Non-interactive zero-knowledge arguments for QMA, with preprocessing

Andrea Coladangelo*[2], Thomas Vidick†[1,2], and Tina Zhang‡[3]

[1]*Department of Computing and Mathematical Sciences, California Institute of Technology*
[2]*Institute for Quantum Information and Matter, California Institute of Technology*
[3]*Division of Physics, Mathematics and Astronomy, California Institute of Technology*

Abstract

A non-interactive zero-knowledge (NIZK) proof system for a language $L \in$ NP allows a prover (who is provided with an instance $x \in L$, and a witness w for x) to compute a *classical certificate* π for the claim that $x \in L$ such that π has the following properties: 1) π can be verified efficiently, and 2) π does not reveal any information about w, besides the fact that it exists (i.e. that $x \in L$). NIZK proof systems have recently been shown to exist for all languages in NP in the common reference string (CRS) model and under the learning with errors (LWE) assumption.

We initiate the study of NIZK *arguments* for languages in QMA. An argument system differs from a proof system in that the honest prover must be efficient, and that it is only sound against (quantum) polynomial-time provers. Our first main result is the following: if LWE is hard for quantum computers, then any language in QMA has an *NIZK argument with preprocessing*. The preprocessing in our argument system consists of (i) the generation of a CRS and (ii) a *single (instance-independent) quantum message* from verifier to prover. Meanwhile, the instance-dependent phase of our argument system involves only a single *classical* message from prover to verifier. Importantly, verification in our protocol is entirely classical, and the verifier needs not have quantum memory; its only quantum actions are in the preprocessing phase. NIZK proofs of (classical) knowledge are widely used in the construction of more advanced cryptographic protocols, and we expect the quantum analogue to likewise find a broad range of applications. In this respect, the fact that our protocol has an entirely classical verification phase is particularly appealing.

Our second contribution is to extend the notion of a classical *proof of knowledge* to the quantum setting. We introduce the notions of *arguments* and *proofs of quantum knowledge* (AoQK/PoQK), and we show that our non-interactive argument system satisfies the definition of an AoQK. In particular, we explicitly construct an extractor which can recover a quantum witness from any prover who is successful in our protocol. We also show that any language in QMA has an (interactive) *proof of quantum knowledge*, again by exhibiting a particular proof system for all languages in QMA and constructing an extractor for it. That our NIZK argument system is also an AoQK further broadens the contexts in which it may be applied.

*Email: acoladan@caltech.edu
†Email: vidick@caltech.edu
‡Email: tinazhang@caltech.edu

Contents

1 Introduction **3**

2 Preliminaries **13**

2.1 Notation . . . 13

2.2 The [BJSW16] protocol . . . 13

2.3 Cryptographic primitives . . . 17

2.3.1 Homomorphic encryption (with circuit privacy) . . . 17

2.3.2 Commitment scheme . . . 18

2.4 Argument systems . . . 20

2.4.1 Interactive quantum machines . . . 20

2.4.2 Argument systems with setup . . . 21

2.5 Zero-knowledge arguments . . . 23

2.5.1 Zero-knowledge . . . 23

2.5.2 Non-interactive zero-knowledge arguments for QMA . . . 24

2.5.3 Non-interactive zero-knowledge for NP . . . 25

2.6 Proofs and arguments of quantum knowledge . . . 27

2.6.1 Reducing the knowledge error sequentially . . . 29

3 The protocol **30**

3.1 Notation and predicates . . . 30

3.2 The protocol . . . 32

4 Soundness **36**

4.1 Overview . . . 36

4.2 Success probability in hybrids . . . 38

4.3 Reduction to [BJSW16] . . . 42

4.4 Parallel amplification of soundness . . . 43

5 Zero-knowledge property **47**

5.1 The original protocol . . . 48
5.2 Replacing the NP proof and the commitment . . . 49
5.3 Replacing ρ with ρ_r . . . 50
5.3.1 Pauli-twirling the verifier . . . 50
5.3.2 Reducing to [BJSW16] . . . 55
5.4 Feasibility of the $\rho \to \rho_r$ replacement . . . 55

6 NIZK argument of quantum knowledge with setup for QMA **56**

7 Proofs of quantum knowledge for QMA **58**

1 Introduction

The paradigm of the interactive proof system is commonly studied in cryptography and in complexity theory. Intuitively speaking, an interactive proof system is a protocol in which an *unbounded* prover attempts to convince an *efficient* verifier that some problem instance x is in some language L. The verifier represents an entity less computationally powerful or less informed than the prover; the prover holds some knowledge that the verifier does not (namely, that $x \in L$), and the prover attempts to convince the verifier of this knowledge. We say that there is an interactive proof system *for a language* L if the following two conditions are satisfied. Firstly, for any $x \in L$, there must exist a prover (the 'honest' prover) which causes the (honest) verifier to accept in the protocol with high probability; and secondly, for any $x \notin L$, there is no prover which can cause the honest verifier to accept, except with some small probability. These two conditions are commonly referred to as the 'completeness' and 'soundness' conditions. We can also consider a relaxed soundness condition where, when $x \notin L$, we require only that it be computationally intractable (rather than impossible) to cause the verifier to accept. A protocol satisfying this relaxed soundness condition, and which has an efficient honest prover, is known as an interactive *argument* system.

Some interactive proof and argument systems satisfy a third property known as *zero-knowledge* [GMR85], which captures the informal notion that the verifier (even a dishonest verifier) 'learns no new information' from an interaction with the honest prover, except for the information that $x \in L$. This idea is formalised through a *simulator*, which has the same computational powers as the verifier V does, and can output transcripts that (for x such that $x \in L$) are indistinguishable from transcripts arising from interactions between V and the honest prover. As such, V intuitively 'learns nothing', because whatever it might have learned from a transcript it could equally have generated by itself. The property of zero-knowledge can be *perfect* (PZK), *statistical* (SZK) or *computational* (CZK). The difference between these three definitions is the extent to which simulated transcripts are indistinguishable from real ones. In a PZK protocol, the simulator's output distribution is *identical* to the distribution of transcripts that the honest prover and (potentially

dishonest) verifier generate when $x \in L$. In SZK, the two distributions have negligible statistical distance, and in CZK, they are computationally indistinguishable. In this work we will primarily be concerned with CZK.

A *non-interactive* proof system (or argument system) is a protocol in which the prover and the verifier exchange only a single message that depends on the problem instance x. (In general, an instance-independent setup phase may be allowed in which the prover and verifier communicate, with each other or with a trusted third party, in order to establish shared state that is used during the protocol execution proper. We discuss this in more detail in the following paragraph.) Non-interactive zero-knowledge (NIZK) proofs and arguments have seen widespread application in classical cryptography, often in venues where their interactive counterparts would be impracticable—including, notably, in CCA-secure public-key cryptosystems [NY90, Sah99], digital signature schemes [BG90, CP92, BMW03], verifiable delegated computation [PHGR13] and, recently, a number of blockchain constructions [GGPR13, Com14, Lab17]. A particularly attractive feature of classical NIZK systems is that they can be amplified *in parallel* to achieve better security parameters [BDSMP91], which is in general not true of their interactive (private-coin) counterparts.

It is known [GO94] that NIZK proofs and arguments in the *standard model* (namely, the model where the only assumption is that adversarial entities are computationally efficient) exist only for languages in BPP. As such, in order to construct NIZK protocols for more interesting languages, it is customary to consider *extended* cryptographic models. Examples of these include the *common reference string* (CRS) model, in which the verifier and the prover are assumed to begin the protocol sharing access to a common string sampled from a specified distribution; and the *random oracle* (RO) model, in which prover and verifier have access to an efficiently evaluable function that behaves like a function sampled uniformly at random from the set of possible functions with some specified, and finite, domain and range. In these extended models, and under certain computational hardness assumptions, non-interactive computational zero-knowledge proof systems for all languages in NP are known. For instance, Blum, Santis, Micali and Persiano [BDSMP91] showed in 1990 that NIZK proofs for all languages in NP exist in the CRS model, assuming that the problem of quadratic residuosity is computationally intractable.

At this point, a natural question arises: what happens in the *quantum* setting? Ever since Shor's algorithm for factoring [Sho95] was published in 1995, it has been understood that the introduction of quantum computers would render a wide range of cryptographic protocols insecure. For example, quadratic residuosity is known to be solvable in polynomial time by quantum computers. Given that this is so, it is natural to ask the following question: in the presence of quantum adversaries, is it still possible to obtain proof systems for all languages in NP that are complete and sound, and if it is, in which extended models is it feasible? This question has been studied in recent years. For example, Unruh showed in [Unr15] that quantum-resistant NIZK proof systems for all languages in NP exist in the quantum random oracle (QRO) model, a quantum generalisation of the random oracle model. More recently, Peikert and Shiehian [PS19] achieved a more direct analogue of Blum et al.'s result, by showing that NIZK proofs for all languages in NP exist in the CRS model, assuming that learning with errors (LWE)—a problem believed to be difficult for quantum computers—is computationally intractable. [1]

[1] Peikert and Shiehan construct, based on LWE, a NI(C)ZK proof system in the common *reference* string model, and

However, the advent of large-scale quantum computers would not only render some cryptosystems insecure; it would also provide us with computational powers that extend those of our current classical machines, and give rise to new cryptographic tasks that were never considered in the classical literature. A second natural question which arises in the presence of quantum computers is the following: in which models is it possible to obtain a NIZK proof or argument system not only for all languages in NP, but for all languages in 'quantum NP' (i.e. QMA)? Loosely speaking, NIZK protocols for NP languages allow the prover to prove any statement that can be checked efficiently by a classical verifier who is given a classical witness. A NIZK protocol for QMA languages would, analogously, allow the prover to prove to the verifier (in a non-interactive, zero-knowledge way) the veracity of statements that require a quantum witness and quantum computing power to check. To our knowledge, the question of achieving NIZK protocols for QMA has not yet been studied. In 2016, Broadbent, Ji, Song and Watrous [BJSW16] exhibited a zero-knowledge proof system for QMA with an efficient honest prover, but their protocol requires both quantum and classical interaction.

In this work, our first contribution is to propose a non-interactive (computational) zero-knowledge argument system for all languages in QMA, based on the hardness of LWE, in which both verifier and prover are quantum polynomial time. The model we consider is the CRS (common reference string) model, augmented by a single message of (quantum) preprocessing. (The preprocessing consists of an instance-independent quantum message from the verifier to the prover.) The post-setup single message that the prover sends to the verifier, after it receives the witness, is classical; the post-setup verifier is also entirely classical; and, if we allow the prover and verifier to share EPR pairs *a priori*, as in a model previously considered by Kobayashi [Kob02], we can also make the verifier's preprocessing message classical. Like classical NIZK protocols, our protocol shows itself to be receptive to parallel repetition (see section 4.4), which allows us to amplify soundness concurrently without affecting zero-knowledge. Our model and our assumptions are relatively standard ones which can be fruitfully compared with those which have been studied in the classical setting. As such, this result provides an early benchmark of the kinds of assumptions under which NIZK can be achieved for languages in QMA.

Our second contribution is to show that our protocol also satisfies a notion of *argument of quantum knowledge*. In the classical setting, some proof systems and argument systems for NP languages satisfy a stronger notion of soundness wherein a witness can be *extracted* from any prover P who convinces the verifier to accept with high probability. More formally, in such a setting, there is an *extractor* machine which—given black-box access to any P who convinces the verifier to accept with high probability (on the input x)—is able to efficiently compute a witness w that testifies that the problem instance x is in the language L. Such protocols are known as *proofs* and *arguments of knowledge* (PoK and AoK). Intuitively speaking, the notion of PoK/AoK is a framework for describing situations where the prover is not necessarily more powerful, but only *better informed*, than the verifier. In these situations, the prover possesses knowledge (the witness w, which could represent a password or some other form of private information) that the verifier does not; and the

a NI(S)ZK argument system in the common *random* string model. They do not explicitly consider the applications of either result to the quantum setting. We show, however, for our own purposes, that the latter of these results generalises to quantum adversaries. In other words, we show (in section 2.5.3) that the Peikert-Shiehan NIZK *argument* system in the common *random* string model is adaptively sound against quantum adversaries and adaptively (quantum computational) zero-knowledge.

prover wishes to convince the verifier, possibly in a zero-knowledge way (i.e. without revealing sensitive information), that it indeed 'knows' or 'possesses' the witness w (so that it might, for example, be granted access to its password-protected files, or cash a quantum cheque). The idea of a machine 'knowing' some witness w is formalised by the existence of the extractor.

Until now, the witness w has always been classical, and the notion of a proof of *quantum* knowledge (PoQK) has not been formally defined or studied. In this paper, we firstly formulate a definition for a PoQK that is analogous to the classical definition of a PoK[2], and we show that there exists a (interactive) PoQK for any language in QMA[3]. Secondly, we introduce the notion of an *argument of quantum knowledge* (AoQK), and we prove that our NIZK protocol for QMA is also (under this definition) a zero-knowledge argument of quantum knowledge. We present our definition of a PoQK in section 2.6. There are two main difficulties in extending the classical notion of a PoK to the quantum setting. The first is that we must precisely specify how the extractor should be permitted to interact with the successful (quantum) prover. For this, we borrow the formalism of quantum interactive machines that Unruh [Unr12] uses in defining quantum proofs of *classical* knowledge. The second difficulty is to give an appropriate definition of success for the extractor. In the classical setting, the NP relation R which defines the set of witnesses w for a problem instance x is binary: a string w is either a witness or it is not. In the quantum setting, on the other hand—unlike in the classical case, in which any witness is as good as any other—different witnesses might be accepted with different probabilities by some verification circuit Q under consideration. In other words, some witnesses may be of better 'quality' than others. In addition, because QMA is a probabilistic class, the choice of Q (which is analogous to the choice of the NP relation R) is more obviously ambiguous than it is in the classical case. Different (and equally valid) choices of verifiers Q for a particular language $L \in \mathrm{QMA}$ might have different probabilities of accepting a candidate witness ρ on a particular instance x. In our definition, we define a 'QMA relation' with respect to a fixed choice of verifying circuit (family) Q; we define the 'quality' of a candidate witness ρ for x to be the probability that Q accepts (x, ρ); and we require that the successful extractor returns a witness whose quality lies strictly above the soundness parameter for the QMA relation.

Since we show that our protocol is a NIZK argument of quantum knowledge under our new definition, it can be used in settings where a prover wishes to prove it 'knows' or 'possesses' a quantum state. For example, this state could be a quantum money state [Wie83]. Suppose that the prover participated in the appropriate setup phase with the bank when the quantum money state was minted. Later on, using our protocol, the prover could—by sending a *single classical* message, which can be processed *classically*—demonstrate to the bank that it still possesses that quantum banknote, allowing the prover to cash a quantum cheque or transfer quantum funds.[4] The fact that our protocol is an *argument of quantum knowledge* means that, when it is used in this way (i.e. in the place of a more traditional quantum money verification protocol), it preserves (computationally speaking) the anti-counterfeiting security properties that are central to quantum money. Another example is a history state for a certain computation. Suppose that the prover

[2] This definition is joint work with Broadbent and Grilo

[3] This result is also obtained in independent and concurrent work of Broadbent and Grilo.

[4] For this to be possible, the prover needs to be provided with a homomorphic encryption of a classical description of the quantum money state. In the case of Wiesner's scheme [Wie83], this is a Hamiltonian that is already in the Clifford form required for our protocol.

is a server to which the verifier wishes to delegate a quantum task, and that the prover and the verifier complete the setup phase of our protocol when the delegation occurs. After the setup phase is complete, *the verifier does not need to preserve any quantum information*, meaning that it could perform the setup phase using borrowed quantum resources, and thereafter return to the classical world. When it receives the prover's single-message zero-knowledge proof, the verifier can verify the computation without performing any additional quantum operations—a property that our protocol shares with protocols that have purely classical verification, such as Mahadev's classical-verifier argument system for QMA [Mah18]. An advantage of our protocol, however, is that the server can free the quantum memory associated with the verifier's computation *immediately* after the computation terminates, rather than holding the history state until the verifier is available to perform the verification.

The interactive protocol from [BJSW16]

Our protocol is inspired by the protocol exhibited in [BJSW16], which can be summarized as follows. (For a more detailed exposition, see section 2.2.)

1. The verifier and the prover begin with an instance x of some interesting problem, the latter of which is represented by a (promise) language $L = (L_{yes}, L_{no}) \in \mathrm{QMA}$. The prover wishes to prove to the verifier that $x \in L_{yes}$. The first step is to map x to an instance H of the QMA-complete *local Clifford Hamiltonian problem*. In the case that x is a yes instance, i.e. $x \in L_{yes}$, the prover, who receives a witness state $|\Phi\rangle$ for x as auxiliary input, performs the efficient transformation that turns the witness $|\Phi\rangle$ for x into a witness $|\Psi\rangle$ for H. (The chief property that witnesses $|\Psi\rangle$ for H have is that $\langle\Psi| H |\Psi\rangle$ is *small*—smaller than a certain threshold—which, rephrased in physics terminology, means that $\langle\Psi|$ has *low energy with respect to* H.) The prover then sends an *encoding* of $|\Psi\rangle$ to the verifier (under a specified authentication code which doubly functions as an encryption scheme). The prover also *commits* to the secret key of the authentication code.

2. The Clifford Hamiltonian H to which x has been mapped can be written as a sum of polynomially many terms of the form $C^* |0^k\rangle \langle 0^k| C$, where C is a Clifford unitary. (This is the origin of the name 'Clifford Hamiltonian'.) The verifier chooses a string r uniformly at random. r plays a role analogous to that of the verifier's choice of edge to check in the 3-colouring zero-knowledge protocol introduced by [GMR85]: intuitively, r determines the verifier's challenge to the prover. Each r corresponds to one of the terms $C_r^* |0^k\rangle \langle 0^k| C_r$ of the Clifford Hamiltonian.

   The verifier then measures the term $C_r^* |0^k\rangle \langle 0^k| C_r$ on the encoded witness (this can be done 'homomorphically' through the encoding). The outcome z obtained by the verifier can be thought of as an encoding of the true measurement outcome, the latter of which should be *small* (i.e. correspond to low energy) if $|\Psi\rangle$ is a true witness. The verifier sends z (its measurement outcomes) and r (its choice of Hamiltonian term) back the prover.

3. Finally, using a zero-knowledge NP proof system,[5] the prover provides an (interactive) ZK proof for the following NP statement: there *exists* an opening to its earlier (perfectly bind-

[5] It is known that there are quantumly sound and quantumly zero-knowledge proof systems for NP: see [Wat09].

ing) commitment such that, if the verifier had the opened encoding keys, it *would* accept. This is an NP statement because the witness string is the encoding keys. Proving that the verifier 'would accept' amounts to proving that the verifier's measurement outcomes z, decoded under the keys which were committed to earlier, would correspond to a low-energy outcome. Because the proof that the prover provides is zero-knowledge, the verifier learns nothing substantial from this exchange, but it becomes convinced that it should accept.

In the protocol from [BJSW16], it is critical to soundness that the prover sends the encoding of the witness to the verifier *before* the verifier chooses r. The zero-knowledge property holds because the encoding that the prover applies to the witness state functions like an authenticated encryption scheme: its encryption-like properties prevent the verifier from learning anything substantial about the witness while handling the encoded state, and its authentication code–like properties ensure that the verifier cannot deviate very far from its honest behaviour.

Our non-interactive protocol

We wish to make the protocol from [BJSW16] *non-interactive*. To start with, we can replace the prover's proof in step 3 with a NIZK proof in the CRS model. NIZK proofs for all languages in NP have recently been shown to exist [CLW19, PS19] based on the hardness of LWE only, and we prove that the Peikert-Shiehian construction from [PS19] remains secure (i.e. quantum computationally sound and zero-knowledge) against quantum adversaries, assuming that LWE is quantum computationally intractable. However, the more substantial obstacle to making the [BJSW16] protocol non-interactive is the following: in order to do away with the verifier's message in step 2, it seems that the prover would have to somehow *predict* z (the verifier's measurement outcomes) and send a NIZK proof corresponding to this z. Unfortunately, in order for the authentication code to work, the number of possible outcomes z has to be exponentially large (and thus the prover cannot provide a NIZK proof of consistency for each possible outcome). Even allowing for an instance-independent preprocessing step between the verifier and the prover, it is unclear how this impasse could be resolved.

Our first main idea is to use *quantum teleportation*. We add an instance-independent preprocessing step in which the verifier creates a number of EPR pairs and sends half of each to the prover. We then have the verifier (prematurely) make her measurement from step 2 *during the preprocessing step* (and hence *independently of the instance!*), and send the measurement outcomes z to the prover. Once x is revealed, the prover *teleports* the encoded witness to the verifier, and sends the verifier the teleportation outcomes d, along with a commitment to his encoding keys. The prover then provides an NIZK proof of an opening to the committed keys such that d, z and the encoding keys are consistent with a low-energy outcome. The hope is that, because the prover's and the verifier's actions commute (at least when the prover is honest), this protocol will be, in some sense, equivalent to one where the prover firstly teleports the witness, *then* the verifier makes the measurements, and finally the prover sends a NIZK proof. This latter protocol would be essentially equivalent to the [BJSW16] protocol.

There are three main issues with this strategy:

1. In the preprocessing step, the verifier does not yet know what the instance x (and hence

what the Clifford Hamiltonian) is. Thus, she cannot measure the term $C_r^* \left|0^k\right\rangle \left\langle 0^k\right| C_r$, as she would have done in what we have called step 2 of the protocol from [BJSW16].

2. The second issue is that the verifier cannot communicate her choice of r in the preprocessing step in the clear. If she does, the prover will easily be able to cheat by teleporting a state that passes the check for the rth Hamiltonian term, but that would not pass the check for any other term.

3. The third issue is a bit more subtle. If the prover knows the verifier's measurement outcomes z before he teleports the witness state to the verifier, he can misreport the teleportation outcomes d, and make a clever choice of d such that d, z and the committed keys are consistent with a low-energy outcome even when he does not possess a genuine witness.

The first issue is resolved by considering the (instance-independent) verifying circuit Q *for the* QMA *language* L (recall that Q takes as input both an instance x and a witness state), and mapping Q itself to a Clifford Hamiltonian $H(Q)$. (For comparison, in the protocol from [BJSW16], it is the circuit $Q(x, \cdot)$ which is mapped to a Clifford Hamiltonian.) In the instance-dependent step, the prover will be asked to teleport a "history state" corresponding to the execution of the circuit Q on input $(x, |\Psi\rangle)$, where $|\Psi\rangle$ is a witness for the instance x. In the preprocessing step, the verifier will measure a uniformly random term from $H(Q)$, and will also perform a special measurement (with some probability) which is meant to certify that the prover put the correct instance x into Q when it was creating the history state. Of course, the verifier does not know x at the time of this measurement, but she will know x at the point where she needs to verify the prover's NIZK proof.

Our second main idea, which addresses the second and the third issues above (at the price of downgrading our proof system to an argument system), is to have the prover *compute his NIZK proof homomorphically*. During the preprocessing step, we have the verifier send the prover a (computationally hiding) commitment σ to her choice of r; and, in addition, we ask the verifier to send the prover a *homomorphic encryption* of r, of the randomness s used to commit to σ, and of her measurement outcomes z. At the beginning of the instance-dependent step, the prover receives a witness $|\Psi\rangle$ for the instance x. During the instance-dependent step, and after having received the verifier's ciphertexts in the preprocessing step, we ask the prover firstly to commit to some choice of encoding keys, and then to teleport to the verifier (an encoding of) the history state corresponding to the execution of Q on input $(x, |\Psi\rangle)$. Let d be the outcome of the teleportation measurements. After the prover has committed to his encoding keys, we ask the prover to homomorphically encrypt d and his encoding keys, and homomorphically run the following circuit: check that r, s is a valid opening to σ, and (using the properties of the authentication code) check also that the verifier performed the honest measurement during preprocessing. If all the checks pass, then the prover *homomorphically* computes an NIZK proof that there exist encoding keys consistent with his commitment such that these keys, together with r, z, d, indicate that the verifier's measurement result was a low-energy outcome. The homomorphic encryption safeguards the verifier against a malicious prover who may attempt to take advantage of knowing r, or of the freedom to cleverly choose d, in order to pass in the protocol without holding a genuine witness.

