Flow Ambiguity: A Path Towards Classically Driven Blind Quantum Computation

Atul Mantri,[1, 2, ∗] Tommaso F. Demarie,[1, 2, ∗] Nicolas C. Menicucci,[3, 4, †] and Joseph F. Fitzsimons[1, 2, ‡]

[1]*Singapore University of Technology and Design, 8 Somapah Road, Singapore 487372*
[2]*Centre for Quantum Technologies, National University of Singapore, Block S15, 3 Science Drive 2, Singapore 117543*
[3]*Centre for Quantum Computation and Communication Technology,*
School of Science, RMIT University, Melbourne, Victoria 3001, Australia
[4]*School of Physics, The University of Sydney, Sydney, New South Wales 2006, Australia*
(Dated: July 25, 2017)

Blind quantum computation protocols allow a user to delegate a computation to a remote quantum computer in such a way that the privacy of their computation is preserved, even from the device implementing the computation. To date, such protocols are only known for settings involving at least two quantum devices: either a user with some quantum capabilities and a remote quantum server or two or more entangled but noncommunicating servers. In this work, we take the first step towards the construction of a blind quantum computing protocol with a completely classical client and single quantum server. Specifically, we show how a classical client can exploit the ambiguity in the flow of information in measurement-based quantum computing to construct a protocol for hiding critical aspects of a computation delegated to a remote quantum computer. This ambiguity arises due to the fact that, for a fixed graph, there exist multiple choices of the input and output vertex sets that result in deterministic measurement patterns consistent with the same fixed total ordering of vertices. This allows a classical user, computing only measurement angles, to drive a measurement-based computation performed on a remote device while hiding critical aspects of the computation.

I. INTRODUCTION

Large-scale quantum computers offer the promise of quite extreme computational advantages over conventional computing technologies for a range of problems spanning cryptanalysis [1], simulation of physical systems [2], and machine learning [3]. Recently, however, a new application has emerged for quantum computers: secure delegated computation [4].

Consider a user wishing to have a computation performed on a remote server. Two main security concerns arising for the user relate to the *privacy* and the *correctness* of the computation. The privacy concern is that the description of their computation, both the program and any input data, remains hidden even from the server. The correctness concern is that a malicious server might tamper with their computation, sending them a misleading result: hence, ideally such behaviour would be detectable. Quantum protocols have been proposed that can mitigate both of these concerns. In the literature, protocols that allow for program and data privacy are known as *blind* quantum computing protocols, while protocols that allow for correctness to be ensured with high probability are known as *verifiable* quantum computing protocols [55].

The first blind quantum computing protocol was proposed by Childs [5]. While functional, this scheme put a rather heavy burden on the client's side in terms of resources, with the client required to control a quantum memory and to perform SWAP gates. A subsequent protocol, from Arrighi and Salvail [6], introduced mechanisms for both verification and blindness for a limited range of functions and can be seen as the start of an intimate link between blindness and verifiability. This link was further established with the discovery of the universal blind quantum computing (UBQC) protocol [7], which allows a client, equipped only with the ability to produce single-qubit states, to delegate an arbitrary quantum computation to a universal quantum server while making it blind with *unconditional security*.

This scheme has been modified and extended several times in the last few years, with works investigating robustness [8–10], optimality [11–13], and issues related to physical implementations [14, 15]. Importantly, the blind computation protocols have proven a powerful tool in the construction of verifiable quantum computing protocols, with a number of protocols emerging in recent years based on the UBQC protocol [16–18] and on an alternative blind protocol from Morimae and Fujii [19] in which the client performs single-qubit measurements rather than state preparations [20–22]. The relatively low overhead in such schemes has made it possible to implement both blind and verifiable quantum computing protocols in quantum optics [23–25].

The question of verifiability, directly rather than as a consequence of blindness, has also attracted attention. This problem was first studied by Aharonov *et al.* [26], who considered the use of a constant-sized quantum computer to verify a larger device. Subsequent work by Broadbent [27] reduced the requirements on the prover to mirror those used in the UBQC protocol. An entirely distinct route to verification has also emerged, which considers a classical user but requires multiple entangled but noncommunicating servers [28, 29]. Surprisingly, perhaps, many of these schemes are also blind, though often this was not the aim of the paper. In fact, only a few examples of verifiable computing schemes exist that are not naturally blind [30, 31], and it is tempting to conjecture a fundamental link between blindness and verifiability.

The verification methods discussed above provide a very

∗These authors contributed equally to this work.
†ncmenicucci@gmail.com
‡joseph_fitzsimons@sutd.edu.sg

strong form of certification, amounting to interactive proofs for correctness which do not rely on any assumptions about the functioning of the device to be tested. From an experimental point of view, the first nonclassically simulable evolution of quantum systems will most likely be implemented by means of nonuniversal quantum simulators rather than fully universal quantum computers. Here, too, the problem arises of certifying the correct functioning of a device [32] that cannot be efficiently simulated. However, in this regime interactive proofs have proven more difficult to construct. Nonetheless, progress has been made in developing a range of certification techniques for various physical systems. These include feasible quantum state tomography of matrix product states [33], certification of the experimental preparation of resources for photonic quantum technologies [34], certification of simulators of frustration-free Hamiltonians [35], and derivation of a statistical benchmark for boson sampling experiments [36].

A common feature among all blind quantum computing protocols and interactive proofs of correctness for quantum computation is that they require that at least two parties possess quantum capabilities. Removing this requirement and allowing a purely classical user to interact with a single quantum server would greatly expand the practicality of delegated quantum computation since it would remove large-scale quantum networks as a prerequisite for verifiability. In the present work, we focus specifically on the question of blind computation with a completely classical client, but given the historic links between progress in blindness and verification, it is natural to expect that progress in either direction will likely be reflected in the other.

While it is presently unknown if such a protocol can exist, a negative result in this context is a *scheme-dependent* impossibility proof presented in Ref. [37]. There, the author considered a scenario where a classical user and a quantum server exchange classical information in a two-step process. First, the classical client encodes their description of the computation using an affine encryption scheme and then sends all the classical encrypted data to the server. The server then performs a quantum computation using the received data and returns the classical output to the client who decrypts the result using their encryption key. For this setting, it was shown that secure blind quantum computation cannot be achieved unless BPP = BQP (i.e., unless a classical computer can efficiently simulate a quantum computer). While this is an interesting result, it imposes strong assumptions on the operational method of blind quantum computation with a classical client and therefore does not seem to limit further studies in this direction. Additionally, Aaronson *et al.* [38] have recently suggested that information-theoretically blind quantum computing with a classical client is not likely to be possible because the existence of such a scheme implies unlikely containments between complexity classes. Additional implications that the development of a classical-client blind-computation protocol would have in complexity theory are discussed in Ref. [39].

Here, we provide evidence in the opposite direction. We introduce a form of delegated quantum computation using measurement-based quantum computing (MBQC) as the underlying framework. This allows us to introduce a model-

specific protocol that achieves a satisfying degree of security by directly exploiting the structure of MBQC. We show that the classical communication received by the party performing quantum operations is insufficient to reconstruct a description of the computation. This insufficiency remains even when the server is required only to identify the computation up to pre- and post-processing by polynomial-sized classical computation, under plausible complexity-theoretic assumptions. We call our scheme *classically driven blind quantum computing (CDBQC)*.

The paper is structured as follows: In Sec. II, to help the readers unfamiliar with MBQC, we present a short introduction to this model of quantum computation. In Sec. III we describe the steps of the CDBQC protocol. In Sec. IV we use mutual information to analyse the degree of blindness for a single round of the CDBQC protocol. In Sec. V we introduce the concept of flow ambiguity and we show how this is used by the client to hide information from the quantum server. Our conclusions are presented in Sec. VI.

II. MEASUREMENT BASED QUANTUM COMPUTATION

In MBQC, a computation is performed by means of single-qubit projective measurements that drive the quantum information across a highly entangled resource state. The most general resources for MBQC are graph states [40]. A graph state is defined by a simple and undirected graph, i.e., a mathematical object $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ composed of a vertex set $\mathcal{V}$ and an edge set $\mathcal{E}$, with cardinality $|\mathcal{V}|$ and $|\mathcal{E}|$, respectively. The vertices of the graph represent the qubits, while their interactions are symbolized by the edges. A graph state $|\mathcal{G}\rangle$ is an N-qubit state, where $N = |\mathcal{V}|$. Each qubit is initialized in the state $|+\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$ and then entangled with its neighbors by controlled-Z gates, $\hat{C}_{Z\,i,j} = |0\rangle\langle 0|_i \otimes \hat{I}_j + |1\rangle\langle 1|_i \otimes \hat{Z}_j$, where $\hat{I}$ and $\hat{Z}$ are the single-qubit identity and Pauli-Z gate respectively. Explicitly, $|\mathcal{G}\rangle = \prod_{(i,j)\in\mathcal{E}} \hat{C}_{Z\,i,j}|+\rangle^{\otimes N}$. Equivalently, a graph state $|\mathcal{G}\rangle$ can be defined by the stabilizer relations $\hat{K}_v|\mathcal{G}\rangle = |\mathcal{G}\rangle$, with stabilizers [40]

$$\hat{K}_v = \hat{X}_v \prod_{w\in\mathcal{N}(v)} \hat{Z}_w, \qquad \forall v \in \mathcal{V}, \tag{1}$$

where $\mathcal{N}(v)$ denotes the neighborhood of v in $\mathcal{G}$. Without loss of generality, the vertices in $\mathcal{G}$ can be labeled $(1,\ldots,N)$ in the order that the corresponding qubits are to be measured. We take this ordering to be implicit in the definition of the graph, for example as the order in which the vertices appear in the adjacency matrix for $\mathcal{G}$. It is also useful to define a specific type of graph state that will be used later. An N-qubit *cluster state* $|\text{CS}\rangle_{n,m}$ is the graph state corresponding to an $n \times m$ regular square-lattice graph $\mathcal{G}_{n,m}$. For such a graph, $N = nm$.

