An exposition on quantum interactive proofs with a classical verifier

Sriram Balasubramanian
University of Maryland, College Park

December 7, 2022

Abstract

Quantum interactive proofs are the quantum analogue of interactive proofs, in which a powerful but untrusted prover attempts to convince a less powerful verifier of the correctness of its computation. In quantum interactive proofs, we consider the case when either (or both) the prover and verifier have quantum capabilities. In practice, we are concerned with the case where the prover is a realistic quantum computer (BQP machine) trying to efficiently convince a classical verifier (a BPP machine), which corresponds to the question: Can quantum computers (efficiently) convince a classical verifier that its computations are correct over a polynomial number of interactions? Recently, this question was settled in the affirmative [3], with the assumption that the BQP prover cannot efficiently solve the learning with errors problem (LWE). In this expository paper, I go over some key results in this field of quantum interactive proofs, with a special focus on the landmark paper [3] by Urmila Mahadev.

1 Introduction

The field of quantum computing has been growing rapidly, with many theoretical and practical advancements in recent years. It is widely believed that quantum computers have a significant edge over their classical counterparts, and can solve efficiently problems like integer factoring and quantum simulation in polynomial time. However, since we are dealing with a new more powerful class of computers, it is natural to ask the question: How can an observer with access to a regular classical computer (the verifier) be convinced that the powerful quantum computer (the prover) is indeed performing the computations correctly? If this is not possible, it would be difficult for a quantum computer to gain trust that it is indeed capable of powerful computations. This would be easy if the powerful computer was an NP machine, we could just ask the prover to encode the NP problem in 3-SAT and get the satisfying assignment as a certificate which the verifier can verify. However, it is not known that BQP $\subseteq$ NP, and it is widely assumed that this is not the case. In fact, in the presence of access to an oracle, BQP can be separated from PH [9]. Another problem is that unlike a classical computer, we can't 'look' into the intermediate computations of a quantum computer because of measurement collapse of the quantum state. Thus, we need to find a verification method which can work when the prover can solve BQP problems efficiently.

One possible line of attack proposed in [5] is to construct an interactive protocol between the prover and the verifier capable of convincing the verifier with a polynomial number of interactions. Typically, repeated interactions are more powerful than a one-time interaction as this enables us to use probabilistic guarantees which can be pumped up by repeated interactions. Thus, [2] introduced a complexity class $QPIP$ (quantum prover interactive proofs). In this class, we have languages that can be verified in the following setup:

1. A quantum *prover*, which can solve BQP problems in polynomial time. The prover also has access to the quantum communication channel which can transmit κ bits at a time.
2. A hybrid quantum-classical *verifier*, which consists of a classical BPP machine and a quantum κ-bit register on which the BPP machine can perform quantum measurements.
3. A classical communication channel which can transmit polynomial number of bits at a time.

A language can be said to be in QPIP_κ (where κ is the number of qubits in the register) if (a) for any YES instance the verifier accepts a proof from the prover, with a probability greater than c, that the instance indeed belongs to the language, and (b) for any NO instance, the verifier rejects any proof from the prover with probability greater than $1 - s$. c and s are constants which parametrize the language QPIP. In [2] and [6], it was proved that for some constants $c, s, \kappa \geq 0$, $\text{QPIP}_\kappa = \text{BPP}$.

However, there are still some caveats in this result that we need to keep in mind. The verifier still has some quantum powers (2) and the prover can send quantum states through a quantum communication channel (1). Ideally, we would want to relax these conditions, as we would like to not rely on any quantum powers at all to verify the quantum computer. That is, we want to show that $\text{QPIP}_0 = \text{BPP}$.

Many papers tried to relax assumptions on the quantum verifier [4, 11] culminating with [3] which showed a measurement protocol which did not require the verifier to have quantum powers. This protocol is based on previous work by [4], which managed to weaken the verifier considerably by restricting it to performing measurements only in the standard or Hadamard basis. Using a cryptographic primitive considered to be hard to crack for quantum computers, we are able to devise a protocol to force the quantum prover to perform the measurements by itself and report it to the classical verifier.

2 Preliminaries

Before proceeding to the main proof, we need to introduce some crucial machinery and prerequisites. The first piece is a cryptographic primitive for which the verifier has a key but the prover does not, and which is believed to be hard for quantum computers to break based on the hardness of learning with errors (LWE). The second piece is a work by [4], which showed that the verifier could be restricted to performing only Hadamard or standard basis measurements on its register. The protocol in [3] acts as an intermediate interface between the prover and the verifier in [4]

2.1 Cryptographic primitives

We first define a family of cryptographic primitives called *trapdoor claw-free functions*, or TCF functions. These are two-to-one functions which are easy to invert given a secret trapdoor, but hard otherwise. More formally, for a function f, it is hard to find x_0, x_1 for a given y such that $f(x_0) = f(x_1) = y$, but easy given a trapdoor t. Equivalently, we can assume that we have a pair of one-to-one functions f_0, f_1 such that they have the same image (For all x_0, there exists x_1 such that $y = f_0(x_0) = f_1(x_1)$). These functions are hard to invert given y normally but easy given a trapdoor. In [7], it is shown how to construct such a family based on the hardness of learning with errors problem. It is believed that quantum computers cannot solve the learning with error problem efficiently.

