Succinct Arguments in the Quantum Random Oracle Model

Alessandro Chiesa
alexch@berkeley.edu
UC Berkeley

Peter Manohar
pmanohar@cs.cmu.edu
Carnegie Mellon University

Nicholas Spooner
nick.spooner@berkeley.edu
UC Berkeley

October 1, 2019

Abstract

Succinct non-interactive arguments (SNARGs) are highly efficient certificates of membership in non-deterministic languages. Constructions of SNARGs in the random oracle model are widely believed to be post-quantum secure, provided the oracle is instantiated with a suitable post-quantum hash function. No formal evidence, however, supports this belief.

In this work we provide the first such evidence by proving that the SNARG construction of Micali is unconditionally secure in the *quantum* random oracle model. We also prove that, analogously to the classical case, the SNARG inherits the zero knowledge and proof of knowledge properties of the PCP underlying the Micali construction. We thus obtain the first zero knowledge SNARG of knowledge (zkSNARK) that is secure in the quantum random oracle model.

Our main tool is a new lifting lemma that shows how, for a rich class of oracle games, we can *generically* deduce security against quantum attackers by bounding a natural classical property of these games. This means that in order to prove our theorem we only need to establish *classical* properties about the Micali construction. This approach not only lets us prove post-quantum security but also enables us to prove explicit bounds that are tight up to small factors.

We additionally use our techniques to prove that SNARGs based on interactive oracle proofs (IOPs) with round-by-round soundness are unconditionally secure in the quantum random oracle model. This result establishes the post-quantum security of many SNARGs of practical interest.

Keywords: succinct arguments; quantum random oracle model; probabilistically checkable proofs

Contents

1 Introduction

The design and analysis of cryptographic primitives that are plausibly secure against quantum attackers is an increasingly important goal. The expected advent of quantum computers demands the cryptography community to be prepared well in advance, so much so that the National Institute of Standards and Technology (NIST) is *already* in the process of selecting, from among many proposals, a new set of cryptography standards that are "post-quantum" [NIS16]. The proposals involve schemes for key agreement, public-key encryption, and digital signatures, and are intended to eventually replace existing standards based on the hardness of factoring or discrete logarithms.

In this paper we study the post-quantum security of a cryptographic primitive that has recently received much attention across theoretical and applied communities: *succinct arguments* [GW11]. These are argument systems for non-deterministic languages where the communication complexity between the prover and verifier is sublinear in the size of the non-deterministic witness.[1] This notion originates in seminal works of Kilian [Kil92] and Micali [Mic00], which construct succinct arguments for languages in $\mathsf{NTIME}(T(n))$ where communication complexity is $\mathsf{poly}(\lambda, \log T(n))$ and the time complexity of the verifier is $\mathsf{poly}(\lambda, n, \log T(n))$; here λ is the security parameter.

Researchers have studied many aspects of succinct arguments in the last two decades, leading to numerous constructions with different tradeoffs [WB15], efficient realizations in code [SCI14; bell15; SCI18; dalek18; stark18; SCI19; iden19], real-world deployments [Zc14; Co17], and standardization efforts [ZKP17]. A particularly useful feature is that many succinct arguments can be made zero knowledge with minimal overhead. At present, however, *most approaches to obtain efficient succinct arguments are "pre-quantum"*, since they rely on the discrete logarithm problem (and more).

A notable exception is a class of succinct arguments obtained by combining two ingredients: (a) probabilistic proof systems, which are unconditionally secure, and (b) cryptographic hash functions, for which we have post-quantum candidates. This class includes the succinct interactive argument of Kilian [Kil92], which use probabilistically checkable proofs (PCPs) [BFLS91; FGLSS96; AS98; ALMSS98] and collision-resistant hash functions. It also includes the succinct non-interactive argument (SNARG) of Micali [Mic00], which uses PCPs and random oracles. More generally, by using random oracles one can construct a SNARG from a multi-round generalization of PCPs known as interactive oracle proofs (IOPs) [BCS16; RRR16]. *All of these succinct arguments are widely believed to be post-quantum*, provided the hash function is suitably instantiated [BBHR19].[2]

There is, however, no formal evidence that supports the above widely-held belief. Since succinct arguments are a fundamental cryptographic primitive with both theoretical and real-world applications, it is important to prove guarantees on their post-quantum security.

1.1 SNARGs with random oracles

In this paper we focus our attention on the SNARG construction of Micali [Mic00], which is unconditionally secure in the random oracle model [BR93; PS96]. SNARGs in the random oracle model are not only plausibly post-quantum secure but also enjoy other desirable features. Namely, the random oracle can be heuristically instantiated via hash functions that avoid expensive public-key cryptographic operations. Moreover, the SNARG uses a transparent (public-coin) setup, because the only public parameter needed to produce/verify proofs is the choice of hash function.

[1] Achieving communication complexity that is sublinear in the witness size is known to *require* relaxing soundness from statistical to computational, provided one assumes standard complexity conjectures [GH98; GVW02].

[2] There is also a class of lattice-based succinct arguments that is plausibly post-quantum; see Section 1.3.

We are thus interested in asking: can we establish formal evidence that the SNARG construction of Micali is post-quantum secure? One way to establish formal evidence is to prove security in a quantum analogue of the random oracle model, as we now explain. A quantum attacker can, among other things, evaluate a hash function in superposition when given the hash function's code. This enables the attacker, for instance, to find pre-images [Gro96] or collisions [BHT98] faster than a classical attacker. In light of this, Boneh et al. [BDFLSZ11] have argued that, in the quantum setting, the correct way to model a random oracle is to allow the attacker to query the random oracle in superposition. The resulting model is known as the *quantum random oracle model* (QROM), and a line of work has established post-quantum security within this model for a variety of cryptographic primitives; see, e.g., [BDFLSZ11; Zha12; Zha15; TU16; Eat17].

Our goal is to study the SNARG construction of Micali in the quantum random oracle model. We also study the SNARG construction of BCS [BCS16], which yields SNARGs of practical interest.

1.2 Our results

The main result of this paper is establishing that the SNARG construction of Micali [Mic00] is unconditionally secure in the quantum random oracle model. This is the first formal evidence that supports the widely-held belief that this construction is post-quantum secure when the oracle is instantiated via a suitable post-quantum hash function.

Theorem 1 (informal)**.** *The non-interactive argument of Micali, when based on a PCP with soundness error ϵ, has soundness error $O(t^2\epsilon + t^3/2^\lambda)$ against quantum attackers that make t queries to a random oracle with output size λ. This soundness error is tight up to small factors.*

A key step in our proof, of independent interest, is a *Lifting Lemma* that shows how, for a rich class of "oracle games", we can *generically* deduce security against quantum attackers by bounding a natural classical property of these games, instability, that we introduce. This means that to prove Theorem 1 we only need to bound the instability of the Micali construction. This approach not only yields the theorem but also enables us to prove explicit bounds that are tight up to small factors.

If we base the Micali construction on suitable PCPs, we obtain new statements about the existence of post-quantum non-interactive arguments. First, if the PCP achieves (honest-verifier) zero knowledge and proof of knowledge then through the Micali construction we obtain a zero knowledge non-interactive argument of knowledge that is *unconditionally* secure in the quantum random oracle model. This strengthens a result of Unruh [Unr15], which assumes the existence of a post-quantum Σ-protocol for NP. Moreover, if the PCP has polylogarithmic query complexity and verifier running time then we obtain the first construction of a zero knowledge succinct non-interactive argument of knowledge (zkSNARK) that is secure in the quantum random oracle model.

Theorem 2 (informal)**.** *There exists a zero knowledge non-interactive argument of knowledge for NP in the quantum random oracle model. Moreover, the non-interactive argument is succinct, in the sense that arguments have size λ^c and can be verified in time $(\lambda \cdot n)^c$, where λ is the random oracle's security parameter, n is instance size, and $c \geq 1$ is a universal constant.*

The above theorem is stated for NP only for simplicity. Analogously to the classical case, a more general statement holds for all non-deterministic time languages by relying on suitable PCPs for non-deterministic time. For example, the PCP in [BFLS91] achieves proof of knowledge, can be made (honest-verifier) zero knowledge [DFKNS92; KPT97], and supports general non-deterministic time computations.

The BCS construction. We conclude with a result that demonstrates how the tools in this paper can be used to study the post-quantum security of protocols that are of practical interest. Since known PCP constructions are expensive, efficient constructions of succinct arguments in the random oracle model are typically based on the BCS construction [BCS16], which instead uses interactive oracle proofs (IOPs) [BCS16; RRR16], a multi-round extension of PCPs. This extension additionally captures IPs [Bab85; GMR89] and IPCPs [KR08] as special cases.

We prove that the BCS construction is unconditionally secure in the quantum random oracle model, if applied to public-coin IOPs that have round-by-round soundness [CCHLRR18]. The resulting argument inherits proof of knowledge and zero knowledge properties of the underlying IOP.

Theorem 3 (informal)**.** *The non-interactive argument of BCS, when based on a public-coin IOP with round-by-round soundness error ϵ, has soundness error $O(t^2\epsilon + t^3/2^\lambda)$ against quantum attackers that make t queries to a random oracle with output size λ. Moreover, it is an argument of knowledge if the IOP has round-by-round proof of knowledge, and it is a (statistical) zero knowledge argument if the IOP is honest-verifier zero knowledge.*

Round-by-round proof of knowledge is a natural notion that we introduce, analogous to round-by-round soundness, and is satisfied by many natural protocols. In particular, Theorem 3 enables us to deduce the post-quantum security of succinct arguments based on well-known IPs such as the sumcheck protocol [LFKN92] and the GKR protocol [GKR15], as well as zkSNARKs based on recent IOPs such as [BBHR19; AHIV17; BCRSVW19]. These protocols (among others) are of interest to practitioners, and our result can be used to guide parameter choices in practice.

1.3 Related work

Argument systems that use random oracles. Several works study the post-quantum security of zero knowledge non-interactive arguments of knowledge that use random oracles, most notably those obtained by applying the Fiat–Shamir transformation [FS86] to a post-quantum Σ-protocol. These are used to achieve post-quantum digital signatures [CDGORRSZ17; KKW18; BN19], and underlie constructions submitted to the NIST call for post-quantum cryptography [NIS16].

A security reduction for the Fiat–Shamir transformation in the quantum random oracle model has been recently achieved [DFMS19; LZ19]. Obtaining a security reduction had been elusive, as the classical approach of rewinding the adversary to reduce to special soundness of the Σ-protocol does not work for quantum adversaries.[3] Before, researchers were only able to prove security if the underlying Σ-protocol satisfies special properties [DFG13; Unr17; KLS18], or resorted to proving security for alternative, less efficient, constructions such as the Unruh transformation [Unr15].

The question that we study in this paper is complementary to these prior works. On the one hand, prior works study the security of the Fiat–Shamir transformation *given* that the underlying Σ-protocol is secure against efficient quantum attackers. On the other hand, we study protocols such as the Micali construction and BCS construction that can be viewed as applying the Fiat–Shamir transformation to specific public-coin protocols that are known to be unconditionally secure in the (classical) random oracle model. In particular, we establish *unconditional* security in the

[3] Rewinding quantum adversaries is a delicate matter [Wat09] and, more importantly, special soundness does *not* imply post-quantum soundness (relative to some oracle) [ARU14]. These difficulties have been circumvented by using new techniques that enable reducing directly to the (post-quantum) soundness of the underlying Σ-protocol.

quantum random oracle model via an approach that considers the protocol as a whole (similarly to the classical analysis of these protocols).

The foregoing differences are reflected in a technical analysis that departs from prior works. Most of the effort in this paper is establishing *classical* security properties of the Micali and BCS constructions, which we then use to generically deduce their quantum security. This approach, besides being intuitive, yields tight bounds that can be used to guide parameter choices in practice.

Succinct arguments based on lattices. Several lattice problems are presumed to remain hard even against quantum adversaries, and researchers have relied on such problems to propose numerous cryptographic constructions that are plausibly post-quantum. A handful of works have used lattices to achieve various notions of succinct arguments that are plausibly post-quantum. Baum et al. [BBCPGL18] rely on the short integer solution (SIS) problem to obtain an argument system for arithmetic circuits where the communication complexity grows with the square-root of circuit size; the argument system is constant-round, public-coin, and honest-verifier zero knowledge. Boneh et al. [BISW17; BISW18] and Gennaro et al. [GMNO18] rely on lattice knowledge assumptions to construct designated-verifier SNARGs for boolean circuits, in the preprocessing model [BCIOP13]. Whether one can use lattices to obtain publicly-verifiable SNARGs remains an open problem.

2 Techniques

We discuss the main ideas behind our results. In Section 2.1 we recall the construction of Micali, and then in Section 2.2 we explain the challenges that arise when trying to prove its security in the quantum random oracle model. In Section 2.3 we outline our approach to obtain a proof of security for the Micali construction (Theorem 1); we elaborate on our approach in Sections 2.4 to 2.7. Finally, in Section 2.8 we discuss how to further establish zero knowledge and proof of knowledge; we thus obtain the first zkSNARK secure in the quantum random oracle model (Theorem 2).

We conclude in Section 2.9 by explaining how our techniques extend to establish post-quantum security for the BCS construction applied to many protocols of practical interest (Theorem 3).

2.1 The construction of Micali

The construction of Micali is a transformation that maps any *probabilistically checkable proof* (PCP) into a corresponding non-interactive argument in the random oracle model. (See Section 3.4 for the definition of a PCP, and Section 3.3 for that of a non-interactive argument.) The resulting non-interactive argument is *succinct*, i.e. a SNARG, provided the PCP has suitable parameters.

Let $(\mathbf{P}, \mathbf{V})$ be a PCP for a relation $\mathcal{R}$ with soundness error ϵ, proof length ℓ over alphabet Σ, and query complexity q. The honest prover $\mathbf{P}$ takes as input an instance-witness pair $(\mathtt{x}, \mathtt{w})$ and outputs a proof string $\Pi\colon [\ell] \to \Sigma$. The honest verifier $\mathbf{V}$ takes as input the instance $\mathtt{x}$, makes q probabilistic queries to a (possibly malicious) proof string $\tilde{\Pi}\colon [\ell] \to \Sigma$, and then accepts or rejects.

The PCP $(\mathbf{P}, \mathbf{V})$ for $\mathcal{R}$ is used to construct a SNARG $(\mathcal{P}, \mathcal{V})$ for $\mathcal{R}$, as follows.

The SNARG prover $\mathcal{P}$ takes as input an instance $\mathtt{x}$ and witness $\mathtt{w}$. First, $\mathcal{P}$ uses the random oracle h to commit to the proof string $\Pi := \mathbf{P}(\mathtt{x}, \mathtt{w})$ via a Merkle tree, obtaining a corresponding root rt. Second, $\mathcal{P}$ applies the random oracle h to the root rt in order to derive randomness r for the PCP verifier $\mathbf{V}$. Third, $\mathcal{P}$ simulates the PCP verifier $\mathbf{V}$ with the proof string Π, input $\mathtt{x}$, and randomness r, in order to deduce the queried locations of Π. Finally, $\mathcal{P}$ assembles a SNARG proof π that contains the root rt, answers to the queries, and an authentication path for each answer.

Observe that the SNARG proof π is succinct because it is small (it has size $|\pi| = O(q \cdot (\log |\Sigma| + \lambda \log \ell)) = O_\lambda(q)$ for $\ell, |\Sigma| = 2^{O(\lambda)}$) and it is cheap to validate via the algorithm described next.

The SNARG verifier $\mathcal{V}$ takes as input an instance $\mathtt{x}$ and a (possibly malicious) SNARG proof $\tilde{\pi}$. First, $\mathcal{V}$ uses the random oracle h to check that each answer in $\tilde{\pi}$ is certified by an authentication path relative to the claimed root $\tilde{\mathsf{rt}}$. Next, $\mathcal{V}$ applies the random oracle h to the root $\tilde{\mathsf{rt}}$ in order to derive randomness $\tilde{\mathsf{r}}$. Finally, $\mathcal{V}$ runs the PCP verifier $\mathbf{V}$ on the instance $\mathtt{x}$ and randomness $\tilde{\mathsf{r}}$, answering $\mathbf{V}$'s queries using the claimed answers in $\tilde{\pi}$.

The intuition behind the construction is that the soundness guarantee of a PCP holds only if the proof string $\tilde{\Pi}$ to be validated is *fixed* before the randomness $\tilde{\mathsf{r}}$ for the PCP verifier is known, and for this reason the SNARG prover must derive $\tilde{\mathsf{r}}$ by hashing a commitment $\tilde{\mathsf{rt}}$ to $\tilde{\Pi}$.

This construction is unconditionally secure in the random oracle model [Mic00; Val08; BCS16]:

Theorem 2.1. *The SNARG $(\mathcal{P}, \mathcal{V})$ has soundness error $O(t\epsilon + t^2/2^\lambda)$ against (classical) attackers that make at most t queries to the random oracle. This soundness error is tight up to small factors.*

A SNARG obtained via the Micali construction also inherits zero knowledge and proof of knowledge properties of the underlying PCP. We discuss these additional properties and how we establish them in the quantum setting later on in Section 2.8. We focus on soundness first.

2.2 Challenges in the quantum setting

Our goal is to show that the SNARG construction of Micali is unconditionally secure in the *quantum* random oracle model. Suppose that $\tilde{\mathcal{P}}$ is a t-query quantum prover that convinces the SNARG verifier $\mathcal{V}$ with probability δ (over the random oracle). We wish to construct a malicious PCP prover $\tilde{\mathbf{P}}$ that, using $\tilde{\mathcal{P}}$ as a subroutine, outputs a proof string $\tilde{\Pi}\colon [\ell] \to \Sigma$ that convinces the PCP verifier $\mathbf{V}$ with related probability $\epsilon(\delta, t)$ (here the probability is over the randomness of $\tilde{\mathbf{P}}$ and $\mathbf{V}$).

A natural approach to reduce the SNARG prover $\tilde{\mathcal{P}}$ to the PCP prover $\tilde{\mathbf{P}}$ would be to try to adapt to the quantum setting the reduction that is used for the classical setting. Below we recall the classical reduction, and then explain why adapting it to the quantum case is challenging.

The reduction for classical attackers. The reduction from a *classical* SNARG prover $\tilde{\mathcal{P}}$ to a PCP prover $\tilde{\mathbf{P}}$ relies on a *straightline extractor*, as we now explain.

While the SNARG prover $\tilde{\mathcal{P}}$ outputs a short proof π that contains a Merkle root and a few decommitted values, the PCP prover $\tilde{\mathbf{P}}$ must output a "long" proof string $\tilde{\Pi}$. How can $\tilde{\mathbf{P}}$ obtain all this information from seeing only π? The answer is that, when running $\tilde{\mathcal{P}}$ as a subroutine, $\tilde{\mathbf{P}}$ observes the queries that $\tilde{\mathcal{P}}$ makes to the oracle, and these queries reveal the proof string $\tilde{\Pi}$.

This is only a caricature of how $\tilde{\mathbf{P}}$ actually works, though. The reason is that $\tilde{\mathcal{P}}$ need not produce a query sequence from which $\tilde{\mathbf{P}}$ can just read off a proof string $\tilde{\Pi}$ consistent with the Merkle root in π. For example, $\tilde{\mathcal{P}}$ could try to commit to many possible proof strings "in its head", derive the corresponding randomness from each commitment, and then select which commitment to include in π. Even worse, $\tilde{\mathcal{P}}$ could try to commit to a *partial* proof string $\tilde{\Pi}$ via an incomplete Merkle tree and, because the PCP verifier inspects only a small fraction of a proof string, hope that queries will land to leaves of the Merkle tree that do exist.

The proof of Theorem 2.1 shows that, despite these complications, there is a way for $\tilde{\mathbf{P}}$ to observe all queries and answers of a single execution of the SNARG prover $\tilde{\mathcal{P}}$, and then run an algorithm on these to extract a suitable proof string $\tilde{\Pi}$.

How to deal with quantum attackers? If we now return to the case where the SNARG prover $\tilde{\mathcal{P}}$ is a quantum attacker, we are immediately confronted with a severe problem. Since $\tilde{\mathcal{P}}$ can query the random oracle in superposition, how can $\tilde{\mathbf{P}}$ "observe" queries and answers to the oracle? If $\tilde{\mathbf{P}}$ were to just measure $\tilde{\mathcal{P}}$'s query register, $\tilde{\mathcal{P}}$ may detect this and stop working. This basic problem has made obtaining security reductions against quantum attackers that access random oracles exceedingly difficult when compared to the case of classical attackers. Papers that study the security of cryptographic primitives in the quantum random oracle model have had to develop clever techniques to somehow circumvent this problem in various settings of interest.

Most relevant to this paper is a work of Zhandry [Zha19] that introduces *compressed oracles*, a set of notions and techniques that enables a quantum algorithm to simulate access to a random oracle for a quantum attacker. This is achieved by replacing a random oracle $h\colon \{0,1\}^m \to \{0,1\}^n$ with the action of a specially-crafted unitary $\mathcal{O}$ that implicitly keeps track of queries. This is a quantum analogue of when, in the classical setting, a simulator merely observes the queries made by the attacker and maintains a database of the query-answer pairs. Formally, the classical simulator keeps track of a database D, which is a partial function $D\colon \{0,1\}^m \rightharpoonup \{0,1\}^n$. The database represents the part of the random oracle that has been "revealed" to the attacker by answering its queries. In the quantum setting, the state space of the quantum attacker is augmented with registers to store the database, which (loosely) keep track of the database D in superposition, as it evolves from query to query. Thus, while the original oracle h operates on the state $|\psi_{\mathcal{A}}\rangle$ of the

adversary, the unitary $\mathcal{O}$ operates on a bipartite state $|\psi_{\mathcal{A}}, \psi_D\rangle$. This extended state represents a purification of the mixed state of the adversary induced by choosing the oracle h at random.

One may conjecture that the compressed oracle technique, by virtue of "exposing" a quantum attacker's queries, makes proving the quantum security of the Micali construction, or indeed of any construction that uses random oracles, straightforward. This is, unfortunately, not the case.

For example, the results of [Zha19] on compressed oracles allow us to argue directly that, given an adversary that outputs a convincing SNARG proof π with high probability, if we measure the database D after the adversary terminates, then with high probability one can find a convincing *SNARG proof* π in the database D. This does not allow us to reduce to soundness of the underlying PCP, however, because to do that we need to argue that one can extract a *PCP proof* Π from D (containing a lot more information than π) that convinces the PCP verifier with high probability.

While the techniques in [Zha19] suffice for analyzing simple protocols like collision finding, it is not clear how to analyze more complex protocols such as the Micali SNARG, as explained above. In the next section we describe a general framework for analyzing complex protocols in the quantum random oracle model.

2.3 Outline of our approach

The ideas that we use in this paper to analyze the Micali construction are almost entirely generic, and can be used to analyze any *oracle game*. Informally, given a "base game" $G \subseteq A^k \times B^k \times C$, an adversary with oracle access to a random oracle h wins the oracle game for G if it outputs a tuple $(\mathbf{a}, \mathbf{b}, c) \in G$ where $h(a_i) = b_i$ for each $i \in [k]$. Oracle games are a natural notion that captures many games of interest, such as finding pre-images or finding collisions. Producing a valid proof in the Micali construction can also be cast as an oracle game,[4] and we shall view the soundness property of the Micali construction as stating that the value (maximum winning probability) of this game is small (when the statement being proved is false).

Our proof of quantum security consists of two main parts. First, we *generically* reduce the value of any oracle game to the *instability* of the game, a purely classical property of the game that we introduce. Second, we analyze the instability of the oracle game induced by the Micali construction. The instability of this oracle game is not too difficult to analyze because it is a classical quantity, and the "hard work" is crisply, and conveniently, encapsulated within our generic reduction. We view bounding values of oracle games via instability as the main technical contribution of this paper.

We now elaborate on our approach: in Section 2.4 we recast prior work in the language of oracle games; in Section 2.5 we explain what is instability and how we use it to bound game values; in Section 2.6 we introduce conditional instability and use it to prove tighter bounds on oracle game values; and in Section 2.7 we outline the analysis of instability for the Micali construction.

2.4 From oracle games to database games

We begin with a sequence of three games whose values are closely related. These games play the role of hybrids in our analysis, and are all defined relative to the given base game $G \subseteq A^k \times B^k \times C$.

- **Oracle game.** This is the game defined earlier that is played in the real world, using a random oracle h. The adversary wins if it outputs a tuple $(\mathbf{a}, \mathbf{b}, c) \in G$ with $h(a_i) = b_i$ for each $i \in [k]$.

[4] At a high level, G is the set of all proofs that cause the verifier to accept relative to some oracle; the $h(a_i) = b_i$ constraints ensure that the verifier accepts this proof relative to the specific oracle h. For more details, see Section 6.2.

- **Simulated oracle game.** The simulator of Zhandry [Zha19] is used to run the adversary and its final state is measured, leading to a tuple $(\mathbf{a}, \mathbf{b}, c)$ *and* a database D. The adversary wins if $(\mathbf{a}, \mathbf{b}, c) \in G$ and $D(a_i) = b_i$ for each $i \in [k]$. (The oracle $h\colon \{0,1\}^m \to \{0,1\}^n$ is now replaced by the database $D\colon \{0,1\}^m \rightharpoonup \{0,1\}^n$ stored by the simulator.)

- **Database game.** Again the simulator of Zhandry is used to run the adversary, leading to a tuple $(\mathbf{a}, \mathbf{b}, c)$ and a database D. However, now we ignore $(\mathbf{a}, \mathbf{b}, c)$ and only consider D. The adversary wins if there *exists* $(\mathbf{a}', \mathbf{b}', c') \in G$ such that $D(a_i) = b_i$ for each $i \in [k]$.

We let $\omega^*_{\mathsf{O}}(G,t)$, $\omega^*_{\mathsf{S}}(G,t)$, and $\omega^*_{\mathsf{D}}(G,t)$ denote the values of the oracle game, simulated oracle game, and database game against quantum adversaries that make at most t oracle queries.

A result of Zhandry [Zha19, Lemma 5], when stated via the notions above, shows that $\sqrt{\omega^*_{\mathsf{O}}(G,t)} \leq \sqrt{\omega^*_{\mathsf{S}}(G,t)} + \sqrt{k/2^n}$. Moreover, $\omega^*_{\mathsf{S}}(G,t) \leq \omega^*_{\mathsf{D}}(G,t)$ holds trivially, because winning the simulated oracle game implies winning the database game, by taking $(\mathbf{a}', \mathbf{b}', c') := (\mathbf{a}, \mathbf{b}, c)$. In sum:

Lemma 2.2. *For any base game G,*

$$\sqrt{\omega^*_{\mathsf{O}}(G,t)} \leq \sqrt{\omega^*_{\mathsf{D}}(G,t)} + \sqrt{k/2^n} \ .$$

The above lemma is a conceptualization of prior work, and is the starting point for the technical contributions of this paper. In particular, the lemma tells us that in order to bound the maximum winning probability of a quantum adversary in an oracle game (played in the real world) it suffices to bound the maximum winning probability of the adversary in the corresponding database game.

See Section 4 for more details.

2.5 A basic lifting lemma for database games

We describe how we use a *classical* quantity $\mathbf{I}(\mathcal{P}_G, t)$ to bound $\omega^*_{\mathsf{D}}(G,t)$, the maximum winning probability of any t-query quantum algorithm in the database game of G. When combined with the hybrids in Section 2.4, this reduces the quantum security of oracle games to studying $\mathbf{I}(\mathcal{P}_G, t)$.

Given a base game G, we let $\mathcal{P}_G$ be the set of databases that win the database game of G. In the classical setting, a natural way to bound the maximum winning probability of the database game is to compute, for each possible database $D \notin \mathcal{P}_G$ (a database that is currently losing the game), the maximum probability that adding a query-answer pair to D puts D in $\mathcal{P}_G$. Assuming that the empty database is not in $\mathcal{P}_G$ (for otherwise one can win trivially), this quantity characterizes the probability that the adversary gets lucky and ends up with a winning database D.

We define the *instability* of $\mathcal{P}_G$ with query bound t, denoted $\mathbf{I}(\mathcal{P}_G, t)$, to be the maximum probability that, for any database D containing less than t queries, making one additional (classical) query changes whether or not D is in $\mathcal{P}_G$. *The foregoing argument explains that the classical value of the database game G is bounded by $t \cdot \mathbf{I}(\mathcal{P}_G, t)$.* Intuitively this is because each query can increase the probability that the database D is in $\mathcal{P}_G$ by at most $\mathbf{I}(\mathcal{P}_G, t)$.

We prove that an analogous result holds for quantum adversaries as well. We call this lemma a lifting lemma, because it enables us to use the *classical* quantity of instability to prove a bound on the maximum winning probability of *quantum* adversaries. The version below is a "basic" version, because we shall ultimately need a stronger statement, as we discuss in Section 2.6. The result below extends an idea of Zhandry sketched in [Zha19, Section 4.3].

Lemma 2.3 (Basic lifting lemma)**.** *For any base game G,*

$$\omega^*_{\mathsf{D}}(G,t) \le O\big(t^2 \cdot \mathbf{I}(\mathcal{P}_G,t)\big) \ .$$

In particular, combining the above with Lemma 2.2, we get

$$\omega^*_{\mathsf{O}}(G,t) \le O\big(t^2 \cdot \mathbf{I}(\mathcal{P}_G,t) + k/2^n\big) \ .$$

Even the above basic lifting lemma is a powerful tool. For example, suppose that G is the collision game, where the adversary wins if it outputs an oracle collision. Then $\mathbf{I}(\mathcal{P}_G,t) < t/2^n$, because if D is a database with no collisions and less than t entries, then making one more query produces a collision with probability less than $t/2^n$, and if D has collisions then it is not possible to make an additional query and remove collisions. Then (since $k=2$ in the collision game) the lifting lemma immediately tells us that $\omega^*_{\mathsf{O}}(G,t) \le O(t^3/2^n)$, which shows that the probability that a t-query quantum oracle algorithm finds a collision is bounded by $O(t^3/2^n)$. This further simplifies the analysis of this fact in [Zha19] and matches the bound of [AS04] (which is tight [BHT98]).

We now sketch the proof of the basic lifting lemma. The proof sketch differs slightly from the actual proof, as in the actual proof we do a slightly more complicated analysis that gives us smaller constants. The main ideas, however, remain the same.

We let P_G be the operator that projects onto databases that win the database game G: for any basis state $|D\rangle$ in the database register, $P_G|D\rangle = |D\rangle$ if $D \in \mathcal{P}_G$, and $P_G|D\rangle = 0$ if $D \notin \mathcal{P}_G$; P_G acts as the identity on other registers. If $|\phi\rangle$ is the final joint state of the quantum adversary and database, then $\|P_G|\phi\rangle\|^2$ is the probability that $D \in \mathcal{P}_G$ after measurement. We will assume that $\emptyset \notin \mathcal{P}_G$, i.e., that the empty database does not win the database game of G (or else the adversary can win by doing nothing).