In the MBQC framework, given a resource state with graph $\mathcal{G}$, the standard procedure to perform a computation is to first identify two sets of qubits $\{I, O\}$ on $\mathcal{G}$. This procedure defines an *open graph* $\mathcal{G}(I, O)$, such that $I, O \subseteq \mathcal{V}$ for a given

$\mathcal{G}$. The set I corresponds to the input set, while O denotes the output set. In general, $0 < |I| \leq |O| \leq |\mathcal{V}|$. Note that the input and output sets can overlap. The complement of I is written I^c, and similarly, the complement of O is O^c. We also denote by $P(I^c)$ the power set of all the subsets of elements in I^c, and we define

$$\text{Odd}(K) \coloneqq \{i : |\mathcal{N}(i) \cap K| = 1 \mod 2\} \qquad (2)$$

as the odd neighborhood of a set of vertices $K \subseteq \mathcal{V}$. In this work, we are only interested in MBQC protocols that implement unitary embeddings. Hence for us, $|I| = |O| \leq N$. Intuitively, the state of the qubits in the input set corresponds to the input state of a computation. Similarly, the qubits in the output set will contain the quantum information corresponding to the result of the computation once all the qubits in O^c have been measured. In the process, the quantum information is transformed by the same principle that governs the generalized one-bit teleportation scheme [41, 42].

For our purposes, we restrict the measurements to be projective measurements in the XY-plane of the Bloch sphere, denoted $M_j^{\alpha_j} = \{|\pm_\alpha\rangle\langle\pm_\alpha|_j\}$ for qubit j, where $|\pm_\alpha\rangle = \frac{1}{\sqrt{2}}(|0\rangle \pm e^{i\alpha}|1\rangle)$. As a convention, we use $b_j = 0$ for the measured qubit collapsing to $|+_\alpha\rangle_j$ and $b_j = 1$ for collapsing to $|-_\alpha\rangle_j$. The computation to be performed is specified both by the choice of open graph $\mathcal{G}(I, O)$ and by a vector $\boldsymbol{\alpha}$ specifying the measurement basis α_i for each qubit i. Note that these are the measurements *that would be made directly on the cluster state if all the measurement outcomes were zero for non-output qubits*—i.e., if one were to implement the positive branch of the MBQC computation. Importantly, by convention, the positive branch corresponds to the target computation. In general, however, these bases need to be updated based on outcomes of earlier measurements in order to ensure the correct computation is performed. The description of the resource state, the order of measurements, and the dependency of the measurement bases on previous measurement outcomes are collectively known as a *measurement pattern*.

Projective measurements are inherently random in quantum mechanics, and one needs a procedure to correct for this randomness. We show that this need for adaptation of future measurements based on previous outcomes is what prevents Bob from knowing the protocol perfectly. Not incidentally, it is also what circumvents the no-go result from Ref. [37]. This is because only Alice knows how she is choosing to adapt future measurement bases dependent on previous measurement outcomes: Our observation is that different choices of adaptation strategy correspond to different computations in general.

The structure that determines how to recover deterministic evolution from a MBQC measurement pattern is called g-flow [43], from *generalized quantum-information flow*. Rigorously, given some resource state $|\mathcal{G}\rangle$ and a measurement pattern on it, if the associated open graph $\mathcal{G}(I, O)$ satisfies certain g-flow conditions (to be described later), then the pattern is runnable, and it is also uniformly, strongly, and stepwise deterministic. This means that each branch of the pattern can be made equal to the positive branch after each measurement by application of local corrections, independently of the measurement angles. We use *deterministic* without ambiguity to indicate all these attributes. Note also that satisfying the g-flow conditions is a necessary and sufficient condition for determinism.

In practice, the g-flow assigns a set of local Pauli corrections to a subset of unmeasured qubits after a measurement. See Ref. [44] for the fine details regarding the practicalities of g-flow. For simplicity, in the definition of g-flow below adapted from Ref. [43], we assume all qubits are measured in the XY-plane of the Bloch sphere. The idea behind g-flow is to determine whether one can find a correction operator (related to a correcting set on the graph) that, in the case of a nonzero measurement outcome, can bring back the quantum state onto the projection corresponding to the zero outcome. This is done by applying stabilizer operators on the state. The g-flow conditions determine whether the geometrical structure of an open graph allows for these corrections after each measurement.

Definition 1 (G-flow). For an open graph $\mathcal{G}(I, O)$, there exists a *g-flow* $(g, \succ)$ if one can define a function $g : O^c \to P(I^c)$ and a partial order $\succ$ on $\mathcal{V}$ such that $\forall i \in O^c$, all of the following conditions hold:

(G1) if $j \in g(i)$ and $j \neq i$, then $j \succ i$;
(G2) if $j \not\succ i$ and $i \neq j$, then $j \notin \text{Odd}(g(i))$; and
(G3) $i \notin g(i)$ and $i \in \text{Odd}(g(i))$.

The successor function $g(i)$ indicates what measurements will be affected by the outcome of the measurement of qubit i, while the partial order $\succ$ should be thought of as the causal order of measurements. The condition (G1) says that if a vertex j is in the correcting set of the vertex i, then j should be measured after the vertex i. In other words, a correction should happen after the assigned measurement. Condition (G2) makes sure that if the correcting set of a vertex i is connected to a vertex j, and j is measured before the vertex i, then the vertex j should have an even number of connections with the correcting set of vertex i. Then vertex j receives an even number of equal Pauli corrections, which is equivalent to receiving none: hence, no correction can affect earlier corrections. Finally, condition (G3) certifies that each vertex i has an odd number of connections with its correcting set, such that a correction is indeed performed on i [44]. In this sense, the g-flow conditions are understood in terms of geometrical conditions on the open graph.

Guided by these conditions, for cluster states, here and in the following, we always adopt the same choice of vertex labeling on the graph as shown in Fig. 1. This choice is motivated by our later goal of counting how many choices of open graphs satisfy the g-flow conditions on a given cluster state. Since a vertex labeling corresponds to a total order of measurement, it is easy to check that, in order to satisfy the g-flow conditions, for any vertex i the quantum information can only move towards the right, move towards the bottom, or stay on that vertex. Furthermore, condition (G3) imposes that the information from a vertex cannot move simultaneously towards the right and towards the bottom. In order to further simplify the process of counting flows, we introduce an additional criterion, which is not strictly required by g-flow:

(G4) If $k \in \mathcal{N}(i) \cup \mathcal{N}(j)$, and if $k \in g(i)$, then $k \notin g(j)$.

For $\mathcal{G}_{n,m}$, as we shall see later, it will prove easier to count flows satisfying (G1)–(G4) than those satisfying (G1)–(G3). This process of course only provides a lower bound on the number of flows rather than the exact number, but will be sufficient for our purposes.

With these four criteria in place, we can define a g-flow graph path, in this restricted version of g-flow, as an ordered set of adjacent edges of the graph, starting from an element of the input set and ending on an element of the output set, such that for each edge ij of the path, we have $j \in g(i)$, with $j \succ i$. Then it follows that (G4) does not allow the g-flow graph paths to cross. To help the understanding of MBQC, one could think of a g-flow graph path as a representation of a wire in the quantum circuit picture. This intuition will be used later in this work to count how many ways one could define an open graph with g-flow for our choice of total ordering, which in turn provides a link to the idea that different open graphs with g-flow lead to different quantum computations.

III. CLASSICALLY DRIVEN BLIND QUANTUM COMPUTATION

We start from the situation where Alice wants to obtain the result of a particular quantum computation. Having no quantum devices of her own, the quantum computation must have a classical output. We allow Alice to control a probabilistic polynomial-time universal Turing machine (i.e., a classical computer with access to randomness). Alice has classical communication lines to and from Bob, the server. Bob has access to a universal (and noiseless) quantum computer. Bob could help Alice, but she does not trust him. Alice wishes to ask Bob to perform a quantum computation for her in a way that Bob obtains as little information as possible about her choice of computation. Without loss of generality, we assume that the quantum systems used in the protocol are qubits (two-level quantum systems [45]). In general, here and in the following, we denote by

$$\Delta_{\mathcal{A}} = \{\rho_I, \hat{U}_A, \mathcal{M}\} \qquad (3)$$

the classical description of Alice's computation, where ρ_I is the n-qubit input state of the computation, $\hat{U}_A$ is the unitary embedding that maps ρ_I to the output state $\rho_O = \hat{U}_A \rho_I \hat{U}_A^\dagger$, and $\mathcal{M}$ is the final set of measurements on ρ_O required to extract the classical output. Note that we are implying that the input state can be efficiently described classically. For instance, it could be a standard choice of input such as the n-qubit computational basis $\rho_I = |0\rangle\langle 0|^{\otimes n}$. We also (rather pedantically) assume that the number of computational steps is at most polynomial in the input size. Making the process abstract, Alice's desired task becomes equal to sampling the string

$$\mathbf{p} = \{p_i\} = \pi(\Delta_{\mathcal{A}}) := \mathcal{M}(\hat{U}_A \rho_I \hat{U}_A^\dagger), \qquad (4)$$

where π is a map that describes the blind operation performed by the protocol, which outputs the correct probability distri-

---

Protocol 1 CDBQC$(\mathcal{G}, \mathcal{A})$: Classically Driven Blind Quantum Computation

Protocol parameters:
- A graph $\mathcal{G}$ with an implicit total ordering of vertices.
- A set of angles $\mathcal{A}$ satisfying Eq. (19).