In summary, the structure of our protocol is as follows. Let Q be a QMA verification circuit, and let $H(Q)$ be the Clifford Hamiltonian obtained from Q by performing a circuit-to-Clifford-Hamiltonian reduction.

1. *(preprocessing step)* The verifier creates a (sufficiently large) number of EPR pairs, and divides them into 'her halves' and 'the prover's halves'. She interprets her halves as the qubits making up (an encoding of) a history state generated from an evaluation of the circuit Q. Then, the verifier samples r (her 'challenge') uniformly at random, and according to its value, does one of two things: either she measures a uniformly random term of $H(Q)$ on 'her halves' of the EPR pairs, or she makes a special measurement (on her halves of the EPR pairs) whose results will allow her later to verify that the circuit Q was evaluated on the correct instance x. Following this, the verifier samples a public-key, secret-key pair (pk, sk) for a homomorphic encryption scheme. She sends the prover:
   (a) pk;
   (b) the 'prover's halves' of the EPR pairs;
   (c) a commitment to her choice of challenge, r;
   (d) homomorphic encryptions of
      i. r,
      ii. the randomness s used in the commitment, and
      iii. the measurement outcomes z.
2. *(instance-dependent step)* Upon receiving x, and a witness $|\Psi\rangle$, the prover computes the appropriate history state, and samples encoding keys. Then, he teleports an encoding of the history state to the verifier using the half EPR pairs that he previously received from her. Let d be the teleportation measurement outcomes. The prover sends to the verifier:
   (a) d;
   (b) a commitment, σ, to his encoding keys;
   (c) a homomorphic encryption of a NIZK proof (homomorphically computed) of the existence of an opening to σ such that the opened keys, together with d, z, r, are consistent with a low-energy outcome.

Analysis

Our protocol is a non-interactive, zero-knowledge argument system in the CRS model with a one-message preprocessing step. It is straightforward to see that the protocol satisfies completeness.

Intuitively, soundness follows from the fact that the encryptions the prover receives in the preprocessing step should be indistinguishable (assuming the prover is computationally bounded) from encryptions of the zero string. As such, the encryptions of z, r, s (and the commitment to r) cannot possibly be helping the prover in guessing r or in selecting a false teleportation measurement outcome d' which makes z, r, d' and the authentication keys consistent with a low-energy outcome. Soundness then essentially reduces to soundness of the protocol in [BJSW16].

The zero-knowledge property follows largely from the properties of the protocol in [BJSW16] that allowed Broadbent, Ji, Song and Watrous to achieve zero-knowledge. One key difference is that, in order to avoid rewinding the (quantum) verifier, the authors of [BJSW16] use the properties of an *interactive coin-flipping protocol* to allow the efficient simulator to recover the string r (recall that r determines the verifier's challenge) with probability 1. (The traditional alternative to this

strategy is to have the simulator guess r, and rewind the verifier if it guessed incorrectly in order to guess again. This is typical in classical proofs of zero-knowledge [GMR85]. However, because quantum rewinding [Wat09] is more delicate, the authors of [BJSW16] avoid it for simplicity.) As our protocol is non-interactive, we are unable to take the same approach. Instead, we ask the verifier to choose r and commit to it using a commitment scheme with a property we call *extractability*. Intuitively, extractability means that the commitment scheme *takes a public key determined by the CRS*. We then show that the simulator can efficiently recover r from the verifier's commitment by taking advantage of the CRS. For a candidate LWE-based extractable commitment scheme, see section 2.3.2.

Another subtlety, unique to homomorphic encryption, is that the verifier may learn something about the homomorphic computations performed by the prover (and hence possibly about the encoding keys) by looking at the *encryption randomness* in the encryption (of an NIZK proof) that the prover sends the verifier. (Recall that the verifier possesses the decryption key sk for the homomorphic encryption scheme.) This leads us to require the use of a fully homomorphic encryption scheme which satisfies the property of *circuit privacy*. For a definition of this property, see section 2.3.1.

A non-interactive argument of quantum knowledge

One desirable feature of our non-interactive argument system is that it is also an *argument of quantum knowledge*. As we mentioned earlier, one of our contributions is to generalize the definitions of PoKs and AoKs for NP-relations to definitions of PoKs and AoKs for *QMA relations*. In the latter setting, the prover wishes to convince the verifier that he 'knows' or 'possesses' the quantum witness for an instance of a QMA problem. In order to show that our protocol satisfies this additional property, we need to exhibit an extractor that, for any yes instance x, and given quantum oracle access to any prover that is accepted with high probability in our protocol, outputs a quantum state which is a witness for x. In section 6, we explicitly construct such an extractor K for our non-interactive protocol. The intuition is the following. K (the extractor) has oracle access to a prover P^*, and it simulates an execution of the protocol between P^* and the honest verifier V. We show that, if P^* is accepted in our protocol with sufficiently high probability, then it must send V (and hence K) the *encoding* $\tilde{\rho}$ of a witness state, and a commitment σ to the encoding keys. If K knew the encoding keys, it would be able to decode $\tilde{\rho}$, but it is not clear *a priori* how K could obtain such keys. Crucially, the same feature of our protocol that allows the *zero-knowledge* simulator to extract r from the verifier's commitment to r also plays in K's favour: when K simulates an execution of the protocol, it samples a common reference string which is given to both V and P^*, and in our protocol, the CRS contains a public key which P^* uses to make his commitment. As such, in order to extract a witness from P^*, the extractor samples a CRS containing a public key pk for which it knows the corresponding secret key sk, and provides this particular CRS as input to P^*. Then, when K receives $\tilde{\rho}$ and σ from P^*, it is able to extract the committed keys from σ, and use these to decode $\tilde{\rho}$.

An interactive proof of quantum knowledge

Our non-interactive protocol is an *argument system*, which means that it is sound only against computationally bounded provers. In Section 7, we add a separate but complementary result to our NIZK argument (of knowledge) for QMA by showing that the zero-knowledge proof system for QMA exhibited in [BJSW16] (with some minor modifications) is also a *proof of quantum knowledge*.

Open questions

One of the intriguing properties of our protocol is that it is both computationally sound and computationally zero-knowledge. It would be interesting to obtain a non-interactive protocol with a similarly simple preprocessing phase for which at least one of these properties (soundness or zero-knowledge) can be shown to hold information-theoretically. (This is the case for the NIZK proof for NP languages from [PS19]: the authors show that it is possible to obtain both NI(S)ZK arguments and NI(C)ZK proofs from LWE assumptions.)

It is also interesting to ask how far our preprocessing phase could be weakened or modified. Our protocol is non-interactive in the *CRS model with a single message of quantum preprocessing*. In the classical literature, a broad variety of models for achieving NIZK exists, and it seems plausible that the potential for variety in the quantum setting is even higher. It might be fruitful to ask: in which *other* models can NIZK for QMA be achieved? In particular, is it possible to weaken our preprocessing phase (while minimally perturbing the other desirable properties of our scheme, including the fact that the instance-dependent phase only involves classical communication and classical verification) so that *all* the information generated during the preprocessing phase is—like the CRS—generated by a trusted entity, and shared by this trusted entity with both prover and verifier?

The technique we proposed to remove interaction from the protocol of [BJSW16] is based on two main ingredients: the use of quantum teleportation, which allows the verifier to *anticipate* her measurements of the state she receives from the prover in the instance-dependent step, and the use of classical homomorphic encryption to allow the prover to demonstrate (homomorphically) that he has performed a certain computation correctly. These two ingredients work in tandem to ensure that the soundness and the zero-knowledge property of the [BJSW16] protocol are preserved. We believe that this technique could find use more broadly, and that it is potentially applicable as a general transformation to any 3-message protocol in which the prover's first message is a quantum state. We leave a more thorough investigation of this as a possible direction for future work.

Acknowledgements. We thank Anne Broadbent and Alex Grilo for useful discussions on the definition of proofs and arguments of quantum knowledge. We thank Zvika Brakerski and Vinod Vaikuntanathan for useful correspondence.

Andrea Coladangelo is supported by AFOSR YIP award number FA9550-16-1-0495. Thomas Vidick is supported by NSF CAREER Grant CCF-1553477, AFOSR YIP award number FA9550-16-1-0495, a CIFAR Azrieli Global Scholar award, 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).

2 Preliminaries

2.1 Notation

For an integer $\ell \geq 1$, $[\ell]$ denotes the set $\{1, \ldots, \ell\}$. We use $\mathrm{poly}(n)$ and $\mathrm{negl}(n)$ to denote an arbitrary polynomial and negligible function of n respectively (a negligible function f is any computable function such that $f(n)q(n) \to_{n\to\infty} 0$ for all polynomials q). For an integer $d \geq 1$, $\mathrm{D}(\mathbb{C}^d)$ denotes the set of density matrices on $\mathbb{C}^d$, i.e. positive semidefinite ρ on $\mathbb{C}^d$ such that $\mathrm{Tr}(\rho) = 1$. For a set S and an element $s \in S$, we write $s \xleftarrow{\$} S$ to mean that s is sampled uniformly at random from S. For an integer l, we denote by $\{0,1\}^{\leq l}$ the set of binary strings of length at most l. We use the notation S_N to denote the set of all permutations of a set of N elements.

We use the terminology PPT for *probabilistic polynomial time* and QPT for *quanutm polynomial time* to describe algorithms; formally, a PPT (resp. QPT) procedure A is a uniformly generated family $\{A_n\}$ of classical (resp. quantum) polynomial-size circuits such that A_n takes classical (resp. quantum) inputs of length $\mathrm{poly}(n)$.

2.2 The [BJSW16] protocol

The following exposition is taken from [VZ19].

In [BJSW16], Broadbent, Ji, Song and Watrous describe a protocol involving a quantum polynomial-time verifier and an unbounded prover, interacting quantumly, which constitutes a zero-knowledge proof system for languages in QMA. (Although it is sound against arbitrary provers, the system in fact only requires an honest prover who is provided with a single witness state to perform quantum polynomial-time computations.) We summarise the steps of their protocol below. For details and fuller explanations, we refer the reader to [BJSW16, Section 3].

Notation. Let L be any language in QMA. For a definition of the *k-local Clifford Hamiltonian problem*, see [BJSW16, Section 2]. The k-local Clifford Hamiltonian problem (with exponentially small ground state energy) is QMA-complete for $k = 5$; therefore, for all possible inputs x, there exists a 5-local Clifford Hamiltonian H (which can be computed efficiently from x) whose terms are all operators of the form $C^* \left|0^k\right\rangle \left\langle 0^k\right| C$ for some Clifford operator C, and such that

1. if $x \in L$, the ground energy of H is $\leq 2^{-p}$,
2. if $x \notin L$, the ground energy of H is $\geq \frac{1}{q}$,

for some positive integers p and q which are bounded above by polynomials in $|x|$.

Parties. The proof system involves

1. A *verifier*, who implements a quantum polynomial-time procedure;

2. A *prover*, who is unbounded, but who is only required by the protocol to implement a quantum polynomial-time procedure.

The verifier and the prover communicate quantumly.

Inputs.

1. Input to the verifier:
   (a) The Hamiltonian H.
   (b) A quantum computationally concealing, perfectly binding (classical) commitment protocol. In this section, we refer to the commitment algorithm from this protocol as commit; $\mathsf{commit}(\mu, s)$ takes as input a message μ and a random string s and produces a commitment string z.
   (c) A proof system for NP sound against arbitrary quantum provers.
2. Input to the prover:
   (a) The Hamiltonian H.
   (b) The n-qubit quantum state ρ, where ρ is a ground state of the Hamiltonian H.
   (c) A quantum computationally concealing, perfectly binding (classical) commitment protocol.
   (d) A proof system for NP sound against arbitrary quantum provers.

Protocol.

1. *The prover's encoding step*. The prover applies the following encoding to the witness state ρ.

Auth.Enc:
Parameters: $N(\cdot)$, a polynomially bounded function in $|x|$. (N functions as a security parameter.)
Input: An m-qubit state ρ.
The prover firstly applies a concatenated Steane code (which maps every one qubit to $N(|x|)$ qubits) to each qubit in ρ. (For details on the concatenated Steane code, see [BJSW16, Appendix A.6]. It will be important to Broadbent et al.'s purposes, and ours, that this code admits transversal applications of Clifford operations.) It then executes the following steps:

(a) Concatenate N trap qubits to the end of each logical qubit (alternatively, to the end of each N-qubit block) in the result of applying the concatenated Steane code to ρ. Each trap qubit is initialised uniformly at random to one of $|0\rangle, |+\rangle, |+_y\rangle$. ($|+_y\rangle$ here refers to the state $\frac{1}{\sqrt{2}}(|0\rangle + i|1\rangle)$.) Denote the string that records the choices of trap qubits for all m logical qubits by $t = t_1, \ldots, t_N \in \{|0\rangle, |+\rangle, |+_y\rangle\}^{mN}$.

(b) Permute each $2N$-tuple of qubits in the result of (a) according to a uniformly random permutation $\pi \in S_{2N}$. (Note that the same permutation π is applied to every $2N$-tuple.)

(c) Apply a Pauli one-time pad $X^a Z^b$, for uniformly random $a, b \in \{0,1\}^{2mN}$, to the entire $2mN$-qubit state.

Figure 1: The authentication code

We refer to t, π, a, b as 'the authentication keys' or 'the encoding keys'.

The prover's encoding applied to ρ is denoted by $E(\rho)$, and the procedure E is fully determined by the encoding key (t, π, a, b) which the prover chose to use. At this point, the prover sends the state $E(\rho)$ to the verifier, along with a commitment (using some perfectly binding, computationally concealing classical commitment protocol) to the tuple (π, a, b). (A commitment to the sequence of trap qubits t is unnecessary because, in a sense, the trap qubits exist only to check the verifier.) Let the prover's commitment string be denoted z.

2. *Coin-flipping protocol.* The prover and the verifier execute a coin-flipping protocol, choosing a string r of fixed length uniformly at random. This random string r determines a local Hamiltonian term $H_r = C_r^* |0^k\rangle \langle 0^k| C_r$ that is to be tested. (This step can be implemented **[?]** using the same classical commitment protocol that the prover employed in the previous step.)

3. *Verifier's challenge.* The verifier applies the Clifford C_r transversally to the qubits on which the k-local Hamiltonian term H_r acts nontrivially, and measures them in the standard basis. It then sends the measurement results $u_{i_1}, \ldots, u_{i_k}$ which it obtained to the prover. (Each u_i is a $2N$-bit string, and $i_1, \ldots, i_k$ are the indices of the logical qubits on which the term H_r acts nontrivially.)

4. *Prover's response.* The prover receives the verifier's measurement results u, and firstly checks whether they cause a predicate $\tilde{Q}(t, \pi, a, b, r, u)$ to be satisfied. (We will explain the predicate $\tilde{Q}$ in more detail shortly. Intuitively, $\tilde{Q}$ is satisfied if and only if both verifier *and* prover behaved honestly. Note that we have used the notation $\tilde{Q}$ to represent this predicate, while

the authors of [BJSW16] simply call it Q. We add the tilde in order to differentiate their predicate from our predicate Q, the latter of which we define in Definition 3.5.) If $\tilde{Q}$ is not satisfied, the prover aborts, causing the verifier to reject. If $\tilde{Q}$ is satisfied, then the prover proves to the verifier, using an NP zero-knowledge protocol, that there exists randomness s_P and an encoding key (t, π, a, b) such that $z = \mathsf{commit}((\pi, a, b), s_P)$ and $\tilde{Q}(t, \pi, a, b, r, u) = 1$.

We now describe the predicate $\tilde{Q}$ in precise terms. For convenience, Broadbent et al. define a predicate $\tilde{R}_r$, which represents the prover's check after it reverses the effects on u of the one-time pad $X^a Z^b$, and then proceed to define $\tilde{Q}$ in terms of $\tilde{R}_r$. (As with Q, we denote the predicate found in [BJSW16] by $\tilde{R}_r$ instead of R_r, in order to differentiate it from our R_r, which we define in Definition 3.2.) We quote below the definition of $\tilde{R}_r$ used in [BJSW16]. (Note that we have altered their notation slightly: the strings that they call y_i and z_i we have called p_i and q_i, and the strings that they call c and d we have called e and f, to avoid conflicts.)

Definition 2.1 (Definition of $\tilde{R}_r$)**.** *Let $u_{i_1}, \ldots, u_{i_k}$ be the measurement results that the verifier sent to the prover in step 3. For each $i \in \{i_1, \ldots, i_k\}$, define strings p_i, q_i in $\{0,1\}^N$ such that $\pi(p_i \| q_i) = u_i$ (alternatively: $\pi^{-1}(u_i) = p_i \| q_i$). The predicate $\tilde{R}_r(t, \pi, u)$ takes the value 1 if and only if the following two conditions are met:*

1. *$p_i \in \mathcal{D}_N$ for every $i \in \{i_1, \ldots, i_k\}$, and $p_i \in \mathcal{D}_N^1$ for at least one index $i \in \{i_1, \ldots, i_k\}$. ($\mathcal{D}_N = \mathcal{D}_N^0 \cup \mathcal{D}_N^1$ is the set of all valid classical N-bit codewords of the concatenated Steane code. We note, as an aside, that $\mathcal{D}_N^0$ and $\mathcal{D}_N^1$ are both sets of size 8^t, where t is an even positive integer such that $7^t = N$; as such, $\mathcal{D}_N$ is polynomially sized.)*
2. *$\langle q_{i_1} \cdots q_{i_k}| \, C_r^{\otimes n} \, |t_{i_1} \cdots t_{i_k}\rangle \neq 0$.*

Here $|t_{i_1} \cdots t_{i_k}\rangle$ is the state of mN qubits obtained by tensoring $|0\rangle, |+\rangle$ and $|+_y\rangle$ in the natural way.

Now we define the predicate $\tilde{Q}(t, \pi, a, b, r, u)$ in terms of $\tilde{R}_r$:

Definition 2.2 (Definition of $\tilde{Q}$)**.** *Let $e_1, \ldots, e_m, f_1, \ldots, f_m \in \{0,1\}^{2N}$ be the unique strings such that*

$$C_r^{\otimes 2N}(X^{a_1} Z^{b_1} \otimes \cdots \otimes X^{a_m} Z^{b_m}) = \alpha(X^{e_1} Z^{f_1} \otimes \cdots \otimes X^{e_m} Z^{f_m}) C_r^{\otimes 2N} \, ,$$

for some $\alpha \in \{1, i, -1, -i\}$. (It is possible to efficiently compute $e = e_1, \ldots, e_m$ and $f = f_1, \ldots, f_m$ given a, b and C_r.) The predicate $\tilde{Q}$ is defined by

$$\tilde{Q}(t, \pi, a, b, r, u) = \tilde{R}_r(t, \pi, u \oplus e_{i_1} \cdots e_{i_k}) \, .$$

Definition 2.3. *For notational convenience, we define a quantum procedure $Dec_{t,\pi,a,b}$ on a $2nN$-qubit state ρ as follows:*

1. *Apply $X^a Z^b$ to ρ, to obtain a state ρ'.*
2. *Apply π^{-1} to each $2N$-qubit block in the state ρ', to obtain a state ρ''.*
3. *Discard the last N qubits of each $2N$-qubit block in ρ'', to obtain a state ρ'''.*
4. *Return ρ'''.*

Intuitively, we think of $Dec_{t,\pi,a,b}$ as an inverse to the prover's encoding procedure E.

2.3 Cryptographic primitives

2.3.1 Homomorphic encryption (with circuit privacy)

The following definitions are taken (with modifications) from [BDPMW16].

A homomorphic encryption scheme $\mathsf{FHE} = (\mathsf{FHE.Gen}, \mathsf{FHE.Enc}, \mathsf{FHE.Dec}, \mathsf{FHE.Eval})$ is a quadruple of PPT algorithms which operate as follows:

- $\mathsf{FHE.Gen}(1^\lambda)$: given the security parameter λ, outputs a secret key sk, a public key pk and an evaluation key evk.
- $\mathsf{FHE.Enc}(pk, \mu)$: using the public key pk, encrypts a message $\mu \in \{0,1\}$ into a ciphertext c and outputs c.
- $\mathsf{FHE.Dec}(sk, c)$: using the secret key sk, decrypts a ciphertext c to recover a message $\mu \in \{0,1\}$.
- $\mathsf{FHE.Eval}(evk, f, c_1, \ldots, c_\ell)$: using the evaluation key evk, applies a function $f : \{0,1\}^\ell \to \{0,1\}$ to ciphertexts $c_1, \ldots, c_\ell$ and outputs a ciphertext c_f.

Evaluation correctness. We say that the $\mathsf{FHE.Eval}$ algorithm correctly evaluates all functions in $\mathcal{F}$ if, for any function $f \in \mathcal{F} : \{0,1\}^\ell \to \{0,1\}$ and respective inputs $x_1, \ldots, x_\ell \in \{0,1\}$ it holds that

$$\Pr[\mathsf{FHE.Dec}(sk, \mathsf{FHE.Eval}(evk, f, c_1, \ldots, c_\ell)) = f(x_1, \ldots, x_\ell)] = 1 - \mathsf{negl}(\lambda) \ ,$$

where $(pk, sk, evk) \leftarrow \mathsf{FHE.Gen}(1^\lambda)$ and $c_i \leftarrow \mathsf{FHE.Enc}(pk, x_i)$.

Semantic security. We say that FHE is semantically secure (or IND-CPA secure) if no PPT adversary $\mathcal{A}$ can distinguish between the encryptions of two plaintexts which it is allowed to specify. More formally, let $(pk, sk, evk) \leftarrow \mathsf{FHE.Gen}(1^\lambda)$ and $\mathcal{O}_b(\mu_0, \mu_1) = \mathsf{FHE.Enc}(pk, \mu_b)$ for $b \in \{0,1\}$. Then FHE is IND-CPA secure if

$$\left|\Pr[\mathcal{A}^{\mathcal{O}_0}(1^\lambda) = 1] - \Pr[\mathcal{A}^{\mathcal{O}_1}(1^\lambda) = 1]\right| = \mathsf{negl}(\lambda) \ ,$$

where the probability is taken over the internal coins of $\mathsf{FHE.Gen}$, $\mathsf{FHE.Enc}$ and $\mathcal{A}$.

Levelled FHE. A fully homomorphic encryption scheme FHE is *levelled* if

1. $\mathsf{FHE.Gen}$ takes an additional parameter 1^L, and
2. $\mathcal{F}$ (the set of functions which can be correctly evaluated using evk) is restricted to functions that can be expressed as branching programs of depth at most L.

We implicitly assume, in the text of this paper, that FHE is levelled; the restriction on program depth can be handled by choosing L to be a sufficiently large polynomial in λ such that 'overflow' will not occur.

Circuit privacy. For the protocol described in section 3, we require an additional property of FHE known as *circuit privacy*. We formalise the property of circuit privacy in the existence of a fifth PPT algorithm, $\mathsf{FHE.Refresh}$, which operates as follows:

- $\mathsf{FHE.Refresh}(pk, evk, c)$: takes a ciphertext c, and outputs another ciphertext c' such that $\mathsf{FHE.Dec}(sk, c) = \mathsf{FHE.Dec}(sk, c')$

We say that FHE is a scheme *with circuit privacy* if there exists a PPT algorithm Enc^* (not to be confused with FHE.Enc) such that, for any branching program Π of length $L = \mathrm{poly}(\lambda)$ on ℓ variables, and any $x_1, \ldots, x_\ell \in \{0,1\}$, the following holds:

$$\Big(\mathsf{FHE.Refresh}\big(\mathsf{FHE.Eval}(evk, \Pi, c_1, \ldots, c_\ell)\big), c_1, \ldots, c_\ell, 1, pk, sk\Big)$$
$$\approx_s \Big(\mathsf{Enc}^*\big(pk, 1^\lambda, \Pi(x_1, \ldots, x_\ell), 1^L, c_1, \ldots, c_\ell\big), c_1, \ldots, c_\ell, 1, pk, sk\Big)$$

where $(pk, sk) \leftarrow \mathsf{FHE.Gen}(1^\lambda)$ and $c_i \leftarrow \mathsf{FHE.Enc}(pk, x_i)$.

