Secure Quantum Computation with Classical Communication

James Bartusek[*]

University of California, Berkeley

Abstract

We construct a constant-round composable protocol for blind and verifiable classical delegation of quantum computation, and show applications to secure quantum computation with classical communication. In particular, we give the first maliciously-secure multi-party protocols for BQP (bounded-error quantum polynomial-time) computation that only require a single party to have quantum capabilities. Assuming QLWE (the quantum hardness of learning with errors), we obtain the following.

- A six-round protocol between one quantum server and multiple classical clients in the CRS (common random string) model.
- A three-round protocol between one quantum server and multiple classical clients in the PKI (public-key infrastructure) + QRO (quantum random oracle) model.
- A two-message protocol between quantum sender and classical receiver (a quantum non-interactive secure computation protocol), in the QRO model.

The only previously known approach for obtaining composable security of blind classical verification of quantum computation (Gheorghiu and Vidick, FOCS 2019) has inverse polynomial security and requires polynomially many rounds of interaction.

The property we require of classical verification of quantum computation that enables composability is *malicious blindness*, which stipulates that the prover does not learn anything about the verifier's delegated computation, even if it is able to observe whether or not the verifier accepted the interaction. To construct a protocol with malicious blindness, we use a classical verification protocol for sampBQP computation (Chung et al., Arxiv 2020), which in general has inverse polynomial soundness error, to prove honest evaluation of QFHE (quantum fully-homomorphic encryption) ciphertexts with negligible soundness error. Obtaining a constant-round protocol requires a strong parallel repetition theorem for classical verification of quantum computation, which we show following the "nearly orthogonal projector" proof strategy (Alagic et al., TCC 2020).

1 Introduction

Secure multi-party computation (MPC) is a fundamental cryptographic task that allows for multiple parties to securely evaluate a function on their joint private inputs. The study of MPC is foundational to the field of modern cryptography [Yao86, GMW87, BGW88, CCD88] and has since only increased in depth and scope.

Naturally, the vast majority of MPC literature considers the task of securely evaluating a classical functionality over classical inputs. However, the emergence of quantum computing technology raises several interesting and increasingly relevant questions for the field of secure computation. Indeed, secure computation of quantum functionalities over potentially quantum inputs also has a rich history of study, including many recent works [CGS02, BCG$^+$06, Unr10, DNS12, KP17, DGJ$^+$20, ACC$^+$20, GLSV21, BCKM21a, BCKM21b, KKMO21].

[*] Email: bartusek.james@gmail.com

One drawback of each of the above multi-party quantum computation (MPQC) protocols is that they require each party to operate a quantum computer, or at least be able to manipulate some quantum information. Indeed, personal quantum computers remain far from a reality, and it appears as if quantum computation will be concentrated in the hands of a few technologically-advanced entities for the foreseeable future. Thus, recent years have seen a major research effort towards the goal of *classical delegation of quantum computation*, which allows a classical client to enlist the resources of a quantum server, ideally without comprising the privacy of the client's data or the integrity of the computation [Mah18a, Mah18b, Bra18, GV19, ACGH20, CCY20, CLLW20].

These works consider a single classical client with (potentially private) input, interacting with an inputless quantum server. Thus, they do not address the possibility of *multi-party* quantum computation with classical communication, and do not attempt to realize fully simulation-secure protocols, which is the gold standard notion of security for distributed computation. In this work, we address the following feasibility question for the first time, where "securely compute" refers to simulation security against arbitrarily malicious parties.

Can multiple parties, some of which do not have any quantum resources, securely compute a quantum functionality over their joint private inputs?

1.1 Results

We study the notion of MPQC with classical communication secure against a dishonest majority of arbitrarily malicious parties. We focus on MPQC for BQP (bounded-error quantum polynomial-time) computation.[1] We capture BQP by considering "pseudo-deterministic" quantum functionalities $D(\cdot)$ that on classical input x, produce a fixed classical output z except with negligible probability.

Composable blind CVQC. We begin by considering a simple two-party functionality between classical client and quantum server, defined by a pseudo-deterministic circuit $D(\cdot)$. It takes an input x from the client and a bit b from the server and delivers the output $D(x)$ to the client if $b = 0$ and the output $\perp$ if $b = 1$. We say that a protocol with classical communication that securely implements this ideal functionality is a *composable blind CVQC* protocol. Our first result is described in the following informal theorem.

Theorem 1.1. *(Informal) Assuming the quantum hardness of learning with errors (QLWE), there exists a four-round composable blind CVQC protocol. The protocol can be made two rounds in the quantum random oracle model.*

Next, we give applications of composable blind CVQC to secure quantum computation.

Multi-party results. In the multi-party setting, we construct two protocols, both of which only require a *single* party (called the server) to have quantum capabilities. This setting of one quantum server and several classical clients can be viewed as a quantum analogue of "cloud-assisted" MPC, introduced by [AJL$^+$12]. In their setting, several clients wish to securely outsource the bulk of some computation to a single powerful server, and they require that the client computation is much smaller than the functionality to be computed. In our setting, we consider several classical clients that wish to outsource a quantum computation to a single quantum server, and we require that the client computation is entirely classical (though it may grow with the size of the functionality to be computed).

The features of our first protocol are described in the following theorem.

Theorem 1.2. *(Informal) Assuming the quantum hardness of learning with errors (QLWE), there exists a six-round protocol (in the common random string model) between multiple classical clients and one quantum server for computing any pseudo-deterministic quantum functionality over the private inputs of the clients. The protocol tolerates any coalition (including client-server collusion) of malicious quantum polynomial-time adversaries.*

[1] As opposed to the more general class of quantum functionalities that output an arbitrary distribution over classical strings (see discussion in Section 1.3). Note that this distinction generally does not arise in the classical setting, since one can make any randomized functionality deterministic by fixing the random coins. In the quantum setting, this strategy will not always work since randomness can come from measurement.

We next study cloud-assisted MPQC with the following interaction pattern.

- Round 1: each classical client P_i computes and broadcasts an encryption ct_i of their input x_i.
- Round 2: the quantum server computes and broadcasts an encryption $\widetilde{\mathsf{ct}}$ of the output.
- Round 3: the clients participate in a one-round decryption procedure that delivers output y_i to each client P_i.

The feasibility of this interaction pattern in the classical setting was established by [AJL+12] in the PKI (public-key infrastructure) model, where each client can publish a succinct and reusable public key before the protocol begins.[2] Here, we show how to achieve three-round cloud-assisted MPQC in the QROM (quantum random oracle model), also assuming a PKI setup, as described in the following theorem.

Theorem 1.3. *(Informal) Assuming QLWE, there exists a three-round protocol (in the QRO + PKI model) between multiple classical clients and one quantum server for computing any psuedo-deterministic quantum functionality over the private inputs of the clients. The protocol tolerates any coalition (including client-server collusion) of malicious quantum polynomial-time adversaries.*

Two-party results. In the two-party setting, we show how to construct a *round-optimal* (two-message) protocol where one party receives output. That is, we consider a quantum sender $\mathcal{S}$ with classical input $x_{\mathcal{S}}$ and a classical receiver $\mathcal{R}$ with classical input $x_{\mathcal{R}}$, and we construct a maliciously-secure two-message protocol that delivers $D(x_{\mathcal{R}}, x_{\mathcal{S}})$ to the receiver. This can be seen as a (non-reusable) NISC (non-interactive secure computation) protocol for BQP. Both non-reusable and reusable NISC protocols have a long history of study in the classical setting [Yao86, IPS08, IKO+11, AMPR14, BGI+17, CDI+19, MPP20], and we give the first construction that supports quantum functionalities while maintaining classical communication.

Theorem 1.4. *(Informal) Assuming QLWE, there exists a NISC for BQP with classical receiver in the quantum random oracle model.*

This result continues a recent line of work that constructs maliciously-secure two-message protocols for quantum functionalities, which we survey in Appendix A.

1.2 Technical overview

Background. Our starting point is two works of Mahadev [Mah18a, Mah18b] on classical delegation of quantum computation. Taken together, they show that a classical client can delegate a BQP computation to a quantum server while maintaining both privacy of the client's input and integrity of the computation performed. Indeed, [Mah18b] shows how to obtain *soundness* via a construction of classical verification of quantum computation (CVQC), meaning that a (computationally bounded) cheating server won't be able to convince the classical client of a false outcome. Furthermore, [Mah18a] shows how to obtain *privacy* of the client's input via a construction of quantum fully-homomorphic encryption (QFHE) with classical keys. Executing CVQC under the hood of QFHE then provides both privacy and soundness, which was recently formalized by [CLLW20].

Our goal is to extend these results to the setting of fully-simulatable maliciously-secure computation, while also enabling *multiple* classical clients to outsource a quantum computation on their private inputs to a single quantum server. A natural idea would be to make use of post-quantum classical MPC to simulate the classical client in the above two-party client-server protocol. That is, n parties engage in classical MPC to set up a joint encryption $\mathsf{QFHE.Enc}(x_1, \ldots, x_n)$ of their inputs, and then they proceed to interact with the quantum server as a single entity.

Unfortunately, the resulting protocol suffers from an input-dependent abort issue, rendering it insecure. Consider a malicious server that colludes with any one of the classical clients P_1. Under QFHE, the server

[2] It was later shown how to remove the PKI via multi-key fully homomorphic encryption [MW16].

can decide whether to honestly complete the CVQC or force an abort *as a function of the clients' private inputs*. Then, P_1's output will signal which was the case, allowing the server (and P_1) to learn any arbitrary predicate of the honest clients' inputs. In fact, the possibility of causing an input-dependent abort also prevents the original protocol between single client and single server from satisfying the standard notion of simulatable *two*-party computation. In other words, executing CVQC under QFHE does not result in a *composable* blind CVQC protocol.

CVQC with malicious blindness. In order to prevent such input-dependent abort attacks, what we need is a CVQC protocol where the prover cannot learn anything about the verifier's input, even if is able to learn whether or not the verifier aborted (rejected its proof). We will refer to this property as *malicious blindness*.

As explained above, executing CVQC under QFHE does not result in malicious blindness. But what if we switch the nesting of CVQC and QFHE, using CVQC to prove that a QHFE evaluation $\mathsf{QFHE.Enc}(x) \rightarrow \mathsf{QFHE.Enc}(y)$ is performed honestly? It is not immediately clear how to do this, since [Mah18b]'s CVQC protocol only works for BQP computation, and QFHE evaluation is a *sampBQP* computation. Indeed, the "encrypted CNOT" operation at the heart of [Mah18a]'s QFHE involves obliviously sampling a classical FHE ciphertext by measuring a superposition over encryption random coins. Thus, performing QFHE evaluation of a pseudo-deterministic quantum functionality $y := D(x)$ will produce a *distribution* over ciphertexts $\mathsf{QFHE.Enc}(y; r)$ with the same outcome y but with varying random coins r.

However, Chung et al. [CLLW20] recently showed how to extend the protocols of [FHM18, Mah18b] to prove the correctness of sampBQP computations. The caveat is that the soundness error of their protocol is non-negligible, due to the following issue. Roughly, the prover prepares multiple copies of the history state of the computation, and the verifier chooses all but one of them to test and one of them to sample from. A malicious prover can always guess which state the verifier will sample from with inverse polynomial probability and cheat only in that copy of the history state, convincing the verifier to accept a completely invalid result. For general sampBQP problems, this issue appears somewhat inherent to their approach, since there is no meaningful way to combine multiple potentially invalid samples into a single valid sample.

We observe that for the special case of proving honest QFHE evaluation of some BQP computation, one *can* meaningfully combine multiple potentially invalid samples. Consider a verifier that requests multiple output ciphertexts, decrypts them all, and then outputs the most frequently occurring plaintext. If the prover can only cheat with some small (say 1/4) probability on each sample, then one should be able to drive the probability of accepting an invalid result down to negligible, with enough samples.

More abstractly, we consider any pseudo-deterministic circuit $D(\cdot)$ that can be written as $C(Q(\cdot))$, where Q is a quantum circuit and C is a classical circuit. On any classical input x, $C(Q(x))$ produces a well-defined output z (with overwhelming probability), while $Q(x)$ may produce any distribution over intermediate classical values y. The goal will be to obtain a protocol for delegating the computation of $C(Q(\cdot))$ where the prover's computation and the verifier's decision to accept or reject is independent of C.

We first use the one-round sampBQP verification protocol of [CLLW20] (with a quantum verifier that performs single-qubit computational and Hadamard basis measurements) to show how to verify such pseudo-deterministic computations with negligible soundness error (details in Section 4.2). The resulting protocol consists of a quantum proof and requires a quantum verifier to perform single-qubit meausurements. The next step is then to incorporate Mahadev's measurement protocol [Mah18b] in order to make the proof and the verifier fully classical.

The measurement protocol as described in [Mah18b] proceeds in a number of rounds, where in each round, the verifier issues a single bit challenge indicating either a "test" round or a "Hadamard" round. The test round is meant to check that the prover is behaving honestly (i.e. it is honestly "committing" to some particular quantum state), while the Hadamard round is meant to produce a sequence of classical measurement results that the verifier can use to produce its verdict. A single round consists of four classical messages between prover and verifier, and [Mah18b] shows that it satisfies the following property: Any prover that passes the test round with overwhelming probability will only be able to "cheat" in a Hadamard round with negligible probability, assuming QLWE.

In Section 4.3, we use this protocol to obtain a four message protocol for verifying circuits $D(\cdot) = C(Q(\cdot))$

that satisfies the following property. The verifier will either choose a test round, in which case they simply accept or reject, or they will choose a Hadamard round, in which case they either reject or obtain a purported sample $y \leftarrow Q(x)$. In the latter case, they compute and output $z := C(y)$. Any prover that passes the test round with overwhelming probability will only be able to force an incorrect output in a Hadamard round with negligible probability.

Parallel repetition. Now, we would like to obtain negligible soundness error, ideally while maintaining the four-message interaction. Recent works [ACGH20, CCY20] have shown how to do this in the setting where the cheating prover is attempting to convince the verifier to accept some false BQP statement. They show that if the four-message protocol is run sufficiently many times n in parallel, then the probability that the verifier accepts on *all* repetitions is negligible. Phrased differently, they show that, conditioned on the verifier accepting each of the (roughly) $n/2$ test rounds, the prover will not be able to successfully "cheat" on *all* the Hadamard rounds, except with negligible probability.

However, this is not quite enough for our setting. Recall that in the i'th Hadamard round, the verifier receives a purported sample $y_i \leftarrow Q(x)$ and computes $z_i := C(y_i)$. Crucially, we want the verifier to have already decided to accept by the time they invoke C to compute the outputs z_i. Thus, to combine these $\{z_i\}_i$ into a final output z, we will simply have the verifier output the most frequently occurring string z in the set (in particular, this final output computation cannot decide to accept/reject based on any properties of the set $\{z_i\}_i$). Now, a prover that successfully cheats in only *half* of the Hadamard rounds may be able to force an invalid output. Our goal is thus to show that this can only happen with negligible probability.

To do so, we take a closer look at the proof of the parallel repetition theorem from [ACGH20]. The following exposition will be simplified and not technically accurate, and is just meant to convey intuition. They consider any cheating prover state $|\psi\rangle$ right before the verifier's challenge $c \leftarrow \{0,1\}^n$ is chosen (n is the number of repetitions, and $c_i = 0$ corresponds to a test round while $c_i = 1$ corresponds to a Hadamard round). Then, for each possible $c \in \{0,1\}^n$, they consider a binary-valued projector Π_c that corresponds to running the prover's remaining strategy and then applying the verifier's verdict function on the resulting proof. Since c is chosen uniformly at random, it suffices to show that $\frac{1}{2^n} \langle\psi| \sum_{c \in \{0,1\}^n} \Pi_c |\psi\rangle$ is negligible. To do so, they square the quantity $\langle\psi| \sum_c \Pi_c |\psi\rangle$ and then show that each cross term $\langle\psi| \Pi_c \Pi_{c'} |\psi\rangle$ for $c \neq c'$ is negligible.

This is possible in their setting because the verifier only accepts if *every* test and Hadamard round accepts. This means that if Π_c and $\Pi_{c'}$ are both accepting for $c \neq c'$, there must be some index i where the prover is being accepted on both a test round and a Hadamard round (any i such that $c_i \neq c'_i$). This can be used to contradict the key property of the single repetition protocol described above, that any prover accepted on a test round with overwhelming probability can only cheat on a Hadamard round with negligible probability.

In our setting, we say that the verifier accepts if *every* test round accepts and *at least half* of the Hadamard rounds accept. Thus, it is no longer true that any noticeably large cross term $\langle\psi| \Pi_c \Pi_{c'} |\psi\rangle$ will imply a contradiction to the single repetition protocol. Indeed, if c and c' are somewhat close in Hamming distance, a prover could be rejected in a Hadamard round on all indices i such that $c_i \neq c'_i$ without causing the overall verifier to reject.

However, we observe that (i) it suffices to bound an overwhelming fraction of the cross terms (rather than all), and (ii) for any c, c' that have sufficiently large Hamming distance, if $\langle\psi| \Pi_c \Pi_{c'} |\psi\rangle$ is large, then there must be some index where the prover is simultaneously passing both the test and Hadamard rounds. Thus, we will need an overwhelming fraction of cross terms to correspond to pairs of challenge strings c, c' with Hamming distance at least as large as the number of Hadamard rounds. To facilitate this, we alter the protocol, setting the number of Hadamard rounds to some fixed λ, the total number of rounds to $n = \lambda^{1+\epsilon}$, and having the verifier sample a challenge string $c \in \{0,1\}^n$ with Hamming weight exactly λ. The full details and proof of this strengthened parallel repetition theorem can be found in Section 3.

Secure quantum computation. The above shows how to obtain negligible soundness for classical verification of quantum-classical circuits $C(Q(\cdot))$, where the verifier decides to accept or reject independently of

C. In the remainder of the paper, we show how to use this primitive to construct composable blind CVQC, and then present applications to secure quantum computation with classical communication.

In Section 4.4 we show that four-message composable blind CVQC follows by letting Q correspond to QFHE evaluation, and C correspond to QFHE decryption. Crucially, the QFHE decryption key is not needed to determine whether the verifier accepts or rejects, which results in the malicious blindness needed to ensure composability. Then, in Section 5.1, we use post-quantum MPC for classical (reactive) functionalities to allow multiple parties to simulate a single verifier participating in the CVQC protocol with the server. This results in a constant-round MPQC protocol in the common random string model from QLWE.

Next, we consider the three-round interaction pattern described in Section 1.1. To implement this, we will need a CVQC protocol with (i) two total messages, and (ii) distributed setup, where multiple parties can encrypt their respective inputs without any interaction. We construct this primitive in Section 4.5 via the following observations. Property (i) can be obtained in the quantum random oracle model (QROM) by appealing to Fiat-Shamir in the QROM [DFMS19, LZ19]. Property (ii) can be obtained via the use of quantum *multi-key* fully homomorphic encryption, which was recently constructed in [ABG$^+$21]. In fact, we will also require the CVQC protocol to satisfy various *perfect correctness* properties to obtain security against verifiers that make a malicious choice of random coins, and we defer discussion of this to the body.

Now, in Section 5.2, we show how to combine the two-message CVQC with distributed setup with a classical *multi-party reusable non-interactive secure computation* protocol (mrNISC) to obtain three-round MPQC in the public-key infrastructure (PKI) model. Maliciously-secure post-quantum mrNISC protocols have recently been constructed from QLWE [AJJM21, BJKL21]. The reason we need a PKI is the following. In the two-message CVQC protocol, the various inputs can be encrypted in a distributed fashion. However, the verifier also needs to set up some (input-independent) public parameters pp along with secret parameters sp kept private from the prover. This part cannot be fully distributed, so before the protocol begins, we will have each party publish a public key consisting of a mrNISC first-round message committing to a PRF key. This message is succinct (does not depend on the size of the functionality to be computed) and reusable across any varying subsets of parties (due to the reusability of the mrNISC protocol), and so it satisfies the requirements of the PKI model. These PRF keys can then be used to compute public parameters in the first round of the MPQC protocol, which are sent to the server along with the encryptions of each party's input.

Finally, in Section 5.3, we show that the two-message CVQC with distributed setup primitive can also be used to construct a *two-message* protocol between two parties: a quantum sender $\mathcal{S}$ and a classical receiver $\mathcal{R}$. Both parties have classical inputs, but can compute a (pseudo-deterministic) quantum functionality over these inputs. This follows by letting each of the sender and receiver independently encrypt their input, having the sender evaluate the CVQC prover, and then executing the CVQC verifier under a *classical* NISC (non-interactive secure computation) protocol. This results in a quantum NISC protocol with classical communication.

1.3 Discussion and open problems

Quantum sampling circuits. The above describes how to achieve standard malicious security for secure multi-party computation of pseudo-deterministic circuits. That is, any adversary will only have negligible advantage in distinguishing the real and simulated worlds. A natural next question is whether this is achievable for the more general class of polynomial-time quantum *sampling* problems, i.e. functionalities that output an arbitrary distribution over classical strings.

It is straightforward to see that the sampBQP protocol of [CLLW20] can be combined with QFHE and post-quantum classical MPC to give MPQC for quantum sampling problems with classical communication, but with some inverse polynomial security. That is, the adversary will be able to distinguish the real and simulated worlds with at most some inverse polynomial probability (but where the communication complexity of the protocol grows with this polynomial). This follows by using their sampBQP protocol to prove the correct computation of $\mathsf{QFHE.Enc}(y) \leftarrow \mathsf{QFHE.Eval}(Q, \mathsf{QFHE.Enc}(x))$, where Q is some quantum sampling circuit, and x is the joint private inputs of the clients.