Alice's input:
- A target computation $\Delta_{\mathcal{A}}$ implemented using MBQC as
$$\Delta_{\mathcal{A}}^M = \{\mathcal{G}, \boldsymbol{\alpha}, \mathbf{f}\},$$
representing a measurement pattern on $\mathcal{G}$ compatible with the total ordering of measurements implicit in $\mathcal{G}$, which describes a unitary embedding $\hat{U}_A$. The set $\boldsymbol{\alpha}$ represents a sequence of N measurement angles over the graph $\mathcal{G}$, with each angle chosen from a set $\mathcal{A}$, which is also taken to be a parameter of the protocol and is known to both parties. The g-flow construction $\mathbf{f}$ fully determines the input state ρ_{in}, through the location of the input and output qubit sets on the graph (I and O, respectively) and the dependency sets $(\mathbf{s}^x, \mathbf{s}^z)$

Steps of the protocol:

1. **State preparation**

   (a) Bob prepares the graph state $|\mathcal{G}\rangle$.

2. **Measurements**
   For $i = 1, \ldots, N$, repeat the following:

   (a) Alice picks a binary digit $r_i \in \mathbb{Z}_2$ uniformly at random. Then, using $r_i, \mathbf{s}^x, \mathbf{s}^z$, and the function in Eq. (18), she computes the angle α_i'. Alice transmits α_i' to Bob.

   (b) Bob measures the ith qubit in the basis $\{|\pm_{\alpha_i'}\rangle\}$ and transmits to Alice the measurement outcome $b_i' \in \mathbb{Z}_2$.

   (c) Alice records $b_i = b_i' \oplus r_i$ in $\mathbf{b}$ and then updates the dependency sets $(\mathbf{s}^x, \mathbf{s}^z)$. If $i \in O$, then she also records b_i in $\mathbf{p}_{\mathcal{B}}^C$.

3. **Post-processing of the output**

   (a) Alice implements the final round of corrections on the output string by calculating $\mathbf{p} = \mathbf{p}^C \oplus \mathbf{s}_O^Z$, with $\mathbf{s}_O^Z$ the set of Z corrections on the output at the end of the protocol.

---

bution $\{p_i\}$ on the joint measurement outcomes given $\Delta_{\mathcal{A}}$ as Alice's delegated target computation. An outline of the protocol is presented in Protocol 1.

Let us now introduce the relevant definitions for the variables used in the protocol and describe the steps thoroughly. The initial step of the protocol is for Bob to prepare the resource state $|\mathcal{G}\rangle$ that will be used to implement the MBQC. Once the graph state $|\mathcal{G}\rangle$ is prepared by Bob, the interactive part of the protocol starts with Alice communicating to Bob the angles to be measured, one by one. Because of the randomness introduced by the results of the projective measurements, there exists the possibility that these angles must be corrected based on the outcomes

$$\mathbf{b} := (b_1, ..., b_N) \in \mathbb{Z}_2^N \qquad (5)$$

of Alice's would-be measurements. Nonetheless, Alice can

pick a canonical set of angles

$$\boldsymbol{\alpha} := (\alpha_1, \ldots, \alpha_N) \in \mathcal{A}^N \qquad (6)$$

corresponding to the positive branch case where $\mathbf{b} = \mathbf{0}$. As discussed earlier, it is possible that the angle for qubit j must be modified based on the outcomes of the preceding $j - 1$ measurements, which we denote:

$$\mathbf{b}_{<j} := (b_1, \ldots, b_{j-1}). \qquad (7)$$

We account for this adaptation in *dependency sets*

$$\mathbf{s}^x := (s_1^x, \ldots, s_N^x) \in \mathbb{Z}_2^N, \qquad (8)$$
$$\mathbf{s}^z := (s_1^z, \ldots, s_N^z) \in \mathbb{Z}_2^N, \qquad (9)$$

which depend on the $\mathbf{b}_{<j}$ and also on the g-flow construction, here represented by a bit string

$$\mathbf{f} := (f_1, \ldots, f_M) \in \mathbb{Z}_2^M \qquad (10)$$

of length M called the *flow control bits* (or just "flow bits"). At this point, we still have to quantify the value of M. Note, though, that it represents the number of bits needed to enumerate all the possible combinations of input and output that satisfy the g-flow conditions [43, 46]. Hence, for a fixed total order of the measurements, it is a function of N. Explicitly, the X and Z corrections associated with the measurement angle of each qubit j are determined by the dependency sets:

$$s_j^x : \mathcal{D}[\mathbf{b}_{<j}] \times \mathcal{D}[\mathbf{f}] \to \mathbb{Z}_2, \qquad (11)$$
$$s_j^z : \mathcal{D}[\mathbf{b}_{<j}] \times \mathcal{D}[\mathbf{f}] \to \mathbb{Z}_2, \qquad (12)$$

where the function $\mathcal{D}$ denotes the domain of the argument. Without loss of generality, we choose $s_1^z = s_1^x = 0$ since there are no previous outcomes on which these could depend. For a fixed open graph $\mathcal{G}(I, O)$, the form of the dependency sets is uniquely defined by the g-flow [4]. Analogously, the flow bits $\mathbf{f}$ fully specify the dependency sets (as functions of $\mathbf{b}$). As such, the quantum circuit that Alice intends to implement is specified by the information

$$(\boldsymbol{\alpha}, \mathbf{f}) \in \mathcal{A}^N \times \mathbb{Z}_2^M, \qquad (13)$$

consisting of N measurement angles and M flow bits for a given graph with fixed total order of measurement. Consequently, once the graph $\mathcal{G}$ is known, there exists a one-to-one correspondence $\mathcal{G}(I, O)_{n,m} \leftrightarrow \mathbf{f}$, and we can accordingly denote the corresponding MBQC measurement pattern as follows:

$$\Delta_{\mathcal{A}}^M = (\mathcal{G}_{n,m}, \boldsymbol{\alpha}, \mathbf{f}). \qquad (14)$$

Explicitly, note that by choosing $\mathbf{f}$, Alice is defining a unique choice of the input and output on the graph state before the protocol begins. In line with the computation description from Eq. (3) we call ρ_I the input state on $\mathcal{G}$.

We now turn our attention to what kind of information Bob receives when Alice asks him to perform the measurements on her behalf. The interactive part of the protocol consists of N steps. At each step i, Alice requests Bob to measure, in the XY-plane of the Bloch sphere, the ith qubit, according to the total order implied by $\mathcal{G}$, and he sends back a bit for each measurement. We identify the measurement instructions Bob receives as a list of angles

$$\boldsymbol{\alpha}' := (\alpha_1', \ldots, \alpha_N') \in \mathcal{A}^N, \qquad (15)$$

and we label the string of bits Alice receives from Bob as

$$\mathbf{b}' := (b_1', \ldots, b_N') \in \mathbb{Z}_2^N, \qquad (16)$$

while remembering that they are communicated alternately (α_1' to Bob, b_1' to Alice, α_2' to Bob, b_2' to Alice, etc.). Note that in the case of a dishonest Bob, the string $\mathbf{b}'$ does not need to correspond to real measurement outcomes but could have been generated by Bob through some alternative process.

Realizing that measuring α can just as easily be effected by asking Bob to measure $\alpha + \pi$ and then flipping the returned outcome bit, we introduce a *uniformly random N-bit string*

$$\mathbf{r} := (r_1, \ldots, r_N) \in \mathbb{Z}_2^N \qquad (17)$$

that Alice will use to pad the angles in an attempt to conceal the measurement outcomes. All that remains is to specify how $\boldsymbol{\alpha}'$ depends on $\boldsymbol{\alpha}$. This is specified by the following functional dependence [16, 46, 47]:

$$\boldsymbol{\alpha}' = (-1)^{\mathbf{s}^x} \boldsymbol{\alpha} + (\mathbf{s}^z + \mathbf{r})\pi \mod 2\pi, \qquad (18)$$

which follows from the g-flow construction and shows how corrections change subsequent measurement angles. Here we have used multi-index notation to present the result concisely as a vector. Note that the dependency sets $(\mathbf{s}^x, \mathbf{s}^z)$ are updated by Alice after each measurement. To make the analysis of the protocol meaningful, we construct a domain for $\boldsymbol{\alpha}$ such that the domain of all valid $\boldsymbol{\alpha}'$ is the same. Thus, in general

$$\mathcal{A} = \left\{ (-1)^x \theta + z\pi \mid \theta \in \mathcal{A}, x \in \mathbb{Z}_2, z \in \mathbb{Z}_2 \right\}. \qquad (19)$$

Also note that now

$$\mathbf{b}' = \mathbf{b} \oplus \mathbf{r}, \qquad (20)$$

where $\oplus$ indicates addition modulo 2 for each bit. We can identify the *data* that Bob receives during the interactive part of the protocol (some from Alice, some from his own measurements) as:

$$(\text{data Bob receives}) := (\mathbf{b}', \boldsymbol{\alpha}') \in \mathbb{Z}_2^N \times \mathcal{A}^N. \qquad (21)$$

The interactive part of the protocol ends when all the qubits have been measured and Alice holds the binary register $\mathbf{b}$, derived from $\mathbf{b}'$ to account for the one-time pad $\mathbf{r}$. Since Alice knows the output set O, whenever the ith qubit belongs to the set of output qubits, Alice saves b_i into a second binary sequence of length $|O|$:

$$\mathbf{p}_{\mathcal{G}}^C := (p_1, \ldots, p_{|O|}) \in \mathbb{Z}_2^{|O|}, \qquad (22)$$

where $p_i = b_i, \forall i \in O$. If Bob is honest, then $\mathbf{p}_{\mathcal{B}}^C$ is equivalent to $\mathbf{p}^C$. At the end of the protocol, this string contains the classical result of the computation, up to classical post-processing. This is accounted for by calculating $\mathbf{p} = \mathbf{p}^C \oplus \mathbf{s}_O^Z$, where $\mathbf{s}_O^Z$ is used to represent the final set of Z corrections on the output qubits. Clearly, the classical nature of the client allows us to consider only quantum computations with classical output.

