Quantum Collision-Finding in Non-Uniform Random Functions

Marko Balogh[1], Edward Eaton[2], and Fang Song[3]

[1]Department of Physics, Portland State University marko_balogh@me.com
[2]Department of Combinatorics and Optimization, University of Waterloo eeaton@uwaterloo.ca
[3]Department of Computer Science, Portland State University fang.song@pdx.edu

Abstract

We give a complete characterization of *quantum* attacks for finding a collision in a *non-uniform* random function whose outputs are drawn according to a distribution of min-entropy k. This can be viewed as showing *generic* security of hash functions under *relaxed* assumptions in contrast to the standard heuristic of assuming uniformly random outputs. It also has applications in analyzing quantum security of the Fujisaki-Okamoto transformation [TU16]. In particular, our results close a gap in the lower bound left open in [TTU16].

Specifically, let D be a min-entropy k distribution on a set Y of size N. Let $f : X \to Y$ be a function whose output $f(x)$ is drawn according to D for each $x \in X$ independently. We show that $\Omega(2^{k/3})$ quantum queries are necessary to find a collision in f, improving the previous bound $\Omega(2^{k/9})$. In fact we show a stronger lower bound $2^{k/2}$ in some special case. For all cases, we also describe explicit quantum algorithms that find a collision with a number of queries matching the corresponding lower bounds.

1 Introduction

Hash functions are central and prominent in modern cryptography, and there have been many ingenious designs of cryptographic hash functions [Riv92, NIS95, BDPA07, NIS14]. Despite being *deterministic* functions that efficiently compute a short digest of long input strings, they exhibit certain *random* behavior. In particular, a cryptographic hash function H is believed, backed with intensive tests in practice, to be *collision resistant*. Namely, it should be computationally unfeasible to find a *collision*, which is a pair of distinct input strings (x, x') with $H(x) = H(x')$. Because of these nice features, hash functions are being used in numerous cryptographic constructions and applications, e.g., protecting passwords [PHC12], constructing message authentication codes and digital signature schemes, as well as various crypto-currencies exemplified by BitCoin [Nak08].

Theoretical analysis of a hash function H often refers to *generic* security, where one ignores the internal designs of H and views it as a black box. Moreover, the output of H is assumed to have been drawn *uniformly* at random from some codomain of size N. The complexity of finding a collision is then measured by the number of evaluations of H, i.e., queries to the black box. By the well-known *birthday* bound, $\Theta(\sqrt{N})$ queries are both sufficient and necessary to find a collision in H. This principle is extended and formalized as the *random oracle* model, in which a hash function is treated as a truly random function that is publicly available but only through oracle queries [BR93]. This heuristic has been widely adopted to construct more efficient cryptosystems and facilitate security reduction proofs which are otherwise challenging or even impossible [BR94, FO13].

However, in reality, there are attacks that perform significantly better than the plain birthday attack. The recent explicit break of full SHA-1 by Google [SBK$^+$17], where two PDF files can be generated that collide on the same 160-bit digest, only takes $\sim 2^{61}$ hash evaluations instead of 2^{80}. Stronger attacks like these are possible because the internal structure of H may create opportunities for more effective cryptanalysis. A natural reaction would be to open up the "black box" and take into account the inner workings when analyzing a hash function. Clearly a case-by-case study of every existing and new construction will require much more work. Alternatively, *can we prove generic security bounds, but under relaxed and/or more accurate assumptions?*

The approaching era of quantum computing will make these challenges far more pressing. The power of quantum computers, while promising in accelerating the resolution of fundamental problems in many areas such as chemistry, biology, etc., represents a tremendous threat to cryptography. Many public key cryptosystems will be broken due the Shor's efficient quantum algorithm for the factoring and discrete logarithm problems upon which they are based [Sho97]. In addition, new features of quantum adversaries are difficult and subtle to deal with, especially in the setting of cryptographic protocols. In fact a lot of classical security analyses become inapplicable or even fail completely in the presence of quantum adversaries [CSST11, Wat09, HSS11].

Pertaining to hash functions, a quantum adversary naturally has the ability to implement the hash function as a quantum circuit and hence evaluate it in quantum *superposition*. In other words, if H is treated as a black box, it is reasonable to give a quantum adversary superposition-access to H by querying H on any set of input strings in the form: $\sum_x \alpha_x |x, 0\rangle \mapsto \sum_x \alpha_x |x, H(x)\rangle$. Although this does not allow the adversary to learn the entire set of hash values in one query, an immediate difficulty one may notice is the failure of the "lazy sampling" trick, where one can simulate a random function by sampling random responses on-the-fly. In fact, a lot of effort has been devoted to extending the results and useful techniques in the classical random oracle model to the quantum setting (formalized as the quantum random oracle model) [BDF$^+$11, Zha15b, ARU14]. Notably, Zhandry [Zha15a] shows that $O(N^{1/3})$ quantum queries are enough to find a collision in a uniformly random function. The same amount $\Omega(N^{1/3})$ is also proven to be necessary. This establishes the generic security of hash functions when treated as truly random black-box functions. But as classical attacks illustrated, the assumption of uniform randomness is sometimes too optimistic and risky. Any non-uniformity may well be exploited in practice by quantum adversaries. Such concerns are becoming more pressing due to recent breakthroughs in physical realizations of quantum computers[1]. Optimized architectures are also reducing the cost of implementing quantum algorithms (see for example an estimation of Grover's quantum search algorithm [ADMG$^+$16]).

This motivates the question we study in this work: *what is the complexity of finding a collision in a* ***non-uniform*** *random function, under quantum attacks in particular*? Specifically we consider a distribution D_k on set Y which has min-entropy k, i.e., the most likely element occurs with probability at most 2^{-k}. We want to find a collision in a function $H : X \to Y$ where for each $x \in X$, $H(x)$ is drawn independently according to D_k. We call it a rand-min-k function hereafter. Note that if D_k is uniform over Y (hence $|Y| = 2^k$), this becomes the standard uniformly random function. Given H as a black-box, we are interested in the number of queries needed by a quantum algorithm to find a collision in H. As a result, this will establish the generic security of hash functions under a *relaxed condition* where the outputs of a hash function are drawn from a distribution of min-entropy

[1]See recent announcements of IBM Q 16-qubit chip and prototype 17-qubit commercial quantum chips [IBM17], as well as Google's testing of a 22-qubit chip and projection of 50-qubit chip [Mar17].

k rather than uniformly random. This condition in some sense matches our intuition of a good hash function more closely. Roughly speaking, a designer will only need to make sure that there is no single value $y \in Y$ that has a large set of preimages (i.e., $f^{-1}(y) := \{x \in X : f(x) = y\}$ with $|f^{-1}(y)| \leq 2^k$), which could be more realistic and easier to check. In contrast, modeling a hash function as a uniformly random function would require certain *regularity* such that the preimage set of every codomain element has the same size on average, which may be difficult to justify and test in practice. We also note that a concrete application of collision finding in rand-min-k functions appears in the famous Fujisaki-Okamoto transformation [FO13], whose quantum security has been studied in [TU16].

Classically, it is not difficult to derive a variation of the birthday bounds, which gives $\Theta(2^{k/2})$ as the query complexity in typical cases. In the quantum setting, Targhi et al. [TTU16] proves that $\Omega(2^{k/9})$ queries are necessary for any quantum algorithm to find a collision with constant probability. Notice that compared to the tight bound $2^{k/3}$ in the uniform case, the bound is unlikely to be optimal and the gap seems significant. In addition, no quantum algorithms are described or analyzed formally, although it is natural to conjecture that the $2^{k/3}$ upper bound for uniform case should carry through. Overall, our understanding of finding a collision in non-uniform random functions is far from satisfying as far as quantum attacks are concerned.

1.1 Our contributions

In this work, we provide a complete characterization of the complexity of finding collisions in a rand-min-k function when it is given as an oracle to a quantum algorithm. In all cases, we are able to prove matching upper and lower bounds. The results are summarized in Table 1.

D_k	M, N, k **settings**	**Upper bound**	**Lower bound**
All	$M = o(\beta^{1/2})$ (inj. by Lemma 2.11)	∞	∞
All	$M = \Omega(\beta^{1/2})$	$\beta^{1/3}$ (Thm. 4.2)	$2^{k/3}$ (Cor. 3.4)
flat-k	$M = \Omega(2^{k/2})$	$2^{k/3}$ (Thm. 4.2)	$2^{k/3}$ (Cor. 3.4)
δ-min-k	$M = \Omega(N^{1/2})$, $2^k \leq N < 2^{3k/2}$	$N^{1/3}$ (Thm. 4.2)	$N^{1/3}$ (Cor. 3.11)
	$M = \Omega(N^{1/2})$, $2^{3k/2} \leq N < 2^{2k}$	$2^{k/2}$ (Thm. 4.4)	$2^{k/2}$ (Cor. 3.11)
	$M = \Omega(2^k)$, $N \geq 2^{2k}$	$2^{k/2}$ (Thm. 4.4)	$2^{k/2}$ (Cor. 3.11, also Cor. B.3)

Table 1: Summary of quantum collision finding in rand-min-k functions. $\beta := \frac{1}{\Pr[x=y:x,y\leftarrow D]}$ is the collision variable, which equals 2^k for flat-distributions (i.e., uniform on a subset of size 2^k), and lies in $[2^k, 2^{2k}]$ for δ-min-k distributions (i.e., peak at one element, and uniform elsewhere), as well as for general δ-min-k distributions.