Another potential approach to secure computation of sampling circuits would be to follow [GV19], which uses ideas from [BCM$^+$18] and [Mah18b] to construct a blind CVQC protocol in the measurement-based quantum computing framework (avoiding the reduction to local Hamiltonian). This framework appears to naturally support quantum sampling circuits, though its soundness for sampling circuits has not been analyzed. However, the inverse soundness error of their protocol also grows with the communication complexity. Thus, we leave MPQC for sampling circuits with classical communication and standard negligible security as an open question. Indeed, the more basic question that remains open is whether it is possible to construct a CVQC protocol for sampling circuits where the verifier only accepts samples that are distributed negligibly close to the real distribution induced by the quantum circuit.

Obfuscation of quantum circuits. As also discussed in [BM21], one approach to obtaining (heuristic) obfuscation of quantum circuits involves the notion of blind CVQC. Given a two-message composable blind CVQC with delayed functionality (meaning that the circuit D can be chosen by the prover after the verifier's message has been sampled) one could imagine obfuscating the (psuedo-deterministic) quantum circuit U as follows. Sample the verifier's first message on input U and output this message along with a classical obfuscation of the verification circuit (with the verification secret key hard-coded). Then, the evaluator with input x can attempt to produce a proof π for the circuit $D_x(\cdot)$ that takes U as input and outputs $U(x)$. It can then query the obfuscated verifier on π to learn $U(x)$. Note that malicious blindness is crucial for making this approach work, as the evaluator can clearly see whether or not the verifier accepts or rejects its proof.

Unfortunately, this is not the only property that is required. It is also crucial that the blind CVQC protocol is *reusably* sound, meaning that it securely implements an ideal functionality that allows the prover to repeatedly query the verifier on different circuits of its choice, learning the verifier's output each time. The composable blind CVQC constructed in this paper does not satisfy this reusable ideal functionality, and more discussion about the difficulties in obtaining reusable security can be found in [BM21]. Thus, the (one-time) composable blind CVQC protocol constructed in this paper can be seen a step towards heuristic obfuscation of quantum circuits, though the possibility of obtaining the crucial property of *reusability* remains open.

1.4 Other related work

Composable security. The notion of *composable blind and verifiable quantum computation* has been studied previously both in the setting of a quantum verifier and a classical verifier. In particular, [DFPR14] showed that the blind and verifiable protocols of [FK17] and [Mor14] with quantum verifier are composable. Next, [GV19] gave a construction of composable blind and verifiable delegation of quantum computation with a classical verifier, and noted that such a protocol should have implications to multi-party quantum computation (though they left this formalization to future work). However, their protocol only achieves inverse polynomial security and requires polynomially many rounds,[3] while ours achieves standard negligible security and has constant rounds.

We also remark that the works of [DFPR14] and [GV19] achieve composable security by combining the properties of standard blindness and "independent" verifiability, which are very similar to the two properties of malicious blindness and standard soundness that we use to enable composable security.

Two-party quantum computation. As discussed in Appendix A, many recent works (e.g. [CVZ20, ACGH20, Shm20, BCKM21a, MY21]) have considered maliciously-secure two-message two-party protocols for computing quantum functionalities.

In addition, there are a couple of recent works that study a relaxed variant of two-party secure computation, called *secure function evaluation* (SFE), in the quantum setting. This notion relaxes security from

[3] This appears to be somewhat inherent to their approach, as they follow the measurement-based computing paradigm, which requires the prover and verifier to interact for each sequential gate being computed.

simulation-based to indistinguishability-based, and in particular does not require correctness against malicious senders, meaning that constructions of quantum SFE do not require (classical) verification of quantum computation.

First, the work of [CCKM20] constructs SFE for quantum functionalities with classical communication, and even achieves *one-sided* simulation security (against malicious receiver). In fact, they show that if the sender has a quantum input, then full simulation-security is *impossible* to achieve if only classical communication is allowed. We evade this barrier by constructing protocols where all parties have *classical* input, showing that full simulation-based security is indeed possible in this setting. Next, the work of [CDM20] constructs SFE for quantum functionalities (where receiver may have a quantum input, and thus communication is quantum) from QLWE, and even show how to achieve rate-1 communication complexity.

Multi-party delegated quantum computation. Multi-party delegated quantum computation was first studied by [KP17], in the setting where multiple computationally weak but *quantum* clients would like to outsource some quantum computation to a powerful quantum server. This work only provides blindness (no correctness against a malicious server) and does not handle client-server collusions. Very recently, [KKMO21] showed how to achieve standard malicious security in this setting, though still with the requirement that the clients have quantum resources. The work of [Goy18] studies the notion of multi-party delegated quantum computation in three rounds, though security is only semi-honest, and again the clients require quantum resources. In summary, our work is the first to construct multi-party delegated quantum computation with entirely classical clients.

2 Preliminaries

Let λ denote the security parameter. A function $f : \mathbb{N} \to [0,1]$ is negligible if for every constant $c \in \mathbb{N}$ there exists $N \in \mathbb{N}$ such that for all $n > N$, $f(n) < n^{-c}$, and we write $\mathrm{negl}(\cdot)$ to denote such a function. Let $\mathcal{HW}_{n,m}$ denote the set of binary strings of length n with Hamming weight m. We will refer to pure quantum states with ket notation $|\psi\rangle$ and mixed quantum states with lowercase Greek letters such as ρ. Throughout, we will consider non-uniform quantum polynomial-time (QPT) adversaries, which are families of polynomial-size quantum circuits $\{\mathsf{Adv}_\lambda\}_{\lambda\in\mathbb{N}}$ along with some polynomial-size quantum advice $\{|\psi_\lambda\rangle\}_{\lambda\in\mathbb{N}}$, though we will often drop the indexing by λ when clear from context. We will also often refer to families of "psuedo-deterministic" quantum circuits, defined below. Again, we usually drop the indexing by λ when clear from context.

Definition 2.1 (Pseudo-Deterministic Quantum Circuit). *A family of psuedo-deterministic quantum circuits $\{D_\lambda\}_{\lambda\in\mathbb{N}}$ is defined as follows. The circuit D_λ takes as input a classical bit string $x \in \{0,1\}^{n(\lambda)}$ and outputs a single classical string $z \leftarrow D(x)$. The circuit is pseudo-deterministic if there exists a negligible function ν such that for every sequence of classical inputs $\{x_\lambda\}_{\lambda\in\mathbb{N}}$, there exists a sequence of outputs $\{z_\lambda\}_{\lambda\in\mathbb{N}}$ such that*

$$\Pr[D_\lambda(x_\lambda) = z_\lambda] = 1 - \nu(\lambda).$$

The following notation will also be useful.

Definition 2.2 ($M_{XZ}(\rho, h)$). *For a string $h \in \{0,1\}^n$ and an n-qubit quantum state ρ, consider the following procedure. For each $i \in [n]$, measure the i'th qubit of ρ in the standard basis if $h_i = 0$ and in the Hadamard basis if $h_i = 1$. Let the resulting random variable (over classical n-bit strings) be denoted by $M_{XZ}(\rho, h)$.*

2.1 Delegation of quantum computation

We will consider protocols $\Pi = (\mathcal{P}, \mathcal{V})$ for delegating the computation of psuedo-deterministic quantum circuits D. In such a protocol, $\mathcal{P}$ and $\mathcal{V}$ interact on input the security parameter 1^λ, and $\mathcal{V}$ has additional (possibly private) inputs D, x. After the interaction, $\mathcal{V}$ outputs (v, z), where $v \in \{\mathsf{acc}, \mathsf{rej}\}$ and $z \in \{0,1\}^*$. We denote this by $(v, z) \leftarrow (\mathcal{P}, \mathcal{V}(D, x))(1^\lambda)$. In general, $\mathcal{V}$ will satisfy some efficiency properties (e.g. it has limited or no quantum capabilities), making the protocol non-trivial.

Definition 2.3. *A protocol $\Pi = (\mathcal{V}, \mathcal{P})$ for delegating the computation of a pseudo-deterministic quantum circuit D should satisfy the following properties.*

- ***Completeness:*** *For any circuit D, input x, and output z such that $\Pr[D(x) = z] = 1 - \text{negl}(\lambda)$, it holds that*
$$\Pr[(\mathsf{acc}, z) \leftarrow (\mathcal{P}, \mathcal{V}(D, x))(1^\lambda)] = 1 - \text{negl}(\lambda).$$
- ***Soundness:*** *For any circuit D, input x, output z such that $\Pr[D(x) = z] = 1 - \text{negl}(\lambda)$, and cheating prover $\mathcal{P}^*$ with advice $|\psi\rangle$, it holds that*
$$\Pr[v = \mathsf{acc} \wedge z' \neq z : (v, z') \leftarrow (\mathcal{P}^*(|\psi\rangle), \mathcal{V}(D, x))(1^\lambda)] = \text{negl}(\lambda).$$
We say that soundness is statistical if $\mathcal{P}^*$ is unbounded, and that soundness is computational if $\mathcal{P}^*$ is a QPT machine with polynomial-size advice.

2.2 Quantum fully-homomorphic encryption

We define quantum fully-homomorphic encryption (QFHE) with classical keys and classical encryption of classical messages. One could also define encryption for quantum states and decryption for quantum ciphertexts, but we will not need that in this work.

Definition 2.4 (Quantum Homomorphic Encryption)**.** *A quantum fully-homomorphic encryption scheme* (QFHE.Gen, QFHE.Enc, QFHE.Eval, QFHE.Dec) *consists of the following efficient algorithms.*

- $\mathsf{QFHE.Gen}(1^\lambda) \to (\mathsf{pk}, \mathsf{sk})$*: On input the security parameter, the PPT key generation algorithm returns public/secret key pair* $(\mathsf{pk}, \mathsf{sk})$*.*
- $\mathsf{QFHE.Enc}(\mathsf{pk}, x) \to \mathsf{ct}$*: On input the public key* pk *and a classical plaintext x, the PPT encryption algorithm returns a classical ciphertext* ct*.*
- $\mathsf{QFHE.Eval}(Q, \mathsf{ct}) \to \widetilde{\mathsf{ct}}$*: On input a quantum circuit Q, and a ciphertext* ct*, the QPT evaluation algorithm returns an evaluated ciphertext* $\widetilde{\mathsf{ct}}$*.*
- $\mathsf{QFHE.Dec}(\mathsf{sk}, \mathsf{ct}) \to x$*: On input the secret key* sk *and a classical ciphertext* ct*, the decryption algorithm returns a message x.*

The scheme should satisfy the standard notion of semantic security. We will require the following notion of correctness for evaluation of pseudo-deterministic quantum circuits. Note that this evaluation correctness holds over *all* key generation and encryption random coins.

Definition 2.5 (Evaluation Correctness)**.** *A QFHE scheme* (QFHE.Gen, QFHE.Enc, QFHE.Eval, QFHE.Dec) *is correct if for every $\lambda \in \mathbb{N}$, every $(\mathsf{pk}, \mathsf{sk}) \in \mathsf{QFHE.Gen}(1^\lambda)$, every input x, every $\mathsf{ct} \in \mathsf{QFHE.Enc}(\mathsf{pk}, x)$, and every polynomial-size pseudo-deterministic quantum circuit Q and output y such that $\Pr[Q(x) = y] = 1 - \text{negl}(\lambda)$, it holds that*
$$\Pr[\mathsf{QFHE.Dec}(\mathsf{sk}, \mathsf{QFHE.Eval}(Q, \mathsf{ct})) = y] = 1 - \text{negl}(\lambda).$$

The works of Mahadev [Mah18a] and Brakerski [Bra18] show that such a QFHE scheme can be constructed from QLWE (we will not consider unlevelled QFHE in this work, which requires circular-security assumptions).

Multi-key. We will also make use of a quantum *multi-key* fully-homomorphic encryption scheme (QMFHE.Gen, QMFHE.KeyGen, QMFHE.Enc, QMFHE.Eval, QMFHE.Dec), which was constructed in [ABG+21]. In such a scheme, the evaluation algorithm QMFHE.Eval now may take as input some n-input circuit Q along with n ciphertexts $(\mathsf{ct}_1, \ldots, \mathsf{ct}_n)$, each encrypted under independently sampled public keys (assume that each ct_i contains a description of the public key pk_i it is encrypted under). Likewise, QMFHE.Dec can decrypt a ciphertext ct that is the result of evaluating ciphertexts encrypted under public keys $\mathsf{pk}_1, \ldots, \mathsf{pk}_n$, given the corresponding secret keys $\mathsf{sk}_1, \ldots, \mathsf{sk}_n$. We require the same evaluation correctness (Definition 2.5) to hold, except over n-input pseudo-deterministic functionalities Q. We have also added a QMFHE.Gen algorithm which samples a common random string crs, to be used by each KeyGen algorithm.

2.3 Multi-party quantum computation

Below we give a definition of maliciously-secure multi-party quantum computation for pseudo-deterministic quantum functionalities, following the standard real/ideal world paradigm for defining secure computation [Gol04]. We assume that parties have access to a (classical) broadcast channel, and we aim for security with unanimous abort.

Consider an n-party quantum functionality specified by a family of pseudo-deterministic quantum circuits $\mathcal{Q} = \{Q_\lambda\}_{\lambda \in \mathbb{N}}$ where Q_λ has classical input of size $m_1(\lambda) + \cdots + m_n(\lambda)$ and classical output of size $\ell_1(\lambda) + \cdots + \ell_n(\lambda)$. We will consider a QPT adversary $\mathsf{Adv} = \{\mathsf{Adv}_\lambda\}_{\lambda \in \mathbb{N}}$ that corrupts any subset $M \subset [n]$ of parties. Let $H \coloneqq [n] \setminus M$.

Let Π be an n-party protocol for computing $\mathcal{Q}$. Consider any collection $(x_1, \ldots, x_n, |\psi\rangle_{\mathsf{Adv},\mathcal{D}})$, where $x_1, \ldots, x_n$ are classical bitstrings and $|\psi\rangle_{\mathsf{Adv},\mathcal{D}}$ is a polynomial-size quantum state on two registers Adv and $\mathcal{D}$. Abusing notation, we let $|\psi\rangle_{\mathsf{Adv}}$ be the part of the state on register Adv and likewise for $|\psi\rangle_{\mathcal{D}}$. Now define quantum random variable $\mathsf{REAL}_{\Pi,\mathsf{Q}}(\mathsf{Adv}_\lambda, \{x_i\}_{i\in[n]}, |\psi\rangle_{\mathsf{Adv}})$ as follows. $\mathsf{Adv}_\lambda(\{x_i\}_{i\in M}, |\psi\rangle_{\mathsf{Adv}})$ interacts with honest party algorithms on inputs $\{x_i\}_{i\in H}$ participating in protocol Π, after which the honest parties output $\{y_i\}_{i\in H}$ and Adv outputs a final state $|\psi_{\mathsf{out}}\rangle$ (an arbitrary function computed on an arbitrary subset of the registers that comprise its view). The random variable $\mathsf{REAL}_{\Pi,\mathsf{Q}}(\mathsf{Adv}_\lambda, \{x_i\}_{i\in[n]}, |\psi\rangle_{\mathsf{Adv}})$ then consists of $\{y_i\}_{i\in H}$ along with $|\psi_{\mathsf{out}}\rangle$.

For any Adv, we require the existence of a simulator $\mathsf{Sim} = \{\mathsf{Sim}_\lambda\}_{\lambda\in\mathbb{N}}$ that takes as input $(\{x_i\}_{i\in M}, |\psi\rangle_{\mathsf{Adv}})$, has access to an ideal functionality $\mathcal{I}[\{x_i\}_{i\in H}](\cdot)$, and outputs a state $|\psi_{\mathsf{out}}\rangle$. The ideal functionality accepts an input $\{x_i\}_{i\in M}$, applies Q_λ to $(x_1, \ldots, x_n)$ to recover $(y_1, \ldots, y_n)$, and returns $\{y_i\}_{i\in M}$ to Sim_λ. Then, it waits for either an abort or ok message from Sim_λ. In the case of ok it includes $\{y_i\}_{i\in H}$ in its output and in the case of abort it includes $\{\perp\}_{i\in H}$. Now, we define the quantum random variable $\mathsf{IDEAL}_{\Pi,\mathsf{Q}}(\mathsf{Sim}_\lambda, \{x_i\}_{i\in[n]}, |\psi\rangle_{\mathsf{Adv}})$ to consist of the output of $\mathcal{I}[\{x_i\}_{i\in H}](\cdot)$ and the final state $|\psi_{\mathsf{out}}\rangle$ of $\mathsf{Sim}_\lambda^{\mathcal{I}[\{x_i\}_{i\in H}](\cdot)}(\{x_i\}_{i\in M}, |\psi\rangle_{\mathsf{Adv}})$.

Definition 2.6 (Secure Multi-Party Quantum Computation)**.** *A protocol Π securely computes Q if for all QPT $\mathsf{Adv} = \{\mathsf{Adv}_\lambda\}_{\lambda\in\mathbb{N}}$ corrupting subset of parties $M \subset [n]$, there exists a QPT $\mathsf{Sim} = \{\mathsf{Sim}_\lambda\}_{\lambda\in\mathbb{N}}$ such that for all $\{x_{1,\lambda}, \ldots, x_{n,\lambda}, |\psi_\lambda\rangle_{\mathsf{Adv},\mathcal{D}}\}_{\lambda\in\mathbb{N}}$ and all QPT $\mathcal{D} = \{\mathcal{D}_\lambda\}_{\lambda\in\mathbb{N}}$, there exists a negligible function $\nu(\cdot)$ such that*

$$\Big| \Pr\left[\mathcal{D}_\lambda\left(|\psi_\lambda\rangle_{\mathcal{D}}, \mathsf{REAL}_{\Pi,\mathsf{Q}}(\mathsf{Adv}_\lambda, \{x_{i,\lambda}\}_{i\in[n]}, |\psi_\lambda\rangle_{\mathsf{Adv}})\right) = 1\right]$$
$$- \Pr\left[\mathcal{D}_\lambda\left(|\psi_\lambda\rangle_{\mathcal{D}}, \mathsf{IDEAL}_{\Pi,\mathsf{Q}}(\mathsf{Sim}_\lambda, \{x_{i,\lambda}\}_{i\in[n]}, |\psi_\lambda\rangle_{\mathsf{Adv}})\right) = 1\right] \Big| \leq \nu(\lambda).$$

2.4 Classical non-interactive secure computation

In Section 5.3, we will make use of a particular type of classical two-party computation protocol, called non-interactive secure computation (NISC). A NISC protocol is a two-message protocol between sender $\mathcal{S}$ with input $x_{\mathcal{S}}$ and receiver $\mathcal{R}$ with input $x_{\mathcal{R}}$. The protocol consists of two total messages, and is defined by four algorithms $(\mathsf{NISC}_{\mathsf{Gen}}, \mathsf{NISC}_1, \mathsf{NISC}_2, \mathsf{NISC}_{\mathsf{out}})$. We require that the receiver's message can be computed independently of the functionality C to be computed. The syntax of these algorithms is as follows.

- $\mathsf{crs} \leftarrow \mathsf{NISC}_{\mathsf{Gen}}(1^\lambda)$. The gen algorithm generates the crs.
- $(m_{\mathcal{R}}, \mathsf{st}) \leftarrow \mathsf{NISC}_1(\mathsf{crs}, x_{\mathcal{R}})$. The first message algorithm takes as input crs and the receiver's input $x_{\mathcal{R}}$, and outputs the receiver's message $m_{\mathcal{R}}$ and private state st.
- $m_{\mathcal{S}} \leftarrow \mathsf{NISC}_2(\mathsf{crs}, C, m_{\mathcal{R}}, x_{\mathcal{S}})$. The second message algorithm takes as input crs, a circuit C, the receiver's message $m_{\mathcal{R}}$, and the sender's input $x_{\mathcal{S}}$, and outputs the sender's message $m_{\mathcal{S}}$.
- $y \leftarrow \mathsf{NISC}_{\mathsf{out}}(\mathsf{st}, C, m_{\mathcal{S}})$. The output algorithm takes as input the receiver's private state st, the circuit C, and the sender's message $m_{\mathcal{S}}$, and outputs the receiver's output y.

A NISC protocol should satisfy standard simulation-based security against arbitrarily malicious adversaries. We note that post-quantum NISC protocols are known from QLWE [IPS08, PVW08].

3 Generalizing the [ACGH20] Parallel Repetition Theorem

Consider the following outline for a four-message protocol between a classical verifier $\mathcal{V}$ and a quantum prover $\mathcal{P}$. For concreteness, one can think of the public key pk as encoding some computation that the verifier would like the prover to perform. However, for the purposes of this section, we will only need to consider a few generic properties of such a protocol.

- $\mathcal{V}(1^\lambda) \to (\mathsf{pk}, \mathsf{sk})$: the verifier, on input the security parameter, generates a public/secret key pair $(\mathsf{pk}, \mathsf{sk})$ and sends pk to $\mathcal{P}$.
- $\mathcal{P}(\mathsf{pk}) \to (y, |\mathsf{st}\rangle)$: the prover, on input the public key, generates a classical string y and a quantum private state $|\mathsf{st}\rangle$, and sends y to $\mathcal{V}$.
- $\mathcal{V}$ samples a random challenge bit $c \leftarrow \{0, 1\}$ and sends c to $\mathcal{P}$.
- $\mathcal{P}(|\mathsf{st}\rangle, c) \to \pi$: the prover, on input its quantum state and the verifier's challenge bit, generates a classical proof π and sends π to $\mathcal{V}$.
- $\mathcal{V}(\mathsf{pk}, \mathsf{sk}, y, c, \pi) \to b$: the verifier, on input the transcript and its secret key, outputs a bit b where $b = 1$ indicates that it accepts and $b = 0$ indicates that it rejects.

