Equivalence in delegated quantum computing

Fabian Wiesner\textsuperscript{1,2}, Jens Eisert\textsuperscript{2,3}, and Anna Pappa\textsuperscript{1,4}

\textsuperscript{1} Electrical Engineering and Computer Science Department, Technische Universität Berlin, 10587 Berlin, Germany
\textsuperscript{2} Dahlem Center for Complex Quantum Systems, Freie Universität Berlin, 14195 Berlin, Germany
\textsuperscript{3} Fraunhofer Heinrich-Hertz Institute, 10587 Berlin, Germany
\textsuperscript{4} Fraunhofer Institute for Open Communication Systems - FOKUS, 10589 Berlin, Germany

Abstract. Delegated quantum computing (DQC) enables limited clients to perform operations that are outside their capabilities remotely on a quantum server. Protocols for DQC are usually set up in the measurement-based quantum computation framework, as this allows for a natural separation of the different parts of the computation between the client and the server. The existing protocols achieve several desired properties, including the security of inputs, the blindness of computation and its verifiability, and have also recently been extended to the multiparty setting. Two approaches are followed in DQC that demand completely different operations on the clients’ side. In one, the clients are able to prepare quantum states, in the other, the clients are able to measure them. In this work, we provide a novel stringent definition of the equivalence of protocols and show that these distinct DQC settings are, in fact, equivalent in this sense. We use the abstract cryptography framework to prove our claims and provide a novel technique that enables changing from one setting to the other. In this way, we demonstrate that both approaches can be used to perform tasks with the same properties. I.e., using our proposed techniques, we can always translate from one setting to the other. We finally use our results to propose a hybrid-client model for DQC.

Keywords: Delegated quantum computation, quantum cryptography, composability.

1 Introduction

Compared to classical computers, current-state quantum computers are excessively expensive, noisy, prone to errors, and massive in size. Nevertheless, due to the promise of unprecedented computational speed, there is growing interest in performing quantum computations in the scientific community and the industry. For this reason, it is envisioned that genuine quantum access to quantum computers may well first be possible via delegated quantum computing (DQC), where clients with limited quantum capabilities will delegate computational tasks to powerful quantum servers.
In such cases, it is likely that the clients might want to keep their inputs, outputs or the computation they delegate hidden from the server. Protocols that achieve these properties commonly make use of the measurement-based quantum computation (MBQC) framework \cite{Bri+09}. In MBQC, the server and the clients can appropriately split up the different parts of the computation, ensuring that the clients perform only minimal quantum operations, while at the same time guaranteeing security. The quantum operations that the clients are able to perform define the model of DQC that will be used. There are two prevalent DQC models, one where the clients can prepare quantum states, which they then send to the server (\textit{prepare-and-send}), and another where the clients receive quantum states from the server and measure them (\textit{receive-and-measure}). These two approaches have until now been studied separately, providing protocols that achieve the desired properties in an ad hoc way.

1.1 Related work

The first proposal of a DQC protocol that achieves security for the input and output, and also keeps the computation secret (\textit{blindness}), has been introduced by Childs in 2005 \cite{Chi05}. The complex requirements on the client side have been drastically reduced in the seminal work of Broadbent et al. \cite{BFK09}. In the proposed protocol, the client prepares simple quantum states and sends them to the server, which subsequently entangles them. We define the communication setting of Ref. \cite{BFK09} as \textit{prepare-and-send} (Fig. 1). Using the abstract cryptography framework (AC) introduced by Maurer and Renner \cite{MR11}, Dunjko et al. \cite{Dun+14} have shown that the protocol in Ref. \cite{BFK09} is in fact not only secure in a stand-alone fashion, but it also maintains its security properties under composition, giving rise to the stringent notion of \textit{composable security}. The proof relies on a reduction to non-signalling via an equivalence of protocols. In 2017, Kashefi and Pappa \cite{KP17} proposed the first multiparty delegated quantum computation protocol in the \textit{prepare-and-send} setting, and proved that it maintains the blindness and secrecy of inputs under composition, against a malicious server or against other malicious clients. A different approach towards a multiparty protocol has been made by Houshmand et al. \cite{Hou+18}. Their protocol is based on single-client DQC and has been proven to be secure against a dishonest server, where the assumption that the global computation is made from local ones, has been made.

In contrast to the above family of schemes, a distinctly different setting for delegated quantum computation has independently been proposed and studied. In this setting, the server prepares a large entangled resource state and then sends the qubits one by one to the client, who measures them sequentially in appropriate bases that encode the desired computation (Fig. 2). This \textit{receive-and-measure} setting (also called \textit{measurement-only}) was first introduced by Morimae and Fujii in 2013 \cite{MF13} and is by definition blind, since the client never sends anything to the server. This protocol has also been shown to be \textit{composably secure} \cite{MK13}.

\footnote{This type of protocols is also known as \textit{universal blind quantum computing} (UBQC).}
In **prepare-and-send**, a client prepares qubits $|P_{r_i}\rangle$ sampled from arbitrary bases $\{P_i\}_{i=1}^n$ and sends them to the server. The server performs a unitary on the qubits and on an additional state $\rho$. Afterwards, it measures the qubit registers in arbitrary bases $\{M_i\}_{i=1}^n$. The measurement result the $i$-th for this register is denoted with $s_i$.

In **receive-and-measure**, the server prepares qubits $|M_{s_i}\rangle$ sampled from arbitrary bases $\{M_i\}_{i=1}^n$ and performs a unitary on the qubits and on an additional state $\rho$. Afterwards, it sends the qubit registers to a client who measures them in arbitrary bases $\{P_i\}_{i=1}^n$. The measurement result for the $i$-th register is denoted with $r_i$. 

Fig. 1. In **prepare-and-send**, a client prepares qubits $|P_{r_i}\rangle$ sampled from arbitrary bases $\{P_i\}_{i=1}^n$ and sends them to the server. The server performs a unitary on the qubits and on an additional state $\rho$. Afterwards, it measures the qubit registers in arbitrary bases $\{M_i\}_{i=1}^n$. The measurement result the $i$-th for this register is denoted with $s_i$.

Fig. 2. In **receive-and-measure**, the server prepares qubits $|M_{s_i}\rangle$ sampled from arbitrary bases $\{M_i\}_{i=1}^n$ and performs a unitary on the qubits and on an additional state $\rho$. Afterwards, it sends the qubit registers to a client who measures them in arbitrary bases $\{P_i\}_{i=1}^n$. The measurement result for the $i$-th register is denoted with $r_i$. 
A notion that has actually been studied early on in both models is that of verifiability. This refers to notions of how clients can verify that the server is performing the desired computation and giving back the correct outcome. In the prepare-and-send setting with a single client, initial efforts by Fitzsimons and Kashefi [FK17] have been followed up by others [KW17]. These have been complemented by Kapourniotis et al. [Kap+21], who have presented a protocol that allows multiple clients to perform a computation on a server and to verify that everyone acted as supposed, as long as at least one client is honest. For that, they assumed a classical secure multiparty computation (SMPC) construction that remains secure when at least one party is honest. In the receive-and-measure setting as well, several protocols achieved verifiability [Mor14; HM15]. An insightful review by Fitzsimons [Fit17] offers a short introduction to the different protocols on blind and verifiable quantum computing, a discussion on the case of multiple quantum servers, as well as on the experimental state-of-affairs in DQC.

1.2 Our contribution

In this work, we take a holistic approach to DQC and aim at providing a unifying framework for previously proposed protocols. Specifically, we examine whether the two settings where DQC is usually set, i.e. prepare-and-send and receive-and-measure, are distinct or whether there is a direct reduction from one to the other. We define what it means for protocols to be equivalent in the AC framework, show a transformation between the settings and prove that this is an equivalence transformation. We therefore manage to connect the year-long studies on DQC in prepare-and-send and receive-and-measure settings by establishing that they are in fact equivalent. We further apply the proposed transformation to the recent prepare-and-send protocol by Kapourniotis et al. [Kap+21] and obtain a new protocol that performs similarly in the receive-and-measure setting. In this way, we demonstrate how to straightforwardly construct a new protocol in one setting, given a protocol in the other, without the need to reprove its cryptographic properties. Since our proposed transformation is applied on elementary subroutines, it can find a broad application on any protocol containing at least one subroutine that fits into one of the two settings. We finally propose a new hybrid protocol that enables clients with different capabilities to jointly delegate a computation to a quantum server, and in this way, we make DQC protocols significantly more flexible.

1.3 Structure

This work is structured as follows. In Section 2, we provide a short introduction to quantum computing, with a focus on measurement-based quantum computation. In Section 3, we describe the abstract cryptography (AC) framework by Maurer and Renner [MR11] and also define new relations for protocols in AC. In Section 4, we formally define the two prevalent routines in prepare-and-send and receive-and-measure and present and prove the equivalence transformation between them.
In Section 5, we apply the transformation on the protocol by Kapourniotis et al. [Kap+21] and find an equivalent protocol in the receive-and-measure setting. Finally, in Section 6, we use the fact that the transformation is applied to individual clients, to propose a novel hybrid protocol based on [Kap+21], which involves clients from both settings.

2 Quantum computing

To keep this article self-contained, in this section, we briefly explain some basic concepts of quantum computing in general (for more details, see, e.g., Ref. [NC10]) and measurement-based quantum computation (for more details, see, e.g., Ref. [Joz05]). Readers familiar with the basics of quantum mechanics and computing can obviously skip this section.

2.1 Basic notions of quantum computing

With the term quantum bit or qubit, we denote the smallest quantum mechanical object that we can manipulate. When in a pure state, a qubit can be written as a linear combination of chosen basis states of a complex vector space. If we consider for example the computational basis vectors \( |0\rangle \) and \( |1\rangle \), a qubit state vector \( |\psi\rangle \) can be written as

\[
|\psi\rangle = \alpha|0\rangle + \beta|1\rangle,
\]

where \( \alpha, \beta \in \mathbb{C} \). We use a specific notation for the states that are ‘equidistant’ from \( |0\rangle \) and \( |1\rangle \); we define \( |+\rangle = (|0\rangle + |1\rangle)/\sqrt{2} \) and \( |-\rangle = (|0\rangle - |1\rangle)/\sqrt{2} \).

We can also consider composite systems of more than single qubits. For example, the state vector of a two-qubit system can be expressed as

\[
|\psi\rangle = \sum_{i,j \in \{0,1\}} \alpha_{i,j} |i,j\rangle,
\]

where \( \sum_{i,j \in \{0,1\}} |\alpha_{i,j}|^2 = 1 \).

In quantum states with more than one qubit, it is possible to observe one of the fundamental non-classical properties of quantum mechanics, that of entanglement. Entanglement is present in a state vector \( |\psi\rangle \) when it cannot be expressed by a tensor product. Probably the most well-known examples of entangled states are the Bell pairs (studied by Bell [Bel64] but also earlier in the EPR paradox [EPR35]). These four vectors

\[
|\Phi^{0,0}\rangle = \frac{1}{\sqrt{2}}(|0,0\rangle + |1,1\rangle), \quad |\Phi^{0,1}\rangle = \frac{1}{\sqrt{2}}(|0,0\rangle - |1,1\rangle),
\]

\[
|\Phi^{1,0}\rangle = \frac{1}{\sqrt{2}}(|0,1\rangle + |1,0\rangle), \quad |\Phi^{1,1}\rangle = \frac{1}{\sqrt{2}}(|0,1\rangle - |1,0\rangle)
\]
constitute a basis of the four-dimensional Hilbert space. These state vectors actually result from applying an entangling operation on two unentangled qubits. A commonly-used entangling gate is the control-\(Z\) gate (or in short CZ), written in matrix form as

\[
CZ = \begin{bmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & -1
\end{bmatrix},
\]

which when applied to \(|+,+\rangle\) gives the \(|\Phi^0,0\rangle\) state vector. Some qubit operators that we will encounter later are the Pauli operators; together with the identity \(1\), they form the basis of the vector space of 2\(\times\)2 Hermitian matrices

\[
X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \quad Y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}, \quad Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}.
\]

Two other operators that are often used for DQC are the \(P\)-gate and the Hadamard \(H\)-gate

\[
P(\theta) = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\theta} \end{bmatrix}, \quad H = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}.
\]

In general, it holds that in a quantum computer all quantum circuits, reflecting operations excluding measurements and state preparations, can be represented by unitary matrices, or short, unitaries. We denote the set of all unitaries in a \(d\)-dimensional complex vector space \(\mathcal{H}\) by \(U(d)\) or \(U(\mathcal{H})\). For these matrices it holds that

\[
U \in U(d) : UU^\dagger = 1_d,
\]

where \(1_d\) is the identity operator in \(\mathcal{H}\), and \(U^\dagger\) the conjugate transpose of \(U\).

Before we explain unitary evolution of states and measurements, we need a more general notion of states and a notion of the conjugate transpose of vectors. For vectors, we write \(\langle \psi | \psi \rangle = (\alpha |0\rangle + \beta |1\rangle)^\dagger = \langle \psi | = \alpha^* \langle 0 | + \beta^* \langle 1 |\), where \(\langle 0 | = \begin{bmatrix} 1 & 0 \end{bmatrix}\), \(\langle 1 | = \begin{bmatrix} 0 & 1 \end{bmatrix}\).

