *randomized quantum oracles*, which are quantum oracles that flip classical random coins before algorithms start.

Definition 3 (Randomized quantum oracles). *Let R_n be a finite set for each n, and $R := \prod_{n=1}^{\infty} R_n$ (note that each element $r \in R$ is an infinite sequence $(r_1, r_2, \cdots)$). A randomized quantum oracle $O := \{O_r\}_{r\in R}$ is a family of quantum oracles such that $O_{r,n} = O_{r',n}$ if $r_n = r'_n$. When we feed $\mathcal{A}$ with an input $x \in \{0,1\}^n$ relative to O, first r_n is randomly chosen from the finite set R_n (according to some distribution), and then $\mathcal{A}$ runs the circuit $\mathcal{A}_n^{O_{r,n}}$ on the initial state $|x\rangle|0\rangle|0\rangle$. We denote $O_{r,n}$ by O_{r_n} and $\{O_{r_n}\}_{r_n\in R_n}$ by O_n, respectively, and identify O with $\{O_n\}_{n\in\mathbb{N}}$.* [11]

[10] We assume that the queries are always performed in a sequential order (e.g., before each query to O_2, the adversary always makes a query to O_1), but there is no reason for an adversary to fix the order. We assume this only for an ease of notation. There are multiple ways to fix it, but changes of the order does not essentially affect (im)possibility of reductions.

[11] Note that the meaning of the symbol O_X changes depending on the set that the index X belongs to. R_n is the set of random coins for the security parameter n, and each coin $r_n \in R_n$ corresponds to one fixed unitary operator O_{r_n}. O_r is an infinite family $\{O_{r_1}, O_{r_2}, \dots\}$ for each fixed $r = (r_1, r_2, \dots) \in R$, and O_n is the finite family $\{O_{r_n}\}_{r_n\in R_n}$ for each fixed n. Each of O_r and O_n can be regarded as a subset of O. In addition, $O_{r,n}$ denotes "the n-th element of O_r" for each fixed r, which is the same as O_{r_n}.

Similarly, when $\mathcal{A}$ is given oracle access to multiple randomized oracles $(O_1, \ldots, O_t)$, we consider that an oracle gate is randomly chosen and fixed for each of the t oracles before $\mathcal{A}$ starts. The distributions of $O_1, \ldots, O_t$ can be highly dependent.

Remark 4. Later we consider the situation that a quantum algorithm $\mathcal{A}$ has access to a randomized quantum oracle O, and another quantum algorithm $\mathcal{B}$ has access to $\mathcal{A}^O$. This is interpreted as follows: Before $\mathcal{B}$ starts, $r_n \in R_n$ is chosen uniformly at random, and $\mathcal{B}$ is given an oracle access to the unitary operator that represents $\mathcal{A}_n^{O_{r_n}}$. In particular we do not change r_n while $\mathcal{B}$ is running.

Next, we define what "a quantum algorithm computes a function" means.

Definition 4 (Functions computed by quantum algorithms). *A quantum algorithm $\mathcal{A}$ computes a function $f : \{0,1\}^* \to \{0,1\}^*$ if we have $\Pr[\mathcal{A}(x) = f(x)] > 2/3$ [12] for all $n \in \mathbb{N}$ and $x \in \{0,1\}^n$. An oracle-aided quantum algorithm $\mathcal{A}$ computes a function $f : \{0,1\}^* \to \{0,1\}^*$ relative to an oracle Γ if we have $\Pr[\mathcal{A}^\Gamma(x) = f(x)] > 2/3$ for all $n \in \mathbb{N}$ and $x \in \{0,1\}^n$.*

2.2 Technical Lemmas

This section introduces some technical lemmas for later use. First, we use the following basic lemma as a fact. See textbooks on quantum computation and quantum information (e.g., [NC10]) for a proof.

Lemma 1. $\mathsf{trD}(|\psi_1\rangle\langle\psi_1|, |\psi_2\rangle\langle\psi_2|) \leq \| |\psi_1\rangle - |\psi_2\rangle \|$ *holds for any pure states $|\psi_1\rangle$ and $|\psi_2\rangle$, where trD denotes the trace distance function.*

By applying the above claim, we can show the following lemma.

Lemma 2. *Let $\Gamma = (f_1, \ldots, f_t), \Gamma' = (f'_1, \ldots, f'_t)$ be sequences of oracles, and assume that $\mathcal{A}$ is given oracle access to either Γ or Γ'. Then,*

$$\left|\Pr\left[\mathcal{A}^\Gamma(x) = z\right] - \Pr\left[\mathcal{A}^{\Gamma'}(x) = z\right]\right| \leq \left\|\mathcal{A}_n^\Gamma |x,0,0\rangle - \mathcal{A}_n^{\Gamma'} |x,0,0\rangle\right\| \quad (2)$$

holds for any input $x \in \{0,1\}^n$ and output z.

It is straightforward to show the lemma. See this paper's full version [HY18] for a complete proof.

[12] Here we are using the value 2/3 for the threshold, but it does not make any essential difference even if we use another constant c such that instead of 2/3, as long as $1/2 < c < 1$.

Swapping Lemma for Multiple Oracles. Next we introduce a generalized version of the *swapping lemma* [Vaz98, Lem. 3.1] for multiple oracles. The original swapping lemma formalizes our intuition that the measurement outcome of oracle-aided algorithm will not be changed so much even if the output values of the oracles are changed on a small fraction of inputs. Since this paper considers the situation that multiple oracles are available to adversaries, we extend the original lemma to a generalized one so that we can treat multiple oracles. To simplify notation, below we often omit the parameter n when it is clear from context (e.g., we write just q instead of $q(n)$). Here we introduce an important notion called *query magnitude*.

Query Magnitude. Let $\Gamma = (f_1, \ldots, f_t)$ be a sequence of quantum oracles, where each f_i is a fixed oracle and not randomized. Let $\mathcal{A}$ be a q-query oracle-aided quantum algorithm relative to the oracle Γ. [13]

Fix an input x, and let $|\phi_j^{f_i}\rangle$ be the quantum state of $\mathcal{A}^\Gamma$ on input $x \in \{0,1\}^n$ just before the j-th query to f_i. Without loss of generality, we consider that the unitary operator O_{f_i} acts on the first $(m_i(n) + \ell_i(n))$-qubits of the quantum system. (Here we assume that f_i is a function from $\{0,1\}^{m_i(n)}$ to $\{0,1\}^{\ell_i(n)}$.) Then $|\phi_j^{f_i}\rangle = \sum_{z \in \{0,1\}^{m_i(n)}} \alpha_z |z\rangle \otimes |\psi_z\rangle$ holds for some complex numbers α_z and quantum states $|\psi_z\rangle$. If we measure the first $m_i(n)$ qubits of the state $|\phi_j^{f_i}\rangle$ with the computational basis, we obtain z with probability $|\alpha_z|^2$. Intuitively, this probability corresponds to the "probability" that z is sent to f_i as the j-th quantum query by $\mathcal{A}$.

Definition 5 (Query magnitude to f_i).

1. *The* query magnitude of the j-th quantum query of $\mathcal{A}$ to f_i at z on input $x \in \{0,1\}^n$ *is defined by*
$$\mu_{z,j}^{\mathcal{A},f_i}(x) := |\alpha_z|^2. \tag{3}$$
2. *The* (total) query magnitude of $\mathcal{A}$ to f_i at z on input $x \in \{0,1\}^n$ *is defined by*
$$\mu_z^{\mathcal{A},f_i}(x) := \sum_j \mu_{z,j}^{\mathcal{A},f_i}(x). \tag{4}$$

The following lemma can be proven in the same way as the original swapping lemma [Vaz98, Lem. 3.1], using the hybrid argument introduced by Bennet et al. [BBBV97]. [14] See the proof for Lemma 3 of this paper's full version [HY18] for a complete proof.

Lemma 3 (Swapping lemma with multiple oracles). *Let $\Gamma = (f_1, \ldots, f_t)$, $\Gamma' = (f'_1, \ldots, f'_t)$ be sequences of oracles, where each f_i and f'_i are fixed oracles*

[13] We sometimes call a sequence of oracles just "oracle".

[14] The original swapping lemma is the special case of Lemma 3 such that $t = 1$.

and not randomized. Assume that $\mathcal{A}$ is given oracle access to either Γ or Γ'. Then

$$\left\| \mathcal{A}_n^{\Gamma} |x,0,0\rangle - \mathcal{A}_n^{\Gamma'} |x,0,0\rangle \right\| \leq 2 \sum_{1 \leq i \leq t} \sqrt{q(n) \sum_{z \in \Delta(f_i, f_i')} \mu_z^{\mathcal{A}, f_i}(x)} \tag{5}$$

holds for all $x \in \{0,1\}^n$.

3 Quantum Primitives and Black-Box Quantum Reductions

Here, we define quantum primitives, which is a quantum counterpart of a primitive, in addition to the notion of fully-black-box reduction in quantum regime (see Def. 2.1 and Def. 2.3 in [RTV04] for classical definitions). Note that we consider reductions that have quantum black-box oracle accesses to primitives. We always consider security of primitives against quantum adversaries, and do not discuss primitives that are only secure against classical adversaries. When we consider primitives with interactions in the quantum setting we have some subtle issues that do not matter in the classical setting (e.g., rewinding is sometimes hard in the quantum setting [ARU14]). Thus we treat only primitives such that both of the primitives themselves and security games are non-interactive.

