Hindawi
International Journal of Intelligent Systems
Volume 2023, Article ID 2662685, 17 pages
https://doi.org/10.1155/2023/2662685

Research Article

Verifiable Multiparty Delegated Quantum Computation

Qin Li,[1] **Can Wang**,[1] **Jiang Zhu**,[1] **Lingling Xu**,[2] **and Zhiwei Sun**[3,4]

[1]*School of Computer Science, Xiangtan University, Xiangtan, China*
[2]*School of Computer Science and Engineering, South China University of Technology, Guangzhou, China*
[3]*School of Artificial Intelligence, Shenzhen Polytechnic, Shenzhen, China*
[4]*Institute of Applied Artificial Intelligence of the Guangdong-Hong Kong-Macao Greater Bay Area, Shenzhen Polytechnic, Shenzhen, China*

Correspondence should be addressed to Zhiwei Sun; smeker@szpt.edu.cn

Received 28 September 2022; Revised 29 March 2023; Accepted 15 April 2023; Published 15 May 2023

Academic Editor: Subrata Kumar Sarker

Multiparty delegated quantum computation (MDQC) allows multiple clients with limited quantum capability to jointly complete a quantum computational task with the aid of an untrusted quantum server. But in existing MDQC protocols, the verifiability that clients should verify whether the server executed the protocol correctly and gave correct results was not handled. Therefore, in this paper, we improve a typical MDQC protocol to enable clients to verify the correctness of computation by inserting trap qubits and develop a novel method to enforce clients to send qubits honestly while avoiding the positions of trap qubits being leaked to the server. The security and verifiability of the improved MDQC protocol are also analyzed. In addition, a specific example of the proposed verifiable MDQC protocol is given and simulated on IBM's quantum platform.

1. Introduction

Quantum computation has some advantages over classical computation. For example, Shor's algorithm [1] and Grover's algorithm [2] can solve corresponding difficult problems faster than the best classical counterparts. Now, it has been applied to many fields, such as blockchains [3, 4], genetic algorithms [5, 6], block ciphers [7, 8], and homomorphic encryption computation [9, 10]. However, due to technical limitations and expensive costs, it is likely that when quantum computers can be built, only a few institutions possess them and other users need to remotely access them. For classical computation, users can store data or perform calculations with the help of servers through cloud [11, 12]. Similarly, users can also use quantum computers remotely via delegated quantum computation (DQC). In such a situation, users usually want to keep their data private during the process of delegation. Blind quantum computation (BQC) can provide a solution, which is a typical type of DQC that allows a client with limited quantum capability to delegate its computation to a quantum server while still keeping the input, output, and algorithm private.

The first BQC protocol was proposed by employing the circuit model where the client needs the ability to implement the SWAP gate and own quantum memory [13]. Then, Arrighi and Salvail proposed a BQC protocol which requires the client to prepare and measure multiqubit entangled states and is not universal [14]. In 2009, Broadbent et al. proposed the first universal BQC protocol [15], namely, the famous BFK protocol, in which the client only needs to prepare single qubits. In 2013, Morimae and Fujii proposed a BQC protocol for the client who is only able to perform quantum measurements [16], also known as the MF protocol. In addition, in order to make the client as classical as possible, some multiserver BQC protocols were proposed [17, 18]. However, the client cannot check correctness of the results in these BQC protocols. Therefore, various verifiable BQC protocols have been proposed [19–26]. For example, Fitzsimons and Kashefi proposed a universal verifiable BQC protocol, called the FK protocol, where the client can detect any malicious behavior of the server with high probability by using trap qubits [19]. Hayashi and Morimae solved the verification problem for the measurement-only BQC by the stabilizer test [24]. Kashefi and Wallden proposed an optimised resource construction method for verifiable universal BQC protocol [25], where the overhead is linear in the size of the computation. In addition, some BQC protocols also have been put forward by combining other properties, such as BQC with identity authentication [27, 28], device-independence BQC [29–31], and blind oracular quantum computation [32].

But the abovementioned BQC protocols did not take into account the case of multiple clients working together to complete quantum computation. In the network environment, some clients may be required to jointly handle a quantum computational task. Then, secure multiparty quantum computation is necessary in which several players collaboratively compute a joint task on their private data, and they want to keep their own data private. The concept of secure multiparty computation was first proposed by Yao [33] and then extended to multiparty quantum computation by Crépeau et al. [34]. It was proved that the computation is secure as long as the number of malicious participants is less than $n/6$, where n is the total number of participants. In [35], the protocol can tolerate any $\lfloor (n-1)/2 \rfloor$ cheaters among n players. Thereafter, various secure multiparty quantum computation protocols have been proposed [36–39].

In 2017, Kashefi and Pappa proposed a multiparty delegated quantum computation (MDQC) protocol [40] based on the BFK protocol [15]. It allows multiple clients to delegate quantum computation to a powerful quantum server and keep the private data hidden from a dishonest server and a coalition of dishonest clients. But in the MDQC protocol, clients can be tricked by servers since verifiability is not considered. Actually, in the quantum cloud environment, verifiability which means that the client should be able to know whether the quantum server behaves honestly and verify the correctness of results is very important. It can get clients to trust quantum cloud servers and quantum cloud more and ensure that clients cannot be fooled by servers. Especially in the case of multiple parties, each client should be able to fairly verify the correctness of their results. In this paper, we solve the problem in the MDQC protocol [40] where dummy qubits to be used may cause the leakage of positions of trap qubits and propose a verifiable MDQC protocol in which clients can verify the correctness of results with high probability.

This paper is organized as follows: Section 2 introduces the model of measurement-based quantum computation (MBQC) and the dotted triple graph DT (G). In Section 3, a typical MDQC protocol is reviewed. Section 4 proposes a verifiable MDQC protocol. The analysis about the security and verifiability of the proposed protocol is carried out in Section 5. Section 6 compares the proposed protocol with other similar BQC protocols. An example of the verifiable MDQC protocol for three clients is given and simulated on IBM's quantum platform in Section 7. The last section makes a conclusion of this paper.

2. Preliminaries

In this section, we give a brief introduction to the MBQC model [41] that has the same computational power as the quantum circuit model [42] and the DT (G) state as a typical quantum resource state.

2.1. The MBQC Model. The MBQC model which is derived from the gate teleportation principle is often used to realize DQC [41]. In the MBQC model, the computation is implemented by creating a highly entangled quantum state, such as a cluster state [43] or a brickwork state [15], and then performing single-qubit measurements on the qubits in that state. This particular entangled state is called the graph state since it can be represented by a graph. The vertices of the graph represent qubits, and the edges represent CZ gates applied to the corresponding vertices.