Using this notation, we can write the inner product \(\langle \psi | \phi \rangle\) as \(\langle \psi | \phi \rangle\) and use for the outer product the notation \(|\phi \rangle \langle \psi |\). With this notion, we introduce mixed states. These states are represented by density matrices, which are characterised by

\[
\rho \geq 0, \quad \rho = \rho^\dagger, \quad \text{Tr}(\rho) = 1.
\]

The representation of a state vector \(|\psi\rangle\) as density matrix is nothing else but the outer product \(|\psi\rangle \langle \psi |\). A quantum state evolves in time in a unitary fashion when undergoing Schrödinger dynamics, which in a quantum computer is represented with the application of gates. Hence, when we apply the gate \(U\) on the state vector \(|\psi\rangle\), we get \(U|\psi\rangle\), and if we apply \(U\) on the state \(\rho\), we get \(U\rho U^\dagger\).

\[\footnote{Up to a \(H\) gate applied on one of the qubits.}\]
We finish this introduction to the very basics of quantum computing with measurements. A measurement is denoted by a set of measurement operators \( \{ M_j \} \), where we associate \( j \) with the outcome. For these operators it holds
\[
\sum_j M_j^\dagger M_j = \mathbb{I}_d, \quad M_j \in \mathbb{C}^{d \times d},
\]
where the states to measure are density matrices in \( \mathbb{C}^{d \times d} \) and the states after measurement are represented by density matrices in \( \mathbb{C}^{d' \times d'} \). When we measure a state \( \rho \) using these measurement operators, we get the result \( j \) with the probability
\[
\Pr[j] = \text{Tr}(M_j \rho M_j^\dagger).
\]
If \( d' > 1 \), the remaining quantum state after this measurement result is given by
\[
\rho'_j = \frac{M_j \rho M_j^\dagger}{\text{Tr}(M_j \rho M_j^\dagger)}.
\]

We will also often use the term ‘measurement basis’. With that, we denote a measurement where the operators have the form \( \langle j | x \otimes \mathbb{I}_y \rangle \), where we measure the qubit-system \( x \) in the basis \( \{ | j \rangle \} \) and do nothing with the rest \( y \).

We finally note that the operations that we consider are many times applied on parts of entangled states. Slightly abusing notation, we will for simplicity denote with Greek letters, e.g., \( \rho, \eta \), both the registers containing the physical systems of the different parties and the systems themselves where the operations are applied.

### 2.2 Measurement-based quantum computing

The DQC protocols that we study in this work are formulated in the measurement-based quantum computing (MBQC) model. However, very little knowledge is in fact necessary in order to follow the rest of this work. Generally, the quantum computation within MBQC is done by (adaptively) measuring an entangled resource state in certain bases. It has been shown that it is enough to use bases from the set
\[
\mathcal{A} = \{ \{ | \pm e^{\pi/4} \rangle = |0\rangle \pm e^{\pi/4} |1\rangle : 0 \leq k < 8 \} \}.
\]

A resource state is commonly made from many \( |+\rangle \)-state vectors that are entangled using \( \text{CZ} \)-gates. In most cases, there are two requirements for the resource state, or rather the family of resource states:

1. The (family of) resource states is computationally universal.
2. There is an efficient classically-computable flow function \( f \) that determines how measurement bases have to be adapted, depending on previous measurement results.
The first requirement implies that every quantum computation can be performed using a specific measurement pattern on the resource state. The second requirement dictates that the measurement instructions on the resource state need to be efficiently updated in order to perform a computation. This is because of the randomness that incurs when measuring parts of the state due to the entanglement present. Finally, if there is quantum input for the computation, that input needs to be entangled with the resource state using CZ gates; in contrast, if the output is quantum, part of the resource state remains unmeasured at the very end of the protocol. The family of cluster states naturally satisfied those conditions [Bri+09], but there are also other families of universal resources for measurement-based quantum computing [GE07].

3 Abstract cryptography

3.1 The different components

In 2011, Maurer and Renner proposed the abstract cryptography (AC) framework in order to facilitate proofs of composable security for cryptographic primitives [MR11]. This work aimed to complement and extend various existing approaches, including the universal composability framework [Can00]. Composable security is a particularly stringent notion of security: It allows performing modular security proofs and extending the validity of these proofs to composed setups. Hence, if one proves the composable security of the different parts of a protocol, the whole protocol is also composable-secure. Thus, one can directly replace a black box in a composable-secure protocol with a composable-secure implementation of the black box and obtain a secure composite implementation. In contrast, one cannot rely on the security of a composite protocol, when the components are proven to be ‘just’ stand-alone secure.

AC follows an ‘ideal-world real-world’ approach. What one aims to do in AC is to show that the implemented protocol is indeed performing the correct task, and does not leak anything more than desired. To prove this, one first has to define what is the ideal resource \( F \) that the protocol is trying to attain, albeit very abstractly. The next step is to show that the proposed protocol is close to \( F \) by some pseudo-metric. In what follows, we give the necessary definitions regarding AC to be able to later prove the composable security of our protocols. Based on Ref. [MR11], we specifically define what a resource⁷, a converter, a filter and a distinguisher are.

Definition 3.1 (Resource). A resource is a system with interfaces for communication with other resources, converters or the environment. We label the interfaces with the elements of an interface-set \( I \). A resource is closed in the sense that there is no way to intervene in the operation of the resource except to use the interfaces one legitimately has access to.

⁷ Note that the word ‘resource’ in AC is used differently than in the previous section, where computational resources are discussed.
We will use calligraphic capital letters like $\mathcal{F}$ for resources. A common resource that is used in real-world implementations of functionalities is a communication channel used by the protocol. A protocol itself is modelled in AC as a converter, i.e., as an algorithm that makes use of a resource to implement another resource. Apart from the protocols, the other common converters we will be using are simulators. We will denote converters with small Greek letters like $\pi$ for protocols and $\sigma$ or $\gamma$ for simulators.

**Definition 3.2 (Converter).** A converter is a system with two interfaces. The inner interface is attached to the resource that should be converted; the outer interface provides the new interface of this composition.

To account for different adversarial behaviours in the protocol, we also need to consider filters that connect to the resources and allow for advanced malicious activity to take place.

**Definition 3.3 (Filter).** An interface can be filtered. This means that there are two different sets of rules about what can happen at this interface, one for honest behaviour (filter active) and the other for dishonest behaviour (filter removed). In our notation, we write $\mathcal{R}_\phi = (\mathcal{R}, \phi)$ when we consider the pair of the filter $\phi$ and a resource $\mathcal{R}$.

Finally, in order to quantify security, one has to be able to measure the distance between an ideal resource and its real-world implementation. This is done in AC with distinguishers.

**Definition 3.4 (Distinguisher).** A distinguisher for two resources $\mathcal{R}$ and $\mathcal{S}$ with $n$ interfaces each, is a resource with $n+1$ interfaces. The distinguisher interacts with $\mathcal{R}$ or $\mathcal{S}$ using $n$ interfaces, i.e., connects via all of $\mathcal{R}$’s or $\mathcal{S}$’s interfaces with the respective resource and outputs either 0 or 1 at the remaining interface.

In most cases, one is interested in either information-theoretical security, i.e. security against unbounded adversaries, or – as in computational security – the security against polynomially bounded adversaries. The limitations of the adversaries determine the set of distinguishers one considers.

**Definition 3.5 (Distinguishing pseudo-metric [MR11][Dun+14]).** Let $D$ be a set of distinguishers. Then we define the distance between two resources $\mathcal{R}$ and $\mathcal{S}$ with respect to $D$ as

$$d(\mathcal{R}, \mathcal{S}) = \max_{D \in D}(|\Pr[D(\mathcal{R}) = 1] - \Pr[D(\mathcal{S}) = 1]|).$$

For the distinguishing pseudo-metric, it holds that

$$d(\mathcal{R}, \mathcal{R}) = 0,$$
$$d(\mathcal{R}, \mathcal{S}) = d(\mathcal{S}, \mathcal{R}),$$
$$d(\mathcal{R}, \mathcal{S}) + d(\mathcal{S}, \mathcal{A}) \geq d(\mathcal{R}, \mathcal{A}).$$
The missing property compared to a full metric, therefore, is that $d(R, S)$ may also vanish for $R$ and $S$ different from each other. Another important property of $d(\cdot, \cdot)$ is

$$\forall \sigma : d(R, S) \geq d(\sigma R, \sigma S). \quad (3.1)$$

The last inequality implies that one cannot increase the distinguishing advantage for two resources by concatenating them with a converter. If this were possible, there would be a better distinguisher for $R$ and $S$ that uses this converter in the first place, which is by definition not possible.

### 3.2 Definitions of protocol relations

We now give some necessary definitions for relations between resources. The first one, secure construction, has already been defined in Ref. [Dun+14]. The other two, translatability and equivalence are notions that are newly introduced here. The introduction of these is necessary for understanding the potential of the various DQC protocols in the different communication models and the relations between them. We note here that all relations of resources defined below depend implicitly on the choice of the distinguisher set. The subtle differences between them are easier to understand using the example of coin flipping presented in the supporting material A.1.

**Definition 3.6 (Secure construction [Dun+14])**. Let $R_\phi = (R, \phi)$ and $S_\psi = (S, \psi)$ be pairs of a resource with interface-set $I$ and a filter. A protocol $\pi = \{\pi_i\}_{i \in I}$ securely constructs $S_\psi$ out of $R_\phi$ within $\epsilon$, conditioned on honest $H \subseteq I$, if there exist simulators $\sigma = \{\sigma_i\}_{i \in I \setminus H}$ such that

$$\forall P \subseteq I, H \subseteq P : d(\pi_P \phi_R, \sigma_{I \setminus P} \psi_P S) \leq \epsilon.$$

In this case, we write $R_\phi \xrightarrow{\pi, H, \epsilon} S_\psi$.

Simulators are specific converters used in the ideal world, i.e. are applied here on $S$, to account for potential malicious activities in the real world, corresponding here to the resource $R$. If we can find simulators such that the two world are indistinguishable, no attack in the real world can achieve more than what is allowed in the ideal world. Or equivalently, if there were an attack that could achieve more, there would be a distinguisher that could use this attack as a subroutine to distinguish between the two worlds.

The definition of secure construction is useful to show relations between an ideal resource and its real-world implementation, but it provides no information about the relation between two real-world implementations. For that, we define translatability.

**Definition 3.7 (Translatability)**. Let $R_\phi = (R, \phi)$ and $Q_\chi = (Q, \chi)$ be pairs of a resource with interface-set $I$ and a filter. $\pi^1 = \{\pi^1_i\}_{i \in I}$ applied to $R_\phi$ is translatable
into $\pi^2 = \{\pi^2_i\}_{i \in I}$ applied to $Q_\chi$ within $\epsilon$, conditioned on honest $H \subseteq I$, if there exist simulators $\sigma = \{\sigma_i\}_{i \in I \setminus H}$ such that

$$\forall P \subseteq I, H \subseteq P: d(\pi^1_P \phi_P R, \sigma_T \setminus P, \pi^2_P \chi_P Q) \leq \epsilon.$$

In this case we write $\{\pi^1 R_\phi\} \overset{\epsilon}{\rightarrow}_H \{\pi^2 Q_\chi\}$. Further, we define translatability within $\epsilon = 0$ as 'perfect translatability'.

Note that the Defs. 3.6 and 3.7 are different, even though at first sight they look similar. A very important distinction is that in Def. 3.7, the right argument of $d(\cdot, \cdot)$ has converters (protocols) $\pi^2_P$ attached that allow us to compare two real-world implementations.

Further, one finds:

$$\{\pi^1 R_\phi\} \overset{\epsilon}{\rightarrow}_H \{\pi^2 Q_\chi\} \Rightarrow \forall P \subseteq I, H \subseteq P: d(\gamma_{I \setminus P} \pi^1_P \phi_P R, \gamma_{I \setminus P} \sigma_{I \setminus P} \pi^2_P \chi_P Q) \leq \epsilon_s,$$

where we have the freedom to choose $\gamma$ such that

$$\forall P \subseteq I, H_i \cup H_s \subseteq P: d(\pi^1_P \phi_P R, \gamma_{I \setminus P} \pi^2_P \chi_P Q) \leq \epsilon_i.$$

Such simulators $\gamma_i$ exist from the translatable relation $\{\pi^1 R_\phi\} \overset{\epsilon}{\rightarrow}_H \{\pi^2 Q_\chi\}$. Using now the triangle inequality which holds for this pseudo-metric, one gets

$$\forall P \subseteq I, H_i \cup H_s \subseteq P: d(\pi^1_P \phi_P R, \gamma_{I \setminus P} \sigma_{I \setminus P} \psi_P S) \leq \epsilon_s + \epsilon_i \Rightarrow R_\phi \overset{\pi^1 \epsilon_i + \epsilon_s}{\rightarrow}_{H_s \cup H_t} S_\psi.$$
Exploiting the definition of secure construction, we now introduce a new notion of equivalence for protocols, which we will later use to show that different DQC protocols achieve the same level of security.

**Definition 3.8 (Equivalence).** We define two protocols $\pi^1$ and $\pi^2$ applied to their filtered resources $R_\phi$ and $Q_\chi$ conditioned on honest $H$ to be equivalent if and only if the secure construction of any filtered ideal resource $S_\psi$ within $\epsilon$, conditioned on honest $H$, by one protocol implies the secure construction within $\epsilon$, conditioned on honest $H$, by the other protocol, i.e., if

$$\forall S_\psi : R_\phi \overset{\pi^1,\epsilon}{\longrightarrow}_H S_\psi \iff Q_\chi \overset{\pi^2,\epsilon}{\longrightarrow}_H S_\psi.$$ 

In this case we write $\{\pi^1 R_\phi\} \overset{H}{\iff} \{\pi^2 Q_\chi\}$.