Note 1

Learning with errors: This problem was first introduced by [8], and is a finite ring analogue of the linear regression problem. Formally, consider an unknown linear function $f : \mathbb{Z}_q^n \to \mathbb{Z}_q$ of the form $f(\mathbf{x}) = \mathbf{m} \cdot \mathbf{x} + c$ and $(\mathbf{x}, y)$, where $\mathbf{x} \in \mathbb{Z}_q^n$ and $y \in \mathbb{Z}_q$, such that $y = f(\mathbf{x}) + e$ where $e \in \mathbb{Z}_q$ is drawn from a known noise distribution. Then, the problem is to estimate $\mathbf{m}$ from the data.

Given a TCF $f : A \to B$, we can construct a quantum gate U_f which acts on $\frac{1}{\sqrt{|A|}} \sum_{x \in A} |x\rangle|0\rangle$ to give $\frac{1}{\sqrt{|A|}} \sum_{x \in A} |x\rangle|f(x)\rangle$. Measuring the second register, we get $y \in B$ and we are left with $\frac{1}{\sqrt{2}}(|x_0\rangle + |x_1\rangle)$ in the first register such that $f(x_0) = f(x_1) = y$. Thus, a quantum computer is able to obtain a superposition of x_0 and x_1, but not both at the same time. We can further apply a Hadamard transform on each of the $N = \log_2 |A|$ qubits in the first register, and thus obtain $\frac{1}{\sqrt{2}}(H_N|x_0\rangle + H_N|x_1\rangle) = \frac{1}{\sqrt{2}^{N+1}}(\sum_{d \in \{0,1\}^N}((-1)^{d \cdot x_0} + (-1)^{d \cdot x_1})|d\rangle)$. The coefficients of $|d\rangle$ are non-zero only if $d \cdot x_0$ and $d \cdot x_1$ have the same parity, or $d \cdot (x_0 + x_1) = 0$. Thus, the quantum computer is able to sample d efficiently so that $d \cdot (x_0 + x_1) = 0$, unlike a classical computer. However, note that this still doesn't mean that the quantum computer can hold any two of x_0, x_1, and d simultaneously in an efficient manner. It was proven that doing so is as hard as solving the LWE problem in [7], if f is TCF. The proof is too involved for this paper, but can be found in [7].

We now describe the cryptographic primitives we need more formally. Consider a trapdoor claw-free function family $\mathcal{F} = \{f_{k,b} : \mathcal{X} \to \mathcal{Y}\}$ $(b \in \{0, 1\})$, such that it is computationally difficult to find a claw (x_0, x_1) where $f_{k,0}(x_0) = f_{k,1}(x_1) = y$, but it is easy to invert these functions and find a claw when given access to a trapdoor t_k. We can construct such a family with two additional *hardcore bit* properties:

1. **First hardcore bit property** : If $d \neq 0$, then for all claws (x_0, x_1), it is hard to compute both $d \cdot (x_0 + x_1)$ and either x_0 or x_1.
2. **Second hardcore bit property** : For each function pair $f_{k,0}, f_{k,1}$, there exists a string d such that for all claws (x_0, x_1), the bit $d \cdot (x_0 + x_1) = c_k$ is the same (depends only on k). Furthermore, it is computationally hard to determine c_k given $f_{k,0}$ and $f_{k,1}$.

Note 2

Construction of function family: Because it is hard to construct such a family with all the desired properties, we actually have to consider the noisy versions of the above families, where the output of function is not a single y but a distribution over y with some small support. Formally, it is a function class $\mathcal{F} = \{f_{k,b} : \mathcal{X} \to \mathcal{D}_{\mathcal{Y}}\}$, where $\mathcal{D}_{\mathcal{Y}}$ refers to the set of distributions over $\mathcal{Y}$. Detailed construction of these families can be found in [3, 7, 10]. We will now assume that these families exist and are constructible.

The hardcore bit properties for TCF functions are important for proving the *soundness* of the protocol and to ensure that that a BQP prover cannot cheat. We also require another family of functions related to the TCF functions known as trapdoor injective functions. This is defined as $\mathcal{G} = \{g_{k,b} : \mathcal{X} \to \mathcal{Y}\}$ where $(b \in \{0, 1\})$. It is computationally difficult to invert $y = g_{k,b}(x)$ given y, but easy when given access to a trapdoor t_k. Also, since it is injective, if $(b, x_0) \neq (b', x_1)$, then $g_{k,b}(x_0) \neq g_{k,b}(x_1)$. It is possible to construct $\mathcal{G}$ such that it is computationally difficult to distinguish it from $\mathcal{F}$.