More specifically, the MBQC model is defined by a graph G (V and E) with the vertex set V, the edge set E, the input set $I \subseteq V$, the output set $O \subseteq V$, the set $I^c = V/I$ for noninput vertices, a sequence of measurement angles $\{\phi_i\}$ for nonoutput vertices in the set $O^c = V \backslash O$, and a flow function $f\colon O^c \longrightarrow I^c$. During calculation, these measurement angles ϕ_i need to be updated depending on the measurement results of previously measured qubits due to the probabilistic nature of the measurement. The flow function f determines the dependency structure and measurement order of qubits [44], such that a qubit i is X-dependent on the qubit $f^{-1}(i)$ and Z-dependent on the qubit j in which $i \in N_G(f(j))$, where $N_G(k)$ represents the set of neighbours of the vertex k in the graph G. The X- and Z-dependency sets of the qubit i are expressed as D_X^i and D_Z^i, respectively. During the execution of the MBQC model, the updated measurement angle of the qubit i is $\phi_i' = (-1)^{s_X^i}\phi_i + s_Z^i \pi$, where $s_X^i = \oplus_{j \in D_X^i} s_j$, $s_Z^i = \oplus_{j \in D_Z^i} s_j$, and s_j represents the measurement result of the qubit j. After measuring all nonoutput qubits in O^c, the result of the computation is obtained by applying the Pauli correction $Z^{s_Z^i} X^{s_X^i}$ on each output qubit i in O.

2.2. The DT (G) State. In 2017, Kashefi and Wallden used the DT (G) construction to simplify the required resources for verification [25]. In the protocol to be proposed, the DT (G) state is used as the graph state of the MBQC model so that clients can verify the correctness of the calculated results. DT (G) is briefly introduced as follows.

DT (G) is constructed from a base graph G that has vertices $v \in V$ and edges $e \in E$. Each vertex v_i in the graph G is replaced with a set of three vertices $P_{v_i} = \{p_{v_i}(1), p_{v_i}(2), p_{v_i}(3)\}$ called primary vertices, and each edge $e(v_i, v_j)$ that connects the vertices v_i and v_j is replaced with a set of nine edges $E_{(v_i, v_j)}$ that connect each of the vertices in P_{v_i} with each of the vertices in P_{v_j}. The resulting graph after performing the previous steps on the graph G is called the triple graph T (G). The operation of replacing each edge in a graph G with a new vertex connected to the two vertices originally joined by that edge is defined as the dotting operation, and the result graph of this operation on a graph G is called the dotted base graph D (G). The dotting operation is then performed on the graph T (G) to obtain the dotted triple graph DT (G), and the new vertices are called added vertices. The whole process of generating DT (G) used for quantum computation is shown in Figure 1.

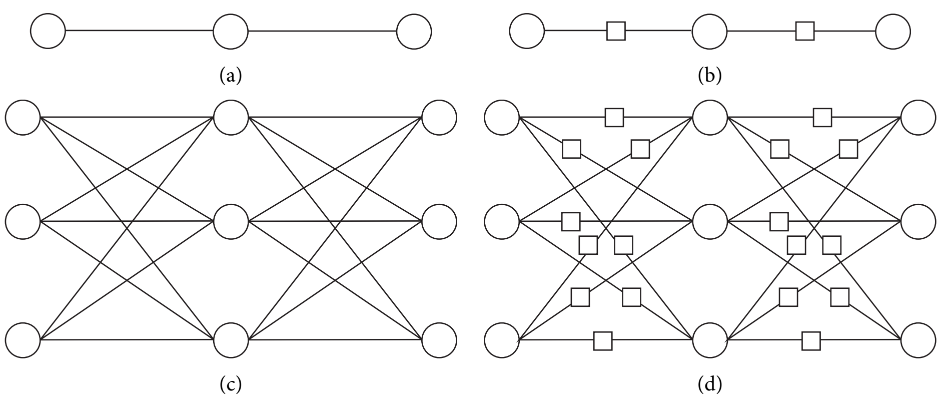


Figure 1: (a) A base graph G. (b) A dotted base graph D (G). (c) A triple graph T (G), where all primary vertices are denoted as circles. (d) A dotted triple graph DT (G), where all added vertices are denoted as squares.

DT (G) is then reduced to three copies of D (G) by using break operation. One of three dotted base graphs is randomly selected for the computation, and the other two copies used as traps are called the trap subgraph. The break operation on a vertex v is equivalent to the operation which removes the vertex v and any adjacent edges to v. It can be carried out by using dummy qubits in a graph state. The dummy qubit represents a qubit in the state $|0\rangle$ or $|1\rangle$; thus, it does not entangle with other qubits in the graph state. The process for choosing which vertices used for traps in DT (G) to form certain subgraph is called trap-coloring. This is carried out by first choosing randomly for each primary set P_{v_i} in DT (G) one white, one green, and one black vertices. Then, the added vertices that connect primary vertices of different colors are colored in red and that the added vertices that connect primary vertices of the same color are also colored in that color. After performing break operations on all red vertices, three copies of D (G) are obtained. Figure 2 illustrates the trap-coloring of DT (G) and three subgraphs of DT (G). Note that, in DT (G), trap qubits in states $|+_\theta\rangle$ should be included for verification. If the measurement angles of some trap qubits are wrong, the client can determine whether the server is honest according to the measurement results of these trap qubits. In addition, it is also necessary to add dummy qubits $|0\rangle$ or $|1\rangle$ to separate trap qubits from computation qubits and other trap qubits in the graph state so as to avoid entanglement among these qubits and influence on the measurement results of them. As shown in Figure 2(a), the red vertices are dummy qubits used to separate trap qubits and computation qubits. In Figure 2(b), assuming that the white subgraph is a trap subgraph and the circle vertices are trap qubits, the square vertices are dummy qubits and are used to isolate trap qubits. Especially, the trap qubits in DT (G) cannot all be the primary vertices (circular vertices) of the graph; otherwise, the server may perform malicious operations only on the added vertices (square vertices) instead of circular vertices, and the client may not be able to detect malicious behavior through trap qubits. Therefore, in two trap subgraphs, if the primary vertices of one subgraph are used as trap qubits, then the other subgraph should use the added vertices as trap qubits.

3. The Review of the MDQC Protocol

The MDQC protocol proposed by Kashefi and Pappa [40] is briefly reviewed in this section. In this protocol, n clients $C_1, C_2, \ldots, C_n$ delegate a quantum computational task to a powerful but untrusted quantum server and still keep their inputs, outputs, and the performed computation secret. For simplicity, this protocol only considers the case where each client has one input qubit and one output qubit, and it assumes that these clients have secure access to classical multiparty computation that is secure against a dishonest majority [45, 46].

The protocol is divided into the state preparation phase and the computation phase. In the state preparation phase, in order to provide security against dishonest participants without the need for quantum communication among clients, a process named "remote state preparation" is proposed. The process has two different algorithms for input qubits in I and noninput/nonoutput qubits in O^c/I. For input qubits, a client sends an encrypted input qubit to the server and other clients send qubits in states $|+_\theta\rangle = 1/\sqrt{2}(|0\rangle + e^{i\theta}|1\rangle)$ to the server, where θ is a rotated angle. Then, the server performs Algorithm 1 on these qubits. For noninput/nonoutput qubits, all clients send qubits in states $|+_\theta\rangle$ to the server who will perform Algorithm 2 related to them. The steps of Algorithms 1 and 2 are given as follows:

By using the method for remote state preparation, clients can remotely prepare quantum states that are encrypted by all clients without quantum communication with each other. However, it is possible that some clients send qubits at a different rotate angle instead of the expected angle and that it will result in an incorrect quantum state. Therefore, Algorithm 3 is proposed to enforce clients to honestly send qubits. In addition, Algorithm 3 requires clients to secretly share their classical angle values via verifiable secret sharing (VSS) schemes [47–49], which can be viewed as a classical secure multiparty computation method. The steps of Algorithm 3 are given as follows:

The previous three algorithms can ensure that clients successfully prepare quantum states during the state preparation phase, and the subsequent steps are to entangle these qubits to form a graph state and measure these qubits to

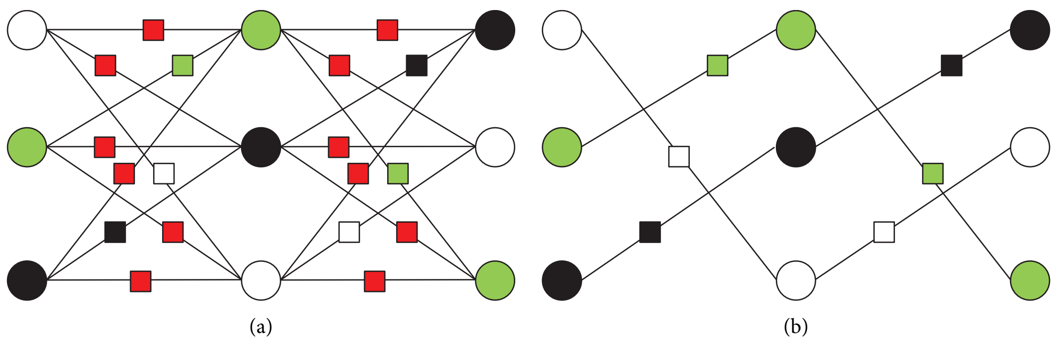


Figure 2: (a) Trap-coloring of DT (G). (b) Three copies of D (G). The green subgraph is used for computations. In the white subgraph, the circle vertices are trap qubits and the square vertices are dummy qubits. In the black subgraph, the square vertices are trap qubits and the circle vertices are dummy qubits.

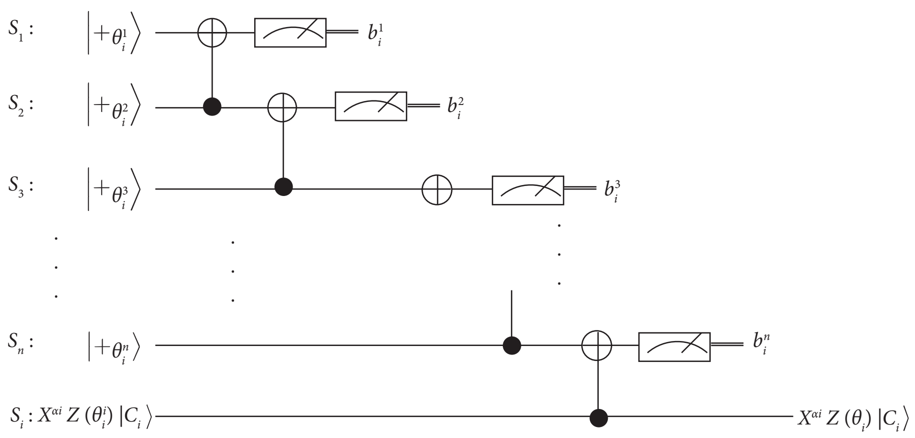


Figure 3: The circuit used for remote state preparation of the quantum input. The resulting state of this circuit is $X^{\alpha_i}Z(\theta_i)|C_i\rangle$, where $\theta_i = \theta_i^i + \sum_{t=1,t\neq i}^{n}(-1)^{\oplus_{k=1}^{n} b_i^k + \alpha_i}\theta_i^t$.

```
A1: The server receives the encrypted input $X^{\alpha_i}Z(\theta_i^i)|C_i\rangle$ from the client $C_i$ and $n-1$ qubits $|+_{\theta_i^j}\rangle$ from other clients $C_j\,(1 \leq j \leq n, j \neq i)$. It also stores the qubit sent by the client $C_k$ in the quantum register $S_k$, where $1 \leq k \leq n$.
A2: For $k = 1, \ldots, n-1$
(1)     The server performs the CNOT gate on $S_k \otimes S_{k+1}$. Note that if $k+1 = i$ and $i \neq n$, the server performs the CNOT gate on $S_k \otimes S_{k+2}$.
(2)     The server measures the qubit in $S_k$ to get the outcome $b_i^k$.
    A3: The server performs the CNOT gate on $S_n \otimes S_i$ for $i \neq n$ and on $S_{n-1} \otimes S_i$ for $i = n$. Then, it measures the qubit in $S_n$ (or $S_{n-1}$) to get the outcome $b_i^n$ (or $b_i^{n-1}$).
    A4: The server gets the qubit $X^{\alpha_i}Z(\theta_i)|C_i\rangle$ that all clients encrypt together and the outcome vector $b_i = (b_i^1, \ldots, b_i^n)$, where $\theta_i = \theta_i^i + \sum_{t=1,t\neq i}^{n}(-1)^{\oplus_{k=1}^{n} b_{k=1}^{n} + \alpha_i}\theta_i^t$. This process is shown in Figure 3.
```

Algorithm 1: Remote state preparation of the quantum input.

```
B1: The server receives $n$ qubits $|+_{\theta_i^k}\rangle$ from clients $C_k$ $(1 \leq k \leq n)$. Then, it stores each qubit received from the client $C_k$ in the quantum register $S_k$.
B2: For $k = 1, \ldots, n-1$, the server applies the CNOT gate on $S_k \otimes S_{k+1}$ and measures the qubit in $S_k$ to get the outcome $b_i^k$.
B3: The server gets a noninput/nonoutput qubit $|+_{\theta_i}\rangle$ and the outcome vector $b_i = (b_i^1, \ldots, b_i^n)$, where $\theta_i = \theta_i^n + \sum_{t=1}^{n-1}(-1)^{\oplus_{k=t}^{n-1} b_i^k}\theta_i^t$. This process is shown in Figure 4.
```

Algorithm 2: Remote state preparation of noninput/nonoutput qubits.

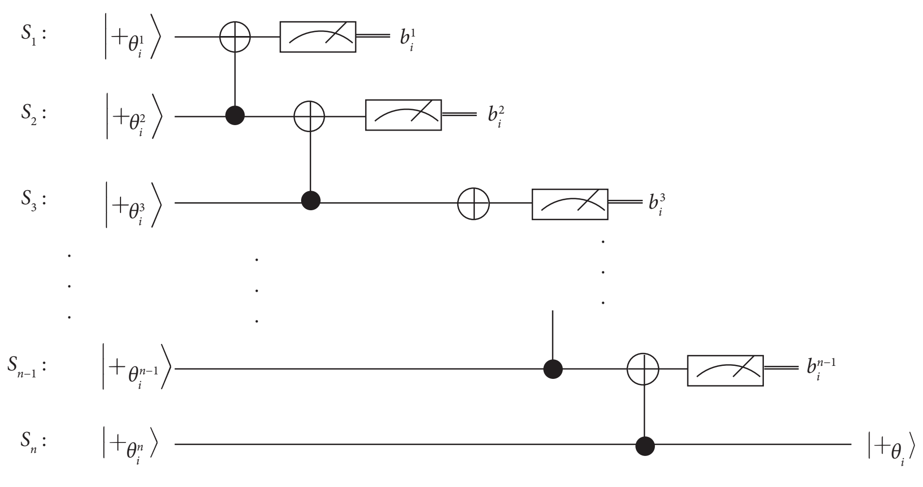


Figure 4: Remote state preparation for noninput/nonoutput quantum qubits. The resulting state of this circuit is $|+_{\theta_i}\rangle$, where $\theta_i = \theta_i^n + \sum_{t=1}^{n-1} (-1)^{\oplus_{k=t}^{n-1} b_i^k} \theta_i^t$.

complete the task in the computation phase. The specific steps of the MDQC protocol are given as follows.