If the verifier sampled $c = 0$, we refer to the execution as a "test round", and if the verifier sampled $c = 1$, we refer to the execution as a "Hadamard round". Now consider a generic prover strategy $(|\psi_{\mathsf{init}}\rangle, U, V_0, V_1)$, where

- $|\psi_{\mathsf{init}}\rangle$ is some initial state on three registers $\mathsf{X}, \mathsf{Y}, \mathsf{Z}$,
- U is a unitary on registers $\mathsf{X}, \mathsf{Y}, \mathsf{Z}, \mathsf{K}$, classically controlled on K,
- and V_0, V_1 are unitaries on registers $\mathsf{X}, \mathsf{Y}, \mathsf{Z}, \mathsf{K}$, classically controlled on Y and K.

Given this generic strategy, the protocol proceeds as follows. The pair $(\mathsf{pk}, \mathsf{sk})$ is sampled by the verifier, and then U is applied to $|\psi_{\mathsf{init}}\rangle |\mathsf{pk}\rangle$ to produce private state $|\psi_{\mathsf{pk}}\rangle$. Then, c is sampled and sent to the prover, who applies V_c to $|\psi_{\mathsf{pk}}\rangle$, and then measures Y, X to produce y, π.[4] For any choice of $(\mathsf{pk}, \mathsf{sk}, c)$, let $\Pi_{\mathsf{pk},\mathsf{sk},c}$ be the binary-valued projector that, when applied to $V_c |\psi_{\mathsf{pk}}\rangle$, corresponds to measuring y, π and then applying the verifier's verdict function on $(\mathsf{pk}, \mathsf{sk}, y, c, \pi)$.

Now suppose that the following two conditions hold (these are used in the proof of Lemma 3.2). Whenever we take an expectation over $(\mathsf{pk}, \mathsf{sk})$, we mean over $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathcal{V}(1^\lambda)$.

1. $\Pi_{\mathsf{pk},\mathsf{sk},0}$ does not depend on sk (in the protocols we are interested in, it simply checks whether π is in some set of classical strings determined by pk and y).
2. For any efficient prover strategy $(|\psi_{\mathsf{init}}\rangle, U, V_0, V_1)$[5] such that $\mathbb{E}_{\mathsf{pk},\mathsf{sk}}[\langle\psi_{\mathsf{pk}}| V_0^\dagger \Pi_{\mathsf{pk},\mathsf{sk},0} V_0 |\psi_{\mathsf{pk}}\rangle] = 1 - \operatorname{negl}(\lambda)$, it holds that $\mathbb{E}_{\mathsf{pk},\mathsf{sk}}[\langle\psi_{\mathsf{pk}}| V_1^\dagger \Pi_{\mathsf{pk},\mathsf{sk},1} V_1 |\psi_{\mathsf{pk}}\rangle] = \operatorname{negl}(\lambda)$.

Consider now an (n, λ)-parallel repeated version of the protocol, with a challenge string d that is chosen uniformly at random from the set $\mathcal{HW}_{n,\lambda}$ of n-bit strings with Hamming weight exactly λ. That is, the protocol is repeated n times in parallel, except that a Hadamard round is performed in exactly λ of the n protocols. The following theorem shows that any efficient cheating prover in this multi-copy protocol cannot make the verifier accept all test rounds and more than half of the Hadamard rounds.

[4] Note that, since V_c must be classically controlled on Y, we can indeed consider Y to measured after applying V_c rather than before.

[5] By efficient, we mean that U, V_0, V_1 are unitaries that can be implemented in quantum polynomial-time, while $|\psi_{\mathsf{init}}\rangle$ is any (potentially inefficiently preparable) state on a polynomial number of registers.

Theorem 3.1. *Let $n(\lambda) \geq \lambda^{1+\epsilon}$ for some constant $\epsilon > 0$. Then in the (n, λ)-parallel repeated protocol, the probability that the verifier accepts every test round (i such that $d_i = 0$) and $\geq \lambda/2$ of the Hadamard rounds (i such that $d_i = 1$) is $\mathrm{negl}(\lambda)$.*[6]

Proof. We can write any prover strategy in the multi-copy protocol as $(|\psi_{\mathsf{init}}\rangle, U, \{V_d\}_{d\in\mathcal{HW}_{n,\lambda}})$. For any $d \in \mathcal{HW}_{n,\lambda}$, let $\Pi_{\mathsf{pk},\mathsf{sk},d}$ be the projector that applies the single-copy verifier's projector $\Pi_{\mathsf{pk},\mathsf{sk},d_i}$ to each corresponding part of the prover's final state, and then accepts if all the $d_i = 0$ positions accept and if $\geq \lambda/2$ of the $d_i = 1$ positions accept. Then define $\Pi^{V_d}_{\mathsf{pk},\mathsf{sk},d} \coloneqq V_d^\dagger \Pi_{\mathsf{pk},\mathsf{sk},d} V_d$.

Now we state the following lemma, which is proven below.

Lemma 3.2. *For any $d_1, d_2 \in \mathcal{HW}_{n,\lambda}$ with Hamming distance at least $\lambda + 1$, it holds that*

$$\mathop{\mathbb{E}}_{\mathsf{pk},\mathsf{sk}}\left[\langle\psi_{\mathsf{pk}}|\,\Pi^{V_{d_2}}_{\mathsf{pk},\mathsf{sk},d_2}\Pi^{V_{d_1}}_{\mathsf{pk},\mathsf{sk},d_1} + \Pi^{V_{d_1}}_{\mathsf{pk},\mathsf{sk},d_1}\Pi^{V_{d_2}}_{\mathsf{pk},\mathsf{sk},d_2}\,|\psi_{\mathsf{pk}}\rangle\right] = \mathrm{negl}(\lambda).$$

Given this lemma we can complete the proof of Theorem 3.1. Let D_{far} be the set of pairs of strings $\{d_1, d_2\} \in (\mathcal{HW}_{n,\lambda})^2$ with Hamming distance at least $\lambda + 1$, and let D_{close} be the set of pairs of strings $\{d_1, d_2\} \in (\mathcal{HW}_{n,\lambda})^2$ with Hamming distance at most λ. The success probability of the prover can be written as follows. Recall that $|\psi_{\mathsf{pk}}\rangle$ is defined to be the prover's state that results from applying the unitary U to its initial state $|\psi_{\mathsf{init}}\rangle$ and the public key pk sampled by the verifier.

$$\begin{aligned}
&\frac{1}{\binom{n}{\lambda}}\mathop{\mathbb{E}}_{\mathsf{pk},\mathsf{sk}}\left[\langle\psi_{\mathsf{pk}}|\sum_{d\in\mathcal{HW}_{n,\lambda}}\Pi^{V_d}_{\mathsf{pk},\mathsf{sk},d}\,|\psi_{\mathsf{pk}}\rangle\right]\\
\leq&\frac{1}{\binom{n}{\lambda}}\mathop{\mathbb{E}}_{\mathsf{pk},\mathsf{sk}}\left[\left(\langle\psi_{\mathsf{pk}}|\left(\sum_{d\in\mathcal{HW}_{n,\lambda}}\Pi^{V_d}_{\mathsf{pk},\mathsf{sk},d}\right)^2|\psi_{\mathsf{pk}}\rangle\right)^{1/2}\right]\\
\leq&\frac{1}{\binom{n}{\lambda}}\mathop{\mathbb{E}}_{\mathsf{pk},\mathsf{sk}}\left[\left(\sum_{d\in\mathcal{HW}_{n,\lambda}}\langle\psi_{\mathsf{pk}}|\,\Pi^{V_d}_{\mathsf{pk},\mathsf{sk},d}\,|\psi_{\mathsf{pk}}\rangle\right)^{1/2}\right]\\
&+\frac{1}{\binom{n}{\lambda}}\mathop{\mathbb{E}}_{\mathsf{pk},\mathsf{sk}}\left[\left(\sum_{\{d_1,d_2\}\in(\mathcal{HW}_{n,\lambda})^2}\langle\psi_{\mathsf{pk}}|\,\Pi^{V_{d_2}}_{\mathsf{pk},\mathsf{sk},d_2}\Pi^{V_{d_1}}_{\mathsf{pk},\mathsf{sk},d_1} + \Pi^{V_{d_1}}_{\mathsf{pk},\mathsf{sk},d_1}\Pi^{V_{d_2}}_{\mathsf{pk},\mathsf{sk},d_2}\,|\psi_{\mathsf{pk}}\rangle\right)^{1/2}\right]\\
\leq&\frac{1}{\binom{n}{\lambda}^{1/2}}+\frac{1}{\binom{n}{\lambda}}\left(\sum_{\{d_1,d_2\}\in(\mathcal{HW}_{n,\lambda})^2}\mathop{\mathbb{E}}_{\mathsf{pk},\mathsf{sk}}\left[\langle\psi_{\mathsf{pk}}|\,\Pi^{V_{d_2}}_{\mathsf{pk},\mathsf{sk},d_2}\Pi^{V_{d_1}}_{\mathsf{pk},\mathsf{sk},d_1} + \Pi^{V_{d_1}}_{\mathsf{pk},\mathsf{sk},d_1}\Pi^{V_{d_2}}_{\mathsf{pk},\mathsf{sk},d_2}\,|\psi_{\mathsf{pk}}\rangle\right]\right)^{1/2}\\
\leq&\mathrm{negl}(\lambda)+\frac{1}{\binom{n}{\lambda}}\left(\sum_{\{d_1,d_2\}\in D_{\mathsf{far}}}\mathrm{negl}(\lambda)\right)^{1/2}\\
&+\frac{1}{\binom{n}{\lambda}}\left(\sum_{\{d_1,d_2\}\in D_{\mathsf{close}}}\mathop{\mathbb{E}}_{\mathsf{pk},\mathsf{sk}}\left[\langle\psi_{\mathsf{pk}}|\,\Pi^{V_{d_2}}_{\mathsf{pk},\mathsf{sk},d_2}\Pi^{V_{d_1}}_{\mathsf{pk},\mathsf{sk},d_1} + \Pi^{V_{d_1}}_{\mathsf{pk},\mathsf{sk},d_1}\Pi^{V_{d_2}}_{\mathsf{pk},\mathsf{sk},d_2}\,|\psi_{\mathsf{pk}}\rangle\right]\right)^{1/2}\\
\leq&\mathrm{negl}(\lambda)+\frac{1}{\binom{n}{\lambda}}\left(2|D_{\mathsf{close}}|\right)^{1/2},
\end{aligned}$$

[6] In fact, it is straightforward to adjust the proof to show that for *any* constant c, the verifier will only accept all test rounds and at least λ/c Hadamard rounds with $\mathrm{negl}(\lambda)$ probability.

where the first inequality follows because $|\psi_{\mathsf{pk}}\rangle\langle\psi_{\mathsf{pk}}| \preceq \mathbb{I}$, the second inequality follows because each projector is idempotent, the third inequality uses Jensen's inequality, and the fourth inequality follows from Lemma 3.2. Now we bound the size of D_{close}. Observe that for each $d \in \mathcal{HW}_{n,\lambda}$, one can count the number of $d' \in \mathcal{HW}_{n,\lambda}$ at Hamming distance at most λ from d by flipping any i of the 0 positions in d to 1 and any i of the 1 positions in d to 0, for each $i \in [\lambda/2]$. This means that

$$|D_{\mathsf{close}}| < \binom{n}{\lambda}\sum_{i=1}^{\lambda/2}\binom{n-\lambda}{i}\binom{\lambda}{i}.$$

Thus, the above expression can be bounded by

$$\mathrm{negl}(\lambda) + \left(\frac{2\sum_{i=1}^{\lambda/2}\binom{n-\lambda}{i}\binom{\lambda}{i}}{\binom{n}{\lambda}}\right)^{1/2} \leq \mathrm{negl}(\lambda) + \left(\frac{\lambda 2^{\lambda}\binom{n-\lambda}{\lambda/2}}{\binom{n}{\lambda}}\right)^{1/2} \leq \mathrm{negl}(\lambda) + \left(\lambda 2^{\lambda}\left(\frac{\lambda}{n}\right)^{\lambda/2}\right)^{1/2} = \mathrm{negl}(\lambda),$$

for $n \geq \lambda^{1+\epsilon}$.

□

Now we prove Lemma 3.2.

Proof. (of Lemma 3.2) This proof follows the outline of [ACGH20, Lemma 4.2], with adjustments to handle the fact that the multi-copy verifier accepts if some large enough subset of the Hadamard round positions accept (rather than all). Thus the lemma is only true for two projectors that correspond to challenges with large Hamming distance.

First, since for any quantum state $|\psi\rangle$ and two projectors Π_1, Π_2,

$$\langle\psi|\Pi_2\Pi_1 + \Pi_1\Pi_2|\psi\rangle \leq 2|\langle\psi|\Pi_2\Pi_1|\psi\rangle| \leq 2\langle\psi|\Pi_2\Pi_1\Pi_2|\psi\rangle^{1/2},$$

it holds that (see also the end of the proof of [ACGH20, Lemma 4.2])

$$\mathop{\mathbb{E}}_{\mathsf{pk},\mathsf{sk}}\left[\langle\psi_{\mathsf{pk}}|\Pi^{V_{d_2}}_{\mathsf{pk},\mathsf{sk},d_2}\Pi^{V_{d_1}}_{\mathsf{pk},\mathsf{sk},d_1} + \Pi^{V_{d_1}}_{\mathsf{pk},\mathsf{sk},d_1}\Pi^{V_{d_2}}_{\mathsf{pk},\mathsf{sk},d_2}|\psi_{\mathsf{pk}}\rangle\right] \leq 2\mathop{\mathbb{E}}_{\mathsf{pk},\mathsf{sk}}\left[\langle\psi_{\mathsf{pk}}|\Pi^{V_{d_2}}_{\mathsf{pk},\mathsf{sk},d_2}\Pi^{V_{d_1}}_{\mathsf{pk},\mathsf{sk},d_1}\Pi^{V_{d_2}}_{\mathsf{pk},\mathsf{sk},d_2}|\psi_{\mathsf{pk}}\rangle\right]^{1/2}.$$

So, it suffices to show that

$$\mathop{\mathbb{E}}_{\mathsf{pk},\mathsf{sk}}\left[\langle\psi_{\mathsf{pk}}|\Pi^{V_{d_2}}_{\mathsf{pk},\mathsf{sk},d_2}\Pi^{V_{d_1}}_{\mathsf{pk},\mathsf{sk},d_1}\Pi^{V_{d_2}}_{\mathsf{pk},\mathsf{sk},d_2}|\psi_{\mathsf{pk}}\rangle\right] = \mathrm{negl}(\lambda).$$

Suppose there exists a polynomial $p(\cdot)$ such that

$$\mathop{\mathbb{E}}_{\mathsf{pk},\mathsf{sk}}\left[\langle\psi_{\mathsf{pk}}|\Pi^{V_{d_2}}_{\mathsf{pk},\mathsf{sk},d_2}\Pi^{V_{d_1}}_{\mathsf{pk},\mathsf{sk},d_1}\Pi^{V_{d_2}}_{\mathsf{pk},\mathsf{sk},d_2}|\psi_{\mathsf{pk}}\rangle\right] \geq 1/p(\lambda).$$

Assume for simplicity that λ is even. Note that since d_1 and d_2 each have Hamming weight λ and are at Hamming distance at least $\lambda+1$ from each other, there exists at most $\lambda/2-1$ indices i such that $d_{1,i} = d_{2,i} = 1$. Let S be the set of at least $\lambda/2+1$ indices i such that $d_{1,i} = 1$ and $d_{2,i} = 0$. Define

$$\Pi^{(i)}_{\mathsf{pk},\mathsf{sk}} := \mathbb{I} \otimes \cdots \otimes \Pi_{\mathsf{pk},\mathsf{sk},1} \otimes \cdots \otimes \mathbb{I},$$

where the $\Pi_{\mathsf{pk},\mathsf{sk},1}$ is applied on the i'th part of the state. This corresponds to the verifier's projector on the i'th single-copy state in the case of a Hadamard round. Now $\Pi^{V_{d_1}}_{\mathsf{pk},\mathsf{sk},d_1}$ can be expressed as applying V_{d_1}, checking that at least $\lambda/2+1$ of the $\Pi^{(i)}_{\mathsf{pk},\mathsf{sk}}$ projectors accept for i such that $d_1 = 1$ (as well as checking that all test rounds accept), and applying $V^{\dagger}_{d_1}$. Since $|S| \geq \lambda/2+1$, it follows that

$$
\begin{aligned}
&\mathop{\mathbb{E}}_{\mathsf{pk},\mathsf{sk}}\left[\langle\psi_{\mathsf{pk}}|\,\Pi^{V_{d_2}}_{\mathsf{pk},\mathsf{sk},d_2}\Pi^{V_{d_1}}_{\mathsf{pk},\mathsf{sk},d_1}\Pi^{V_{d_2}}_{\mathsf{pk},\mathsf{sk},d_2}\,|\psi_{\mathsf{pk}}\rangle\right]\\
&\leq \mathop{\mathbb{E}}_{\mathsf{pk},\mathsf{sk}}\left[\langle\psi_{\mathsf{pk}}|\,\Pi^{V_{d_2}}_{\mathsf{pk},\mathsf{sk},d_2}V_{d_1}^{\dagger}\left(\mathbb{I}-\prod_{i\in S}\left(\mathbb{I}-\Pi^{(i)}_{\mathsf{pk},\mathsf{sk}}\right)\right)V_{d_1}\Pi^{V_{d_2}}_{\mathsf{pk},\mathsf{sk},d_2}\,|\psi_{\mathsf{pk}}\rangle\right]\\
&\leq \sum_{i\in S}\mathop{\mathbb{E}}_{\mathsf{pk},\mathsf{sk}}\left[\langle\psi_{\mathsf{pk}}|\,\Pi^{V_{d_2}}_{\mathsf{pk},\mathsf{sk},d_2}V_{d_1}^{\dagger}\,\Pi^{(i)}_{\mathsf{pk},\mathsf{sk}}V_{d_1}\Pi^{V_{d_2}}_{\mathsf{pk},\mathsf{sk},d_2}\,|\psi_{\mathsf{pk}}\rangle\right],
\end{aligned}
$$

where the second inequality is a union bound, using the fact that the $\Pi^{(i)}_{\mathsf{pk},\mathsf{sk}}$ commute. Thus, there must exist an $i \in S$ and a polynomial $q(\cdot)$ such that

$$
\mathop{\mathbb{E}}_{\mathsf{pk},\mathsf{sk}}\left[\langle\psi_{\mathsf{pk}}|\,\Pi^{V_{d_2}}_{\mathsf{pk},\mathsf{sk},d_2}V_{d_1}^{\dagger}\,\Pi^{(i)}_{\mathsf{pk},\mathsf{sk}}V_{d_1}\Pi^{V_{d_2}}_{\mathsf{pk},\mathsf{sk},d_2}\,|\psi_{\mathsf{pk}}\rangle\right] \geq 1/q(\lambda).
$$

At this point, we have essentially established that the prover must be passing *both* the test round and the Hadamard round on the i'th index with noticeable probability. To complete the proof, we show that such a prover can be used to contradict condition 2 on the CVQC protocol. This follows the proof given in [ACGH20], so we defer the remainder of this proof to Appendix B.

□

4 Composable Blind CVQC

In this section, we show how to construct (constant-round) composable blind classical verification of quantum computation. The main property we need to achieve composability (other than standard blindness and soundness) is that the prover cannot obtain information about the verifier's input even if it gets to observe whether or not the verifier accepted its proof.

In fact, we will first construct a more general notion that we call *CVQC for quantum-classical circuits*, which allows for delegation of pseudo-deterministic quantum-classical circuits $C(Q(\cdot))$, where the verifier's decision to accept or reject does not depend on the description of C. In Section 4.1, we give a formal definition of this primitive. In Section 4.2, we use [CLLW20]'s sampBQP protocol to show how to delegate quantum-classical circuits with a quantum verifier that only performs single qubit measurements. In Section 4.3, we compile the protocol with [Mah18b]'s measurement protocol to make the verifier classical, and then apply the parallel repetition theorem from last section to obtain negligible soundness error, satisfying our definition in Section 4.1. In Section 4.4, we formally define the notion of composable blind CVQC, and show how to construct it from CVQC for quantum-classical circuits plus quantum fully-homomorphic encryption. Finally, in Section 4.5, we give an alternative CVQC protocol that satisfies various extra properties that will be useful later for obtaining round-optimized secure computation protocols.

4.1 CVQC for quantum-classical circuits

Below we define CVQC for quantum-classical circuits. We crucially split the final verification into two parts, requiring that the classical part of the circuit is not needed to determine whether or not the verifier will accept the prover's proof.

We require the protocol to satisfy the standard completeness and soundness properties, along with *semi-malicious correctness*, which will be useful when building secure computation from this primitive. This property guarantees that even if the verifier's initial keys are computed honestly, but with a malicious choice of random coins, the output of the honest verification procedure applied to an honest proof must still be correct (if it doesn't abort). Below we present the two-message syntax, but such a definition easily extends to more general interactive protocols.