2.2 Interactive proofs when the verifier can perform restricted quantum measurements

In [3], Mahadev shows how to reduce the class of languages which can be verified by performing only X *or* Z *measurements* on a quantum state given by the BQP prover to the class of languages which can be verified by a classically with only classical access to the quantum prover. This first class of languages is known to contain BQP (proven in [4]). We now give a sketch of this proof below.

Consider the 2-local Hamiltonian ground state problem which is QMA-complete, that is, there exists a 2-local Hamiltonian for which there exists a low energy ground state iff the verifying circuit accepts some witness state from Merlin. For BQP problems (which are also QMA problems), the verifier is just the circuit which solves the problem by itself and ignores the witness state returned by the prover. Thus, the witness state is not needed and can be assumed to be a trivial state. Using the Feynman-Kitaev clock construction, we can encode the history of the BQP circuit computation (with the language instance as only input) into a quantum state ρ, and this state ρ can be computed efficiently by a BQP prover. To compute ρ, run the verifying circuit on the trivial state for n steps to get the state $|\psi\rangle_n$ and tensor product it with $|n\rangle$ to get $|\psi\rangle_n \otimes |n\rangle$. Repeat for all n from 1 to L where L is the length of the circuit, and add all of them to get ρ. This can be done efficiently since L is only polynomial in input length. This state ρ is then handed to the verifier.

Now, it is known that the 2-local Hamiltonian can be written as a sum of tensor products with only X and Z operators. Thus, we have the total Hamiltonian $H = \sum_S d_S S$ where d_S is a real number (scale factor) and S is the tensor product of some combination of I, X and Z with only two operators not I. The verifier picks an S randomly with probability proportional to $|d_S|$ and measures two qubits in ρ with respect to S using only standard or Hadamard basis measurements to get s_1 and s_2. The verifier accepts if $(-1)^{s_1+s_2} = -\text{sign}(d_S)$. This procedure was proven to have a significant acceptance probability in [11] and is thus able to verify if the ground state energy is $\leq a$ or $\geq b$. The prover cannot cheat the verifier repeatedly for the same reason it cannot cheat in the QMA version of the problem – the Hamiltonian includes all ways a cheater may try to cheat using penalty terms.

3 Measurement protocol

We now relax the assumption that the verifier has a quantum channel and can perform quantum measurements. Now, the prover has to perform the measurements according to the verifier's instructions and report the results to the verifier classically. The verifier also needs to be convinced that the prover is faithfully performing its measurements. We now give an outline of the protocol below. It gives a method to reliably perform Hadamard and standard basis measurements on the history state (for the 2-local Hamiltonian ground state problem) held by the prover. Once we have this, we can use the verification procedure in [4] to verify any language in BQP.

3.1 Obtaining Hadamard basis measurements and testing the prover

1. The verifier samples a function pair f_0, f_1 and a trapdoor from the TCF family $\mathcal{F}$ and sends the functions to the prover, while keeping the trapdoor secret.

2. **Commitment round**: The prover entangles his ground state (history state for the 2-local Hamiltonian ground state problem) $|\psi\rangle = \alpha_0|0\rangle + \alpha_1|1\rangle$ with x_0 and x_1 and then partially measures it as follows:

$$|\psi\rangle = \sum_{b\in\{0,1\}} \alpha_b|b\rangle \to \sum_{b\in\{0,1\},x} \alpha_b|b\rangle|x\rangle|f_b(x)\rangle \xrightarrow{\text{Measure } y=f_b(x)} \sum_{b\in\{0,1\}} \alpha_b|b\rangle|x_b\rangle$$

   (For simplicity, the normalization factor is omitted and can be inferred from the coefficients.) Here, $f(x_0) = f(x_1) = y$. The prover then tells the verifier the measured y. We call y the *commitment string* and $|b\rangle$ the commitment bit. Since the prover "commits" to a particular superposition of claws entangled with the history state, this is called a commitment round.

3. The verifier then uses its trapdoor and y to get x_0 and x_1, and then can proceed to either 'test' the prover, or 'delegate Hadamard measurements' to the prover. The verifier chooses this at random.

4. **Test round**: If the verifier chooses to 'test', then it asks for a pre-image (b, x_b) of y and verifies if $f_b(x_b) = y$. If not, the verifier rejects the prover. If the prover passes the test round, then at some point in the prover's history, it held a superposition of pre-images of the form $\sum |b\rangle|x_b\rangle|y\rangle$, or just a single pre-image $|b\rangle|x_b\rangle|y\rangle$. This helps to ensure that the prover does not deviate from intended behaviour.