3.1. The State Preparation Phase

D1: For each input qubit $i \in I$

(1) The client C_i that holds an input qubit sends the encrypted quantum input $X^{\alpha_i} Z(\theta_i^i)|C_i\rangle$ to the server and shares the values of α_i and θ_i^i with other clients via a VSS scheme, where $\alpha_i \in \{0,1\}$ and $\theta_i^i \in \{l\pi/4 | l = 0, 1, \ldots, 7\}$.
(2) Other clients $C_j\,(1 \le j \le n, j \ne i)$ perform Algorithm 3. If these $n-1$ clients pass the test, the server has $n-1$ qubits $|+_{\theta_i^j}\rangle$. The server performs Algorithm 1 on the input qubit $X^{\alpha_i} Z(\theta_i^i)|C_i\rangle$ and $n-1$ qubits $|+_{\theta_i^j}\rangle$.
(3) The server gets the outcome vector b_i and the qubit $X^{\alpha_i} Z(\theta_i)|C_i\rangle$ and then announces the vector b_i to all clients, where $\theta_i = \theta_i^i + \sum_{t=1, t\ne i}^{n} (-1)^{\oplus_{k=t}^{n} b_i^k + \alpha_i} \theta_i^t$.

D2: For each nonoutput/noninput qubit $i \in O^c/I$

(1) All clients perform Algorithm 3. If all clients pass the test, the server has n qubits $|+_{\theta_k^j}\rangle\,(1 \le j \le n)$.
(2) After running Algorithm 2 on n qubits, the server obtains the qubit $|+_{\theta_i}\rangle$ and the outcome vector b_i, where $\theta_i = \theta_i^n + \sum_{t=1}^{n-1} (-1)^{\oplus_{k=t}^{n-1} b_i^k} \theta_i^t$. Then, it announces the vector b_i to all clients.

D3: For each output qubit, the server prepares a qubit $|+\rangle$.

D4: The server creates an entangled brickwork state by applying CZ gates on corresponding qubits.

3.2. The Computation Phase

D5: For each nonoutput qubit $i \in O^c$

(1) Each client $C_k\,(1 \le k \le n)$ chooses random $r_i^k \in \{0,1\}$ and shares the value r_i^k with other clients via a VSS scheme. Then, by using classical multiparty computation, clients compute the measurement angle $\delta_i = \phi_i' + \pi r_i + \theta_i$ of the qubit i and send it to the server, where $r_i = \oplus_{k=1}^{n} r_i^k$, $\phi_i' = (-1)^{\alpha_i + s_X^i} \phi_i + s_Z^i \pi + \alpha_{f^{-1}(i)} \pi$ is the updated measurement angle, and $\alpha_i = 0$ if i is a noninput qubit.
(2) The server receives the measurement angle δ_i and measures the qubit i on the basis $\{|+_{\delta_i}\rangle, |-_{\delta_i}\rangle\}$. Then, the server transmits the measurement result t_i to all clients.
(3) The clients compute the actual measurement result $s_i = t_i \oplus r_i$.

D6: For each output qubit $i \in O$, the server sends the output qubit i to the corresponding client C_i. Then, all clients jointly compute the value of s_X^i and s_Z^i and send them to the client C_i. Then, C_i applies $Z^{s_Z^i} X^{s_X^i}$ on the output qubit i to get the actual quantum output.

4. The Proposed Verifiable MDQC Protocol

In this section, we construct a verifiable MDQC protocol based on the reviewed MDQC protocol. In the protocol to be presented, clients can not only delegate the quantum computational task to the server while keeping their data private but also verify the correctness of the results by using DT (G) states and inserting trap qubits.

In the reviewed MDQC protocol, it is not possible to directly use trap qubits and dummy qubits due to the following facts: In Algorithm 3, in order to enforce the clients to send qubits honestly, the server needs to measure the qubits sent by clients on XY-plane bases. If the dummy qubit is used, the server should measure these dummy qubits on the Z basis. Different measurement bases for dummy qubits and nondummy qubits leak their positions to the server. From the positions of dummy qubits, the server can deduce the positions of trap qubits. Therefore, it is difficult for clients to detect whether the server behaves maliciously by measuring the trap qubits in the reviewed MDQC protocol.

C1: The client C_k sends m qubits $|+_{\theta_k^j}\rangle$ $(1 \leq j \leq m)$ to the server and shares the angle values θ_k^j with other clients via a classical VSS scheme.
C2: The server randomly chooses $m-1$ qubits from m qubits sent by the client C_k and requests the shared angle values of $m-1$ qubits from all clients.
C3: The server reconstructs these angle values of $m-1$ qubits and measures $m-1$ qubits in corresponding basis. If all results of the measurements are correct as expected, then the remaining qubit is correct with high probability.

Algorithm 3: The specific steps for enforcing clients to send qubits honestly.

Thus, we propose Algorithm 4 where clients should send qubits honestly while ensuring that the positions of dummy qubits and trap qubits are not leaked. The specific steps of Algorithm 4 are given as follows:

In Algorithm 4, the client needs to send $2m+2$ qubits to the server. The server randomly chooses m pairs of qubits from $2m+2$ qubits and verifies the correctness of each pair of qubits with a new method. Two cases of this method are shown in Figures 5 and 6, respectively. In the case of nondummy qubits, we suppose a pair of qubits is in states $|+_{\theta_1+\alpha_1\pi}\rangle$ and $|+_{\theta_2+\alpha_2\pi}\rangle$, where $\alpha \in \{0,1\}$. The CNOT gate is applied to the two qubits to entangle them, and the measurement result s_1 is obtained by measuring the target qubit on the Z basis. According to the value of s_1, the measurement angle $\theta_2 + (-1)^{s_1}\theta_1$ of the control qubit is calculated. The result s_2 is obtained by measuring the control qubit on the basis $|\pm_{\theta_2+(-1)^{s_1}\theta_1}\rangle$. If $s_2 = \alpha_1 \oplus \alpha_2$, the pair of nondummy qubits are correct with high probability. In the case of dummy qubits, we suppose a pair of qubits is in states $|\alpha_1\rangle$ and $|\alpha_2\rangle$, where $\alpha_1, \alpha_2 \in \{0,1\}$. The server applies the CNOT gate to two qubits and measures the target qubit on the Z basis to get the result s_1. If $s_1 = \alpha_1 \oplus \alpha_2$, the pair of dummy qubits is correct with high probability. In addition, in order to be consistent with the operations of nondummy qubits, in the case of dummy qubits, control qubits also need to be measured on XY-plane bases. Furthermore, in the case of dummy qubits, if the states of $2m+2$ qubits are completely different from what the client promises, i.e., the client originally promises to send $|0\rangle$ (or $|1\rangle$) but actually sends $|1\rangle$ (or $|0\rangle$), then the given method cannot detect errors. The server can avoid this case by randomly measuring one of the two remaining qubits on the Z basis. If the measurement result of the qubit measured on the Z basis is correct as expected and m pairs of qubits pass the test, the state of the remaining qubit is real with high probability. In this way, Algorithm 4 can avoid the positions of dummy qubits and trap qubits being leaked and enforce clients to send qubits honestly.