We can represent any simulated quantum adversary making at most t queries as a sequence of unitary operators $U = A_t\mathcal{O}A_{t-1}\mathcal{O}\dots A_1\mathcal{O}$ applied to an initial state $|\phi_0,\emptyset\rangle := |\phi_0\rangle \otimes |\emptyset\rangle$, where $\mathcal{O}$ is the compressed oracle and $|\emptyset\rangle$ is the state of the empty database. Each A_i acts non-trivially only on the registers of the adversary being simulated and P_G acts non-trivially only on the database registers, so P_G and A_i commute. So, if P_G and $\mathcal{O}$ were to also commute, then we could simply conclude that $P_GU|\phi_0,\emptyset\rangle = UP_G|\phi_0,\emptyset\rangle = 0$, i.e., that the adversary never wins. (Here we used the fact that $\emptyset \notin \mathcal{P}_G$.)

However, it is *not* the case that P_G and $\mathcal{O}$ commute. This should be expected because in general an adversary can win with some positive probability. However, if we could show that they *almost* commute, then we could apply the previous argument to show that $P_GU|\phi_0,\emptyset\rangle \approx UP_G|\phi_0,\emptyset\rangle = 0$; i.e., the adversary wins with small probability. The notion of "almost" commuting we use is that the operator norm $\|[P_G,\mathcal{O}]\|$ of the commutator $[P_G,\mathcal{O}] := P_G\mathcal{O} - \mathcal{O}P_G$ is small.

Unfortunately, for interesting games the operator norm $\|[P_G,\mathcal{O}]\|$ may not be small. For example, if G is the collision game and D is a database with a pre-image of every $y \in \{0,1\}^n$ but no collisions, then $\|[P_G,\mathcal{O}]|x,u,D\rangle\| = 1$. Generally, this norm may be large if D has many entries.

Query-bounded adversaries, however, cannot produce nonzero amplitudes on databases with more entries than the query bound. Hence, intuitively we should not consider states that correspond to large databases when bounding the operator norm of the aforementioned commutator. We follow this intuition by introducing the notion of a *projected oracle*, which acts as the compressed oracle except that it discards databases that do not belong to a certain subset.

Definition 2.4. *Let P be the operator that projects onto databases that belong to a given subset $\mathcal{P}$ of databases. A* **projected oracle** *is an operator of the form $P\mathcal{O}P$.*

We thus consider the projected oracle $P_t\mathcal{O}P_t$, where P_t is operator that projects onto databases containing at most t queries. For adversaries that make at most t queries, replacing $\mathcal{O}$ with $P_t\mathcal{O}P_t$ has no effect because the adversary cannot create a database that contains more than t entries. Moreover, $\|[P_G, P_t\mathcal{O}P_t]\,|D\rangle\| = 0$ if D contains more than t entries, so the operator norm of $[P_G, P_t\mathcal{O}P_t]$ accounts for the action of $\mathcal{O}$ *only* on databases containing at most t entries.

In sum, projected oracles allow us to cleanly compute the operator norm only over databases that are reachable by an adversary making a bounded number of queries. By carefully analyzing the action of $\mathcal{O}$, we show that

$$\|[P_G, P_t\mathcal{O}P_t]\|^2 \leq O\big(\mathbf{I}(\mathcal{P}_G, t)\big) \ .$$

We additionally prove that $\|P_G U\,|\phi_0, \emptyset\rangle - U P_G\,|\phi_0, \emptyset\rangle\| \leq t\|[P_G, P_t\mathcal{O}P_t]\|$. Combining these two inequalities yields the lifting lemma.

See Section 5.1 for more details.

2.6 Stronger lifting via conditional instability

The lifting lemma implies that to prove soundness of the Micali construction, it suffices to bound the instability of the Micali database game. Unfortunately, the instability of the Micali database game is actually large, even given the query bound. For example, suppose that D is a database containing Merkle trees for many different proof strings, but each of these Merkle trees has (miraculously) the same root due to collisions. Then, the probability that querying the root yields a good randomness for the underlying PCP verifier is large, because the answer to the query only needs to be a good random string for any one of the many proofs that D contains.

This counterexample, however, should not be of concern because it relies on the database having many collisions, and we have already argued that creating even a single collision in the database is difficult. To deal with this issue, we introduce the notion of *conditional instability*: $\mathbf{I}(\mathcal{P} \mid \mathcal{Q}, t)$. This is a refined notion of instability that allows us to condition on events, e.g., that the database has no collisions. Our main technical contribution is the following stronger variant of Lemma 2.3.

Definition 2.5. *A database property $\mathcal{P}$ is a set of databases. The complement of $\mathcal{P}$ is $\bar{\mathcal{P}}$.*

Lemma 2.6 (Lifting lemma)**.** *For any base game G and database property $\mathcal{Q}$,*

$$\omega^*_{\mathsf{D}}(G, t) \leq O\Big(t^2 \cdot \big(\mathbf{I}(\mathcal{P}_G \mid \bar{\mathcal{Q}}, t) + \mathbf{I}(\mathcal{Q}, t)\big)\Big) \ .$$

In particular, combining the above with Lemma 2.2, we get

$$\omega^*_{\mathsf{O}}(G, t) \leq O\Big(t^2 \cdot \big(\mathbf{I}(\mathcal{P}_G \mid \bar{\mathcal{Q}}, t) + \mathbf{I}(\mathcal{Q}, t)\big) + k/2^n\Big) \ .$$

The above statement is an "instability analogue" of the standard fact that for any two events E_1 and E_2, $\Pr[E_1] \leq \Pr[E_1 \cup E_2] \leq \Pr\big[E_1 \mid \bar{E_2}\big] + \Pr[E_2]$.

The proof of Lemma 2.6 has three steps. First, we relax the database game $\mathcal{P}_G$ so that the adversary wins if the database is in $\mathcal{P}_G \cup \mathcal{Q}$. Clearly, winning the relaxed game is only easier than the original database game. Lemma 2.3 then implies that $\omega^*_{\mathsf{D}}(G, t) \leq O\Big(t^2 \cdot \mathbf{I}(\mathcal{P}_G \cup \mathcal{Q}, t)\Big)$. Finally, we show that for any two database properties $\mathcal{P}$ and $\mathcal{Q}$ it holds that $\mathbf{I}(\mathcal{P} \cup \mathcal{Q}, t) \leq \mathbf{I}(\mathcal{P} | \bar{\mathcal{Q}}, t) + \mathbf{I}(\mathcal{Q}, t)$, which completes the proof.

We remark that Lemma 2.6 cannot be proved by simply arguing that $\mathbf{I}(\mathcal{P}, t) \leq \mathbf{I}(\mathcal{P} \cup \mathcal{Q}, t)$ and then applying Lemma 2.3. This is because $\mathbf{I}(\mathcal{P}, t)$ and $\mathbf{I}(\mathcal{P} \cup \mathcal{Q}, t)$ are in general *incomparable* (see Proposition 5.12 for examples).

See Section 5.2 for more details.

2.7 Instability of the Micali oracle game

Armed with our lifting lemma, establishing the quantum security of the Micali construction is now relatively straightforward. Let $\mathcal{P}_{\mathsf{Mic}}$ be the database property for the Micali game, and let $\bar{\mathcal{P}}_{\mathsf{col}}$ be the no-collision property (the set of databases that do not contain collisions). We show that, for a random oracle of the form $h\colon \{0,1\}^{2\lambda} \to \{0,1\}^{\lambda}$,

$$\mathbf{I}(\mathcal{P}_{\mathsf{col}}, t) < t/2^{\lambda} \quad \text{and} \quad \mathbf{I}(\mathcal{P}_{\mathsf{Mic}} \mid \bar{\mathcal{P}}_{\mathsf{col}}, t) < \varepsilon + O(t/2^{\lambda}) \ .$$

Proving each of these inequalities is merely a classical argument.

- $\mathbf{I}(\mathcal{P}_{\mathsf{col}}, t)$: If D is a database containing less than t entries and has a collision, then adding an entry to D cannot remove the collision, so the probability that adding a new entry to D makes D have no collisions is 0. Let D be a database containing less than t entries and no collisions. For any new query x, adding the query-answer pair (x, y) to D for a random y will contain a collision with probability less than $t/2^{\lambda}$. Thus, $\mathbf{I}(\mathcal{P}_{\mathsf{col}}, t) < t/2^{\lambda}$.
- $\mathbf{I}(\mathcal{P}_{\mathsf{Mic}} \mid \bar{\mathcal{P}}_{\mathsf{col}}, t)$: It is impossible to go from a database D in $\mathcal{P}_{\mathsf{Mic}}$ to a database D not in $\mathcal{P}_{\mathsf{Mic}}$ by adding entries. Let D be a database not in $\mathcal{P}_{\mathsf{Mic}}$ containing less than t entries that contains no collisions. There are two ways to make D in $\mathcal{P}_{\mathsf{Mic}}$: either the new query is for the randomness of the PCP verifier in the Micali construction, in which case this finds a good choice of randomness with probability at most ε, or the new query extends one of the Merkle trees that the adversary is constructing. To extend the Merkle tree the adversary must find a pre-image, which happens with probability less than $O(t/2^{\lambda})$. Hence, $\mathbf{I}(\mathcal{P}_{\mathsf{Mic}} \mid \bar{\mathcal{P}}_{\mathsf{col}}, t) < \varepsilon + O(t/2^{\lambda})$, completing the proof.

Combining these bounds on instability with the lifting lemma completes the proof of soundness, and completes a proof sketch for Theorem 1. See Section 6 for more details.

2.8 zkSNARKs in the QROM

We have so far discussed how to establish soundness of the Micali construction in the quantum setting. We now discuss how to further establish zero knowledge and proof of knowledge, obtaining the first zkSNARKs secure in the quantum random oracle model (and thereby proving Theorem 2).

Zero knowledge. In the classical setting, the Micali construction achieves statistical zero knowledge provided the underlying PCP is (honest-verifier) statistical zero knowledge (and leaves in the Merkle tree are suitably salted to ensure statistical hiding of unrevealed leaves) [IMSX15; BCS16]. In the quantum setting, an analogous statement is immediate simply because the zero knowledge property holds against computationally unbounded verifiers *that make an unbounded number of queries to the random oracle*, and any quantum verifier can be simulated by an unbounded verifier.

Proof of knowledge. In the classical setting, the Micali construction achieves proof of knowledge provided the underlying PCP is a proof of knowledge [Val08]. The quantum analogue of this statement, however, does *not* immediately follow from our soundness analysis. Recall that our

strategy was to bound the instability of the Micali property for $x \notin \mathcal{L}$, conditioned on no collisions. But when $x \in \mathcal{L}$ this approach will not work, because the instability of the Micali property even conditioned on the absence of collisions is 1 (as witnessed by the existence of the honest prover).

Nevertheless, the tools that we develop in this work are flexible enough that we can apply them to also establish proof of knowledge. We consider the following natural extractor strategy: run the prover until completion, and measure the database. Then, for each entry in the database, try to extract a PCP proof rooted at that entry, and then run the PCP extractor on this proof.

Let $\mathcal{P}$ be the set of databases D where there exists a root rt such that D wins the Micali game with a SNARG proof rooted at rt, but the PCP extractor does not extract a valid witness from the PCP proof rooted at rt. If the prover wins the Micali game but the extractor fails, then D must be in $\mathcal{P}$. We then argue that $\mathbf{I}(\mathcal{P} \mid \bar{\mathcal{P}}_{\mathsf{col}}, t)$ is at most $\mathsf{k} + O(t/2^\lambda)$, where k is the knowledge error of the underlying PCP. Intuitively, this is because if the PCP extractor fails to extract a witness from the PCP proof Π rooted at rt, then Π convinces the verifier with probability at most k, and hence the probability of finding good randomness for Π is at most k. Combining this with Lemma 2.6 implies that the probability that the prover wins the Micali game but the extractor fails is at most $O(t^2\mathsf{k} + t^3/2^\lambda)$. Hence, if μ is the probability that the prover wins the Micali game, then the probability that the extractor succeeds is at least $\Omega(\mu - t^2\mathsf{k} - t^3/2^\lambda)$.

See Section 7 for more details.

2.9 The BCS construction: succinct arguments beyond Micali

We apply our techniques to prove post-quantum security of the BCS construction [BCS16], when the underlying public-coin IOP satisfies a notion of soundness achieved by many protocols of practical interest. The notion is *round-by-round soundness*, and was introduced for IPs in [CCHLRR18] for the purposes of facilitating proofs of security of the Fiat–Shamir transformation for correlation-intractable hash functions. The notion can be extended in a straightforward way to any IOP, and this is the notion that we consider in this work. We further show that if the underlying IOP is honest-verifier zero knowledge and/or has round-by-round proof of knowledge, then the BCS argument inherits these properties. Round-by-round proof of knowledge is a type of knowledge property that is analogous to round-by-round soundness (and is also achieved by many protocols of practical interest). Below we sketch our analysis; see Section 8 for details.

Soundness. An IOP has round-by-round soundness if, for any partial transcript tr of the protocol, one can tell if tr is "doomed", i.e., that it is highly unlikely to be accepted by the verifier when completed to a full transcript; a doomed full transcript is never accepted by the verifier.

By the lifting lemma, in order to prove the post-quantum security of the BCS construction it suffices to bound the conditional instability of the database property $\mathcal{P}$, where $D \in \mathcal{P}$ if D contains a partial transcript where the last verifier message has flipped the transcript from "doomed" to "not doomed". We argue that $\mathbf{I}(\mathcal{P} \mid \bar{\mathcal{P}}_{\mathsf{col}}, t) < \epsilon + O(t/2^\lambda)$, where ϵ is the round-by-round soundness error of the IOP. The proof is similar to the proof for the Micali construction. If $D \notin \mathcal{P}$, there are two ways to add an entry and make $D \in \mathcal{P}$: either the new query is for the randomness of the next verifier message in the IOP for some doomed transcript tr, in which case we find a message that makes tr not doomed with probability ϵ; or the new query extends one of the Merkle trees that the adversary is constructing, which happens with probability less than $O(t/2^\lambda)$ as this implies finding a pre-image. Hence, $\mathbf{I}(\mathcal{P} \mid \bar{\mathcal{P}}_{\mathsf{col}}, t) < \epsilon + O(t/2^\lambda)$, which completes the proof.

Zero knowledge. As in the case of Micali, zero knowledge is straightforward, as the BCS construc-

tion classically achieves *statistical* zero knowledge when the IOP is honest-verifier zero knowledge.

Proof of knowledge. Analogously to our analysis of the Micali construction, we define a property $\mathcal{Q}$, where $D \in \mathcal{Q}$ if D contains a partial transcript that is in $\mathcal{P}$ but the BCS extractor fails to extract a valid witness. We then argue that $\mathbf{I}(\mathcal{Q} \mid \bar{\mathcal{P}}_{\mathsf{col}}) < \mathsf{k} + O(t/2^{\lambda})$, where k is the round-by-round knowledge error of the IOP; the proof of this fact is similar to the proof of soundness. We conclude that if the prover causes the verifier to accept with probability at least μ, then the probability that the extractor succeeds is at least $\Omega(\mu - t^2\mathsf{k} - t^3/2^{\lambda})$.

3 Preliminaries

We denote by $\mathcal{R}$ a binary relation of instance-witness pairs $(\mathtt{x}, \mathtt{w})$, and by $\mathcal{L}(\mathcal{R})$ its corresponding language, which is the set $\{\mathtt{x} \mid \exists\, \mathtt{w} \text{ s.t. } (\mathtt{x}, \mathtt{w}) \in \mathcal{R}\}$. We denote by $f\colon X \to Y$ a function from a set X to a set Y; similarly, we denote by $f\colon X \rightharpoonup Y$ a *partial* function from a set X to a set Y, i.e., a function $f\colon X \to Y \cup \{\perp\}$, where $\perp \notin Y$ is a special symbol indicating that $f(x)$ is undefined.

3.1 Quantum notation

We briefly recall standard quantum notation. We let $|\phi\rangle$ denote an arbitrary quantum state, and let $|x\rangle$ denote an element of the standard (computational) basis. The norm of a state $|\phi\rangle$ is $\||\phi\rangle\| := \sqrt{\langle\phi|\phi\rangle}$. In general, the states that we consider will have norm 1. The operator norm of an operator A is $\|A\| := \max_{|\phi\rangle:\||\phi\rangle\|=1} \|A\,|\phi\rangle\|$. Note that if A is unitary then $\|A\| = 1$. The commutator of two operators A and B is $[A, B] := AB - BA$. The following proposition relates operator norms and commutators.

Proposition 3.1. *Let A, B, C be operators with $\|B\|, \|C\| \leq 1$. Then*

$$\|[A, BC]\| \leq \|[A, B]\| + \|[A, C]\| \ .$$

Proof. By definition, $[A, BC] = ABC - BCA = ABC - BAC + BAC - BCA = [A, B]C + B[A, C]$. Therefore, $\|[A, BC]\| \leq \|[A, B]C\| + \|B[A, C]\| \leq \|[A, B]\| + \|[A, C]\|$, as $\|B\|, \|C\| \leq 1$. □

A projector P is an idempotent linear operator (i.e., $P^2 = P$). Throughout, we will only consider orthogonal projectors of the form $P_S := \sum_{x \in S} |x\rangle\langle x|$, where S is a set of binary strings. Measuring a state $|\phi\rangle$ in the standard basis results in an output that is in S with probability equal to $\|P_S\,|\phi\rangle\|^2$. Since all P_S are diagonal in the same basis, they commute with each other. Note that for any non-zero orthogonal projector P it holds that $\|P\| = 1$. In particular, since $\|AB\| \leq \|A\|\|B\|$, we see that if A is the product of projectors and unitaries then $\|A\| \leq 1$.

3.2 Oracle algorithms

Let $f\colon \{0,1\}^m \to \{0,1\}^n$ be a function. The standard way to model oracle access to f in the quantum setting is via a unitary operator O_f that acts as $|x, y\rangle \mapsto |x, y \oplus f(x)\rangle$ for all $x \in \{0,1\}^m$ and $y \in \{0,1\}^n$. We label the input and output registers X and Y, respectively.

A *t-query quantum oracle algorithm* $\mathcal{A}$ is specified via $m, n \in \mathbb{N}$, t unitary operators $A_1, \ldots, A_t$ and an initial state $|\phi_0\rangle$ on four registers $\mathsf{X}, \mathsf{Y}, \mathsf{S}, \mathsf{T}$. The register X is on m qubits and is for queries to the oracle; the register Y is on n qubits and is for answers from the oracle; the register S is for the output of $\mathcal{A}$; and the register T is for scratch space of $\mathcal{A}$. The initial state $|\phi_0\rangle$ and unitary operators A_i need not be efficiently computable.

We write $\left|\mathcal{A}^f\right\rangle$ to denote $A_t O_f A_{t-1} O_f \cdots A_1 O_f\,|\phi_0\rangle$, the final state of the adversary before measurement. (We implicitly extend O_f to act as the identity on S, T.) We write $\mathcal{A}^f$ to denote the random variable which is the outcome of measuring the register S of $\left|\mathcal{A}^f\right\rangle$ in the computational basis. This is the output of $\mathcal{A}$ when accessing the oracle f.

A *random oracle* is a function $h\colon \{0,1\}^m \to \{0,1\}^n$ sampled from $\mathcal{U}(m, n)$, the uniform distribution over functions from $\{0,1\}^m$ to $\{0,1\}^n$. We write $h \leftarrow \mathcal{U}(m, n)$ to say that h is sampled from $\mathcal{U}(m, n)$. In the quantum random oracle model [BDFLSZ11], we study $\mathcal{A}^h$ for $h \leftarrow \mathcal{U}(m, n)$.

3.3 Non-interactive arguments in the quantum random oracle model

Let $(\mathcal{P}, \mathcal{V})$ be two polynomial-time (classical) algorithms, known as the prover and verifier. We say that $(\mathcal{P}, \mathcal{V})$ is a *non-interactive argument in the quantum random oracle model* (QROM) with soundness error ϵ for a relation $\mathcal{R}$ if it satisfies the following properties.

- **Completeness.** For every $(\mathtt{x}, \mathtt{w}) \in \mathcal{R}$ and function $h \in \mathcal{U}(2\lambda, \lambda)$, $\mathcal{P}^h(\mathtt{x}, \mathtt{w})$ outputs a (classical) proof string π for which $\mathcal{V}^h(\mathtt{x}, \pi) = 1$.
- **Soundness.** For every $\mathtt{x} \notin \mathcal{L}(\mathcal{R})$ and t-query quantum oracle algorithm $\tilde{\mathcal{P}}$, the probability over a function $h \leftarrow \mathcal{U}(2\lambda, \lambda)$ and (classical) proof string $\tilde{\pi} \leftarrow \tilde{\mathcal{P}}^h$ that $\mathcal{V}^h(\mathtt{x}, \tilde{\pi}) = 1$ is at most $\epsilon(t, \lambda)$.

We say that $(\mathcal{P}, \mathcal{V})$ has *argument size* s if a proof π output by $\mathcal{P}^h(\mathtt{x}, \mathtt{w})$ consists of $s(|\mathtt{x}|)$ bits.

We also consider non-interactive arguments that additionally achieve *proof of knowledge* and *zero knowledge*. The first property will hold against query-bounded adversaries (that are otherwise all-powerful), while the second property will hold against unbounded adversaries (and in particular need not refer to quantum algorithms). We define both of these properties below.

Knowledge. The non-interactive argument $(\mathcal{P}, \mathcal{V})$ is an *argument of knowledge* with extraction probability κ if there exists a polynomial-time quantum extractor $\mathcal{E}$ such that, for every instance $\mathtt{x}$ and t-query quantum oracle algorithm $\tilde{\mathcal{P}}$, if, over a random oracle $h \leftarrow \mathcal{U}(2\lambda, \lambda)$, for $\pi := \tilde{\mathcal{P}}^h$ it holds that $\mathcal{V}^h(\mathtt{x}, \pi) = 1$ with probability μ, the probability that $\mathcal{E}^{\tilde{\mathcal{P}}}(\mathtt{x}, 1^t, 1^\lambda)$ outputs a valid witness for $\mathtt{x}$ is at least $\kappa(t, \mu, \lambda)$. Here the notation $\mathcal{E}^{\tilde{\mathcal{P}}}$ denotes that $\mathcal{E}$ has black-box access to $\tilde{\mathcal{P}}$ as defined by Unruh [Unr17]. Informally, this means that if $\tilde{\mathcal{P}} = (A_1, \ldots, A_t)$ with initial state $|\phi_0\rangle$, then $\mathcal{E}$ is given an auxiliary register containing $|\phi_0\rangle$ and may apply, in addition to any efficient quantum operation, any A_i to any of its registers.

Zero knowledge. The non-interactive argument $(\mathcal{P}, \mathcal{V})$ has (statistical) zero knowledge if there exists a probabilistic polynomial-time simulator $\mathcal{S}$ such that for every instance-witness pair $(\mathtt{x}, \mathtt{w}) \in \mathcal{R}$ the distributions below are statistically close (as a function of λ):

$$\left\{ (h, \pi) \;\middle|\; \begin{array}{r} h \leftarrow \mathcal{U}(2\lambda, \lambda) \\ \pi \leftarrow \mathcal{P}^h(\mathtt{x}, \mathtt{w}) \end{array} \right\} \quad \text{and} \quad \left\{ (h[\mu], \pi) \;\middle|\; \begin{array}{r} h \leftarrow \mathcal{U}(2\lambda, \lambda) \\ (\mu, \pi) \leftarrow \mathcal{S}^h(\mathtt{x}) \end{array} \right\} .$$

Above, $h[\mu]$ is the function that, on input x, equals $\mu(x)$ if μ is defined on x, or $h(x)$ otherwise. This definition uses explicitly-programmable random oracles [BR93]. (Non-interactive zero knowledge with non-programmable random oracles is impossible for non-trivial languages [Pas03; BCS16].)

Succinctness for non-deterministic time. A zkSNARK for $\mathsf{NTIME}(T(n))$ in the QROM is a non-interactive argument for $\mathsf{NTIME}(T(n))$ in the QROM such that: (a) it has (statistical) zero knowledge; (b) it has extraction probability $\mathsf{poly}(\mu, 1/t) - \mathsf{poly}(\mu, t)/2^\lambda$; (c) arguments have size $\mathsf{poly}(\lambda, \log T(n))$, the prover runs in time $\mathsf{poly}(\lambda, n, T(n))$, and the verifier runs in time $\mathsf{poly}(\lambda, n, \log T(n))$.

3.4 Probabilistically checkable proofs

A *probabilistically checkable proof* (PCP) for a relation $\mathcal{R}$ with soundness error ϵ, proof length ℓ, and alphabet Σ is a pair of polynomial-time algorithms $(\mathbf{P}, \mathbf{V})$ for which the following holds.

- **Completeness.** For every instance-witness pair $(\mathtt{x}, \mathtt{w}) \in \mathcal{R}$, $\mathbf{P}(\mathtt{x}, \mathtt{w})$ outputs a proof string $\Pi\colon [\ell] \to \Sigma$ such that $\Pr\big[\mathbf{V}^\Pi(\mathtt{x}) = 1\big] = 1$.

- **Soundness.** For every instance $\mathtt{x} \notin \mathcal{L}(\mathcal{R})$ and proof string $\Pi\colon [\ell] \to \Sigma$, $\Pr\big[\mathbf{V}^{\Pi}(\mathtt{x}) = 1\big] \leq \epsilon$.

The quantities ϵ, ℓ, Σ can be functions of the instance size $|\mathtt{x}|$. Probabilities are taken over the randomness of $\mathbf{V}$. The *randomness complexity* is the number of random bits used by $\mathbf{V}$, and the *query complexity* q is the number of locations of Π read by $\mathbf{V}$. (Both can be functions of $|\mathtt{x}|$.)

We also consider PCPs that achieve *proof of knowledge* and (honest-verifier) *zero knowledge*. We define both of these properties below.

Proof of knowledge. The PCP $(\mathbf{P}, \mathbf{V})$ has knowledge error k if there exists a polynomial-time extractor $\mathbf{E}$ such that for every instance $\mathtt{x}$ and proof string $\Pi\colon [\ell] \to \Sigma$ if $\Pr[\mathbf{V}(\mathtt{x}, \Pi) = 1] > \mathsf{k}$ then $\mathbf{E}(\mathtt{x}, \Pi)$ outputs a valid witness for $\mathtt{x}$.

Zero knowledge. The PCP $(\mathbf{P}, \mathbf{V})$ is (perfect) honest-verifier zero knowledge if there exists a probabilistic polynomial-time simulator $\mathbf{S}$ such that for every instance-witness pair $(\mathtt{x}, \mathtt{w}) \in \mathcal{R}$ the view of $\mathbf{V}(\mathtt{x})$ when given access to a proof string sampled as $\Pi \leftarrow \mathbf{P}(\mathtt{x}, \mathtt{w})$ equals the view of $\mathbf{V}(\mathtt{x})$ when given access to $\mathbf{S}(\mathtt{x})$. In the latter case, $\mathbf{S}(\mathtt{x})$ adaptively answers queries received from $\mathbf{V}(\mathtt{x})$.

3.5 Databases

A *database* mapping X to Y is a partial function $D\colon X \rightharpoonup Y$. The support of a database D is $\mathrm{supp}(D) := \{x \in X\colon D(x) \neq \perp\}$ and its image is $\mathrm{im}(D) := \{D(x)\colon x \in \mathrm{supp}(D)\}$. The size of a database is the size of its support: $|D| := |\mathrm{supp}(D)|$. Given two databases D and D', we write $D \subseteq D'$ if $\mathrm{supp}(D) \subseteq \mathrm{supp}(D')$ and $D(x) = D'(x)$ for every $x \in \mathrm{supp}(D)$.

We define two operations on databases, corresponding to deletions and insertions. Given a database D, input values $x, x' \in X$, and output value $y \in Y$, we define the two databases

$$(D - x)(x') := \begin{cases} \perp & \text{if } x = x' \\ D(x') & \text{if } x \neq x' \end{cases} \quad \text{and} \quad (D + [x \mapsto y])(x') := \begin{cases} y & \text{if } x = x' \\ D(x') & \text{if } x \neq x' \end{cases}.$$

For $D\colon \{0,1\}^m \rightharpoonup \{0,1\}^n$ and $t \in \mathbb{N}$ with $|D| \leq t \leq 2^m$, we define the pure quantum state

$$|D_t\rangle \;:=\; \big|x_1, y_1, \ldots, x_{|D|}, y_{|D|}\big\rangle \otimes |\perp, 0^n\rangle^{\otimes(|D|-t)}$$

where $x_1, \ldots, x_{|D|}$ is the lexicographic ordering of $\mathrm{supp}(D)$ and $y_i := D(x_i)$ for each $i \in [|D|]$. We will write $|D\rangle$ for $|D_t\rangle$ when the bound t is clear from context.

3.6 Compressed phase oracle

The standard method to encode a function $h\colon \{0,1\}^m \to \{0,1\}^n$ as a quantum operation is the unitary matrix O_h defined in Section 3.2, which acts as $|x, y\rangle \mapsto |x, y \oplus h(x)\rangle$. Another method is to encode h in the *phase* of a quantum state, via the unitary matrix O'_h that acts as $|x, u\rangle \mapsto (-1)^{u \cdot h(x)} |x, u\rangle$. These two encodings are equivalent under an efficient change of basis: $O_h = (I^m \otimes H^n) O'_h (I^m \otimes H^n)$ where I^m is the identity on the first m qubits and H^n is the Hadamard transformation on the other n qubits. Thus, choosing between the *standard oracle* O_h or the *phase oracle* O'_h is a matter of convenience. For example, the Deutsch–Josza algorithm [DJ92] is easier to describe with a standard oracle, while Grover's algorithm [Gro96] is easier with a phase oracle.

In this paper it is more convenient to *always work with phase oracles*. All quantum query algorithms will thus have an oracle phase register U instead of the oracle answer register Y. Moreover, since h is sampled at random from the set of all functions from m bits to n bits, we follow Zhandry

[Zha19] and extend the adversary's initial state with a random superposition of all functions h, which represents a purification of the adversary's mixed state relative to the random oracle.

In fact, instead of considering a superposition of functions h, we will consider a superposition of databases D, according to the *compressed oracle* formalism of [Zha19]. Specifically, throughout this paper we will only deal with the *compressed phase oracle* with m input bits and n output bits, which we denote by $\mathcal{O}$. We fix the database query bound of the compressed oracle to be t in advance. For the purposes of this paper, we will only use the fact that $\mathcal{O}$ is a certain unitary matrix, indistinguishable from a real random oracle, whose action is given by the following lemma, which we prove in Appendix A for completeness. We refer the reader to [Zha19] for more details.