5. **Hadamard round**: If the verifier chooses to 'delegate Hadamard measurements', it asks for a Hadamard basis measurement of the ground (history) state $|\psi\rangle$. To measure the ground state in the Hadamard basis, the prover needs to apply a Hadamard transform on $|\psi\rangle = \alpha_b|b\rangle$, but it only has $\sum_b \alpha_b|b\rangle|x_b\rangle$. Because of entanglement with $|x_0\rangle$ and $|x_1\rangle$, it is not possible to directly measure the state $|\psi\rangle$ in the Hadamard basis. Therefore, the prover applies the Hadamard transform on the entire state (normalization factors omitted):

$$\begin{aligned} H\sum_b \alpha_b|b\rangle|x_b\rangle &= \sum_b \alpha_b H|b\rangle \sum_t (-1)^{x_b\cdot t}|t\rangle \\ &= \alpha_0(|0\rangle + |1\rangle)\sum_t (-1)^{x_0\cdot t}|t\rangle + \alpha_1(|0\rangle - |1\rangle)\sum_t (-1)^{x_1\cdot t}|t\rangle \\ &= \sum_t (-1)^{x_0\cdot t}(\alpha_0(|0\rangle + |1\rangle) + \alpha_1(|0\rangle - |1\rangle)(-1)^{(x_0\oplus x_1)\cdot t})|t\rangle \end{aligned}$$

   The prover then measures $|t\rangle$ to obtain d. The corresponding state will then be (ignoring global phase):

$$\begin{aligned} \alpha_0(|0\rangle + |1\rangle) + \alpha_1(|0\rangle - |1\rangle)(-1)^{(x_0\oplus x_1)\cdot d} &= \begin{cases} (\alpha_0 + \alpha_1)|0\rangle + (\alpha_0 - \alpha_1)|1\rangle & \text{if } (x_0 \oplus x_1)\cdot d = 0 \\ (\alpha_0 + \alpha_1)|1\rangle + (\alpha_0 - \alpha_1)|0\rangle & \text{if } (x_0 \oplus x_1)\cdot d = 1 \end{cases} \\ &= X^{(x_0\oplus x_1)\cdot d} H|\psi\rangle \end{aligned}$$

The prover then measures this state and sends the result h along with d to the verifier. The verifier already has y, so it can get the claw (x_0, x_1) using the trapdoor and compute $d \cdot (x_0 \oplus x_1)$ using the trapdoor. It can then flip the bit if $d \cdot (x_0 \oplus x_1) = 1$ to recover back the correct measurement. The probabilities are $P(a) = \frac{1}{2}|\alpha_0 + (-1)^a \alpha_1|^2$ which is identical to the Hadamard basis measurement probabilities of $|\psi\rangle$

3.2 Obtaining standard basis measurements

1. The verifier samples a function pair g_0, g_1 and a trapdoor from the trapdoor injective function family and sends the functions to the prover. Crucially, the prover cannot efficiently distinguish between g_0, g_1 and f_0, f_1.

2. **Commitment round**: The prover entangles his ground state (history state for the 2-local Hamiltonian problem)$|\psi\rangle = \alpha_0|0\rangle + \alpha_1|1\rangle$ with x_0 and x_1 as follows:

   $$|\psi\rangle = \sum_b \alpha_b |b\rangle \to \sum_{b,x} \alpha_b |b\rangle |x\rangle |g_b(x)\rangle \xrightarrow{\text{Measure } y = g_b(x)} |b\rangle |x_b\rangle$$

   Since $g_b(x)$ is injective, there is no superposition over states possible and there exists a unique pre-image x_b for each y. The prover has actually performed a standard basis measurement on $|\psi\rangle$ with $P(b) = |\alpha_b|^2$. The prover then tells the verifier the measured y, who can then invert g to obtain (b, x_b), where b is the standard basis measurement.

3. The verifier goes through the motions as in the previous part, but it has no use for the prover's further measurements as the state has collapsed.

Thus, if the verifier measures the qubit in state $|\psi\rangle = \alpha_0|0\rangle + \alpha_1|1\rangle$ in the computational basis, the probabilities are $P(a) = |\alpha_a|^2$.

4 Proof of soundness

The goal of the measurement protocol is to get the prover to perform the measurements in the Hadamard/standard basis by itself and then report the measurements to the verifier. However, since the prover is not trusted, the measurements reported can also not be trusted as the prover can simply fake its measurements. In particular, we need to show that there exists an underlying quantum state ρ consistent with the measurements reported by the prover. The key idea is to use the cryptographic primitive described (TCFs and TIFs) in 2.1 to make it hard for the prover to fake the measurements.