In order for the protocol to have any utility, we require that the output $\mathbf{p}$ satisfies Eq. (4), a property known as correctness. The correctness of this protocol can be proved straightforwardly. Note that the positive branch of the MBQC pattern $\Delta_{\mathcal{A}}^M$ [that is, where all the measurement outcomes happen to be equal to zero ($\mathbf{b} = \mathbf{0}$)] implements Alice's target computation $\Delta_{\mathcal{A}}$ by definition. In the circuit model, this corresponds to a quantum circuit that implements the unitary $\hat{U}_A$ over the correct input state ρ_I and a final round of measurements whose output is the binary string $\mathbf{p}$ [48]. Below, we give a proof of the correctness of the CDBQC protocol.

Theorem 1 (Correctness). *For honest Alice and Bob, the outcome of Protocol 1 is correct.*

Proof. There are only two differences between Protocol 1 and a conventional MBQC implementation of $\Delta_{\mathcal{A}}^M$. The first is the use of $\mathbf{r}$ to hide measurement outcomes. The effect of $\mathbf{r}$ is to add an additional π to the measurement angle on certain qubits, resulting in a bit flip on the corresponding measurement result b_i'. However, since this is immediately undone, it has no effect on the statistics of the measurement results obtained after decoding $\mathbf{b}$.

The other difference is that the g-flow construction, and hence the dependency sets, is only known to Alice and not to Bob. However, this does not affect the input state, which is equivalent to the usual case if Alice is honest (i.e., if she correctly performs her role in implementing the protocol). Furthermore, if Alice updates the measurement angles correctly using the dependency sets as dictated by the g-flow, and Bob measures them accordingly, every branch of $\Delta_{\mathcal{A}}^M$ is equivalent to the positive branch. Then the measurement pattern correctly implements the unitary transformation $\rho_{out} = \hat{U}_A \rho_{in} \hat{U}_A^\dagger$. The protocol also allows Alice to identify the elements of the output string $\mathbf{p}^C$ in $\mathbf{b}$, since she knows the position of the output on the graph. Hence, when both Alice and Bob follow the protocol, the output string $\mathbf{p} = \mathbf{p}^C \oplus \mathbf{s}_O^Z$ is the desired probability distribution that follows from the joint measurement of the correct quantum output. $\square$

IV. BLINDNESS ANALYSIS

We now look at the degree of blindness for a single round of Protocol 1. In this setting, we consider a cheating Bob with unbounded computational power, able to deviate from the protocol and follow any strategy allowed by the laws of physics. Our aim, however, is not to verify that Bob is indeed performing the correct quantum computation as requested. Instead, we want to quantify the amount of information that Bob can access when Protocol 1 is run only once (stand-alone) and

compare it against the total amount of information needed to describe the computation. To completely identify Alice's computation, Bob needs to know the description $\Delta_{\mathcal{A}}^M$.

We identify variables with uppercase letters and particular instances of such variables with lowercase letters. The probability of a given instance $\mathbf{x}$ of a random variable $\mathbf{X}$ is denoted $\Pr(\mathbf{x})$, and averaging over $\mathbf{X}$ is denoted $\langle \cdot \rangle_{\mathbf{X}}$ or $\langle \cdot \rangle$ when there is no ambiguity. Given a random variable $\mathbf{X}$, we call $N_{\mathbf{X}}$ the number of possible outcomes for the variable and $n_{\mathbf{X}} := \log_2 N_{\mathbf{X}}$ the number of bits required to enumerate them.

We denote the Shannon entropy [49] of a random variable $\mathbf{X}$ by $H(\mathbf{X}) := \langle -\log_2 \Pr(\mathbf{x}) \rangle_{\mathbf{X}} \leq n_{\mathbf{X}}$, with equality if and only if $\mathbf{X}$ is uniformly random. For two random variables $\mathbf{X}$ and $\mathbf{Y}$, their joint entropy is written $H(\mathbf{X}, \mathbf{Y}) := \langle -\log_2 \Pr(\mathbf{x}, \mathbf{y}) \rangle_{\mathbf{X}, \mathbf{Y}}$, and the conditional entropy of $\mathbf{X}$ given $\mathbf{Y}$ is $H(\mathbf{X}|\mathbf{Y}) := \langle -\log_2 \Pr(\mathbf{x}|\mathbf{y}) \rangle_{\mathbf{X}, \mathbf{Y}}$. These satisfy

$$H(\mathbf{X}|\mathbf{Y}) = H(\mathbf{X}, \mathbf{Y}) - H(\mathbf{Y}). \qquad (23)$$

The mutual information of $\mathbf{X}$ and $\mathbf{Y}$ is

$$\begin{aligned} I(\mathbf{X}; \mathbf{Y}) &:= H(\mathbf{X}) + H(\mathbf{Y}) - H(\mathbf{X}, \mathbf{Y}) \\ &= H(\mathbf{X}) - H(\mathbf{X}|\mathbf{Y}) \\ &= H(\mathbf{Y}) - H(\mathbf{Y}|\mathbf{X}), \qquad (24) \end{aligned}$$

which will be our main tool of analysis. Intuitively, $I(\mathbf{X}; \mathbf{Y})$ measures how much information $\mathbf{Y}$ has about $\mathbf{X}$. More precisely, it quantifies how much the entropy of $\mathbf{X}$ is reduced, on average, when the value of $\mathbf{Y}$ is known. Because of the symmetry of the definition, these statements also hold when the roles of $\mathbf{X}$ and $\mathbf{Y}$ are swapped.

Let us call the angles variable $\mathbf{A}$ and the flow variable $\mathbf{F}$. Specifying $\Delta_{\mathcal{A}}^M$, in general, therefore requires $n_{\mathbf{A}} + n_{\mathbf{F}}$ bits. In addition, we use $\mathbf{B}$ for the eventual measurement outcomes and $\mathbf{R}$ for the (uniformly random) string of π-shift bits that is known only to Alice. In any given run of the protocol, $\mathbf{A}$ and $\mathbf{F}$ are drawn from a joint prior probability $\Pr(\boldsymbol{\alpha}, \mathbf{f})$, which is known to Bob. Thus $H(\mathbf{A}, \mathbf{F}) \leq n_{\mathbf{A}} + n_{\mathbf{F}}$ bits, with equality if and only if the prior is uniform over $\mathbf{F}$ and $\mathbf{A}$. Note that we do not make any assumptions about this prior in what follows.

We have seen before that a single instance of the data Bob receives at the end of Protocol 1 is equal to $(\mathcal{G}, \mathbf{b}', \boldsymbol{\alpha}')$. In a stand-alone setting, this is the only data available to Bob from which he might be able to gain some information about the circuit chosen by Alice. If this protocol were to be used as a subroutine or in parallel with another protocol, then one must analyze the security in a composable framework. Such an analysis is beyond the scope of the present work and is left as an open problem. Note that the graph is considered a parameter of the protocol and not part of Alice's secret. Bob's useful information at the end of a single run of Protocol 1 is then equal to the mutual information $I(\mathbf{B}', \mathbf{A}'; \mathbf{A}, \mathbf{F})$ between the variables associated with the circuit $(\mathbf{A}, \mathbf{F})$ and Bob's data $(\mathbf{B}', \mathbf{A}')$.

In other words, we are modeling the leakage of information as an unintentional classical channel between Alice and Bob, where $(\mathbf{A}, \mathbf{F})$ is the input of the channel and $(\mathbf{B}', \mathbf{A}')$ is

the output at Bob's side. Then, the mutual information tells us how many bits of the original message Bob receives on average, when averaged over many uses of the channel. Importantly, one cannot recover from mutual information what bits of the original message are passed to Bob. For our protocol, the mutual information satisfies the following bound, which does not rely on any computational assumption but it is entirely derived from information theory.

