How to Verify that a Small Device is Quantum, Unconditionally

Giulio Malavolta[*] and Tamer Mour[†]

Bocconi University

Abstract

A proof of quantumness (PoQ) allows a classical verifier to efficiently test if a quantum machine is performing a computation that is infeasible for any classical machine. In this work, we propose a new approach for constructing PoQ protocols where soundness holds *unconditionally* assuming a bound on the *memory* of the prover, but otherwise no restrictions on its runtime. In this model, we propose two protocols:

- A simple protocol with a quadratic gap between the memory required by the honest parties and the memory bound of the adversary. The soundness of this protocol relies on Raz's (classical) memory lower bound for matrix inversion (Raz, FOCS 2016).
- A protocol that achieves an exponential gap, building on techniques from the literature on the bounded storage model (Dodis et al., Eurocrypt 2023).

Both protocols are also efficiently verifiable. Despite having worse asymptotics, our first protocol is conceptually simple and relies only on arithmetic modulo 2, which can be implemented with one-qubit Hadamard and CNOT gates, plus a *single* one-qubit non-Clifford gate.

[*] giulio.malavolta@unibocconi.it. Work supported by the European Research Council through an ERC Starting Grant (Grant agreement No. 101077455, ObfusQation) and partially funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy - EXC 2092 CASA – 390781972.

[†] tamer.mour@unibocconi.it. Work supported by European Research Council (ERC) under the EU's Horizon 2020 research and innovation programme (Grant agreement No. 101019547).

Contents

1 Introduction

Quantum computing promises to bring the computing sciences to a new world, where the laws of quantum mechanics are harnessed to do computation. The first step towards this ambitious goal is to obtain an experimental demonstration of *quantum advantage*: The construction of a quantum apparatus capable of performing computations that are manifestly infeasible for any classical machine. Existing approaches combine techniques from cryptography, complexity theory, and physics, in order to design a *proof of quantumness* (PoQ) that is (i) efficient enough to be run on existing quantum machines and (ii) *efficiently verifiable* by a fully classical computer. Unfortunately, despite a massive industrial and academic effort devoted to achieving this milestone, a convincing experimental demonstration remains elusive.

To set some context, let us discuss existing approaches for the experimental demonstration of quantum advantage. On the one hand, the *circuit sampling* approach asks the quantum machine to (approximately) sample from a distribution defined by a quantum circuit, that is conjectured to be hard to simulate classically. The advantage of this approach is that it is feasible for near-term quantum machines, and is in fact the approach adopted by Google's first experiment [AAB+19], and more recently in [AABA+24]. On the downside, the classical hardness of sampling from these distributions has been called into question [LLL+21]. Moreover, by far the biggest problem of this approach is that a classical machine cannot efficiently verify that the computation was done correctly.

To overcome this drawback, researchers have turned their attention to *cryptography*. The simplest proposal that allows efficient verification would be to run Shor's algorithm [Sho94] to factor a large number. Then, given its prime decomposition, it is easy to check that the computation was done correctly. More sophisticated and general approaches exist, where the classical hardness is proven assuming the intractability of computational problems related to lattices [BCM+18], group actions [AMR22], or discrete logarithms [KMCVY22]. More recently, proposals based on compiling two-player non-local games [KLVY23a] have emerged as an alternative pathway to PoQ. Common to all of these approaches is that the (classical) verification is always efficient, making them an appealing alternative to the circuit sampling approach. On the flip-side, their soundness also relies on unproven computational conjectures and, furthermore, the quantum operations are somewhat more complex than the ones involved in the sampling method, making this approach less suitable for experiments. To the best of our knowledge, none of the existing experiments [LZG+24] was performed in a regime where the problem is classically hard.

To summarize, all known PoQ protocols rely on computational assumptions, and are either inefficiently verifiable or beyond the current technological reach. We emphasize that the former limitation is for a good reason: It was recently shown that PoQ protocols *imply* the existence of (quantum) computationally hard problems [MSY24]. In this work, we propose a new approach for testing quantumness. Instead of placing assumptions on the computational power of the prover, we constrain its *memory*.

1.1 Our Contributions

We consider a model where the adversary has a limited amount of memory, but is otherwise computationally unbounded. This is commonly referred to as the *bounded storage model* [Mau92, CM97, Din01, Vad04, DHRS04, GZ19, DQW23]. We show that in this model, one can construct protocols that are (i) efficiently verifiable and (ii) unconditionally sound.

Our first result is a simple PoQ protocol with a quadratic gap between the space complexity of the honest execution and the memory bound on the malicious prover.[1] The protocol is extremely simple; the honest quantum prover performs simple arithmetics modulo 2, and can be implemented with one-qubit Hadamard and CNOT gates and a *single* one-qubit non-Clifford gate. The prover's circuit is illustrated in Figure 1. The protocol is inspired by techniques from the cryptographic literature [Mah18b, BCM$^+$21, BGK$^+$23, NZ23, BK24, BKM$^+$24], and its soundness relies on Raz's (classical) memory lower bound for learning parities [Raz18]. We summarize our result with the following statement.

Theorem 1 (Simple PoQ with Quadratic Gap, Informal)**.** *There exists a proof of quantumness protocol with constant gap between completeness and soundness, where the honest execution runs $O(n^2)$ time, the verifier has $O(n)$ memory, and the honest prover has $n+2$ qubits. Soundness holds unconditionally against any classical attacker that uses less than $n^2/20$ bits of memory.*

We stress that, although the verifier's runtime and the prover's quantum circuit size are quadratic, an honest execution of the protocol has linear parallel runtime. In particular, the quantum circuit implementing the prover has depth $O(n)$.

Despite its simplicity, an unsatisfactory aspect of the above protocol is that it has only a quadratic gap between the memory of the honest and the malicious prover. Leveraging techniques from the literature of protocols in the bounded storage model [DQW23], we show that this gap can be made arbitrarily large. The following theorem establishes the theoretical feasibility of a PoQ with an exponential gap.

Theorem 2 (PoQ with Exponential Gap, Informal)**.** *There exists a proof of quantumness protocol with constant gap between completeness and soundness, where the honest execution execution runs* $\mathrm{poly}(n)$ *time, the verifier has* $\mathrm{polylog}(n)$ *memory, and the honest prover has* $\mathrm{polylog}(n)$ *qubits. Soundness holds unconditionally against any classical attacker that uses less than n bits of memory.*

As our final contribution, we show that the above protocol achieves a form of soundness also against *quantum provers*. In more details, it allows the generation of a state of the form

$$\frac{|x_0\rangle + |x_1\rangle}{\sqrt{2}}$$

such that even the prover that holds that state in memory cannot guess both x_0 and x_1. In the literature, this is known as a *claw state generation* protocol [BK24].

[1] For the moment, it suffices to think of the space complexity of a quantum prover as the logarithm of the dimension of his Hilbert space. We make this more precise as the discussion progresses.

Theorem 3 (Claw Generation with Quantum Soundness, Informal)**.** *There exists a claw generation protocol, where the honest execution runs* $\mathrm{poly}(n)$ *time, the verifier has* $\mathrm{polylog}(n)$ *memory, and the honest prover has* $\mathrm{polylog}(n)$ *qubits. Soundness holds unconditionally against any quantum attacker that uses less than* n *qubits.*

Claw generation protocols have proven to be extremely valuable in quantum cryptography. For instance, all known protocols to classically verify BQP computation rely on claw states, one way or another [Mah18b, BCM$^+$21, BGK$^+$23, NZ23, BK24, BKM$^+$24]. To demonstrate the usefulness of our claw-generation protocol, we sketch how to derive analogous PoQ protocols with unconditional security in the bounded storage model, by adapting analysis from prior work [BK24] (Appendix A).

1.2 Technical Overview

Our starting point is a recent result of Raz [Raz18], who considers the following experiment between a verifier and a prover:

1. The verifier samples a uniform $s \leftarrow \mathbb{F}_2^n$.
2. For $i = 1 \dots m$, the verifier samples a uniform $v_i \leftarrow \mathbb{F}_2^n$ and sends $(v_i, v_i^\intercal s)$ to the prover.
3. The prover returns some $\tilde{s} \in \mathbb{F}_2^n$.

The main theorem of [Raz18] shows that, if a prover uses less than $n^2/20$ memory bits, then the probability that he outputs s is $2^{-\Omega(n)}$, for any $m = \mathrm{poly}(n)$. This implies a strong lower bound on the memory needed to solve linear equation systems, for any classical computer. One could speculate that quantum algorithms are better suited for solving this kind of tasks, thus immediately yielding a PoQ protocol based on the memory-hardness of solving linear equations. Alas, we are not aware of any quantum algorithm solving this problem with $O(n)$ qubits, bringing us back to square one.

Nevertheless, inspired by a recent work from Guan and Zhandry [GZ19], we observe that the above problem has sufficient *structure* to allow us to leverage techniques developed in the context of computationally security. Turning our attention to cryptographic PoQs, we see that all recent works [BCM$^+$21, KCVY22, KLVY23b, BGK$^+$23] follow the same three-phase template:

- (Claw Generation) In the first phase, the prover and verifier engage in an interaction, at the end of which the internal state of the (honest) prover is of the form
$$\frac{|x_0\rangle + |x_1\rangle}{\sqrt{2}},$$
where x_0 and x_1 are two bitstrings that are known to the verifier. Furthermore, it is required that the prover cannot output both x_0 and x_1 simultaneously. Thus, this phase relies on cryptographic hardness and is typically realized using a *trapdoor claw-free function* (TCF) [Mah18a]. We refer to the above state as a *claw state*, and to this interaction as a *claw generation* subroutine.

- (Commitment) In the second phase, the prover is instructed to convert the claw state into a single qubit $b \in \{|0\rangle, |1\rangle, |+\rangle, |-\rangle\}$. The security guarantee from the claw state generation phase translates to the fact that the prover cannot tell whether b is in the Z-basis or in the X-basis, as this would imply learning the values x_0 and x_1.
- (Non-Local Game) The protocol then completes with a *CHSH test*: A test of quantumness that is derived by the CHSH non-local game, where a quantum prover is challenged to produce a correlation with the "hidden basis" that a classical prover cannot.

Note that after the first phase, all guarantees are information-theoretic. In other words, cryptography is used only to derive the hardness of recovering the values in the claw state. Thus, our task to design a PoQ in the bounded storage model reduces to the task of designing a claw state generation protocol with security against memory-bounded provers.

A Simple PoQ from Learning Parities. The above experiment considered by Raz already suggests a construction for a claw state generation protocol. Consider the linear map $x \mapsto Ax$, where $A = (V, Vs) \in \mathbb{F}^{m \times (n+1)}$ is induced by samples in the experiment, namely composed by rows of the form $(v_i, v_i^\intercal s)$. We make two simple observations:

1. A is two-to-one with overwhelming probability, since V is full rank with overwhelming probability.
2. Finding a *claw*, namely x_0, x_1 such that $Ax_0 = Ax_1$, implies finding s since $x_0 + x_1 = (s, -1)$ is the only non-trivial element in $\ker(A)$. Computing s given samples of the form $(v_i, v_i^\intercal s) \in \mathbb{F}^{n+1}$ is precisely the problem of learning parities considered in the experiment, and cannot be done with less than $n^2/20$ memory.

Hence, $x \mapsto Ax$ is a *claw-free function* in the bounded storage model, namely a 2-to-1 function where it is hard to find a claw. Claw generation from such a function is straightforward: the verifier sends the function to the prover, the prover computes the function over the uniform superposition then measures an outcome y, reducing the state to a superposition of the two pre-images of y.

Naively performing the above, however, requires the parties to communicate A and, in particular, to remember it using quadratic space. Instead, we perform the evaluation in rounds: at rounds $i = 1, \ldots, m$ of the interaction, the verifier samples a row from A, i.e. a vector $a_i = (v_i, v_i^\intercal s) \in \mathbb{F}_2^{n+1}$, and sends it to the prover. The prover starts with a uniform superposition over $n+1$ qubits in a register X and, at round i, applies the isometry

$$|x\rangle_{\mathsf{X}} \mapsto |x\rangle_{\mathsf{X}} |a_i^\intercal x\rangle_{\mathsf{Y}},$$

measures Y and returns the outcome $y_i \in \mathbb{F}_2$ to the verifier. Overall, such an interaction corresponds to the prover computing the mapping $|x\rangle_{\mathsf{X}} \mapsto |x\rangle_{\mathsf{X}} |Ax\rangle_{\mathsf{Y}}$, then measuring the m qubits in Y to obtain an outcome $y = (y_1, \ldots, y_m)$ and a residual claw state $(|x_0\rangle + |x_1\rangle)/\sqrt{2}$, where x_0, x_1 are the two pre-images of y under A.

While the above protocol seems to give us the claw generation sufficient for a PoQ, we have omitted an important detail: How can the verifier compute x_0 and x_1, which are generally necessary to carry out the CHSH test? Once again, computing x_0 and x_1 given y requires the verifier to invert the matrix A. Since A has quadratic size, it is infeasible for the verifier in our model to remember it, let alone invert it. Taking a closer look at the literature, we observe that existing instantiations of this outline, e.g., [KCVY22], do not require the verifier to extract x_0 and x_1 entirely but rather only the bits $b_0 = r^\intercal x_0$ and $b_1 = r^\intercal x_1$ for a uniformly random $r \leftarrow \mathbb{F}_2^{n+1}$ *sampled by the verifier* in the commitment phase, independently of x_0 and x_1. In fact, it is required that the verifier can tell if b_0 and b_1 are equal – in which case $r^\intercal(s, -1) = r^\intercal(x_0 + x_1) = 0$ – and, only when they are, to know their value.

Given this observation, we propose a memory-efficient implementation of the verifier. At the beginning of the protocol, the verifier flips a coin to decide whether r satisfies $r^\intercal(s, -1) = 0$ or $r^\intercal(s, -1) = 1$. In the former case, the verifier is not required to learn b_0 and b_1 and, therefore, the CHSH test can be executed with a uniformly random $r \notin \ker((s, -1))$, independently of x_0, x_1. In the latter case, the verifier must know $r^\intercal x_0$. We use the fact that the prover communicates y, which is a linear function of x_0, to the verifier. Instead of sampling $r \leftarrow \ker((s, -1))$ directly, the verifier samples r as a linear combination of the rows of A: when he sends a_i at round i, he decides "on-the-fly" whether to include it in the linear combination or not, with probably 1/2 each. Remembering his choices, the verifier completes the claw generation with a linear combination $u \in \mathbb{F}_2^m$ satisfying $u^\intercal A = r$. In particular, it holds that $r^\intercal x_0 = u^\intercal A x_0 = u^\intercal y$. This the verifier can compute given u and y in linear space, allowing him to execute the CHSH test also in this case.