To highlight a few, a simple special case is the flat distribution which is uniform on a subset of size 2^k. In this case, not surprisingly, the same bound $2^{k/3}$ for the uniform random function holds. Another special case, which represents the hardest instances, concerns the δ-min-k distributions, where there is a mode element with probability mass 2^{-k} and the remaining probability mass is distributedly uniformly throughout all of the codomain. Here we basically show that $2^{k/2}$ queries are both sufficient and necessary. In general, the complexity is characterized by the *collision variable* $\beta(D)$ for a distribution D, which is the reciprocal of the probability that two independent samples from D collide. We prove a generic upper bound $\beta^{1/3}$, and a lower bound $2^{k/3}$ (corresponding to the case of flat distributions). For comparison, one can show that classically $\Theta(\beta^{1/2})$ precisely depicts the hardness of finding a collision, which we also derive for completeness by adapting the

birthday bound to the min-k setting.
Technical overview. For the generic lower bound $2^{k/3}$, we follow the natural idea of reducing from collision finding in uniform random functions (Theorem 3.3). We describe a sequence of reductions showing that finding a collision in uniform random functions of codomain size 2^k reduces to that in flat distributions, and then to general min-k distributions. Therefore the $2^{k/3}$ lower bound follows. To illustrate our new approach, we review that in [TTU16], they extract close-to uniform bits from the output of a rand-min-k function f by composing f with a universal hash function h. Note that a collision in f is also a collision in $h \circ f$. In addition, $h \circ f$ can be shown to be quantum indistinguishable from a uniformly random function by a general theorem of Zhandry [Zha12], which relates sample-distinguishability to oracle-distinguishability. Therefore any adversary for rand-min-k can be turned into an adversary for $h \circ f$, which is close to the uniform case. As a result finding a collision must be hard in $h \circ f$. However, the discrepancy between $h \circ f$ and uniform accumulates and gets amplified in the sample-to-oracle lifting step, and this may explain the loss in their lower bound $2^{k/9}$. In contrast, our reductions employ a general randomized *collision-conversion* procedure, which perfectly converts a function f drawn according to one distribution D (e.g., uniform) to a function f' distributed according to another distribution D' (e.g., general D_k). Therefore an adversary for D_k can be turned into an adversary for the uniform case. The only downgrade stems from the unfortunate fact a collision in f' does not always gives a collision in f, but we can show that this bad event only occurs bounded constant probability. This is the main distinction that enables our improvement. Basically, our conversion introduces a conditional probability distribution $p(\cdot|z)$ for each element z in the support of D, under the constraint that $\sum_z p(x|z)$ should agree with the probability of sampling x according to D'. Therefore given a function f, whose output z is drawn according to D, we perform a sub-sampling according to $p(\cdot|z)$ to define the output of f'. Then f' will be the correct distribution according to D'. The conversion may not be time efficient, but it preserves query complexity in the oracle model.

We note that along the same lines, it is possible to demonstrate that finding collision in δ-min-k distributions is the hardest case. In fact, we are able to establish rigorously a strengthened lower bound in this case (Theorem 3.10). Our proof proceeds by first proving a distinguishability result between a random δ-min-k function on codomain of size N and a uniformly random function on the same codomain, and then the lower bound in the uniform case will translate to a lower bound for the δ-min-k case. The exact bounds vary a bit for different relative sizes between N and k.

Establishing upper bounds is relatively easy (Theorem 4.2). We adapt the quantum algorithm of [Zha15a] in the uniform case. Basically we partition the domain of a rand-min-k function f into subsets of proper size, so that when restricting f on every subset, it contains a collision with at least constant probability. Next, we can invoke the collision finding algorithm by Ambainis [Amb07] on each restricted function, and with a few iterations, a collision will be found.

Moreover, we give alternative proofs showing the lower bound for δ-min-k distributions (Theorem B.1) and upper bound for all min-k distributions (Theorem 4.4). They are helpful to provide more insights and explain the bounds intuitively. Specifically, we reduce an average-case search problem, of which the hardness has been studied [HRS16], to finding collision in a δ-min-k random function. On the other hand, when the mode element of a min-k distribution is known, we show that applying Grover's quantum search algorithm almost directly will find a collision within $O(2^{k/2})$ queries. This actually strengthens the upper bound above in some parameter settings.

1.2 Discussion

Collision finding is an important problem in quantum computing, and we mention a few more in the literature. Brassard et al. [BHT97] give a quantum algorithm that finds a collision in any two-to-one function $f : [M] \to [N]$ with $O(N^{1/3})$ quantum queries. Ambainis [Amb07] gives an algorithm based on quantum random walks that finds a collision using $O(M^{2/3})$ queries whenever there is at least on collision in the function. Aaronson and Shi [AS04] and Ambainis [Amb05] give a $\Omega(N^{1/3})$ lower bound for a two-to-one function f with the same domain and co-domain of size N. Yuen [Yue14] proves an $\Omega(N^{1/5}/\text{polylog}N)$ lower bound for finding a collision in a uniformly random function with codomain at least as large as the domain. This is later improved by Zhandry [Zha15a] to $\Theta(N^{1/3})$ and for general domain and codomain as we mentioned earlier.

We stress that typically in quantum computing literature, the lower bounds are proven for the worst-case scenario and with constant success probability. This in particular does not rule out adversaries that succeed with inverse polynomial probability which is usually considered a break of a scheme in cryptography. Hence a more appropriate bound in cryptography would be showing the number of queries needed for achieving any (possibly low) success probability, or equivalently upper bounding the success probability of any adversary with certain number of queries. Our results, as in [Zha15a, TTU16], are proven in the strong sense that is more applicable in cryptographic settings.

Our work leaves many interesting possible directions for future work. One immediate unsatisfying feature of our collision-conversion reductions is that they may take a long time to implement. Can they be made time efficient? We have been mainly concerned with finding one collision; it is interesting to investigate the complexity of finding *multiple* collisions in a non-uniform random function. There are other important properties of hash functions such as preimage resistance and second-preimage resistance, which are both weaker than and implied by collision resistance. Hence, our lower bound results also demonstrate the hardness of finding a preimage and second preimage. But they are not necessarily tight, and finding out the optimal quantum algorithms for solving them is also crucial. Finally, we note that a stronger notion for hash functions called *collapsing* has been proposed which is very useful in the quantum setting [Unr16]. Can we prove that rand-min-k functions are collapsing? Note that a uniform random function is known be collapsing, and more recently it has been shown that the sponge construction in SHA-3 is collapsing (in the quantum random oracle model) [CBHS17, Unr17].

2 Preliminaries

Here we introduce the reader to a few important notations and definitions. We also discuss some basic results concerning the collision probability and birthday bound in min-k distributions.

We consider functions $f : X \to Y$ for some arbitrary X and Y and use Y^X to denote the set of all such functions. The notation $f \leftarrow Y^X$ indicates that f is a function sampled uniformly from Y^X. In general, we consider distributions on the co-domain Y. The notation $f \leftarrow D^X$ indicates that f is a function from X to Y sampled from the distribution of functions induced by sampling each image independently from a distribution D on Y.

Definition 2.1 (Min-Entropy)**.** Let D be a distribution on some set Y. Let $D(y)$ denote the probability mass corresponding to a $y \in Y$ under distribution D. D is said to have min-entropy k if $k = -\log_2(\max_{x \in X}\{D(x)\})$. We refer to a distribution of min-entropy k as a min-k distribution or simply a k-distribution.

Definition 2.2 (Flat-k-Distribution)**.** We call a k-distribution D on set Y with support S a flat-k-distribution, denoted $D_{k,\flat}$, if the cardinality of S is exactly 2^k. It follows that $\forall x \in S$, $D(x) = 2^{-k}$.

Definition 2.3 (δ-k-Distribution)**.** We call a k-distribution D on set Y a δ-k-distribution if there is a unique $m \in Y$ such that $\forall y \in Y$

$$D(y) = \begin{cases} 2^{-k} & \text{if } y = m\,; \\ \frac{1-2^{-k}}{|Y|-1} & \text{otherwise}\,, \end{cases}$$

and may denote such a distribution $D_{k,\delta}$. Notice that m is the mode of D, since D has min-entropy k. The support of D is the entire set Y, and remaining probability mass $1 - 2^{-k}$ is distributed uniformly among all elements in Y other than the mode.

Definition 2.4 (Function of min-entropy k)**.** Let D be a distribution with min-entropy k on some set Y. Let $f : X \to Y$ be a (in general non-uniformly) random function such that the image of each input under f is an independent random sample from set Y according to distribution D. We denote sampling of a function in this way as $f \leftarrow D^X$. We say that f is a function of min-entropy k.

Definition 2.5 (Collision Problem)**.** Let $f : X \to Y$ be a function of min-entropy k, whose images are distributed according to D. A pair of elements $x_1 \in X$ and $x_2 \in X$ such that $x_1 \neq x_2$ and $f(x_1) = f(x_2)$ is called a *collision* in f. We refer to the problem of producing such a pair as the *collision finding problem* or *collision finding problem* in D. Suppose an algorithm $\mathcal{A}$ outputs a collision in f. Then we say that $\mathcal{A}$ has *solved* collision finding in D.

Definition 2.6 (Quantum Oracle Access)**.** Suppose $\mathcal{O}$ is an oracle for some function f. Suppose $\mathcal{A}$ is a quantum algorithm which can submit queries to $\mathcal{O}$ in a quantum superposition (called a *quantum query*) and receive $\mathcal{O}$'s responses in a quantum superposition. Specifically, $\mathcal{A}$ can implement $\mathcal{O}$ as a unitary transformation $\sum \alpha_{x,y,z}|x, y, z\rangle \mapsto \sum \alpha_{x,y,z}|x, y + f(x), z\rangle$. Then we say that $\mathcal{A}$ has *quantum oracle access* to f and denote this as $\mathcal{A}^f$.

2.1 Collision probability and birthday bound

For a discrete probability distribution D on set Y, let $D(y)$ be its probability mass function. The support of D is $\mathrm{Supp}(D) = \{y \in Y : D(y) > 0\}$.

Definition 2.7. The *collision probability* of a probability distribution D is defined to be the probability that two independent samples from D are equal. Namely

$$\mathrm{CP}(D) := \Pr_{y_1,y_2 \leftarrow D}[y_1 = y_2] = \sum_{y \in Y} D(y)^2\,.$$

Define $\beta(D) := \frac{1}{\mathrm{CP}(D)}$ to be the reciprocal of the collision probability. It will be an important variable determining the complexity of collision finding. We characterize $\beta(D)$ for min-k distributions.

Lemma 2.8. *Let D_k be a min-k distribution on set Y with $|Y| = N$, $k \geq 1$ and $N \geq 2^k$.*

- *If D_k is a flat-k distribution, $\beta(D) = 2^k$.*
- *For δ-min-k distribution $D_{k,\delta}$, $\beta(D_{k,\delta}) \approx \begin{cases} N & \text{if } N < 2^{2k}\,; \\ 2^{2k} & \text{if } N \geq 2^{2k}\,. \end{cases}$*
- *For a general min-k distribution D_k, $\beta(D_k) \in [2^k, 2^{2k}]$.*

Proof. For flat-k D_k, $D_k(y) = \frac{1}{2^k}$ for $y \in Y' \subseteq Y$ with $|Y'| = 2^k$. Hence $\beta(D_k) = \frac{1}{\sum_{y \in Y'} 2^{-2k}} = 2^k$.