Theorem 2 (Blindness). *In a single instance of Protocol 1 the mutual information between the client's secret input $\{\boldsymbol{\alpha}, \mathbf{f}\}$ and the information received by the server is bounded by*

$$I(\mathbf{B}', \mathbf{A}'; \mathbf{A}, \mathbf{F}) \leq H(\mathbf{A}'). \tag{25}$$

Proof. From the definition of mutual information, we have

$$I(\mathbf{B}', \mathbf{A}'; \mathbf{A}, \mathbf{F}) = H(\mathbf{B}', \mathbf{A}') - H(\mathbf{B}', \mathbf{A}' | \mathbf{A}, \mathbf{F}).$$

Applying the inequality $H(\mathbf{X}, \mathbf{Y}) \leq H(\mathbf{X}) + H(\mathbf{Y})$, together with the fact that $H(\mathbf{B}') \leq n_{\mathbf{B}'} = N$, to the above equation yields

$$I(\mathbf{B}', \mathbf{A}'; \mathbf{A}, \mathbf{F}) \leq H(\mathbf{A}') + N - H(\mathbf{B}', \mathbf{A}' | \mathbf{A}, \mathbf{F}). \tag{26}$$

What remains to be shown is that $H(\mathbf{B}', \mathbf{A}' | \mathbf{A}, \mathbf{F}) \geq N$. This result is proved as Lemma 4 in the Appendix A by bounding $\Pr(\mathbf{b}', \boldsymbol{\alpha}' | \boldsymbol{\alpha}, \mathbf{f}) \leq 2^{-N}$ based on the full joint probability for the protocol. With this bound in place, Eq. (25) directly follows. $\qquad\square$

The conditional entropy $H(\mathbf{A}, \mathbf{F} | \mathbf{B}', \mathbf{A}')$ quantifies the amount of information that, on average, remains unknown to Bob about Alice's computation at the end of Protocol 1. As mentioned previously, in the case where Alice chooses the measurement angles $\mathbf{A}$ uniformly randomly from a finite set, one A_j for each qubit, and she chooses the flow $\mathbf{F}$ uniformly randomly from the set of all flows compatible with the total order implicit in $\mathcal{G}$, then

$$H(\mathbf{A}, \mathbf{F}) = n_{\mathbf{A}} + n_{\mathbf{F}}. \tag{27}$$

In this case by calculating the conditional entropy

$$H(\mathbf{A}, \mathbf{F} | \mathbf{B}', \mathbf{A}') = H(\mathbf{A}, \mathbf{F}) - I(\mathbf{B}', \mathbf{A}'; \mathbf{A}, \mathbf{F}), \tag{28}$$

we have $H(\mathbf{A}, \mathbf{F} | \mathbf{B}', \mathbf{A}') \geq n_{\mathbf{F}}$ because of Theorem 2. Note that Theorem 2 guarantees zero mutual information for a single run of Protocol 1 only if $n_{\mathbf{A}} = 0$, which means only one choice of measurement angle for each qubit. However, the structure of the domain of $\boldsymbol{\alpha}$ and $\boldsymbol{\alpha}'$ [see Eq. (19)] forbids such a choice. A minimal choice of angles that is not classically simulable (via the Gottesman-Knill theorem [50]) is given by

$$\mathcal{A} = \left\{ \frac{\pi}{4}, \frac{3\pi}{4}, \frac{5\pi}{4}, \frac{7\pi}{4} \right\}. \tag{29}$$

In this case, for each angle α_j, one has $n_{\alpha_j} = 2$, so $n_{\mathbf{A}} = 2N$. Since $H(\mathbf{A}') \leq n_{\mathbf{A}'} = n_{\mathbf{A}}$, Bob gains at most two bits of information per qubit measured, with this information being a nontrivial function of both $\boldsymbol{\alpha}$ and $\mathbf{f}$.
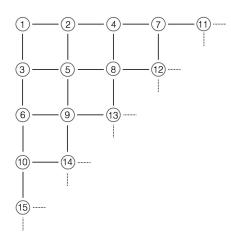

FIG. 1: Total order of the measurements for a generic $n \times m$ cluster state used as a resource state in Protocol 1.

V. APPLICATION TO CLUSTER STATES

To conclude the security analysis of the stand-alone scenario, it is necessary to calculate the exact value of $N_{\mathbf{F}}$, which in turn gives us the value of $n_{\mathbf{F}}$ and hence the lower bound of the conditional entropy for the case of uniform variables $\mathbf{A}, \mathbf{F}$ as explained above. Clearly this depends on the choice of $\mathcal{G}$. Here, we consider the case of cluster states, where $\mathcal{G}$ is taken to be $\mathcal{G}_{n,m}$ with implicit total ordering of vertices as illustrated in Fig. 1. Note that M, the length of the bit string $\mathbf{f}$, is equal to $\log_2 N_{\mathbf{F}}$. When condition (G4) is included, the g-flows we consider correspond to *focused g-flows* [51]. Hence, there is a one-to-one correspondence between an instance of a g-flow $\mathbf{f}$ of $\mathbf{F}$ and a choice of input and output set on the graph [51]. Here, we place a lower bound on M by counting flows that satisfy conditions (G1)–(G4). The use of the additional constraint (G4), which is not implicit in the definition of g-flow, implies that we are undercounting the total number of flows: hence,

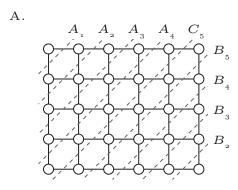$$N_{\mathbf{F}} \geq \#\mathcal{G}(I, O)_{n,m}, \tag{30}$$

where $\#\mathcal{G}(I, O)_{n,m}$ corresponds to the number of possible ways one can define an open graph that satisfies conditions (G1)–(G4). We now show that this quantity can grow exponentially in the dimensions of the cluster state such that $n_{\mathbf{F}} \propto N$.

Theorem 3. *For a cluster state corresponding to $\mathcal{G}_{n,m}$ with fixed total order as depicted in Fig. 1, the number of different open graphs $\mathcal{G}(I, O)$ satisfying conditions (G1)–(G4) is given by*

$$\#\mathcal{G}(I, O)_{n,m} = F_{2\min(n,m)+1}^{|n-m|} \prod_{\mu=2}^{\min(n,m)} F_{2\mu}^2. \tag{31}$$

where F_i is the ith Fibonacci number.

Proof. The proof of this theorem is somewhat involved. We begin by considering a set of diagonal cuts across $\mathcal{G}_{n,m}$, as depicted in Fig. 2(a). As we are considering only those flows that
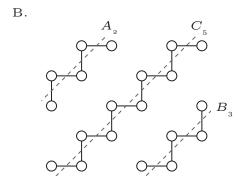
A.

B.

FIG. 2: (a) A cluster-state graph $\mathcal{G}_{n,m}$ with diagonal cuts imposed. The flow across each cut is independent, and the number of possible flows across each cut is indicated. (b) Several cuts with their neighboring vertices isolated.

FIG. 3: $A_\mu^\rightarrow$ and $A_\mu^{\nrightarrow}$ can be constructed recursively, as shown above. Here arrows indicate information flow, while edges indicate the possibility of information flow.

satisfy condition (G4), there is a straightforward constraint on the information flow, which can be seen by isolating a single cut and the vertices linked by edges that the cut passes through [see Fig. 2(b)]. In the following discussion, we consider only the vertices connected by edges through which a particular cut passes. Because of the total ordering imposed on the vertices of $\mathcal{G}_{n,m}$, conditions (G1)–(G3) ensure that information can only pass through a cut from the left side to the right side and not in the reverse direction. Condition (G4) then allows exactly the set of flows where for any vertex k on the right side of the cut, information flows to k from at most one of its neighbors on the left side of the cut. So, if $i, j \in \mathcal{N}(k)$, then $k \notin g(i) \cap g(j)$.

We divide the cuts into three types: (i) those having one less neighboring vertex on the left side of the cut than on the right, (ii) those having one more neighboring vertex on the left side of the cut than on the right, and (iii) those having an equal number of neighboring vertices on both sides of the cut. We label the total number of flows for each type as A_μ, B_μ, and C_μ, respectively, where μ indicates the number of neighboring vertices on the left side of the cut, as shown in Fig 2(b).

In order to quantify these, we begin by noting that $A_\mu = A_\mu^\rightarrow + A_\mu^{\nrightarrow}$, where $A_\mu^\rightarrow$ denotes the number of flows with the restriction that information flows from the uppermost neighboring vertex on the left side of the cut to the uppermost neighboring vertex on the right side of the cut, and $A_\mu^{\nrightarrow}$ denotes the

number of flows where this constraint is not satisfied. These quantities can be calculated using a simple recursion relation, as follows.

Here, $A_\mu^\rightarrow$ allows precisely one possibility for flow between the uppermost vertices of the cut, precluding flow from the uppermost vertex on the left side of the cut to lower vertices on the right side. Hence the remaining $\mu-1$ vertices on the left side and $\mu - 1$ on the right side will be isolated and identical to the situation where the cut partitions one fewer vertex on each side. Thus, $A_\mu^\rightarrow = A_{\mu-1} = A_{\mu-1}^\rightarrow + A_{\mu-1}^{\nrightarrow}$.

Calculating $A_\mu^{\nrightarrow}$ is a little more involved, as there are two possibilities to consider. The first is that no information flows from the uppermost vertex on the left side of the cut across the cut (in which case it is an output). In this case, isolation of the lower vertices occurs as in the analysis of $A_\mu^\rightarrow$; hence, there are $A_{\mu-1}^\rightarrow + A_{\mu-1}^{\nrightarrow}$ possible flows. In the second case, no information can flow into the second uppermost vertex on the right side of the cut; hence, only $A_{\mu-1}^{\nrightarrow}$ flows are possible. Thus, $A_\mu^{\nrightarrow} = A_{\mu-1}^\rightarrow + 2A_{\mu-1}^{\nrightarrow} = A_\mu^\rightarrow + A_{\mu-1}^{\nrightarrow}$. These correspondences are depicted in Fig. 3.

Note that $A_\mu^\rightarrow$ and $A_\mu^{\nrightarrow}$ satisfy the same recursion relation as the Fibonacci sequence when ordered as $(A_1^\rightarrow, A_1^{\nrightarrow}, A_2^\rightarrow, A_2^{\nrightarrow}, \dots)$ and starting with $A_1^\rightarrow = 1 = F_2$ and $A_1^{\nrightarrow} = 2 = F_3$. It follows that $A_\mu = F_{2\mu+2}$. By similar arguments, we have $B_\mu = F_{2\mu}$ and $C_\mu = F_{2\mu+1}$.