**Theorem 3.2.** Two protocols $\pi^1$ and $\pi^2$ are equivalent if and only if one perfectly translates to the other, i.e.:

$$\left(\{\pi^1 R_\phi\} \overset{0}{\longrightarrow}_H \{\pi^2 Q_\chi\} \land \{\pi^2 Q_\chi\} \overset{0}{\longrightarrow}_H \{\pi^1 R_\phi\}\right) \iff \left(\{\pi^1 R_\phi\} \overset{H}{\iff} \{\pi^2 Q_\chi\}\right).$$

**Proof.** One direction of the proof follows directly from Theorem 3.1:

$$\forall S_\psi : R_\phi \overset{\pi^1,\epsilon}{\longrightarrow}_H \{\pi^2 Q_\chi\} \land Q_\chi \overset{\pi^2,\epsilon}{\longrightarrow}_H S_\psi \Rightarrow R_\phi \overset{\pi^1,\epsilon}{\longrightarrow}_H S_\psi$$

$$\{\pi^2 Q_\chi\} \overset{0}{\longrightarrow}_H \{\pi^1 R_\phi\} \land R_\phi \overset{\pi^1,\epsilon}{\longrightarrow}_H S_\psi \Rightarrow Q_\chi \overset{\pi^2,\epsilon}{\longrightarrow}_H S_\psi.$$ 

The other direction is more involved and is presented in the supporting material A.2. □

### 4 Equivalence of DQC protocols

We are now ready to further discuss the different proposals for delegated quantum computing and how they precisely relate to each other. We will specifically focus on protocols in the *prepare-and-send* and the *receive-and-measure* settings. Protocols of the first flavour assume that the client can prepare arbitrary single qubits [BFK09; KW17; Kap+21], which they then send to the server. Upon reception, the server entangles them using appropriate pre-fixed operations and waits for instructions from the client regarding which qubits to measure and on what basis. The server then sends the measurement outcomes back to the client and awaits further instructions. The client can ensure that their input and measurement bases remain secret by applying one-time pads on both the quantum and classical communicated information.

For our constructions, nothing else is required. Indeed, we do not need to go into details regarding the type of measurements, preparation bases or resource states.
used. Similarly, we work with a very general model of receive-and-measure, which is a model where the server prepares a large resource state which is then sent to the client for measurement \[\text{MF13}\]. Again, the generality of our constructions allows us to be vague about the specific measurement and preparation processes, and therefore our results apply to all possible protocols that achieve DQC in the receive-and-measure model.

The main result of this section is a generic transformation that, when applied to a protocol in one model, yields an equivalent protocol in the other. This constitutes a proof of equivalence. To find this result, we (abstractly) define the two routines (prepare-and-send and receive-and-measure) and show an equivalence transformation between them. To achieve this, we define a subset of distinguishers, which we call Pauli-blind-distinguisher-set (PBDS) and use it to prove two theorems. The first one (Section 4.1) concerns the translatability of protocols with respect to the PBDS and more general distinguishers. The second theorem (Section 4.2) states that prepare-and-send and receive-and-measure are equivalent with respect to the PBDS.

**Definition 4.1 (Pauli-blind-distinguisher-set).** Let \(S\) be a resource with interfaces in \(\mathcal{I}\) and \(\mathcal{I}_Q\) the subset of \(\mathcal{I}\) that outputs a qubit. Further, let \(\gamma\) be a set of converters those map a qubit \(p\) to \(\sigma_P \rho \sigma_P\), with \(\sigma_P\) being a Pauli randomly drawn according to any distribution \(\chi\). We define the PBDS to be the subset of unbounded distinguishers, such that

\[
\forall D \in \text{PBDS} : |D[\gamma_{I_Q}S] - D[S]| = 0.
\]

Hence, the PBDS is the set of unbounded distinguishers that can not distinguish between resources which only differ by a Pauli operation.

The motivation to use the PBDS lies in the proof of translatability between prepare-and-send and receive-and-measure. While we found simple PBDS-simulators that lead to both-sided translatability (Theorems 4.2, 4.3), it seems to be much more challenging to find simulators that show translatability for all unbounded distinguishers. Hence, we use Theorem 4.1 combined with equivalence with respect to the PBDS (Corollary 4.1) in order to prove the equivalence for arbitrary distinguishers (Theorem 4.4, Section 4.3).

### 4.1 Translatability with respect to the PBDS

In this section, we show how, under certain conditions, we can generalise our results in the unbounded-distinguisher setting. We do this by lifting our results from Pauli-blind to arbitrary unbounded distinguishers. Similar to Theorem 3.1, Theorem 4.1 shows that under certain conditions, translatability with respect to the PBDS suffices to get secure constructions with respect to arbitrary distinguishers. To be able to formulate Theorem 4.1, we need to define a type of adapted converters as follows.

**Definition 4.2 (Heralding-adaption).** A converter \(\sigma\) has a heralding-adaption \(\tilde{\sigma}\) for a message set \(M = \{m_i\}\) if and only if one finds \(\sigma = \beta^M \circ \tilde{\sigma}\), where \(\beta^M\) has the following properties:
– $\beta^M$ gets messages at the inner interface from $\tilde{\sigma}$. All messages not in $M$ are forwarded, while messages in $M$ are ignored.

– $\beta^M$ forwards all messages received at the outer interface to $\tilde{\sigma}$.

One can see that for any converter, there exist many heralding-adaptions. In most relevant cases, $M$ is a set of messages that contain information about the inner processes of $\tilde{\sigma}$. Equipped with this definition, we can now lift the secure construction that follows from translatability, from the PBDS-setting to the general unbounded-distinguisher-setting.

**Theorem 4.1 (General unbounded-distinguisher-setting).** Let us assume we have perfect translatability with respect to the PBDS between two protocols $\pi_1$ and $\pi_2$, i.e. $\{\pi_1 R_\phi\} \xrightarrow{H} \{\pi_2 Q_\chi\}$, and also the following properties.

1. There are heralding-adaptions for the simulators that were used in the translatability relation; these adapted simulators output messages about the additional Pauli operations that were applied.

2. The probability that the identity operator was applied instead of any other Pauli-operator is always larger than 0 at every simulated interface.

Under these assumptions, we can show with respect to arbitrary distinguishers that

$$\forall S_\psi : Q_\chi \xrightarrow{H} S_\psi \Rightarrow R_\phi \xrightarrow{H} S_\psi$$

**Proof.** Let us assume

$$\exists S_\psi : \forall \sigma \exists P \subseteq I, H \subseteq P : \epsilon_D' = d(\pi_1 R_\phi \sigma, \pi_2 Q_\chi) > \epsilon.$$ We denote with $D'$ the distinguisher that performs with $\epsilon_D'$ accuracy. We can find a distinguisher $D$ that distinguishes between the simulation and the real world if one used $\pi_2$ as follows.

1. Apply the heralding-adaption of the simulators $\tilde{\gamma}_i$ that were used in the translatability relation between $\{\pi_1 R_\phi\}$ and $\{\pi_2 Q_\chi\}$.

2. Forward any communication to $D'$ except for the Pauli messages from the simulators.

3. If all Pauli messages correspond to the identity (denoted with $1$), output the result from $D'$, else rerun.

If all Pauli messages correspond to the identity operator $1$, we find translatability with respect to arbitrary distinguisher. This is because the simulators only induce $1$ in this case but no other Pauli operation. Further, $D$ only accepts the result from $D'$ if all Pauli messages correspond to the identity. The probability for that event
is close to 1 in the asymptotic limit. Hence, one finds
\[
|\Pr[D(\pi_2^P \chi P Q) = 1] - \Pr[D(\sigma_2^P \psi_p S) = 1]| = |\Pr[D'(\gamma_2 \chi P \chi P) = 1] - \Pr[D'(\gamma_1 \chi P \sigma_2^P \psi_p S) = 1]| = |\Pr[D'(\pi_1^P \phi_1 P \phi_1 P) = 1] - \Pr[D'(\sigma_1^P \psi_p S) = 1]| = \epsilon_D > \epsilon,
\]
where we have set \( \sigma_i^1 = \gamma_i \circ \sigma_i^2 \), and used the fact that there exists perfect translatability between the protocols applied to their respective resources in the second and third line.

\[\square\]

### 4.2 Equivalence of prepare-and-send and receive-and-measure with respect to the PBDS

In this section, we show the equivalence of the prepare-and-send and receive-and-measure settings with respect to the PBDS. We start with defining two routines for prepare-and-send and receive-and-measure (Defs. 4.3 and 4.4), which are the main building blocks of all proposals of secure delegated quantum computation to date. Following that, we prove perfect translatability in both directions (Theorems 4.2, 4.3) and conclude that these routines are equivalent with respect to the PBDS (Corollary 4.1).

The index notation \( a | b \) is used for the object being the \( a \)th in a set or sum and acting on system \( b \). Let also \( \mathcal{P} \) and \( \mathcal{M} \) denote choices of bases for the two-dimensional Hilbert space \( \mathcal{H}_q \), \( \rho_x \) denotes a density matrix residing in an arbitrary finite-dimensional Hilbert space \( \mathcal{H}_x \) and \( \Omega_{q,x} \) denotes a unitary in \( U(\mathcal{H}_q \otimes \mathcal{H}_x) \).

**Definition 4.3 (Prepare-and-send).** We define the prepare-and-send routine \( \pi_{ps} \) performed by a Server \( S \) and a client \( C \) as follows.

1. \( C \) inputs \( \mathcal{P} \); \( S \) inputs \( \Omega_{q,x}, \mathcal{M}, \rho_x \).
2. \( C \) prepares a pure state \( |P_r|_q \rangle \) drawn uniformly at random from a basis \( \mathcal{P} \); \( r \) denotes the choice.
3. \( S \) applies \( \Omega_{q,x} \cdot \rho_{x} = \Omega_{q,x}(|P_r|_q \langle P_r|_q \rangle \otimes \rho_x) \Omega_{q,x}^\dagger \).
4. \( S \) measures the qubit register labelled with \( q \) in the basis \( \{ |M_0 \rangle, |M_1 \rangle \} \), where \( s \) is the measurement result. The state of register \( x \) is now \( \rho'_x \).
5. \( C \) outputs \( r \) and \( S \) outputs \( s \) and \( \rho'_x \).

We will denote with \( \mathcal{K} \) and \( |\mathcal{K}_k\rangle \) the complex conjugates of a basis \( \mathcal{K} \) and a state vector \( |k_k\rangle \) respectively, i.e.,
\[
\overline{\mathcal{K}} = \{ |\overline{\mathcal{K}}_k\rangle \} = \{ \mathcal{K}_k^\dagger |0\rangle + \mathcal{K}_k^\dagger |1\rangle \}.
\]

We also denote with \( \Omega_{q,x}^T = \sum_{a,b=0}^1 |b_q\rangle \langle a_q| \otimes \Theta_{a,b|q} \), where \( \Theta_{a,b|q} \) acts in \( \mathcal{H}_x \), the partial transpose of \( \Omega_{q,x} = \sum_{a,b=0}^1 |a_q\rangle \langle b_q| \otimes \Theta_{a,b|q} \), i.e. the transpose of the \( q \)-part in the compound vector space.
Definition 4.4 (Receive-and-measure).

We define the receive-and-measure routine $\pi_{rm}$ performed by a Server $S$ and a client $C$ as follows.

1. $C$ inputs $\mathcal{P}$; $S$ inputs $\Omega_{q,x}, \mathcal{M}, \rho_x$.
2. $S$ prepares a pure state $|\mathcal{M}_{s|q}\rangle$ drawn uniformly at random from a basis $\mathcal{M}$; $s$ denotes the choice.
3. $S$ applies $\Omega_{q,x}^T: \rho_{q,x} = \Omega_{q,x}^T (|\mathcal{M}_{s|q}\rangle \langle \mathcal{M}_{s|q}| \otimes \rho_x) (\Omega_{q,x}^T)^\dagger$ and sends the qubit register to $C$.
4. $C$ measures the qubit register labelled with $q$ in the basis $\{|\mathcal{P}_0\rangle, |\mathcal{P}_1\rangle\}$, where $r$ is the measurement result. The state of register $x$ is now $\rho_{x}'$.
5. $C$ outputs $r$ and $S$ outputs $s$ and $\rho_{x}'$.

The routines are depicted schematically in Fig. 3. Having defined the routines formally, we are now ready to prove the following translatability theorems.

**Theorem 4.2.** The prepare-and-send routine $\pi_{ps}$ (Def. 4.3) is perfectly translatable with respect to the PBDS into the receive-and-measure routine $\pi_{rm}$ when both are applied on a quantum communication channel $Q$, i.e.:

$$\{\pi_{ps}^Q\}^{0,\mathcal{P}} \rightarrow \{\pi_{rm}^Q\}$$

**Proof.** We here give a sketch of the proof, which can be found in full detail in the supporting material [A.3]. We start with the case of honest participants. At the end of the prepare-and-send routine, $S$ outputs (up to normalisation) the state $\rho_{x}' = A_{r,s}\rho_x A_{r,s}^\dagger$ where

$$A_{r,s} = (|\mathcal{M}_{s|q}\rangle \otimes 1_x) \Omega_{q,x}(|\mathcal{P}_r\rangle \otimes 1_x) = \sum_{b,c=0}^{1} \Theta_{b,c|x} \mathcal{M}_{s,b}^* \mathcal{P}_{r,c}.$$
and $\Omega_{q,x} = \sum_{b,c=0}^{1} |b_q\rangle\langle c_q| \otimes \Theta_{b,c|x}$.

For the same input, the output of $S$ at the end of the receive-and-measure routine is (up to normalisation) $\tilde{\rho}'_x = B_{r,s} \rho_x B_r^\dagger$, where

$$B_{r,s} = (\langle P_r | \otimes 1_x) \Omega_{q,x}^T (|M_{s,q}| \otimes 1_x) = \sum_{b,c=0}^{1} \Theta_{b,c|x} M_{s,b}^* P_{r,c} = A_{r,s}.$$ 

This means that $\rho'_x = \tilde{\rho}'_x$ and we immediately have

$$d(\pi^{ps}_C Q^{ps}_S, \pi^{rm}_C Q^{rm}_S) = 0.$$ 

Fig. 4. Simulators for the translatability between the prepare-and-send and receive-and-measure routines. $\sigma^{B, prep}$ outputs half of $|\Phi_{i,o}^{0,0}\rangle$ at each interface. $\sigma^{B, meas}$ receives at each interface a qubit and performs a measurement in the Bell-basis.

If one or both parties are dishonest, we need simulators. For the remainder of the proof we use the simulators depicted in Fig. 4. Further, we note that in favour of readability, we sometimes omit normalisation since the proofs are not affected in any way.