To simplify the soundness proof, we assume that the history state is only 1 qubit large. The generalization for n qubits follows by utilizing the fact that we can choose n different functions independently, the full proof can be found in [3].

4.1 Proof sketch

The proof proceeds as follows. First, the behaviour of an arbitrary prover is characterized using two unitary matrices. We then construct an underlying quantum state ρ for each prover which passes the test round. We then prove that the distribution of the Hadamard/standard basis measurements on this state ρ is computationally indistinguishable from the distribution of the Hadamard/standard basis measurements for any prover which passes the test round with some restrictions on the unitary matrices which characterize the prover. This restriction is then relaxed to include all unitary matrices. Thus, it is proved that there exists a well-defined underlying state ρ on which the prover performs its measurements and reports them. After verifying whether this state ρ is indeed low energy (by making Hadamard and standard basis measurements via the protocol in [4]), the proof can be accepted with arbitrarily high confidence.

4.2 Characterizing prover behaviour

First note that the protocol can be broken down into *rounds*, where each round consists of a verifier requesting something from the prover, and the prover giving some information to the verifier. The key insight is that any *general* prover can be modelled as behaving like an honest prover but applying an arbitrary unitary transformation to his state in each round. This implies that any general prover can be thought of as measuring the same registers as the honest prover – doing something else doesn't give the prover additional power. We can thus characterize an arbitrary prover by three unitary matrices (U_C, U_T, U_H) corresponding to the transformations applied before the commitment, test, or Hadamard rounds.

We can simplify this prover characterization a bit. Note that U_T can be assumed to not act on the register holding the commitment string - if it does, then we could copy the commitment string to an auxiliary register before performing the measurement, and U_T could act on this auxiliary register. Therefore, U_T commutes with the measurement of the commitment string, and can be pushed before the measurement in the commitment round. After the commitment measurement, the auxiliary register entangled with the commitment string also collapses to the same value, and the output of the test round is the same as if U_T had been applied after the measurement. Therefore, the general prover applies a unitary $U_1 = U_T U_C^{honest} U_C$ before the commitment bit measurement, and applies $U_2 = U_H U_T^\dagger$ before performing the Hadamard basis measurements, and does not do anything in the test round. Here U_C^{honest} refers to the unitary transformations applied by the honest prover. Thus, we can characterize prover behaviour using a pair of unitaries (U_1, U_2).

The test round acts as a check on the prover state. Now, just before the prover applies U_2 and then measures in Hadamard basis, it has to maintain a superposition over the claw entangled with the history state. That is, we can assume that the prover state before Hadamard measurement and after reporting y (with pre-images $x_{b,y}$) looks like this:

$$\sum_b |b\rangle|x_{b,y}\rangle|\psi_{b,x_{b,y}}\rangle$$

where $|\psi_{b,x_{b,y}}\rangle$ contains all additional qubits held by the prover. This is because the verifier can be assumed to perform the test round honestly, as argued earlier. If the commitment round was actually a standard basis measurement, the state is collapsed and resembles $|b\rangle|x_{y,b}\rangle|\psi_{b,x_{b,y}}\rangle$ for some random y, b with probabilities $|\alpha_b|^2$.

4.3 Underlying quantum state

We need to show that for any general prover $\mathbb{P}$, there exists a state ρ such that the distribution of measurements $D_{h,\rho}$ of this state in some basis h is computationally indistinguishable from the measurements returned by the prover $D_{h,\mathbb{P}}$ for basis h. Let ρ' be the state of the committed qubit prior to the prover's measurement in the Hadamard round. This ρ' can be thought of as the true history state ρ encoded with $Z^{d\cdot(x_{0,y}\oplus x_{1,y})}$. We define ρ as the state that the prover measures in the Hadamard round after being decoded by $Z^{d\cdot(x_{0,y}\oplus x_{1,y})}$. Thus, $D_{1,\mathbb{P}} = D_{1,\rho}$ by construction.

4.4 When U_2 is X-trivial

Let U_2 be X-trivial if it can commute with the standard basis measurement of the commitment bit. To simplify notation, we will use equality to mean computational indistinguishability. We now prove that if U_2 is X-trivial, then $D_{0,\mathbb{P}} = D_{0,\rho}$. Here, $D_{0,\rho}$ is the distribution over measurements we would have obtained if we could perform standard measurement on the state ρ that the prover obtains (encoded with a Z operator) in the Hadamard round.

There are two differences between $D_{0,\mathbb{P}}$ and $D_{0,\rho}$. First, when $h = 0$, the functions given to the prover are trapdoor injective, but ρ is constructed using a trapdoor claw-free function pair. Secondly, the prover measures the history state immediately in the commitment round via the commitment bit, but ρ is only measured later in the Hadamard round after application of U_2 (where Hadamard measurement is substituted with standard).