Definition 4.1 (CVQC for Quantum-Classical Circuits). *A two-message CVQC protocol for a quantum-classical pseudo-deterministic circuit $D(\cdot) = C(Q(\cdot))$ has the following syntax.*

- $\mathcal{V}_{\mathsf{QC}}^{\mathsf{setup}}(1^\lambda, Q, x) \to (\mathsf{pk}, \mathsf{sk})$*: the verifier takes the security parameter 1^λ, a quantum circuit Q, and a classical input x, and outputs a public key secret key pair $(\mathsf{pk}, \mathsf{sk})$.*
- $\mathcal{P}_{\mathsf{QC}}(\mathsf{pk}, Q, x) \to \pi$*: the prover takes the public key pk, a quantum circuit Q, and classical input x, and outputs a classical proof π.*
- $\mathcal{V}_{\mathsf{QC}}(Q, C, x, \mathsf{pk}, \mathsf{sk}, \pi) \to (v, z)$*: the final verification circuit is split up into the following two parts.*
  - $\mathcal{V}_{\mathsf{QC}}^{\mathsf{vrfy}}(Q, x, \mathsf{pk}, \mathsf{sk}, \pi) \to (v, \widetilde{z})$*: the first part of the final verification circuit takes the quantum circuit Q, the input x, public key secret key pair $(\mathsf{pk}, \mathsf{sk})$, and proof π, and outputs $(v, \widetilde{z})$, where $v \in \{\mathsf{acc}, \mathsf{rej}\}$, and $\widetilde{z} \in \{0,1\}^*$. If $v = \mathsf{acc}$, then the second part of the verification is invoked on $\widetilde{z}$ to produce z, and the final output is (acc, z). Otherwise, the final output is $(\mathsf{rej}, \bot)$.*
  - $\mathcal{V}_{\mathsf{QC}}^{\mathsf{out}}(C, \widetilde{z}) \to z$*: the second part of the verification takes as input a classical circuit C and a string $\widetilde{z}$, and outputs a string z.*

Crucially, both the prover and the first part of the final verification circuit do not depend on the classical circuit C. This protocol should satisfy the following standard completeness and soundness properties.

- ***Completeness:*** *For any quantum-classical circuit $D(\cdot) = C(Q(\cdot))$, input x, and output z such that $\Pr[D(x) = z] = 1 - \mathrm{negl}(\lambda)$, it holds that*

$$\Pr\left[v = \mathsf{acc} \wedge z' = z : \begin{array}{r} (\mathsf{pk}, \mathsf{sk}) \leftarrow \mathcal{V}_{\mathsf{QC}}^{\mathsf{setup}}(1^\lambda, Q, x) \\ \pi \leftarrow \mathcal{P}_{\mathsf{QC}}(\mathsf{pk}, Q, x) \\ (v, z') \leftarrow \mathcal{V}_{\mathsf{QC}}(Q, C, x, \mathsf{pk}, \mathsf{sk}, \pi) \end{array}\right] = 1 - \mathrm{negl}(\lambda).$$

- ***Soundness:*** *For any quantum-classical circuit $D(\cdot) = C(Q(\cdot))$, input x, output z such that $\Pr[D(x) = z] = 1 - \mathrm{negl}(\lambda)$, and cheating prover $\mathcal{P}_{\mathsf{QC}}^*$ with advice $|\psi\rangle$, it holds that*

$$\Pr\left[v = \mathsf{acc} \wedge z' \neq z : \begin{array}{r} (\mathsf{pk}, \mathsf{sk}) \leftarrow \mathcal{V}_{\mathsf{QC}}^{\mathsf{setup}}(1^\lambda, Q, x) \\ \pi \leftarrow \mathcal{P}_{\mathsf{QC}}^*(|\psi\rangle, \mathsf{pk}) \\ (v, z') \leftarrow \mathcal{V}_{\mathsf{QC}}(Q, C, x, \mathsf{pk}, \mathsf{sk}, \pi) \end{array}\right] = \mathrm{negl}(\lambda).$$

In addition, we may want the protocol to satisfy the following property.

- ***Semi-malicious correctness:*** *For any quantum-classical circuit $D(\cdot) = C(Q(\cdot))$, input x, output z such that $\Pr[D(x) = z] = 1 - \mathrm{negl}(\lambda)$, and $(\mathsf{pk}, \mathsf{sk}) \in \mathcal{V}_{\mathsf{QC}}^{\mathsf{setup}}(1^\lambda, Q, x)$, it holds that*

$$\Pr\left[v = \mathsf{acc} \wedge z' \neq z : \begin{array}{r} \pi \leftarrow \mathcal{P}_{\mathsf{QC}}(\mathsf{pk}, Q, x) \\ (v, z') \leftarrow \mathcal{V}_{\mathsf{QC}}(Q, C, x, \mathsf{pk}, \mathsf{sk}, \pi) \end{array}\right] = \mathrm{negl}(\lambda).$$

4.2 Delegation of quantum-classical circuits with quantum verifier

Quantum sampling protocol. Recently, [CLLW20] constructed a one-message protocol $\Pi_{\mathsf{Samp}} = (\mathcal{P}_{\mathsf{Samp}}, \mathcal{V}_{\mathsf{Samp}})$ for delegating the computation of a quantum circuit Q on classical input x. The soundness of their protocol stipulates that if the verifier accepts an output z, then z is ϵ-close to a sample from the classical distribution induced by running Q on x and then measuring the output.

- $\mathcal{P}_{\mathsf{Samp}}(1^\lambda, 1^{1/\epsilon}, Q, x) \to |\psi\rangle$: the prover takes the security parameter 1^λ, the accuracy parameter $1^{1/\epsilon}$, a quantum circuit Q, and a classical input x, and samples a quantum proof $|\psi\rangle$.

- $\mathcal{V}_{\mathsf{Samp}}(1^\lambda, 1^{1/\epsilon}, Q, x, |\psi\rangle) \to (v, z)$: the verifier samples a binary string h where each position indicates a computational basis or Hadamard basis measurement, samples $e \leftarrow M_{XZ}(|\psi\rangle, h)$ (see Definition 2.2), and applies a classical circuit $\mathcal{V}_{\mathsf{out}}(Q, x, h, e)$ to obtain output (v, z) where $v \in \{\mathsf{acc}, \mathsf{rej}\}$ and $z \in \{0,1\}^*$.

[CLLW20] show that the protocol satisfies the following.

- **Completeness:** For any $\epsilon(\lambda) = 1/\mathrm{poly}(\lambda)$ and quantum circuit Q with input x,

$$\Pr[(\mathsf{rej}, \bot) \leftarrow \mathcal{V}_{\mathsf{Samp}}(1^\lambda, 1^{1/\epsilon}, Q, x, |\psi\rangle) : |\psi\rangle \leftarrow \mathcal{P}_{\mathsf{Samp}}(1^\lambda, 1^{1/\epsilon}, Q, x)] = \mathrm{negl}(\lambda).$$

- **Soundness:** For any circuit Q, input x, cheating prover $\mathcal{P}^*_{\mathsf{Samp}}$, $\epsilon(\lambda) = 1/\mathrm{poly}(\lambda)$, and sufficiently large $\lambda \in \mathbb{N}$,
  - let $(v, z) \leftarrow \mathcal{V}_{\mathsf{Samp}}(1^\lambda, 1^{1/\epsilon}, Q, x, \mathcal{P}^*_{\mathsf{Samp}}(1^\lambda, 1^{1/\epsilon}, Q, x))$,
  - and define $z_{\mathsf{ideal}} = \bot$ if $v = \mathsf{rej}$ and $z_{\mathsf{ideal}} \leftarrow Q(x)$ if $v = \mathsf{acc}$.

  Then it holds that $\|(v, z) - (v, z_{\mathsf{ideal}})\|_1 \leq \epsilon(\lambda)$.

Quantum-classical circuits. Now consider any pseudo-deterministic quantum circuit D that can be split into two parts Q, C, where Q is quantum and C is classical. That is, $D(x) = C(Q(x))$, where $Q(x)$ outputs a classical string $\widehat{z}$, and then $C(\widehat{z})$ outputs z. Since D is pseudo-deterministic, each input x results in some fixed z with overwhelming probability. However, x still may induce an arbitrary distribution over intermediate values $\widehat{z}$.

We now show that parallel repetition of Π_{Samp} gives a one-message protocol for delegating computation of $D(x)$ with *negligible* soundness, and where the prover's algorithm depends *only* on Q. In particular, consider the following protocol $\Pi_{\mathsf{QVer}} = (\mathcal{P}_{\mathsf{QVer}}, \mathcal{V}_{\mathsf{QVer}})$ for delegating the computation of $D(\cdot) = C(Q(\cdot))$.

- $\mathcal{P}_{\mathsf{QVer}}(1^\lambda, Q, x) \to |\psi\rangle$: the prover obtains a description of Q and an input x, sets $\epsilon = 1/4$, and runs λ copies of $\mathcal{P}_{\mathsf{Samp}}(1^\lambda, 1^{1/\epsilon}, Q, x)$ to produce a proof $|\psi\rangle := (|\psi_1\rangle, \dots, |\psi_\lambda\rangle)$.
- $\mathcal{V}_{\mathsf{QVer}}(Q, C, x, |\psi\rangle) \to (v, z)$: we split this verifier into three parts $(\mathcal{V}^{\mathsf{meas}}_{\mathsf{QC}}, \mathcal{V}^{\mathsf{test}}_{\mathsf{QC}}, \mathcal{V}^{\mathsf{out}}_{\mathsf{QC}})$.
  - $\mathcal{V}^{\mathsf{meas}}_{\mathsf{QVer}}(Q, x, |\psi\rangle)$ samples h_i according to the distribution defined by $\mathcal{V}_{\mathsf{Samp}}$ and obtains $e_i \leftarrow M_{X,Z}(|\psi_i\rangle, h_i)$ for each $i \in [\lambda]$. Set $h := (h_1, \dots, h_\lambda)$ and $e := (e_1, \dots, e_\lambda)$.
  - $\mathcal{V}^{\mathsf{test}}_{\mathsf{QVer}}(Q, x, h, e)$ applies $\mathcal{V}_{\mathsf{out}}(Q, x, h_i, e_i)$ for each $i \in \lambda$ to obtain $(v_i, \widehat{z}_i)$. If any $v_i = \mathsf{rej}$, then output $(\mathsf{rej}, \bot)$ (and do not proceed to $\mathcal{V}^{\mathsf{out}}_{\mathsf{QVer}}$), and otherwise set $\widehat{z} := (\widehat{z}_1, \dots, \widehat{z}_\lambda)$ and continue.
  - $\mathcal{V}^{\mathsf{out}}_{\mathsf{QVer}}(C, \widehat{z})$ computes $z_i := C(\widehat{z}_i)$ for each $i \in [\lambda]$, determines the most frequently occuring z in $\{z_i\}_{i \in [\lambda]}$, and outputs (acc, z).

Lemma 4.2. Π_{QVer} *satisfies completeness and statistical soundness as defined in Definition 2.3 for delegating the computation of a pseudo-deterministic circuit* $D(\cdot) = C(Q(\cdot))$.

Proof. Fix an x and let z be such that $C(Q(x)) = z$ with probability $1 - \mathrm{negl}(\lambda)$. Consider the projector P that corresponds to running the single-copy verifier $\mathcal{V}_{\mathsf{Samp}}$ on some part $|\psi_i\rangle$ of the prover's proof and accepting if the output is (acc, z') for some $z' \neq z$. By the soundness of Π_{Samp}, this projector will accept with probability at most $1/4 + \mathrm{negl}(\lambda)$ on *any* prover state $|\psi_i\rangle$.

Note that soundness of Π_{QVer} is violated only if the following procedure accepts. Partition the prover's proof $|\psi\rangle$ into λ registers $|\psi_1\rangle, \dots, |\psi_\lambda\rangle$, apply P to each, and accept if at least $\lambda/2$ of the projectors accept. Even though $|\psi_1\rangle, \dots, |\psi_\lambda\rangle$ may be entangled, it still holds that P accepts on each $|\psi_i\rangle$ individually with probability at most $1/4 + \mathrm{negl}(\lambda)$, since the registers are disjoint. That is, conditioned on any sequence of previous results of measuring $|\psi_1\rangle, \dots, |\psi_i\rangle$, applying P to $|\psi_{i+1}\rangle$ will accept with probability at most $1/4 + \mathrm{negl}(\lambda)$. Thus, the distribution on number of acceptances is stochastically dominated by the distribution arising from independent Bernoulli trials that each output 1 with probability $1/4 + \mathrm{negl}(\lambda)$. Applying Chernoff to this distribution, we see that the probability that at least $\lambda/2$ projectors accept is $\mathrm{negl}(\lambda)$.

To show completeness, we know from the completeness of Π_{Samp} and a union bound that $\mathcal{V}_{\mathsf{QVer}}$ will accept all parts $|\psi_i\rangle$ of the honest prover's proof, except with negligible probability. Conditioned on this, soundness of Π_{Samp} implies that for each i, the verifier will obtain $z' \neq z$ with probability at most $1/4 + \mathrm{negl}(\lambda)$, and so again by Chernoff, the verifier will output (acc, z) except with negligible probability.

□

4.3 Making the verifier classical

Measurement protocol. [Mah18b] constructed a four-message protocol $\Pi_{\mathsf{meas}} = (\mathcal{P}_{\mathsf{meas}}, \mathcal{V}_{\mathsf{meas}})$ between a quantum prover and a classical verifier, with the following syntax.

- $\mathcal{V}_{\mathsf{meas}}(1^\lambda, h) \to (\mathsf{pk}, \mathsf{sk})$: the verifier, on input a string of basis choices h, samples a public key pk and a secret key sk, and sends pk to the prover.
- $\mathcal{P}_{\mathsf{meas}}(\mathsf{pk}, |\psi\rangle) \to (y, |\mathsf{st}\rangle)$: the prover generates a classical commitment y, which it sends to the verifier, and a quantum internal state $|\mathsf{st}\rangle$.
- The verifier samples $c \leftarrow \{0,1\}$ and sends c to the prover, where $c = 0$ indicates a *test* round, and $c = 1$ indicates a *Hadamard* round.
- $\mathcal{P}_{\mathsf{meas}}(|\mathsf{st}\rangle, c) \to \pi$: The prover generates a classical proof π and sends it to the verifier.
- $\mathcal{V}_{\mathsf{meas}}(\mathsf{pk}, \mathsf{sk}, y, c, \pi) \to \mathsf{out}$: If $c = 0$, the verifier computes some classical circuit $\mathcal{V}_{\mathsf{meas},T}(\mathsf{pk}, y, \pi) \to \mathsf{out}$, where $\mathsf{out} \in \{\mathsf{acc}, \mathsf{rej}\}$, and if $c = 1$, the verifier computes some classical circuit $\mathcal{V}_{\mathsf{meas},H}(\mathsf{sk}, y, \pi) \to \mathsf{out}$, where $\mathsf{out} \in \{0,1\}^*$.

This protocol satisfies the following property.

Lemma 4.3 ([Mah18b]). *Let $(\mathcal{P}^*_{\mathsf{meas}}, |\psi_{\mathsf{init}}\rangle)$ be any polynomial-size cheating prover for Π_{meas}, and suppose that there exists an h such that the probability that the verifier accepts if their basis choice was h and $c = 0$ is $1 - \mathrm{negl}(\lambda)$. Then there exists a state ρ such that for all h, the verifier's output if $c = 1$ is computationally indistinguishable from $M_{XZ}(\rho, h)$.*

As observed in [ACGH20], a similar lemma follows by combining the claims [Mah18b, Claim 5.7] and [Mah18b, Claim 7.3]. The lemma stated above is potentially stronger in that (i) it considers non-uniform cheating provers with advice $|\psi_{\mathsf{init}}\rangle$, and (ii) in the premise, it is only required that $\mathcal{P}^*_{\mathsf{meas}}$ is accepted in the test round with probability $1-\mathrm{negl}(\lambda)$ for a *single* basis choice h rather than all. However, it is easy to see that (i) follows from the (standard) assumption that LWE is hard against non-uniform QPT adversaries. Next, (ii) follows due to properties of the extended trapdoor claw-free function used in [Mah18b]'s protocol. Indeed, [Mah18b] shows that for any two basis choices h_0, h_1, no QPT prover will be able to distinguish between pk sampled by $\mathcal{V}(1^\lambda, h_0)$ and pk sampled by $\mathcal{V}(1^\lambda, h_1)$, assuming QLWE. Then since the circuit $\mathcal{V}_{\mathsf{meas},T}$ does not depend on sk, it follows that the probability that $\mathcal{P}^*_{\mathsf{meas}}$ is accepted in a test round is negligibly close for *all* values of h. Thus, if $\mathcal{P}^*_{\mathsf{meas}}$ passes the test round with $1 - \mathrm{negl}(\lambda)$ probability for any h, it will pass with $1 - \mathrm{negl}(\lambda)$ for all h.

A single repetition. We now combine the measurement protocol with Π_{QVer} to obtain Π_{single}, which is a delegation protocol for $D(\cdot) = C(Q(\cdot))$ with a classical verifier and a non-trivial soundness property (though not negligibly sound).

- $\mathcal{V}_{\mathsf{single}}(1^\lambda, Q, x) \to (\mathsf{pk}, \mathsf{sk})$: the verifier samples h according to the distribution defined by $\mathcal{V}_{\mathsf{QVer}}$, samples $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathcal{V}_{\mathsf{meas}}(1^\lambda, h)$, and sends (pk, Q, x) to the prover.

- $\mathcal{P}_{\mathsf{single}}(\mathsf{pk}, Q, x) \to (y, |\mathsf{st}\rangle)$: the prover samples $|\psi\rangle \leftarrow \mathcal{P}_{\mathsf{QVer}}(1^\lambda, Q, x)$, computes $(y, |\mathsf{st}\rangle) \leftarrow \mathcal{P}_{\mathsf{meas}}(\mathsf{pk}, |\psi\rangle)$, and sends y to the verifier.
- The verifer samples $c \leftarrow \{0,1\}$ and sends c to the prover.
- $\mathcal{P}_{\mathsf{single}}(|\mathsf{st}\rangle, c) \to \pi$: the prover samples $\pi \leftarrow \mathcal{P}_{\mathsf{meas}}(|\mathsf{st}\rangle, c)$ and sends π to the verifier.
- $\mathcal{V}_{\mathsf{single}}(Q, C, x, \mathsf{pk}, \mathsf{sk}, y, c, \pi) \to (v, z)$: we split the verifier into two parts $\mathcal{V}^{\mathsf{vrfy}}_{\mathsf{single}}$ and $\mathcal{V}^{\mathsf{out}}_{\mathsf{single}}$.
  - $\mathcal{V}^{\mathsf{vrfy}}_{\mathsf{single}}(Q, x, \mathsf{pk}, \mathsf{sk}, y, c, \pi)$ does the following. If $c = 0$, compute $\mathsf{out} \leftarrow \mathcal{V}_{\mathsf{meas},T}(\mathsf{pk}, y, \pi)$, where $\mathsf{out} \in \{\mathsf{acc}, \mathsf{rej}\}$, and output $(\mathsf{out}, \bot)$. If $c = 1$, compute $e \leftarrow \mathcal{V}_{\mathsf{meas},H}(\mathsf{sk}, y, \pi)$ and run $\mathcal{V}^{\mathsf{test}}_{\mathsf{QVer}}(Q, x, h, e)$ to obtain either rej or $\widehat{z}$. In the first case, output $(\mathsf{rej}, \bot)$, and in the second case, continue.
  - $\mathcal{V}^{\mathsf{out}}_{\mathsf{single}}(C, \widehat{z})$ computes and outputs $(\mathsf{acc}, z) := \mathcal{V}^{\mathsf{out}}_{\mathsf{QVer}}(C, \widehat{z})$.

We show that this protocol satisfies the following.

Lemma 4.4. *Let $(\mathcal{P}^*_{\mathsf{single}}, |\psi_{\mathsf{init}}\rangle)$ be any polynomial-size cheating prover for Π_{single}, and let Q, C, x, z be such that $C(Q(x)) = z$ with probability $1 - \mathrm{negl}(\lambda)$. Suppose that the verifier outputs $(\mathsf{acc}, \bot)$ when $c = 0$ with probability $1 - \mathrm{negl}(\lambda)$. Then assuming QLWE, the probability that the verifier outputs (acc, z') for $z' \neq z$ when $c = 1$ is $\mathrm{negl}(\lambda)$.*

Proof. Fix Q, C, x, z. Consider the predicate F that has Q, C, x, z hard-coded, and on input basis choice h and measurement results e, checks whether $\mathcal{V}_{\mathsf{single}}$ outputs (acc, z') for $z' \neq z$. That is, F first runs $\mathcal{V}^{\mathsf{test}}_{\mathsf{QVer}}(Q, x, h, e)$. If this procedure accepted and output $\widehat{z}$, F computes $(\mathsf{acc}, z') := \mathcal{V}^{\mathsf{out}}_{\mathsf{QVer}}(C, \widehat{z})$ and outputs 1 if $z' \neq z$. Let $p(h)$ be the probability that basis choice h is sampled by $\mathcal{V}_{\mathsf{single}}$. Then we want to show that

$$\sum_h p(h) F(h, e) = \mathrm{negl}(\lambda),$$

where e is obtained by running the protocol with basis choice h and $c = 1$. Now since $\mathcal{P}^*_{\mathsf{single}}$ is accepted with overwhelming probability in the test round, it satisfies the premise in Lemma 4.3. So by Lemma 4.3, the above expression is negligibly close to

$$\sum_h p(h) F(h, M_{X,Z}(\rho, h)),$$

since F is an efficient function outputting a single bit. Finally, this expression is negligible due to the soundness of the underlying information-theoretic protocol given by Lemma 4.2.

□

Parallel repetition. Now we repeat the above protocol in parallel to obtain a CVQC protocol Π_{QC} that satisfies Definition 4.1.

