Classical proofs of quantum knowledge

Thomas Vidick[⋆] and Tina Zhang[⋆⋆]

California Institute of Technology

Abstract. We define the notion of a *proof of knowledge* in the setting where the verifier is classical, but the prover is quantum, and where the witness that the prover holds is in general a quantum state. We establish simple properties of our definition, including that, if a *non-destructive* classical proof of quantum knowledge exists for some state, then that state can be cloned by an unbounded adversary, and that, under certain conditions on the parameters in our definition, a proof of knowledge protocol for a hard-to-clone state can be used as a (destructive) quantum money verification protocol. In addition, we provide two examples of protocols (both inspired by private-key classical verification protocols for quantum money schemes) which we can show to be proofs of quantum knowledge under our definition. In so doing, we introduce techniques for the analysis of such protocols which build on results from the literature on nonlocal games. Finally, we show that, under our definition, the verification protocol introduced by Mahadev (FOCS 2018) is a classical *argument* of quantum knowledge for QMA relations. In all cases, we construct an explicit quantum extractor that is able to produce a quantum witness given black-box quantum (rewinding) access to the prover, the latter of which includes the ability to coherently execute the prover's black-box circuit controlled on a superposition of messages from the verifier.

1 Introduction

The notion of a *proof of knowledge* was first considered in the classical setting in [GMR89] and subsequently formalized in [TW87,FFS88] and [BG92].[1] Intuitively, a proof of knowledge protocol allows a prover to convince a verifier that it 'knows' or 'possesses' some piece of secret information (a 'witness', w) which satisfies a certain relation R relative to a publicly known piece of information x. (Symbolically, we might say that the prover wants to convince its verifier that, for a particular x, it knows w such that $R(x, w) = 1$.) For example, the witness w might be a private password corresponding to a particular public username

⋆ Department of Computing and Mathematical Sciences, California Institute of Technology, USA. vidick@caltech.edu

⋆⋆ Division of Physics, Mathematics and Astronomy, California Institute of Technology, USA. tinazhang@caltech.edu

[1] These three works give inequivalent definitions, but the differences are not important for the purpose of this introduction.

x, and a proof of knowledge protocol in this setting could allow the prover to demonstrate that it possesses the credentials to access sensitive information.

The formal definition of a classical proof of knowledge for NP relations R was settled in a series of works (see [BG92] for a summary) in the 1990s. The standard definition is as follows: the prover P is said to 'know' a witness w if there is an extractor E which, given black-box access to P (including the ability to rewind P and run it again on different messages from the verifier), can efficiently compute w. The applications of classical proofs of knowledge include identification protocols [FFS88], signature schemes [CL06], and encryption schemes secure against chosen-ciphertext attack [SJ00].

In this work, we consider a particular generalisation of the classical concept of a proof of knowledge to the quantum setting. We imagine a situation where the *verifier* remains classical, but the *prover* is quantum, and where the witness w is in general a quantum state; and we ask the prover to 'convince' the verifier that it knows that state. We call this type of protocol a *classical proof of quantum knowledge*. Recently, there have been works which show how a fully classical verifier can, under cryptographic assumptions, delegate a quantum computation on encrypted data to a quantum server [Mah18a], verify that such a server performed the computation correctly [Mah18b], delegate the preparation of single-qubit states to the server in a composable fashion [GV19], and test classically that the server prepared an EPR pair in its own registers [MV20]. In short, as long as classical computational resources and classical communication channels remain less expensive than their quantum counterparts, it will be natural to wish to use classical devices to test quantum functionality. Although we focus here on information-theoretic rather than computational security, the current paper can be considered part of the preceding line of work.

Quantum proofs of quantum knowledge (i.e. proof of knowledge protocols for quantum witnesses in which quantum interaction is allowed) have recently been explored by [BG19] and [CVZ19]; these two papers give a definition for quantum proofs of quantum knowledge, and exhibit several examples which meaningfully instantiate the definition. Here, we consider the more challenging question of defining and constructing proofs of knowledge for quantum witnesses in which the verifier and the interaction are *classical*. In this setting there is an interesting difficulty involved in constructing an extractor: how does one argue that a quantum prover 'knows' a certain quantum state if the only information which the prover 'reveals' is classical? A first approach, following the classical definition, would only allow the extractor to access classical transcripts from the protocol. Under such a restriction, the problem the extractor faces becomes one of reconstructing a witness ρ based entirely on classical measurement outcomes. It is not hard to convince ourselves that this problem probably has no solution for any non-trivial class of quantum states, as indeed it may be as hard as quantum state tomography [HHJ+17]. This observation makes it clear that we must allow the extractor to engage in some sort of quantum interaction with the prover.

Our first contribution in this paper is to provide an adequate definition of a proof of quantum knowledge for the setting where the communication between verifier and prover is classical. In order to circumvent the difficulty described in the preceding paragraph, we adopt a definition of 'black-box access to the prover' which is naturally suited to the quantum setting. Informally speaking, we model the prover as a unitary map U that acts on two quantum registers, one which is private (and which is used for storing its internal state) and one which is public (used for sending and receiving messages). In each round of the real protocol, the verifier places a classical message in the public 'message register', and the prover then runs the unitary U, before the message register is measured in the computational basis; the measurement result is the message that the prover sends to the verifier for that round. We define 'black-box access to the prover' as follows: we allow the extractor to place any quantum state in the public 'message register', as well as run the prover's unitary U, which acts on both registers, or its inverse $U^\dagger$; we do not allow the extractor to access the prover's private register except through U or $U^\dagger$. We do, however, allow it to place a coherent superposition of messages in the message register, even though the verifier (in a real protocol) would only ever put one classical message there. We make use of this latter possibility in our instantiations of this definition.

This definition matches the definition of 'black-box access to a quantum machine' used in previous works [Unr12]. We emphasise that, even though we consider protocols with purely classical communication, the extractor according to this definition of 'black-box access' is allowed to coherently manipulate a unitary implementation of the prover, and the message registers are not necessarily measured after each round of interaction. This possibility was allowed in [Unr12], but not used; here we make essential use of it when we construct our extractors. We note also that this definition of 'black-box access' matches the definition given in prior works (e.g. [Wat09]) of the 'black-box access' to a malicious verifier which a zero-knowledge simulator for a post-quantumly zero-knowledge proof of knowledge is allowed to have.

Having formalised what 'black-box access to the prover' means in our context, we move to the task of defining a 'proof of quantum knowledge' for our setting. We have two main applications in mind for a 'proof of quantum knowledge': one of them (proofs of knowledge for QMA witnesses) is natural given the standard formulation of classical proofs of knowledge for NP witnesses, but the other (proofs of knowledge for *quantum money states* [AFG+12]) is both natural and unique to the quantum setting. The quantum money application does not fit well into the standard formalism which is used for NP and QMA verification. Therefore, in order to formulate our definition of a 'proof of quantum knowledge' generally enough that we can capture both applications, we introduce a broader framework that mirrors frameworks recently introduced for similar purposes in the classical literature. Formally, we base our definition of a 'proof of quantum knowledge' on the notion of an 'agree-and-prove scheme' introduced recently in [BJM19]. The main innovation in this framework is that it allows the instance x and the proof relation R to be determined dynamically through interactions between the prover, the verifier, and possibly a trusted setup entity (such as the provider of a common random string or a random oracle). This framework lends itself remarkably well to our applications. Since we do not need all the possibilities that it allows, we introduce a somewhat simplified version which is sufficient for our purposes; details are given in Section 3.

In Section 4 we show two elementary but potentially interesting properties of our definition of a 'proof of quantum knowledge'. The first property is that, if a classical proof of quantum knowledge leaves the witness state intact, then the witness state can be cloned by an unbounded adversary. This is a simple no-go result which precludes certain types of proofs of quantum knowledge in the scenarios which we consider. The second property is that, under certain conditions on the parameters in the definition, a proof of knowledge protocol for a hard-to-clone witness state can also be used as a quantum money verification protocol. This result formalises the intuition that the property of being a 'proof of quantum knowledge' is stronger than the property of being a quantum money verification protocol: the latter implies that no adversary can pass verification twice given access to only one money bill, and the former formalises the notion that no adversary can pass even once unless it is possible to efficiently compute the money bill by interacting with said adversary.

Our second main contribution is to provide several examples of protocols which can be shown to be proofs of knowledge under our definition, and in so doing introduce some techniques that may possibly find use in the analysis of such protocols. As we have mentioned, instantiating a secure quantum money scheme is a natural application for a proof of quantum knowledge protocol. Conversely, quantum money verification protocols are natural candidates for examples of proofs of quantum knowledge: in a quantum money protocol, there is a prover who holds a purported money state, and who wishes to demonstrate to the verifier (who might be the bank or an independent citizen) that it does indeed 'hold' or 'possess' the quantum money state. The first person to describe quantum money was Wiesner [Wie83], who proposed money states that are tensor products of n qubits, each qubit of which is chosen uniformly at random from the set $\{|0\rangle, |1\rangle, |+\rangle, |-\rangle\}$. Wiesner's states can be described classically by $2n$ classical bits, and in a quantum money scheme this classical description is kept secret by the bank; a typical classical description is the pair of strings (x, θ), where the money state can be described (denoting by H_i a Hadamard gate on the ith qubit of the state and identities on all other qubits) as $|\$\rangle_{x,\theta} = \prod_i H_i^{\theta_i} |x\rangle$. We choose to analyse as our first example of a proof of knowledge a *private-key*, destructive classical money verification protocol between a prover and the bank for Wiesner's quantum money states which has been described previously in [MVW12]. The protocol is as follows: the verifier issues a uniformly random challenge string c to the prover, which encodes the bases (standard or Hadamard) in which the prover should measure the money state; the prover measures the ith qubit of the state in the standard basis if $c_i = 0$, or in the Hadamard basis if $c_i = 1$, and sends all the measurement outcomes as a string m to the verifier; and the verifier checks that, whenever $c_i = \theta_i$, $m_i = x_i$. The property which makes this protocol and these states interesting is that no prover who is given only one copy of the money state can pass verification twice.

Perhaps surprisingly, showing even that this simple protocol is a proof of knowledge according to our definition turns out to be a non-trivial task. We may examine the following illustration of the difficulty. Consider, firstly, the following naïve approach to designing an extractor for the protocol described in the preceding paragraph. Recall that, according to our model of 'black-box access', the prover can be considered a unitary process; we denote by U_c the unitary that the prover applies to its private register and the message register in response to challenge c. The extractor could pick a challenge c at random, apply U_c, and then attempt to apply some unitary to the message register to 'correct' for the challenge bases in order to recover the original money state. (For example, if $n = 4$, and $c = 0110$, the extractor could apply the unitary U_{0110}, and then apply $H_2 H_3$ to the message register in hopes of recovering the original money state. This strategy would work on the honest prover, who simply measures the real money state in the bases indicated by c in order to obtain its message to the verifier; we may imagine that few meaningful deviations from this pattern are possible.) However, the prover (upon receipt of the challenge) may take its honest money state and decide to apply Pauli X (bit-flip) gates to some arbitrary subset of the qubits of the money state which it was told to measure in the Hadamard basis, and Pauli Z (phase-flip) gates to a subset of the qubits which it was told to measure in the standard basis. If the prover now measures the result in the bases indicated by c, it will pass with probability 1—but the state that it measures in the c basis in this scenario is almost certainly not the correct money state. (The exception is when $c = \theta$.)

A little thought will show that this is a fairly general obstacle to the extractor's constructing the money state from the state residing in the prover's message registers immediately before it performs the measurement whose outcomes it will send to the verifier. Since we know very little about what the prover might be doing to the money state at any other stage in its execution, meanwhile, it is difficult to reason about finding the money state in the prover's registers at other points in its operation. This simple argument shows that, in order to design an effective extractor, it is crucial to consider the prover's responses to all challenges c at once—the question, of course, is one of how.