Lemma 3.2 ([Zha19])**.** *The compressed phase oracle $\mathcal{O}$ (with query bound t) acts on a quantum state $|x,u,z,D\rangle$, where $x \in \{0,1\}^m$, $u \in \{0,1\}^n$, $z \in \{0,1\}^*$, and $D\colon \{0,1\}^m \rightharpoonup \{0,1\}^n$ is a database with $|D| \leq t$, as follows.*

- *If $|D| = t$ or $u = 0^n$, then $\mathcal{O}|x,u,z,D\rangle = (-1)^{u\cdot D(x)}|x,u,z,D\rangle$, where $u \cdot \perp := 0$.*
- *If $D(x) = \perp$, $|D| < t$, and $u \neq 0^n$, then $\mathcal{O}|x,u,z,D\rangle = |x,u,z\rangle \otimes |\phi\rangle$ where*

$$|\phi\rangle := \frac{1}{\sqrt{2^n}} \sum_{y\in\{0,1\}^n} (-1)^{u\cdot y} |D + [x \mapsto y]\rangle \quad .$$

- *If $D(x) \neq \perp$, $|D| < t$, and $u \neq 0^n$, then $\mathcal{O}|x,u,z,D\rangle = |x,u,z\rangle \otimes |\phi\rangle$ where*

$$|\phi\rangle := (-1)^{u\cdot D(x)}|D\rangle + \frac{(-1)^{u\cdot D(x)}}{\sqrt{2^n}}|D-x\rangle + \frac{1}{2^n}\sum_{y\in\{0,1\}^n}\left(1-(-1)^{u\cdot y}-(-1)^{u\cdot D(x)}\right)|D-x+[x\mapsto y]\rangle \quad .$$

Given a quantum algorithm $\mathcal{A}$ described by unitaries $A_1,\ldots,A_t$ and initial state $|\phi_0\rangle$, we write $|\mathsf{Sim}^*(\mathcal{A})\rangle$ to represent the final state of $\mathcal{A}$ before measurement when simulated using $\mathcal{O}$ as the oracle. Formally, $|\mathsf{Sim}^*(\mathcal{A})\rangle = A_t\mathcal{O}\cdots A_1\mathcal{O}|\phi_0,\emptyset\rangle$, where $\emptyset$ denotes that the D register holds the empty database with t slots, and we implicitly extend each A_i to act as the identity on D.

The following lemma of [Zha19] shows simulating $\mathcal{A}$ by using $\mathcal{O}$ as the oracle is perfectly indistinguishable from running $\mathcal{A}$ with access to a random oracle.

Lemma 3.3 ([Zha19, Lemma 4])**.** *For any quantum oracle algorithm $\mathcal{A}$ making at most t queries,*

$$\mathrm{Tr}_{\mathsf{D}}(|\mathsf{Sim}^*(\mathcal{A})\rangle\langle\mathsf{Sim}^*(\mathcal{A})|) = \frac{1}{(2^n)^{2^m}} \sum_{h\colon \{0,1\}^m\to\{0,1\}^n} |\mathcal{A}^h\rangle\langle\mathcal{A}^h| \quad .$$

I.e., $|\mathsf{Sim}^*(\mathcal{A})\rangle$ purifies the mixed state of $\mathcal{A}$ when interacting with a random oracle $h \leftarrow \mathcal{U}(m,n)$.

The notation Tr_{D} denotes the partial trace over the D (database) register, defined as the unique linear operator such that $\mathrm{Tr}_{\mathsf{D}}(|a\rangle\langle a|_{\mathsf{Z}} \otimes |b\rangle\langle b|_{\mathsf{D}}) := \langle b|b\rangle\, |a\rangle\langle a|_{\mathsf{Z}}$ for all vectors $|a\rangle, |b\rangle$. Here Z denotes all the registers of the adversary.

4 From oracle games to database games

In this section we formulate three distinct, but closely related, types of games that represent different hybrids in our proof of security. The material in this section is mostly a conceptualization of prior work, and forms the starting point for the technical contributions presented in later sections.

The first type of games that we consider are *oracle games*, which correspond to when an adversary "in the real world" is granted access to a random oracle and must produce a certain output in order to win. The second type of games are *simulated oracle games*, which differ in that the adversary is executed by a probabilistic process that efficiently simulates the random oracle (without actually sampling a random function). The third type of games are *database games*, which differ in that we only consider properties of the query-answer pairs recorded in the simulation.

We shall see that the *value* (i.e., the maximum winning probability) across all these games are closely related. In particular, in order to bound the value of an oracle game it suffices to bound the value of the corresponding database game. For this reason in Section 5 we shall focus our attention on developing new techniques to bound the values of database games.

We now proceed to formalize the foregoing notions. We find it helpful to structure our exposition by first discussing the special case of *classical* adversaries (in Section 4.1), and then discussing the general case of *quantum* adversaries (Section 4.2). In either case, we will derive each type of game from a *base game* that represents a target set of "winning outputs".

Definition 4.1. *Let A, B, C be finite sets, $k \in \mathbb{N}$. A* **base game** *is a set of tuples $G \subseteq A^k \times B^k \times C$.*

We will use $\mathcal{C}_t$ to denote the set of all classical algorithms that make at most t queries to an oracle, and $\mathcal{C}_t^*$ to denote the set of all quantum algorithms that make at most t queries to an oracle.

4.1 The case of classical adversaries

We associate to a base game G three classical notions: the *classical oracle game* G_{O}, the *classical simulated oracle game* G_{S}, and the *classical database game* G_{D}. These notions are distinct, but they are also closely related as shown in Lemma 4.5 below.

A classical algorithm $\mathcal{A}$ plays the *classical oracle game* of G as follows: $\mathcal{A}$ has access to a random oracle $h\colon \{0,1\}^m \to \{0,1\}^n$ and wins if it outputs $(\mathbf{a}, \mathbf{b}, c) \in G$ with $h(a_i) = b_i$ for all $i \in [k]$.

Definition 4.2. *The* **value of the classical oracle game** *of G is*

$$\omega_{\mathsf{O}}(G,t) := \max_{\mathcal{A}\in\mathcal{C}_t} \Pr[\mathcal{A} \textit{ wins } G_{\mathsf{O}}] = \max_{\mathcal{A}\in\mathcal{C}_t} \Pr\left[\begin{array}{c|c} (\mathbf{a},\mathbf{b},c) \in G \textit{ and} & h \leftarrow \mathcal{U}(m,n) \\ \forall\, i \in [k]\,,\ h(a_i) = b_i & (\mathbf{a},\mathbf{b},c) \leftarrow \mathcal{A}^h \end{array} \right] .$$

Next, we discuss the *classical simulated oracle game* of G, which is defined via a simulator. We denote by $\mathsf{Sim}(\mathcal{A})$ the probabilistic process that executes the classical algorithm $\mathcal{A}$ while simulating a random oracle with random consistent answers,[5] and then outputs $((\mathbf{a}, \mathbf{b}, c), D)$ where $(\mathbf{a}, \mathbf{b}, c)$ is the output of $\mathcal{A}$ and D is the database containing all the simulated query-answer pairs. The classical algorithm $\mathcal{A}$ wins if $((\mathbf{a}, \mathbf{b}, c), D)$ is such that $(\mathbf{a}, \mathbf{b}, c) \in G$ and $D(a_i) = b_i$ for all $i \in [k]$.

Definition 4.3. *The* **value of the classical simulated oracle game** *of G is*

$$\omega_{\mathsf{S}}(G,t) := \max_{\mathcal{A}\in\mathcal{C}_t} \Pr[\mathcal{A} \textit{ wins } G_{\mathsf{S}}] = \max_{\mathcal{A}\in\mathcal{C}_t} \Pr\left[\begin{array}{c|c} (\mathbf{a},\mathbf{b},c) \in G \textit{ and} & \big((\mathbf{a},\mathbf{b},c), D\big) \leftarrow \mathsf{Sim}(\mathcal{A}) \\ \forall\, i \in [k]\,,\ D(a_i) = b_i & \end{array} \right] .$$

[5] Initialize an empty database $D\colon \{0,1\}^m \rightharpoonup \{0,1\}^n$. Whenever A queries the oracle at $x \in \{0,1\}^m$, if $x \in \mathrm{supp}(D)$ then answer $D(x)$; else if $x \notin \mathrm{supp}(D)$ then answer with a random $y \in \{0,1\}^n$ and update as $D := D + [x \mapsto y]$.

Finally, we discuss the *classical database game* of G. We again simulate the classical algorithm $\mathcal{A}$ using $\mathsf{Sim}(\mathcal{A})$ to obtain $((\mathbf{a},\mathbf{b},c),D)$, but now we *discard* the tuple $(\mathbf{a},\mathbf{b},c)$ and only consider the database D. Concretely, the classical algorithm $\mathcal{A}$ wins if there *exists* a tuple $(\mathbf{a}',\mathbf{b}',c')$, possibly different from $(\mathbf{a},\mathbf{b},c)$, such that $(\mathbf{a}',\mathbf{b}',c') \in G$ and $D(a'_i) = b'_i$ for all $i \in [k]$.

Definition 4.4. *The* **value of the classical database game** *of G is*

$$\omega_{\mathsf{D}}(G,t) := \max_{\mathcal{A}\in\mathcal{C}_t} \Pr[\mathcal{A} \text{ wins } G_{\mathsf{D}}] = \max_{\mathcal{A}\in\mathcal{C}_t} \Pr\left[\begin{array}{c} \exists\,(\mathbf{a}',\mathbf{b}',c') \in G \text{ s.t.} \\ \forall\, i \in [k]\,,\ D(a'_i) = b'_i \end{array} \;\middle|\; \big((\mathbf{a},\mathbf{b},c),D\big) \leftarrow \mathsf{Sim}(\mathcal{A}) \right] \ .$$

The following lemma relates the three classical notions introduced above.

Lemma 4.5. *For every classical algorithm $\mathcal{A}$:*

$$\begin{aligned} \Pr[\mathcal{A} \text{ wins } G_{\mathsf{O}}] &\le \Pr[\mathcal{A} \text{ wins } G_{\mathsf{S}}] + 1/2^n \ , \\ \Pr[\mathcal{A} \text{ wins } G_{\mathsf{S}}] &\le \Pr[\mathcal{A} \text{ wins } G_{\mathsf{D}}] \ . \end{aligned}$$

In particular, the values of the three types of classical games are related:

$$\begin{gathered} \forall\, t \in \mathbb{N}\,,\ \omega_{\mathsf{O}}(G,t) \le \omega_{\mathsf{S}}(G,t) + 1/2^n \\ \text{and} \quad \omega_{\mathsf{S}}(G,t) \le \omega_{\mathsf{D}}(G,t) \ . \end{gathered}$$

Proof. We first compare G_{O} and G_{S}. Consider the adversary $\mathcal{A}'$ for G_{O} given by running $\mathsf{Sim}(\mathcal{A})$ and answering a query to x with $y := h(x)$. Let $(\mathbf{a},\mathbf{b},c)$ be the output of $\mathcal{A}'$, and let i be the first index such that $(a_i,b_i) \notin D$. Let E be the event that $h(a_i) = b_i$. Then $\Pr[E] \le 1/2^n$, so $\Pr[\mathcal{A} \text{ wins } G_{\mathsf{O}}] \le \Pr[\mathcal{A}' \text{ wins } G_{\mathsf{O}} \wedge \neg E] + \Pr[E]$. Observe that if $\mathcal{A}'$ wins G_{O} and $\neg E$ holds, then the simulated $\mathcal{A}$ wins G_{S}. Since the $\mathcal{A}$ simulated by $\mathcal{A}'$ is indistinguishable from $\mathsf{Sim}(\mathcal{A})$, it follows that $\Pr[\mathcal{A} \text{ wins } G_{\mathsf{O}}] \le \Pr[\mathcal{A} \text{ wins } G_{\mathsf{S}}] + 1/2^n$.

We also note that $\Pr[\mathcal{A} \text{ wins } G_{\mathsf{S}}] \le \Pr[\mathcal{A}' \text{ wins } G_{\mathsf{O}}] = \Pr[\mathcal{A} \text{ wins } G_{\mathsf{O}}]$, which shows additionally that the first inequality is tight up to the additive term of $1/2^n$.

We now compare G_{S} and G_{D}. Let $\mathcal{A}$ be an adversary, and let $((\mathbf{a},\mathbf{b},c),D)$ be the output of $\mathsf{Sim}(\mathcal{A})$. Observe that if $\mathcal{A}$ wins G_{S} then $\mathcal{A}$ wins G_{D} trivially, by taking $\mathbf{a}' = \mathbf{a}$, $\mathbf{b}' = \mathbf{b}$, and $c' = c$. Hence, $\Pr[\mathcal{A} \text{ wins } G_{\mathsf{S}}] \le \Pr[\mathcal{A} \text{ wins } G_{\mathsf{D}}]$.

We additionally note that $\omega_{\mathsf{S}}(G,t) = \omega_{\mathsf{D}}(G,t)$. This is because if $\mathcal{A}$ wins G_{D}, then $\mathcal{A}'$ (the adversary that runs $\mathsf{Sim}(\mathcal{A})$ and uses h to answer oracle queries) can search over all satisfying tuples in G and see if any of them win G_{S}. Since the database stored by $\mathcal{A}'$ is equal to the database stored by $\mathsf{Sim}(\mathcal{A}')$, it follows that $\mathcal{A}'$ wins G_{S} whenever $\mathcal{A}$ wins G_{D}, and hence $\omega_{\mathsf{S}}(G,t) = \omega_{\mathsf{D}}(G,t)$. □

4.2 The case of quantum adversaries

This section extends the notions and statements of Section 4.1 to the quantum setting, and provides a conceptualization of results of Zhandry [Zha19] that is useful in this paper. Analogously to the classical setting, we associate to a base game G three notions: its *quantum oracle game* G^*_{O}, its *quantum simulated oracle game* G^*_{S}, and its *quantum database game* G^*_{D}. Similarly to before, these notions are distinct, but they are also closely related as shown in Lemma 4.9 below. Unlike before, however, the notions and statements are *not* elementary because executing a quantum algorithm while recording its queries and answers to the oracle into a database D is not a simple problem.

A quantum algorithm A plays the *quantum oracle game* of G as follows: A has access to a random oracle $h\colon \{0,1\}^m \to \{0,1\}^n$ and wins if it outputs $(\mathbf{a},\mathbf{b},c) \in G$ with $h(a_i) = b_i$ for all $i \in [k]$.

Definition 4.6. *The* **value of the quantum oracle game** *of G is*

$$\omega^*_{\mathsf{O}}(G,t) := \max_{\mathcal{A}\in\mathcal{C}^*_t} \Pr[\mathcal{A} \textit{ wins } G^*_{\mathsf{O}}] = \max_{\mathcal{A}\in\mathcal{C}^*_t} \Pr\left[\begin{array}{c|c} (\mathbf{a},\mathbf{b},c)\in G \textit{ and} & h\leftarrow\mathcal{U}(m,n) \\ \forall\, i\in[k]\,,\ h(a_i)=b_i & (\mathbf{a},\mathbf{b},c)\leftarrow\mathcal{A}^h \end{array} \right] .$$

Next, we discuss the *quantum simulated oracle game* of G. One can simulate the interaction of a quantum algorithm $\mathcal{A}$ with a random oracle via the compressed phase oracle $\mathcal{O}$ of Zhandry [Zha19]. (See Section 3.6 for the definition of $\mathcal{O}$.) We denote by $\mathsf{Sim}^*(\mathcal{A})$ the probabilistic process that executes $\mathcal{A}$ while answering its queries with $\mathcal{O}$, and then outputs $((\mathbf{a},\mathbf{b},c),D)$ where $(\mathbf{a},\mathbf{b},c)$ is the output of $\mathcal{A}$ (obtained by measuring its output register in the computational basis) and D is the result of measuring the database registers in the computational basis after $\mathcal{A}$ has halted. The quantum algorithm $\mathcal{A}$ wins if $((\mathbf{a},\mathbf{b},c),D)$ is such that $(\mathbf{a},\mathbf{b},c)\in G$ and $D(a_i)=b_i$ for all $i\in[k]$.

Definition 4.7. *The* **value of the quantum simulated oracle game** *of G is*

$$\omega^*_{\mathsf{S}}(G,t) := \max_{\mathcal{A}\in\mathcal{C}^*_t} \Pr[\mathcal{A} \textit{ wins } G^*_{\mathsf{S}}] = \max_{\mathcal{A}\in\mathcal{C}^*_t} \Pr\left[\begin{array}{c|c} (\mathbf{a},\mathbf{b},c)\in G \textit{ and} & \big((\mathbf{a},\mathbf{b},c),D\big)\leftarrow\mathsf{Sim}^*(\mathcal{A}) \\ \forall\, i\in[k]\,,\ D(a_i)=b_i & \end{array} \right] .$$

Finally, we discuss the *quantum database game* of G. We again simulate the quantum algorithm $\mathcal{A}$ using $\mathsf{Sim}^*(\mathcal{A})$ to obtain $((\mathbf{a},\mathbf{b},c),D)$ but now we *discard* the tuple $(\mathbf{a},\mathbf{b},c)$ and only consider the database D. Concretely, the quantum algorithm $\mathcal{A}$ wins if there exists a tuple $(\mathbf{a}',\mathbf{b}',c)\in G$ such that $D(a'_i)=b'_i$ for all $i\in[k]$.

Definition 4.8. *The* **value of the quantum database game** *of G is*

$$\omega^*_{\mathsf{D}}(G,t) := \max_{\mathcal{A}\in\mathcal{C}^*_t} \Pr[\mathcal{A} \textit{ wins } G^*_{\mathsf{D}}] = \max_{\mathcal{A}\in\mathcal{C}^*_t} \Pr\left[\begin{array}{c|c} \exists\,(\mathbf{a}',\mathbf{b}',c')\in G \textit{ s.t.} & \big((\mathbf{a},\mathbf{b},c),D\big)\leftarrow\mathsf{Sim}^*(\mathcal{A}) \\ \forall\, i\in[k]\,,\ D(a'_i)=b'_i & \end{array} \right] .$$

The following lemma relates the three quantum notions introduced above.

Lemma 4.9 ([Zha19])**.** *For every quantum algorithm $\mathcal{A}$:*

$$\sqrt{\Pr[\mathcal{A} \textit{ wins } G^*_{\mathsf{O}}]} \le \sqrt{\Pr[\mathcal{A} \textit{ wins } G^*_{\mathsf{S}}]} + \sqrt{k/2^n} \ ,$$
$$\Pr[\mathcal{A} \textit{ wins } G^*_{\mathsf{S}}] \le \Pr[\mathcal{A} \textit{ wins } G^*_{\mathsf{D}}] \ .$$

In particular, the values of the three types of quantum games are related:

$$\forall\, t\in\mathbb{N}\,,\ \sqrt{\omega^*_{\mathsf{O}}(G,t)} \le \sqrt{\omega^*_{\mathsf{S}}(G,t)} + \sqrt{k/2^n}$$
$$\textit{and}\quad \omega^*_{\mathsf{S}}(G,t) \le \omega^*_{\mathsf{D}}(G,t) \ .$$

Proof. We first compare G^*_{O} and G^*_{S}. The fact that $\sqrt{\Pr[\mathcal{A} \text{ wins } G^*_{\mathsf{O}}]} \le \sqrt{\Pr[\mathcal{A} \text{ wins } G^*_{\mathsf{S}}]} + \sqrt{k/2^n}$ is a restatement of [Zha19, Lemma 5].

We now compare G^*_{S} and G^*_{D}. As in the classical setting, if $\mathcal{A}$ wins G^*_{S} then $\mathcal{A}$ wins G^*_{D} trivially, so $\Pr[\mathcal{A} \text{ wins } G^*_{\mathsf{S}}] \le \Pr[\mathcal{A} \text{ wins } G^*_{\mathsf{D}}]$. Note that in the classical case the value of the simulated oracle and database games are the same by a simple simulation argument, but in the quantum setting the simulation argument does not go through due to the no-cloning theorem. □

5 A lifting lemma for database games

In this section we show how to bound the value of a (classical or quantum) database game via the *instability* of the game, a purely classical quantity that we introduce in this paper. As we will see shortly, it is straightforward to argue that for any base game G (Definition 4.1), the value $\omega_{\mathsf{D}}(G,t)$ is at most t times the instability of G. The goal of this section is to to prove that the (quantum) value $\omega^*_{\mathsf{D}}(G,t)$ is at most t^2 times the instability of G. In particular, we enable *lifting* a bound on the (classical) instability of G to a bound on the (quantum) value $\omega^*_{\mathsf{D}}(G,t)$. Combining the lifting lemma with the fact that oracle games can be generically reduced to database games (as described in Section 4.2), we are able to establish the post-quantum security of the Micali construction solely by analyzing classical properties of it.

5.1 Database properties and the basic lifting lemma

A database property is a more general notion of a database game.

Definition 5.1. *A* **database property** $\mathcal{P}$ *is a set of databases* $D\colon X \rightharpoonup Y$*. The negation of* $\mathcal{P}$*, denoted* $\bar{\mathcal{P}}$*, is the set* $(X \rightharpoonup Y) \setminus \mathcal{P}$*.*

Given a base game, we define a corresponding database property as follows.

Definition 5.2. *The database property of a base game* $G \subseteq A^k \times B^k \times C$ *is*

$$\mathcal{P}_G := \{D : \exists\, (\mathbf{a}, \mathbf{b}, c) \in G \text{ with } D(a_i) = b_i \ \forall\, i \in [k]\} \ .$$

For a base game G, the database property $\mathcal{P}_G$ is closely related to the database game of G. This is because winning the database game is equivalent to the database outputted by $\mathsf{Sim}^*(\mathcal{A})$ being in $\mathcal{P}_G$. In particular, the following proposition holds.

Proposition 5.3. *For every base game* $G \subseteq A^k \times B^k \times C$ *and quantum algorithm* $\mathcal{A}$*,*

$$\Pr[\mathcal{A} \text{ wins } G^*_{\mathsf{D}}] = \Pr\left[D \in \mathcal{P}_G \;\middle|\; \big((\mathbf{a}, \mathbf{b}, c), D\big) \leftarrow \mathsf{Sim}^*(\mathcal{A})\right] \ .$$

We define the *flip probability* of a pair of database properties.

Definition 5.4. *The* **flip probability** $\mathrm{flip}(\mathcal{P} \to \mathcal{Q}, t)$ *from property* $\mathcal{P}$ *to property* $\mathcal{Q}$ *is the quantity*

$$\mathrm{flip}(\mathcal{P} \to \mathcal{Q}, t) := \max_{\substack{D\colon \{0,1\}^m \rightharpoonup \{0,1\}^n \\ |D| < t\,,\, D \in \mathcal{P}}} \ \max_{x \notin \mathrm{supp}(D)} \Pr_y \left[D + [x \mapsto y] \in \mathcal{Q}\right] \ ,$$

and $\mathrm{flip}(\emptyset \to \mathcal{Q}, t) := 0$*.*

Intuitively, this is the maximum probability over all databases $D \in \mathcal{P}$ with less than t entries that making an additional query puts $D \in \mathcal{Q}$. The following properties can be obtained easily from the above definition.

Proposition 5.5 (Properties of the flip probability)**.** *Let* $\mathcal{P}, \mathcal{P}', \mathcal{Q}, \mathcal{Q}'$ *be database properties.*

(i) If $\mathcal{P} \subseteq \mathcal{P}'$ *and* $\mathcal{Q} \subseteq \mathcal{Q}'$ *then* $\mathrm{flip}(\mathcal{P} \to \mathcal{Q}) \leq \mathrm{flip}(\mathcal{P}' \to \mathcal{Q}')$*.*

(ii) $\mathrm{flip}(\mathcal{P} \cup \mathcal{P}' \to \mathcal{Q}) = \max\big(\mathrm{flip}(\mathcal{P} \to \mathcal{Q}), \mathrm{flip}(\mathcal{P}' \to \mathcal{Q})\big)$*.*

(iii) $\mathrm{flip}(\mathcal{P} \to \mathcal{Q} \cup \mathcal{Q}') \leq \mathrm{flip}(\mathcal{P} \to \mathcal{Q}) + \mathrm{flip}(\mathcal{P} \to \mathcal{Q}')$.

The instability of a database property is the following classical quantity.

Definition 5.6. *The* **instability** $\mathbf{I}(\mathcal{P}, t)$ *of a database property* $\mathcal{P}$ *with query bound* t *is the maximum probability that, for any database* D *containing less than* t *queries, making one additional (classical) query changes whether or not* D *has the property* $\mathcal{P}$*. Formally, we let*

$$\mathbf{I}(\mathcal{P}, t) := \max\{\mathrm{flip}(\bar{\mathcal{P}} \to \mathcal{P}, t), \mathrm{flip}(\mathcal{P} \to \bar{\mathcal{P}}, t)\} \ .$$

Note that instability is symmetric: $\mathbf{I}(\mathcal{P}, t) = \mathbf{I}(\bar{\mathcal{P}}, t)$. There is a direct argument that shows that $\omega_{\mathsf{D}}(G, t)$ is bounded by $t\mathbf{I}(\mathcal{P}_G, t)$.[6] Similarly, our basic lifting lemma shows that $\omega^*_{\mathsf{D}}(G, t)$ is bounded by the instability of the database property $\mathcal{P}_G$. Thus, it lifts a *classical* notion to prove a bound on the *quantum* value of a database game.

Lemma 5.7 (Basic lifting lemma)**.** *For any base game* G*,*

$$\omega^*_{\mathsf{D}}(G, t) \leq t^2 \cdot 6\mathbf{I}(\mathcal{P}_G, t) \ .$$

Before we proceed to the proof of Lemma 5.7, we first introduce some quantum notation. Recall that we let $|\mathsf{Sim}^*(\mathcal{A})\rangle$ denote the final quantum state of the simulated adversary. Using the definition of measurement, we can express the probability that the final measured database D is in a database property $\mathcal{P}$ in terms of the state $|\mathsf{Sim}^*(\mathcal{A})\rangle$.

Proposition 5.8. *For every database property* $\mathcal{P}$ *and quantum adversary* $\mathcal{A}$*,*

$$\Pr\left[D \in \mathcal{P} \;\middle|\; \big((\mathbf{a}, \mathbf{b}, c), D\big) \leftarrow \mathsf{Sim}^*(\mathcal{A})\right] = \|P|\mathsf{Sim}^*(\mathcal{A})\rangle\|^2 \ ,$$

where $P := I \otimes \sum_{D \in \mathcal{P}} |D\rangle\langle D|$ *is the projector that maps all basis states of the form* $|x, u, z\rangle \otimes |D\rangle$ *to* 0 *if* $D \notin \mathcal{P}$*, and is otherwise the identity.*

We learn that in order to bound $\omega^*_{\mathsf{D}}(G, t)$ it suffices to bound $\|P_G\,|\mathsf{Sim}^*(\mathcal{A})\rangle\|$ for every $\mathcal{A} \in \mathcal{C}^*_t$.

Next, define $P_t := I \otimes \sum_{D:|D| \leq t} |D\rangle\langle D|$ to be the projector that maps all basis states of the form $|x, u, z\rangle \otimes |D\rangle$ to 0 if $|D| > t$, and is otherwise the identity.

The proof of Lemma 5.7 follows from two lemmas. The first lemma shows that $\|P|\mathsf{Sim}^*(\mathcal{A})\rangle\|$ is bounded by $t\|P(P_t\mathcal{O}P_t)\bar{P}\|$. Intuitively, this is because if P and $P_t\mathcal{O}P_t$ almost commute (i.e., P and $\mathcal{O}$ almost commute when acting on databases with at most t entries) then each oracle query cannot change the probability that the database is in $\mathcal{P}$ by too much. The second lemma shows that $\|P(P_t\mathcal{O}P_t)\bar{P}\|^2$ is bounded by $\mathbf{I}(\mathcal{P}, t)$. Combining the two lemmas with Proposition 5.8 completes the proof of Lemma 5.7.

Lemma 5.9. *Let* $\mathcal{P}$ *be a database property with* $\emptyset \notin \mathcal{P}$*. For every* $\mathcal{A} \in \mathcal{C}^*_t$*,*

$$\|P|\mathsf{Sim}^*(\mathcal{A})\rangle\| \leq t \cdot \|P(P_t\mathcal{O}P_t)\bar{P}\| \ .$$

Lemma 5.10. *For any database property* $\mathcal{P}$*,*

$$\|P(P_t\mathcal{O}P_t)\bar{P}\|^2 \leq 6\mathbf{I}(\mathcal{P}, t) \ .$$

[6] Let $\mathcal{A}$ be a classical adversary, and let $\mathcal{A}_i$ be the adversary obtained by stopping $\mathcal{A}$ immediately before its i-th query. Then $|\Pr[\mathcal{A}_{i+1} \text{ wins } G_{\mathsf{D}}] - \Pr[\mathcal{A}_i \text{ wins } G_{\mathsf{D}}]| \leq \mathbf{I}(\mathcal{P}, t)$ holds for each $i \in [t]$ by definition of instability, and $\Pr[\mathcal{A}_1 \text{ wins } G_{\mathsf{D}}] = 0$ since $\emptyset \notin \mathcal{P}_G$. Therefore, $\Pr[\mathcal{A} \text{ wins } G_{\mathsf{D}}] \leq t\mathbf{I}(\mathcal{P}, t)$.

Lemmas 5.9 and 5.10 strengthen the proof sketch outlined in Section 2.5. This is because for any operator A and projector P, $[P,A] = PA - AP = (PAP + PA\bar{P}) - (PAP + \bar{P}AP) = PA\bar{P} - \bar{P}AP$, and so $\|[P,A]\|^2 = \|PA\bar{P}\|^2 + \|\bar{P}AP\|^2$. Hence, Lemma 5.9 implies that $\|P|\mathsf{Sim}^*(\mathcal{A})\rangle\| \leq t \cdot \|[P, P_t\mathcal{O}P_t]\|$ and Lemma 5.10 implies that $\|[P, P_t\mathcal{O}P_t]\|^2 \leq 12\mathbf{I}(\mathcal{P}, t)$.

We now prove Lemma 5.9, and postpone the proof of Lemma 5.10 to Section 5.3.

Proof of Lemma 5.9. Recall that the quantum algorithm $\mathcal{A}$ is described by some unitaries $(A_1, \ldots, A_t)$ and initial state $|\phi_0\rangle$. We can thus describe the quantum algorithm $\mathsf{Sim}^*(\mathcal{A})$ via the cumulative unitary $U := A_t\mathcal{O}A_{t-1}\cdots\mathcal{O}A_1\mathcal{O}$ acting on the initial state $|\phi_0, \emptyset\rangle$ where $\emptyset$ denotes the empty database. (We abuse notation and implicitly extend A_i to act as the identity on the database register.) The final state is $|\mathsf{Sim}^*(\mathcal{A})\rangle := U|\phi_0, \emptyset\rangle$.

