2018 IEEE 59th Annual Symposium on Foundations of Computer Science

Classical Verification of Quantum Computations

Urmila Mahadev
Department of Computer Science, UC Berkeley
mahadev@berkeley.edu

***Abstract*—We present the first protocol allowing a classical computer to interactively verify the result of an efficient quantum computation. We achieve this by constructing a measurement protocol, which enables a classical verifier to use a quantum prover as a trusted measurement device. The protocol forces the prover to behave as follows: the prover must construct an n qubit state of his choice, measure each qubit in the Hadamard or standard basis as directed by the verifier, and report the measurement results to the verifier. The soundness of this protocol is enforced based on the assumption that the learning with errors problem is computationally intractable for efficient quantum machines.**

I. Introduction

We propose a solution to the open question of verifying quantum computations through purely classical means. The question is as follows: is it possible for an efficient classical verifier (a BPP machine) to verify the output of an efficient quantum prover (a BQP machine)? This question was first raised by Daniel Gottesman in 2004 ([1]). In the absence of any techniques for tackling this question, two weaker formulations were considered. In the first, it was shown that if the verifier had access to a small quantum computer, verification of all efficient quantum computations was possible ([2], [3], [4], [5]). The second formulation considered a classical polynomial time verifier interacting with two entangled, non communicating quantum provers (rather than just one machine), and showed that in this setting it was possible to verify the result of an arbitrary quantum computation ([6]). Although both lines of work initiated extensive research efforts, the question of classical verification by interaction with a single quantum computer has remained elusive.

In this paper, we answer this question affirmatively: we show that a classical polynomial time verifier (a BPP machine) can interact with an efficient quantum prover (a BQP machine) in order to verify BQP computations. We rely on the additional assumption that the verifier may use post-quantum cryptography that the BQP prover cannot break. More specifically, we rely on quantum secure classical encryption schemes, such as those based on the learning with errors problem ([7]). These schemes can be used to encrypt classical bits (or quantum states) in a way in which an efficient quantum machine cannot extract any information.

The core of our construction is a *measurement protocol*, an interactive protocol between an efficient quantum prover and a classical verifier which is used to force the prover to behave as the verifier's trusted measurement device. To formalize the idea of a measurement protocol, we now describe its completeness and soundness conditions, beginning with a small amount of necessary notation. In our measurement protocol, an honest prover constructs an n qubit quantum state ρ of his choice, and the verifier would like each qubit to be measured in either the standard basis or the Hadamard basis. Denote the choice of measurement basis by an n bit string $h = (h_1, \ldots, h_n)$. For a prover $\mathbb{P}$ and a measurement basis choice $h = (h_1, \ldots, h_n)$, we define $D_{\mathbb{P},h}$ to be the resulting distribution over the measurement result $m \in \{0,1\}^n$ obtained by the verifier. For an n qubit state ρ, we define $D_{\rho,h}$ to be the distribution obtained by measuring ρ in the basis corresponding to h.

Our measurement protocol is complete, in the following sense: for all efficiently computable n qubit states ρ, there exists a prover $\mathbb{P}$ such that $D_{\mathbb{P},h}$ is approximately equal to $D_{\rho,h}$ for all h. Moreover, $\mathbb{P}$ is accepted by the verifier with all but negligible probability. Our soundness notion for the measurement protocol is slightly more complex, but a simplified form is as follows: if the prover $\mathbb{P}$ is accepted by the verifier with perfect probability, there exists an efficiently computable n qubit quantum state ρ underlying the distribution over m. More precisely, for all $h \in \{0,1\}^n$, $D_{\rho,h}$ is computationally indistinguishable from $D_{\mathbb{P},h}$. The full soundness guarantee is achieved by making this statement robust: we will show that if a prover $\mathbb{P}$ is accepted by the verifier on basis choice h with probability $1 - p_h$, there exists a prover $\mathbb{P}'$ who is always accepted by the verifier and the statistical distance between $D_{\mathbb{P},h}$ and $D_{\mathbb{P}',h}$ is approximately $\sqrt{p_h}$ for all h.

So far, we have described the goal of our measurement protocol, which is to force the prover to behave as the verifier's trusted standard/Hadamard basis measurement device. To link our measurement protocol to verification, we simply need to show that a classical verifier who also has access to such a trusted measurement device can verify the result of BQP computations.

To describe how such verification can be done, we briefly recall a method of verifying classical computations. Assume that, for a language $L \in$ BPP and an instance x, the verifier wishes to check whether $x \in L$. To do so, the verifier can reduce x to a 3-SAT instance, ask the prover for a satisfying variable assignment, and verify that the assignment satisfies the instance. There is an analogous setting for quantum computations, in which the language $L \in$ BQP, the instance x can be reduced to a local Hamiltonian instance H_x, an n bit variable assignment corresponds to an n qubit quantum state $|\psi\rangle$, and the fraction of unsatisfied clauses corresponds to the energy of the Hamiltonian H_x with respect to the state $|\psi\rangle$ ([8]). If $x \in L$, there exists a state with low energy with respect

2575-8454/18/$31.00 ©2018 IEEE
DOI 10.1109/FOCS.2018.00033

to H_x; if not, all states will have sufficiently high energy. We will rely on the fact that the energy of $|\psi\rangle$ with respect to H_x can be estimated by performing only standard/Hadamard basis measurements ([9]).

With this analogy, verification of a quantum computation can be performed by a classical verifier with access to a trusted standard/Hadamard basis measurement device as follows ([10]): the verifier first reduces the instance x to be verified to a local Hamiltonian instance H_x, then requests an n qubit state from the prover, and finally checks (via standard/Hadamard basis measurements) if the received state has low energy with respect to H_x. If so, the verifier is assured that $x \in L$.

Now that we have seen how to link our measurement protocol to verification, we proceed to describing how our measurement protocol works; essentially, it is a weak quantum state commitment procedure. Ideally, this commitment would operate as follows: at the start of the protocol, the verifier asks the prover for a classical commitment to a quantum state. This commitment should guarantee that the prover is forced to perform either a Hadamard or standard basis measurement as directed by the verifier on the committed state. Such a commitment scheme is enough to satisfy the soundness condition of the measurement protocol described above. Note the difference between this notion of commitment and the standard cryptographic notion: our commitment differs in that it does not need to hide the quantum state, but is similar since it is binding the prover to measuring the state he has committed to.