It remains only to be noted that the configuration of the information flow across one cut is independent of the information flow across other cuts; hence, the total number of possible flows is given by the product of the possible flows across each cut. Therefore,

$$\#\mathcal{G}(I,O)_{n,m} =$$
$$\left(\prod_{\mu=1}^{\min(n,m)-1} F_{2\mu+2} \right) F_{2\min(n,m)+1}^{|n-m|} \left(\prod_{\nu=2}^{\min(n,m)} F_{2\nu} \right), \quad (32)$$

which simplifies to Eq. (31) as required. $\qquad\square$

The Fibonacci numbers can be written exactly in terms of

the golden ratio $\phi = \frac{1}{2}(1 + \sqrt{5})$ as

$$F_k = \frac{\phi^k - (-\phi)^{-k}}{\sqrt{5}}. \tag{33}$$

For large cluster states ($n, m \gg 1$), the number of possible flows is given by

$$\#\mathcal{G}(I, O)_{n,m} \approx 5^{-\frac{|n-m|}{2}} \phi^{(2\lambda+1)|n-m|} \prod_{\nu=2}^{\lambda} \frac{\phi^{4\nu}}{5} \tag{34}$$

$$= 5^{-\frac{(n+m-2)}{2}} \phi^{2mn+m+n-4}, \tag{35}$$

where $\lambda = \min(n, m)$. The above approximation is obtained by noting that $F_k \approx \frac{\phi^k}{\sqrt{5}}$, since $\left|(-\phi)^{-k}\right| \ll 1$ for large k, and using this to approximate Eq. (31).

Taking $N = nm$ and assuming m grows polynomially in n, then $m = \text{poly}(n)$, and

$$\#\mathcal{G}(I, O)_{n,m} = 2^{2N \log_2 \phi + O(N^\epsilon)} \tag{36}$$

for some $\epsilon < 1$. In such a case, using Eq. (30) and evaluating to leading order,

$$n_{\mathbf{F}} \geq \log_2 \#\mathcal{G}(I, O)_{n,m} \approx 1.388\,N. \tag{37}$$

This result implies that the conditional entropy $H(\mathbf{A}, \mathbf{F} | \mathbf{B}', \mathbf{A}') \geq 1.388\,N$. For the case of a computation chosen uniformly at random by Alice, the total number of bits required to entirely describe her computation is approximately equal to $3.388\,N$. However Bob only receives exactly $2N$ bits of information from Alice (the angles $\boldsymbol{\alpha}'$). From Theorem 2.1 in Ref. [52], it is easy to verify that Bob cannot decode Alice's computation entirely with unit probability. Additionally, Theorem 2.4 in Ref. [53] shows that Bob cannot guess Alice's computation with probability greater than $2^{-1.388N}$.

To make sense of this result, one should remember that a particular *deterministic* MBQC computation is characterized by identifying an input and output set on the underlying graph of the resource state, together with an information flow construction. This structure determines how the quantum information is deterministically transferred via projective measurements from the physical location of the input to the output. Furthermore, once the input and output systems are fixed on the graph, the flow, if it can be constructed, is unique. Hence, in the canonical approach to MBQC, the usual procedure is to fix the input and the output and assign a partial order of measurements that guarantees determinism under a specific set of rules. Consequently, the flow construction imposes a total order of measurements, which must respect the partial one.

Here, we *have reversed* this point of view. As such, Theorem 2 is based on the nontrivial observation that, for a given MBQC resource state with a fixed total order of measurements, choices of g-flow, i.e. choices of input and output vertices on the graph that correspond to different deterministic quantum computations, are generally not unique. Nonetheless, Alice's choice of input and output enforces a unique computation among all the possible choices. This choice of
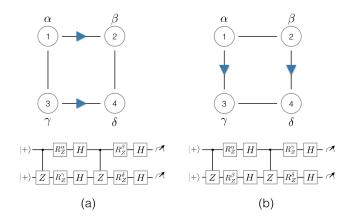

FIG. 4: A 2×2 cluster state with measurement angles $\{\alpha, \beta, \gamma, \delta\}$. In this example we show how to encode two different computations using a fixed total order of measurements $\{1, 2, 3, 4\}$. The difference follows from the choice of the $\mathcal{G}(I, O)$. In diagram (a), the input set is $\{1, 3\}$, and the output set is $\{2, 4\}$, with g-flow function $g(1) = \{2\}$, and $g(3) = \{4\}$. The equivalent circuit associated with the positive branch of this MBQC pattern is shown below. Note that any final round of corrections is pushed into the classical post-processing of the output. In diagram (b), the input set is $\{1, 2\}$, the output set is $\{3, 4\}$, $g(1) = \{3\}$, and $g(2) = \{4\}$. Similarly to (a), we show the circuit of the positive branch of the MBQC pattern, and the final round of corrections is classically post-processed.

g-flow is not communicated to the server and is kept hidden by Alice, who uses it to update the classical instructions sent to Bob. This observation makes it possible for a client to conceal the flow of quantum information from a quantum server classically instructed on what operations to perform. In particular, since a large number of other computations are still compatible with the information Bob receives, the achieved blindness follows from the ambiguity about the flow of information on the graph. Furthermore, our protocol circumvents the scheme-dependent no-go theorem for classical blind quantum computing stated in Ref. [37]. Here, we do not make use of any affine encryption on the client's side, but as mentioned above, we use flow ambiguity to encode a part of the client's computation. As a consequence of this encoding, Protocol 1 requires multiple rounds of communication between the client and server. This requirement is in direct contrast with the assumptions of Ref. [37], where only one round of communication is allowed.

We can additionally make two important observations. The first is that the circuits implementable on $\mathcal{G}_{n,m}$ are not classically simulable unless BPP = BQP. This stems from the fact that the cluster state is universal with only XY-plane measurements, as has recently been proven in Ref. [54]. The above bound provides an exponential lower bound on the number of consistent flows for all cluster states. The second observation is that the computations corresponding to different choices of flow are not equivalent, even when classical post-processing is allowed. This can most easily be seen by considering an example. We consider the simplest case of the 2×2 plaquette $|\text{CS}\rangle_{2,2}$. In Fig. 4 we show an example of two different

choices of open graphs compatible with the underlying total order of measurements. Both choices satisfy the g-flow conditions and as a result they correspond to two deterministic MBQC patterns, i.e. two different and well-defined computations. Since in this particular case the input state ($\hat{C}_Z|++\rangle$) is equivalent, the difference is dictated by the unitary transformation (specified by the measurement angles) that acts on it. As can be seen from Fig. 4, the corresponding circuits are different and perform different unitaries. Because of the flow ambiguity and the obfuscation due to the one-time pad, a quantum server that was to perform the measurements following Protocol 1, at the end of the procedure, would not have enough information to exactly identify Alice's choice of open graph. For $\mathcal{G}_{2,2}$ there are nine possible flow configurations as expected from Eq. (31), which are depicted in Fig. 5. Given any fixed transcript of the protocol, for each flow there exists a choice of $\boldsymbol{\alpha}$ such that it is consistent with the transcript. An example run of Protocol 1 is presented in Fig. 6 for the $\mathcal{G}_{2,2}$ case.

As a final comment, it is clear that there exist cases where, for a fixed graph and choice of angles, different choices of flows will correspond to the same computation. For instance, referring to Fig. 5, measuring all qubits with the same angle would give a two-to-one correspondence for some of the computations. However, it is reasonable to conjecture that when the angles are chosen from sets of large cardinality the mapping will be close to one to one. The full characterization of the mapping is left as an open problem.
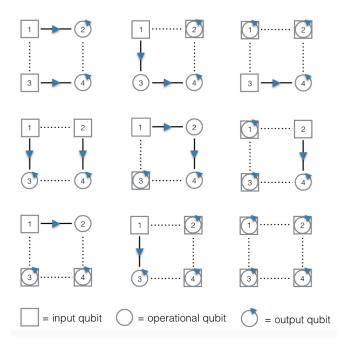

FIG. 5: List of the nine possible $\mathcal{G}(I,O)_{2,2}$ combinations (and associated patterns) with g-flow for the cluster state $|\text{CS}\rangle_{2,2}$. The arrows indicate the direction of the quantum information flow. Note that overlapping input and output sets are allowed. All the patterns implement unitary embeddings on the input state.
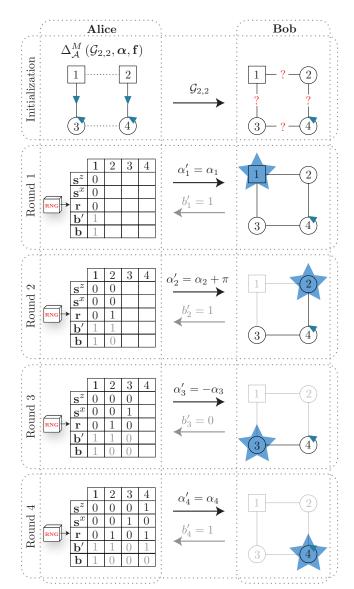

FIG. 6: Illustration of an exemplary run of Protocol 1. At the start of the protocol, Alice's computation is expressed as a measurement pattern on a graph, in this case $\mathcal{G}_{2,2}$. This is communicated to Bob, who prepares the initial state. The computation then proceeds in rounds with Alice computing the relevant entries of $\mathbf{s}^x$ and $\mathbf{s}^z$ and, using these together with r_i, the measurement angle α'_i. The measurement angle α'_i is communicated to Bob, who performs the measurement and returns the result to Alice as b'_i. From this, Alice computes b_i as $b'_i \oplus r_i$. This process is repeated until all qubits have been measured. At the end of the protocol, for any fixed transcript of the communication (composed of $\boldsymbol{\alpha}'$ and $\mathbf{b}'$), it is always possible to find a choice for $\boldsymbol{\alpha}$ consistent with the transcript for any choice of $\mathbf{f}$. In other words, for any of the possible g-flow configurations shown in Fig. 5, Bob can find an $\boldsymbol{\alpha}$ that would have led to the transcript he recorded, which means any of those g-flows is possible. This ambiguity is responsible for partially hiding Alice's computation from Bob.