Let $U' := A_t(P_t\mathcal{O}P_t)A_{t-1}\cdots(P_t\mathcal{O}P_t)A_1(P_t\mathcal{O}P_t)$. We have that $U'|\phi_0, \emptyset\rangle = U|\phi_0, \emptyset\rangle$, as applying each P_t has no effect, since the database can only have at most t queries when P_t is applied.

For any operators $C_1, \ldots, C_t$ and projector P, we have that

$$C_t \cdots C_1 = \bar{P}C_t\bar{P}C_{t-1}\bar{P}\cdots C_1\bar{P} + \sum_{i=0}^{t}(C_t\cdots C_{i+1})\cdot P\cdot(C_i\bar{P}\cdots C_1\bar{P}) \ . \tag{1}$$

To see this, we observe that

$$C_t\cdots C_1 = (C_t\cdots C_2)(C_1\bar{P}) + (C_t\cdots C_1)\cdot P \ ,$$

which implies Eq. (1) by induction.

Let $C_i = A_i(P_t\mathcal{O}P_t)$. Then we have that

$$\|P|\mathsf{Sim}^*(\mathcal{A})\rangle\| = \|PU'|\phi_0,\emptyset\rangle\| = \|\Big(P\bar{P}C_t\bar{P}C_{t-1}\bar{P}\cdots C_1\bar{P} + \sum_{i=0}^{t}P(C_t\cdots C_{i+1})\cdot P\cdot(C_i\bar{P}\cdots C_1\bar{P})\Big)|\phi_0,\emptyset\rangle\|$$

$$\leq \sum_{i=0}^{t}\|P(C_t\cdots C_{i+1})\cdot P\cdot(C_i\bar{P}\cdots C_1\bar{P})|\phi_0,\emptyset\rangle\|$$

$$\leq \|P(C_t\cdots C_1)\cdot P|\phi_0,\emptyset\rangle\| + \sum_{i=1}^{t}\|P(C_t\cdots C_{i+1})\|\cdot\|P\cdot(C_i\bar{P}\cdots C_1\bar{P})|\phi_0,\emptyset\rangle\|$$

$$\leq 0 + \sum_{i=1}^{t}\|PC_i\bar{P}\|\cdot\|(C_i\bar{P}\cdots C_1\bar{P})|\phi_0,\emptyset\rangle\|$$

$$\leq \sum_{i=1}^{t}\|PA_i(P_t\mathcal{O}P_t)\bar{P}\| \ ,$$

where we use the fact that the operator norm of a product of unitaries/projectors is at most 1, and that $\emptyset \notin \mathcal{P}$. Since P and A_i commute for every i, we get that $\|PA_i(P_t\mathcal{O}P_t)\bar{P}\| = \|A_iP(P_t\mathcal{O}P_t)\bar{P}\| \leq \|A_i\|\|P(P_t\mathcal{O}P_t)\bar{P}\| = \|P(P_t\mathcal{O}P_t)\bar{P}\|$. Hence, $\|P|\mathsf{Sim}^*(\mathcal{A})\rangle\| \leq t\|P(P_t\mathcal{O}P_t)\bar{P}\|$. □

5.2 Conditional instability and the lifting lemma

Lemma 5.7 is not quite sufficient to analyze the database game that corresponds to the Micali construction. In fact, the instability of this game is high because we take a maximum over all

bounded databases, including those which contain collisions. If we were to only take the maximum over databases that do not contain collisions, then the instability would be low. Moreover, the instability of the "no collision" property is itself low.

In this section, we strengthen the results of the previous section by introducing the notion of *conditional* instability, which allows us to analyze the value $\omega^*_{\mathsf{D}}(G,t)$ by splitting its database property $\mathcal{P}_G$ into subproperties and analyzing the subproperties separately, analogous to conditioning in probability. In particular, we can then analyze the Micali game by analyzing the no collision property and the instability of the Micali database property conditioned on the no collision property.

For the entirety of this section we will let $\mathcal{P}$ and $\mathcal{Q}$ be database properties, and we will analyze quantities about $\mathcal{P}$ conditioned on $\mathcal{Q}$. These results strengthen the results of Section 5.1, as the previous results can be recovered by setting $\mathcal{Q}$ to be the database property containing all databases.

Definition 5.11. *Let $\mathcal{P}$ and $\mathcal{Q}$ be two database properties, and let t be a query bound. We define*

$$\mathrm{flip}(\mathcal{P} \mid \mathcal{Q}, t) := \mathrm{flip}(\bar{\mathcal{P}} \cap \mathcal{Q} \to \mathcal{P} \cap \mathcal{Q}, t) \ .$$

The **conditional instability** $\mathbf{I}(\mathcal{P} \mid \mathcal{Q}, t)$ *is defined as*

$$\mathbf{I}(\mathcal{P} \mid \mathcal{Q}, t) := \max\{\mathrm{flip}(\mathcal{P} \mid \mathcal{Q}, t),\ \mathrm{flip}(\bar{\mathcal{P}} \mid \mathcal{Q}, t)\} \ .$$

Before we state the lifting lemma, we observe the following properties of instability.

Proposition 5.12. *Let $\mathcal{P}$ and $\mathcal{Q}$ be two database properties. Then*

1. $\mathbf{I}(\mathcal{P}, t)$ *and* $\mathbf{I}(\mathcal{P} \cup \mathcal{Q}, t)$ *are incomparable.*
2. $\mathrm{flip}(\mathcal{P} \mid \mathcal{Q}, t) \le \mathrm{flip}(\bar{\mathcal{P}} \to \mathcal{P}, t)$*, and therefore* $\mathbf{I}(\mathcal{P} \mid \mathcal{Q}, t) \le \mathbf{I}(\mathcal{P}, t)$*.*
3. $\mathbf{I}(\mathcal{P} \cup \mathcal{Q}, t) \le \mathbf{I}(\mathcal{P} \mid \bar{\mathcal{Q}}, t) + \mathbf{I}(\mathcal{Q}, t)$*.*

Proof. To show Item 1, we give database properties $\mathcal{P}, \mathcal{Q}$ such that $\mathbf{I}(\mathcal{P}, t) > \mathbf{I}(\mathcal{P} \cup \mathcal{Q}, t)$ and properties $\mathcal{P}', \mathcal{Q}'$ such that $\mathbf{I}(\mathcal{P}', t) < \mathbf{I}(\mathcal{P}' \cup \mathcal{Q}', t)$. Let $\mathcal{P}$ be the property that $D \neq \emptyset$. Then clearly $\mathbf{I}(\mathcal{P}, t) \ge \mathrm{flip}(\bar{\mathcal{P}} \to \mathcal{P}, t) = 1$. Let $\mathcal{Q}$ be the property that $D = \emptyset$. Now $\mathcal{P} \cup \mathcal{Q}$ is the set of all databases, so $\mathbf{I}(\mathcal{P} \cup \mathcal{Q}, t) = 0$.

On the other hand, let $\mathcal{P}' = \emptyset$ be the empty property, and let $\mathcal{Q}'$ be the property that $D = \emptyset$. Then, $\mathbf{I}(\mathcal{P}', t) = 0$, and $\mathbf{I}(\mathcal{P}' \cup \mathcal{Q}', t) = \mathbf{I}(\mathcal{Q}', t) = 1$.

Item 2 holds since

$$\mathrm{flip}(\mathcal{P} \mid \mathcal{Q}, t) = \mathrm{flip}(\bar{\mathcal{P}} \cap \mathcal{Q} \to \mathcal{P} \cap \mathcal{Q}, t) \le \mathrm{flip}(\bar{\mathcal{P}} \to \mathcal{P}, t) \ .$$

Finally, for Item 3 we observe that

$$\begin{aligned}\mathrm{flip}(\overline{\mathcal{P} \cup \mathcal{Q}} \to \mathcal{P} \cup \mathcal{Q}, t) &= \mathrm{flip}(\bar{\mathcal{P}} \cap \bar{\mathcal{Q}} \to \mathcal{P} \cup \mathcal{Q}, t) \le \mathrm{flip}(\bar{\mathcal{P}} \cap \bar{\mathcal{Q}} \to \mathcal{P} \cap \bar{\mathcal{Q}}, t) + \mathrm{flip}(\bar{\mathcal{P}} \cap \bar{\mathcal{Q}} \to \mathcal{Q}, t) \\ &\le \mathrm{flip}(\mathcal{P} \mid \bar{\mathcal{Q}}, t) + \mathrm{flip}(\bar{\mathcal{Q}} \to \mathcal{Q}, t) \ .\end{aligned}$$

On the other hand,

$$\begin{aligned}\mathrm{flip}(\mathcal{P} \cup \mathcal{Q} \to \overline{\mathcal{P} \cup \mathcal{Q}}, t) &= \mathrm{flip}(\mathcal{P} \cup \mathcal{Q} \to \bar{\mathcal{P}} \cap \bar{\mathcal{Q}}, t) = \max(\mathrm{flip}(\mathcal{P} \cap \bar{\mathcal{Q}} \to \bar{\mathcal{P}} \cap \bar{\mathcal{Q}}, t), \mathrm{flip}(\mathcal{Q} \to \bar{\mathcal{P}} \cap \bar{\mathcal{Q}}, t)) \\ &\le \max(\mathrm{flip}(\bar{\mathcal{P}} \mid \bar{\mathcal{Q}}, t), \mathrm{flip}(\mathcal{Q} \to \bar{\mathcal{Q}}, t)) \ .\end{aligned}$$

Therefore, we get that $\mathbf{I}(\mathcal{P} \cup \mathcal{Q}) \le \max\big(\mathrm{flip}(\mathcal{P} \mid \bar{\mathcal{Q}}, t) + \mathrm{flip}(\bar{\mathcal{Q}} \to \mathcal{Q}, t), \mathrm{flip}(\bar{\mathcal{P}} \mid \bar{\mathcal{Q}}, t), \mathrm{flip}(\mathcal{Q} \to \bar{\mathcal{Q}}, t)\big)$, which is at most $\mathbf{I}(\mathcal{P} \mid \bar{\mathcal{Q}}, t) + \mathbf{I}(\mathcal{Q}, t)$.

□

We now state the lifting lemma.

Lemma 5.13 (Lifting lemma)**.** *Let G be a base game. Then for any database property $\mathcal{Q}$,*

$$\omega^*_{\mathsf{D}}(G,t) \leq t^2 \cdot 6\left(\mathbf{I}(\mathcal{P}_G \mid \bar{\mathcal{Q}},t) + \mathbf{I}(\mathcal{Q},t)\right) \ .$$

Proof. Let $\mathcal{P}$ and $\mathcal{Q}$ be two database properties. We show that for every $\mathcal{A} \in \mathcal{C}^*_t$ it holds that

$$\|P|\mathsf{Sim}^*(\mathcal{A})\rangle\|^2 \leq t^2 \cdot 6\left(\mathbf{I}(\mathcal{P} \mid \bar{\mathcal{Q}},t) + \mathbf{I}(\mathcal{Q},t)\right) \ .$$

Let $\mathcal{R} = \mathcal{P} \cup \mathcal{Q}$. Then by Lemmas 5.9 and 5.10 we have that

$$\|P|\mathsf{Sim}^*(\mathcal{A})\rangle\|^2 \leq \|R\,|\mathsf{Sim}^*(\mathcal{A})\rangle\|^2 \leq t^2 \cdot \|[R, P_t\mathcal{O}P_t]\|^2 \leq t^2 \cdot 6\mathbf{I}(\mathcal{R},t) \ ,$$

where the first inequality holds since $\mathcal{P} \subseteq \mathcal{R}$. Finally, we use the fact that $\mathbf{I}(\mathcal{R},t) = \mathbf{I}(\mathcal{P} \cup \mathcal{Q},t) \leq \mathbf{I}(\mathcal{P} \mid \bar{\mathcal{Q}},t) + \mathbf{I}(\mathcal{Q},t)$, which completes the proof. □

We dedicate the remainder of the section to proving Lemma 5.10.

5.3 Proof of Lemma 5.10

Let $|\Phi\rangle$ be an arbitrary state in the image of $P_t\bar{P}$, i.e. $|\Phi\rangle = \sum_{x,u,z,D} \alpha_{x,u,z,D}\,|x,u,z,D\rangle$, where the sum is over all $D\colon \{0,1\}^m \rightharpoonup \{0,1\}^n$ such that $D \in \bar{\mathcal{P}}$ and $|D| \leq t$. By Lemma 3.2,

$$\mathcal{O}\,|\Phi\rangle = |\Phi_1\rangle + |\Phi_2\rangle + |\Phi_3\rangle + |\Phi_4\rangle + |\Phi_5\rangle \ ,$$

where $|\Phi_1\rangle$, $|\Phi_2\rangle$, $|\Phi_3\rangle$, $|\Phi_4\rangle$, and $|\Phi_5\rangle$ are the following states:

$$|\Phi_1\rangle = \sum_{\substack{x,u,z,D \\ D\in\bar{\mathcal{P}} \\ (|D|=t \text{ or } u=0^n)}} \alpha_{x,u,z,D}\,|x,u,z\rangle \otimes (-1)^{u\cdot D(x)}\,|D\rangle \ ,$$

$$|\Phi_2\rangle = \sum_{\substack{x,u,z,D \\ D(x)=\perp, D\in\bar{\mathcal{P}} \\ |D|<t, u\neq 0^n}} \alpha_{x,u,z,D}\,|x,u,z\rangle \otimes \frac{1}{\sqrt{2^n}}\sum_y (-1)^{u\cdot y}\,|D+[x\mapsto y]\rangle \ ,$$

$$|\Phi_3\rangle = \sum_{\substack{x,u,z,D \\ D(x)\neq\perp, D\in\bar{\mathcal{P}} \\ |D|<t, u\neq 0^n}} \alpha_{x,u,z,D}\,|x,u,z\rangle \otimes (-1)^{u\cdot D(x)}\,|D\rangle \ ,$$

$$|\Phi_4\rangle = \sum_{\substack{x,u,z,D \\ D(x)\neq\perp, D\in\bar{\mathcal{P}} \\ |D|<t, u\neq 0^n}} \alpha_{x,u,z,D}\,|x,u,z\rangle \otimes \frac{(-1)^{u\cdot D(x)}}{\sqrt{2^n}}\,|D-x\rangle \ ,$$

$$|\Phi_5\rangle = \sum_{\substack{x,u,z,D \\ D(x)\neq\perp, D\in\bar{\mathcal{P}} \\ |D|<t, u\neq 0}} \alpha_{x,u,z,D}\,|x,u,z\rangle \otimes \frac{1}{2^n}\sum_y \left(1-(-1)^{u\cdot D(x)}-(-1)^{u\cdot y}\right)|D-x+[x\mapsto y]\rangle \ .$$

Let $|\Psi_i\rangle = PP_t\,|\Phi_i\rangle$ for $i \in [5]$. We have that

$$|\Psi_1\rangle = 0 \ ,$$

$$|\Psi_2\rangle = \sum_{\substack{x,u,z,D \\ D(x)=\perp, D\in\bar{\mathcal{P}} \\ |D|<t, u\neq 0^n}} \alpha_{x,u,z,D}\,|x,u,z\rangle \otimes \frac{1}{\sqrt{2^n}} \sum_{\substack{y \\ D+[x\mapsto y]\in\mathcal{P}}} (-1)^{u\cdot y}\,|D+[x\mapsto y]\rangle \ ,$$

$$|\Psi_3\rangle = 0 \ ,$$

$$|\Psi_4\rangle = \sum_{\substack{x,u,z,D \\ D(x)\neq\perp, D\in\bar{\mathcal{P}}, D-x\in\mathcal{P} \\ |D|<t, u\neq 0^n}} \alpha_{x,u,z,D}\,|x,u,z\rangle \otimes \frac{(-1)^{u\cdot D(x)}}{\sqrt{2^n}}\,|D-x\rangle$$

$$= \sum_{\substack{x,u,z,D' \\ D'(x)=\perp, D'\in\mathcal{P} \\ |D'|<t-1, u\neq 0^n}} \Big(\sum_{\substack{w \\ D'+[x\mapsto w]\in\bar{\mathcal{P}}}} \alpha_{x,u,z,D'+[x\mapsto w]} \frac{(-1)^{u\cdot w}}{\sqrt{2^n}} \Big)\,|x,u,z\rangle \otimes \big|D'\big\rangle \ ,$$

$$|\Psi_5\rangle = \sum_{\substack{x,u,z,D \\ D(x)\neq\perp, D\in\bar{\mathcal{P}} \\ |D|<t, u\neq 0^n}} \alpha_{x,u,z,D}\,|x,u,z\rangle \otimes \frac{1}{2^n} \sum_{\substack{y \\ D-x+[x\mapsto y]\in\mathcal{P}}} \big(1-(-1)^{u\cdot D(x)}-(-1)^{u\cdot y}\big)\,|D-x+[x\mapsto y]\rangle$$

$$= \sum_{\substack{x,u,z,D' \\ D'(x)=\perp \\ |D'|<t-1, u\neq 0^n}} \sum_{\substack{y \\ D'+[x\mapsto y]\in\mathcal{P}}} \Big(\sum_{\substack{w \\ D'+[x\mapsto w]\in\bar{\mathcal{P}}}} \alpha_{x,u,z,D'+[x\mapsto w]} \frac{1}{2^n}\big(1-(-1)^{u\cdot w}-(-1)^{u\cdot y}\big)\Big)\,|x,u,z\rangle \otimes \big|D'+[x\mapsto y]\big\rangle$$

Let $S_{\bar{\mathcal{P}}}(D',x) := \{y \in \{0,1\}^n : D'+[x\mapsto y] \in \bar{\mathcal{P}}\}$, and let $S_{\mathcal{P}}(D',x) := \{y \in \{0,1\}^n : D'+[x\mapsto y] \in \mathcal{P}\}$. Observe that $|\Psi_4\rangle$ is orthogonal to $|\Psi_2\rangle$ and $|\Psi_5\rangle$, as every database D with nonzero amplitude in $|\Psi_4\rangle$ has $D(x)=\perp$.

Define

$$s_1 := \sum_{\substack{x,u,z,D \\ D(x)\neq\perp, D\in\bar{\mathcal{P}}, D-x\in\mathcal{P} \\ |D|<t, u\neq 0^n}} |\alpha_{x,u,z,D}|^2 \ ,$$

$$s_2 := \sum_{\substack{x,u,z,D' \\ D'(x)=\perp, D'\in\bar{\mathcal{P}} \\ |D'|=t-1, u\neq 0^n}} \big|\alpha_{x,u,z,D'}\big|^2 \ ,$$

$$s_3 := \sum_{\substack{x,u,z,D' \\ D'(x)=\perp, D'\in\bar{\mathcal{P}} \\ |D'|<t-1, u\neq 0^n}} \big|\alpha_{x,u,z,D'}\big|^2 \ ,$$

$$s_4 := \sum_{\substack{x,u,z,D \\ D(x)\neq\perp, D\in\bar{\mathcal{P}}, D-x\in\bar{\mathcal{P}} \\ |D|<t, u\neq 0^n}} |\alpha_{x,u,z,D}|^2 \ .$$

Note that by normalization, we have that $s_1+s_2+s_3+s_4 \leq 1$.

By Cauchy-Schwarz, we have that

$$\begin{aligned}
\||\Psi_4\rangle\|^2 &= \sum_{\substack{x,u,z,D' \\ D'(x)=\perp, D'\in\mathcal{P} \\ |D'|<t-1, u\neq 0^n}} \left| \sum_{\substack{w \\ D'+[x\mapsto w]\in\bar{\mathcal{P}}}} \alpha_{x,u,z,D'+[x\mapsto w]} \frac{(-1)^{u\cdot w}}{\sqrt{2^n}} \right|^2 \\
&\leq \sum_{\substack{x,u,z,D' \\ D'(x)=\perp, D'\in\mathcal{P} \\ |D'|<t-1, u\neq 0^n}} \Big(\sum_{\substack{w \\ D'+[x\mapsto w]\in\bar{\mathcal{P}}}} \left|\alpha_{x,u,z,D'+[x\mapsto w]}\right|^2 \Big) \cdot \frac{\left|S_{\bar{\mathcal{P}}}(D',x)\right|}{2^n} \\
&\leq \mathrm{flip}(\mathcal{P}\to\bar{\mathcal{P}},t) \sum_{\substack{x,u,z,D' \\ D'(x)=\perp, D'\in\mathcal{P} \\ |D'|<t-1, u\neq 0^n}} \Big(\sum_{\substack{w \\ D'+[x\mapsto w]\in\bar{\mathcal{P}}}} \left|\alpha_{x,u,z,D'+[x\mapsto w]}\right|^2 \Big) \\
&\leq \mathrm{flip}(\mathcal{P}\to\bar{\mathcal{P}},t) \sum_{\substack{x,u,z,D \\ D(x)\neq\perp, D\in\bar{\mathcal{P}}, D-x\in\mathcal{P} \\ |D|<t, u\neq 0^n}} \left|\alpha_{x,u,z,D}\right|^2 \\
&= \mathrm{flip}(\mathcal{P}\to\bar{\mathcal{P}},t)\cdot s_1 \ .
\end{aligned}$$

Let $\beta_{w,y} = 1-(-1)^{u\cdot w}-(-1)^{u\cdot y}$. Observe that $|\Psi_2\rangle + |\Psi_5\rangle = |\Xi_1\rangle + |\Xi_2\rangle + |\Xi_3\rangle$, where $|\Xi_1\rangle$, $|\Xi_2\rangle$ and $|\Xi_3\rangle$ are the following orthogonal states:

$$|\Xi_1\rangle = \sum_{\substack{x,u,z,D' \\ D'(x)=\perp, D'\in\bar{\mathcal{P}} \\ |D'|=t-1, u\neq 0^n}} \alpha_{x,u,z,D'} \frac{1}{\sqrt{2^n}} \sum_{\substack{y \\ D'+[x\mapsto y]\in\mathcal{P}}} (-1)^{u\cdot y} |x,u,z\rangle \otimes \left|D'+[x\mapsto y]\right\rangle$$

$$|\Xi_2\rangle = \sum_{\substack{x,u,z,D' \\ D'(x)=\perp, D'\in\mathcal{P} \\ |D'|<t-1, u\neq 0^n}} \sum_{D'+[x\mapsto y]\in\mathcal{P}} \Big(\sum_{\substack{w \\ D'+[x\mapsto w]\in\bar{\mathcal{P}}}} \alpha_{x,u,z,D'+[x\mapsto w]} \frac{1}{2^n}\beta_{w,y} \Big) |x,u,z\rangle \otimes \left|D'+[x\mapsto y]\right\rangle$$

$$|\Xi_3\rangle = \sum_{\substack{x,u,z,D' \\ D'(x)=\perp, D'\in\bar{\mathcal{P}} \\ |D'|<t-1, u\neq 0^n}} \sum_{D'+[x\mapsto y]\in\mathcal{P}} \Big(\frac{(-1)^{u\cdot y}\alpha_{x,u,z,D'}}{\sqrt{2^n}} + \sum_{\substack{w \\ D'+[x\mapsto w]\in\bar{\mathcal{P}}}} \alpha_{x,u,z,D'+[x\mapsto w]} \frac{1}{2^n}\beta_{w,y} \Big) |x,u,z\rangle \otimes \left|D'+[x\mapsto y]\right\rangle \ .$$

We also observe that $\sum_y \left|\frac{1}{2^n}\beta_{w,y}\right|^2 = \frac{3-2(-1)^{u\cdot w}}{2^n} \leq \frac{5}{2^n}$, and $\sum_w \left|\frac{1}{2^n}\beta_{w,y}\right|^2 = \frac{3-2(-1)^{u\cdot y}}{2^n} \leq \frac{5}{2^n}$.
We have that

$$\begin{aligned}
\||\Xi_1\rangle\|^2 &= \sum_{\substack{x,u,z,D' \\ D'(x)=\perp, D'\in\bar{\mathcal{P}} \\ |D'|=t-1, u\neq 0^n}} \sum_{D'+[x\mapsto y]\in\mathcal{P}} \left| \alpha_{x,u,z,D'} \frac{1}{\sqrt{2^n}} (-1)^{u\cdot y} \right|^2 = \sum_{\substack{x,u,z,D' \\ D'(x)=\perp, D'\in\bar{\mathcal{P}} \\ |D'|=t-1, u\neq 0^n}} \left|\alpha_{x,u,z,D'}\right|^2 \frac{|S_{\mathcal{P}}(D',x)|}{2^n} \\
&\leq \mathrm{flip}(\bar{\mathcal{P}}\to\mathcal{P},t) \sum_{\substack{x,u,z,D' \\ D'(x)=\perp, D'\in\bar{\mathcal{P}} \\ |D'|=t-1, u\neq 0^n}} \left|\alpha_{x,u,z,D'}\right|^2 = \mathrm{flip}(\bar{\mathcal{P}}\to\mathcal{P},t)\cdot s_2 \ .
\end{aligned}$$

By Cauchy-Schwarz, we get that

$$
\begin{aligned}
\||\Xi_2\rangle\|^2 &= \sum_{\substack{x,u,z,D' \\ D'(x)=\perp, D'\in\mathcal{P} \\ |D'|<t-1, u\neq 0^n}} \sum_{y\ D'+[x\mapsto y]\in\mathcal{P}} \left| \sum_{w\ D'+[x\mapsto w]\in\bar{\mathcal{P}}} \alpha_{x,u,z,D'+[x\mapsto w]} \frac{1}{2^n} \beta_{w,y} \right|^2 \\
&\leq \sum_{\substack{x,u,z,D' \\ D'(x)=\perp, D'\in\mathcal{P} \\ |D'|<t-1, u\neq 0^n}} \sum_{y\ D'+[x\mapsto y]\in\mathcal{P}} \Big(\sum_{w\ D'+[x\mapsto w]\in\bar{\mathcal{P}}} \left|\alpha_{x,u,z,D'+[x\mapsto w]}\right|^2 \Big) \Big(\sum_{w\ D'+[x\mapsto w]\in\bar{\mathcal{P}}} \left| \frac{1}{2^n}\beta_{w,y} \right|^2 \Big) \\
&= \sum_{\substack{x,u,z,D' \\ D'(x)=\perp, D'\in\mathcal{P} \\ |D'|<t-1, u\neq 0^n}} \Big(\sum_{w\ D'+[x\mapsto w]\in\bar{\mathcal{P}}} \left|\alpha_{x,u,z,D'+[x\mapsto w]}\right|^2 \Big) \sum_{y\ D'+[x\mapsto y]\in\mathcal{P}} \Big(\sum_{w\ D'+[x\mapsto w]\in\bar{\mathcal{P}}} \left| \frac{1}{2^n}\beta_{w,y} \right|^2 \Big) \\
&= \sum_{\substack{x,u,z,D' \\ D'(x)=\perp, D'\in\mathcal{P} \\ |D'|<t-1, u\neq 0^n}} \Big(\sum_{w\ D'+[x\mapsto w]\in\bar{\mathcal{P}}} \left|\alpha_{x,u,z,D'+[x\mapsto w]}\right|^2 \Big) \Big(\sum_{w\ D'+[x\mapsto w]\in\bar{\mathcal{P}}} \sum_{y\ D'+[x\mapsto y]\in\mathcal{P}} \left| \frac{1}{2^n}\beta_{w,y} \right|^2 \Big) \\
&\leq \sum_{\substack{x,u,z,D' \\ D'(x)=\perp, D'\in\mathcal{P} \\ |D'|<t-1, u\neq 0^n}} \Big(\sum_{w\ D'+[x\mapsto w]\in\bar{\mathcal{P}}} \left|\alpha_{x,u,z,D'+[x\mapsto w]}\right|^2 \Big) \sum_{w\ D'+[x\mapsto w]\in\bar{\mathcal{P}}} \frac{5}{2^n} \\
&\leq \sum_{\substack{x,u,z,D' \\ D'(x)=\perp, D'\in\mathcal{P} \\ |D'|<t-1, u\neq 0^n}} \Big(\sum_{w\ D'+[x\mapsto w]\in\bar{\mathcal{P}}} \left|\alpha_{x,u,z,D'+[x\mapsto w]}\right|^2 \Big) \frac{5\left|S_{\bar{\mathcal{P}}}(D',x)\right|}{2^n} \\
&\leq 5\cdot \mathrm{flip}(\mathcal{P}\to\bar{\mathcal{P}},t) \sum_{\substack{x,u,z,D' \\ D'(x)=\perp, D'\in\mathcal{P} \\ |D'|<t-1, u\neq 0^n}} \Big(\sum_{w\ D'+[x\mapsto w]\in\bar{\mathcal{P}}} \left|\alpha_{x,u,z,D'+[x\mapsto w]}\right|^2 \Big) \\
&= 5\cdot \mathrm{flip}(\mathcal{P}\to\bar{\mathcal{P}},t) \sum_{\substack{x,u,z,D \\ D(x)\neq\perp, D\in\bar{\mathcal{P}}, D-x\in\mathcal{P} \\ |D|<t, u\neq 0^n}} \left|\alpha_{x,u,z,D}\right|^2 \\
&= 5\cdot \mathrm{flip}(\mathcal{P}\to\bar{\mathcal{P}},t)\cdot s_1 \ .
\end{aligned}
$$