Intuitively, circuit privacy guarantees that 'refreshed' ciphertexts are distributed like some known distribution which can be sampled from without the knowledge of Π. (This is formalised by the property that the distribution of refreshed ciphertexts is negligibly close to the distribution of ciphertexts which Enc^* produces, where Enc^* does not have access to information about Π. In [BDPMW16], Enc^* is known as Sim.) In other words, refreshed ciphertexts do not leak information about the nature of the homomorphic computations that were performed on the ciphertext prior to the refresh operation (excepting information about the length of Π).

Remark 2.4. *We refer to ciphertexts which are sampled by* Enc^* *as 'fresh' ciphertexts. This makes it convenient to state that ciphertexts originating from* FHE.Refresh *are 'indistinguishable from fresh ciphertexts'.*

Levelled fully homomorphic encryption schemes with circuit privacy are known to exist, predicated on the assumption that LWE is (quantum) computationally intractable [BDPMW16]. The circuit-private FHE scheme exhibited in [BDPMW16] actually has $\mathsf{FHE.Refresh} = I$, where I is the identity circuit (meaning that circuit privacy is 'built into' evaluation). For clarity we explicitly consider a 'refreshing' algorithm in our use of the primitive.

2.3.2 Commitment scheme

In the protocol described in section 3, we will twice make use of a commitment scheme consisting of a tuple of PPT algorithms (gen, commit, reveal, verify) which operate as follows:

- $\mathsf{gen}(1^\ell)$ takes as input a security parameter, and generates a public key pk and a secret key sk. (We remark that sk is never used in the real commitment procedure; it exists purely for simulation purposes.)
- $\mathsf{commit}(pk, b, s)$ takes as input a public key pk, a bit $b \in \{0,1\}$ to which to commit, and a random string s, and produces a commitment string z.
- $\mathsf{reveal}(pk, z, b, s)$ outputs the inputs it is given.
- $\mathsf{verify}(pk, z, b, s)$ takes as argument a purported public key, commitment string, committed bit and random string, and outputs either 1 (accept) or 0 (reject).

We require the following properties of this commitment scheme:

- *Perfectly binding*: If $\mathsf{commit}(pk, b, s) = \mathsf{commit}(pk, b', s')$, then $b = b'$.
- *(Quantum) computationally concealing*: For any public key $pk \leftarrow \mathsf{gen}(1^\ell)$ and any two messages b, b', the distributions over s of $\mathsf{commit}(pk, b, s)$ and $\mathsf{commit}(pk, b', s)$ are quantum computationally indistinguishable.
- *Extractable*: There is a PPT algorithm $\mathsf{recover}(pk, sk, z)$ which, for any $z \leftarrow \mathsf{commit}(pk, b, s)$, outputs the unique b such that $z = \mathsf{commit}(pk, b, s)$.

Instantiation based on LWE. There are a number of folklore results on commitment schemes based on the intractability of LWE. Below, we sketch a scheme (one sufficient for our purposes) which is inspired by the LWE-based public-key encryption scheme presented by Regev in [Reg09]. Before we introduce our commitment scheme, we firstly provide an overview of the [Reg09] public-key encryption scheme.

Definition 2.5 ([Reg09] public-key encryption scheme)**.** *Let m, n be security parameters, and let $q = \mathrm{poly}(n)$ be a prime. The [Reg09] public-key encryption scheme is a tuple of algorithms $(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ of the following descriptions:*

- *$\mathsf{KeyGen}(1^n, 1^m, q)$ generates a uniformly random matrix $A \in \mathbb{Z}_q^{m \times n}$, along with a uniformly random vector $\vec{s} \in \mathbb{Z}_q^n$ and a 'short' noise vector $e \in \mathbb{Z}_q^n$. (We say that the vector e is 'short' because it has bounded norm. We will not go into detail here about the distribution from which e is drawn. It suffices to state that this distribution is well-defined and efficiently sampleable.) It outputs $pk = (A, A\vec{s} + e)$ and $sk = \vec{s}$.*
- *$\mathsf{Enc}(pk, b)$ takes a message $b \in \{0,1\}$ and chooses a vector $w \in \{0,1\}^m$ uniformly at random. It outputs a ciphertext $c = (c_0, c_1) = (w^T A, w^T(A\vec{s} + e) + b \cdot \lfloor \frac{q}{2} \rfloor)$.*
- *$\mathsf{Dec}(sk, c)$ is 0 if $c_1 - \langle c_0, \vec{s} \rangle$ is closer to 0 than to $\lfloor \frac{q}{2} \rfloor \bmod q$, and 1 otherwise.*

We refer the reader to [Reg09, Section 5] for the security analysis of the cryptosystem defined in Definition 2.5.

We now provide a sketch of an LWE-based commitment scheme satisfying the properties specified at the start of this section. In addition to the [Reg09] cryptosystem, we use the algorithms $\mathsf{GenTrap}$ and Invert from [MP12, Theorem 2]. $\mathsf{GenTrap}$ is an algorithm which generates a matrix A with a distribution negligibly close to the distribution of A in Definition 2.5, and in addition generates a *trapdoor* R for A that allows the algorithm Invert to (given R) efficiently recover $(\vec{s}, e)$ from any $A\vec{s} + e$.

Definition 2.6 (Perfectly binding, computationally hiding, extractable commitment scheme based on LWE)**.** *Let m, n be security parameters, and let $q = \mathrm{poly}(n)$ be a prime.*

- *$\mathsf{gen}(1^n, 1^m, q)$ outputs $(A, R) \leftarrow \mathsf{GenTrap}(1^n, 1^m, q)$ as (pk, sk). The distribution of A is negligibly close to uniform.*
- *$\mathsf{commit}(pk, b, s)$ uses s to generate*
  1. *a uniformly random vector $\vec{s} \in \mathbb{Z}_q^n$,*
  2. *a short vector e sampled from the appropriate distribution, and*

3. *a uniformly random string* $w \in \{0,1\}^m$.

It then outputs $z = (z_0, z_1, z_2, z_3) = (A, A\vec{s} + e, w^T A, w^T(A\vec{s} + e) + b \cdot \lfloor \frac{q}{2} \rfloor)$.

- $\mathsf{reveal}(pk, z, b, s)$ *outputs the inputs it is given.*
- $\mathsf{verify}(pk, z, b, s)$ *checks that* $e(s)$ *is sufficiently short, checks that* $z_1 = z_0\vec{s} + e(s)$*, and then decrypts* (z_2, z_3) *using* $w(s)$ *and* $\vec{s}(s)$*. It outputs 1 if and only if the result is* b*.*

In our setting, we are guaranteed that A is generated honestly (i.e. that it comes from running $\mathsf{GenTrap}$), because A is embedded into a CRS. Given that this is so, the commitment scheme we have defined in Definition 2.6 is perfectly binding because, for an A originating from $\mathsf{GenTrap}$ and a sufficiently short e, the vector $A\vec{s} + e$ perfectly determines $\vec{s}$. Then, if $\vec{s}$ is determined, $w^T(A\vec{s} + e) + b \cdot \lfloor \frac{q}{2} \rfloor$ cannot decrypt to a value other than b, by correctness of the [Reg09] encryption scheme. That the commitment scheme is computationally hiding follows directly from the IND-CPA security of the [Reg09] scheme.

Finally, the commitment scheme is extractable because the trapdoor R allows an algorithm to efficiently invert A (as described in [MP12, Theorem 12]). Specifically, $\mathsf{recover}(pk = A, sk = R, z)$ runs $\mathsf{Invert}(R, A, A\vec{s} + e)$ to recover $(\vec{s}, e)$, and then recovers b by decrypting $w^T(A\vec{s} + e) + b \cdot \lfloor \frac{q}{2} \rfloor$.

Remark 2.7. *It is possible to strengthen the commitment scheme in Definition 2.6 so that it satisfies the following two properties:*

1. *The perfectly binding property holds for the randomness as well as for the committed bit. In other words, if* $\mathsf{commit}(pk, b, s) = \mathsf{commit}(pk, b', s')$*, then we can conclude not only that* $b = b'$ *but also that* $(b, s) = (b', s')$*.*
2. *The algorithm* $\mathsf{recover}(pk, sk, z)$ *recovers the randomness* s *as well as the committed bit* b *from a commitment* z*.*

In order to achieve these two properties, we simply have gen *sample a second matrix* A' *from* $\mathsf{GenTrap}$ *during key generation, along with its trapdoor* R'*, and set* $pk = (A, A'), sk = (R, R')$*. Then we have* commit *append the value* $z_4 = A'w + e'$ *to the commitment* z *that it generates, for some appropriately sampled short vector* e'*, where* w *is the uniformly random string in* $\{0,1\}^m$ *that is used to compute* z_2 *and* z_3*. We also modify* verify *appropriately, so that, in addition to the checks it performs in Definition 2.6, it also checks that* $z_4 = A'w + e'$ *and that* $(z_2, z_3) = (w^T A, w^T(A\vec{s} + e) + b \cdot \lfloor \frac{q}{2} \rfloor)$*. The purpose of these changes is to ensure that the value of* w *which is used to produce a given commitment* z *is also perfectly determined by the value of* z *(which was not necessarily the case before, as* A *is in general not invertible). Because* A' *is sampled independently from* A*, the commitment scheme remains computationally hiding if LWE is computationally intractable.*

2.4 Argument systems

2.4.1 Interactive quantum machines

The definitions of *interactive quantum machines*, their *executions* and *oracle access* to an interactive quantum machine are taken largely from [Unr12].

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 $\mu \in \mathbb{N}$, a classical input $x \in \{0,1\}^*$, and quantum input $|\Phi\rangle$, which is stored in S. Formally, machine M is specified by a family of circuits $\{M_{\mu x}\}_{\mu\in\mathbb{N},x\in\{0,1\}^*}$, and a family of integers $\{r_{\mu x}\}_{\mu\in\mathbb{N},x\in\{0,1\}^*}$. $M_{\mu x}$ is the quantum circuit that M performs on the registers S and N upon invocation. $r_{\mu x}$ determines the total number of messages/invocations. We might omit writing the security parameter when it is clear from the context. We say that M is *quantum-polynomial-time* (QPT for short) if the circuit $M_{\mu x}$ has polynomial size in $\mu + |x|$, the description of the circuit is computable in deterministic polynomial time in $\mu + |x|$ given μ and x, and $r_{\mu,x}$ is polynomially bounded in μ and x.

Execution of interactive quantum machines For a pair of interactive quantum machines M and M' with internal registers S, S' respectively and network register N, a security parameter μ, a pair of quantum states $|\Phi\rangle, |\Phi'\rangle$ and a pair of strings x, x', we define the *execution* $(M(x, |\Phi\rangle), M'(x', |\Phi'\rangle))$ via the following process: initialize registers $\mathsf{S}, \mathsf{S}', \mathsf{N}$ as $|\Phi\rangle, |\Phi'\rangle$ and $|0\rangle$; alternately apply $M_{\mu x}$ to registers S and N and $M_{\mu x'}$ to registers S' and N; stop applying $M_{\mu x}$ after $r_{\mu x}$ times and $M_{\mu x'}$ after $r'_{\mu x}$ times, and then measure the S register in the computational basis. Let $\langle M(x, |\Phi\rangle), M'(x', |\Phi'\rangle)\rangle$ be the random variable for the outcome of this measurement. For a k-tuple of interactive machines $M_1, \ldots, M_k$, we assume that there is a message register N_{ij} between any two machines. Machine M_i has an internal register S_i, an invocation register INV_i, and is specified by a unitary $M_{i,\mu x}$, which acts on S_i, on all invocation registers INV_j, and on all of M_i's message registers (N_{ij} for all j). $M_{i,\mu x}$ has the additional property that it always leaves all invocation registers in the state $|0\rangle$, except for one in the state $|1\rangle$. This determines the next machine to be invoked.

Oracle access to an interactive quantum machine We say that a quantum algorithm A has oracle access to an interactive quantum machine M (and we write this as A^M, or sometimes $A^{|M\rangle}$ to emphasize that M is a quantum machine and that oracle access includes the ability to apply the inverse of M) to mean the following. Besides the security parameter and its own classical input x, we allow A to execute the quantum circuit $M_{\mu x}$ specifying M, and 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 $M_{\mu x}$ or its inverse.

2.4.2 Argument systems with setup

First we define the kinds of relations that underlie our argument systems. Classically, a relation over finite sets $\mathcal{X} \times \mathcal{Y}$ is a subset $R \subseteq \mathcal{X} \times \mathcal{Y}$. An NP relation $R = \{(x, w) : V_{|x|}(x, w) = 1\}$ has the additional property that given any $x \in \mathcal{X}$ and $w \in \mathcal{Y}$, the claim that $(x, w) \in R$ can be verified by a uniformly generated family of circuits $V = \{V_n\}$ (the "verifier").

In the quantum case the "input" x (the first argument to the relation) remains classical, but the "witness" w (the second argument) can be a quantum state $|\psi\rangle$. Before we give our definition of a

QMA relation we introduce some notation. Fix a uniformly generated family of polynomial-size quantum circuits $Q = \{Q_n\}_{n\in\mathbb{N}}$ such that for every n, Q_n takes as input a string $x \in \{0,1\}^n$ and a quantum state σ on $p(n)$ qubits (for some polynomial $p(n)$) and returns a single bit as output. For any $0 \leq \gamma \leq 1$ define

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

and

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

Note the presence of the parameter γ, that quantifies the expected success probability for the verifier; γ can be thought of as a measure of the "quality" of a witness $|\psi\rangle$ (or mixture theoreof, as represented by the density matrix σ) that is sufficient for the witness to be acceptable with respect to the relation R.

Definition 2.8 (QMA relation)**.** *A* QMA relation *is specified by triple (Q,α,β) where $Q = \{Q_n\}_{n\in\mathbb{N}}$ 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 and returns a single bit, and $\alpha,\beta : \mathbb{N} \to [0,1]$ are such that $\alpha(n) - \beta(n) \geq 1/p(n)$ for some polynomial p and all $n \in \mathbb{N}$. The* QMA *relation associated with (Q,α,β) is the pair of sets $R_{Q,\alpha}$ and $N_{Q,\beta}$.*

For (Q,α,β) a QMA *relation, we define the (promise)* language $L = (L_{yes}, L_{no})$ specified by (Q,α,β) *as*

$$L_{yes} = \bigcup_{n\in\mathbb{N}} \left\{x \in \{0,1\}^n | \exists \sigma \in \mathrm{D}(\mathbb{C}^{p(n)}) \text{ s.t. } (x,\sigma) \in R_{Q,\alpha}\right\}, \tag{1}$$

and $L_{no} = N_{Q,\beta}$.

Note that in contrast to an NP relation, we define a QMA relation using two sets: the first set, $R_{Q,\alpha}$, is the set of (instance, witness) pairs that are deemed to form part of the relation. The second set, $N_{Q,\beta}$, is the set of instances that are deemed to be such that they are in relation to no witness. Some instances may lie in neither (the projection of) $R_{Q,\alpha}$ or $N_{Q,\beta}$; this is analogous to the necessity for a "promise" between the completeness and soundness parameters α and β in the definition of the class QMA, that do not appear in the definition of NP. In particular, note that whenever $\alpha - \beta > 1/\operatorname{poly}(n)$ the language L defined in (1) lies in QMA.

Definition 2.9. *A* protocol with setup *is a triple of interactive machines (S,P,V) with the following properties:*

1. *$S = \{S_{\mu n}\}_{\mu\in\mathbb{N}}$ depends on the security parameter μ and an instance size n, takes no input and returns a classical output in the message registers N_{SP} and N_{SV}. When the output in both registers is the same, we refer to it as "common reference string".*
2. *Each of P and V has two phases: $P = (P_1, P_2)$ and $V = (V_1, V_2)$. $P_1 = \{P_{1,\mu n}\}$ and $V_1 = \{V_{1,\mu n}\}$ are interactive machines that depend on the security parameter μ and an instance size parameter n, take a classical message input in register N_{SP} and N_{SV} respectively and return a quantum message as output in registers $\mathsf{N}_{P_1P_2}$ and $\mathsf{N}_{V_1V_2}$ respectively. $P_2 = \{P_{2,\mu n}\}$ and $V_2 = \{V_{2,\mu n}\}$ are interactive machines that depend on the security parameter μ and an input size n. V_2 takes as input the output*

of V_1, in register $\mathsf{N}_{V_1V_2}$, as well as an instance x such that $|x| = n$. P_2 takes as input the output of P_1, in register $\mathsf{N}_{P_1P_2}$, an instance x such that $|x| = n$, and a quantum state ρ. V_2 returns a single bit $b \in \{0,1\}$ as output, and P_2 returns no output. If $b = 1$ then we say that V accepts, and otherwise we say that it rejects.

We refer to the first phase of P and V as the *preprocessing phase*, and to the second phase as the *instance-dependent phase*.

Definition 2.10. *Let (Q, α, β) be a* QMA *relation and $s, c : \mathbb{N} \to [0,1]$. An* argument system *(with setup) for (Q, α, β), with completeness c and soundness s, is a protocol with setup (S, P, V) such that S, P, V are quantum polynomial-time and, in addition, the following hold:*

1. *(Completeness) For all $(x, \rho) \in R_{Q,\alpha}$, for all integer μ, the execution $(S, P(x, \rho), V(x))$ returns 1 with probability at least $c(\mu)$.*
2. *(Soundness) For all $x \in N_{Q,\beta}$, all integer μ and all polynomial-time P^* the execution $(S, P^*(x), V(x))$ returns 1 with probability at most $s(\mu)$.*

When the second phase of a protocol with setup (S, P, V) consists of a single message from P to V we refer to it as a *non-interactive* protocol with setup. If it is a an argument system with setup, we refer to it as a *non-interactive* argument system with setup. When the first phase involves some communication between P and V, we specify that it is a non-interactive argument system with setup *and preprocessing*. When S outputs a common reference string (as defined in 2.9), we refer to it as an argument system *with CRS setup* (possibly with preprocessing).

Note that Definition 2.10 requires that the execution $(S, P(x, \rho), V(x))$ returns 1 with probability at least $c(\mu)$. In the case of sequential or parallel repetition of a protocol, it may not be possible for the prover to succeed with a single copy of the witness ρ as input. In this case we may considering relaxing the definition as follows.

Definition 2.11 (Completeness of argument system with setup — alternative definition)**.** *There exists a polynomial $q > 0$, such that for all $(x, \rho) \in R_{Q,\alpha}$, for all integers μ, the execution $(S, P(x, \rho^{\otimes q(\mu)}), V(x))$ returns 1 with probability at least $c(\mu)$.*

We will clarify, whenever we refer to an argument system with setup, which definition we refer to.

2.5 Zero-knowledge arguments

2.5.1 Zero-knowledge

The notion of Zero-Knowledge in the presence of quantum verifiers and quantum auxiliary information was first formulated by Watrous [Wat09]. Here, we adapt Unruh's version of this definition. We first give a definition for argument systems with setup for NP relations, and then we give the generalization to QMA relations.

Definition 2.12. *An interactive argument system with setup (S, P, V) for an* NP *relation R is quantum computational zero-knowledge if for every quantum polynomial-time verifier V^* there is a quantum polynomial-time simulator Sim such that, for any quantum polynomial-time distinguisher D and any polynomial $l > 0$, there is a negligible ν such that for any $(x, w) \in R$ with $|x|, |w| \leq l(\mu)$, and for any quantum state $|\Psi\rangle$, we have:*

$$\Pr\left(b = 1 : ZE \leftarrow |\Psi\rangle, (S, P(x, w), V^*(Z)), b \leftarrow D(Z, E)\right) - \Pr\left(b = 1 : ZE \leftarrow |\Psi\rangle, S(x, Z), b \leftarrow D(Z, E)\right) \leq \nu(\mu) .$$

Here $ZE \leftarrow |\Psi\rangle$ denotes that the quantum registers Z,E are initialized jointly in state $|\Psi\rangle$. And $(S, P(x, \sigma), V^*(Z))$ denotes an execution where V^* gets access to the quantum register Z. Note that after that execution V^* may have changed the state of Z. $S(x, Z)$ also gets access to and may change Z.

The only difference in the definition for QMA relations is that the prover receives a state σ such that $(x, \sigma) \in R_{Q,\alpha}$. We report the full definition for completeness.

Definition 2.13. *An interactive argument system with setup (S, P, V) for a* QMA *relation (Q, α, β) is quantum computational zero-knowledge if for every quantum polynomial-time verifier V^* there is a quantum polynomial-time simulator Sim such that, for any quantum polynomial-time distinguisher D and any polynomial $l > 0$, there is a negligible ν such that for any $(x, \sigma) \in R_{Q,\alpha}$ with $|x| \leq l(\mu)$, and for any quantum state $|\Psi\rangle$,*

$$\Pr\left(b = 1 : ZE \leftarrow |\Psi\rangle, (S, P(x, \sigma), V^*(Z)), b \leftarrow D(Z, E)\right) - \Pr\left(b = 1 : ZE \leftarrow |\Psi\rangle, S(x, Z), b \leftarrow D(Z, E)\right) \leq \nu(\mu) .$$

2.5.2 Non-interactive zero-knowledge arguments for QMA

In this section we formally define the notion of NIZK arguments for QMA. This is the main object we aim to construct in this paper.

Definition 2.14. *An NIZK argument system with CRS setup for a* QMA *relation (Q, α, β) is a non-interactive protocol with CRS setup $\Pi = (S, P, V)$ such that:*

1. *Π is quantum computational zero-knowledge, as in Definition 2.13.*
2. *An execution of Π involves a single message from P to V.*
3. *Completeness of Π is $1 - \text{negl}$.*

If P and V exchange additional messages in the preprocessing phase (but not in the instance-dependent phase), we refer to Π as an NIZK argument sytem with CRS setup and preprocessing.

Definition 2.15 (Argument system for a language in QMA)**.** *We say that a language $L \in QMA$ has an NIZK argument system with CRS setup (and preprocessing) if there exists a QMA relation $(Q, 1 - \text{negl}(n), \text{negl}(n))$ which specifies L and such that there exists a NIZK argument system with setup (and preprocessing) (S, P, V) for $(Q, 1 - \text{negl}(n), \text{negl}(n))$.*

Using the amplification technique from [MW05] it follows that any QMA relation that specifies a QMA language L can be amplified to a QMA relation of the form $(Q, 1 - \text{negl}(n), \text{negl}(n))$, such that moreover the reduction preserves the honest quantum witness for positive instances of L.

2.5.3 Non-interactive zero-knowledge for NP

A building block in our construction of NIZK argument systems for QMA are non-interactive argument systems for NP in the CRS model (or with CRS setup). Based on the work of [CLW19, PS19] it is possible to construct such argument systems satisfying both adaptive soundness and adaptive zero-knowledge assuming only the hardness of the LWE problem. Here, we require such arguments that maintain the soundness and zero-knowledge properties in case the adversarial party may be quantum. We give the definitions and sketch how quantum security follows almost immediately by adapting the same arguments from [CLW19] and using quantum-security of the LWE-based correlation intractable hash family from [PS19].

For an NP relation R we let $\mathcal{L}(R) = \{x : \exists w, (x,w) \in R\}$.

Definition 2.16. *An NIZK argument system for an* NP *relation R is a non-interactive protocol with setup $\Pi = (S, P, V)$ such that in addition:*

1. *$S = \{S_{\mu n}\}$ is a classical polynomial-time family of circuits that depend on the security parameter μ and an instance size parameter n and return a common reference string* crs.
2. *$P = \{P_{\mu n}\}$ is a classical polynomial-time family of circuits that take as input* crs*, an instance x and a witness w and return a proof π.*
3. *$V = \{V_{\mu n}\}$ is a classical polynomial-time family of circuits that take as input* crs*, an instance x and a proof π and return a decision $b \in \{0,1\}$. If $b = 1$ then we say that V accepts, and otherwise we say that it rejects.*
4. *P and V do not require a setup phase (other than receiving the common reference string* crs *from S).*
5. *Π is quantum computational zero-knowledge, as in Definition 2.12.*

The argument system should satisfy the completeness requirement: for every integer μ, n, for every $(x,w) \in R$ such that $|x| = n$, $V_{\mu n}(\text{crs}, x, \pi) = 1$ with probability 1 when $\text{crs} \leftarrow S_{\mu n}()$ *and* $\pi \leftarrow P_{\mu n}(\text{crs}, x, w)$.