A PoQ with an Exponential Gap. Our first PoQ, while extremely simple, achieves only a quadratic gap between the space complexity of the honest parties and the memory needed for a successful attacker. A natural question is whether we can achieve a better gap.

We positively answer this question in our second result. We devise an interactive claw generation protocol that is secure against attackers with memory m while requiring only $\operatorname{polylog} m$ memory for an honest execution. The protocol is based on *interactive hashing* [NOVY92, DHRS04]. In prior work [CCM98, Din01, DQW23], (classical) interactive hashing was used to construct oblivious transfer (OT) protocols in the BSM. Most relevant to us is the work by Dodis et al. [DQW23] where, roughly speaking, interactive hashing is used to let a sender and a receiver jointly choose two bit values from a long stream $u_1, \ldots, u_k \in \{0, 1\}$, such that the verifier knows both values but a bounded-memory receiver can remember only one of them by the end of the protocol. We observe that realizing this outline with a *quantum* receiver, using a coherent implementation of interactive hashing [MY23], allows the receiver to obtain a superposition of the two chosen bits while still preventing him from recovering both simultaneously.

Let us recall the outline from [DQW23] in more detail. A sender sends a stream of $k \gg m$ uniform bits $u_1, \ldots, u_k \leftarrow \{0, 1\}$, one at a time, to the receiver. The sender remembers the bits at two random locations $v_0^*, v_1^* \leftarrow [k]$ of his choice. After the streaming is complete, the two parties perform interactive hashing. The transcript of the interactive hashing protocol defines a 2-to-1 hash function h over the domain $[k]$ and a hash value y, and guarantees the following: (i) On the one hand, the receiver can control one pre-image of y under h and, in fact, he can make it so $y = h(v)$ for an apriori arbitrarily chosen input v. (ii) On the other hand, the receiver cannot control *both* pre-images of y under h. Namely, for any bounded-size set of inputs B, apriori chosen by the receiver, it holds that $h^{-1}(y) \subseteq B$ with a very small probability. Consequently, the interactive hashing protocol defines a pair of indices $v_0, v_1 \in [k]$, and the two chosen bits are set to be u_{v_0} and u_{v_1}. By the soundness of the interactive hashing ((ii)), the receiver cannot obtain both u_{v_0} and u_{v_1} by the end of the protocol. This is shown in [DQW23] as follows: Define B to be the set of bits in the stream about which the receiver remembers sufficient information. Via standard incompressibility argument, due to the bounded memory of the receiver, B cannot be too large. Hence, the receiver cannot have too much information about both u_{v_0} and u_{v_1}.

We carry the above outline where the verifier plays the role of the sender and the quantum prover plays the role of the classical receiver, coherently. That is, the prover prepares a uniform superposition over $[k]$, next to an ancilla qubit. In the streaming phase, upon the receipt of the v^{th} bit u_v, the prover maps the basis vector $|v\rangle|0\rangle$ to $|v\rangle|u_v\rangle$, entangling the two registers. Consequently, the streaming completes with the prover's internal state being

$$\frac{1}{\sqrt{k}} \sum_{v \in [k]} |v\rangle_{\mathsf{V}} |u_v\rangle_{\mathsf{U}} .$$

Next, the verifier and the prover, with input register V, perform interactive hashing to select two values $v_0, v_1 \in [k]$. The residual state of the prover after this is complete is

$$\frac{|v_0, u_{v_0}\rangle + |v_1, u_{v_1}\rangle}{\sqrt{2}},$$

where v_0, v_1 are the two pre-images of y under h, for y and h defined by the transcript of the interactive hashing (recall in a classical invocation, it holds by (i) that $h(v) = y$ for the prover's classical input v). The verifier then checks that $h(v_0^*) = h(v_1^*) = y$. If not, the protocol is repeated with a new random stream of bits. Otherwise, the parties have performed successful claw generation: the prover has a superposition of two values that the verifier knows, yet of which he can remember at most one. By extending the analysis from [DQW23] to the setting of a memory-bounded *quantum* adversary, we additionally show that such claw generation guarantees security also against quantum adversaries. The analysis turns out to be quite technical. In particular, we rely on a lemma from [BBK22] that can be seen as an analog of incompressibility arguments for quantum information. For more details we refer the reader to Section 4.5.

While the above claw generation already provides non-trivial hardness for computing the values in the claw, such hardness is limited to that of predicting a *single bit* (note we do not claim that the prover cannot obtain the indices v_0, v_1). Naturally, such a "1-bit claw generation" cannot provide too much security. We propose a simple security amplification strategy via *stitching*. In stitching, we convert many "1-bit claws" into a single claw that is as hard to predict as predicting all of the 1-bit claws that compose it, simultaneously.

Let us describe how to stitch two claws together. Stitching more claws generalizes straight-forwardly. Let the prover's state be $(|v_0^1, u_0^1\rangle + |v_1^1, u_1^1\rangle) \otimes (|v_0^2, u_0^2\rangle + |v_1^2, u_1^2\rangle)$ after the generation of two claw states. The verifier, who has the values $v_0^1, v_1^1, v_0^2, v_1^2$, sends to the prover two predicates $g^1, g^2 : [k] \to \{0,1\}$ such that $g^i(v^i_b) = b$. Stitching is complete with the prover applying the following isometry over the registers containing the v values

$$\left|v^1\right\rangle_{\mathsf{V}^1} \left|v^2\right\rangle_{\mathsf{V}^2} \mapsto \left|v^1\right\rangle_{\mathsf{V}^1} \left|v^2\right\rangle_{\mathsf{V}^2} \left|g^1(v^1) \oplus g^2(v^2)\right\rangle_{\mathsf{B}},$$

then measuring the qubit in B to obtain a bit $b \in \{0,1\}$, which he sends to the verifier. This operation entangles the two claws, resulting in the following stitched claw state

$$\frac{|v^1_0, u^1_0\rangle \, |v^2_b, u^2_b\rangle + |v^1_1, u^1_1\rangle \left|v^2_{1-b}, u^2_{1-b}\right\rangle}{\sqrt{2}},$$

which the verifier can anticipate given b.

To conclude, our claw generation protocol performs the above 1-bit claw generation sequentially λ times, where λ is a security parameter, then performs stitching to stitch together the λ 1-bit claws into a claw that a memory-bounded attacker can break with only negligible probability.

2 Preliminaries

We denote by $[n]$ the set $\{1, \dots, n\}$. We recall the following fundamental fact about the rank of random binary matrices (see, e.g. [BKW97]).

Proposition 4 (Rank of a Random Matrix)**.** *Let $M \leftarrow \mathbb{F}_2^{m\times n}$ be a uniformly sampled random matrix, with $m = 2n$. Then,*

$$\Pr\left(\mathrm{rank}(M) = n\right) = 1 - O(2^{-n}).$$

2.1 Memory Bounded Algorithms

We say that an adversary is *memory bounded* if the size of its state at any point in the computation is bounded by some parameter m. In fact, for most of our bounds (for instance the one in Section 4) we can also consider a slightly stronger adversary that is allowed to have an unlimited amount of short-term memory, but can only store an m-bit state after the end of each round.

Lemma 5 ([Raz18])**.** *For any $C < 1/20$ there exists $\alpha > 0$, such that the following holds. For $m \leq 2^{\alpha n}$ and an algorithm $\mathcal{A}$, consider the experiment:*

- *Sample a uniform $s \in \mathbb{F}_2^n$.*
- *For $i = 1 \dots m$: Sample a uniform $v_i \in \mathbb{F}_2^n$ and send $(v_i, v_i \cdot s)$ to $\mathcal{A}$.*
- *$\mathcal{A}$ returns some $\tilde{s}$.*

If $\mathcal{A}$ uses less than Cn^2 memory bits, then:

$$\Pr\left(s = \tilde{s}\right) \leq O(2^{-\alpha n}).$$

2.2 The Goldreich-Levin Extractor

We recall the well-known Goldreich-Levin extractor [GL89], highlighting its memory complexity.

Lemma 6 ([GL89])**.** *Let $f : \mathbb{F}_2^n \to \mathbb{F}_2$ be a function such that, for some $x \in \mathbb{F}_2^n$,*

$$\Pr_{r \in \mathbb{F}_2^n} \left(f(r) = r^\intercal x \right) \geq 1/2 + \varepsilon(n).$$

for inverse-polynomial $\varepsilon(n)$. Then there exists an extractor with memory $O(n \log n)$ running in time polynomial in n and making oracle calls to f, that returns x with probability inverse-polynomial in n.

Proof Sketch. We briefly recall the description of the extractor. For $t = O(\log n/\varepsilon^2(n)) = O(\log n)$ the extractor proceeds as follows:

1. Sample uniformly random $(r_1, \ldots, r_t) \leftarrow \mathbb{F}_2^n$ and $(b_1, \ldots, b_t) \leftarrow \mathbb{F}_2$.
2. For $i = 1 \ldots n$,

   2.1. For any $S \subseteq [t]$, compute

   $$x_{i,S} = f\left(\sum_{j \in S} r_j\right) + \sum_{j \in S} b_j.$$

   2.2. Set $x_i = \text{maj}\left\{x_{i,S}\right\}_S$.
3. Return $(x_1, \ldots, x_n)$.

It is clear that the runtime of the algorithm is polynomial in n and it is shown in [GL89] that the above extractor succeeds with at least inverse-polynomial probability in n. As for the memory complexity, the algorithm must store $(r_1, \ldots, r_t)$ and $(b_1, \ldots, b_t)$ and, for each bit, compute the majority function which is computable in constant memory due to Barrington's theorem [Bar89]. Overall, the memory required by the extractor is bounded by $O(n \log n)$. □

2.3 Information Theory

The *Shannon entropy* (or simply entropy) of a random variable X is defined as $H(X) = \mathbb{E}_{x \leftarrow X}(-\log \Pr_X(X = x))$. The conditional entropy of X given another random variable Y is defined as $H(X \mid Y) = \mathbb{E}_{y \leftarrow Y}(H(X \mid Y = y))$.

The *min-entropy* of a random variable X is defined as $H_\infty(X) = -\log \max_x \Pr(X = x)$. We define the *conditional min-entropy* of X given another random variable Y as:

$$H_\infty(X \mid Y) = -\log \mathbb{E}_{y \leftarrow Y}(\max_x \Pr(X = x \mid Y = y)) = -\log \mathbb{E}_{y \leftarrow Y}(2^{-H_\infty(X|Y=y)}).$$

Note that $H_\infty(X \mid Y) \leq H(X \mid Y)$ for any X, Y (in particular this holds for the non-conditional notions).

We recall the following basic facts on min-entropy from the literature.

Lemma 7 ([DORS06])**.** *For any random variables X, Y, Z, where Y is over $\{0,1\}^m$, we have $H_\infty(X \mid Y, Z) \geq H_\infty(X \mid Z) - m$.*

Lemma 8 ([DORS06])**.** *For any random variables X, Y and any $\varepsilon > 0$, it holds that*

$$\Pr_{y \leftarrow Y}(H_\infty(X \mid Y = y) \geq H_\infty(X \mid Y) - \log(1/\varepsilon)) \geq 1 - \varepsilon.$$

Proposition 9 ([DQW23])**.** *For random variables X, Y, Z where X and Y are independent conditioned on Z, it holds that $H_\infty(X \mid Y) \geq H_\infty(X \mid Y, Z) \geq H_\infty(X \mid Z)$ and $H_\infty(X, Y \mid Z) \geq H_\infty(X|Z) + H_\infty(Y|Z)$.*

Proposition 10 ([DQW23])**.** *For any random variables $X = X_1, \ldots, X_k$ and Y, it holds that*

$$H_\infty(X \mid Y) \leq \sum_{i \in [k]} H(X_i \mid Y).$$

Proof. The inequality follows easily from the chain-rule of Shannon entropy

$$H_\infty(X \mid Y) \leq H(X \mid Y) = \sum_{i \in [k]} H(X_i \mid Y, X_1, \ldots, X_{i-1}) \leq \sum_{i \in [k]} H(X_i \mid Y).$$

□

In the following proposition, we give a lower bound on the min-entropy of a binary random variable with high Shannon entropy.

Proposition 11 (Binary Entropy Bound)**.** *For any binary random variable X, letting $H(X) = h$, it holds that*

$$H_\infty(X) \geq 1 - \log\Big(1 + \sqrt{1 - h^{\ln 4}}\Big).$$

Proof. Let $p = \max_{b \in \{0,1\}} \Pr(X = b) = 2^{-H_\infty(X)} > 1/2$. We start from the following known upper bound on the binary entropy function [Top01]: $H(X) \leq (4p(1-p))^{1/\ln 4}$. The bound implies $4p - 4p^2 - h^{\ln 4} \geq 0$ and, consequently, $p \leq \frac{1}{2}(1 + \sqrt{1 - h^{\ln 4}})$. □

We recall some useful concentration bounds.

Lemma 12 (Markov Inequality)**.** *Let X be a non-negative random variable and let $\alpha > 0$, then:*

$$\Pr\left(X \geq \alpha \cdot \mathbb{E}(X)\right) \leq 1/\alpha.$$

Lemma 13 (Chernoff Inequality)**.** *Let $X_1 \ldots X_n$ be independent random variables such that $X_i \in \{0,1\}$ and let $\tilde{X} = \mathbb{E}(\sum_i X_i)$. Then:*

$$\Pr\left(\sum_i X_i \geq (1 + \delta) \cdot \tilde{X}\right) \leq e^{-\frac{\tilde{X}\delta^2}{3}}.$$

We say that a set of random variables $X_1 \dots X_n$ is *negatively correlated* if for every subset $S \subseteq [n]$ it holds that:

$$\Pr\left(\prod_{i \in S} X_i = 1\right) \le \prod_{i \in S} \Pr\left(X_i = 1\right).$$

In [PS97], it is shown that the Chernoff bound (Lemma 13) continues to hold also for negatively correlated random variables.

2.4 Quantum Information

In quantum mechanics, physical systems are identified with Hilbert spaces $\mathcal{H}$, and the states of the system are identified with positive semidefinite operators (PSD) ρ with unit trace, called *density operators*. A state is called *pure* if the density operator has rank one, and otherwise it is called *mixed*. Any unit vector $|\psi\rangle \in \mathcal{H}$ determines a pure state by the formula $\rho = |\psi\rangle\langle\psi|$, and conversely any pure state can be written in this way. We often associate a Hilbert space with a register X, and we denote a state in such register as $|\psi\rangle_{\mathsf{X}}$.