For $D_{k,\delta}$ distribution

$$\beta(D_{k,\delta}) = \frac{1}{\mathrm{CP}(D_{k,\delta})} = \frac{1}{2^{-2k} + \frac{(1-2^{-k})^2}{N-1}} = \frac{2^{2k}(N-1)}{N - 2 \cdot 2^k + 2^{2k}} \approx \frac{2^{2k} \cdot N}{2^{2k} + N} .$$

By considering different range of N, we obtain the estimation for $\beta(D_{k,\delta})$ in the lemma.

For general D_k, it is easy to observe that $2^{-2k} \le \mathrm{CP}(D_k) \le 2^{-k}$ and hence $\beta(D_k) \in [2^k, 2^{2k}]$. □

A few useful lemmas by Wiener [Wie05]. Let D be a discrete probability distribution. Let R_D be the random variable denoting the number of i.i.d. samples from D when a collision appears for the first time.

Lemma 2.9. *([Wie05, Theorem 3]) Let $q \ge 1$ be an integer and $\gamma_q := \frac{q-1}{\sqrt{\beta(D)}}$*

$$\Pr(R_D > q) \le e^{-\gamma_q}(1 + \gamma_q) .$$

An immediately corollary follows.

Corollary 2.10. *There is a constant $c > 2$ such that if $q \ge c\sqrt{\beta(D)}$, and let $y_1, \ldots, y_q$ be i.i.d. samples from D. Let $\textsc{Col}^q(D)$ be the event that $y_i = y_j$ for some $i, j \in [q]$. Then $\Pr(\textsc{Col}^q(D)) \ge 2/3$.*

Proof. Let E be the event that $y_i = y_j$ for some $i, j \in [q]$. Then

$$\Pr[E] \ge 1 - \Pr[X_D > q] \ge 1 - e^{-\gamma_q}(1 + \gamma_q) \ge 2/3 ,$$

when $q \ge c\sqrt{\beta(D)}$ because $\frac{1+\gamma_q}{e^{\gamma_q}} < 0.3$ whenever $\gamma_q = \frac{q-1}{\sqrt{\beta(D)}} > 2$. □

We can also derive an upper bound on $\Pr[\textsc{Col}^q(D)]$ by standard approach.

Lemma 2.11. $\Pr[\textsc{Col}^q(D)] \le \frac{q^2}{\beta(D)}$.

Proof. For any pair $i \in [q]$ and $j \in [q]$, Let $\textsc{Col}_{ij}$ be the event that $y_i = y_j$. Then $\Pr[\textsc{Col}_{ij}] = \mathrm{CP}(D)$. Therefore by union bound, we have

$$\Pr[\textsc{Col}^q(D)] = \Pr[\cup_{i,j \in [q]} \textsc{Col}_{ij}] \le \binom{q}{2} \cdot \mathrm{CP}(D) \le \frac{q^2}{\beta(D)} .$$

□

As a result, when $q = o(\sqrt{\beta(D)})$, essentially no collision will occur. Namely q needs to be $\Omega(\sqrt{\beta(D)})$ to see a collision, which we have seen is also sufficient by Corollary 2.10. This is summarized below as a birthday bound for general distributions.

Theorem 2.12. $\Theta(\sqrt{\beta(D)})$ *samples according to D are sufficient and necessary to produce a collision with constant probability for any classical algorithms.*

3 Lower bounds: finding a collision is difficult

To prove our query complexity lower bounds, we make use of proof by reduction. First we recall the hardness result for uniform distributions shown by Zhandry [Zha15a].

Lemma 3.1 ([Zha15a] Theorem 3.1)**.** *There is a universal constant C such that the following holds. Let $f : [M] \to [N]$ be a uniformly random function. Then any algorithm making q quantum queries to f outputs a collision in f with probability at most $C(q+1)^3/N$.*

We show that collision finding in any min-k distribution is at least as difficult as collision finding in a uniform distribution on a set of size 2^k. We begin by demonstrating a reduction of collision finding in a uniform distribution to collision finding in a flat-k distribution. Then we show a reduction of collision finding in a flat-k distribution to collision finding in a general k-distribution. These reductions show that any algorithm which is not able to solve collision finding in a uniform distribution on a set of size 2^k is also not able to solve collision finding in any k-distribution. Namely we prove the following. Note that we write all of the constant factors in the probabilities as C, even though they will not all take the same numerical value, in recognition that they are not interesting for the study of asymptotic query complexity.

Theorem 3.2. *Let $f_{flat} \leftarrow D_{k,\flat}^X$ be a random function whose outputs are chosen according to a flat-k-distribution $D_{k,\flat}$. Then any algorithm making q queries to f_{flat} outputs a collision with probability at most $C(q+1)^3/2^k$, for some constant C.*

Theorem 3.3. *Let $f_D \leftarrow D^X$ be a random function whose outputs are chosen according to a distribution D of min-entropy k. Then any algorithm making q queries to f_D outputs a collision with probability at most $C(q+1)^3/2^k$, for some constant C.*

Corollary 3.4. *Any quantum algorithm needs at least $\Omega(2^{k/3})$ queries to find a collision with constant probability in a random function $f_D \leftarrow D^X$ whose outputs are chosen according to a distribution D of min-entropy k.*

Each of the reduction proofs describe an algorithm (which we may refer to as 'the reduction') attempting to find a collision in a random function f to which it has oracle access. The algorithm will run as a subroutine another algorithm which is capable of finding a collision in another random function g when given oracle access to g. Therefore, the proofs must account for how the reduction algorithm satisfies two requirements:

a. it interacts with the subroutine algorithm in such a way that, from the perspective of the subroutine, the subroutine is interacting with an oracle for the random function g; and

b. the collision in g returned by the subroutine can be converted into a collision in f with sufficiently high probability.

The way the reduction satisfies requirement b by necessity depends on the way it satisfied requirement a, since converting a collision in g into a collision in f necessarily depends on how the values of g are generated from the values of f. Hence a single procedure should be used by the reduction to satisfy both requirements. We will refer to such a procedure as a *collision-conversion procedure*. Intuitively, a collision-conversion procedure is an algorithm that in some way processes the responses of the an oracle f so that its output (perfectly) simulates an oracle for g, and then inverts that process to convert a collision in g into a collision in f. A describes a conversion procedure (flat distribution to arbitrary min-k distribution) with visual aides. The formal definition follows.

Definition 3.5 ($p^{D \to D'}$ collision-conversion procedure **).** Let $\mathcal{C}$ be an algorithm that is given access to an oracle f whose responses are distributed according some distribution D. For each unique query submitted to it, $\mathcal{C}$ returns an independent random sample drawn according to some distribution D'. Furthermore, when given as input a pair (x_1, x_2), $\mathcal{C}$ returns either some collision in f, (x'_1, x'_2), in which case we say $\mathcal{C}$ *succeeds*, or returns $\perp$ otherwise. Then we call $\mathcal{C}$ a collision-conversion procedure from D to D'. We say $\mathcal{C}$ *succeeds with probability at least* p if the conditional probability that $\mathcal{C}$ succeeds, given that $\mathcal{C}$'s responses to queries x_1 and x_2 were equal, is at least p, regardless of the values x_1 and x_2.

We assume the following about the conversion procedure and hence the reductions:

- The reduction has a full description of the distribution in which the subroutine can solve the collision problem, since it is possible that an adversary will have some or even all information relating to the image distribution of the function it is attacking. Formally, suppose the subroutine solves the collision problem in D, a distribution on Y. We assume that the reduction has perfect knowledge of the probability mass function $D(y)$ for all $y \in Y$. .

- The reduction has access to a "sufficiently large" amount of randomness. While we will be more explicit about this later, the intuition is that the reduction will need to sample from some distribution D. As we will not put any restriction on this distribution (other than having min-entropy k), it is not clear how much randomness is needed to sample from it. However it is clear that a large but finite amount of randomness is sufficient to sample arbitrarily close to this distribution.

- The reduction is not computationally limited. It is conceivable that an adversary with advance knowledge of the function it is attacking may perform an exponential amount of precomputation to speed up collision finding.

We give an simple example below that converts from uniform to a flat distribution.

Algorithm 1 Collision-conversion: uniform to flat

```
Input: Let $f : X \to Y$, where $|Y| = 2^k$, be a uniformly random function.
1: Upon initialization, prepare any injective mapping $g : S \to Y$. This can be done by randomly
   sampling from $Y$, without replacement, for each element in $S$.
2: For each query $x$, forward the query to the oracle $f$ which $\mathcal{C}$ has access to, and respond with
   $g^{-1}(f(x))$.
3: When a pair $(x_1, x_2)$ is received, output $(x_1, x_2)$.
```

This is indeed a collision-conversion procedure from a flat-k-distribution to a uniform distribution on a set of size 2^k. Since g is a permutation, g^{-1} will map a uniform element of Y to a uniform element of S. Since $f(x)$ is, by the definition of a uniformly random function, a uniform element of Y, $g^{-1}(f(x))$ is a uniform element of S. This means that the distribution of $g^{-1}(f(x))$ is the flat-k-distribution on support S. Therefore, when queried, $\mathcal{C}$ returns an independent random sample from the flat-k-distribution, as required. If the responses of $\mathcal{C}$ to the queries x_1 and x_2 are equal, then $g^{-1}(f(x_1)) = g^{-1}(f(x_2))$. Since g is a permutation, this implies $f(x_1) = f(x_2)$. Hence the conditional probability that $f(x_1) = f(x_2)$ given that the responses of $\mathcal{C}$ to the queries x_1 and x_2 are equal is 1, so $\mathcal{C}$ is a collision-conversion procedure from a flat-k-distribution to a uniform distribution on a set of size 2^k which succeeds with probability 1.