Finally, we have that by the (square of the) triangle inequality,

$$
\begin{aligned}
\||\Xi_3\rangle\|^2 &= \sum_{\substack{x,u,z,D' \\ D'(x)=\perp, D'\in\bar{\mathcal{P}} \\ |D'|<t-1, u\neq 0^n}} \sum_{y\ D'+[x\mapsto y]\in\mathcal{P}} \left| \frac{(-1)^{u\cdot y}\alpha_{x,u,z,D'}}{\sqrt{2^n}} + \sum_{w\ D'+[x\mapsto w]\in\bar{\mathcal{P}}} \alpha_{x,u,z,D'+[x\mapsto w]} \frac{1}{2^n}\beta_{w,y} \right|^2 \\
&\leq \varepsilon + \varepsilon' + 2\sqrt{\varepsilon\cdot\varepsilon'} \ ,
\end{aligned}
$$

where

$$\varepsilon = \sum_{\substack{x,u,z,D' \\ D'(x)=\perp, D'\in\bar{\mathcal{P}} \\ |D'|<t-1, u\neq 0^n}} \sum_{y \; D'+[x\mapsto y]\in\mathcal{P}} \left| \frac{(-1)^{u\cdot y}\alpha_{x,u,z,D'}}{\sqrt{2^n}} \right|^2 \leq \sum_{\substack{x,u,z,D' \\ D'(x)=\perp, D'\in\bar{\mathcal{P}} \\ |D'|<t-1, u\neq 0^n}} \left|\alpha_{x,u,z,D'}\right|^2 \cdot \frac{|S_{\mathcal{P}}(D',x)|}{2^n}$$

$$\leq \mathrm{flip}(\bar{\mathcal{P}}\to\mathcal{P},t) \sum_{\substack{x,u,z,D' \\ D'(x)=\perp, D'\in\bar{\mathcal{P}} \\ |D'|<t-1, u\neq 0^n}} \left|\alpha_{x,u,z,D'}\right|^2 = \mathrm{flip}(\bar{\mathcal{P}}\to\mathcal{P},t)\cdot s_3 \ ,$$

and

$$\varepsilon' = \sum_{\substack{x,u,z,D' \\ D'(x)=\perp, D'\in\bar{\mathcal{P}} \\ |D'|<t-1, u\neq 0^n}} \sum_{y \; D'+[x\mapsto y]\in\mathcal{P}} \left| \sum_{w \; D'+[x\mapsto w]\in\bar{\mathcal{P}}} \alpha_{x,u,z,D'+[x\mapsto w]} \frac{1}{2^n}\beta_{w,y} \right|^2$$

$$\leq \sum_{\substack{x,u,z,D' \\ D'(x)=\perp, D'\in\bar{\mathcal{P}} \\ |D'|<t-1, u\neq 0^n}} \sum_{y \; D'+[x\mapsto y]\in\mathcal{P}} \Big(\sum_{w \; D'+[x\mapsto w]\in\bar{\mathcal{P}}} \left|\alpha_{x,u,z,D'+[x\mapsto w]}\right|^2 \Big) \cdot \Big(\sum_w \left| \frac{1}{2^n}\beta_{w,y} \right|^2 \Big)$$

$$\leq \sum_{\substack{x,u,z,D' \\ D'(x)=\perp, D'\in\bar{\mathcal{P}} \\ |D'|<t-1, u\neq 0^n}} \sum_{y \; D'+[x\mapsto y]\in\mathcal{P}} \Big(\sum_{w \; D'+[x\mapsto w]\in\bar{\mathcal{P}}} \left|\alpha_{x,u,z,D'+[x\mapsto w]}\right|^2 \Big) \cdot \frac{5}{2^n}$$

$$\leq \sum_{\substack{x,u,z,D' \\ D'(x)=\perp, D'\in\bar{\mathcal{P}} \\ |D'|<t-1, u\neq 0^n}} \frac{5|S_{\mathcal{P}}(D',x)|}{2^n} \sum_{w \; D'+[x\mapsto w]\in\bar{\mathcal{P}}} \left|\alpha_{x,u,z,D'+[x\mapsto w]}\right|^2$$

$$\leq 5\cdot \mathrm{flip}(\bar{\mathcal{P}}\to\mathcal{P},t) \sum_{\substack{x,u,z,D \\ D(x)\neq\perp, D\in\bar{\mathcal{P}}, D-x\in\bar{\mathcal{P}} \\ |D|<t, u\neq 0^n}} \left|\alpha_{x,u,z,D}\right|^2$$

$$= 5\cdot \mathrm{flip}(\bar{\mathcal{P}}\to\mathcal{P},t)\cdot s_4 \ .$$

Hence,

$$\||\Xi_3\rangle\|^2 \leq \mathrm{flip}(\bar{\mathcal{P}}\to\mathcal{P},t)\Big(s_3+5s_4+2\sqrt{5s_3s_4}\Big) \leq 6\cdot\mathrm{flip}(\bar{\mathcal{P}}\to\mathcal{P},t)\cdot(s_3+s_4) \ .$$

Putting it all together (and recalling that $s_1+s_2+s_3+s_4\leq 1$), we get that

$$\|\bar{Q}P_t\mathcal{O}\Phi\|^2 = \||\Psi_4\rangle\|^2 + \||\Xi_1\rangle\|^2 + \||\Xi_2\rangle\|^2 + \||\Xi_3\rangle\|^2$$
$$\leq \mathrm{flip}(\mathcal{P}\to\bar{\mathcal{P}},t)\cdot s_1 + \mathrm{flip}(\bar{\mathcal{P}}\to\mathcal{P},t)\cdot s_2 + \mathrm{flip}(\mathcal{P}\to\bar{\mathcal{P}},t)\cdot 5s_1 + \mathrm{flip}(\bar{\mathcal{P}}\to\mathcal{P},t)\cdot 6(s_3+s_4)$$
$$\leq \mathbf{I}(\mathcal{P},t)\Big(6s_1+s_2+6s_3+6s_4\Big)$$
$$\leq 6\mathbf{I}(\mathcal{P},t) \ ,$$

which completes the proof.

6 Soundness of the Micali construction

We use the lifting lemma to prove that the Micali construction is sound in the quantum random oracle model. First, we show that convincing the Micali verifier amounts to winning a corresponding oracle game, in the sense of Definition 4.6. Second, we show that the database property induced by this game is stable when conditioned on the database having no collisions. Both of these are *classical* properties of the Micali construction. Indeed, our results show how the *existing* proof of soundness for this protocol fits into our instability framework, making quantum security immediate.

Theorem 6.1. *Let $(\mathbf{P}, \mathbf{V})$ be a PCP for a relation $\mathcal{R}$ that has soundness error ϵ, proof length ℓ, and query complexity q. The Micali construction, when based on $(\mathbf{P}, \mathbf{V})$, is a non-interactive argument for $\mathcal{R}$ that has soundness error $O(t^2\epsilon + t^3/2^\lambda)$ against quantum attackers that make at most $t - O(q \log \ell)$ queries to the random oracle. This soundness error is tight up to small factors.*

The rest of this section is organized as follows: in Section 6.1 we describe the properties of Merkle trees used in the proof of security; in Section 6.2 we describe the oracle game that corresponds to the Micali construction; and in Section 6.3 we prove Theorem 6.1.

6.1 Some algorithms for Merkle trees

The Micali construction uses Merkle trees [Mer89] based on random oracles as succinct commitments to lists of values that permit cheaply decommitting to chosen entries in the list. While Merkle trees are well known, we rely on properties of Merkle trees that, while simple, are less well known. In this section we introduce notations needed to specify these properties.

Given a domain Z and a power of two $\ell = 2^d$, a *Merkle tree* on a list $\mathbf{v} = (v_i)_{i\in[\ell]} \in Z^\ell$ with respect to the function $h\colon Z \times Z \to Z$ is a list of values $(z_{k,i})_{k\in\{0,\ldots,d\},i\in[2^k]} \in Z^{2\ell}$, where

$$\forall\, i \in [\ell]\,,\ z_{d,i} := v_i\ ,$$
$$\forall\, k \in \{d-1,\ldots,1,0\}\,,\ \forall\, i \in [2^k]\,,\ z_{k,i} := h(z_{k+1,2i-1}, z_{k+1,2i})\ .$$

The *root* of the Merkle tree, denoted rt, is $z_{0,1}$. Given $i \in [\ell]$, the *authentication path* for the i-th leaf in the Merkle tree is the list of values $\mathsf{ap} := (a_k)_{k\in[d]} \in Z^d$ where a_k is the sibling of the k-th node on the path from $v_i = z_{d,i}$ to the root $\mathsf{rt} = z_{0,1}$. An authentication path is *valid* for v at i with respect to rt, h if "hashing up" the path from v at index i to the root using ap yields rt. We denote by $\mathsf{CheckPath}^h(\mathsf{rt}, i, v, \mathsf{ap})$ the (efficient) algorithm that checks this condition.

Below we describe two algorithms that will be used in the proof of security.

Expanding paths. We define an algorithm Expand that outputs all the input-output pairs that arise when validating a given authentication path. The name "expand" denotes the fact the algorithm outputs not only the authentication path but also the nodes from the given leaf to the root. The algorithm Expand is used in Section 6.2 to define the oracle game for the Micali construction.

In more detail, Expand is granted oracle access to a function $h\colon Z \times Z \to Z$ and receives as input an index $i \in [2^d]$, a value $v \in Z$, and an authentication path $\mathsf{ap} := (a_k)_{k\in[d]} \in Z^d$. Then Expand sets $y_d := v$ and proceeds as follows for $k = d-1,\ldots,0$: if the k-th bit of i is 0, set $x_k := (y_{k+1}, a_{k+1})$; else if the k-th bit of i is 1, set $x_k := (a_{k+1}, y_{k+1})$; set $y_k := h(x_k)$. Finally, Expand outputs the d input-output pairs $(\mathbf{x}, \mathbf{y}) := ((x_i)_{i=0}^{d-1}, (y_i)_{i=0}^{d-1})$. The following property holds.

Proposition 6.2. $\mathsf{CheckPath}^h(\mathsf{rt}, i, v, \mathsf{ap}) = 1$ *if and only if* $(\mathbf{x}, \mathbf{y}) \leftarrow \mathsf{Expand}^h(i, v, \mathsf{ap})$ *has* $y_0 = \mathsf{rt}$.

Extracting trees. We define an algorithm $\mathsf{Extract}$ that is used to obtain a Merkle tree rooted at a chosen entry in a database. This algorithm is similar to the one presented in [Val08; BCS16], and below we give a description of it because we use it in Section 6.3 within the security proof.

In more detail, $\mathsf{Extract}$, receives a database $D\colon Z \times Z \rightharpoonup Z$, a root value $\mathsf{rt} \in Z$, and a height bound d, and outputs a rooted binary tree $T = (V, E)$ of depth at most d where $V \subseteq Z \times \{0,1\}^{\leq d}$. The algorithm $\mathsf{Extract}$ is defined as follows.

$\mathsf{Extract}(D, \mathsf{rt}, d)$:

1. Initialize the vertex set with the root $V := \{(\mathsf{rt}, \varnothing)\}$ and the edge set to be empty $E := \emptyset$.
2. While there exists an unmarked vertex $(u, s) \in V$ such that $u \in \mathrm{im}(D)$ and $s \in \{0,1\}^{<d}$:
   (a) Mark the vertex (u, s).
   (b) If u is the result of a collision ($D(x) = D(x') = u$ for distinct x and x'), return $\perp$.
   (c) Let $x_0, x_1 \in Z$ be the unique values such that $D(x_0, x_1) = u$.
   (d) Update the vertex set $V := V \cup \{(x_0, s\|0), (x_1, s\|1)\}$.
   (e) Update the edge set $E := E \cup \{((u, s), (x_0, s\|0)), ((u, s), (x_1, s\|1))\}$.
3. Output the tree $T := (V, E)$.

Given a tree $T := \mathsf{Extract}(D, \mathsf{rt}, d)$, we can define a string $\mathsf{leaves}(T)$ where for each $i \in [2^d] \cong \{0,1\}^d$ if there exists a vertex $(x, i) \in T$ then $\mathsf{leaves}(T)_i := x$ else $\mathsf{leaves}(T)_i := \perp$.

It will be useful in the analysis of Merkle tree algorithms to define the set $S(D)$ of "half-preimages" in a database D:

Definition 6.3. *Let $D\colon Z \times Z \rightharpoonup Z$ be a database. Then $S(D) := \{u : \exists\, u' \in Z \text{ s.t. } (u, u') \in \mathrm{supp}(D) \vee (u', u) \in \mathrm{supp}(D)\}$.*

The following two lemmas are about properties of the algorithm $\mathsf{Extract}$. The first lemma follows from the algorithm description, and the second one is proved below.

Lemma 6.4. *Let $D\colon Z \times Z \rightharpoonup Z$ be a database, $x \in Z \times Z$ an input with $x \notin \mathrm{supp}(D)$, $y \in Z$ an output, $\mathsf{rt} \in Z$, and $d \in \mathbb{N}$. For the rooted binary trees $T_1 := \mathsf{Extract}(D, \mathsf{rt}, d)$ and $T_2 := \mathsf{Extract}(D + [x \mapsto y], \mathsf{rt}, d)$, if the database $D + [x \mapsto y]$ has no collisions then T_1 is a subtree of T_2.*

Lemma 6.5. *Let D be a database and let $x \notin \mathrm{supp}(D)$. Let $D' = D + [x \mapsto y]$. Suppose that $D, D' \in \bar{\mathcal{P}}_{\mathsf{col}}$. For any $\mathsf{rt} \in Z$, if $\mathsf{Extract}(D, \mathsf{rt}, d) \neq \mathsf{Extract}(D', \mathsf{rt}, d)$, then $y \in \{\mathsf{rt}\} \cup S(D)$.*

Proof. Let $\mathsf{rt} \in Z$, and let $T_1 := \mathsf{Extract}(D, \mathsf{rt}, d)$ and $T_2 := \mathsf{Extract}(D + [x \mapsto y], \mathsf{rt}, d)$, and suppose that $T_1 \neq T_2$. By Lemma 6.4, we get that $T_1 \subsetneq T_2$. Therefore, there must be a vertex (z, s) that is marked in T_2 but unmarked in T_1. This means that $z \notin \mathrm{im}(D)$ but $z \in \mathrm{im}(D')$, and hence that $z = y$. If $(z, s) = (\mathsf{rt}, \emptyset)$, then $y = z = \mathsf{rt}$. If $(z, s) \neq (\mathsf{rt}, \emptyset)$, then there exists $z' \in Z$ such that either $(z, z') \in \mathrm{supp}(D)$ or $(z', z) \in \mathrm{supp}(D)$, and hence $y \in S(D)$. □

6.2 The oracle game for the Micali construction

We have summarized the Micali construction in Section 2.1. In order to define the oracle game that represents it, though, we need to recall in more detail how the verifier works. The Micali verifier $\mathcal{V}$ receives as input an instance $\mathtt{x}$ and proof $\pi = (\mathsf{rt}, (\mathsf{ap}_i)_{i\in[q]}, (v_i)_{i\in[q]})$, and has oracle access to a function $h\colon \{0,1\}^{2\lambda} \to \{0,1\}^{\lambda}$. First, $\mathcal{V}$ ensures that all authentication paths are for a Merkle tree with ℓ leaves (recall that ℓ is the PCP proof length). After that, $\mathcal{V}$ runs the underlying PCP verifier $\mathbf{V}$ on input $\mathtt{x}$ and random string $h(\mathsf{rt}, 0^{\lambda})$. When $\mathbf{V}$ makes its i-th query to location $j \in [\ell]$, $\mathcal{V}$ runs

$\mathsf{CheckPath}^h(\mathsf{rt}, \mathsf{ap}_i, v_i, j)$ and answers with v_i if this check passes (if not then it rejects). Once $\mathbf{V}$ halts, $\mathcal{V}$ accepts if and only if $\mathbf{V}$ accepted.

Given a proof π as above, we define

$$\mathsf{Expand}^h(\pi) := \Big(\big(\mathbf{x}_1, \ldots, \mathbf{x}_q, (\mathsf{rt}, 0^\lambda)\big), \big(\mathbf{y}_1, \ldots, \mathbf{y}_q, \mathsf{r}\big)\Big) \in (\{0,1\}^{2\lambda})^a \times (\{0,1\}^\lambda)^a$$

where $\mathsf{r} := h(\mathsf{rt}, 0^\lambda)$ is the derived randomness, each $(\mathbf{x}_i, \mathbf{y}_i) := \mathsf{Expand}^h(j_i, \mathsf{ap}_i, v_i)$ is an expanded authentication path, and $j_i \in [\ell]$ is the location of the i-th query of $\mathbf{V}(\mathbb{x}; \mathsf{r})$. Note that $a := O(q \log \ell)$.

Definition 6.6. *Let $\mathbf{V}$ be a PCP verifier for a relation $\mathcal{R}$ and let $\mathbb{x}$ be an instance. The* **Micali game** *for $(\mathbf{V}, \mathbb{x})$ is the set $G^{\mathsf{Mic}}_{\mathbf{V},\mathbb{x}} := \{\mathsf{Expand}^h(\pi) : h\colon \{0,1\}^{2\lambda} \to \{0,1\}^\lambda, \pi \in \{0,1\}^s, \mathcal{V}^h(\mathbb{x}, \pi) = 1\}$.*

6.3 Proof of Theorem 6.1

In order to show that the Micali construction is sound in the (quantum) random oracle model, it suffices to show that the oracle-game value of $G^{\mathsf{Mic}}_{\mathbf{V},\mathbb{x}}$ is small whenever $\mathbb{x} \notin \mathcal{L}(\mathcal{R})$.

Proposition 6.7. *The probability that a t-query classical adversary $\mathcal{A}$ causes the Micali verifier $\mathcal{V}$ to accept $\mathbb{x}$ is at most $\omega_{\mathsf{O}}(G^{\mathsf{Mic}}_{\mathbf{V},\mathbb{x}}, t + O(q \log \ell))$. The probability that a t-query quantum adversary $\mathcal{A}$ causes the Micali verifier $\mathcal{V}$ to accept $\mathbb{x}$ is at most $\omega^*_{\mathsf{O}}(G^{\mathsf{Mic}}_{\mathbf{V},\mathbb{x}}, t + O(q \log \ell))$.*

Proof. Let $\mathcal{B}$ be the algorithm that, when given access to an oracle h, runs $\mathcal{A}^h$ to obtain π and then outputs $\mathsf{Expand}^h(\pi)$. The query complexity of $\mathcal{B}$ is $t + O(q \log \ell)$ because $\mathcal{A}$ makes t queries and Expand makes $O(q \log \ell)$ queries. Let $((\mathbf{x}_i)_{i=1}^q, (\mathsf{rt}, 0^\lambda), (\mathbf{y}_i)_{i=1}^q, \mathsf{r})$ be the output of $\mathcal{B}$. By definition of Expand, $h((\mathbf{x}_i)_j) = (\mathbf{y}_i)_j$ for all i, j. Hence $\mathcal{B}^h$ wins $G^{\mathsf{Mic}}_{\mathbf{V},\mathbb{x}}$ if and only if $\mathcal{V}^h(\mathbb{x}, \pi) = 1$. □

We now need to establish an upper bound on $\omega^*_{\mathsf{O}}(G^{\mathsf{Mic}}_{\mathbf{V},\mathbb{x}}, t)$ given that $\mathbb{x} \notin \mathcal{L}(\mathcal{R})$. By Lemmas 4.9 and 5.13, it suffices to study the instability of the database property $\mathcal{P}_{\mathsf{Mic}} := \mathcal{P}_{G^{\mathsf{Mic}}_{\mathbf{V},\mathbb{x}}}$ associated with the oracle game $G^{\mathsf{Mic}}_{\mathbf{V},\mathbb{x}}$ (in the sense of Definition 5.2). In particular, we upper bound the instability of $\mathcal{P}_{\mathsf{Mic}}$ conditioned on the absence of collisions in the database. To simplify this task, we first "break down" $\mathcal{P}_{\mathsf{Mic}}$ as the union of simpler database properties, at least for databases that do not contain collisions. Below we assume that the PCP length ℓ equals 2^d for some $d \in \mathbb{N}$.

Definition 6.8. *For $\mathsf{rt} \in \{0,1\}^\lambda$, $\mathcal{P}_{\mathsf{rt}}$ is the set of databases D such that $(\mathsf{rt}, 0^\lambda) \in \mathrm{supp}(D)$ and for $T := \mathsf{Extract}(D, \mathsf{rt}, d)$ the PCP verifier $\mathbf{V}^{\mathsf{leaves}(T)}(\mathbb{x}; D(\mathsf{rt}, 0^\lambda))$ accepts (where the verifier immediately rejects if it queries a location i with $\mathsf{leaves}(T)_i = \perp$).*

Proposition 6.9. $\mathcal{P}_{\mathsf{Mic}} \subseteq \mathcal{P}_{\mathsf{col}} \cup \bigcup_{\mathsf{rt} \in \{0,1\}^\lambda} \mathcal{P}_{\mathsf{rt}}$.

Proof. By definition, a database D is in $\mathcal{P}_{\mathsf{Mic}}$ if and only if there exists $(\mathbf{x}, \mathbf{y}) \in G^{\mathsf{Mic}}_{\mathbf{V},\mathbb{x}}$ such that for all $i \in [k]$, $D(x_i) = y_i$. Suppose that $D \in \mathcal{P}_{\mathsf{Mic}}$, and let $w = ((\mathbf{x}_1, \ldots, \mathbf{x}_q, (\mathsf{rt}, 0^\lambda)), (\mathbf{y}_1, \ldots, \mathbf{y}_q, \mathsf{r})) \in G^{\mathsf{Mic}}_{\mathbf{V},\mathbb{x}}$ be a witness to this. If $D \in \mathcal{P}_{\mathsf{col}}$ then we are done, so suppose $D \in \bar{\mathcal{P}}_{\mathsf{col}}$. Then $D \in \mathcal{P}_{\mathsf{rt}}$, because if D has no collisions then the proof π extracted from root rt must be consistent with w; in particular $\mathbf{V}^{\mathsf{leaves}(T)}(\mathbb{x}; D(\mathsf{rt}, 0^\lambda))$ accepts. □

We now prove two lemmas about instability: first we bound the instability of $\mathcal{P}_{\mathsf{Mic}}$ when conditioned on databases having no collisions; then we bound the instability for the set of databases that contain collisions. We denote the latter property by $\mathcal{P}_{\mathsf{col}}$.

Lemma 6.10. *If* $\mathtt{x} \notin \mathcal{L}(\mathcal{R})$ *then* $\mathbf{I}(\mathcal{P}_{\mathsf{Mic}} \mid \bar{\mathcal{P}}_{\mathsf{col}}, t) < \varepsilon + (2t+1)/2^{\lambda}$.

Proof. Fix a database $D \in \bar{\mathcal{P}}_{\mathsf{Mic}} \cap \bar{\mathcal{P}}_{\mathsf{col}}$ with $|D| < t$ and fix an input $(x_0, x_1) \in (\{0,1\}^{\lambda})^2$. To bound the instability $\mathbf{I}(\mathcal{P}_{\mathsf{Mic}} \mid \bar{\mathcal{P}}_{\mathsf{col}}, t)$, it suffices to bound $\Pr_y[D + [(x_0, x_1) \mapsto y] \in \mathcal{P}_{\mathsf{Mic}} \cap \bar{\mathcal{P}}_{\mathsf{col}}]$ because if $D \in \mathcal{P}_{\mathsf{Mic}}$ then $D' \in \mathcal{P}_{\mathsf{Mic}}$ for any D' with $D \subseteq D'$.

Let $S' = \{r \in \{0,1\}^{\lambda} : \mathbf{V}^{\Pi}(\mathtt{x}; r) \text{ accepts}, \Pi = \mathsf{leaves}(\mathsf{Extract}(D, x_0, d))\}$. We will show that if $D + [(x_0, x_1) \mapsto y] \in \mathcal{P}_{\mathsf{Mic}} \cap \bar{\mathcal{P}}_{\mathsf{col}}$, then $y \in S(D) \cup \{x_0\} \cup S'$.

Let $D' := D + [(x_0, x_1) \mapsto y]$. Proposition 6.9 implies that if $D' \in \mathcal{P}_{\mathsf{Mic}} \cap \bar{\mathcal{P}}_{\mathsf{col}}$ then D' belongs to the set $\mathcal{P}_{\mathsf{rt}}$ for some $\mathsf{rt} \in \{0,1\}^{\lambda}$ and the database D does not belong to the set $\mathcal{P}_{\mathsf{rt}}$. Consider the two rooted binary trees $T_1 := \mathsf{Extract}(D, \mathsf{rt}, d)$ and $T_2 := \mathsf{Extract}(D', \mathsf{rt}, d)$. We have two cases.

- Case 1: $T_1 \neq T_2$. By Lemma 6.5 we get that $y \in S(D) \cup \{\mathsf{rt}\}$. Since $D' \in \mathcal{P}_{\mathsf{rt}}$, we see that $(\mathsf{rt}, 0^{\lambda}) \in \mathrm{supp}(D')$. Thus, either $\mathsf{rt} \in S(D)$ or $(x_0, x_1) = (\mathsf{rt}, 0^{\lambda})$. Hence, $y \in S(D) \cup \{x_0\}$.
- Case 2: $T_1 = T_2$. It must be that $(\mathsf{rt}, 0^{\lambda}) \notin \mathrm{supp}(D)$ but $(\mathsf{rt}, 0^{\lambda}) \in \mathrm{supp}(D')$. Hence $x_0 = \mathsf{rt}$ and $x_1 = 0^{\lambda}$. Letting $\Pi_1 := \mathsf{leaves}(T_1)$ and $\Pi_2 := \mathsf{leaves}(T_2)$, we have that $\Pi_1 = \Pi_2$. Now because $D' \in \mathcal{P}_{\mathsf{rt}}$, $\mathbf{V}^{\Pi_2}(\mathtt{x}; y)$ accepts, so $\mathbf{V}^{\Pi_1}(\mathtt{x}; y)$ accepts, and $y \in S'$ as $\Pi_1 = \mathsf{leaves}(\mathsf{Extract}(D, \mathsf{rt}, d))$ and $x_0 = \mathsf{rt}$.

We deduce that either $y \in S(D)$, $y = x_0$, or $y \in S'$. Note that these three sets are independent of rt, so that in particular this holds for all choices of rt. Hence, $\Pr_y[D + [(x_0, x_1) \mapsto y] \in \mathcal{P}_{\mathsf{Mic}} \cap \bar{\mathcal{P}}_{\mathsf{col}}] \leq \Pr_y[y \in S(D) \cup \{x_0\} \cup S']$. Clearly, $|S(D)| \leq 2|D| < 2t$ and $|\{x_0\}| = 1$. We have that $|S'| \leq \varepsilon 2^{\lambda}$, by the soundness of the PCP, as $\Pi = \mathsf{leaves}(\mathsf{Extract}(D, x_0, d))$ depends only on x_0 and D, and in particular is independent of y. Thus, we conclude that $\Pr_y[D + [(x_0, x_1) \mapsto y] \in \mathcal{P}_{\mathsf{Mic}} \cap \bar{\mathcal{P}}_{\mathsf{col}}] < \varepsilon + (2t+1)/2^{\lambda}$, so $\mathbf{I}(\mathcal{P}_{\mathsf{Mic}} \mid \bar{\mathcal{P}}_{\mathsf{col}}, t) < \varepsilon + (2t+1)/2^{\lambda}$, as required. □

Lemma 6.11. $\mathbf{I}(\mathcal{P}_{\mathsf{col}}, t) < t/2^{\lambda}$.

Proof. Fix $D \notin \mathcal{P}_{\mathsf{col}}$ with $|D| < t$ and $x \in \{0,1\}^{2\lambda}$. We have that $\Pr_y[D + [x \mapsto y] \in \mathcal{P}_{\mathsf{col}}] < t/2^{\lambda}$, as in order for this event to occur it must be that $y \in \mathrm{im}(D)$, and $|\mathrm{im}(D)| < t$. For any $D \in \mathcal{P}_{\mathsf{col}}$, we trivially have that $\Pr_y[D + [x \mapsto y] \notin \mathcal{P}_{\mathsf{col}}] = 0$, which completes the proof. □

We can now deduce the following statements:

$$\text{Lemma 4.9} \quad \longrightarrow \quad \sqrt{\omega^*_{\mathsf{O}}(G^{\mathsf{Mic}}_{\mathbf{V},\mathtt{x}}, t)} \leq \sqrt{\omega^*_{\mathsf{D}}(G^{\mathsf{Mic}}_{\mathbf{V},\mathtt{x}}, t)} + O\left(\sqrt{q \log \ell / 2^{\lambda}}\right) \ ,$$

$$\text{Lemma 5.13} \quad \longrightarrow \quad \omega^*_{\mathsf{D}}(G^{\mathsf{Mic}}_{\mathbf{V},\mathtt{x}}, t) \leq t^2 \cdot 6\big(\mathbf{I}(\mathcal{P}_{\mathsf{Mic}} \mid \bar{\mathcal{P}}_{\mathsf{col}}, t) + \mathbf{I}(\mathcal{P}_{\mathsf{col}}, t)\big) \ .$$

By combining the above statements, we get that $\omega^*_{\mathsf{O}}(G^{\mathsf{Mic}}_{\mathbf{V},\mathtt{x}}, t) = O(t^2\varepsilon + t^3/2^{\lambda} + q \log \ell / 2^{\lambda})$, from which we obtain the stated upper bound on the soundness error via Proposition 6.7.

Tightness. For simplicity we assume that the leaves in the Merkle tree commitments are salted, as is the case when one wishes to preserves zero knowledge (Section 7.1). We prove tightness of the $t^2\varepsilon$ term, which is the term that dominates in the expression for the soundness error.

Consider the following adversary. The adversary starts with a proof Π that is accepted by the PCP with probability ε and, by making $O(\ell)$ classical queries, builds a Merkle tree for Π. Let rt be the root of the tree. By changing the salt for one of the leaves in the tree and "hashing up", the adversary can construct another valid Merkle tree for Π with a newly sampled root rt' by making

only $O(\log \ell)$ additional queries. The adversary then uses Grover's algorithm [Gro96] to search for a new salt so that the new root rt' has $\mathbf{V}^{\Pi}(\mathtt{x}; h(\mathsf{rt}', 0^{\lambda})) = 1$.

With high probability over the choice of h, it holds that at least $(\varepsilon/2)$-fraction of the choices of salt will have a root rt' with $\mathbf{V}^{\Pi}(\mathtt{x}; h(\mathsf{rt}', 0^{\lambda})) = 1$. By running Grover's algorithm with t queries, the adversary thus finds a good salt with probability $\Omega(t^2\varepsilon)$. Each of the t queries made by Grover's algorithm corresponds to $O(t \log \ell)$ queries to h. The adversary makes $O(\ell)$ queries initially to construct the tree, so this gives us a quantum adversary that wins with probability $\Omega(t^2\varepsilon)$ by making $O(t \log \ell + \ell)$ queries.

7 zkSNARKs in the QROM

We prove the existence of zkSNARKs that are unconditionally secure in the quantum random oracle model, based on the Micali construction.

First, we prove that the Micali construction is a zero knowledge non-interactive argument of knowledge if the underlying PCP is both honest-verifier zero knowledge and a proof of knowledge.

Theorem 7.1. *Let $(\mathbf{P}, \mathbf{V})$ be a PCP for a relation $\mathcal{R}$ that is honest-verifier zero knowledge and has knowledge error k, and suppose that $(\mathbf{P}, \mathbf{V})$ has proof length ℓ and query complexity q. The Micali construction, when based on $(\mathbf{P}, \mathbf{V})$, is a non-interactive argument in the QROM for $\mathcal{R}$ that is (statistical) zero knowledge and is an argument of knowledge with extraction probability $\kappa(\mu, \lambda, t + O(q \log \ell)) = \Omega(\mu - t^2\mathsf{k} - t^3/2^\lambda)$.*

We separately prove zero knowledge and proof of knowledge, respectively in Lemma 7.3 and in Lemma 7.4 below, thereby establishing the above theorem. Observe that the extraction probability κ achieved in the theorem statement *implies* the soundness bound given Theorem 6.1, because the extraction probability is positive when the verifier's acceptance probability μ is $\Omega(t^2\mathsf{k} + t^3/2^\lambda)$.