Based on Algorithm 4, we propose a verifiable MDQC protocol. In the protocol to be proposed, we use the DT (G) state as the resources state. The protocol consists of two phases, namely, the state preparation phase and the computation phase. In the state preparation phase, all clients generate the dotted triple graph DT (G) according to the base graph G and choose a kind of trap-coloring. Since the input qubits and noninput qubits are handled differently in the process of remote state preparation, the server knows where input qubits are and each client should send three input qubits to the server, namely, one real input qubit, one trap-input qubit, and one dummy-input qubit. The client is able to know if the server is correctly performing the measurement of the input qubit via the measurement result of the trap-input qubit. In the computation phase, clients update the measurement angles of nonoutput qubits by using classical multiparty computation and send these measurement angles to the server. Then, clients can determine whether the server performed the measurements correctly based on the measurement results of trap qubits. After measuring all nonoutput qubits, the server returns three output qubits to each client, including one trap-output qubit, one dummy-output qubit, and one real output qubit. Each client can know if the server honestly returns the results by measuring the trap-output qubit. The specific verifiable MDQC protocol is described in Protocol 1.

5. Security Analysis and Verifiability

In this section, we analyze the security of Algorithm 4 and the proposed verifiable MDQC protocol if there is a dishonest server and a coalition of dishonest clients and the verifiability of the proposed protocol. The proposed MDQC protocol is considered to be secure if it is blind which means that the input, output, and even the algorithm of honest clients cannot be learned by quantum servers and the input and output of each client cannot be known by other malicious clients. For the verifiability of the proposed protocol, the probability that clients accept the incorrect result calculated by the server is analyzed.

Theorem 1. *In Algorithm 4, if the server is malicious, it cannot know the positions of trap qubits and the state of qubits.*

Proof. In Algorithm 4, clients send $2m+2$ qubits in states $\{|0\rangle, |1\rangle\}$ or $2m+2$ qubits in states $|+_{\theta_i}\rangle$ to the server. The state of the qubit that the server received from clients is

$$\begin{aligned}\rho &= \frac{1}{10}\sum_{\theta_i \in \{\pi l/4\}_{l=0}^{7}} \left(\left|+_{\theta_i}\right\rangle\left\langle+_{\theta_i}\right|\right) + |1\rangle\langle 1| + |0\rangle\langle 0| \\ &= \frac{1}{10}\left(|+\rangle\langle+| \ldots + |+_{7\pi/4}\rangle\langle+_{7\pi/4}| + |0\rangle\langle 0| + |1\rangle\langle 1|\right) \\ &= \frac{I}{2}.\end{aligned} \tag{1}$$

For nondummy qubits

E1: The client C_k sends $2m+2$ qubits $|\varphi_k^j\rangle = 1/\sqrt{2}(|0\rangle + (-1)^{a_j} e^{i\theta_k^j} 1\rangle)$ $(1 \le j \le 2m+2)$ to the server and shares the values of θ_k^j and a_j with other clients by using a VSS scheme.

E2: The server randomly chooses $2m$ qubits from $2m+2$ qubits sent by the client and then divides $2m$ qubits into m pairs $\{(|\varphi_k^{q_1}\rangle, |\varphi_k^{p_1}\rangle), \ldots, (|\varphi_k^{q_m}\rangle, |\varphi_k^{p_m}\rangle)\}$.

E3: For each pair of qubits $(|\varphi_k^{q_l}\rangle, |\varphi_k^{p_l}\rangle)$

(1) The server applies the CNOT gate to these two qubits. Then, the server measures the target qubit $|\varphi_k^{q_l}\rangle$ on the Z basis and sends the measurement result b_{q_l} and the indexes q_l, p_l of two qubits to all clients.
(2) The clients send the shared values of $\theta_k^{q_l}$ and $\theta_k^{p_l}$ to the server. Then, the server reconstructs the angle values $\theta_k^{q_l}$ and $\theta_k^{p_l}$ and computes the measurement angle $\delta_k^{p_l} = \theta_k^{p_l} + (-1)^{b_{q_l}} \theta_k^{q_l}$ of the control qubit. The server measures the control qubit $|\varphi_k^{p_l}\rangle$ on the basis $\{|+_{\delta_k^{p_l}}\rangle, |-\delta_k^{p_l}\rangle\}$ and sends the measurement result b_{p_l} to all clients.
(3) The clients determine whether the result b_{p_l} is correct. If $b_{p_l} = \alpha_{p_l} \oplus \alpha_{q_l}$, the result is correct with high probability, so the states of two qubits also are correct with high probability.Algorithm 4: The specific steps used to enforce clients to send qubits honestly while ensuring that the positions of trap qubits and dummy qubits in the graph state are not leaked to the server.

E4: The server randomly chooses one of the remaining two qubits and measures the qubit on the Z basis to get the result b_z. Then, it sends b_z and the index z to all clients.

E5: If the quantum states of m pairs of qubits selected by the server in step E2 are all considered to be correct in step E3, then the remaining qubits are also correct with high probability.

For dummy qubits

E1: The client C_k sends $2m+2$ qubits$|\varphi_k^j\rangle = |a_j\rangle$ $(a_j \in \{0,1\}, 1 \le j \le 2m+2)$ to the server. Then, it randomly chooses $2m+2$ arbitrary angle values $\{\theta_k^j\}_{j=1}^{2m+2}$ and shares the values of a_j and θ_k^j with other clients by using a VSS scheme.

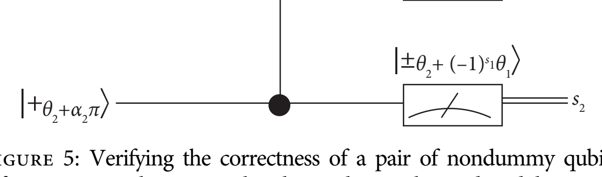
E2: The server randomly chooses $2m$ qubits from the qubits sent by the client and then divides $2m$ qubits into m pairs $\{(|\varphi_k^{q_1}\rangle, |\varphi_k^{p_1}\rangle), \ldots, (|\varphi_k^{q_m}\rangle, |\varphi_k^{p_m}\rangle)\}$.

E3: For each pair of qubits $(|\varphi_k^{q_l}\rangle, |\varphi_k^{p_l}\rangle)$

(1) The server applies the CNOT gate to these two qubits. Then, the server measures the target qubit $|\varphi_k^{q_l}\rangle$ on the Z basis and sends the result b_{q_l} and the indexes q_l, p_l to all clients.
(2) The clients send the shared values of $\theta_k^{q_l}$ and $\theta_k^{p_l}$ to the server. Then, the server reconstructs the values and computes the measurement angle $\delta_k^{p_l}$. It measures the control qubit $|\varphi_k^{p_l}\rangle$ on the basis $\{|+_{\delta_k^{p_l}}\rangle, |-_{\delta_k^{p_l}}\rangle\}$ and sends the measurement result b_{p_l} to all clients.
(3) The clients determine whether the result b_{q_l} is correct. If $b_{q_l} = \alpha_{p_l} \oplus \alpha_{q_l}$, the result is considered to be right with high probability, so the states of two qubits also are correct with high probability.