Definition 6 (Quantum primitives). *A quantum primitive $\mathcal{P}$ is a pair $\langle F_{\mathcal{P}}, R_{\mathcal{P}} \rangle$, where $F_{\mathcal{P}}$ is a set of quantum algorithms $\mathcal{I}$, and $R_{\mathcal{P}}$ is a relation over pairs $\langle \mathcal{I}, \mathcal{A} \rangle$ of quantum algorithms $\mathcal{I} \in F_{\mathcal{P}}$ and $\mathcal{A}$. A quantum algorithm $\mathcal{I}$ implements $\mathcal{P}$ or is an implementation of $\mathcal{P}$ if $\mathcal{I} \in F_{\mathcal{P}}$. If $\mathcal{I} \in F_{\mathcal{P}}$ is efficient, then $\mathcal{I}$ is an efficient implementation of $\mathcal{P}$. A quantum algorithm $\mathcal{A}$ $\mathcal{P}$-breaks $\mathcal{I} \in F_{\mathcal{P}}$ if $\langle \mathcal{I}, \mathcal{A} \rangle \in R_{\mathcal{P}}$. A secure implementation of $\mathcal{P}$ is an implementation $\mathcal{I}$ of $\mathcal{P}$ such that no efficient quantum algorithm $\mathcal{P}$-breaks $\mathcal{I}$. The primitive $\mathcal{P}$ quantumly exists if there exists an efficient and secure implementation of $\mathcal{P}$.*

Definition 7 (Quantum primitives relative to oracle). *Let $\mathcal{P} = \langle F_{\mathcal{P}}, R_{\mathcal{P}} \rangle$ be a quantum primitive, and $\Gamma = (O_1, \ldots, O_t)$ be a family of (possibly randomized) quantum oracles. An oracle-aided quantum algorithm $\mathcal{I}$ implements $\mathcal{P}$ relative to Γ or is an implementation of $\mathcal{P}$ relative to Γ if $\mathcal{I}^{\Gamma} \in F_{\mathcal{P}}$. If $\mathcal{I}^{\Gamma} \in F_{\mathcal{P}}$ is efficient, then $\mathcal{I}$ is an efficient implementation of $\mathcal{P}$ relative to Γ. A quantum algorithm $\mathcal{A}$ $\mathcal{P}$-breaks $\mathcal{I} \in F_{\mathcal{P}}$ relative to Γ if $\langle \mathcal{I}^{\Gamma}, \mathcal{A}^{\Gamma} \rangle \in R_{\mathcal{P}}$. A secure implementation of $\mathcal{P}$ is an implementation $\mathcal{I}$ of $\mathcal{P}$ relative to Γ such that no efficient quantum algorithm $\mathcal{P}$-breaks $\mathcal{I}$ relative to Γ. The primitive $\mathcal{P}$ quantumly exists relative to Γ if there exists an efficient and secure implementation of $\mathcal{P}$ relative to Γ.*

Remark 5. In the above definition, $\mathcal{I}^{\Gamma}$ and $\mathcal{A}^{\Gamma}$ are considered to be quantum algorithms (rather than oracle-aided quantum algorithms) once an oracle Γ is fixed so that $\mathcal{I}^{\Gamma} \in F_{\mathcal{P}}$ and $\langle \mathcal{I}^{\Gamma}, \mathcal{A}^{\Gamma} \rangle \in R_{\mathcal{P}}$ are well-defined. This is possible since we assume that an oracle Γ is stateless. (If Γ is randomized, we regard the randomness of Γ as a part of the randomness of the quantum algorithms $\mathcal{I}^{\Gamma}$ and $\mathcal{A}^{\Gamma}$. See also Remark 4.)

Next we define quantum fully-black-box reductions, which is a quantum counterpart of fully-black-box reductions [RTV04, Def. 2.3].

Definition 8 (Quantum fully-black-box reductions). *A pair (G, S) of efficient oracle-aided quantum algorithms is a quantum fully-black-box reduction from a quantum primitive $\mathcal{P} = \langle F_{\mathcal{P}}, R_{\mathcal{P}} \rangle$ to a quantum primitive $\mathcal{Q} = \langle F_{\mathcal{Q}}, R_{\mathcal{Q}} \rangle$ if the following two conditions are satisfied:*

1. *(Correctness.) For every implementation $\mathcal{I} \in F_{\mathcal{Q}}$, we have $G^{\mathcal{I}} \in F_{\mathcal{P}}$.*
2. *(Security.) For every implementation $\mathcal{I} \in F_{\mathcal{Q}}$ and every quantum algorithm $\mathcal{A}$, if $\mathcal{A}$ $\mathcal{P}$-breaks $G^{\mathcal{I}}$, then $S^{\mathcal{A},\mathcal{I}}$ $\mathcal{Q}$-breaks $\mathcal{I}$.*

Hsiao and Reyzin showed that if there exists an oracle (family) that separates primitives $\mathcal{P}$ and $\mathcal{Q}$, then there is no fully-black-box reduction from $\mathcal{P}$ to $\mathcal{Q}$ [HR04, Prop. 1]. The following lemma guarantees that a similar claim holds in the quantum setting. Although we need no arguments which is specific to the quantum setting, we give a proof for completeness.

Lemma 4 (Two oracle technique). *There exists no quantum fully-black-box reduction from $\mathcal{P}$ to $\mathcal{Q}$ if there exist families of quantum oracles Γ_1 and $\Gamma_2 = \{\Psi_\lambda^\Phi\}_{\Phi \in \Gamma^1, \lambda \in \Lambda}$, where Λ is a non-empty set, and the following two conditions hold.*

1. **Existence of $\mathcal{Q}$.** *There exists an efficient oracle-aided quantum algorithm $\mathcal{J}_0$ that satisfies the following conditions:*
   1. *$\mathcal{J}_0^\Phi \in F_{\mathcal{Q}}$ holds for any $\Phi \in \Gamma_1$.*
   2. *For any efficient oracle-aided algorithm $\mathcal{B}$ and any $\lambda \in \Lambda$, there exists $\Phi \in \Gamma_1$ such that $\mathcal{B}^{\Phi,\Psi_\lambda^\Phi}$ does not $\mathcal{Q}$-break $\mathcal{J}_0^\Phi$.*
2. **Non-Existence of $\mathcal{P}$.** *For any efficient oracle-aided quantum algorithm $\mathcal{I}$ such that $\mathcal{I}^\Phi \in F_{\mathcal{P}}$ holds for any $\Phi \in \Gamma_1$, there exists an efficient oracle-aided quantum algorithm $\mathcal{A}_{\mathcal{I}}$ and $\lambda \in \Lambda$ such that $\mathcal{A}_{\mathcal{I}}^{\Psi_\lambda^\Phi}$ $\mathcal{P}$-breaks $\mathcal{I}^\Phi$ for any $\Phi \in \Gamma_1$.*

Proof. We prove the claim by contradiction. Suppose that there exists a quantum fully-black-box reduction (G, S) from $\mathcal{P} = \langle F_{\mathcal{P}}, R_{\mathcal{P}} \rangle$ to $\mathcal{Q} = \langle F_{\mathcal{Q}}, R_{\mathcal{Q}} \rangle$. Let $\mathcal{J}_0$ be an algorithm that satisfies the conditions on existence of $\mathcal{Q}$ in Lemma 4. Then $\mathcal{J}_0^\Phi \in F_{\mathcal{Q}}$ holds for arbitrary $\Phi \in \Gamma_1$. Hence, from the correctness of the quantum fully-black-box reductions (in Definition 8), it follows that $G^{\mathcal{J}_0^\Phi} \in F_{\mathcal{P}}$ holds for arbitrary $\Phi \in \Gamma_1$. Thus, if we set $\mathcal{I}_0 := G^{\mathcal{J}_0}$, from the second condition of Lemma 4, it follows that there exists an efficient oracle-aided quantum algorithm $\mathcal{A}_{\mathcal{I}_0}$ and $\lambda \in \Lambda$ such that $\mathcal{A}_{\mathcal{I}_0}^{\Psi_\lambda^\Phi}$ $\mathcal{P}$-breaks ${\mathcal{I}_0}^\Phi$ for any $\Phi \in \Gamma_1$. Therefore, from the second property of quantum fully-black-box reduction ("security" in Definition 8), it follows that $S^{\mathcal{A}_{\mathcal{I}_0}^{\Psi_\lambda^\Phi}, \mathcal{J}_0^\Phi}$ $\mathcal{Q}$-breaks $\mathcal{J}_0^\Phi$ for any $\Phi \in \Gamma_1$. Since G, $\mathcal{A}_{\mathcal{I}_0}$, and $\mathcal{J}_0$ are all efficient, there exists an efficient oracle-aided quantum algorithm $\mathcal{B}$ such that $\mathcal{B}^{\Phi,\Psi_\lambda^\Phi} = S^{\mathcal{A}_{\mathcal{I}_0}^{\Psi_\lambda^\Phi}, \mathcal{J}_0^\Phi}$. Now we have that there exists an efficient oracle-aided algorithm $\mathcal{B}$ and $\lambda \in \Lambda$ such that $\mathcal{B}^{\Phi,\Psi_\lambda^\Phi}$ $\mathcal{Q}$-breaks $\mathcal{J}_0^\Phi$ for any $\Phi \in \Gamma_1$. However, it contradicts the second part of the first condition of Lemma 4, which completes the proof. □

Note that, due to Lemma 4, if we want to show that there *does not exist any quantum fully-black-box reductions* from a quantum primitive $\mathcal{P}$ to another quantum primitive $\mathcal{Q}$, it suffices to show that there exists *at least one pair* of quantum oracles (Γ_1, Γ_2) that satisfies the two conditions.

Remark 6. Remember that each fixed (resp., randomized) quantum oracle O is an infinite family of unitary gates $\{O_n\}_{n\in\mathbb{N}}$ (resp., $O = \{O_n\}_{n\in\mathbb{N}}$ and $O_n = \{O_{r_n}\}_{r_n\in R_n}$, where R_n is the set of random coins), where O_n is used when an oracle-aided algorithm runs relative to O on an input in $\{0,1\}^n$. For example, (the quantum oracle of) a permutation $f \in \mathsf{Perm}(\{0,1\}^*)$ is represented as a family $\{f_n\}_{n\in\mathbb{N}}$, where $f_n = f|_{\{0,1\}^n}$. We implicitly assume that $\Psi^{\Phi}_{\lambda,n}$ depends only on Φ_n and is independent of Φ_m for $m \neq n$.

Later, to prove impossibility of quantum fully-black-box reductions from collision resistant hash functions to one-way permutations, we will apply this lemma with the condition that Λ is the set of all polynomials in n, $\Gamma_1 = \mathsf{Perm}(\{0,1\}^*)$, and $\Gamma_2 = \{\mathsf{ColFinder}^f_\lambda\}_{f\in\Gamma_1,\lambda\in\Lambda}$. Here, $\mathsf{ColFinder}^f_\lambda$ is a randomized oracle that takes, as inputs, oracle-aided quantum circuits that computes functions, and returns collision of the functions. The number $\lambda(n)$ denotes the maximum size of circuits that $\mathsf{ColFinder}^f_{\lambda,n}$ takes as inputs for each $n \in \mathbb{N}$.