VI. DISCUSSION AND CONCLUSIONS

Our overall motivation in this work has been to explore the possibility of classically driven blind quantum computation. While this may seem an impossible task, the fact that multiple nonequivalent computations in the MBQC model can yield the same transcript of measurement angles and results, even when the resource state and order of measurements are fixed, allows the tantalizing possibility that it may be possible for a classical user to hide a computation from a quantum server. Protocol 1 makes use of this flow ambiguity to provide some measure of hiding for quantum computations chosen from certain restricted sets. Our intention in introducing this protocol is not to provide a practical cryptographic protocol but rather to demonstrate that it is indeed possible to hide nonequivalent quantum computations using this flow ambiguity. As such, we concentrate on showing that in a single run of the protocol, the amount of information obtained by the server is bounded, rather than introducing a composable security definition, which is nontrivial given the dependence of the leaked information on the responses of the server.

Our results provoke a couple of questions. The first and most obvious is whether the flow ambiguity effect can be exploited to hide a universal set of computations even after the measurement angles have been communicated to the server. A second, and perhaps even more important question is whether this phenomenon can be used as a building block for verification of quantum computers by completely classical users.

[1] P. W. Shor, "Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer," SIAM review **41**, 303 (1999).

[2] S. Lloyd, "Universal quantum simulators," Science **273**, 1073 (1996).

[3] S. Lloyd, M. Mohseni, and P. Rebentrost, "Quantum algorithms for supervised and unsupervised machine learning," arXiv:1307.0411 (2013).

[4] V. Dunjko, J. Fitzsimons, C. Portmann, and R. Renner, *Advances in Cryptology - ASIACRYPT 2014* (Springer Berlin Heidelberg, 2014).

[5] A. Childs, "Secure assisted quantum computation," Quant. Inf. Comp. **5**, 456 (2005).

[6] P. Arrighi and L. Salvail, "Blind quantum computation," Int. J. Quantum Information **4**, 883 (2006).

[7] A. Broadbent, J. Fitzsimons, and E. Kashefi, "Universal blind quantum computation," in *Proceedings of the 50th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pp. 517–526 (2009).

[8] T. Morimae and K. Fujii, "Blind topological measurement-based quantum com- putation," Nat. Comm. **3**, 1036 (2012).

[9] T. Sueki, T. Koshiba, and T. Morimae, "Ancilla-driven universal blind quantum computation," Phys. Rev. A **87**, 060301 (2013).

[10] C.-H. Chien, R. V. Meter, and S.-Y. Kuo, "Fault-Tolerant Operations for Universal Blind Quantum Computation," J. Emerg. Technol. Comput. Syst. **12**, 9:1 (2015).

[11] A. Mantri, C. Perez-Delgado, and J. Fitzsimons, "Optimal blind quantum computation," Phys. Rev. Lett. **111**, 230502 (2013).

[12] V. Giovannetti, L. Maccone, T. Morimae, and T. Rudolph, "Efficient universal blind quantum computation," Phys. Rev. Lett. **111**, 230501 (2013).

[13] C. Perez-Delgado and J. Fitzsimons, "Iterated Gate Teleportation and Blind Quantum Computation," Phys. Rev. Lett. **114**, 220502 (2015).

[14] T. Morimae, "Continuous-variable blind quantum computation," Phys. Rev. Lett. **109**, 230502 (2012).

[15] V. Dunjko, E. Kashefi, and A. Leverrier, "Blind quantum computing with weak coherent pulses," Phys. Rev. Lett. **108**, 200502 (2012).

[16] J. F. Fitzsimons and E. Kashefi, "Unconditionally verifiable blind quantum computation," Physical Review A **96**, 012303 (2017).

[17] M. Hajdušek and C. Perez-Delgado and J. Fitzsimons, "Device-Independent Verifiable Blind Quantum Computation," arXiv:1502.02563v1 (2015).

[18] A. Gheorghiu, E. Kashefi, and P. Wallden, "Robustness and device independence of verifiable blind quantum computing," New J. Phys. **17**, 083040 (2015).

[19] T. Morimae and K. Fujii, "Blind quantum computation protocol in which Alice only makes measurements," Physical Review A **87**, 050301 (2013).

[20] T. Morimae, "Verification for measurement-only blind quantum computing," Phys. Rev. A **89**, 060302 (2014).

[21] M. Hayashi and T. Morimae, "Verifiable measurement-only blind quantum computing with stabilizer testing," Phys. Rev. Lett. **115**, 220502 (2015).

[22] M. Hayashi and M. Hajdusek, "Self-guaranteed measurement-based quantum computation," arXiv preprint arXiv:1603.02195 (2016).

[23] S. Barz, E. Kashefi, A. Broadbent, J. Fitzsimons, A. Zeilinger, and P. Walther, "Demonstration of blind quantum computing," Science **335**, 303 (2012).

[24] S. Barz, J. Fitzsimons, E. Kashefi, and P. Walther, "Experimental verification of quantum computation," Nat. Phys. **9**, 727

[25] C. Greganti, M.-C. Roehsner, S. Barz, T. Morimae, and P. Walther, "Demonstration of measurement-only blind quantum computing," New. J. Phys. **18**, 013020 (2016).

[26] D. Aharonov, M. Ben-Or, and E. Eban, in *Proceedings of Innovations in Computer Science*, (Tsinghua University Press, 2010).

[27] A. Broadbent, "How to verify a quantum computation," arXiv:1509.09180 (2015).

[28] B. Reichardt, F. Unger, and U. Vazirani, "Classical command of quantum systems," Nature **496**, 7446 (2013).

[29] M. McKague, "Interactive Proofs for BQP via Self-Tested Graph States," Theory of Computing **12**, 1 (2016).

[30] J. F. Fitzsimons and M. Hajdušek, "Post hoc verification of quantum computation," arXiv:1512.04375 (2015).

[31] T. Morimae and J. F. Fitzsimons, "Post hoc verification with a single prover," arXiv:1603.06046 (2016).

[32] P. Hauke, F. M. Cucchietti, L. Tagliacozzo, I. Deutsch, and M. Lewenstein, "Can one trust quantum simulators?," Reports on Progress in Physics **75**, 082401 (2012).

[33] M. Cramer *et al.*, "Efficient quantum state tomography," Nat Comms **1**, 149 (2010).

[34] L. Aolita, C. Gogolin, M. Kliesch, and J. Eisert, "Reliable quantum certification of photonic state preparations," Nature Communications **6**, 8498 EP (2015).

[35] D. Hangleiter, M. Kliesch, M. Schwarz, and J. Eisert, "Direct certification of a class of quantum simulations," Quantum Science and Technology **2**, 015004 (2017).

[36] M. Walschaers *et al.*, "Statistical benchmark for BosonSampling," New Journal of Physics **18**, 032001 (2016).

[37] T. Morimae and T. Koshiba, "Impossibility of secure cloud quantum computing for classical client," arXiv:1407.1636 (2014).

[38] S. Aaronson, A. Cojocaru, A. Gheorghiu, and E. Kashefi, "On the implausibility of classical client blind quantum computing," arXiv:1704.08482v1 (2017).

[39] V. Dunjko and E. Kashefi, "Blind quantum computing with two almost identical states," arXiv:1604.01586 (2016).

[40] R. Raussendorf and H. J. Briegel, "A one-way quantum computer," Phys. Rev. Lett. **86**, 5188 (2001).

[41] D. Gottesman and I. L. Chuang, "Demonstrating the viability of universal quantum computation using teleportation and single-qubit operations," Nature (London) **402**, 390 (1999).

[42] X. Zhou, D. W. Leung, and I. L. Chuang, "Methodology for quantum logic gate constructions," Phys. Rev. A **62**, 052316 (2000).

[43] D. Browne, E. Kashefi, M. Mhalla, and S. Perdrix, "Generalized flow and determinism in measurement-based quantum computation," New J. Phys. **9**, 250 (2007).

[44] D. Markham and E. Kashefi, "Entanglement, Flow and Classical Simulatability in Measurement Based Quantum Computation," arXiv:1311.3610 (2013).

[45] M. Nielsen and I. Chuang, *Quantum Computation and Quantum Information* (Cambridge University Press, 2000).

[46] V. Danos and E. Kashefi, "Determinism in the one-way model," Phys. Rev. A **74**, 052310 (2006).

[47] V. Danos, E. Kashefi, and P. Panangaden, "The measurement calculus," Journal of ACM **54**, 8 (2007).

[48] R. Raussendorf, D. E. Browne, and H. J. Briegel, "Measurement-based quantum computation on cluster states," Phys. Rev. A **68**, 022312 (2003).

[49] T. M. Cover and J. A. Thomas, *Elements of information theory* (John Wiley & Sons, 2012).

[50] D. Gottesman, "The Heisenberg representation of quantum computers," arXiv:9807006 (1998).

[51] M. Mhalla, M. Murao, S. Perdrix, M. Someya, and P. S. Turner, "Which graph states are useful for quantum information processing?," in *Conference on Quantum Computation, Communication, and Cryptography*, pp. 174–187 (2011).

[52] A. Ambainis, A. Nayak, A. Ta-Shma, and U. Vazirani, "Dense quantum coding and a lower bound for 1-way quantum automata," Proceedings of the thirty-first annual ACM symposium on Theory of Computing pp. 376–383 (1999).