E4: The server randomly chooses one of the remaining two qubits and measures the qubit on the Z basis to get the result b_z. Then, it sends b_z and the index z to all clients.

E5: If the quantum states of m pairs of qubits selected by the server in step E2 are all considered to be correct in step E3 and $b_z = \alpha_z$, then the remaining qubits are also correct with high probability.



$|+_{\theta_1+\alpha_1\pi}\rangle$
$|+_{\theta_2+\alpha_2\pi}\rangle$
Z
s_1
$|\pm\theta_2 + (-1)^{s_1}\theta_1\rangle$
s_2

FIGURE 5: Verifying the correctness of a pair of nondummy qubits. After measuring the target qubit, the result s_1 is obtained, and the state of control qubits is $|+_{\theta_2+\alpha_2\pi+(-1)^{s_1}(\theta_1+\alpha_1\pi)}\rangle$. Then, the control qubit is measured on the basis $|\pm_{\theta_2+(-1)^{s_1}\theta_1}\rangle$ to get the result $s_2 = \alpha_1 \oplus \alpha_2$.

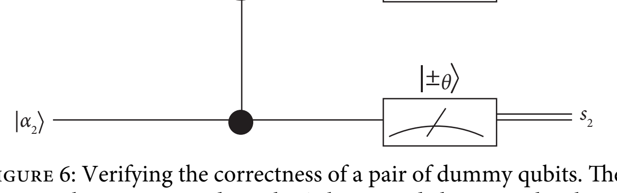


FIGURE 6: Verifying the correctness of a pair of dummy qubits. The target qubit is measured on the Z basis, and the control qubit is measured on the basis $|\pm_\theta\rangle$ where θ is any angle. The measurement result of the target qubit is $s_1 = \alpha_1 \oplus \alpha_2$.

It is a maximally mixed state so that the server cannot know the state of each qubit sent by clients.

Because $\langle 0|+_\theta\rangle \ne 0$ and $\langle 1|+_\theta\rangle \ne 0$, $\{|0\rangle, |1\rangle\}$ and $|+_\theta\rangle$ are not orthogonal. According to the indistinguishability of nonorthogonal states, the server cannot distinguish dummy qubits from nondummy qubits based on the state of the qubits. In addition, the operation process in Algorithm 4 for making clients to send dummy qubits or nondummy qubits honestly is the same for the server, since it always measures the target qubit on the Z basis and measures the control qubit on the XY-plane basis for each pair of qubit. So the server cannot distinguish between dummy qubits and nondummy qubits, and hence, the server does not know

Input: Each client has three input qubits, one real input qubit, one trap-input qubit, and one dummy-input qubit.
Output: Each client has three output qubits, one real output qubit, one trap-output qubit, and one dummy-output qubit.
The state preparation phase
(1) The clients generate the dotted triple graph DT (G) in terms of a base graph G and choose a kind of trap-coloring of DT (G). Then, DT (G) is sent to the server.
For each input qubit $i \in I$
(1) The client C_i that holds an input qubit sends the encrypted quantum input $X^{\alpha_i} Z(\theta_i^i) |C_i\rangle$ to the server and shares the values of α_i and θ_i^i with other clients. If i is a trap qubit, $|C_i\rangle = |+\rangle$, and if i is a dummy qubit, $|C_i\rangle = |0\rangle$ or $|1\rangle$.
(2) Other $n-1$ clients run Algorithm 4. If $n-1$ clients pass the test, the server has $n-1$ qubits.
(3) The server performs Algorithm 1 to get the outcome vector b_i and an input qubit $X^{\alpha_i} Z(\theta_i) |C_i\rangle$ that are encrypted by all clients. Then, it announces the outcome vector b_i.
For each noninput qubit $i \in I^c$
(1) All clients run Algorithm 4. If all clients pass the test, the server has n qubits.
(2) The server performs Algorithm 2 similar to the reviewed MDQC protocol to get the outcome vector b_i. If i is a nondummy qubit, the server gets the resulting state $|+_{\theta_i}\rangle$. If i is a dummy qubit, the resulting state is $|d_i\rangle$, where $d_i \in \{0,1\}$. Then, the server announces the outcome vector b_i.
Graph state: the server entangles these qubits generated during the state preparation phase to form a DT (G) state.
The computation phase
For each nonoutput qubit $i \in O^c$
(1) Each client C_k $(1 \le k \le n)$ randomly chooses $r_i^k \in \{0,1\}$ and shares the value with other clients via a VSS scheme. Then, by using classical multiparty computation, the clients compute the measurement angle $\delta_i = \phi_i' + \pi r_i + \theta_i + \sum_{j \in D_i} d_j \pi$ of the qubit i and send it to the server, where $r_i = \oplus_{k=1}^{n} r_i^k$, ϕ_i' is the updated measurement angle in MBQC, and D_i is the set of dummy qubits that are neighbours of the qubit i. If i is a trap qubit, $\phi_i' = 0$.
(2) The server measures the qubit i on the basis $\{|+_{\delta_i}\rangle, |-_{\delta_i}\rangle\}$ and sends the measurement result t_i to all clients. Then, the clients get the actual measurement result $s_i = t_i \oplus r_i$.
(3) If qubit i is a trap qubit, the clients check whether the measurement result t_i is right. If $t_i \neq r_i$, the clients reject it and terminate the protocol.
For each output qubit $i \in O$
(1) The server sends the output qubit i to the corresponding client C_i.
(2) If qubit i is a trap-output qubit, all clients compute the measurement angle θ_i for the trap qubit by using classical multiparty computation. Then, C_i measures the qubit on the basis $\{|+_{\theta_i}\rangle, |-_{\theta_i}\rangle\}$ and determines whether the measurement result is correct.
(3) If qubit i is a real output qubit, all clients compute the values of s_Z^i, s_X^i, and θ_i and send them to C_i. The client C_i applies $Z^{s_Z^i} X^{s_X^i} Z(-\theta_i)$ to the qubit i and gets the actual quantum output.

Protocol 1: The verifiable MDQC protocol.

which are dummy qubits and also cannot deduce the positions of trap qubits. □

Theorem 2. *The proposed MDQC protocol is secure against a malicious server and against a coalition of malicious clients.*

Proof. If the protocol is secure against a malicious server and against a coalition of malicious clients, each client's input and output information is not leaked to other participants and the server also does not know which algorithm is being performed.

Since the input qubit of each client is encrypted with the quantum one-time pad and the server does not know the encryption key, the malicious server cannot obtain the input information of clients. Similarly, the output qubit is also encrypted by the quantum one-time pad $X^{s_X^i} Z^{s_Z^i}$. Since the encryption keys s_X^i, s_Z^i are related to previous actual measurement outcomes $s_i = t_i \oplus r_i$ where the server does not know r_i, the keys s_X^i, s_Z^i are unknown to the server. So the server also cannot obtain the output information of clients.