The actual commitment performed in our measurement protocol has a weaker guarantee than the ideal scheme described above: the prover is asked to commit to a state ρ, but his hands are not entirely tied when it comes to the measurement of the state. The prover must perform the standard basis measurement on the committed state ρ, but he can perform a specific type of deviation operator prior to Hadamard measurement: this operator must commute with standard basis measurement (for example, it could be a Pauli Z operator). The key point here is that this weaker commitment protocol is still strong enough to guarantee the soundness of the measurement protocol. To see why, observe that the deviation operator could have been applied prior to the commitment, creating a different committed state ρ'. Due to the fact that the deviation operator commutes with standard basis measurement, the measurement distribution obtained by the verifier (for both the Hadamard and standard bases) would have been equivalent had the prover instead committed to the state ρ', and honestly performed the measurement requested by the verifier on ρ'.

The commitment structure described above is obtained from a classical cryptographic primitive called a trapdoor claw-free function, a function f which is two to one, easy to invert with access to a trapdoor, and for which it is computationally difficult to find any pair of preimages with the same image. Such a pair of preimages is called a claw, hence the name claw-free. These functions are particularly useful in the quantum setting, due to the fact that a quantum machine can create a uniform superposition over a random claw (x_0, x_1): $\frac{1}{\sqrt{2}}(|x_0\rangle + |x_1\rangle)$. This superposition can be used to obtain information which is hard to obtain classically: the quantum machine can obtain *either* a string $d \neq 0$ such that $d \cdot (x_0 \oplus x_1) = 0$ or one of the two preimages x_0, x_1. In [11], this advantage was introduced and used to show how to generate information theoretic randomness from a single quantum device.

Trapdoor claw-free function are used in our measurement protocol as follows. The prover is first asked to commit to a state of his choice (assume the state is $\alpha_0 |0\rangle + \alpha_1 |1\rangle$) by entangling it with a random claw (x_0, x_1), creating the state:

$$\alpha_0 |0\rangle |x_0\rangle + \alpha_1 |1\rangle |x_1\rangle$$

The corresponding classical commitment, which is sent to the verifier, is the image $y = f(x_0) = f(x_1)$. In order to force the prover to perform the desired measurement on the committed state, the claw-free property is strengthened in two different ways and is then used to randomize any operator applied by the prover which is a deviation from the requested measurement, rendering the deviation of the prover essentially useless. We provide the definition and construction of an *extended trapdoor claw-free family* which satisfies the strengthened claw-free properties. This family is an extension of the family given in [11]. Like the construction in [11], our construction relies on the computational assumption that a BQP machine cannot solve the learning with errors problem with superpolynomial noise ratio ([7]).

The main result of our paper is stated informally as follows:

Theorem I.1. ***(Informal)*** *Assuming the existence of an extended trapdoor claw-free family, all decision problems which can be efficiently computed in quantum polynomial time (the class BQP) can be verified by an efficient classical machine through interaction (formally, BQP = QPIP$_0$).*

In the full version of this paper ([12]), we also provide a learning with errors based construction of an extended trapdoor family, giving a proof of the following theorem:

Theorem I.2. ***(Informal)*** *Under the assumption that the learning with errors problem with superpolynomial noise ratio is computationally intractable for an efficient quantum machine, there exists an extended trapdoor claw-free family.*

A. Related Work

The basic idea of using trapdoor claw-free functions to constrain the power of the prover in the context of randomness generation was introduced in [11]. This idea was further used in a computational setting to hide information from the prover (but not constrain the prover) in [13]. What is new to our paper is that it constrains the prover in the context of carrying out a particular computation. This requires the verifier to exert a much greater degree of control over the prover than in randomness generation. To accomplish this goal, we develop a new protocol and proof techniques for strongly characterizing the state of an untrusted prover.

B. Outline

Our measurement protocol relies on two cryptographic primitives which give a BPP verifier some leverage over a BQP prover. We begin by describing these primitives in Section II. Using these primitives, we can describe our measurement protocol in Section III. In Sections IV - V, we show how the two cryptographic primitives can be used to guarantee soundness of the measurement protocol, in the sense that all provers must essentially be creating a state and measuring it in the basis chosen by the verifier. In Section VI, we show how to extend our measurement protocol to a verification protocol for all of BQP. This paper is an overview of our result; formal statements and complete proofs are deferred to the full version ([12]).

II. Cryptographic Primitives

A. Trapdoor Claw-Free Families

The first cryptographic primitive we will use is a function family $\mathcal{F} = \{f_{k,b} : \mathcal{X} \to \mathcal{Y}\}$ (for $b \in \{0,1\}$) called a *trapdoor claw-free* function family. For convenience, we will assume in this overview that $\mathcal{X} = \{0,1\}^w$. A trapdoor claw-free function family is a family of functions which are two to one (both $f_{k,0}(\cdot)$ and $f_{k,1}(\cdot)$ are injective and their images are equal) and for which it is computationally difficult to find a *claw*, i.e. a pair of points x_0 and x_1 which have the same image ($f_{k,0}(x_0) = f_{k,1}(x_1)$). Given y in the image of $f_{k,0}$ or $f_{k,1}$, the trapdoor t_k of the functions $f_{k,0}, f_{k,1}$ allows recovery of both preimages of y. The trapdoor claw-free family also satisfies two hardcore bit properties, which are stronger versions of the claw-free property: roughly, they state that it is computationally difficult to find a string d and the bit $d \cdot (x_0 \oplus x_1)$, where (x_0, x_1) form a claw. These two properties are specified as needed (in Claim V.1 and Claim V.2).

In this version of the paper, we will assume the existence of the function family described above for simplicity. However, since we do not know how to construct such a family, the full version of the paper ([12]) instead relies on an approximate version of this family, called an extended trapdoor claw-free family. Both the definition and construction (from learning with errors) are extensions of those given in [11].