Note that if the measurement is in standard basis, the Z encoding doesn't affect the measurement. This means that to construct $D_{0,\rho}$, we don't need the trapdoor to decode the measurement. Thus, we can replace the TCF pair $f_0, f_1 \in \mathcal{F}$ used for constructing ρ with trapdoor injective pair $g_0, g_1 \in \mathcal{G}$ to obtain ρ_0 instead of $Z^{d\cdot(x_{0,y}\oplus x_{1,y})}\rho$ in the Hadamard round. Since f_0, f_1 and g_0, g_1 are computationally indistinguishable and the Z encoding does not matter for standard basis measurements, $D_{0,\rho_0} = D_{0,\rho}$.

We now have to show that $D_{0,\rho_0} = D_{0,\mathbb{P}}$, where $D_{0,\mathbb{P}}$ is the measurement via the commitment bit in the commitment round. Recall that after the commitment bit measurement, the state collapses to $|b\rangle|x_{y,b}\rangle|\psi_{b,x_{b,y}}\rangle$ (as verified in the test round), thus all further measurements will reveal the same value reported to the verifier. Also, U_2 is X-trivial and it commutes with this standard basis measurement, so instead of measuring after applying U_2 to the state, we could measure it beforehand. Thus, $D_{0,\rho_0} = D_{0,\mathbb{P}}$. Combined with $D_{0,\rho_0} = D_{0,\rho}$, we have $D_{0,\mathbb{P}} = D_{0,\rho}$. Therefore, $D_{0,\rho}$ is computationally indistinguishable from $D_{0,\mathbb{P}}$. Since $D_{1,\mathbb{P}} = D_{1,\rho}$ by construction, $D_{h,\mathbb{P}} = D_{h,\rho}$ when U_2 is X-trivial.

4.5 When U_2 is arbitrary unitary

We now show that there exists $\mathbb{P}'$ such that $D_{h,\mathbb{P}'}$ is computationally indistinguishable from $D_{h,\mathbb{P}}$, where $\mathbb{P}'$ is constrained to an X-trivial U_2, but $\mathbb{P}$ is a general prover with arbitrary U_2 which passes the test round. Both $\mathbb{P}$ and $\mathbb{P}'$ can be considered to have the same U_1. For $h = 0$, U_2 plays no part in $D_{0,\mathbb{P}'}$ or $D_{0,\mathbb{P}}$, and U_1 is the same in both cases. Therefore, $D_{0,\mathbb{P}}$ is indistinguishable from $D_{0,\mathbb{P}'}$. We will now consider the case where $h = 1$, that is, the verifier chooses the Hadamard round.

Let U_2 of $\mathbb{P}$ be denoted by U, and U_2 for $\mathbb{P}'$ be $\{U_x\}_{x\in\{0,1\}}$. Here,

$$U = \sum_{x,z\in\{0,1\}} X^x Z^z \otimes U_{xz} \qquad\qquad U_x = \sum_{z\in\{0,1\}} Z^z \otimes U_{xz}$$

The decomposition for U is in terms of Pauli matrices and is completely general. The map $\{U_x\}_{x\in\{0,1\}}$ is a equal mixture of U_x for $x \in \{0,1\}$. It is completely positive and trace preserving. Since each U_x is X-trivial (assuming that the committed bit register is first, corresponding to Z^z), the map $\{U_x\}_{x\in\{0,1\}}$ is also X-trivial. We now need to replace U with $\{U_x\}_{x\in\{0,1\}}$ and simulate U in the Hadamard round to show the indistinguishability. This replacement can be done using a construct called the Pauli twirl, which will allow us to replace $\{U_x\}_{x\in\{0,1\}}$ with $\{\frac{1}{\sqrt{2}}(Z^z\otimes I)U(Z^z\otimes I)\}_{x\in\{0,1\}}$, which is the equal mixture of $\frac{1}{\sqrt{2}}(Z\otimes I)U(Z\otimes I)$ and $\frac{1}{\sqrt{2}}U$. Observe that if we replace the first map with $\frac{1}{\sqrt{2}}U$ too, we get back U and we are done. We need to thus show that if U is replaced by $(Z\otimes I)U(Z\otimes I)$, the distribution over measurements $D_{h,\mathbb{P}}$ is changed by a computationally indistinguishable amount.

The proof strategy here is to show that if any algorithm $\mathcal{A}$ could could detect this change, it would violate one of the hardcore bit properties outlined in 2.1. Intuitively, we need to exploit the (apparent) randomness introduced by the $X^{d\cdot(x_1\oplus x_2)}$ encoding in the Hadamard measurement step and the entanglement of the history state with the claw in the commitment stage.