In the state preparation phase, the quantum state for each noninput qubit received by the server is

$$\frac{1}{8} \sum_{\theta_i \in \{\pi l/4\}_{l=0}^{7}} \left(|+_{\theta_i}\rangle\langle +_{\theta_i}| \right) = \frac{I}{2}, \tag{2}$$

if the qubit is not a dummy qubit or

$$\frac{1}{2} \left(|0\rangle\langle 0| + |1\rangle\langle 1| \right) = \frac{I}{2}, \tag{3}$$

if the qubit is a dummy qubit. Hence, the qubits obtained by the server are always in maximum mixed states, and the server does not know the states of these qubits it received. Furthermore, during Algorithms 1 and 2, the server only knows the measurement result b_i^k and does not know the angle θ_i^k of these qubits. Due to $\theta_i = \theta_i^i + \sum_{t=1,t\neq i}^{n} (-1)^{\oplus_{k=t}^{n-1} b_i^k + \alpha_i} \theta_i^t$ in Algorithm 1 and $\theta_i = \theta_i^n + \sum_{t=1}^{n-1} (-1)^{\oplus_{k=t}^{n-1} b_i^k} \theta_i^t$ in Algorithm 2, the server ultimately cannot learn the states of the qubits used in calculation.

In addition, another kind of information that the server receives from clients is the measurement angles of the qubits in the graph state. In the MBQC model, the actual measurement angle of a nonoutput qubit in the graph state should be $\phi'(i)$ after correcting, but in the protocol, the measurement angle sent by the clients to the server is $\delta_i = \phi'(i) + \pi r_i + \theta_i + \sum_{j \in D_i} d_j \pi$. Obviously, the measurement angles δ_i are related to $r_i = \oplus_{k=1}^{n} r_i^k$, θ_i, and d_j. However, r_i^k is randomly chosen by each client, θ_i is composed of θ_i^k randomly chosen by each client, and d_j represents the state of the adjacent dummy qubit, and they are unknown to the server. Therefore, the server cannot derive the actual measurement angle $\phi'(i)$ from the measurement angle δ_i which implies that the server cannot know the real computation which the client performed.

Then, we analyze the case of coalition of malicious clients in the proposed MDQC protocol. Clients never receive any quantum information from other clients, and the only information they know is the shared values about the rotation angles θ of quantum states $|+_\theta\rangle$ and partial information related to the encryption keys of quantum inputs shared by other clients through the classical VSS scheme. Since in the classical VSS scheme, the reconstruction of the original value needs to combine the shared values of most participants, and the coalition of malicious clients is difficult to obtain the rotation angles of the quantum states and the encryption keys of quantum input. Hence, a coalition of malicious clients cannot obtain the input and output information of other clients if the used classical multiparty computation and VSS scheme are secure. □

Theorem 3. *The proposed MDQC protocol is* (8/9) *verifiable.*

Proof. The deviation between the correct result and the incorrect result can be defined by Kraus operators [25], which are expressed as linear combinations of Pauli operators $\sigma_i \in \{I, X, Y, Z\}$. Pauli σ_i acting on the qubit k is denoted as $\sigma_{i|k}$, and if the qubit k is an output qubit, $\sigma_{i|k} \in \{X, Y, Z\}$; otherwise, $\sigma_{i|k} \in \{X, Y\}$.

We denote P_{inc} as the maximum probability that clients accept an incorrect computation result. If clients accept the incorrect computation result, it means that the attack did not disturb any trap. Then, P_{inc} is determined by the probability of the attack acting on the trap qubit, different σ_i acting on the trap qubit, and the values of θ and r related to the trap qubit. Let T represents the positions of trap qubits and $p(T)$ denotes the probability that the qubits in positions $T$are trap qubits. $p(\theta_t)$ and $p(r_t)$ represent the probabilities of choosing θ_t and r_t, respectively. According to [25], the following formula can be obtained:

$$P_{\text{inc}} \le \max_{i\in E_i} \sum_T p(T) \prod_{t\in T} \left(\sum_{\theta_t, r_t} p(\theta_t) p(r_t) \left(\langle \eta_t^{v_T} | \sigma_{i|t} | \eta_t^{v_T} \rangle \right)^2 \right), \tag{4}$$

where $|\eta_{t_\beta}^{v_T}\rangle = |+_{\theta_{t_\beta}}\rangle$if the qubit t is an output qubit; otherwise,$|\eta_{t_\beta}^{v_T}\rangle = |r_{t_\beta}\rangle$.

To maximize the value of the probability P_{inc}, the best strategy is to make only a single attack. Assume that the position of the single attack is β and F_β represents the location of qubit β, $F_\beta = P_v^O$ if β belongs to the output qubit, and $F_\beta = P_v^{NO}$ if βbelongs to the nonoutput qubit. According to $\sum_T p(T) = \sum_{t_\beta\in F_\beta} \sum_{t\notin F_\beta} p(T) = \sum_{t_\beta\in F_\beta} p(t_\beta)$, we can obtain the probability as follows:

$$P_{\text{inc}} \le \max_{i\in E_i} \sum_{t_\beta\in F_\beta} \sum_{\theta_{t_\beta}, r_{t_\beta}} p(t_\beta) p\left(\theta_{t_\beta}\right) p\left(r_{t_\beta}\right) \left(\langle \eta_{t_\beta}^{v_T} \middle| \sigma_{i|t_\beta} \middle| \eta_{t_\beta}^{v_T} \rangle \right)^2, \tag{5}$$

where $\sigma_{i|t_\beta}$ is the identity if $t_\beta \neq \beta$; otherwise, it is nontrivial, and $p(t_\beta)$ represents the probability of the qubit β as a trap qubit.

If the nontrivial attack is acted on the output qubits, we can obtain $F_\beta = P_v^O$ and

$$\begin{aligned} P_{\text{inc}} &\le \sum_{t_\beta\in P_{v\beta}^O} \sum_{\theta_{t_\beta}, r_{t_\beta}} p(t_\beta) p\left(\theta_{t_\beta}\right) p\left(r_{t_\beta}\right) \left(\langle +_{\theta_{t_\beta}} \middle| \sigma_{i|t_\beta} \middle| +_{\theta_{t_\beta}} \rangle \right)^2 \\ &= \frac{1}{16\left|P_{v\beta}^O\right|} \sum_{t_\beta\in P_{v\beta}^O} \sum_{\theta_t, r_t} \left(\langle +_\theta \middle| \sigma_{i|t_\beta} \middle| +_\theta \rangle \right)^2 \\ &\le \frac{1}{16} \left(16 \cdot \frac{\left(\left|P_{v\beta}^O\right| - 1\right)}{\left|P_{v\beta}^O\right|} + 8 \cdot \frac{1}{\left|P_{v\beta}^O\right|} \right) \\ &= \frac{5}{6}, \end{aligned} \tag{6}$$

where $p(\theta_{t_\beta}) p(r_{t_\beta}) = 1/16$ since $\theta_{t_\beta} \in \{0, \pi/4, \ldots, 7\pi/4\}$ and $r_{t_\beta} \in \{0, 1\}$, $\sum_\theta (\langle +_\theta | \sigma_i | +_\theta \rangle)^2 \le 4$ for $\sigma_i \in \{X, Y, Z\}$ and $1/|P_{v\beta}^O| = 1/3$ since trap qubits account for 1/3 of the output qubits.