We now describe a BQP process we call *state commitment*, which requires a function key k corresponding to functions $f_{k,0}, f_{k,1} \in \mathcal{F}$ (we assume that computing the functions $f_{k,0}, f_{k,1}$ only requires access to the function key k). The state commitment process is performed with respect to an arbitrary single qubit state $|\psi\rangle$:

$$|\psi\rangle = \sum_{b \in \{0,1\}} \alpha_b |b\rangle \tag{1}$$

The commitment process consists of two steps. First, the functions $f_{k,0}, f_{k,1}$ are applied in superposition, using $|\psi\rangle$ to determine whether to apply $f_{k,0}$ or $f_{k,1}$ and a uniform superposition over $x \in \mathcal{X}$ as the input to $f_{k,0}$ or $f_{k,1}$:

$$\frac{1}{\sqrt{|\mathcal{X}|}} \sum_{b \in \{0,1\}} \sum_{x \in \mathcal{X}} \alpha_b |b\rangle |x\rangle |f_{k,b}(x)\rangle \tag{2}$$

Second, the final register of the resulting state is measured, obtaining $y \in \mathcal{Y}$. At this point, the state is:

$$\sum_{b \in \{0,1\}} \alpha_b |b\rangle |x_{b,y}\rangle \tag{3}$$

where $x_{0,y}$ and $x_{1,y}$ are the two preimages of y. We will call the qubit containing b the *committed qubit*, the register containing $x_{b,y}$ the *preimage register* and the string y the *commitment string*. The crucial point here is that, due to the claw-free nature of the functions $f_{k,0}, f_{k,1}$, it is computationally difficult for a BQP machine to compute both inverses $x_{0,y}$ and $x_{1,y}$ given only y. However, with access to the trapdoor t_k, both inverses can be computed from y. If we think of the state commitment process in an interactive setting, in which the verifier selects the function key and the trapdoor and the BQP prover performs the commitment process (sending the commitment y to the verifier), the BQP prover cannot compute both inverses, but the verifier can. This gives the verifier some leverage over the prover's state.

A key property of the committed state in (3) is that it allows a logical Hadamard measurement up to an X Pauli operator, which is performed as follows. First, a Hadamard transform is applied to both the committed qubit and preimage register of the state in (3):

$$\frac{1}{\sqrt{|\mathcal{X}|}} \sum_{d \in \mathcal{X}} X^{d \cdot (x_{0,y} \oplus x_{1,y})} H |\psi\rangle \otimes Z^{x_{0,y}} |d\rangle \tag{4}$$

The next step in applying the logical Hadamard measurement is to measure the second (preimage) register, obtaining $d \in \mathcal{X}$. The state at this point is:

$$X^{d \cdot (x_{0,y} \oplus x_{1,y})} H |\psi\rangle \tag{5}$$

To obtain the Hadamard measurement of $|\psi\rangle$, the operator $X^{d \cdot (x_{0,y} \oplus x_{1,y})}$ (which we call the decoding operator and requires the trapdoor) is first applied, followed by a standard basis measurement of $H|\psi\rangle$. Note that these two operations commute: it is equivalent to first perform a standard basis measurement of the state in (5) followed by applying the X decoding operator. The X decoding operator applied after measurement is simply the classical XOR operation.

We can again think of this logical Hadamard transform in the interactive setting, in which the BQP prover applies the Hadamard transform to obtain the state in (4) and then measures the committed qubit and preimage register, sending the measurement results $b' \in \{0,1\}$ and $d \in \{0,1\}^w$ to the verifier. The verifier decodes the measurement b' by XORing it with $d \cdot (x_{0,y} \oplus x_{1,y})$ (which can be computed using the trapdoor) to obtain the bit m, which the verifier stores as the result of the Hadamard basis measurement.

B. Trapdoor Injective Function Families

The second primitive is a function family $\mathcal{G} = \{g_{k,b} : \mathcal{X} \to \mathcal{Y}\}$ (for $b \in \{0,1\}$) called a *trapdoor injective* function family. A trapdoor injective function family is a family of injective functions such that the images of $g_{k,0}(\cdot)$ and $g_{k,1}(\cdot)$ are disjoint. Given $y = g_{k,b}(x_{b,y})$, the trapdoor t_k of the

functions $g_{k,0}, g_{k,1}$ allows recovery of $b, x_{b,y}$. We will also require that the trapdoor injective family is computationally indistinguishable from the trapdoor claw-free family: given a function key k, it must be computationally difficult to determine whether k belongs to an injective or claw-free family. As in the case of trapdoor claw-free families, we will assume the existence of the function family described above for this version of the paper, but in the full version ([12]) we will rely on an approximate version of this function family.

The state commitment process described in Section II-A can also be performed with a function key k corresponding to functions $g_{k,0}, g_{k,1} \in \mathcal{G}$. At the stage of (2), the following state has been created:

$$\frac{1}{\sqrt{|\mathcal{X}|}} \sum_{b\in\{0,1\}} \sum_{x\in\mathcal{X}} \alpha_b |b\rangle |x\rangle |g_{k,b}(x)\rangle \tag{6}$$

However, in this case when the last register is measured to obtain $y \in \mathcal{Y}$, the superposition over b collapses. This is because the images of $g_{k,0}(\cdot)$ and $g_{k,1}(\cdot)$ are disjoint. It follows that with probability $|\alpha_b|^2$, $y \in g_{k,b}(\cdot)$. In this case, the state after measurement of y is:

$$|b\rangle |x_{b,y}\rangle \tag{7}$$

where $x_{b,y}$ is the unique preimage of y.

If we think of this process in the interactive setting (in which the BQP prover performs the commitment and sends the verifier y), the verifier can use the trapdoor t_k to extract $(b, x_{b,y})$ from y. Therefore, the verifier can obtain b, which is the result of standard basis measurement of the prover's state, simply by asking for the commitment y.

III. Measurement Protocol

Given the two primitives described in Section II, we can now describe our measurement protocol. Before the protocol begins, the verifier will select the basis for which he would like a measurement result (either Hadamard or standard) for each of n qubits. We will represent this basis choice by a string $h \in \{0,1\}^n$. If $h_i = 0$, it indicates that the standard basis was chosen for qubit i.

The protocol begins with the commitment round. For all i for which $h_i = 1$, the verifier samples a key k_i and a trapdoor t_{k_i} corresponding to a pair of trapdoor claw-free functions $f_{k_i,0}, f_{k_i,1} \in \mathcal{F}$. For all i for which $h_i = 0$, the verifier samples a key k_i and a trapdoor t_{k_i} corresponding to a pair of trapdoor injective functions $g_{k_i,0}, g_{k_i,1} \in \mathcal{G}$. The verifier sends the prover the function keys $k_1, \ldots, k_n$. The honest prover first constructs an n qubit state of his choice and then commits to each qubit of the state using the corresponding function key, as described in Section II. The verifier receives the measurement results $y_1, \ldots, y_n \in \mathcal{Y}$ from the prover.

