Selectively Blind Quantum Computation

Abbas Poshtvan[1,*], Oleksandra Lapiha[2,*,†], Mina Doosti[1], Dominik Leichtle[1], Luka Music[3], and Elham Kashefi[1,4]

[1]School of Informatics, University of Edinburgh, UK
[2]Royal Holloway, University of London, UK
[3]Quandela, Massy, France
[4]LIP6, CNRS, Sorbonne Université, Paris, France

Abstract

Known protocols for the secure delegation of quantum computations from a client to a server in an information-theoretic setting require quantum communication. In this work, we investigate methods to reduce the communication overhead. First, we establish an impossibility result by proving that server-side local processes cannot decrease the quantum communication requirements of secure delegation protocols. We develop a series of no-go results that prohibit such a process within an information-theoretic framework.

Second, we present a possibility result by introducing Selectively Blind Quantum Computing (SBQC), a novel functionality which allows the client to hide one among a known set of possible computations. We precisely characterise how the differences between the computations in the protected set influence the number of qubits sent during our SBQC implementation, therefore yielding a communication-optimal protocol. This approach can reduce qubit communication drastically and demonstrates the trade-off between information leaked to the server and the protocol's communication cost.

1 Introduction

1.1 Context

Currently, in order to delegate a quantum computation to a quantum service provider, a client sends over the input and the computation they want to perform, and the quantum server is supposed to do the computation and send back the outcome. For some important computations, the client may want the following privacy and security guarantees: the input, computation, and the output should stay unknown to the server (blindness); furthermore, the client should be

---

able to check that the computation has been performed correctly (verifiability). Delegating quantum computations in this secure way to a powerful server is a crucial step toward making quantum computing widely accessible and establishing a secure quantum cloud. Hence, improving the efficiency and practicality of these functionalities together with the fundamental limitations on them has been a long-lasting topic of interest in quantum cryptography, complexity theory and physics [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12]. In this manuscript, we focus on the blindness property only.

The Universal Blind Quantum Computing (UBQC) protocol proposed by [1] achieves this functionality provided that the client can generate random single qubits and send them over a quantum channel to the server. However, the number of qubits that must be sent in this protocol (i.e. its communication complexity) is linear in the size of the circuit. From a practical point of view, the flying qubit from client has to be integrated into the remote quantum computer of the server. In the process, it might be affected by the noise of the channel established between the parties, which results in a reduction of the computation performance on the server's side. The main question addressed in this paper is whether techniques can be used to reduce this communication complexity and also establish lower bounds on the number of encrypted qubits necessary to achieve blindness.

There are three existing approaches for constructing the blind quantum computation functionality, each using a specific resource:

1. Blind quantum computing with Prepare-and-Send [1]. This approach has been formulated in the measurement based quantum computation (MBQC) model [13]. Here, the client is able to apply random single qubit rotations and sends multiple encrypted qubits to the server (where the keys are the rotation angles). The server uses them to construct an encrypted graph state. The computation is performed interactively with the client sending measurement instructions and the server returning the measurement outcomes for subsets of the graph's qubits. At the end, the server sends back the output qubits to the client. The main resource for this variant of UBQC is the ability to prepare single qubits on the server's machine without the server knowing the state of these qubits, also called secure remote state preparation (RSP). This is the main variant of UBQC we will be considering throughout this paper, and will discuss it in more details later. A closely related alternative approach are Receive-and-Measure protocols [14, 3], in which the server prepares the computation graph state and sends it qubit-by-qubit to the client, who has the ability of measuring these qubits in a certain number of bases.

2. Blind quantum computing with non-communicating entangled servers [6]. This device-independent approach requires two non-communicating servers to share maximally entangled states with each other prior to the protocol's execution. Then the client instructs one of the servers to measure their part of this resource state and send back the outcome without the other server learning it. This information asymmetry is leveraged by the client to both perform the computation and test the servers' honesty by utilizing the rigidity of non local games [15]. However, sharing high amounts of entanglement between distant servers and making sure that the severs are not communicating (e.g.

by enforcing space-like separations so the parties can not have causal effects on each other) are the downsides of this approach.

3. Blind quantum computing with computational hardness assumptions and classical client [16]. The idea behind this approach is to perform secure remote state preparations using only classical public-key cryptography [5, 17], in particular Trapdoor Claw-Free Functions (TCF) based only the Learning with Errors problem. In this approach, quantum communication is removed entirely, at the cost of having to implement the TCFs complex and deep circuits on the server's quantum hardware. Compared to the other approaches which are information-theoretically secure, these protocols provide only computational security.

Among these approaches, the first one seems closer to implementation, as it has already been demonstrated in the lab [9]. From a practical perspective, the main obstacle here is the quantum communication cost which is equal to the size of the graph state implemented on the server (i.e. proportional to the number of gates in the circuit that the client wants to run). Reducing the communication cost and computing its fundamental lower bounds is an important step toward making this approach more feasible.

Apart from the experimental significance of achieving UBQC with reduced qubit communication, the possibility of accomplishing this with minimal resources has important implications for complexity theory. The first question in this direction is whether it is possible to make the client fully classical while preserving information-theoretic security. As discussed, this can be done with computational security using TCFs. However, achieving blindness with a fully classical client in an information-theoretic context is unlikely, as it would lead to unlikely collapses in complexity classes. As shown in [18], delegating Boson Sampling in such a way would imply the collapse of the polynomial hierarchy to the third level. This relates to the foundational question of verifying BQP computations with BPP verifiers, which remains an open question in the field. Therefore, studying the communication requirements of the [1] protocol can shed light into possible answers to this question.

An important question in that direction is: *Can we construct more efficient RSPs with less quantum communication?* This question was positively addressed in the Random Oracle Model [19]. They demonstrate that UBQC can be done by sharing keys encrypted in quantum states, with a communication complexity that scales polynomially with the security parameter and remains independent of the computation's size. To this low number of keyed states, they apply a gadget-increasing protocol to generate enough encrypted states to perform the client's computation blindly. However, while the Random Oracle Model is a nice theoretical model, it is not practical and similar protocols based on concrete instantiations have not yet been developed.

From another point of view, for many practical applications, hiding all aspects of the computation, i.e. the graph, the input, the algorithm, etc, might be unnecessary. For example, in the federated machine learning protocol that uses delegated quantum computing [20], the only sensitive parameters are the weights of a run, and the other information about their circuit can be publicly revealed. For many cases, hiding only the input and some other qubits affected by their measurement

outcome suffices. In that case, it would be sufficient to hide only the input and its propagation through the circuit [21].

1.2 Overview of the Main Results

Questions addressed. A more comprehensive understanding of the minimal resources needed for blind quantum computations can be obtained from the answers to the following open questions:

1. For now, one RSP call generates a single keyed quantum state. Ideally, we would like the server to be able to expand this state into multiple encrypted states as in [19]. *Is it possible to design more efficient RSPs from similar server-side processes in the plain model?*[1] This would then imply that a run of the UBQC protocol requires less quantum communication for while preserving the same information-theoretic security.

2. The full UBQC protocol perfectly hides all the computation angles in the graph, the flow, the inputs and the outputs – full blindness. However, perhaps there are settings in which the client does not care about the secrecy of some of these values. *Is the full UBQC protocol still necessary and, if not, what is the minimal amount of quantum communication that is required to selectively blind a subset of these parameters?*

Impossibility results – Section 3. In the first part of our work, we prove a sequence of no-go results which rule out some scenarios for reducing communication complexity of a protocol for implementing UBQC with full blindness.

More specifically, we study the existence of a process that allows the RSP process to be bootstrapped: implemented on the server's machine, it takes one keyed qubit as input and outputs two (or more) keyed qubits with different keys. We prove in the information theoretic setting that such a functionality, either as an isometry 1A or as a general quantum channel 2, can not be achieved due to the basic laws of quantum mechanics, i.e. linearity. We also consider the foundational aspect of this result and show that it can not be reduced to the no-cloning theorem. In lemma 1 We also show that it is impossible to construct a protocol that has the same security guarantees as the original UBQC by only having access to a process which replicates some but not all of the encrypted states.

Next, we turn our attention to variants of the the UBQC protocol of [1] which rely on entangled states or a more limited set of states than the original one. We again show the impossibility of amplifying such resource states to decrease the communication complexity. We rule out the entangled version by analyzing the symmetry of bipartite entangled systems in quantum mechanics – the equality of the Von Neumann entropy of subsystems 3. This result fundamentally relies on the fact that entanglement cannot be increased in a bipartite system by local operations.

We then show that relaxing the requirements of the amplification processes described above is also not enough to yield a secure UBQC protocol. Indeed, we

---

[1]Note that this new functionality is not equivalent to entropy expansion, but a distribution or splitting process

exhibit explicit differential attacks on a protocol makes use of amplifiers which can approximately prepare the correct states, i.e. states that have correlated parameters in the classical description of each subsystem, but the correlation is supposed to be minimal 4. All these results are encompassed in 1.

Finally, we prove the existence of a lower bound on the amount of quantum communication needed for blind quantum computing protocols which hide the same information as the original UBQC protocol: the number of unknown states sent to the server (or more generally, the number of unknown independent parameters) must be superlogarithmic in the size of the computation 5.

Possibility results – Section 4. In the second part of the paper, guided by the impossibility results above, we focus on optimizing the UBQC protocol by going back and analyzing the functionality that it achieves. We introduce a more general blind functionality (Resource 1) which we call Selectively Blind Quantum Computing (SBQC): *It allows the client to delegate one computation from a known set of unitaries to the server while hiding the choice itself.* UBQC then corresponds to the subcase of SBQC when the set of unitaries contains all the computations that are possible on a given graph. We then show that we only need to mask the differences between computational paths rather than the entire computation. SBQC demonstrates its full advantage when unitary operations differ minimally — for instance, by just one or a few gates. The protocol uses two main techniques:

Section 4.1 – Angle masking: We study here the hiding cost generated by a node in the computation graph which has different default measurement angles in the known computations. This is done by first defining the notion of a *future cone* (Definition 6) of a given node: all the nodes in the graph which are affected by – i.e. which can get corrections directly or indirectly from – the measurement outcome of that node. We then show that if only the measurement angles of some nodes are different between the graphs and all the other properties of the computations are the same, it is sufficient to hide with quantum communication (i) those specific nodes and (ii) the nodes in their future cones which have non Clifford measurement angles, i.e. odd multiples of $\pi/4$.

Section 4.2 Graph masking (Merge-and-Break): If the computation graphs have different shapes, we give conditions for finding a *merger graph* (Definition 8 and Lemma 6) that contains all computational graphs. We then introduce methods to decompose the graph into the desired configuration while concealing its shape, size, and flow. We achieve this by leveraging bridge-and-break operations [22], alongside established techniques from Measurement-Based Quantum Computing [23].

We then combine these two techniques into a single full protocol for SBQC for delegating one out of two computations (Protocols 5 and 4). This protocol can be easily generalized to arbitrary sets of computations by applying similar rules to combine all the graphs and measurement angles one-by-one. We finally prove that it is information-theoretic secure in the composable Abstract Cryptography framework (Theorem 2), meaning that it remains secure even if run sequentially or concurrently with other protocols.

Thus, by relaxing full universal blindness to SBQC, we significantly reduce communication overhead while maintaining robust information-theoretic security. Ap-

plying SBQC to practical scenarios such as algorithm auditing, servers can validate algorithmic structures without accessing hidden user data or selected computational choices. This approach effectively bridges theoretical UBQC with practical deployment within emerging distributed quantum computing architectures.

Note. We note that a result similar to our angle masking technique, stating that hiding only non-Clifford gates is sufficient, has been previously known in the circuit model [24]. We have rediscovered this fact within the framework of Measurement-Based Quantum Computation and utilize it as a subroutine to build a complete protocol, SBQC.

Additionally, a somewhat similar objective to SBQC was recently pursued in [25] to create a protocol that allows for concealing specific or sensitive parts of a quantum circuit. This is achieved by employing quantum fully homomorphic encryption (QFHE) at the input and output, while also hiding the sensitive portion of the circuit, such as the oracle in the Grover algorithm's search function. While both approaches share a goal of selective concealment, our SBQC protocol employs entirely distinct techniques. Furthermore, our work provides a clearly defined functionality for this selective hiding and includes a comprehensive analysis of the communication cost increases that may arise from variations among the unitaries in the publicly known set, offering a detailed perspective.

2 Preliminaries

In this section we briefly introduce the notions and technical tools that we will use throughout the paper.

2.1 Notations

We list the necessary Quantum Computing notations we use in this paper. The bra-ket notation is used to represent pure quantum states, such as $|\psi\rangle$. The symbol $|0\rangle$ represents the quantum state equal to $\binom{1}{0}$ in the vector notation. Then $|1\rangle$ equals $\binom{0}{1}$ and $\alpha|0\rangle + \beta|1\rangle$ where $\alpha, \beta \in \mathbb{C}$ and $|\alpha|^2 + |\beta|^2 = 1$ equals $\binom{\alpha}{\beta}$ in the vector notation. Specifically, we widely use the following quantum state parametrised by an angle θ as: $|\pm_\theta\rangle = \frac{1}{\sqrt{2}}(|0\rangle \pm e^{i\theta}|1\rangle)$. We also use ρ to represent the density matrix of an arbitrary qubit. Quantum systems are composed via tensor products. We use $\rho^{\otimes n}$ to show the tensor product of n copies of ρ

A unitary operation describes the reversible evolution of quantum systems. A unitary matrix $U : L(\mathbb{C}^d) \longrightarrow L(\mathbb{C}^d)$ is a linear operator where the input and output have the same dimension and $UU^\dagger = U^\dagger U = \mathbb{I}$. Also, an isometry D in this manuscript refers to an operator which is inner product preserving, but in a more general sense than unitary operations. In an isometry $D : L(\mathbb{C}^{d_1}) \longrightarrow L(\mathbb{C}^{d_2})$ the input and output dimensions are different and we have $D^\dagger D = \mathbb{I}$. In other words, isometry is a generalization of a unitary where you can tensor product your input state with another state and perform a unitary on the whole system. The most general mathematical model for describing a quantum process is a quantum

channel, $\Lambda : L(\mathbb{C}^{d_1}) \longrightarrow L(\mathbb{C}^{d_2})$ which is a completely positive and trace preserving (CPTP) map that maps the initial state of a quantum system to its final state. It can be shown that the action of a quantum channel on an input state can be written as a partial trace on the evolved state of the input by an isometry [26]:

$$\forall \, \Lambda \, \exists \, D, E \ s.t. \quad \Lambda(\rho) = tr_E(D\rho D^\dagger) \tag{1}$$

Where E is the space of the tensor product state with the input ρ.

One of the most widely used set of unitary operations are Pauli matrices, also known as Pauli gates, which form a basis for the operations over a single qubit system, described as follows:

$$\mathbb{I}, \quad X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \quad Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}, \quad Y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix} \tag{2}$$

Another important gate is the Hadamard gate, which transforms the computational basis (Z) to plus-minus basis (X), and has the following matrix form.

$$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \tag{3}$$

Also, an arbitrary rotation around the Z-axis is given by

$$Z(\theta) = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\theta} \end{pmatrix} \tag{4}$$

Finally, we introduce a two-qubit controlled operation gate known as CZ. This gate applies a Z gate on the target qubit if the control qubit is in state $|1\rangle$, and applies an identity if it is in state $|0\rangle$. The CZ gate can act as an entangling operation as is given by the following matrix:

$$CZ = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix} \tag{5}$$

2.2 Measurement-Based Quantum Computing.

Quantum computation can be described in various ways universally. In this paper, we use a universal model known as Measurement-Based Quantum Computation (MBQC) [27]. This model is well-suited for delegating quantum computation and the desired properties of it.