We describe the procedures as fundamentally *classical* algorithms. This is especially visible in the fact that individual queries are processed sequentially by oracles and the algorithms that interact with them. The definition of a collision-conversion procedure above is also implicitly classical in this way. Nonetheless, the reductions below can easily be generalized to the quantum setting because nowhere do the reductions utilize knowledge of the characteristics of the set of prior queries and responses. This is an important feature because quantum algorithms can query an oracle on a superposition of inputs and receive the responses in a superposition. Hence any sort of dynamic behavior in which an oracle's response depends on some analysis of prior behavior is not possible in the quantum setting. The fact that the reductions we present below can be (with the exception of the final step) parallelized over the set of all possible queries indicates their compatibility with quantum computing.

We begin by showing some reusable general results that will allow us to quickly construct reductions and extend query complexity lower bounds by simply demonstrating the existence of a satisfactory collision-conversion procedure for use in each reduction.

Lemma 3.6. *Suppose there exists an algorithm $\mathcal{A}$ which solves collision finding in a particular distribution $D_{\mathcal{A}}$ with probability at least $P_{\mathcal{A}}$, using q queries to an oracle whose responses are distributed according to $D_{\mathcal{A}}$. Suppose there exists a collision-conversion procedure from $D_{\mathcal{A}}$ to a distribution $D_{\mathcal{R}}$ that succeeds with probability at least p . Then there exists an algorithm which solves collision finding in $D_{\mathcal{R}}$ with probability at least $p \cdot P_{\mathcal{A}}$ using q queries to an oracle whose responses are distributed according to $D_{\mathcal{R}}$.*

Proof. We describe such an algorithm, which we call $\mathcal{R}$ for reduction. Upon initialization, run $\mathcal{A}$ and the collision-conversion procedure, which we call $\mathcal{C}$, simultaneously. Let $f_{\mathcal{R}}$ denote the oracle whose responses are distributed according to distribution $D_{\mathcal{R}}$. Grant $\mathcal{C}$ access to the oracle $f_{\mathcal{R}}$ and access to all other resources available to the reduction. For each of $\mathcal{A}$'s queries, forward the query to $\mathcal{C}$, and return $\mathcal{C}$'s response back to $\mathcal{A}$. When $\mathcal{A}$ outputs a collision candidate, forward the collision candidate to $\mathcal{C}$. Return the output of $\mathcal{C}$.

That this algorithm works is clear. By the definition of a collision-conversion procedure, the algorithm $\mathcal{A}$ interacts with an oracle (call it $f_{\mathcal{A}}$) whose responses are distributed according to $D_{\mathcal{A}}$. Hence $\mathcal{A}$ runs as usual, producing a collision in $f_{\mathcal{A}}$ with probability $P_{\mathcal{A}}$. Again, by the definition of a collision-conversion procedure, $\mathcal{C}$ will return a collision in $f_{\mathcal{R}}$ at least half of the time that it is given collision in $f_{\mathcal{A}}$. It follows that, at a minimum, the probability that $\mathcal{R}$ outputs a collision in $f_{\mathcal{R}}$ is $p \cdot P_{\mathcal{A}}$. Observe also that $\mathcal{R}$ uses exactly one query for every query used by $\mathcal{A}$. Hence lemma 3.6 follows. □

Corollary 3.7. *Suppose there is some upper bound $G(q)$ on the probability that any algorithm making q queries to an oracle $f_{\mathcal{R}}$, whose responses are distributed according to a distribution $D_{\mathcal{R}}$, finds a collision in $f_{\mathcal{R}}$. Suppose there exists a collision-conversion procedure from a distribution $D_{\mathcal{A}}$ to $D_{\mathcal{R}}$ that succeeds with probability at least p. Then $G(q)/p$ is an upper bound on the probability that any algorithm making q queries to an oracle $f_{\mathcal{A}}$, whose responses are distributed according to $D_{\mathcal{A}}$, finds a collision in $f_{\mathcal{A}}$.*

Proof. This follows directly from contraposition of lemma 3.6. The negation of the consequent of lemma 3.6 is that the probability that any algorithm solves collision finding in $D_{\mathcal{R}}$ using q queries to an oracle whose responses are distributed according to $D_{\mathcal{R}}$ must be less than $p \cdot P_{\mathcal{A}}$. The negation of the antecedent is that the probability any algorithm solves collision finding in $D_{\mathcal{A}}$ using q queries to an oracle whose responses are distributed according to $D_{\mathcal{A}}$ must be less than $P_{\mathcal{A}}$. Expressing this contrapositive with $G(q)/p$ taking the role of $P_{\mathcal{A}}$ gives the above corollary. □

Proof of Theorem 3.2 & Theorem 3.3. Both are proven the same way by combining Corollary 3.7 with Lemma 3.1. For Theorem 3.2, we replace N with 2^k, and use the collision-conversion procedure $\mathcal{C}$ described in Algorithm 1.

Likewise Theorem 3.3 follows by combining Corollary 3.7 and Theorem 3.2. All we need is a conversion procedure $\mathcal{C}$. We give the complete description (Algorithm 4) and explanation of why $\mathcal{C}$ qualifies as a collision-conversion procedure from a flat-k-distribution to an arbitrary k-distribution in appendix A. $\square$

Finally we remark that it is possible to show a reduction of collision finding in an arbitrary k-distribution to collision finding in a δ-k-distribution. This is interesting because it affirms that the δ-k-distribution case is the most difficult out of all k-distributions. The proof is easy to find by mimicking the proof of theorem 3.2 but replacing all references of 2^{-k} as the probability of sampling each element from the flat distribution with a general probability $D(x)$, and replacing the general distribution D with a δ-k-distribution D_δ. This works in the case that no elements in the support of D are associated with a probability mass less than $1/N$. One has to employ the idea of computational indistinguishability to extend this result to the case that there are some elements associated with smaller probability mass than $1/N$.

3.1 Lower bound for δ-min-k distributions

As noted above, δ-min-k distributions represent the hardest instances for collision finding. In this section we give further evidence and establish an even stronger bound for finding collision in the δ-k-distribution case (Theorem 3.10). We first prove the following theorem.

Theorem 3.8. *For any q-query algorithm A,*

$$\left| \Pr_{f \leftarrow {D_{k,\delta}}^X} (A^f(\cdot) = 1) - \Pr_{f \leftarrow Y^X} (A^f(\cdot) = 1) \right| \leq 8q^2/2^k + 1/N\,.$$

Lemma 3.9. *[Zha12, Theorem 7.2] Fix q, and let F_λ be a family of distributions on Y^X indexed by $\lambda \in [0,1]$. Suppose there is an integer d such that for every $2q$ pairs $(x_i, y_i) \in X \times Y$, the function $p_\lambda := \Pr_{f \leftarrow F_\lambda}(f(x_i) = y_i, \forall i \in \{1, \ldots, 2q\})$ is a polynomial of degree at most d in λ. Then any quantum algorithm A making q queries can only distinguish F_λ from F_0 with probability at most $2\lambda d^2$.*

Proof of Theorem 3.8. For every $\lambda \in [0,1]$, define D_λ on Y such that there is an element $m \in Y$ with $D_\lambda(m) = \lambda$ and for any $y \neq m$ $D_\lambda(y) = \frac{1-\lambda}{|Y|-1}$. Denote $\hat{Y} := Y \backslash \{m\}$ Then Define a family of distributions F_λ on Y^X where $F_\lambda := {D_\lambda}^X$, i.e., the output of each input is chosen independently according to D_λ.

Consider any sequence $\{(x_i, y_i)\}_{i=1}^{2q}$ $p_\lambda := \Pr_{f \leftarrow F_\lambda}(f(x_i) = y_i, \forall i \in \{1, \ldots, 2q\}) = \lambda^t (\frac{1-\lambda}{|Y|-1})^{2q-t}$, where t is the number of occurrences of m in $\{y_i\}_{i=1}^{2q}$. Clearly p_λ is a polynomial in λ with degree at most $2q$.

Notice that $F_{2^{-k}}$ is exactly δ-min-k distribution $D_{k,\delta}$, and F_0 is uniformly random distribution $Y \backslash \{m\}^X$. Therefore applying Lemma 3.9, we have that

$$\left| \Pr_{f \leftarrow {D_{k,\delta}}^X} (A^f(\cdot) = 1) - \Pr_{f \leftarrow \hat{Y}^X} (A^f(\cdot) = 1) \right| \leq 2(2q)^2 \cdot 2^{-k} = 8q^2/2^k\,.$$

Notice that the statistical distance between Y^X and $\hat{Y}^X$ is $\frac{1}{2}\left((N-1)(\frac{1}{N-1} - \frac{1}{N}) + (\frac{1}{N} - 0)\right) = 1/N$. Hence $\left|\Pr_{f \leftarrow {D_{k,\delta}}^X}(A^f(\cdot) = 1) - \Pr_{f \leftarrow Y^X}(A^f(\cdot) = 1)\right| \leq 8q^2/2^k + 1/N$. $\square$

This allows us to show stronger complexity for finding collision in a δ-min-k random function.

Theorem 3.10. *For any q-query algorithm A,*

$$\Pr_{f \leftarrow D_{k,\delta}{}^X}[f(x) = f(x') : (x, x') \leftarrow A^f(\cdot)] \le O\left(\frac{(q+2)^2}{2^k} + \frac{(q+2)^3}{N}\right).$$

Proof. This follows from a simple reduction. Suppose that there is an A with $\Pr_{f \leftarrow X^{D_{k,\delta}}}(f(x) = f(x') : (x, x') \leftarrow A^f(\cdot)) = \varepsilon$ using q queries. Then construct A' which on input oracle f, runs A and receives (x, x') from A. A' then output 1 iff. $f(x) = f(x')$. By definition, we have that $\Pr_{f \leftarrow D_{k,\delta}{}^X}(A'^f(\cdot) = 1) = \varepsilon$. Meanwhile, note that A' makes $q + 2$ queries. Therefore by Zhandry's lower bound on finding collision in uniform random function (Lemma 3.1), we know that $\Pr_{f \leftarrow Y^X}(A'^f(\cdot) = 1) \le O(\frac{(q+3)^3}{N})$. Then Theorem 3.8 implies that

$$\varepsilon \le O(\frac{(q+3)^3}{N}) + 8(q+2)^2/2^k + 1/N = O(\frac{(q+2)^2}{2^k} + \frac{(q+3)^3}{N}).$$

□

Corollary 3.11. *Any quantum algorithm needs $\min\{2^{k/2}, N^{1/3}\}$ queries to find a collision with constant probability. Specifically we need $\Omega(N^{1/3})$ if $2^k \le N < 2^{\frac{3k}{2}}$, and $\Omega(2^{k/2})$ when $N \ge 2^{\frac{3k}{2}}$.*

We give an alternative proof in Section B based on a reduction from an average version of a search problem which is hard to solve from the literature. This may serve as an intuitive explanation of the hardness of non-uniform collision finding. It also connects to the quantum algorithm we develop in Sect. 4.1 based on Grover's search algorithm. We describe it in the case that the domain X is much smaller than the codomain.