We define adaptive soundness and adaptive ZK for the case of quantum adversaries.

Definition 2.17 (Adaptive soundness)**.** *A NIZK argument Π for R is* adaptively sound *if for every quantum polynomial time $P^* = \{P^*_{\mu n}\}$ there is a negligible function ν such that for all μ,*

$$\Pr_{\text{crs}\leftarrow S_{\mu n}(),\ (x,\pi)\leftarrow P^*_{\mu n}(\text{crs})} \left(x \notin \mathcal{L}(R) \wedge V(\text{crs}, x, \pi) = 1\right) \leq \nu(\mu) \ .$$

Definition 2.18 (Adaptive zero-knowledge)**.** *A NIZK argument Π for R is* adaptive zero-knowledge *if for every quantum polynomial time $V^* = \{V^*_{\mu n}\}$ there is a quantum polynomial time simulator $Sim = \{Sim_{\mu n}\}$ such that for all quantum states σ the following ensembles are computationally indistinguishable (with respect to quantum polynomial time distinguishers):*

$$\left\{\text{crs} \leftarrow S_{\mu n}()\ ,\ (x, w, \tau) \leftarrow V^*_{\mu n}(\text{crs}, \sigma)\ :\ (\text{crs}, P_{\mu n}(\text{crs}, x, w), \tau)\right\}$$

and

$$\left\{Sim_{\mu n}(\sigma)\right\} \ .$$

Note that in the first ensemble, τ is a quantum state that may be correlated in an arbitrary way with the classical (x,w).

Lemma 2.19. *For any* NP *relation* R *there is a NIZK argument system that is adaptively sound and adaptively ZK against quantum adversaries, assuming the hardness of LWE and the existence of statistically binding and quantum computationally concealing commitment schemes.*

Proof. Fix μ and n and let Com be a statistically binding and quantum computationally concealing commitment scheme that can be used to commit to $n \times n$ Boolean matrices. We write a key for Com as pk.

Our starting point is the 3-message protocol for graph Hamiltonicity from [FLS99], as described in [CLW19, Section 5.1], and the NIZK argument $\tilde{\Pi}$ for it given in [CLW19, Construction 5.2].

Argument system. We recall the protocol to establish notation, referring to [CLW19, Section 5.1] for more details. In the 3-message protocol, the prover first sends a commitment $a = \mathsf{Com.commit}(pk, H, s)$ to a graph H that is a uniformly random permutation $\pi(C_n)$, for C_n the adjacency matrix of a canonically fixed cycle. The verifier sends a random challenge $e \in \{0,1\}$. If $e = 0$, the prover reveals H. If $e = 1$, the prover, given $(x, w) \in R$ where x is a graph and w a Hamiltonian cycle, selects a permutation σ such that $\sigma^{-1}(C_n)$ is the cycle w. The prover sends $\pi \circ \sigma$ as well as decommitments to entries of H that correspond to non-edges of $\pi \circ \sigma(G)$. (In the actual protocol, the above steps are repeated t times in parallel. For simplicity, we describe the protocol for a single run.)

Next we recall the non-interactive variant, protocol $\tilde{\Pi}$ in [CLW19, Construction 5.2]. Here $\mathrm{crs} = (pk, k)$ where pk is a key for the commitment scheme and k is a key for a family $\mathcal{H}$ of hash functions satisfying certain properties (correlation intractability and programmability, see [CLW19]). The prover then computes a as above, sets $e = h_k(a)$, and computes the third message z as above. The prover returns $\pi = (a, e, z)$.

The only difference with [CLW19, Construction 5.2] is that we use a statistically binding commitment scheme instead of their PKE, as this is needed for soundness. We instantiate $\mathcal{H}$ using the correlation intractable family of hash functions from [PS19].

In [CLW19, Theorem 5.5] it is shown that $\tilde{\Pi}$ is a NIZK argument system for NP in the common reference string model satisfying both adaptive soundness and adaptive zero-knowledge against classical adversaries. We briefly sketch how their argument extends to the quantum case.

Adaptive soundness. First we consider adaptive soundness. Here the argument is essentially identical; we follow the proof of [CLW19, Lemma 5.6]. Let $(\mathrm{crs}, x, \pi = (a, e, z))$ be such that $V(\mathrm{crs}, x, \pi) = 1$. Then for every $i \in \{1, \ldots, t\}$ (recall that the basic 3-message protocol is repeated t times before applying Fiat-Shamir), using that the commitment is unconditionally binding it must be that $e_i = 0$ if and only if a_i is a commitment to a cycle. This means that any P^* which breaks the adaptive soundness condition from Definition 2.17, given only crs, returns $(a, e) \in R$, where is the set of pairs where for all i, a_i is a commitment to a cycle if and only if $e_i = 0$, with probability at least $\nu(\mu)$. By an averaging argument, there exists a fixed crs for which the same holds. This contradicts the correlation intractability of $\mathcal{H}$ which by [PS19] reduces to the quantum hardness of LWE.

Adaptive zero-knowledge. Next we consider adaptive zero-knowledge. For this we construct a simulator, mostly following the proof of [CCRR18, Proposition 7.6]. Let V^* be quantum polynomial time. First, $Sim_{\mu n}(\sigma)$ generates a key pk for Com. Then, it selects uniformly random challenges $\{e_i\}$ and permutations $\{\pi_i\}$. If $e_i = 0$, it sets $a_i \leftarrow \mathsf{Com.commit}(pk, \pi_i(C_n), s)$. If $e_i = 1$, it sets $a_i \leftarrow \mathsf{Com.commit}(pk, 0, s)$ (for some randomness s).

The simulator then samples a key k^* such that $h_{k^*}(a_i) = e_i$ for all $i \in \{1, \dots, t\}$ and creates $\mathrm{crs} = (pk, k^*)$.[6] Then, the simulator executes $V^*(\mathrm{crs}, \sigma)$ to obtain (x, w, τ). Finally, if $e_i = 0$ the simulator sets $z_i = \pi_i(C_n)$. If $e_i = 1$ it selects a uniformly random permutation σ_i and sets z_i to be σ_i together with decommitments to the entries of a_i that correspond to non-edges of $\sigma_i(G)$.

This simulator is "straight-line simulator" and does not involve rewinding V^*. Therefore, the computational zero-knowledge property follows from the computationally hiding property of Com and the programmability of $\mathcal{H}$ (to argue that k^* is indistinguishable from a random key). □

2.6 Proofs and arguments of quantum knowledge

The content of this subsection, as it pertains to *proofs of quantum knowledge*, was written in collaboration with Broadbent and Grilo, and appears with slight differences in their concurrent work.

A *Proof of Knowledge (PoK)* is an interactive proof system for some relation R such that if the verifier accepts some input x with high enough probability, then she is "convinced" that the prover "knows" some witness w such that $(x, w) \in R$. This notion is formalized by requiring the existence of an efficient *extractor* K that is able to return a witness for x when given oracle access to the prover (including the ability to rewind its actions, in the classical case).

Definition 2.20 (Classical Proof of Knowledge)**.** *Let $R \subseteq \mathcal{X} \times \mathcal{Y}$ be a relation. A proof system (P, V) for R is a Proof of Knowledge for R with knowledge error κ if there exists a polynomial $p > 0$ and a polynomial-time machine K such that for any classical interactive machine P^*, any $\mu \in \mathbb{N}$, any polynomial $l > 0$, any instance $x \in \{0,1\}^n$ for $n = \mathrm{poly}(\mu)$ and any string y: if the execution $(P^*(x, y), V(x))$ returns 1 with probability $\varepsilon > \kappa(\mu)$, we have*

$$\Pr\left(\left(x, K^{P^*(x,y)}(x)\right) \in R\right) \geq p\left(\varepsilon - \kappa(\mu), \frac{1}{\mu}\right).$$

In this definition, y corresponds to the side information that P^* has, possibly including some w such that $(x, w) \in R$.

PoKs were originally defined only considering classical adversaries, and this notion was first studied in the quantum setting by Unruh [Unr12]. The first issue that arises in the quantum setting is to formalize the type of query that the extractor K is able to make. In order to do so, we assume that P^* always performs a fixed unitary operation U when invoked. Notice that this can be assumed without loss of generality since (i) we can always consider a purification of P^*, (ii) all measurements can be performed coherently, and (iii) P^* can keep track of the round of communication

[6] The possibility for selecting k^* in a way that is indistinguishable from a random key comes from the *programmability* of $\mathcal{H}$; see [CLW19].

in some internal register and U can implicitly control on this value. Then, the quantum extractor K has oracle access to P^* in the sense that it may perform U and $U^\dagger$ on the message register and private register of P^*, but has no direct access to the latter. We denote the extractor K with such oracle access to P^* by $K^{|P^*(x,\rho)\rangle}$, where ρ is some (quantum) side information held by P^*.

Definition 2.21 (Quantum Proof of (Classical) Knowledge)**.** *Let $R \subseteq \mathcal{X} \times \mathcal{Y}$ be a relation. A proof system (P, V) for R is a Quantum Proof of Knowledge for R with knowledge error κ if there exists a polynomial $p > 0$ and a quantum polynomial-time machine K such that for any quantum interactive machine P^*, any $\mu \in \mathbb{N}$, any polynomial $l > 0$, any instance $x \in \{0,1\}^n$ for $n = \text{poly}(\mu)$ and any state ρ: if the execution $(P^*(x,\rho), V(x))$ returns 1 with probability $\varepsilon > \kappa(\mu)$, we have*

$$\Pr\left(\left(x, K^{|P^*(x,\rho)\rangle}(x)\right) \in R\right) \geq p\left(\varepsilon - \kappa(\mu), \frac{1}{\mu}\right).$$

Remark 2.22. *In the fully classical case of 2.20, the extractor could repeat the procedure in sequence polynomially many times in order to increase the probability of a successful extraction (which, in Definitions 2.20 and 2.21, is allowed to be inverse-polynomially small in the security parameter). This is not known to be possible for a general quantum P^*, since the final measurement to extract the witness could possibly disturb the internal state of P^*, making it impossible to simulate the side information that P^* had originally in the subsequent simulations.*

We finally move on to the full quantum setting, where we want a *Proof of Quantum Knowledge* (PoQK). Intuitively, at the end of the protocol, we would like the verifier to be 'convinced' that the prover 'has' a *quantum witness* for the input x. The main difference from Quantum Proofs of (classical) Knowledge is that in the case of QMA relations, as defined in section 2.4.2, the notion of a witness is not as unambiguous as in the case of NP relations. We introduce a parameter q which quantifies the probability that the witness returned by the extractor makes the verifying circuit accept. We refer to this parameter as the "quality" of the PoQK. We also allow the extractor K to return a special symbol "$\perp$" in a designated portion of the output register, and we require that either the extractor returns "$\perp$" or it returns a witness of a certain quality. Formally, we assume that the output of the extractor is measured according to $\{|\perp\rangle\langle\perp|, I - |\perp\rangle\langle\perp|\}$. We ask that the outcome of this measurement be the latter with at least inverse-polynomial probability, and that, conditioned on the latter outcome, the post-measurement state be a witness (of a certain quality).

Definition 2.23 (Proof of Quantum Knowledge)**.** *Let (Q, α, β) be a* QMA *relation. A proof system (P, V) is a Proof of Quantum Knowledge for (Q, α, β) with knowledge error κ and quality $q > \beta$, if there exists a polynomial $p > 0$ and a quantum polynomial-time machine K such that for any quantum interactive machine P^*, any $\mu \in \mathbb{N}$, any polynomial $l > 0$, any instance $x \in \{0,1\}^n$ for $n = \text{poly}(\mu)$ and any state ρ: if the execution $(P^*(x,\rho), V(x))$ returns 1 with probability $\varepsilon > \kappa(\mu)$, we have*

$$\Pr\left(K^{|P^*(x,\rho)\rangle}(x) \neq \text{"}\perp\text{"} \text{ and } \left(x, \frac{(I - |\perp\rangle\langle\perp|)K^{|P^*(x,\rho)\rangle}(x)}{\text{Tr}[(I - |\perp\rangle\langle\perp|)K^{|P^*(x,\rho)\rangle}(x)]}\right) \in R_{Q,q(|x|,\varepsilon)}\right) \geq p\left(\varepsilon - \kappa(\mu), \frac{1}{\mu}\right).$$

We also define *arguments* of quantum knowledge (with a setup). The main difference is that the proof system is replaced by an argument system with setup. Moreover, the extractor is allowed to create the setup as they wish (they can "impersonate" the setup procedure S).

Definition 2.24 (Quantum Argument of (Classical) Knowledge). *Let $R \subseteq \mathcal{X} \times \mathcal{Y}$ be a relation. An argument system with setup $\Pi = (S, P, V)$ for R is a Quantum Argument of Knowledge with setup for R with knowledge error κ if there exists a polynomial $p > 0$ and a quantum polynomial-time machine K such that for any quantum polynomial-time interactive machine P^*, any $\mu \in \mathbb{N}$, any polynomial $l > 0$, any instance $x \in \{0,1\}^n$ for $n = \mathrm{poly}(\mu)$ and any state ρ: if the execution $(S, P^*(x, \rho), V(x))$ returns 1 with probability $\varepsilon > \kappa(\mu)$, we have*

$$\Pr\left(\left(x, K^{|P^*(x,\rho)\rangle}(x)\right) \in R\right) \geq p\left(\varepsilon - \kappa(\mu), \frac{1}{\mu}\right) .$$

Definition 2.25 (Argument of Quantum Knowledge). *Let (Q, α, β) be a* QMA *relation. An argument system with setup $\Pi = (S, P, V)$ is an Argument of Quantum Knowledge with setup for (Q, α, β) with knowledge error κ and quality $q > \beta$ if there exists a polynomial $p > 0$ and a quantum polynomial-time interactive machine K such that for any quantum polynomial-time interactive machine P^*, any $\mu \in \mathbb{N}$, any polynomial $l > 0$, any instance $x \in \{0,1\}^n$ for $n = \mathrm{poly}(\mu)$ and any state ρ: if the execution $(S, P^*(x, \rho), V(x))$ returns 1 with probability $\varepsilon > \kappa(\mu)$, we have*

$$\Pr\left(K^{|P^*(x,\rho)\rangle}(x) \neq \text{“}\perp\text{”} \text{ and } \left(x, \frac{(I - |\perp\rangle\langle\perp|)K^{|P^*(x,\rho)\rangle}(x)}{\mathrm{Tr}[(I - |\perp\rangle\langle\perp|)K^{|P^*(x,\rho)\rangle}(x)]}\right) \in R_{Q,q(|x|,\varepsilon)}\right) \geq p\left(\varepsilon - \kappa(\mu), \frac{1}{\mu}\right) .$$

As for the several possible specializations to the definition of Argument of Quantum Knowledge with setup based on the properties of the underlying argument system (NIZK, CRS setup, preprocessing etc.), we naturally apply the terminology introduced in sections 2.4.2, 2.5.1 and 2.5.2.

2.6.1 Reducing the knowledge error sequentially

One of the most natural properties of Proofs of Knowledge that one investigates in the classical setting is reducing the knowledge error by sequential repetition. Classically, it is well-known that the knowledge error drops exponentially fast in the number of sequential repetitions [BG92]. Just like in the classical case, sequential repetition of a proof of quantum knowledge reduces the knowledge error exponentially fast. This is an immediate consequence of the proof of a lemma from Unruh [Unr12] for the case of quantum Proofs of (classical) Knowledge.

Lemma 2.26. *Let $n = n(\mu)$ be a polynomially bounded and efficiently computable function. Let (P, V) be a Proof of Quantum Knowledge for a* QMA *relation (Q, α, β) with knowledge error κ. Let (P', V') be the proof system consisting of n sequential executions of (P, V) (where V' accepts iff all executions accept). Then (P', V') is a Proof of Quantum Knowledge for (Q, α, β) with knowledge error κ^n.*

Proof. Unruh's argument applies to interactive machines with quantum auxiliary input, which is our setting, and does not make use at all of the fact that the extractor's output is a classical string (the argument proceeds entirely at the level of the success probability of the extractor for the sequentially composed proof and for the atomic proof). □

The only difference from [Unr12] in the Quantum Knowledge case is that if the original PoQK had the additional property of existence of an efficient prover which is accepted with probability close to 1 when given a witness as auxiliary input, the corresponding efficient prover for the

n-sequentially repeated protocol requires n copies of a witness. Formally, this does not affect the completeness property of the proof system since this is formulated with respect to computationally unbounded provers, who can generate as many witnesses as they wish on their own.

An analogous lemma holds for arguments of quantum knowledge. However, since in this case the prover is computationally bounded, the completeness parameter will, in general, drop below soundness, unless we give the prover multiple copies of the witness, instead of a single one. In this case we say that the argument has completeness with respect to the alternative definition, Definition 2.11.

3 The protocol

3.1 Notation and predicates

For a circuit Q_n, we denote by $H(Q_n)$ the local Clifford Hamiltonian obtained by performing the circuit-to-Clifford-Hamiltonian reduction from [BJSW16, Section 2]. In the rest of this section, Q_n will always be taken from a family $Q = \{Q_n\}_{n\in\mathbb{N}}$, where Q specifies a QMA relation (Q, α, β), and we will let the r-th term of the Clifford Hamiltonian $H(Q_n)$ be $C_r^* \left|0^k\right\rangle \left\langle 0^k\right| C_r$. So,

$$H(Q_n) = \sum_{r=1}^{m} C_r^* \left|0^k\right\rangle \left\langle 0^k\right| C_r \,, \tag{2}$$

where each C_r is a k-local Clifford unitary. (Following [BJSW16], we use the short-hand $\left|0^k\right\rangle \left\langle 0^k\right|$ to denote a projector which is $\left|0\right\rangle \left\langle 0\right|$ on at most k qubits and identity everywhere else. As shown in [BJSW16], we can take $k = 5$ without loss of generality.)

We denote by $\mathcal{H}_{\mathsf{clock}} \otimes \mathcal{H}_{\mathsf{instance}} \otimes \mathcal{H}_{\mathsf{witness}}$ the Hilbert space that $H(Q_n)$ acts on. For notational convenience, we assume in the rest of this section that $\mathcal{H}_{\mathsf{instance}}$ is n qubits, that is, $\mathcal{H}_{\mathsf{instance}} = \mathbb{C}^{2^n}$.

For clarity and notational convenience, we define predicates R_r and Q below (along the lines of Definitions 2.1 and 2.2) which we will refer to in our description of our protocol.

Remark 3.1. *Predicates Q and R_r are defined with respect to a fixed problem instance x and a fixed Clifford Hamiltonian H, where*

$$H = \sum_{r=1}^{m} C_r^* \left|0^k\right\rangle \left\langle 0^k\right| C_r$$

for some m that is polynomial in n.

Definition 3.2 (Definition of R_r)**.** *As in section 2.2, we write $\mathcal{D}_N$ to represent the set of all valid (classical) N-bit codewords of a particular error-correcting code. We will generally refer to this error-correcting code as 'the concatenated Steane code'. (This code is the same concatenated Steane code which is outlined in [BJSW16, Appendix A.6].) We may write $\mathcal{D}_N = \mathcal{D}_N^0 \cup \mathcal{D}_N^1$, where $\mathcal{D}_N^0$ is the set of all codewords that encode 0, and $\mathcal{D}_N^1$ is defined analogously.*

We assume that r takes values in $[m+1]$, where m is the number of terms in the Clifford Hamiltonian H. Our R_r is defined differently when $r \in [m]$ and when $r = m+1$.

If $r \in [m]$: we define $R_r(t, \pi, u) = \tilde{R}_r(t, \pi, u)$, where $\tilde{R}_r$ is the predicate defined in Definition 2.1.

If $r = m+1$, then we set $R_r = R_{m+1}$, where R_{m+1} is defined below (Definition 3.3).

Definition 3.3 (Definition of R_{m+1})**.** *Let $u = u_{\mathsf{clock}_1}, u_{\mathsf{instance}_1}, \ldots, u_{\mathsf{instance}_\ell}$ be a string in $\{0,1\}^{2N(\ell+1)}$.*

Remark 3.4. *Each u_{label}, for $\mathsf{label} \in \{\mathsf{clock}_1, \mathsf{instance}_1, \ldots, \mathsf{instance}_\ell\}$, is a $2N$-bit string, and intuitively represents the result of measuring the* logical *qubit with an index specified by* label*. (For notational convenience in the exposition below, we replace the iterator* label *by the iterator* i*.) For example, u_{clock_1} is the string that results from measuring the first* logical *qubit of the clock register. The logical clock register consists of many logical qubits, and each logical qubit is encoded in $2N$ physical qubits as a result of applying the authentication code described in Figure 1.*

For $\pi \in S_{2N}$, and for each $i \in \{\mathsf{clock}_1, \mathsf{instance}_1, \ldots, \mathsf{instance}_\ell\}$, define strings p_i, q_i in $\{0,1\}^N$ such that $\pi(p_i \| q_i) = u_i$ (alternatively: $\pi^{-1}(u_i) = p_i \| q_i$). The predicate $R_{m+1}(t, \pi, u)$ takes the value 1 if and only if the following two conditions (1. and 2.) are met:

1. Either

   $p_{\mathsf{clock}_1} \in \mathcal{D}_N^1$ *(this corresponds to the first qubit of the clock register, expressed in unary, being in state* 1*, i.e. the clock register is* not *at time* 0*),*

   or

   For every $i \in \{\mathsf{instance}_1, \ldots, \mathsf{instance}_\ell\}$, $p_i \in \mathcal{D}_N^{x_i}$.

2. $\langle q_{\mathsf{clock}_1} q_{\mathsf{instance}_1} \cdots q_{\mathsf{instance}_\ell} | t_{\mathsf{clock}_1} t_{\mathsf{instance}_1} \cdots t_{\mathsf{instance}_\ell} \rangle \neq 0.$

We now define our predicate Q in terms of the R_r defined in Definition 3.2.

Definition 3.5 (Definition of Q)**.** *Let $d = (a', b') = (a'_1, \ldots, a'_{2N(\ell+1)}, b'_1, \ldots, b'_{2N(\ell+1)})$ be a string in $\{0,1\}^{4N(\ell+1)}$. Define*

$$\begin{aligned}\mathbb{P}_{m+1} = &|0\rangle\langle 0|_{\mathsf{clock}_1} \otimes \left(I - |x\rangle\langle x|\right)_{\mathsf{instance}} \otimes I_{\mathsf{witness}} \\ &+ (I - |0\rangle\langle 0|)_{\mathsf{clock}_1} \otimes I_{\mathsf{instance}} \otimes I_{\mathsf{witness}}\end{aligned}$$

where $|x\rangle\langle x|$ is a shorthand for the projector onto the standard-basis bitstring $\langle x \rangle$, and

$$C_{m+1} = I_{\mathsf{clock}} \otimes I_{\mathsf{instance}} \otimes I_{\mathsf{witness}}.$$

For $r \in [m+1]$, define

$$\mathbb{P}_r = \begin{cases} C_r^* |0^k\rangle\langle 0^k| C_r & r \in [m] \\ \mathbb{P}_{m+1} & r = m+1 \end{cases}$$

Let $i_1, .., i_k$ be the indices of the qubits on which $\mathbb{P}_r$ acts non-trivially, and let $e_{i_1}, \ldots, e_{i_k}$ be the unique strings such that

$$C_r^{\otimes 2N}(X^{(a\oplus a')_{i_1}} Z^{(b\oplus b')_{i_1}} \otimes \cdots \otimes X^{(a\oplus a')_{i_k}} Z^{(b\oplus b')_{i_k}}) = \alpha(X^{e_{i_1}} Z^{f_{i_1}} \otimes \cdots \otimes X^{e_{i_k}} Z^{f_{i_k}}) C_r^{\otimes 2N} \tag{3}$$

for some $\alpha \in \{1, i, -1, -i\}$ and some $f_{i_1}, \ldots, f_{i_k} \in \{0,1\}^{2N}$. (It is possible to efficiently compute $e = e_{i_1}, \ldots, e_{i_k}$ and $f = f_{i_1}, \ldots, f_{i_k}$ given a, b and C_r.)