Our way of overcoming these difficulties builds on results from the literature on nonlocal games. The key idea of our security proof for the Wiesner money verification protocol is as follows. Let the party which chooses and prepares the money state $|\$\rangle_{x,\theta} = \prod_i H_i^{\theta_i} |x\rangle$ that the prover receives be known as Alice, and let the prover be known as Bob. Consider the following thought experiment: instead of preparing $|\$\rangle_{x,\theta}$, Alice could prepare n EPR pairs and send half of each one to Bob. Let $E(\theta) = \{|\$\rangle\langle\$|_{x,\theta} \mid x \in \{0,1\}^n\}$ be a general measurement (POVM). Then, if Alice measures $E(\theta)$ on her side of the state, and obtains the outcome x, Alice's and Bob's joint state will collapse to two copies of $|\$\rangle_{x,\theta}$. Note that, from Bob's perspective, the protocol is the same regardless of whether Alice sent EPR pairs and then measured $E(\theta)$, or whether she chose x and θ uniformly at random and sent him $|\$\rangle_{x,\theta}$ to begin with. However, if Bob succeeds with high probability in the money verification protocol, then he also succeeds with high probability at recovering a subset of the string x which represents Alice's measurement outcomes after she measures the POVM $E(\theta)$, and which also forms part of the classical description of the money state $|\$\rangle_{x,\theta}$. This observation makes it possible to apply a theorem from [NV16] which states that, if two noncommunicating parties exhibit correlations like those which Alice and Bob exhibit in this thought experiment, then they must once have shared EPR pairs, up to local isometry. Since Alice is honest and did nothing to her shares of the EPR pairs, the local isometry on her side is the identity map. Then, in order to recover the original money state, the proof-of-knowledge extractor simply has to execute the correct isometry on Bob's side. This isometry can be implemented efficiently using only black-box access to the prover; this step, however, crucially makes use of the fact that the extractor can implement controlled versions of the prover's unitaries on a superposition of messages of its choice. A detailed analysis is given in section 5.1.

Although the efficacy of this technique for showing that a protocol is a proof of knowledge depends strongly on the structure of the Wiesner verification protocol, we are also able to apply it to one other example. Wiesner states were the earliest and are the best-known kind of quantum money states, but there are other kinds, and one sort which has received some recent attention is the class of *subspace states* introduced in a quantum money context by [AC12]. Subspace states are states of the form $\frac{1}{\sqrt{|A|}}\sum_{x\in A}|x\rangle$ for some $n/2$-dimensional subspace $A \in \mathbb{Z}_2^n$, and they have similar no-cloning properties to those of Wiesner states; they are also of additional interest because they have been used in several schemes which make steps toward the goal of public-key quantum money [AC12], [Zha19], and in constructions of other quantum-cryptographic primitives such as quantum signing tokens [BDS16]. We were not able to find a simple classical verification protocol for subspace states that we could show to be a proof of quantum knowledge. Nonetheless, in Section 5.2, we propose a classical (private-key) verification protocol for what we call *one-time-padded subspace states* (that is, subspace states which have had random Pauli one-time-pads applied to them by the bank), and we are able to show under our new definition, using similar techniques to those which we applied to Wiesner states, that this simple verification protocol is a proof of knowledge for one-time-padded subspace states. This verification protocol is remarkable for having a challenge from the verifier that is only one bit long.

Our final contribution is to show that, under our definition, a classical *argument* of quantum knowledge exists for any relation in the class QMA.[2] The notion of a *QMA relation* was formalised jointly by [BG19] and [CVZ19], as a quantum analogue to the idea of an *NP relation* which was described in the first paragraphs of this introduction. [BG19] and [CVZ19] show that any QMA relation has a *quantum* proof of quantum knowledge. The protocol that we show

[2] Argument systems differ from proof systems only in that the honest prover must be efficient, and that soundness is required to hold only against efficient provers. In this case, 'efficient' means quantum polynomial-time.

to be a *classical* argument of quantum knowledge for QMA relations, meanwhile, is the classical verification protocol introduced recently by [Mah18b]. Mahadev [Mah18b] shows, under cryptographic assumptions, that quantum properties (in her case, any language in BQP) can be decided by a classical polynomial-time verifier through classical interaction alone with a quantum polynomial-time prover. We note that the proofs of the main results in [Mah18b] include statements which can be used to make the verification protocol which [Mah18b] introduces into a classical argument of quantum knowledge in the sense in which we have defined the latter. The main work that needs to be done in order to show this is to establish that the quantum witness, which as shown in [Mah18b] always *exists* for the case of a successful prover, can be *extracted* from the prover in a black-box manner. While all the required technical components for establishing this are already present in [Mah18b], we make the statement explicit. (In comparison, our proofs that specific quantum money schemes satisfy our definition of a proof of quantum knowledge do not use any cryptographic assumptions, and the protocols which we consider are very simple compared with the [Mah18b] protocol.) The [Mah18b] verification protocol can be shown to be an argument of quantum knowledge for any QMA relation; the only caveat, which was also a caveat for the *quantum* proofs of quantum knowledge for QMA exhibited by [BG19] and [CVZ19], is that an honest prover in the protocol may require multiple copies of a witness in order that the extractor can succeed in extracting *one* copy. We refer the reader to Section 6 for details.

Acknowledgements. We thank Alexandru Gheorghiu for useful feedback and Or Sattath for comments. Thomas Vidick is supported by NSF CAREER Grant CCF-1553477, AFOSR YIP award number FA9550-16-1-0495, MURI Grant FA9550-18-1-0161 and the IQIM, an NSF Physics Frontiers Center (NSF Grant PHY-1125565) with support of the Gordon and Betty Moore Foundation (GBMF-12500028). This material is based upon work supported by DARPA under Agreement No. HR00112020023. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the United States Government or DARPA.

2 Preliminaries

2.1 Terminology and notation

Due to space constraints, we refer the reader to the full version [VZ21] for basic notation and terminology.

2.2 Black-box quantum provers

Due to space constraints, we refer the reader to the full version [VZ21] for a discussion of the definitions we will now present relating to the notion of 'black-box access'. Formally, we use a similar framework to that which is described in [Unr12, Section 2.1] in order to capture black-box access to quantum provers. The following definitions of *interactive quantum machines* and *oracle access* to an interactive quantum machine are taken (with some modifications) from [Unr12]; a similar formulation of these definitions of [Unr12] appears in [CVZ19]. The modifications which we introduce are primarily for convenience in dealing with the situation where the verifier is known to be classical, instead of (potentially) quantum as it is in [Unr12].

Remark 1. Even though the possibility is not used in [Unr12], the framework presented there explicitly allows the extractor to coherently implement controlled versions of the prover's unitaries on a superposition of messages from the verifier. As we argued in the introduction, it is necessary to give this power to the extractor in our context (see e.g. [BCC+20] for impossibility results in related settings).

Interactive quantum machines. An *interactive quantum machine* is a machine M with two quantum registers: a register S for its internal state, and a register N for sending and receiving messages (the network register). Upon activation, M expects in N a message, and in S the state at the end of the previous activation. At the end of the current activation, N contains the outgoing message of M, and S contains the new internal state of M. A machine M gets as input: a security parameter $\lambda \in \mathbb{N}$, a classical input $x \in \{0,1\}^*$, and a quantum input $|\Phi\rangle$, which is stored in S. Formally, machine M is specified by a family of unitary circuits $\{M_{\lambda x}\}_{\lambda \in \mathbb{N}, x \in \{0,1\}^*}$ and a family of integers $\{r^M_{\lambda x}\}_{\lambda \in \mathbb{N}, x \in \{0,1\}^*}$. $M_{\lambda x}$ is the quantum circuit that M performs on the registers S and N upon invocation. $r_{\lambda x}$ determines the total number of messages/invocations. We might omit writing the security parameter and/or the classical input x when they are clear from the context. We say that M is *quantum-polynomial-time* (QPT for short) if the circuit $M_{\lambda x}$ has size polynomial in $\lambda + |x|$, the description of the circuit is computable in deterministic polynomial time in $\lambda + |x|$ given λ and x, and $r_{\lambda,x}$ is polynomially bounded in λ and x.

Oracle access to an interactive quantum machine. We say that a quantum algorithm A has oracle access to an interactive quantum machine M (with internal register S and network register N) *running on* $|\Phi\rangle$ to mean the following. We initialise S to $|\Phi\rangle$ and N to $|0\rangle$, we give A the security parameter λ and its own classical input x, and we allow A to execute (a controlled version of) the quantum circuit $M_{\lambda x'}$ (for any x') specifying M, and (a controlled version of) its inverse (recall that these act on the internal register S and on the network register N of M). Moreover, we allow A to provide and read messages from M (formally, we allow A to act freely on the network register N). We do not allow A to act on the internal register S of M, except via $M_{\lambda x'}$ or its inverse.

Interactive classical machines. An *interactive classical machine* is a machine C with two classical registers: a register T for its internal state, and a register N for sending and receiving messages (the network register). Upon activation, C expects in N a message, and in T the state at the end of the previous activation.

At the end of the current activation, N contains the outgoing message of C, and T contains the new internal state of M. A machine C gets as input: a security parameter $\lambda \in \mathbb{N}$, a classical input $x \in \{0,1\}^*$, a random input $u \in \{0,1\}^{p(\lambda+|x|)}$ for some function $p \in \mathbb{N}$, and a classical auxiliary input $t \in \{0,1\}^{|\mathsf{T}|}$, which is stored in T. Formally, machine C is specified by a function $p \in \mathbb{N}$, a family of classical circuits $\{C_{\lambda x u}\}_{\lambda \in \mathbb{N}, x \in \{0,1\}^*, u \in \{0,1\}^{p(\lambda+|x|)}}$ and a family of integers $\{r^C_{\lambda x}\}_{\lambda \in \mathbb{N}, x \in \{0,1\}^*}$. $C_{\lambda x u}$ is the classical circuit that C performs on the registers T and N upon invocation. Without loss of generality, for convenience's sake, we assume that $C_{\lambda x u}$ is reversible. $r^C_{\lambda x}$ determines the total number of messages/invocations. We might omit writing the security parameter and/or the input when they are clear from the context. We say that C is *probabilistic-polynomial-time* (PPT for short) if p is a polynomial, the circuit $C_{\lambda x u}$ has size polynomial in $\lambda + |x|$, the description of the circuit is computable in deterministic polynomial time in $\lambda + |x|$ given λ, x and u, and $r^C_{\lambda x}$ is polynomially bounded in λ and x.

Oracle access to an interactive classical machine We say that a quantum algorithm A has oracle access to an interactive classical machine C *running on string* t to mean the following. We initialise C's internal register T to t and the network register N to the all-zero string. We give A the security parameter and its own classical input x. Each time A wishes to run C (or its inverse), it must submit an input x' on which to run C (or its inverse). Upon receiving A's choice of x', we choose u uniformly at random, and then we run the classical circuit $C_{\lambda x' u}$ (or its inverse); recall that these act on the internal register T and on the network register N of C. Moreover, we allow A to provide and read messages from C (formally, we allow A to act freely on the network register N). We do not allow A to act on the internal register T of C, except via $C_{\lambda x' u}$ or its inverse.

Definition 1. We use the terminology *interactive Turing machine* (ITM) to refer to either an interactive classical machine or an interactive quantum machine. If the ITM is bounded-time, we may refer to a PPT ITM or a QPT ITM to clarify which model is used. An interactive oracle machine is an ITM that in addition has query access to an oracle.

Interaction between an interactive quantum machine and an interactive classical machine Let $M = (\{M_{\lambda x}\}, \{r^M_{\lambda x}\})$ be an interactive quantum machine with internal register S and network register N. Let $C = (\{p, C_{\lambda x' u}\}, \{r^C_{\lambda x'}\})$ be an interactive classical machine with internal register T and network register N. For a given CQ state $\rho_{\mathsf{TS}} \in \mathrm{D}(\mathcal{H}_{\mathsf{T}} \otimes \mathcal{H}_{\mathsf{S}})$, we define the interaction $(C(x'), M(x))_{\rho_{\mathsf{TS}}}$ as the following quantum process: initialize register N to $|0\rangle$; initialise registers S and T to the CQ state ρ_{TS}; alternately apply $M_{\lambda x}$ to registers S and N and $C_{\lambda x' u}$ (for a uniformly chosen $u \in \{0,1\}^{p(\lambda+|x'|)}$ each time) to registers T and N, measuring N in the computational basis after each application of either $M_{\lambda x}$ or $C_{\lambda x' u}$; stop applying $M_{\lambda x}$ after $r^M_{\lambda x}$ times and $C_{\lambda x' u}$ after $r^C_{\lambda x'}$ times, and finally output the output of the circuit $C_{\lambda x' u}$. We denote the random variable representing this output by $\langle C(x'), M(x)\rangle_{\rho_{\mathsf{TS}}}$. We call the $r^M_{\lambda x} + r^C_{\lambda x'}$ measurement outcomes which are obtained after performing as many standard basis measurements of N during a single execution of the interaction $(C(x'), M(x))_{\rho_{\mathsf{TS}}}$ the *transcript* for this execution.

2.3 Implementing oracles

Some of our formal definitions rely on 'oracles', which we generally visualise as functions $\mathcal{O} : \{0,1\}^* \to \{0,1\}^*$ to which query access is given. We refer the reader to the full version [VZ21] for some brief remarks on how query access to these oracles (which, expressed as functions, may take an exponential number of bits to specify) can be implemented efficiently in the number of queries made to the oracle, and also on our assumption that any query submitted to an oracle is measured in the standard basis before being answered.