In MBQC, the computation is defined with a pattern (also called a resource graph) which is a set of qubits prepared in $|+\rangle$ states, where some of them are entangled using a CZ operator. It is usually represented by a graph G where only entangled nodes are connected with an edge.

$$|G\rangle = \prod_{(i,j)\in E_G} CZ_{(i,j)} |+\rangle^{\otimes V_G}$$

If the computation has quantum input, a subset of the nodes of G is reserved for the input states, and another subset is reserved for the output states. We call such a graph G an open graph.

Definition 1 (Open graph). *An open graph is a tuple (G, I, O), where $G = (V, E)$ is an undirected graph and $I, O \subseteq V$ are the sets of input and output vertices, respectively.*

To compute a general unitary computation in MBQC, one needs to measure all non-output nodes sequentially. The set of universal gates can be implemented only using the measurements in the XY-plane of the Bloch sphere and the bases $|+_\theta\rangle, |-_\theta\rangle$ where the angles $\theta \in \{\frac{k\pi}{4} : k = 0, \ldots, 7\}$ suffice for an arbitrary quantum computation [28]. In what follows whenever measuring with an angle ψ is mentioned, we refer to measuring in the basis $|+_\psi\rangle, |-_\psi\rangle$.

As the measurements are probabilistic by nature, in the MBQC picture, to obtain a deterministic[2] outcome, the intermediate measurement outcomes need to be taken into account and corrected at the end. It has been shown in [29] that there exist a necessary and sufficient condition on an open graph known as *g-flow* to achieve such deterministic computation. This condition is defined formally as:

Definition 2 (G-flow). *For an open graph (G, I, O) a g-flow is a tuple $(g, \prec)$, where $g : V \setminus O \to 2^{V \setminus I} \setminus \{\emptyset\}$ and $\prec$ is a partial order on V, satisfying the following conditions:*

- *If $j \in g(i)$, then $i \prec j$.*
- *If $j \in \mathrm{Odd}(g(i))$, then $i \prec j$.*
- *For all i we have: $i \in \mathrm{Odd}(g(i))$.*

In this context, $\mathrm{Odd}(K) = \{u : |N(u) \cap K| = 1 \pmod 2\}$ for $K \subseteq V$ is the set of odd neighbours of K.

Given a g-flow, the computation can proceed by correcting the measurement angles of each node depending on the previous measurement outcomes [27]. A node v of the resource graph propagates an X-correction to every node in $g(v)$ and a Z-correction to every odd neighbour of $g(v)$ (See an example in Figure 1). These corrections are equivalent to applying X and Z Pauli gates to the corresponding nodes before the measurement, but they can be incorporated into the measurement angles.

We can also compute the "inverse" formula and obtain all the corrections that *affect* a particular node. Then we have a formula for correcting the measurement angle of a node w:

$$\varphi'_w = (-1)^{s_X} \varphi_w + s_Z \pi \quad \text{where} \quad s_X = \sum_{j : w \in g(j)} s_j, \, s_Z = \sum_{j : w \in Odd(g(j)), j \neq w} s_j. \tag{6}$$

Finally, a measurement pattern is defined as below:

Definition 3 (Measurement pattern). *A pattern in measurement based quantum computation is given by a graph $G(V, E)$, input and output vertex sets I and O, a flow g which induces a partial ordering of the qubits V, and a set of measurement angles $\{\phi_i\}_{i \in O}$ in the $X - Y$ plane of the bloch sphere.*

---

[2]We call a computation pattern deterministic if its final outcome does not depend on intermediate measurement results.
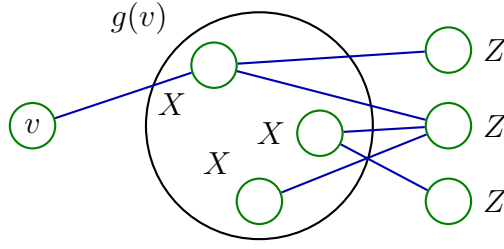
Figure 1: In green we show all the nodes for which the measurement angle has to be adapted depending on the measurement result of v. The node v induces an X-correction on every node in the set $g(v)$. It also induces a Z-correction on every odd neighbour of $g(v)$. Odd neighbours are the vertices that have an odd number of edges connecting them to $g(v)$.

2.2.1 Delegated computing in MBQC

To perform a delegated computation in this framework between the Client with a small quantum device and the Server with a quantum computer, we proceed as follows. Firstly, the Client sends to the Server the qubits of the input and the classical description of the resource graph. Then the Server prepares a $|+\rangle$ state for each non-input vertex and entangles prepared and received qubits with respect to the graph. After that, the Client sends to the Server a measurement angle for every non-output node. The Server measures them according to the g-flow and sends the measurement results to the Client. In the end, the Server sends to the Client the output qubits. These qubits can have extra X- and Z- Z-rotations because of the corrections. So the Client uncomputes them and recovers the final result. We note that performing delegation computation in this way, without any additional procedure, does not satisfy any security requirement, i.e. the server has full knowledge over the data and the details of the computations of the client.

2.3 Universal Blind Quantum Computation Protocol.

The delegated protocol explained in Section 2.2.1 does not provide any privacy for the client. However, there exist protocols for this model that can hide input, computation, and output of the client from the server. One protocol, known as Universal Blind Quantum Computation (UBQC), was first proposed by Kashefi et al. in [30]. We first describe informally how the protocol works. For a more detailed review, we refer the reader to [31].

Firstly, we implement the computation on a brickwork state to ensure that the graph always has the same structure and the same g-flow; for more details, see [30]. To fully conceal the computation, it remains to hide the measurement angles. Consider a qubit in the resource graph that does not belong to the input or output sets; we refer to such a qubit as a *computational* node. In the UBQC protocol, for each such node, the Client prepares a qubit in the state $|+_\theta\rangle$ with $\theta \in \frac{k\pi}{4}, ; k = 0, \ldots, 7$ and sends it to the Server. The Server then entangles this qubit with the rest of the graph. When the time comes to measure the node, the Server requests a measurement angle from the Client. The Client responds with

9

$\delta = \varphi' + \theta$. A short calculation shows that this measurement produces the same effect as measuring a $|+\rangle$ state with angle φ', except that the angle sent to the Server is now hidden using a one-time pad. However, this masking alone introduces a subtle information leak. Suppose we measure a qubit ρ with angle ψ and obtain the result s. If $s = 1$, we can be certain that $\rho \neq |+\psi\rangle$; otherwise, we know $\rho \neq |-\psi\rangle$. Either way, we gain some information about ρ from the outcome s. Since in our case $\rho = |+_\theta\rangle$, the measurement result s can leak partial information about the secret key θ. To prevent this, the measurement outcome of a masked node must also be hidden. The Client does this by sampling $r \xleftarrow{\$} 0, 1$ and sending the measurement angle $\delta = \varphi' + \theta + r\pi$. The term $r\pi$ flips the measurement basis when $r = 1$. The Server performs the measurement and returns the outcome b to the Client. If the Server is honest, the Client can recover the true outcome as $s = b \oplus r$. Thus, the value of s remains perfectly hidden. The input qubits to the computation are encrypted using $X^a Z(\theta)$. For these nodes, the Client sends a measurement angle of the form $\delta = (-1)^a \varphi + \theta + r\pi$. Because the Server does not know the actual measurement outcomes, the resulting X- and Z-corrections appear random to them. As a result, the output qubits are encrypted with a quantum one-time pad of the form $X^a Z^b$. The Client must apply the inverse of these corrections on their own device to recover the final output of the computation. We will now give a more formal description of the protocol.

Protocol 1 Universal blind quantum computation [1]

---

1. **Alice's Preparation**

   (a) For each column $x = 1, ..., n$ and each row $y = 1, ..., m$, Alice (the client) prepares the single-qubit state $|\psi_{x,y}\rangle$ as $|+_{\theta_{x,y}}\rangle$ where $\theta_{x,y} \in \{\frac{k\pi}{4} \mid k = 0, 1, ..., 7\}$, and sends the qubits to Bob (the server).

2. **Bob's Preparation**

   (a) Bob creates an entangled state from all received qubits, according to their indices, by applying CZ gates between the qubits in order to create a Brickwork state.

3. **Interaction and measurement**

   (a) For each column $x = 1, ..., n$ and each row $y = 1, ..., m$:
      i. Alice computes $\phi'_{x,y}$ where $s^X_{0,y} = s^Z_{0,y} = 0$.
      ii. Alice chooses $r_{x,y} \in \{0, 1\}$ and computes $\delta_{x,y} = \phi'_{x,y} + \theta_{x,y} + \pi r_{x,y}$.
      iii. Alice transmits $\delta_{x,y}$ to Bob. Bob measures in the basis $\{|+_{\delta_{x,y}}\rangle, |-_{\delta_{x,y}}\rangle\}$.
      iv. Bob transmits the result $s_{x,y} \in \{0, 1\}$ to Alice.
      v. If $r_{x,y} = 1$, Alice flips $s_{x,y}$, otherwise she does nothing.

In the language of Abstract Cryptography (see Section 2.4), the functionality and security of UBQC is captured by Figure 2 [32].
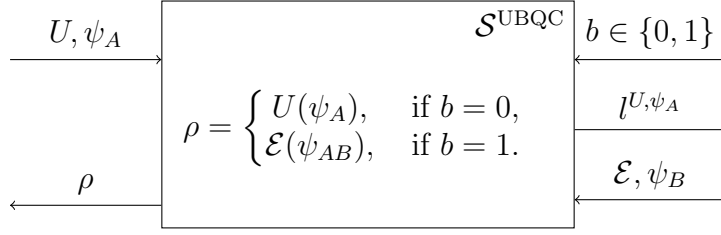
Figure 2: The ideal functionality of UBQC, in the language of Abstract Cryptography.

2.4 Abstract Cryptography Framework

There exist several formal frameworks to study and analyse the security properties of protocols such as UBQC. One of these frameworks is Abstract Cryptography (AC) [33, 34]. One of the advantages of AC over other formal ways, such as game-based security models, is its inherent composability: if a protocol is proven secure within the AC framework, it remains secure when composed, either sequentially or in parallel, with other secure protocols. For example, the confidentiality of a key established by a key exchange protocol holds even when multiple sessions run concurrently, provided each session satisfies the composable security definition. Another strength of AC lies in its modularity and clarity. The framework allows for precise specifications of what a protocol is intended to achieve (via ideal resources), as well as clear descriptions of the adversarial capabilities permitted in the model. This separation of concerns helps avoid ambiguity in security definitions and facilitates reasoning about complex protocol interactions.

At the core of the AC framework are resources — abstract entities (or "boxes") that encapsulate a functionality and interact with external parties via interfaces. These resources are assumed to have unbounded computational power. When a party is honest, it interacts with the resource through a converter, a predefined, efficient CPTP (completely positive trace-preserving) map that mediates input/output behavior. Dishonest parties, on the other hand, can replace their converters with arbitrary CPTP maps. Certain interfaces may be filtered, meaning they are accessible only to malicious parties and are used to model adversarial control points. Capturing all the aspects of the protocol in such ideal resources is crucial and one of the sources of complications in AC. If the ideal resource fails to capture all relevant adversarial behaviors, the resulting security guarantee may be incomplete or even misleading. This places a strong burden on the designer to carefully and comprehensively define the security functionality.

Security in AC is formalized via resource construction. We say that a protocol π ϵ-constructs an ideal resource $\mathcal{S}$ from a real resource (or collection of resources) $\mathcal{R}$, written as:

$$\pi\mathcal{R} \approx_\epsilon \mathcal{S} \tag{7}$$

This expression means that the behavior of the protocol execution using real resources is ϵ-indistinguishable from the ideal resource $\mathcal{S}$, according to a defined metric. As such, the resource construction can be defined as:

Definition 4 (Construction of Resources)**.** *Let $\epsilon(k)$ be a function of the security*

parameter k. We say that a two party protocol $\pi = (\pi_1, \pi_2)$ ϵ-statistically-constructs a resource $\mathcal{S}$ from resource $\mathcal{R}$ if:

- *It is correct: $\pi_1 \mathcal{R} \pi_2 \approx_\epsilon \mathcal{S}$.*
- *It is secure when the second party is corrupted: there exists a simulator σ_2 such that $\pi_1 \mathcal{R} \approx_\epsilon \mathcal{S} \sigma_2$.*

To formalize indistinguishability, we introduce a Distinguisher, an entity whose goal is to decide whether it is interacting with the real protocol $\pi \mathcal{R}$ or the ideal resource $\mathcal{S}$. The Distinguisher may choose all inputs to the system, simulate the behavior of corrupted parties, and observe all outputs. The nature of the Distinguisher's power determines the type of security achieved: if the Distinguisher is computationally unbounded, we obtain statistical security; if it is limited to efficient computations, the result is computational security.

Definition 5 (Statistical Indistinguishability of Resources). *Let $\epsilon(k)$ be a function of security parameter k and let $\mathcal{R}_1$ and $\mathcal{R}_2$ be two resources with the same input and output interfaces. The resources are ϵ-statistically indistinguishable if, for all distinguishers $\mathcal{D}$, we have:*

$$\left| Pr(b = 1 | b \leftarrow \mathcal{D}\mathcal{R}_1) - Pr(b = 1 | b \leftarrow \mathcal{D}\mathcal{R}_2) \right| \leq \epsilon$$

Then we write $\mathcal{R}_1 \approx_\epsilon \mathcal{R}_2$

To prove the indistinguishability between the real-world protocol execution and the ideal resource, we often construct an efficient Simulator. The Simulator connects to the ideal resource via the interface of the corrupted (malicious) party and exposes an additional interface that interacts with the Distinguisher. The role of the Simulator is to mimic the behavior of the real-world adversary in such a way that the Distinguisher cannot tell whether it is interacting with the real protocol execution or with the ideal resource and the Simulator. In other words, the Simulator must generate interaction patterns, based only on the limited information available through the ideal resource, that are indistinguishable from those produced in the real world. The overall goal is to show that for any Distinguisher, there exists a Simulator such that the joint system composed of the Simulator and the ideal resource produces an output distribution that is ϵ-close (in the appropriate metric) to the output of the real protocol execution. This guarantees that the ideal resource is a faithful abstraction of the protocol's security in the presence of adversaries.

3 Impossibility Results: No randomness distribution

In this section, we study the possibility of implementing a local quantum process on the server side that takes as input one of the $|+_\theta\rangle$ states and outputs two such states (or more general states but in the product form or even entangled form with distinct independent parameters) which could serve as the vertices of the ultimate graph state we want to run the computations on. We name this machine a remote

state expander (RSE).