Once the verifier receives $y_1, \ldots, y_n$, he chooses at random either to run a test round or a Hadamard round. In the case of a test round, the verifier requests standard basis measurements of the committed qubit and preimage register for all n qubits. He receives $b'_i \in \{0,1\}$ and $x'_i \in \mathcal{X}$ from the prover and checks if the measurement is a preimage of y_i.

In the case of a Hadamard round, the verifier requests a Hadamard basis measurement of the committed qubit and preimage register for all i. The verifier receives $b'_i \in \{0,1\}, d_i \in \mathcal{X}$ for all i. For all i for which $h_i = 0$, the verifier ignores the measurement results b'_i and d_i and uses the trapdoor of g_{k_i} to invert y_i. The verifier stores the first bit of the inverted value as the standard basis measurement result m_i. For all i for which $h_i = 1$, the verifier first decodes b'_i by XORing it with $d_i \cdot (x_{0,y_i} \oplus x_{1,y_i})$ (this can equivalently be thought of as applying the decoding operator $X^{d\cdot(x_{0,y_i}\oplus x_{1,y_i})}$- see the end of Section II-A). The verifier stores the result $m_i = b'_i \oplus d_i \cdot (x_{0,y_i} \oplus x_{1,y_i})$ as the Hadamard basis measurement result.

Completeness of our measurement protocol (as defined in the introduction) follows immediately from the description of the state commitment process given in Sections II-A and II-B.

IV. Measurement Protocol Soundness

We now give an overview of our soundness guarantee: we describe how to show that for $n = 1$ and a prover $\mathbb{P}$ who passes the test round perfectly, there exists a quantum state underlying the distribution over measurement results obtained by the verifier. The generalization to arbitrary n (given in the full version) follows easily due to the fact that all n function keys are drawn independently. The generalization to provers $\mathbb{P}$ who do not pass perfectly (also given in the full version) is straightforward as well; it is done by conditioning $\mathbb{P}$ on acceptance in a test round, thereby creating an efficient prover who passes the test round perfectly as long as $\mathbb{P}$ is accepted with non negligible probability. In this section, we begin by characterizing the behavior of a general prover. We then show that if this characterization satisfies a certain requirement, we can prove the existence of an underlying quantum state. In Section V (which is the crux of this paper), we show how to enforce this requirement on general provers.

A. Prover Behavior

The analysis of the measurement protocol is based on understanding and characterizing the prover's Hilbert space and operations. We will rely on the following principle behind interactive proofs between a BQP prover and a BPP verifier. A round of the protocol begins with the verifier's message and ends with the prover's message. A general prover is equivalent, from the verifier's perspective, to a prover who begins each round by applying an arbitrary unitary operator to his space and then behaves exactly the same as an honest prover, culminating in a measurement (the result of which is sent to the verifier). This principle implies that an arbitrary prover measures the same registers that an honest prover does in each round, which will be particularly useful in our protocol.

Let $\mathbb{P}_0$ be an honest prover in our measurement protocol and assume the unitary operator he applies in the commitment round is $U_{C,0}$, after which he measures the commitment string register in the standard basis. As described in Section II, $\mathbb{P}_0$ has three designated registers: the register containing the committed qubit, the preimage register, and the commitment

string register. Each message of $\mathbb{P}_0$ to the verifier is the result of the measurement of one of these registers.

It follows from the principle above that a general prover $\mathbb{P}$ has the same designated registers as $\mathbb{P}_0$ and is characterized by 3 unitary operators: the unitary U_C applied in the commitment round, the unitary U_T applied in the test round, and the unitary U_H applied in the Hadamard round. We assume that both U_T and U_H do not act on the commitment string register, since it has already been measured; the measurement result could have been copied into the auxiliary space, on which U_T and U_H can act.

We now use the structure of our protocol to simplify the general prover one step further. There are only two possible messages of the verifier for the second round of our protocol: the message indicates either a test or Hadamard round. Due to this property, we can assume that the test round attack U_T is equal to the identity operator. To see this, we only need to make one observation: the attack U_T applied in the test round commutes with the measurement of the commitment string. Therefore, it could have been applied prior to reporting the commitment string y.

It follows that the general prover $\mathbb{P}$ described above is identical, from the perspective of the verifier, to a prover who applies the unitary $U_0 = U_T U_{C,0} U_C$ immediately prior to measuring the commitment string register and applies $U = U_H U_T^\dagger$ prior to performing Hadamard basis measurements of the committed qubit and preimage register in the Hadamard round. We will say that such a prover is *characterized* by (U_0, U). For a formal statement and proof of the above argument, see the full version ([12]).

The characterization of all provers by two unitary attacks allows us to use the test round of the measurement protocol to enforce that the prover's state has a specific structure, which is derived from the cryptographic primitives in Section II. Let $\mathbb{P}$ be a prover who passes the test round perfectly. If $h = 1$, the state of $\mathbb{P}$ at the start of either the test or the Hadamard round (i.e. immediately after reporting y) can be written as follows (the two preimages of y are $x_{0,y}, x_{1,y}$):

$$\sum_{b \in \{0,1\}} |b\rangle |x_{b,y}\rangle \left|\psi_{b,x_{b,y}}\right\rangle \tag{8}$$

where $\left|\psi_{b,x_{b,y}}\right\rangle$ contains all additional qubits held by the prover. This is because the verifier checks, in a test round, if he receives a valid pre-image from the prover. Since the prover simply measures the requested registers when asked by the verifier in a test round (i.e. he does not apply an attack in the test round), these registers must be in a superposition over the two preimages of the reported measurement result y.

If $h = 0$ and $\mathbb{P}$ reports y, there is only one inverse of y. If we assume this inverse is $x_{b,y}$ (i.e. $g_{k,b}(x_{b,y}) = y$), the state of $\mathbb{P}$ at the start of the test or Hadamard round can be written as follows, due to the same reasoning used in (8):

$$|b\rangle |x_{b,y}\rangle \left|\psi_{b,x_{b,y}}\right\rangle \tag{9}$$

This structure enforced by the test run is the key to proving the existence of an underlying quantum state, as we will see shortly.