3 Quantum Agree-and-Prove schemes

To define the intuitive notion of a 'proof of quantum knowledge' in sufficient generality so that we can capture both quantum money verification and QMA verification we introduce a quantum variant of the 'agree-and-prove' framework from [BJM19], extending their formalism to our setting in which the prover and the witness are quantum, and simplifying some aspects of the formalism that are less important for the applications we have in mind. For convenience, we preserve much of the notation from [BJM19]. The reader might wish to consult the full version of this paper [VZ20] for a discussion of the intuition behind this agree-and-prove framework; the reader can also refer to [BJM19] for additional motivation and explanations relating to the framework.

In the next subsection we formalise the notion of a scenario. The following section discusses input generation algorithms; the one after that formalises protocols, and the one after that lays down the security conditions for agree-and-prove schemes.

3.1 Scenario

Definition 2 (Agree-and-Prove Scenario for quantum relations). An agree-and-prove (AaP for short) scenario for quantum relations is a triple $(\mathcal{F}, \mathcal{R}, \mathcal{C})$ of interactive oracle machines satisfying the following conditions:

– The *setup functionality* $\mathcal{F}$ is a QPT ITM taking a unary encoding of a security parameter λ as input. The ITM $\mathcal{F}$ runs an initialization procedure $\boldsymbol{init}$, and in addition returns the specification of an oracle (which we also model as an ITM) $\mathcal{O}_{\mathcal{F}}(i, \boldsymbol{q}, arg)$. The oracle function takes three arguments: $i \in \{I, P, V\}$ denotes a 'role', $\boldsymbol{q}$ denotes a keyword specifying a query type, and arg denotes the argument for the query.[3]

[3] In [BJM19], $\mathcal{O}_{\mathcal{F}}$ has an additional function: when it is called with the argument `QUERIES`, $\mathcal{O}_{\mathcal{F}}$(`QUERIES`) returns a list of tuples representing all of the queries made

There are three different options for the 'role' parameter, which exists to allow $\mathcal{F}$ to release information selectively depending on the party asking for it. The roles I, P and V correspond respectively to the *input generator* (Definition 3), the prover, and the verifier.

– The *agreement relation* $\mathcal{C}$ is a QPT oracle machine taking a unary encoding of the security parameter λ and a statement as inputs, and producing a decision bit as output.[4]
– The *proof relation* $\mathcal{R}$ is a QPT oracle machine taking a unary encoding of the security parameter λ, a (classical) statement x and a (quantum) witness ρ_{W} as inputs, and outputting a decision bit.

3.2 Input generation

Before we formalise the notion of an agree-and-prove protocol, we introduce the notion of an *input generation algorithm*, which is an algorithm that produces the auxiliary inputs that the prover and the verifier receive before they begin interacting. The input generation algorithm models 'prior knowledge' which the prover and the verifier may possess. For a fuller discussion of the motivation for the input generation algorithm, please see the full version [VZ21].

Definition 3 (Input Generation Algorithm). An input generation algorithm I for an agree-and-prove scenario $\mathcal{S}$ is a machine I taking a unary encoding of the security parameter λ as input and producing a CQ state $\rho_{\mathsf{AUX}_V \mathsf{AUX}_P}$ specifying the auxiliary inputs for the verifier (in the classical register AUX_V) and prover (in the quantum register AUX_P) respectively as output. We may use the shorthand $\rho_{\mathsf{AUX}_P} \equiv \mathrm{Tr}_{\mathsf{AUX}_V}\left(\rho_{\mathsf{AUX}_V \mathsf{AUX}_P}\right)$ and $\rho_{\mathsf{AUX}_V} \equiv \mathrm{Tr}_{\mathsf{AUX}_P}\left(\rho_{\mathsf{AUX}_V \mathsf{AUX}_P}\right)$.

3.3 Protocol

Once a scenario has been fixed we can define a *protocol* for that scenario. Informally, the protocol specifies the actions of the honest parties. Each party, prover and verifier, is decomposed into two entities that correspond to the two phases, "agree" and "prove", of the protocol.

Definition 4 (Agree-and-prove protocol). An agree-and-prove protocol is a tuple $(\mathcal{I}, P_1, P_2, V_1, V_2)$ consisting of a set $\mathcal{I}$ of input generation algorithms together with the following four interactive oracle machines (P_1, P_2, V_1, V_2):

to $\mathcal{O}_{\mathcal{F}}$ by the prover P and the replies that were given. This functionality is available only to the extractor, not to the parties I, P and V, and it is necessary in order to permit the design of an efficient extractor for some protocols, particularly those in the random oracle model (see, for example, the discussion at the bottom of page 10 in [BJM19]). Since we do not need to use this functionality in our protocols, we omit it here.

[4] In [BJM19] the agreement relation also takes two auxiliary inputs. We will not need this.

- A (honest) first phase QPT prover P_1 taking a unary encoding of the security parameter λ and a (quantum) auxiliary input ρ_{AUX_P} as inputs. It produces a (classical) statement x_P or $\perp$ as output, as well as a (quantum) state ρ_{st_P}.
- A (honest) first phase PPT verifier V_1 taking a unary encoding of the security parameter λ and a (classical) auxiliary input AUX_V as inputs. It produces a (classical) statement x_V or $\perp$ as output, as well as a (classical) state st_V.
- A (honest) second phase QPT prover P_2 taking a classical instance x and a quantum state ρ_{st_P} as input, as well as a unary encoding of the security parameter λ, and producing as output a bit that indicates whether the proof has been accepted.
- A (honest) second phase PPT verifier V_2 taking a classical instance x and a state string st_V as input, as well as a unary encoding of the security parameter λ, and producing as output a bit that indicates whether it accepts or rejects.

Note that in this definition the verifier is required to be a classical probabilistic polynomial time ITM. In general one may extend the definition to allow for quantum polynomial time verifiers; since our focus is on classical protocols we restrict our attention to classical verifiers. We also restrict the honest prover to run in quantum polynomial time; for soundness, this restriction will be lifted for the case of proofs of knowledge and maintained for the case of arguments of knowledge.

3.4 Security conditions

We now specify the correctness and soundness conditions associated with an agree-and-prove scenario $\mathcal{S}$.

Definition 5 (Completeness experiment). We define the following *completeness experiment* for an agree-and-prove protocol $\mathcal{K} = (\mathcal{I}, P_1, P_2, V_1, V_2)$ in the context of a scenario $\mathcal{S} = (\mathcal{F}, \mathcal{C}, \mathcal{R})$:

1. An input generation algorithm $I \in \mathcal{I}$ is executed. It is allowed to query $\mathcal{O}_{\mathcal{F}}(I, \cdot, \cdot)$. It produces the CQ state $\rho_{\mathsf{AUX}_V \mathsf{AUX}_P}$, and passes input ρ_{AUX_P} to P_1 and ρ_{AUX_V} to V_1.
2. The interaction $(V_1, P_1)_{\rho_{\mathsf{AUX}_V \mathsf{AUX}_P}}$ is executed (during which V_1 and P_1 are allowed to query $\mathcal{O}_{\mathcal{F}}(V, \cdot, \cdot)$ and $\mathcal{O}_{\mathcal{F}}(P, \cdot, \cdot)$, respectively), and if either V_1 or P_1 returns $\perp$, or if $x_V \neq x_P$, the agree phase returns 0. Otherwise, the outputs of V_1 and P_1 are passed to V_2 and P_2, respectively, and the agree phase returns 1. If the agree phase returns 1, let the CQ state representing the joint distribution of st_V and ρ_{st_P} be denoted by $\rho_{st_V\, st_P}$, and let $x = x_P = x_V$ be the instance that V_1 and P_1 have agreed on.
3. The interaction $(V_2(x), P_2(x))_{\rho_{st_V\, st_P}}$ is executed (during which V_2 and P_2 are allowed to query $\mathcal{O}_{\mathcal{F}}(V, \cdot, \cdot)$ and $\mathcal{O}_{\mathcal{F}}(P, \cdot, \cdot)$, respectively), and the outcome of the proof phase is set to the value which V_2 returns at the end of the protocol.

The completeness experiment returns 1 if the agree phase and the proof phase both return 1.

Definition 6 (Soundness experiment). We define the following *soundness experiment* for an agree-and-prove protocol $\mathcal{K} = (\mathcal{I}, P_1, P_2, V_1, V_2)$ and an extractor E, in the context of a scenario $\mathcal{S} = (\mathcal{F}, \mathcal{C}, \mathcal{R})$:

1. An input generation algorithm $\hat{I}$ is executed. It is allowed to query $\mathcal{O}_{\mathcal{F}}(I, \cdot, \cdot)$. It produces the CQ state $\rho_{\mathsf{AUX}_V \mathsf{AUX}_P}$, and passes input ρ_{AUX_P} to $\hat{P}_1$ and ρ_{AUX_V} to V_1.
2. The interaction $(V_1, \hat{P}_1)_{\rho_{\mathsf{AUX}_V \mathsf{AUX}_P}}$ is executed (during which V_1 and $\hat{P}_1$ are allowed to query $\mathcal{O}_{\mathcal{F}}(V, \cdot, \cdot)$ and $\mathcal{O}_{\mathcal{F}}(P, \cdot, \cdot)$, respectively), and if either V_1 or P_1 returns $\perp$, or if $x_V \neq x_P$, the agree phase returns 0. Otherwise, the outputs of V_1 and $\hat{P}_1$ are passed to V_2 and $\hat{P}_2$, respectively, and the agree phase returns 1. If the agree phase returns 1, let the CQ state representing the joint distribution of st_V and ρ_{st_P} be denoted by $\rho_{st_V\, st_P}$, and let $x = x_P = x_V$ be the instance that V_1 and $\hat{P}_1$ have agreed on.
3. If the agree phase returns 1 in step 2, the extractor E is provided with the transcript of the interaction $(V_1, \hat{P}_1)_{\rho_{\mathsf{AUX}_V \mathsf{AUX}_P}}$ and the instance x resulting from the agree phase, along with oracle access to $\hat{P}_2$ running on input ρ_{st_P} (where ρ_{st_P} is the prover's half of the joint CQ state $\rho_{st_V\, st_P}$). In addition the extractor can access the oracle $\mathcal{O}_{\mathcal{F}}$ using any of the roles in $\{I, P\}$. It outputs a state ρ.

We are now ready to give the formal definition of security.

Definition 7 (Security of Protocol for Quantum Agree-and-Prove Scenario). Let λ be a security parameter. Let $c, \kappa, \delta : \mathbb{N} \to [0,1]$. A protocol $\mathcal{K} = (\mathcal{I}, P_1, V_1, P_2, V_2)$ for a scenario $(\mathcal{F}, \mathcal{C}, \mathcal{R})$ is secure with completeness c, up to knowledge error κ, and with extraction distance parameter δ if the following conditions hold:

- *Correctness:* The completeness experiment (Definition 5) returns 1 with probability at least c, and in addition the statement $x = x_V = x_P$ that is agreed on during the completeness experiment is such that $\mathcal{C}(1^\lambda, x) = 1$, whenever the honest parties P and V are provided with their inputs by some input generation algorithm $I \in \mathcal{I}$. [5]
- *Soundness:* There exists a QPT ITM E (called the "extractor") such that the following holds. Let $\hat{P} = (\hat{P}_1, \hat{P}_2)$ be a potentially dishonest prover for $\mathcal{K}$ and $\hat{I}$ an arbitrary input generation algorithm. Let x be an instance such that, conditioned on the agree phase of $\mathcal{K}$ returning 1 and the instance x being agreed upon, the prover $\hat{P}_2$ succeeds with probability $p > \kappa$ in the proof phase of $\mathcal{K}$. Then the state ρ returned by the extractor in the soundness experiment

[5] Note that, for completeness, we require that the input generation algorithm is chosen from a set $\mathcal{I}$ of 'honest' algorithms. Here we depart from [BJM19], where input generation is always unrestricted (even when the verifier and the prover are honest). We refer the reader to the full version [VZ21] for a fuller discussion of this subject.

(Definition 6), conditioned on the agree phase of the soundness experiment returning 1 and x being agreed on, is such that $\Pr[\mathcal{R}(1^\lambda, x, \rho) = 1] > 1 - \delta(p)$, where δ, which may depend on λ, is such that $\delta(p) < 1$ for all $p > \kappa$. The expected number of steps of extractor E is required to be bounded by a polynomial in $\lambda/(p - \kappa)$, if executing the prover's unitary on any input counts as a unit-time procedure.

When the soundness condition only holds under the restriction that $\hat{P}$ must be implemented by a QPT ITM we say that the protocol is *computationally secure*, or that it is an *argument system* (as opposed to a *proof system*, which is sound against all possible provers).