If we additionally ensure that the PCP in the Micali construction has small query complexity and verifier running time (via standard PCPs), then we obtain zkSNARKs that are unconditionally secure in the quantum random oracle model (see definition in Section 3.3).

Corollary 7.2. *There exist zkSNARKs for $\mathsf{NTIME}(T(n))$ in the quantum random oracle model.*

Proof. The PCP in [BFLS91] supports $\mathsf{NTIME}(T(n))$ with query complexity $\mathsf{poly}(\log T(n))$, a prover that runs in time $\mathsf{poly}(n, T(n))$, and a verifier that runs in time $\mathsf{poly}(n, \log T(n))$. The PCP is a proof of knowledge, and can be modified to achieve honest-verifier zero knowledge and a negligible soundness error [DFKNS92; KPT97]. We can then apply Theorem 7.1 to the resulting PCP. □

7.1 Zero knowledge

In the classical setting, the Micali construction achieves statistical zero knowledge if the underlying PCP is *honest-verifier zero knowledge* (see definition in Section 3.4), and if leaves in the Merkle tree are salted to ensure statistical hiding of unrevealed leaves [IMSX15; BCS16]. The following lemma states that the same is true in the quantum setting.

Lemma 7.3. *The construction of Micali (modified to use salted Merkle trees), when based on an honest-verifier zero knowledge PCP, is a zero knowledge non-interactive argument in the QROM.*

Proof. A straightforward adaptation of Theorem 6.1 establishes soundness in the case where leaves in the Merkle tree are salted by the prover with fresh randomness. Moreover, the classical zero knowledge guarantee of the Micali construction is statistical, i.e., it holds against verifiers that are unbounded, in computation *and* in queries to the random oracle. Such verifiers can in particular simulate any quantum adversary (including queries in superposition). This means that the zero knowledge guarantee also holds in the quantum setting. (Indeed, the definition of zero knowledge that we provide in Section 3.3 for non-interactive arguments in the QROM does not even have to mention quantum algorithms; it requires the real view and ideal view to be statistically close.) □

7.2 Proof of knowledge

In the classical setting, the Micali construction is an argument of knowledge if the underlying PCP is a proof of knowledge (see definition in Section 3.4). The following lemma states that the same is true in the quantum setting.

Lemma 7.4. *The Micali construction, when based on a PCP of knowledge with knowledge error* k*, is a non-interactive argument of knowledge in the QROM with extraction probability* $\Omega(\mu - t^2\mathsf{k} - t^3/2^\lambda)$ *against quantum attackers that make at most* $t - O(q \log \ell)$ *queries and win with probability at least* μ*. In particular, the extraction probability is positive for large enough* $\mu = \Omega(t^2\mathsf{k} + t^3/2^\lambda)$*.*

We prove the lemma by building on ideas in the proof of Section 6, and also on some additional generic machinery. Below we assume that the proof length ℓ equals 2^d for some $d \in \mathbb{N}$.

Let $\mathcal{A}$ be a t'-query quantum adversary $\mathcal{A}$ that causes the Micali verifier to accept an instance $\mathtt{x}$ with probability at least μ. Following the proof of Proposition 6.7, we obtain a t-query quantum adversary $\mathcal{B}$ that wins the Micali oracle game $G^{\mathsf{Mic}}_{\mathbf{V},\mathtt{x}}$ with probability at least μ, for $t := t' + O(q \log \ell)$. This transformation can be efficiently performed with black-box access to $\mathcal{A}$, because $\mathcal{B}$ merely extends $\mathcal{A}$ with some classical computation that depends on $\mathcal{A}$'s (classical) output.

We now describe the quantum extractor $\mathcal{E}$ for the Micali game, and then argue why it works.

$\mathcal{E}^{\mathcal{A}}(\mathtt{x}, 1^{t'}, 1^\lambda)$:

1. Set $t := t' + O(q \log \ell)$.
2. Compute the quantum state $|\mathsf{Sim}^*(\mathcal{B})\rangle$, by simulating $\mathcal{B}$.
3. Measure the database register of $|\mathsf{Sim}^*(\mathcal{B})\rangle$ to get a database D.
4. Run $\mathtt{w} \leftarrow \mathbf{E}(\mathtt{x}, 0^\ell)$, where 0^ℓ denotes the all 0's proof. If $\mathtt{w}$ is a valid witness, return it, otherwise continue.
5. For each $\mathsf{rt} \in \mathrm{im}(D)$:
   Run $T \leftarrow \mathsf{Extract}(D, \mathsf{rt}, d)$,
   Set Π_T equal to $\mathsf{leaves}(T)$ on all points where $\mathsf{leaves}(T)$ is not $\perp$, and 0 otherwise.
   Run $\mathtt{w} \leftarrow \mathbf{E}(\mathtt{x}, \Pi_T)$. If $\mathtt{w}$ is a valid witness, return it, otherwise continue.

Let $\mathcal{P}_{\mathbf{E},\mathsf{rt}}$ be the set of databases D where running $T \leftarrow \mathsf{Extract}(D, \mathsf{rt}, d)$ and then $\mathtt{w} \leftarrow \mathbf{E}(\mathtt{x}, \Pi_T)$ yields $(\mathtt{x}, \mathtt{w}) \in \mathcal{R}$, where Π_T is defined as above. Note that $\mathcal{E}^{\mathcal{A}}(\mathtt{x}, 1^{t'}, 1^\lambda)$ outputs a valid witness if and only if $D \in \cup_{\mathsf{rt} \in \{0,1\}^\lambda} \mathcal{P}_{\mathbf{E},\mathsf{rt}}$. This is because if $\mathsf{rt} \in \mathrm{im}(D)$ then $\mathcal{E}$ tries to extract from rt, and if $\mathsf{rt} \notin \mathrm{im}(D)$ then $T \leftarrow \mathsf{Extract}(D, \mathsf{rt}, d)$ has one vertex (just rt), and so $\Pi_T = 0^\ell$.

We can lower bound the probability that the extractor succeeds as follows:

$$
\begin{aligned}
&\Pr\Big[\mathcal{E}^{\mathcal{A}}(\mathtt{x}, 1^{t'}, 1^\lambda) \text{ outputs a valid witness}\Big] = \Pr\Big[D \in \cup_{\mathsf{rt} \in \{0,1\}^\lambda} \mathcal{P}_{\mathbf{E},\mathsf{rt}}\Big] \\
&\geq \Pr[D \in \cup_{\mathsf{rt}} \mathcal{P}_{\mathbf{E},\mathsf{rt}} \cap \mathcal{P}_{\mathsf{Mic}}] = \Pr[D \in \mathcal{P}_{\mathsf{Mic}}] - \Pr\big[D \in \cap_{\mathsf{rt}} \bar{\mathcal{P}}_{\mathbf{E},\mathsf{rt}} \cap \mathcal{P}_{\mathsf{Mic}}\big] \\
&\geq \Pr[D \in \mathcal{P}_{\mathsf{Mic}}] - \Pr\Big[D \in \cap_{\mathsf{rt} \in \{0,1\}^\lambda} \bar{\mathcal{P}}_{\mathbf{E},\mathsf{rt}} \cap ((\cup_{\mathsf{rt}} \mathcal{P}_{\mathsf{rt}}) \cup \mathcal{P}_{\mathsf{col}})\Big] \\
&\geq \Pr[D \in \mathcal{P}_{\mathsf{Mic}}] - \Pr\big[D \in \mathcal{P}_{\mathsf{col}} \cup (\cup_{\mathsf{rt}} (\mathcal{P}_{\mathsf{rt}} \cap \bar{\mathcal{P}}_{\mathbf{E},\mathsf{rt}}))\big] \\
&\geq \Pr[\mathcal{B} \text{ wins the Micali database game}] - t^2 \cdot 6\mathbf{I}(\mathcal{P}_{\mathsf{col}} \cup (\cup_{\mathsf{rt}} (\mathcal{P}_{\mathsf{rt}} \cap \bar{\mathcal{P}}_{\mathbf{E},\mathsf{rt}})), t) \ ,
\end{aligned}
$$

where the second inequality follows by Proposition 6.9 and the third by the fact that $\cap_{\mathsf{rt} \in \{0,1\}^\lambda} \bar{\mathcal{P}}_{\mathbf{E},\mathsf{rt}} \cap ((\cup_{\mathsf{rt}} \mathcal{P}_{\mathsf{rt}}) \cup \mathcal{P}_{\mathsf{col}}) \subseteq \mathcal{P}_{\mathsf{col}} \cup (\cup_{\mathsf{rt}} (\mathcal{P}_{\mathsf{rt}} \cap \bar{\mathcal{P}}_{\mathbf{E},\mathsf{rt}}))$. This latter property can be interpreted as follows: either there is a collision, or there is some $\mathsf{rt} \in \mathrm{supp}(D)$ such that we can produce a correct SNARG proof rooted at rt but the PCP extractor cannot extract a valid witness from the PCP proof rooted at rt.

We have that the probability that $\mathcal{B}$ wins the quantum database game of $G^{\mathsf{Mic}}_{\mathbf{V},\mathtt{x}}$ is $\Omega(\mu - q\log\ell/2^\lambda)$ by Lemma 4.9. Hence, in order to complete the proof it suffices to show the following proposition.

Proposition 7.5. $\mathbf{I}(\mathcal{P}_{\mathsf{col}} \cup (\cup_{\mathsf{rt}}(\mathcal{P}_{\mathsf{rt}} \cap \bar{\mathcal{P}}_{\mathbf{E},\mathsf{rt}})), t) < \mathsf{k} + (2t+1)/2^\lambda$.

Proof. We have that $\mathbf{I}(\mathcal{P}_{\mathsf{col}} \cup (\cup_{\mathsf{rt}}(\mathcal{P}_{\mathsf{rt}} \cap \bar{\mathcal{P}}_{\mathbf{E},\mathsf{rt}})), t) \leq \mathbf{I}(\mathcal{P}_{\mathsf{col}}, t) + \mathbf{I}(\cup_{\mathsf{rt}}(\mathcal{P}_{\mathsf{rt}} \cap \bar{\mathcal{P}}_{\mathbf{E},\mathsf{rt}}) \mid \bar{\mathcal{P}}_{\mathsf{col}}, t)$. By Lemma 6.11, we have that $\mathbf{I}(\mathcal{P}_{\mathsf{col}}, t) < t/2^\lambda$.

We now bound $\mathbf{I}(\cup_{\mathsf{rt}}(\mathcal{P}_{\mathsf{rt}} \cap \bar{\mathcal{P}}_{\mathbf{E},\mathsf{rt}}) \mid \bar{\mathcal{P}}_{\mathsf{col}}, t)$. Let $D \in \bar{\mathcal{P}}_{\mathsf{col}}$ be a database, and fix $x = (x_0, x_1) \notin \mathrm{supp}(D)$. Let $S' = \{r \in \{0,1\}^\lambda : \mathbf{V}^\Pi(\mathtt{x}; r) \text{ accepts}, \Pi = \mathsf{leaves}(\mathsf{Extract}(D, x_0, d))\}$.

Suppose first that $D \notin \cup_{\mathsf{rt}}(\mathcal{P}_{\mathsf{rt}} \cap \bar{\mathcal{P}}_{\mathbf{E},\mathsf{rt}})$; we bound $\Pr_y[D + [x \mapsto y] \in \cup_{\mathsf{rt}}(\mathcal{P}_{\mathsf{rt}} \cap \bar{\mathcal{P}}_{\mathbf{E},\mathsf{rt}}) \cap \bar{\mathcal{P}}_{\mathsf{col}}]$. Fix a y such that this holds, and let $D' := D + [x \mapsto y]$. Then there exists rt^* such that $D' \in \mathcal{P}_{\mathsf{rt}^*} \cap \bar{\mathcal{P}}_{\mathbf{E},\mathsf{rt}^*}$ and $D \in \bar{\mathcal{P}}_{\mathsf{rt}^*} \cup \mathcal{P}_{\mathbf{E},\mathsf{rt}^*}$. There are two cases.

- Case 1: $D \in \mathcal{P}_{\mathbf{E},\mathsf{rt}^*}$. Since $D' \notin \mathcal{P}_{\mathbf{E},\mathsf{rt}^*}$, we get that $\mathsf{Extract}(D, \mathsf{rt}^*, d) \neq \mathsf{Extract}(D', \mathsf{rt}^*, d)$. Hence, by Lemma 6.5 we get that $y \in S(D)$ or $y = \mathsf{rt}^*$. Since $D' \in \mathcal{P}_{\mathsf{rt}^*}$, we have that $(\mathsf{rt}^*, 0^\lambda) \in \mathrm{supp}(D')$, and hence either $\mathsf{rt}^* \in S(D)$ or $\mathsf{rt}^* = x_0$. Thus, either $y \in S(D)$ or $y = x_0$.

- Case 2: $D \in \bar{\mathcal{P}}_{\mathsf{rt}^*} \cap \bar{\mathcal{P}}_{\mathbf{E},\mathsf{rt}^*}$. We have that $D' \in \mathcal{P}_{\mathsf{rt}^*}$. If $\mathsf{Extract}(D, \mathsf{rt}^*, d) \neq \mathsf{Extract}(D', \mathsf{rt}^*, d)$, then again we get that $y \in S(D)$ or $y = x_0$. If $\mathsf{Extract}(D, \mathsf{rt}^*, d) = \mathsf{Extract}(D', \mathsf{rt}^*, d)$, then by Case 2 of Lemma 6.10 we get that $x_0 = \mathsf{rt}^*$ and $y \in S'$. In particular, this implies that $D \in \bar{\mathcal{P}}_{\mathbf{E},x_0}$.

We conclude that either $y \in S(D) \cup \{x_0\}$, or $D \in \bar{\mathcal{P}}_{\mathbf{E},x_0}$ and $y \in S'$. We know that $|S(D) \cup \{x_0\}| < 2t+1$. We show that $|S'| \leq \mathsf{k} \cdot 2^\lambda$ if $D \in \bar{\mathcal{P}}_{\mathbf{E},x_0}$. Let $T = \mathsf{Extract}(D, x_0, d)$, $\Pi = \mathsf{leaves}(T)$, and Π_T be as defined earlier. Suppose that $|S'|/2^\lambda = \Pr_r[\mathbf{V}^\Pi(\mathtt{x}; r) \text{ accepts}] > \mathsf{k}$. Then, $\Pr_r[\mathbf{V}^{\Pi_T}(\mathtt{x}; r) \text{ accepts}] \geq \Pr_r[\mathbf{V}^\Pi(\mathtt{x}; r) \text{ accepts}] > \mathsf{k}$ and so $\mathbf{E}(\mathtt{x}, \Pi_T)$ outputs a valid witness for $\mathtt{x}$. Therefore, $D \in \mathcal{P}_{\mathbf{E},x_0}$, a contradiction, and so $|S'| \leq \mathsf{k} \cdot 2^\lambda$.

We thus conclude that if $D \in \mathcal{P}_{\mathbf{E},x_0}$ then $\Pr_y[D' \in \bar{\mathcal{P}}_{\mathsf{col}} \cap (\cup_{\mathsf{rt}}(\mathcal{P}_{\mathsf{rt}} \cap \bar{\mathcal{P}}_{\mathbf{E},\mathsf{rt}}))] < (2t+1)/2^\lambda$, and if $D \in \bar{\mathcal{P}}_{\mathbf{E},x_0}$ then $\Pr_y[D' \in \bar{\mathcal{P}}_{\mathsf{col}} \cap (\cup_{\mathsf{rt}}(\mathcal{P}_{\mathsf{rt}} \cap \bar{\mathcal{P}}_{\mathbf{E},\mathsf{rt}}))] < \mathsf{k} + (2t+1)/2^\lambda$. Hence, $\mathrm{flip}(\cup_{\mathsf{rt}}(\mathcal{P}_{\mathsf{rt}} \cap \bar{\mathcal{P}}_{\mathbf{E},\mathsf{rt}}) \mid \bar{\mathcal{P}}_{\mathsf{col}}, t) < \mathsf{k} + (2t+1)/2^\lambda$.

Now suppose that $D \in \cup_{\mathsf{rt}}(\mathcal{P}_{\mathsf{rt}} \cap \bar{\mathcal{P}}_{\mathbf{E},\mathsf{rt}})$; we bound $\Pr_y[D + [x \mapsto y] \in \overline{\cup_{\mathsf{rt}}(\mathcal{P}_{\mathsf{rt}} \cap \bar{\mathcal{P}}_{\mathbf{E},\mathsf{rt}})} \cap \bar{\mathcal{P}}_{\mathsf{col}}]$. Fix a y such that this holds, and let $D' := D + [x \mapsto y]$. Then there exists rt^* such that $D \in \mathcal{P}_{\mathsf{rt}^*} \cap \bar{\mathcal{P}}_{\mathbf{E},\mathsf{rt}^*}$ and $D' \in \bar{\mathcal{P}}_{\mathsf{rt}^*} \cup \mathcal{P}_{\mathbf{E},\mathsf{rt}^*}$. There are two cases.

- Case 1: $D' \in \bar{\mathcal{P}}_{\mathsf{rt}^*}$. This is impossible: since $D \in \mathcal{P}_{\mathsf{rt}^*} \cap \bar{\mathcal{P}}_{\mathsf{col}}$ and $D' \in \bar{\mathcal{P}}_{\mathsf{col}}$, it follows by Lemma 6.4 that $D' \in \mathcal{P}_{\mathsf{rt}^*}$.

- Case 2: $D' \in \mathcal{P}_{\mathsf{rt}^*} \cap \mathcal{P}_{\mathbf{E},\mathsf{rt}^*}$. Since $D \in \bar{\mathcal{P}}_{\mathbf{E},\mathsf{rt}^*}$ and $D' \in \mathcal{P}_{\mathbf{E},\mathsf{rt}^*}$, it follows that $\mathsf{Extract}(D, \mathsf{rt}^*, d) \neq \mathsf{Extract}(D', \mathsf{rt}^*, d)$. Hence, by Lemma 6.5 we get that $y \in S(D)$ or $y = \mathsf{rt}^*$. Since $D \in \mathcal{P}_{\mathsf{rt}^*}$, it follows that $(\mathsf{rt}^*, 0^\lambda) \in \mathrm{supp}(D)$, so $\mathsf{rt}^* \in S(D)$. Hence, $y \in S(D)$.

We conclude that $\mathrm{flip}(\overline{\cup_{\mathsf{rt}}(\mathcal{P}_{\mathsf{rt}} \cap \bar{\mathcal{P}}_{\mathbf{E},\mathsf{rt}})} \mid \bar{\mathcal{P}}_{\mathsf{col}}, t) < 2t/2^\lambda$.

Putting it together, we get that $\mathbf{I}(\cup_{\mathsf{rt}}(\mathcal{P}_{\mathsf{rt}} \cap \bar{\mathcal{P}}_{\mathbf{E},\mathsf{rt}}) \mid \bar{\mathcal{P}}_{\mathsf{col}}, t) < \mathsf{k} + (2t+1)/2^\lambda$. □

8 The BCS construction in the QROM

We prove that non-interactive arguments obtained via the BCS construction applied to IOPs with round-by-round soundness are unconditionally secure in the quantum random oracle model. Moreover, if the IOP is honest-verifier zero knowledge and has round-by-round knowledge then BCS construction yields a zero knowledge non-interactive argument of knowledge. This section is organized as follows: in Section 8.1 we define IOPs; in Section 8.2 we define the oracle game that corresponds to the BCS construction; in Section 8.3 we define round-by-round soundness and round-by-round knowledge; in Section 8.4 we state our result; and in Section 8.5 we prove our result.

8.1 Interactive oracle proofs

A (public-coin) *interactive oracle proof* (IOP) [BCS16; RRR16] is a multi-round extension of the notion of a PCP where in each round the verifier sends a random message m_i and the prover replies with a proof string Π_i. After the interaction, the verifier queries the proof strings $(\Pi_1, \Pi_2, \dots)$ received from the prover, and then accepts or rejects. A PCP is then a non-interactive IOP.

An IOP for a relation $\mathcal{R}$ with soundness error ϵ is a pair of polynomial-time interactive algorithms $(\mathbf{P}, \mathbf{V})$ for which the following holds.

- **Completeness.** For every instance-witness pair $(\mathtt{x}, \mathtt{w}) \in \mathcal{R}$, the probability that $\mathbf{P}(\mathtt{x}, \mathtt{w})$ convinces $\mathbf{V}(\mathtt{x})$ to accept is 1.
- **Soundness.** For every instance $\mathtt{x} \notin \mathcal{L}(\mathcal{R})$ and unbounded malicious prover $\tilde{\mathbf{P}}$, the probability that $\tilde{\mathbf{P}}$ convinces $\mathbf{V}(\mathtt{x})$ to accept is at most ϵ.

Like the IP model, a fundamental measure of efficiency is the round complexity k. Like the PCP model, two additional fundamental measures of efficiency are the *proof length* ℓ, which is the total number of alphabet symbols in all of the prover's messages, and the *query complexity* q, which is the total number of locations queried by the verifier across all of the prover's messages.

Transcripts. We denote by tr a *transcript* of $(\mathbf{P}, \mathbf{V})$, which means all verifier messages and proof strings up to a point where either the prover or the verifier is about to move. In more detail, tr is one of the following cases (with Π_0 defined as the empty string):

- the empty transcript, denoted by the symbol $\emptyset$ (the verifier is about to move);
- a partial transcript where the prover is about to move, which is a pair of the form $(\mathbf{m}, \mathbf{\Pi}) = ((m_1, \dots, m_i), (\Pi_1, \dots, \Pi_{i-1}))$ for some $i \in [k]$;
- a partial transcript where the verifier is about to move, which is a pair of the form $(\mathbf{m}, \mathbf{\Pi}) = ((m_1, \dots, m_i), (\Pi_1, \dots, \Pi_i))$ for some $i \in [k]$;
- a full transcript, which is a pair of the form $(\mathbf{m}, \mathbf{\Pi}) = ((m_1, \dots, m_{k+1}), (\Pi_1, \dots, \Pi_k))$.

We allow the proof strings Π_i to be partial functions, i.e., databases (Section 3.5).

Given a transcript $\mathsf{tr} = (\mathbf{m}, \mathbf{\Pi}) = ((m_1, \dots, m_i), (\Pi_1, \dots, \Pi_{i-1}))$ where the prover is about to move, we let $\mathsf{tr} \| \Pi$ denote the transcript $((m_1, \dots, m_i), (\Pi_1, \dots, \Pi_{i-1}, \Pi))$. Similarly, given a transcript $\mathsf{tr} = (\mathbf{m}, \mathbf{\Pi}) = ((m_1, \dots, m_i), (\Pi_1, \dots, \Pi_i))$ where the verifier is about to move, we let $\mathsf{tr} \| m$ denote the transcript $(\mathbf{m}, \mathbf{\Pi}) = ((m_1, \dots, m_i, m), (\Pi_1, \dots, \Pi_i))$.

8.2 The BCS construction and its oracle game

The BCS construction transforms a public-coin IOP into a corresponding non-interactive argument, using random oracles. Informally, this is achieved by repeating the idea of the Micali construction

in each round, and then forcing rounds to be in order via a hash chain; see [BCS16] for details.

In order to define the oracle game that represents the BCS construction, we need to recall how its verifier works. The BCS verifier $\mathcal{V}$ receives as input an instance $\mathbb{x}$ and proof

$$\pi = (\sigma, (\mathsf{rt}_j)_{j\in[k]}, (\mathsf{ap}_i)_{i\in[q]}, (v_i)_{i\in[q]}) \ ,$$

and has oracle access to a function $h\colon \{0,1\}^{2\lambda} \to \{0,1\}^{\lambda}$. First, $\mathcal{V}$ checks the hash chain: it initializes $\sigma_0 := 0^{\lambda}$; then, for each $j \in [k]$, it computes $m_j := h(\sigma_{j-1}, \text{“}\mathsf{j}\text{”})$ and $\sigma_j := h(m_j, \mathsf{rt}_j)$, where $\text{“}\mathsf{j}\text{”} \in \{0,1\}^{\lambda}$ is some unique encoding of the integer j; finally, $\mathcal{V}$ checks that $\sigma = \sigma_k$. Next, $\mathcal{V}$ computes the randomness for the query phase as $m_{k+1} := h(\sigma, \text{“}\mathsf{k{+}1}\text{”})$. The randomness for the IOP verifier $\mathbf{V}$ is $\mathbf{m} := (m_1, \ldots, m_k, m_{k+1})$. Finally, $\mathcal{V}$ simulates the IOP verifier $\mathbf{V}$ on $\mathbf{m}$ by answering its queries using the values contained in π. In more detail, it runs $\mathbf{V}(\mathbb{x}; \mathbf{m})$ and answers its i-th query, for $i \in [q]$, as follows: if the query is to the j-th proof string at a location s then check that $\mathsf{CheckPath}^h(\mathsf{rt}_j, s, v_i, \mathsf{ap}_i)$ accepts and answer with v_i if this check passes (else reject). Once $\mathbf{V}$ halts, $\mathcal{V}$ accepts if and only if $\mathbf{V}$ accepted. (The BCS verifier $\mathcal{V}$ should also do basic syntactic checks such as ensuring that if the j-th proof string is supposed to have length ℓ_j then authentication paths relative to the j-th root have an appropriate length.)

Given a BCS proof π as above, we define

$$\mathsf{BCSExpand}^h(\pi) := \Big(\big((\mathbf{x}_i)_{i\in[q]}, (\sigma_{j-1}, \text{“}\mathsf{j}\text{”})_{j\in[k+1]}, (m_j, \mathsf{rt}_j)_{j\in[k]}\big), \big((\mathbf{y}_i)_{i\in[q]}, (m_j)_{j\in[k+1]}, (\sigma_j)_{j\in[k]}\big)\Big)$$

where $\sigma_0 := 0^{\lambda}$, $m_j := h(\sigma_{j-1}, \text{“}\mathsf{j}\text{”})$ for $j \in [k+1]$ is the derived message of the verifier in the j-th round, $\sigma_j := h(m_j, \mathsf{rt}_j)$ for $j \in [k]$, each $(\mathbf{x}_i, \mathbf{y}_i) := \mathsf{Expand}^h(s_i, \mathsf{ap}_i, v_i)$ is an expanded authentication path, and s_i is the location of the i-th query of $\mathbf{V}(\mathbb{x}; \mathbf{m})$. Note that $\mathsf{BCSExpand}^h(\pi) \in (\{0,1\}^{2\lambda})^a \times (\{0,1\}^{\lambda})^a$ for $a := O(q \log \ell)$ where $\ell := \sum_{j=1}^{k} \ell_j$ is the proof length of the IOP.

Definition 8.1. *Let $\mathbf{V}$ be an IOP verifier for a relation $\mathcal{R}$ and let $\mathbb{x}$ be an instance. The* **BCS game** *for $(\mathbf{V}, \mathbb{x})$ is the set $G^{\mathrm{BCS}}_{\mathbf{V},\mathbb{x}} := \{\mathsf{BCSExpand}^h(\pi) : h\colon \{0,1\}^{2\lambda} \to \{0,1\}^{\lambda}, \pi \in \{0,1\}^{s}, \mathcal{V}^h(\mathbb{x}, \pi) = 1\}$.*

Remark 8.2. Our definition of the BCS construction uses a slightly different hash chain from the one used in [BCS16]. This is due to our proof technique. Soundness against quantum attackers also holds for the hash chain in [BCS16], provided that the underlying IOP satisfies a slightly stronger variant of round-by-round soundness. This issue is discussed in Section 8.6.

8.3 Round-by-round soundness and knowledge

We define round-by-round soundness (adapted from [CCHLRR18]) and round-by-round knowledge (introduced in this work), which are the notions of soundness and knowledge that we consider in Theorem 8.6 below. We first define an IOP state function, and then give the two definitions.

Definition 8.3. *Let $(\mathbf{P}, \mathbf{V})$ be an IOP for a relation $\mathcal{R}$. A* **state function** *for $(\mathbf{P}, \mathbf{V})$ is a deterministic (possibly inefficient) function state that receives as input an instance $\mathbb{x}$ and a transcript $\mathsf{tr} = (\mathbf{m}, \mathbf{\Pi})$ and outputs a bit for which the following holds.*

- Empty transcript: *if $\mathsf{tr} = \emptyset$ is the empty transcript then $\mathsf{state}(\mathbb{x}, \mathsf{tr}) = 0$.*
- Prover moves: *if tr is a transcript where the prover is about to move and $\mathsf{state}(\mathbb{x}, \mathsf{tr}) = 0$, then*

$$\forall\ \Pi,\ \mathsf{state}(\mathbb{x}, \mathsf{tr} \| \Pi) = 0 \ .$$

- Full transcript: *if* tr *is a full transcript and* $\mathsf{state}(\mathbb{x}, \mathsf{tr}) = 0$*, then* $\mathbf{V}^{\mathbf{\Pi}}(\mathbb{x}; \mathbf{m}) = 0$.

Definition 8.4 ([CCHLRR18] adapted to IOP)**.** *An IOP* $(\mathbf{P}, \mathbf{V})$ *for a relation* $\mathcal{R}$ *has* **round-by-round soundness error** ϵ *if there exists a state function* state *such that for all* $\mathbb{x} \notin \mathcal{L}(\mathcal{R})$ *and every transcript* tr *where the verifier is about to move and* $\mathsf{state}(\mathbb{x}, \mathsf{tr}) = 0$ *it holds that*

$$\Pr_m[\mathsf{state}(\mathbb{x}, \mathsf{tr} \| m) = 1] \leq \epsilon \ .$$

Definition 8.5. *An IOP* $(\mathbf{P}, \mathbf{V})$ *for a relation* $\mathcal{R}$ *has* **round-by-round knowledge error** k *if there exists a polynomial-time extractor* $\mathbf{E}$ *and state function* state *such that for all* $\mathbb{x}$ *and every transcript* tr *where the verifier is about to move and* $\mathsf{state}(\mathbb{x}, \mathsf{tr}) = 0$*, if* $\Pr_m[\mathsf{state}(\mathbb{x}, \mathrm{tr} \,\|m) = 1] > \mathsf{k}$ *then* $(\mathbb{x}, \mathbf{E}(\mathbb{x}, \mathsf{tr})) \in \mathcal{R}$.