3.1 Concrete Primitives

In this section, we define one-way permutations, trapdoor permutations, and collision-resistant hash functions.

We define two quantum counterparts for each classical primitive. One is the *classical-computable* primitive that can be implemented on classical computers, and the other is the *quantum-computable* primitive that can be implemented on quantum computers but may not be implemented on classical computers. Here we note that, in this paper, all adversaries are quantum algorithms for both of classical-computable and quantum-computable primitives.

Definition 9 (One-way permutation). *Quantum-computable (resp., classical-computable) quantum-secure one-way permutation* QC-qOWP*(resp.,* CC-qOWP*) is a quantum primitive defined as follows: Implementation of* QC-qOWP *(resp.,* CC-qOWP*) is an efficient quantum (resp., classical) algorithm* Eval *that computes a function $f : \{0,1\}^* \to \{0,1\}^*$ such that $f_n := f|_{\{0,1\}^n}$ is a permutation over $\{0,1\}^n$. For an implementation $\mathcal{I}$ of* QC-qOWP *(resp.,* CC-qOWP*) that computes f and a quantum algorithm $\mathcal{A}$, we say that $\mathcal{A}$* QC-qOWP*-breaks $\mathcal{I}$ (resp.,* CC-qOWP*-breaks $\mathcal{I}$) if and only if*

$$\Pr\left[x \xleftarrow{\$} \{0,1\}^n; y \leftarrow f_n(x); x' \leftarrow \mathcal{A}(y) : x' = x\right] \tag{6}$$

is non-negligible.

Remark 7. Since there is no function generation algorithm Gen in the above definition, this captures "public-coin" one-way permutations. This makes the definition of one-way permutations stronger, and thus makes our negative result stronger.

Definition 10 (Trapdoor permutation). *Quantum-computable (resp., classical-computable) quantum-secure trapdoor permutation* QC-qTDP *(resp.,* CC-QTDP*) is a quantum primitive defined as follows: Implementation of* QC-qTDP *(resp.,* CC-qTDP*) is a triplet of efficient quantum (resp., classical) algorithms* $(\mathsf{Gen}, \mathsf{Eval}, \mathsf{Inv})$*. In addition, we require* $(\mathsf{Gen}, \mathsf{Eval}, \mathsf{Inv})$ *to satisfy the following:*

1. *For any* $(\mathsf{pk}, \mathsf{td})$ *generated by* $\mathsf{Gen}(1^n)$*,* $\mathsf{Eval}(\mathsf{pk}, \cdot)$ *computes a permutation* $f_{\mathsf{pk},n}\{0,1\}^n \to \{0,1\}^n$*.*
2. *For any* $(\mathsf{pk}, \mathsf{td})$ *generated by* $\mathsf{Gen}(1^n)$ *and any* $x \in \{0,1\}^n$*, we have that the inequality* $\Pr[\mathsf{Inv}(\mathsf{td}, f_{\mathsf{pk},n}(x)) = x] > 2/3$ *holds (i.e.,* $\mathsf{Inv}(\mathsf{td}, \cdot)$ *computes* $f^{-1}_{\mathsf{pk},n}(\cdot)$*).*

For an implementation $\mathcal{I} = (\mathsf{Gen}, \mathsf{Eval}, \mathsf{Inv})$ *of* QC-qTDP *(resp.,* CC-qTDP*) and a quantum algorithm* $\mathcal{A}$*, we say that* $\mathcal{A}$ QC-qTDP*-breaks* $\mathcal{I}$ *(resp.,* CC-qTDP*-breaks* $\mathcal{I}$*) if and only if*

$$\Pr\left[(\mathsf{pk}, \mathsf{td}) \leftarrow \mathsf{Gen}(1^n); x \xleftarrow{\$} \{0,1\}^n; y \leftarrow f_{\mathsf{pk},n}(x); x' \leftarrow \mathcal{A}(\mathsf{pk}, y) : x' = x\right] \quad (7)$$

is non-negligible.

Definition 11 (Collision-resistant hash function). *Quantum-computable (resp., classical-computable) quantum-collision-resistant hash function* QC-qCRH *(resp.,* CC-qCRH*) is a quantum primitive defined as follows: Implementation of* QC-qCRH *(resp.,* CC-qCRH*) is a pair of efficient quantum (resp., classical) algorithms* $(\mathsf{Gen}, \mathsf{Eval})$*.*

$\mathsf{Gen}(1^n)$**:** *This algorithm is given* 1^n *as input, and outputs a function index.*
$\mathsf{Eval}(\sigma, x)$**:** *This algorithm is given a function index* $\sigma \in \{0,1\}^{s(n)}$ *and* $x \in \{0,1\}^{m(n)}$ *as input, and outputs* $y \in \{0,1\}^{\ell(n)}$*.*

In addition, we require $(\mathsf{Gen}, \mathsf{Eval})$ *to satisfy the following:*

1. *We have* $m(n) > \ell(n)$ *for all sufficiently large* $n \in \mathbb{N}$*.*
2. $\mathsf{Eval}(\cdot, \cdot)$ *computes a function* $H(\cdot, \cdot) : \{0,1\}^{s(n)} \times \{0,1\}^{m(n)} \to \{0,1\}^{\ell(n)}$*.*

For an implementation $\mathcal{I} = (\mathsf{Gen}, \mathsf{Eval})$ *of* QC-qCRH *(resp.,* CC-qCRH*) and a quantum algorithm* $\mathcal{A}$*, we say that* $\mathcal{A}$ QC-qCRH*-breaks* $\mathcal{I}$ *(resp.,* CC-qCRH*-breaks* $\mathcal{I}$*) if and only if*

$$\Pr\left[\sigma \leftarrow \mathsf{Gen}(1^n); (x, x') \leftarrow \mathcal{A}(\sigma) : H(\sigma, x) = H(\sigma, x')\right] \quad (8)$$

is non-negligible.

Remark 8. If we replace "quantum algorithm" with "probabilistic Turing machine" verbatim, Definition 11 completely matches the classical definition [HR04].

Remark 9. Though trapdoor permutations and collision-resistant hash functions are defined to be a tuple of algorithms, we can capture them as quantum primitives as defined in Definition 6 by considering a unified quantum algorithm that runs either of these algorithms depending on prefix of its input. We also remark that any classical algorithm can be seen as a special case of quantum computation, and thus classical-computable variants are also captured as quantum primitives.

4 Impossibility of Reduction from QC-qCRH to CC-qOWP

The goal of this section is to show the following theorem.

Theorem 3. *There exists no quantum fully-black-box reduction from* QC-qCRH *to* CC-qOWP*.*

To show this theorem, we define two (families of) oracles that separate QC-qCRH from CC-qOWP. That is, we define an oracle that implements CC-qOWP, in addition to an oracle that finds collisions of functions, and then apply the two oracle technique (Lemma 4). Our oracles are quantum analogues of those in previous works on impossibility results [Sim98,HHRS07,AS15] in the classical setting. Roughly speaking, we simply use random permutations f to implement one-way permutations. As for an oracle that finds collisions of functions, we use a randomized oracle ColFinder.

Remark 10. The statement of Theorem 3 is the strongest result among possible quantum (fully-black-box) separations of CRH from OWP, since it also excludes reductions from CC-qCRH to CC-qOWP, reductions from QC-qCRH to QC-qOWP, and reductions from CC-qCRH to QC-qOWP. [15]

Oracle ColFinder.

Intuitive Idea. Intuitively, our oracle $\mathsf{ColFinder}^f$ works as follows for each fixed permutation f. As an input, $\mathsf{ColFinder}^f$ takes an oracle-aided quantum circuit C. We say that C is a *valid* input if it computes a function $F_C^{f'} : \{0,1\}^m \to \{0,1\}^\ell$ relative to the oracle f', for arbitrary permutation f' (here we assume that m and ℓ are independent of the permutation f'). We say that C is *invalid* if it is not valid. Given the input C, first $\mathsf{ColFinder}^f$ checks whether C is invalid, and return $\perp$ if it is. Second, $\mathsf{ColFinder}^f$ chooses $w_{Cf}^{(1)} \in \{0,1\}^m$ uniformly at random, and computes $u = F_C^f(w_{Cf}^{(1)})$ by running the circuit C on input $w_{Cf}^{(1)}$ relative to f. Third, $\mathsf{ColFinder}^f$ chooses $w_{Cf}^{(2)}$ from $(F_C^f)^{-1}(u)$ uniformly at random. Finally $\mathsf{ColFinder}^f$ returns $(w_{Cf}^{(1)}, w_{Cf}^{(2)}, u)$. If F_C^f has many collisions (for example, if $m > \ell$), $\mathsf{ColFinder}^f$ returns a collision of F_C^f with a high probability. The idea of the above oracle ColFinder originally comes from the seminal work by Simon [Sim98]. Below we give a formal description of ColFinder, following the formalization of Asharov and Segev [AS15].

Formal Description. Here we give a formal description of ColFinder. Let valid and invalid denote the set of valid and invalid circuits, respectively. Let $\lambda : \mathbb{N} \to \mathbb{R}_{\geq 0}$ be a function, and $\mathsf{Circ}(\lambda(n))$ denote the set of oracle-aided quantum circuits C of which size is less than or equal to $\lambda(n)$. Note that $\mathsf{Circ}(\lambda(n))$ is a finite

[15] Note that it also excludes possible quantum (fully-black-box) reductions from collapsing hash functions to one-way permutations, since the notion of collapsing is stronger than collision-resistance.

set for each n. Let $\mathit{\Pi}_n = \{\pi_C^{(1)}, \pi_C^{(2)}\}_{C \in \mathsf{Circ}(\lambda(n)) \cap \mathsf{valid}}$ be a set of permutations, where $\pi_C^{(1)}, \pi_C^{(2)}$ are permutations over $\{0,1\}^m$, which is the domain of F_C that the circuit C computes. Let $R_{\lambda,n}$ be the set of all possible such assignments $\mathit{\Pi}_n$, and R_λ be the product set $\prod_{n=1}^{\infty} R_{\lambda,n}$.