Remark 2. When we wish to emphasize the connection between secure agree-and-prove protocols and the more usual notion of a 'proof of knowledge', we sometimes refer to an AaP scenario that satisfies Definition 7 as a 'classical proof (or argument) of quantum knowledge'. (Formally, proofs and arguments of knowledge can be formulated as protocols for AaP scenarios which have trivial agreement phases and which have as a proof relation an NP or a QMA relation; see Section 6.) When we use this terminology, it will be clear from context what the 'knowledge' is that we are referring to.

3.5 Agree-and-Prove scenario for quantum money

As an example of a concrete agree-and-prove scenario, we define an agree-and-prove scenario that captures the scenario which arises in the problem of verifying quantum money. We firstly lay down the 'standard' security definitions for a quantum money scheme, and in so doing introduce some notation and some objects that will be useful in formulating quantum money in the agree-and-prove framework.

Definition 8. A "quantum money scheme" is specified by the following objects, each of which is parametrized by a security parameter λ:

- A algorithm Bank taking a string r as a parameter which initialises a database of valid *money bills* in the form of a table of tuples $(\mathsf{id}, \mathsf{public}, \mathsf{secret}, |\$\rangle_{\mathsf{id}})$. id represents a unique identifier for a particular money bill; public and secret represent, respectively, public and secret information that may be necessary to run the verification procedure for the bill labeled by id; and $|\$\rangle_{\mathsf{id}}$ is the quantum money state associated with the identifier id. The string r should determine a classical map H_r such that $H_r(\mathsf{id}) = (\mathsf{public}, \mathsf{secret})$.[6]
- A verification procedure $\mathit{Ver}(x, \mathsf{public}, \mathsf{secret}, \rho_W)$ that is a QPT algorithm which decides when a bill is valid.

In addition the scheme should satisfy the following conditions:

[6] The string r represents any random choices that Bank might make while generating valid bills; we make this string explicit for later convenience.

1. Completeness: for any valid money bill $(\mathsf{id}, \mathsf{public}, \mathsf{secret}, |\$\rangle_{\mathsf{id}})$ in the database created by *Bank*,

$$\Pr\big(\mathit{Ver}(\mathsf{id}, \mathsf{public}, \mathsf{secret}, |\$\rangle\langle\$|_{\mathsf{id}})\big) \geq c_M(\lambda) ,$$

for some function $c_M(\cdot)$. We refer to c_M as the *completeness parameter* of the money scheme.
2. No-cloning: Consider the following game played between a challenger and an adversary: the challenger selects a valid money bill $(\mathsf{id}, \mathsf{public}, \mathsf{secret}, |\$\rangle_{\mathsf{id}})$ and sends $(\mathsf{id}, \mathsf{public}, |\$\rangle_{\mathsf{id}})$ to the adversary; the adversary produces a state σ_{AB}. Then for any adversary in this game,[7]

$$\Pr_r\big(\mathit{Ver}(\mathsf{id}, \mathsf{public}, \mathsf{secret}, \mathrm{Tr}_B(\sigma_{AB})) = 1 \\ \text{and} \quad \mathit{Ver}(\mathsf{id}, \mathsf{public}, \mathsf{secret}, \mathrm{Tr}_A(\sigma_{AB})) = 1\big) \leq \mu_M(\lambda) ,$$

for some function $\mu_M(\cdot)$. We refer to μ_M as the *cloning parameter* of the money scheme. Note that the probability of the adversary's success is calculated assuming that the string r which *Bank* takes is chosen uniformly at random.

Fix a quantum money scheme according to Definition 8, with completeness parameter c_M and cloning parameter μ_M. We call an agree-and-prove scenario $(\mathcal{F}_M, \mathcal{C}_M, \mathcal{R}_M)$ that takes the form below a 'quantum money scenario with completeness parameter c_M and cloning parameter μ_M'.

– Setup functionality $\mathcal{F}_M(1^\lambda)$: The setup should run an initialization procedure init_M that instantiates[8] a database B_M whose records are of the form (and the distribution) that *Bank* would have produced running on a uniformly random input r. The setup should also return a specification of how the following oracles should be implemented:
  - $\mathcal{O}_{\mathcal{F}_M}(I, \mathit{id})$: returns an identifier id such that the bill $(\mathsf{id}, \mathsf{public}, \mathsf{secret}, |\$\rangle_{\mathsf{id}})$ is in B_M.[9]
  - $\mathcal{O}_{\mathcal{F}_M}(\cdot, \mathit{public}, \mathsf{id})$: Returns the public string associated with id. Returns $\perp$ if no record in B_M with the identifier id exists.
  - $\mathcal{O}_{\mathcal{F}_M}(I, \mathit{getMoney}, \mathsf{id})$: If no record in B_M with identifier id exists, returns $\perp$. Otherwise, returns $|\$\rangle_{\mathsf{id}}$ the first time it is called. If called again with the same id argument, returns $\perp$.

[7] Many quantum money schemes are information-theoretically secure; however, it is also possible to consider computationally secure schemes by replacing 'any' with 'any QPT'.

[8] init_M doesn't necessarily need to actually allocate memory for the database; since the database will only ever be accessed through the oracle $\mathcal{O}_{\mathcal{F}_M}$, it is possible to 'instantiate' the database using the method described in Section 2.3.

[9] Which identifier is returned is at the discretion of any particular instantiation of this function. Intuitively, this oracle is used to represent identifiers of bills that have been generated in the past and are thus available in an "environment" that I may have access to.

  - $\mathcal{O}_{\mathcal{F}_M}(V, \mathtt{secret}, \mathsf{id})$: accesses B_M and returns the secret string associated with id. Returns $\perp$ if no record in B_M with the identifier id exists.
- Agreement relation $\mathcal{C}^{\mathcal{O}_{\mathcal{F}_M}}(1^\lambda, \mathsf{id})$: outputs 1 if and only if a record in B_M with identifier id exists.
- Proof relation $\mathcal{R}^{\mathcal{O}_{\mathcal{F}_M}}(1^\lambda, x, \rho_W)$: interprets x as an id (outputting $\perp$ if this fails), sets $\mathsf{public} \leftarrow \mathcal{O}_{\mathcal{F}_M}(V, \mathtt{public}, x)$ and $\mathsf{secret} \leftarrow \mathcal{O}_{\mathcal{F}_M}(V, \mathtt{secret}, x)$, and executes $\mathit{Ver}(x, \mathsf{public}, \mathsf{secret}, \rho_W)$.

4 Simple properties

4.1 Nondestructive proofs of quantum knowledge imply cloning

In this section we formalize the intuitive claim that a non-destructive proof of quantum knowledge implies the ability to clone the underlying witness state. To formalize this statement we make a number of assumptions that help simplify the presentation. More general statements can be proven depending on one's needs; see the end of the section for further discussion.

We use definitions and notation from Section 2.2 and Section 3.

Definition 9 (Nondestructive interaction). Let $P = (\{P_{\lambda x}\}, \{r^P_{\lambda x}\})$ be an interactive quantum machine, and let $V = (p, \{V_{\lambda x u}\}, \{r^V_{\lambda x}\})$ be an interactive classical machine. Fix a security parameter λ. A *nondestructive interaction* $(V(x), P(x'))_{\rho_{\mathsf{TS}}}$ between V and P for some CQ state ρ_{TS} is an interaction in which the execution of $(V(x), P(x'))_{\rho_{\mathsf{TS}}}$ is unitary (including the standard-basis measurements of the network register that take place during the execution) for all possible random inputs u to V. More formally, for any choice of $r^V_{\lambda x}$ random strings $u_1, \dots, u_{r^V_{\lambda x}}$ used during the interaction $(V(x), P(x'))_{\rho_{\mathsf{TS}}}$, there exists a unitary U acting on registers N, T and S such that the joint state of the registers N, T and S is identical after U has been applied to them (assuming they are initialised as described in Section 2.2) to their joint state after the execution of $(V(x), P(x'))_{\rho_{\mathsf{TS}}}$ using the random strings $u_1, \dots, u_{r^V_{\lambda x}}$.

Definition 10 (Oracle access to an interactive quantum machine with power of initialisation). Recall the definition of oracle access to an interactive quantum machine given in Section 2.2. In that section, the initial state $|\Phi\rangle$ on which the quantum machine is run is fixed. We say that a quantum algorithm A has *oracle access to an interactive quantum machine M with power of initialisation* if A can do all the things described in Section 2.2, and in addition can initialise M's internal register S to a state of its choosing (but cannot read S, only write to it).

Proposition 1. *Let λ be a security parameter, let $(\mathcal{F}, \mathcal{C}, \mathcal{R})$ be an agree-and-prove scenario, and let $\mathcal{K} = (\mathcal{I}, P_1, P_2, V_1, V_2)$ be a protocol for $(\mathcal{F}, \mathcal{C}, \mathcal{R})$ with a classical honest verifier $V = (V_1, V_2)$, knowledge error κ and extraction distance δ. Let $\hat{P} = (\hat{P}_1, \hat{P}_2)$ be a prover for $\mathcal{K}$.*

Let $\hat{I}$ be any input generation algorithm, and x and ρ_{TS} an instance and a CQ state respectively such that the agree phase of $\mathcal{K}$, executed with $\hat{I}$, V_1 and $\hat{P}_1$, *has positive probability of ending with x being agreed on, and such that the joint state of st_V and st_P conditioned on x being agreed on is ρ_{TS}.*

Suppose further that (i) the interaction $\Big(V_2(x), \hat{P}_2(x)\Big)_{\rho_{\mathsf{TS}}}$ is nondestructive, (ii) the oracle $\mathcal{O}_{\mathcal{F}}$ does not keep state during the second phase of the protocol, i.e. any query to it by V_2 or $\hat{P}_2$ can be repeated with the same input-output behavior, and (iii) the success probability of $\hat{P}_2$ conditioned on instance x being agreed on is at least κ. Then there exists a procedure A [10] *such that the following holds.*

Let $\mathcal{R}^{\mathcal{O}_{\mathcal{F}}}_{\lambda x}(\cdot)$ be the function such that $\mathcal{R}^{\mathcal{O}_{\mathcal{F}}}_{\lambda x}(\rho) = \mathcal{R}^{\mathcal{O}_{\mathcal{F}}}(1^\lambda, x, \rho)$, and let the single-bit-valued function $\big(\mathcal{R}^{\mathcal{O}_{\mathcal{F}}}_{\lambda x}\big)^{\otimes 2}(\cdot)$ be the function whose output is the AND *of the outcomes obtained by executing the tensor product of two copies of $\mathcal{R}^{\mathcal{O}_{\mathcal{F}}}_{\lambda x}(\cdot)$ on the state that is given as argument. Then the procedure A, given as input x, a copy of a communication transcript from the agree phase that led to x, and black-box access to V_2 and $\hat{P}_2$ as interactive machines (including any calls they might make to $\mathcal{O}_{\mathcal{F}}$) running on ρ_{TS}, with power of initialisation for $\hat{P}_2$, can produce a state σ such that*

$$\Pr[\big(\mathcal{R}^{\mathcal{O}_{\mathcal{F}}}_{\lambda x}\big)^{\otimes 2}(\sigma) = 1] > 1 - 2\delta - \mathsf{negl}(\lambda). \tag{1}$$

Proof. Due to space constraints, we refer the reader to the full version [VZ21] for the proof.

Discussion. Due to space constraints, we refer the reader to the full version [VZ21] for a discussion of potential extensions of Proposition 1, including to the case where the protocol is not perfectly nondestructive but only 'slightly destructive', and to the case where computationally efficient cloning might be desirable.

4.2 Proofs of quantum knowledge are also quantum money verification protocols

The other simple property which we prove is that, under certain assumptions on the parameters in Definition 7, any protocol satisfying Definition 7 can be used as a quantum money verification protocol. Proposition 2 formalises the intuition that the property of being a 'proof of quantum knowledge' is stronger than the property of being a quantum money verification protocol: the latter implies that no adversary can pass verification twice given access to only one money bill, and the former formalises the notion that no adversary can pass even once unless it is possible to efficiently compute the money bill by interacting with said adversary.

[10] A is in general not efficient. It is also allowed slightly more invasive access to $\hat{P}_2$ than a typical extractor. This is acceptable because A is not an extractor, but a cloning procedure. We refer the reader to the full version [VZ21] for a fuller discussion of this topic.

Formalising interactive quantum money verification. Before we state Proposition 2, we must formalise what it means to 'be a quantum money verification protocol'. The standard definition of quantum money security (Definition 8) indicates what this means for a *passive* verification procedure, in which the verification procedure `Ver` is just an isometric map, but we need to formalise what it means for an *interactive* protocol. Due to space constraints, we refer the reader to the full version [VZ21] for a fuller motivation of the definition that we state below, and in particular of the no-communication assumption between provers $\hat{P}_A$ and $\hat{P}_B$.