If such a process exists, it can clearly reduce the communication cost in blind quantum computing. We will be studying the existence of such a machine in the information theoretic setting and in the standard model. We show that the separable output variant of this machine can not exist due to the linearity of quantum mechanics, which is also the root of no cloning theorem, a well known and fundamental theorem in quantum mechanics [35]. Finally in the last subsection, we provide a counting argument that puts a lower bound on the number of unknown parameters to the server, indicating a limitation of reducing the quantum communication complexity. Let us start by stating our main no-go results in the following proposition:

Proposition 1 (No-go on Communication Complexity reduction for UBQC). *In any UBQC protocol between a client and a server, there exists no local quantum operation Λ_S (prescribed by the client and performed by the server) mapping a resource state $|+_\theta\rangle$ to a state ρ used on two or multiple nodes of the underlying graph state, such that the new resource, satisfies the completeness and blindness of the protocol simultaneously. In particular, no such server-side process can generate or recycle resource states in a way that reduces the quantum communication from the client without compromising security.*

We support our claim in the above proposition through a central, fundamental no-go theorem and a lemma concerning UBQC. Our first no-go result rules out the possibility of a universal gadget for RSE. Nonetheless, this no-go result is not limited to the scope of UBQC and delegated computation; more generally, it demonstrates that no universal quantum machine exists for distributing a random resource state into multiple useful resource states related to the original random resource. We begin by stating the theorem, and to capture this no-go in full generality, we provide a sequence of lemmas that establish the result across all relevant scenarios.

Theorem 1. *There exists no quantum machine $D : L(\mathbb{C}^2) \longrightarrow L((\mathbb{C}^2)^{\otimes 2})$ that takes as input a state $|+_\theta\rangle$, and maps it into a bi-partite system $\rho_{A,B}$ as $D(|+_\theta\rangle) = \rho_{A,B}(\theta_1, \theta_2)$, such that all the following three conditions hold:*

(1) $tr_B(\rho_{A,B}(\theta_1, \theta_2)) = \rho_A(\theta_1)$
(2) $tr_A(\rho_{A,B}(\theta_1, \theta_2)) = \rho_B(\theta_2)$
(3) If the parameter θ is unknown, then no information about θ_1 and θ_2 is accessible from the output $\rho_{A,B}$.

$$(8)$$

The proof of the theorem is given after Lemma 3. The above general no-go has a particularly important consequence in the context of UBQC, where the instruction for the description of D comes from the client and is implemented by the server. If such a gadget D were to exist, the server could locally generate additional independent resource states from a given $|+_\theta\rangle$ state, where the only available information was the linear dependency equation between the input and output angles. These newly generated states could then be used in UBQC without compromising

13

blindness. With such a gadget, it would be possible to reduce the quantum communication overhead. However, since such a linear operation is quantum mechanically impossible, no general strategy can include such a resource optimisation subroutine. This significantly limits the possibility of designing a more communication-efficient UBQC protocol. Also note that here D is any general linear operation. More specifically, in the following lemma, we assume D to be an isometry and we show that no such isometry can exist to provide a separable structure, as

$$D(|+_\theta\rangle) = |+_{\theta_1}\rangle \otimes |+_{\theta_2}\rangle \quad , \forall\, \theta. \tag{9}$$

where condition 3 of Theorem 1 is satisfied. i.e. θ_1 and θ_2 cannot recovered if θ is fully unknown.

Lemma 1. *There exists no isometry operation $D : L(\mathbb{C}^2) \longrightarrow L((\mathbb{C}^2)^{\otimes 2})$ outputting according to 9, while no information about θ_1 and θ_2 is accessible, if θ is unknown.*

Proof. We give a constructive proof here, and we also provide a second general proof by contraposition in Appendix A. The second proof can also be seen as a no-cloning type of proof for random state distribution by any quantum operation.

We first note that for this lemma, we explicitly require the output to satisfy a separable product from. Although D itself can be a separable or an entangling process. We consider each case separately, starting from the separable case. Also, the isometry can take any arbitrary ancillary system as an input. Without loss of generality, we can assume this ancillary state is instantiated in state $|+_{\theta'}\rangle$ (as D is arbitrary, it can include mapping the initial ancillary state into any arbitrary state). Let's assume now that D consists of a separable one-qubit unitary as:

$$D(|+_\theta\rangle) = U_1 \otimes U_2 \ (|+_\theta\rangle \otimes |+_{\theta'}\rangle) \tag{10}$$

where $U_1, U_2 : L(\mathbb{C}^2) \longrightarrow L(\mathbb{C}^2)$ are the single qubit operations. With no mixing between the states of the two input qubits, the description of U_1 and U_2 fully characterizes at least one of the two output angles. Thus, clearly no separable operation can satisfy condition (3).

We then conclude that such D needs to include entangling operations. We ask if we can construct an isometry which outputs two product states of the desired form in Equation 9 for all input angles θ. Assume that such an isometry exists. We write its action on a full basis and without loss of generality, we select the Hadamard basis:

$$\begin{aligned} D(|+\rangle) &= |+_\alpha\rangle \otimes |+_\beta\rangle, \\ D(|-\rangle) &= |+_\gamma\rangle \otimes |+_\delta\rangle \end{aligned} \tag{11}$$

Using rotational symmetry, and given by two local operations on each qubit, $|+\rangle$ can be mapped to $|+\rangle \otimes |+\rangle$, the number of parameters of D is reduced by two, and we can rewrite a new isometry D_2 such that:

$$D_2 = (Z(-\alpha) \otimes Z(-\beta))D, \tag{12}$$

where D_2 acts as $D_2(|+\rangle) = |+\rangle \otimes |+\rangle$. As the isometry should preserve the inner products implies that:

$$\langle +|-\rangle = ((\langle +|D_2^\dagger)(D_2|-\rangle)) = \langle +, +|+_\gamma, +_\delta\rangle = 0 \tag{13}$$

which forces one of the remaining parameters, γ or δ, to be equal π. Let us choose $\gamma = \pi$, leading to D_2 satisfying:

$$D_2(|+\rangle) = |+\rangle \otimes |+\rangle, \quad D_2(|-\rangle) = |-\rangle \otimes |+_\delta\rangle, \tag{14}$$

Using the above equation, we can easily derive the matrix form of this isometry in the Hadamard basis to be the following:

$$D_2 = \begin{pmatrix} 1 & 0 \\ 0 & 0 \\ 0 & cos(\delta) \\ 0 & sin(\delta) \end{pmatrix} \tag{15}$$

We now require the action of such an isometry on a $|+_\theta\rangle$ to satisfy the desired tensor product form. Writing the planar input state as $|+_\theta\rangle = \frac{1}{2}((1+e^{i\theta})|+\rangle + (1-e^{i\theta})|-\rangle)$, we have:

$$D_2(|+_\theta\rangle) = \frac{1}{2} \begin{pmatrix} 1 + e^{i\theta} \\ 0 \\ cos\delta(1 - e^{i\theta}) \\ sin\delta(1 - e^{i\theta}) \end{pmatrix} \tag{16}$$

The separability condition implies that:

$$(1 + e^{i\theta})(1 - e^{i\theta}) \sin\delta = 0 \tag{17}$$

The possible solutions to the above equation are: either $\theta = 0$ and $\theta = \pi$, or $\sin\delta = 0$ which leads to $\delta = 0, \pi$. In $\delta = \pi$, the problem reduces to quantum cloning, and hence impossible. The $\delta = 0$ case reduces to the separable case, which we discarded earlier, where the condition (3) cannot be achieved. This means that no such isometry can exist for all θ, and the proof is complete. $\qquad \square$

We concluded that no reversible linear operation exists for our desired gadget. We note that this lemma not only applies to the original UBQC protocol, but also to extensions that only require non-identical states with zero overlaps as shown in [36]. This is due to the fact that we put no restrictions on θ_1 and θ_2 apart from the blindness requirement captured by condition (3). Now we generalise the no-go to quantum channels, by relaxing the purity. We are still requiring the tensor product form of the output. We state the no-go in the following lemma:

Lemma 2. *There exists no quantum channel* $\Lambda : L(C^2) \longrightarrow L(C^2)$, *satisfying the following output separability, while no information about* θ_1 *and* θ_2 *is accessible, if* θ *is unknown.*

$$\Lambda(|+_\theta\rangle\langle +_\theta|) = \rho(\theta_1) \otimes \rho(\theta_2). \tag{18}$$

Proof. We assume such a CPTP map exists. We use the Stinespring dilation theorem [26] to represent the channel via an isometry G, such that:

$$\Lambda(|+_\theta\rangle\langle+_\theta|) = tr_{3,4}(G|+_\theta\rangle\langle+_\theta|G^\dagger) \tag{19}$$

More precisely the existence of Λ implies the existence of an isometry $G : L(H_A) \longrightarrow L(H_B \otimes H_D \otimes H_C \otimes H_E)$ such that $G|+_\theta\rangle = |\psi_{\theta_1}\rangle \otimes |\psi_{\theta_2}\rangle$ where $\rho(\theta_1) = tr_D(|\psi_{\theta_1}\rangle\langle\psi_{\theta_1}|)$ and $\rho(\theta_2) = tr_E(|\psi_{\theta_2}\rangle\langle\psi_{\theta_2}|)$. However, in Lemma 1 we showed no such isometry exists. Hence, we reach a contradiction and conclude the proof by contraposition. $\square$

We now relax the separability condition. We assume an RSE machine which outputs an entangled state with two parameters, such that the reduced density matrix of each subsystem is only a function of one of the parameters. Even though, to the best of our knowledge, no UBQC protocol has been designed working with such entangled states, we argue that it might still be possible to do so. However, the independence of the two subsystems in terms of the secret parameter is the minimal requirement for such states to be considered a resource state for UBQC (we will clarify this later as well) Hence, proving a no-go under this assumption will rule out any information-theoretic strategy or subroutines of a new UBQC protocol that can be reduced to the creation of such resource states. In the next lemma, we give this no-go for the isometry, which can always be extended to the channel form using Steinspring.

Lemma 3. *There exists no isometry operation $D : L(\mathbb{C}^2) \longrightarrow L((\mathbb{C}^2)^{\otimes 2})$ outputting an entangled state of two parameters θ_1, θ_2 as $|\psi_{\theta_1,\theta_2}\rangle$, while no information about θ_1 and θ_2 is accessible, if θ is unknown, and the following condition is satisfied:*

$$\begin{aligned}
\rho_A(\theta_1) &= tr_B(|\psi_{\theta_1,\theta_2}\rangle\langle|\psi_{\theta_1,\theta_2}|) \\
\rho_B(\theta_2) &= tr_A(|\psi_{\theta_1,\theta_2}\rangle\langle|\psi_{\theta_1,\theta_2}|)
\end{aligned} \tag{20}$$

Proof. For the conditions of Equation 3 to be satisfied, we need the subsystems A and B to have at least one independent pair of parameters. i.e. they can still be correlated via other parameters potentially included in the description of D. Since $|\psi_{\theta_1,\theta_2}\rangle$ is a pure bipartite state, due to the Schmidt decomposition, we know that the entropy of subsystems should always be equal:

$$S(\rho_A) = S(\rho_B) \tag{21}$$

where $S(\rho) = -tr(\rho \log \rho)$ denotes the Von Neumann entropy. For the independence condition to be satisfied, the left-hand side of 21 is only a function of θ_1 (and not θ_2), and vice versa for the right-hand side. Therefore, the entropy identity implies that:

$$S(\rho_A(\theta_1)) = S(\rho_B(\theta_2)) =: s,$$

for some constant s independent of both θ_1 and θ_2. This means that for a given input, $S(\rho_A(\theta_1))$ must be constant for all values of θ_1, and similarly for $\rho_B(\theta_2)$. But this contradicts the assumption that ρ_A and ρ_B encode the parameters θ_1

and θ_2 respectively, unless ρ_A or ρ_B are constant with respect to their associated parameters. This is not the case, as we assume the density matrices of the two subsystems are distinct and are characterisable by parameters θ_1 and θ_2, such that the change of the input state θ results in changing the output parameters. If the marginals carry no information about θ_1 or θ_2, then no information about these parameters is accessible from any subsystem.

This would imply that either the state $|\psi_{\theta_1,\theta_2}\rangle$ is a separable product state, which is in contradiction with the initial assumption that it is entangled, or that the entropy of the subsystems must change with respect to the change of one of the parameters, which contradicts the Schmidt decomposition, Therefore, no such isometry D can exist. $\qquad\square$

We are now ready to give the proof of Theorem 1.

Proof of Theorem 1. Through Lemma 1, we showed that no isometric map can satisfy all three conditions of Equation 9 simultaneously for all input angles θ when restricted to pure product outputs. In Lemma 2, we extended the analysis to general quantum channels, allowing mixed separable outputs, and showed that even under this relaxation, such a map remains impossible. Finally, in Lemma 3, we relaxed the separability condition and considered general entangled outputs, while maintaining condition (3) in the form of independence of output parameters. We demonstrated that no quantum-mechanically valid map satisfies all required conditions in this setting either. These results collectively rule out all possible constructions of D, establishing that no such map exists. $\qquad\square$

So far, we have shown that generating more resource states with independent parameters on the server side is impossible. To complete our argument for justifying Proposition 1, we need to show that the set of assumptions in Theorem 1 is minimal for UBQC. This means that relaxing the assumption of independence of the parameters will lead to states that cannot be used in UBQC. To this end, we assume that our RSE subroutine is an approximate random state distributor, producing minimally dependent outputs.

We describe such an approximate quantum machine as a process that takes an input state $|+_\theta\rangle$ and produces a two-qubit output state, where each subsystem depends on both output parameters. In the spirit of approximate quantum operations — such as approximate quantum cloning [37, 38] — one can design such a machine to minimize the degree of parameter cross-dependence. While the study of these approximate quantum random state distributors is an interesting direction within quantum information theory, we do not explore their full characterization here and leave it for future work.

Instead, we focus on whether such approximate machines can be useful in the context of UBQC, particularly when their outputs are used to construct graph states, as a means to reduce the quantum communication cost. Suppose an approximate distributor is used to prepare qubits in a graph state $|G\rangle$, where certain angle parameters θ_i are functionally dependent on others (e.g., $\theta_i = f(\theta_j)$), and this dependency is known to the server. We analyze this setting in the pure-state separable case. We show that using such dependent outputs in UBQC compromises

the protocol's blindness. Although the measurement outcomes remain secure due to the one-time pad, the encryption of measurement angles can leak information if the server leverages the known parameter dependency. Therefore, we rule out the use of such approximate subroutines in UBQC through the following lemma.

Lemma 4. *Let $|G\rangle$ be a graph state used in a UBQC protocol, where the state of each node is described by a reduced density matrix $\rho(\theta_i)$. Suppose there exist indices i, j such that the angle θ_j is functionally dependent on θ_i, i.e., $\theta_j = f(\theta_i)$ for some deterministic function f, and this dependency is known to the server, even through the values of θ_i remain hidden. Then, the UBQC protocol may not satisfy blindness.*

Proof. We prove the lemma by showing that a UBQC protocol using such a graph state $|G\rangle$ cannot guarantee blindness. In particular, the blindness condition reduces to the security of a classical one-time pad with correlated keys, which is, in general, insecure. As a result, the server can gain non-trivial information about the client's encrypted measurement angles, thereby violating the blindness requirement.

UBQC protocols employ two layers of encryption: *1) Encryption of Measurement Outcomes*, and *2) Encryption of Measurement Angles*.