B. Construction of Underlying Quantum State

We begin by using the characterization of general provers in Section IV-A to define a single qubit state ρ corresponding a prover $\mathbb{P}$ who is characterized by (U_0, U). Recall that $\mathbb{P}$ has a well defined committed qubit, which he measures when the verifier asks for the measurement of a committed qubit. Let ρ' be the state of the committed qubit prior to the prover's measurement in the Hadamard round in the case that $h = 1$. We can think of ρ' as encoded by the operator $Z^{d \cdot (x_{0,y} \oplus x_{1,y})}$, which is determined by the prover's measurements d and y. This Z operator is derived from the verifier's X decoding operator applied in the measurement protocol; we have used a Z operator here since the Hadamard measurement has not yet been performed. The single qubit state ρ will be the result of applying the Z decoding operator to the committed qubit ρ'.

Define X-trivial operators to be those which commute with standard basis measurement of the committed qubit. We now show that if the prover's Hadamard round attack U is an X-trivial operator, the distribution $D_{\mathbb{P},h}$ obtained by the verifier in the measurement protocol is computationally indistinguishable from the distribution which is obtained by measuring ρ in basis specified by h.

Recall that $D_{\rho,h}$ is the distribution obtained by measuring ρ in the basis corresponding to h. By construction, $D_{\rho,1} = D_{\mathbb{P},1}$. If $h = 0$, there are two differences between the distribution $D_{\rho,h}$ and the distribution $D_{\mathbb{P},h}$. The first differences lies in the function sampling: in our measurement protocol, an injective function is sampled if $h = 0$, but in the state ρ, a claw-free function is sampled. The second difference comes from how the standard basis measurement is obtained: in $D_{\mathbb{P},h}$ the standard basis measurement is obtained from the commitment string y, but in $D_{\rho,h}$ the standard basis measurement is obtained by measuring ρ (the committed qubit) in the standard basis.

We can handle the first difference by making two key observations. First, the Z decoding operator has no effect if $h = 0$; in this case, the committed qubit will be measured in the standard basis immediately after application of Z in order to obtain $D_{\rho,h}$. Second, if the Z decoding operator is not applied, the trapdoor t_k is no longer needed to construct the distribution $D_{\rho,h}$. If $D_{\rho,h}$ is only dependent on the function key k (and not the trapdoor t_k), the function key k can be replaced with a function key which corresponds to a pair of trapdoor injective functions, rather than a pair of trapdoor claw-free functions, to obtain a computationally indistinguishable distribution. This is due to the computational indistinguishability between keys drawn from the trapdoor claw-free family $\mathcal{F}$ and the trapdoor claw-free family $\mathcal{G}$.

Let ρ_0 be the committed qubit of the prover prior to measurement in the Hadamard round in the case that $h = 0$. Due to the argument above, the distribution $D_{\rho,0}$ is computationally indistinguishable from $D_{\rho_0,0}$. To address the second

difference, we now show that measuring ρ_0 in the standard basis produces the same distribution obtained from extracting the standard basis measurement from the commitment string y. First, note that measuring the committed qubit prior to application of U (i.e. at the start of the Hadamard round) results in the same measurement obtained from y; as seen in (9), the value of the committed qubit is equal to the value m extracted from y, since the prover passes the test round perfectly. To complete our proof, recall that U is X-trivial with respect to the committed qubit, and therefore commutes with standard basis measurement of the committed qubit.

To recap, the argument above shows that there exists a quantum state underlying the distribution $D_{\mathbb{P},h}$ as long as the prover's attack operator in the Hadamard round is an X-trivial operator. For a formal statement and complete proof of this argument, see the full version of this paper ([12]).

V. Replacement of a General Attack with an X-Trivial Attack

We can now proceed to the crux of this paper: assuming that $n = 1$ and $\mathbb{P}$ passes the test round perfectly, we show that there exists a prover $\mathbb{P}'$ such that $D_{\mathbb{P},h}$ is computationally indistinguishable from $D_{\mathbb{P}',h}$ for both h and $\mathbb{P}'$ attacks with an X-trivial operator in the Hadamard round. By the argument in Section IV-B and the triangle inequality, this implies that there exists a state ρ for which $D_{\mathbb{P},h}$ and $D_{\rho,h}$ are computationally indistinguishable, thereby proving our soundness guarantee.

Assume $\mathbb{P}$ is characterized by (U_0, U). Then $\mathbb{P}'$ is characterized by $(U_0, \{U_x\}_{x\in\{0,1\}})$, where $\{U_x\}_{x\in\{0,1\}}$ is an X-trivial CPTP map[1]:

$$U = \sum_{x,z\in\{0,1\}} X^x Z^z \otimes U_{xz} \tag{10}$$

$$U_x = \sum_{z\in\{0,1\}} Z^z \otimes U_{xz} \tag{11}$$

Observe that if $h = 0$, $D_{\mathbb{P},h} = D_{\mathbb{P}',h}$; this is simply because the standard basis measurement is obtained from the commitment y, which is measured prior to the Hadamard round attack U. This argument requires a bit more detail for $n > 1$ and is given in the full version ([12]). We proceed to describing how to replace the attack U in (10) with the CPTP map $\{U_x\}_{x\in\{0,1\}}$ in (11) in the case that the verifier chooses the Hadamard basis ($h = 1$). We will rely heavily on the structure of the prover's state, as written in (8).

The replacement of U with $\{U_x\}_{x\in\{0,1\}}$ will be done by using the Z Pauli twirl. The Z Pauli twirl is a technique which allows the replacement of U with the CPTP map $\{U_x\}_{x\in\{0,1\}}$ by conjugating U by a random Z Pauli operator. More formally, the Z Pauli twirl amounts to proving that the following two CPTP maps are equivalent when followed by Hadamard basis measurement:

$$\{\frac{1}{\sqrt{2}}(Z^r \otimes \mathcal{I})U(Z^r \otimes \mathcal{I})\}_{r\in\{0,1\}} \tag{12}$$

[1] U can be written in the form in (10) by decomposing its action on the first qubit in the Pauli basis; the matrix U_{xz} is not necessarily unitary.

$$\{U_x\}_{x\in\{0,1\}} \tag{13}$$

To apply the Z Pauli twirl in this setting, it suffices to show that replacing the prover's attack U with the unitary attack $(Z\otimes\mathcal{I})U(Z\otimes\mathcal{I})$ results in a computationally indistinguishable distribution.