Note 3

Pauli twirl: The Pauli twirl is obtained when we conjugate a unitary operation U using a random Pauli gate. Formally, a Pauli twirl of U would be denoted as $\{(X^xZ^z)^\dagger U(X^xZ^z)\}_{x,z\in\{0,1\}}$ or $\{Z^zX^xU(X^xZ^z)\}_{x,z\in\{0,1\}}$. We will use a Z-Pauli twirl which is $\{\frac{1}{\sqrt{2}}(Z^z\otimes I)U(Z^z\otimes I)\}_{z\in\{0,1\}}$. It can be proven that this Z-Pauli twirl is equal to $\{(X^x\otimes I)U_x\}_{x\in\{0,1\}}$ (Proof in [3]). In the Hadamard round, the X^x has no effect on the Hadamard basis measurement and the resulting distribution is identical to that of $\{U_x\}_{x\in\{0,1\}}$.

Let the distributions $D_{h,\mathbb{P}}$ and $D_{h,\mathbb{P}'}$ corresponding to U and $(Z\otimes I)U(Z\otimes I)$ be represented by density matrices σ_0 and σ_1. Recall that if the prover passes the test round, the state $|\phi_y\rangle$ obtained just after the commitment round is constrained to be in a superposition $\sum_b |b\rangle|x_{b,y}\rangle|\psi_{b,x_{b,y}}\rangle$. If we form the density matrix $|\phi_y\rangle\langle\phi_y|$ and we perform the prover and verifier operations, we get σ_0 or σ_1 depending on if the prover uses U or $(Z\otimes I)U(Z\otimes I)$. We split $\sigma_0 = \sigma_0^D + \sigma_0^C$ and $\sigma_1 = \sigma_1^D + \sigma_1^C$. Here, σ_r^D and σ_r^C are defined as what we get if the prover and the verifier operations in the Hadamard round is applied to the diagonal and cross terms of $|\phi_y\rangle\langle\phi_y|$ respectively. This decomposition is possible because quantum operations are linear. Notice that σ_r^D is a legitimate density matrix (trace is 1), but σ_r^C (trace is 0) is not. If we now prove that (a) σ_0^D and σ_0^D are computationally indistinguishable, and (b) the trace distance between σ_0^C and σ_1^C is computationally negligible (i.e. cannot be amplified by a BQP machine), we prove that σ_0 and σ_1 is computationally indistinguishable

4.5.1 Diagonal Terms

The creation of σ_0^D proceeds from the state represented by the diagonal terms of $|\phi_y\rangle\langle\phi_y|$, that is,

$$\sum_b |b\rangle\langle b| \otimes |x_b\rangle\langle x_b| \otimes |\psi_{b,x_b}\rangle\langle\psi_{b,x_b}|$$

This initial state can be obtained by following the prover's steps in the commitment round, measuring the commitment bit b and the pre-image x_b and throwing them away to zero out the off-diagonal terms. The prover applies U, then measures in the Hadamard basis, and the verifier decodes using its X operator. This state is σ_0^D, and if U is replaced by $(Z\otimes I)U(Z\otimes I)$, it would be σ_1^D. The first Z operation leaves the state unchanged, as there are only diagonal terms in the density matrix. The second Z operation can be pushed through the Hadamard measurement to become X, therefore the state $\sigma_1^D = X\sigma_0^D X$.

Now, consider any algorithm $\mathcal{A}'$ which can distinguish between σ_1^D and σ_0^D. Then, we construct $\mathcal{A}$ which violates the first hardcore bit property as follows. $\mathcal{A}$ first creates σ^D by following the steps of the prover $\mathbb{P}$ but stops short of the verfier's decoding. In the process, it obtains d, b, x_b by measuring respective registers. Now, it runs $\mathcal{A}'$ on $\sigma^D = \sigma_r^D$ where $r = d\cdot(x_0\oplus x_1)$, and thus finds r. Thus $\mathcal{A}$ can compute $d, b, x_b, r = d\cdot(x_0\oplus x_1)$, contradicting the first hardcore bit property (1).

4.5.2 Cross Terms

We say that σ_0^C and σ_1^C are computationally distinguishable if the trace distance is not negligible, that is $|\mathrm{Tr}\mathcal{S}(\sigma_0^C - \sigma_1^C)|$ is not negligible for some quantum map $\mathcal{S}$. For the cross diagonal terms, the first Z of $(Z \otimes I)U(Z \otimes I)$ flips the phase to -1, and the second Z can be pushed through the Hadamard measurement as before. Therefore, $\sigma_1^C = -X\sigma_0^C X$, and $\sigma_0^C - \sigma_1^C = \sigma_0^C + X\sigma_0^C X$.