For each fixed permutation f and a function λ, we define a randomized quantum oracle $\mathsf{ColFinder}_\lambda^f = \{\mathsf{ColFinder}_{\lambda,\mathit{\Pi}}^f\}_{\mathit{\Pi} \leftarrow R_\lambda}$. Here, by $\mathit{\Pi} \leftarrow R_\lambda$ we ambiguously denote the procedure that $\mathit{\Pi}$ is chosen uniformly at random before adversaries make queries to $\mathsf{ColFinder}_\lambda^f$, and $\mathsf{ColFinder}_{\lambda,\mathit{\Pi}}^f = \{\mathsf{ColFinder}_{\lambda,\mathit{\Pi},n}^f\}_{n \in \mathbb{N}}$ is a fixed quantum oracle for each $\mathit{\Pi}$. When we feed an algorithm $\mathcal{A}$ with an input $x \in \{0,1\}^n$ relative to $\mathsf{ColFinder}_\lambda^f$, first $\mathit{\Pi}_n \in R_{\lambda,n}$ is chosen uniformly at random (i.e., two permutations $\pi_C^{(1)}, \pi_C^{(2)}$ are chosen uniformly at random for each oracle-aided quantum circuit $C \in \mathsf{Circ}(\lambda(n)) \cap \mathsf{valid}$), and then $\mathcal{A}$ runs the circuit $\mathcal{A}_n^{\mathsf{ColFinder}_{\lambda,\mathit{\Pi},n}^f}$ on the initial state $|x\rangle |0\rangle |0\rangle$. For each fixed n and $\mathit{\Pi}_n$, the deterministic function $\mathsf{ColFinder}_{\lambda,\mathit{\Pi},n}^f$ is defined by the following procedures:

1. Take an input C, where C is an oracle-aided quantum circuit of which size is less than or equal to $\lambda(n)$.
2. Check if C is a valid input by checking whether the following condition is satisfied: For arbitrary $f'_n \in \mathsf{Perm}(\{0,1\}^n)$ and $x \in \{0,1\}^m$, there exists $y \in \{0,1\}^\ell$ such that $\Pr[C^{f'_n}(x) = y] > 2/3$. If C is an invalid input, return $\perp$.
3. Compute $w_{Cf}^{(1)} := \pi_C^{(1)}(0^m)$.
4. Compute $F_C^f(w_C^{(1)})$. That is, compute the output distribution of C^f on input $w_{Cf}^{(1)}$, find the element y such that $\Pr[C^f(w_{Cf}^{(1)}) = y] > 2/3$, and set $u \leftarrow y$.
5. Search for the minimum $t \in \{0,1\}^m$ such that $F_C^f(\pi_C^{(2)}(t)) = u$ by checking whether
$$\Pr\left[C^f\left(\pi_C^{(2)}(i)\right) = u\right] > 2/3$$
holds for $i = 0, 1, 2, \ldots$ in a sequential order, and set $w_{Cf}^2 := \pi_C^{(2)}(t)$ (note that such t always exists since $F_C^f(w_{Cf}^{(1)}) = u$).
6. Return $(w_{Cf}^{(1)}, w_{Cf}^{(2)}, u)$.

Later we will apply Lemma 4 (the two oracle technique) with $\mathit{\Gamma}_1 := \mathsf{Perm}(\{0,1\}^*)$ and $\mathit{\Gamma}_2 := \{\mathsf{ColFinder}_\lambda^f\}_{f \in \mathit{\Gamma}_1, \lambda \in \mathit{\Lambda}}$, where $\mathit{\Lambda}$ is the set of polynomials in n.

4.1 The Technically Hardest Part

The technically hardest part of proving Theorem 3 is to show the following proposition, which states that the random permutation f is hard to invert even if the additional oracle $\mathsf{ColFinder}^f$ is available for adversaries. Note that the oracle gate $\mathsf{ColFinder}_{\lambda,\mathit{\Pi},n}^f$ is (and thus the circuit $\mathcal{A}_n^{f_n, \mathsf{ColFinder}_{\lambda,\mathit{\Pi},n}^f}$ is) fixed once f_n and $\mathit{\Pi}_n$ are fixed, since the output values of $\mathsf{ColFinder}_{\lambda,\mathit{\Pi},n}^f$ are independent of f_m and $\mathit{\Pi}_m$ for $m \neq n$.

Proposition 1. *Let λ, q, ϵ be functions such that $1 \leq \lambda(n), q(n)$ and $0 < \epsilon(n) \leq 1$. Let $\mathcal{A}$ be a q-query oracle-aided quantum algorithm. Suppose that there is a function $\eta(n) \leq \lambda(n)$ such that, for each circuit C that $\mathcal{A}_n$ queries to* ColFinder*, C makes at most $\eta(n)$ queries. If*

$$\Pr_{\substack{f_n, \Pi_n \\ y \leftarrow \{0,1\}^n}} \left[x \leftarrow \mathcal{A}_n^{f_n, \mathsf{ColFinder}^f_{\lambda, \Pi, n}}(y) : f_n(x) = y \right] \geq \epsilon(n) \tag{9}$$

holds for infinitely many n, then there exists a constant const *such that*

$$\max\{q(n), \eta(n)\} \geq \mathsf{const} \cdot \epsilon(n) \cdot 2^{n/7} \tag{10}$$

holds for infinitely many n.

Below we prove Proposition 1. See Section 1.3 for an intuitive overview of our proof idea. We begin with describing some technical preparations.

Preparations. We construct another algorithm $\hat{\mathcal{A}}$ that iteratively runs $\mathcal{A}$ to increase the success probability, and then apply the encoding technique to $\hat{\mathcal{A}}$.

Let c be a positive integer. Let $\mathcal{B}_c$ be an oracle-aided quantum algorithm that runs as follows, relative to the oracles f and $\mathsf{ColFinder}^f_\lambda$. [16]

1. Take an input y. Set guess $\leftarrow \perp$.
2. For $i = 1, \ldots, c\lceil 1/\epsilon(n) \rceil$ do:
3. Run $\mathcal{A}^{f, \mathsf{ColFinder}^f_\lambda}$ on the input y. Let x denote the output.
4. Query x to f. If $f(x) = y$, then set guess $\leftarrow x$.
5. End For
6. Return guess.

Let $Q(n) := c\lceil 1/\epsilon(n) \rceil (\max\{q(n), \eta(n)\} + 1)$. Then $\mathcal{B}_c$ can be regarded as a Q-query algorithm, and for each quantum circuit C that $\mathcal{B}_c$ queries to $\mathsf{ColFinder}^f_{\lambda,n}$, C makes at most $Q(n)$ queries[17].

Remark 11. The randomness Π_n of $\mathsf{ColFinder}^f_\lambda$ is chosen before $\mathcal{B}_c$ starts, and unchanged while $\mathcal{B}_c$ is running (see Remark 4).

Lemma 5. *Let p_1, p_2 be any positive constant values such that $0 < p_1, p_2 < 1$. For a sufficiently large integer c, the following condition is satisfied for infinitely many n:*

[16] Later, we will set $\hat{\mathcal{A}} := \mathcal{B}_c$ for a constant c.

[17] We introduced Q here just for convenience. Q is an upper bound of both of i) The number of queries made by $\mathcal{B}_c$ to f and ColFinder, and ii) The number of queries to f made by quantum circuits that are queried by $\mathcal{B}_c$ to ColFinder. Because the notations in later proofs become simpler when i) and ii) are the same (i.e., $q = \eta$), we introduced Q here.

Condition. *There exist* $X \subset \mathsf{Perm}(\{0,1\}^n)$ *and* Π_n *such that* $|X| \geq p_1 \cdot |\mathsf{Perm}(\{0,1\}^n)|$ *and*

$$\Pr_{y \leftarrow \{0,1\}^n}\left[\Pr\left[x \leftarrow \mathcal{B}_{c,n}^{f_n, \mathsf{ColFinder}^f_{\lambda,\Pi,n}}(y) : f_n(x) = y\right] \geq 2/3\right] \geq p_2 \tag{11}$$

for all $f_n \in X$.

This lemma can be shown by simple averaging arguments. See the proof for Lemma 5 of this paper's full version [HY18] for a complete proof.

In what follows, we fix constants p_1, p_2 such that $0 < p_1, p_2 < 1$ arbitrarily. Then, from the above lemma, it follows that there exists a constant c that satisfies the condition in Lemma 5 for infinitely many n. Let us denote $\mathcal{B}_c$ by $\hat{\mathcal{A}}$. We use the encoding technique to this Q-query algorithm $\hat{\mathcal{A}}$, here $Q(n) = c\lceil 1/\epsilon(n)\rceil(\max\{q(n), \eta(n)\} + 1)$. Below we fix a sufficiently large n in addition to Π_n and X such that the condition in Lemma 5 is satisfied. For simplicity, we write Q, q, ϵ, η, f, and $\mathsf{ColFinder}^f$ instead of $Q(n)$, $q(n)$, $\epsilon(n)$, $\eta(n)$, f_n, and $\mathsf{ColFinder}^f_{\lambda,\Pi,n}$ respectively, for simplicity.

Information Theoretic Property of Randomized Compression Scheme. Here we introduce an information theoretic property of a randomized compression scheme $(E_r : X \to Y \cup \{\bot\}, D_r : Y \to X \cup \{\bot\})$, where r is chosen according to a distribution $\mathcal{R}$. Generally, if encoding and subsequent decoding succeed with a constant probability p, then $|Y|$ cannot be much smaller than $|X|$:

Lemma 6 ([DTT10], Fact 10.1). *If there exists a constant* $0 \leq p \leq 1$ *such that* $\Pr_{r\sim\mathcal{R}}[D_r(E_r(x)) = x] \geq p$ *holds for all* $x \in X$*, then* $|Y| \geq p \cdot |X|$ *holds.*

Below we formally define an encoder E and a decoder D that compress elements (truth tables of permutations) in X. In the encoder E, random coin r is chosen according to a distribution $\mathcal{R}$. On the other hand, we consider that D is deterministic rather than randomized, and regard r as a part of inputs to D. Note that we do not care whether encoding and decoding can be efficiently done, since Lemma 6 describes a purely information theoretic property.