The two simulators that we propose make use of the properties of Bell states, specifically that all four states can be generated from any of them using Pauli operators:

$$\left\{ \left| \Phi_{a,b}^{0,0}\right\rangle \right\}_{B_0,B_1=0}^{1} = \left\{ \left(1_a \otimes X_b B_1 Z_b B_0 \right) \left|0_a,0_b\right\rangle + \left|1_a,1_b\right\rangle \right\}_{B_0,B_1=0}^{1} \sqrt{2}.$$ 

In case of a dishonest server, we use the simulator $\sigma^{B, prep}$ for the server in receive-and-measure. Using

$$\left(1 \otimes \langle P_r | \otimes 1\right) \left|\Phi_{i,o}^{0,0}\right\rangle \otimes 1 = \frac{1}{\sqrt{2}} (|P_r\rangle \otimes 1). \quad (4.1)$$

we find

$$d(\pi^{ps}_C Q^{ps}_S, \pi^{rm}_C Q^{B, prep}_S) = 0. \quad (4.2)$$
Hence, we find indistinguishably using $\sigma^{B,\text{prep}}$ is the server is dishonest.

In the next case, the client is dishonest but the server is honest. Here we use $\sigma^{B,\text{meas}}$ for the client in receive-and-measure and find

\[
\langle\Phi_B^0, B^1 | \otimes \mathbb{1} \rangle (\mathbb{1} \otimes T_\psi) (\mathbb{1} \otimes |\mathcal{M}_s\rangle \otimes \mathbb{1}) = \frac{1}{\sqrt{2}} (\mathcal{M}_s |\Omega X^{B_1} Z^{B_0}, \quad (4.3)
\]

which implies

\[
d(\mathcal{Q}\pi^{ps}_S, \sigma^{B,\text{meas}}_C \pi^{rm}_S, \sigma^{B,\text{prep}}_S) = 0. \quad (4.4)
\]

We finally assume that both parties are dishonest. For this case we use the property

\[
(1 \otimes \langle\Phi_{B_0, B_1} | (|\Phi_{B_0}^0, 0\rangle \otimes \mathbb{1}) = \frac{1}{2} Z^{B_0} X^{B_1}, \quad (4.5)
\]

of the Bell states. One can already see in Fig. 4, that the concatenation of $\sigma^{B,\text{prep}}$ and $\sigma^{B,\text{meas}}$ is exactly described by Eq. (4.5) and this gives

\[
d(\mathcal{Q}, \sigma^{B,\text{meas}}_C \mathcal{Q} \sigma^{B,\text{prep}}_S) = 0. \quad (4.6)
\]

\[\Box \quad \Box\]

**Theorem 4.3.** The receive-and-measure routine $\pi^{rm}_C$ (Def. 4.4) is perfectly translatable with respect to the PBDS into the prepare-and-send routine $\pi^{ps}_S$ when both are applied on a quantum communication channel $\mathcal{Q}$, i.e.:

\[
\{\pi^{rm}_C \mathcal{Q}\} \xrightarrow{0,\mathcal{Q}} \{\pi^{ps}_S \mathcal{Q}\}
\]

**Proof.** (Sketch, the detailed proof can be found in supplementary material [A.3].)

First note, that if the server and the client are honest we can use the same calculation as in Theorem 4.2. The same holds if both are dishonest.

The remaining cases are those where only one of the parties is dishonest. If the server is dishonest we almost have the same situation as in the case of a dishonest client in Theorem 4.2, and find

\[
\langle\Phi_{B_0, B_1} | \otimes \mathbb{1} \rangle (\mathbb{1} \otimes |\mathcal{P}_r\rangle \otimes \mathbb{1}) = \frac{1}{\sqrt{2}} (\mathcal{P}_r |\Omega X^{B_1} Z^{B_0}, \quad (4.7)
\]

which implies

\[
d(\mathcal{Q}\pi^{rm}_S, \sigma^{B,\text{meas}}_C \mathcal{Q} \sigma^{B,\text{prep}}_S) = 0. \quad (4.8)
\]

At last we assume a dishonest client, which is similar to a dishonest server in Theorem 4.2. Using

\[
(1 \otimes (\mathcal{M}_s |\otimes \mathbb{1} \rangle (\mathbb{1} \otimes \Omega) (|\Phi_{0, 0}^0 \rangle \otimes \mathbb{1}) = \frac{1}{\sqrt{2}} T_\psi (|\mathcal{M}_r\rangle \otimes \mathbb{1}). \quad (4.9)
\]

we find

\[
d(\mathcal{Q}\pi^{rm}_S, \sigma^{B,\text{prep}}_C \mathcal{Q} \pi^{ps}_S) = 0. \quad (4.10)
\]

\[\Box\]
Corollary 4.1. The prepare-and-send routine $\pi^{ps}$ (Def. 4.3) and the receive-and-measure routine $\pi^{rm}$ (Def. 4.4) are equivalent with respect to the PBDS, i.e.:

$$\{\pi^{ps} Q\} \iff \{\pi^{rm} Q\}.$$ 

Proof. The proof follows directly using Theorems 3.2, 4.2 and 4.3.

\[ \square \]

4.3 General equivalence of prepare-and-send and receive-and-measure

Combining Theorem 4.1 and Corollary 4.1 we can now prove the general equivalence of prepare-and-send and receive-and-measure routines.

Theorem 4.4. The prepare-and-send routine $\pi^{ps}$ and the receive-and-measure routine $\pi^{rm}$ are equivalent with respect to arbitrary distinguishers, i.e.,

$$\{\pi^{ps} Q\} \iff \{\pi^{rm} Q\}.$$ 

Proof. We prove the claim using the following observations, which allow us to deploy Theorem 4.1:

1. There is a heralding-adaption $\tilde{\sigma}^{B,\text{meas}}$ for $\sigma^{B,\text{meas}}$, which outputs the tuple $(B_0, B_1)$ at the outer interface.
2. $\sigma^{B,\text{prep}}$ does not induce a Pauli operation different from $I$.
3. If either $S$ or $C$ are dishonest and $\sigma^{B,\text{meas}}$ is used to simulate the dishonest party, there exists no distinguisher such that $Pr[(B_0, B_1) = (0, 0)] = 0$.
4. In case of both parties being dishonest, one finds $Pr[(B_0, B_1) = (0, 0)] = \frac{1}{4}$.

Except for the third statement, these observations are easy to see. The reasoning behind the third statement is the following: To get $Pr[(B_0, B_1) = (0, 0)] = 0$ the distinguisher has to input at the interface of the dishonest party a qubit $|\psi_D\rangle$ such that $|\psi_D\rangle \otimes |\psi_A\rangle = \alpha|0, 1\rangle + \beta|1, 0\rangle$, where $|\psi_A\rangle$ is the state vector prepared by the honest party. The only way a distinguisher can achieve that with probability 1 is by waiting for the information about $|\psi_A\rangle$ and preparing a suitable state. In the real world, that means the distinguisher never halts since it can not send the correct state without receiving the output of the honest party. However, the honest party also waits for input from the distinguisher before outputting the measurement result.

\[ \square \]

We complete the methodological part of our paper with a definition of the equivalence transformation for protocols that use prepare-and-send and receive-and-measure routines.
Definition 4.5 (T-Transformation). Let $\pi^1$ be a protocol that uses the prepare-and-send routine at a step $t$. By applying the following exchanges we replace the prepare-and-send routine with a receive-and-measure routine:

1. $\Omega_{q,x} \leftrightarrow \Omega^T_{j,x}$,
2. $\mathcal{P} \leftrightarrow \mathcal{M}$,
3. $\mathcal{M} \leftrightarrow \mathcal{P}$.

We define the resulting protocol $\pi^2$ to be the result of the transformation of $\pi^1$, i.e., $\pi^2 = T(\pi^1,t)$.

$T$ is an equivalence transformation since the two protocols differ only regarding the two routines, and by Theorem 4.4 these are equivalent. Furthermore, one can see that $T(T(\pi,t),t) = \pi$, i.e., $T$ is self-inverse.

Most DQC protocols use either exclusively prepare-and-send routines [BFK09; FKP17; Kap+21] or receive-and-measure routines [MF13; HEB04; HM15]. Of course, one can apply $T$ to a 'pure' prepare-and-send protocol to get a receive-and-measure one and vice versa, but one can also apply the transformation only to some routines. This freedom allows for hybrid protocols like the one we present later in Section 6.

5 Using the equivalence transformation

In this section, we apply our newly-defined equivalence transformation (Def. 4.5), first on a single-client DQC protocol [BFK09] and then on a multi-client one [Kap+21].

5.1 Blind delegated quantum computation (BDQC)

The protocol by Broadbent et al. [BFK09] achieves perfect blindness, i.e., constructs an ideal BDQC resource with $\epsilon = 0$. In 2014, Dunjko et al. [Dun+14] showed that the properties of the protocol hold under composition. Starting from the prepare-and-send protocol of [BFK09], they use intermediate protocols to transform it into a no-signalling protocol and deploy the AC framework to show that these sequential transformations preserve the protocol’s properties. On the other hand, there exist receive-and-measure protocols that also achieve perfect blindness [MF13]. Here, we show how to use our equivalence transformation (Def. 4.5) to connect the two approaches to BDQC. In this way, we provide a straightforward way that ensures that proven properties still hold when passing from one setting to the other. We start by briefly describing the two protocols for BDQC ([BFK09] and [MF13]) and then prove that they are in fact equivalent.
BDQC in prepare-and-send \cite{BFK09}. Let us assume that the client \( C \) has \( m \) bits or qubits as input and wants to delegate the computation of a function on that input to the server \( S \). The main idea of the protocol in \cite{BFK09} is that \( C \) encrypts both quantum and classical information sent to \( S \) in such a way that it cancels out and gives the correct outcome in the end. More specifically, \( C \) one-time pads their input and sends it to \( S \) together with auxiliary qubits prepared with offsets \( \theta_{i,j} \), i.e., in \(|+\theta_{i,j}\rangle\). \( S \) then entangles them to build up the brickwork state. \( C \) instructs \( S \) on to measure the state node by node in \(|\pm \delta_{i,j}\rangle\) with
\[
\delta_{i,j} = \phi_{i,j} + \theta_{i,j} + r_{i,j} \pi.
\] (5.1)
Since the qubit rotations applied by \( C \) commute with the entangling operations of \( S \), the offsets in the preparation and the measurement cancel out, and \( S \) effectively measures in \(|\pm \phi_{i,j}\rangle\). The protocol is noted as Protocol 5 in the supporting material.

BDQC in receive-and-measure \cite{MF13}. In the protocol proposed by Morimae and Fujii \cite{MF13}, the client does not have the ability to create quantum states, only to measure them. Hence, the server prepares qubits in \(|+\rangle\), entangles them and sends them to the client for measurement. In the original paper, the client was only thought to have classical input, and therefore no offsets were necessary, as there was no information sent to the server. If we want to account for quantum input that might not be known to the client (could, e.g., come from another process), we need to allow for the client to encrypt their input and send it to the server, using, for example, the same encryption as in \cite{BFK09}. In what follows, we consider such an enhanced version of the \cite{MF13} that also includes quantum input (Protocol 6 in the supporting material).

Equivalence of the BDQC protocols. We will now see how one can straightforwardly retrieve the \cite{MF13} protocol from the protocol by Broadbent et al. \cite{BFK09} by applying the equivalence-transformation of Def. 4.5. We only need to apply the transformation to qubits that do not belong to the input or output in \cite{BFK09}, since the two protocols differ only in what concerns these cases. Hence, after applying the transformation, the server now creates the qubits in \(|+\delta_{x,y}\rangle\), entangles them and sends them to the client. The client now measures the qubits in \(|\pm \theta_{x,y}\rangle\). The multi-qubit gate \( \Omega_{q,x} \) does not change during the transformation, since the entangling gates \( CZ \) used for the preparation of the resource state are diagonal for each qubit in the computational basis. We can assume that the server prepares the output qubits; since there is no measurement on these qubits and they do not contain any secret information from the client, we do not need to consider an equivalence transformation for these. The resulting protocol \( \tau \) is equivalent to the protocol in \cite{BFK09}, however instantiated in the receive-and-measure setting.

It now suffices to show that \( \tau \) and Protocol 6 are equivalent. As \( P(\pm \delta) \) commutes with \( CZ \) it does not matter, if the server or a simulator applies \( P(\pm \delta) \). Since the client always communicates \( \delta \) to the server, the simulators can induce or cancel offsets. Hence, \( \tau \) and Protocol 6 are in both directions perfectly translatable and therefore, equivalent.
5.2 Delegated multiparty quantum computation (DMPQC)

In [Kap+21], the authors have proposed a delegated quantum computation protocol for multiple clients that is blind and verifiable secure against a coalition of up to \( N - 1 \) corrupted clients and the server. To achieve this, a classical secure multiparty computation (SMPC), which is also composable-secure against \( N - 1 \) malicious clients, is used as a resource. The basic idea is to use the double-blind state preparation (DBSP) protocol [KP17] and the H/I-gadget [Kap+21] in order to allow for multiple clients to interact with the server in the same way as a single client would, using the single-party verifiable blind delegated quantum computing protocol (VBDQC) [KW17]. The verifiability at intermediate steps allows aborting before an attack can succeed. To better understand the DMPQC protocol [Kap+21], it is helpful to get into the details of the VBDQC protocol by Kashefi and Wallden [KW17]; we refer the reader to the supporting material A.5 for a presentation of the prepare-and-send version as well as an equivalent protocol in the receive-and-measure.

In order to find an equivalent DMPQC protocol in the receive-and-measure setting, we first apply the transformation \( T \) on the DBSP protocol [KP17], that is used to prepare/rotate states in the \( XY \)-plane with an angle only known if all parties cooperate. After that, we apply \( T \) on the H/I-gadget, which allows for secretly turning states in the \( XY \)-plane into \( Z \)-eigenstates. These two parts suffice for using the single-client protocol of [KW17] in the multiparty setting. Finally, we present the protocol in a way that is agnostic to the communication setting and can therefore be instantiated both in prepare-and-send and in receive-and-measure.