To prove this statement, we will rely on the fact that there is already computational randomness, due to the trapdoor claw-free function, which is hiding both the Z operator applied prior to U and the Z operator applied after. The computational randomness hiding the posterior Z operator comes from the verifier's decoding operator $X^{d\cdot(x_{0,y}\oplus x_{1,y})}$ applied at the end of the measurement protocol (see Section III); if this decoding operator is shifted prior to the Hadamard transform on the committed qubit, it acts as a Z operator immediately after the attack U. The computational randomness hiding the anterior Z operator results from the format of the prover's state. Recall that, since the prover is perfect, we can assume the prover begins the Hadamard round with a state of the form in (8):

$$|\phi_y\rangle = \sum_{b\in\{0,1\}} |b\rangle |x_{b,y}\rangle \left|\psi_{b,x_{b,y}}\right\rangle \tag{14}$$

Intuitively, the prover's inability to determine the claw $(x_{0,y}, x_{1,y})$ prevents him from being able to distinguish whether or not a Z operator is applied to $|\phi_y\rangle$. More formally, we show in Section V-A2 that distinguishing between the states $|\phi_y\rangle$ and $(Z\otimes\mathcal{I})|\phi_y\rangle$ boils down to the ability to determine the bit $d\cdot(x_{0,y}\oplus x_{1,y})$ for an arbitrary string d.

In order to use these two sources of computational randomness to hide the difference between U and $(Z\otimes\mathcal{I})U(Z\otimes\mathcal{I})$, it must be the case that the bit $d\cdot(x_{0,y}\oplus x_{1,y})$ is computationally indistinguishable from a uniformly random bit. Formalizing this requirement is a bit tricky, since d is sampled from the state created by the prover. In the next section, we show how to prove computational indistinguishability between the distributions resulting from U and $(Z\otimes\mathcal{I})U(Z\otimes\mathcal{I})$. As part of this process, we formalize the computational randomness requirement regarding $d\cdot(x_{0,y}\oplus x_{1,y})$ as two different hardcore bit conditions for the function pair $f_{k,0}, f_{k,1}$.

A. Computational Indistinguishability of Phase Flip

Let $\mathbb{P}$ be the prover characterized by (U_0, U) and let $\hat{\mathbb{P}}$ be the prover characterized by $(U_0, (Z\otimes\mathcal{I})U(Z\otimes\mathcal{I}))$. In this section, we will show that the distributions resulting from the two provers ($D_{\mathbb{P},h}$ and $D_{\hat{\mathbb{P}},h}$) are computationally indistinguishable for all h. For convenience, we will instead refer to these two distributions as mixed states; let σ_0 be the mixed state corresponding to $D_{\mathbb{P},h}$ and let σ_1 be the mixed state corresponding to $D_{\hat{\mathbb{P}},h}$, i.e.

$$\sigma_0 = \sum_{m\in\{0,1\}} D_{\mathbb{P},h}(m) |m\rangle \langle m| \tag{15}$$

$$\sigma_1 = \sum_{m\in\{0,1\}} D_{\hat{\mathbb{P}},h}(m) |m\rangle \langle m| \tag{16}$$

To prove the computational indistinguishability of σ_0 and σ_1, each state is split into two terms (for $r \in \{0,1\}$):

$$\sigma_r = \sigma_r^D + \sigma_r^C \tag{17}$$

By a straightforward application of the triangle inequality, we obtain that if σ_0 is computationally distinguishable from σ_1, either σ_0^D and σ_1^D are computationally distinguishable or σ_0^C and σ_1^C are. Note that even if the terms are not quantum states, the notion of computational indistinguishability is still well defined: to show that two terms, for example σ_0^C and σ_1^C, are computationally indistinguishable, we need to show (informally) that there does not exist an efficiently computable CPTP map $\mathcal{S}$ such that the following expression is non negligible

$$|\mathrm{Tr}((|0\rangle\langle 0| \otimes \mathcal{I})\mathcal{S}(\sigma_0^C - \sigma_1^C)| \tag{18}$$

In more detail, the density matrices σ_0 and σ_1 are created by beginning with the state $|\phi_y\rangle$ in (14) and applying the operations of both the prover and verifier in the Hadamard round, followed by tracing out all but the first qubit. Therefore, to split σ_0 and σ_1 into two parts, we can equivalently split the density matrix of $|\phi_y\rangle$ into the following two parts, corresponding to the diagonal and cross terms:

$$\sum_{b\in\{0,1\}} |b\rangle\langle b| \otimes |x_{b,y}\rangle\langle x_{b,y}| \otimes \left|\psi_{b,x_{b,y}}\right\rangle\left\langle\psi_{b,x_{b,y}}\right| \tag{19}$$

$$\sum_{b\in\{0,1\}} |b\rangle\langle b\oplus 1| \otimes |x_{b,y}\rangle\langle x_{b\oplus 1,y}| \otimes \left|\psi_{b,x_{b,y}}\right\rangle\left\langle\psi_{b,x_{b\oplus 1,y}}\right| \tag{20}$$

Let σ_0^D and σ_1^D be the result of applying the operations of both the prover and the verifier in the Hadamard round to (19), followed by tracing out all but the first qubit. Recall the difference between σ_0^D and σ_1^D: in the latter, the prover's attack U is conjugated by $(Z \otimes \mathcal{I})$. Define σ_0^C and σ_1^C similarly, but replace (19) with (20). In the following two sections, we show that both pairs of terms are computationally indistinguishable.