The first encryption operates as follows: for a measurement on qubit i, the outcome s_i is one-time padded with a random classical bit r_i, resulting in $b_i = s_i + r_i$. The server receives b_i. This encryption remains secure regardless of any dependencies among the θ_i, because r_i is chosen uniformly at random, independently of θ_i, and remains unknown to the server. Therefore, b_i reveals no information about s_i.

The second encryption works as follows: the client instructs the server to measure qubit i at angle $\delta_i = \phi_i + \theta_i + r_i\pi$. Here, the server receives δ_i and is assumed to know the dependency structure among the θ_i, for example, $\theta_j = f(\theta_i)$. Since the θ_i serve as keys in this one-time pad, the security of the angle encryption reduces to the security of a one-time pad with correlated keys. It is well known in classical cryptanalysis that such schemes are vulnerable to differential attacks [39, 40]. For instance, if $\theta_j = f(\theta_i)$, the server can analyze the pair (δ_i, δ_j). Given knowledge of f, the server can mount differential attacks to infer relationships between ϕ_i and ϕ_j, leaking information about the underlying computation. Consequently, the second layer of encryption is compromised. More generally, no UBQC protocol relying on information-theoretic encryption can satisfy blindness when using a resource state $|G\rangle$ of the form described in the lemma. This completes the proof. □

The consequence of the above lemma is that, even if an optimal approximate RSE machine exists, it cannot be employed within a UBQC protocol to reduce communication resources, as it would compromise the security of the protocol. Furthermore, we observe that any attempt to reduce the communication overhead of UBQC without introducing additional assumptions, i.e., relying solely on information-theoretic constructions, while involving a local process by the server, must involve a subroutine that falls within the class of processes described in Theorem 1. However, as proven, no such subroutine exists. Alternatively, such attempts would result in a graph state as described in Lemma 4, which violates the blindness condition, thus failing to preserve the security of UBQC. This leads to the conclusion stated in Proposition 1.

We finish this section with a couple of remarks to point out some interesting conceptual aspects of our no-go results.

Remark 1. *(Distinction from Entropy Expansion)*: *It is important to emphasize that the functionality we aim to achieve in Theorem 1 is not equivalent to entropy expansion, but rather to a distribution or splitting process of a random bitstring into different states. Thus, it cannot be trivially addressed by entropic arguments. In other words, the process does not seek to generate additional random bits from a bitstring encoded in θ without any extra resources. Instead, it splits the same amount of randomness into two states, with the outputs being non-trivially linked to the inputs. This distinction allows the process to remain consistent with entropic principles, thereby highlighting the non-triviality of our no-go result*

Remark 2. *(Distinction from the No-Cloning Theorem)*: *The functionality described in Theorem 1 may remind us the no-cloning theorem [35]. In this context, we are again considering a machine $C : L(H_A) \longrightarrow L(H_B \otimes H_C)$, which operates as a cloner, such that $C(\rho) = \rho \otimes \rho$. This raises the question of whether the no-distribution theorem we established earlier can be reduced to the no-cloning theorem. We will argue that this is not the case, and this distinction can be seen from two key perspectives:*

1. First, although both theorems are based on the linearity of quantum mechanics, the no-randomness distribution theorem is more general. In the proof of Lemma 1, we recover a cloning scenario for a fixed parameter $\delta = \pi$, while for a general range of the parameter, our machine is allowed to produce more general outputs other than clones. We can also refer to our second proof of Lemma 1 in Appendix A for a similar argument based on dimensionality, which again subsumes the perfect cloning machines.

2. One might attempt to reduce Theorem 1 to the no-cloning theorem by suggesting that if quantum process existed, it could be transformed into a cloning machine by applying local rotations, i.e., $Z(\theta - \theta_1) \otimes Z(\theta - \theta_2)$. Thus, the no-go on RSE would follow from the no-cloning theorem. However, this reduction fails for one key reason: the unknown nature of the input. An essential assumption in both our theorem and the no-cloning theorem is that the input state, which is intended to be cloned or distributed, is unknown. Consequently, a local rotation of the form $Z(\theta - \theta_1) \otimes Z(\theta - \theta_2)$ cannot be applied, since we do not know the value of the input angle θ.

Therefore, while the no-randomness distribution theorem shares a common root with the no-cloning theorem, namely, the linearity of quantum mechanics (which can also be interpreted as the difficulty of disentanglement), it cannot be directly reduced to the no-cloning theorem, and can be seen as a new fundamental no-go result in quantum information.

Remark 3. *(On the approximate randomness distributor)*: *The concept of an approximate randomness distributor, as motivated by the previous lemma, is not the primary focus of this work. However, it offers a promising direction for future research into the properties of quantum machines. Specifically, it refers to*

a machine that generates a bipartite entangled state, where the correlation between the subsystems is minimized. In this context, we examine the correlation between the parameters that each state depends on. Unlike the cloning case, where the objective is to maximize fidelity, the appropriate measure of correlation in this case is not straightforward. To quantify the functionality of an approximate randomness distributor, suitable measures of correlation, such as quantum mutual information [41] or quantum discord [42], could serve as potential candidates.

3.1 A Lower Bound on Communication Complexity of UBQC Based on a Counting Argument

In the previous section, we showed an important limitation for reducing the quantum communication of UBQC through the impossibility of RSE. In this section, we use a counting argument to establish a lower bound on the number of qubits that must be communicated for a blind quantum computing protocol to remain secure. This can also be interpreted as a lower bound for any collective generalization of the randomness-distribution machine discussed earlier.

The lower bound is derived by considering a simple attack strategy that a malicious server could employ to compromise the security of a blind quantum computing protocol. Specifically, the server could randomly guess the input parameters $\{\theta_i\}_{i=1,2,\ldots,m}$ (the keys) and then implement and interpret the corresponding computation based on those values. We aim to identify a necessary condition for ensuring the security of a UBQC protocol against this type of attack. To illustrate this concept, consider the case where the only input the server receives from the client is a single-keyed qubit $|+_\theta\rangle$. In this scenario, the server could randomly guess the parameter θ and proceed to implement and interpret the computation accordingly. In such a case, the server would correctly guess the computation with a probability of $\frac{1}{8}$, which is a non-negligible probability. We formalise this argument in the following lemma.

Lemma 5. *The number of qubits m needed to be communicated for a blind quantum computation is lower bounded by $\omega(\log n)$, where n is the number of vertices of the underlying graph state.*

Proof. Let n be the number of vertices in the underlying graph state for UBQC. Each of these vertices corresponds to a $|+_\theta\rangle$ state where $\theta \in \Theta$ and where Θ is a finite set. Assume a subset of these qubits are not blinded and known to the server, and some others are encrypted and sent by the client. If the number of sent qubits from the client to the server is m, then there are $|\Theta|^m$ possibilities for the whole graph state, which we denote each by the set $\{G_i\}_{i=1}^{|\Theta|^m}$. As the probability distribution on this is uniform when the server guesses one of them randomly, the probability of each of these graphs (G_i) to be the desired graph state (G^*) which the client wishes to be executed is exponentially decreasing in m, and it needs to be a negligible function of n, the size of the graph (proportional to the size of computation) for the protocol to stay secure against random guess attack, which

means we should have:

$$Pr[G(i) = G^*] = \frac{1}{|\Theta|^m} = negl(n) \qquad (22)$$

Solving this asymptotic equality for m easily yields $m = \omega(\log n)$. $\qquad \square$

This argument can be generalized in two ways:

1. If the protocol uses other types of states that are not $|+_\theta\rangle$, but still contain m independent parameters (e.g., a highly entangled GHZ state with m rotated qubits, where the rotation angles of each pair of qubits are independent), the same argument holds. Specifically, the number of possible underlying graph states will be exponentially large ($|\Theta|^m$). This lower bound arises from the fact that, as the size of the graph increases, the dimensionality of the input parameter space must also increase to keep the success probability of the random guess strategy negligible with respect to the computation size.

2. If the set Θ is not discrete but continuous (and symmetric, as it should be), the claim still holds approximately. In this case, the server can discretize the space of input states by dividing it into $|\Theta|$ regions, each with a length of $\epsilon = \frac{2\pi}{|\Theta|}$. For each of the unknown angles θ_i, the server will randomly guess an index $n_i \in \{0, 1, \ldots, |\Theta|\}$ and select a random angle $n_i\epsilon \leq \theta_i^\sim \leq (n_i + 1)\epsilon$. If all their guesses are correct, they can successfully execute the computation, since $|\theta_i - \theta_i^\sim| \leq \epsilon$, and for sufficiently small ϵ, the statistics of measurement outcomes corresponding to these two angles will be similar. Thus, in this case, the dimension of the parameter space must grow similarly to prevent the random guess attack from succeeding.

Our lower bound is also consistent with other types of blind quantum computing protocols that utilize resources other than $|+_\theta\rangle$. For example, the authors in [43] proposed an efficient blind quantum computing protocol with a communication complexity of $O(J \log n)$, where n is the number of qubits needed for the computation (i.e., the size of the graph state in our case), and J is the computational depth. This upper bound is also consistent with our result and can be derived from the no-programming theorem [44].

4 Possibility Results: Selective blindness

In earlier sections, the client's goal was to delegate an arbitrary computation to the server, with the server only learning an upper-bound on the size of the computation. In this section we assume that the client wishes to delegate one of two known computations. The server should not be able to guess with probability higher than $1/2$ which of these two computations the client decided to perform. This covers use-cases in which the Client is willing to leak some information about the computation they are performing.[3] We show that restricting our security guarantees in this

---

[3]This is the case for instance if the server knows already the ansatz that the client wants to use is a parametrized quantum circuit (for some QML application) but the client does not want to leak the parameters of the trained model.

way allows us to decrease drastically the qubit communication complexity of the protocol.

We first describe how to mask all possible differences between two unitary transformations expressed as MBQC measurement patterns and then give the protocol which achieves this new masking goal. Our protocol can be generalized in a straightforward way to allow the client to delegate one computation within a family of n unitaries without revealing which one has been performed. This is done by considering the set of all pairwise differences between the measurement patterns in the set and masking them accordingly.

In the language of Abstract Cryptography, we design a protocol realizing the ideal functionality of *Selectively Blind Quantum Computation* described in Figure 3.
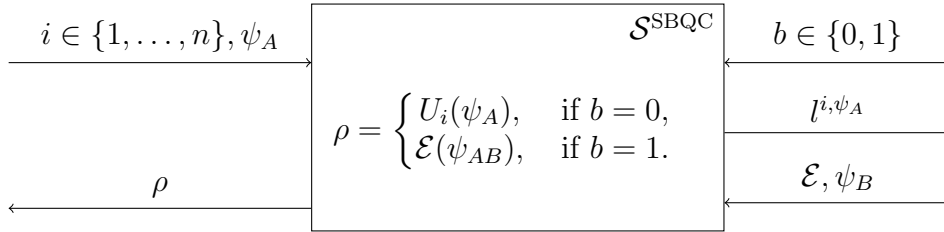
$$i \in \{1, \ldots, n\}, \psi_A \qquad \mathcal{S}^{\mathrm{SBQC}} \qquad b \in \{0, 1\}$$

$$\rho = \begin{cases} U_i(\psi_A), & \text{if } b = 0, \\ \mathcal{E}(\psi_{AB}), & \text{if } b = 1. \end{cases} \qquad l^{i, \psi_A}$$

$$\rho \qquad \qquad \mathcal{E}, \psi_B$$

Figure 3: The ideal functionality of *Selectively Blind Quantum Computation*. Note, that the selection of unitaries $\{U_i\}_{i=1,\ldots,n}$ is fixed as a public parameter of the functionality, which is available to all participating parties. An honestly participating server does not have access to its interfaces, and its input b is filtered to be 0.

4.1 Angle Masking Techniques

In this section we describe how to hide a difference in a default measurement angle of a node in the computation. We assume that the client wishes to delegate either U_0 or U_1 on input $|0^m\rangle$ for input register size m, whose measurement patterns when expressed in MBQC differ only in the default measurement angle at node v: it is φ_0 for U_0 and φ_1 for U_1. They have the same resource graph noted G and the same g-flow g. We assume that the server knows the descriptions of U_0 and U_1 including φ_0, φ_1.

4.1.1 Target Node

We call v a *target node*. To mask the associated angle we apply the technique of the original UBQC described in the Section 2.3: the client sends a qubit $|+_{\theta_v}\rangle$ and during the computation it sends the associated angle δ_v. This perfectly hides the measurement angle for this node.

However, this is not sufficient to hide the outcome of the computation. We must also mask all the qubits influenced by the measurement on qubit v, otherwise the server may use the difference in outputs to distinguish U_0 and U_1. As explained in Section 2.2 the measurement outcome s_v propagates through the graph in the form of measurement corrections. For example, the target node induces an X-correction

on the qubits at positions $w \in g(v)$. If this is the only correction for one such node w, then w has to be measured with the updated angle $(-1)^{s_v}\varphi_w$. The Server is aware of the default measurement angle for node w and they need to know the updated angle to perform the computation, hence they can calculate s_v.[4] This means that the measurement angles that depend on s_v have to be masked.

As seen in Figure 1, the target node propagates different corrections to different qubits. We analyze which masking techniques are required depending on the type of correction and the value of the measurement angle of a qubit.

4.1.2 X Corrections

Consider $w \in g(v)$. The corrected angle for it is defined in the Equation 6. Since v belongs to s_X of w, the value of s_v has the effect of changing the sign of φ_w. We consider the following three cases depending on the value of φ_w.

No effect. If $\varphi_w \in \{0, \pi\}$, the value $(-1)^{s_v}\varphi_w$ does not change depending on s_v and we do not need to mask it.

Classical hiding. Suppose now that $\varphi_w \in \{\frac{\pi}{2}, \frac{3\pi}{2}\} = \{\frac{\pi}{2} + r_1\pi | r_1 \in \{0,1\}\}$. The X-correction maps these angle to one another. To mask this change it is enough to sample $r_2 \overset{\$}{\leftarrow} \{0,1\}$ and ask the qubit to be measured with angle $\varphi_w'' := \varphi_w' + r_2\pi$. Then we have

$$\varphi_w'' = \frac{\pi}{2} + (r_1 + r_2)\pi + s_Z\pi$$

It is easy to see that r_1 is perfectly hidden. Also, we can deduce the result of the original measurement by flipping the outcome transmitted by the Server when $r_2 = 1$ or keeping it as it is when $r_2 = 0$.

Quantum hiding. Lastly, assume $\varphi_w \in \{\frac{(2k+1)\pi}{4} | k = 0, \ldots, 3\}$. For these nodes the X-correction changes the angle from $\pi/4$ to $7\pi/4$ or from $3\pi/4$ to $5\pi/4$. It is not possible to deterministically deduce the result of a measurement in the $\pi/4$ basis from a measurement outcome in the $7\pi/4$ basis and vice versa, and similarly for the other two bases, since they differ by a value of $\pi/2$. Hence, to mask them the client sends a $|+_{\frac{r_2\pi}{2}}\rangle$ qubit to the Server, where $r_2 \overset{\$}{\leftarrow} \{0, \ldots, 3\}$ and change the measurement angle to $\varphi_w'' = \varphi_w' + \frac{r_2\pi}{2} + r_3\pi$. Here $r_3 \overset{\$}{\leftarrow} \{0,1\}$ is hiding the measurement outcome. The resulting measurement angle is uniformly random among the four possible choices, so it is perfectly hidden. This last case therefore incurs a communication cost of 1 qubit.