8.4 Our result

We prove that the BCS construction is sound in the quantum random oracle model if the underlying IOP has round-by-round soundness, is a proof of knowledge if the IOP has round-by-round knowledge, and is statistical zero knowledge if the IOP is honest-verifier zero knowledge. Our result establishes the post-quantum security of many zkSNARKs of practical interest, which rely on zero knowledge IOPs (and IPs) that have round-by-round soundness and round-by-round knowledge.

Theorem 8.6. *Let* $(\mathbf{P}, \mathbf{V})$ *be an IOP for a relation* $\mathcal{R}$ *with proof length* ℓ *and query complexity* q*. Then the BCS construction, when based on* $(\mathbf{P}, \mathbf{V})$*, is a non-interactive argument for* $\mathcal{R}$ *such that:*

1. *(Soundness) if* $(\mathbf{P}, \mathbf{V})$ *has round-by-round soundness error* ϵ*, then the argument has soundness error* $O(t^2\epsilon + t^3/2^\lambda)$ *against quantum attackers that make at most* $t - O(q \log \ell)$ *queries to the random oracle.*

2. *(Knowledge) if* $(\mathbf{P}, \mathbf{V})$ *has round-by-round knowledge error* k*, then the argument is an argument of knowledge with extraction probability* $\Omega(\mu - t^2\mathsf{k} - t^3/2^\lambda)$ *against quantum attackers that make at most* $t - O(q \log \ell)$ *queries and win with probability at least* μ*. The extraction probability is positive for large enough* $\mu = \Omega(t^2\mathsf{k} + t^3/2^\lambda)$.

3. *(Zero Knowledge) if* $(\mathbf{P}, \mathbf{V})$ *is honest-verifier zero knowledge, then the argument is (statistical) zero knowledge.*

8.5 Proof of Theorem 8.6

We begin by noting that preservation of zero knowledge is straightforward. This is because, analogous to the Micali construction Section 7.1, the BCS construction achieves *statistical* zero knowledge provided the underlying IOP is honest-verifier zero knowledge. See [BCS16] for details.

Next, we prove soundness in Section 8.5.1 and knowledge in Section 8.5.2. To start, we define two extraction algorithms that operate on databases, which are similar to those defined in Section 6.1.

Extracting hash chains. We define an algorithm $\mathsf{HExtract}$ that is used to extract a hash chain ending at a chosen entry in a database. $\mathsf{HExtract}$ receives a database $D \colon Z \times Z \rightharpoonup Z$, an end value $y \in Z$, and a length bound k, and outputs an integer ℓ and two tuples $(x_1, \ldots, x_\ell), (y_0, \ldots, y_\ell)$ where $\ell \in [k]$, each x_i and $y_i \in Z$, $D(y_{i-1}, x_i) = y_i$ for each $i \in [\ell]$, and $y_\ell = y$.

Formally, the algorithm $\mathsf{HExtract}$ is defined as follows.

$\mathsf{HExtract}(D, y, k)$:

1. Initialize two lists $\mathsf{chain} = (y)$ and $\mathsf{input} = ()$, and a counter $\ell \leftarrow 0$.
2. While $\ell < k$:
   (a) If y is the result of a collision ($D(z, x) = D(z', x') = y$ for distinct $(z, x), (z', x')$ in Z^2), return $\perp$.
   (b) Let $z, x \in Z$ be the unique values such that $D(z, x) = y$. If none exist, then exit the loop.
   (c) Prepend z to chain: $\mathsf{chain} := \{z\} \cup \mathsf{chain}$.
   (d) Prepend x to input: $\mathsf{input} := \{x\} \cup \mathsf{input}$.
   (e) Update $y \leftarrow z$ and increment $\ell \leftarrow \ell + 1$.
3. Output $(\ell, \mathsf{input}, \mathsf{chain})$.

The BCS extractor. We now define the BCS extractor, which finds a transcript where the hash chain of roots ends at a chosen entry in the database. The algorithm $\mathsf{BCSExtract}$ is given below.

$\mathsf{BCSExtract}(D, \sigma, k, d_1, \ldots, d_k)$:

1. Let $(\ell, (x_1, \ldots, x_\ell), (y_0, \ldots, y_\ell)) \leftarrow \mathsf{HExtract}(D, \sigma, k)$. If $\mathsf{HExtract}(D, \sigma, k)$ outputs $\perp$ instead, output $\perp$.
2. If $y_0 \neq 0^\lambda$ or $x_{2i-1} \neq$ "i" for some $i \in \{1, \ldots, \lfloor (\ell+1)/2 \rfloor\}$, output the empty transcript $\mathsf{tr} = \emptyset$.
3. Let $\mathsf{rt}_i = x_{2i}$ for $i \in \{1, \ldots, \lfloor \ell/2 \rfloor\}$, let $m_i = y_{2i-1}$ for $i \in \{1, \ldots, \lfloor (\ell+1)/2 \rfloor\}$.
4. Let $T_i = \mathsf{Extract}(D, \mathsf{rt}_i, d_i)$ for each $i \in \{1, \ldots, \lfloor \ell/2 \rfloor\}$. If $T_i = \perp$ for some i, output $\perp$.
5. Output the transcript $\mathsf{tr} = \Big(\big(m_1, \ldots, m_{\lfloor (\ell+1)/2 \rfloor}\big), \big(\mathsf{leaves}(T_1), \ldots, \mathsf{leaves}(T_{\lfloor \ell/2 \rfloor})\big) \Big)$.

Lemma 8.7. *Let D be a database, and suppose that $D \in \bar{\mathcal{P}}_{\mathsf{col}}$. Then $\mathsf{BCSExtract}(D, \sigma, k, d_1, \ldots, d_k)$ always outputs a valid transcript tr. In particular, $\mathsf{tr} \neq \perp$.*

The following is an analogue of Lemma 6.5 for the BCS extractor.

Lemma 8.8. *Let D be a database and let $x \notin \mathrm{supp}(D)$. Let $D' = D + [x \mapsto y]$, and suppose that $D, D' \in \bar{\mathcal{P}}_{\mathsf{col}}$. Let $\sigma \in Z$. If $\mathsf{BCSExtract}(D, \sigma, k, d_1, \ldots, d_k) \neq \mathsf{BCSExtract}(D', \sigma, k, d_1, \ldots, d_k)$, then $y \in \{\sigma\} \cup S(D)$.*

Lemma 8.8 follows immediately from the following two claims.

Claim 8.9. *If $\mathsf{HExtract}(D, \sigma, k) \neq \mathsf{HExtract}(D', \sigma, k)$ for some $\sigma \in Z$, then $y \in S(D)$ or $y = \sigma$.*

Claim 8.10. *If $\mathsf{BCSExtract}(D, \sigma, k, d_1, \ldots, d_k) \neq \mathsf{BCSExtract}(D', \sigma, k, d_1, \ldots, d_k)$ and $\mathsf{HExtract}(D, \sigma, k) = \mathsf{HExtract}(D', \sigma, k)$, then $y \in S(D)$.*

Proof of Claim 8.9. Let $(\ell, (x_1, \ldots, x_\ell), (y_0, \ldots, y_\ell))$ be the output of $\mathsf{HExtract}(D', \sigma, k)$. Since $D, D' \in \bar{\mathcal{P}}_{\mathsf{col}}$, it follows that $\mathsf{HExtract}(D, \sigma, k)$ outputs $(\ell - i, (x_{i+1}, \ldots, x_\ell), (y_i, \ldots, y_\ell))$ for some $i \in [\ell]$. Therefore, the entry $((y_{i-1}, x_i), y_i)$ is in D' but not in D. Hence, $x = (y_{i-1}, x_i)$ and $y = y_i$. If $i = \ell$ then $y = y_\ell = \sigma$, and if $i < \ell$, then the entry $((y_i, x_{i+1}), y_{i+1})$ is in D, so $y = y_i \in S(D)$. □

Proof of Claim 8.10. If $\mathsf{HExtract}(D, \sigma, k) = \mathsf{HExtract}(D', \sigma, k)$, then there must exist some root rt_i in the execution of $\mathsf{BCSExtract}$ such that $\mathsf{Extract}(D, \mathsf{rt}_i, d_i) \neq \mathsf{Extract}(D', \mathsf{rt}_i, d_i)$. Hence, by Lemma 6.5 we get that $y \in S(D)$ or $y = \mathsf{rt}_i$. Assume that $y = \mathsf{rt}_i$. We must have that $(u, \mathsf{rt}_i) \in \mathrm{supp}(D)$ for some $u \in Z$, as $\mathsf{rt}_i = x_{2i}$ in the hash chain extracted by $\mathsf{HExtract}$, so $(y_{2i-1}, x_{2i}) \in \mathrm{supp}(D)$. Since $(u, \mathsf{rt}_i) \in \mathrm{supp}(D)$, it follows that $y \in S(D)$. □

8.5.1 Soundness

Let $\mathcal{R}$ be a relation, and let $\mathtt{x} \notin \mathcal{L}(\mathcal{R})$. Let $(\mathbf{P}, \mathbf{V})$ be an IOP for a relation $\mathcal{R}$ that has round-by-round error ϵ, proof length ℓ, query complexity q, and k rounds. Let $\ell_1, \ldots, \ell_k$ be the proof lengths of the individual rounds. Without loss of generality assume that for each i, $\ell_i = 2^{d_i}$ for some d_i. For ease of notation, we will omit the arguments $k, d_1, \ldots, d_k$ from the input to $\mathsf{BCSExtract}$.

We first show the following simple proposition.

Proposition 8.11. *The probability that a t-query classical adversary $\mathcal{A}$ causes the BCS verifier $\mathcal{V}$ to accept $\mathtt{x}$ is at most $\omega_{\mathsf{O}}(G^{\mathrm{BCS}}_{\mathbf{V},\mathtt{x}}, t + O(q \log \ell))$. The probability that a t-query quantum adversary $\mathcal{A}$ causes the BCS verifier $\mathcal{V}$ to accept $\mathtt{x}$ is at most $\omega^*_{\mathsf{O}}(G^{\mathrm{BCS}}_{\mathbf{V},\mathtt{x}}, t + O(q \log \ell))$.*

Proof. Let $\mathcal{A}$ be a (either classical or quantum) oracle algorithm. Let $\mathcal{B}$ be the algorithm that, when given access to an oracle h, runs $\mathcal{A}^h$ to obtain π and then outputs $\mathsf{BCSExpand}^h(\pi)$. The query complexity of $\mathcal{B}$ is $t + O(q \log \ell)$ because $\mathcal{A}$ makes t queries and $\mathsf{BCSExpand}$ makes $O(q \log \ell)$ queries. Let $((x_i)_{i=1}^a, (y_i)_{i=1}^a)$ be the output of $\mathcal{B}$. By definition of $\mathsf{BCSExpand}$, $h(x_i) = y_i$ for all $i \in [a]$. Hence $\mathcal{B}^h$ wins $G^{\mathrm{BCS}}_{\mathbf{V},\mathtt{x}}$ if and only if $\mathcal{V}^h(\mathtt{x}, \pi) = 1$. □

Let $\mathcal{P}_{\mathsf{BCS}} := \mathcal{P}_{G^{\mathrm{BCS}}_{\mathbf{V},\mathtt{x}}}$. Let $\mathcal{P}_{\mathsf{col}}$ be the collision property.

Define $\mathcal{P} := \cup_\sigma \mathcal{P}_\sigma$, where for $\sigma \in \{0,1\}^\lambda$, $\mathcal{P}_\sigma$ is the set of databases D where (1) D contains no collisions, (2) $\mathsf{tr} = \mathsf{BCSExtract}(D, \sigma)$ is a transcript where the verifier is about to move, (3) $\mathsf{state}(\mathtt{x}, \mathsf{tr}) = 0$, (4) $D(\sigma, \text{“i+1”}) \neq \perp$, where i is the number of rounds in tr, and (5) $\mathsf{state}(\mathtt{x}, \mathsf{tr} \| D(\sigma, \text{“i+1”})) = 1$.

Proposition 8.12. $\mathcal{P}_{\mathsf{BCS}} \subseteq \mathcal{P}_{\mathsf{col}} \cup \mathcal{P}$.

Proof. Suppose $D \in \mathcal{P}_{\mathsf{BCS}}$. If D has a collision, then $D \in \mathcal{P}_{\mathsf{col}}$. Suppose D has no collisions. Since $D \in \mathcal{P}_{\mathsf{BCS}}$, there exists $w = \Big(\big((\mathbf{x}_i)_{i\in[q]}, (\sigma_{j-1}, \text{“j”})_{j\in[k+1]}, (m_j, \mathsf{rt}_j)_{j\in[k]}\big), \big((\mathbf{y}_i)_{i\in[q]}, (m_j)_{j\in[k+1]}, (\sigma_j)_{j\in[k]}\big)\Big) \in G^{\mathrm{BCS}}_{\mathbf{V},\mathtt{x}}$ consistent with D. Let $\mathsf{tr} = \mathsf{BCSExtract}(D, m_{k+1})$. Since D has no collisions the transcript tr extracted from m_{k+1} by $\mathsf{BCSExtract}$ is a full transcript and consistent with w, i.e. $\mathsf{tr} = ((m_1, \ldots, m_{k+1}), (\Pi_1, \ldots, \Pi_k))$. Since $w \in G^{\mathrm{BCS}}_{\mathbf{V},\mathtt{x}}$ the IOP verifier accepts tr. Therefore, $\mathsf{state}(\mathtt{x}, \mathsf{tr}) = 1$. Since $\mathsf{state}(\mathtt{x}, \emptyset) = 0$ and adding a prover message cannot change state from 0 to 1, there exists i such that $\mathsf{tr}_1 = ((m_1, \ldots, m_i), (\Pi_1, \ldots, \Pi_i))$ and $\mathsf{tr}_2 = ((m_1, \ldots, m_{i+1}), (\Pi_1, \ldots, \Pi_i))$ satisfy $\mathsf{state}(\mathtt{x}, \mathsf{tr}_1) = 0$ and $\mathsf{state}(\mathtt{x}, \mathsf{tr}_2) = 1$. Note that by Item 2 of Definition 8.3 it must be that tr_1 is a transcript where the verifier is about to move. We have that $\mathsf{tr}_1 = \mathsf{BCSExtract}(\sigma_i, D)$, and $D(\sigma_i, \text{“i+1”}) = m_{i+1}$, so $\mathsf{tr}_2 = \mathsf{tr}_1 \| D(\sigma_i, \text{“i+1”})$. Hence, $D \in \mathcal{P}_{\sigma_i}$. □

Lemma 8.13. $\mathbf{I}(\mathcal{P} \mid \bar{\mathcal{P}}_{\mathsf{col}}, t) < \varepsilon + (2t+1)/2^\lambda$.

Proof. Fix a database $D \in \bar{\mathcal{P}}_{\mathsf{col}}$ with $|D| < t$, and fix an input $x \in \{0,1\}^{2\lambda}$. Suppose that $D \in \mathcal{P} \cap \bar{\mathcal{P}}_{\mathsf{col}}$. Then, $\Pr_y[D + [x \mapsto y] \in \bar{\mathcal{P}} \cap \bar{\mathcal{P}}_{\mathsf{col}}] < 2t/2^\lambda$. This is because since $D \in \mathcal{P}$, $D \in \mathcal{P}_\sigma$ for some σ. Hence, if $D + [x \mapsto y] \in \bar{\mathcal{P}}$ then $D + [x \mapsto y] \notin \mathcal{P}_\sigma$, and so $\mathsf{BCSExtract}(D, \sigma) \neq \mathsf{BCSExtract}(D + [x \mapsto y], \sigma)$. It follows by Lemma 8.8 that $y \in S(D) \cup \{\sigma\}$. Since $D \in \mathcal{P}_\sigma$, we see that $\sigma \in S(D)$. Thus, $y \in S(D)$. Since $|S| < 2t$, we see that $\Pr_y[D + [x \mapsto y] \in \bar{\mathcal{P}} \cap \bar{\mathcal{P}}_{\mathsf{col}}] < 2t/2^\lambda$.

Now, suppose that $D \notin \mathcal{P}$. We bound $\Pr_y[D + [x \mapsto y] \in \mathcal{P} \cap \bar{\mathcal{P}}_{\mathsf{col}}]$. Proposition 8.12 implies that, if the event inside the probability statement occurs, then the database $D' := D + [x \mapsto y]$ belongs to the set $\mathcal{P}_\sigma$ for some $\sigma \in \{0,1\}^\lambda$ and the database D does not belong to the set $\mathcal{P}_{\sigma'}$ for all σ'.

Let z denote the first λ bits of x, and let $\mathsf{tr} := \mathsf{BCSExtract}(D, z)$. If $\mathsf{state}(\mathtt{x}, \mathsf{tr}) = 1$ then set $S' := \emptyset$; otherwise set $S' := \{m : \mathsf{state}(\mathtt{x}, \mathsf{tr}\|m) = 1\}$, where we use the convention that $\mathsf{state}(\mathtt{x}, \mathsf{tr}\|m) = 0$ if tr is not a transcript where the verifier is about to move, i.e. that $\mathsf{tr}\|m$ is a malformed transcript. Note that this implies that $|S'| \leq \varepsilon 2^\lambda$, since the IOP has round-by-round soundness error ϵ. We have two cases.

- Case 1: $\mathsf{BCSExtract}(D, \sigma) \neq \mathsf{BCSExtract}(D', \sigma)$. By Lemma 8.8, we see that $y \in S(D)$ or $y = \sigma$. Suppose that $y = \sigma$. Since $D' \in \mathcal{P}_\sigma$, we get that $(\sigma, u') \in \mathrm{supp}(D')$ for some $u' \in \{0,1\}^\lambda$. Hence, either $\sigma \in S(D)$ or $\sigma = z$, and so either $y \in S(D)$ or $y = z$.
- Case 2: $\mathsf{BCSExtract}(D, \sigma) = \mathsf{BCSExtract}(D', \sigma)$. Let tr be equal to $\mathsf{BCSExtract}(D', \sigma)$. Since $D' \in \mathcal{P}_\sigma$, we have that tr is a transcript where the verifier is about to move and $\mathsf{state}(\mathtt{x}, \mathsf{tr}) = 0$. Let i be the number of rounds in tr. Since $D' \in \mathcal{P}_\sigma$, we also have that $D'(\sigma, \text{“i+1”}) \neq \perp$ and $\mathsf{state}(\mathtt{x}, \mathsf{tr}\|D'(\sigma, \text{“i+1”})) = 1$. Since $D \notin \mathcal{P}_\sigma$, we must have that $D(\sigma, \text{“i+1”}) = \perp$. Hence, $z = \sigma$, so it follows that $y \in S'$.

We deduce that in all cases, either $y \in S(D)$, $y = z$, or $y \in S'$. Note that these events are independent of σ, so in particular this holds for all choices of σ. Clearly, $|S(D)| \leq 2|D| < 2t$ and $|\{z\}| = 1$. As discussed above, $|S'| \leq \varepsilon 2^\lambda$. Hence $\Pr_y[D + [x \mapsto y] \in \mathcal{P}_{\mathsf{BCS}} \cap \bar{\mathcal{P}}_{\mathsf{col}}] \leq \Pr_y[y \in S(D) \cup \{z\} \cup S'] < \varepsilon + (2t+1)/2^\lambda$. It follows that $\mathbf{I}(\mathcal{P}_{\mathsf{BCS}} \mid \bar{\mathcal{P}}_{\mathsf{col}}, t) < \varepsilon + (2t+1)/2^\lambda$, as required. □

Finishing the proof of soundness. Since $\mathcal{P}_{\mathsf{BCS}} \subseteq \mathcal{P}_{\mathsf{col}} \cup \mathcal{P}$, we get that for any $\mathcal{A} \in \mathcal{C}_t^*$, $\Pr_{\mathcal{A}}[\mathcal{A} \text{ wins } G^{\mathrm{BCS}}_{\mathbf{V},\mathtt{x}}] \leq \Pr\big[D \in \mathcal{P}_{\mathsf{col}} \cup \mathcal{P} \;\big|\; \big((\mathbf{a}, \mathbf{b}, c), D\big) \leftarrow \mathsf{Sim}^*(\mathcal{A})\big]$. We also have that $\mathbf{I}(\mathcal{P}_{\mathsf{col}} \cup \mathcal{P} \mid \bar{\mathcal{P}}_{\mathsf{col}}, t) = \mathbf{I}(\mathcal{P} \mid \bar{\mathcal{P}}_{\mathsf{col}}, t)$ and $\emptyset \notin \mathcal{P} \cup \mathcal{P}_{\mathsf{col}}$ (as $\mathsf{state}(\mathtt{x}, \emptyset) = 0$). Therefore, by Lemmas 5.9 and 5.10 we have that $\Pr\big[D \in \mathcal{P}_{\mathsf{col}} \cup \mathcal{P} \;\big|\; \big((\mathbf{a}, \mathbf{b}, c), D\big) \leftarrow \mathsf{Sim}^*(\mathcal{A})\big] \leq t^2 \cdot 6\big(\mathbf{I}(\mathcal{P}|\bar{\mathcal{P}}_{\mathsf{col}}, t) + \mathbf{I}(\mathcal{P}_{\mathsf{col}}, t)\big) \leq 6(t^2\epsilon + 4t^3/2^\lambda)$. Thus, $\omega^*_{\mathsf{D}}(G^{\mathrm{BCS}}_{\mathbf{V},\mathtt{x}}, t) \leq 6(t^2\epsilon + 4t^3/2^\lambda)$.

By Lemma 4.9, we have that $\sqrt{\omega^*_{\mathsf{O}}(G^{\mathrm{BCS}}_{\mathbf{V},\mathtt{x}}, t)} \leq \sqrt{\omega^*_{\mathsf{D}}(G^{\mathrm{BCS}}_{\mathbf{V},\mathtt{x}}, t)} + O\left(\sqrt{q \log \ell / 2^\lambda}\right)$. We thus conclude that $\omega^*_{\mathsf{O}}(G^{\mathrm{BCS}}_{\mathbf{V},\mathtt{x}}, t) = O(t^2\varepsilon + t^3/2^\lambda + q \log \ell / 2^\lambda)$. Applying Proposition 8.11 completes the proof of soundness in Theorem 8.6.

8.5.2 Knowledge

Let $\mathcal{R}$ be a relation, and let $(\mathbf{P}, \mathbf{V})$ be an IOP for a relation $\mathcal{R}$ that has round-by-round knowledge error k, proof length ℓ, query complexity q, and k rounds. Let $\ell_1, \ldots, \ell_k$ be the proof lengths of the individual rounds. Without loss of generality assume that for each i, $\ell_i = 2^{d_i}$ for some d_i.

Let $\mathcal{A}$ be a t'-query quantum adversary $\mathcal{A}$ that causes the BCS verifier to accept an instance $\mathtt{x}$ with probability at least μ. Following the proof of Proposition 8.11, we obtain a t-query quantum adversary $\mathcal{B}$ that wins the BCS oracle game $G^{\mathrm{BCS}}_{\mathbf{V},\mathtt{x}}$ with probability at least μ, for $t := t' + O(q \log \ell)$. This transformation can be efficiently performed with black-box access to $\mathcal{A}$, because $\mathcal{B}$ merely extends $\mathcal{A}$ with some classical computation that depends on $\mathcal{A}$'s (classical) output.

We now describe the quantum extractor $\mathcal{E}$ for the BCS game, and then argue why it works.

$\mathcal{E}^{\mathcal{A}}(\mathtt{x}, 1^{t'}, 1^{\lambda})$:

1. Set $t := t' + O(q \log \ell)$.
2. Compute the quantum state $|\mathsf{Sim}^*(\mathcal{B})\rangle$, by simulating $\mathcal{B}$ via $\mathcal{A}$.
3. Measure the database register of $|\mathsf{Sim}^*(\mathcal{B})\rangle$ to obtain a database D.
4. Run $\mathtt{w} \leftarrow \mathbf{E}(\mathtt{x}, \emptyset)$. If $\mathtt{w}$ is a valid witness, return it, otherwise continue.
5. For each $\sigma \in \mathrm{im}(D)$:
   Run $\mathsf{tr} \leftarrow \mathsf{BCSExtract}(D, \sigma)$, then run $\mathtt{w} \leftarrow \mathbf{E}(\mathtt{x}, \mathsf{tr})$. If some Π in tr has an entry that is $\perp$, treat it as if it were 0.
   If $\mathtt{w}$ is a valid witness, return it, otherwise continue.

Let $\mathcal{P}_{\mathbf{E},\sigma}$ be the set of databases D where running $\mathsf{tr} \leftarrow \mathsf{BCSExtract}(D, \sigma)$ and then $\mathtt{w} \leftarrow \mathbf{E}(\mathtt{x}, \mathsf{tr})$ has $(\mathtt{x}, \mathtt{w}) \in \mathcal{R}$. Note that $\mathcal{E}^{\mathcal{A}}(\mathtt{x}, 1^{t'}, 1^{\lambda})$ outputs a valid witness if and only if $D \in \cup_{\sigma \in \{0,1\}^{\lambda}} \mathcal{P}_{\mathbf{E},\sigma}$. This is because if $\sigma \in \mathrm{im}(D)$ then $\mathcal{E}$ tries to extract from σ, and if $\sigma \notin \mathrm{im}(D)$ then $\mathsf{BCSExtract}(D, \sigma)$ is the empty transcript, and $\mathcal{E}$ always tries to extract from the empty transcript.

As in the case of Micali, we lower bound the probability that the extractor succeeds as follows:

$$\begin{aligned}
&\Pr\Big[\mathcal{E}^{\mathcal{A}}(\mathtt{x}, 1^{t'}, 1^{\lambda}) \text{ outputs a valid witness}\Big] = \Pr\Big[D \in \cup_{\sigma \in \{0,1\}^{\lambda}} \mathcal{P}_{\mathbf{E},\sigma}\Big] \\
&\geq \Pr[D \in \cup_{\sigma} \mathcal{P}_{\mathbf{E},\sigma} \cap \mathcal{P}_{\mathsf{BCS}}] = \Pr[D \in \mathcal{P}_{\mathsf{BCS}}] - \Pr\big[D \in \cap_{\sigma} \bar{\mathcal{P}}_{\mathbf{E},\sigma} \cap \mathcal{P}_{\mathsf{BCS}}\big] \\
&\geq \Pr[D \in \mathcal{P}_{\mathsf{BCS}}] - \Pr\Big[D \in \cap_{\sigma \in \{0,1\}^{\lambda}} \bar{\mathcal{P}}_{\mathbf{E},\sigma} \cap ((\cup_{\sigma} \mathcal{P}_{\sigma}) \cup \mathcal{P}_{\mathsf{col}})\Big] \\
&\geq \Pr[D \in \mathcal{P}_{\mathsf{BCS}}] - \Pr\big[D \in \mathcal{P}_{\mathsf{col}} \cup (\cup_{\sigma}(\mathcal{P}_{\sigma} \cap \bar{\mathcal{P}}_{\mathbf{E},\sigma}))\big] \\
&\geq \Pr[\mathcal{B} \text{ wins the BCS database game}] - t^2 \cdot 6\mathbf{I}(\mathcal{P}_{\mathsf{col}} \cup (\cup_{\sigma}(\mathcal{P}_{\sigma} \cap \bar{\mathcal{P}}_{\mathbf{E},\sigma})), t) \ ,
\end{aligned}$$

where the second inequality follows by Proposition 8.12 and the third by the fact that $\cap_{\sigma \in \{0,1\}^{\lambda}} \bar{\mathcal{P}}_{\mathbf{E},\sigma} \cap ((\cup_{\sigma} \mathcal{P}_{\sigma}) \cup \mathcal{P}_{\mathsf{col}}) \subseteq \mathcal{P}_{\mathsf{col}} \cup (\cup_{\sigma}(\mathcal{P}_{\sigma} \cap \bar{\mathcal{P}}_{\mathbf{E},\sigma}))$.

We have that the probability that $\mathcal{B}$ wins the quantum database game of $G^{\mathsf{Mic}}_{\mathbf{V},\mathtt{x}}$ is $\Omega(\mu - q \log \ell / 2^{\lambda})$ by Lemma 4.9. Hence, in order to complete the proof it suffices to show the following proposition.

Proposition 8.14. $\mathbf{I}(\mathcal{P}_{\mathsf{col}} \cup (\cup_{\sigma}(\mathcal{P}_{\sigma} \cap \bar{\mathcal{P}}_{\mathbf{E},\sigma})), t) < \mathsf{k} + (2t+1)/2^{\lambda}$.

Proof. We have that $\mathbf{I}(\mathcal{P}_{\mathsf{col}} \cup (\cup_{\sigma}(\mathcal{P}_{\sigma} \cap \bar{\mathcal{P}}_{\mathbf{E},\sigma})), t) \leq \mathbf{I}(\mathcal{P}_{\mathsf{col}}, t) + \mathbf{I}(\cup_{\sigma}(\mathcal{P}_{\sigma} \cap \bar{\mathcal{P}}_{\mathbf{E},\sigma}) \mid \bar{\mathcal{P}}_{\mathsf{col}}, t)$. By Lemma 6.11, we have that $\mathbf{I}(\mathcal{P}_{\mathsf{col}}, t) < t/2^{\lambda}$.

We now bound $\mathbf{I}(\cup_{\sigma}(\mathcal{P}_{\sigma} \cap \bar{\mathcal{P}}_{\mathbf{E},\sigma}) \mid \bar{\mathcal{P}}_{\mathsf{col}}, t)$. Let $D \in \bar{\mathcal{P}}_{\mathsf{col}}$ be a database, and let $x \notin \mathrm{supp}(D)$. Let z be the first λ bits of x. Let $\mathsf{tr} := \mathsf{BCSExtract}(D, z)$. If $\mathsf{state}(\mathtt{x}, \mathsf{tr}) = 1$ then set $S' := \emptyset$; otherwise let $S' := \{m : \mathsf{state}(\mathtt{x}, \mathsf{tr} \| m) = 1\}$, where we use the convention that $\mathsf{state}(\mathtt{x}, \mathsf{tr} \| m) = 0$ if tr is not a transcript where the verifier is about to move, i.e. that $\mathsf{tr} \| m$ is a malformed transcript.

Suppose first that $D \notin \cup_{\sigma}(\mathcal{P}_{\sigma} \cap \bar{\mathcal{P}}_{\mathbf{E},\sigma})$; we bound $\Pr_y[D + [x \mapsto y] \in \cup_{\sigma}(\mathcal{P}_{\sigma} \cap \bar{\mathcal{P}}_{\mathbf{E},\sigma}) \cap \bar{\mathcal{P}}_{\mathsf{col}}]$. Fix a y such that this holds, and let $D' := D + [x \mapsto y]$. Then there exists σ^* such that $D' \in \mathcal{P}_{\sigma^*} \cap \bar{\mathcal{P}}_{\mathbf{E},\sigma^*}$ and $D \in \bar{\mathcal{P}}_{\sigma^*} \cup \mathcal{P}_{\mathbf{E},\sigma^*}$. There are two cases.