4 Upper bounds: (optimal) quantum algorithms

Lemma 4.1. *([Amb07, Theorem 3]) Let $f : X' \to Y$ be a function that has at least one collision. Then there is a quantum algorithm* ColF *making $O(|X'|^{2/3})$ quantum queries to f that finds the collision with constant bounded error.*

We derive a generic upper bound for finding collision in any min-k random functions. We adapt Ambainis's algorithm and describe a quantum algorithm NU-ColF below.

Algorithm 2 Collision Finding in Non-uniform Function NU-ColF

```
Input: $f \leftarrow D_k{}^X$ as an oracle. Let $s, t$ be parameters to be specified later.
Output: Collision $(x, x')$ or $\perp$.
1: Divide $X$ in to subsets $X_i$ of equal size (ignoring the boundary case) $|X_i| = s$.
2: Construct $f_i : X_i \to Y$ as the restriction of $f$ on $X_i$.
3: For $i = 1, \ldots, t$, Run Ambainis's algorithm ColF on $f_i$, and get candidate collision $x_i$ and $x_i'$. if
   $f(x_i) = f(x_i')$, output $(x_i, x_i')$ and abort.
4: Output $\perp$.
```

Since f is generated according to the min-k distribution, when restricting to any subset X_i, we can think of drawing each function value independently from D_k. Namely $f_i \sim D_k{}^{X_i}$ holds for all i. Therefore, by Lemma 2.10, we have that when $s \ge c\sqrt{\beta(D)}$ for some $c > 2$, f_i contains a collision with constant probability. If that is the case, Ambainis's algorithm will find a collision with constant probability using $O(|X_i|^{2/3}) = O(\beta(D)^{1/3})$. We only need to repeat $t = O(k)$ times to succeed except with error negligible in k.

Theorem 4.2. *Assume Let $\beta := \beta(D_k)$. Let X be a set with $|X| = M = \Omega(\sqrt{\beta})$. There is a quantum algorithm* NU-ColF *that finds a collision in $f \leftarrow X^{D_k}$ within $O(\beta^{1/3})$ queries with constant probability. Moreover with $O(k\beta^{1/3})$ queries the algorithm succeeds except with probability negligible in k.*

More specifically, by our characterization of $\beta(D_k)$ in Lemma 2.8,

- flat-k: $O(\beta^{1/3}) = O(2^{k/3})$ and it is tight (when $M = \Omega(2^{k/2})$).
- δ-min-k: $O(\beta^{1/3}) = \begin{cases} O(N^{1/3}) & 2^k \leq N < 2^{2k}, \text{ tight when } N \leq 2^{3k/2} \\ O(2^{\frac{2k}{3}}) & N \geq 2^{2k} \end{cases}$

Note that our algorithm NU-ColF is generic, and needs no additional information about D_k.

4.1 Quantum algorithm for min-k distribution with a mode known

We state a version of Grover's algorithm [Gro96, BBHT96] in a universe of multiple marked items.

Lemma 4.3. *Let $f : X \to \{0,1\}$ be an oracle function and let $Z_f = |\{x \in X : f(x) = 1\}|$. Then there is a quantum algorithm* QSEARCH *using q queries that finds an $x \in X$ such that $f(x) = 1$ with success probability $\Omega(q^2 \frac{Z_f}{|X|})$.*

Algorithm 3 Collision Finding in Non-uniform Function **with a mode known** NU-ColF-Mode

```
Input: f ← D_k^X as an oracle. A mode element m of D_k.
Output: Collision (x, x') or ⊥.
1: Run Grover's algorithm QSEARCH on f to find x with f(x) = m.
2: Run Grover's algorithm QSEARCH on f to find x' with f(x) = m and x' ≠ x.
3: Output ⊥ if any invocations of the Grover's algorithm fails. Otherwise output (x, x').
```

Theorem 4.4. *NU-ColF-Mode finds a collision using $O(2^{k/2})$ queries with constant probability.*

Proof. Let $Z_f := |f^{-1}(m)|$. Let p_f be the probability that f is chosen, when drawn from ${D_k}^X$. Since we invoke QSEARCH twice, we find (x, x') with probability $\Omega\left((\frac{q^2 Z_f}{|X|})^2\right)$. Then the success probability of the algorithm NU-ColF-Mode is

$$\sum_f p_f \Omega\left(\frac{q^4}{M^2} Z_f^2\right) = \Omega\left(\frac{q^4}{M^2} \sum_f p_f Z_f^2\right) = \Omega(\frac{q^4}{M^2}\mathbb{E}[Z_f^2]) \,.$$

To compute $\mathbb{E}[Z_f^2]$, we define for every $x \in X$ an indicator variable $Z_x = \begin{cases} 1 & \text{if } f(x) = m; \\ 0 & \text{otherwise.} \end{cases}$, where $f \leftarrow {D_k}^X$, and clearly $Z_f = \sum_{x \in X} Z_x$. Since each output of x is drawn independently according to $D_{k,\delta}$, $\mathbb{E}[Z_x] = \varepsilon := 2^{-k}$ for all x, it follows that $\mathbb{E}[Z_x] = \mathbb{E}[Z_x^2] = \varepsilon$, and $\mathbb{E}[Z_x \cdot Z_{x'}] = \mathbb{E}[E_x] \cdot \mathbb{E}[E_{x'}] = \varepsilon^2$ for any $x \neq x'$ by independence. Therefore

$$\mathbb{E}[Z_f^2] = \sum_x \mathbb{E}[Z_x^2] + \sum_{x \neq x'} \mathbb{E}[Z_x Z_{x'}] = \Omega(M^2 \varepsilon^2) \,.$$

Hence the algorithm succeeds with probability $\Omega(q^4 \varepsilon^2) = \Omega((\frac{q^2}{2^k})^2)$. As a result, with $q = O(2^{k/2})$ many queries, we find a collision with constant probability. □

Remark 1. Note that we still need $M = \Omega(\sqrt{\beta(D)})$ to ensure existence of collisions. When $N \geq 2^{3k/2}$, Theorem 4.4 gives a better bound $(2^{k/2})$ than Theorem 4.2 ($N^{1/3}$ when $2^{3k/2} \leq N < 2^{2k}$ and $2^{2k/3}$ when $N \geq 2^{2k}$).

References

[ADMG$^+$16] Matthew Amy, Olivia Di Matteo, Vlad Gheorghiu, Michele Mosca, Alex Parent, and John Schanck. Estimating the cost of generic quantum pre-image attacks on SHA-2 and SHA-3. *arXiv preprint arXiv:1603.09383*, 2016.

[Amb05] Andris Ambainis. Polynomial degree and lower bounds in quantum complexity: Collision and element distinctness with small range. *Theory of Computing*, 1(3):37–46, 2005.

[Amb07] Andris Ambainis. Quantum walk algorithm for element distinctness. *SIAM Journal on Computing*, 37(1):210–239, 2007. Preliminary version in FOCS 2004. Available at arXiv:quant-ph/0311001.

[ARU14] Andris Ambainis, Ansis Rosmanis, and Dominique Unruh. Quantum attacks on classical proof systems (the hardness of quantum rewinding). In *FOCS 2014*, pages 474–483. IEEE, October 2014. Preprint on IACR ePrint 2014/296.

[AS04] Scott Aaronson and Yaoyun Shi. Quantum lower bounds for the collision and the element distinctness problems. *Journal of the ACM (JACM)*, 51(4):595–605, 2004.

[BBHT96] Michel Boyer, Gilles Brassard, Peter Høyer, and Alain Tapp. Tight bounds on quantum searching. *arXiv preprint quant-ph/9605034*, 1996.

[BDF$^+$11] Dan Boneh, Özgür Dagdelen, Marc Fischlin, Anja Lehmann, Christian Schaffner, and Mark Zhandry. Random oracles in a quantum world. In *Advances in Cryptology – ASIACRYPT 2011*, pages 41–69. Springer, 2011.

[BDPA07] Guido Bertoni, Joan Daemen, Michaël Peeters, and Gilles Van Assche. The Keccak sponge function family, 2007. `http://keccak.noekeon.org/`.

[BHT97] Gilles Brassard, Peter Hoyer, and Alain Tapp. Quantum algorithm for the collision problem. *arXiv preprint quant-ph/9705002*, 1997.

[BR93] Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *Proceedings of the 1st ACM conference on Computer and Communications Security*, pages 62–73. ACM, 1993.

[BR94] Mihir Bellare and Phillip Rogaway. Optimal asymmetric encryption. In *Advances in Cryptology–EUROCRYPT 1994*, pages 92–111. Springer, 1994.

[CBHS17] Jan Czajkowski, Leon Groot Bruinderink, Andreas Hülsing, and Christian Schaffner. Quantum preimage, 2nd-preimage, and collision resistance of sha3. Cryptology ePrint Archive, Report 2017/302, 2017. `http://eprint.iacr.org/2017/302`.

[CSST11] Claude Crépeau, Louis Salvail, Jean-Raymond Simard, and Alain Tapp. Two provers in isolation. In *Advances in Cryptology–ASIACRYPT 2011*, pages 407–430. Springer, 2011.

[FO13] Eiichiro Fujisaki and Tatsuaki Okamoto. Secure integration of asymmetric and symmetric encryption schemes. *Journal of Cryptology*, 26(1):80–101, 2013. Preliminary version in CRYPTO 1999.

[Gro96] Lov K Grover. A fast quantum mechanical algorithm for database search. In *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, pages 212–219. ACM, 1996.

[HRS16] Andreas Hülsing, Joost Rijneveld, and Fang Song. Mitigating multi-target attacks in hash-based signatures. In *Public-Key Cryptography – PKC 2016*, pages 387–416. Springer, 2016.