Predicate Q is defined as follows:

$$Q(t, \pi, a, b, r, z, d) = R_r(t, \pi, z \oplus e_{i_1} \cdots e_{i_k}).$$

3.2 The protocol

Parties. The argument system involves

1. A (QPT) *verifier* V,
2. A (QPT) *prover* P, and
3. A (classical PPT) *setup machine* S.

The verifier sends a single quantum message to the prover in the preprocessing phase of the protocol, and the prover sends the verifier a single classical message in the instance-dependent phase of the protocol. S sends an identical classical message to both the prover and the verifier during the preprocessing phase.

Inputs. (Unless otherwise stated, all inputs are common to all three parties.)

1. Preprocessing stage:
   (a) An instance size parameter n and a security parameter μ.
   (b) A QMA relation (Q, α, β).
   (c) The Clifford Hamiltonian $H(Q_n)$ (see equation (2)).
   (d) Other parameters:
      i. $c(n)$, an upper bound on the number of qubits in a witness state;
      ii. $p(n)$, an upper bound on the number of qubits in a history state corresponding to an execution of Q_n on a witness state of length $c(n)$ and an instance of size n;
      iii. $m = \text{poly}(n)$, the number of terms in the Clifford Hamiltonian (equation (2));
      iv. $N = \text{poly}(n)$, the number of physical qubits per logical qubit in the Steane code introduced in section 2.2.
   (e) A perfectly binding, quantum computationally concealing (classical) commitment scheme $\mathsf{Com} = (\mathsf{Com.gen}, \mathsf{Com.commit}, \mathsf{Com.reveal}, \mathsf{Com.verify}, \mathsf{Com.recover})$, of the form described in section 2.3.2.
   (f) A NIZK argument system with setup for NP, such as the one described in section 2.5.3. We denote this argument system by a triple $(\mathsf{NIZK}.S, \mathsf{NIZK}.P, \mathsf{NIZK}.V)$.

(g) A levelled fully homomorphic encryption scheme with circuit privacy, $\mathsf{FHE} = (\mathsf{FHE.Gen}, \mathsf{FHE.Enc}, \mathsf{FHE.Dec}, \mathsf{FHE.Eval}, \mathsf{FHE.Refresh})$.

2. Instance-dependent stage:
   (a) The instance x.
   (b) Auxiliary (private) input to the prover: a witness ρ for the claim that $x \in R_{Q,\alpha}$.

Protocol.

Refer to Figure 6 for a diagram charting the essential structure of the protocol below.

Preprocessing step:

(S) S samples $\gamma_P, \gamma_V \leftarrow \mathsf{NIZK}.S(1^{q(n)}, 1^{\mu})$, for a sufficiently large polynomial $q > 0$, in addition to $(pk_P, sk_P), (pk_V, sk_V) \leftarrow \mathsf{Com.gen}(1^{\mu})$ (the two pairs are sampled independently). S then outputs (γ, pk_P, pk_V).

(i) V samples $r' \xleftarrow{\$} [m+n]$. Sets $r = r'$ if $r \leq m$, and $r = m+1$ otherwise. Computes $\sigma \leftarrow \mathsf{Com.commit}(pk_V; r)$. Let s_V be the randomness used.

(ii) V creates $2Np(n)$ EPR pairs ($2N$ for each qubit of the history state). The second qubit of each EPR pair will be sent to P in step (iv). We refer to the remaining qubits (i.e. the first qubit in each EPR pair) as "V's qubits".

(iii) V interprets her $2Np(n)$ qubits as the encoding (according to the authentication code of Fig. 1) of a history state for $H(Q_n)$.

- If $r \in [m]$: V applies C_r transversally to the subset of her $2Np(n)$ qubits which encode the k logical qubits on which C_r acts. Measures those qubits in the computational basis, obtaining an outcome string z.
- If $r = m+1$: V measures the subset of the $2Np(n)$ qubits corresponding to the first qubit of $\mathcal{H}_{\mathsf{clock}}$ and all the qubits of $\mathcal{H}_{\mathsf{instance}}$ in the computational basis, obtaining an outcome string z.

(iv) V samples $(pk_E, sk_E) \leftarrow \mathsf{FHE.Gen}(1^n)$. V sends to P:

- $\alpha \leftarrow \mathsf{FHE.Enc}(pk_E, (r, s_V, z))$.
- pk_E and σ.
- An NIZK argument that there exists $e = (r', s', z')$ and t such that $\mathsf{FHE.Enc}(pk_E, e; t) = \alpha$ (t are the random coins used by $\mathsf{FHE.Enc}$). V uses γ_V in order to compute this argument.

Figure 2: The preprocessing step of our NIZK argument for QMA

Instance-dependent step:

- **Prover's message:**
  1. P executes NIZK.V to check the validity of the NIZK argument received in step (iv) of the preprocessing step. If it is valid, P proceeds to the next step. Otherwise P aborts.
  2. P computes the history state corresponding to an evaluation of the circuit Q_n on the input $|x\rangle \otimes |\psi\rangle$. This is the state $|\Psi\rangle = \sum_{t=0}^{T} |t\rangle_{\mathsf{clock}} \otimes \Pi_{j=1}^{t} U_j \big(|x\rangle \otimes |\psi\rangle \otimes |0\rangle^{\otimes n} \big)$ for some unitary U_j, which can be computed efficiently. P computes $|\tilde{\Psi}\rangle \leftarrow \mathsf{Auth.Enc}(|\Psi\rangle)$ according to the authentication scheme of Fig. 1. Let the sampled authentication keys be:
     (a) $a = a_1, .., a_{p(n)}$, $b = b_1, .., b_{p(n)}$ for $a_1, .., a_{p(n)}, b_1, .., b_{p(n)} \in \{0,1\}^{2N}$,
     (b) $\pi \in S_{2N}$,
     (c) $t = t_1, .., t_{p(n)}$ where $t_1, .., t_{p(n)} \in \{0, +, +_y\}^N$.
     P samples commitment randomness s_P, and computes $\sigma_{\mathsf{keys}} \leftarrow \mathsf{Com.commit}(pk, (t, \pi, a, b), s_P)$.
  3. P teleports the state ρ to V using his halves of the $2Np(n)$ shared EPR pairs received in step (iv) of the preprocessing step. Let $d = (x_1, \ldots, x_{2Np(n)}, y_1, \ldots, y_{2Np(n)}) \in \{0,1\}^{4Np(n)}$ be the Bell basis measurement outcomes obtained during the teleportation.
  4. P computes $\beta \leftarrow \mathsf{FHE.Enc}\big(pk_E, (d, \sigma, \sigma_{\mathsf{keys}}, (t, \pi, a, b), s_P)\big)$, where σ is the commitment received in step (iv) of the preprocessing step. P homomorphically evaluates the following circuit C using β and the ciphertext α that it received from the verifier. (Recall that α is an encryption of (r, s_V, z).)

     C takes as input $d, \sigma, r, s_V, z, \sigma_{\mathsf{keys}}, t, \pi, a, b, s_P$. It checks that (r, s_V) is a valid opening for σ, and that $Q(t, \pi, a, b, r, z, d) = 1$, where Q is defined in Definition 3.5. If its checks pass, using γ_P it computes an NIZK argument for the existence of an opening to σ_{keys} such that the opened value (t, π, a, b) satisfies $Q(t, \pi, a, b, r, z, d) = 1$. If its checks do not pass, it outputs "$\perp$".
  5. Let $\tilde{\pi}$ be the encrypted proof that P obtains in step 4. P computes $\tilde{\pi}' \leftarrow \mathsf{FHE.Refresh}(\tilde{\pi})$. Sends d, σ_{keys} and $\tilde{\pi}'$ to V.
- **Verifier's check:** V decrypts $\tilde{\pi}'$, and executes NIZK.V to check the decrypted proof. It checks that the d received from P is the same d that appears in the statement being proven.

Figure 3: The instance-dependent step of our NIZK argument for QMA

Theorem 3.6. *Every language in QMA has a NIZK argument argument system with CRS setup and preprocessing, with completeness* $1 - \mathrm{negl}$ *and soundness* negl. *Moreover, the preprocessing phase consists of a single quantum message from the verifier to the prover.*

We refer to the combination of the protocols of Figures 2 and 3 as "the protocol".

To show Theorem 3.6 we start with an arbitrary language $L \in$ QMA. Using standard amplification

techniques, for any polynomial t there is a family of polynomial-size verification circuits Q such that L is the language associated with the QMA relation $(Q, 1, 2^{-t})$ as in Definition 2.8. We show that the protocol associated to this relation is an NIZK argument with setup for $(Q, 1 - 2^{-t}, 2^{-t})$. Completeness is easy to verify, as for any $(x, \rho) \in R_{Q,1-2^{-t}}$ the prover described in Figure 3 is accepted with probability negligibly close to 1, given access to ρ. In Section 4 we prove soundness inverse polynomially close to 1, and in Section 4.4 we show how soundness can be amplified in parallel to any 2^{-p} for polynomial p (provided t is taken large enough compared to p). Finally, in Section 5 we prove the zero-knowledge property.

4 Soundness

4.1 Overview

The structure of the proof is as follows. We show through a sequence of hybrids that it is possible to transform an execution of our protocol on some instance x, into an execution of the protocol from [BJSW16] on a specific local Clifford Hamiltonian derived from x. We show that such transformation can at most negligibly decrease the optimal acceptance probability of the prover. Thus, soundness of our protocol reduces to soundness of the protocol from [BJSW16]. The main steps in our sequence of hybrids are the following:

- Remove the zero knowledge proof that V sends to the prover in step (iv) of the preprocessing step;
- Remove the encryption of V's choice of r, randomness s_V and measurement outcomes z sent in step (iv) of the preprocessing step.
- Replace the step where P teleports the encoded witness to V through shared EPR pairs (step 3 in Fig. 3) with one where P directly sends the qubits of the encoded witness to V.
- Remove the CRS, and replace the NIZK argument sent by the prover in step 5 of Fig. 3 with a ZK proof.

In more detail, fix a QMA relation (Q, α, β) and an instance x of size n. Consider the following sequence of hybrid experiments. Each experiment describes a modified verifier, and thus a modified protocol. We will argue that the optimal probability of a prover being accepted in any of the protocols can only increase (or at most negligibly decrease) across hybrids.

H_0: This is the real experiment in which V behaves as described in Figure 2 and Figure 3.

H_1: Same as H_0, except V_1 does not send the NIZK proof from step (iv) of the preprocessing step.

H_2: Same as H_1, except V_2 does not send to P an encryption of r, s_V and her outcomes z. Instead, she sends an encryption of the zero string. The instance-dependent step is now interactive, and proceeds as follows:

(a) V_2 expects d and σ from P.

(b) Let z be the outcome of V_2's measurements from step (iii) of the preprocessing step. V_2 Sends z, r and s_V to P (in the clear).

(c) V_2 expects an NIZK argument (in the clear) of existence of an opening of σ such that the opened keys t, π, a, b satisfy $Q(t, \pi, a, b, r, z, d) = 1$.

H_3: Same as H_2 except V_3 does not send the encryption of the zero string.

H_4: Same as H_3, except V_4's measurements in step (iii) of the preprocessing step are postponed. V_4 proceeds directly to the instance-dependent step as in H_3. Only after receiving d, σ, V_4 measures according to step (iii), and sends the outcome z of these measurements to P. The rest is the same.

H_5: We modify H_4 as follows: V_5 does not send any EPR pairs to P. In step 1 of the instance-dependent step, V_5 expects P to send a $2Np(n)$-qubit state directly, together with a string d and a commitment σ. V_5 obtains z by measuring the received state. The rest is the same.

H_6: Same as H_6, except V_6 does not expect any string d. The relation for which V_6 expects an NIZK argument is modified so that d (which was previously part of the instance) is now fixed to 0 (or equivalently $\widetilde{Q}$ is replaced with Q, from Definition 2.2).

H_7 : Same as H_6, except V_7 does the following: in part (i) of the preprocessing step, V_7 samples $r' \xleftarrow{\$} \in [m+n]$. V_7 computes $\sigma \leftarrow \mathsf{Com.commit}(pk_V; r')$. In part (iii) of Fig. 2,

- If $r \in \{1, \ldots, m\}$: same as before.
- If $r = m + i$, $i \in [n]$: V_7 measures the subset of the $2Np(n)$ qubits corresponding to the first qubit of $\mathcal{H}_{\mathsf{clock}}$ and to the i-th qubit of $\mathcal{H}_{\mathsf{circuit\text{-}desc}}$ in the computational basis, obtaining an outcome string z.

In the final step, V_7 expects from P an NIZK argument for the existence of an opening to the committed keys such that the opened keys t, π, a, b satisfy $Q'(t, \pi, a, b, r, z) = 1$, where Q' is defined as follows from R'_r.

Definition 4.1 (Definition of R'_r, $r \in \{m+1, \ldots, m+n\}$)**.** *Let $u_{\mathsf{clock}_1}, u_{\mathsf{instance}_1}, .., u_{\mathsf{instance}_n} \in \{0,1\}^{2N}$ (these represent the measurement results that the verifier sends to the prover in step (iii) of Fig 2 when $r = m + 1$). For $\pi \in S_{2N}$, for each $i \in \{\mathsf{clock}_1, \mathsf{instance}_1, \ldots, \mathsf{instance}_n\}$, define strings p_i, q_i in $\{1,1\}^N$ such that $\pi(p_i \| q_i) = u_i$ (alternatively: $\pi^{-1}(u_i) = p_i \| q_i$). Let $u = u_{\mathsf{clock}_1}, u_{\mathsf{instance}_1}, .., u_{\mathsf{instance}_n}$. The predicate $R'_{m+i}(t, \pi, u)$ takes the value 1 if and only if the following two conditions are met:*

1. *$p_{\mathsf{clock}_1} \in \mathcal{D}_N^1$ (this corresponds to the first qubit of the clock register, expressed in unary, being in state 1, i.e. the clock register is not at time 0), or $p_i \in \mathcal{D}_N^{x_i}$.*
2. *$\langle q_{\mathsf{clock}_1}\, q_{\mathsf{instance}_1} \cdots q_{\mathsf{instance}_n} | t_{\mathsf{clock}_1}\, t_{\mathsf{instance}_1} \cdots t_{\mathsf{instance}_n} \rangle \neq 0$*

Definition 4.2 (Definition of Q′)**.** • *If $r \in \{1, \ldots, m\}$,*

$$Q'(t, \pi, a, b, r, u) = R'_r(t, \pi, u \oplus e_{i_1} \cdots e_{i_k}),$$

where $i_1, .., i_k$ are the indices of the qubits on which the C_r acts non-trivially and $e_{i_1}, \ldots, e_{i_k}$ are the

unique strings such that

$$C_r^{\otimes 2N}(X^{a_{i_1}}Z^{b_{i_1}} \otimes \cdots \otimes X^{a_{i_k}}Z^{b_{i_k}}) = \alpha(X^{e_{i_1}}Z^{f_{i_1}} \otimes \cdots \otimes X^{e_{i_k}}Z^{f_{i_k}})C_r^{\otimes 2N}$$

for some $\alpha \in \{1, i, -1, -i\}$ *and some* $f_{i_1}, \ldots, f_{i_k} \in \{0,1\}^{2N}$.

- *If* $r = m + i$, *for* $i \in [n]$:

$$Q'(t, \pi, a, b, r, u) = R'_r(t, \pi, u \oplus a_{clock_1} a_{instance_1} \cdots a_{instance_n}).$$

H_8: Same as H_7, except V_8 does not send a commitment to the randomness r. We also drop the associated public key pk_V from the CRS. The first message in the protocol is from P who is expected to send a state and a commitment. After that, V_8 sends z, r to P. The rest is the same.

H_9: Same as H_8, except we remove γ_P from the CRS, and the NIZK proof from P to V_9 in the last round is replaced by a (interactive) ZK proof.

H_{10}: Same as H_9, except that after the first message of the prover, V_{10} and P engage in a coin flipping protocol. This determines the randomness r. In the next round V_{10} sends her measurement outcome z to P, and the rest is the same.

H_{11}: Same as H_{10}, except we remove pk_P from the CRS, and we replace the public-key commitment scheme Com with the commitment scheme commit used by the prover in the [BJSW16] protocol.

In Section 4.2 we show that, up to negligible quantity, the maximum success probability of a prover in the protocol described in H_{11} can only be higher than the maximum success probability of a prover in the protocol described in H_0. In Section 4.3 we show that this implies soundness of the protocol. Finally, in Section 4.4 we show that soundness can be amplified in parallel.

4.2 Success probability in hybrids

Lemma 4.3. *The optimal probability of a prover being accepted in* H_0 *is at most negligibly higher than in* H_{11}.

Proof. We argue that the optimal probability of the prover can decrease at most negligibly across hybrids.

$H_0 \approx H_1$. We show that given any prover P for H_0, we can find a prover P' for H_1 which is accepted with negligibly close probability. Since the converse is clear, the maximum success probability of a prover in H_0 and H_1 is negligibly close.

H_0 and H_1 only differ in step (iv) of the preprocessing protocol, where, in the latter, V does not provide any NIZK proof. The prover P' acts as follows: P' forwards to P the message it receives from V in step (iv). P' then runs the simulator *Sim* for the NIZK proof of step (iv) (which is guaranteed to exist by the ZK property of the proof; this step does not need adaptive ZK since the verifier is honest and the statement to be proven is sampled independently of the CRS) on the

(quantum) registers of P. Then, P' continues the rest of the protocol by running P and V. Suppose for a contradiction that the probabilities of P and P' being accepted differ more than negligibly. Then there must exist a polynomial q such that, for infinitely many n, there exist a ciphertext $\alpha^{(n)}$ and a joint state $\rho^{(n)}$ of V and P (before the point where H_1 and H_2 differ), such that

$$\left| \Pr\left(V \text{ accepts } P \text{ in } H_0 | \alpha^{(n)}, \rho^{(n)}\right) - \Pr\left(V \text{ accepts } P' \text{ in } H_1 | \alpha^{(n)}, \rho^{(n)}\right) \right| > \frac{1}{q(n)} . \tag{4}$$

Suppose that it is the first probability that is inverse-polynomially higher infinitely often (the other case is analogous). We construct a distinguisher $\mathcal{D}$ that breaks the zero-knowledge property of the argument.

Let $\mathcal{D}$ be the following (non-uniform) distinguisher: $\mathcal{D}$ takes as input a string x (the instance) and a quantum register, which it interprets as being a register of V and a register of P. It runs V and P using these registers: $\mathcal{D}$ outputs 0 (i.e. guesses that it received a sample from a true transcript) if V accepts, and outputs 1 (i.e. guesses that it received a sample from the simulator) if V rejects (note that $\mathcal{D}$ is running V so it has the secret key that V uses to decrypt the ciphertext received by the prover in the very last step).

It is straightforward to see that on the sequence of instance, auxliary input pairs $\alpha^{(n)}, \rho^{(n)}$, $\mathcal{D}$ succeeds at distinguishing the (quantum) transcript of a true NIZK proof from that of *Sim* with probability at least $\frac{1}{q(n)}$. In the former case, the output of $\mathcal{D}$ is distributed exactly as an experiment H_0 between V and P conditioned on $\alpha^{(n)}$ being the ciphertext sent by V in step (iv), and $\rho^{(n)}$ being the joint state of V and P right before the ZK proof. In the latter case, it is distributed exactly as an experiment H_1 between V and P' conditioned on α being the ciphertext sent by V and $\rho^{(n)}$ being the joint state of V and (the invoked) P right before P' runs *Sim* to simulate a transcript of the ZK proof.

$H_1 \approx H_2$. We start with the following claim.

Claim 4.4. *For any prover P^* that plays H_1 or H_2 up to step (a) of H_2, the probability that P^* outputs a pair (d, σ) such that, letting t, π, a, b be the value committed in σ and z, r be V's message in step (b), we have $\widetilde{Q}(t, \pi, a, b, r, z, d) = 1$ is negligibly close for H_1 and H_2.*

Proof. Suppose for a contradiction that there is a P^* such that the probability that P^* outputs a good d in H_1 and in H_2 differ non-negligibly. Then, there exists a polynomial $q > 0$ such that the difference between the probabilities is at least $1/q(\mu)$ for infinitely many values of the security parameter μ. Suppose it is the first probability that is at least at least $1/q(\mu)$ larger infinitely often. (the reverse case is similar).

We construct an adversary $\mathcal{A}$ that breaks CPA security of FHE. $\mathcal{A}$ proceeds as follows:

- $\mathcal{A}$ receives a public key pk_E for the FHE scheme from the challenger.
- $\mathcal{A}$ runs $\gamma_P, \gamma_V \leftarrow \mathsf{NIZK\text{-}CRS.Setup}$ and $(\tilde{pk}_P, \tilde{sk}_P), (\tilde{pk}_V, \tilde{sk}_V) \leftarrow \mathsf{Com.gen}(1^\mu)$. $\mathcal{A}$ then runs the honest verifier V and P^* on common input the CRS $(\gamma_V, \gamma_P, \tilde{pk}_P, \tilde{pk}_V)$, except that if forwards pk_E to V in step (iv) (it does not matter that $\mathcal{A}$ cannot give the secret key to V, since V only uses this in the very final "Verifier's check" of Fig. 3, which $\mathcal{A}$ will not need to run).

Let z be the outcomes of V's measurements in step (iii). Let r and s_V be respectively V's committed random choice of Hamiltonian term and the randomness used for committing to it. $\mathcal{A}$ sets $m_0 = r, s_V, z$, and $m_1 = 0$. Sends m_0 and m_1 to the challenger.

- The challenger returns a ciphertext c. $\mathcal{A}$ forwards this to P^* as the first part of the verifier's message in step (iv), and runs P^* obtaining output d, σ where σ is some commitment.
- $\mathcal{A}$ decrypts σ using $\tilde{sk}_P$, obtaining (t, π, a, b). If the output is not of this form or (t, π, a, b) is not consistent with d, r, z, then $\mathcal{A}$ guesses that the ciphertext received was an encryption of m_1; otherwise of m_0.

Denote by b the bit sampled by the challenger. Notice that in the case that $b = 0$, the distribution of the output d that $\mathcal{A}$ obtains from P^* is precisely that of H_1. When $b = 1$, it is precisely that of H_2. Given the hypothesis, this implies that $\mathcal{A}$ wins the CPA-security game with $1/q$ advantage infinitely often, which contradicts CPA-security of FHE. □

Next, we show that the optimal probabilities of a prover being accepted in H_1 and in H_2 are negligibly close. Let P be a prover for H_1. We construct a prover P' for H_2 such that

$$\Pr\left(P' \text{ is accepted}\right) > \Pr\left(P \text{ is accepted}\right) - \text{negl}(|x|)\ .$$

P' runs as follows:

- As part of H_2, P' receives a CRS $(\gamma_V, \gamma_P, pk_P, pk_V)$. It then receives a commitment $\tilde{\sigma}$ and a ciphertext c. P' runs $(\tilde{pk}_P, \tilde{sk}_P) \leftarrow \mathsf{Com.gen}(1^\mu)$. It provides $(\gamma, \tilde{pk}_P, pk_V)$ to P as CRS, and c as the ciphertext. P returns d and σ. P' decrypts σ using $\tilde{sk}_P$. Let t, π, a, b be the committed value (if the committed value is not of this form or if decryption fails, P' aborts). P' computes $\sigma' \leftarrow \mathsf{Com.commit}(pk_P; t, \pi, a, b)$. Let s' be the randomness used. P' returns d, σ' to the verifier of H_2.
- P' receives z, r, s_V. If z, r, d, t, π, a, b are consistent, P' computes an NIZK proof of the existence of an opening to σ' such that the opened keys are consistent with z, r, d.

Since the NIZK argument system employed has negligible soundness (we chose the NIZK argument system from Section 2.5.3), the probability that P is accepted in H_1 conditioned on returning a bad (d, σ) is negligible. By Claim 4.4, it follows that

$$\begin{aligned}\Pr\left(P'' \text{ is accepted}\right) &> \Pr\left(P' \text{ returns a good } (d, \sigma)\right) - \text{negl}(|x|) \\ &> \Pr\left(P' \text{ is accepted}\right) - \text{negl}(|x|)\end{aligned}$$

where the first inequality follows from Claim 4.4, and the second is because the NIZK argument system employed has negligible soundness, and so the probability that P' is accepted in H_1 conditioned on returning a bad (d, σ) is negligible.