A *measurement* with a finite outcome set $\mathcal{O}$ is described by a collection of bounded operators $\{M_x\}_{x \in \mathcal{O}}$ acting on $\mathcal{H}$ such that $\sum_{x \in \mathcal{O}} M_x = Id$, referred to as a POVM. A *quantum circuit* is a unitary operator that operates on $\mathcal{H}$ and is given by the composition of unitary gates (taken from some fixed universal gate set). The size of a quantum circuit is the number of gates used in that circuit. The qubits are typically split into input qubits and ancillas, which are assumed to be initialized in the $|0\rangle$. In its most general form, any physically-admissible quantum operation is described by a *completely-positive trace-preserving* (CPTP) map from linear operators $\mathrm{L}(\mathsf{X})$ on a register X to linear operators $\mathrm{L}(\mathsf{Y})$ on a register Y. The *trace distance* between two states ρ and σ is defined as:

$$\mathbf{TD}(\rho, \sigma) = \frac{1}{2}\|\rho - \sigma\|_1 = \frac{1}{2}\mathsf{Tr}\left(\sqrt{(\rho - \sigma)^\dagger(\rho - \sigma)}\right)$$

where Tr is the trace of a matrix. The operational meaning of the trace distance is that $1/2 \cdot (1 + \mathbf{TD}(\rho, \sigma))$ is the maximal probability that two states ρ and σ can be distinguished by any (possibly unbounded) quantum channel or algorithm.

Lastly, we recall a useful lemma from [BBK22].

Lemma 14 (Plug-In Lemma [BBK22])**.** *Let $X = X_1, \dots, X_k$ and W be arbitrarily dependent random variables, where W is over m qubits. Then, it holds that:*

$$\mathbf{TD}\left((i, X_{<i}, X_i, W), (i, X_{<i}, X_i', W)\right) \le \sqrt{m/2k},$$

where i is uniformly random over $[k]$ and X_i' is sampled according to the marginal distribution of X_i given $X_{<i}$ (and is otherwise independent in W).

3 Simple Proof of Quantumness with Quadratic Gap

The first proof of quantumness we present in this work is based on the space lower bound for learning with parities by Raz [Raz18] (Lemma 5) and, consequently, achieves a quadratic gap between the space complexity of an honest execution and the space complexity of the best attack.

Theorem 15 (PoQ with Quadratic Gap)**.** *There exists a protocol between a classical verifier and a quantum prover where the two parties take as input a security parameter $n \in \mathbb{N}$ and, at the end of the interaction, the verifier either accepts or rejects, and*

- *(Completeness) The verifier accepts with probability $\cos^2(\pi/8) - O(2^{-n})$ when interacting with an honest prover.*
- *(Soundness) Any non-uniform classical prover $\mathcal{P}^*$ that on input n uses less than $n^2/20$ memory bits has success probability at most $3/4 + O(2^{-n})$ in making the verifier accept.*
- *(Complexity) The verifier and (honest) prover run in time $O(n^2)$ and in space $O(n)$. In particular, the prover uses $n+2$ qubits.*

3.1 The Protocol

Our protocol follows the general framework of basing PoQ protocol on *claw generation* [BCM+21, KCVY22, BGK+23]. We perform claw generation using on a simple linear function $x \mapsto Ax$, where $A = (V, Vs) \in \mathbb{F}^{m\times(n+1)}$ for a random $V \leftarrow \mathbb{F}^{m\times n}$ and $s \leftarrow \mathbb{F}^n$. The function is 2-to-1 with overwhelming probability and it is claw-free for any adversary who has space at most quadratic due to the space complexity of learning parities by [Raz18] (Lemma 5).

While PoQ based on claw generation generally requires the verifier to extract the values in the claw, for the above function this demands inverting A (and, in particular, remembering it) and therefore considered infeasible in our bounded-storage model. We show how to nevertheless implement the verifier using only linear space. Due to the simplicity of our claw-free function, the prover in our protocol is not only space-efficient, but can be implemented by a very simple quantum circuit, which we illustrate in Fig. 1.

Protocol 1 (Parity-based PoQ)**.** Let n be the security paramter of the protocol and let $m = 2n$. The protocol consists of an interaction between a quantum prover and a classical verifier, described as follows.

- (Claw Generation)

  1. The verifier samples a uniform $s \leftarrow \mathbb{F}_2^n$ and a uniform $u = (u_1, \dots, u_m) \leftarrow \mathbb{F}_2^m$ at the beginning of the protocol. The verifier sets $t = (s, -1)$ and $r = 0^{n+1}$.

     The prover prepares the uniform superposition:

     $$|\psi\rangle = \frac{1}{\sqrt{2^{n+1}}} \sum_{x \in \mathbb{F}_2^{n+1}} |x\rangle \in \mathbb{C}^{2^{n+1}}.$$

  2. For $i = 1 \dots m$,

     2.1. The verifier samples a uniform $v_i \leftarrow \mathbb{F}_2^n$ and sends to the prover $(v_i, v_i^\intercal s) = a_i$. The verifier updates $r = r + u_i \cdot a_i$.

2.2. The prover applies the following isometric mapping to his state

$$|x\rangle_{\mathsf{X}} \mapsto |x\rangle_{\mathsf{X}} |a_i^{\intercal} x\rangle_{\mathsf{Y}}$$

and measures the qubit in Y in the computational basis to obtain a bit $y_i \in \mathbb{F}_2$ and sends it to the verifier.

- (Commitment)

1. The verifier samples a random bit $c \leftarrow \{0,1\}$, then:
   - If $c = 0$, sends r to the prover.
   - If $c = 1$, samples a fresh uniform $r \leftarrow \mathbb{F}_2^{n+1}$ conditioned on $r^{\intercal} t = 1$ and sends it to the prover.
2. The prover applies the following isometric mapping to his state

$$|x\rangle_{\mathsf{X}} \mapsto |x\rangle_{\mathsf{X}} |r^{\intercal} x\rangle_{\mathsf{B}} ,$$

then measures X in Hadamard basis to obtain $d \in \mathbb{F}_2^{n+1}$ and sends it to the verifier.

- (CHSH Test)

1. The verifier samples an angle $\theta \in \{\pi/8, -\pi/8\}$ and sends θ to the prover.
2. The prover measures the qubit in B in the basis

$$\{\cos(\theta) |0\rangle + \sin(\theta) |1\rangle , \cos(\theta) |0\rangle - \sin(\theta) |1\rangle\}$$

to obtain an outcome $b \in \{0,1\}$ and sends it to the verifier.
3. The verifier accepts if the following conditions are satisfied:
   - If $c = 0$, accept iff $u^{\intercal} y = b$, where $y = (y_1, \ldots, y_m)$.
   - Otherwise: If $\theta = \pi/8$, accept iff $d^{\intercal} t = b$.
     If $\theta = -\pi/8$, accept iff $d^{\intercal} t \neq b$.

3.2 Analysis

First, notice that the honest parties only require storage linear in n to run the protocol. Specifically, the verifier only needs to keep in memory the variables s, a_i (one at a time), u, d and y, all of size $O(n)$. On the other hand, the prover's memory consists of $n+1$ classical bits (to store each a_i), along with $n+1$ qubits in X and an additional qubit in Y and B (one at a time).

We start with showing that the protocol is complete, namely that an honest quantum prover is able to convince the verifier with high probability.

Lemma 16 (Completeness)**.** *The verifier in an honest execution of the protocol accepts with probability* $\cos^2(\pi/8) - O(2^{-n})$.

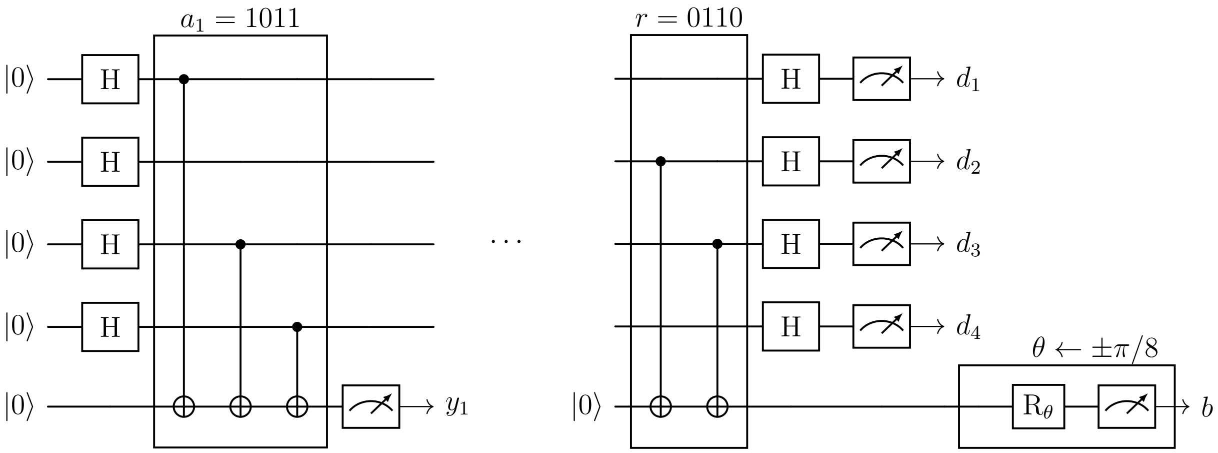

Figure 1: The quantum circuit executed by the prover in a run of the protocol with $n = 4$. In each of the first $2n$ rounds the prover receives $a_i \in \mathbb{F}_2^n$ from the verifier and responds by $y_i \in \mathbb{F}_2$. In round $2n+1$ he receives $r \in \mathbb{F}_2^n$ and returns $d \in \mathbb{F}_2^n$. In the last round, he receives an angle $\theta \leftarrow \{\pi/8, -\pi/8\}$ and responds by a bit b.

Proof. Let $\ell = n + 1$ and let $A \in \mathbb{F}^{m\times\ell}$ denote the matrix that is composed by the vectors a_i, sent by the verifier in Step 2.1. in the claw generation of Protocol 1, as its rows. Let $V \in \mathbb{F}^{m\times n}$ denote the matrix similarly composed by the random vectors v_i. It holds that $A = (V, Vs)$. By Proposition 4, V has full rank with probability at least $1 - O(2^{-n})$ and, therefore, A defines a 2-to-1 linear map with such a probability.

We show that the verifier accepts with probability $\cos^2(\pi/8)$ conditioned on A is 2-to-1. By the above, this is sufficient to derive the lemma.

It is convenient to delay the measurements done by the prover in Step 2.2. until after all iterations in claw generation are complete. This results in an equivalent residual state by the principle of delayed measurement. Applying the isometric map defined by a_i to the prover state results into

$$\frac{1}{\sqrt{2^{n+1}}} \sum_{x\in\mathbb{F}_2^{n+1}} |x, \psi_x\rangle_{\mathsf{XY}_{1\ldots i-1}} \mapsto \frac{1}{\sqrt{2^{n+1}}} \sum_{x\in\mathbb{F}_2^{n+1}} |x, \psi_x, a_i^\intercal x\rangle_{\mathsf{XY}_{1\ldots i}} .$$

We can then rewrite the state of the prover in the end of claw generation (but before any measurement) as

$$\frac{1}{\sqrt{2^{n+1}}} \sum_{x\in\mathbb{F}_2^{n+1}} |x, Ax\rangle_{\mathsf{XY}_{1\ldots m}} .$$

Measuring the last m qubits in registers $\mathsf{Y}_1, \ldots, \mathsf{Y}_m$ in the computational basis, the prover obtains a $y \in \mathbb{F}_2^m$ and the state collapses to

$$\frac{1}{\sqrt{2}} \sum_{x\in\mathbb{F}_2^{n+1} \,:\, Ax=y} |x\rangle = \frac{1}{\sqrt{2}} (|x_0\rangle + |x_1\rangle) .$$

Let $r \in \mathbb{F}_2^\ell$ be the string sent by the verifier in the commitment phase. The prover maps its state to

$$\frac{1}{\sqrt{2}}\left(|x_0\rangle + |x_1\rangle\right)_{\mathsf{X}} \mapsto \frac{1}{\sqrt{2}}\left(|x_0, r^\intercal x_0\rangle + |x_1, r^\intercal x_1\rangle\right)_{\mathsf{XB}}\,, \tag{1}$$

then measures X in the Hadamard basis and gets $d \in \mathbb{F}_2^\ell$. We distinguish between two cases.

- (Case I) $c = 0 \implies r^\intercal x_0 = r^\intercal x_1 = \delta$ for some $\delta \in \mathbb{F}_2$: In such a case, the prover's state before measuring B (RHS of Equation (1)) is a tensor product between a claw state in X and the qubit in B, therefore, the measurement of X has no effect on B. Thus, the prover's residual state after measuring is $|\delta\rangle$. In the CHSH test, the verifier verifies b against the most likely outcome of the prover's measurement, which is $\delta = r^\intercal x_0 = u^\intercal y$. It follows, then, that the prover succeeds with probability exactly $\cos^2(\pi/8)$.

- (Case II) $c = 1 \implies r^\intercal x_0 \neq r^\intercal x_1$: The residual state of the prover after measuring X is

  $$\frac{1}{\sqrt{2}}\left(|0\rangle + (-1)^{d\cdot(x_0+x_1)}|1\rangle\right).$$

  Thus, if $\theta = \pi/8$, the most likely outcome of the prover's measurement is $d^\intercal(x_0 + x_1)$, whereas if $\theta = -\pi/8$, the most likely outcome is $1 - d^\intercal(x_0 + x_1)$. Notice that if A has rank n then $\ker(A)$ has dimension 1, namely a single non-trivial element. Therefore, it must be the case that $x_0 + x_1 = t$. We conclude that the honest prover succeeds with probability exactly $\cos^2(\pi/8)$ in this case as well.

□

It remains to show that Protocol 1 is sound against classical cheating provers. In the following lemma, we show that if a classical prover is able to convince the verifier with probability significantly better than $3/4$, then he is able to find the vector s sampled by the verifier at the first step of claw generation. The hardness of finding s is implied directly by the bound on space complexity of learning parities from Lemma 5; computing s correspond precisely to learning parities give the verifier messages $(v_i, v_i^\intercal s)$ at claw generation.

Hence, together with Lemma 16, the following lemma completes the proof of Theorem 15.