Now, the map converting the cross terms of the initial state $|\phi_y\rangle\langle\phi_y|$ to σ_0^C is efficient except for the $X^{d\cdot(x_1 \oplus x_2)}$ term of the verifier's X decoding. But it is not necessary to perform this step as $\sigma_0^C + X\sigma_0^C X$ remains unchanged whether this step is included or not. Thus, there is an efficient map from the cross terms of $|\phi_y\rangle\langle\phi_y|$ to $\sigma_0^C - \sigma_1^C$.

We now define $\hat{\sigma}_r = (Z^r \otimes I)|\phi_y\rangle\langle\phi_y|(Z^r \otimes I)$. It can be observed that

$$\hat{\sigma}_0 - \hat{\sigma}_1 = 2\sum_b |b\rangle\langle b \oplus 1| \otimes |x_b\rangle\langle x_{b\oplus 1}| \otimes |\psi_{b,x_b}\rangle\langle\psi_{b,x_{b\oplus 1}}|$$

which is nothing but 2 times the cross terms of $|\phi_y\rangle\langle\phi_y|$. Thus, there is an efficient map between $\hat{\sigma}_0 - \hat{\sigma}_1$ and $\sigma_0^C - \sigma_1^C$. We now prove that if there is an efficient algorithm $\mathcal{A}'$ which can distinguish between $\hat{\sigma}_0$ and $\hat{\sigma}_1$ (equivalently σ_0^C and σ_1^C), there exists $\mathcal{A}$ which violates the second hardcore bit property (2).

Observe that for any arbitrary string d, we have

$$(I \otimes Z^d \otimes I)\sum_b |b\rangle|x_{b,y}\rangle|\psi_{b,x_{b,y}}\rangle = (Z^{d\cdot(x_{1,y}\oplus x_{0,y})} \otimes I \otimes I)\sum_b |b\rangle|x_{b,y}\rangle|\psi_{b,x_{b,y}}\rangle$$

Let $\mathcal{A}$ take any arbitrary string d and apply $I \otimes Z^d \otimes I$ on $|\phi_y\rangle$. This is equivalent to applying $Z^{d\cdot(x_{1,y}\oplus x_{0,y})} \otimes I \otimes I$. It then runs $\mathcal{A}'$ which can tell the value $d \cdot (x_{1,y} \oplus x_{0,y})$. Thus, $\mathcal{A}$ is able to find $d \cdot (x_{1,y} \oplus x_{0,y})$ for any string d efficiently, which violates the second hardcore bit property.

We have now proven that there exists a well-defined ρ on which the prover reports its measurements to the verifier. Whether this ρ is the correct low energy ground state (history state) will be decided by the verification algorithm in [4]. This means that the protocol is *sound* – it will only accept the proof if there is a appropriate history state ρ for which the ground state is low energy.

5 Conclusion

Thus, relying on the hardness of LWE for quantum computers, we are able to show that by using the described protocol, the operations of a quantum computer can be verified using strictly classical means by encoding the problem in a 2-local Hamiltonian. Using this protocol, we can be certain that the prover was actually constructing the ground state and reporting the measurements on it faithfully.

References

[1] P. W. Shor, *Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer*, SIAM Journal on Computing 26, 1484–1509 (1997).

[2] Dorit Aharonov, Michael Ben-Or, Elad Eban, and Urmila Mahadev. Interactive Proofs for Quantum Computations. Arxiv preprint 1704.04487, 2017

[3] Mahadev, Urmila. "Classical verification of quantum computations." 2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS). IEEE, 2018.

[4] Morimae, Tomoyuki, and Joseph F. Fitzsimons. ”Post hoc verification with a single prover.” arXiv preprint arXiv:1603.06046 (2016).

[5] Daniel Gottesman, 2004. As referenced in [http://www.scottaaronson.com/blog/?p= 284;]

[6] Broadbent, Anne, Joseph Fitzsimons, and Elham Kashefi. ”Universal blind quantum computation.” 2009 50th Annual IEEE Symposium on Foundations of Computer Science. IEEE, 2009.

[7] Zvika Brakerski, Paul Christiano, Urmila Mahadev, Umesh Vazirani, and Thomas Vidick. Certifiable randomness from a single quantum device. Arxiv preprint 1804.00640, 2018.

[8] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In Proceedings of the Thirty-seventh Annual ACM Symposium on Theory of Computing, STOC ’05

[9] Ran Raz and Avishay Tal. 2019. Oracle separation of BQP and PH. In Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing (STOC 2019). Association for Computing Machinery, New York, NY, USA, 13–23. https://doi.org/10.1145/3313276.3316315

[10] Urmila Mahadev. Classical Homomorphic Encryption for Quantum Circuits. Arxiv preprint arXiv:1708.02130, 2017

[11] Morimae, Tomoyuki, Daniel Nagaj and Norbert Schuch. (2015). ”Quantum proofs can be verified using only single qubit measurements.” Physical Review A. 93. 10.1103/PhysRevA.93.022326.