We have thus shown that the optimal probability of acceptance in H_2 is at least that of H_1, up to a negligible quantity. An analogous argument shows the converse. Hence, the optimal probabilities of acceptance in H_1 and H_2 are negligibly close.

$H_2 \approx H_3$**.** The encryption of the zero string can be simulated by the prover itself, so clearly the optimal acceptance probabilities in H_2 and H_3 are exactly equal.

$H_3 \approx H_4$. For any prover P^*, since the action of P^* to produce d, σ and the measurements of V from step (iii) of the preprocessing step commute, performing them before receiving d, σ or after does not change the acceptance probability of P^*.

$H_4 \approx H_5$ We show that the optimal probability of a prover being accepted in H_4 and H_5 is the same. Given a prover P for H_4 one can construct a prover P' that is accepted in H_5 with the same probability: P' starts by sharing $2Np(n)$ EPR pairs with P. When P outputs (d, σ), P' forwards (d, σ) as well as his half of the $2Np(n)$ EPR pairs to V. When V returns a message of the form (z, r, s_V), P' forwards this to P. P' returns the final message of P to V. It is clear that the acceptance probability of P' is the same as that of P, as the distribution of the whole transcript is identical to that of experiment H_4. The reverse direction is similar.

$H_5 \approx H_6$. Again, the optimal probabilities in H_5 and H_6 are the same. For any prover P for H_5, we construct P' for H_6 which is accepted with the same probability. P' runs P. The only difference is that when P outputs $d = (a, b)$ and a state ρ to send to V, P' sends $X^a Z^b \rho (X^a Z^b)^\dagger$ to V.

$H_6 \approx H_7$. We show that the optimal acceptance probability can only increase. Let P be a prover in H_6. We construct a prover P' for H_7 as follows. P' samples a CRS (γ, pk_P, pk_V) such that it knows the secret key sk_P corresponding to pk_P. Upon receiving a commmitment (to some randomness) from the verifier, it runs P obtaining $\sigma, \tilde{\rho}$. It decrypts σ using sk_P. Let (t, π, a, b) be the opened value (if it is not of this form, P' aborts). P' sends $\sigma, \tilde{\rho}$ to the verifier. Upon receiving z, r from the verifier, P' does the following:

- If $r \in \{1, \ldots, m\}$: P' obtains an NIZK proof from P and forwards this to the verifier.
- If $r = m + i$, $i \in [l(n)]$: P' computes an NIZK proof for the existence of an opening to σ such that the opened keys (t, π, a, b) satisfy $Q'(t, \pi, a, b, r, z) = 1$ (where Q' is as defined in 4.2).

Notice that in H_7, the verifier only measures the subset of physical qubits corresponding to the i-th qubit of the circuit description, and the check that it performs is strictly less stringent than in H_6. A simple calculation shows that the probability of acceptance of P' in H_7 is at least that of P in H_6.

$H_7 \approx H_8$. We first show that the acceptance probability of a prover P cannot change more than negligibly if the commitment to r in H_7 is replaced by a commitment to a zero string. Suppose for a contradiction that there exists P such that the probability of acceptance in H_7 differs non-negligibly from H_8. This implies that there exists a polynomial $q > 0$ such that the two probabilities differ by at least $1/q(\mu)$ for infinitely many values of the security parameter μ. Suppose the first probability is higher by at least $1/q$ infinitely often (the reverse case is similar). Then, we can construct an adversary $\mathcal{A}$ that breaks the hiding property of the commitment. $\mathcal{A}$ samples a random r and sends $m_0 = r$ and $m_1 = 0$ to the challenger. Upon receiving a commitment and a public key pk_P from the challenger, $\mathcal{A}$ simulates a CRS which includes pk_P and an execution of the verifier from V_7 (or equivalently V_8) except that it forwards to P the commitment received. If the verifier accepts, $\mathcal{A}$ guesses that it received a commitment to r, otherwise that it received a commitment to 0. Notice that when the challenger sends a commitment to r the view of P is exactly as in H_7, and when it sends a commitment to 0 the view of P is exactly as in H_8. Thus $\mathcal{A}$ has $1/q$ distinguishing advantage infinitely often, which gives the desired contradiction.

$H_8 \approx H_9$. We show that the optimal acceptance probability decreases at most negligibly. Let P be a prover for H_8. We construct a prover P' for H_9 as follows. P' gets as input the CRS pk_P. Then, P' samples a CRS $\tilde{pk}_P$ such that it knows the secret key $\tilde{sk}_P$ corresponding to $\tilde{pk}_P$. It forwards $\tilde{pk}$ to P as the CRS, and obtains $\sigma, \tilde{\rho}$. It decrypts σ using $\tilde{sk}_P$. Let (t, π, a, b) be the opened value (if it is not of this form P' aborts). P' computes $\sigma' \leftarrow \mathsf{Com.commit}(pk_P; t, \pi, a, b)$. Let s' be the randomness used. Sends $\tilde{\rho}, \sigma'$ to the verifier. P' receives z, r from the verifier, and checks that $Q(z, r, t, \pi, a, b) = 1$. If so, it engages with the verifier in a ZK proof of existence of an opening to σ' such that the opened keys, together with z, r, satisfy the predicate Q. Since the NIZK proof has negligible adaptive soundness, the probability that P is accepted in H_9 conditioned on z, r, t, π, a, b being inconsistent is negligible. This implies that

$$\Pr\left(P' \text{ is accepted in } H_9\right) > \Pr\left(P \text{ is accepted in } H_8\right) - negl(|x|) \ .$$

$H_9 \approx H_{10}$. From a prover P for H_9, we obtain a prover P' for H_{10} with the same acceptance probability as follows: P' runs P for the first round. Then P' engages in an honest coin flipping protocol with the verifier and forwards the outcome r to P. Finally, P' runs P until the end.

$H_{10} \approx H_{11}$. We show that the optimal acceptance probability can only increase. Let P be a prover for H_{10}. We construct a prover P' for H_{11} as follows. Similar to the earlier argument for $H_8 \approx H_9$, P' samples a CRS $\tilde{pk}_P$ such that it knows the corresponding secret key $\tilde{sk}_P$. It forwards $\tilde{pk}_P$ as CRS to P who returns $\tilde{\rho}, \sigma$. P' decrypts σ using $\tilde{sk}_P$. Let (t, π, a, b) be the opened value. P' computes $\sigma' \leftarrow \mathsf{commit}(t, \pi, a, b)$. Let s' be the randomness used. P' Sends $\sigma, \tilde{\rho}$ to the verifier. P' engages honestly in a coin-flipping protocol with V. Let r be the outcome. Upon receiving z from the verifier, P' checks that $Q(z, r, t, \pi, a, b) = 1$. If so, it engages in a ZK proof of an opening to σ' such that the opened keys, together with z, r satisfy the predicate Q. Since the ZK proof has negligible soundness, the probability that P is accepted in H_{10} conditioned on $Q(z, r, t, \pi, a, b) = 0$ is negligible. Moreover, since the interaction of P with V in the coin-flipping protocol could have been simulated by a uniformly random r together with some local operation on the register of P, the probability that $Q(z, r, t, \pi, a, b) = 1$ is exactly the same for P and P'. This implies that

$$\Pr\left(P' \text{ is accepted in } H_{11}\right) > \Pr\left(P \text{ is accepted in } H_{10}\right) - \mathrm{negl}(|x|) \ .$$

□

4.3 Reduction to [BJSW16]

Consider the following Hamiltonian:

$$H = H(Q_n) + |0\rangle\langle 0|_{\mathsf{clock}} \otimes \sum_i |\overline{x_i}\rangle\langle\overline{x_i}|_{\mathsf{instance}_i} \tag{5}$$

Notice that this is a *local* Clifford Hamiltonian (once the clock is expressed in unary). It is immediate to verify that the protocol described in H_{11} is identical to an execution of the protocol from [BJSW16] on common input H (see Section 2.2). By invoking the soundness of the [BJSW16] protocol, we deduce that protocol H_{11} is sound, and as a consequence of Lemma 4.3, protocol H_0 (the original protocol) is also sound. This concludes the soundness analysis.

4.4 Parallel amplification of soundness

If $Q = \{Q_n\}$ is a family of quantum circuits that each return a single bit as output, for every $k \geq 1$ we define $Q^{(k)} = \{Q_n^{(k)}\}$ where for every n, $Q_n^{(k)}$ executes Q_n k times in parallel and returns the AND of their output bits.

Let $\Pi = (S, P, V)$ be a non-interactive argument with setup. Let $k \geq 1$ be an integer. We define the k-th parallel repetition of Π, $\Pi^{(k)} = (S^{(k)}, P^{(k)}, V^{(k)})$ as follows:

- For every μ and n, $S_{\mu n}^{(k)}$ executes k independent copies of $S_{\mu n}$ and returns their outputs.
- For every μ and n, $V_{\mu n}^{(k)} = (V_{1,\mu n}^{(k)}, V_{2,\mu n}^{(k)})$ and $P_{\mu n}^{(k)} = (P_{1,\mu n}^{(k)}, P_{2,\mu n}^{(k)})$. $V_{1,\mu n}^{(k)}$ and $P_{1,\mu n}^{(k)}$ take as input a classical string, interpreted as k outputs of $S_{\mu n}$. They execute k copies of $V_{1,\mu n}$ and $P_{1,\mu n}$ respectively on the k inputs and they each produce a k-register quantum state as output. $V_{2,\mu n}^{(k)}$ takes as input the output of $V_{1,\mu n}^{(k)}$ and an instance x. $P_{2,\mu n}^{(k)}$ takes as input the output of $P_{1,\mu n}^{(k)}$, x and a k-register quantum state $\rho^{(k)}$. Each of them executes k copies of $V_{2,\mu n}$ and $P_{2,\mu n}$ respectively, where the i-th copy of $P_{2,\mu n}$ is executed with the i-th register of $\rho^{(k)}$ as its quantum input. $V_{\mu n}^{(k)}$ returns 1 if and only if each copy of $V_{\mu n}$ returns 1.

For classical non-interactive arguments (and more generally arguments of up to three rounds) it is well-known that parallel repetition amplifies soundness exponentially [BIN97]. In our setting the argument is made more delicate by the fact that the message sent from verifier to prover in the setup phase is quantum. We do not know if parallel repetition amplifies soundness in a black-box way for (quantum) non-interactive arguments with setup. However, using the specific structure of our protocol we can give a tailored argument.

Lemma 4.5. *Let (Q, α, β) be a* QMA *relation and $\Pi = (S, P, V)$ the argument system for (Q, α, β) described in Section 3. Let c and s be the completeness and soundness of the protocol respectively. Then for any integer $k \geq 1$, $(S^{(k)}, P^{(k)}, V^{(k)})$ is an argument system for $(Q^{(k)}, \alpha, \beta^k)$ with completeness $1 - k(1-c)$ and soundness s', where s' is negligibly close to s^k.*

Proof. Let $k \geq 1$ and for ease of notation write $\Pi' = (S', P', K') = (S^{(k)}, P^{(k)}, V^{(k)})$ and $(Q', \alpha', \beta') = (Q^{(k)}, \alpha, \beta^k)$. We also fix a μ and an integer n and omit the indices.

Completeness. Let $(x, \rho) \in R_{Q',\alpha'}$. Then by definition Q' accepts ρ with probability at least α. By definition of $Q' = Q^{(k)}$, this implies that if ρ_i is the reduced density of ρ on the i-th register, Q accepts ρ_i with probability at least α. By the completeness property of Π, $(S, P(x, \rho_i), V(x))$ returns 1 with probability at least c. By a union bound, $(S', P'(x, \rho), V'(x))$ returns 1 with probability at least $1 - k(1-c)$.

Soundness. For ease of notation we give the proof for $k = 2$, as the general case is similar. Suppose for contradiction that there exists an $x \in N_{Q',\beta'}$ and a quantum polynomial-time P^* such that $(S', P^*(x), V'(x))$ returns 1 with probability non-negligibly larger than $s' = s^2$.

As a first step we show that there exists a $\tilde{P}^*$ that succeeds with probability non-negligibly larger than $s' = s^2$ in an (interactive) hybrid variant H_2' of the protocol, that is identical to Hybrid H_2

considered in the soundness proof (see Section 4.1), except for the following modifications: First, the protocol described in H_2 is executed twice in parallel. Second, at step (a) of H_2, the prover is expected to report, in addition to the commitments σ_i, valid openings (t_i, π_i, a_i, b_i) for $i \in \{1,2\}$. (The verifier checks that the openings are valid.) We let $\tilde{V}$ be the verifier in the hybrid H_2', and denote by $\tilde{\Pi}'$ the resulting interactive protocol.

Claim 4.6. *The maximum success probability of a prover in protocol Π' and in protocol $\tilde{\Pi}'$ are negligibly close.*

Proof. The fact that the maximum success probability of a prover in H_2 is negligibly close to that in Π' follows from the soundness analysis in Section 4.2, with straightforward modifications to handle the parallel repetition.

Clearly, since the prover is asked for more information in H_2' than in H_2 the maximum success probability cannot increase. To show that it can only decrease negligibly, let P^* be a prover that succeeds in H_2 with some probability, and consider the following prover $\tilde{P}$ for H_2'.

$\tilde{P}$ first executes the setup phase with $\tilde{V}$ exactly as P^* would. As a result, it obtains two CRS $(\gamma_1, pk_{P,1}, pk_{V,1})$ and $(\gamma_2, pk_{P,2}, pk_{V,2})$ as well as two $2Np(n)$-qubit states ρ_1 and ρ_2 and classical information $(pk_{E,1}, \sigma_1, \alpha_1)$ and $(pk_{E,2}, \sigma_2, \alpha_2)$ from the preprocessing phase $\tilde{V}_1$ of $\tilde{V}$.

Next, for $i \in \{1,2\}$ $\tilde{P}$ generates new CRS $(\tilde{\gamma}_i, \tilde{pk}_{P,i}, \tilde{pk}_{V,i})$ such that $\tilde{\gamma}_i = \gamma_i$ and $\tilde{pk}_{V,i} = pk_{V,i}$ but $\tilde{pk}_{P,i}$ is sampled jointly with a secret key $\tilde{sk}_{P,i}$ for $\mathsf{Com}_{\mathsf{stat\text{-}bind}}$ using $\mathsf{Com}_{\mathsf{stat\text{-}bind}}.\mathsf{Gen}(1^n)$.

In the instance-dependent interactive phase, after having received $\tilde{V}_2$'s first message (encryptions of 0), $\tilde{P}$ simulates P^*, *except* that it uses the new CRS in place of the older one. Receiving P^*'s messages (d_i, σ_i) for $i \in \{1,2\}$, $\tilde{P}$ uses $\tilde{sk}_{P,i}$ to open σ_i and sends (d_i, σ_i) together with the opening (t_i, π_i, a_i, b_i) to the verifier. □

Let P^* be a prover with optimal success probability in H_2'. Let p_1 be the probability that the first proof returned by P^* is accepted by the first verifier, and p_2 the probability that the second proof is accepted, conditioned on the first one being accepted. Since the maximum success probability in Π is s, it follows from (the single-repetition analogue of) Claim 4.6 that the maximum success probability of any prover in the single-repetition analogue $\tilde{\Pi}$ of $\tilde{\Pi}'$ is at most neglibly higher than s. By a straightforward simulation argument it follows that p_1 is at most negligibly larger than s. Applying Bayes' rule, p_2 is non-negligibly higher than s. In the remainder of the proof we derive a contradiction by showing that there is an (efficient) P' such that P' succeeds in the single-repetition hybrid $\tilde{\Pi}$ with probability negligibly close to p_2.

Definition of P'. Let ρ_2, m_2 be respectively the quantum state and the classical message received from V_1 in the preprocessing phase. P' independently simulates another execution of V_1 and V_2 to obtain ρ_1 and m_1. Instead of executing directly P^* on these inputs, it executes it on the state ρ_1, but the state ρ_2 is replaced by half-EPR pairs.

Let (d_i', σ_i) and (t_i, π_i, a_i, b_i), for $i \in \{1,2\}$, be the outcomes obtained from P^*. Note that at this point P' can already verify if the outcomes associated with the first repetition $i = 1$ will lead to acceptance in the protocol, simply by running the simulated verifier one step forward to obtain

(z_1, r_1, s_1) and checking whether $Q(t_1, \pi_1, a_1, b_1, r_1, z_1, d_1) = 1$. If this is not the case, P' repeats the first part with a new execution of the setup phase V_1 and fresh half-EPR pairs.

Suppose now that P' has obtained (d_i, σ_i) and (t_i, π_i, a_i, b_i), for $i \in \{1, 2\}$ from P^* such that the execution of the first copy of the verifier leads to acceptance.

To complete the protocol, P' now performs a Bell basis measurement between the half-EPR pairs not given to P^* and the state ρ_2, that until now has not been used. Let $d_2' = (x_i', z_i')$ denote the teleportation outcomes obtained. P' then pursues the interaction with V_2, except that it replaces the outcome $d_2 = (x_i, z_i)$ obtained from P^* by $d_2 \oplus d_2' = (x_i \oplus x_i', z_i \oplus z_i')$. All other messages are the same. This completes the description of P'.

Success probability of P'. To conclude we claim that P' succeeds with probability non-negligibly larger than s in $\tilde{\Pi}$. Note that the result of the "teleportation" operation performed by P' is that the prover P^* is executed on the state $(X^{x'}Z^{z'})\rho_2(X^{x'}Z^{z'})^\dagger$ instead of the state ρ_2. Since the state ρ_2 is obtained from half-EPR pairs that were prepared and measured by the verifier V_1, the effect of this one-time pad is identical to not modifying ρ_2, but instead modifying the verifier's measurement outcomes z by XORing them with a string e_i' defined as in (3) with (a, b) on the left-hand side replaced by (x', z'). Let $\tilde{z}$ be the modified outcomes. Using that by definition for any t, π, a, b, r, d it holds that

$$\tilde{Q}(t, \pi, a, b, r, \tilde{z}, d) = \tilde{Q}(t, \pi, a, b, r, z, \tilde{d}) \ ,$$

where $\tilde{d}$ is computed from d as done by P' (described above), it follows that the simulation of P^* that is performed by P' is indistinguishable (from the point of view of P^*) of an actual execution of protocol $\tilde{\Pi}'$. Furthermore, since P' repeatedly executes P^* until the first repetition of the protocol accepts, the probability that P' succeeds in its interaction with V_2 is identical to the probability that P^* succeeds in the second repetition in $\tilde{\Pi}'$, conditioned on the first repetition succeeding. This probability is exactly p_2, which according to the discussion above is non-negligibly larger than s. This is a contradiction and concludes the proof.

□

Lemma 4.7. *Let (Q, α, β) be a* QMA *relation, and let $\Pi = (S, P, V)$ be the argument system for (Q, α, β) which is described in Section 3. Suppose that Π is computationally zero-knowledge with respect to any quantum polynomial time verifier with (quantum) auxiliary input.*[7] *Then $\Pi^{(k)} = (S^{(k)}, P^{(k)}, V^{(k)})$ is also computationally zero-knowledge with respect to any quantum polynomial time verifier with quantum auxiliary input.*

Proof. Consider a single execution of $\Pi^{(k)}$, in which $V^{*(k)}$, a potentially cheating verifier, outputs k quantum messages, and receives k proofs in return:

[7] This will be proven in section 5.

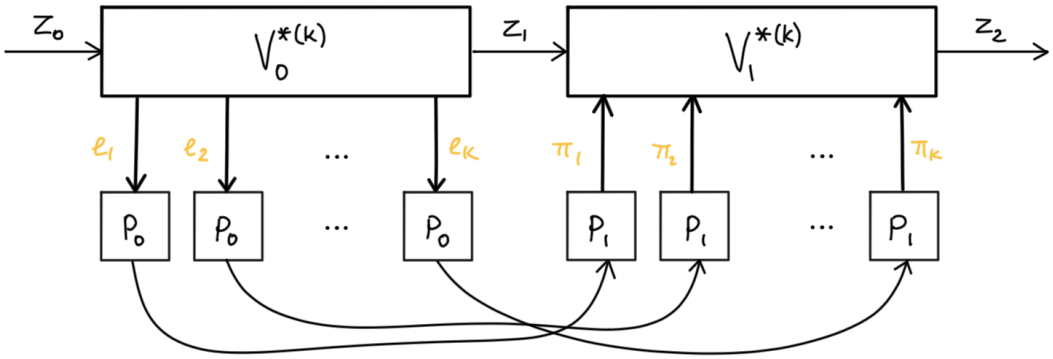

Figure 4: The first stage of the cheating verifier, $V_0^{*(k)}$, takes in an auxiliary input Z_0 and outputs k (quantum) setup messages $e_1, \ldots, e_k$, each of which goes into one copy of P_0. $V_0^{*(k)}$ then communicates with $V_1^{*(k)}$, the second stage of the cheating verifier, by sending an arbitrary quantum state Z_1. $V_1^{*(k)}$ takes proofs $\pi_1, \ldots, \pi_k$ from the k copies of P_1, along with Z_1 from $V_0^{*(k)}$, and outputs an arbitrary quantum state Z_2.

Notice that Z_1 is entirely independent of the behaviour of $P^{(k)}$. As such, in order to guarantee that Z_2 remains computationally indistinguishable from what it would be in a real execution of the protocol, we need only simulate the following:

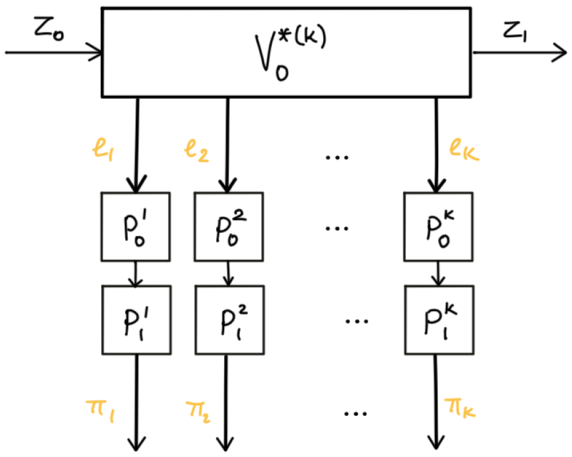

Figure 5: If we can create a simulation which takes in Z_0 and outputs $\pi_1', \ldots, \pi_k'$ which are indistinguishable from $\pi_1, \ldots, \pi_k$ that arise from a real execution of the protocol, then we can also simulate Figure 4.

That Figure 5 is efficiently simulable follows directly from the fact that Π is zero-knowledge. The simulator $S_{\Pi^{(k)}}$ would run $V_0^{*(k)}$ on Z_0 to obtain $Z_1, e_1, \ldots, e_k$, and then replace each of the k pairs $(P_0^i, P_1^i), i \in \{1, \ldots, k\}$ in the diagram above with a simulator S^i (which is guaranteed to exist by the auxiliary-input zero-knowledge property of Π).

Remark 4.8. *Note that it is in general not possible to extend this simple argument to a multi-round protocol. The intuitive reason is that, in a multi-round protocol, the parallel instances of the protocol may depend on each other, because the cheating verifier may introduce cross-correlations. More formally, the assumption we made that the internal communications of the cheating verifier do not depend on the behaviour of $P^{(k)}$ is false if the protocol is multi-round.*

□

5 Zero-knowledge property

Lemma 5.1. *Let $V = \{V_{\mu n}\}$ be an arbitrary quantum polynomial time (QPT) verifier for the protocol of Section 3. There exists a QPT simulator $S = \{S_{\mu n}\}$ such that, for any μ, n and yes-instance x with $|x| = n$, and for any auxiliary quantum input Z_0 to the verifier, the distribution of V's final output after its interaction with the honest prover P in the protocol is quantum computationally indistinguishable from S's output distribution on auxiliary input Z_0.*

We show that our protocol is zero-knowledge by replacing the components of the honest prover with components of a simulator one at a time, and demonstrating that, when the input is a yes-instance, the dishonest verifier's output after each replacement is made is at the least computationally indistinguishable from its output before. The end result, after all replacements have been made, is an efficient simulation which does not require access to the witness ρ. The proof follows an outline that is similar to that of the proof of zero-knowledge in [BJSW16]. We emphasise those aspects of our proof which do not appear in the proof of zero-knowledge in [BJSW16], and refer the reader to [BJSW16, Section 5] for a more detailed exposition of the steps that the two proofs share.