Lemma 17 (Soundness)**.** *Assume there exists a classical adversary $\mathcal{P}^*(1^\lambda)$ that makes the verifier from Protocol 1 accept with probability larger than $3/4 + \varepsilon(\lambda)$ for some function $\varepsilon = 1/\operatorname{poly}$. Then, there exists an adversary $\mathcal{A}(1^\lambda)$ that interacts with the verifier in the claw generation sub-protocol (the first phase) and, at the end of the interaction, outputs s with probability at least $1/\operatorname{poly}(\lambda)$, where s is the vector sampled by the verifier at the beginning of the protocol. Further, the space complexity of $\mathcal{A}$ is larger than the space complexity of $\mathcal{P}^*$ by an additive factor of at most $O(\ell \log \ell)$.*

Proof. First, we switch to a modified experiment where the matrix A defined by the rows a_i sent by the verifier in claw generation is a rank-n matrix. Since this occurs with probability all but $O(2^{-n})$, the assumption on $\mathcal{P}^*$ that it convinces the verifier holds also in the modified experiment.

Let W be the internal state of the prover after the completion of claw generation (the first phase). Let $\texttt{accept}$ denote the event that the verifier accepts at the end of the interaction with $\mathcal{P}^*$. We define a "good" set G over the support of W as

$$G = \left\{ w : \Pr\left(\texttt{accept} \mid W = w\right) \geq \frac{3}{4} + \frac{\varepsilon(\lambda)}{2} \right\}.$$

We claim that $\Pr\left(W \in G\right) \geq \varepsilon(\lambda)/2$, where probability is over the random choices of $\mathcal{P}^*$ and the verifier. Assume not, then

$$\begin{aligned} \Pr\left(\texttt{accept}\right) &= \Pr\left(\texttt{accept} \mid W \in G\right) \cdot \Pr\left(W \in G\right) + \Pr\left(\texttt{accept} \mid W \notin G\right) \cdot \Pr\left(W \notin G\right) \\ &< \frac{\varepsilon(\lambda)}{2} + \frac{3}{4} + \frac{\varepsilon(\lambda)}{2} = \frac{3}{4} + \varepsilon(\lambda), \end{aligned}$$

in contradiction to the initial assumption on $\mathcal{P}^*$.

Next, we argue that the vector r that the verifier sends in Step 1 of the commitment phase distributes uniformly at random in the eyes of the prover. To see this, observe that r is sampled as follows:

- With probability $1/2$, r is sampled uniformly conditioned on $r^\intercal t = 1$.
- With probability $1/2$, r is uniformly sampled from the row-span of A. Conditioned on A is rank-n, t is the only non-trivial element in $\ker(A)$. Thus, this is equivalent to sampling a uniform r, conditioned on $r^\intercal t = 0$.

Such a distribution is equivalent to uniform since $r^\intercal t = 0$ with probability exactly half for a uniformly random r.

We describe an adversary $\mathcal{B}$ that takes as input a $\mathcal{P}^*$'s internal state after claw generation, and aims to predict $r^\intercal t$ for a uniformly random $r \leftarrow \mathbb{F}_2^\ell$ (where $\ell = n + 1$). $\mathcal{B}(w)$ performs the following:

1. On input a uniformly sampled $r \leftarrow \mathbb{F}_2^\ell$, $\mathcal{B}$ sends r to $\mathcal{P}^*(w)$ as the message from Step 1 of the commitment phase.
2. $\mathcal{P}^*$ returns some $d \in \mathbb{F}_2^\ell$.
3. Simulate $\mathcal{P}^*$ on both $\theta = \pi/8$ and $\theta = -\pi/8$, rewinding the algorithm in-between. If $\mathcal{P}^*$ outputs the same answer for both cases return 0, else return 1.

Let us denote by $\texttt{equal}(w)$ the event where the answers of simulated $\mathcal{P}^*$ in Step 3 of $\mathcal{B}(w)$ are indeed identical for both $\theta = \pi/8$ and $\theta = -\pi/8$. To conclude the proof of the theorem it suffices to show that

$$\Pr_{r,\mathcal{P}^*}\left(\texttt{equal}(w) \mid r^\intercal t = 0, w \in G\right) - \Pr_{r,\mathcal{P}^*}\left(\texttt{equal}(w) \mid r^\intercal t = 1, w \in G\right) \geq 2\varepsilon(\lambda). \qquad (2)$$

This is indeed sufficient, since it implies the existence of a predictor that has bias ε against $r^\intercal t$ conditioned on $w \in G$. Consequently, by Lemma 6 (Goldreich-Levin), there exists an extractor with memory $O(\ell \log \ell)$ that outputs t, given oracle access to $\mathcal{B}(w)$ that is successful when $w \in G$. Given such an extractor, we define $\mathcal{A}$ as follows: Simulate $\mathcal{P}^*$ in the claw generation with the verifier and obtain an internal state w, then use the extractor with access to $\mathcal{B}(w)$ to extract t. Since $w \in G$ with probability at least $\varepsilon/2$, it suffices to argue inverse-polynomial success probability of $\mathcal{A}$ given $w \in G$, which follows by the success of the Goldreich-Levin extractor.

To conclude, we turn to the proof of Eq. (2), which is derived by the following

$$\begin{aligned}
\frac{3}{4} + \frac{\varepsilon(n)}{2} &\leq \Pr_{r \in \mathbb{F}_2^{n+1}} (\texttt{accept} \mid w \in G) \\
&= \frac{1}{2} \Pr_{r \in \mathbb{F}_2^{n+1}} (\texttt{accept} \mid r^\intercal t = 0, w \in G) + \frac{1}{2} \Pr_{r \in \mathbb{F}_2^{n+1}} (\texttt{accept} \mid r^\intercal t = 1, w \in G) \\
&\leq \frac{1}{4} + \frac{1}{4} \Pr_{r \in \mathbb{F}_2^{n+1}} (\texttt{equal}(w) \mid r^\intercal t = 0, w \in G) + \frac{1}{2} \Pr_{r \in \mathbb{F}_2^{n+1}} (\texttt{accept} \mid r^\intercal t = 1, w \in G) \\
&\leq \frac{1}{4} + \frac{1}{4} \Pr_{r \in \mathbb{F}_2^{n+1}} (\texttt{equal}(w) \mid r^\intercal t = 0, w \in G) + \frac{1}{2} - \frac{1}{4} \Pr_{r \in \mathbb{F}_2^{n+1}} (\texttt{equal}(w) \mid r^\intercal t = 1, w \in G) \\
&= \frac{3}{4} + \frac{1}{4} \left(\Pr_{r \in \mathbb{F}_2^{n+1}} (\texttt{equal}(w) \mid r^\intercal t = 0, w \in G) - \Pr_{r \in \mathbb{F}_2^{n+1}} (\texttt{equal}(w) \mid r^\intercal t = 1, w \in G) \right).
\end{aligned}$$

□

4 Proof of Quantumness with Arbitrary Gap

Our second main result is a proof of quantumness protocol that can exhibit up to an exponential gap between the space complexity of the honest parties and the space complexity of the best attack. The protocol instantiates a template laid down by [BGK+23] which, similarly to our protocol from Section 3, follows the general framework from the literature [BCM+21, KCVY22, KLVY23b] that bases PoQ on claw generation. Unlike the simpler protocol from Section 3, the claw here is generated in an interactive process, namely via *interactive hashing* [NOVY92]. While the protocol, just like our first one, guarantees only a constant gap between completeness and soundness, the gap can be naturally amplified by sequential repetition. Formally, we obtain the following result.

Theorem 18 (PoQ with Arbitrary Gap). *There exists a protocol between a classical verifier and a quantum prover where the two parties take as input a security parameter $\lambda \in \mathbb{N}$ and, at the end of the interaction, the verifier either accepts or rejects, and*

- *(Completeness) The verifier accepts with probability $\cos^2(\pi/8)$ when interacting with an honest prover.*
- *(Soundness) Any non-uniform classical prover $\mathcal{P}^*$ that on input λ uses less than $m(\lambda)$ memory bits has success probability at most $3/4 + 2^{-\Omega(\lambda)}$ in making the verifier accept.*
- *(Complexity) The verifier and (honest) prover run in time $O(\lambda^7 m^3 \cdot \operatorname{polylog} m)$ and in space $O(\lambda \cdot \operatorname{polylog} m)$.*

4.1 PoQ from Claw Generation

We recall the PoQ outline from [BGK$^+$23, Figure 4], which assumes the existence of a claw generation sub-protocol. In fact, [BGK$^+$23] assumes a special case of claw generation, namely claw generation via *trapdoor claw-free functions* (TCF). In contrast, we consider a more general outline where claw generation may be performed arbitrarily as long as it guarantees completeness, i.e. that the prover obtains a claw state, and soundness, i.e. that it is hard for the verifier to compute both values in the claw simultaneously. This does not affect the analysis of the protocol at all: its completeness and soundness follow from those of claw generation, just as in [BGK$^+$23]. Nevertheless, we provide proofs for the sake of completeness.

Protocol 2 (PoQ from Claw Generation)**.** The following proof of quantumness is parameterized by a security parameter $\lambda \in \mathbb{N}$ and consists of three phases:

- (Claw Generation) The prover and verifier engage in a *claw generation sub-protocol* at the end of which the verifier has a pair of *claw values* $x_0, x_1 \in \{0,1\}^\ell$, where $\ell := \ell(\lambda)$ is a polynomial, and the prover's residual state is the *claw state*

$$\frac{1}{\sqrt{2}}(|x_0\rangle + |x_1\rangle)_{\mathsf{X}}. \tag{3}$$

  Additionally, we assume that the prover obtains a function $g : \{0,1\}^\ell \to \{0,1\}$ that satisfies $g(x_0) = 0$ and $g(x_1) = 1$.

- (Commitment)

  1. The verifier samples uniformly random $r_0, r_1 \leftarrow \mathbb{F}_2^\ell$ and sends it to the prover.
  2. The prover applies the following isometric mapping to his state

$$|x\rangle_{\mathsf{X}} \mapsto |x\rangle_{\mathsf{X}} \left|r_{g(x)}{}^{\intercal} x\right\rangle_{\mathsf{B}},$$

     then measures X in Hadamard basis to obtain $d \in \mathbb{F}_2^\ell$ and sends it to the verifier.

- (CHSH Test)

  1. The verifier samples an angle $\theta \leftarrow \{\pi/8, -\pi/8\}$ and sends θ to the prover.
  2. The prover measures the qubit in B in the basis

$$\{\cos(\theta)\,|0\rangle + \sin(\theta)\,|1\rangle\,, \cos(\theta)\,|0\rangle - \sin(\theta)\,|1\rangle\}$$

     to obtain an outcome $b \in \{0,1\}$ and sends it to the verifier.
  3. Let $\alpha = r_0^\intercal x_0 \oplus r_1^\intercal x_1 = (r_0||r_1)^\intercal(x_0||x_1)$. The verifier accepts if the following conditions are satisfied (hereby, arithmetics are over the integers):
     - If $\theta = \pi/8$, accept iff

$$(-1)^b = (1-\alpha)(-1)^{r_0 x_0} + \alpha(-1)^{d^\intercal(x_0 \oplus x_1)}. \tag{4}$$

- If $\theta = -\pi/8$, accept iff

$$(-1)^b = (1-\alpha)(-1)^{r_0 x_0} - \alpha(-1)^{d^\intercal(x_0 \oplus x_1)}. \tag{5}$$

In the following, we confirm that if the prover in Protocol 2 completes claw generation with the claw state from Equation (3), then he is able to convince the verifier into accepting with good probability. The proof follows the lines of the proof of Proposition 5.4 in [BGK+23].

Lemma 19 (Completeness [BGK+23])**.** *The verifier in Protocol 2 accepts with probability* $\cos^2(\pi/8) \approx 0.853$ *when interacting with an honest quantum prover.*

Proof. Assuming the completeness of claw generation, the prover's state at the end of the first phase of Protocol 2 is $(|x_0\rangle + |x_1\rangle)/\sqrt{2}$. Let $r_0, r_1 \in \mathbb{F}_2^\ell$ be the strings sent by the verifier in Step 1 of the commitment phase. The prover maps its state to

$$\frac{1}{\sqrt{2}}\left(|x_0\rangle + |x_1\rangle\right)_{\mathsf{X}} \mapsto \frac{1}{\sqrt{2}}\left(|x_0, r_0^\intercal x_0\rangle + |x_1, r_1^\intercal x_1\rangle\right)_{\mathsf{XB}}, \tag{6}$$

then measures X in the Hadamard basis and gets $d \in \mathbb{F}_2^\ell$, which leaves him with

$$\frac{(-1)^{d^\intercal x_0}}{\sqrt{2}}\left(|r_0^\intercal x_0\rangle + (-1)^{d^\intercal(x_0 \oplus x_1)}\,|r_1^\intercal x_1\rangle\right).$$

From here, it follows by inspection that, when the above state is measured in the basis $\{|\pi/8\rangle, |5\pi/8\rangle\}$ (i.e. when $\Theta = \pi/8$), then b satisfying Equation (4) is the most likely outcome with probability $\cos^2(\pi/8)$. Otherwise, when the state is measured in the basis $\{|-\pi/8\rangle, |3\pi/8\rangle\}$, the most likely outcome is b satisfying Equation (5). □

The soundness of Protocol 2 holds whenever it is hard for any classical prover to compute both values in the claw from Equation (3) simultaneously, i.e. x_0 and x_1.

Lemma 20 (Soundness [BGK+23])**.** *Assume there exists a classical adversary* $\mathcal{P}^*(1^\lambda)$ *that makes the verifier from Protocol 2 accept with probability larger than* $3/4 + \varepsilon(\lambda)$ *for some function* $\varepsilon = 1/\operatorname{poly}$*. Then, there exists an adversary* $\mathcal{A}(1^\lambda)$ *that interacts with the verifier in the claw generation sub-protocol (the first phase) and, at the end of the interaction, outputs* (x_0, x_1) *with probability at least* $1/\operatorname{poly}(\lambda)$*, where* $x_0, x_1 \in \{0,1\}^\ell$ *are the claw values that the verifier obtains. Further, the space complexity of* $\mathcal{A}$ *is larger than the space complexity of* $\mathcal{P}^*$ *by an additive factor of at most* $O(\ell \log \ell)$*.*

Proof. The soundness analysis from [BGK+23] is in two steps: (i) First, they observe that a classical prover that succeeds with probability better than 3/4 can essentially predict the value $\alpha = (r_0||r_1)^\intercal(x_0||x_1)$ used for the verification in Step 3 in the CHSH test. (ii) Second, by a Goldreich-Levin argument similar to that from the proof of soundness of our first protocol (Lemma 17), they show that a predictor of $(r_0||r_1)^\intercal(x_0||x_1)$ for random r_0, r_1 can be transformed into an algorithm that computes x_0, x_1.