Remark 4. *Note that X- and Z-corrections do not change the angle type. That is why instead of talking about the type of the corrected angle $(-1)^{s_X}\varphi_w + s_Z\pi$ we just consider the default value φ_w.*

---

[4]If w receives corrections from other nodes, the other correction bits are known to the server and they can again recover s_v with a simple XOR.

4.1.3 Z Corrections

We now consider nodes u where $u \in Odd(g(v))$. If s_v is equal to 1 it has an effect of adding π to the measurement angle of u. To achieve indistinguishability, the Client transmits a corrected measurement angle $\varphi_u'' = \varphi_u' + r\pi$ with $r \xleftarrow{\$} \{0,1\}$. Since r is not known to the adversary, the value of s_v in φ_u' is perfectly hidden. The real measurement result can be deduced similarly to the previous cases.

4.1.4 Propagating Corrections

Similarly to the corrections coming directly from the target node, we need to mask the ones produced by the dependent nodes of the target. We define the *future cone* as the set of vertices that need to be masked. Intuitively, the future cone of a node v contains all the qubits whose measuring angle depends on the measurement outcome of v. See Figure 4 for an example of a future cone.

Definition 6 (Future cone). *The* future cone *of a node v in a resource graph for MBQC computation is defined recursively. A qubit w belongs to the future cone of v if*

- *It belongs to the set of X or Z dependencies of v.*
- *It belongs to the set of X or Z dependencies of a node in the future cone.*

The set of qubits that receive an X-correction from the target node or from the other nodes of the future cone are called the interior *of the future cone.*

Definition 7 (Qubit-masked node). *A node w is called* qubit-masked *if it belongs to the interior of the future cone of the target qubit and its default measurement angle belongs to the set $\{\frac{\pi}{4}, \frac{3\pi}{4}, \frac{5\pi}{4}, \frac{7\pi}{4}\}$.*
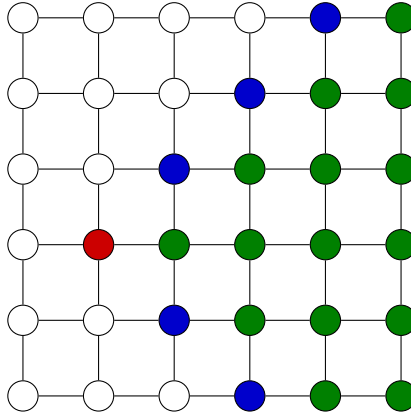

Figure 4: The future cone of the red node is shown in green and blue. The green nodes denote the interior of the future cone. The term is inspired by the shape of this dependence set in the resource graph equal to the $\mathbb{Z}_2$ lattice with the g-flow $g(x,y) = \{(x, y+1)\}$.

Recap 1 Angle hiding procedure.

- A target node generates a future cone.
- The target node and qubit-masked nodes are masked by having the client send additional qubits and adapting the measurement angle.
- The client hides the other nodes of the cone classically by adding a random $r\pi$ rotation to the measurement angle.

Remark 5 (Generalization to private quantum input.). *To hide a quantum input we use the quantum one time pad $Z(\theta_i)X^{a_i}(\rho_{in})$. For every index i we can see the part a_i of the secret key as an X-correction to an input qubit. Since these qubits receive this X-correction and some of them might be qubit-masked we hide the effect of this correction using θ_i. That is why instead of a simple Z^{b_i} we use $Z(\theta_i)$ in the encryption.*

We need to hide the measurement outcome of these qubits. We can see this scenario as an instance of angle hiding where all input nodes are target nodes. The quantum communication cost in this case is equal to

$$\#\{input/output\ qubits\} + \#\left\{\frac{(2k+1)\pi}{4}\ angles\ on\ dependent\ qubits\right\}.$$

Remark 6 (Further optimizations.). *Note that we can optimize the angle hiding algorithm even further for the qubit that is sent for the target node. Consider the case when $\varphi_0 - \varphi_1 = \pi \pmod{2\pi}$ and both angles φ_0 and φ_1 are Clifford angles. Then the corrections can only change these angles by π. Therefore, this difference can be concealed using classical masking and we do not need to send a qubit for this node. Nevertheless, we will have to mask the future cone of this node.*

Additionally, for most of the other cases it is enough to send a qubit of the form $|+_\theta\rangle$ where $\theta \in \frac{k\pi}{2}, k = 0, \ldots, 3$. The only time when all 8 possibilities are required is when the difference $\varphi_0 - \varphi_1 = \frac{(2k+1)\pi}{4} \pmod{2\pi}$, with $k = 0, \ldots, 3$.

4.2 Graph Masking Techniques

Suppose now that the two unitaries $U_0 := (G_0, g_0, (\varphi_i^0)_{i=0}^{n_0})$ and $U_1 := (G_1, g_1, (\varphi_i^1)_{i=0}^{n_1})$ that have different resource graphs, g-flow functions, input and output spaces, but all default measurement angles are equal to 0: $\forall i \forall b : \varphi_i^b = 0$. We add this constraint to only deal with the difference in the graphs and g-flows as we have already discussed how to mask the default angles in Section 4.1.

We first give a necessary and sufficient condition for merging G_0 and G_1 into one graph. Then we describe a number of graph masking techniques that can be applied to the merger of G_0 and G_1 to make the computations defined on these graphs indistinguishable. We leave the question of finding the optimal merger graph for future work.

4.2.1 Condition for Merging Graphs.

Suppose we are given a pair of resource graphs with g-flows on them $(G_0, g_0, \prec_0)$ and $(G_1, g_1, \prec_1)$. We define a third graph $(G_M, \prec_{G_M})$ such that it "contains" both G_0 and G_1 and has an order of measurements that does not violate the orderings on G_0 and on G_1.

Having G_M we can use the techniques described below to reduce G_M to G_0 or G_1 following the choice of the client, while keeping the measurement order the same for both computations.

Definition 8 (Merger graph)**.** *We define a graph G_M to be the merger of two given graphs G_0 and G_1 with their respective partial orders $\prec_0$ and $\prec_1$ if it has the following properties*

1. *$\exists$ an embedding function $i_0 : G_0 \to G_M$ preserving edges.*
2. *$\exists$ an embedding function $i_1 : G_1 \to G_M$ preserving edges.*
3. *$\exists$ a total order $\prec_{G_M}$ such that, if $a \prec_0 b$, for some $a, b \in G_0$ than $i_0(a) \prec_{G_M} i_0(b)$ and similarly for G_1.*

In the rest of the document we will abuse notation and say that some of the nodes of G_M belong to G_0 or G_1 instead of saying that they are an image of a node from G_0 or G_1 under an embedding function.

A trivial instantiation of G_M is a graph that contains G_0 and G_1 as two non intersecting components. Then one can measure all the non-output vertices of G_0 first and followed by all of the non-output vertices of G_1. This approach is obviously not the most efficient. In this section we set the necessary and sufficient conditions on G_M to be a merger of G_0 and G_1.

Lemma 6. *A graph G_M is a merger of G_0 and G_1 if and only if the transitive closure of $\prec_0 \cup \prec_1$ (after the respective renaming of vertices) is a strict partial order on G_M i.e. it is irreflexive, antisymmetric and transitive.*

Proof. Firstly, note that irreflexivity of the union of two partial orders is obvious, so we are not going to discuss it in the proof.

We assume G_M is a merger of G_0 and G_1 and prove that the transitive closure of $\prec_u := \prec_0 \cup \prec_1$ is a partial order. By the definition, two arbitrary elements $a, b \in G_M$ have $a \prec_u b$ if and only if $i_0^{-1}(a) \prec_0 i_0^{-1}(b)$ or $i_1^{-1}(a) \prec_1 i_1^{-1}(b)$.

Take the total order $\prec_{G_M}$ from part 3 of the merger graph definition. From the statement above, we can see that if $a \prec_u b$ then $a \prec_{G_M} b$. Therefore, $\prec_u$ is a suborder of $\prec_{G_M}$ hence it cannot have relations that violate transitivity or antisymmetry.

Nevertheless, it is possible that we have $a \prec_u b$ and $b \prec_u c$ but nothing for a and c. It can happen if b has a preimage in G_0 and in G_1 and the first relation comes from $\prec_0$ but the second from $\prec_1$. The relations that complete transitivity can be safely added to $\prec_u$ since they are present in $\prec_{G_M}$. In conclusion, we have obtained that a transitive closure of $\prec_u$ is a strict partial order.

Suppose now that we have some embeddings for G_0 and G_1 into G_M and $\prec_u := \prec_0 \cup \prec_1$ is a strict partial order. We would like to prove that there exists a total order compatible with $\prec_0$ and $\prec_1$. Let us take $\prec_{G_M}$ to be a linear extension of $\prec_u$ (it exists

by the order-extension principle) and check if property 3 holds. For an arbitrary pair $a, b \in G_0$ if we have $a \prec_0 b$ then $i_0(a) \prec_u i_0(b)$ and therefore $i_0(a) \prec_{G_M} i_0(b)$. Similarly for $a, b \in G_1$ if we have $a \prec_1 b$ then $i_1(a) \prec_u i_1(b)$ and so $i_1(a) \prec_{G_M} i_1(b)$. Hence G_M is a merger of G_0 and G_1.

$\square$

Remark 7. *Note that the ordering on the merger does not depend on g-flow functions g_0 and g_1. These functions guarantee that $\prec_0$ and $\prec_1$ satisfy the properties needed for the computation to be deterministic. Therefore every extension of $\prec_i$ to a total order will be compatible with computation i. It is thus sufficient to find a total order that is an extension of both $\prec_0$ and $\prec_1$.*

4.2.2   Masking the Edges.

In this section we only consider the case where G_0, G_1 and G_M have the same number of vertices and the same input and output nodes, so after they are merged we only need to delete edges from the merger graph to obtain either G_0 or G_1. The technique we use in this section utilizes the bridge and break protocol, firstly developed by [22]. Suppose we have three nodes of the resource graph prepared as in Figure 5a.

The bridge operation deletes the vertex in the middle and creates an edge between the side vertices. The break operation also deletes the middle vertex but leaves the side vertices disconnected.
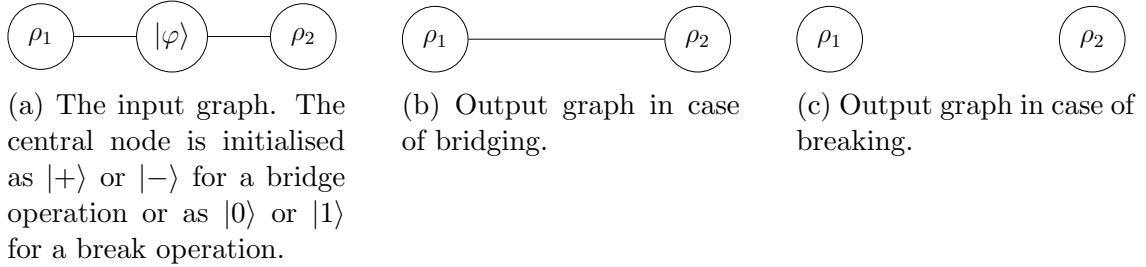

(a) The input graph. The central node is initialised as $|+\rangle$ or $|-\rangle$ for a bridge operation or as $|0\rangle$ or $|1\rangle$ for a break operation.

(b) Output graph in case of bridging.

(c) Output graph in case of breaking.

Figure 5: Bridge and break states before and after the protocol.

Let us give a detailed description of both protocols in the following diagrams.

Protocol 2 Bridge operation.

Inputs: The state $CZ_{1,2}CZ_{2,3}(\rho_1|\varphi\rangle\rho_2)$ which is equivalent to the Figure 5A. Here $|\varphi\rangle = \frac{1}{\sqrt{2}}(|0\rangle + (-1)^c|1\rangle)$, with $c \in \{0, 1\}$
Outputs: The state $CZ_{1,2}(\rho_1\rho_2)$ (as on the Figure 5B).
Protocol:

1. Measure the second qubit in $|+_{\pi/2}\rangle, |-_{\pi/2}\rangle$ basis. Record the result in b.
2. Apply a rotation $Z(-\frac{\pi}{2})$ to the side qubits.
3. Apply a correction Z^{b+c} to the side qubits.

Protocol 3 Break operation.

> **Inputs:** The state $CZ_{1,2}CZ_{2,3}(\rho_1|\varphi\rangle\rho_2)$. Here $|\varphi\rangle = |c\rangle$, with $c \in \{0,1\}$
> **Outputs:** Qubits $\rho_1\rho_2$ disentangled (as on the Figure 5C).
> **Protocol:**
>
> 1. Measure the second qubit in an arbitrary basis.
> 2. Apply a correction Z^c to both side qubits.

Given the description of bridge and break operations, The idea behind our edge masking technique is to add a node in the middle of each edge that originally belonged only to one of the graphs G_0 or G_1, called *middle nodes*. Then the server will transform G_M into G_0 or G_1 using bridge and break operations (Protocol 2 and Protocol 3) in a blind way. Both of these operations consist of a measurement of the middle node and rotations to the side vertices linked to this middle node. Our goal is to incorporate these procedures into a single protocol so that they are indistinguishable for the server.

This is done by first having the client send a qubit in a state chosen at random from $\{|0\rangle, |1\rangle\}$ if they want to break the edge, or from $\{|+\rangle, |-\rangle\}$ if they want to bridge it. The server is then instructed in both cases to measure all middle qubits in the basis $|+_{\frac{\pi}{2}}\rangle, |-_{\frac{\pi}{2}}\rangle$ before any other operation for the computation.

The only other difference between the Protocol 2 and Protocol 3 consists of $Z(-\frac{\pi}{2})$ rotations on the side vertices and Z-corrections. We incorporate these operations into the measurement of the corresponding nodes. To hide whether a $(-\frac{\pi}{2})$-rotation is performed, the client must send a qubit in a state $|+_\theta\rangle$ with $\theta \xleftarrow{\$} \{\frac{k\pi}{2}, k = 0, \ldots, 3\}$ for both of the side vertices. Later in the computation phase of the protocol we measure this node in the basis $|+_\delta\rangle, |-_\delta\rangle$ where

$$\delta = \psi' + \theta - \frac{\pi}{2} + (b+c)\pi + r\pi \text{ for the bridging,}$$

$$\delta = \psi' + \theta + c\pi + r\pi \text{ for the breaking}$$

with $r \xleftarrow{\$} \{0,1\}$ and following the notations of Protocols 2 and 3. Then θ and r fully hide the difference between bridging and breaking and the server is incapable of distinguishing the two operation based on the qubits received from the client.

Similarly to the angle hiding, we need to mask the measurement results of these nodes, so we apply our hiding methods to their future cones. The qubit cost of this operation is equal to 1 qubit for the middle vertex plus 2 qubits for the side vertices, plus the number of qubit-masked nodes in their future cones.

Recap 2 Masking the shape of the graph.

- The middle vertex is in one of the states: $|0\rangle, |1\rangle, |+\rangle, |-\rangle$ depending on the operation.
- It is measured in $|+_{\frac{\pi}{2}}\rangle, |-_{\frac{\pi}{2}}\rangle$ basis.
- Side vertices need a qubit to mask possible corrections and generate future cones.