Double-blind state preparation. The purpose of the DBSP is a collective preparation/modification of the state by all clients and the server in such a way that only all parties together can learn the state. Hence, one honest client is enough in order to keep the state secret. The protocol used in [Kap+21] is a prepare-and-send protocol taken from [KP17]. The circuit in the latter implements the unitary

\[
U = P \left( \sum_{i=1}^{k} (-1)^{s_i} \theta_i \right)
\]

on the input. In Fig. 5a, we slightly modify this by allowing the clients also to prepare \( | - \theta_i \rangle \), which leads to an implementation of unitary

\[
U = P \left( \sum_{i=1}^{k} (-1)^{s_i} \theta_i + \pi \bigoplus_{j=1}^{k} r_i \right),
\]

where \( r_i = 1 \) if client \( i \) prepared \( | - \theta_i \rangle \). This modification does not have a real effect, however it allows to directly apply the equivalence transformation.

---

8. We note that without any further assumptions (e.g., secure setup), such a resource cannot be implemented [CKL03]. An implementation would imply composable-secure two-party computation, which implies composable-secure bit commitment, which, as has been shown in Ref. [CF01], is not possible.
Fig. 5. The two circuits used for DBSP. a) The circuit in prepare-and-send and b) the circuit in receive-and-measure. The two circuits are the same up to the $T$ transformation and they both implement $P_{\theta}(\rho) = P(\theta)\rho P(\theta)^\dagger$ with $\theta = \sum_{i=1}^{k} (-1)^{s_i}\theta_i + \pi \oplus r_i$. In prepare-and-send, $r_i$ marks the choice of the basis state in the preparation while in receive-and-measure, this is denoted by $s_i$.

Protocol 1: DBSP of input state $\rho$ provided by party $i$.

**Require:** $i, \rho$

1.1: If in prepare-and-send: All clients except $i$ sample an angle $\theta_c \leftarrow R_A$ and $r_c \leftarrow R\{0,1\}$ for the preparation.

1.2: If in receive-and-measure: All clients except $i$ sample an angle $\theta_c \leftarrow R_A$ for the measurement; The server samples $s_c$ for the preparation.

1.3: Depending on the setting, all parties run the relevant circuit in Fig. 5.

1.4: The measurement results, preparation choices and the angles $\theta_c$ are sent to the SMPC.

1.5: **return** $P(\theta)(\rho)$ to the server and $\theta = \sum_{i=1}^{k} (-1)^{s_i}\theta_i + \pi \oplus r_i$ to the SMPC.

If we apply the equivalence-transformation $T$ on the target-wires of circuit 5a, the resulting circuit is the one shown in Fig. 5b. The implemented unitary is still the same. However, now we can not choose the $r_i$’s anymore, since $r_i$ is the measurement result.

**H/I-gadget.** In order to use the verification mechanism of the dotted triple-graph, one needs to be able to turn the quantum states of some nodes into $Z$-eigenstates to perform the break-operation on the graph. It is not sufficient to let anyone prepare $Z$-eigenstates in the first place, as this party would be able to deduce the positions of the trap qubits from where the dummy qubits are. For this reason, a subroutine has been introduced in Ref. [Kap+21] that is called the **H/I-gadget** and allows performing either $H$ or $I$ in a double-blind fashion (since the parties can only learn which one was applied if they all collaborate).
Fig. 6. Circuits for the $H/I$-gadget. The upper circuit is for the prepare-and-send, the lower one for the receive-and-measure version. $\rho'$ is determined by $P$ and $\rho$ up to a Pauli operation depending on $s$ and $r$. On the left side $r$ determines which of the basis states is used as auxiliary state, as $s$ does on the right side. $\eta$ labels the lower register at this position.

Protocol 2: $H/I$-gadget in prepare-and-send $[\text{Kap+21}]$

Require: $b, \rho$

2.1: if $b=1$ then
2.2: SMPC samples $\gamma \leftarrow \{0, \pi\}$, $r$ marks the choice.
2.3: else
2.4: SMPC samples $\gamma \leftarrow \{\pi/2, 3\pi/2\}$, $r$ marks the choice.
2.5: end if
2.6: All parties run $\text{DBSP}(|+\rangle, S)$ which yields the output $|\psi\rangle = |\theta\rangle$, $\theta$, where $S$ is the server (Protocol [1]).
2.7: The SMPC calculates $\delta = \gamma - \theta$ and instructs the server to perform $|\psi\rangle = P(\delta)|\psi\rangle$
2.8: The server performs their part of the circuit in Fig. 6a and sends $s$ to SMPC.
2.9: return $\rho'$ to $S$.
2.10: return $r,s$ to the SMPC.

In $[\text{Kap+21}]$ it has been shown that the circuit in Fig. 6a, which is in the prepare-and-send setting, implements (up to a Pauli correction) $H$ on $\rho$ if $P$ is the $Y$-basis and $\mathds{1}$ if $P$ is the $X$-basis. We can achieve the same functionality in receive-and-measure using the equivalence-transformation $T$. The result is shown in Fig. 6b. We note that here $\Omega_{q,x} = CX_{q,x}X_q CZ_{q,x}X_q P(\pi/2)H_q P(\pi/2)q$.

Note that protocols 2 and 3 implementing the $H/I$-gadget are, in fact, equivalent. This is because, in order to retrieve the receive-and-measure protocol, we simply applied the equivalence-transformation $T$ on the prepare-and-send one, and instead of the DBSP routine, we implemented a double-blind measurement.
Protocol 3: H/I-gadget in receive-and-measure

Require: $b, \rho$

1. if $b=1$ then
2. SMPC: $\gamma \leftarrow \{0, \pi\}$, $r'$ marks the choice.
3. else
4. SMPC: $\gamma \leftarrow \{\frac{\pi}{2}, \frac{3\pi}{2}\}$, $r'$ marks the choice.
5. end if
6. The server performs their part of the circuit in Fig. 6b.
7. All parties run Protocol 1: $(\eta', \theta) \leftarrow \text{DBSP}(\eta, S)$.
8. The SMPC calculates $\delta = \gamma - \theta$ and instructs the server to apply $P(\delta)$:
   $\eta'' = P(\delta)\eta'$.
9. The server follows the instructions and measures $\eta''$ in $X$, getting the result $\tilde{r}$.
10. return $\rho'$ to $S$.
11. return $(r = r' \oplus \tilde{r}, s)$ to the SMPC.

Combination of the parts. As we now have all the building blocks in both settings, we can create a general DMPQC protocol which consists of three sub-protocols. In the first sub-protocol (4.1), the parties build the dotted triple-graph. In the second sub-protocol (4.2), the SMPC and the server run the VBQC protocol from [KW17]. In sub-protocol 4.3, the SMPC lets the clients measure the last traps and gives the decryption keys to the clients if everything is correct.

We use the following notation for the sub-protocols:

- $V$ is the set of all nodes in the dotted triple-graph.
- $C$ is the set of the nodes that are used for the actual computation, $D$ is the set of dummy nodes, and $T$ is the set of trap nodes.
- $C(\cdot)$ returns the computation node corresponding to a trap or a dummy node; $D(\cdot)$ and $T(\cdot)$ work similarly, returning dummy and trap nodes respectively.
- $I \subseteq C$ and $O \subseteq C$ are the sets of input and output nodes, respectively, of the dotted triple-graph.

We assume, that for each input $v$, every client has two additional $|+\rangle$-state vectors which will later serve as $T(v)$ and $D(v)$. Note that there is a small difference between our protocol and the one proposed in [Kap+21]. In the latter, the input for the DBSP for a node in $C \setminus (I \cup T(I) \cup D(I))$ is prepared by client $N$. For our equivalence transformations to work, it makes more sense (especially for receive-and-measure protocols) to assume that the server performs this task. This difference, however, does not affect security since neither the clients nor the server are guaranteed to be honest.
Sub-Protocol 4.1: Setup of graph state

4.1: All clients with inputs tell the SMPC the positions of their inputs in the dotted triple-graph.
4.2: \textbf{for} \( v \in V \) \textbf{do}
4.3: If \( v \in I \), the owner \( i \) encrypts their input \( \rho_v' = P(\theta_v)X^{a_v}(\rho) \), with \( a_v \in \{0,1\} \) and \( \theta_v \in A \) random, and sends \( a_v, \theta_v \) to the SMPC.
4.4: If \( v \in T(I) \cup D(I) \), the owner sends one of their corresponding \(|+\rangle \) state vectors: \( \rho_v'' = |+\rangle \).
4.5: In all other cases the server prepares \( \rho_v'' = |+\rangle \).
4.6: All run Protocol 1: \( \rho_v, \theta \leftarrow \text{DBSP}(S, \rho_v'') \).
4.7: If \( v \in D \) the SMPC sets \( b = 1 \) and \( b = 0 \) otherwise.
4.8: All run the H/I-gadget: \( \rho_v, s \leftarrow \text{H/I-gadget}(b, \rho_v') \).
4.9: The SMPC notes all classical values for the corrections of the measurement angles and to know the state.
4.10: \textbf{end for}
4.11: The server applies \( CZ \) according to the topology of the dotted triple-graph.

Sub-Protocol 4.2: Measurement of the state

4.1: \textbf{for} \( v \in V \setminus \{O \cup T(O) \cup D(O)\} \) \textbf{do}
4.2: SMPC samples \( r_v \leftarrow R \{0,1\} \).
4.3: \textbf{if} \( v \in D \) \textbf{then}
4.4: SMPC samples \( \delta_v \leftarrow R \mathcal{A} \) and sends \( \delta_v \) to the server.
4.5: The server measures register \( v \) in \(|\pm \delta_v\rangle \) and sends the result \( s_v \) to the SMPC.
4.6: \textbf{else if} \( v \in T \) \textbf{then}
4.7: SMPC sets \( \delta_v = \theta_v + r\pi \) and sends \( \delta_v \) to the server.
4.8: The server measures register \( v \) in \(|\pm \delta_v\rangle \) and sends the result \( s_v \) to the SMPC.
4.9: \textbf{else}
4.10: SMPC sets \( \delta_v = \theta_v + \phi_v + r\pi \) and sends \( \delta_v \) to the server.
4.11: The server measures register \( v \) in \(|\pm \delta_v\rangle \) and sends the result \( s_v \) to the SMPC.
4.12: \textbf{end if}
4.13: \textbf{end for}

Sub-Protocol 4.3: Check and output

4.1: \textbf{for} \( v \in O \) \textbf{do}
4.2: The server sends \( v, T(v) \) and \( D(v) \) to the corresponding client.
4.3: The SMPC instructs the client to measure \( T(v) \) in \( \delta_{T(v)} = \theta_{T(v)} + r_{T(v)} \pi \) with \( r_{T(v)} \leftarrow R \{0,1\} \). The clients sends the result \( s_{T(v)} \) to the SMPC.
4.4: \textbf{end for}
4.5: If for all \( v \in T: r_v = s_v \), i.e. all traps were measured correctly, the SMPC tells the clients the remaining Pauli-corrections for the output and which of the two remaining qubits is the dummy.
By putting these three sub-protocols together, we now obtain a general DMPQC protocol that can readily be instantiated in both the prepare-and-send and receive-and-measure setting by simply calling the two equivalent versions of the DBSP and H/I-gadget. The security of the protocol is guaranteed by the equivalence transformation between the two settings and the security proof of the prepare-and-send protocol in Ref. [Kap+21].

Finally, until now, we have assumed that all clients are of the same type, i.e. all are in prepare-and-send or all are in receive-and-measure. However, there is nothing that imposes this restriction, and we could in fact envision a setting where clients with different capabilities interact during a DMPQC protocol. Our constructions enable building up such a protocol for the first time; that is the topic of the next section.

6 Mixing client types

The equivalence transformation between the receive-and-measure and prepare-and-send routines allows us to consider different types of clients in the same DMPQC protocol. This novel approach does not restrict the clients’ capabilities to only one type (either receive-and-measure or prepare-and-send) and provides a great degree of flexibility compared with all previous proposals. We can therefore consider a set of clients where some of them can prepare while others can measure states in a chosen basis. We assume that the server can perform both functionalities, i.e. from the prepare-and-send and the receive-and-measure scheme. This assumption allows us to build a compatible DBSP protocol.

Fig. 7. An example of a DBSP protocol with mixed client types. For any prepare-and-send-client $i$ we have $r_i = 0$ and for any receive-and-measure client $j$ we set $s_j = 0$.

Fig. 7 exhibits an example for such a DBSP protocol. One can easily see that

$$\theta = \theta_0 + ((-1)^{s_1} \theta_1) + (\theta_2 + r_2 \pi) + \cdots + (\theta_N + \pi r_N)$$

$$= \theta_0 + \sum_{i \in P} (-1)^{s_i} \theta_i + \sum_{i \in R} \theta_i + r_i \pi,$$
where $P$ is the set of prepare-and-send clients and $R$ is the set of receive-and-measure clients. The scheme shown in the figure remains totally blind for colluding server and N-1 clients. The SMPC has to consider the different types of corrections for the result, and the server has to prepare the $|0\rangle$ state vectors for the receive-and-measure clients.

7 Discussion and open questions

In this work, we have comprehensively examined the relation between the different models of delegated quantum computation, aimed at providing a unifying framework. By introducing a new notion of protocol equivalence in the abstract cryptography framework, we have shown that the two widely-used models for delegated quantum computation, namely the prepare-and-send and receive-and-measure are in fact equivalent. The new definitions and theorems that we provide are of independent interest and could potentially be used in other settings as well. In our work, they have enabled us to show how, given a protocol in one of the two delegated computation models, we can directly obtain a protocol in the other. Additionally, we have used this new notion of equivalence to introduce a new delegated multiparty quantum computing protocol that works in a hybrid setting, i.e., it can accommodate clients from the prepare-and-send and receive-and-measure setting.