Similarly to the proof of Lemma 17, we begin by fixing an internal state of $\mathcal{P}^*$ after the claw generation is complete, with which he succeeds to convince the verifier at the end with good probability. Specifically, letting W denote $\mathcal{P}^*$'s state after claw generation (the first phase) and `accept` denote the event that the verifier accepts, we again define a "good" set G over the support of W as

$$G = \left\{ w : \Pr\left(\texttt{accept} \mid W = w\right) \geq \frac{3}{4} + \frac{\varepsilon(\lambda)}{2} \right\}.$$

By an averaging argument, it holds that $\Pr\left(W \in G\right) \geq \varepsilon(\lambda)/2$.

Next, we proceed with showing (i). Denote by b_+ the value of b that satisfies Equation (4) and by b_- the value that satisfies Equation (5). Then, it holds that $(-1)^{b_+ \oplus b_-} = (1-\alpha)^2 - \alpha^2 = 1 - 2\alpha = (-1)^{\alpha}$. Hence, predicting the parity $b_+ \oplus b_-$ is equivalent to predicting α. We describe an adversary $\mathcal{B}$ that takes as input a $\mathcal{P}^*$'s internal state after claw generation, and aims to predict $\alpha = (r_0||r_1)^\intercal(x_0||x_1)$ for uniformly random $r_0, r_1 \leftarrow \mathbb{F}_2^\ell$. $\mathcal{B}(w)$ performs the following:

1. On input uniformly sampled $r_0, r_1 \leftarrow \mathbb{F}_2^\ell$, $\mathcal{B}$ sends r_0, r_1 to $\mathcal{P}^*(w)$ as the message from Step 1 of the commitment phase.

2. $\mathcal{P}^*$ returns some $d \in \mathbb{F}_2^\ell$.

3. Simulate $\mathcal{P}^*$ on both $\theta = \pi/8$ and $\theta = -\pi/8$, rewinding the algorithm in-between. If $\mathcal{P}^*$ outputs the same answer for both cases return 0, else return 1.

We argue that, given $w \in G$, $\mathcal{B}(w)$ predicts $b_+ \oplus b_-$, and therefore α, with advantage at least $2\varepsilon(\lambda)$. This follows by the same derivation with which we proved the advantage of the same predictor in the proof of Lemma 17, namely Equation (2) (there, the corresponding parity $b_+ \oplus b_-$ is precisely the inner product $r^\intercal t$).

The proof is then complete via (ii), again similarly to the proof of Lemma 17. The predictor $\mathcal{B}$ against $(r_0||r_1)^\intercal(x_0||x_1)$ implies, via Lemma 6 (Goldreich-Levin), an algorithm $\mathcal{A}$ that computes x_0, x_1 as follows: $\mathcal{A}$ simulates $\mathcal{P}^*$ in the claw generation with the verifier and obtains an internal state w. Then, $\mathcal{A}$ applies the Goldreich-Levin extractor with access to $\mathcal{B}(w)$. By Lemma 6, $\mathcal{A}$ uses at most $O(\ell \log \ell)$ additional memory compared to $\mathcal{P}^*$. □

4.2 Interactive Hashing

In a (classical) *interactive hashing* protocol [NOVY92], two parties, say Alice (to be thought of as a challenger, or the verifier later in our context) and Bob (a challengee, or the prover), engage in an interaction the defines a 2-to-1 hash function h over $[k]$ and a hash value y. An interactive hashing protocol allows Bob to control one of the pre-images of y under h, specifically to make it so $h(v) = y$ for an input v of his choice. However, the soundness of interactive hashing prevents Bob from controlling *both* pre-images.

Definition 21 (Interactive Hashing [NOVY92])**.** *An interactive hashing protocol is a classical protocol between a public-coin Alice (the verifier in our context) and a deterministic Bob (the prover). Alice has no input and Bob has an input $v \in [k]$. For any choice of Alice's public randomness h and Bob's input v, we denote by $y = h(v)$ the deterministic answers computed by (an honest) Bob. The protocol satisfies the following:*

- *(2-to-1 Hash) Any random choice of h is a 2-to-1 function. That is, for any h and y, $|h^{-1}(y)| = 2$.*
- *((α, β)-Security) For any fixed set $B \subseteq [k]$ of size at most βk, for any (possibly malicious, unbounded) Bob's strategy which, on input public coins h results in an arbitrary y such that $h^{-1}(y) = \{v_0, v_1\}$, it holds that:*

$$\Pr\left(\{v_0, v_1\} \subseteq B\right) \leq \alpha.$$

We say that an interactive hashing protocol is stateless if Bob is not required to keep an intermediate state between the rounds of the protocol. Namely, if Bob's message at any round is a deterministic function of its input v and Alice's public coins at that round.

The seminal work of Naor et al. [NOVY92] devises a simple interactive hashing protocol, where the hash function is a random 2-to-1 linear map, sent by Alice one row at a time. Bob, in turn, answers by multiplying the rows with its input v at every round, hence the protocol is stateless and has a linear number of rounds. Ding et al. [DHRS04] propose an improvement over the [NOVY92] protocol where Alice sends random hash functions with certain t-wise independence properties, and requires only 4 rounds of interaction. The protocol from [DHRS04] is also stateless as per Definition 21: at every round, Bob merely applies the functions sent by Alice on his input and sends back the result.

Theorem 22 (Constant-round Interactive Hashing [DHRS04])**.** *There exists a stateless 4-round interactive hashing protocol that is (α, β)-secure for any $\beta > 0$ with $\alpha = O(\beta \log k)$. The execution of the protocol and the computation of h^{-1} can be done in time and space polylogarithmic in k.*

Morimae and Yamakawa [MY23] show how to perform the interactive hashing protocol from [NOVY92] in a setting where Bob's input is a quantum state rather than a classical input. They propose a coherent implementation where Bob's state at the end of the protocol is a superposition over all possible classical inputs consistent with the obtained transcript. In the following lemma, we generalize their implementation to any stateless interactive hashing protocol.

Lemma 23 (Coherent Implementation of Interactive Hashing)**.** *For any stateless interactive hashing protocol over k-bit input, there exists a coherent implementation of the protocol between a classical Alice and a quantum Bob with k-qubit input register V, satisfying the following:*

- *(Correctness) If Bob's state before the protocol is:*

$$\rho_{\mathsf{VW}} = \sum_{v \in \{0,1\}^k} |v\rangle_{\mathsf{V}} |\psi_v\rangle_{\mathsf{W}},$$

and the transcript of the protocol upon its completion is (h, y), then Bob's state at the end of the protocol is:

$$\tilde{\rho}_{\mathsf{VW}} = \sum_{v:\ h(v)=y} |v\rangle_{\mathsf{V}} |\psi_v\rangle_{\mathsf{W}}.$$

- *(Complexity) The time and space complexity of quantum Bob in the implementation is polynomial in the time and, resp., space complexity of Bob in the classical protocol.*

Proof. Let h_i and y_i denote Alice's public coins and, respectively, Bob's message at round i of the protocol. Let $y_i \leftarrow f_i(v, h_i)$ denote Bob's computation at round i. The coherent interactive hashing over Bob's quantum input performs the protocol as follows. At round i, upon receiving Alice's random coins, Bob applies the following mapping to its state

$$|v, z\rangle_{\mathsf{VY}} \mapsto |v, z \oplus f_i(v, h_i)\rangle_{\mathsf{VY}} ,$$

where Y is an additional ancilla register (created new and initialized to $|0\rangle$ at every round). Bob measures the register Y to obtain y_i and sends it to Alice.

To see why the implementation satisfies correctness, note that we may purify the execution of the protocol as follows: Apply the following for a random choice of $h = (h_1, \dots, h_r)$

$$|v, z_1, \dots, z_r\rangle \mapsto |v, z_1 \oplus f_1(v, h_1), \dots, z_r \oplus f_r(v, h_r)\rangle_{\mathsf{VY}_1 \dots \mathsf{Y}_r} ,$$

then measure the registers $\mathsf{Y}_1, \dots, \mathsf{Y}_r$ to obtain y (recall h is sampled independently in Bob's input). In particular, when applied over ρ_{VW} and Y_i registers that are initiated to ancillas, the above mapping gives

$$\begin{aligned} \sum_{v \in \{0,1\}^k} |v\rangle_{\mathsf{V}} |0\rangle_{\mathsf{Y}_1 \dots \mathsf{Y}_r} |\psi_v\rangle_{\mathsf{W}} \mapsto & \sum_{v \in \{0,1\}^k} |v\rangle_{\mathsf{V}} |f_1(v, h_1), \dots, f_r(v, h_r)\rangle_{\mathsf{Y}_1 \dots \mathsf{Y}_r} |\psi_v\rangle_{\mathsf{W}} \\ = & \sum_{v \in \{0,1\}^k} |v\rangle_{\mathsf{V}} |h(v)\rangle_{\mathsf{Y}} |\psi_v\rangle_{\mathsf{W}} . \end{aligned}$$

Evidently, when we measure Y in the above state and obtain y, we obtain the claimed residual state $\tilde{\rho}_{\mathsf{VW}}$. □

4.3 Claw Generation

Equipped with the formulation of coherent interactive hashing, we are prepared to present our claw generation protocol. The following lemma immediately implies Theorem 18 through Lemmas 19 and 20. In addition, we prove that our protocol provides soundness against space-bounded quantum attackers, albeit with inverse-polynomial soundness error. We elaborate on the applications of quantum soundness in Section 4.5.

Lemma 24 (Claw Generation)**.** *Let $m := m(\lambda)$ be a bound on memory. Let $k := k(\lambda)$ be an additional security parameter such that $k / \log k > \lambda(m + \Omega(\lambda))$. There exists a protocol between a classical verifier and a quantum prover, where both parties take as input a security parameter 1^λ and at the end of the protocol the verifier outputs a pair of values $x_0, x_1 \in \{0,1\}^{\ell(\lambda)}$, that satisfies the following:*

- *(Correctness) The residual state of the prover at the end of the protocol is*

$$\frac{1}{\sqrt{2}}(|x_0\rangle + |x_1\rangle),$$

with probability 1, where (x_0, x_1) is the output of the verifier. Additionally, the prover obtains a function $g : \{0,1\}^\ell \to \{0,1\}$ that satisfies $g(x_0) = 0$ and $g(x_1) = 1$.

- *(Claw-Finding Hardness) Any non-uniform classical, or quantum, adversary with memory of at most m bits (resp., qubits) that interacts with the verifier, outputs (x_0, x_1) with probability at most $2^{-\Omega(\lambda)}$, respectively $O(\lambda^2\sqrt{m/k})$, at the end of the interaction, where (x_0, x_1) is the output of the verifier.*
- *(Complexity) Both the prover and the verifier run in time $O(\lambda \cdot k^3)$ in expectation (and with probability all but negligible in λ) and use $O(\lambda \cdot \operatorname{polylog} k)$ space.*

In particular, setting $k = \Theta(\lambda(m+\lambda)\log m)$ gives a protocol with classical soundness (with negligible soundness error), runtime $O(\lambda m^3 \operatorname{polylog} m)$ and space $O(\lambda \operatorname{polylog} m)$. Setting $k = \Theta(\lambda^{2c} m)$ gives a protocol with quantum soundness up to soundness error $1/\lambda^c$, polynomial runtime and space $\operatorname{polylog} m$.

The rest of this subsection is dedicated to presenting the protocol underlying Lemma 24 and analyzing its correctness and complexity. The proof of security (hardness of finding a claw) is deferred to Sections 4.4 and 4.5.

The Protocol. Our claw generation protocol is parametrized by a *stream length* $k := k(\lambda)$ which is chosen such that $k/(\lambda \log k) > m + \Omega(\lambda)$. We assume for notational convenience that k is always a power of 2. The protocol makes use of a coherent interactive hashing sub-routine (see Definition 21 and Lemma 23), which in particular may be based on the protocol from Theorem 22.

The protocol proceeds as follows:

1. (1-bit Claws Generation) For $j \in [\lambda]$, the prover creates new $\log k$-qubit register V_j and 1-qubit register U_j and engages in the following interaction with the verifier:

   1.1. (Setup) The verifier samples two uniform indices $v_0, v_1 \leftarrow [k]$ and the prover prepares the uniform superposition over V_j and the 0 state in U_j:

$$|\psi\rangle = \sum_{v \in [k]} |v\rangle_{\mathsf{V}_j} \otimes |0\rangle_{\mathsf{U}_j} .$$

   1.2. (Streaming) The verifier uniformly samples and transmits to the prover a stream of k bits $U = (U_1, \ldots, U_k)$, one bit at a time. The verifier stores (v_0, U_{v_0}) and (v_1, U_{v_1}) in his memory. Upon receiving the i-th bit U_i, the prover applies to its state the unitary defined by the following map:

$$|v\rangle_{\mathsf{V}_j} \otimes |u\rangle_{\mathsf{U}_j} \mapsto |v\rangle_{\mathsf{V}_j} |u \oplus f_{i,U_i}(v)\rangle_{\mathsf{U}_j} \quad \text{where} \quad f_{i,U_i}(v) = \begin{cases} U_i & \text{if } v = i \\ 0 & \text{otherwise.} \end{cases}$$

1.3. (Interactive Hashing) The verifier and the prover engage in a run of the coherent interactive hashing protocol over $[k]$, where the input of the prover is the register V_j. Let (h, y) be the transcript of the interactive hashing upon its completion. If $h^{-1}(y) \neq \{v_0, v_1\}$, the verifier aborts and starts over with the Setup phase. Otherwise, the verifier stores:

$$v_0^j = v_0 \qquad v_1^j = v_1 \qquad z_0^j = U_{v_0} \qquad z_1^j = U_{v_1}.$$

2. (Amplification) The prover and the verifier engage in the following protocol:

2.1. For any $j \in [\lambda]$, the verifier arbitrarily picks a function $g^j : [k] \to \{0, 1\}$ with short description satisfying $g^j(v_b^j) = b$ for $b \in \{0, 1\}$,[2] The verifier sends $(g^1, \dots, g^\lambda)$ to the prover.