[HSS11] Sean Hallgren, Adam Smith, and Fang Song. Classical cryptographic protocols in a quantum world. In *Advances in Cryptology–CRYPTO 2011*, pages 411–428. Springer, 2011.

[IBM17] Ibm q quantum experience, May 17, 2017. https://www.research.ibm.com/ibm-q/.

[JKMR09] Rahul Jain, Alexandra Kolla, Gatis Midrijanis, and Ben W Reichardt. On parallel composition of zero-knowledge proofs with black-box quantum simulators. *Quantum Information & Computation*, 9(5):513–532, 2009. Available at arXiv:quant-ph/0607211.

[Mar17] People of ACM - John Martinis, May 16, 2017. https://www.acm.org/articles/people-of-acm/2017/john-martinis.

[Nak08] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system, 2008. https://bitcoin.org/bitcoin.pdf.

[NIS95] National Institute of Standards and Technology. FIPS 180-1: Secure hash standard, April 1995.

[NIS14] National Institute of Standards and Technology. SHA-3 standard: Permutation-based hash and extendable-output functions, 2014. Available at http://csrc.nist.gov/publications/drafts/fips-202/fips_202_draft.pdf.

[PHC12] Password hashing competition, 2012. https://password-hashing.net/.

[Riv92] Ronald L. Rivest. RFC 1321: The MD5 message-digest algorithm, April 1992. https://www.ietf.org/rfc/rfc1321.txt.

[SBK$^+$17] Marc Stevens, Elie Bursztein, Pierre Karpman, Ange Albertini, and Yarik Markov. The first collision for full SHA-1. Cryptology ePrint Archive, Report 2017/190, 2017. https://shattered.io/.

[Sho97] Peter W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM J. Comput.*, 26(5):1484–1509, 1997.

[TTU16] Ehsan Ebrahimi Targhi, Gelo Noel Tabia, and Dominique Unruh. Quantum collision-resistance of non-uniformly distributed functions. In *International Workshop on Post-Quantum Cryptography*, pages 79–85. Springer, 2016.

[TU16] Ehsan Ebrahimi Targhi and Dominique Unruh. Post-quantum security of the fujisaki-okamoto and oaep transforms. In *Theory of Cryptography Conference*, pages 192–216. Springer, 2016.

[Unr16] Dominique Unruh. Computationally binding quantum commitments. In *Advances in Cryptology–EUROCRYPT 2016*, pages 497–527. Springer, 2016.

[Unr17] Dominique Unruh. Collapsing sponges: Post-quantum security of the sponge construction. Cryptology ePrint Archive, Report 2017/282, 2017. http://eprint.iacr.org/2017/282.

[Wat09] John Watrous. Zero-knowledge against quantum attacks. *SIAM J. Comput.*, 39(1):25–58, 2009. Preliminary version in STOC 2006.

[Wie05] Michael J. Wiener. Bounds on birthday attack times. Cryptology ePrint Archive, Report 2005/318, 2005. http://eprint.iacr.org/2005/318.

[Yue14] Henry Yuen. A quantum lower bound for distinguishing random functions from random permutations. *Quantum Information & Computation*, 14(13-14):1089–1097, 2014.

[Zha12] Mark Zhandry. How to construct quantum random functions. In *FOCS 2012*, pages 679–687. IEEE, 2012. Full version available at http://eprint.iacr.org/2012/182.

[Zha15a] Mark Zhandry. A note on the quantum collision and set equality problems. *Quantum Information and Computation*, 15(7 & 8), 2015.

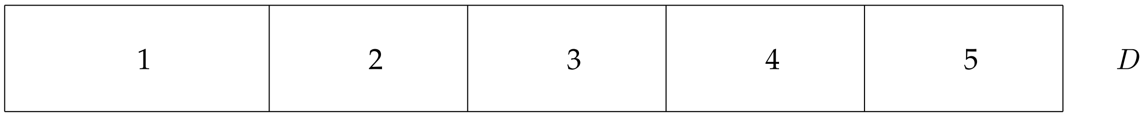
[Zha15b] Mark Zhandry. Secure identity-based encryption in the quantum random oracle model. *International Journal of Quantum Information*, 13(4), 2015. Preliminary version in Crypto 2012. Full version available at http://eprint.iacr.org/2012/076.

A Collision conversion procedure from a flat-k-distribution to an arbitrary k-distribution

Algorithm 4 Collision-conversion: flat to arbitrary min-k

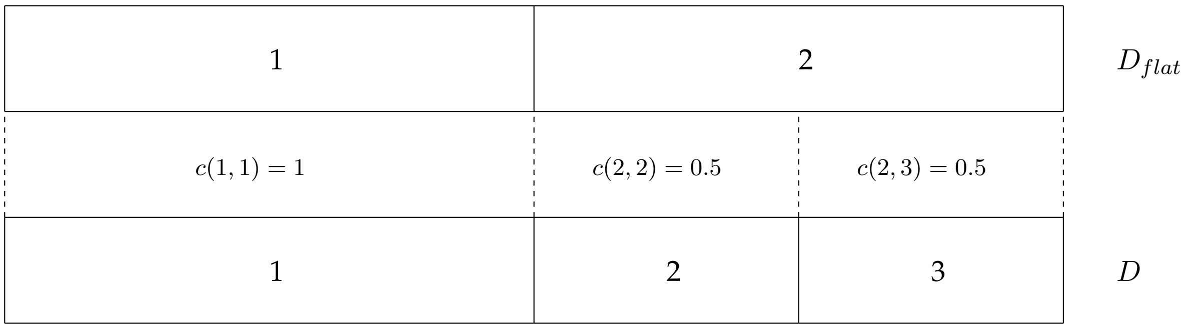
```
Input: Let $D$ be an arbitrary $k$-distribution on support $S$. Let $D_{flat}$ be a flat-$k$-distribution on
    support $S_{flat}$. Let $\mathcal{C}$ denote the collision-conversion procedure. Upon initialization, $\mathcal{C}$ does the
    following:
1: Prepare to store a lookup table for a function $c : S \times S_{flat} \to [0,1]$
2: Sort and label the elements $y_i$ of $S$ in order of decreasing probability mass under distribution
   $D$, so that $D(y_1) = 2^{-k}$ (by the definition of min-entropy, there must be one or more $y_i \in S$
   with $D(y_i) = 2^{-k}$)
3: Arbitrarily label the elements $z_j$ of $S_{flat}$ with index $j = 1, 2, \ldots, 2^k$.
4: Iterate the following over $i = 1, 2, \ldots, |S|$:
     • If $D(y_i) = 2^{-k}$, set $c(y_i, z_i) = 1$. Set $c(y_i, z_j) = 0$ for all $j \neq i$. Continue to the $i$.
     • If $D(y_i) < 2^{-k}$, compute $\sum_{j=1}^{i-1} D(y_j)$ (here $j$ is a dummy-index for the sum and is unre-
       lated to the labeling of the elements of $S_{flat}$) and save the result as the image of $i$ under
       a function $g : [|S|] \to [0,1]$. Check if $g(i)$ is a multiple of $2^{-k}$.
         – If so, set $c(y_i, z_{((g(i)/2^{-k})+1)}) = 2^k D(y_i)$ and $c(y_i, z_j) = 0$ for all $j \neq (g(i)/2^{-k}) + 1$.
         – If not, set $c(y_i, z_{\lceil g(i)/2^{-k}\rceil}) = 2^k \min(\lceil g(i)/2^{-k}\rceil - g(i), D(y_i))$, set $c(y_i, z_{\lceil g(i)/2^{-k}\rceil+1}) =$
           $2^k(D(y_i) - \min(\lceil g(i)/2^{-k}\rceil - g(i), D(y_i)))$, and set $c(y_i, z_j) = 0$ for all remaining
           $z_j \in S_{flat}$
5: For each $z_j \in S_{flat}$, construct the set $W_j$ containing all $y_i \in S$ for which $c(y_i, z_j) \neq 0$. Store the
   set $W_j$ as the image of $z_j$ in a function $b : S_{flat} \to \mathcal{P}(S)$.
6: Now $\mathcal{C}$ enters the query-response phase. Recall that $\mathcal{C}$ has access to an oracle whose responses
   are distributed according to $D_{flat}$; denote this oracle $f_{flat}$. Suppose that $\mathcal{C}$ is queried on some
   $x \in S$. $\mathcal{C}$ responds via the following:
     • Query $f_{flat}$ on $x$. Fix some mapping from an index $t$ to each element in $b(f_{flat}(x))$.
       Compute $m_x(y_t) = c(y_t, f_{flat}(x))$ for each $y_t \in b(f_{flat}(x))$. Sample $r \leftarrow [0,1]$. Note: In
       practice, sampling such an $r$ would of course take an infinite amount of randomness. If
       we only use a finite amount of randomness, then for some distributions we may intro-
       duce some small amount of error. However by increasing this amount of randomness,
       we can make this error arbitrarily small such that any $q$ query adversary $A$ cannot de-
       tect the error. As we do not care about the efficiency of the reduction $\mathcal{C}$, only the query
       complexity, the actual amount of randomness needed for this is not relevant.
     • Iterate through $i = 1, \ldots, |b(f_{flat}(x))|$ until $i$ has a value such that the following condition
       holds: $\sum_{t=1}^{i} m_x(y_t) \geq r$. Return $y_i$.
7: Finally, when $\mathcal{C}$ receives a pair $(x_1, x_2)$, it outputs the same pair $(x_1, x_2)$.
```

To facilitate intuitive understanding of how $\mathcal{C}$ fulfills the requirements of a collision-conversion procedure, we visualize a distribution D as a rectangle divided into disjoint regions, each region representing one element of the support S. The total area of the rectangle is 1, representing the total probability mass in D. For simplicity, consider momentarily a distribution of min-entropy 2 on a set of size 5, whose elements we label with the first 5 positive integers. In the distribution represented below, the 1 element has 25% of the probability mass, while the others share the remaining 75%. Thus 1 is the mode element and its probability mass determines the min-entropy of the distribution.