5.1 The original protocol

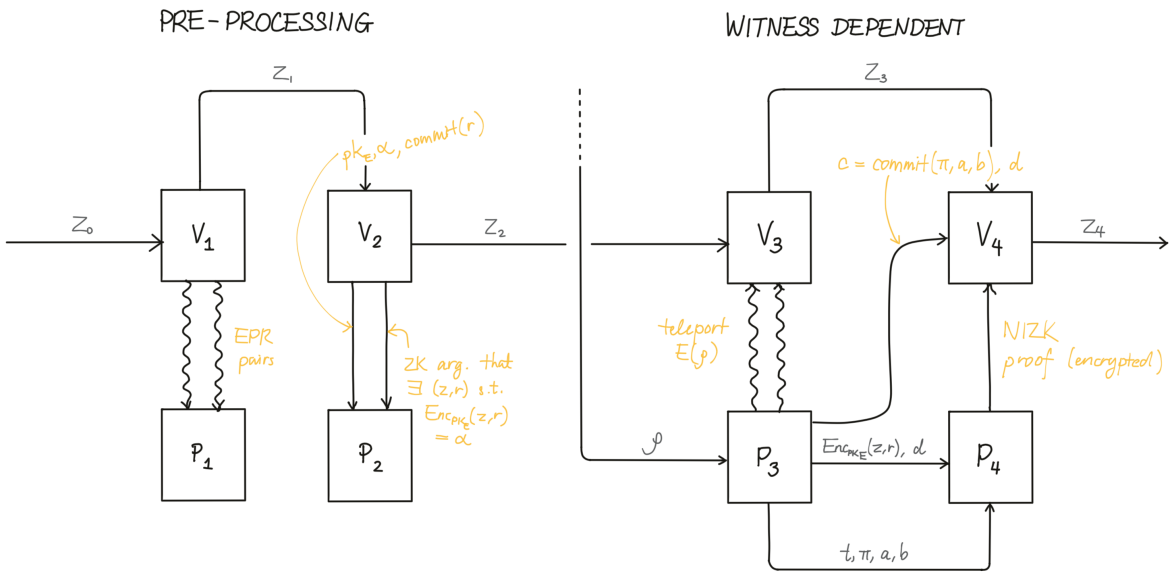

Figure 6: Diagram representing the original protocol. Communications between verifier and prover are labelled in orange; internal communications on either side are labelled in grey. In the two subsequent diagrams, we will omit the auxiliary input Z_0 that the cheating verifier receives, as well as the internal communications Z_1, Z_2, Z_3 between the different parts of the cheating verifier. (Note, however, that the auxiliary input Z_0 and these internal communications make a reappearance in Figure 9.) In the two subsequent diagrams, we will highlight replacements in green.

5.2 Replacing the NP proof and the commitment

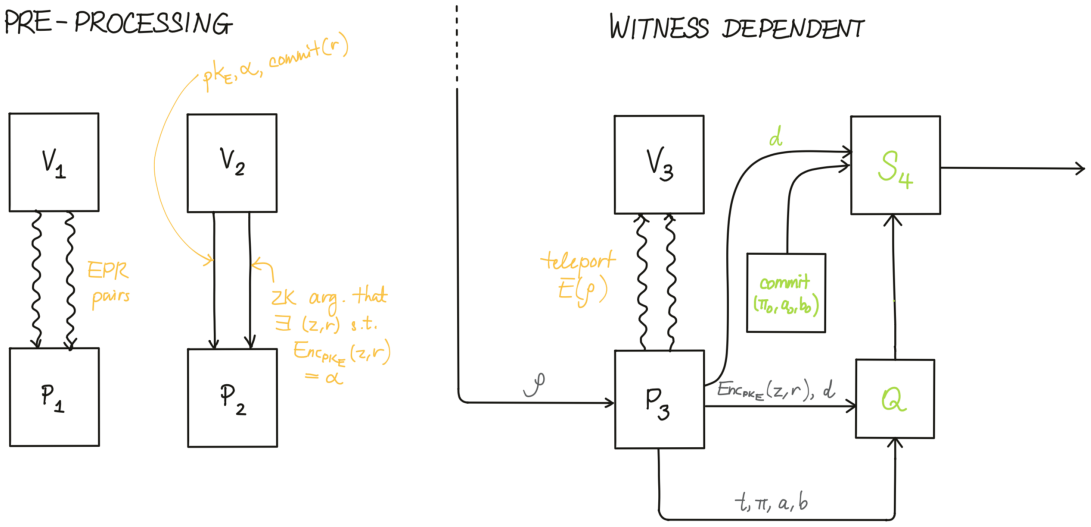

Figure 7: Figure 6 after the NIZK NP proof has been replaced by a simulation, and the commitment z has been replaced by a commitment to a fixed string.

We replace the interaction between P_4 and V_4 with the entities Q and S_4. (Recall that P_4 is the part of the prover which executes steps 4 and 5 in Figure 3, and that V_4 is the part of the verifier which is meant to perform the 'verifier's check' described in the same figure.) Q is an entity which evaluates the predicate Q (see Definition 3.5 for the definition of the predicate Q) homomorphically, and outputs an encryption of the result, which is either 0 or 1. S_4 takes in Q's output, and homomorphically does one of two things. If the output of Q is (homomorphically) 0, S_4 runs V_4 on an encryption of the abort symbol. If the output of Q is homomorphically 1, then S_4 homomorphically runs the simulation that is guaranteed to exist by the zero-knowledge property of the NIZK NP proof system that V_4 and P_4 use. From the simulation, S_4 obtains an encryption of a bogus (simulated) proof that is computationally indistinguishable from a real proof. S_4 then runs V_4 with this encryption of a simulated proof as input. (In each case, S_4 ensures circuit privacy by *refreshing* the ciphertext that it puts into V_4 using FHE.Refresh. See section 2.3.1 for a definition of FHE.Refresh and its properties.)

We argue that V_4's output after this replacement occurs is computationally indistinguishable from its output before. In the original protocol, P_4 homomorphically evaluates the predicate Q to check that it is satisfied, and if it is, P_4 provides an encryption of a zero-knowledge proof to V_4 of the NP statement 'There exists a commitment string s_P and an encoding key (π, a, b) such that $c = \mathsf{Com.commit}(pk, (\pi, a, b), s_P)$ and $Q(t, \pi, a, b, r, z, d) = 1$.' (We are guaranteed that this encryption of a zero-knowledge proof, generated by P_4 through a homomorphic computation, is indistinguishable from a fresh encryption of the proof, by the circuit privacy property of the FHE scheme in use.) Let S be the predicate that is 1 if and only if the NP statement 'There exists a com-

mitment string s_P...' is true. Then, because P_4 is honest, S is satisfied whenever Q is satisfied; and, by definition, Q is satisfied whenever S is satisfied. Therefore, if the simulation S_4 aborts if and only if Q is not satisfied, S_4 will abort exactly when P_4 would have aborted. When Q is satisfied, meanwhile, the bogus proof that the simulation of P_4 generates is computationally indistinguishable from a real yes-instance proof produced by P_4, by the (adaptive) zero-knowledge property of the NIZK NP proof system in the presence of auxiliary input which was shown in section 2.5.3. Furthermore, by the circuit privacy property of the FHE scheme in use, we are guaranteed that the encryptions generated by P_4 and by S_4 are both indistinguishable from fresh encryptions of, respectively, the real proof and the bogus proof. Therefore, V_4 (being efficient) is computationally unable to distinguish between P_4 and the combination of Q and S_4, which means its output after the replacement occurs must be computationally indistinguishable from its output before.

Having made this first replacement (i.e. turning P_4 and V_4 into Q and S_4), we can then replace the commitment c with a commitment to a fixed encoding key (π_0, a_0, b_0) that is independent of the problem instance. The reasoning that justifies this replacement is the same as in [BJSW16, Section 5, step 3]: because the commitment is computationally hiding, and because it is never opened after the NP proof has been replaced by a simulation, the real commitment c is computationally indistinguishable from $c' = \mathsf{Com.commit}(pk, (\pi_0, a_0, b_0), s_P)$ to all of the entities in the diagram above, given that all of said entities are quantum polynomial-time. As such, after both replacements have been made, the output of S_4 is computationally indistinguishable from the output of V_4 in the original protocol.

5.3 Replacing ρ with ρ_r

We now claim that, if we replace the real witness ρ with a simulated witness ρ_r in Figure 7, S_4's output is computationally indistinguishable from what it was in Figure 7. (This claim holds regardless of the verifier's choice of r.) The simulated witness ρ_r is a state that, given r, can be efficiently constructed.

For the remainder of section 5.3, we assume that r is chosen and published by the verifier before the witness-dependent part of the protocol begins, so that r is fixed and known to all entities which make use of it (such as Q). Although this is, in the real world, untrue—in the real protocol, the verifier only sends a *commitment* to r, which hides the true value of r from the prover—we will show in section 5.4 that the simulator can *extract* r from the verifier's perfectly binding commitment, so that the effect is the same (for the simulator, not the real prover) as if the verifier had published r.

5.3.1 Pauli-twirling the verifier

Recall that P_3 is the part of the prover which receives the witness ρ, applies an encoding to ρ, and then teleports $E(\rho)$ to the verifier. In order to facilitate the analysis, we firstly separate the action of P_3 from Figure 7 into two parts: we make applying the encoding E the responsibility of a separate entity labelled E, and delegate the remainder of P_3's actions to a new entity P_3'. Since P_3 was honest, this replacement results in no change in the output of S_4.

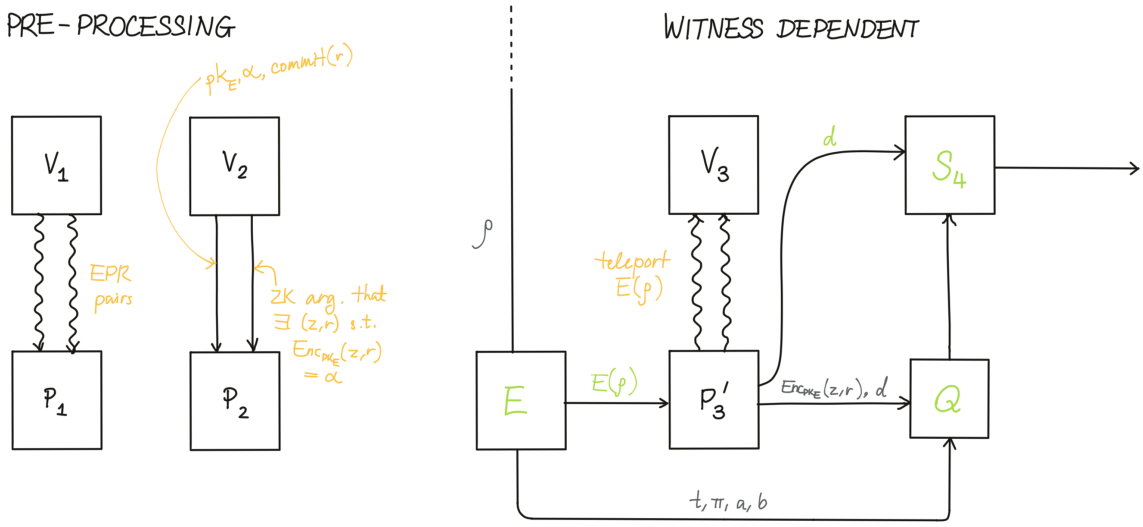

Figure 8: E represents a procedure that samples an encoding key from the same distribution from which P_3 would have sampled its encoding key in Figure 7, applies the encoding represented by that key to ρ, and sends the encoded ρ to P_3 and the key it chose to Q. Because P_3 was honest, the output distribution of S_4 in this diagram is exactly the same as the output distribution of S_4 in Figure 7.

Now, we consider $(V_1, P_1), (V_2, P_2), (V_3, P_3')$ taken together to be a cheating verifier V':

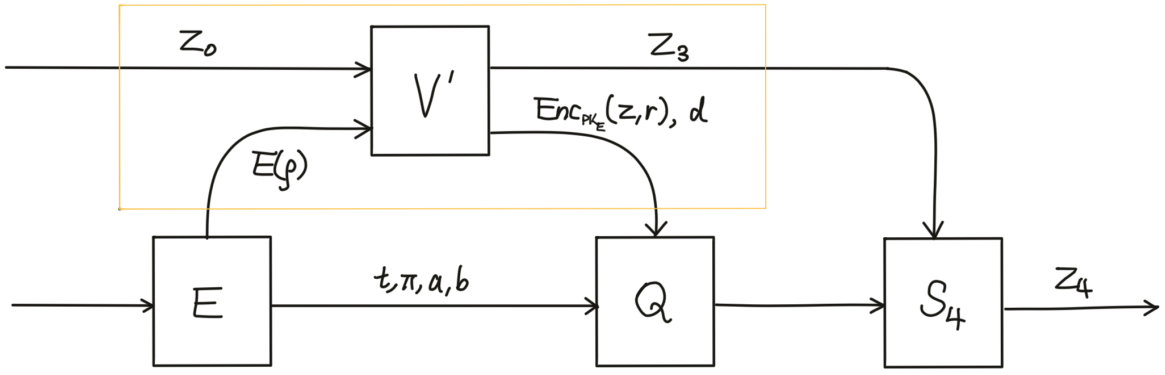

Figure 9: V' receives the encoded witness, $E(\rho)$, along with auxiliary information Z_0. It sends some arbitrary auxiliary Z_3 to S_4, and also sends $d = (a', b')$ and the homomorphic encryptions of z, r to Q. Q receives its other inputs from E.

We will now make a series of simplifications to Figure 9 such that, after each simplification, the distribution of Z_4 is statistically indistinguishable from what it was before the simplification.

Firstly, consider the region of Figure 9 inside the orange box, taken together with

1. the *last* step involved in applying E (that is, the application of the Pauli one-time-pad $X^a Z^b$),
2. the additional one-time-pad applied by the prover's teleportation that resulted in d (which is $X^{a'} Z^{b'}$), and
3. the *first* step involved in calculating Q (which is to 'undo' the Pauli one-time-pad $X^{a \oplus a'} Z^{b \oplus b'}$, keeping in mind that the honest verifier would have applied C_r transversally in the interim).

Let e, f be the unique strings such that $X^{a \oplus a'} Z^{b \oplus b'} = C_r X^e Z^f C_r^*$. Recall that V_2 (of Figure 8 and earlier) provided a proof that the string which we have called $\mathsf{Enc}_{pk_E}(z, r)$ is in fact a valid encryption under pk_E of some message (z, r). (For notational simplicity, we will from now until the end of this section omit the subscript E when we refer to the public key for FHE.) We assume the proof has negligible soundness error. If the prover P_2 accepted that proof, then (except with negligible probability) there exists a process $\tilde{V}'$ and a process $\widetilde{\mathsf{Enc}}_{pk}$ (the latter of which samples randomness of encryption from some well-defined, though not necessarily efficiently sampleable, distribution) such that the figure below is equivalent to the contents of the orange box in Figure 9 plus the last step of E, the prover's teleportation one-time-pad, and the first step of Q:

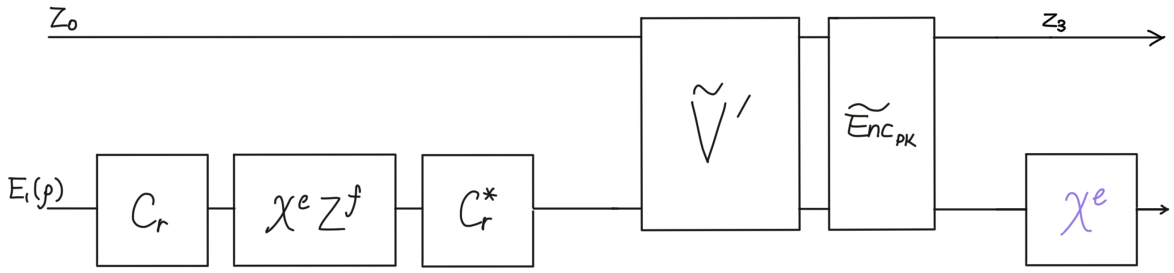

Figure 10: E_1 denotes the application of the first three steps of the encoding E. The last X^e operation is shown in to indicate that it is applied *homomorphically* on an encrypted input. We emphasise that $\widetilde{\mathsf{Enc}}_{pk}$ is not an arbitrary operation: due to the zero-knowledge proof that V_2 provides, its action is certainly to encrypt the input on the bottom wire under the encryption key pk. The only freedom it has is to sample the randomness of encryption from some arbitrary distribution, which may depend upon the auxiliary input Z_0.

By the correctness property of the homomorphic encryption scheme, Figure 10 is equivalent to the below (for some $\widetilde{\mathsf{Enc}}'_{pk}$ that may be different from $\widetilde{\mathsf{Enc}}_{pk}$):

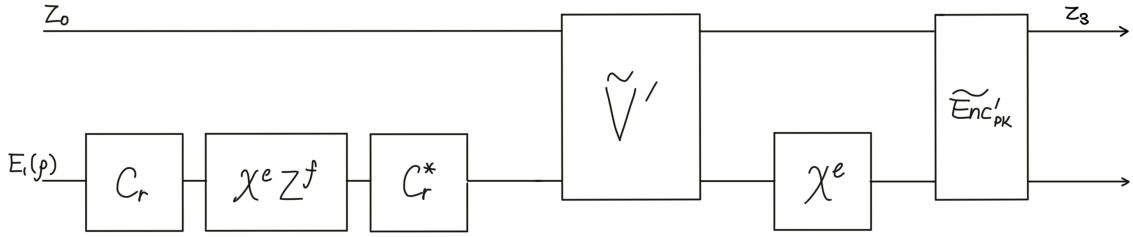

Figure 11: Note that the application of X^e is no longer homomorphic.

Following [BJSW16], we merge the operation C_r^* into $\tilde{V}'$, and apply a Pauli twirl in order to limit the action of $\tilde{V}'$ to an XOR attack which does not depend on the state in the lower wire:

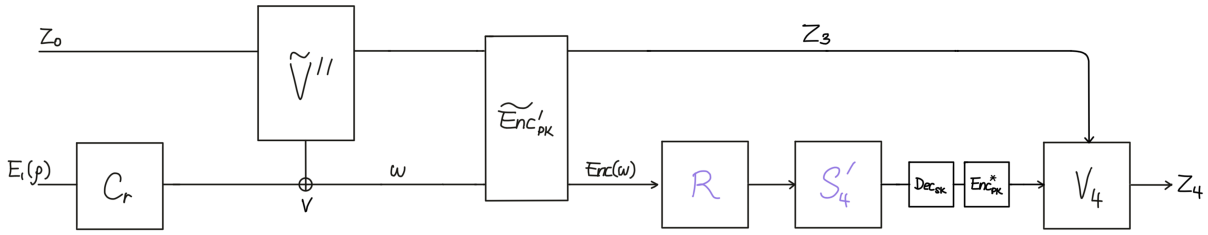

Figure 12: That which remains of the cheating verifier after the Pauli twirl is represented by the box $\tilde{V}''$, shown on the left. We have 'zoomed out' to show the entities into which the output wires in Figure 11 are fed. R represents the procedure that evaluates the remainder of Q ('the remainder' being that part of Q which remains after the correction X^e has already been made). S'_4 represents the entity, described in section 5.2, which homomorphically runs a simulation of the NP ZK proof system in order to obtain a simulated proof. The two boxes Dec_{sk} and Enc^*_{pk} make explicit the fact that the ciphertext (i.e. the encryption of the simulated proof obtained by S'_4) is 'refreshed' before it is fed into V_4. Enc^* is defined in section 2.3.1; its chief property is that it samples randomness of encryption from some known distribution.

Using again the correctness property of the homomorphic encryption, we can 'commute' the boxes with purple text in the diagram above past Dec_{sk} and turn them into boxes with black text. (Recall that boxes with purple text represent circuits that are evaluated homomorphically, and boxes with black text represent circuits evaluated in the clear.)

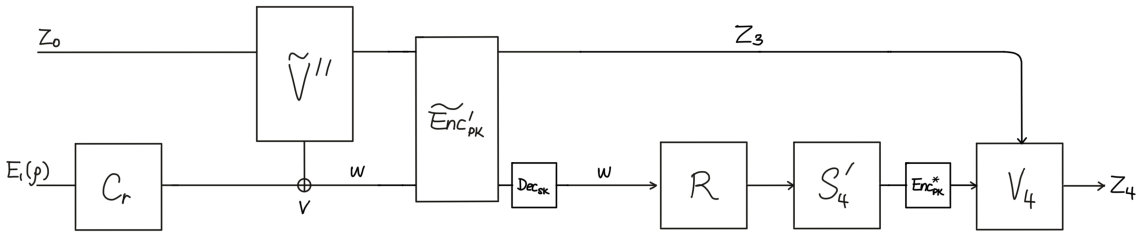

Figure 13: R and S_4' have been 'commuted' past Dec_{sk}.

Notice now that any action which $\widetilde{\mathsf{Enc}}_{pk}$ might have had on the lower wire is entirely cancelled by the Dec_{sk} that follows it, since the state of the wire before and after the action of both boxes is w. We can therefore reduce $\widetilde{\mathsf{Enc}}_{pk}$ to its action on the upper wire, and then merge its residual action into that of the cheating verifier $\tilde{V}''$.

Since Figure 13 is equivalent to (a subset of) Figure 9, it suffices to show that the output of Enc_{pk} in Figure 13 when $E_1(\rho_r)$ is fed in on the left (statistically) indistinguishable from its output when $E_1(\rho)$ is fed in on the left. This statement is implied by the statement that the output of R in Figure 13 when $E_1(\rho_r)$ is fed in on the left (statistically) indistinguishable from its output when $E_1(\rho)$ is fed in on the left. If we can show the latter, then we can substitute Figure 13 back into Figure 9, and conclude that—by the statistical indistinguishability which the preceding argument has proven—the verifier's final output Z_4 when ρ_r is used is statistically indistinguishable from its output when the real witness ρ is used.

As such, it suffices to show that the output of R_r (we have made the fact that r is fixed at the start of the protocol explicit now) in Figure 14 when ρ_r is given as input is (statistically) indistinguishable from its output when ρ is given as input.

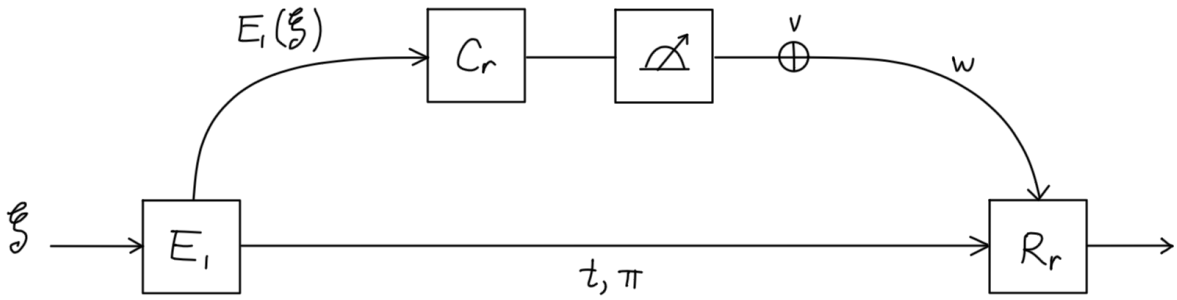

Figure 14: It is sufficient to argue that the output of R_r when $\xi = \rho_r$ and when $\xi = \rho$ are statistically indistinguishable.

5.3.2 Reducing to [BJSW16]

Figure 14 is identical to Figure 11 from [BJSW16]. As such, we can apply step 6 of their zero-knowledge proof, almost in a black-box fashion, in order to argue that the output of R_r when $\xi = \rho_r$ is statistically indistinguishable from its output when $\xi = \rho$.