Encoder E. Let δ be a sufficiently small constant ($\delta = (1/8)^4$ suffices). When we feed E with $f \in X$ as an input, E first chooses subsets $R, R' \subset \{0,1\}^n$ by the following sampling: For each $x \in \{0,1\}^n$, x is added to R with probability $\delta^{3/2}/Q^2$, and independently added to R' with probability $\delta^{5/2}/Q^4$. (The pair (R, R') is the random coin of E.)

According to the choice of R', "bad" inputs (oracle-aided quantum circuits) to $\mathsf{ColFinder}^f$ are defined for each $x \in \{0,1\}^n$ as follows. Note that now $\pi_C^{(1)}$ and $\pi_C^{(2)}$ have been fixed for each oracle-aided quantum circuit $C \in \mathsf{Circ}(\lambda(n)) \cap \mathsf{valid}$, and thus the output $\mathsf{ColFinder}^f(C) = (w_{C^f}^{(1)}, w_{C^f}^{(2)}, F_C^f(w_{C^f}^{(1)}))$ is uniquely determined. Since C is an oracle-aided quantum circuit, we can define the query magnitude of C to f on input $w_{C^f}^{(1)}$ and $w_{C^f}^{(2)}$ at $z \in \{0,1\}^n$ (see Definition 5).

We say that a quantum circuit $C \in \mathsf{Circ}(\lambda(n)) \cap \mathsf{valid}$ is *bad* relative to x if $\sum_{z \in R' \setminus \{x\}} \mu_z^{C,f}(w_{Cf}^{(1)}) > \delta/Q$ or $\sum_{z \in R' \setminus \{x\}} \mu_z^{C,f}(w_{Cf}^{(2)}) > \delta/Q$ hold, and otherwise we say that C is *good* relative to x. Let $\mathsf{badC}(R', x)$ denote the set of bad circuits relative to x, for each $R' \subset \{0,1\}^n$.

Next, E constructs a set $G \subset \{0,1\}^n$ depending on the input f. Let $I \subset \{0,1\}^n$ be the set of elements x such that $\hat{\mathcal{A}}$ successfully inverts $f(x)$, i.e., $I := \{x \mid \Pr[x' \leftarrow \hat{\mathcal{A}}^{f,\mathsf{ColFinder}^f}(f(x)) : x' = x] \geq 2/3\}$. Then $|I| \geq p_2 \cdot 2^n$ holds by definition of X (Remember that X is chosen in such a way as to satisfy the condition in Lemma 5). Now, a set G is defined to be the set of elements $x \in I$ that satisfies the following conditions:

Conditions for G.

(Cond. 1) $x \in R \cap R'$.
(Cond. 2) $\sum_{z \in R \setminus \{x\}} \mu_z^{\hat{\mathcal{A}},f}(f(x)) \leq \delta/Q$.
(Cond. 3) $\sum_{C \in \mathsf{badC}(R',x)} \mu_C^{\hat{\mathcal{A}},\mathsf{ColFinder}^f}(f(x)) \leq \delta/Q$.

Finally, E encodes f into $(f|_{\{0,1\}^n \setminus G}, f(G))$ if $|G| \geq \theta$, where $\theta = (1 - 60\sqrt{\delta})\delta^4 p_2 2^n/2Q^6$. Otherwise E encodes f into $\perp$.

In addition, here we formally define the set Y (the range of E) as

$$Y := \left\{(f|_{\{0,1\}^n \setminus G}, f(G)) \mid f \in \mathsf{Perm}(\{0,1\}^n), G \subset \{0,1\}^n, |G| \geq \theta\right\}. \qquad (12)$$

In fact $E((R,R'),f) \in Y \cup \{\perp\}$ holds for any choice of (R,R') and any permutation $f \in X$.

Decoder D. D takes $(\tilde{f}, \tilde{G})$ as an input in addition to (R,R'), where $\tilde{G} \subset \{0,1\}^n$ and $\tilde{f}$ is a bijection from a subset of $\{0,1\}^n$ onto $\{0,1\}^n \setminus \tilde{G}$, and R, R' are subsets of $\{0,1\}^n$. If $\{0,1\}^n \setminus$ (the domain of $\tilde{f}$) $\not\subset R \cap R'$ holds, then D outputs $\perp$. Otherwise, D decodes $(\tilde{f}, \tilde{G})$ and reconstructs the truth table of a permutation $f \in \mathsf{Perm}(\{0,1\}^n)$ as follows.

For each x in the domain of $\tilde{f}$, D infers the value $f(x)$ as $f(x) := \tilde{f}(x)$. For other elements $x \in \{0,1\}^n$ which is not contained in the domain of $\tilde{f}$, what D now knows is only that $f(x)$ is contained in $\tilde{G}$. To determine the remaining part of the truth table of f, D tries to recover the value $f^{-1}(y)$ for each $y \in \tilde{G}$ by using $\hat{\mathcal{A}}$.

For each fixed $y \in \tilde{G}$, D could succeed to recover the value $f^{-1}(y)$ if D were able to determine the output distribution of $\hat{\mathcal{A}}$ on input y relative to oracles f and $\mathsf{ColFinder}^f$. However, D cannot determine the distribution even though D has no limitation on its running time, since f itself is the permutation of which D wants to reconstruct the truth table, and the behavior of $\mathsf{ColFinder}^f$ depends on f. Thus D instead prepares oracles h_y and SimCF^{h_y} which approximates f and $\mathsf{ColFinder}^f$, respectively, and computes the output distribution of $\hat{\mathcal{A}}^{h_y,\mathsf{SimCF}^{h_y}}$ on input y. SimCF^{h_y} uses a subroutine CalC_y that takes (C,w) as an input (C is a valid oracle-aided circuit that may make queries to f and computes a function

F_C^f, and w is an element of the domain of F_C^f) and simulates the evaluation of $F_C^f(w)$. D finally infers that $f^{-1}(y)$ is the element which $\hat{\mathcal{A}}^{h_y,\mathsf{SimCF}^{h_y}}$ outputs with probability greater than $1/2$. (If there does not exist such an element, then D outputs $\perp$.) Below we describe h_y, CalC_y, and SimCF^{h_y}.

Oracle h_y. The oracle (function) $h_y : \{0,1\}^n \to \{0,1\}^n$ is defined by

$$h_y(z) = \begin{cases} \tilde{f}(z) \text{ if } z \notin R \cap R', \\ y \text{ otherwise.} \end{cases} \tag{13}$$

Subroutine CalC_y. Let $P_{\mathrm{candidate}} := \{h' \in \mathsf{Perm}(\{0,1\}^n)) \mid \Delta(h', h_y) \subset R \cap R'\}$. CalC_y is defined as the following procedures.

1. Take an input (C, w), where C is an oracle-aided circuit and w is an element of the domain of the function F_C.
2. Compute the output distribution of the quantum circuit $C^{h'}$ on input w for each $h' \in P_{\mathrm{candidate}}$, and find $u(C, w, h') \in \{0,1\}^\ell$ such that $\Pr[C^{h'}(w) = u(C, w, h')] > 1/2$. If there is no such value $u(C, w, h')$ for a fixed h', set $u(C, w, h') := \perp$.
3. If $u(C, w, h') = u(C, w, h'') \neq \perp$ for all $h', h'' \in P_{\mathrm{candidate}}$, return the value $u(C, w, h')$. Otherwise return $\perp$.

Oracle SimCF^{h_y}. SimCF^{h_y} is defined as the following procedures:

1. Take an input C, where C is an oracle-aided quantum circuit of which size is less than or equal to $\lambda(n)$.
2. Check if C is a valid input by checking whether the following condition is satisfied: For arbitrary $f'_n \in \mathsf{Perm}(\{0,1\}^n)$ and $x \in \{0,1\}^m$, there exists $y \in \{0,1\}^\ell$ such that $\Pr[C^{f'_n}(x) = y] > 2/3$. If C is an invalid input, return $\perp$.
3. Compute $\tilde{w}_{Cf}^{(1)} := \pi_C^{(1)}(0^m)$.
4. If $\mathsf{CalC}_y(C, \tilde{w}_{Cf}^{(1)}) = \perp$, return $\perp$.
5. Otherwise, search the minimum $t \in \{0,1\}^m$ such that $\mathsf{CalC}_y(C, \tilde{w}_{Cf}^{(1)}) = \mathsf{CalC}_y(C, \pi_C^{(2)}(t))$ by checking whether $\mathsf{CalC}_y(C, \tilde{w}_{Cf}^{(1)}) = \mathsf{CalC}_y(C, \pi_C^{(2)}(i))$ holds for $i = 0, 1, 2, \ldots$ in a sequential order, and set $\tilde{w}_{Cf}^{(2)} := \pi_C^{(2)}(t)$.
6. Return $(\tilde{w}_{Cf}^{(1)}, \tilde{w}_{Cf}^{(2)}, \mathsf{CalC}_y(C, \tilde{w}_{Cf}^{(1)}))$.

Note that D is an information theoretic decoder, and we do not care whether CalC_y and SimCF^{h_y} run efficiently.

Analysis. Here we provide a formal analysis of encoding scheme's success probability. See Section 1.3 for an intuitive overview. The following lemma shows that h_y, CalC_y, and SimCF^{h_y} satisfy some suitable properties. Here we consider the situation that D takes an input $(\tilde{f}, \tilde{G})$ such that $(\tilde{f}, \tilde{G}) = E((R, R'), f)$ for some subsets $R, R' \subset \{0,1\}^n$ and a permutation $f \in \mathsf{Perm}(\{0,1\}^n)$, and tries to recover the value $f^{-1}(y)$ for some $y \in \tilde{G}$.