Denote the oracle simulated by the query-response phase of $\mathcal{C}$ as f_D. Before it terminates, $\mathcal{C}$ receives a pair (x_1, x_2), which it attempts to convert into a collision in f_{flat}. In order to convert a collision in f_D into a collision in f_{flat}, $\mathcal{C}$ must map the elements of S into the elements of S_{flat}. Additionally, this mapping must be consistent with $\mathcal{C}$'s responses to queries, so that a collision another algorithm discovers from $\mathcal{C}$'s responses is likely to correspond to a collision in the oracle f_{flat}.

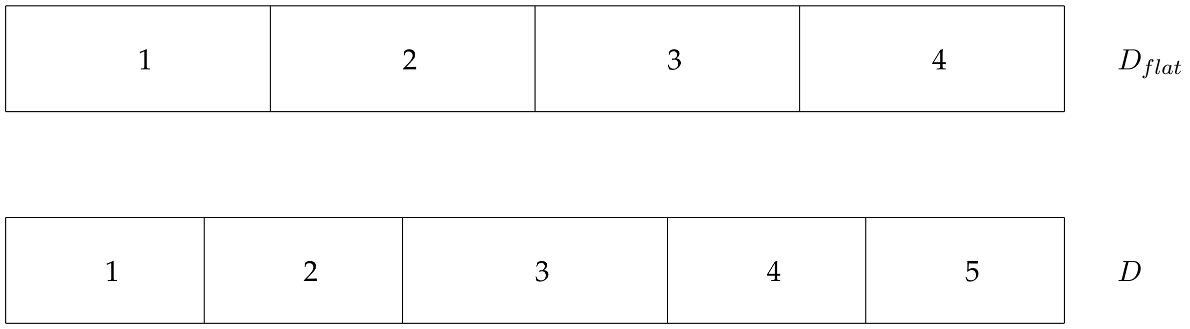
Clearly, an oracle f_D with responses distributed according to D generally cannot be perfectly simulated by a deterministic mapping of the responses of the oracle f_{flat}. Additional randomness generally must be added by $\mathcal{C}$ because the distribution D may have higher Shannon entropy than the distribution D_{flat}, as $|S| \geq |S_{flat}|$. However, the oracle f_D can be simulated by treating the elements of S_{flat} as 'bins', each associated with one or more elements of S. In order to generate a sample from S, a sample from S_{flat} first selects a 'bin', and then one of the elements of S associated with that bin is chosen randomly according a conditional probability distribution such that the marginal probability of sampling that element is equal to probability associated with that element under distribution D. This process can be visualized intuitively by vertically aligning rectangular representations of the distributions D_{flat} and D. The conditional probability of sampling each element in S given a bin in S_{flat} can be illustrated by projecting the dividers between elements in the rectangle representing D into the space between the two rectangles. We show a trivial example below, using distributions of min-entropy 1.



The diagram above also illustrates the role played by the function $c : S \times S_{flat} \to [0, 1]$. This function specifies the conditional probability that $\mathcal{C}$ returns a sample of a certain element of S given that the oracle f_{flat} responded with a certain element from S_{flat}. Formally, $c(y, z) = \Pr[f_D(x) = y | f_{flat}(x) = z]$, where x is a query and $f_D(x)$ is $\mathcal{C}$'s response. All that remains of the collision-conversion procedure then is sampling from the chosen bin according to the conditional probabilities that $c(y, z)$ specifies. The values encoded into the function c therefore make up the non-trivial part of the protocol.

In the example above, the elements 2 and 3 have a total probability mass of 0.5 in D, so they can be 'binned' into element 2 in D_{flat}. Supposing that element 2 is returned by the oracle f_{flat}, a single uniformly random bit would determine whether $\mathcal{C}$ will return 2 or 3 as the response of f_D, since each are equally likely under D. If element 1 is returned by f_{flat}, no additional randomness is needed, as $c(1,1) = 1$. This is always the case for the mode element, because its probability mass under D must exactly equal the probability mass of any of the elements of D_{flat}, since D and D_{flat} have equal min-entropy.

This procedure will perfectly simulate responses from an oracle f_D whose responses are distributed according to D, since the marginal probability of sampling each element in this fashion exactly replicates the associated probability in D. In this simple case, any collision found in f_D will necessarily be a collision in f_{flat}. However, this is not true in general. If the elements of S cannot be grouped into bins each with total probability mass under D equal to 2^{-k}, then some elements of S must have their probability mass split among multiple bins. An example of such a case is shown below.

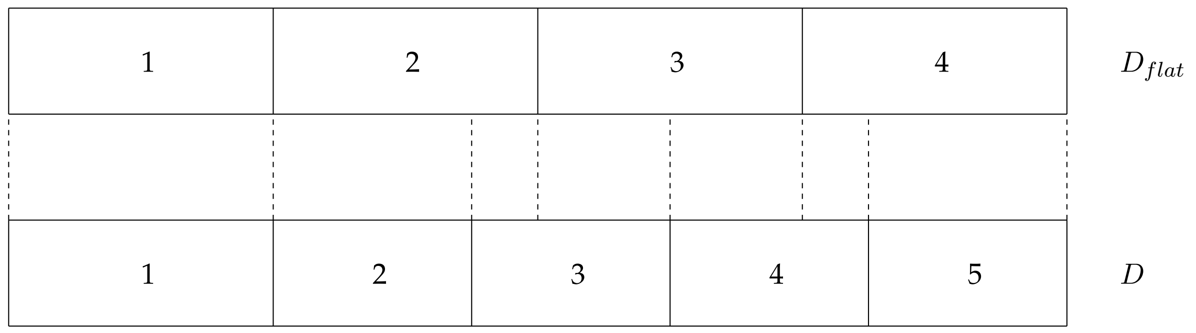


In cases like these, it is possible that a pair of identical responses from $\mathcal{C}$, constituting an apparent collision in f_D, do not actually originate from identical responses from f_{flat}, and therefore do not constitute an actual collision in f_{flat}. Luckily, it is possible to construct the function c such that the probability that a collision in f_D corresponds to a collision in f_{flat} is bounded below by one half, so that C qualifies as a collision-conversion procedure and theorem 3.3 follows. The initialization stage of $\mathcal{C}$ contains a general method for constructing such a function, which we explain now.

The first step is to sort the elements of S in order of decreasing probability under distribution D. The utility of this is that it guarantees that any elements of S which can be trivially associated with an element in S_{flat}, because they have a probability mass equal to 2^{-k}, are mapped to a single element in S_{flat} with probability 1.

Next, $\mathcal{C}$ iterates over the elements y_i of S, setting the value of $c(y_i, z_j)$ for all elements z_j of S_{flat} for each. This may be visualized as moving across the rectangular representations of D and D_{flat} from left to right, determining the values of the collision conversion function along the way. If the probability mass of y_i under D is 2^{-k}, then all of its probability mass is associated with the element in S_{flat} with the same index, so $c(y_i, z_i) = 1$ (and of course zero for all other z_j). If the probability mass corresponding to y_i in D is less than 2^{-k}, then the probability mass from y_i will not 'occupy' an entire bin in D_{flat}, so it must share a bin with other elements of S. But if the current bin is already partially occupied, it may be the case that the probability mass of y_i has to be split between the current bin and the next bin, where the 'current bin' and 'next bin' refer to the elements of S_{flat} which, at this point in execution of $\mathcal{C}$, have the highest index out of the elements for which c has already been assigned and the lowest index out of the elements for which c has not already been assigned, respectively.

To check whether this is the case, $\mathcal{C}$ computes the total probability mass in S that has already been assigned, which it saves as $g(i)$, and checks whether this value is a multiple of 2^{-k}. If it is, then the current bin must be completely occupied, and the probability mass corresponding to y_i will fit completely inside the next bin. The next bin will in this case be indexed by $(g(i)/2^{-k}) + 1$. If $g(i)$ is not a multiple of 2^{-k}, then the probability mass from element y_i may need to be split between the current bin and the next bin. In this case, the index of the current bin will be $\lceil g(i)/2^{-k} \rceil$, because for example if $g(i)/2^{-k} = 2.1$ then the first two bins are completely occupied, and one tenth of the third bin is completely occupied, making the index of the current bin 3. The index of the next bin will thus be the index of the current bin plus one. The value of $c(y_i, z_{\lceil g(i)/2^{-k} \rceil})$, representing the conditional probability of sampling y_i given the current bin has been sampled from D_{flat}, is set to $2^k \min(\lceil g(i)/2^{-k} \rceil - g(i), P(y_i))$. The minimum function guarantees that if it is possible to fit all the probability mass from y_i into the current bin then this is done, and if not, whatever probability mass can fit into the current bin is assigned to the current bin. Naturally, whatever probability mass is not assigned to the current bin must be assigned to the next bin, to conserve marginal probability. The factors of 2^k come from the denominator of 2^{-k} in the conditional probability. Continuing like this, the entire collision-conversion function is constructed. It should be clear that the procedure just described would lead to a c function like the one illustrated below, for the example distribution D which we introduced earlier. In general, the c function that results from this procedure can be visualized by projecting the dividers between elements in *both* rectangles into the space between the two rectangles.



$c(1,1) = 1$

$c(2,2) = 0.75$

$c(3,2) = 0.25$

$c(3,3) = 0.5$

$c(4,3) = 0.5$

$c(4,4) = 0.25$

$c(5,4) = 0.75$

At the end of the initialization stage, for each element in S_{flat}, $\mathcal{C}$ saves the set of all elements in S which are associated with it via the c function. These sets are saved in a function b and will be used to 'invert' (using the term loosely) the c function for the purpose of simulating the responses of an oracle f_D whose responses are distributed according to D.

Now we explain the query-response phase of C. Suppose x is one of the queries submitted to $\mathcal{C}$. The query is forwarded to the oracle f_{flat}, and then retrieves the set $b(f_{flat}(x))$ and the associated conditional probabilities from values of the c function. Next, all $\mathcal{C}$ has to do is sample from the conditional distribution specified by c. It is simple to verify that the next few steps properly do so via inverse transform sampling.

Finally, $\mathcal{C}$ outputs whatever pair of elements is given to it. In order to qualify as a collision-conversion procedure, the probability that $\mathcal{C}$'s output is a collision in f_{flat} conditioned on the fact that it is given a pair that is a collision in f_D must be greater than $1/2$. We now show that this is indeed the case.