- $\mathcal{V}^{\mathsf{setup}}_{\mathsf{QC}}(1^\lambda, Q, x) \to (\mathsf{pk}, \mathsf{sk})$: set $n \geq \lambda^{1+\epsilon}$ and for each $i \in [n]$, sample $(\mathsf{pk}_i, \mathsf{sk}_i) \leftarrow \mathcal{V}_{\mathsf{single}}(1^\lambda, Q, x)$. Set $\mathsf{pk} := (\mathsf{pk}_1, \ldots, \mathsf{pk}_n)$ and $\mathsf{sk} := (\mathsf{sk}_1, \ldots, \mathsf{sk}_n)$, and send (pk, Q, x) to the prover.
- $\mathcal{P}_{\mathsf{QC}}(\mathsf{pk}, Q, x) \to (y, |\mathsf{st}\rangle)$: for each $i \in [n]$, the prover samples $(y_i, |\mathsf{st}_i\rangle) \leftarrow \mathcal{P}_{\mathsf{single}}(\mathsf{pk}_i, Q, x)$, sets $y := (y_1, \ldots, y_n)$ and $|\mathsf{st}\rangle := (|\mathsf{st}_1\rangle, \ldots, |\mathsf{st}_n\rangle)$, and sends y to the verifier.
- The verifier samples $d \leftarrow \mathcal{HW}_{n,\lambda}$ and sends d to the prover.
- $\mathcal{P}_{\mathsf{QC}}(|\mathsf{st}\rangle, d) \to \pi$: for each $i \in [n]$, the prover samples $\pi_i \leftarrow \mathcal{P}_{\mathsf{single}}(|\mathsf{st}_i\rangle, d_i)$ and sends $\pi := (\pi_1, \ldots, \pi_n)$ to the verifier.
- $\mathcal{V}_{\mathsf{QC}}(Q, C, x, \mathsf{pk}, \mathsf{sk}, y, d, \pi) \to \mathsf{out}$: we split the verifier into two parts $\mathcal{V}^{\mathsf{vrfy}}_{\mathsf{QC}}$ and $\mathcal{V}^{\mathsf{out}}_{\mathsf{QC}}$.

- $\mathcal{V}^{\mathsf{vrfy}}_{\mathsf{QC}}(Q, x, \mathsf{pk}, \mathsf{sk}, y, d, \pi)$ runs $\mathcal{V}^{\mathsf{vrfy}}_{\mathsf{single}}(Q, x, \mathsf{pk}_i, \mathsf{sk}_i, y_i, d_i, \pi_i)$ for each $i \in [n]$. If any outputs $(\mathsf{rej}, \bot)$ then output $(\mathsf{rej}, \bot)$. Otherwise, obtain strings $\{\widehat{z}_i\}_{i:d_i=1}$.
- $\mathcal{V}^{\mathsf{out}}_{\mathsf{QC}}(C, \{\widehat{z}_i\}_{i:d_i=1})$ computes the λ outputs $(\mathsf{acc}, z_i) := \mathcal{V}^{\mathsf{out}}_{\mathsf{single}}(C, \widehat{z}_i)$. Then determine the most frequently occurring z in the resulting set and output (acc, z).

Lemma 4.5. *Assuming QLWE, Π_{QC} satisfies Definition 4.1*

Proof. First, we show soundness. Fix C, Q, x, z such that $C(Q(x)) = z$ with probability $1 - \mathrm{negl}(\lambda)$. Define a Hadamard-round verification projector for Π_{single} that checks whether $\mathcal{V}_{\mathsf{single}}$ outputs (acc, z') for $z' \neq z$. Lemma 4.4 and Theorem 3.1 imply that, conditioned on all test rounds (i such that $d_i = 0$) accepting in Π_{QC}, this verifier only accepts at most $\lceil \lambda/2 \rceil - 1$ of the indices for which $d_i = 1$, except with negligible probability. Thus, conditioned on $\mathcal{V}^{\mathsf{vrfy}}_{\mathsf{QC}}$ accepting, a strict majority of the Hadamard rounds will result in the outcome z, except with negligible probability.

It remains to argue semi-malicious correctness. We observe that the strings $\{\widehat{z}_i\}_{i:d_i=1}$ are obtained via computational basis measurements (which follows from the description of [CLLW20]'s sampBQP protocol). Since we are considering an honest prover, it suffices to argue that computational basis measurements are always performed correctly under $(\mathsf{pk}, \mathsf{sk})$. The parts of $(\mathsf{pk}, \mathsf{sk})$ that are used for computational basis measurements are injective trapdoor keys, and the measurements will be correct if the keys are indeed injective. The sampling procedure for injective keys given in [Mah18b, Section 9.2] can be made to produce an injective key with probability 1 over its random coins, by outputting a fixed injective key in the case that the initial sampling failed to produce an injective key. □

4.4 Four-message CVQC

In this section, we use CVQC for quantum-classical circuits in combination with quantum fully-homomorphic encryption to construct a four-message blind CVQC protocol that enjoys composability. That is, the protocol that we construct satisfies an ideal functionality that either delivers the correct output $D(x)$ or a $\bot$ symbol to the verifier, depending on a bit b input by the prover. Security is argued in the simulation sense according to Definition 2.6, which in particular implies that the prover's bit b cannot depend on the verifier's input x. We first give our definition of composable blind CVQC, which is essentially the same as the notion of *blind and verifiable delegated quantum computation* studied by [DFPR14], but adapted to the classical verifier setting.

Definition 4.6 (Composable Blind CVQC). *Consider the two-party functionality $\mathcal{F}_D$ between classical verifier and quantum prover defined by a psuedo-determistic circuit $D(\cdot)$. It takes an input string x from the verifier and an input bit b from the prover (which is always 0 if the prover is honest). If $b = 0$, it delivers $D(x)$ to the verifier (and no output to the prover), and if $b = 1$, it delivers $\bot$ to the verifier (and no output to the prover). Protocol Π is a* composable blind CVQC *protocol if for any pseudo-deterministic D, it satisfies Definition 2.6 for functionality $\mathcal{F}_D$.*

To construct such a protocol, we use the following ingredients.

- A four-message CVQC protocol for quantum-classical circuits Π_{QC} (Definition 4.1).
- Quantum fully-homomorphic encryption with classical keys $(\mathsf{QFHE.Gen}, \mathsf{QFHE.Enc}, \mathsf{QFHE.Eval}, \mathsf{QFHE.Dec})$ (Definition 2.4).

The construction is given in Protocol 1.

Theorem 4.7. *Assuming QLWE, Protocol 1 is a composable blind CVQC protocol satisfying definition Definition 4.6.*

Proof. First, we argue that in case neither party is corrupted, the verifier's output is correct with overwhelming probability. This follows from the completeness of Π_{QC}, since for any psuedo-deterministic circuit D, input x, $(\mathsf{pk}_{\mathsf{QFHE}}, \mathsf{sk}_{\mathsf{QFHE}}) \in \mathsf{QFHE.Gen}(1^\lambda)$, and $\mathsf{ct} \in \mathsf{QFHE.Enc}(\mathsf{pk}_{\mathsf{QFHE}}, x)$, it holds that $\mathsf{QFHE.Dec}(\mathsf{sk}_{\mathsf{QFHE}}, \mathsf{QFHE.Eval}(\mathsf{pk}_{\mathsf{QFHE}}, D, \cdot))$ applied to ct outputs $D(x)$ with overwhelming probability, so the computation

Protocol 1: Composable Blind CVQC

- $\mathcal{V}_{\mathsf{blind}}^{\mathsf{setup}}(1^\lambda, D, x)$: the verifier samples

$$(\mathsf{pk}_{\mathsf{QFHE}}, \mathsf{sk}_{\mathsf{QFHE}}) \leftarrow \mathsf{QFHE.Gen}(1^\lambda), \mathsf{ct} \leftarrow \mathsf{QFHE.Enc}(\mathsf{pk}_{\mathsf{QFHE}}, x),$$

  then it samples $(\mathsf{pk}_{\mathsf{eval}}, \mathsf{sk}_{\mathsf{eval}}) \leftarrow \mathcal{V}_{\mathsf{QC}}^{\mathsf{setup}}(1^\lambda, \mathsf{QFHE.Eval}(D, \cdot), \mathsf{ct})$, sets

$$\mathsf{pk} := (\mathsf{pk}_{\mathsf{eval}}, \mathsf{QFHE.Eval}(D, \cdot), \mathsf{ct}), \mathsf{sk} := (\mathsf{sk}_{\mathsf{eval}}, \mathsf{sk}_{\mathsf{QFHE}}),$$

  and sends pk to the prover.

- $\mathcal{P}_{\mathsf{blind}}(\mathsf{pk})$: the prover samples $(y, |\mathsf{st}\rangle) \leftarrow \mathcal{P}_{\mathsf{QC}}(\mathsf{pk})$, and sends y to the verifier.
- The verifier samples d according to Π_{QC}.
- $\mathcal{P}_{\mathsf{blind}}(|\mathsf{st}\rangle, d)$: the prover samples $\pi \leftarrow \mathcal{P}_{\mathsf{QC}}(|\mathsf{st}\rangle, d)$, and sends π to the verifier.
- $\mathcal{V}_{\mathsf{blind}}(\mathsf{pk}, \mathsf{sk}, y, d, \pi)$: the verifier first runs

$$\mathcal{V}_{\mathsf{QC}}^{\mathsf{vrfy}}(\mathsf{QFHE.Eval}(D, \cdot), \mathsf{ct}, \mathsf{pk}_{\mathsf{eval}}, \mathsf{sk}_{\mathsf{eval}}, y, d, \pi)$$

  to obtain either rej or $\widetilde{z}$. In the first case, it outputs $\bot$. In the second case, it computes

$$z := \mathcal{V}_{\mathsf{QC}}^{\mathsf{out}}(\mathsf{QFHE.Dec}(\mathsf{sk}_{\mathsf{QFHE}}, \cdot), \widetilde{z}),$$

  and outputs z.

Figure 1: A four-message composable blind CVQC from QLWE.

performed by Π_{QC} is pseudo-deterministic. Next, note that simulation in case the verifier is corrupted is trivial, since the honest prover has no input or output.

It remains to argue security when the prover is corrupted, for which we define the following simulator. The simulator samples a QFHE encryption of 0, and then interacts with the prover as the honest verifier delegating the circuit $\mathsf{QFHE.Eval}(D, \cdot)$. At the end of the protocol, the simulator first runs $\mathcal{V}_{\mathsf{QC}}^{\mathsf{vrfy}}$. If this results in rej, send $b = 1$ to the ideal functionality, and otherwise send $b = 0$ to the ideal functionality. Now consider the following sequence of hybrids.

- $\mathcal{H}_0$: This is the real interaction between honest verifier and malicious prover, resulting in a verifier output in $\{0, 1\}^* \cup \{\bot\}$ and a final prover state.

- $\mathcal{H}_1$: In this hybrid, we change how the final verifier output is computed. In particular, in the case that $\mathcal{V}_{\mathsf{QC}}^{\mathsf{vrfy}}$ does not reject, compute the output $z = D(x)$. This is computationally indistinguishable from $\mathcal{H}_0$ due to the soundness of Π_{QC}, which guarantees that, conditioned on the verifier accepting, the output will be equal to $D(x)$ except with negligible probability.

- $\mathcal{H}_2$: In this hybrid, we change how the verifier's first message is computed. In particular, set ct to be a QFHE encryption of 0 rather than x. This is computationally indistinguishable from $\mathcal{H}_1$ due to the semantic security of QFHE, since the QFHE secret key is no longer needed to compute the output of the experiment.

- $\mathcal{H}_3$: In this hybrid, if $\mathcal{V}_{\mathsf{QC}}^{\mathsf{vrfy}}$ rejects, then send $b = 1$ to the ideal functionality, and otherwise send $b = 0$. This is identical to $\mathcal{H}_2$, since we have just moved the computation of $z = D(x)$ to the ideal functionality. This is the simulator, completing the proof.

□

4.5 Two-message CVQC with extra properties

As we show in Section 5.1, it is straightforward to compose the above CVQC protocol with multi-party computation for classical circuits to obtain multi-party quantum computation with classical communication. However, we would also like to optimize the round complexity of the resulting MPQC. Towards this goal, we define and construct an alternative CVQC protocol with various extra properties that will be useful in constructing round-efficient protocols in Section 5.2 and Section 5.3.

Definition 4.8 (Two-message semi-malicious CVQC with malicious blindness and distributed setup)**.** *A two-message semi-malicious CVQC with malicious blindness and distributed setup for n-input pseudo-deterministic circuit $D(\cdot,\ldots,\cdot)$ has the following syntax.*

- $\mathsf{Gen}(1^\lambda) \to \mathsf{crs}$: *the generation algorithm outputs a common random string* crs.
- $\mathcal{V}^{\mathsf{params}}_{\mathsf{blind}}(1^\lambda, D) \to (\mathsf{pp},\mathsf{sp})$*: the* params *part of the setup takes the security parameter* 1^λ *and a quantum circuit* D*, and outputs public parameters* pp *and secret parameters* sp.
- $\mathcal{V}^{\mathsf{inp}}_{\mathsf{blind}}(1^\lambda, \mathsf{crs}, x_i) \to (\mathsf{ct}_i,\mathsf{sk}_i)$*: for each* $i \in [n]$*, the* inp *part of the setup takes the security parameter* 1^λ*, the crs* crs*, and an input* x_i*, and outputs a ciphertext* ct_i *and secret key* sk_i.
- $\mathcal{P}_{\mathsf{blind}}(\mathsf{pp},\mathsf{ct}_1,\ldots,\mathsf{ct}_n) \to \pi$*: the prover takes as input the public parameters* pp *and* n *ciphertexts* $\mathsf{ct}_1,\ldots,\mathsf{ct}_n$*, and outputs a proof* π.
- $\mathcal{V}_{\mathsf{blind}}(\mathsf{pp},\mathsf{ct}_1,\ldots,\mathsf{ct}_n,\mathsf{sp},\mathsf{sk}_1,\ldots,\mathsf{sk}_n,\pi) \to (v,z)$*: the verifier takes as input the public parameters* pp*,* n *ciphertexts* $\mathsf{ct}_1,\ldots,\mathsf{ct}_n$*, the secret parameters* sp*,* n *secret keys* $\mathsf{sk}_1,\ldots,\mathsf{sk}_n$*, and a proof* π*, and outputs* $v \in \{\mathsf{acc},\mathsf{rej}\}$ *and* $z \in \{0,1\}^*$.

We require that the protocol satisfies the standard notion of completeness, along with semi-malicious versions of correctness, soundness, and malicious blindness.

- ***Completeness:*** *For any* n*-input circuit* $D(\cdot,\ldots,\cdot)$*, inputs* $x_1,\ldots,x_n$*, and output* z *such that* $\Pr[D(x_1,\ldots,x_n) = z] = 1-\mathrm{negl}(\lambda)$*, it holds that*

$$\Pr\left[v=\mathsf{acc}\wedge z'=z : \begin{array}{r} \mathsf{crs}\leftarrow\mathsf{Gen}(1^\lambda)\\ (\mathsf{pp},\mathsf{sp})\leftarrow\mathcal{V}^{\mathsf{params}}_{\mathsf{blind}}(1^\lambda,D)\\ \forall i\in[n],(\mathsf{ct}_i,\mathsf{sk}_i)\leftarrow\mathcal{V}^{\mathsf{inp}}_{\mathsf{blind}}(1^\lambda,\mathsf{crs},x_i)\\ \pi\leftarrow\mathcal{P}_{\mathsf{blind}}(\mathsf{pp},\mathsf{ct}_1,\ldots,\mathsf{ct}_n)\\ (v,z')\leftarrow\mathcal{V}_{\mathsf{blind}}(\mathsf{pp},\mathsf{ct}_1,\ldots,\mathsf{ct}_n,\mathsf{sp},\mathsf{sk}_1,\ldots,\mathsf{sk}_n,\pi)\end{array}\right] = 1-\mathrm{negl}(\lambda).$$

- ***Semi-malicious correctness:*** *For any* n*-input circuit* $D(\cdot,\ldots,\cdot)$*, inputs* $x_1,\ldots,x_n$*, output* z *such that* $\Pr[D(x_1,\ldots,x_n)=z] = 1-\mathrm{negl}(\lambda)$*,* $\mathsf{crs}\leftarrow\mathsf{Gen}(1^\lambda)$*,* $(\mathsf{pp},\mathsf{sp})\in\mathcal{V}^{\mathsf{params}}_{\mathsf{blind}}(1^\lambda,D)$*, and* $(\mathsf{ct}_i,\mathsf{sk}_i)\in\mathcal{V}^{\mathsf{inp}}_{\mathsf{blind}}(1^\lambda,\mathsf{crs},x_i)$ *for all* $i\in[n]$*, it holds that*

$$\Pr\left[v=\mathsf{acc}\wedge z'\neq z : \begin{array}{r} \pi\leftarrow\mathcal{P}_{\mathsf{blind}}(\mathsf{pp},\mathsf{ct}_1,\ldots,\mathsf{ct}_n)\\ (v,z')\leftarrow\mathcal{V}_{\mathsf{blind}}(\mathsf{pp},\mathsf{ct}_1,\ldots,\mathsf{ct}_n,\mathsf{sp},\mathsf{sk}_1,\ldots,\mathsf{sk}_n,\pi)\end{array}\right] = \mathrm{negl}(\lambda).$$

- ***Semi-malicious soundness:*** *For any* n*-input circuit* $D(\cdot,\ldots,\cdot)$*, inputs* $x_1,\ldots,x_n$*, output* z *such that* $\Pr[D(x_1,\ldots,x_n) = z] = 1-\mathrm{negl}(\lambda)$*,* $\mathsf{crs}\leftarrow\mathsf{Gen}(1^\lambda)$*,* $(\mathsf{ct}_i,\mathsf{sk}_i)\in\mathcal{V}^{\mathsf{inp}}_{\mathsf{blind}}(1^\lambda,\mathsf{crs},x_i)$ *for each* $i\in[n]$*, and cheating prover* $\mathcal{P}^*_{\mathsf{blind}}$ *with advice* $|\psi\rangle$*, it holds that*

$$\Pr\left[v=\mathsf{acc}\wedge z'\neq z : \begin{array}{r} (\mathsf{pp},\mathsf{sp})\leftarrow\mathcal{V}^{\mathsf{params}}_{\mathsf{blind}}(1^\lambda,D)\\ \pi\leftarrow\mathcal{P}^*_{\mathsf{blind}}(|\psi\rangle,\mathsf{pp},\mathsf{ct}_1,\ldots,\mathsf{ct}_n)\\ (v,z')\leftarrow\mathcal{V}_{\mathsf{blind}}(\mathsf{pp},\mathsf{ct}_1,\ldots,\mathsf{ct}_n,\mathsf{sp},\mathsf{sk}_1,\ldots,\mathsf{sk}_n,\pi)\end{array}\right] = \mathrm{negl}(\lambda).$$

- ***Malicious blindness:*** *For any n-input circuit $D(\cdot,\ldots,\cdot)$, index $i^* \in [n]$, inputs $\{x_i\}_{i\neq i^*}$,* $\mathsf{crs} \leftarrow \mathsf{Gen}(1^\lambda)$, $(\mathsf{pp},\mathsf{sp}) \in \mathcal{V}_{\mathsf{blind}}^{\mathsf{params}}(1^\lambda, D)$, $(\mathsf{ct}_i,\mathsf{sk}_i) \in \mathcal{V}_{\mathsf{blind}}^{\mathsf{inp}}(1^\lambda,\mathsf{crs},x_i)$ *for each $i \neq i^*$, inputs $x_{i^*}^{(0)}, x_{i^*}^{(1)}$, and two-part non-uniform cheating prover $\mathcal{P}_1^*, \mathcal{P}_2^*, |\psi\rangle$, it holds that*

$$\left|\Pr\left[\mathcal{P}_2^*(|\mathsf{st}\rangle, v) = 1 : \begin{array}{r}(\mathsf{ct}_{i^*},\mathsf{sk}_{i^*}) \leftarrow \mathcal{V}_{\mathsf{blind}}^{\mathsf{inp}}(1^\lambda,\mathsf{crs},x_{i^*}^{(0)})\\ (\pi,|\mathsf{st}\rangle) \leftarrow \mathcal{P}_1^*(|\psi\rangle,\mathsf{pp},\mathsf{ct}_1,\ldots,\mathsf{ct}_n)\\ (v,z') \leftarrow \mathcal{V}_{\mathsf{blind}}(\mathsf{pp},\mathsf{ct}_1,\ldots,\mathsf{ct}_n,\mathsf{sp},\mathsf{sk}_1,\ldots,\mathsf{sk}_n,\pi)\end{array}\right]\right.$$
$$\left.-\Pr\left[\mathcal{P}_2^*(|\mathsf{st}\rangle, v) = 1 : \begin{array}{r}(\mathsf{ct}_{i^*},\mathsf{sk}_{i^*}) \leftarrow \mathcal{V}_{\mathsf{blind}}^{\mathsf{inp}}(1^\lambda,\mathsf{crs},x_{i^*}^{(1)})\\ (\pi,|\mathsf{st}\rangle) \leftarrow \mathcal{P}_1^*(|\psi\rangle,\mathsf{pp},\mathsf{ct}_1,\ldots,\mathsf{ct}_n)\\ (v,z') \leftarrow \mathcal{V}_{\mathsf{blind}}(\mathsf{pp},\mathsf{ct}_1,\ldots,\mathsf{ct}_n,\mathsf{sp},\mathsf{sk}_1,\ldots,\mathsf{sk}_n,\pi)\end{array}\right]\right| = \mathrm{negl}(\lambda).$$

Protocol 2: Semi-malicious CVQC with malicious blindness and distributed setup

- $\mathsf{Gen}(1^\lambda)$: sample $\mathsf{crs} \leftarrow \mathsf{QMFHE.Gen}(1^\lambda)$. Also let $\mathcal{H}$ be a random oracle.
- $\mathcal{V}_{\mathsf{blind}}^{\mathsf{params}}(1^\lambda, D)$: the parameter generation algorithm samples

$$(\mathsf{pp},\mathsf{sp}) \leftarrow \mathcal{V}_{\mathsf{QC}}^{\mathsf{setup}}(1^\lambda, \mathsf{QMFHE.Eval}(D,\cdot)).$$

- $\mathcal{V}_{\mathsf{blind}}^{\mathsf{inp}}(1^\lambda,\mathsf{crs},x_i)$: the input generation algorithm samples

$$(\mathsf{pk}_i,\mathsf{sk}_i) \leftarrow \mathsf{QMFHE.KeyGen}(1^\lambda,\mathsf{crs}), \mathsf{ct}_i \leftarrow \mathsf{QMFHE.Enc}(\mathsf{pk}_i, x_i).$$

- $\mathcal{P}_{\mathsf{blind}}(\mathsf{pp},\mathsf{ct}_1,\ldots,\mathsf{ct}_n)$: the prover computes

$$(y,|\mathsf{st}\rangle) \leftarrow \mathcal{P}_{\mathsf{QC}}(\mathsf{pp},\mathsf{QMFHE.Eval}(D,\cdot),(\mathsf{ct}_1,\ldots,\mathsf{ct}_n)), d := \mathcal{H}(y)^{a}, \pi \leftarrow \mathcal{P}_{\mathsf{QC}}(|\mathsf{st}\rangle, d),$$

  and returns (y,d,π).