Lemma 7. *h_y, CalC_y, and SimCF_{h_y} satisfy the following properties.*

1. *$\Delta(h_y, f) = R \cap R' \setminus \{f^{-1}(y)\}$ holds.*
2. *$\mathsf{CalC}_y(C, w) = F_C^f(w)$ or $\perp$ holds for any $C \in \mathsf{Circ}(\lambda(n)) \cap \mathsf{valid}$ and w.*
3. *$\mathsf{CalC}_y(C, w_{Cf}^{(1)}) = F_C^f(w_{Cf}^{(1)})$ and $\mathsf{CalC}_y(C, w_{Cf}^{(2)}) = F_C^f(w_{Cf}^{(2)})$ hold for each circuit $C \in \mathsf{Circ}(\lambda(n)) \cap \mathsf{valid}$ which is good relative to $f^{-1}(y)$.*
4. *$\mathsf{SimCF}^{h_y}(C) = \mathsf{ColFinder}^f(C)$ holds for each circuit $C \in \mathsf{Circ}(\lambda(n)) \cap \mathsf{valid}$ which is good relative to $f^{-1}(y)$. In particular, $\Delta(\mathsf{ColFinder}^f, \mathsf{SimCF}^{h_y}) \subset \mathsf{badC}(R', f^{-1}(y))$ holds.*

Proof. The first property is obviously satisfied by definition of h_y.

For the second property, since $f \in P_{\mathrm{candidate}}$, if $\mathsf{CalC}_y(C, w) \neq \perp$ then we have $\mathsf{CalC}_y(C, w) = u(C, w, f) \neq \perp$ by definition of CalC_y, and $u(C, w, f) = F_C^f(w)$ always holds. Hence the second property holds.

For the third property, for each $h' \in P_{\mathrm{candidate}}$, from Lemma 2 we have

$$\Pr\left[C^{h'}(w_{Cf}^{(1)}) = F_C^f(w_{Cf}^{(1)})\right] \geq \Pr\left[C^f(w_{Cf}^{(1)}) = F_C^f(w_{Cf}^{(1)})\right] - \left\| C^f\,|w_{Cf}^{(1)}, 0, 0\rangle - C^{h'}\,|w_{Cf}^{(1)}, 0, 0\rangle \right\|. \tag{14}$$

From the swapping lemma (Lemma 3) it follows that

$$\left\| C^f\,|w_{Cf}^{(1)}, 0, 0\rangle - C^{h'}\,|w_{Cf}^{(1)}, 0, 0\rangle \right\| \leq 2\sqrt{Q \sum_{z \in \Delta(f,h')} \mu_z^{C,f}(w_{Cf}^{(1)})}. \tag{15}$$

Since $\Delta(f, h') \subset R \cap R' \setminus \{f^{-1}(y)\} \subset R' \setminus \{f^{-1}(y)\}$ holds for all $h' \in P_{\mathrm{candidate}}$, and C is a good circuit relative to $f^{-1}(y)$, the right hand side of the above inequality is upper bounded by $2\sqrt{\delta}$. Thus, for a sufficiently small δ we have

$$\Pr\left[C^{h'}(w_{Cf}^{(1)}) = F_C^f(w_{Cf}^{(1)})\right] \geq \frac{2}{3} - 2\sqrt{\delta} > \frac{1}{2}, \tag{16}$$

which implies that $u(C, w_{Cf}^{(1)}, h') = F_C^f(w_{Cf}^{(1)})$ holds for every $h' \in P_{\mathrm{candidate}}$. Thus $\mathsf{CalC}_y(C, w_{Cf}^{(1)}) = F_C^f(w_{Cf}^{(1)})$ holds if C is good relative to $f^{-1}(y)$. The equality $\mathsf{CalC}_y(C, w_{Cf}^{(2)}) = F_C^f(w_{Cf}^{(2)})$ can be shown in the same way.

The fourth property follows from the definition of SimCF^{h_y}, the second property, and the third property. □

The following lemma shows that the decoding always succeeds if the encoding succeeds.

Lemma 8. *If $E((R, R'), f) \neq \perp$, then $D((R, R'), E((R, R'), f)) = f$ holds.*

Proof (of Lemma 8). Let $\tilde{f} := f|_{\{0,1\}^n \setminus G}$ and $\tilde{G} := f(G)$. We show that D can correctly recover $x = f^{-1}(y)$ for each $y \in \tilde{G}$.

We apply the swapping lemma (Lemma 3) to the oracle pairs $(f, \mathsf{ColFinder}^f)$ and $(h_y, \mathsf{SimCF}^{h_y})$. Then we have

$$\begin{aligned}&\left\|\hat{\mathcal{A}}_n^{f,\mathsf{ColFinder}^f}\,|f(x),0,0\rangle - \hat{\mathcal{A}}_n^{h_y,\mathsf{SimCF}^{h_y}}\,|f(x),0,0\rangle\right\| \\ &\quad\le 2\sqrt{Q\sum_{z\in\Delta(f,h_y)}\mu_z^{\hat{\mathcal{A}},f}(f(x))} + 2\sqrt{Q\sum_{C\in\Delta(\mathsf{ColFinder}^f,\mathsf{SimCF}^{h_y})}\mu_C^{\hat{\mathcal{A}},\mathsf{ColFinder}^f}(f(x))}.\end{aligned} \tag{17}$$

Since $\Delta(f,h_y) = R\cap R'\setminus\{f^{-1}(y)\} \subset R\setminus\{f^{-1}(y)\} = R\setminus\{x\}$ and $\Delta(\mathsf{ColFinder}^f, \mathsf{SimCF}^{h_y}) \subset \mathsf{badC}(R', f^{-1}(y)) = \mathsf{badC}(R', x)$ from Lemma 7, the right hand side of inequality (17) is upper bounded by

$$2\sqrt{Q\sum_{z\in R\setminus\{x\}}\mu_z^{\hat{\mathcal{A}},f}(f(x))} + 2\sqrt{Q\sum_{C\in\mathsf{badC}(R',x)}\mu_C^{\hat{\mathcal{A}},\mathsf{ColFinder}^f}(f(x))}. \tag{18}$$

Due to the conditions (Cond. 2) and (Cond. 3) (see p. 21), each term of the above expression is upper bounded by $2\sqrt{\delta}$. Thus, eventually we have

$$\left\|\hat{\mathcal{A}}_n^{f,\mathsf{ColFinder}^f}\,|f(x),0,0\rangle - \hat{\mathcal{A}}_n^{h_y,\mathsf{SimCF}^{h_y}}\,|f(x),0,0\rangle\right\| \le 4\sqrt{\delta} \tag{19}$$

Finally, from Lemma 2, for sufficiently small δ it follows that

$$\begin{aligned}&\Pr\left[\hat{\mathcal{A}}^{h_y,\mathsf{SimCF}^{h_y}}(f(x)) = x\right] \\ &\quad\ge \Pr\left[\hat{\mathcal{A}}^{f,\mathsf{ColFinder}^f}(f(x)) = x\right] \\ &\qquad - \left\|\hat{\mathcal{A}}_n^{f,\mathsf{ColFinder}^f}\,|f(x),0,0\rangle - \mathcal{A}_n^{h_y,\mathsf{ColFinder}^h}\,|f(x),0,0\rangle\right\| \\ &\quad\ge 2/3 - 4\sqrt{\delta} > 1/2,\end{aligned} \tag{20}$$

which implies that D correctly recovers $x = f^{-1}(y)$. □

The following lemma is a generalization of a claim showed by Nayebi et al [NABT15, Claim 8], which shows that our E and D work well with a constant probability. See the proof for Lemma 9 of this paper's full version [HY18] for a complete proof.

Lemma 9. *If $Q^6 \le \delta^4 p_2 2^n/32$,*

$$\Pr_{(R,R')}\left[D((R,R'), E((R,R'), f) = f\right] \ge 0.7 \tag{21}$$

holds for each $f \in X$.

Finally, we show that Proposition 1 follows from the above lemmas.

Proof (of Proposition 1). First, remember that the set Y is defined as

$$Y := \left\{ (f|_{\{0,1\}^n \setminus G}, f(G)) \mid f \in \mathsf{Perm}(\{0,1\}^n), G \subset \{0,1\}^n, |G| \geq \theta \right\}. \tag{22}$$

For each fixed positive integer $\theta \leq M \leq 2^n$, the cardinality of the set

$$Y_M := \{ (f|_{\{0,1\}^n \setminus G}, f(G)) \mid f \in \mathsf{Perm}(\{0,1\}^n), G \subset \{0,1\}^n, |G| = M \} \tag{23}$$

is equal to $(2^n - M)! \cdot \binom{2^n}{M} = (2^n)!/M!$. Thus $|Y|$ is upper bounded as

$$|Y| = \sum_{M=\lceil \theta \rceil}^{2^n} \frac{(2^n)!}{M!} \leq 2^n \cdot \frac{(2^n)!}{(\lceil \theta \rceil)!} \tag{24}$$

for sufficiently large n. Here we show the following claim.

Claim. If $Q^6 \leq \delta^4 p_2 2^n/32$, there exists a constant const_1 such that $Q^6 \geq \mathsf{const}_1 \cdot 2^n/n$ holds. We can choose const_1 independently of n.

Proof (of Claim). By definition of X, $|X| \geq p_1(2^n)!$ holds. In addition, from inequality (24), we have $|Y| \leq 2^n \cdot \frac{(2^n)!}{(\lceil \theta \rceil)!}$. Moreover, since now we are assuming that $Q^6 \leq \delta^4 p_2 2^n/32$ holds, it follows that $|Y| \geq 0.7|X|$ from Lemma 6 and Lemma 9. Hence we have $2^n \cdot \frac{(2^n)!}{(\lceil \theta \rceil)!} \geq 0.7 \cdot p_1(2^n)!$, which is equivalent to $\frac{2^n}{0.7 p_1} \geq \lceil \theta \rceil !$.

Since p_1 is a constant and $n! \geq 2^n$ holds for $n \geq 4$, there exists a constant const_2, which can be taken independently of n, such that $\lceil \mathsf{const}_2 \cdot n \rceil ! \geq 2^n/(0.7 p_1)$ holds. Now we have $\lceil \mathsf{const}_2 \cdot n \rceil \geq \lceil \theta \rceil$, which implies that $\mathsf{const}_2 \cdot n + 1 \geq \theta = \delta^4 \left(1 - 60\sqrt{\delta}\right) \frac{p_2 2^n}{2Q^6}$ holds. Moreover, since δ and p_2 are also constants, there exists a constant const_1 that is independent of n and $Q^6 \geq \mathsf{const}_1 \cdot 2^n/n$ holds, which completes the proof of the claim. □

Let $\mathsf{const}_3 := \min\{\delta^4 p_2/32, \mathsf{const}_1\}$. Then, from the the above claim, it follows that $Q^6 \geq \mathsf{const}_3 \cdot 2^n/n$ holds. Since $Q = c \left\lceil \frac{1}{\epsilon} \right\rceil (\max\{q, \eta\} + 1)$ by definition of Q, we have $c^6 \left\lceil \frac{1}{\epsilon} \right\rceil^6 (\max\{q, \eta\} + 1)^6 \geq \mathsf{const}_3 \cdot 2^n/n$. Hence there exists a constant const such that $\max\{q, \eta\} \geq \mathsf{const} \cdot \epsilon \cdot 2^{n/6}/n^{1/6} \geq \mathsf{const} \cdot \epsilon \cdot 2^{n/7}$ holds for all sufficiently large n, which completes the proof. □

4.2 Proof of Theorem 3

This section shows that Theorem 3 follows from Proposition 1. First, we can show that the following lemma follows from Proposition 1.