2.2. For $j = 2 \dots \lambda$, the prover adds a new 1-qubit ancilla register B_j and applies the following mapping over $\mathsf{V}_1\mathsf{V}_j\mathsf{B}_j$:

$$\left|v^1\right\rangle_{\mathsf{V}_1} \otimes \left|v^j\right\rangle_{\mathsf{V}_j} \otimes |b\rangle_{\mathsf{B}_j} \mapsto \left|v^1\right\rangle_{\mathsf{V}_1} \otimes \left|v^j\right\rangle_{\mathsf{V}_j} \otimes \left|b \oplus g^1(v^1) \oplus g^j(v^j)\right\rangle_{\mathsf{B}_j}.$$

2.3. The prover measures $\mathsf{B}_2 \dots \mathsf{B}_\lambda$ to obtain an outcome $(b_2, \dots, b_\lambda) \in \{0, 1\}^{\lambda-1}$, which he sends to the verifier. The verifier outputs:

$$x_0 = (v_0^1, z_0^1, v_{b_2}^2, z_{b_2}^2, \dots, v_{b_\lambda}^\lambda, z_{b_\lambda}^\lambda) \quad x_1 = (v_1^1, z_1^1, v_{1-b_2}^2, z_{1-b_2}^2, \dots, v_{1-b_\lambda}^\lambda, z_{1-b_\lambda}^\lambda).$$

Complexity. The runtime of one attempt of a 1-bit claw generation is linear in k for both parties (note interactive hashing is performed over the domain $[k]$ which is of polylogarithmic size). An attempt succeeds with probability $1/k^2$ since the verifier samples v_0, v_1 uniformly and independently of the other random choices in the protocol. Therefore, in expectation, the claw generation for a single bit takes $O(k^2)$ time. It also takes $O(k^2)$ time with probability all but negligible in k via a standard Chernoff argument (Lemma 13). Since 1-bit claw generation is repeated λ times, this results in $O(\lambda k^3)$ runtime in total, which easily dominates over the runtime of the amplification phase. As for space complexity, observe that, for each of the λ bits in the claw, both parties require memory at most polylogarithmic in k since they both store a constant amount of variables over $[k]$ and apply interactive hashing over this domain.

Correctness. We show that the output of the protocol results into a well-formed claw (see Lemma 24), with certainty, provided that the protocol terminates.

Consider a prover that follows the protocol honestly. At any round $j \in [\lambda]$, once the streaming in Step 1.2. is complete, the prover obtains the superposition $\sum_{v \in [k]} |v\rangle |U_v\rangle$ in the registers $\mathsf{V}_j\mathsf{U}_j$, where U is the communicated stream at that round. By Lemma 23, the following coherent interactive hashing step, if succeeds, results in the state $\left|v_0^j\right\rangle \left|z_0^j\right\rangle + \left|v_1^j\right\rangle \left|z_1^j\right\rangle$ in these registers. Notice that the generation of these 1-bit claws is independent across all j and, hence, the prover's state prior to the stitching stage may be written as:

$$\left(\sum_{\beta \in \{0,1\}} \left|v_\beta^1\right\rangle_{\mathsf{V}_1} \left|z_\beta^1\right\rangle_{\mathsf{U}_1}\right) \otimes \cdots \otimes \left(\sum_{\beta \in \{0,1\}} \left|v_\beta^\lambda\right\rangle_{\mathsf{V}_\lambda} \left|z_\beta^\lambda\right\rangle_{\mathsf{U}_\lambda}\right).$$

[2] For instance, g^j can be a dictator function with description size $\log \log k + 1$.

By inspection, since $g^1(v^1_{\beta_1}) \oplus g^j(v^j_{\beta_j}) = \beta_1 \oplus \beta_j$ for any j, the mapping performed at Step 2.2. gives:

$$\sum_{\beta_1 \in \{0,1\}} \left|v^1_{\beta_1}\right\rangle_{\mathsf{V}_1} \left|z^1_{\beta_1}\right\rangle_{\mathsf{U}_1} \otimes \bigotimes_{j=2\ldots\lambda} \left(\sum_{\beta_j \in \{0,1\}} \left|v^j_{\beta_j}\right\rangle_{\mathsf{V}_j} \left|z^j_{\beta_j}\right\rangle_{\mathsf{U}_j} \left|\beta_1 \oplus \beta_j\right\rangle_{\mathsf{B}_j} \right).$$

Thus, when applying the measurement at Step 2.3. and obtaining $(b_2, \ldots, b_\lambda)$, the prover is left with the following state:

$$\sum_{\beta_1 \in \{0,1\}} \left|v^1_{\beta_1}\right\rangle_{\mathsf{V}_1} \left|z^1_{\beta_1}\right\rangle_{\mathsf{U}_1} \otimes \left|v^2_{\beta_1 \oplus b_2}\right\rangle_{\mathsf{V}_2} \left|z^2_{\beta_1 \oplus b_2}\right\rangle_{\mathsf{U}_2} \otimes \cdots \otimes \left|v^\lambda_{\beta_1 \oplus b_\lambda}\right\rangle_{\mathsf{V}_\lambda} \left|z^\lambda_{\beta_1 \oplus b_\lambda}\right\rangle_{\mathsf{U}_\lambda},$$

which is a superposition of the two values x_0 and x_1 that the verifier outputs.

Lastly, recall that we require the prover to obtain a function $g : \{0,1\}^\ell \to \{0,1\}$ that differentiates between the two values of the claw by satisfying $g(x_b) = b$. This function may be set to be g^1 (which the prover receives from the verifier), applied to the first $\log k$ bits of x_b, namely the part containing the index v^1_b.

4.4 Classical Hardness of Finding a Claw

We show a bound on the success probability of any *classical* prover to output a claw. For $j \in [\lambda]$, let U^j be the random variable taking the value of the string U streamed in Step 1.2., before the first successful completion of the interactive hashing step, namely the first attempt where the verifier does not abort and start over in Step 1.3.. Let W^j_{pre} denote the random variable consisting of the adversary's internal state, i.e., its m-bit memory, right after the streaming of U^j has completed and just before the start of the interactive hashing. Let W^j_{post} denote its internal state after the completion of the interactive hashing. For $b \in \{0,1\}$, we let the random variable Z^j_b take the value of the bit z^j_b that is stored by the verifier in Step 1.3. at round j and let $Z_b = (Z^1_b, \ldots, Z^\lambda_b)$ (note these values are part of the claw x_0, x_1).

By definition, the following lemma implies the classical hardness of finding a claw.

Lemma 25 (Classical Hardness of Claw-Finding)**.** *It holds that* $H_\infty(Z_0, Z_1 \mid W^\lambda_{post}) \geq \Omega(\lambda)$.

Let $t = k/(\lambda \log k) - m = \Omega(\lambda)$. Denote by Bad_j the event that $W^j_{pre} = \omega$ for $\omega \in \{0,1\}^m$ satisfying:

$$H_\infty(U^j \mid W^j_{pre} = \omega, W^j_{post}) < k - (m + t).^{3} \tag{7}$$

Let us first bound the probability that the above bad event occurs for any round j.

Claim 26 (Bad Events)**.** $\Pr\left(\bigcup_j \mathrm{Bad}_j\right) < 2^{-\Omega(\lambda)}$.

Proof. By Proposition 9, since U^j is independent in W^j_{post} given W^j_{pre}, it holds that $H_\infty(U^j \mid W^j_{pre} = \omega, W^j_{post}) \geq H_\infty(U^j \mid W^j_{pre} = \omega)$. Thus, for any j, we have

$$\Pr(\mathrm{Bad}_j) < 2^{k-(m+t)-H_\infty(U^j|W^j_{pre})} < 2^{k-t-H_\infty(U^j)} = 2^{-t},$$

[3] Note this is the conditional entropy of the marginal distribution of U_j when $W^j_{pre} = \omega$, conditioned on the random variable W^j_{post}.

where the first inequality is by Lemma 8 and the second by Lemma 7. A union bound is then sufficient to derive the inequality. □

In the next proposition, we invoke the soundness of the interactive hashing to argue the unpredictability of *both* Z_0^j and Z_1^j, for any round j, even given the adversary's bounded storage.

Claim 27 (Entropy Bound). *Assume* Bad_j *does not occur, and fix any* $W_{pre}^j = \omega$ *in the support.Then, it holds that:*

$$\Pr\left(H_\infty(Z_0^j, Z_1^j \mid \omega, W_{post}^j) > 0.8\right) = 1 - O(1/\lambda)$$

where the probability is taken over the randomness of the interactive hashing at round j*.*

Proof. Conditioned on Bad_j not occurring, fix any $W^j = \omega$ in the support. We have:

$$\begin{aligned}\mathbb{E}_{i\leftarrow[k]}\left(H(U_i^j \mid W_{pre}^j = \omega, W_{post}^j)\right) &= \frac{1}{k}\sum_{i\in[k]} H(U_i^j \mid \omega, W_{post}^j)\\ &\geq \frac{1}{k} H_\infty(U^j \mid \omega, W_{post}^j)\\ &\geq \frac{1}{k}(k-(m+t)) = 1-(m+t)/k,\end{aligned}$$

where the first inequality follows by Proposition 10 and the second one follows be the definition of the Bad_j event in Eq. (7). Define the set:

$$B_\omega^j := \left\{i \in [k] \mid H(U_i^j \mid W_{pre}^j = \omega, W_{post}^j) < 0.99\right\}.$$

By an averaging argument, it must hold that $\Pr_{i\leftarrow[k]}(i \in B_\omega^j) \leq 100(m+t)/k$, and consequently that $|B_\omega^j| \leq 100(m+t)$. Thus, by Theorem 22, we have that:

$$\Pr\left(\{v_0^j, v_1^j\} \subseteq B_\omega^j\right) = O((m+t)\log k/k) = O(1/\lambda),$$

where probability is over the randomness of the interactive hashing in the choice of $\{v_0^j, v_1^j\}$ and it holds for any (possibly unbounded) prover. By definition, we have:

$$\Pr\left(\exists b \in \{0,1\}: \ H(Z_b^j \mid W_{pre}^j = \omega, W_{post}^j) = 0.99\right) > 1 - O(1/\lambda).$$

From Proposition 11, this implies:

$$\Pr\left(\exists b \in \{0,1\}: \ H_\infty(Z_b^j \mid W_{pre}^j = \omega, W_{post}^j) > 0.8\right) = 1 - O(1/\lambda),$$

and, in particular, the proposition follows. □

Fix any sequence $\omega = (\omega^1, \ldots, \omega^\lambda)$ of values taken by $W_{pre}^1, \ldots, W_{pre}^\lambda$ throughout the protocol. Define the random variable:

$$J_\omega := \left\{j : H_\infty(Z_0^j, Z_1^j \mid W_{pre}^j = \omega^j, W_{post}^j) < 0.8\right\} \subseteq [\lambda].$$

Claim 28. *Assume* Bad_j *does not occur for any* j*. Then, it holds that* $\Pr(|J_\omega| \geq \lambda/2) < 2^{-\Omega(\lambda^2)}$ *for any* ω *in the support, where probability is over the randomness of the interactive hashing invocations.*

Proof. Assuming that ω are such that Bad_j does not hold for any j, we have that

$$\mathbb{E}\left(|J_\omega|\right) = \lambda \cdot O(1/\lambda) = O(1)$$

by Claim 27 (expectation is taken over the interactive hash invocations). Further, notice that upon fixing ω, the membership of the different j's in J_ω constitute anti-correlated predicates, since the probability of each is bounded by the soundness of the interactive hash, independently of what happens in other rounds. Hence, we may invoke Chernoffs's inequality (Lemma 13) to obtain the following tail bound on the size of J_ω:

$$\Pr\left(|J_\omega| \geq \lambda/2\right) < e^{-\Omega(\lambda^2)} = e^{-\Omega(\lambda^2)}.$$

□

We consider a run of the protocol conditioned on Bad_j not occurring for any j and $|J_\omega| < \lambda/2$, where $\omega = (\omega^1, \ldots, \omega^\lambda)$ are the values taken by $W^1_{pre}, \ldots, W^\lambda_{pre}$. By Claims 26 and 28, if an adversary succeeds in the original experiment with certain probability, he must succeed in the conditional experiment with probability smaller by at most $2^{-\Omega(\lambda)}$. It remains, then, to bound the success probability under the above two conditions.

We denote the distributions in the conditional experiment by $\tilde{Z}$, $\tilde{W}_{pre}$ and $\tilde{W}_{post}$. Note that $\tilde{Z}^j_0, \tilde{Z}^j_1$ are independent in all other values in $\tilde{Z}_0$ and $\tilde{Z}_1$ given the snapshots of adversary's state before and after they are determined, namely $\tilde{W}^j_{pre}$ and $\tilde{W}^j_{post}$. Hence, by Proposition 9,

$$\begin{aligned}
H_\infty(\tilde{Z}_0, \tilde{Z}_1 \mid \tilde{W}^\lambda_{post}) &\geq H_\infty(\tilde{Z}_0, \tilde{Z}_1 \mid \tilde{W}^\lambda_{post}, \ldots, \tilde{W}^1_{post}, \tilde{W}^\lambda_{pre}, \ldots, \tilde{W}^1_{pre}) \\
&\geq \sum_{j=1}^{\lambda} H_\infty(\tilde{Z}^j_0, \tilde{Z}^j_1 \mid \tilde{W}^\lambda_{post}, \ldots, \tilde{W}^1_{post}, \tilde{W}^\lambda_{pre}, \ldots, \tilde{W}^1_{pre}) \\
&\geq \sum_{j=1}^{\lambda} H_\infty(\tilde{Z}^j_0, \tilde{Z}^j_1 \mid \tilde{W}^j_{post}, \tilde{W}^j_{pre}) \\
&\geq \min_{\omega} \sum_{j \in J_\omega} H_\infty(\tilde{Z}^j_0, \tilde{Z}^j_1 \mid \tilde{W}^j_{post}, \tilde{W}^j_{pre} = \omega^j) \\
&\geq 0.8(\lambda/2).
\end{aligned}$$

The above implies that the probability of success in the hybrid experiment is at most $2^{-0.4\lambda} = 2^{-\Omega(\lambda)}$ and, consequently, is at most $2^{-\Omega(\lambda)}$ in the original experiment as well. This completes the proof of Lemma 25.

4.5 Quantum Hardness of Finding a Claw

We prove hardness of finding a claw also against a quantum (but memory-bounded) attacker, with a somewhat worse bound. Although this statement is not necessary for the proof of our proof of quantumness theorem (Theorem 18), it enables new applications in the context of verification of quantum computation, which we outline in Appendix A. Before starting with the analysis, let us make the notion of a memory-bounded quantum adversary more precise.

Quantum Adversaries. A memory-bounded quantum adversary is modeled as a quantum channel acting on a fixed-size register $\mathsf{M} \simeq \mathbb{C}^{2^m}$ and on a register N, which corresponds to the next message of the protocol. We can model any memory-bounded quantum adversary without loss of generality as follows:

- The adversary starts with an initial state ρ_{M} in the memory register.
- For each round i of the protocol and each incoming message μ_i, the adversary applies an arbitrary CPTP linear map:

$$\Phi^{i,\mu_i}_{\mathsf{M}\to\mathsf{MN}} : \mathrm{L}(\mathsf{M}) \mapsto \mathrm{L}(\mathsf{M} \otimes \mathsf{N}).$$

- The message sent by the adversary as a response is determined by measuring N in the computational basis.
- The updated state of the attacker is the reduced density matrix on M.