- Case 1: $D \in \mathcal{P}_{\mathbf{E},\sigma^*}$. Since $D' \notin \mathcal{P}_{\mathbf{E},\sigma^*}$, we get that $\mathsf{BCSExtract}(D, \sigma^*) \neq \mathsf{BCSExtract}(D', \sigma^*)$. Hence, by Lemma 8.8 we get that $y \in S(D)$ or $y = \sigma^*$. Since $D' \in \mathcal{P}_{\sigma^*}$, we have that $(\sigma^*, u') \in \mathrm{supp}(D')$ for some $u' \in \{0,1\}^{\lambda}$, and hence either $\sigma^* \in S(D)$ or $\sigma^* = z$. Thus, either $y \in S(D)$ or $y = z$.

- Case 2: $D \in \bar{\mathcal{P}}_{\sigma^*} \cap \bar{\mathcal{P}}_{\mathbf{E},\sigma^*}$. We have that $D' \in \mathcal{P}_{\sigma^*}$. If $\mathsf{BCSExtract}(D, \sigma^*) \neq \mathsf{BCSExtract}(D', \sigma^*)$, then again we get that $y \in S(D)$ or $y = z$. If $\mathsf{BCSExtract}(D, \sigma^*) = \mathsf{BCSExtract}(D', \sigma^*)$, then by Case 2 of Lemma 8.13 we get that $z = \sigma^*$ and $y \in S'$. In particular, this implies that $D \in \bar{\mathcal{P}}_{\mathbf{E},z}$.

We conclude that either $y \in S(D) \cup \{z\}$, or it holds that $D \in \bar{\mathcal{P}}_{\mathbf{E},z}$ and $y \in S'$. We know that $|S(D) \cup \{z\}| < 2t+1$. We show that $|S'| \leq \mathsf{k} \cdot 2^\lambda$ if $D \in \bar{\mathcal{P}}_{\mathbf{E},z}$. If $\mathsf{state}(\mathbb{x}, \mathsf{tr}) = 1$ then $|S'| = 0 < \mathsf{k} \cdot 2^\lambda$, so suppose otherwise. Suppose that $|S'|/2^\lambda = \Pr_m[\mathsf{state}(\mathbb{x}, \mathsf{tr} \| m) = 1] > \mathsf{k}$. Then, since $\mathsf{state}(\mathbb{x}, \mathsf{tr}) = 0$, by the definition of round-by-round knowledge we get that $\mathbf{E}(\mathbb{x}, \mathsf{tr})$ outputs a valid witness for $\mathbb{x}$. Therefore, $D \in \mathcal{P}_{\mathbf{E},z}$, a contradiction, and so $|S'| \leq \mathsf{k} \cdot 2^\lambda$.

We thus conclude that if $D \in \mathcal{P}_{\mathbf{E},z}$ then $\Pr_y[D' \in \bar{\mathcal{P}}_{\mathsf{col}} \cap (\cup_\sigma(\mathcal{P}_\sigma \cap \bar{\mathcal{P}}_{\mathbf{E},\sigma}))] < (2t+1)/2^\lambda$, and if $D \in \bar{\mathcal{P}}_{\mathbf{E},z}$ then $\Pr_y[D' \in \bar{\mathcal{P}}_{\mathsf{col}} \cap (\cup_\sigma(\mathcal{P}_\sigma \cap \bar{\mathcal{P}}_{\mathbf{E},\sigma}))] < \mathsf{k} + (2t+1)/2^\lambda$. Hence, $\mathrm{flip}(\cup_\sigma(\mathcal{P}_\sigma \cap \bar{\mathcal{P}}_{\mathbf{E},\sigma}) | \bar{\mathcal{P}}_{\mathsf{col}}, t) < \mathsf{k} + (2t+1)/2^\lambda$.

Now suppose that $D \in \cup_\sigma(\mathcal{P}_\sigma \cap \bar{\mathcal{P}}_{\mathbf{E},\sigma})$. We bound $\Pr_y[D + [x \mapsto y] \in \overline{\cup_\sigma(\mathcal{P}_\sigma \cap \bar{\mathcal{P}}_{\mathbf{E},\sigma})} \cap \bar{\mathcal{P}}_{\mathsf{col}}]$. Fix a y such that this holds, and let $D' := D + [x \mapsto y]$. Then there exists σ^* such that $D \in \mathcal{P}_{\sigma^*} \cap \bar{\mathcal{P}}_{\mathbf{E},\sigma^*}$ and $D' \in \bar{\mathcal{P}}_{\sigma^*} \cup \mathcal{P}_{\mathbf{E},\sigma^*}$. There are two cases.

- Case 1: $D' \in \bar{\mathcal{P}}_{\sigma^*}$. Since $D \in \mathcal{P}_{\sigma^*} \cup \bar{\mathcal{P}}_{\mathsf{col}}$ and $D' \in \bar{\mathcal{P}}_{\mathsf{col}}$, it follows that $\mathsf{BCSExtract}(D, \sigma^*) \neq \mathsf{BCSExtract}(D', \sigma^*)$. Hence, by Lemma 8.8 we get that $y \in S(D) \cup \{\sigma^*\}$. Since $D \in \mathcal{P}_{\sigma^*}$, we see that $\sigma \in S(D)$, and so $y \in S(D)$.

- Case 2: $D' \in \mathcal{P}_{\sigma^*} \cap \mathcal{P}_{\mathbf{E},\sigma^*}$. Since $D \in \mathcal{P}_{\sigma^*}$, we have that $\sigma^* \in \mathrm{im}(D)$, and hence also in $\mathrm{im}(D')$. Since σ^* is in both $\mathrm{im}(D)$ and $\mathrm{im}(D')$ and that $D \in \bar{\mathcal{P}}_{\mathbf{E},\sigma^*}$ and $D' \in \mathcal{P}_{\mathbf{E},\sigma^*}$, it follows that $\mathsf{BCSExtract}(D, \sigma^*) \neq \mathsf{BCSExtract}(D', \sigma^*)$. Hence, by Lemma 8.8 we get that $y \in S(D)$ or $y = \sigma^*$. Since $D \in \mathcal{P}_{\sigma^*}$, it follows that $(\sigma^*, u') \in \mathrm{supp}(D)$ for some $u' \in \{0,1\}^\lambda$, so $\sigma^* \in S(D)$. Hence, $y \in S(D)$.

We conclude that $\mathrm{flip}(\overline{\cup_\sigma(\mathcal{P}_\sigma \cap \bar{\mathcal{P}}_{\mathbf{E},\sigma})} \mid \bar{\mathcal{P}}_{\mathsf{col}}, t) < 2t/2^\lambda$.

Putting it together, we get that $\mathbf{I}(\cup_\sigma(\mathcal{P}_\sigma \cap \bar{\mathcal{P}}_{\mathbf{E},\sigma}) \mid \bar{\mathcal{P}}_{\mathsf{col}}, t) < \mathsf{k} + (2t+1)/2^\lambda$. □

8.6 On the difference in hash chains

In the original BCS construction, the messages are generated "outside" of the hash chain. In the modified construction in this paper, we consider the messages as *part* of the hash chain. This is used by our proof technique, which forces the prover to "fix" the message in a given round before moving to the next, which allows us to argue from round-by-round soundness as defined in [CCHLRR18].

We now describe how to prove soundness for the hash chain in [BCS16], provided that the underlying IOP satisfies a slightly stronger (and arguably natural) notion of round-by-round soundness. The modifications to Definition 8.4 are as follows (we make analogous modifications to Definition 8.5). We allow the verifier messages in a partial transcript to be $\perp$ (i.e., undefined), and we additionally require that if tr is a transcript with $m_i = \perp$ for some i and $\mathsf{state}(\mathbb{x}, \mathsf{tr}) = 0$, then with probability at most ϵ it holds that $\mathsf{state}(\mathbb{x}, \mathsf{tr}') = 1$, where tr' is the transcript obtained by "filling in" m_i with a uniformly random message. We note that like round-by-round soundness, this stronger definition is also satisfied by many natural protocols, such as the sumcheck protocol. Indeed, it seems that any natural protocol that is round-by-round sound should also be sound under this stronger definition. There exists a (contrived) two-round protocol which has a quadratic gap between its round-by-round soundness and "strong" round-by-round soundness, and it is open to determine the correct relationship for protocols that have many rounds.

Given this stronger soundness guarantee, the proof then proceeds as follows. The algorithm $\mathsf{BCSExtract}$ is modified to fit the new hash chain, and also so that it extracts the verifier messages (which are now not in the hash chain), including the verifier message for the next round. We then

define $\mathcal{P}_\sigma$ to be the set of databases where $D \in \bar{\mathcal{P}}_{\mathsf{col}}$ and $\mathsf{state}(\mathtt{x}, \mathsf{BCSExtract}(D, \sigma)) = 1$. The rest of the proof is then nearly identical to the proof in Section 8.5.

A Proof of Lemma 3.2

Recall from [Zha19] that the compressed phase oracle $\mathcal{O}$ is defined as $\mathsf{Dec} \circ \mathcal{O}' \circ \mathsf{Dec}$ where $\mathcal{O}'$ is the unitary that acts as $|x,u,z,D\rangle \mapsto (-1)^{u \cdot D(x)} |x,u,z,D\rangle$ (where $u \cdot \perp := 0$), Dec is the unitary that acts as $|x,u,z,D\rangle \mapsto |x,u,z\rangle \mathsf{Dec}_x |D\rangle$, and Dec_x is defined as follows:

$$\mathsf{Dec}_x |D\rangle := \begin{cases} \frac{1}{\sqrt{2^n}} \sum_y |D + [x \mapsto y]\rangle & \text{if } D(x) = \perp \text{ and } |D| < t \\ |D\rangle & \text{if } D(x) = \perp \text{ and } |D| = t \\ |D\rangle + \frac{1}{\sqrt{2^n}} |D'\rangle - \frac{1}{\sqrt{2^n}} |\phi_{0^n}\rangle & \text{if } D(x) \neq \perp \end{cases},$$

where $D' := D - x$, and $|\phi_w\rangle := \frac{1}{\sqrt{2^n}} \sum_y (-1)^{y \cdot w} |D' + [x \mapsto y]\rangle$, so that $|D\rangle = \frac{1}{\sqrt{2^n}} \sum_w (-1)^{w \cdot D(x)} |\phi_w\rangle$. Note that $\mathsf{Dec}_x |\phi_w\rangle = |\phi_w\rangle + \big(\frac{1}{2^n} \sum_y (-1)^{y \cdot w}\big)(|D'\rangle - |\phi_{0^n}\rangle)$, which equals $|\phi_w\rangle$ for $w \neq 0^n$ and $|D'\rangle$ for $w = 0^n$.

We now prove each of the cases in the statement of Lemma 3.2. If $|D| = t$, then Dec_x does nothing, so $\mathcal{O} |x,u,z,D\rangle = \mathcal{O}' |x,u,z,D\rangle = (-1)^{u \cdot D(x)} |x,u,z,D\rangle$. If $|D| < t$ and $u = 0^n$, then $\mathcal{O}'$ does nothing. Since $\mathsf{Dec}_x \circ \mathsf{Dec}_x$ is the identity, we get that $\mathcal{O} |x,0^n,z,D\rangle = |x,0^n,z,D\rangle$.

Now, assume that $|D| < t$ and $u \neq 0^n$. We first consider the case where $D(x) = \perp$. We apply Dec, then $\mathcal{O}'$, and then Dec again to $|x,u,z,D\rangle$:

$$\begin{aligned} &\xrightarrow{\mathsf{Dec}} |x,u,z\rangle \otimes \frac{1}{\sqrt{2^n}} \sum_y |D + [x \mapsto y]\rangle \\ &\xrightarrow{\mathcal{O}'} |x,u,z\rangle \otimes \frac{1}{\sqrt{2^n}} \sum_y (-1)^{u \cdot y} |D + [x \mapsto y]\rangle \\ &\xrightarrow{\mathsf{Dec}} |x,u,z\rangle \otimes \frac{1}{\sqrt{2^n}} \sum_y (-1)^{u \cdot y} |D + [x \mapsto y]\rangle \quad , \end{aligned}$$

as the second application of Dec does nothing since $u \neq 0^n$.

Finally, suppose $D(x) = y^*$. As before, we apply Dec, then $\mathcal{O}'$, and then Dec again to $|x,u,z,D\rangle$:

$$\begin{aligned} &\xrightarrow{\mathsf{Dec}} |x,u,z\rangle \otimes \big(|D\rangle + \frac{1}{\sqrt{2^n}} \big|D'\big\rangle - \frac{1}{\sqrt{2^n}} |\phi_{0^n}\rangle \big) \\ &= |x,u,z\rangle \otimes \big(|D\rangle + \frac{1}{\sqrt{2^n}} \big|D'\big\rangle - \frac{1}{2^n} \sum_y \big|D' + [x \mapsto y]\big\rangle \big) \\ &\xrightarrow{\mathcal{O}'} |x,u,z\rangle \otimes \big((-1)^{u \cdot y^*} |D\rangle + \frac{1}{\sqrt{2^n}} \big|D'\big\rangle - \frac{1}{2^n} \sum_y (-1)^{u \cdot y} \big|D' + [x \mapsto y]\big\rangle \big) \\ &= |x,u,z\rangle \otimes \big((-1)^{u \cdot y^*} |D\rangle + \frac{1}{\sqrt{2^n}} \big|D'\big\rangle - \frac{1}{\sqrt{2^n}} |\phi_u\rangle \big) \\ &\xrightarrow{\mathsf{Dec}} |x,u,z\rangle \otimes \Big((-1)^{u \cdot y^*} \big(|D\rangle + \frac{1}{\sqrt{2^n}} \big|D'\big\rangle - \frac{1}{\sqrt{2^n}} |\phi_{0^n}\rangle \big) + \frac{1}{\sqrt{2^n}} |\phi_{0^n}\rangle - \frac{1}{\sqrt{2^n}} |\phi_u\rangle \Big) \\ &= |x,u,z\rangle \otimes \Big((-1)^{u \cdot y^*} |D\rangle + \frac{(-1)^{u \cdot y^*}}{\sqrt{2^n}} \big|D'\big\rangle + \frac{(1 - (-1)^{u \cdot y^*})}{\sqrt{2^n}} |\phi_{0^n}\rangle - \frac{1}{\sqrt{2^n}} |\phi_u\rangle \Big) \\ &= |x,u,z\rangle \otimes \Big((-1)^{u \cdot y^*} |D\rangle + \frac{(-1)^{u \cdot y^*}}{\sqrt{2^n}} \big|D'\big\rangle + \frac{1}{2^n} \sum_y (1 - (-1)^{u \cdot y^*} - (-1)^{u \cdot y}) \big|D' + [x \mapsto y]\big\rangle \Big) \quad , \end{aligned}$$

which completes the proof.

References

[AHIV17] Scott Ames, Carmit Hazay, Yuval Ishai, and Muthuramakrishnan Venkitasubramaniam. "Ligero: Lightweight Sublinear Arguments Without a Trusted Setup". In: *Proceedings of the 24th ACM Conference on Computer and Communications Security*. CCS '17. 2017, pp. 2087–2104.

[ALMSS98] Sanjeev Arora, Carsten Lund, Rajeev Motwani, Madhu Sudan, and Mario Szegedy. "Proof verification and the hardness of approximation problems". In: *Journal of the ACM* 45.3 (1998). Preliminary version in FOCS '92., pp. 501–555.

[ARU14] Andris Ambainis, Ansis Rosmanis, and Dominique Unruh. "Quantum Attacks on Classical Proof Systems: The Hardness of Quantum Rewinding". In: *Proceedings of the 55th Annual IEEE Symposium on Foundations of Computer Science*. FOCS '14. 2014, pp. 474–483.

[AS04] Scott Aaronson and Yaoyun Shi. "Quantum lower bounds for the collision and the element distinctness problems". In: *Journal of the ACM* 51.4 (2004), pp. 595–605.

[AS98] Sanjeev Arora and Shmuel Safra. "Probabilistic checking of proofs: a new characterization of NP". In: *Journal of the ACM* 45.1 (1998). Preliminary version in FOCS '92., pp. 70–122.

[BBCPGL18] Carsten Baum, Jonathan Bootle, Andrea Cerulli, Rafaël del Pino, Jens Groth, and Vadim Lyubashevsky. "Sub-linear Lattice-Based Zero-Knowledge Arguments for Arithmetic Circuits". In: *Proceedings of the 38th Annual International Cryptology Conference*. CRYPTO '18. 2018, pp. 669–699.

[BBHR19] Eli Ben-Sasson, Iddo Bentov, Yinon Horesh, and Michael Riabzev. "Scalable Zero Knowledge with No Trusted Setup". In: *Proceedings of the 39th Annual International Cryptology Conference*. CRYPTO '19. 2019, pp. 733–764.

[BCIOP13] Nir Bitansky, Alessandro Chiesa, Yuval Ishai, Rafail Ostrovsky, and Omer Paneth. "Succinct Non-Interactive Arguments via Linear Interactive Proofs". In: *Proceedings of the 10th Theory of Cryptography Conference*. TCC '13. 2013, pp. 315–333.

[BCRSVW19] Eli Ben-Sasson, Alessandro Chiesa, Michael Riabzev, Nicholas Spooner, Madars Virza, and Nicholas P. Ward. "Aurora: Transparent Succinct Arguments for R1CS". In: *Proceedings of the 38th Annual International Conference on the Theory and Applications of Cryptographic Techniques*. EUROCRYPT '19. Full version available at `https://eprint.iacr.org/2018/828`. 2019, pp. 103–128.

[BCS16] Eli Ben-Sasson, Alessandro Chiesa, and Nicholas Spooner. "Interactive Oracle Proofs". In: *Proceedings of the 14th Theory of Cryptography Conference*. TCC '16-B. 2016, pp. 31–60.

[BDFLSZ11] Dan Boneh, Özgür Dagdelen, Marc Fischlin, Anja Lehmann, Christian Schaffner, and Mark Zhandry. "Random Oracles in a Quantum World". In: *Proceedings of the 17th International Conference on the Theory and Application of Cryptology and Information Security*. ASIACRYPT '11. 2011, pp. 41–69.

[BFLS91] László Babai, Lance Fortnow, Leonid A. Levin, and Mario Szegedy. "Checking computations in polylogarithmic time". In: *Proceedings of the 23rd Annual ACM Symposium on Theory of Computing*. STOC '91. 1991, pp. 21–32.

[BHT98] Gilles Brassard, Peter Høyer, and Alain Tapp. "Quantum Cryptanalysis of Hash and Claw-Free Functions". In: *Proceedings of the 3rd Latin American Symposium on Theoretical Informatics*. LATIN '98. 1998, pp. 163–169.

[BISW17] Dan Boneh, Yuval Ishai, Amit Sahai, and David J. Wu. "Lattice-Based SNARGs and Their Application to More Efficient Obfuscation". In: *Proceedings of the 36th Annual International Conference on Theory and Applications of Cryptographic Techniques*. EUROCRYPT '17. 2017, pp. 247–277.

[BISW18] Dan Boneh, Yuval Ishai, Amit Sahai, and David J. Wu. "Quasi-Optimal SNARGs via Linear Multi-Prover Interactive Proofs". In: *Proceedings of the 37th Annual International Conference on Theory and Application of Cryptographic Techniques*. EUROCRYPT '18. 2018, pp. 222–255.

[BN19] Carsten Baum and Ariel Nof. *Concretely-Efficient Zero-Knowledge Arguments for Arithmetic Circuits and Their Application to Lattice-Based Cryptography*. Cryptology ePrint Archive, Report 2019/532. 2019.

[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*. CCS '93. 1993, pp. 62–73.

[Bab85] László Babai. "Trading group theory for randomness". In: *Proceedings of the 17th Annual ACM Symposium on Theory of Computing*. STOC '85. 1985, pp. 421–429.

[CCHLRR18] Ran Canetti, Yilei Chen, Justin Holmgren, Alex Lombardi, Guy N. Rothblum, and Ron D. Rothblum. *Fiat–Shamir From Simpler Assumptions*. Cryptology ePrint Archive, Report 2018/1004. 2018.

[CDGORRSZ17] Melissa Chase, David Derler, Steven Goldfeder, Claudio Orlandi, Sebastian Ramacher, Christian Rechberger, Daniel Slamanig, and Greg Zaverucha. "Post-Quantum Zero-Knowledge and Signatures from Symmetric-Key Primitives". In: *Proceedings of the 24th ACM Conference on Computer and Communications Security*. CCS '17. 2017, pp. 1825–1842.

[Co17] O(1) Labs. *Coda Cryptocurrency*. `https://codaprotocol.com/`. 2017.

[DFG13] Özgür Dagdelen, Marc Fischlin, and Tommaso Gagliardoni. "The Fiat-Shamir Transformation in a Quantum World". In: *Proceedings of the 19th International Conference on the Theory and Application of Cryptology and Information Security*. ASIACRYPT '13. 2013, pp. 62–81.

[DFKNS92] Cynthia Dwork, Uriel Feige, Joe Kilian, Moni Naor, and Shmuel Safra. "Low Communication 2-Prover Zero-Knowledge Proofs for NP". In: *Proceedings of the 11th Annual International Cryptology Conference*. CRYPTO '92. 1992, pp. 215–227.

[DFMS19] Jelle Don, Serge Fehr, Christian Majenz, and Christian Schaffner. "Security of the Fiat–Shamir Transformation in the Quantum Random-Oracle Model". In: *Proceedings of the 39th Annual International Cryptology Conference*. CRYPTO '19. 2019, pp. 356–383.

[DJ92] David Deutsch and Richard Jozsa. "Rapid solution of problems by quantum computation". In: *Proceedings of the Royal Society of London. Series A: Mathematical and Physical Sciences* 439.1907 (1992), pp. 553–558.

[Eat17] Edward Eaton. "Leighton-Micali Hash-Based Signatures in the Quantum Random-Oracle Model". In: *Proceedings of the 24th International Conference on Selected Areas in Cryptography*. SAC '17. 2017, pp. 263–280.

[FGLSS96] Uriel Feige, Shafi Goldwasser, Laszlo Lovász, Shmuel Safra, and Mario Szegedy. "Interactive proofs and the hardness of approximating cliques". In: *Journal of the ACM* 43.2 (1996). Preliminary version in FOCS '91., pp. 268–292.

[FS86] Amos Fiat and Adi Shamir. "How to prove yourself: practical solutions to identification and signature problems". In: *Proceedings of the 6th Annual International Cryptology Conference*. CRYPTO '86. 1986, pp. 186–194.

[GH98] Oded Goldreich and Johan Håstad. "On the complexity of interactive proofs with bounded communication". In: *Information Processing Letters* 67.4 (1998), pp. 205–214.

[GKR15] Shafi Goldwasser, Yael Tauman Kalai, and Guy N. Rothblum. "Delegating Computation: Interactive Proofs for Muggles". In: *Journal of the ACM* 62.4 (2015), 27:1–27:64.

[GMNO18] Rosario Gennaro, Michele Minelli, Anca Nitulescu, and Michele Orrù. "Lattice-Based zk-SNARKs from Square Span Programs". In: *Proceedings of the 25th ACM Conference on Computer and Communications Security*. CCS '18. 2018, pp. 556–573.

[GMR89] Shafi Goldwasser, Silvio Micali, and Charles Rackoff. "The knowledge complexity of interactive proof systems". In: *SIAM Journal on Computing* 18.1 (1989). Preliminary version appeared in STOC '85., pp. 186–208.

[GVW02] Oded Goldreich, Salil Vadhan, and Avi Wigderson. "On interactive proofs with a laconic prover". In: *Computational Complexity* 11.1/2 (2002), pp. 1–53.

[GW11] Craig Gentry and Daniel Wichs. "Separating Succinct Non-Interactive Arguments From All Falsifiable Assumptions". In: *Proceedings of the 43rd Annual ACM Symposium on Theory of Computing*. STOC '11. 2011, pp. 99–108.

[Gro96] Lov K. Grover. "A Fast Quantum Mechanical Algorithm for Database Search". In: *Proceedings of the 28th Annual ACM Symposium on Theory of Computing*. STOC '96. 1996, pp. 212–219.

[IMSX15] Yuval Ishai, Mohammad Mahmoody, Amit Sahai, and David Xiao. *On Zero-Knowledge PCPs: Limitations, Simplifications, and Applications*. Available at `http://www.cs.virginia.edu/~mohammad/files/papers/ZKPCPs-Full.pdf`. 2015.

[KKW18] Jonathan Katz, Vladimir Kolesnikov, and Xiao Wang. "Improved Non-Interactive Zero Knowledge with Applications to Post-Quantum Signatures". In: *Proceedings of the 25th ACM Conference on Computer and Communications Security*. CCS '18. 2018, pp. 525–537.

[KLS18] Eike Kiltz, Vadim Lyubashevsky, and Christian Schaffner. "A Concrete Treatment of Fiat-Shamir Signatures in the Quantum Random-Oracle Model". In: *Proceedings of the 37th Annual International Conference on Theory and Application of Cryptographic Techniques*. EUROCRYPT '17. 2018, pp. 552–586.

[KPT97] Joe Kilian, Erez Petrank, and Gábor Tardos. "Probabilistically checkable proofs with zero knowledge". In: *Proceedings of the 29th Annual ACM Symposium on Theory of Computing*. STOC '97. 1997, pp. 496–505.

[KR08] Yael Kalai and Ran Raz. "Interactive PCP". In: *Proceedings of the 35th International Colloquium on Automata, Languages and Programming*. ICALP '08. 2008, pp. 536–547.

[Kil92] Joe Kilian. "A note on efficient zero-knowledge proofs and arguments". In: *Proceedings of the 24th Annual ACM Symposium on Theory of Computing*. STOC '92. 1992, pp. 723–732.

[LFKN92] Carsten Lund, Lance Fortnow, Howard J. Karloff, and Noam Nisan. "Algebraic Methods for Interactive Proof Systems". In: *Journal of the ACM* 39.4 (1992), pp. 859–868.

[LZ19] Qipeng Liu and Mark Zhandry. "Revisiting Post-Quantum Fiat–Shamir". In: *Proceedings of the 39th Annual International Cryptology Conference*. CRYPTO '19. 2019, pp. 326–355.

[Mer89] Ralph C. Merkle. "A certified digital signature". In: *Proceedings of the 9th Annual International Cryptology Conference*. CRYPTO '89. 1989, pp. 218–238.

[Mic00] Silvio Micali. "Computationally Sound Proofs". In: *SIAM Journal on Computing* 30.4 (2000). Preliminary version appeared in FOCS '94., pp. 1253–1298.

[NIS16] NIST. *Post-Quantum Cryptography*. 2016. URL: `https://csrc.nist.gov/Projects/Post-Quantum-Cryptography`.

[PS96] David Pointcheval and Jacques Stern. "Security Proofs for Signature Schemes". In: *Proceedings of the 14th Annual International Conference on Theory and Application of Cryptographic Techniques*. EUROCRYPT '96. 1996, pp. 387–398.

[Pas03] Rafael Pass. "On Deniability in the Common Reference String and Random Oracle Model". In: *Proceedings of the 23rd Annual International Cryptology Conference*. CRYPTO '03. 2003, pp. 316–337.

[RRR16] Omer Reingold, Ron Rothblum, and Guy Rothblum. "Constant-Round Interactive Proofs for Delegating Computation". In: *Proceedings of the 48th ACM Symposium on the Theory of Computing*. STOC '16. 2016, pp. 49–62.

[SCI14] SCIPR Lab. *libsnark: a C++ library for zkSNARK proofs*. 2014. URL: `https://github.com/scipr-lab/libsnark`.

[SCI18] SCIPR Lab. *DIZK: Java library for distributed zero knowledge proof systems*. 2018. URL: `https://github.com/scipr-lab/dizk`.

[SCI19] SCIPR Lab. *libiop: C++ library for IOP-based zkSNARKs*. 2019. URL: `https://github.com/scipr-lab/libiop`.

[TU16] Ehsan Ebrahimi Targhi and Dominique Unruh. "Post-Quantum Security of the Fujisaki–Okamoto and OAEP Transforms". In: *Proceedings of the 14th Theory of Cryptography Conference*. TCC '16-B. 2016, pp. 192–216.

[Unr15] Dominique Unruh. "Non-Interactive Zero-Knowledge Proofs in the Quantum Random Oracle Model". In: *Proceedings of the 34th Annual International Conference on Theory and Application of Cryptographic Techniques*. EUROCRYPT '15. 2015, pp. 755–784.

[Unr17] Dominique Unruh. "Post-quantum Security of Fiat-Shamir". In: *Proceedings of the 23rd International Conference on the Theory and Applications of Cryptology and Information Security*. ASIACRYPT '17. 2017, pp. 65–95.

[Val08] Paul Valiant. "Incrementally Verifiable Computation or Proofs of Knowledge Imply Time/Space Efficiency". In: *Proceedings of the 5th Theory of Cryptography Conference*. TCC '08. 2008, pp. 1–18.

[WB15] Michael Walfish and Andrew J. Blumberg. "Verifying Computations Without Reexecuting Them". In: *Communications of the ACM* 58.2 (Jan. 2015), pp. 74–84.

[Wat09] John Watrous. "Zero-Knowledge against Quantum Attacks". In: *SIAM Journal on Computing* 39.1 (2009). Preliminary version appeared in STOC '06., pp. 25–58.

[ZKP17] ZKP Standards. *Zero Knowledge Proof Standardization*. `https://zkproof.org/`. 2017.

[Zc14] Electric Coin Company. *Zcash Cryptocurrency*. `https://z.cash/`. 2014.

[Zha12] Mark Zhandry. "Secure Identity-Based Encryption in the Quantum Random Oracle Model". In: *Proceedings of the 32nd Annual International Cryptology Conference*. CRYPTO '12. 2012, pp. 758–775.

[Zha15] Mark Zhandry. "A note on the quantum collision and set equality problems". In: *Quantum Information & Computation* 15.7&8 (2015), pp. 557–567.

[Zha19] Mark Zhandry. "How to Record Quantum Queries, and Applications to Quantum Indifferentiability". In: *Proceedings of the 39th Annual International Cryptology Conference*. CRYPTO '19. 2019, pp. 239–268.

[bell15] Sean Bowe. *bellman: a zk-SNARK library*. 2015. URL: `https://github.com/zkcrypto/bellman`.

[dalek18] dalek cryptography. *A pure-Rust implementation of Bulletproofs using Ristretto*. 2018. URL: `https://github.com/dalek-cryptography/bulletproofs`.

[iden19] iden3. *websnark: A fast zkSNARK proof generator written in native Web Assembly*. 2019. URL: `https://github.com/iden3/websnark`.

[stark18] libstark. *libstark: a C++ library for zkSTARK systems*. 2018. URL: `https://github.com/elibensasson/libSTARK`.