Definition 11 (Interactive quantum money verification procedure). Let λ be a security parameter, and let $(\mathcal{F}_M, \mathcal{C}_M, \mathcal{R}_M)$ be a quantum money scenario (as defined in Section 3.5). A protocol $\mathcal{K} = (I, P_1, P_2, V_1, V_2)$ for $(\mathcal{F}_M, \mathcal{C}_M, \mathcal{R}_M)$ (see Definition 7) is an *interactive verification procedure* with completeness c and cloning error s for the quantum money scenario $(\mathcal{F}_M, \mathcal{C}_M, \mathcal{R}_M)$ if the following two conditions hold.[11] (Probabilities in these conditions are calculated assuming that r, the randomness that ***Bank*** takes as input, is drawn from the uniform distribution. See Definition 8 for a definition of `Bank`.)

1. Completeness: The protocol $\mathcal{K}$ has completeness c according to Definition 7.
2. Soundness: let $\hat{P}_A = (\hat{P}_{A,1}, \hat{P}_{A,2})$ and $\hat{P}_B = (\hat{P}_{B,1}, \hat{P}_{B,2})$ be two provers for $\mathcal{K}$, and let $\hat{I}$ be an algorithm that generates inputs for both of them. We define a no-cloning game as follows:
   (a) $\hat{I}$ prepares a (potentially entangled) joint state ρ_{AB}. During this phase, $\hat{I}$ is allowed to call the oracle $\mathcal{O}_{\mathcal{F}_M}$ using the role I. At the end of this phase, $\hat{I}$ gives $\rho_A = \mathrm{Tr}_B(\rho_{AB})$ to $\hat{P}_A$, and $\rho_B = \mathrm{Tr}_A(\rho_{AB})$ to $\hat{P}_B$.
   (b) Holding ρ_A, $\hat{P}_A$ executes $\mathcal{K}$ with a copy of the honest verifier of $\mathcal{K}$, the latter of which we denote by $V_A = (V_{A,1}, V_{A,2})$. Likewise, holding ρ_B, $\hat{P}_B$ also executes $\mathcal{K}$ with a copy of the honest verifier of $\mathcal{K}$, which we denote by $V_B = (V_{B,1}, V_{B,2})$. During the protocol executions, $\hat{P}_A$ and $\hat{P}_B$ are not allowed to communicate, but they are allowed to call the oracle $\mathcal{O}_{\mathcal{F}}$ using the role P.
   (c) $\hat{P}_A$ and $\hat{P}_B$ win the game if and only if the same instance x is agreed upon in the agree phases of both copies of $\mathcal{K}$ played in step 2, and in addition both V_A and V_B output 1 at the end of the game.

   We say that the protocol $\mathcal{K}$ for the quantum money scenario $(\mathcal{F}_M, \mathcal{C}_M, \mathcal{R}_M)$ is *secure against cloning* with cloning error s if any pair of provers $(\hat{P}_A, \hat{P}_B)$ with any input generation algorithm $\hat{I}$ wins the no-cloning game with probability less than s.

[11] This definition is distinct from the definition of security of a protocol $\mathcal{K}$ described in Definition 7. The latter is a security definition that can apply to any AaP scenario, and the present definition is a new definition tailored to quantum money that is a natural extension of the standard "no-cloning"-based definition recalled in Section 8. Our aim in this section, in fact, is to show that (qualitatively speaking) Definition 7 implies Definition 11.

We are now ready to formally present our lemma which captures the fact that a secure agree-and-prove protocol can be used as a quantum money verification procedure. We refer the reader to the full version [VZ21] for an exposition of the parameters that appear in Proposition 2.

Proposition 2. *Let λ be a security parameter, and let $(\mathcal{F}_M, \mathcal{C}_M, \mathcal{R}_M)$ be a quantum money scenario (as defined in Section 3.5). Let $\mathcal{K} = (I, P_1, P_2, V_1, V_2)$ be a protocol for a quantum money agree-and-prove scenario $(\mathcal{F}_M, \mathcal{C}_M, \mathcal{R}_M)$. Let μ_M be the cloning parameter for the quantum money scenario $(\mathcal{F}_M, \mathcal{C}_M, \mathcal{R}_M)$.*

Define $\delta_0 \equiv \frac{2-\sqrt{3}}{2}$. Suppose there is a function $\kappa(\cdot)$ such that $\mathcal{K}$ is a $(c = 1-\mathsf{negl}(\lambda), \delta)$-secure protocol with knowledge error $\kappa(\lambda)$ and extraction distance δ (the latter of which we assume is a function of the prover's success probability p as well as the security parameter λ) such that $\delta_0 - \delta(p(\lambda), \lambda) > \frac{1}{2}\frac{\mu_M(\lambda)}{\epsilon \cdot \kappa(\lambda)}$ for arbitrary $\epsilon > 0$ and sufficiently large λ whenever p is a function such that $p(\lambda) > \kappa(\lambda)$ for sufficiently large λ.

Then $\mathcal{K}$ is an interactive quantum money verification protocol for the money scenario $(\mathcal{F}_M, \mathcal{C}_M, \mathcal{R}_M)$ (in the sense defined in Definition 11) with completeness $1 - \mathsf{negl}(\lambda)$ and cloning error $(1 + \epsilon)\kappa$.

Proof. Due to space constraints, we refer the reader to the full version [VZ21] for the proof.

Amplification. The bound on the maximum success probability of a cloning adversary which comes out of Proposition 2 is linear in the knowledge error of the agree-and-prove protocol which is being used as a money verification protocol. A typical expectation in a quantum money scenario is that any cloning adversary will only succeed with negligible probability (see Definition 11 of [AC12], for example), but in our analyses of quantum money verification protocols in Sections 5.1 and 5.2, we only get constant (and not negligible) knowledge error. Therefore, in the full version [VZ21], we present a sequential amplification lemma which shows that a money scheme equipped with a classical agree-and-prove protocol that has constant knowledge error (and other parameters similar to those which we obtain in Sections 5.1 and 5.2) can be modified into a money scheme which admits only cloning adversaries that pass with negligible probability.

5 Proofs of quantum knowledge for quantum money states

In this section we apply our notion of proofs of quantum knowledge to the problem of certifying quantum money. We give two examples for two protocols from the literature, Wiesner's quantum money in Section 5.1 and Aaronson and Christiano's public-key quantum money based on hidden subspaces in Section 5.2.

5.1 PoQK for Wiesner money states

Our first concrete example of an Agree-and-Prove scheme for a quantum property is a verification protocol for Wiesner's quantum money states. Any Wiesner state can be described by 2λ classical bits; a typical classical description is the pair of strings $\$ = (v, \theta) \in \{0,1\}^{2\lambda}$, where the associated money state is

$$|\$\rangle_{v,\theta} = \Big(\prod_i H_i^{\theta_i}\Big) |v\rangle \;, \tag{2}$$

in which $|v\rangle = \otimes_i |v_i\rangle$ and H_i denotes a Hadamard on the ith qubit and identities on all other qubits. In the notation of Definition 8, valid bills in this scheme are quadruples $(\mathsf{id}, \mathsf{public}, \mathsf{secret}, |\$\rangle_{\mathsf{id}})$ such that id is an arbitrary string, public is empty, $\mathsf{secret} = (v,\theta) \in \{0,1\}^\lambda \times \{0,1\}^\lambda$ and $|\$\rangle_{\mathsf{id}} = |\$\rangle_{v,\theta}$. The verification procedure $\mathit{Ver}(x, \mathsf{public}, \mathsf{secret}, \rho_W)$ parses $\mathsf{secret} = (v, \theta)$ and measures each qubit of ρ_W in the basis indicated by θ. It accepts if and only if the outcomes obtained match v. This scheme clearly has completeness parameter 1, and it was shown in [MVW12] that its cloning parameter is $\mu_W(\lambda) = (3/4)^\lambda$.

Scenario 12 *We instantiate the generic AaP scenario for quantum money described in Section 3.5 as follows:*

- *Setup functionality $\mathcal{F}_W(1^\lambda)$:*
  - $\mathtt{init}_W$ *initializes a random oracle H taking strings of length 2λ to strings of length 2λ.*[12] *In addition, it initializes an empty database B_W that is destined to contain a record of all quantum money bills in circulation.*
  - $\mathcal{O}_{\mathcal{F}_W}(I, \mathtt{getId})$*: generates* $\mathsf{id} \in \{0,1\}^{2\lambda}$ *uniformly at random. Sets $(v,\theta) = H(\mathsf{id})$,* $\mathsf{secret} = (v,\theta)$ *and $|\$\rangle = |\$\rangle_{v,\theta}$. If* id *already appears in B_W, then returns $\perp$. Otherwise, add* $(\mathsf{id}, \mathsf{public}, \mathsf{secret}, |\$\rangle_{v,\theta})$ *to B_W. Return* id.
  - $\mathcal{O}_{\mathcal{F}_W}(\cdot, \mathtt{public}, \mathsf{id})$, $\mathcal{O}_{\mathcal{F}_W}(I, \mathtt{getMoney}, \mathsf{id})$, *and* $\mathcal{O}_{\mathcal{F}_W}(V, \mathtt{secret}, \mathsf{id})$*: as described in Section 3.5.*
- *Agreement relation $\mathcal{C}_W^{\mathcal{O}_{\mathcal{F}_W}}(1^\lambda, \mathsf{id})$: The agreement relation is the same as it is in Section 3.5.*
- *Proof relation $\mathcal{R}_W^{\mathcal{O}_{\mathcal{F}_W}}(1^\lambda, x, \rho_W)$: The proof relation firstly queries $\mathcal{O}_{\mathcal{F}_W}(V, \mathtt{secret}, x)$ in order to get a tuple (v,θ). (If $\mathcal{O}_{\mathcal{F}}(V, \mathtt{secret}, x)$ returns $\perp$, then $\mathcal{R}$ rejects.) Then it implements the Wiesner money verification procedure: it applies $\prod_i H_i^{\theta_i}$ to its quantum input ρ_W, measures all qubits in the computational basis, and accepts if and only if the outcome is v.*

Protocol 13 *We define our proof of knowledge protocol $\mathcal{K}_W = (\mathcal{I}_W, P_1, P_2, V_1, V_2)$ for the scenario $(\mathcal{F}_W, \mathcal{C}_W, \mathcal{R}_W)$. An honest input generation algorithm $I \in \mathcal{I}_W$ calls $\mathcal{O}_{\mathcal{F}_W}(I, \mathtt{getId})$ repeatedly until it obtains a string* $\mathsf{id} \in \{0,1\}^{2\lambda}$ *such that* $\mathsf{id} \neq \perp$*. It then queries $\mathcal{O}_{\mathcal{F}}(I, \mathtt{getMoney}, \mathsf{id})$, obtains a quantum state ρ_W, and gives (id, ρ_W) to the prover (it gives nothing to the verifier). In the agreement phase, the prover P_1 parses the auxiliary input ρ_{AUX_P} which it gets from I as a classical string* $\mathsf{id} \in \{0,1\}^{2\lambda}$ *in addition to a quantum state ρ_W. (If this fails,*

[12] Formally the oracle is implemented in the standard way, recalled in Section 2.3.

the prover halts.) Then the prover sends id *to* V_1 *and outputs the statement* $x_P = \mathsf{id}$ *and the quantum state* $\rho_{st_P} = \rho_W$*.* V_1*, upon receiving* id *from* P_1*, queries* $\mathcal{O}_{\mathcal{F}_W}(V, \texttt{public}, \mathsf{id})$*. If this returns* $\perp$ *the verifier aborts. Otherwise,* V_1 *outputs* $x_V = \mathsf{id}$ *and* $st_V = \perp$*.*

This completes the description of the (honest) prover and verifier in the first (agree) phase. We now describe the interaction between the (honest) prover and verifier, P_2 *and* V_2*, in the second (proof) phase:*

1. V_2 *queries* $\mathcal{O}_{\mathcal{F}_W}(V, \texttt{secret}, \mathsf{id})$*. If it obtains* $\perp$*,* V_2 *aborts. Otherwise, let* $\$ = (v, \theta)$ *be the classical description obtained.*
2. V_2 *sends a uniformly random* $c \in \{0,1\}^\lambda$ *to the prover.*
3. *For each* $i \in \{1, \ldots, n\}$ *if the ith bit of* c *is 0,* P_2 *measures the ith qubit of* ρ_{st_P} *in the standard basis; and if it is 1, it measures the ith qubit in the Hadamard basis. Let* $\beta \in \{0,1\}^\lambda$ *denote the outcomes obtained.* P_2 *sends* β *to* V_2*.*
4. *Let* $s = c \cdot \theta + \bar{c} \cdot \bar{\theta}$*, where* $\cdot$ *denotes componentwise multiplication. In other words,* $s_i = 1$ *if and only if* $c_i = \theta_i$*.* V_2 *checks that, whenever* $s_i = 1$*, it holds that* $v_i = \beta_i$*. If not, then it returns* 0*.*

Lemma 1. *There is a constant* $\kappa < 1$ *such that Protocol* $\mathcal{K}_W$ *(Protocol 13) is a secure agree-and-prove protocol for* $(\mathcal{F}_W, \mathcal{C}_W, \mathcal{R}_W)$ *(Scenario 12) with completeness* 1*, knowledge error* κ*, and extraction distance* $\delta = O(\mu^{1/4})$*, where* $\mu = 1 - p$ *and* p *is the prover's success probability.*

Proof. Due to space constraints, we refer the reader to the full version [VZ21] for the full proof. For intuition, we provide below a sketch of the main step, the design of the extraction procedure.