4.2.3 Deleting a Computational Vertex.

Here we describe a procedure for blindly deleting a computation vertex $v \in G_M$ that has a preimage under the embedding function in G_0 but no preimage in G_1. Input and output vertices are discussed respectively in Section 4.2.4 and Section 4.2.5.

If the client wants to execute U_1 it deletes v from G_M by sending a state $|b\rangle \in \{|0\rangle, |1\rangle\}$ for this node which disentangles v from the rest of the graph. Every neighboring state it corrected by applying Z^b, i.e. adding $b\pi$ to their measurement angle. The node v is measured by the server in the basis $\delta = \psi' + \theta$ for a randomly chosen $\theta \xleftarrow{\$} \{\frac{k\pi}{2} : k = 0 \ldots 3\}$.

If the client want to execute U_0, it sends a qubit for vertex v in a states chosen at random from $\{|+_{\frac{k\pi}{2}}\rangle : k = 0 \ldots 3\}$[5] and adapts the measurement angle to $\delta = \psi' + \theta + r\pi$, where θ is the angle chosen for the state and r is a random bit.

In both cases, the client will have to hide the measurement outcome of this node and the neighbors so they have to apply the hiding techniques to the future cones of these qubits.

Recap 3 Masking a deleted vertex.

- The Client sends a $|0\rangle, |1\rangle$ state to delete a vertex and $|+_\theta\rangle$ with $\theta \in \{\frac{k\pi}{2}, k = 0, \ldots, 3\}$ to keep it.
- The node is measured with angle $\delta = \psi' + \theta + r\pi$ and generates a future cone.
- Its neighbours are masked with an $r\pi$ rotation and generate future cones.

4.2.4 Quantum Input Vertices.

We do not require the embeddings of G_0 and G_1 to have the same position and number of the input vertices. To make the two computations indistinguishable the client sends qubits for every vertex of G_M such that at least one of its preimages under the embedding functions is an input vertex. Let v be an input node of G_0.

If the initial state ρ_{in} for this input vertex is known to the server (public input), its vertex in G_M is mapped to a public input in G_1 and they have the same known input state distribution then no masking is required. In other cases if the client whishes to run U_0 – including if the input is private to the client – the client encrypts the state it sends v with $Z(\theta)X^a$ where $\theta \xleftarrow{\$} \{\frac{k\pi}{2}, k = 0, \ldots, 3\}$ and $a \xleftarrow{\$} \{0, 1\}$.

If the client wishes to run U_1 instead, the behavior depends on the type of the node. If v is also an input of G_1 but they do not have the same distribution, then the state sent by the client for vertex v in U_1 should be encrypted in the same way. If v is a deleted node in G_1, the client applies the procedure from Section 4.2.3.

Otherwise, v is a computational node of G_1. If these nodes have a difference in their measurement angles, the client applies the procedure from the Section 4.1.

---

[5]Note that in this situation the Client never sends one of 8 angles for this node as it only happens when there is an angle difference between U_0 and U_1.

If not, the client sends a qubit $|+_\theta\rangle$, with $\theta \overset{\$}{\leftarrow} \{\frac{k\pi}{2}, k = 0, \ldots, 3\}$ and adapt the measurement angle accordingly.

Recap 4 Masking input vertices.

Case 1. When a public input node collides with another public input node with the same distribution then it is transmitted in clear and does not generate a cone.
Case 2. When an input node of U_0 is either private, or public and:

    a. collides with a computational node of U_1.
    b. collides with an input node of U_1 with a different distribution.
    c. collides with an encrypted input.
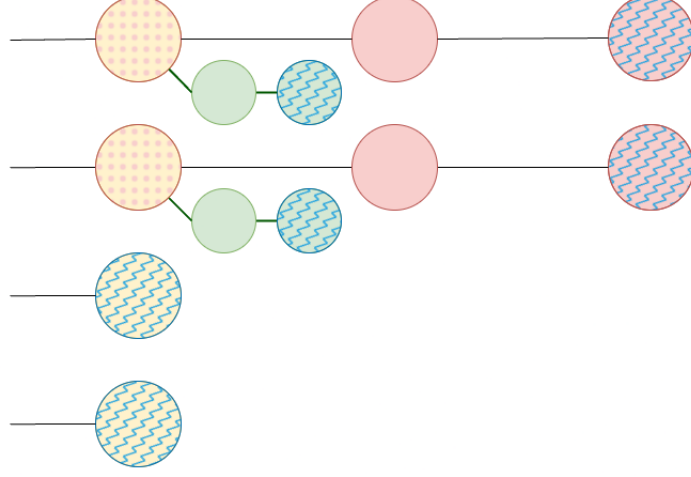   d. collides with a deleted node of U_1.

Actions:

    1. Send $X^a Z(\theta)(\rho_{in})$ for U_0.
    2. For the same order of cases for U_1 we send a qubit:

        a. Send $|+_\theta\rangle$ with $\theta \overset{\$}{\leftarrow} \{\frac{k\pi}{2}, k = 0, \ldots, 3\}$.
        b. Send $X^a Z(\theta)(\rho_{in})$ with $\theta \overset{\$}{\leftarrow} \{\frac{k\pi}{2}, k = 0, \ldots, 3\}$.
        c. Send $X^a Z(\theta)(\rho_{in})$ with $\theta \overset{\$}{\leftarrow} \{\frac{k\pi}{2}, k = 0, \ldots, 3\}$.
        d. Send a $|b\rangle$ state with $b \overset{\$}{\leftarrow} \{0, 1\}$.

    3. Measure with angle $\delta = (-1)^a \psi + \theta + r\pi$.
    4. Mask the future cone generated by the node.

4.2.5   Quantum Output Vertices.

The server should not be able to distinguish the two unitaries by the location or value of the output. If our computation has classical output, then the problem is trivial since the operation of obtaining the output is equivalent to measuring a computational vertex and the future cone masking technique hides the outputs of exactly all vertices that may be influenced by the difference between the two computations. One the other hand, if the output is quantum then the output qubits have to be sent to the client. The set of qubits the server returns to the client should be the same for both computations.

    When the locations of the output nodes do not coincide we will teleport the output qubits of the "smaller graph" into the output set of G_M. Let v be an output vertex of G_0 and a computational vertex of G_1. We add to G_M two vertices v_1, v_1 and the edges (v, v_1) and (v_1, v_2) as shown in Figure 6. These additional vertices are considered as part of G_0. The new default measurement angles are 0 for v and v_1. This means that the operation applied is the identity. To mask this procedure, the client only needs to delete all vertices that do not exist for the computation being executed as described in Section 4.2.3. The Server prepares the qubit for v_2 in the state $|+\rangle$. The qubit for v_2 is sent back to the client at the end of the computation as it contains the quantum output if the computation is U_0.

Figure 6: Teleporting the output values. Red vertices (including dotted red) belong to G_1, yellow vertices belong to G_0. The newly created vertices and edges are shown in green and the output of G_M are the nodes in zigzag blue.

Recap 5 Masking output vertices.

Consider a node v that is used as a computational vertex in one of the unitaries and as the output vertex in the other. Let v_1 and v_2 be the added vertices. The client does the following:

- When v is used as a computational vertex:
  1. Send $|b\rangle$ with $b \xleftarrow{\$} \{0,1\}$ for v_1.
  2. Measure v with angle $(r+b)\pi$ for $r \xleftarrow{\$} \{0,1\}$.
  3. Measure v_1 with angle $r_1\pi$ for $r_1 \xleftarrow{\$} \{0,1\}$.
  4. Mask v's future cone.

- When v used as output:
  1. Send a state from $\{|+\rangle, |-\rangle\}$ for v_1.
  2. Measure v with the default angle $r\pi$ for $r \xleftarrow{\$} \{0,1\}$.
  3. Measure the v_1 with default angle $r_1\pi$, $r_1 \xleftarrow{\$} \{0,1\}$.
  4. Mask v's future cone.

4.2.6 Masking the G-flow.

The last aspect of the computation that has to be hidden is the g-flow. From the definition of g-flow, each node receives the following corrections:

$$\varphi_i' = (-1)^{s_X}\varphi_i + s_Z\pi$$

where

$$s_X = \sum_{j:i\in g(j)} s_j, \ s_Z = \sum_{j:i\in Odd(g(j)), j\neq i} s_j$$

We define a X-(Z-)correction set of qubit i to be the set of qubits that may give i an X-(Z-)correction. Here the X-correction set of i is equal to $\{j : i \in g(j), j \neq i\}$ and Z-correction set is $\{j : i \in Odd(g(j))\}$.

The masking of a node is going to depend on the difference in correction sets. On one hand, if both U_0 and U_1 produce the same correction sets for the node i we apply no hiding.

If only Z-correction sets are different then $\varphi_i^0 - \varphi_i^1 = k\pi \pmod{2\pi}$ and classically $r\pi$-masking the node and masking the future cone of this node will suffice.

If there is a difference in X-correction sets, this is equivalent to hiding an X-correction on a node of a future cone from the Section 4.1. Therefore, it can be hidden with an $r\pi$ rotation if its measurement angle is one of $\{\frac{k\pi}{2}, k = 0, \dots, 3\}$, or the client sends an additional qubit for it otherwise. In both cases we mask the future cone of this qubit.

Remark 8. *Note that these influences are measurement result dependent. Suppose we are working with a node v and every node in the difference between its X-correction sets is already masked with an $r\pi$ rotation. Then the attacker does not know the measurement outcome for these nodes and cannot tell where a possible X-correction of v is coming from. Hence, we do not need to hide the X-correction difference for v. The same holds for the Z-correction sets.*

Recap 6 Masking the g-flow difference.

1. Compute the correction sets of the node.
2. If there is a difference in Z-correction sets mask the qubit with an $r\pi$ rotation.
3. Else if there is a difference in X-correction sets mask the qubit with an $r\pi$ rotation when the angle is from $\{\frac{k\pi}{2}, k = 0, \dots, 3\}$ or using an additional qubit when it is from $\{\frac{(2k+1)\pi}{4}, k = 0, \dots, 3\}$ otherwise.
4. In both cases mask the future cone of the qubit.

4.3 Full Masking Protocol

After describing in the previous two sections the various cases which require some hiding technique to be applied, we combine these cases here and characterize more generally when future cone masking is necessary. This will facilitate the description of our full protocol later.

We define influence sets of every node with respect to the operations induced by the masking techniques or by the MBQC corrections. These operations include Z-corrections, X-corrections and $\frac{\pi}{2}$-rotations.

Definition 9 (X-influence set). *The X-influence set of a node v in graph G contains all nodes of G that require applying an X gate to v at least in one of the branches of the computation to keep the result deterministic and correct.*

Definition 10 (Z-influence set)**.** *The Z-influence set of a node v in graph G contains all nodes of G that require applying a Z gate to v at least in one of the branches of the computation to keep the result deterministic and correct.*

Definition 11 (R-influence set)**.** *The R-influence set of a node v in graph G contains all the nodes of G that require applying a rotation gate $Z(-\frac{\pi}{2})$ to v to keep the result correct.*

We can now define the set of vertices in G_M that generate future cones. This set includes every vertex for which an influence set in G_0 and in G_1 is different. The other two types of vertices that generate a cone are vertices having a difference in the default measurement angle in U_0 and U_1 and the input vertices that need to be encrypted.

Remark 9. *Since the g-flow in G_0 and G_1 can differ, the future cones of a node are going to be different in the two cases. Therefore, we need to hide the shape of the future cone of a vertex in both graphs. When we say a future cone of a vertex in this part of the document we mean the union of the future cones of this node in both graphs.*

We formally define the full masking procedure in Protocol 4 and Protocol 5.

4.4 Proof of Blindness in AC Framework.

In the following section we prove the blindness of this protocol. We start by giving a high-level definition of the Ideal Resource for the 1-of-2 DQC. Then we formulate the Security Theorem 2. To prove it we modify the original Protocols 4 and 5 in a series of reductions that are statistically equivalent. Lastly, we complete the definition describing the 1-of-2 DQC Resource and we complete the proof by designing the Simulator that makes the real world protocol statistically indistinguishable from the ideal world simulation. This implies the blindness of the scheme in the AC framework.

4.4.1 1-of-2 Delegated Quantum Computation Resource Definition.

Our protocol implements the 1-of-2 Delegated Quantum Computation Resource ($\mathcal{R}$) for an honest Client, a possibly malicious Server. The Server has two filtered interfaces that are controlled by bits (c, d). We define the 1-of-2 DQC Resource for two fixed unitary transformations U_0 and U_1 that are embedded in $\mathcal{R}$.

Protocol 4 Full Masking. Preparation Phase.

Inputs: A graph G_M compatible with masking techniques described above. We are assuming that the output teleportation circuits are already a part of G_M.

Protocol: (steps 1-4 are executed by the Client)

1. Compute the input maskings depending on the input distribution as discussed in section 4.2.4.
2. For every node in G_M, compute the influence sets of this node. To do this we need to consider influences of the g-flow in both graphs, the locations of bridge and break operations, deleted vertices and output teleportation circuits.
3. Compute the list of nodes that generate future cones. This list should include every vertex that has different influence sets in G_0 and G_1, the vertices that have an angle difference in two graphs and encrypted input vertices.
4. For every node v send the following qubits to the Server:
   (a) ρ_{in} if v is a clear text input.
   (b) $X^{a_v} Z(\theta_v)(\rho_{in})$ with $\theta_v \xleftarrow{\$} \{\frac{k\pi}{2}, k = 0, \ldots, 3\}$ and $a_v \xleftarrow{\$} \{0, 1\}$ if v is an encrypted input.
   (c) $|0\rangle/|1\rangle$ if v is a deleted node or middle break vertex.
   (d) $|+\rangle/|-\rangle$ if v is a middle bridge vertex or a middle teleportation qubit when output is being teleported.
   (e) $|+_{\theta_v}\rangle$ where θ_v is defined by $\theta_v \xleftarrow{\$} \{\frac{k\pi}{2}, k = 0, \ldots, 3\}$ if v is a computational vertex of the current graph that would be deleted or would be an input in the other graph. Also if v has an X-influence difference and has a qubit-masked measurement angle or it is a qubit-masked node that belongs to a future cone.
   (f) $|+_{\theta_v}\rangle$ where θ_v is defined by $\theta_v \xleftarrow{\$} \{\frac{k\pi}{4}, k = 0, \ldots, 7\}$ if v has different default measurement angles in U_0 and U_1.
5. The Server prepares other nodes of the graph in a $|+\rangle$ state.
6. The Server entangles the qubits received from the Client and the $|+\rangle$ states with respect to the resource graph G_M.

Protocol 5 Full Masking. Computation Phase.

1. The Server measures every middle bridge/break vertex with angle $\frac{\pi}{2}$.
2. The Server measures every middle teleportation node with angle $r\pi$, with $r \xleftarrow{\$} \{0,1\}$.
3. For every remaining non-output node for G_M, let us call it v, repeat the following:
   (a) The Client computes the corrected measurement angle φ' depending on the default angle and the real measurement outcomes of previous vertices.
   (b) The Client instructs the Server to measure v with angle δ where:
      i. $\delta = \varphi' + \theta_v + r\pi$ if v is an output and computational node at the same time, or has a default angle difference, or has R-influence set difference, or it is in a future cone, or it has X-influence difference and is qubit-masked; or if v is a node that is deleted in one of the graphs; or if v is an input and computational node at the same time; or if v is an encrypted input.
      ii. $\delta = \varphi' + r\pi$ if v is a part of a future cone or has an X- or Z- influence set difference and is not qubit-masked.
      iii. $\delta = \varphi'$ if there is no masking on v; or if there is v is a public input vertex.