- $\mathcal{V}_{\mathsf{blind}}(\mathsf{pp},\mathsf{ct}_1,\ldots,\mathsf{ct}_n,\mathsf{sp},\mathsf{sk}_1,\ldots,\mathsf{sk}_n,y,d,\pi)$: the verifier first checks that $\mathcal{H}(y) = d$ and if not outputs $(\mathsf{rej},\bot)$. Otherwise, it computes

$$\mathcal{V}_{\mathsf{QC}}^{\mathsf{vrfy}}(\mathsf{QMFHE.Eval}(D,\cdot),(\mathsf{ct}_1,\ldots,\mathsf{ct}_n),\mathsf{pp},\mathsf{sp},y,d,\pi)$$

  to obtain either rej or $\widetilde{z}$. In the first case, it outputs $(\mathsf{rej},\bot)$. In the second case, it computes

$$z := \mathcal{V}_{\mathsf{QC}}^{\mathsf{out}}(\mathsf{QMFHE.Dec}(\mathsf{sk}_1,\ldots,\mathsf{sk}_n,\cdot),\widetilde{z})$$

  and outputs (acc, z).

[a] Technically, the string d in Π_{QC} is not a uniformly random string, rather, it is uniform over all binary strings with a particular Hamming weight. However, this is still a public-coin message, and we can generate these coins with $\mathcal{H}$. We simplify notation by writing $\mathcal{H}(y) = d$ even though $\mathcal{H}(y)$ actually outputs the uniformly random coins used to determine d.

Figure 2: A two-message semi-malicious CVQC with malicious blindness and distributed setup from QLWE in the QROM.

We now construct a protocol satisfying Definition 4.8 using the following ingredients.

- A four-message CVQC protocol for quantum classical circuits Π_{QC} (see Section 4.3).

- Quantum multi-key fully-homomorphic encryption with classical keys $(\mathsf{QMFHE.Gen}, \mathsf{QMFHE.Enc}, \mathsf{QMFHE.Eval}, \mathsf{QMFHE.Dec})$ (Definition 2.4).

Theorem 4.9. *Assuming QLWE, Protocol 2 satisfies Definition 4.8 in the QROM.*

Proof. First, note that we changed the syntax of $\mathcal{V}_{\mathsf{QC}}^{\mathsf{setup}}$ to not depend on the input $(\mathsf{ct}_1, \ldots, \mathsf{ct}_n)$ of the computation that will be performed by Π_{QC}. This is possible due to the observation from [ACGH20, Section 3] that the first message in CVQC can be made instance-independent, since the sampling of the measurement bases in the underlying information-theoretic protocol [FHM18, CLLW20] can be made instance-independent.

Now we prove security of the protocol. First, by the statistical correctness of QMFHE, the computation performed by Π_{QC} will always be pseudo-deterministic for inputs $(\mathsf{ct}_i, \mathsf{sk}_i) \in \mathcal{V}_{\mathsf{blind}}^{\mathsf{inp}}(1^\lambda, \mathsf{crs}, x_i)$. Next, note that we have actually collapsed the second and third messages of Π_{QC} using the random oracle $\mathcal{H}$. This clearly does not affect completeness or semi-malicious correctness, so these properties follow directly from the corresponding properties of Π_{QC}, which were shown in Lemma 4.5. For semi-malicious soundness, we appeal to "Fiat-Shamir for generalized Σ-protocols" [ACGH20, Theorem 6.3]. Since Π_{QC} satisfies the syntax of a generalized Σ-protocol, this theorem shows that, assuming Π_{QC} is sound, the two-message collapsed protocol is sound in the QROM. It was shown in Lemma 4.5 that Π_{QC} is sound for the delegation of any pseudo-deterministic circuit, and thus semi-malicious soundness of Protocol 2 follows. Finally, malicious blindness follows from the semantic security of QMFHE, since sk_{i^*} is not needed to determine whether the verifier accepts or rejects. □

5 Secure Quantum Computation

In this section, we show applications of the CVQC protocols constructed in Section 4 to secure quantum computation with classical communication (where only a single party requires quantum capabilities).

First, in Section 5.1, we give a generic compiler from composable blind CVQC to multi-party quantum computation with classical computation, assuming the existence of post-quantum oblivious transfer. In fact, assuming two-message post-quantum oblivious transfer, the compiler only adds two rounds of interaction to the composable blind CVQC protocol. Next, in Section 5.2, we show how to optimize the round complexity of multi-party quantum computation, achieving a three-round protocol. This protocol requires a (succinct and reusable) PKI setup and security follows from QLWE in the quantum random oracle model. Finally, in Section 5.3, we show how to construct a two-message two-party protocol between quantum sender and classical receiver (a quantum NISC protocol). Security of this protocol follows from QLWE in the quantum random oracle model.

5.1 A generic construction of multi-party quantum computation

In this section, we show that, assuming post-quantum two-message oblivious transfer, k-message composable blind CVQC implies $k+2$-round multi-party quantum computation between any n classical clients and a single quantum server. The protocol (in fact, all protocols in this section) is described for functionalities with a single public output, but this is easy to generalize to multiple private outputs using secret-key encryption.

Ingredients.

- A post-quantum round-optimal MPC protocol for classical reactive functionalities in the CRS model, to be treated as an oracle called MPC. Such a protocol is known from post-quantum two-message oblivious transfer, which is known from QLWE . See Appendix C for more discussion, and a precise description of the ideal functionality implemented by the oracle MPC.
- A k-message composable blind CVQC protocol (Definition 4.6). We assume without loss of generality that k is even and the verifier sends the first message. The transcript of messages between verifier and prover are denoted $\mathsf{msg}_1^{(V)}, \mathsf{msg}_2^{(P)}, \ldots, \mathsf{msg}_{k-1}^{(V)}, \mathsf{msg}_k^{(P)}$.

Protocol 3: Multi-Party Quantum Computation with Classical Communication

Public Information: An n-party pseudo-deterministic quantum functionality $D(\cdot,\ldots,\cdot)$, security parameter λ, n classical client parties $P_1,\ldots,P_n$, and a designated server S with quantum capabilities.

- **Round 1:** Each party P_i sends their input x_i to MPC.
- **Round 2:** MPC computes $\mathsf{msg}_1^{(V)}$ and sends it to S.
- **Round 3:** S computes and broadcasts $\mathsf{msg}_2^{(P)}$.

$\vdots$

- **Round k:** MPC computes $\mathsf{msg}_{k-1}^{(V)}$ and sends it to S.
- **Round $k+1$:** S computes and broadcasts $\mathsf{msg}_k^{(P)}$.
- **Round $k+2$:** MPC computes the output $z \in \{0,1\}^* \cup \{\perp\}$ of the CVQC protocol and delivers z to each P_i.

Figure 3: An MPQC protocol with classical communication.

Theorem 5.1. *Protocol 3 satisfies Definition 2.6.*

Proof. First consider any adversary $\{\mathsf{Adv}_\lambda\}_{\lambda\in\mathbb{N}}$ that corrupts a set of parties M such that $S \in M$, and let $H := [n] \setminus M$. The simulator is defined below. We allow Sim to maintain the MPC oracle, intercepting the adversary's inputs and computing the outputs.

$\mathsf{Sim}(\{x_i\}_{i\in M}, |\psi\rangle_{\mathsf{Adv}})$:

- Obtain $\{x_i\}_{i\in M}$ from Adv's initial query to MPC.
- Invoke the malicious prover simulator for the composable blind CVQC protocol, which computes messages on behalf of MPC, and interacts with Adv (controlling the server) until after Round $k+1$. The simulator outputs a bit b.
- If $b = 0$, query the ideal functionality with $\{x_i\}_{i\in M}$ to obtain z and deliver z to Adv. Otherwise, if $b = 1$, send abort to the ideal functionality and deliver $\perp$ to Adv.

Now consider the following hybrids.

- $\mathcal{H}_0$: This is the real distribution $\mathsf{REAL}_{\Pi,\mathsf{Q}}(\mathsf{Adv}_\lambda, \{x_i\}_{i\in[n]}, |\psi\rangle_{\mathsf{Adv}})$, where MPC is implemented honestly as the CVQC verifier with input $x_1,\ldots,x_n$.
- $\mathcal{H}_1$: Invoke the malicious prover simulator for the composable blind CVQC protocol to simulate the interaction between Adv and MPC through Round $k+1$. Also, change how the final message from MPC is computed. That is, if the CVQC simulator output $b = 0$, compute $z = D(x_1,\ldots,x_n)$ (where $\{x_i\}_{i\in M}$ were obtained from Adv's first query to MPC, and $\{x_i\}_{i\in H}$ are the honest party inputs) and deliver z to Adv, and otherwise, deliver $\perp$ to Adv. This is indistinguishable from $\mathcal{H}_0$ due to the security of the CVQC protocol.
- $\mathcal{H}_2$: Obtain the output z by querying the ideal functionality with $\{x_i\}_{i\in M}$, which is perfectly indistinguishable from $\mathcal{H}_1$. This is the simulator.

Now consider any adversary $\{\mathsf{Adv}_\lambda\}_{\lambda\in\mathbb{N}}$ that corrupts a set of parties $M \subset \{P_1, \ldots, P_n\}$. Security in this case is almost immediate. Indeed, the simulator will obtain $\{x_i\}_{i\in M}$ from Adv's query to MPC in Round 1, query the ideal functionality to obtain z and then deliver z to Adv in Round $k+2$. This is indistinguishable from the real distribution due to the correctness of the CVQC protocol, that is, the requirement that the correct output is generated when neither of the parties in corrupted.

□

Combining with our four-message composable blind CVQC protocol from Section 4.4, we obtain the following corollary.

Corollary 5.2. *Assuming QLWE, there exists a six-round multi-party quantum computation protocol in the CRS model between n classical clients and one quantum server for computing any n-party pseudo-deterministic functionality $D(\cdot, \ldots, \cdot)$.*

5.2 Three-round multi-party quantum computation

Our next protocol will make use of a special type of two-round MPC, which has recently been referred to as multi-party reusable non-interactive secure computation (mrNISC) [BL20]. Such a protocol consists of four algorithms $(\mathsf{mrNISC.Gen}, \mathsf{mrNISC.Com}, \mathsf{mrNISC.Encode}, \mathsf{mrNISC.Eval})$, and proceeds in three phases, after crs is sampled by $\mathsf{mrNISC.Gen}(1^\lambda)$.

- Input Encoding: A party P_i encodes their private input x_i by running $(\widehat{x}_i, \mathsf{st}_i) \leftarrow \mathsf{mrNISC.Com}(1^\lambda, \mathsf{crs}, x_i)$. They broadcast $\widehat{x}_i$ and keep their secret state st_i.
- Computation: Given a public n-party functionality C, a set I of n parties $\{P_i\}_{i\in I}$ generates an encoding of the output $y := C(x_1, \ldots, x_n)$ by running $\widehat{y}_i \leftarrow \mathsf{mrNISC.Encode}(C, \{\widehat{x}_i\}_{i\in I}, \mathsf{st}_i)$.
- Output: An evaluator can publicly reconstruct the output $y := \mathsf{mrNISC.Eval}(C, \{\widehat{x}_i, \widehat{y}_i\}_{i\in I})$.

The protocol should satisfy standard malicious security against a dishonest majority of parties (in the CRS model). Moreover, the input encodings should be *reusable* across any number of computations performed by arbitrary subsets of the parties, and security should be maintained in this setting. Such a protocol has recently been constructed from QLWE [AJJM21, BJKL21], and these protocols are post-quantum secure since the simulators and reductions are straight-line.[7]

We now give a three-round MPQC protocol in the PKI + QRO model, assuming QLWE.

Ingredients.

- A post-quantum mrNISC protocol $\mathsf{mrNISC} = (\mathsf{mrNISC.Gen}, \mathsf{mrNISC.Com}, \mathsf{mrNISC.Encode}, \mathsf{mrNISC.Eval})$.
- A two-message semi-malicious CVQC protocol with malicious blindness and distributed setup $\Pi_{\mathsf{blind}} = (\mathsf{Gen}, \mathcal{P}_{\mathsf{blind}}, \mathcal{V}^{\mathsf{params}}_{\mathsf{blind}}, \mathcal{V}^{\mathsf{inp}}_{\mathsf{blind}}, \mathcal{V}_{\mathsf{blind}})$ (Protocol 2). Let $\mathcal{V}^*_{\mathsf{blind}}$ be the circuit $\mathcal{V}_{\mathsf{blind}}$ with the random oracle check $\mathcal{H}(y) = d$ removed.

Theorem 5.3. *Protocol 4 satisfies Definition 2.6 in the PKI model.*

Proof. Consider any adversary $\{\mathsf{Adv}_\lambda\}_{\lambda\in\mathbb{N}}$ that corrupts a set of parties M such that $S \in M$, and let $H = [n] \setminus M$. Like in the previous protocol, security against adversaries that don't corrupt M is the simpler case, and the proof is omitted.

We demonstrate simulation for a single invocation of the protocol on circuit D, but note that the PKI can be reused across any number of invocations due to the reusability of mrNISC.

$\mathsf{Sim}(\{x_i\}_{i\in M}, |\psi\rangle_{\mathsf{Adv}})$:

[7] Technically, these works only show semi-malicious security, but such protocols can be compiled (in a post-quantum manner) to maliciously-secure protocols via UC-secure NIZKs from QLWE [AJL+12, PS19].

Protocol 4: Three-Round MPQC

Public Information: Security parameter λ, $\mathsf{crs} \leftarrow \mathsf{Gen}(1^\lambda)$, $\mathsf{mrNISC.crs} \leftarrow \mathsf{mrNISC.Gen}(1^\lambda)$. Let $\mathcal{H}$ be a random oracle, used in Π_{blind}.
PKI Setup: Party P_i samples PRF key k_i, runs $(\mathsf{pk}_i, \mathsf{st}_{i,\mathsf{key}}) \leftarrow \mathsf{mrNISC.Com}(1^\lambda, \mathsf{mrNISC.crs}, k_i)$, and broadcasts pk_i.

- **Input Round:** Let $\{P_i\}_{i\in[n]}$ be a set of n parties with inputs $\{x_i\}_{i\in[n]}$ that wish to compute some circuit $D(x_1, \ldots, x_n)$. Let $\widetilde{D}(\cdot, \ldots, \cdot)$ be the circuit that takes as input n PRF keys $k_1, \ldots, k_n$ and computes $\mathcal{V}^{\mathsf{params}}_{\mathsf{blind}}(1^\lambda, D; \bigoplus_i \mathsf{PRF}_{k_i}(D))$. Each $i \in [n]$ will
  - sample random coins $r_i \leftarrow \{0,1\}^\lambda$, compute $(\mathsf{ct}_i, \mathsf{sk}_i) := \mathcal{V}^{\mathsf{inp}}_{\mathsf{blind}}(1^\lambda, \mathsf{crs}, x_i; r_i)$,
  - compute $(\widehat{(x_i, r_i)}, \mathsf{st}_{i,\mathsf{inp}}) \leftarrow \mathsf{mrNISC.Com}(1^\lambda, \mathsf{mrNISC.crs}, (x_i, r_i))$,
  - compute $\widehat{\mathsf{pp}}_i \leftarrow \mathsf{mrNISC.Encode}(\widetilde{D}, \{\mathsf{pk}_i\}_{i\in[n]}, \mathsf{st}_{i,\mathsf{key}})$,
  - and broadcast $(\mathsf{ct}_i, \widehat{(x_i, r_i)}, \widehat{\mathsf{pp}}_i)$.
- **Server Round:** The server S computes
  - $\mathsf{pp} := \mathsf{mrNISC.Eval}(\widetilde{D}, \{\mathsf{pk}_i, \widehat{\mathsf{pp}}_i\}_{i\in[n]})$,
  - $(y, d, \pi) \leftarrow \mathcal{P}_{\mathsf{blind}}(\mathsf{pp}, \mathsf{ct}_1, \ldots, \mathsf{ct}_n)$,
  - and broadcasts (y, d, π).
- **Output Round:** Let $E[\mathsf{ct}_1, \ldots, \mathsf{ct}_n, y, d, \pi](\cdot, \ldots, \cdot)$ be the circuit that
  - takes as input $(k_1, x_1, r_1), \ldots, (k_n, x_n, r_n)$,
  - verifies that each ct_i matches $(\mathsf{ct}_i, \mathsf{sk}_i) := \mathcal{V}^{\mathsf{inp}}_{\mathsf{blind}}(1^\lambda, \mathsf{crs}, x_i; r_i)$,
  - and if so runs $(\mathsf{pp}, \mathsf{sp}) := \mathcal{V}^{\mathsf{params}}_{\mathsf{blind}}(1^\lambda, D; \bigoplus_i \mathsf{PRF}_{k_i}(D))$,
  - and $(v, z) \leftarrow \mathcal{V}^*_{\mathsf{blind}}(\mathsf{pp}, \mathsf{ct}_1, \ldots, \mathsf{ct}_n, \mathsf{sp}, \mathsf{sk}_1, \ldots, \mathsf{sk}_n, y, d, \pi)$, and outputs z.

  Each P_i then checks if $\mathcal{H}(y) = d$ and if so computes
  - $\widehat{z}_{i,1} \leftarrow \mathsf{mrNISC.Encode}(E[\mathsf{ct}_1, \ldots, \mathsf{ct}_n, y, d, \pi], \{\mathsf{pk}_i, \widehat{(x_i, r_i)}\}_{i\in[n]}, \mathsf{st}_{i,\mathsf{key}})$
  - $\widehat{z}_{i,2} \leftarrow \mathsf{mrNISC.Encode}(E[\mathsf{ct}_1, \ldots, \mathsf{ct}_n, y, d, \pi], \{\mathsf{pk}_i, \widehat{(x_i, r_i)}\}_{i\in[n]}, \mathsf{st}_{i,\mathsf{inp}})$,
  - and broadcasts $\widehat{z}_{i,1}, \widehat{z}_{i,2}$.
- **Output Reconstruction:** Any party P_i can reconstruct the output by running

$$z \leftarrow \mathsf{mrNISC.Eval}(E[\mathsf{ct}_1, \ldots, \mathsf{ct}_n, y, d, \pi], \{\widehat{z}_{i,1}, \widehat{z}_{i,2}\}_{i\in[n]}).$$

Figure 4: Three-Round MPQC with classical communication from QLWE, in the PKI + QRO model.

- Setup: invoke the mrNISC simulator to obtain honest party commitments $\{\mathsf{pk}_i\}_{i\in H}$, and send them to Adv. Adv sends a circuit D to be computed.
- Input Round: for each $i \in H$, compute $(\mathsf{ct}_i, \mathsf{sk}_i) \leftarrow \mathcal{V}^{\mathsf{inp}}_{\mathsf{blind}}(1^\lambda, 0)$. Invoke the mrNISC simulator to obtain honest party commitments $\{\widehat{(x_i, r_i)}\}_{i\in H}$. Compute $(\mathsf{pp}, \mathsf{sp}) \leftarrow \mathcal{V}^{\mathsf{params}}_{\mathsf{blind}}(1^\lambda, D)$ and invoke the mrNISC

simulator on pp to obtain $\{\widehat{\mathsf{pp}}_i\}_{i\in H}$. Send $\{(\mathsf{ct}_i, \widehat{(x_i, r_i)}, \widehat{\mathsf{pp}}_i)\}_{i\in H}$ to Adv. Receive from the adversary $\{(\mathsf{ct}_i, \widehat{(x_i, r_i)}, \widehat{\mathsf{pp}}_i)\}_{i\in M}$ and invoke the mrNISC simulator on $\{\widehat{(x_i, r_i)}\}_{i\in M}$ to obtain $\{(x_i, r_i)\}_{i\in M}$. Invoke the mrNISC simulator on $\{\widehat{\mathsf{pp}}_i\}_{i\in M}$ and in the case of abort send $\perp$ to the ideal functionality. Otherwise, continue.

- Server Round: receive (y, d, π) from Adv.
- Output Round: first, check that $\mathcal{H}(y) = d$. If not, then abort (send nothing to Adv), and send $\perp$ to the ideal functionality. Otherwise, for each $i \in M$, verify that each ct_i matches $(\mathsf{ct}_i, \mathsf{sk}_i) := \mathcal{V}^{\mathsf{inp}}_{\mathsf{blind}}(1^\lambda, x_i; r_i)$. If not, let $\mathsf{out} = \perp$ and send $\perp$ to the ideal functionality. Otherwise, compute $(v, z) \leftarrow \mathcal{V}^*_{\mathsf{blind}}(\mathsf{pp}, \mathsf{ct}_1, \ldots, \mathsf{ct}_n, \mathsf{sp}, \mathsf{sk}_1, \ldots, \mathsf{sk}_n, y, d, \pi)$ and discard z. If $v = \mathsf{rej}$, let $\mathsf{out} = \perp$ and send $\perp$ to the ideal functionality. Otherwise, if $v = \mathsf{acc}$, then query the ideal functionality with $\{x_i\}_{i\in M}$ to obtain z and let $\mathsf{out} = z$. Finally, invoke the mrNISC simulator on out to obtain $\{\widehat{z}_{i,1}, \widehat{z}_{i,2}\}_{i\in H}$ and forward these to Adv.