Our first step is to argue that we can replace $\mathcal{F}_W$ from Scenario 12 with a new setup functionality $\mathcal{F}'_W$ such that the prover is (perfectly) unable to distinguish the two. $\mathcal{F}_W$ and $\mathcal{F}'_W$ differ mainly in their implementations of $\mathcal{O}_{\mathcal{F}}(I, \texttt{getMoney}, \mathsf{id})$: while $\mathcal{F}_W$ chooses v and θ uniformly and returns a money state $|\$\rangle_{v,\theta}$ to the prover when $\texttt{getMoney}$ is called, $\mathcal{F}'_W$ returns *half EPR pairs* to the prover. (The number of such half-pairs is λ, the length of the money state.) $\mathcal{F}'_W$ keeps the halves of the EPR pairs that it does not give to the prover in a register A. Then, when the verifier calls $\mathcal{O}_{\mathcal{F}'_W}(V, \texttt{secret}, \mathsf{id})$ in step 1 of Protocol 13, $\mathcal{F}'_W$ chooses a basis string $\theta \in \{0,1\}^\lambda$ uniformly and *measures* the state in A in the bases determined by θ. This action collapses the state in A to the state $|\$\rangle_{v,\theta}$ for some uniformly random $v \in \{0,1\}^\lambda$. For convenience, we will refer to a version of Scenario 12 with $\mathcal{F}_W$ replaced with $\mathcal{F}'_W$ as 'the purified Scenario 12'.

We then use the prover's unitary ($M_{\lambda x'}$ in Section 2.2, which the extractor has black-box access to) in order to define a set of $2 \cdot 2^\lambda$ measurement operators $X^B(s), Z^B(s)$ for $s \in \{0,1\}^\lambda$ that act on the prover's private space P as well as the message register M. We design $X^B(s), Z^B(s)$ so that, for any $c, \theta \in \{0,1\}^\lambda$, the outcome of measuring $Z^B(c \cdot \theta)$ is equal to the single bit $\oplus_{i:c_i=\theta_i=0}\beta_i$, and likewise the outcome of measuring $X^B(\bar{c} \cdot \bar{\theta})$ is equal to $\oplus_{i:c_i=\theta_i=1}\beta_i$ (given that the prover's private state is initialised the way that it is at the start of the prove stage of Protocol 13 in the purified Scenario 12). [13]

We define corresponding measurement operators $X^A(s), Z^A(s)$ which act on the register A, and which simply act as $\sigma_X(s), \sigma_Z(s)$ on the A register (where $\sigma_Z(s) \equiv \bigotimes_i \sigma_{Z,i}^{s_i}$, with $\sigma_{Z,i}$ representing σ_Z on the ith qubit, and $\sigma_X(s)$ is defined analogously). Recall the verifier's check in step 4 of Protocol 13. We argue that, if the verifier's check passes, the outcomes of measuring $Z^A(c\cdot\theta)$ and $Z^B(c\cdot\theta)$ (on the registers on which they are respectively defined) are equal, and likewise the outcomes of measuring $X^A(\bar{c}\cdot\bar{\theta})$ and $X^B(\bar{c}\cdot\bar{\theta})$ are equal. [14]

We then apply a theorem similar to [NV16, Theorem 14] which states that, if we can define operators $X^A(s), Z^A(s), X^B(s), Z^B(s)$ for all $s \in \{0,1\}^\lambda$ satisfying certain conditions (which we satisfy), and if we can show that, for some state $|\psi\rangle_{AB}$, $\langle\psi|_{AB}\, Z^A(c\cdot\theta) \otimes Z^B(c\cdot\theta)\,|\psi\rangle_{AB} = 1$ and $\langle\psi|_{AB}\, X^A(\bar{c}\cdot\bar{\theta}) \otimes X^B(\bar{c}\cdot\bar{\theta})\,|\psi\rangle_{AB} = 1$ with high probability over uniformly chosen c, θ (in the previous paragraph we argued that these relations hold when $|\psi\rangle_{AB}$ is the joint state of registers $\mathsf{A}, \mathsf{M}, \mathsf{P}$ at the start of the prove phase of Protocol 13 in the purified Scenario 12, with $A = \mathsf{A}$ and $B = \mathsf{MP}$), then there is a local isometry of the form $\Phi^A \otimes \Phi^B$ which, applied to $|\psi\rangle_{AB}$, transforms $|\psi\rangle_{AB}$ (approximately) into shared EPR pairs between A and B. In our case, this means that we can recover the shared entanglement which initially existed between registers A and PM due to the EPR pairs which $\mathcal{F}'_W$ created and shared with the prover. In our case, it is also true that Φ^A is the identity map. We show in the full version of this proof that this conclusion about the purified Scenario 12 implies that we can recover the money state up to some error in the real Scenario 12 by applying Φ^B to the registers P and M, and also that the extractor can apply Φ^B efficiently using black-box access to the prover. (This step uses the fact that the extractor can execute the prover's unitary coherently on a message register which has been initialized in a quantum superposition.)

5.2 PoQK for subspace money states

Our second example of a proof of quantum knowledge protocol is a verification protocol for a modification of Aaronson and Christiano's *subspace states* [AC12]. Aaronson and Christiano present a quantum money scheme in which a λ-qubit money state, with $\lambda \in \mathbb{N}$ a security parameter, is specified by a (secret) $(\lambda/2)$-dimensional subspace $A \subseteq \mathbb{Z}_2^\lambda$, and defined as $|A\rangle = \frac{1}{\sqrt{|A|}}\sum_{x\in A}|x\rangle$. In the notation of Definition 8, valid bills in this scheme are quadruples $(\mathsf{id}, \mathsf{public}, \mathsf{secret}, |\$\rangle_{\mathsf{id}})$ such that id is an arbitrary string, public is empty[15], $\mathsf{secret} = \mathcal{Z} = \{z_1, \ldots, z_{\lambda/2}\}$

[13] Formally, we mean that $Z^B(c\cdot\theta)$ and $X^B(\bar{c}\cdot\bar{\theta})$ both commute with the measurement that produces β when $\mathcal{F}'_W$'s choice of basis string is θ and when the verifier's choice of challenge is c, and that performing the measurement which produces β and computing $\oplus_{i:c_i=\theta_i=0}\beta_i$ (resp. $\oplus_{i:c_i=\theta_i=1}\beta_i$) always gives the same outcome as measuring $Z^B(c\cdot\theta)$ (resp. $X^B(\bar{c}\cdot\bar{\theta})$).

[14] The reader should feel free to check that this holds given the previous paragraph.

[15] What we describe here is actually a *private-key* version of the Aaronson-Christiano scheme, equipped with a verification procedure which is similar to a verification pro-

is a basis for a $(\lambda/2)$-dimensional subspace A of $\mathbb{Z}_2^\lambda$, and $|\$\rangle_{\mathsf{id}} = |A\rangle$. One possible (quantum-verifier) verification procedure $\mathit{Ver}(x, \mathsf{public}, \mathsf{secret}, \rho_W)$ for these bills parses $\mathsf{secret} = \mathcal{Z}$ and then performs the projective measurement $H^{\otimes\lambda}\mathbb{P}_{A^\perp}H^{\otimes\lambda}\mathbb{P}_A$ on ρ_W (where $\mathbb{P}_A$ is a projection onto all standard basis strings in A, i.e. $\mathbb{P}_A = \sum_{x\in A}|x\rangle\langle x|$, and $\mathbb{P}_{A^\perp}$ is a projection onto all standard basis strings in $A^\perp$), and accepts if and only if the outcome is 1. The scheme (when equipped with this verification procedure) has completeness parameter 1, and it was shown in [AC12] that its cloning parameter is $\mu_{AC}(\lambda) \le c^\lambda$ for some constant $c < 1$. [16]

As we mentioned in the introduction, we do not know if it is possible to devise a natural proof of quantum knowledge for the Aaronson-Christiano subspace states as they have thus far been described. Nonetheless, we are able to give a proof of knowledge for a version of the subspace scheme in which a (secret) quantum one-time pad has been applied to every subspace state.

Notation. Before we define the associated AaP scenario, we introduce some notation:

– Let $|\$\rangle_{v,\theta}$ be a Wiesner money state representing the string v encoded in bases θ, as in (2).
– Let $\{s_i : i \in \{1,\dots,\lambda\}\} = \{100...0, 010...0, 001...0, \dots, 000...1\}$ be the standard basis for $\mathbb{Z}_2^\lambda$.
– Let $\mathcal{Z} = \{z_i : i \in \{1,\dots,\lambda\}\}$ be a basis for $\mathbb{Z}_2^\lambda$.
– Let W be the unitary on $(\mathbb{C}^2)^{\otimes\lambda}$ defined as follows:

$$\begin{aligned} W : W|x\rangle &= W|x_1 s_1 + \dots + x_\lambda s_\lambda\rangle \\ &= |x_1 z_1 + \dots + x_\lambda z_\lambda\rangle \ . \end{aligned} \tag{3}$$

– Let L_θ for a string $\theta \in \{0,1\}^\lambda$ be the subspace of $\mathbb{Z}_2^\lambda$ whose elements are always 0 in the positions where $\theta_i = 0$, and can be either 0 or 1 in the positions where $\theta_i = 1$.
– Let $X(a)$ for some vector $a = (a_1,\dots,a_\lambda) \in \mathbb{Z}_2^\lambda$ denote the tensor product of λ single-qubit gates which is Pauli X in those positions i where $a_i = 1$, and I otherwise. Define $Z(b)$ similarly. Let $X_{\mathcal{Z}}(d)$, for a basis $\mathcal{Z} = \{z_j\}$, denote the operator

$$\prod_j \left(X(z_j)\right)^{d_j} \ ,$$

where z_j denotes a particular vector from the basis set $\mathcal{Z}$, and d_j denotes the jth bit of d. Define $Z_{\mathcal{Z}}(e)$ similarly.

cedure used in [BDS16]. Aaronson and Christiano originally proposed this subspace scheme with the idea of making progress towards public-key quantum money. As such, in their original scheme, public is not empty.

[16] In fact Aaronson and Christiano show the stronger result that this bound holds even if the adversary is given black-box access to a pair of measurement operators that respectively implement projections on A and $A^\perp$.

– Let

$$|\$\rangle_{v,\theta,\mathcal{Z}} \equiv \frac{1}{\sqrt{|L_\theta|}} \sum_{\lambda \in L_\theta} X_{\mathcal{Z}}(d) Z_{\mathcal{Z}}(e) \, |\lambda_1 z_1 + \cdots + \lambda_n z_n\rangle \ , \tag{4}$$

with $d_i = v_i$ for i such that $\theta_i = 0$ (and $d_i = 0$ for all other i), and $e_i = v_i$ for i such that $\theta_i = 1$ (and $e_i = 0$ for all other i). Note that the distribution of $|\$\rangle_{v,\theta,\mathcal{Z}}$ over uniform $v, \theta, \mathcal{Z}$ is identical (ignoring global phase) to that of a uniformly random subspace state with a uniformly random Pauli one-time-pad applied to it, because the coordinates of d and e which we have forced to be zero (instead of uniformly random) would only add a global phase to the state.