Lemma 10. *For any efficient oracle-aided quantum algorithm $\mathcal{B}$ and for any polynomial λ, there exists a permutation $f : \{0,1\}^* \to \{0,1\}^*$ such that*

$$\Pr_{y \leftarrow \{0,1\}^n} \left[x \leftarrow \mathcal{B}^{f, \mathsf{ColFinder}^f_\lambda}(y) : f(x) = y \right] < 2^{-n/8} \tag{25}$$

holds for all sufficiently large n.

The proof of the lemma is straightforward. See the proof of Lemma 10 in this paper's full version [HY18] for a complete proof.

Proof (of Theorem 3). Let $\Gamma_1 := \mathsf{Perm}(\{0,1\}^*)$ and $\Gamma_2 := \{\mathsf{ColFinder}^f_\lambda\}_{f\in\Gamma_1,\lambda\in\Lambda}$, where Λ is the set of all polynomials in n. (If $\lambda(n) \leq 0$ for some n, we assume that $\mathsf{ColFinder}^f_{\lambda,n}$ does not take any inputs.) Below we show that the two conditions of Lemma 4 are satisfied.

For the first condition of Lemma 4, we define an oracle-aided quantum algorithm $\mathcal{J}_0$ as follows: When we feed $\mathcal{J}_0$ with an input x relative to a permutation f, $\mathcal{J}_0$ queries x to f and obtains the output $f(x)$. Then $\mathcal{J}_0$ returns $f(x)$ as its output. We show that this algorithm $\mathcal{J}_0$ satisfies the first condition of Lemma 4 (existence of $\mathsf{CC\text{-}qOWP}$). It is obvious that $\mathcal{J}_0^f \in F_{\mathsf{CC\text{-}qOWP}}$ for any permutation f, by definition of $\mathcal{J}_0$. Let $\mathcal{B}$ be an efficient oracle-aided quantum algorithm, and λ be a polynomial in n.

From Lemma 10, it follows that, for any efficient oracle-aided quantum algorithm $\mathcal{B}$ and any $\lambda \in \Lambda$, there exists a permutation f such that

$$\Pr_{y\leftarrow\{0,1\}^n}\left[x \leftarrow \mathcal{B}^{f,\mathsf{ColFinder}^f_\lambda}(y) : f(x) = y\right] < \mathsf{negl}(n) \tag{26}$$

holds, which implies that $\mathcal{B}^{f,\mathsf{ColFinder}^f_\lambda}$ does not $\mathsf{CC\text{-}qOWP}$-break $\mathcal{J}_0^f$ relative to $(f, \mathsf{ColFinder}^f_\lambda)$. Hence the first condition (existence of $\mathsf{CC\text{-}qOWP}$) of Lemma 4 is satisfied.

Next, we show that the second condition (non-existence of $\mathsf{QC\text{-}qCRH}$) of Lemma 4 is satisfied. For any efficient oracle-aided quantum algorithm $\mathcal{I} = (\mathsf{Gen}, \mathsf{Eval})$ such that $\mathcal{I}^f \in F_{\mathsf{CC\text{-}qCRH}}$ holds for any permutation f, let λ be a polynomial such that $\lambda(n) > |\mathcal{I}_n|$ for all n. We define a family of oracle-aided quantum algorithms $\mathcal{A}_\mathcal{I}$ as follows: Given an input σ, $\mathcal{A}_\mathcal{I}$ queries the oracle-aided quantum circuit $\mathsf{Eval}_n(\sigma,\cdot)$ to $\mathsf{ColFinder}^f_\lambda$, obtains an answer $(w^{(1)}, w^{(2)}, H^f(\sigma, w^{(1)}))$ [18], and finally outputs $(w^{(1)}, w^{(2)})$. When $\mathcal{A}_\mathcal{I}^{\mathsf{ColFinder}^f_\lambda}$ is given an input σ, the output will be $(w^{(1)}, w^{(2)})$, where $w^{(1)}$ is uniformly distributed over the domain of $H^f(\sigma,\cdot) : \{0,1\}^{m(n)} \to \{0,1\}^{\ell(n)}$ and $w^{(2)}$ is uniformly distributed over the set $(H^f(\sigma,\cdot))^{-1}(H^f(\sigma, w^{(1)}))$. Since $m(n) > \ell(n)$ holds by definition of implementations of $\mathsf{QC\text{-}qCRH}$, the probability that $w^{(1)} \neq w^{(2)}$, which implies that $(w^{(1)}, w^{(2)})$ is a collision of $H^f(\sigma,\cdot)$, is at least $1/4$. Thus it follows that there exists $\mathcal{A}_\mathcal{I}$ and $\lambda \in \Lambda$ such that $\mathcal{A}_\mathcal{I}^{\mathsf{ColFinder}^f_\lambda}$ $\mathsf{CC\text{-}qCRH}$-breaks $\mathcal{I}^f$ for any permutation f. Hence the second condition of Lemma 4 is satisfied. □

5 Impossibility of Reduction from QC-qCRH to CC-qTDP

As well as the impossibility of reduction from $\mathsf{QC\text{-}qCRH}$ to $\mathsf{CC\text{-}qOWP}$, we can show the following theorem.

[18] Since $\mathcal{I}^{f'} \in F_{\mathsf{CC\text{-}qCRH}}$ for any permutation f', $\mathsf{Eval}^{f'}_n(\cdot,\cdot)$ computes a function $H^{f'}(\cdot,\cdot)$ for any permutation f' by definition of $\mathsf{QC\text{-}qCRH}$. In particular, even when σ is generated by $\mathsf{Gen}^f(1^n)$ and $f' \neq f$, $\mathsf{Eval}^{f'}_n(\sigma,\cdot)$ computes the function $H^{f'}(\sigma,\cdot)$. Hence $\mathsf{ColFinder}^f_\lambda$ does not return $\perp$ on the input $\mathsf{Eval}_n(\sigma,\cdot)$.

Theorem 4. *There exists no quantum fully-black-box reduction from* QC-qCRH *to* CC-qTDP*.*

Remark 12. The statement of Theorem 4 is the strongest result among possible quantum (fully-black-box) separations of CRH from TDP, since it also excludes reductions from CC-qCRH to CC-qTDP, reductions from QC-qCRH to QC-qTDP, and reductions from CC-qCRH to QC-qTDP. [19]

Here we give only a proof intuition. See Section 5 of this paper's full version [HY18] for a complete proof.

Proof intuition. To show this theorem, again we define two oracles that separate QC-qCRH from CC-qTDP. That is, we define an oracle (g, f, f^{inv}) that implements random trapdoor permutations, in addition to an oracle $\mathsf{ColFinder}^{g,f,f^{\mathrm{inv}}}$ that finds collisions of functions, and then apply Lemma 4 (the two oracle technique). Here, $g : \{0,1\}^n \to \{0,1\}^n$ is a random permutation and $f : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$ is a family of random permutations $(f(z,\cdot) : \{0,1\}^n \to \{0,1\}^n$ is a random permutation for each $z \in \{0,1\}^n)$. f^{inv} is the inverse of f defined by $f^{\mathrm{inv}}(z,\cdot) := (f(g(z),\cdot))^{-1}$. At the beginning of each game, a trapdoor $\mathsf{td} \in \{0,1\}^n$ is chosen randomly, and a public key pk is set as $\mathsf{pk} := g(\mathsf{td}) \in \{0,1\}^n$. We consider the situation that each adversary $\mathcal{A}$ is given the public key pk and a randomly chosen target y to invert, in addition to oracle accesses to (g, f, f^{inv}) and $\mathsf{ColFinder}^{g,f,f^{\mathrm{inv}}}$, and $\mathcal{A}$ tries to find x such that $f(\mathsf{pk}, x) = y$.

Recall that the most technically difficult part of the proof in Section 4 was to show that, if $\mathcal{A}$ inverts a random permutation with a high probability, it has to make exponentially many queries. Similarly the most technically difficult part to prove Theorem 4 is to show that, if $\mathcal{A}$ inverts y in $f(\mathsf{pk},\cdot)$ (with a high probability), $\mathcal{A}$ has to make exponentially many queries.

We consider three separate cases: The first case is the one that $\mathcal{A}$'s query magnitude on the trapdoor td to f^{inv} is large (we denote this event by TDHIT_1). The second case is the one that $\mathcal{A}$'s query magnitude on a quantum circuit C to $\mathsf{ColFinder}^{g,f,f^{\mathrm{inv}}}$ that queries the trapdoor td to f^{inv} is large (we denote this event by TDHIT_2). The third case is the one that both of TDHIT_1 and TDHIT_2 do not occur.

In the first and second cases, by using $\mathcal{A}$ we can construct another algorithm $\mathcal{B}$ that makes almost as much queries as $\mathcal{A}$ and inverts $\mathsf{pk} = g(\mathsf{td})$ in g (with a high probability). Since g is just a random permutation, from the results shown in Section 4 it follows that $\mathcal{B}$ has to make exponentially many queries, which implies that $\mathcal{A}$ has to make exponentially many queries. In the third case, intuitively, we can construct a randomized compression scheme that compresses the truth table of the random permutation $f(\mathsf{pk},\cdot)$ without the inverse oracle $f^{\mathrm{inv}}(\mathsf{td},\cdot)$ since the query magnitude to $f^{\mathrm{inv}}(\mathsf{td},\cdot)$ is always small if $\neg(\mathsf{TDHIT}_1 \vee \mathsf{TDHIT}_2)$ occurs.