Consider the following hybrids.

- $\mathcal{H}_0$: The real distribution $\mathsf{REAL}_{\Pi,\mathsf{Q}}(\mathsf{Adv}_\lambda, \{x_i\}_{i\in M}, |\psi\rangle_{\mathsf{Adv}})$.
- $\mathcal{H}_1$: Simulate the mrNISC. In particular:
  - During Setup, simulate the honest party commitments pk_i, and extract k_i from adversary commitments pk_i.
  - During Server Round, simulate the honest party commitments $\widehat{(x_i, r_i)}$, simulate the honest party second round messages $\widehat{\mathsf{pp}}_i$ using output $\widetilde{D}(k_1, \ldots, k_n)$, and extract (x_i, r_i) from adversary commitments $\widehat{(x_i, r_i)}$.
  - During Output Round, simulate the honest party second round messages $\widehat{z}_{i,1}, \widehat{z}_{i,2}$ using output $E[\mathsf{ct}_1, \ldots, \mathsf{ct}_n, y, d, \pi](k_1, x_1, r_1, \ldots, k_n, x_n, r_n)$.

  This is computationally indistinguishable from $\mathcal{H}_0$ due to the security of mrNISC.
- $\mathcal{H}_2$: Replace $\widetilde{D}(k_1, \ldots, k_n)$ with pp computed as $(\mathsf{pp}, \mathsf{sp}) \leftarrow \mathcal{V}^{\mathsf{params}}_{\mathsf{blind}}(1^\lambda, D)$. This is computationally indistinguishable from $\mathcal{H}_1$ due to the security of the PRF.
- $\mathcal{H}_3$: Alter the computation $E[\mathsf{ct}_1, \ldots, \mathsf{ct}_n, y, d, \pi](k_1, x_1, r_1, \ldots, k_n, x_n, r_n)$ as follows. Compute (v, z) (or $\perp$) as before but discard z. Then, in the case that $\mathcal{H}(y) = d$ and $v = \mathsf{acc}$, directly compute $z \leftarrow D(x_1, \ldots, x_n)$. This is computationally indistinguishable from $\mathcal{H}_2$ due to the semi-malicious soundness of Π_{blind}.
- $\mathcal{H}_4$: During Input Round, compute honest party ct_i as $(\mathsf{ct}_i, \mathsf{sk}_i) \leftarrow \mathcal{V}^{\mathsf{inp}}_{\mathsf{blind}}(1^\lambda, 0)$. This is computationally indistinguishable from $\mathcal{H}_3$ due to the malicious blindness of Π_{blind}. Indeed, the computation of the output z is unchanged, so the adversary could only possibly notice a difference in the ct_i and $v \in \{\mathsf{acc}, \mathsf{rej}\}$.
- $\mathcal{H}_5$: Obtain the output z by querying the ideal functionality on $\{x_i\}_{i\in M}$. In the case of an abort (either $\mathcal{H}(y) \neq d$, some ct_i is malformed, or $\mathcal{V}^*_{\mathsf{blind}}$ outputs rej), send $\perp$ to the ideal functionality. This just moves the computation to the ideal functionality, so is perfectly indistinguishable from $\mathcal{H}_4$. This is the simulator.

□

5.3 Quantum non-interactive secure computation

In this section, we consider a quantum sender $\mathcal{S}$ with classical input $x_{\mathcal{S}}$ and a classical receiver $\mathcal{R}$ with classical input $x_{\mathcal{R}}$. The receiver would like to learn the value $D(x_{\mathcal{S}}, x_{\mathcal{R}})$ for some pseudo-deterministic quantum functionality D.

Note first that the protocol from last section can be used to accomplish this in three rounds, by having the sender implement both the server and one of the clients. However, in this section we show how to accomplish this in just two messages, satisfying the syntax of a (non-reusable) NISC (non-interactive secure computation) protocol.

Ingredients.

- A post-quantum NISC protocol $\mathsf{NISC} = (\mathsf{NISC}_{\mathsf{Gen}}, \mathsf{NISC}_1, \mathsf{NISC}_2, \mathsf{NISC}_{\mathsf{out}})$ (Section 2.4).
- A two-message semi-malicious CVQC protocol with malicious blindness and distributed setup $\Pi_{\mathsf{blind}} = (\mathsf{Gen}, \mathcal{P}_{\mathsf{blind}}, \mathcal{V}^{\mathsf{params}}_{\mathsf{blind}}, \mathcal{V}^{\mathsf{inp}}_{\mathsf{blind}}, \mathcal{V}_{\mathsf{blind}})$ (Protocol 2). Let $\mathcal{V}^{*}_{\mathsf{blind}}$ be the circuit $\mathcal{V}_{\mathsf{blind}}$ with the random oracle check $\mathcal{H}(y) = d$ removed.

Protocol 5: Quantum NISC

Public Information: A two-party pseudo-deterministic quantum functionality $D(\cdot, \cdot)$, security parameter λ, $\mathsf{crs} \leftarrow \mathsf{Gen}(1^{\lambda})$, $\mathsf{NISC.crs} \leftarrow \mathsf{NISC}_{\mathsf{Gen}}(1^{\lambda})$. Let $\mathcal{H}$ be a random oracle, used in Π_{blind}.

- The receiver $\mathcal{R}$ has input $x_{\mathcal{R}}$. It uses random coins $r_{\mathcal{R}}$ to sample outputs in the first two steps below.
  - Sample $(\mathsf{pp}, \mathsf{sp}) \leftarrow \mathcal{V}^{\mathsf{params}}_{\mathsf{blind}}(1^{\lambda}, D)$.
  - Sample $(\mathsf{ct}_{\mathcal{R}}, \mathsf{sk}_{\mathcal{R}}) \leftarrow \mathcal{V}^{\mathsf{inp}}_{\mathsf{blind}}(1^{\lambda}, \mathsf{crs}, x_{\mathcal{R}})$.
  - Compute $(m_{\mathcal{R}}, \mathsf{st}) \leftarrow \mathsf{NISC}_1(\mathsf{NISC.crs}, (x_{\mathcal{R}}, r_{\mathcal{R}}))$.
  - Send $(\mathsf{pp}, \mathsf{ct}_{\mathcal{R}}, m_{\mathcal{R}})$ to the sender $\mathcal{S}$.
- The sender $\mathcal{S}$ has input $x_{\mathcal{S}}$. It uses random coins $r_{\mathcal{S}}$ to sample output in the first step below.
  - Sample $(\mathsf{ct}_{\mathcal{S}}, \mathsf{sk}_{\mathcal{S}}) \leftarrow \mathcal{V}^{\mathsf{inp}}_{\mathsf{blind}}(1^{\lambda}, \mathsf{crs}, x_{\mathcal{S}})$.
  - Compute $(y, d, \pi) \leftarrow \mathcal{P}_{\mathsf{blind}}(\mathsf{pp}, \mathsf{ct}_{\mathcal{R}}, \mathsf{ct}_{\mathcal{S}})$.
  - Compute $m_{\mathcal{S}} \leftarrow \mathsf{NISC}_2(\mathsf{NISC.crs}, C[\mathsf{pp}, \mathsf{ct}_{\mathcal{R}}, \mathsf{ct}_{\mathcal{S}}, y, d, \pi], m_{\mathcal{R}}, (x_{\mathcal{S}}, r_{\mathcal{S}}))$, where the circuit $C[\mathsf{pp}, \mathsf{ct}_{\mathcal{R}}, \mathsf{ct}_{\mathcal{S}}, y, d, \pi]$ does the following.
    - ∗ Take as input strings $(x_{\mathcal{R}}, x_{\mathcal{S}})$ and random coins $(r_{\mathcal{R}}, r_{\mathcal{S}})$ and first verify that $(\mathsf{pp}, \mathsf{ct}_{\mathcal{R}}, \mathsf{ct}_{\mathcal{S}})$ are well-formed.
    - ∗ If so, output $\mathcal{V}^{*}_{\mathsf{blind}}(\mathsf{pp}, \mathsf{ct}_{\mathcal{R}}, \mathsf{ct}_{\mathcal{S}}, \mathsf{sp}, \mathsf{sk}_{\mathcal{R}}, \mathsf{sk}_{\mathcal{S}}, y, d, \pi)$, and otherwise output $(\mathsf{rej}, \bot)$.
  - Send $(\mathsf{ct}_{\mathcal{S}}, y, d, \pi, m_{\mathcal{S}})$ to the receiver.
- The receiver $\mathcal{R}$ first checks that $\mathcal{H}(y) = d$ and if so computes and outputs $\mathsf{NISC}_{\mathsf{out}}(\mathsf{st}, C[\mathsf{pp}, \mathsf{ct}_{\mathcal{R}}, \mathsf{ct}_{\mathcal{S}}, y, d, \pi], m_{\mathcal{S}})$, and otherwise outputs $\bot$.

Figure 5: Quantum NISC with a classical receiver, from QLWE in the QROM.

Theorem 5.4. *Protocol 5 satisfies Definition 2.6.*

Proof. First consider the case where the adversary Adv is corrupting the sender. We define the simulator Sim below.

$\mathsf{Sim}(|\psi\rangle_{\mathsf{Adv}})$:

- Invoke the NISC simulator to sample crs.
- Compute an honest receiver's message $(\mathsf{pp}, \mathsf{ct}_{\mathcal{R}}, m_{\mathcal{R}})$ except that $\mathsf{ct}_{\mathcal{R}}$ is sampled as $\mathcal{V}^{\mathsf{inp}}_{\mathsf{blind}}(1^\lambda, 0)$, and $m_{\mathcal{R}}$ is computed by the classical NISC simulator. Send this message to Adv.
- When Adv returns $(\mathsf{ct}_{\mathcal{S}}, y, d, \pi, m_{\mathcal{S}})$, use the classical NISC simulator to extract input $(x_{\mathcal{S}}, r_{\mathcal{S}})$ from $m_{\mathcal{S}}$. Check that $\mathcal{H}(y) = d$ and if not, send $\perp$ to the ideal functionality. Otherwise, check that $\mathsf{ct}_{\mathcal{S}}$ is well-formed with respect to $x_{\mathcal{S}}, r_{\mathcal{S}}$ and if the check does not pass, send $\perp$ to the ideal functionality. Otherwise, compute $(v, z) \leftarrow \mathcal{V}^*_{\mathsf{blind}}(\mathsf{pp}, \mathsf{ct}_{\mathcal{R}}, \mathsf{ct}_{\mathcal{S}}, \mathsf{sp}, \mathsf{sk}_{\mathcal{R}}, \mathsf{sk}_{\mathcal{S}}, y, d, \pi)$ and discard z. If $v = \mathsf{rej}$, send $\perp$ to the ideal functionality. Otherwise, if $v = \mathsf{acc}$, then query the ideal functionality with $x_{\mathcal{S}}$.

Now consider the following hybrids.

- $\mathcal{H}_0$: The real distribution.
- $\mathcal{H}_1$: Simulate the classical NISC. That is, simulate the crs, extract the adversary's input $(x_{\mathcal{S}}, r_{\mathcal{S}})$ from $m_{\mathcal{S}}$, and set the receiver's output by computing $C[\mathsf{pp}, \mathsf{ct}_{\mathcal{R}}, \mathsf{ct}_{\mathcal{S}}, y, d, \pi]$ on the extracted input. This is indistinguishable from $\mathcal{H}_0$ due to the security of classical NISC.
- $\mathcal{H}_2$: Change how $C[\mathsf{pp}, \mathsf{ct}_{\mathcal{R}}, \mathsf{ct}_{\mathcal{S}}, y, d, \pi]$ is computed. First check that $\mathcal{H}(y) = d$ and that $\mathsf{ct}_{\mathcal{S}}$ is well-formed, and if so, compute $(v, z) \leftarrow \mathcal{V}^*_{\mathsf{blind}}(\mathsf{pp}, \mathsf{ct}_{\mathcal{R}}, \mathsf{ct}_{\mathcal{S}}, \mathsf{sp}, \mathsf{sk}_{\mathcal{R}}, \mathsf{sk}_{\mathcal{S}}, y, d, \pi)$ and discard z. If $\mathcal{H}(y) \neq d$, $x_{\mathcal{S}}$ was not well-formed, or there was a reject, output $(\mathsf{rej}, \perp)$. Otherwise, compute $z \leftarrow D(x_{\mathcal{R}}, x_{\mathcal{S}})$ and output (acc, z). This is indistinguishable from $\mathcal{H}_1$ due to the semi-malicious soundness of Π_{blind}.
- $\mathcal{H}_3$: Switch $\mathsf{ct}_{\mathcal{R}}$ to be sampled as $\mathcal{V}^{\mathsf{inp}}_{\mathsf{blind}}(1^\lambda, 0)$. This is indistinguishable from $\mathcal{H}_2$ due to the semi-malicious malicious blindness of Π_{blind}.
- $\mathcal{H}_4$: Query the ideal functionality as described in the simulator. This is equivalent to $\mathcal{H}_3$.

Next, consider the case where the adversary Adv is corrupting the receiver. We define the simulator Sim below.

$\mathsf{Sim}(|\psi\rangle_{\mathsf{Adv}})$:

- Invoke the NISC simulator to sample crs.
- Receive $(\mathsf{pp}, \mathsf{ct}_{\mathcal{R}}, m_{\mathcal{R}})$ from Adv, and use the classical NISC simulator to extract input $(x_{\mathcal{R}}, r_{\mathcal{R}})$.
- Sample $\mathsf{ct}_{\mathcal{S}}$ as $\mathcal{V}^{\mathsf{inp}}_{\mathsf{blind}}(1^\lambda, 0)$, and then sample (y, d, π) as in the honest protocol.
- Check that pp and $\mathsf{ct}_{\mathcal{R}}$ are well-formed with respect to $x_{\mathcal{R}}, r_{\mathcal{R}}$. If the check does not pass, invoke the classical NISC simulator on circuit $C[\mathsf{pp}, \mathsf{ct}_{\mathcal{R}}, \mathsf{ct}_{\mathcal{S}}, y, d, \pi]$ and output $(\mathsf{rej}, \perp)$ to obtain $m_{\mathcal{S}}$. Otherwise compute $(v, z) \leftarrow \mathcal{V}^*_{\mathsf{blind}}(\mathsf{pp}, \mathsf{ct}_{\mathcal{R}}, \mathsf{ct}_{\mathcal{S}}, \mathsf{sp}, \mathsf{sk}_{\mathcal{R}}, \mathsf{sk}_{\mathcal{S}}, y, d, \pi)$ and discard z. If $v = \mathsf{rej}$, invoke the classical NISC simulator on circuit $C[\mathsf{pp}, \mathsf{ct}_{\mathcal{R}}, \mathsf{ct}_{\mathcal{S}}, y, d, \pi]$ and output $(\mathsf{rej}, \perp)$ to obtain $m_{\mathcal{S}}$. Otherwise, send $x_{\mathcal{R}}$ to the ideal functionality, receive y, and invoke the classical NISC simulator on circuit $C[\mathsf{pp}, \mathsf{ct}_{\mathcal{R}}, \mathsf{ct}_{\mathcal{S}}, y, d, \pi]$ and output (acc, z) to obtain $m_{\mathcal{S}}$. Send $(\mathsf{ct}_{\mathcal{S}}, y, d, \pi, m_{\mathcal{S}})$ to Adv.

Now consider the following hybrids.

- $\mathcal{H}_0$: The real distribution.

- $\mathcal{H}_1$: Simulate the classical NISC. That is, simulate the crs, extract the adversary's input $(x_{\mathcal{R}}, r_{\mathcal{R}})$, and simulate $m_{\mathcal{S}}$ by computing the circuit $C[\mathsf{pp}, \mathsf{ct}_{\mathcal{R}}, \mathsf{ct}_{\mathcal{S}}, y, d, \pi]$ and feeding the output into the NISC simulator. This is indistinguishable from $\mathcal{H}_0$ due to the security of classical NISC.

- $\mathcal{H}_2$: Change how $C[\mathsf{pp}, \mathsf{ct}_{\mathcal{R}}, \mathsf{ct}_{\mathcal{S}}, y, d, \pi]$ is computed. First check that pp and $\mathsf{ct}_{\mathcal{R}}$ are well-formed with respect to $x_{\mathcal{R}}, r_{\mathcal{R}}$. If the check does not pass, feed $(\mathsf{rej}, \bot)$ into the NISC simulator. Otherwise compute $(v, z) \leftarrow \mathcal{V}^*_{\mathsf{blind}}(\mathsf{pp}, \mathsf{ct}_{\mathcal{R}}, \mathsf{ct}_{\mathcal{S}}, \mathsf{sp}, \mathsf{sk}_{\mathcal{R}}, \mathsf{sk}_{\mathcal{S}}, y, d, \pi)$ and discard z. If $v = \mathsf{rej}$, feed $(\mathsf{rej}, \bot)$ into the NISC simulator. Otherwise, compute $z \leftarrow D(x_{\mathcal{R}}, x_{\mathcal{S}})$ and feed (acc, z) into the NISC simulator. This is indistinguishable from $\mathcal{H}_1$ due to the semi-malicious correctness of Π_{blind}.

- $\mathcal{H}_3$: Switch $\mathsf{ct}_{\mathcal{S}}$ to be sampled as $\mathcal{V}^{\mathsf{inp}}_{\mathsf{blind}}(1^\lambda, 0)$. This is indistinguishable from $\mathcal{H}_2$ due to the malicious blindness of Π_{blind}.

- $\mathcal{H}_4$: Query the ideal functionality as described in the simulator. This is equivalent to $\mathcal{H}_3$.

□

Acknowledgements

Thank you to Dakshita Khurana and Giulio Malavolta for helpful discussions.

References

[ABG+21] Amit Agarwal, James Bartusek, Vipul Goyal, Dakshita Khurana, and Giulio Malavolta. Post-quantum multi-party computation. *Eurocrypt*, 2021.

[ACC+20] Bar Alon, Hao Chung, Kai-Min Chung, Mi-Ying Huang, Yi Lee, and Yu-Ching Shen. Round efficient secure multiparty quantum computation with identifiable abort. Cryptology ePrint Archive, Report 2020/1464, 2020. https://eprint.iacr.org/2020/1464.

[ACGH20] Gorjan Alagic, Andrew M. Childs, Alex B. Grilo, and Shih-Han Hung. Non-interactive classical verification of quantum computation. LNCS, pages 153–180. Springer, Heidelberg, March 2020.

[AJJM21] Prabhanjan Ananth, Abhishek Jain, Zhengzhong Jin, and Giulio Malavolta. Unbounded multi-party computation from learning with errors. *Eurocrypt*, 2021.

[AJL+12] Gilad Asharov, Abhishek Jain, Adriana López-Alt, Eran Tromer, Vinod Vaikuntanathan, and Daniel Wichs. Multiparty computation with low communication, computation and interaction via threshold FHE. In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 483–501. Springer, Heidelberg, April 2012.

[AMPR14] Arash Afshar, Payman Mohassel, Benny Pinkas, and Ben Riva. Non-interactive secure computation based on cut-and-choose. In Phong Q. Nguyen and Elisabeth Oswald, editors, *EUROCRYPT 2014*, volume 8441 of *LNCS*, pages 387–404. Springer, Heidelberg, May 2014.

[BCG+06] Michael Ben-Or, Claude Crépeau, Daniel Gottesman, Avinatan Hassidim, and Adam Smith. Secure multiparty quantum computation with (only) a strict honest majority. In *47th FOCS*, pages 249–260. IEEE Computer Society Press, October 2006.

[BCKM21a] James Bartusek, Andrea Coladangelo, Dakshita Khurana, and Fermi Ma. On the round complexity of secure quantum computation. *CRYPTO*, 2021.

[BCKM21b] James Bartusek, Andrea Coladangelo, Dakshita Khurana, and Fermi Ma. One-way functions imply secure computation in a quantum world. *CRYPTO*, 2021.

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

[BGI+17] Saikrishna Badrinarayanan, Sanjam Garg, Yuval Ishai, Amit Sahai, and Akshay Wadia. Two-message witness indistinguishability and secure computation in the plain model from new assumptions. In Tsuyoshi Takagi and Thomas Peyrin, editors, *ASIACRYPT 2017, Part III*, volume 10626 of *LNCS*, pages 275–303. Springer, Heidelberg, December 2017.

[BGW88] Michael Ben-Or, Shafi Goldwasser, and Avi Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computation (extended abstract). In *20th ACM STOC*, pages 1–10. ACM Press, May 1988.

[BJKL21] Fabrice Benhamouda, Aayush Jain, Ilan Komargodski, and Huijia Lin. Multi-party reusable non-interactive secure computation from lwe. *Eurocrypt*, 2021.

[BL20] Fabrice Benhamouda and Huijia Lin. Mr NISC: Multiparty reusable non-interactive secure computation. In *TCC 2020, Part II*, LNCS, pages 349–378. Springer, Heidelberg, March 2020.

[BM21] James Bartusek and Giulio Malavolta. Candidate obfuscation of null quantum circuits and witness encryption for qma. Cryptology ePrint Archive, Report 2021/421, 2021. https://eprint.iacr.org/2021/421.

[Bra18] Zvika Brakerski. Quantum FHE (almost) as secure as classical. In Hovav Shacham and Alexandra Boldyreva, editors, *CRYPTO 2018, Part III*, volume 10993 of *LNCS*, pages 67–95. Springer, Heidelberg, August 2018.