An implication of this fact is that the initial state of the adversary, along with transcript of the protocol, uniquely determine the state of the attacker at any given round.

Analysis. Just like in the proof for classical soundness in Section 4.4, let us denote by Z the random variable containing the 2λ-bit bits $\{z_b^1, \dots, z_b^\lambda\}_{b\in\{0,1\}}$ that are stored by the verifier in Step 1.3., and by W the random variable containing the m-qubit state of the attacker at the end of the protocol. Further, for a set $J \subseteq [\lambda] \times \{0,1\}$, we denote by Z_J the restriction of Z to J.

Lemma 29 (Quantum Hardness of Claw-Finding)**.** *There exists a random variable* $J \subseteq [\lambda] \times \{0,1\}$ *(which depends on* Z *and* W*) such that*

$$\mathbf{TD}\left((J, Z_J, W), (J, Z', W)\right) \leq \lambda^2 \sqrt{m/2k},$$

where Z' *is a uniformly random string of length* $|J|$*, and, further,* $|J| \geq \lambda/2$ *with probability all but* $\exp(-\Omega(\lambda))$ *over the random coins of the protocol.*

Before proving Lemma 29, we first observe that it indeed implies quantum hardness of finding a claw. This is because an adversary that guesses the claw must in particular guess any subset of its bits, including J. However, the success probability of the best attacker on the RHS distribution is at most $2^{-\lambda/2}$. By a triangle inequality, this implies that the success probability in guessing Z is bounded by $2^{-\lambda/2} + \lambda^2\sqrt{m/2k}$.

We now proceed to prove Lemma 29. Similarly as above, for $j \in [\lambda]$, let U^j be the random variable taking the value of the string U streamed in Step 1.2., before the first successful completion of the interactive hashing step, namely the first attempt where the verifier does not abort and start over in Step 1.3.. Let W^j_{pre} denote the random variable consisting of the adversary's internal state, i.e., its m-qubit memory, right after the streaming of U^j has completed. We consider the marginal distributions of U^j and W^j_{pre} given a fixed transcript of the protocol up to (and excluding) the streaming of U^j (recall the transcript, together with the adversary's initial state, determine the state of the adversary and, therefore the aforementioned marginal distributions are well-defined). For any such possible transcript τ, we denote the corresponding marginals by $U^j(\tau)$ and $W^j_{pre}(\tau)$. By Lemma 14, for any τ, we have that:

$$\mathbb{E}_{i\leftarrow[k]} \underbrace{\mathbf{TD}\left((U^j_{<i}(\tau), U^j_i(\tau), W^j_{pre}(\tau)), (U^j_{<i}(\tau), \tilde{U}^j_i, W^j_{pre}(\tau))\right)}_{\delta^j_i(\tau)} \leq \sqrt{m/2k} \tag{8}$$

where $\tilde{U}^j_i$ is a uniformly sampled bit. Let $t = \sqrt{2k/\lambda^2 m}$. For all $j \in [\lambda]$ and any τ define the set:

$$B^j_\tau := \left\{i : \delta^j_i(\tau) > 1/t\right\} \subseteq [k].$$

Using this definition, we define the set J as follows: For any round $j \in [\lambda]$, we let τ^j denote the history of the protocol up to (and excluding) the streaming of U^j. We add (j, b) to J if $b \in \{0,1\}$ is the smallest such that $v^j_b \notin B^j_{\tau^j}$, where v^j_0, v^j_1 are the outcome of the (successful) interactive hashing at round j. To complete the proof of Lemma 29, it suffices the bound the size of J, and the trace distance between the following two experiments:

- The first experiment runs the protocol and outputs the adversary's state W at its completion and the bits in Z_J.
- The second experiment does the same thing, except that it outputs uniformly sampled bits, along with W.

We do so in the following two propositions.

Claim 30 (J is Large)**.** *Let J be defined as above, then:*

$$\Pr(|J| < \lambda/2) < 2^{-\Omega(\lambda)}.$$

Proof. By Equation (8) and Lemma 12 (Markov) we have that:

$$\Pr_{i\leftarrow[k]}\left(\delta^j_i(\tau^j) > 1/t\right) < t/\sqrt{m/2k},$$

which implies that $|B^j_{\tau^j}| < t\sqrt{km/2}$. By Theorem 22, we can bound the probability that the pre-images of the interactive hashing belong to such a set by

$$\Pr\left(\{v^j_0, v^j_1\} \subseteq B^j_{\tau^j}\right) \leq O\left(t \log k \sqrt{m/2k}\right), \tag{9}$$

where the probability is taken over the random coins of the interactive hashing.

Let us denote by E^j the predicate for $\{v_0^j, v_1^j\} \subseteq B_\tau^j$. To prove the proposition, it suffices to show that $E^j = 0$ for at least half of the j's with overwhelming probability. By Eq. (9), we have:

$$\tilde{E} := \mathbb{E}\left(\sum_{j\in[\lambda]} E^j\right) = O\left(\lambda t \log k \sqrt{m/2k}\right) = O(\log k).$$

Further, observe that the random variables E^j are negatively correlated, since interactive hashing soundness holds for any round independently of the others and, therefore, the adversary cannot increase probability that the event E^j happens across several rounds. Thus, by Lemma 13 (Chernoff) we can bound:

$$\Pr\left(\sum_{j\in[\lambda]} E_\tau^j \geq \lambda/2\right) \leq e^{-\Omega(\lambda^2/\log\lambda)}$$

which completes the proof of the claim. □

Claim 31 (Distance of the Experiments)**.** *Let J be defined as above. It holds that:*

$$\mathbf{TD}((J, Z_J, W), (J, Z', W)) \leq \lambda/t$$

where Z' is a uniformly random string of length $|J|$.

Proof. Let us denote $J = \{(j, b_j)\}$ (note for every j there exists at most one element in J for some $b_j \in \{0,1\}$). We bound the distance incurred by each of the swaps. Let $J^{\geq j} = \{(j', b_{j'}) \in J \mid j' \geq j\}$ and let $t_j = |J| - |J^{\geq j}|$. Then, for any $j = 1, \ldots, \lambda$, our goal is to prove

$$\mathbf{TD}((J, Z'_{t_j}, Z_{J^{\geq j}}, W), (J, Z'_{t_j+1}, Z_{J^{\geq j+1}}, W)) \leq 1/t,$$

which implies the claim by triangle inequality. Now, for any $(j, b_j) \in J$, let $v^j = v_{b_j}^j$; this is the location of the swapped bit in U^j. Then, by monotonicity of trace distance, it holds that

$$\begin{aligned}&\mathbf{TD}((J, Z'_{t_j}, Z_{J^{\geq j}}, W), (J, Z'_{t_j+1}, Z_{J^{\geq j+1}}, W))\\ &\quad\leq \mathbf{TD}((J^{\leq j}, Z'_{t_j}, U^j_{<v^j}, U^j_{v^j}, W^j_{pre}), (J^{\leq j}, Z'_{t_j}, U^j_{<v^j}, \tilde{U}^j_{v^j}, W^j_{pre})).\end{aligned}$$

The above holds since the prover's final state W, the final set J and the bits Z_J swapped with random until round $j-1$ (i.e. $Z'_{t_j}, Z_{J^{\geq j}}$) or, respectively, round j (i.e. $Z'_{t_j+1}, Z_{J^{\geq j+1}}$), may be produced from the prover's state after the streaming of U^j, i.e. W^j_{pre}, the choice of J up until the j^{th} round, i.e. $J^{\leq j}$, Z_J swapped up to round $j-1$, i.e. Z'_{t_j}, and the stream U^j up until the v_j^{th} bit which is either swapped (i.e. $\tilde{U}^j_{v^j}$) or not ($U^j_{v^j}$), respectively. To produce the final distributions, simply carry on with the protocol given W^j_{pre} and $J^{\leq j}$ to obtain the final state W and J in full, together with the bits of Z_J at rounds $j+1, \ldots, \lambda$. The bit in Z_J at round j is simulated by $U^j_{v^j}$ or $\tilde{U}^j_{v^j}$ and the bits before round j are given as Z'_{t_j}.

Lastly, since anything that occurs up till round j is a function of the transcript of the protocol up to that round, i.e. τ^j, we may finish as follows

$$\begin{aligned}&\mathbf{TD}((J^{\leq j}, Z'_{t_j}, U^j_{<v^j}, U^j_{v^j}, W^j_{pre}), (J^{\leq j}, Z'_{t_j}, U^j_{<v^j}, \tilde{U}^j_{v^j}, W^j_{pre}))\\ &\quad\leq \mathbb{E}_{\tau^j}\mathbf{TD}((U^j_{<v^j}(\tau^j), U^j_{v^j}(\tau^j), W^j_{pre}(\tau^j)), (U^j_{<v^j}(\tau^j), \tilde{U}^j_{v^j}, W^j_{pre}(\tau^j)))\\ &\quad= \mathbb{E}_{\tau^j}\delta^j_{v^j}(\tau^j) \leq 1/t.\end{aligned}$$

□

References

[AAB+19] Frank Arute, Kunal Arya, Ryan Babbush, Dave Bacon, Joseph C Bardin, Rami Barends, Rupak Biswas, Sergio Boixo, Fernando GSL Brandao, David A Buell, et al. Quantum supremacy using a programmable superconducting processor. *Nature*, 574(7779):505–510, 2019. 3

[AABA+24] Rajeev Acharya, Laleh Aghababaie-Beni, Igor Aleiner, Trond I Andersen, Markus Ansmann, Frank Arute, Kunal Arya, Abraham Asfaw, Nikita Astrakhantsev, Juan Atalaya, et al. Quantum error correction below the surface code threshold. *arXiv preprint arXiv:2408.13687*, 2024. 3

[AMR22] Navid Alamati, Giulio Malavolta, and Ahmadreza Rahimi. Candidate trapdoor claw-free functions from group actions with applications to quantum protocols. In Eike Kiltz and Vinod Vaikuntanathan, editors, *TCC 2022: 20th Theory of Cryptography Conference, Part I*, volume 13747 of *Lecture Notes in Computer Science*, pages 266–293, Chicago, IL, USA, November 7–10, 2022. Springer, Cham, Switzerland. 3

[Bar89] David A. Barrington. Bounded-width polynomial-size branching programs recognize exactly those languages in nc1. *Journal of Computer and System Sciences*, 38(1):150–164, 1989. 10

[BBK22] Nir Bitansky, Zvika Brakerski, and Yael Tauman Kalai. Constructive post-quantum reductions. In Yevgeniy Dodis and Thomas Shrimpton, editors, *Advances in Cryptology - CRYPTO 2022 - 42nd Annual International Cryptology Conference, CRYPTO 2022, Santa Barbara, CA, USA, August 15-18, 2022, Proceedings, Part III*, volume 13509 of *Lecture Notes in Computer Science*, pages 654–683. Springer, 2022. 8, 12

[BCM+18] Zvika Brakerski, Paul Christiano, Urmila Mahadev, Umesh V. Vazirani, and Thomas Vidick. A cryptographic test of quantumness and certifiable randomness from a single quantum device. In Mikkel Thorup, editor, *59th Annual Symposium on Foundations of Computer Science*, pages 320–331, Paris, France, October 7–9, 2018. IEEE Computer Society Press. 3

[BCM$^+$21] Zvika Brakerski, Paul Christiano, Urmila Mahadev, Umesh Vazirani, and Thomas Vidick. A cryptographic test of quantumness and certifiable randomness from a single quantum device. *J. ACM*, 68(5), August 2021. 4, 5, 13, 18

[BGK$^+$23] Zvika Brakerski, Alexandru Gheorghiu, Gregory D. Kahanamoku-Meyer, Eitan Porat, and Thomas Vidick. Simple tests of quantumness also certify qubits. In Helena Handschuh and Anna Lysyanskaya, editors, *Advances in Cryptology - CRYPTO 2023 - 43rd Annual International Cryptology Conference, CRYPTO 2023, Santa Barbara, CA, USA, August 20-24, 2023, Proceedings, Part V*, volume 14085 of *Lecture Notes in Computer Science*, pages 162–191. Springer, 2023. 4, 5, 13, 18, 19, 20

[BK24] James Bartusek and Dakshita Khurana. On the power of oblivious state preparation. *CoRR*, abs/2411.04234, 2024. 4, 5, 36, 37, 38, 39

[BKM$^+$24] Kaniuar Bacho, Alexander Kulpe, Giulio Malavolta, Simon Schmidt, and Michael Walter. Compiled nonlocal games from any trapdoor claw-free function. Cryptology ePrint Archive, Paper 2024/1829, 2024. 4, 5, 36

[BKW97] Johannes Blömer, Richard Karp, and Emo Welzl. The rank of sparse random matrices over finite fields. *Random Struct. Algorithms*, 10(4):407–419, July 1997. 9

[CCM98] C. Cachin, C. Crepeau, and J. Marcil. Oblivious transfer with a memory-bounded receiver. In *Proceedings 39th Annual Symposium on Foundations of Computer Science (Cat. No.98CB36280)*, pages 493–502, 1998. 7

[CM97] Christian Cachin and Ueli M. Maurer. Unconditional security against memory-bounded adversaries. In Burton S. Kaliski Jr., editor, *Advances in Cryptology - CRYPTO '97, 17th Annual International Cryptology Conference, Santa Barbara, California, USA, August 17-21, 1997, Proceedings*, volume 1294 of *Lecture Notes in Computer Science*, pages 292–306. Springer, 1997. 4

[DHRS04] Yan Zong Ding, Danny Harnik, Alon Rosen, and Ronen Shaltiel. Constant-round oblivious transfer in the bounded storage model. In Moni Naor, editor, *Theory of Cryptography, First Theory of Cryptography Conference, TCC 2004, Cambridge, MA, USA, February 19-21, 2004, Proceedings*, volume 2951 of *Lecture Notes in Computer Science*, pages 446–472. Springer, 2004. 4, 7, 22

[Din01] Yan Zong Ding. Oblivious transfer in the bounded storage model. In Joe Kilian, editor, *Advances in Cryptology — CRYPTO 2001*, pages 155–170, Berlin, Heidelberg, 2001. Springer Berlin Heidelberg. 4, 7

[DORS06] Yevgeniy Dodis, Rafail Ostrovsky, Leonid Reyzin, and Adam D. Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. *CoRR*, abs/cs/0602007, 2006. 11