      In case of ambiguity the Client should apply the first rule from the above list that fits. The parameter r is sampled uniformly at random from $\{0,1\}$.
   (c) The Server returns the measurement outcome to the Client.
4. When every non-output qubit has been measured the Server sends the output qubits to the Client.
5. The Client calculates the output X- and Z-corrections that depend on the masking and the measurement outcomes and applies them to the output qubits.

Resource 1 1-of-2 Delegated Quantum Computation Resource.

1. Client sends $i \in \{0,1\}$ and ρ_C to the $\mathcal{R}$. The bit i is a control bit to tell $\mathcal{R}$ which unitary the Client wants to execute and ρ_C is an n-qubit long input to the requested unitary.
2. If $d = 1$ $\mathcal{R}$ sends the leakage l^{U_0,U_1} and l^{ρ_C} to the Server.
3. Case 1 ($c = 0$):

   (a) $\mathcal{R}$ responds with $U_i(\rho_C)$ to the Client.

   Case 2 ($c = 1$):

   (a) The Server sends $(\mathcal{E}, \rho_S)$ to $\mathcal{R}$.

   (b) $\mathcal{R}$ sends $\mathcal{E}(i, \rho_C, \rho_S)$ to the Client.

In our case the leakage l^{U_0,U_1} is defined to be the full description of U_0 and U_1 in terms of the default angles, resource graphs and g-flows. It also includes the length of the input and the output for both unitaries. The other part of the leakage: l^{ρ_C} is equal to the state of the input qubits that are public. The exact format of the deviation $(\mathcal{E}, \rho_S)$ is described in section 4.4.2.

$$i \in \{0,1\}, \psi_A \qquad \mathcal{S}^{\text{1-of-2-DQC}} \qquad b \in \{0,1\}$$

$$\rho = \begin{cases} U_i(\psi_A), & \text{if } b = 0, \\ \mathcal{E}(\psi_{AB}), & \text{if } b = 1. \end{cases} \qquad l^{i,\psi_A}$$
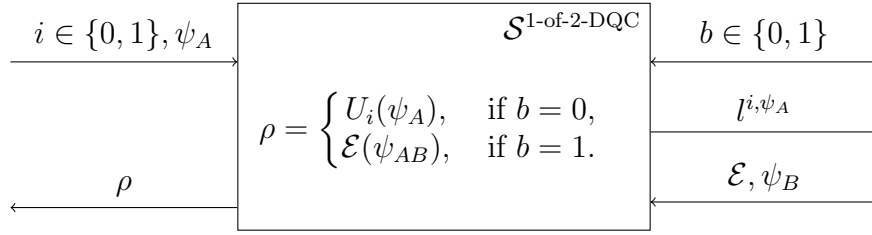
$$\rho \qquad \mathcal{E}, \psi_B$$

Figure 7: 1-of-2 DQC Resource. On the Client's interface, the Resource receives the input to the computation and sends the results. The Server's interface is inactive unless the Server decides to deviate maliciously from the protocol. If the latter is the case, the Server receives the leakage l^{i,ψ_A} from the resource and submits its deviation $(\mathcal{E}, \psi_B)$.

Remark 10. *If the protocol constructs the 1-of-2 DQC resource it provides perfect blindness for the Clients input against an unbounded adversary.*

4.4.2 Proof of Blindness.

Theorem 2. *The Protocols 4 and 5 perfectly construct the 1-of-2 DQC Resource 1 from a quantum channel.*

Proof. In what follows we are going to describe a sequence of reductions and mention in every line which steps of the Protocols 4 and 5 are being replaced. All lines that are not mentioned stay the same.

In Reduction 1 we replace the masked qubits which the Client prepares for the Server with EPR-pairs. One half of every pair the Client sends to the Server and the other half they keep as their state.

This approach allows us to delay the choice of secret parameters θ and r until later in the protocol. When the Server is requesting a measurement angle for one of those qubits the Client teleports there a correct value depending on the type of the qubit.

There are a few different cases to consider. Firstly, if the qubit is a bridge/break middle vertex the Client either teleports a $|0\rangle/|1\rangle$ state or $|+\rangle/|-\rangle$ to this node. This kind of qubits will be measured with angle $\frac{\pi}{2}$. If the qubit has to be deleted the Client teleports a $|0\rangle/|1\rangle$ state there as well. For these deleted qubits the Client also samples uniformly at random the parameters θ_v and r and asks the Server to measure it with angle $\delta = \varphi' + \theta_v + r\pi$. For the rest of the qubits the Client samples θ_v from the set associated to this node defined in Section 4.3 for this node and teleports the value $|+_{\theta_v + r\pi}\rangle$ to the qubit. Then this qubit is measured with an angle $\delta = \varphi' + \theta_v$. It is equivalent to measuring $|+_{\theta_v}\rangle$ with an angle $\delta = \varphi' + \theta_v + r\pi$ as in the original protocol.

It is easy to see that if the new parameters follow the uniform distributions as in the original protocol other steps of the computation are not affected by our modification. If for θ_v it is evident that the distribution is uniform we have to argue why r is also uniformly random when it is recovered as a measurement outcome.

The Server can manipulate their part of the EPR-pair which introduces changes to the Client's state. However Client's actions on their half of the pair do not depend on the messages sent earlier in the Protocol. It means that the situation is equivalent to not communicating at all. Hence, we can apply the No-Communication Theorem to justify why local operations on an entangled state do not transmit information. By no-communication theorem, all the measurement results that the Client obtains have the same distribution as if they shared a $\frac{1}{\sqrt{2}}(|+_{-\theta_v}\rangle|+_{\theta_v}\rangle + |-_{-\theta_v}\rangle|-_{\theta_v}\rangle)$ state with the Server. Therefore, when the Client measures their state in $|+_{-\theta_v}\rangle, |-_{-\theta_v}\rangle$ basis they are equally likely to obtain 0 or 1.

A similar argument holds to prove that the EPR half sent by the Client is indistinguishable from a correctly prepared $|+_{\theta_v}\rangle$ state, $|0\rangle/|1\rangle$ state or a $|+\rangle/|-\rangle$ state. Consider the following two situations. In the first one the Client simply sends the EPR half to the Server, and in the second one they measure their qubit to teleport a correct value to the Server's half and send it to the Server. Due to No-Communication Theorem the Server cannot distinguish between these two cases, so all the measurements the Server can do on their EPR half will give the same results as when the qubit was prepared correctly.

In conclusion we have the following:

- The parameters θ_v, r and δ are uniformly random from the corresponding sets. Hence, indistinguishable from original parameters.
- The operation order is different only on the side of the Client.
- The qubits that the Client sends to the Server are indistinguishable from correctly prepared qubits.
- After the modified steps the computation is exactly the same as in the original protocol for the secret parameters that we calculated.

That is why the protocol after the modification is perfectly equivalent to the original protocol from the point of view of the Distinguisher.

Reduction 1 Using EPR pairs.

1. (Lines 4b - 4f of PP.)
   (a) Public input qubits are sent to the Server.
   (b) For every other qubit that should be sent to the Server, the Client prepares an EPR-pair and sends one half to the Server.

2. (Line 1 of CP.) Break operation: the Client measures their half of the EPR-pair in the computational basis. If the outcome equals b - the Server has $|b\rangle$ state. Bridge operation: they need to apply H to their half of the EPR pair and measure it in the computational basis. If the measurement outcome is equal to r, the Server holds $\frac{1}{\sqrt{2}}(|0\rangle + (-1)^r|1\rangle)$ state.

3. (Lines 3(b)i of CP.)
   (a) The Client samples θ_v from the same set as in the original protocol.
   (b) (for deleted qubits) The Client measures their half of the EPR-pair in the computational basis. If the outcome equals b - the Server has $|b\rangle$ state. The Client sends the measurement angle for v: $\delta = \varphi' + \theta_v + r\pi$, where $r \xleftarrow{\$} \{0,1\}$.
   (c) (for non-deleted qubits) The Client measures the EPR pair that corresponds to v in basis $|+_{-\theta_v}\rangle, |-_{-\theta_v}\rangle$ with outcome r. Due to the teleportation principle the other half is now in the state $|+_{\theta_v+r\pi}\rangle$. The Client sends the measurement angle for v: $\delta = \varphi' + \theta_v$.

Let us now discuss Reduction 2. Here we let the Client first sample a random measurement angle δ and then calculate which secret parameters it would correspond to. This way, when we substitute the Client with the Simulator, the actions of the Simulator will not require access to Client's input or their secret keys.

To start, we note that the angles δ follow the same distribution as in the previous protocol. Let us now prove that we can change the order of the operations.

Suppose we are treating a non-deleted node. The angle $\theta_v = \delta - \varphi' - r\pi$ is calculated using the default angle, the angle δ that we just sent to the Server and the measurement outcomes we have already received from the Server. Hence, θ_v can be deduced up to a π rotation.

The operations we perform to our half of the EPR pair teleport a $|+_{\theta_v+r\pi}\rangle$ state to the Server's half. This qubit has already been measured, but the measurement is an operator of the form $(M(\delta) \otimes Id)(\rho_{1,2})$, where $\rho_{1,2}$ is the current state of the EPR pair. Therefore, it commutes with any operation we apply on the second qubit including $Id \otimes HZ(\theta_v)$. It means that our actions in the Reduction 2 are equivalent to first teleporting the correct value into the qubit and then performing the measurement (as we do in Reduction 1). The same proof applies to the deleted nodes and the bridge/break nodes.

We also make modifications of another kind. For the classically masked angles in lines 3(b)ii and 3c of CP we replace $\delta = \varphi' + r\pi$ with $\delta = \varphi + r\pi$. The angle φ is no longer updated with respect to the X and Z corrections. We know that if this angle is masked classically then $(\varphi' - \varphi) \in \{0, \pi\} \pmod{2\pi}$. So by adding $r\pi$ with a uniformly random r in both cases we make δ have the same distribution before and

after the modification. An updated procedure for dealing with the measurement outcome described in step 3c of Reduction 2 allows us to recover the correct result of the measurement. In conclusion, the two reductions are perfectly indistinguishable.

Reduction 2 Uniform measuring angles.

1. (Line 1 of CP.)
   (a) The Server measures all bridge/break vertices with angle $\frac{\pi}{2}$
   (b) Break operation: the Client measures their half of the EPR-pair in the computational basis. If the outcome equals b - the Server was holding $|b\rangle$ state. Bridge operation: they apply H to their half of the EPR pair and measure it in the computational basis. If the measurement outcome is equal to r, the Server was holding $\frac{1}{\sqrt{2}}(|0\rangle + (-1)^r|1\rangle)$ state.

2. (Line 3(b)i of CP.)
   (a) The Client samples δ from the set of possible angles for v and sends δ as a measurement angle for v.
   (b) The Client computes $\theta_v = \delta - \varphi'$.
   (c) (for deleted qubits) They measure their half of the EPR-pair in the computational basis. If the outcome equals b - the Server had $|b\rangle$ state. The measurement result does not depend on the Z-rotations so we can set r equal a random bit.
   (d) (for non-deleted qubits) They measure the qubit in $|+_{-\theta_v}\rangle, |-_{-\theta_v}\rangle$ with the outcome recorded into the variable r.

3. (Lines 3(b)ii and 3c of CP.)
   (a) The Client sends $\delta = \varphi + r\pi$ to the Server with $r \xleftarrow{\$} \{0, 1\}$. Note that now we are using the default angle φ.
   (b) The Server returns a measurement outcome s.
   (c) The Client computes φ' and if $\varphi + r\pi = \varphi' + \pi$ the Client flips s and keeps it for the future corrections; otherwise they keep s as it was.

As the last part of the proof we construct the Simulator that makes the Real world and the Ideal world protocol executions perfectly indistinguishable. Intuitively, in the ideal world we push the measurements of the Client's EPR halves to the very end of the protocol and delegate them to the Ideal Resource that knows Client's inputs.

With the information from the Simulator and the EPR halves the Ideal Resource reconstructs the output that the Client would have obtained if they ran the UBQC protocol with the Adversary in the real world. The set of Client's secret keys for this execution is decided by the Ideal Resource.

Here we give the definition of the Simulator 1 , the description of the deviation provided to the Resource and which computation the Resource performs to reconstruct the output.

It is easy to see that the messages the Simulator sends to the Adversary (run by a Distinguisher) have the same distributions as the messages after the Reduction 2.

In the description of the Resource 2 the reader can see how the information

---

Simulator 1 The UBQC protocol simulator.

---

1. For every public input qubit the Simulator transmits it to the Server in clear from the leakage.
2. For every other qubit that has to be sent to the Server the Simulator prepares an EPR pair and sends one half to the Server.
3. For every masked qubit v the Simulator does the following:
   (a) If v belongs to the case 3(b)i of CP the Simulator samples δ from a set of possible measurement angles for v and sends δ as a measurement angle for v.
   (b) Else if v belongs to case 3(b)ii of CP Simulator sends $\delta = \varphi + r\pi$ to the Server with $r \xleftarrow{\$} \{0, 1\}$.
   (c) Else if v belongs to case 3(b)iii of CP Simulator sends $\delta = \varphi'$ corresponding to the published measurement outcomes.
   (d) The Simulator receives $s \in \{0, 1\}$ from the Server.
4. The Simulator receives the output qubits from the Server.
5. In the end the Simulator sends to the 1-of-2 DQC Resource the following: the generated half EPR pairs, the output qubits, all measurement outcomes s, angles δ for the masked qubits, bits r from the case 3(b)ii.

---

which the Simulator obtains from the Adversary can be used to reconstruct real UBQC computation results including the deviation of the Adversary.

To see the correctness of this computation we argue that combining the Simulator and the Resource into one party that executes these two protocols sequentially is equivalent to the original protocol after the Reduction 2.

When the Resource reconstructs the computation its main objective is to recover the correct measurement outcomes one by one to know how to update the following angles and the following outcomes. Similarly to the Reduction 2 when treating a particular node the Resource has already computed a correct measurement outcome for every node in its past so now the Resource can determine φ' for the current node and follow the instructions in steps 4c - 4e to compute the correct value of s.

We proved that every modification of the protocols was perfectly indistinguishable. Therefore, the Protocol 4 and the Protocol 5 ϵ-construct the Ideal Resource defined in Resource 1 and Resource 2 for $\epsilon = 0$. This equivalence implies perfect blindness for the Clients input i and the private input qubits. $\qquad\square$

5 Conclusion and Future Directions

In this work, we have explored the fundamental question of whether it is possible to reduce the quantum communication complexity in universal blind quantum computing. Through a sequence of impossibility results, we showed that no quantum process, whether exact, approximate, separable or entangled, performed solely on the server side can distribute or expand encrypted resource states without violating blindness. These results go beyond the no-cloning theorem, revealing that even

Resource 2 1-of-2 Delegated Quantum Computation Resource. Full description.