Still, some interesting questions remain to be answered. The most prominent one is to examine whether other tasks can also profit from our proposed equivalence transformation. This would entail figuring out what other routines can be similarly transformed in different communication settings in order to utilise our general definitions of equivalence. Another topic is the choice of the security framework. We have chosen the abstract cryptography framework as a natural composability framework to explore, also due to the existing previous studies, but there are other ones, e.g., the categorical composable cryptography framework recently proposed by Broadbent and Karvonen \cite{BK22}, where similar equivalence transformations can exist. Finally, it remains to be seen which of the different models of delegated computation will be more adequate for specific communication settings in real-life implementations once we are able to perform universal quantum computations. Until then, smaller implementations of delegated quantum computation \cite{Bar+12, Gre+16} might provide hints on the potential of the different models. It is the hope that our work stimulates such further developments. On a higher level, this work is aimed at contributing to bringing the fields of quantum computation and communication more closely together.

8 Acknowledgements

F. W. and A. P. acknowledge support from the Emmy Noether DFG grant No. 418294583, J. E. from the BMBF (QR.X). J. E. and A. P. also acknowledge support from the Einstein Research Unit on Quantum Devices, for which this is a joint node project.
References

[Bar+12] Stefanie Barz et al. “Experimental demonstration of blind quantum computing”. In: Science 335 (2012), pp. 303–308. DOI: 10.1126/science.1214707

[Bel64] John S. Bell. “On the Einstein Podolsky Rosen paradox”. In: Physics Physique Fizika 1 (1964), p. 195.

[BFK09] Anne Broadbent, Joseph Fitzsimons, and Elham Kashefi. “Universal blind quantum computation”. In: 2009 50th Annual IEEE Symposium on Foundations of Computer Science (2009). DOI: 10.1109/focs.2009.38

[BK22] Anne Broadbent and Martti Karvonen. “Categorical composable cryptography”. In: Lecture Notes in Computer Science. Springer International Publishing, 2022, pp. 161–183. DOI: 10.1007/978-3-030-99253-8_9

[Bri+09] Hans J. Briegel et al. “Measurement-based quantum computation”. In: Nature Physics 5 (2009), pp. 19–26. DOI: 10.1038/nphys1157

[Can00] Ran Canetti. “Universally composable security: A new paradigm for cryptographic protocols”. In: (2000). https://ia.cr/2000/067

[CF01] Ran Canetti and Marc Fischlin. “Universally composable commitments”. In: (2001). https://eprint.iacr.org/2001/055

[Chi05] Andrew M. Childs. “Secure assisted quantum computation”. In: Quantum Information & Computation 5 (6 2005), pp. 456–466.

[CKL03] Ran Canetti, Eyal Kushilevitz, and Yehuda Lindell. “On the limitations of universally composable two-party computation without set-up assumptions”. In: Journal of Cryptology. 2003, pp. 68–86.

[Dir39] Paul A. M. Dirac. “A new notation for quantum mechanics”. In: Mathematical Proceedings of the Cambridge Philosophical Society 35 (1939), pp. 416–418. DOI: 10.1017/S0305004100021162

[Dun+14] Vedran Dunjko et al. “Composable security of delegated quantum computation”. In: Advances in Cryptology – ASIACRYPT 2014. Ed. by Palash Sarkar and Tetsu Iwata. Berlin, Heidelberg: Springer Berlin Heidelberg, 2014, pp. 406–425.

[EPR35] Albert Einstein, Boris Podolsky, and Nathan Rosen. “Can quantum-mechanical description of physical reality be considered complete?” In: Phys. Rev. 47 (1935), p. 777.

[Fit17] Joseph F. Fitzsimons. “Private quantum computation: an introduction to blind quantum computing and related protocols”. In: npj Quantum Information 3 (2017), p. 23. DOI: 10.1038/s41534-017-0025-3

[FK17] Joseph F. Fitzsimons and Elham Kashefi. “Unconditionally verifiable blind quantum computation”. In: Phys. Rev. A 96 (2017). DOI: 10.1103/physreva.96.012303

[GE07] David Gross and Jens Eisert. “Novel schemes for measurement-based quantum computation”. In: Phys. Rev. Lett. 98 (2007), p. 220503. DOI: 10.1103/PhysRevLett.98.220503
[Gre+16] Chiara Greganti et al. “Demonstration of measurement-only blind quantum computing”. In: New J. Phys. 18 (2016), p. 013020. DOI: 10.1088/1367-2630/18/1/013020

[HEB04] Marc Hein, Jens Eisert, and Hans J. Briegel. “Multiparty entanglement in graph states”. In: Phys. Rev. A 69 (6 2004), p. 062311. DOI: 10.1103/PhysRevA.69.062311.

[HM15] Masahito Hayashi and Tomoyuki Morimae. “Verifiable measurement-only blind quantum computing with stabilizer testing”. In: Phys. Rev. Lett. 115 (22 2015), p. 220502. DOI: 10.1103/PhysRevLett.115.220502.

[Hou+18] Monireh Houshmand et al. “Composable secure multi-client delegated quantum computation”. In: (2018). arXiv: 1811.11929 [quant-ph].

[Joz05] Richard Jozsa. “An introduction to measurement based quantum computation”. In: (2005). URL: https://arxiv.org/pdf/quant-ph/0508124.pdf.

[Kap+21] Theodoros Kapourniotis et al. “Delegating multi-party quantum computations vs. dishonest majority in two quantum rounds”. In: (2021). arXiv: 2102.12949 [quant-ph].

[KP17] Elham Kashefi and Anna Pappa. “Multiparty delegated quantum computing”. In: Cryptography 1 (2017), p. 12. DOI: 10.3390/cryptography-2020012.

[KW17] Elham Kashefi and Petros Wallden. “Optimised resource construction for verifiable quantum computation”. In: J. Phys. A 50 (2017), p. 145306. DOI: 10.1088/1751-8121/aa5dac.

[MF13] Tomoyuki Morimae and Keisuke Fujii. “Blind quantum computation protocol in which Alice only makes measurements”. In: Phys. Rev. A 87 (2013). DOI: 10.1103/physreva.87.050301.

[MK13] Tomoyuki Morimae and Takeshi Koshiba. “Composable security of measuring-Alice blind quantum computation”. In: arXiv:1306.2113 [quant-ph] (2013). arXiv: 1306.2113.

[Mor14] Tomoyuki Morimae. “Verification for measurement-only blind quantum computing”. In: Phys. Rev. A 89 (6 2014), p. 060302. DOI: 10.1103/PhysRevA.89.060302.

[MR11] Ueli Maurer and Renato Renner. “Abstract cryptography”. In: Innovations in Computer Science. Tsinghua University Press, 2011.

[NC10] Michael A. Nielsen and Isaac L. Chuang. Quantum computation and quantum information: 10th Anniversary Edition. Cambridge University Press, 2010. DOI: 10.1017/CBO9780511976667.
A Support material

A.1 An example: Quantum coin tossing

We now turn to discussing an example to highlight the subtle differences between the notions defined above. A specifically suitable example that elucidates the different definitions is that of coin tossing. We start by defining the ideal resource for coin tossing $F^C_{CT}$, which gives the same random bit to two parties, Alice and Bob (Fig. 8).

![Fig. 8. Ideal resource of coin tossing. $F^C_{CT}$ sends a random bit $b$ to Alice and Bob.](image)

To securely construct the ideal resource $F^C_{CT}$, let us assume that there exists a classical resource $R^{⊕}$ that computes the XOR of two bits. $R^{⊕}$ has two interfaces connecting to the protocols of Alice and Bob (Fig. 9), where it receives one bit from each and outputs the XOR of the two bits to both parties. We assume that at least one of the two parties is honest (since it does not make sense from a security perspective to guarantee security when both parties are dishonest). Without loss of generality, we assume that Alice is always honest, but a similar analysis also holds for Bob. If Bob is also honest, the outcome $b$ is uniformly random. It, therefore, holds that

$$d(\pi_C^{AR} \otimes \pi_C^{BR}, F^C_{CT}) = 0.$$  \hspace{1cm} (A.1)

![Fig. 9. Visualisation of the coin tossing protocol $\pi^C$ when both parties are honest. Alice and Bob sample random bits and send these to the resource $R^{⊕}$, which outputs the XOR of the bits.](image)

To prove security when Bob is dishonest, we need to introduce a simulator $\sigma^C_B$ for Bob in the ideal world. $\sigma^C_B$ will receive $x_B$ from the right interface and $b$ from the left (coming from the ideal resource $F^C_{CT}$). The simulator will forward $b$ and ignore
\( x_B \). As the distinguisher \( D \) does not learn \( x_A \) when Alice is honest, \( b = x_A \oplus x_B \) is uniformly random for \( D \) in both settings, i.e.,

\[
d(\pi^C_A \mathcal{R}^\oplus, \mathcal{F}^\text{CT} \sigma^C_B) = 0.
\] (A.2)

From Eqs. (A.1) and (A.2), we have \( \mathcal{R}^\oplus \xrightarrow{(A)} \mathcal{F}^\text{CT} \), which means that, conditioned on Alice being honest, protocol \( \pi^C \) securely constructs \( \mathcal{F}^\text{CT} \) from \( \mathcal{R}^\oplus \).

To test the definition of translatability (Def. 3.7) for our example, we need a second implementation of \( \mathcal{F}^\text{CT} \).

We find \( \{\pi^Q_{Q^\text{BS}}\} \xrightarrow{0}{\{A\}} \{\pi^C \mathcal{R}^\oplus\} \). For Alice and Bob being honest, we find perfectly correlated but uniformly random measurement results, as the qubits of Alice and Bob are maximally entangled, so it holds that

\[
d(\pi^Q_A \mathcal{Q}^\text{BS} \sigma^Q_B, \pi^C_A \mathcal{R}^\oplus \sigma^C_B) = 0.
\] (A.3)

Next, we assume Bob is dishonest. His simulator \( \sigma^C_B \) gets the query ‘q’ from Bob, samples randomly \( x_B \) and sends it to \( \mathcal{R}^\oplus \). After receiving \( b \), it creates \( |b_B\rangle \) and sends it to Bob. In the simulation, \( D \) gets at Alice’s side a random \( b \) and at Bob’s side \( |b_B\rangle \).

In the real world, we can assume that Alice measures first as the measurements commute. Afterwards, Bob’s qubit is in \( |b_B\rangle \) and Alice outputs \( b \). Hence, in both settings \( D \) gets the same output and we find

\[
d(\pi^Q_A \mathcal{Q}^\text{BS} \sigma^Q_B \sigma^C_B) = 0.
\] (A.4)

From Eqs. (A.3) and (A.4), we have \( \{\pi^Q \mathcal{Q}^\text{BS}\} \xrightarrow{0}{\{A\}} \{\pi^C \mathcal{R}^\oplus\} \), which means that, conditioned on Alice being honest, protocol \( \pi^Q \) applied on \( \mathcal{Q}^\text{BS} \) is perfectly translatable to \( \pi^C \) applied on \( \mathcal{R}^\oplus \). From that we can directly follow \( \mathcal{Q}^\text{BS} \xrightarrow{(A)} \mathcal{F}^\text{CT} \) using

---

**Fig. 10. Visualisation of the coin-tossing protocol \( \pi^Q \) when both parties are honest.** The parties Alice and Bob query the functionality \( \mathcal{Q}^\text{BS} \). This functionality creates the state vector \( |0_A,0_B\rangle + |1_A,1_B\rangle/\sqrt{2} \) and sends one qubit to each party. Both parties measure their qubit in the Pauli Z-basis and output the measurement result.
Theorem 3.1. At last we show \( \{\pi^Q_{\text{BS}}\} \) \( \xrightarrow{\text{L}} \} \{\pi^C_{\text{R}} \} \) by using Theorem 3.2. To use this theorem, we only have to prove perfect translatability from the classical setting to the quantum setting, conditioned on Alice being honest, i.e. \( \{\pi^C_{\text{R}} \} \xrightarrow{0} \)} \{\pi^Q_{\text{BS}}\}. If Bob is honest, we can use Eq. (A.3), as \( d(\cdot,\cdot) \) is symmetric. If Bob is dishonest, we use the simulator \( \sigma^Q_{\text{BS}} \). The simulator receives \( x_B \) and afterwards sends ‘q’. After that, it receives a qubit and measures this qubit in the \( Z \)-basis, the result is \( b \). In the last step, \( \sigma^Q_{\text{BS}} \) outputs \( b \). As \( x_A \) remains secret in the classical setting and the output \( b \) is at both sides uniformly random but the same in both settings, we find

\[
d(\pi^C_{\text{R}} \oplus \pi^Q_{\text{BS}}, \sigma^Q_{\text{BS}}) = 0.
\]

From Eqs. (A.3),(A.4), (A.5) and Theorem 3.2, we find that these two implementations of \( \mathcal{F}_{\text{FC}} \) are equivalent if Alice is always honest, i.e., \( \{\pi^Q_{\text{BS}}\} \xrightarrow{\text{L}} \) \{\pi^C_{\text{R}} \} \( \xrightarrow{\text{L}} \} \{\pi^Q_{\text{BS}}\} \)

In fact, we find similar results if Bob is always honest and Alice is not trusted. This is because \( \mathcal{F}_{\text{FC}} \), \( \text{R}^\oplus \), \( \text{BS}^Q \) and the protocols are symmetric.

A.2 Proof of Theorem 3.2

In this supporting material, some proofs are presented that are omitted in the main text. Theorem 3.2 states that two protocols \( \pi^1 \) and \( \pi^2 \) are equivalent if and only if one perfectly translates to the other, i.e.,

\[
\left( \{\pi^1_{\text{R}} \} \xrightarrow{0} \{\pi^2_{\text{Q}} \} \land \{\pi^2_{\text{Q}} \} \xrightarrow{0} \{\pi^1_{\text{R}} \} \right) \iff \left( \{\pi^1_{\text{R}} \} \xrightarrow{\mathcal{H}} \{\pi^2_{\text{Q}} \} \right).
\]

One direction of the proof follows directly from Theorem 3.1 as stated in the main text. To prove the other direction we need the following propositions and lemmata.