[CCD88] David Chaum, Claude Crépeau, and Ivan Damgård. Multiparty unconditionally secure protocols (abstract) (informal contribution). In Carl Pomerance, editor, *CRYPTO'87*, volume 293 of *LNCS*, page 462. Springer, Heidelberg, August 1988.

[CCKM20] Michele Ciampi, Alexandru Cojocaru, Elham Kashefi, and Atul Mantri. Secure quantum two-party computation: Impossibility and constructions. Cryptology ePrint Archive, Report 2020/1286, 2020. https://eprint.iacr.org/2020/1286.

[CCY20] Nai-Hui Chia, Kai-Min Chung, and Takashi Yamakawa. Classical verification of quantum computations with efficient verifier. LNCS, pages 181–206. Springer, Heidelberg, March 2020.

[CDI+19] Melissa Chase, Yevgeniy Dodis, Yuval Ishai, Daniel Kraschewski, Tianren Liu, Rafail Ostrovsky, and Vinod Vaikuntanathan. Reusable non-interactive secure computation. In Alexandra Boldyreva and Daniele Micciancio, editors, *CRYPTO 2019, Part III*, volume 11694 of *LNCS*, pages 462–488. Springer, Heidelberg, August 2019.

[CDM20] Orestis Chardouvelis, Nico Doettling, and Giulio Malavolta. Rate-1 secure function evaluation for bqp. Cryptology ePrint Archive, Report 2020/1454, 2020. https://eprint.iacr.org/2020/1454.

[CGS02] Claude Crépeau, Daniel Gottesman, and Adam Smith. Secure multi-party quantum computation. In *34th ACM STOC*, pages 643–652. ACM Press, May 2002.

[CLLW20] Kai-Min Chung, Yi Lee, Han-Hsuan Lin, and Xiaodi Wu. Constant-round blind classical verification of quantum sampling, 2020.

[CVZ20] Andrea Coladangelo, Thomas Vidick, and Tina Zhang. Non-interactive zero-knowledge arguments for qma, with preprocessing. In Daniele Micciancio and Thomas Ristenpart, editors, *Advances in Cryptology – CRYPTO 2020*, pages 799–828, Cham, 2020. Springer International Publishing.

[DFMS19] Jelle Don, Serge Fehr, Christian Majenz, and Christian Schaffner. Security of the Fiat-Shamir transformation in the quantum random-oracle model. In Alexandra Boldyreva and Daniele Micciancio, editors, *CRYPTO 2019, Part II*, volume 11693 of *LNCS*, pages 356–383. Springer, Heidelberg, August 2019.

[DFPR14] Vedran Dunjko, Joseph Fitzsimons, Christopher Portmann, and Renato Renner. Composable security of delegated quantum computation. In Palash Sarkar and Tetsu Iwata, editors, *ASIACRYPT 2014, Part II*, volume 8874 of *LNCS*, pages 406–425. Springer, Heidelberg, December 2014.

[DGJ$^+$20] Yfke Dulek, Alex B. Grilo, Stacey Jeffery, Christian Majenz, and Christian Schaffner. Secure multi-party quantum computation with a dishonest majority. In Anne Canteaut and Yuval Ishai, editors, *EUROCRYPT 2020, Part III*, volume 12107 of *LNCS*, pages 729–758. Springer, Heidelberg, May 2020.

[DNS12] Frédéric Dupuis, Jesper Buus Nielsen, and Louis Salvail. Actively secure two-party evaluation of any quantum operation. In Reihaneh Safavi-Naini and Ran Canetti, editors, *CRYPTO 2012*, volume 7417 of *LNCS*, pages 794–811. Springer, Heidelberg, August 2012.

[FHM18] Joseph F. Fitzsimons, Michal Hajdusek, and Tomoyuki Morimae. Post hoc verification of quantum computation. *Phys. Rev. Lett.*, 120:040501, Jan 2018.

[FK17] Joseph F. Fitzsimons and Elham Kashefi. Unconditionally verifiable blind quantum computation. *Phys. Rev. A*, 96:012303, Jul 2017.

[GLSV21] Alex B. Grilo, Huijia Lin, Fang Song, and Vinod Vaikuntanathan. Oblivious transfer is in miniqcrypt. *Eurocrypt*, 2021.

[GMW87] Oded Goldreich, Silvio Micali, and Avi Wigderson. How to play any mental game or A completeness theorem for protocols with honest majority. In Alfred Aho, editor, *19th ACM STOC*, pages 218–229. ACM Press, May 1987.

[Gol04] Oded Goldreich. *The Foundations of Cryptography - Volume 2, Basic Applications*. Cambridge University Press, 2004.

[Goy18] Rishab Goyal. Quantum multi-key homomorphic encryption for polynomial-sized circuits. Cryptology ePrint Archive, Report 2018/443, 2018. https://eprint.iacr.org/2018/443.

[GS18] Sanjam Garg and Akshayaram Srinivasan. Two-round multiparty secure computation from minimal assumptions. In Jesper Buus Nielsen and Vincent Rijmen, editors, *EUROCRYPT 2018, Part II*, volume 10821 of *LNCS*, pages 468–499. Springer, Heidelberg, April / May 2018.

[GV19] Alexandru Gheorghiu and Thomas Vidick. Computationally-secure and composable remote state preparation. In David Zuckerman, editor, *60th FOCS*, pages 1024–1033. IEEE Computer Society Press, November 2019.

[IKO$^+$11] Yuval Ishai, Eyal Kushilevitz, Rafail Ostrovsky, Manoj Prabhakaran, and Amit Sahai. Efficient non-interactive secure computation. In Kenneth G. Paterson, editor, *EUROCRYPT 2011*, volume 6632 of *LNCS*, pages 406–425. Springer, Heidelberg, May 2011.

[IPS08] Yuval Ishai, Manoj Prabhakaran, and Amit Sahai. Founding cryptography on oblivious transfer - efficiently. In David Wagner, editor, *CRYPTO 2008*, volume 5157 of *LNCS*, pages 572–591. Springer, Heidelberg, August 2008.

[KKMO21] Theodoros Kapourniotis, Elham Kashefi, Luka Music, and Harold Ollivier. Delegating multiparty quantum computations vs. dishonest majority in two quantum rounds, 2021.

[KP17] Elham Kashefi and Anna Pappa. Multiparty delegated quantum computing. *Cryptography*, 1:12, 07 2017.

[LZ19] Qipeng Liu and Mark Zhandry. Revisiting post-quantum Fiat-Shamir. In Alexandra Boldyreva and Daniele Micciancio, editors, *CRYPTO 2019, Part II*, volume 11693 of *LNCS*, pages 326–355. Springer, Heidelberg, August 2019.

[Mah18a] Urmila Mahadev. Classical homomorphic encryption for quantum circuits. In Mikkel Thorup, editor, *59th FOCS*, pages 332–338. IEEE Computer Society Press, October 2018.

[Mah18b] Urmila Mahadev. Classical verification of quantum computations. In Mikkel Thorup, editor, *59th FOCS*, pages 259–267. IEEE Computer Society Press, October 2018.

[Mor14] Tomoyuki Morimae. Verification for measurement-only blind quantum computing. *Phys. Rev. A*, 89:060302, Jun 2014.

[MPP20] Andrew Morgan, Rafael Pass, and Antigoni Polychroniadou. Succinct non-interactive secure computation. In Anne Canteaut and Yuval Ishai, editors, *EUROCRYPT 2020, Part II*, volume 12106 of *LNCS*, pages 216–245. Springer, Heidelberg, May 2020.

[MW16] Pratyay Mukherjee and Daniel Wichs. Two round multiparty computation via multi-key FHE. In Marc Fischlin and Jean-Sébastien Coron, editors, *EUROCRYPT 2016, Part II*, volume 9666 of *LNCS*, pages 735–763. Springer, Heidelberg, May 2016.

[MY21] Tomoyuki Morimae and Takashi Yamakawa. Classically verifiable (dual-mode) nizk for qma with preprocessing, 2021.

[PS19] Chris Peikert and Sina Shiehian. Noninteractive zero knowledge for NP from (plain) learning with errors. In Alexandra Boldyreva and Daniele Micciancio, editors, *CRYPTO 2019, Part I*, volume 11692 of *LNCS*, pages 89–114. Springer, Heidelberg, August 2019.

[PVW08] Chris Peikert, Vinod Vaikuntanathan, and Brent Waters. A framework for efficient and composable oblivious transfer. In David Wagner, editor, *CRYPTO 2008*, volume 5157 of *LNCS*, pages 554–571. Springer, Heidelberg, August 2008.

[Shm20] Omri Shmueli. Multi-theorem (malicious) designated-verifier nizk for qma, 2020.

[Unr10] Dominique Unruh. Universally composable quantum multi-party computation. In Henri Gilbert, editor, *EUROCRYPT 2010*, volume 6110 of *LNCS*, pages 486–505. Springer, Heidelberg, May / June 2010.

[Yao86] Andrew Chi-Chih Yao. How to generate and exchange secrets. In *FOCS*, 1986.

A Recent Work on Two-Message Secure Quantum Computation

Many recent works have considered maliciously-secure quantum protocols that satisfy the following interaction pattern. First, a common random string (crs) is sampled by a trusted party. Then, one party (the *receiver* or *verifier*) sends a message to the other party (the *prover* or *sender*), who responds with a message of their own. Finally, the receiver/verifier computes some output.

If the functionality these parties are computing is some quantum circuit Q with single-bit output, and only the prover has a (quantum) input $|\psi\rangle$, then this constitutes a *malicious designated-verifier non-interactive zero-knowledge protocol for QMA*, which we call MDV-NIZK for QMA. More generally, if both parties have an input, then this constitutes a *quantum non-interactive secure computation protocol*, or Q-NISC.

The following additional distinctions can also be drawn. First, the verifier/receiver's message may either be *one-time* or *reusable*. Reusability requires that a malicious prover cannot recover information about the verifier's inputs, or cause an incorrect output, even if they use the same first message across multiple computations, while observing whether or not the verifier rejects their messages. Next, we can distinguish between whether the verifier is quantum or fully classical. The following table outlines the eight settings that one could consider along these lines.

Table 1: Secure Two-Message Quantum Computation

Functionality	Reusability	Verifier	First constructed by	Weakest assumption
MDV-NIZK	One-time	Quantum	[CVZ20]	OT[8] [BCKM21a]
MDV-NIZK	One-time	Classical	[ACGH20]	QLWE in QROM [ACGH20]
MDV-NIZK	Reusable	Quantum	[Shm20]	OT + MDV-NIZK for NP [BCKM21a]
MDV-NIZK	Reusable	Classical	Open	
Q-NISC	One-time	Quantum	[BCKM21a]	OT [BCKM21a]
Q-NISC	One-time	Classical	**This work**	QLWE in QROM
Q-NISC	Reusable	Quantum	[BCKM21a]	OT + MDV-NIZK for NP [BCKM21a]
Q-NISC	Reusable	Classical	Open[9]	

B Completing the Proof of Lemma 3.2

Consider the following single-copy prover $\mathcal{P}^*$, which is initialized with $q(\lambda)^4$ copies of the multiple-copy prover's initial state $|\psi_{\mathsf{init}}\rangle$. $\mathcal{P}^*$ interacts with the single-copy verifier $\mathcal{V}^*$. The following describes how $\mathcal{P}^*$ prepares its intermediate state $|\phi_{\mathsf{pk}}\rangle$.

1. $\mathcal{P}^*$ receives pk^* from its verifier and sets $\mathsf{pk}_i := \mathsf{pk}^*$. It also samples $(\mathsf{pk}_j, \mathsf{sk}_j) \leftarrow \mathcal{V}^*(1^\lambda)$ for each $j \in [n] \setminus \{i\}$, and sets $\mathsf{pk} := (\mathsf{pk}_1, \dots, \mathsf{pk}_n)$.
2. It tries the following for at most $q(\lambda)^4$ steps:
   - Prepare the state $|\psi_{\mathsf{pk}}\rangle$ using a copy $|\psi_{\mathsf{init}}\rangle$ and the public keys pk, and then apply V_{d_2}.
   - Apply the measurement determined by $\Pi_{\mathsf{pk},\mathsf{sk},d_2}$, which is possible because it does not require sk_i due to condition 1 on the CVQC protocol, and the fact that $d_{2,i} = 0$.
   - If the measurement rejects, go back to step 1. Otherwise apply $V_{d_2}^\dagger$, output the resulting state $|\phi_{\mathsf{pk}}\rangle$, and terminate.
3. If $\mathcal{P}^*$ has not terminated, then prepare and output a dummy state $|\phi^*_{\mathsf{pk}}\rangle$ that always passes the test round.

Note that if $\mathcal{P}^*$ terminated during the second step, then its output state is

$$|\phi_{\mathsf{pk}}\rangle := \frac{\Pi^{V_{d_2}}_{\mathsf{pk},\mathsf{sk},d_2} |\psi_{\mathsf{pk}}\rangle}{\|\Pi^{V_{d_2}}_{\mathsf{pk},\mathsf{sk},d_2} |\psi_{\mathsf{pk}}\rangle\|}.$$

[8] Here, OT refers to post-quantum fully-simulatable two-message OT.

[9] This would have implications to (heuristic) obfuscation of quantum circuits, though there appear to be significant barriers to achieving this with known CVQC techniques (see discussion in [BM21]).

$\mathcal{P}^*$ then measures its intermediate state to produce its commitment $y = (y_1, \ldots, y_n)$, and sends y_i to the verifier $\mathcal{V}^*$. The verifier then samples a challenge $c \in \{0, 1\}$. If $c = 0$, the prover applies V_{d_2} (if its intermediate state was $|\phi_{\mathsf{pk}}\rangle$) or $\mathbb{I}$ (if its intermediate state was the dummy state $|\phi^*_{\mathsf{pk}}\rangle$). If $c = 1$, the prover applies V_{d_1}. In both cases the prover then measures to produce $\pi = (\pi_1, \ldots, \pi_n)$ and returns π_i to the verifier.

Note that, by definition, $\mathcal{P}^*$ always passes the test round. Now consider what happens in the Hadamard round. Let Ω be a set of keys $(\mathsf{pk}, \mathsf{sk})$ for which $\mathcal{P}^*$ has an overwhelming probability of terminating in step 2 above. That is, define

$$\Omega := \{(\mathsf{pk}, \mathsf{sk}) : \langle\psi_{\mathsf{pk}}| \Pi^{V_{d_2}}_{\mathsf{pk},\mathsf{sk},d_2} |\psi_{\mathsf{pk}}\rangle\} > 1/q(\lambda)^2,$$

and note that for any such key pair, the probability that $\mathcal{P}^*$ does not terminate is $(1 - 1/q(\lambda)^2)^{q(\lambda)^4} \leq e^{-q(\lambda)^2} = \text{negl}(\lambda)$.

Now we can write

$$\begin{aligned}
&\mathop{\mathbb{E}}_{\mathsf{pk},\mathsf{sk}} \left[\langle\psi_{\mathsf{pk}}| \Pi^{V_{d_2}}_{\mathsf{pk},\mathsf{sk},d_2} V^\dagger_{d_1} \Pi^{(i)}_{\mathsf{pk},\mathsf{sk}} V_{d_1} \Pi^{V_{d_2}}_{\mathsf{pk},\mathsf{sk},d_2} |\psi_{\mathsf{pk}}\rangle\right] \\
&\quad = \sum_{(\mathsf{pk},\mathsf{sk})\in\Omega} \langle\psi_{\mathsf{pk}}| \Pi^{V_{d_2}}_{\mathsf{pk},\mathsf{sk},d_2} V^\dagger_{d_1} \Pi^{(i)}_{\mathsf{pk},\mathsf{sk}} V_{d_1} \Pi^{V_{d_2}}_{\mathsf{pk},\mathsf{sk},d_2} |\psi_{\mathsf{pk}}\rangle \Pr[(\mathsf{pk},\mathsf{sk})] \\
&\qquad + \sum_{(\mathsf{pk},\mathsf{sk})\notin\Omega} \langle\psi_{\mathsf{pk}}| \Pi^{V_{d_2}}_{\mathsf{pk},\mathsf{sk},d_2} V^\dagger_{d_1} \Pi^{(i)}_{\mathsf{pk},\mathsf{sk}} V_{d_1} \Pi^{V_{d_2}}_{\mathsf{pk},\mathsf{sk},d_2} |\psi_{\mathsf{pk}}\rangle \Pr[(\mathsf{pk},\mathsf{sk})] \\
&\quad \leq \sum_{(\mathsf{pk},\mathsf{sk})\in\Omega} \langle\psi_{\mathsf{pk}}| \Pi^{V_{d_2}}_{\mathsf{pk},\mathsf{sk},d_2} V^\dagger_{d_1} \Pi^{(i)}_{\mathsf{pk},\mathsf{sk}} V_{d_1} \Pi^{V_{d_2}}_{\mathsf{pk},\mathsf{sk},d_2} |\psi_{\mathsf{pk}}\rangle \Pr[(\mathsf{pk},\mathsf{sk})] + 1/q(\lambda)^2,
\end{aligned}$$

which implies that

$$\sum_{(\mathsf{pk},\mathsf{sk})\in\Omega} \langle\psi_{\mathsf{pk}}| \Pi^{V_{d_2}}_{\mathsf{pk},\mathsf{sk},d_2} V^\dagger_{d_1} \Pi^{(i)}_{\mathsf{pk},\mathsf{sk}} V_{d_1} \Pi^{V_{d_2}}_{\mathsf{pk},\mathsf{sk},d_2} |\psi_{\mathsf{pk}}\rangle \Pr[(\mathsf{pk},\mathsf{sk})] \geq 1/q(\lambda) - 1/q(\lambda)^2 = 1/\text{poly}(\lambda).$$

However, the left-hand side is at most $\text{negl}(\lambda)$ greater than $\mathcal{P}^*$'s probability of success in the Hadamard round, since for each $(\mathsf{pk}, \mathsf{sk}) \in \Omega$, the prover's intermediate state is $|\phi_{\mathsf{pk}}\rangle$ defined above, except with $\text{negl}(\lambda)$ probability. This is a contradiction, since $\mathcal{P}^*$'s probability of success in the Hadamard round must be $\text{negl}(\lambda)$, due to the fact that it always passes the test round, and condition 2 on the CVQC protocol.

C MPC for Classical Reactive Functionalities

Define a d-level n-party randomized functionality $\mathcal{F} = (\mathcal{F}_1, \ldots, \mathcal{F}_d)$ as follows. Let r be random coins. Then each $\mathcal{F}_j$ can be defined as

$$\left(y_1^{(j)}, \ldots, y_n^{(j)}\right) := \mathcal{F}_j\left(\left(x_1^{(1)}, \ldots, x_1^{(j)}\right), \ldots, \left(x_n^{(1)}, \ldots, x_n^{(j)}\right), w^{(j)}, r\right),$$

where $\left(x_i^{(1)}, \ldots, x_i^{(d)}\right)$ are the set of party i's private inputs, $(w^{(1)}, \ldots, w^{(d)})$ are some public inputs, and $\left(y_i^{(1)}, \ldots, y_i^{(d)}\right)$ are the set of party i's outputs. We allow $x_i^{(j)}$ to be an arbitrary function of $y_i^{(1)}, \ldots, y_i^{(j-2)}$.

Define an ideal functionality $\mathcal{I}_\mathcal{F}$ for computing $\mathcal{F}$ in $d+1$ rounds. Let $\mathcal{H} \subset [n]$ be a subset of honest parties and $\mathcal{M} := [n] \setminus \mathcal{H}$ be the corresponding subset of malicious parties. $\mathcal{I}_\mathcal{F}$ is initialized with honest party inputs $\left\{x_i^{(1)}\right\}_{i\in\mathcal{H}}$.

- In round 1, accept and store private inputs $\left\{x_i^{(1)}\right\}_{i\in\mathcal{M}}$.

- In round j, for $j \in \{2, \ldots, d\}$, do the following. Accept public input $w^{(j-1)}$ and compute

$$\left(y_1^{(j-1)}, \ldots, y_n^{(j-1)}\right) \coloneqq \mathcal{F}_j\left(\left(x_1^{(1)}, \ldots, x_1^{(j-1)}\right), \ldots, \left(x_n^{(1)}, \ldots, x_n^{(j-1)}\right), w^{(j-1)}, r\right).$$

  Output $\left\{y_i^{(j-1)}\right\}_{i \in \mathcal{M}}$. Accept either $\left(\mathsf{ok}, \left\{x_i^{(j)}\right\}_{i \in \mathcal{M}}\right)$ or abort as input. If ok, set level $j-1$ honest party outputs to $\left\{y_i^{(j-1)}\right\}_{i \in \mathcal{H}}$ and compute the next set of honest party inputs $\left\{x_i^{(j)}\right\}_{i \in \mathcal{H}}$. If abort, set honest party outputs to $\perp$ for each level $k \geq j-1$.

- In round $d+1$, accept public input $w^{(d)}$ and compute

$$\left(y_1^{(d)}, \ldots, y_n^{(d)}\right) \coloneqq \mathcal{F}_j\left(\left(x_1^{(1)}, \ldots, x_1^{(d)}\right), \ldots, \left(x_n^{(1)}, \ldots, x_n^{(d)}\right), w^{(d)}, r\right).$$

  Output $\left\{y_i^{(d)}\right\}_{i \in \mathcal{M}}$. Accept either ok or abort as input. If ok, set level d honest party outputs to $\left\{y_i^{(d)}\right\}_{i \in \mathcal{H}}$. If abort, set level d honest party outputs to $\perp$.

As observed in [BCKM21a], the protocol of [GS18] can be used to implement this ideal functionality for any d-level n-party functionality (with post-quantum security), assuming post-quantum oblivious transfer. The required post-quantum oblivious transfer protocol can be constructed from QLWE [PVW08].