1) Diagonal Terms: In this section, we will show that if there exists a BQP attacker $\mathcal{A}'$ who can distinguish between the terms σ_0^D and σ_1^D, then there exists a BQP attacker $\mathcal{A}$ who can violate the following informal hardcore bit property of the function family $\mathcal{F}$:

Claim V.1. *Assume $f_{k,0}$ and $f_{k,1}$ are sampled from a trapdoor claw-free family $\mathcal{F}$. Then there does not exist a BQP attacker who, on input k, can produce $b \in \{0,1\}$, $x_b \in \mathcal{X}$, $d \in \{0,1\}^w \setminus \{0^w\}$ and $c \in \{0,1\}$ such that $c = d \cdot (x_0 \oplus x_1)$ where $f_{k,0}(x_0) = f_{k,1}(x_1)$.*

We first describe the state σ_0^D, which is created by beginning with the state in (19), in more detail. Note that the state in (19) can be efficiently created by following the prover's commitment process and then measuring the committed qubit and preimage register. To create σ_0^D, the attack U is applied to the state in (19), followed by Hadamard measurement of the committed qubit and preimage register and application of the verifier's X decoding operator. Finally, all qubits but the first are traced out. σ_1^D is almost the same as σ_0^D, except the attack U is replaced with the attack $(Z \otimes \mathcal{I})U(Z \otimes \mathcal{I})$. Note that the initial phase operator has no effect, since it acts on the diagonal state in (19). The final phase flip, once it is shifted past the Hadamard transform, is equivalent to flipping the decoding bit of the verifier; it follows that $\sigma_1^D = X\sigma_0^D X$.

We now construct the BQP attacker $\mathcal{A}$ who will use $\mathcal{A}'$ to violate Claim V.1. Let σ_D be the state σ_0^D except for the verifier's decoding. Observe from the description in the previous paragraph that this state can be efficiently created, and as part of creating the state, the measurements $b, x_{b,y}$ and d are obtained. The attacker $\mathcal{A}$ creates the state σ_D. For the string d obtained by $\mathcal{A}$, the decoding bit $d \cdot (x_{0,y} \oplus x_{1,y})$ determines which of the two states σ_0^D and σ_1^D $\mathcal{A}$ has created; if $d \cdot (x_{0,y} \oplus x_{1,y}) = r$, $\mathcal{A}$ has created σ_r^D. Now $\mathcal{A}$ can run $\mathcal{A}'$ on the resulting mixed state in order to learn $d \cdot (x_{0,y} \oplus x_{1,y})$. As a result, $\mathcal{A}$ holds the following information: $b, x_{b,y}, d$, and $d \cdot (x_0 \oplus x_1)$, therefore violating Claim V.1.

2) Cross Terms: In this section, we will show that the cross terms σ_0^C and σ_1^C are computationally indistinguishable. Since the cross terms are not quantum states, we first show below that if there exists a CPTP map $\mathcal{S}$ which distinguishes between σ_0^C and σ_1^C, i.e. if the following expression is non negligible:

$$|\mathrm{Tr}((|0\rangle\langle 0| \otimes \mathcal{I})\mathcal{S}(\sigma_0^C - \sigma_1^C)| \tag{21}$$

then there exists an efficiently computable CPTP map $\mathcal{S}'$ such that the CPTP map $\mathcal{S}\mathcal{S}'$ distinguishes between the quantum states $\hat{\sigma}_0$ and $\hat{\sigma}_1$, defined as follows. The density matrix $\hat{\sigma}_r$ corresponds to the following pure state (recall $|\phi_y\rangle$ from (14)):

$$(Z^r \otimes \mathcal{I})|\phi_y\rangle = (Z^r \otimes \mathcal{I})\Big(\sum_{b\in\{0,1\}} |b\rangle|x_{b,y}\rangle\left|\psi_{b,x_{b,y}}\right\rangle\Big) \tag{22}$$

To do this, it suffices to show that $\sigma_0^C - \sigma_1^C = \mathcal{S}'(\hat{\sigma}_0 - \hat{\sigma}_1)$. This equality is straightforward for two reasons. First, $\frac{1}{2}(\hat{\sigma}_0 - \hat{\sigma}_1)$ is equal to the cross term in (20). Second, both σ_0^C and σ_1^C also begin with (20), but followed by a CPTP map which is inefficient due to the verifier's decoding. To prove the existence of $\mathcal{S}'$, we show that taking the difference between σ_0^C and σ_1^C effectively removes the verifier's decoding, creating an efficient CPTP map $\mathcal{S}'$.

Finally, we will show that an attacker who can distinguish between $\hat{\sigma}_0$ and $\hat{\sigma}_1$ can violate the following informal hardcore bit property of the function family $\mathcal{F}$:

Claim V.2. *Assume $f_{k,0}$ and $f_{k,1}$ are sampled from a trapdoor claw-free family $\mathcal{F}$. Then there exists $d \in \{0,1\}^w$ which satisfies two conditions. First, there exists a bit c_k such that $d \cdot (x_0 \oplus x_1) = c_k$ for all claws (x_0, x_1) ($f_{k,0}(x_0) = f_{k,1}(x_1)$). Second, there does not exist a BQP attacker who, on input k, can determine the bit c_k.*

We begin by describing the cross term σ_0^C (which is not a quantum state) in more detail. σ_0^C is created by beginning with the expression in (20), copied here for reference:

$$\sum_{b\in\{0,1\}} |b\rangle\langle b\oplus 1| \otimes |x_{b,y}\rangle\langle x_{b\oplus 1,y}| \otimes \left|\psi_{b,x_{b,y}}\right\rangle\left\langle\psi_{b,x_{b\oplus 1,y}}\right| \tag{23}$$

then applying the attack U, followed by Hadamard measurement of the committed qubit and preimage register and application of the verifier's X decoding operator. Finally, all qubits but the first are traced out. σ_1^C is almost the same, except the attack U is replaced with the attack $(Z \otimes \mathcal{I})U(Z \otimes \mathcal{I})$. As in Section V-A1, the phase flip acting after U is equivalent to flipping the decoding operator of the verifier (i.e. applying an X operator to the matrix σ_0^C). The initial phase flip, which acts on the first qubit of (23), results in a phase of -1. Combining these two observations yields the following equality:

$$\sigma_1^C = -X\sigma_0^C X \tag{24}$$

Taking the difference between σ_0^C and σ_1^C results in a matrix which has a uniform X operator applied:

$$\sigma_0^C - \sigma_1^C = \sum_{r\in\{0,1\}} X^r \sigma_0^C X^r \tag{25}$$

Observe that the CPTP map applied to (23) to create σ_0^C is efficiently computable except for the verifier's X decoding operator. In (25), there is a uniform X operator acting on σ_0^C, effectively replacing the verifier's decoding operator. Let $\mathcal{S}'$ be the resulting efficiently computable CPTP map. It follows immediately that $\sigma_0^C - \sigma_1^C = \mathcal{S}'(\hat{\sigma}_0 - \hat{\sigma}_1)$.