1. The Resource receives from the Client the input qubits and the bit i that indicates which unitary they would like to perform.
2. From the Server's interface the Resource receives the information describing the deviation in the following form: a qubit for every masked qubit sent to the Server in the preparation phase, the output qubits, all measurement outcomes s, angles δ for the masked qubits, bits r from the case 3(b)ii.
3. For every bridge/break qubit:
   (a) Break operation: the Resource measures the corresponding in the computational basis and record the outcome b.
   (b) Bridge operation: the Resource applies H to the corresponding qubit and measures it in the computational basis and records the measurement outcome in r.
4. Then for every masked qubit the Resource does the following:
   (a) The Resource computes the correct φ' from the measurement outcomes received on the Server's interface and the correct measurement outcome the Resource recomputed.
   (b) The Resource computes $\theta_v = \delta - \varphi'$.
   (c) (for deleted nodes) The Resource measures the corresponding in the computational basis and records the outcome b. The resource also samples r at random. If $r = 1$ the Resource flips the measurement outcome s.
   (d) (for non-deleted nodes) The Resource measures the corresponding qubit in $|+_{-\theta_v}\rangle, |-_{-\theta_v}\rangle$ basis with outcome r. If $r = 1$ the Resource flips the measurement outcome s.
   (e) (for classical masking nodes) If $\delta = \varphi' + \pi$ the Resource flips the measurement outcome of this node.
5. The Resource now computes the final correction, applies it to the output qubits and outputs these qubits at the Client's interface.

approximate distributor machines introduce dependencies exploitable via differential attacks, thereby breaking security. The linearity of quantum mechanics, while being the root of both no-cloning and our no-distribution theorem, is not sufficient to reduce our result to the former, as the randomness distribution task is more general.

As a constructive result, we introduced a new paradigm we call selectively blind quantum computing (SBQC), which considers the case where a client wishes to delegate one computation from a known set, without revealing which one has been chosen. We formalized this in the language of composable cryptography and presented protocols that reduce the required number of quantum communications depending on the differences between the computation graphs required to implement various computations in the target set. The core idea is that only certain parts of the computation need to be hidden. We presented masking techniques for both angle differences and structural differences using a merged graph formalism.

A key insight from our protocol is the direct relationship between the placement of non-Clifford gates and the cost of hiding. In particular, non-Clifford gates are the ones that depending on their location within the circuit they might require masking via additional qubits sent from the client. If a circuit is compiled in such a way that these non Clifford gates are located in non-overlapping regions of the propagation cones of client's secret parameter then less masking is necessary leading to reduced quantum communication. This provides a new perspective on circuit compilation, distinct from existing optimization strategies for fault-tolerant computing (e.g., minimizing T-count or T-depth). Our approach introduces a new objective: structuring circuits for optimal hiding, opening a new direction in quantum compilation informed by security rather than error correction.

Although we focused exclusively on Blind quantum computing in this work, it would be important to explore whether these hiding techniques can be adapted to verifiable blind quantum computation protocols. Whether the same trade-offs apply, or new impossibility results arise in that context, remains an open problem. A crucial component of the proof of verifiability is Pauli twirling that cannot be applied to our protocol if some nodes do not have any masking. But in the private input scenario of our protocol, every qubit-masked node is masked by encrypting the measurement angle, and every other node is masked with an $r\pi$ rotation. One future direction would be to investigate whether this masking is sufficient for the Pauli twirling and making our protocol verifiable.

It is also natural to ask whether the notion of SBQC can scale efficiently beyond two computations, and what limitations emerge when attempting to balance leakage with communication cost. For example it is not clear what is the best strategy for merging two given graphs in an optimised way. Having this algorithm would allow us to estimate the improvement that our protocol gives in terms of the size of the entangled state of the Server side. More precisely, we expect a trade-off between the size of the resource graph and the qubit communication complexity. Although this problem seems difficult to solve for arbitrary graphs, it would already be very useful to find an optimal merger for $\mathbb{Z}_2$ lattices of different size or other regular graphs used in practice. Altogether, our work both constrains and extends the design space for blind quantum computation and introduces new tools to navigate it.

6   Acknowledgments

References

[1] A. Broadbent, J. Fitzsimons, and E. Kashefi, "Universal blind quantum computation," in *2009 50th annual IEEE symposium on foundations of computer science*, pp. 517–526, IEEE, 2009.

[2] P. Drmota, D. Nadlinger, D. Main, B. Nichol, E. Ainley, D. Leichtle, A. Mantri, E. Kashefi, R. Srinivas, G. Araneda, *et al.*, "Verifiable blind quantum computing with trapped ions and single photons," *Physical Review Letters*, vol. 132, no. 15, p. 150604, 2024.

[3] A. M. Childs, "Secure assisted quantum computation," *arXiv preprint quant-ph/0111046*, 2001.

[4] J. F. Fitzsimons and E. Kashefi, "Unconditionally verifiable blind quantum computation," *Physical Review A*, vol. 96, no. 1, p. 012303, 2017.

[5] U. Mahadev, "Classical verification of quantum computations," in *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*, pp. 259–267, IEEE, 2018.

[6] B. W. Reichardt, F. Unger, and U. Vazirani, "Classical command of quantum systems," *Nature*, vol. 496, no. 7446, pp. 456–460, 2013.

[7] C. Badertscher, A. Cojocaru, L. Colisson, E. Kashefi, D. Leichtle, A. Mantri, and P. Wallden, "Security limitations of classical-client delegated quantum computing," in *International Conference on the Theory and Application of Cryptology and Information Security*, pp. 667–696, Springer, 2020.

[8] S. Barz, J. F. Fitzsimons, E. Kashefi, and P. Walther, "Experimental verification of quantum computation," *Nature physics*, vol. 9, no. 11, pp. 727–731, 2013.

[9] S. Barz, E. Kashefi, A. Broadbent, J. F. Fitzsimons, A. Zeilinger, and P. Walther, "Demonstration of blind quantum computing," *science*, vol. 335, no. 6066, pp. 303–308, 2012.

[10] A. Mantri, T. F. Demarie, N. C. Menicucci, and J. F. Fitzsimons, "Flow ambiguity: A path towards classically driven blind quantum computation," *Physical Review X*, vol. 7, no. 3, p. 031004, 2017.

[11] B. Polacchi, D. Leichtle, L. Limongi, G. Carvacho, G. Milani, N. Spagnolo, M. Kaplan, F. Sciarrino, and E. Kashefi, "Multi-client distributed blind quantum computation with the qline architecture," *Nature Communications*, vol. 14, no. 1, p. 7743, 2023.

[12] C. A. Pérez-Delgado and J. F. Fitzsimons, "Iterated gate teleportation and blind quantum computation," *Physical review letters*, vol. 114, no. 22, p. 220502, 2015.

[13] R. Raussendorf and H. J. Briegel, "A one-way quantum computer," *Physical review letters*, vol. 86, no. 22, p. 5188, 2001.

[14] T. Morimae and K. Fujii, "Blind quantum computation protocol in which alice only makes measurements," *Physical Review A—Atomic, Molecular, and Optical Physics*, vol. 87, no. 5, p. 050301, 2013.

[15] C. Palazuelos and T. Vidick, "Survey on nonlocal games and operator space theory," *Journal of Mathematical Physics*, vol. 57, no. 1, 2016.

[16] U. Mahadev, "Classical homomorphic encryption for quantum circuits," *SIAM Journal on Computing*, vol. 52, no. 6, pp. FOCS18–189, 2020.

[17] A. Cojocaru, L. Colisson, E. Kashefi, and P. Wallden, "Qfactory: classically-instructed remote secret qubits preparation," in *Advances in Cryptology–ASIACRYPT 2019: 25th International Conference on the Theory and Application of Cryptology and Information Security, Kobe, Japan, December 8–12, 2019, Proceedings, Part I 25*, pp. 615–645, Springer, 2019.

[18] S. Aaronson, A. Cojocaru, A. Gheorghiu, and E. Kashefi, "On the implausibility of classical client blind quantum computing," in *7th International Conference on Quantum Cryptography*, 2017.

[19] J. Zhang, "Succinct blind quantum computation using a random oracle," in *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*, pp. 1370–1383, 2021.

[20] W. Li, S. Lu, and D.-L. Deng, "Quantum private distributed learning through blind quantum computing," *Preprint arXiv*, vol. 2103, 2021.

[21] M. C. Caro, J. Eisert, M. Hinsche, M. Ioannou, A. Nietner, and R. Sweke, "Interactive proofs for verifying (quantum) learning and testing," *arXiv preprint arXiv:2410.23969*, 2024.

[22] D. Mills, A. Pappa, T. Kapourniotis, and E. Kashefi, "Information theoretically secure hypothesis test for temporally unstructured quantum computation (extended abstract)," in *Proceedings 14th International Conference on Quantum Physics and Logic* (B. Coecke and A. Kissinger, eds.), vol. 266 of *Electronic Proceedings in Theoretical Computer Science*, pp. 209–221, Open Publishing Association, 2018.

[23] H. J. Briegel, D. E. Browne, W. Dür, R. Raussendorf, and M. Van den Nest, "Measurement-based quantum computation," *Nature Physics*, vol. 5, no. 1, pp. 19–26, 2009.

[24] A. Broadbent, "Delegating private quantum computations," *Canadian Journal of Physics*, vol. 93, no. 9, pp. 941–946, 2015.

[25] Y. Lee and D. Chung, "Partial blind quantum computation," 2025.

[26] W. F. Stinespring, "Positive functions on c*-algebras," *Proceedings of the American Mathematical Society*, vol. 6, no. 2, pp. 211–216, 1955.

[27] V. Danos, E. Kashefi, and P. Panangaden, "The measurement calculus," *J. ACM*, vol. 54, Apr. 2007.

[28] S. Perdrix, "State transfer instead of teleportation inmeasurement-based quantum computation," *International Journal of Quantum Information*, vol. 03, no. 01, pp. 219–223, 2005.

[29] D. E. Browne, E. Kashefi, M. Mhalla, and S. Perdrix, "Generalized flow and determinism in measurement-based quantum computation," *New Journal of Physics*, vol. 9, no. 8, p. 250, 2007.

[30] A. Broadbent, J. Fitzsimons, and E. Kashefi, *Measurement-Based and Universal Blind Quantum Computation*, pp. 43–86. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010.

[31] J. F. Fitzsimons, "Private quantum computation: an introduction to blind quantum computing and related protocols," *npj Quantum Information*, vol. 3, no. 1, p. 23, 2017.

[32] V. Dunjko, J. F. Fitzsimons, C. Portmann, and R. Renner, "Composable security of delegated quantum computation," in *Advances in Cryptology – ASIACRYPT 2014* (P. Sarkar and T. Iwata, eds.), (Berlin, Heidelberg), pp. 406–425, Springer Berlin Heidelberg, 2014.

[33] U. Maurer and R. Renner, "Abstract cryptography," in *Innovations in Computer Science*, pp. 1 – 21, Tsinghua University Press, jan 2011.

[34] U. Maurer, "Constructive cryptography – a new paradigm for security definitions and proofs," in *Theory of Security and Applications* (S. Mödersheim and C. Palamidessi, eds.), (Berlin, Heidelberg), pp. 33–56, Springer Berlin Heidelberg, 2012.

[35] W. K. Wootters and W. H. Zurek, "A single quantum cannot be cloned," *Nature*, vol. 299, no. 5886, pp. 802–803, 1982.

[36] V. Dunjko and E. Kashefi, "Blind quantum computing with two almost identical states," *arXiv preprint arXiv:1604.01586*, 2016.

[37] N. J. Cerf, "Asymmetric quantum cloning in any dimension," *Journal of modern optics*, vol. 47, no. 2-3, pp. 187–209, 2000.

[38] V. Scarani, S. Iblisdir, N. Gisin, and A. Acín, "Quantum cloning," *Reviews of Modern Physics*, vol. 77, no. 4, pp. 1225–1256, 2005.

[39] A. Biryukov and D. Khovratovich, "Related-key cryptanalysis of the full aes-192 and aes-256," in *Advances in Cryptology–ASIACRYPT 2009: 15th International Conference on the Theory and Application of Cryptology and Information Security, Tokyo, Japan, December 6-10, 2009. Proceedings 15*, pp. 1–18, Springer, 2009.

[40] E. Filiol, "Key-dependent security of stream ciphers," *arXiv preprint arXiv:2001.00515*, 2020.

[41] M. M. Wilde, *Quantum information theory*. Cambridge university press, 2013.

[42] H. Ollivier and W. H. Zurek, "Quantum discord: a measure of the quantumness of correlations," *Physical review letters*, vol. 88, no. 1, p. 017901, 2001.

[43] V. Giovannetti, L. Maccone, T. Morimae, and T. G. Rudolph, "Efficient universal blind quantum computation," *Physical review letters*, vol. 111, no. 23, p. 230501, 2013.

[44] M. A. Nielsen and I. L. Chuang, "Programmable quantum gate arrays," *Physical Review Letters*, vol. 79, no. 2, p. 321, 1997.

A Alternative Proof of Lemma 1

Here, we present an alternative proof of Equation (9) to highlight the connection between the lemma and the linearity of quantum mechanics

Proof. Assume the existence of an isometry as described. The output of this isometry takes the form $|+\theta_1\rangle \otimes |+\theta_2\rangle$. Expressing each state in the Hadamard basis, the resulting state can be written as: $|\psi\rangle = |+_{\theta_1}\rangle \otimes |+_{\theta_2}\rangle = a|++\rangle + b|+-\rangle + c|-+\rangle + d|--\rangle$ where the coefficients a, b, c, d satisfy the normalization condition. This implies that, by varying the input parameter θ, the isometry's output spans the full four-dimensional Hilbert space of two qubits:

$$D|+_\theta\rangle \in \text{span}\{|++\rangle, |+-\rangle, |-+\rangle, |--\rangle\} \tag{23}$$

Now, consider an alternative perspective on the output dimensionality. Decompose the input state as $|+_\theta\rangle = \alpha|+\rangle + \beta|-\rangle$. Using Equation (14) and the linearity of quantum mechanics, the output of the isometry becomes:

$$D|+_\theta\rangle = D(\alpha|+\rangle + \beta|-\rangle) = \alpha|++\rangle + \beta_2|-+\rangle + \gamma_2|--\rangle \in \text{span}\{|++\rangle, |-+\rangle, |--\rangle\} \tag{24}$$

where β_2 and γ_2 depend on the isometry's parameters.

A contradiction arises between two ways of evaluating the dimensionality of the output. The former suggests the output spans a four-dimensional space, while the latter indicates it is confined to a three-dimensional subspace. This inconsistency demonstrates a conflict between the assumed existence of the isometry in Equation and the linearity of quantum mechanics.

Furthermore, the lemma's validity extends beyond inputs of the form $|+_\theta\rangle$. Both proofs' mathematical arguments hold even when the input is a generic one-qubit state, rather than a planar state. If we generalize the input condition to an arbitrary state, the lemma's statement depends not on the number of parameters characterizing the input and output states, but on their functional dependence. Specifically, the condition in Equation (14) would be modified such that $D|-\rangle = |-\rangle \otimes |\phi\rangle$ replaces the second condition. This alteration again results in a dimensionality reduction, as the basis state $|+-\rangle$ is excluded from the output span.

Moreover, extending the output to multiple registers — e.g. requiring a pair of two-qubit product states (where each may be entangled) — is similarly unattainable. The same dimensionality arguments preclude achieving such an output configuration. Thus, no such isometry exists. □