**Proposition 1.** Let \( \hat{\mathcal{S}}_{\psi} \) an ideal resource such that

\[
\forall \mathcal{P} \subseteq \mathcal{I}, \mathcal{P} \subseteq \mathcal{H}: d(\pi^1_{\text{R}} \phi \mathcal{P}, \sigma^1_{\mathcal{I}\backslash \mathcal{P}} \psi \mathcal{P} \hat{\mathcal{S}}) = d(\pi^2_{\text{Q}} \chi \mathcal{P}, \sigma^2_{\mathcal{I}\backslash \mathcal{P}} \psi \mathcal{P} \hat{\mathcal{S}}) = 0 \quad (A.6)
\]

and \( \sigma^1 = \gamma \sigma^2 \). Then it holds that \( \{\pi^1_{\text{R}} \} \xrightarrow{\mathcal{H}} \{\pi^2_{\text{Q}} \} \).

**Proof.** This can be proven as follows. We have that

\[
\forall \mathcal{P} \subseteq \mathcal{I}, \mathcal{P} \subseteq \mathcal{H}: d(\pi^1_{\text{R}} \phi \mathcal{P}, \sigma^1_{\mathcal{I}\backslash \mathcal{P}} \psi \mathcal{P} \hat{\mathcal{S}}) = d(\pi^2_{\text{Q}} \chi \mathcal{P}, \sigma^2_{\mathcal{I}\backslash \mathcal{P}} \psi \mathcal{P} \hat{\mathcal{S}}) = 0
\]

\[
\Rightarrow \forall \mathcal{P} \subseteq \mathcal{I}, \mathcal{P} \subseteq \mathcal{H}: d(\pi^1_{\text{R}} \phi \mathcal{P}, \gamma \chi \mathcal{P} \circ \sigma^2_{\mathcal{I}\backslash \mathcal{P}} \psi \mathcal{P} \hat{\mathcal{S}}) = d(\pi^2_{\text{Q}} \chi \mathcal{P}, \sigma^2_{\mathcal{I}\backslash \mathcal{P}} \psi \mathcal{P} \hat{\mathcal{S}}) = 0
\]

\[
\Rightarrow \forall \mathcal{P} \subseteq \mathcal{I}, \mathcal{P} \subseteq \mathcal{H}: d(\pi^1_{\text{R}} \phi \mathcal{P}, \gamma \chi \mathcal{P} \circ \pi^2_{\text{Q}} \chi \mathcal{P} \mathcal{Q}) = 0,
\]

from which the claim follows. \( \square \)
Proposition 2. For every protocol $\pi^2$ applied on $Q_\chi$ there exists an ideal resource $\tilde{S}_\psi$ such that
\[
\forall P \subseteq I : d(\pi^2_{\chi \psi} Q_\chi, \sigma^2_{\chi \psi} \psi P \tilde{S}) = 0,
\] (A.7)
where all $\sigma_j^2$ just send $c_j = 1$ and forward any message. We name $\tilde{S}_\psi$ the trivial resource and $\sigma^2 = \{\sigma_j^2\}_{j \in I}$ the trivial simulators of $\pi^2$ applied on $Q_\chi$.

Proof. We can easily define $\tilde{S}_\psi$ to act for an honest configuration, which it learns due to the simulators, as $\pi^2_{\chi \psi} Q_\chi$. $\square$

Lemma 1. Assume $\{\pi^1 R_\phi\} \xrightarrow{H} \{\pi^2 Q_\chi\}$, then it holds
\[
\{\pi^1 R_\phi\} \xrightarrow{H} \{\pi^2 Q_\chi\} \land \{\pi^2 Q_\chi\} \xrightarrow{H} \{\pi^1 R_\phi\}.
\]

Proof. We can directly use Proposition 2 and find the trivial resources and simulators for $\pi^1$ applied on $R_\phi$, and $\pi^2$ applied on $Q_\chi$. To use Proposition 2, we note that for every ideal resource the distinguishing-advantage for the two protocols is the same, as we have equivalence. Further, we show that any simulator $\lambda_j$ can be implemented by $\gamma_j \circ \sigma_j$, where $\sigma_j$ is the trivial simulator for party $j$.

If $\lambda_j$ sends $c_j = 1$ it already contains $\sigma_j$ and $\gamma_j$ is determined by the rest of $\lambda_j$. If $\lambda_j$ sends $c_j = 0$ we find $\lambda_j = \gamma_j \circ \sigma_j$, where $\gamma_j$ blocks all messages from the resource, that would not be sent, if $c_j = 0$. We can now use Proposition 2 and thus complete the proof of the Theorem. $\square$

A.3 Detailed proof for section 4.2

We start with the detailed proof for Theorem 4.2

The prepare-and-send routine $\pi^{ps}$ (Def. 4.3) is perfectly translatable with respect to the PBDS into the receive-and-measure routine $\pi^{rm}$ when both are applied on a quantum communication channel $Q$, i.e.:
\[
\{\pi^{ps} Q\} \xrightarrow{0, \phi} \{\pi^{rm} Q\}
\]

Proof. The output of a prepare-and-send protocol $\rho_x'$ is given up to normalisation by
\[
\rho_x' = (|M_{s|x}| \otimes 1_x) \Omega_{q,x} (|P_{r|x}| \otimes \rho_x) \Omega_{q,x}^\dagger (|M_s| \otimes 1_x) = A_{r,s} \rho_x A_{r,s}^\dagger
\]
with $A_{r,s} = (\langle M_{s|q} \otimes I_x \rangle \Omega_{q,x} (\langle P_{r|q} \otimes I_x \rangle).

Writing everything acting in $q$ in a basis one finds

$$A_{r,s} = \left( \sum_{a=0}^{1} M_{s,a}^* \langle a_q | \otimes I_x \right) \left( \sum_{b,c=0}^{1} | b_q \rangle \langle c_q | \otimes \Theta_{b,c|x} \right) \left( \sum_{d=0}^{1} | d_q \rangle \otimes I_x \right)$$

$$= \sum_{b,c=0}^{1} \Theta_{b,c|x} M_{s,b}^* P_{r,c}$$

(A.8)

where we have denoted $\Omega_{q,x} = \sum_{b,c=0}^{1} | b_q \rangle \langle c_q | \otimes \Theta_{b,c|x}$ with $\Theta_{b,c|x} \in \mathbb{C}^d \otimes d$ acting on the register labelled with $x$. The output of the receive-and-measure routine $\tilde{\rho}'_x$ for the same input is given up to normalisation by

$$\rho'_x = B_{r,s} \rho_x B_{r,s}^\dagger$$

with

$$B_{r,s} = (\langle P_{r|q} \otimes I_x \rangle \Omega_{q,x}^T (\langle M_{s|q} \otimes I_x \rangle)$$

$$= \left( \sum_{a=0}^{1} P_{r,a} \langle a_q | \otimes I_x \right) \left( \sum_{b,c=0}^{1} | c_q \rangle \langle b_q | \otimes \Theta_{b,c|x} \right) \left( \sum_{d=0}^{1} M_{s,d}^* | d_q \rangle \otimes I_x \right)$$

$$= \sum_{b,c=0}^{1} \Theta_{b,c|x} M_{s,b}^* P_{r,c} = A_{r,s}$$

(A.9)

This means that $\rho'_x = \tilde{\rho}'_x$ and therefore we have

$$d(\pi_C^{ps} \pi_C^{\pi_{ps}} \pi_S^{rm} \pi_S^{\pi_{rm}}) = 0.$$ 

If one or both parties are dishonest, we need simulators. For the remainder of the proof we use the simulators depicted in Fig. 4. Further, we note that in favour of readability, we sometimes omit normalisation since the proofs are not affected in any way.

The two simulators that we propose make use of the properties of Bell states, specifically that all four states can be generated from any of them using Pauli operators:

$$\{ | \Phi^{B_0,B_1}_{a,b} \rangle \}_{B_0,B_1=0}^1 = \{ | 1_a \otimes X_B | 0_a = 0 \rangle + | 1_a \otimes 1_b \rangle \}_{B_0,B_1=0} \sqrt{2}.$$ 

In case of a dishonest server, we use the simulator $\sigma^{B, prep}$ for the server in receive-and-measure. Using

$$\langle P_{r|q} | \Phi^{0,0}_{q,i} \rangle = \left( \sum_{a=0}^{1} P_{r,a} \langle a_q | \right) \left( \sum_{b=0}^{1} \frac{1}{\sqrt{2}} | b_q \rangle \langle b_i | \right)$$

$$= \sum_{b=0}^{1} \frac{1}{\sqrt{2}} P_{r,b} | b_i \rangle = | P_{r|i} \rangle$$

(A.10)
Hence, we find indistinguishably using $\sigma^{B,\text{prep}}$ is the server is dishonest.
In the next case, the client is dishonest but the server is honest. Here we use $\sigma^{B,\text{meas}}$ for the client in receive-and-measure and find

$$
\begin{align*}
&\langle \Phi_{B_0, B_1} \rangle (1 \otimes \Omega^T_q, x) (1 \otimes |M_{s,q} \rangle \otimes 1_x) = \\
&= \left( \sum_{a=0}^1 (-1)^{B_0 \cdot a} \langle a, (a \oplus B_1) \rangle \right) \left( \sum_{b, c=0}^1 |b, c \rangle \otimes \theta_{b, c} \right) \left( \sum_{d=0}^1 \text{M}_{s, d}^* |d_q \rangle \right) \\
&= \left( \sum_{b, c=0}^1 \frac{1}{\sqrt{2}} \text{M}_{s, b}^* (c \otimes \theta_{b, c}) \right) X_{B_0} B_1 Z_{B_0} \\
&= \left( \sum_{c=0}^1 \frac{1}{\sqrt{2}} \text{M}_{s, c}^* (c) \right) \left( \sum_{b, c=0}^1 |b, c \rangle \otimes \theta_{b, c} \right) X_{B_0} B_1 Z_{B_0} \\
&= \frac{1}{\sqrt{2}} \langle \text{M}_{s, c} | \Omega_{i, x} X_{B_0} B_1 Z_{B_0} \rangle,
\end{align*}
$$

which implies

$$
d(Q \pi^{ps}_S \sigma^{B,\text{meas}} \sigma^{rm}_S Q_{\pi^{rm}}) = 0. \tag{A.12}
$$

We finally assume that both parties are dishonest. For this case we use the property

$$
\langle 1 \otimes \langle \Phi_{B_0, B_1} \rangle (|\Phi^{0, 0} \rangle \otimes 1) = \left( \sum_{a=0}^1 (-1)^{B_0 \cdot a} \langle a, (a \oplus B_1) \rangle \right) \left( \sum_{b=0}^1 \frac{1}{\sqrt{2}} |b, b \rangle \right) \\
= \frac{1}{2} \sum_{a=0}^1 (-1)^{B_0 \cdot a} |a, a \rangle \langle (a \oplus B_1) \rangle = \frac{1}{2} Z_{B_0} X_{B_1} \tag{A.13}
$$

of the Bell states. One can already see in Fig. 4 that the concatenation of $\sigma^{B,\text{prep}}$ and $\sigma^{B,\text{meas}}$ is exactly described by Eq. (4.5) and this gives

$$
d(Q, \sigma^{B,\text{meas}} \sigma^{B,\text{prep}} Q) = 0. \tag{A.14}
$$

Next we give the detailed proof of Theorem 4.3.

The receive-and-measure routine $\pi^{rm}$ (Def. 4.4) is perfectly translatable with respect to the PBDS into the prepare-and-send routine $\pi^{ps}$ when both are applied on a quantum communication channel $Q$, i.e.:

$$
\{ \pi^{rm} Q \} \xrightarrow{0,\varnothing} \{ \pi^{ps} Q \}
$$
Proof. First note, that if the server and the client are honest we can use the same calculation as in Theorem 4.2. The same holds if both are dishonest. The remaining cases are those where only one of the parties is dishonest. If the server is dishonest we almost have the same situation as in the case of a dishonest client in Theorem 4.2 and find

\[
\langle q_{i,q} | B_{1} | P r_{i} \rangle = \left( \sum_{a=0}^{1} (-1)^{a} \frac{B_{0}}{\sqrt{2}} \langle a_i, (a \oplus B_{1}) | q \rangle \right) \left( \sum_{b=0}^{1} P r_{b} | b_q \rangle \right)
\]

\[
= \sum_{a=0}^{1} \frac{(-1)^{(a \oplus B_{1})}B_{0}}{\sqrt{2}} P r_{a} | (a \oplus B_{1}) | i \rangle \langle i | B_{1} | Z_{0} \rangle
\]

\[
= \frac{1}{\sqrt{2}} \langle P r_{i} | i \rangle X_{i}^{B_{1}} Z_{i}^{B_{0}},
\]

(A.15)

which implies

\[
d(\pi^{rm}_{C} Q_{i}^{ps} C_{Q} \pi^{prep}_{S} \sigma^{B meas}_{S}) = 0.
\]

(A.16)

At last we assume a dishonest client, which is similar to a dishonest server in Theorem 4.2. Using

\[
(1_i \otimes (M_{s,q} | \otimes I_{x}) (1_i \otimes \Omega_{q,x}) \left( \sum_{i}^{0,0} q_{i,q} \right) \otimes I_{x})
\]

\[
= \left( \sum_{a=0}^{1} M_{s,a}^{*} | a_q \rangle \right) \left( \sum_{b,c=0}^{1} | b_q \rangle \otimes \theta_{b,c} | c_q \rangle \right) \left( \sum_{d=0}^{1} \frac{1}{\sqrt{2}} | d_q \rangle \right)
\]

\[
= \left( \sum_{b,c=0}^{1} \frac{M_{s,b}^{*} | c_q \rangle \otimes \theta_{b,c} | b_q \rangle}{\sqrt{2}} \right)
\]

\[
= \frac{1}{\sqrt{2}} \left( \sum_{b,c=0}^{1} | c_q \rangle \otimes \theta_{b,c} | b_q \rangle \right) \left( \sum_{d=0}^{1} M_{s,d}^{*} | d_q \rangle \right) = \frac{1}{\sqrt{2}} \Omega^{T_{i,x}}_{i} (| \mathcal{M}_{s} | i \rangle \otimes I_{x}).
\]

(A.17)

we find

\[
d(Q \pi^{rm}_{S} \sigma^{B prep}_{C} \pi^{prep}_{S} Q_{s}) = 0.
\]

(A.18)

□
A.4 BDQC-protocols

Protocol 5: The BDQC-protocol proposed by Broadbent et al. [BFK09]. Alice (the client) has the computation compiled into measurement angles $\{\phi_{x,y}\}_{x=0,y=0}^{n-1,m}$.