Suppose $\mathcal{C}$ is given a pair (x_1, x_2). Let y denote $f_D(x_1)$, $\mathcal{C}$'s response to query x_1, and likewise let y' denote $f_D(x_2)$. Let z denote $f_{flat}(x_1)$ and z' denote $f_{flat}(x_2)$. Then the probability that $\mathcal{C}$ *succeeds* in the sense defined in definition 3.5 can be expressed as $\Pr[z = z'|y = y']$. In case of any confusion, we stress that the probability here is taken over the random choice of $f_{flat} \leftarrow {D_{k,\flat}}^X$ and the randomness of $\mathcal{C}$. We derive a property about $\mathcal{C}$, which holds regardless of how it is used in a reduction with any adversaries under consideration.

Lemma A.1. *For any (x_1, x_2), and the induced (y, y') and (z, z') as defined above, $\Pr[z = z'|y = y'] \geq \frac{1}{2}$. Namely $\mathcal{C}$ has success probability at least $1/2$, and hence gives a ${\frac{1}{2}}^{D_{k,\flat} \to D_k}$ conversion procedure.*

Proof. By Bayes' theorem,

$$\Pr[z = z'|y = y'] = \frac{\Pr[z = z']}{\Pr[y = y']} \Pr[y = y'|z = z'] \,. \tag{1}$$

Applying the law of total probability, we may write

$$\begin{aligned}
\Pr[z = z'] &= \sum_{j=1}^{2^k} \Pr[z' = z_j | z = z_j] \cdot \Pr[z = z_j] \\
&= \sum_{j=1}^{2^k} \Pr[z' = z_j] \cdot \Pr[z = z_j] \quad (z \,\&\, z' \textit{ independent}) \\
&= \sum_{j=1}^{2^k} (2^{-k})^2 = 2^{-k} \quad (\text{definition of a flat-}k\text{-distribution})\,.
\end{aligned}$$

Following the same reasoning just used other than the final step, we may also write $\Pr[y = y'] = \sum_{i=1}^{|S|} (D(y_i))^2$.

Then equation (1) can be rewritten as

$$\Pr[z = z'|y = y'] = \frac{2^{-k}}{\sum_{i=1}^{|S|} (D(y_i))^2} \Pr[y = y'|z = z']. \tag{2}$$

Now we turn our attention to the conditional probability on the right. Applying the law of total conditional probability, we decompose the conditional probability into a sum over all possible values of the random variable z, so

$$\Pr[y = y'|z = z'] = \sum_{j=1}^{2^k} \Pr[y = y'|z = z' \wedge z = z_j] \cdot \Pr[z = z_j | z = z'].$$

Applying Bayes' Theorem again, this time to the conditional expression on the far right, inside the summand, we get

$$\Pr[y = y'|z = z'] = \sum_{j=1}^{2^k} \Pr[y = y'|z = z' \wedge z = z_j] \cdot \frac{\Pr[z = z_j]}{\Pr[z = z']} \cdot \Pr[z = z'|z = z_j].$$

Using our prior result for $\Pr[z = z']$, and the fact that all samples according to D_{flat} have probability 2^{-k}, we can see that the ration in the above equation must be exactly one. Hence we get

$$\Pr[y = y'|z = z'] = \sum_{j=1}^{2^k} \Pr[y = y'|z = z' = z_j] \cdot \Pr[z' = z_j] = \sum_{j=1}^{2^k} \Pr[y = y'|z = z' = z_j] \cdot 2^{-k}.$$

Once again applying the law of total conditional probability, this time decomposing the conditional probability in the summand into a sum over all possible values of the random variable y, we get

$$\begin{aligned} \Pr[y = y'|z = z'] &= 2^{-k} \sum_{j=1}^{2^k} \sum_{i=1}^{|S|} \Pr[y = y'|z = z' = z_j \wedge y = y_i] \cdot \Pr[y = y_i|z = z' = z_j] \\ &= 2^{-k} \sum_{j=1}^{2^k} \sum_{i=1}^{|S|} \Pr[y' = y_i|z' = z_j] \cdot \Pr[y = y_i|z = z_j]. \end{aligned}$$

This step is only possible because y' and z are independent, so the presence of z' in the conditional involving y' can be ignored. The same goes for y and z' in the second conditional. Now, recall that the function c is defined by $c(y, z) = \Pr[f_D(x) = y|f_{flat}(x) = z]$. It follows, by the definitions of y, y', z, and z', that each of the factors in the summand are equal to $c(y_i, z_j)$. Hence we may write

$$\Pr[y = y'|z = z'] = 2^{-k} \sum_{i=1}^{|S|} \sum_{j=1}^{2^k} (c(y_i, z_j))^2,$$

in which we have reversed the order of summation. From the details of how the values of c are assigned when $\mathcal{C}$ is initialized, the number of j values for which $c(y_i, z_j)$ is non-zero is either one (if all of y_i's probability mass is associated with a single bin), or two (if the probability mass is split over two bins). If, for a given i only one value of j corresponds to a non-zero $c(y_i, z_j)$, then $c(y_i, z_j)$ must be $2^k D(y_i)$. But we are interested in the worst case, in order to establish a lower bound. If, for a given i, two values of j correspond to a non-zero $c(y_i, z_j)$, then we know that the two values of c must sum to $2^k D(y_i)$. In this case, the sum of the squares of these values will be less than $2^k D(y_i)$. We can express this sum as $c_1^2 + c_2^2 = c_1^2 + (2^k D(y_i) - c_1)^2$. The minimum value for this parabola is $2^{2k-1} D(y_i)^2$, when $c = 2^{k-1} D(y_i)$. Therefore we may write,

$$\Pr[y = y'|z = z'] \geq 2^{-k} \sum_{i=1}^{|S|} 2^{2k-1} (D(y_i))^2 = 2^{k-1} \sum_{i=1}^{|S|} (D(y_i))^2.$$

Substituting this expression into equation (2), we get

$$\Pr[z = z'|y = y'] \geq \frac{2^{-k}}{\sum_{i=1}^{|S|} (D(y_i))^2} \cdot 2^{k-1} \sum_{i=1}^{|S|} (D(y_i))^2 = \frac{1}{2}.$$

Therefore we conclude that $\Pr[\mathcal{C}\ succeeds] \geq \frac{1}{2}$. □

Hence $\mathcal{C}$ satisfies the definition of a collision-conversion procedure from a flat-k-distribution to an arbitrary k-distribution.

B Alternative proof of lower bound

Theorem B.1. *Suppose $|X| = M = o(\sqrt{N})$. Any q-query quantum algorithm finds a collision in $f \leftarrow X^{D_{k,\delta}}$ with probability $O(q^2/2^k)$.*

We prove this by reducing a variant of Grover's search problem in [HRS16] to finding a collision here. Define the following distribution E_λ on $\mathcal{F} : X \to \{0,1\}$: if $F \leftarrow E_\lambda$, then for any $x \in X$

$$F(x) = \begin{cases} 1 & \text{with prob. } \lambda\,; \\ 0 & \text{with prob. } 1-\lambda\,. \end{cases}$$

It has been shown in [HRS16] that searching for a preimage of q in a function drawn according to F_λ is difficult. More precisely, for any quantum algorithm A making q queries, we define its success probability as

$$\mathsf{Succ}^{\lambda}_{A,q} := \Pr_{F \leftarrow E_\lambda}[F(x) = 1 : x \leftarrow A^F(\cdot)]\,.$$

Lemma B.2. *([HRS16, Theorem 1]) $\mathsf{Succ}^{\lambda}_{A,q} \leq 8\lambda(q+1)^2$ holds for any quantum algorithm A making at most q queries*

Proof of Theorem B.1. Let A be any quantum algorithm that makes at most q queries to $f \leftarrow {D_{k,\delta}}^X$ and finds a collision in f with probability ε. We show how to construct $\mathcal{B}$ that solves the above search problem for $\lambda = 1/2^k$ making $2q$ queries with probability $\varepsilon' = \varepsilon - \gamma$ and Lemma B.2 then implies that $\varepsilon \leq O(q^2/2^k)$.

$\mathcal{B}$ is given quantum access to $F \leftarrow F_\lambda$, and the mode m of $D_{k,\delta}$. Let $h : X \to Y\backslash\{m\}$ be a random function [2] It simulates $\hat{f} : X \to Y$ which answers the queries from A:

$$\hat{f}(x) = \begin{cases} m & \text{if } F(x) = 1\,; \\ h(x) & \text{o.w.}\,. \end{cases}$$

After $\mathcal{A}$ has made q queries to $\hat{f}$, $\mathcal{A}$ outputs x and x'. $\mathcal{B}$ outputs one of them, e.g., x.

Note that $\mathcal{B}$ can implement each evaluation of $\hat{f}$ by two queries to F. Therefore $\mathcal{B}$ makes $2q$ queries to F at most. Next observe that $\hat{f}$ is distributed *identically* as $f \leftarrow X^{D_{k,\delta}}$, because $\Pr[\hat{f}(x) = m] = \Pr_{F \leftarrow F_\lambda}[F(x) = 1] = 1/2^k$ and the rest of $\hat{f}(x)$ is uniform over $Y\backslash\{m\}$. Therefore we know that $\hat{f}(x) = \hat{f}(x')$ with probability ε. Finally notice that when $M = o(\sqrt{N})$, h will be injective except with probability negligible in k. Therefore the collision only occurs at the mode, which implies taht $F(x) = 1$ and B successfully finds a marked element in F. $\square$

Corollary B.3. *Any quantum algorithm needs $\Omega(2^{k/2})$ queries to find a collision in $X^{D_{k,\delta}}$ with constant probability even if the mode m of D is known, when $M = o(\sqrt{N})$.*

Remark 2. To see that this result is basically subsumed by Theorem 3.10. Note that when $N < 2^{2k}$, $\beta(D) = N$. Therefore $M = o(\sqrt{N}) = o(\sqrt{\beta(D)})$, and the function drawn is almost always injective. Hence the lower bound trivially holds. When $N \geq 2^{2k}$, Corollary 3.11 gives the same lower bound $2^{k/2}$.

The same proof strategy also works for general M, but then the probability that the collision occurs elsewhere other than the mode will introduce error, and it will match the bound we obtain from Theorem 3.10.

[2]This can be efficiently simulated by a $2q$-wise independent hash function as justified by [Zha15a, Theorem 6.1] and [JKMR09, Lemma 2].