We now proceed to showing that an attacker $\mathcal{A}'$ who can distinguish between $\hat{\sigma}_0$ and $\hat{\sigma}_1$ can be used to violate Claim V.2. Since the state $\hat{\sigma}_r$ is the state $|\phi_y\rangle$ from (14) with the operator Z^r applied to the committed qubit, an attacker who can distinguish between $\hat{\sigma}_0$ and $\hat{\sigma}_1$ can distinguish whether or not a Z operator is applied to the committed qubit of $|\phi_y\rangle$. The following equality (which holds up to a global phase) shows that a Z operator on the preimage register is equivalent to a Z operator on the committed qubit:

$$\begin{aligned}&(\mathcal{I} \otimes Z^d \otimes \mathcal{I})(\sum_{b\in\{0,1\}} |b\rangle\,|x_{b,y}\rangle\,\left|\psi_{b,x_{b,y}}\right\rangle)\\ &= (Z^{d\cdot(x_{0,y}\oplus x_{1,y})} \otimes \mathcal{I})(\sum_{b\in\{0,1\}} |b\rangle\,|x_{b,y}\rangle\,\left|\psi_{b,x_{b,y}}\right\rangle)\end{aligned} \tag{26}$$

This equality, along with the attacker $\mathcal{A}'$, can be used to construct a BQP attacker $\mathcal{A}$ who can determine $d\cdot(x_{0,y}\oplus x_{1,y})$ for an arbitrary fixed string d. $\mathcal{A}$ first constructs $|\phi_y\rangle$ (this is simply the prover's state after reporting the commitment string y, so it can be constructed efficiently). Next, $\mathcal{A}$ applies Z^d to the preimage register of $|\phi_y\rangle$. Due to the equality in (26), this is equivalent to instead applying $Z^{d\cdot(x_{0,y}\oplus x_{1,y})}$ to the committed qubit. By running the attacker $\mathcal{A}'$, $\mathcal{A}$ can determine $d \cdot (x_{0,y} \oplus x_{1,y})$, therefore violating Claim V.2.

VI. Extension of Measurement Protocol to a Verification Protocol for BQP

Our goal is to verify that an instance $x \in L$ for a language $L \in$ BQP. Recall that each instance can be converted into a local Hamiltonian H with the following property: if $x \in L$, H has ground state energy at most a and if $x \notin L$, H has ground state energy at least b, where the gap $b - a$ is inverse polynomial. Therefore, to verify that an instance $x \in L$, a verifier with a quantum computer can simply ask the prover for the ground state and estimate the energy of the received state with respect to the Hamiltonian H. The soundness of such a protocol rests on the fact that if an instance $x \notin L$, all possible states sent by the prover will have energy $\geq b$.

To use such a verification procedure in our setting, we need to rely on one more fact: the Hamiltonian H can be written as a sum over terms which are each a product of X and Z operators [9]. Therefore, when the verifier is estimating the energy of a state sent by the prover, he only needs to perform Hadamard or standard basis measurements on each individual qubit. In [10], the authors formalize the resulting protocol and use it to build a protocol in which a verifier with access to a single qubit register can verify the result of a BQP computation. Their protocol achieves a completeness/ soundness gap which is negligibly close to 1 by performing polynomially many repetitions of the energy estimation process described above.

In [10], the prover sends single qubits to the verifier, who performs either Hadamard or standard basis measurements. To obtain a verification protocol for BQP, we simply replace this step of their protocol with our measurement protocol. Completeness and soundness follow, since our measurement protocol allows the verifier to collect standard and Hadamard basis measurements of a given state, and our soundness claim guarantees that the distribution over measurement results obtained by the verifier comes from the measurement of an underlying quantum state.

Acknowledgment

Thanks to Dorit Aharonov, Zvika Brakerski, Zeph Landau, Umesh Vazirani and Thomas Vidick for many useful discussions. The author is supported by Templeton Foundation Grant 52536, ARO Grant W911NF-12-1-0541, NSF Grant CCF-1410022 and MURI Grant FA9550-18-1-0161.

References

[1] D. Gottesman, 2004, as referenced in [http://www.scottaaronson.com/blog/?p=284; accessed 13-Apr-2017].

[2] A. Broadbent, J. F. Fitzsimons, and E. Kashefi, "Universal blind quantum computation," *Arxiv preprint arXiv:0807.4154*, 2008.

[3] J. F. Fitzsimons and E. Kashefi, "Unconditionally verifiable blind quantum computation," *Phys. Rev. A*, vol. 96, p. 012303, 07 2017. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevA.96.012303

[4] D. Aharonov, M. Ben-Or, and E. Eban, "Interactive Proofs For Quantum Computations," *Arxiv preprint arXiv:0810.5375*, 2008.

[5] D. Aharonov, M. Ben-Or, E. Eban, and U. Mahadev, "Interactive Proofs for Quantum Computations," *Arxiv preprint 1704.04487*, 2017.

[6] B. Reichardt, F. Unger, and U. Vazirani, "A classical leash for a quantum system," *Arxiv preprint arXiv:1209.0448*, 2012.

[7] O. Regev, "On lattices, learning with errors, random linear codes, and cryptography," in *Proceedings of the Thirty-seventh Annual ACM Symposium on Theory of Computing*, ser. STOC '05. New York, NY, USA: ACM, 2005, pp. 84–93.

[8] A. Kitaev, A. Shen, and M. Vyalyi, *Classical and Quantum Computation*. American Mathematical Society, 2002.

[9] J. D. Biamonte and P. J. Love, "Realizable hamiltonians for universal adiabatic quantum computers," *Phys. Rev. A*, vol. 78, p. 012352, 07 2008. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevA.78.012352

[10] T. Morimae and J. F. Fitzsimons, "Post hoc verification with a single prover," *Arxiv preprint arXiv:1603.06046*, 2016.

[11] Z. Brakerski, P. Christiano, U. Mahadev, U. Vazirani, and T. Vidick, "Certifiable randomness from a single quantum device," *Arxiv preprint 1804.00640*, 2018.
[12] U. Mahadev, "Classical verification of quantum computations," *Arxiv preprint 1804.01082*, 2018.
[13] ——, "Classical homomorphic encryption for quantum circuits," *Arxiv preprint 1708.02130*, 2017.