[53] A. Nayak, "Optimal lower bounds for quantum automata and random access codes," Foundations of Computer Science pp. 369,376 (1999).

[54] A. Mantri, T. F. Demarie, and J. F. Fitzsimons, "Universality of quantum computation with cluster states and (X, Y)-plane measurements," Scientific Reports **7**, 42861 (2017).

[55] Formal definitions of blindness and verifiability can be found in [4].

Appendix A: Full joint probability for the protocol and conditional entropy bound

Lemma 4. $H(\mathbf{B}', \mathbf{A}'|\mathbf{A}, \mathbf{F}) \geq N$ *regardless of Bob's strategy.*

Proof. We construct the full joint probability for all of the variables in Protocol 1 and use it to explicitly derive the desired result. Direct dependencies in the joint probability will be limited by causality and the assumptions that Alice's and Bob's laboratories are secure and free of each others' espionage. These limitations are as follows:

- The flow bits $\mathbf{F}$ and ideal measurement angles $\mathbf{A}$ directly depend on no other variables. They are inputs to the problem chosen by Alice and can be correlated.

- Each π-shift bit R_j in $\mathbf{R}$ is chosen by flipping a fair coin; thus, it directly depends on no other variables.

- Alice assigns A'_j based directly on the current A_j, all flow bits $\mathbf{F}$, the current R_j, and any prior decoded bits $\mathbf{B}_{<j}$.

- Each decoded bit B_j directly depends only on R_j (the π-shift bit) and the bit B'_j received from Bob.

- Each bit B'_j that Bob returns to Alice directly depends only on the information Bob has on hand at the time (specifically, $\mathbf{B}'_{<j}$ and $\mathbf{A}'_{\leq j}$), as well as any (classical or quantum) stochastic strategy he wishes to employ.

With these direct-dependency limitations, we can immediately write down the form of the full joint probability for the entire protocol:

$$\Pr(\mathbf{b}', \boldsymbol{\alpha}', \boldsymbol{\alpha}, \mathbf{f}, \mathbf{b}, \mathbf{r}) = \Pr(\boldsymbol{\alpha}, \mathbf{f}) \prod_{j=1}^{N} \Pr(b'_j | \mathbf{b}_{<j}, \boldsymbol{\alpha}'_{\leq j}) \Pr(\alpha'_j | \alpha_j, \mathbf{f}, \mathbf{b}_{<j}, r_j) \Pr(b_j | b'_j, r_j) \Pr(r_j). \tag{A1}$$

Furthermore, we can explicitly write several of these probabilities:

$$\Pr(b_j | b'_j, r_j) = \delta^{b_j}_{b'_j \oplus r_j}, \tag{A2}$$

$$\Pr(\alpha'_j | \alpha_j, \mathbf{f}, \mathbf{b}_{<j}, r_j) = \delta^{\alpha'_j}_{G_j(\alpha_j, \mathbf{f}, \mathbf{b}_{<j}, r_j)}, \tag{A3}$$

$$\Pr(r_j) = \frac{1}{2}, \tag{A4}$$

with the deterministic function

$$G_j(\alpha_j, \mathbf{f}, \mathbf{b}_{<j}, r_j) := (-1)^{s^x_j(\mathbf{f}, \mathbf{b}_{<j})} \alpha_j + \pi s^z_j(\mathbf{f}, \mathbf{b}_{<j}) + \pi r_j \mod 2\pi \tag{A5}$$

obtained from Eq. (18). These hold for all j. At this point, we have the most general form of the full joint probability consistent with the protocol:

$$\Pr(\mathbf{b}', \boldsymbol{\alpha}', \boldsymbol{\alpha}, \mathbf{f}, \mathbf{b}, \mathbf{r}) = \frac{\Pr(\boldsymbol{\alpha}, \mathbf{f})}{2^N} \delta^{\mathbf{b}}_{\mathbf{b}' \oplus \mathbf{r}} \prod_{j=1}^{N} \Pr(b'_j | \mathbf{b}_{<j}, \boldsymbol{\alpha}'_{\leq j}) \delta^{\alpha'_j}_{G_j(\alpha_j, \mathbf{f}, \mathbf{b}_{<j}, r_j)}, \tag{A6}$$

where we have left Bob's strategy arbitrary but consistent with the direct-dependency restrictions given above. Marginalizing over $\mathbf{B}$ gives

$$\Pr(\mathbf{b}', \boldsymbol{\alpha}', \boldsymbol{\alpha}, \mathbf{f}, \mathbf{r}) = \sum_{\mathbf{b}} \Pr(\mathbf{b}', \boldsymbol{\alpha}', \boldsymbol{\alpha}, \mathbf{f}, \mathbf{b}, \mathbf{r}) \tag{A7}$$

$$= \frac{\Pr(\boldsymbol{\alpha}, \mathbf{f})}{2^N} \prod_{j=1}^{N} \Pr(b'_j | \mathbf{b}'_{<j}, \boldsymbol{\alpha}'_{\leq j}) \delta^{\alpha'_j}_{G_j(\alpha_j, \mathbf{f}, \mathbf{b}'_{<j} \oplus \mathbf{r}_{<j}, r_j)}. \tag{A8}$$

From this joint probability distribution we can compute

$$\Pr(\mathbf{b}', \boldsymbol{\alpha}' | \boldsymbol{\alpha}, \mathbf{f}) = \sum_{\mathbf{r}} \frac{\Pr(\mathbf{b}', \boldsymbol{\alpha}', \boldsymbol{\alpha}, \mathbf{f}, \mathbf{r})}{\Pr(\boldsymbol{\alpha}, \mathbf{f})} \tag{A9}$$

$$= \frac{1}{2^N} \sum_{r_1} \cdots \sum_{r_{N-2}} \sum_{r_{N-1}} \sum_{r_N} \prod_{j=1}^{N} \Pr(b'_j | \mathbf{b}'_{<j}, \boldsymbol{\alpha}'_{\leq j}) \delta^{\alpha'_j}_{G_j(\alpha_j, \mathbf{f}, \mathbf{b}'_{<j} \oplus \mathbf{r}_{<j}, r_j)} \tag{A10}$$

$$= \frac{1}{2^N} \left[\prod_{j=1}^{N} \Pr(b'_j | \mathbf{b}'_{<j}, \boldsymbol{\alpha}'_{\leq j}) \right] \sum_{r_1} \cdots \sum_{r_{N-2}} \sum_{r_{N-1}} \left[\prod_{j=1}^{N-1} \delta^{\alpha'_j}_{G_j(\alpha_j, \mathbf{f}, \mathbf{b}'_{<j} \oplus \mathbf{r}_{<j}, r_j)} \right]$$
$$\times \underbrace{\left(\delta^{\alpha'_N}_{G_j(\alpha_N, \mathbf{f}, \mathbf{b}'_{<N} \oplus \mathbf{r}_{<N}, 0)} + \delta^{\alpha'_N}_{G_j(\alpha_N, \mathbf{f}, \mathbf{b}'_{<N} \oplus \mathbf{r}_{<N}, 1)} \right)}_{\text{at most one term is nonzero}} \tag{A11}$$

$$\leq \frac{1}{2^N} \left[\prod_{j=1}^{N} \Pr(b'_j | \mathbf{b}'_{<j}, \boldsymbol{\alpha}'_{\leq j}) \right] \sum_{r_1} \cdots \sum_{r_{N-2}} \sum_{r_{N-1}} \prod_{j=1}^{N-1} \delta^{\alpha'_j}_{G_j(\alpha_j, \mathbf{f}, \mathbf{b}'_{<j} \oplus \mathbf{r}_{<j}, r_j)} \tag{A12}$$

$$\leq \frac{1}{2^N} \left[\prod_{j=1}^{N} \Pr(b'_j | \mathbf{b}'_{<j}, \boldsymbol{\alpha}'_{\leq j}) \right] \sum_{r_1} \cdots \sum_{r_{N-2}} \prod_{j=1}^{N-2} \delta^{\alpha'_j}_{G_j(\alpha_j, \mathbf{f}, \mathbf{b}'_{<j} \oplus \mathbf{r}_{<j}, r_j)} \tag{A13}$$

$$\vdots$$

$$\leq \frac{1}{2^N} \left[\prod_{j=1}^{N} \Pr(b'_j | \mathbf{b}'_{<j}, \boldsymbol{\alpha}'_{\leq j}) \right] \sum_{r_1} \delta^{\alpha'_1}_{G_1(\alpha_1, \mathbf{f}, r_1)} \tag{A14}$$

$$\leq \frac{1}{2^N} \prod_{j=1}^{N} \Pr(b'_j | \mathbf{b}'_{<j}, \boldsymbol{\alpha}'_{\leq j}) \tag{A15}$$

$$\leq \frac{1}{2^N} \tag{A16}$$

In the above, we have repeatedly used the fact that G_j has at most one r_j that makes it equal to α'_j for any given $(\alpha_j, \mathbf{f}, \mathbf{b}_{<j})$. Therefore, substituting the above bound into the conditional entropy formula gives

$$H(\mathbf{B}', \mathbf{A}' | \mathbf{A}, \mathbf{F}) = \sum_{\boldsymbol{\alpha}, \mathbf{f}} \Pr(\boldsymbol{\alpha}, \mathbf{f}) H(\mathbf{B}', \mathbf{A}' | \mathbf{A} = \boldsymbol{\alpha}, \mathbf{F} = \mathbf{f}) \geq \sum_{\boldsymbol{\alpha}, \mathbf{f}} \Pr(\boldsymbol{\alpha}, \mathbf{f}) N = N, \tag{A17}$$

which was to be proven. □