[DQW23] Yevgeniy Dodis, Willy Quach, and Daniel Wichs. Speak much, remember little: Cryptography in the bounded storage model, revisited. In Carmit Hazay and Martijn Stam, editors, *Advances in Cryptology - EUROCRYPT 2023 - 42nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Lyon, France, April 23-27, 2023, Proceedings, Part I*, volume 14004 of *Lecture Notes in Computer Science*, pages 86–116. Springer, 2023. 4, 7, 8, 11

[GL89] O. Goldreich and L. A. Levin. A hard-core predicate for all one-way functions. In *Proceedings of the Twenty-First Annual ACM Symposium on Theory of Computing*, STOC '89, page 25–32, New York, NY, USA, 1989. Association for Computing Machinery. 10

[GZ19] Jiaxin Guan and Mark Zhandry. Simple schemes in the bounded storage model. In Yuval Ishai and Vincent Rijmen, editors, *Advances in Cryptology – EUROCRYPT 2019, Part III*, volume 11478 of *Lecture Notes in Computer Science*, pages 500–524, Darmstadt, Germany, May 19–23, 2019. Springer, Cham, Switzerland. 4, 5

[KCVY22] Gregory Kahanamoku-Meyer, Soonwon Choi, Umesh Vazirani, and Norman Yao. Classically verifiable quantum advantage from a computational bell test. *Nature Physics*, 18:1–7, 08 2022. 5, 7, 13, 18

[KLVY23a] Yael Kalai, Alex Lombardi, Vinod Vaikuntanathan, and Lisa Yang. Quantum advantage from any non-local game. In Barna Saha and Rocco A. Servedio, editors, *55th Annual ACM Symposium on Theory of Computing*, pages 1617–1628, Orlando, FL, USA, June 20–23, 2023. ACM Press. 3

[KLVY23b] Yael Kalai, Alex Lombardi, Vinod Vaikuntanathan, and Lisa Yang. Quantum advantage from any non-local game. In Barna Saha and Rocco A. Servedio, editors, *Proceedings of the 55th Annual ACM Symposium on Theory of Computing, STOC 2023, Orlando, FL, USA, June 20-23, 2023*, pages 1617–1628. ACM, 2023. 5, 18, 36, 38

[KMCVY22] Gregory D Kahanamoku-Meyer, Soonwon Choi, Umesh V Vazirani, and Norman Y Yao. Classically verifiable quantum advantage from a computational bell test. *Nature Physics*, 18(8):918–924, 2022. 3

[LLL$^+$21] Yong Liu, Xin Liu, Fang Li, Haohuan Fu, Yuling Yang, Jiawei Song, Pengpeng Zhao, Zhen Wang, Dajia Peng, Huarong Chen, et al. Closing the" quantum supremacy" gap: achieving real-time simulation of a random quantum circuit using a new sunway supercomputer. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 1–12, 2021. 3

[LZG$^+$24] Laura Lewis, Daiwei Zhu, Alexandru Gheorghiu, Crystal Noel, Or Katz, Bahaa Harraz, Qingfeng Wang, Andrew Risinger, Lei Feng, Debopriyo Biswas, et al.

Experimental implementation of an efficient test of quantumness. *Physical Review A*, 109(1):012610, 2024. 3

[Mah18a] Urmila Mahadev. Classical homomorphic encryption for quantum circuits. In *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 332–338, 2018. 5

[Mah18b] Urmila Mahadev. Classical verification of quantum computations. In *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 259–267. IEEE, 2018. 4, 5

[Mau92] Ueli M. Maurer. Conditionally-perfect secrecy and a provably-secure randomized cipher. *J. Cryptol.*, 5(1):53–66, 1992. 4

[MSY24] Tomoyuki Morimae, Yuki Shirakawa, and Takashi Yamakawa. Cryptographic characterization of quantum advantage. Cryptology ePrint Archive, Paper 2024/1536, 2024. 3

[MY23] Tomoyuki Morimae and Takashi Yamakawa. Proofs of quantumness from trapdoor permutations. In Yael Tauman Kalai, editor, *14th Innovations in Theoretical Computer Science Conference, ITCS 2023, January 10-13, 2023, MIT, Cambridge, Massachusetts, USA*, volume 251 of *LIPIcs*, pages 87:1–87:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2023. 7, 22

[NOVY92] Moni Naor, Rafail Ostrovsky, Ramarathnam Venkatesan, and Moti Yung. Perfect zero-knowledge arguments for NP can be based on general complexity assumptions (extended abstract). In Ernest F. Brickell, editor, *Advances in Cryptology - CRYPTO '92, 12th Annual International Cryptology Conference, Santa Barbara, California, USA, August 16-20, 1992, Proceedings*, volume 740 of *Lecture Notes in Computer Science*, pages 196–214. Springer, 1992. 7, 18, 21, 22

[NZ23] Anand Natarajan and Tina Zhang. Bounding the quantum value of compiled nonlocal games: From CHSH to BQP verification. In *64th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2023, Santa Cruz, CA, USA, November 6-9, 2023*, pages 1342–1348. IEEE, 2023. 4, 5, 36, 38

[PS97] Alessandro Panconesi and Aravind Srinivasan. Randomized distributed edge coloring via an extension of the chernoff–hoeffding bounds. *SIAM Journal on Computing*, 26(2):350–368, 1997. 12

[Raz18] Ran Raz. Fast learning requires good memory: A time-space lower bound for parity learning. *J. ACM*, 66(1), December 2018. 4, 5, 9, 12, 13

[Sho94] Peter W. Shor. Algorithms for quantum computation: Discrete logarithms and factoring. In *35th Annual Symposium on Foundations of Computer Science*, pages 124–134, Santa Fe, NM, USA, November 20–22, 1994. IEEE Computer Society Press. 3

[Top01] Flemming Topsøe. Bounds for entropy and divergence for distributions over a two-element set. *JIPAM. Journal of Inequalities in Pure & Applied Mathematics [electronic only]*, 2(2):Paper No. 25, 13 p.–Paper No. 25, 13 p., 2001. 11

[Vad04] Salil P. Vadhan. Constructing locally computable extractors and cryptosystems in the bounded-storage model. *Journal of Cryptology*, 17(1):43–77, January 2004. 4

A Classical Verification of BQP

It is shown in [NZ23, BK24, BKM$^+$24] that a claw-generation protocol with quantum soundness implies the existence of a protocol where a quantum prover can demonstrate to a completely classical verifier the validity of any statement in BQP. The protocol roughly goes as follows: First, it turns the hardness of finding a claw into a protocol for *blind quantum computation*, i.e., a protocol where the circuit that is computed by the quantum prover is computationally hidden, and the verifier is fully classical. Second, it uses blind quantum computation to *compile* a two-player non-local game for BQP verification [NZ23, KLVY23b] into a single-player one.

By plugging our claw-generation protocol into the framework of [BK24], one can obtain a protocol for general verification of BQP computation against *memory-bounded* quantum adversaries, with *unconditional* security. However, the straightforward combination of Lemma 24 with the [BK24] theorem meets two main discrepancies.

First, the reductions as stated in [BK24] are not concerned with preserving the memory complexity of the attacker, which on the other hand is necessary in our setting. In Appendix A.1, we outline how to adapt all reductions involved to be “memory preserving”.

Second, the analysis from [BK24] requires a claw-generation protocol where advantage of claw-finding is negligible, whereas Lemma 29 only provides an inverse-polynomial bound on the success probability of the attacker. We show how to bridge this gap in Appendix A.2.

A.1 From Claw-Generation to Verification of BQP

We outline how to construct a classical verification for BQP computations, unconditionally secure against memory-bounded adversaries. In what follows, we assume that we have a claw-generation protocol with negligible soundness error (the success probability of any memory-bounded attacker) and obtain a classical verification protocol for BQP computations with inverse-polynomial gap between completeness (the success probability of an honest prover) and soundness error.

In a nutshell, we follow the strategy from [BK24] where an analogous implication is shown in the standard cryptographic setting (against computationally-bounded attackers). In fact, all the steps are precisely identical, except that we need to argue why security holds against memory-bounded adversaries. Given that the protocol and the arguments are unchanged, we only provide a proof sketch.

Step I: Oblivious State Preparation. An oblivious state preparation (OSP) [BK24] is a protocol between a classical verifier and a quantum prover. At the end of the interaction, the prover holds the state

$$H^\theta |b\rangle$$

whereas the verifier holds the bits (θ, b). Security requires that any QPT prover cannot guess the bit of the verifier θ with probability non-negligibly greater than $1/2$. It is shown in [BK24] (Theorem 4.7) that a claw-state generation protocol can be generically used to construct an OSP. First it is shown that a claw-state generation protocol can always be assumed without loss of generality (Lemma 4.6 in [BK24]) to prepare a state of the form

$$\frac{|0, x_0\rangle + |1, x_1\rangle}{\sqrt{2}}$$

for $x_0, x_1 \in \{0,1\}^n$. Then the prover and the verifier engage in the following interactive protocol:

- The verifier samples two random bitstrings $r_0, r_1 \leftarrow \{0,1\}^n$.
- The prover applies the map
$$\frac{|0, x_0\rangle + |1, x_1\rangle}{\sqrt{2}} \mapsto \frac{|0, x_0, x_0^\intercal r_0\rangle + |1, x_1, x_1^\intercal r_1\rangle}{\sqrt{2}}$$
and measures all but the last qubit in the Hadamard basis to obtain a string $d \in \{0,1\}^{n+1}$.
- The verifier sets $\theta = (x_0, x_1)^\intercal (r_0, r_1)$. If $\theta = 0$, then it sets $b = x_0^\intercal r_0 = x_1^\intercal r_1$. Otherwise, it sets $b = d^\intercal (1, x_0 \oplus x_1)$.

Correctness follows by direct calculation, whereas security follows by reducing the hardness of guessing θ to the hardness of computing (x_0, x_1), with the Goldreich-Levin search-to-decision reduction. Using Lemma 6, the same reduction holds in the memory-bounded settings.

Step II: Blind Delegation. A blind delegation protocol allows a verifier to delegate the computation of a quantum circuit Q on a classical input $|y\rangle$, while keeping y hidden. It is shown in [BK24] (Theorem 6.11) that OSP implies a blind delegation protocol. The protocol starts with a one-time padded input $X^x |y\rangle$ and proceeds while the verifier keeps track of the one-time pad keys and the prover performs the quantum gates. For a Clifford gates C, the following identity is used:

$$CX^x Z^z = X^{x'} Z^{z'} C$$

where x', z' are functions of x, z. For non-Clifford gates, [BK24] (Theorem 6.10) shows an interactive protocol to correct the errors, based on OSP. This subroutine, referred to as *encrypted phase*, allows a prover to obliviously apply a phase, conditioned on a bit that is only known to the verifier. The protocol proceeds as follows:

- The verifier and the prover engage in an OSP protocol, with the verifier's bit (θ, b). The prover applies a Hadamard gate H and a $\sqrt{X}$ gate to its state. This results into the state
$$Z^b P^\theta \left|+\right\rangle .$$
- The prover CNOTs the target register onto the above state, and measures it in the computational basis to obtain a bit $m \in \{0, 1\}$.
- The prover sents m to the verifier, who applies the appropriate correction to the one-time pad.

We omit most details from this outline, but what is important for us is that the blindness follows directly from the indistinguishability property of the OSP. Thus, precisely the same analysis holds in the memory-bounded settings.

Step III: Computationally Non-Local Strategies. The last step in the strategy is to use the blind-delegation protocol to *compile* a two-prover non-local game into a single prover one. This idea was first proposed in [KLVY23b] and it soundness was analyzed in [NZ23], for the case of a particular blind-delegation protocol, based on quantum fully-homomorphic encryption. In [BK24] (Theorem 6.15) this approach is extended to *any* blind delegation protocol, and it shown that the compilation results into a single-prover *computationally non-local strategy*. Blindness of the delegation protocol is only used to establish that the maximal success probability of any QPT prover in the compiled protocol is bounded by the supremum over all computationally non-local strategies (Theorem 6.23). This is again a direct reduction to the blind delegation protocol that also works in the memory-bounded settings (provided of course that the prover is memory bounded). All subsequent steps of the analysis are information-theoretic and thus trivially hold for our setting as well.

A.2 Tolerating Non-Negligible Soundness Error

Next, we show that our claw-generation protocol (Lemma 24) implies classical verification of BQP computation, despite its inverse-polynomial soundness error against quantum attackers, thus relaxing the assumption of negligible soundness error made in Appendix A.1.

One simple way to derive this is by relying on the fact that our quantum soundness argument from Lemma 29 stems from an indistinguishability argument between the real experiment and an ideal experiment, where soundness holds *statistically*.

Let us recall some notation from the proof of Lemma 29: Z is the random variable containing the "hard" part of the claw consisting of bits $\{z_b^1, \ldots, z_b^\lambda\}_{b \in \{0,1\}}$ and W is the adversary's m-qubit state at the end of the protocol. Lemma 29 bounds that trace-distance between (J, Z_J, W) and (J, Z', W), where J is a subset of size at least $\lambda/2$ and Z' is uniformly random, by $\varepsilon = \lambda^2\sqrt{m/2k}$. Since the polynomial k can be chosen to be arbitrarily larger than the bound on memory m, we can choose ε to be an arbitrarily small inverse-polynomial function of λ. Specifically, let δ denote the inverse-polynomial gap between completeness and soundness error in the protocol from [BK24] when instantiated using claw-generation with negligible soundness error, which we inherit in the bounded-memory setting as shown in Appendix A.1. Let $T = \mathrm{poly}(\lambda)$ denote the number of times claw-generation is invoked in the protocol to prove a given BQP instance. We set ε to be small enough so that $T \cdot \varepsilon < \delta/2$. We argue that, when the protocol is instantiated using our claw-generation from Lemma 24, we obtain gap between completeness and soundness error that is at most $\delta/2$.

To see this, consider an ideal experiment where after any invocation of claw-generation throughout the protocol, the verifier modifies the claw values in its memory by replacing the bits at locations J in Z with freshly sampled uniformly random bits. By Lemma 29, any such replacement deviates us from the real experiment by at most ε in trace-distance. In total, we obtain an experiment that is at most $\delta/2$-far from the real experiment by triangle inequality.

In the ideal experiment, claw-generation provides soundness up to error $2^{-\lambda/2}$, which is negligible in the security parameter. Following the adaption of [BK24] to our setting (Appendix A.1), this implies that an attacker against the protocol in the experiment succeeds with soundness error that is smaller than completeness by at least δ. Therefore, this gap in the real experiment is at least $\delta/2$ by triangle inequality.