Scenario 14 *We instantiate the generic AaP scenario for quantum money described in Section 3.5 as follows:*

– *Setup functionality $\mathcal{F}_{AC}(1^\lambda)$:*
  - $\texttt{init}_{AC}$ *initializes a random oracle H taking strings of length $2\lambda + \lambda^2$ to strings of length $2\lambda + \lambda^2$.*[17] *In addition, it initializes an empty database B_{AC} that is destined to contain a record of all quantum money bills in circulation.*
  - $\mathcal{O}_{\mathcal{F}_W}(I, \texttt{getId})$*: generates $v \in \{0,1\}^\lambda$ and $\theta \in \{0,1\}^\lambda$ such that $|\theta|_H = \frac{n}{2}$ uniformly at random and selects $\mathcal{Z} = \{z_i : i \in \{1,\ldots,\lambda\}\}$ a uniformly random basis for $\mathbb{Z}_2^\lambda$. Sets $\mathsf{id} = H^{-1}((v,\theta,\mathcal{Z}))$,*[18] *$\mathsf{secret} = (v,\theta,\mathcal{Z})$ and $|\$\rangle_{\mathsf{id}} = |\$\rangle_{v,\theta,\mathcal{Z}}$ defined in (4). Adds $(\mathsf{id}, \mathsf{public}, \mathsf{secret}, |\$\rangle_{\mathsf{id}})$ to B_{AC}. Returns id.*
  - $\mathcal{O}_{\mathcal{F}_{AC}}(\cdot, \texttt{public}, \mathsf{id})$*, $\mathcal{O}_{\mathcal{F}_{AC}}(I, \texttt{getMoney}, \mathsf{id})$, and $\mathcal{O}_{\mathcal{F}_{AC}}(V, \texttt{secret}, \mathsf{id})$: as described in Section 3.5.*

– *Agreement relation $\mathcal{C}_W^{\mathcal{O}_{\mathcal{F}_{AC}}}(1^\lambda, \mathsf{id})$: The agreement relation is the same as it is in Section 3.5.*

– *Proof relation $\mathcal{R}_{AC}^{\mathcal{O}_{\mathcal{F}_W}}(1^\lambda, x, \rho_W)$: The proof relation firstly queries $\mathcal{O}_{\mathcal{F}_W}(V, \texttt{secret}, x)$ in order to get a tuple $(v,\theta,\mathcal{Z})$. (If $\mathcal{O}_{\mathcal{F}}(V, \texttt{secret}, x)$ returns $\perp$, then $\mathcal{R}$ rejects.) Then it applies $Z(e)X(d)$ to its quantum input ρ_W, where d and e are defined in terms of (v,θ) the same way that they are below equation (4). Following that, it defines A to be the subspace generated by the vectors $z_i \in \mathcal{Z}$ such that $\theta_i = 1$, and then it follows the subspace money verification procedure: it performs the projective measurement $H^{\otimes\lambda}\mathbb{P}_{A^\perp}H^{\otimes\lambda}\mathbb{P}_A$ on $Z(e)X(d)\rho_W X(d)Z(e)$ (where $\mathbb{P}_A$ is a projection onto all standard basis strings in A, i.e. $\mathbb{P}_A = \sum_{x\in A}|x\rangle\langle x|$, and $\mathbb{P}_{A^\perp}$ is a projection onto all standard basis strings in $A^\perp$), and accepts if and only if the outcome is 1.*

Protocol 15 *We define a proof of knowledge protocol $\mathcal{K}_{AC}$ for the scenario $(\mathcal{F}_{AC}, \mathcal{C}_{AC}, \mathcal{R}_{AC})$. The agreement phase is identical to that in Protocol 13, except that now id has length $2\lambda + \lambda^2$. The second phase is similar but not identical, as the verifier's challenge now consists of a single bit:*

[17] Formally the oracle is implemented in the standard way, recalled in Section 2.3.
[18] We use H^{-1} and not H here because we specified in Section 3.5 that H maps ids to $(\mathsf{public}, \mathsf{secret})$ pairs.

1. V_2 queries $\mathcal{O}_{\mathcal{F}_{AC}}(V, \texttt{secret}, \mathsf{id})$. If it obtains $\perp$, V_2 aborts. Otherwise, let $\$ = (v, \theta, \mathcal{Z})$ be the classical description obtained.
2. V_2 sends a uniformly random bit $c \in \{0,1\}$ to the prover.
3. If $c = 0$ the prover P_2 measures the state ρ_{st_P} it received from P_1 in the standard basis, obtaining a λ-bit string of outcomes $m \in \{0,1\}^\lambda$, and sends m to the verifier. If $c = 1$ then P_2 measures in the Hadamard basis and likewise sends the outcomes m to V_2.
4. If $c = 0$ the verifier V_2 checks that $m + Wd$ is in the subspace A spanned by $\{z_i : \theta_i = 1\}$, where $\mathcal{Z} = \{z_1, \dots, z_\lambda\}$. If $c = 1$ then V_2 checks that $m + We$ is in $A^\perp$.

Finally, the class of input generation algorithms $\mathcal{I}_{AC}$ used for completeness is the same as the class $\mathcal{I}_W$ in Protocol 13.

Lemma 2. *There exists a constant $\kappa < 1$ such that Protocol $\mathcal{K}_{AC}$ (Protocol 15) is secure with completeness 1, up to knowledge error κ, and with extraction distance $\delta = O(\mu^{1/4})$, where $\mu = 1 - p$ and p the prover's success probability, for the subspace AaP scenario $(\mathcal{F}_{AC}, \mathcal{C}_{AC}, \mathcal{R}_{AC})$ (Scenario 14).*

Proof. The proof is similar to that of Lemma 1. Due to space constraints, we refer the reader to the full version [VZ21] for the proof.

6 Arguments of Quantum Knowledge for QMA relations

The main result of this section is Theorem 1, which gives a classical-verifier protocol to verify any QMA relation (a natural quantum analogue of an NP relation; we recall the definition in Section 6.1 below). Since this protocol is only sound against QPT provers, we refer to it as a 'classical argument of quantum knowledge'. We note that, for general QMA relations, the completeness property from Definition 5 requires the honest prover to hold multiple copies of the QMA witness in order to succeed with high probability. It is still possible for completeness to hold with a single witness if one assumes that the QMA relation takes a specific form; see the statement of Theorem 1 below.

Our construction is based on the classical verification protocol for QMA introduced in [Mah18b], which we review in Section 6.2. Before doing so we introduce the Agree-and-Prove scenario.

6.1 Agree-and-Prove scenario for QMA relations

We first recall the quantum extension of an NP relation $\mathcal{R}$, following [CVZ19,BG19].

Definition 16 (QMA relation). A QMA *relation* is specified by a triple (Q, α, β) where $\alpha, \beta : \mathbb{N} \to [0,1]$ satisfy $\beta(n) \leq \alpha(n)$ for all $n \in \mathbb{N}$ and $Q = \{Q_n\}_{n \in \mathbb{N}}$ is a uniformly generated family of quantum circuits such that for every n, Q_n takes as input a string $x \in \{0,1\}^n$ and a quantum state $|\psi\rangle$ on $p(n)$ qubits (i.e. Q_n takes $n + p(n)$ input qubits for some polynomial p that is implicitly specified by Q, and is assumed to immediately measure its first n input qubits in the computational basis) and returns a single bit.

To a QMA relation (Q, α, β) we associate two sets

$$R_{Q,\alpha} = \bigcup_{n\in\mathbb{N}} \Big\{(x,\sigma) \in \{0,1\}^n \times \mathrm{D}(\mathbb{C}^{p(n)}) \,\Big|\, \Pr(Q_n(x,\sigma) = 1) \geq \alpha\Big\}$$

and

$$N_{Q,\beta} = \bigcup_{n\in\mathbb{N}} \Big\{x \in \{0,1\}^n \,\Big|\, \forall\sigma \in \mathrm{D}(\mathbb{C}^{p(n)})\,,\ \Pr(Q_n(x,\sigma) = 1) < \beta\Big\}\,.$$

We say that a (promise) *language* $L = (L_{yes}, L_{no})$ *is specified by the* QMA *relation* (Q, α, β) if

$$L_{yes} \subseteq \Big\{x \in \{0,1\}^* \Big|\, \exists\sigma \in \mathrm{D}(\mathbb{C}^{p(n)})\,,\ (x,\sigma) \in R_{Q,\alpha}\Big\}\,, \tag{5}$$

and $L_{no} \subseteq N_{Q,\beta}$. Note that, whenever $\alpha - \beta > 1/\operatorname{poly}(n)$, any language L that is specified by (Q, α, β) lies in QMA. Conversely, any language in QMA is specified by some QMA relation in a straightforward (non-unique) way.

The local Hamiltonian problem In the following, we make use of Kitaev's circuit-to-Hamiltonian construction [KSVV02,KR03], which associates with any promise language $L = (L_{yes}, L_{no}) \in$ QMA and $x \in L_{yes} \cup L_{no}$ an instance of the *local Hamiltonian problem*. An instance of the local Hamiltonian problem is specified by a local Hamiltonian operator H and two real numbers $\alpha > \beta$. The instance is a 'YES instance' if H has smallest eigenvalue at most α, and a 'NO instance' if it is at least β.

Agree-and-Prove scenario. Fix a QMA relation (Q, α, β). We associate an AaP scenario to Q as follows.

- Setup functionality $\mathcal{F}_Q(1^\lambda)$. We consider a "trivial" setup, i.e. the initialization procedure does nothing and there is no associated oracle $\mathcal{O}_{\mathcal{F}_Q}$.
- Agreement relation $\mathcal{C}_Q(1^\lambda, x)$: returns 1 for any λ and x.[19]
- Proof relation $\mathcal{R}_Q(1^\lambda, x, \rho)$: executes the verification circuit $Q_{|x|}$ on the pair (x, ρ) and returns the outcome.

We end by presenting some assumptions on a QMA relation under which our results will hold. Let (Q, α, β) be a QMA relation. We require that the relation satisfies the following properties:

(Q.1) The completeness parameter α is negligibly close to 1, and the soundness parameter β is bounded away from 1 by an inverse polynomial.

[19] The agreement relation does not even require that $x \in R_{Q,\alpha} \cup N_{Q,\beta}$, as in general this cannot be efficiently verified.

(**Q.2**) For any $x \in \{0,1\}^n$ there is a local Hamiltonian $H = H_x$ that is efficiently constructible from x and satisfies the following. First, we assume that H is expressed as a linear combination of tensor products of Pauli operators with real coefficients chosen such that $-\mathrm{Id} \leq H \leq \mathrm{Id}$. Second, whenever there is σ such that $(x,\sigma) \in R_{Q,\alpha}$, then $\mathrm{Tr}(H\sigma)$ is negligibly close to -1 and moreover any σ such that $\mathrm{Tr}(H\sigma) \leq -1+\delta$ satisfies $\Pr(Q_{|x|}(x,\sigma)=1) \geq 1-r(|x|)q(\delta)$ for some polynomials q, r depending on the relation only. Third, whenever $x \in N_{Q,\beta}$ then the smallest eigenvalue of H is larger than $-1+1/s(|x|)$, where s is another polynomial depending on the relation only.

The first of these assumptions is benign and can be made without loss of generality; the second assumption is a little more restrictive. For a fuller discussion of these assumptions, we refer the reader to the full version [VZ21].

6.2 The protocol

In the following subsection we recall the high-level structure of the verification protocol from [Mah18b], on which our AaP protocol for the scenario given in Section 6.1 will be based.

The verification protocol from [Mah18b] In the protocol from [Mah18b], which we will refer to as the *verification protocol*, the input to the verifier is an n-qubit Hamiltonian H that is expressed as a linear combination of tensor products of σ_X and σ_Z Pauli operators. The input to the prover is a ground state of H. Both parties also receive a security parameter λ. At a high level, the verification protocol has the following structure:

1. The verifier selects a *basis string* $h \in \{0,1\}^n$ according to a distribution that depends on H. The verifier then randomly samples a pair of keys, (pk, sk), consisting of a public key pk and secret key sk. (The distribution according to which (pk, sk) is sampled depends on h.) The choice of keys specifies an integer w of size $\mathrm{poly}(n,\lambda)$. The verifier sends pk to the prover.
2. The prover returns an n-tuple of *commitment strings* $y = (y_1, \ldots, y_n)$, where each y_i lies in some alphabet $\mathcal{Y}$.
3. The verifier selects a *challenge bit* $c \in \{0,1\}$ and sends c to the prover.
4. If $c = 0$ ("test round"), the prover returns a string $b \in \{0,1\}^n$ and $x_1, \ldots, x_n \in \{0,1\}^w$. If $c = 1$ ("Hadamard round"), the prover returns a string $b \in \{0,1\}^n$ and $d_1, \ldots, d_n \in \{0,1\}^w$.
5. In case $c = 0$ the verifier uses pk, y, b and $x_1, \ldots, x_n$ to make a decision to accept or reject. (In a test round the verifier never checks any properties of the prover's state; it only checks that the prover is, loosely speaking, doing the correct operations.) In case $c = 1$ the verifier uses sk to decode y, b and $d_1, \ldots, d_n$ into *decoded measurement outcomes* $(m_1, \ldots, m_n) \in \{0,1\}^n$. (For the case of a honest prover, the decoded outcomes m correspond to the outcomes of measuring a ground state of H in the bases indicated by h, with $h_i = 0$ indicating that the ith qubit should be measured in the computational

basis and $h_i = 1$ that the ith qubit should be measured in the Hadamard basis. The prover remains ignorant throughout the entire protocol of the verifier's choice of h.)