One difference between their situation and ours of which we must be mindful is that our Hamiltonian (effectively) includes one extra term designed to check the registers of the witness state in which the prover is supposed to put a classical description of the instance. When any other term in our Hamiltonian is selected according to r, our predicate R_r can be defined as $R_r(t, \pi, u) = \tilde{R}_r(t, \pi, u)$, where $\tilde{R}_r$ is the predicate known as R_r in [BJSW16]. As such, in these cases, the argument in step 6 of [BJSW16, Section 5] applies verbatim. In the case when the special instance-check term is selected, the translation between our R_r and theirs is not so direct, but the substance of their argument is not affected. It is still true that, in order to change the *logical* meaning of u, the verifier must apply an XOR attack of large Hamming weight, with the result that the probability its XOR attack goes undetected by the traps t becomes negligibly small. As such, even though the specific form of the correctness check condition (condition (a) in Definition 2.1 and condition 1. in Definition 3.2) is different when the instance-check term from our Hamiltonian is selected, the argument made in step 6 of [BJSW16] still holds.

It follows from the argument in [BJSW16], then, that when ρ in Figure 8 is replaced by ρ_r, where ρ_r is a state that passes the particular challenge determined by r with probability 1, the output of S_4 is computationally indistinguishable from what it was before the replacement occurred. ρ_r in our case can be constructed efficiently, just as it can be in the [BJSW16] protocol: for those rs that do not correspond with the instance check term, ρ_r is the same that it would be in the [BJSW16] protocol, and for those rs that do correspond with the instance check term, a state which had the correct instance description in the relevant registers and the state $|0\rangle$ on all other qubits would pass with probability 1.

5.4 Feasibility of the $\rho \to \rho_r$ replacement

It remains only to argue that, as we claimed at the start of section 5.3, the simulator can efficiently extract r from the verifier's commitment to r. (We assume the latter to be perfectly binding, so that, conditioned on the prover accepting when it checks the verifier's commitment homomorphically, r is well-defined.) The simulator does this by taking advantage of the CRS. The verifier expects as input a CRS which contains in it the public key pk_V that it is to use for the commitment scheme Com. We assume, however, that Com is *extractable* (see section 2.3.2), meaning that every valid public key for Com has a corresponding secret key sk_V such that the efficient algorithm $\mathsf{Com.recover}$ can, given sk_V and a commitment $z = \mathsf{Com.commit}(pk_V, b, s)$, recover (b, s) from z. Then, in order to extract r, the simulator simply samples a public key, secret key *pair* $(pk_V^*, sk_V^*) \leftarrow \mathsf{Com.gen}$; inserts pk_V^* into the CRS when it gives the verifier the CRS as input; and recovers r from the verifier's commitment to r using $\mathsf{Com.recover}$ and sk_V^*.

6 NIZK argument of quantum knowledge with setup for QMA

In this section we show that for any QMA relation the NIZK argument system with CRS setup described in section 3 is also a NIZK Argument of Quantum Knowledge with CRS setup (as defined in section 2.6). The intuition for this is simple. From the proof of soundness of the protocol from [BJSW16], to which soundness of our argument system reduces, we are able to infer that any prover which is accepted in our protocol with high probability must be teleporting to the verifier *an encoding* of a low-energy witness state for the given instance of the 5-local Clifford Hamiltonian problem. Then, all that an extractor (given oracle access to such a prover) has to do in order to output a good witness is:

- Simulate an honest verifier so as to receive (by teleportation) such an encoded witness from the prover,
- Find a way to recover the committed encoding keys and use them to decode the received state.

The extractor recovers the keys by using the same technique used multiple times in the soundness proof of section 4. Namely, the extractor samples a CRS which includes a public key for the prover's commitment scheme for which it knows the associated secret key. The distribution of such CRS is identical to the true distribution, but the additional side information allows the extractor to recover the authentication keys once the prover sends his commitment. In the rest of this section, we formalize this sketch.

The following lemma, that is a direct consequence of the soundness proof in [BJSW16], will be useful in the analysis.

Lemma 6.1. *For any $N \in \mathbb{N}$ there exists an efficiently implementable quantum channel Φ from $2N$ qubits to 1 such that the following holds for any 5-local Clifford Hamiltonian H on n qubits: let P be a prover for the protocol of [BJSW16] (where N is the number of physical qubits per logical qubit in the concatenated Steane code), and suppose P is accepted with probability p on common input H. Let $\sum_\sigma p_\sigma \rho_\sigma \otimes |\sigma\rangle\langle\sigma|$ be the state of the message register of P after the first round. Assume $\sum_{\sigma \text{ is valid}} p_\sigma = 1$. Let ξ_σ be the state obtained from ρ_σ after applying the decoding map $Dec_{t,\pi,a,b}$ (from Definition 2.3), where t, π, a, b are the values committed in σ. Let $\xi = \sum_\sigma p_\sigma \xi_\sigma$. Then*

$$\mathrm{Tr}[H\Phi^{\otimes n}(\xi)] < 1 - p + \mathrm{negl}(n) \ .$$

Consider a QMA relation (Q, α, β) with $\alpha - \beta > 1/poly$ and the language $L \in$ QMA that it specifies. Given an instance x for the associated decision promise problem, let $H(x)$ be an instance of the (QMA-complete) 5-local Clifford Hamiltonian problem that is associated to it. We assume without loss of generality that for all $x \in L$, there exists a state σ such that $\mathrm{Tr}(H(x)\sigma) < \mathrm{negl}(|x|)$. Moreover, there exists $\gamma = 1/\text{ poly}$ such that for all $x \notin L$, for all σ, $\mathrm{Tr}(H(x)\sigma) > \gamma(|x|)$. This reduction has the property that there exists an efficiently implementable unitary U which maps a witness state for x, tensored with some auxiliary qubits in the state $|0^t\rangle$ for some t, to a witness state for $H(x)$. Let τ be the efficiently implementable quantum channel that takes as input a state σ, applies U^{-1} and traces out the auxiliary qubits. It is not hard to verify from the history state construction used to obtain the witness for $H(x)$ from a witness for x that the map τ has the prop-

erty that there exists a universal polynomial t such that whenever $\mathrm{Tr}(H(x)\sigma) \leq 1-\varepsilon$ for some state σ and $\varepsilon \leq 1$, it holds that $(x, \tau(\sigma)) \in R_{Q,1-\mathrm{poly}((1-\varepsilon)t)}$.

We are ready to state the main result of this section. For simplicity we only consider relations (Q, α, β) such that $\alpha = 1 - \mathrm{negl}(n)$ and $\beta = \mathrm{negl}(n)$; by [MW05] any relation can be amplified to such parameters without changing the witness for the case of a yes instance.

Theorem 6.2. *Let $(Q, 1-\mathrm{negl}(n), \mathrm{negl}(n))$ be a* QMA *relation. There is a polynomial t such that the following hold. There is a NIZK Argument of Quantum Knowledge with CRS setup and single-message preprocessing for (Q, α, β) with completeness $1-\mathrm{negl}$, soundness s (where s is the soundness from Thereom 3.6), knowledge error $1 - \frac{1}{2t}$ and quality $q(n, \varepsilon) = 1 - \mathrm{poly}((1-\varepsilon)t(n))$.*

Proof. Let $(Q, 1-\mathrm{negl}(n), \mathrm{negl}(n))$ be a QMA relation. We prove that the argument system with CRS setup from section 3 is also an argument of Quantum Knowledge with CRS setup. We describe an extractor K. Fix an instance x and a prover P' that is accepted with probability $p > 1 - \frac{1}{2t}$ in the protocol of section 3 on common input x. At a high level, K does the following:

1. K uses P' to construct a prover P'' for the protocol of [BJSW16]. The crucial part is that K is able to know the authentication keys that P'' commits to.
2. K runs P'' and obtains a state ρ of $2nN$ qubits and a commitment σ. Let (t, π, a, b) be the committed keys. K outputs $\Phi^{\otimes n}(Dec_{t,\pi,a,b}(\rho))$.

In more detail, the hybrid argument in section 4 implies that there exists an efficient A such that $A^{|P'\rangle}$ is accepted with probability at least $p - \mathrm{negl}(|x|)$ in the protocol from hybrid H_{11}, i.e. the protocol from [BJSW16]. In particular, the prover reduction in the last replacement $H_{10} \approx H_{11}$ implies that we can choose A such that after the first step of the instance-dependent part of the protocol the state of $A^{|P'\rangle}$ is (where the first two registers are message registers and the third register is an internal register):

$$\sum_{\sigma} p_\sigma \, \rho_\sigma \otimes |\sigma\rangle\langle\sigma| \otimes |(t, \pi, a, b)_\sigma\rangle\langle(t, \pi, a, b)_\sigma| \ , \tag{6}$$

where $(t, \pi, a, b)_\sigma$ is the value committed in σ, and all σ's that appear in the sum are valid commitments. The extractor K runs $A^{|P'\rangle}$ to obtain (6), and efficiently maps this to $\Xi = \sum_\sigma p_\sigma \Phi^{\otimes n} Dec_{t,\pi,a,b}(\rho_\sigma)$. Since $A^{|P'\rangle}$ is accepted with probability at least $p - \mathrm{negl}(|x|)$ in the protocol of [BJSW16], Lemma 6.1 implies that Ξ satisfies $\mathrm{Tr}(H(x)\Xi) < 1 - p + \mathrm{negl}(|x|)$. The final output of the extractor is $\tau(\Xi)$, where τ is the efficiently implementable channel defined at the start of this section. The claim on the quality of K follows from the property of τ stated below Lemma 6.1.

□

Remark 6.3. *Notice that K succeeds at extracting with probability 1. This is perhaps surprising given that the only known zero-knowledge proofs or arguments of knowledge for* NP *in the quantum settting [Unr12] have an extractor that only succeeds with inverse-polynomial probability. Here, our argument is non-interactive and it works for* QMA*! The main difference, which grants the additional power to the extractor, is the presence of the CRS setup.*

Remark 6.4. *The knowledge error in Theorem 6.2 is $1 - 1/poly$. This can reduced to negligible by sequential repetition according to Lemma 2.26, but this would forego non-interactiveness. We believe that*

starting from the parallel amplified version of our protocol from section 4.4, it should be possible to obtain a non-interactive argument of knowledge with negligible knowledge error, but we leave a more thorough investigation of this for future work.

7 Proofs of quantum knowledge for QMA

The interactive protocol that we show is a proof of quantum knowledge is identical to the protocol from [BJSW16], as recalled in Section 2.2, except for one modification: at the same time as the prover sends the encoded state $E(\rho)$ and the commitment σ to the verifier (end of step 1 of the protocol), the prover also sends a classical zero-knowledge PoK of an opening to the commitment. More precisely, define a relation R such that $R(\sigma, z) = 1$ if z is a valid opening for the commitment σ. V and P engage in a ZK PoK protocol for the relation R on common input σ, as defined in Definition 2.20. If the verifier rejects in this protocol, then the verifier outputs "reject" for the whole protocol; otherwise the verifier proceeds to the next phase.

We denote by Π_{PoK} the ZK PoK protocol for relation R. Using [Unr12, Corollary 21], we may assume that this protocol has negligible knowledge error.

The main result of this section is that with this modification, the protocol from [BJSW16] is a quantum proof of quantum knowledge for any language in QMA. As in Section 6 for simplicity we only consider relations (Q, α, β) such that $\alpha = 1 - \mathrm{negl}(n)$ and $\beta = \mathrm{negl}(n)$.

Theorem 7.1. *There is a polynomial t such that the following holds. Let $(Q, 1 - \mathrm{negl}(n), \mathrm{negl}(n))$ be a* QMA *relation. There is a Zero-Knowledge PoQK for $(Q, 1 - \mathrm{negl}(n), \mathrm{negl}(n))$ with completeness $1 - \mathrm{negl}(n)$, negligible soundness and knowledge error, and quality $q(n, \varepsilon) = 1 - \mathrm{poly}((1 - \varepsilon)t(n))$.*

Proof. Given a QMA relation $(Q, 1 - \mathrm{negl}(n), \mathrm{negl}(n))$ and an instance x, as in Section 6 we can efficiently construct a 5-local Clifford Hamiltonian $H(x)$ and an efficient quantum channel τ such that there is a polynomial t such that for any state ρ such that $\mathrm{Tr}(H\rho) \leq \varepsilon$ it holds that $(x, \tau(\rho)) \in R_{Q, 1-\mathrm{poly}(\varepsilon t)}$. We then consider the protocol from [BJSW16] for the Hamiltonian $H(x)$, modified as described at the start of the section. Completeness holds trivially. Provided t is chosen large enough, the soundness error is at most $1 - 1/t$. We first show that the knowledge error is at most $1 - 1/(2t)$. Using Lemma 2.6.1, both soundness and knowledge error can be improved by sequential repetition.

The Zero-Knowledge property follows essentially as in [BJSW16], with the only difference being the addition of the quantum zero-knowledge PoK of an opening to the commitment, that also has the Zero-Knowledge property.

We show that the protocol is a quantum proof of quantum knowledge by constructing an extractor K as follows. Let P^* be an arbitrary (quantum polynomial-time) prover in the protocol, and (x, ρ) a pair of a problem instance and a quantum witness such that the protocol execution $(P^*(x, \rho), V(x))$ returns 1 with probability at least $1 - s$ for some $s \leq \min\{1/t, 1/2\}$.

Informally, the extractor K first takes the quantum state ρ^* sent by P^* in the first message. It then executes an extractor K' for an opening to the commitment sent in the first message, that must exist

by the quantum proof of knowledge property for the sub-protocol. If K' succeeds in recovering the committed keys, K decodes the state received in the first message using these keys and returns the decoded state. Otherwise, K returns an abort symbol "$\perp$".

We provide more detail. Denote by N and S_P the message and state registers of P respectively. Initially, S_P contains the witness ρ and N is initialized to a zero string. Let U_P be the unitary applied by the prover at the first step of the protocol. (For simplicity, we omit dependence on the instance x.)

The extractor K operates as follows. (In the following description, we sometimes write for simplicity that "K measures a certain register". Formally, however, K does not perform any measurement, and what we mean is that K uses its auxiliary register $\mathsf{S}_{K_{aux}}$ to implement the channel corresponding to these measurements in a coherent way.)

(i) K executes U_P on registers N and S_P. Subsequently, K swaps the first $2nN$ qubits of N (the ones that the prover would have sent to the verifier) in an auxiliary register S_{K_q}, and the qubits corresponding to the "commitment" into an auxiliary register S_{K_c}. K measures S_{K_c} to obtain a string σ.

(ii) Denote by $\tilde{\rho}$ the state of S_{K_q}. Let ρ' be the leftover state of register S_P. Let K' be the knowledge extractor for Π_{QPoK}. K executes K' on the remaining registers of the prover to obtain an outcome z. If $R(\sigma, z) = 1$ then K proceeds to the next step. Otherwise K halts with the outcome "$\perp$" on its output register.

(iii) Let the opened commitment be $z = (t, \pi, a, b)$. K applies the map $Dec_{t,\pi,a,b}$ from Definition 2.3 to the state $\tilde{\rho}$ contained in register S_{K_q}, followed by the channel $\Phi^{\otimes n}$ from Lemma 6.1, followed by the map τ described at the start of the proof. K returns the resulting n-qubit state in its output register.

We show that K is a valid extractor for the PoQK property. First note that the state of $\mathsf{S}_{K_q} \otimes \mathsf{S}_{K_c}$ after step (i), conditioned on not returning "$\perp$", takes the form

$$\sum_v p_\sigma \rho_\sigma \otimes |\sigma\rangle\langle\sigma| \ , \tag{7}$$

where each σ is a valid commitment. Let $1 - s(\sigma)$ be the probability that V accepts P in the protocol, conditioned on V having received commitment σ. Then $\sum_\sigma p_\sigma s(\sigma) \leq s$. Let p, q be a polynomials such that K' succeeds at extracting an opening for a valid commitment with probability at least $\frac{1}{q(\mu)} p(1 - s(\sigma))$. Denote by ξ_σ the state returned by the verifier at step (iii) prior to applying the map τ, when the commitment is σ and conditioned on the extraction performed by K' having succeeded. Thus as long as $s(\sigma) < 1$ (which implies that σ is a valid commitment), the probability that the prover is eventually accepted in the protocol, conditioned on having given commitment σ and the extraction having succeeded is at least $1 - \frac{s(\sigma)q(\mu)}{p(1-s(\sigma))}$. Applying Lemma 6.1, we get

$$\mathrm{Tr}\big(H\xi_\sigma\big) \leq \frac{s(\sigma)q(\mu)}{p(1 - s(\sigma))} + \mathrm{negl}(n) \ .$$

By Markov's inequality, a fraction at least $1/2$ (under the distribution (p_σ)) of commitments σ are such that $s_\sigma \leq 2s$. For any such σ, conditioned on the extraction performed by K' succeeding, K

obtains a state such that $\mathrm{Tr}(H\xi_\sigma) < 2sq(\mu)/p(1-2s) + \mathrm{negl}(n)$. By the discussion at the start of the proof, in any such case the extractor returns $\tau(\xi_\sigma)$ such that $(x, \tau(\xi_\sigma)) \in R_{Q,1-\mathrm{poly}(2stq/p(1-2s))}$. To obtain the desired conclusion, we combine the polynomials q and t. □

References

[BDPMW16] Florian Bourse, Rafaël Del Pino, Michele Minelli, and Hoeteck Wee. Fhe circuit privacy almost for free. In Matthew Robshaw and Jonathan Katz, editors, *Advances in Cryptology – CRYPTO 2016*, pages 62–89, Berlin, Heidelberg, 2016. Springer Berlin Heidelberg. 2.3.1, 2.3.1, 2.3.1

[BDSMP91] Manuel Blum, Alfredo De Santis, Silvio Micali, and Giuseppe Persiano. Noninteractive zero-knowledge. *SIAM J. Comput.*, 20(6):1084–1118, December 1991. 1

[BG90] Mihir Bellare and Shafi Goldwasser. New paradigms for digital signatures and message authentication based on non-interactive zero knowledge proofs. In Gilles Brassard, editor, *Advances in Cryptology — CRYPTO' 89 Proceedings*, pages 194–211, New York, NY, 1990. Springer New York. 1

[BG92] Mihir Bellare and Oded Goldreich. On defining proofs of knowledge. In *Annual International Cryptology Conference*, pages 390–420. Springer, 1992. 2.6.1

[BIN97] Mihir Bellare, Russell Impagliazzo, and Moni Naor. Does parallel repetition lower the error in computationally sound protocols? In *Proceedings 38th Annual Symposium on Foundations of Computer Science*, pages 374–383. IEEE, 1997. 4.4

[BJSW16] Anne Broadbent, Zhengfeng Ji, Fang Song, and John Watrous. Zero-knowledge proof systems for QMA. In *2016 IEEE 57th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 31–40. IEEE, 2016. (document), 1, 1, 1, 1, 1, 1, 1, 1, 1, 2.2, 1, 4, 2.2, 3.1, 3.1, 3.2, 4.1, 4.1, 4.3, 4.3, 5, 5.2, 5.3.1, 5.3.2, 6, 6.1, 1, 6, 6, 7, 7

[BMW03] Mihir Bellare, Daniele Micciancio, and Bogdan Warinschi. Foundations of group signatures: Formal definitions, simplified requirements, and a construction based on general assumptions. In Eli Biham, editor, *Advances in Cryptology — EUROCRYPT 2003*, pages 614–629, Berlin, Heidelberg, 2003. Springer Berlin Heidelberg. 1

[CCRR18] Ran Canetti, Yilei Chen, Leonid Reyzin, and Ron D Rothblum. Fiat-shamir and correlation intractability from strong kdm-secure encryption. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 91–122. Springer, 2018. 2.5.3

[CLW19] Ran Canetti, Alex Lombardi, and Daniel Wichs. Fiat-shamir: From practice to theory, part ii. 2019. 1, 2.5.3, 2.5.3, 2.5.3, 2.5.3, 6

[Com14] Electric Coin Company. Zcash Cryptocurrency . 2014. 1

[CP92] David Chaum and Torben Pryds Pedersen. Wallet databases with observers. In *Annual International Cryptology Conference*, pages 89–105. Springer, 1992. 1

[FLS99] Uriel Feige, Dror Lapidot, and Adi Shamir. Multiple noninteractive zero knowledge proofs under general assumptions. *SIAM Journal on Computing*, 29(1):1–28, 1999. 2.5.3

[GGPR13] Rosario Gennaro, Craig Gentry, Bryan Parno, and Mariana Raykova. Quadratic span programs and succinct NIZKs without PCPs. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 626–645. Springer, 2013. 1

[GMR85] S Goldwasser, S Micali, and C Rackoff. The knowledge complexity of interactive proof-systems. In *Proceedings of the Seventeenth Annual ACM Symposium on Theory of Computing*, STOC '85, pages 291–304, New York, NY, USA, 1985. ACM. 1, 2, 1

[GO94] Oded Goldreich and Yair Oren. Definitions and properties of zero-knowledge proof systems. *J. Cryptol.*, 7(1):1–32, December 1994. 1

[Kob02] Hirotada Kobayashi. Non-Interactive Quantum Statistical and Perfect Zero-Knowledge. *arXiv e-prints*, pages quant–ph/0207158, Jul 2002. 1

[Lab17] O(1) Labs. Coda Cryptocurrency . 2017. 1

[Mah18] Urmila Mahadev. Classical verification of quantum computations. In *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 259–267. IEEE, 2018. 1

[MP12] Daniele Micciancio and Chris Peikert. Trapdoors for lattices: Simpler, tighter, faster, smaller. In David Pointcheval and Thomas Johansson, editors, *Advances in Cryptology – EUROCRYPT 2012*, pages 700–718, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg. 2.3.2, 2.3.2

[MW05] Chris Marriott and John Watrous. Quantum arthur–merlin games. *Computational Complexity*, 14(2):122–152, 2005. 2.5.2, 6

[NY90] M. Naor and M. Yung. Public-key cryptosystems provably secure against chosen ciphertext attacks. In *Proceedings of the Twenty-second Annual ACM Symposium on Theory of Computing*, STOC '90, pages 427–437, New York, NY, USA, 1990. ACM. 1

[PHGR13] Bryan Parno, Jon Howell, Craig Gentry, and Mariana Raykova. Pinocchio: Nearly practical verifiable computation. In *2013 IEEE Symposium on Security and Privacy*, pages 238–252. IEEE, 2013. 1

[PS19] Chris Peikert and Sina Shiehian. Noninteractive zero knowledge for np from (plain) learning with errors. *IACR Cryptology ePrint Archive*, 2019:158, 2019. 1, 1, 1, 2.5.3, 2.5.3, 2.5.3

[Reg09] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. *Journal of the ACM (JACM)*, 56(6):34, 2009. 2.3.2, 2.5, 2.3.2, 2.3.2

[Sah99] Amit Sahai. Non-malleable non-interactive zero knowledge and adaptive chosen-ciphertext security. In *Proceedings of the 40th Annual Symposium on Foundations of*

Computer Science, FOCS '99, pages 543–, Washington, DC, USA, 1999. IEEE Computer Society. 1

[Sho95] Peter W. Shor. Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer. *arXiv e-prints*, pages quant–ph/9508027, Aug 1995. 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, 2.4.1, 2.6, 2.6.1, 2.6.1, 6.3, 7

[Unr15] Dominique Unruh. Non-interactive zero-knowledge proofs in the quantum random oracle model. In Elisabeth Oswald and Marc Fischlin, editors, *Advances in Cryptology - EUROCRYPT 2015*, pages 755–784, Berlin, Heidelberg, 2015. Springer Berlin Heidelberg. 1

[VZ19] Thomas Vidick and Tina Zhang. Classical zero-knowledge arguments for quantum computations. *arXiv e-prints*, page arXiv:1902.05217, Feb 2019. 2.2

[Wat09] John Watrous. Zero-knowledge against quantum attacks. *SIAM Journal on Computing*, 39(1):25–58, 2009. 5, 1, 2.5.1

[Wie83] Stephen Wiesner. Conjugate coding. *ACM Sigact News*, 15(1):78–88, 1983. 1, 4