1. Input encryption
1.1: for Input qubit in first column ($x=0,y=1,...,m$) do
1.2: Alice samples $\theta_{0,y} \leftarrow R$ and $i_{0,y} \leftarrow R \{0,1\}$, applies $P(\theta_{0,y})X^{i_{0,y}}$ and sends the qubit to Bob.
1.3: end for

2. Auxiliary qubit preparation
2.1: for Column $x, x=1,...,n-1$ do
2.2: for Row $y, y=1,...,m$ do
2.3: Alice samples $\theta_{x,y} \leftarrow R$ sends $\left|+\theta_{x,y}\right>$ to Bob (the server).
2.4: end for
2.5: end for

3. Output preparation
3.1: for Output qubit in last column ($x=n,y=1,...,m$) do
3.2: Alice prepares $\left|+\right>$ and sends it to Bob.
3.3: end for

4. Bob builds up the state
4.1: Bob entangles the qubits with CZ according to the specifications of the brickwork state.

5. Interaction, measurement and output
5.1: for Column $x, x=1,...,n-1$ do
5.2: for Row $y, y=1,...,m$ do
5.3: Alice calculates the corrected secret measurement angle $\phi'_{x,y}$ with the special case $\phi'_{0,y} = (-1)^{i_{x,y}} \phi_{0,y}$.
5.4: Alice samples $r_{x,y} \leftarrow R \{0,1\}$ and computes $\delta_{x,y} = \phi'_{x,y} + \theta'_{x,y} + r_{x,y} \pi$ and sends $\delta_{x,y}$ to Bob.
5.5: Bob measures the node in $\{\pm\delta_{x,y}\}$ and sends the result $s_{x,y}$ to Alice.
5.6: Alice computes $s_{x,y} \leftarrow s_{x,y} \oplus r_{x,y}$.
5.7: end for
5.8: end for
5.9: Bob sends the output column $x=n$ to Alice.
5.10: Alice applies the Pauli-corrections, which she knows due to the flow-function.
Protocol 6: BDQC-protocol by Morimae and Fujii [MF13]. We assume that the brickwork state is used and also, that the computation is compiled into secret measurement angles \( \{\phi_{x,y}\}_{x=0,y=0}^{n-1,m} \).

1. Input encryption
   1.1: for Input qubit in first column \((x=0,y=1,...,m)\) do
   1.2: Alice samples \( \theta_{0,y} \leftarrow R.A \) and \( i_{0,y} \leftarrow R\{0,1\} \), applies \( P(\theta_{0,y})X^{i_{0,y}} \) and sends the qubit to Bob.
   1.3: end for

2. Auxiliary qubit and output preparation
   2.1: for Column \( x, x=1,...,n \) do
   2.2: for Row \( y, y=1,...,m \) do
   2.3: Bob prepares \( |+\rangle \).
   2.4: end for
   2.5: end for

3. Bob builds up the state
   3.1: Bob entangles the qubits with CZ according to the specifications of the brickwork state.

4. Interaction, measurement and output
   4.1: for Column \( x, x=1,...,n-1 \) do
   4.2: for Row \( y, y=1,...,m \) do
   4.3: Alice calculates the corrected secret measurement angle \( \phi'_{x,y} \) with the special case \( \phi'_{0,y} = (-1)^{i_{0,y}} \phi_{0,y} + \theta_{x,y} \).
   4.4: Bob sends the qubit to Alice, who measures in \( \{±\phi'_{x,y}\} \). The result is used for further corrections of the angles.
   4.5: end for
   4.6: end for
   4.7: Bob sends the output column \( x=n \) to Alice.
   4.8: Alice applies the Pauli-corrections, which she knows due to the flow-function.
A.5 Verifiable blind quantum computation (VBDQC)

In this section, we first present the protocol by Kashefi and Wallden [KW17], which achieves blindness and verifiability. Afterwards, we will use the equivalence-transformation to obtain a new equivalent protocol in receive-and-measure.

**VBDQC in prepare-and-send [KW17]**

To understand the VBDQC-protocol by Kashefi and Wallden [KW17], one first needs to know the implementation of the traps, i.e., the isolated qubits that are in a state only known by the client. In order to include such isolated qubits, Kashefi and Wallden proposed the construction of the dotted triple-graph. In this graph, there exists for every computational node a trap and a dummy; the dummies are implemented with $Z$-eigenstates; the traps are isolated, as all their neighbours are dummies. This idea is similar to the approach by Fitzsimons and Kashefi [FK17]. But while in [FK17] a high verifiability advantage for the client with cost of quadratic scaling of the overhead is achieved, [KW17] achieved linear scaling. Protocol 7 describes purely mathematically how one obtains from an arbitrary graph $G$ the coloured dotted triple-graph $G'$. A visualisation with an example is in Fig. 11.

Kashefi and Wallden [KW17] use offsets in the preparation as it is done in Protocol 5. For that, we present a translation of Protocol 7 into physical operations in Protocol 8. As Bob has the coloured dotted triple-graph with unknown trap positions and offsets, Alice and Bob can run Protocol 5 beginning with Step 5. Alice uses random $\delta$ for the dummies and $\delta = \theta + r \pi$ for the traps. Also, the labelling of the nodes may differ from the one of the brickwork-state. Alice knows that Bob cheated if there is for any trap a measurement result which is not $r$. 

---

**Fig. 11.** Visualisation of Protocol 7 with the input graph as shown in a).
**Protocol 7:** Mathematical construction of the coloured dotted triple-graph.

**Require:** Base graph \( G = (V, E) \)

7.1: Let \( V' = \{\} \) be the list of vertices of the output graph and \( E' = \{\} \) the corresponding edges.

7.2: for every \( v \in V \) do

7.3: Append \( v_1, v_2 \) and \( v_3 \) to \( V' \)

7.4: for \((u,v) \in E \) and \((i,j) \in \{1,2,3\}^2 \) do

7.5: Append \( d_{u_i,v_j} \) to \( V' \) and append \((u,d_{u_i,v_j})\) and \((d_{u_i,v_j},v_j)\) to \( E' \). We name the set of this new vertices "dots", noted with \( \Delta, \Delta' \subset V' \).

7.6: end for

7.7: end for

7.8: We name this graph \( DT(G) \), Alice saves a copy of that.

7.9: For every \( v \in V \) assign randomly the colours Green, Blue and Gray to the node \( v_1, v_2, v_3 \) in \( V' \).

7.10: For every \( d \in \Delta \), if the two neighbours have the same colour, \( d \) get also this colour, Red otherwise.

7.11: Delete all Red nodes in \( V' \), all Green nodes in \( V' \) that are also in \( \Delta \) and all Blue nodes in \( V' \) that are not in \( \Delta \) and the corresponding edges in \( E' \). Alice saves the positions of this deleted nodes in \( D \).

7.12: return \( G' = (V', E') \), the colouring, \( DT(G) = (V_{DT(G)}, E_{DT(G)}) \), \( D, T \)

---

**Protocol 8:** Preparation of the coloured dotted triple-graph state. Alice has a set of input- and output nodes in her resource state.

8.1: Alice runs Protocol 7 with the graph of her resource state.

8.2: Alice declares all the nodes that are deleted in Protocol 7 to be dummies. This set is \( D \).

8.3: Alice declares all the nodes that are not Gray and in \( V' \) to be traps, which we note with \( T \).

8.4: All Gray nodes in \( V' \) are computation nodes, noted with \( C \).

8.5: Alice sends the graph topology of \( DT(G) \) to Bob.

8.6: Alice maps the input nodes \( (I) \) and the output nodes \( (O) \) to the corresponding nodes in \( C \).

8.7: for \( v \in V_{DT(G)} \) do

8.8: If \( v \in D \) Alice samples \( d \sim_R \{0,1\} \) and sends \(|d\rangle\) to Bob.

8.9: If \( v \in T \cup C \land v \notin I \cup O \) Alice samples \( \theta_v \sim_R A \) and sends \(|+\rangle\).

8.10: If \( v \in I \) samples \( \theta_v \sim_R A \) and \( i_v \sim_R \{0,1\} \) and sends \( P(\theta_v)X^{i_v}(\rho_v) \) to Bob.

8.11: If \( c \in O \) Alice sends \(|+\rangle\).

8.12: end for

8.13: Bob connects the qubit according to the topology he got from Alice.

8.14: Alice needs to correct her notes about the offsets, as the dummies may introduce a phase of \( \pi \). For \( v \in T \cup C; \theta_v \sim -\theta_v + \pi \bigoplus_{d \in D \cap N(v)} d \), where \( N(v) \) are the neighbouring nodes of \( v \).
Protocol 9: VBDQC by Kashefi and Wallden in prepare-and-send. \cite{KW17}

9.1: Alice and Bob create the coloured dotted triple-graph by using Protocol 8.

9.2: for every node \( v \) of the original graph of the underlying resource state \( G \) in \( O \) or in the to \( O \) corresponding sets of traps and dummies do

9.3: for each corresponding qubit \( u \in \{v_0, v_1, v_2\} \) do

9.4: If \( u \in C \) Alice calculates the corrected secret measurement angle \( \phi'_u \) with further corrections if \( u \in I \):

\[ \phi'_u \leftarrow (-1)^u \phi'u. \]

9.5: If \( u \in T \) Alice sets \( \phi'_u = 0 \).

9.6: If \( u \in D \) Alice randomly chose \( \phi'_u \).

9.7: Alice samples \( r_u \leftarrow R \{0, 1\} \) and computes \( \delta_u = \phi'_u + \theta'_u + r_u \pi \) and sends \( \delta_u \) to Bob.

9.8: Bob measures the node in \( \{\pm \delta_u\} \) and sends the result \( s_u \) to Alice.

9.9: Alice computes \( s_u \leftarrow s_u \oplus r_u \).

9.10: end for

9.11: end for

9.12: Bob sends remaining qubits to Alice.

9.13: Alice measures the remaining traps.

9.14: If all trap measurement result were correct, Alice performs the final corrections on her output, otherwise she knows that Bob cheated.

For the universality of the protocol, one needs two graph operations. The first is the break-operation, which is implemented by sending a \( Z \)-eigenstates as dots. The second one is the bridge-operation, which is implemented by measuring with an angle, different by \( \pi/2 \) to the preparation angle. Hence, one can measure the dots of \( G' \) that way and gets up to known local rotations \( G \). By that, the dotted triple-graph inherits the universality of the resource state that was the input to create it. For details about these operations, see Ref. \cite{FK17}.

**VBDQC in receive-and-measure** When we apply our equivalence transformation, we do not need to hide the measurement angles. Hence, we can also cancel the offset, except for the input nodes and the corresponding traps and dummies. We had already argued in Section 5.4 that this does not influence equivalence. Similar to the transformation between Protocols 5 and 6, we can not apply the transformation for the input and the output. But as the transformation works qubit-wise, that does not constitute a complication.
**Protocol 10:** VBDQC in receive-and-measure. We define $D(\cdot)$, $C(\cdot)$ and $T(\cdot)$ as private functions from Alice that map from an arbitrary node to the corresponding dummy, computational node or trap respectively. Also we assume, that Alice already used Protocol 7 to create the description of the dotted triple-graph with the deleted nodes.

1. State preparation
   1.1: For all traps Alice initialises $\phi_v = 0$.
   1.2: for $v \in I \cup T(I) \cup D(I)$ do
      1.3: If $v \in D$, Alice prepares $|+\rangle_v$ and sends it to Bob.
      1.4: If $v \in T$, Alice prepares $|+\rangle_v, \theta_v \leftarrow_R A$ and sends it to Bob and sets $\phi_v = \theta_v$.
      1.5: If $v \in C$, Alice applies $P(\theta_v)X^{i_v}, \theta_v \leftarrow_R A, i_v \leftarrow_R \{0,1\}$ to the input $\rho_v$ and sends the qubit to Bob.
   1.6: end for
   1.7: Bob creates for every node in $V_{DT(G)} \setminus (I \cup T(I) \cup D(I))$ a $|+\rangle$ state, and creates with these and the received states the state corresponding to $DT(G)$.

2. Measurement, interaction and output
   2.1: for $v \in V_{DT(G)} \setminus (O \cup T(O) \cup D(O))$ do
      2.2: Bob sends the corresponding qubit to Alice.
      2.3: If $v \in D$ Alice measures the State in $Z$-basis, the result is $r_v$. For all neighbouring qubits she adds to the measurement angle $r_v \pi$.
      2.4: Else, she corrects her angle and gets $\phi'_v$. If $v \in I$ she also corrects $\phi'_v \leftarrow (-1)^v \phi'_v + \theta_v$ and measures the qubit in $\{|\pm \phi'_v\rangle\}$, the result is $r_v$.
   2.5: end for
   2.6: Alice measures the remaining traps.
   2.7: If all trap measurement result were correct, Alice performs the final corrections on her output, otherwise she knows that Bob cheated.