[19] Note that it also excludes possible quantum (fully-black-box) reductions from collapsing hash functions to trapdoor permutations, since the notion of collapsing is stronger than collision-resistance.

Acknowledgements

We thank anonymous reviewers for their insightful comments. Especially, we thank reviewers of STOC 2019 and CRYPTO 2020 who pointed out technical errors in previous versions of this paper.

References

Aar09. Scott Aaronson. Quantum copy-protection and quantum money. In *CCC 2009, Proceedings*, pages 229–242, 2009.

ABF+16. Gorjan Alagic, Anne Broadbent, Bill Fefferman, Tommaso Gagliardoni, Christian Schaffner, and Michael St. Jules. Computational security of quantum encryption. In Anderson C. A. Nascimento and Paulo Barreto, editors, *ICITS 16*, volume 10015 of *LNCS*, pages 47–71. Springer, Heidelberg, August 2016.

AC12. Scott Aaronson and Paul Christiano. Quantum money from hidden subspaces. In Howard J. Karloff and Toniann Pitassi, editors, *44th ACM STOC*, pages 41–60. ACM Press, May 2012.

AGM18. Gorjan Alagic, Tommaso Gagliardoni, and Christian Majenz. Unforgeable quantum encryption. In Jesper Buus Nielsen and Vincent Rijmen, editors, *EUROCRYPT 2018, Part III*, volume 10822 of *LNCS*, pages 489–519. Springer, Heidelberg, April / May 2018.

Ajt96. Miklós Ajtai. Generating hard instances of lattice problems (extended abstract). In *28th ACM STOC*, pages 99–108. ACM Press, May 1996.

ARU14. Andris Ambainis, Ansis Rosmanis, and Dominique Unruh. Quantum attacks on classical proof systems: The hardness of quantum rewinding. In *55th FOCS*, pages 474–483. IEEE Computer Society Press, October 2014.

AS15. Gilad Asharov and Gil Segev. Limits on the power of indistinguishability obfuscation and functional encryption. In Venkatesan Guruswami, editor, *56th FOCS*, pages 191–209. IEEE Computer Society Press, October 2015.

BB84. Charles H. Bennett and Gilles Brassard. Quantum cryptography: Public key distribution and coin tossing. In *Proceedings of IEEE International Conference on Computers, Systems, and Signal Processing*, pages 175–179, India, 1984.

BBBV97. Charles H Bennett, Ethan Bernstein, Gilles Brassard, and Umesh Vazirani. Strengths and weaknesses of quantum computing. *SIAM journal on Computing*, 26(5):1510–1523, 1997.

BD19. Nir Bitansky and Akshay Degwekar. On the complexity of collision resistant hash functions: New and old black-box separations. In *TCC 2019, Part I*, LNCS, pages 422–450. Springer, Heidelberg, March 2019.

BJ15. Anne Broadbent and Stacey Jeffery. Quantum homomorphic encryption for circuits of low T-gate complexity. In Rosario Gennaro and Matthew J. B. Robshaw, editors, *CRYPTO 2015, Part II*, volume 9216 of *LNCS*, pages 609–629. Springer, Heidelberg, August 2015.

BL17. Daniel J. Bernstein and Tanja Lange. Post-quantum cryptography. *Nature*, 549:188–194, 2017.

BLP+13. Zvika Brakerski, Adeline Langlois, Chris Peikert, Oded Regev, and Damien Stehlé. Classical hardness of learning with errors. In Dan Boneh, Tim Roughgarden, and Joan Feigenbaum, editors, *45th ACM STOC*, pages 575–584. ACM Press, June 2013.

Bra18. Zvika Brakerski. Quantum FHE (almost) as secure as classical. In Hovav Shacham and Alexandra Boldyreva, editors, *CRYPTO 2018, Part III*, volume 10993 of *LNCS*, pages 67–95. Springer, Heidelberg, August 2018.

BS16. Anne Broadbent and Christian Schaffner. Quantum cryptography beyond quantum key distribution. *Des. Codes Cryptography*, 78(1):351–382, 2016.

CHS18. Nai-Hui Chia, Sean Hallgren, and Fang Song. On basing one-way permutations on NP-hard problems under quantum reductions. *CoRR*, abs/1804.10309, 2018.

CLMP13. Kai-Min Chung, Huijia Lin, Mohammad Mahmoody, and Rafael Pass. On the power of nonuniformity in proofs of security. In Robert D. Kleinberg, editor, *ITCS 2013*, pages 389–400. ACM, January 2013.

DTT10. Anindya De, Luca Trevisan, and Madhur Tulsiani. Time space tradeoffs for attacks against one-way functions and PRGs. In Tal Rabin, editor, *CRYPTO 2010*, volume 6223 of *LNCS*, pages 649–665. Springer, Heidelberg, August 2010.

GC01. Daniel Gottesman and Isaac Chuang. Quantum digital signatures. *CoRR*, abs/quant-ph/0105032, 2001.

GT00. Rosario Gennaro and Luca Trevisan. Lower bounds on the efficiency of generic cryptographic constructions. In *41st FOCS*, pages 305–313. IEEE Computer Society Press, November 2000.

HHRS07. Iftach Haitner, Jonathan J. Hoch, Omer Reingold, and Gil Segev. Finding collisions in interactive protocols - a tight lower bound on the round complexity of statistically-hiding commitments. In *48th FOCS*, pages 669–679. IEEE Computer Society Press, October 2007.

HL18. Justin Holmgren and Alex Lombardi. Cryptographic hashing from strong one-way functions (or: One-way product functions and their applications). In Mikkel Thorup, editor, *59th FOCS*, pages 850–858. IEEE Computer Society Press, October 2018.

HR04. Chun-Yuan Hsiao and Leonid Reyzin. Finding collisions on a public road, or do secure hash functions need secret coins? In Matthew Franklin, editor, *CRYPTO 2004*, volume 3152 of *LNCS*, pages 92–105. Springer, Heidelberg, August 2004.

HXY19. Minki Hhan, Keita Xagawa, and Takashi Yamakawa. Quantum random oracle model with auxiliary input. In *ASIACRYPT 2019, Part I*, LNCS, pages 584–614. Springer, Heidelberg, December 2019.

HY18. Akinori Hosoyamada and Takashi Yamakawa. Finding collisions in a quantum world: Quantum black-box separation of collision-resistance and one-wayness. *IACR Cryptology ePrint Archive: Report 2018/1066*, 2018.

IR89. Russell Impagliazzo and Steven Rudich. Limits on the provable consequences of one-way permutations. In *21st ACM STOC*, pages 44–61. ACM Press, May 1989.

JF11. David Jao and Luca De Feo. Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies. In *Post-Quantum Cryptography - 4th International Workshop, PQCrypto 2011, Taipei, Taiwan, November 29 - December 2, 2011. Proceedings*, pages 19–34, 2011.

KSVV02. Alexei Yu Kitaev, Alexander Shen, Mikhail N Vyalyi, and Mikhail N Vyalyi. *Classical and quantum computation*. Number 47. American Mathematical Soc., 2002.

Mah18. Urmila Mahadev. Classical homomorphic encryption for quantum circuits. In Mikkel Thorup, editor, *59th FOCS*, pages 332–338. IEEE Computer Society Press, October 2018.

McE78. Robert J. McEliece. A public-key cryptosystem based on algebraic coding theory. *DSN Progress Report*, 44:114–116, 1978.

NABT15. Aran Nayebi, Scott Aaronson, Aleksandrs Belovs, and Luca Trevisan. Quantum lower bound for inverting a permutation with advice. *Quantum Information & Computation*, 15(11&12):901–913, 2015.

NC10. Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information: 10th Anniversary Edition*. Cambridge University Press, 2010.

NIS16. NIST. Post-quantum cryptography standardization. 2016. See `https://csrc.nist.gov/Projects/Post-Quantum-Cryptography`.

Pei09. Chris Peikert. Public-key cryptosystems from the worst-case shortest vector problem: extended abstract. In Michael Mitzenmacher, editor, *41st ACM STOC*, pages 333–342. ACM Press, May / June 2009.

Reg05. Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In Harold N. Gabow and Ronald Fagin, editors, *37th ACM STOC*, pages 84–93. ACM Press, May 2005.

RS18. Lior Rotem and Gil Segev. Injective trapdoor functions via derandomization: How strong is Rudich's black-box barrier? In Amos Beimel and Stefan Dziembowski, editors, *TCC 2018, Part I*, volume 11239 of *LNCS*, pages 421–447. Springer, Heidelberg, November 2018.

RTV04. Omer Reingold, Luca Trevisan, and Salil P. Vadhan. Notions of reducibility between cryptographic primitives. In Moni Naor, editor, *TCC 2004*, volume 2951 of *LNCS*, pages 1–20. Springer, Heidelberg, February 2004.

Rud88. Steven Rudich. *Limits on the Provable Consequences of One-way Functions*. PhD thesis, University of California, Berkeley, 1988.

Sho94. Peter W. Shor. Algorithms for quantum computation: Discrete logarithms and factoring. In *35th FOCS*, pages 124–134. IEEE Computer Society Press, November 1994.

Sim98. Daniel R. Simon. Finding collisions on a one-way street: Can secure hash functions be based on general assumptions? In Kaisa Nyberg, editor, *EUROCRYPT'98*, volume 1403 of *LNCS*, pages 334–345. Springer, Heidelberg, May / June 1998.

Vaz98. Umesh Vazirani. On the power of quantum computation. *PHILOSOPHICAL TRANSACTIONS-ROYAL SOCIETY OF LONDON SERIES A MATHEMATICAL PHYSICAL AND ENGINEERING SCIENCES*, pages 1759–1767, 1998.

Wie83. Stephen Wiesner. Conjugate coding. *SIGACT News*, 15(1):78–88, January 1983.

Yao93. Andrew Chi-Chih Yao. Quantum circuit complexity. In *34th Annual Symposium on Foundations of Computer Science, Palo Alto, California, USA, 3-5 November 1993*, pages 352–361, 1993.

Zha19. Mark Zhandry. Quantum lightning never strikes the same state twice. In Vincent Rijmen and Yuval Ishai, editors, *EUROCRYPT 2019, Part III*, LNCS, pages 408–438. Springer, Heidelberg, May 2019.