6. In case $c = 1$ the verifier makes a decision based on the decoded measurement outcomes and the instance x, as described in [Mah18c, Protocol 8.1].

To model the verifier and prover in the protocol as ITMs, in accordance with the formalism in Definition 6, we introduce registers associated with each party and the messages that they send. Let K and C denote registers that contain the verifier's first and second messages respectively, i.e. the key pk and the challenge bit c. Let T denote the verifier's private space. Let Y denote the register measured by the prover to obtain the prover's first message y, and M the register measured to obtain the prover's second message $(b, x_1, \ldots, x_n)$ or $(b, d_1, \ldots, d_n)$, depending on $c = 0$ or $c = 1$ respectively. Let S denote the prover's private space.

The natural description of the prover as an ITM consists of (i) its initial state $\sigma \in \mathrm{D}(\mathcal{H}_{\mathsf{YMS}})$, (ii) a unitary V_0 acting on KYMS, and (iii) two unitaries V and V' acting on MS, where V is the action of the prover on challenge $c = 0$ and V' its action on challenge $c = 1$. In either case the register M is measured in the computational basis to obtain the prover's answer.[20]

For convenience we introduce a slightly different representation of the prover, that matches the presentation from [Mah18b] and which can be straightforwardly simulated given black-box access to the natural representation described in the previous paragraph. First, we replace V_0 by the unitary $U_0 = VV_0$. Note that this is well-defined and does not change the prover's first message, since V does not act on Y. Second, we define $U = H^{\otimes(n+nw)}V'V^\dagger$, where the Hadamard gates act on the $(n + nw)$ qubits in register M. It is then immediate that given a natural representation of the prover as three unitaries (V_0, V, V') the pair of unitaries (U_0, U) provides a different representation of the same prover, who now behaves as follows:

1. Upon reception of pk, the prover applies U_0 to its initial state (to which $|pk\rangle$ has been appended), measures the first $n \log |\mathcal{Y}|$ qubits in the computational basis and returns the outcome;
2. Upon reception of $c = 0$, the prover directly measures the first (remaining) $n + nw$ qubits in the computational basis and returns the outcome;
3. Upon reception of $c = 1$, the prover applies the unitary U, measures the first (remaining) $n + nw$ qubits in the Hadamard basis and returns the outcome.

In both cases $c = 0$ and $c = 1$ we denote the first n qubits measured by the prover (in step 2 or in step 3, respectively), whose associated measurement outcomes are denoted by b in the protocol, the *committed qubits*.

[20] This description slightly departs from the 'canonical' formalism introduced in Section 2.2 by using different symbols for the prover's unitaries associated with different rounds as well as different challenges. It is not hard to find an equivalent description that uses the language from Section 2.2. In this case, the four registers KYCM would all be considered network registers, and are thus accessible to the extractor.

The Agree-and-Prove protocol In this section we define a protocol $\mathcal{K}_Q$ for the AaP scenario $(\mathcal{F}_Q, \mathcal{C}_Q, \mathcal{R}_Q)$ associated to a QMA relation (Q, α, β) as in Section 6.1. Recall that an Agree-and-Prove protocol consists of two phases, an "agree" phase and a "prove" phase. The agree phase in protocol $\mathcal{K}_Q$ is simple:

– The prover P_1 takes as input 1^λ and a CQ state ρ_{AUX_P}. It interprets the classical part of ρ as a string $z \in \{0,1\}^n$ and the quantum part as ℓ witnesses $\sigma_1, \ldots, \sigma_\ell$ each of the same number of qubits. (We assume that the integers n and ℓ are both encoded in a canonical way in the state ρ_{AUX_P}.) It sends z to the verifier and outputs the statement $x_p = z$ and the quantum state $\rho_{st_P} = (\sigma_1, \ldots, \sigma_\ell)$ (which may in general be entangled).
– The verifier V_1 takes as input 1^λ and a classical auxiliary input ρ_{AUX_v}. It parses ρ_{AUX_v} as the specification (in binary) of an input length n followed by a string $x \in \{0,1\}^n$. It receives z from P_1. If $z \neq x$ it aborts. Otherwise, it produces the statement $x_v = x$.

For the proof phase V_2 and P_2 behave exactly as the verifier and prover do in the verification protocol described in Section 6.2, first defining the Hamiltonian H_v and H_p from their respective statements x_v and x_p according to assumption (**Q.2**). Note that H_v (resp. H_p) acts on poly(n) qubits, with $n = |x_v|$ (resp. $n = |x_p|$). Of course, when all parties are honest, $x_v = x_p$.

To complete the description of the protocol we define a class of of input-generation algorithms under which completeness holds. We consider only input generation algorithms that generate positive instances of the language, accompanied with ℓ copies of a valid proof, where $\ell \geq 1$ is a parameter. That is, for any $\ell \geq 1$, $\mathcal{I}_Q^{(\ell)}$ contains any input generation algorithm I that returns a CQ state of the form

$$\sum_{x \in \{0,1\}^*} p_x \, ||x|, x\rangle\langle |x|, x|_{\mathsf{AUX}_V} \otimes \big(|x\rangle\langle x| \otimes \sigma_x^{\otimes \ell} \big)_{\mathsf{AUX}_P} \,, \quad (6)$$ [21]

where (p_x) is any distribution over positive instances for the QMA relation, i.e. the set

$$\{x : \exists \sigma, (x, \sigma) \in R_{Q,\alpha}\} \,,$$

and moreover for each x, σ_x is such that $(x, \sigma_x) \in R_{Q,\alpha}$.

6.3 Arguments of Quantum Knowledge for QMA relations

We state the main result of this section.

Theorem 1. *Let (Q, α, β) be a* QMA *relation that satisfies properties* ***(Q.1)*** *and* ***(Q.2)*** *described in Section 6.1. There exists a polynomially bounded $\ell = \ell(n)$ such that the following holds. Under the Learning with Errors assumption the protocol presented in Section 6.2 is secure with completeness c (for the class of input generation algorithms $\mathcal{I}_Q^{(\ell)}$), up to knowledge error κ and with extraction*

[21] For clarity we omit explicitly writing out $|x|$ in both registers.

distance δ for the Agree-and-Prove scenario $(\mathcal{F}_Q, \mathcal{C}_Q, \mathcal{R}_Q)$, where: c is negligibly close to 1; κ is bounded away from 1 by an inverse polynomial; $\delta = \mathrm{poly}(1 - p)\,\mathrm{poly}(n)$ (for any prover success probability $p > \kappa$).

Proof. Due to space constraints, we refer the reader to the full version [VZ21] for the proof.

6.4 Sequential amplification

Due to space constraints, we refer the reader to the full version [VZ21] for a treatment of sequential amplification of the [Mah18c] protocol.

References

AC12. Scott Aaronson and Paul Christiano. Quantum money from hidden subspaces. In *Proceedings of the forty-fourth annual ACM Symposium on Theory of Computing*, 2012. 1, 4.2, 5.2

AFG$^+$12. Scott Aaronson, Edward Farhi, David Gosset, Avinatan Hassidim, Jonathan Kelner, and Andrew Lutomirski. Quantum money. *Commun. ACM*, 55(8):84–92, August 2012. 1

BCC$^+$20. Christian Badertscher, Alexandru Cojocaru, Léo Colisson, Elham Kashefi, Dominik Leichtle, Atul Mantri, and Petros Wallden. Security limitations of classical-client delegated quantum computing. *arXiv preprint arXiv:2007.01668*, 2020. 1

BDS16. Shalev Ben-David and Or Sattath. Quantum tokens for digital signatures. *arXiv preprint arXiv:1609.09047*, 2016. 1, 15

BG92. Mihir Bellare and Oded Goldreich. On defining proofs of knowledge. In *Annual International Cryptology Conference*, pages 390–420. Springer, 1992. 1

BG19. Anne Broadbent and Alex B Grilo. Zero-knowledge for QMA from locally simulatable proofs. *arXiv preprint arXiv:1911.07782*, 2019. 1, 6.1

BJM19. Christian Badertscher, Daniel Jost, and Ueli Maurer. Agree-and-prove: Generalized proofs of knowledge and applications. *IACR Cryptol. ePrint Arch.*, 2019:662, 2019. 1, 3, 3, 4, 5

CL06. Melissa Chase and Anna Lysyanskaya. On signatures of knowledge. In Cynthia Dwork, editor, *Advances in Cryptology - CRYPTO 2006*, pages 78–96, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg. 1

CVZ19. Andrea Coladangelo, Thomas Vidick, and Tina Zhang. Non-interactive zero-knowledge arguments for QMA, with preprocessing. *arXiv preprint arXiv:1911.07546*, 2019. 1, 2.2, 6.1

FFS88. Uriel Feige, Amos Fiat, and Adi Shamir. Zero-knowledge proofs of identity. *Journal of cryptology*, 1(2):77–94, 1988. 1

GMR89. Shafi Goldwasser, Silvio Micali, and Charles Rackoff. The knowledge complexity of interactive proof systems. *SIAM Journal on Computing*, 18(1):186–208, 1989. 1

GV19. Alexandru Gheorghiu and Thomas Vidick. Computationally-secure and composable remote state preparation. In *2019 IEEE 60th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 1024–1033. IEEE, 2019. 1

HHJ$^+$17. Jeongwan Haah, Aram W Harrow, Zhengfeng Ji, Xiaodi Wu, and Nengkun Yu. Sample-optimal tomography of quantum states. *IEEE Transactions on Information Theory*, 63(9):5628–5641, 2017. 1

KR03. Julia Kempe and Oded Regev. 3-local Hamiltonian is QMA-complete. *Quantum Information and Computation*, 3(3):258–264, 2003. 6.1

KSVV02. Alexei Yu Kitaev, Alexander Shen, Mikhail N Vyalyi, and Mikhail N Vyalyi. *Classical and quantum computation.* Number 47. American Mathematical Soc., 2002. 6.1

Mah18a. Urmila Mahadev. Classical homomorphic encryption for quantum circuits. In *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 332–338. IEEE, 2018. 1

Mah18b. Urmila Mahadev. Classical verification of quantum computations. In *Foundations of Computer Science (FOCS), 2018 IEEE 59th Annual Symposium on*, pages 259–267, Oct 2018. 1, 6, 6.2, 6.2, 6.2

Mah18c. Urmila Mahadev. Classical verification of quantum computations. *arXiv preprint arXiv:1804.01082*, 2018. 6, 6.4

MV20. Tony Metger and Thomas Vidick. Self-testing of a single quantum device under computational assumptions. *arXiv preprint arXiv:2001.09161*, 2020. 1

MVW12. Abel Molina, Thomas Vidick, and John Watrous. Optimal counterfeiting attacks and generalizations for Wiesner's quantum money. In *Conference on Quantum Computation, Communication, and Cryptography.* Springer, 2012. 1, 5.1

NV16. Anand Natarajan and Thomas Vidick. Robust self-testing of many-qubit states. *arXiv e-prints*, page arXiv:1610.03574, Oct 2016. 1, 5.1

SJ00. Claus Schnorr and Markus Jakobsson. Security of signed ElGamal encryption. In *International Conference on the Theory and Application of Cryptology and Information Security*, volume 1976, pages 73–89, 12 2000. 1

TW87. Martin Tompa and Heather Woll. Random self-reducibility and zero knowledge interactive proofs of possession of information. In *28th Annual Symposium on Foundations of Computer Science (sfcs 1987)*, pages 472–482. IEEE, 1987. 1

Unr12. Dominique Unruh. Quantum proofs of knowledge. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 135–152. Springer, 2012. 1, 2.2, 1

VZ20. Thomas Vidick and Tina Zhang. Classical zero-knowledge arguments for quantum computations. *Quantum*, 4:266, 2020. 3

VZ21. Thomas Vidick and Tina Zhang. Classical proofs of quantum knowledge, 2021. 2.1, 2.2, 2.3, 3.2, 5, 4.1, 4.1, 4.2, 10, 4.2, 4.2, 4.2, 5.1, 5.2, 6.1, 6.3, 6.4

Wat09. John Watrous. Zero-knowledge against quantum attacks. *SIAM Journal on Computing*, 39(1):25–58, 2009. 1

Wie83. Stephen Wiesner. Conjugate coding. *ACM SIGACT News*, 15(1):78–88, 1983. 1

Zha19. Mark Zhandry. Quantum lightning never strikes the same state twice. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 408–438. Springer, 2019. 1