If the nontrivial attack is acted on the nonoutput qubits, we can obtain $F_\beta = P_v^{NO}$and

$$\begin{aligned} P_{\text{inc}} &\le \sum_{t_\beta\in P_v^{NO}} \sum_{\theta_{t_\beta}, r_{t_\beta}} p(t_\beta) p\left(\theta_{t_\beta}\right) p\left(r_{t_\beta}\right) \left(\langle r_{t_\beta} | \sigma_{i|t_\beta} | r_{t_\beta} \rangle \right)^2 \\ &= \frac{1}{16\left|P_v^{NO}\right|} \sum_{t_\beta\in P_v^{NO}} \sum_{\theta_t, r_t} \left(\langle r_{t_\beta} | \sigma_{i|t_\beta} | r_{t_\beta} \rangle \right)^2 \\ &= \frac{1}{16\left|P_v^{NO}\right|} \sum_{r_t} \left(8 \cdot \left(\left|P_v^{NO}\right| - 1 \right) + 8 \cdot \left(\langle r_{t_\beta} | \sigma_{i|t_\beta} | r_{t_\beta} \rangle \right)^2 \right) \\ &= \frac{1}{16} \left(16 \cdot \frac{\left(\left|P_v^{NO}\right| - 1\right)}{\left|P_v^{NO}\right|} \right) \\ &= 1 - \frac{1}{\left|P_v^{NO}\right|} \\ &\le \frac{8}{9}, \end{aligned} \tag{7}$$

where $P_{\text{inc}} = 8/9$ if the attack is applied to added vertices of DT (G), since trap qubits account for 1/9 of the added qubits.

Thus, the probability that clients accept the incorrect computation result in the worst case is 8/9. In other words, the protocol is $(8/9)$ verifiable. If the protocol is repeated d times, the probability can be $(8/9)^d$ which approaches 0 if d is large enough. □

6. Comparison with Typical DQC Protocols

In this section, we compare the proposed MDQC protocol with other typical DQC protocols [15, 16, 19, 40] in Table 1 from the following five aspects: the quantum capability of the client, the quantum capability of the server, the number of clients, verifiability of computational results, and the required qubits.

In the BFK protocol [15], the MF protocol [16], and the FK protocol [19], only the case of one server and one client is considered, while in the proposed protocol and the KP protocol [40], the case of multiple clients cooperating together to complete the computation on a server is considered. In terms of the quantum capability of the client, in the proposed protocol and the FK protocol, the client requires the ability to prepare quantum states $|+_\theta\rangle$, $|0\rangle$, and $|1\rangle$. In the BFK protocol and the KP protocol, the client requires the ability to prepare quantum states $|+_\theta\rangle$, while in the MF protocol, the client requires the ability to measure single qubits on XY-plane bases. In addition, all these protocols require the server to be full quantum.

As for the verifiability of computational results, it cannot be achieved in the BFK protocol, the MF protocol, and the KP protocol. However, in the proposed MDQC protocol, each client can verify the correctness of their calculation results by using trap qubits, and it is $(8/9)^d$ verifiable as analyzed in the previous section. Besides, the FK protocol can be $(2/3)^{\lceil 2d/5 \rceil}$ verifiable.

The number of the qubits required in these DQC protocols is also analyzed. In the proposed protocol, the construction of a graph state DT (G) requires $3N + 9cN$ qubits, where N is the number of vertices in the base graph G and c is the maximum degree of the vertices of G. In the remote state preparation of the proposed protocol, each client needs to send one qubit to the server for each vertex in the graph state, and in Algorithm 4, to enforce the client to send qubits honestly, the server selects only one of $2m + 2$ qubits sent by the client for computations. Therefore, the number of the qubits required in this proposed protocol is $2mn(3N + 9cN)$. Since n, c, and m are much smaller than N, they can be treated as constants, and the number of the required qubits is linearly dependent on N which can be denoted as $O(N)$. Similarly, the number of the qubits required in the KP protocol is also $O(N)$. The FK protocol requires $3N(3N + 1)/2$ qubits for the computation and verification, and the qubit cost can be taken as $O(N^2)$. Both the BFK protocol and the MF protocol use only one graph state to complete the computation, so the number of the required qubits is $O(N)$.

7. An Example of the Verifiable MDQC Protocol for Three Parties and Its Simulation

In this section, a three-party example of the verifiable MDQC protocol is given and considered to be implemented on IBM's quantum platform. We suppose that there are three clients, C_1, C_2, and C_3, and the states of their real input qubits are $|+\rangle$, $|+_{\pi/2}\rangle$, and $|+_{\pi/4}\rangle$, respectively. They want to implement the target circuit as shown in Figure 7 on the quantum server.

Example 1. A specific three-party verifiable MDQC protocol and its simulation are given.

Input: The client C_1 generates three input quantum sates, a real input state $|+\rangle$, a trap state $|+\rangle$, and a dummy state $|1\rangle$. Similarly, C_2 generates $|+_{\pi/2}\rangle$, $|+\rangle$, and $|0\rangle$, and C_3 produces $|+_{\pi/4}\rangle$, $|+\rangle$, and $|0\rangle$.

7.1. The State Preparation Phase

(1) The three clients generate the corresponding DT (G) graph shown in Figure 8 according to the circuit in Figure 7 and send it to the server.

For each input qubit $i \in I$

(1) One client C_1 chooses $\alpha_1 = 1$ and $\theta_1 = 2\pi/3$ to encrypt the input $|+\rangle$ and sends the encrypted state $X^{\alpha_1} Z(\theta_1)|+\rangle$ to the server. Then, C_1 shares α_1 and θ_1 with other two clients by using a classical VSS scheme.

(2) The other two clients C_2 and C_3 also send qubits to the server. The server determines whether these qubits are correct or not by using Algorithm 4. We suppose that C_2 sends five qubits $|+_{\pi/2}\rangle$, $|+_{\pi/3}\rangle, |+_{\pi/4}\rangle$, $|+_{\pi/2+1\times\pi}\rangle$,and $|+_{\pi/3}\rangle$ to the server, and the server randomly selects four of them into two pairs of qubits $\{|+_{\pi/2}\rangle, |+_{\pi/3}\rangle\}(\gamma_1 = 0, \gamma_2 = 0)$, $\{|+_{\pi/2+1\times\pi}\rangle, |+_{\pi/3}\rangle\}(\gamma_1 = 1, \gamma_2 = 0)$ and uses the circuit in Figure 5 to verify whether the two pairs of qubits are correct. For each pair of qubit, if the measurement result of the second qubit in the circuit is equal to $\gamma_1 \oplus \gamma_2$, this pair of qubits is correct with high probability. Note that the circuit in Figure 5 is equal to that in Figure 9 which can be implemented on IBM's quantum platform by using the principle of deferred measurement. After the verification circuit for these two pairs of qubits is run 1024 times on the platform, the probabilities of the measurement results are shown in Figure 10. The probability that the measurement result of the second qubit is equal to $\gamma_1 \oplus \gamma_2$ is 100%, so the remaining quantum state $|+_{\pi/4}\rangle$ is correct with high probability, and the server obtains $|+_{\pi/4}\rangle$ from C_2. Similarly, the server can obtain a quantum state such as $|+_{3\pi/4}\rangle$ from C_3 via Algorithm 4.

(3) The three clients need to jointly encrypt the input quantum state $XZ(2\pi/3)|+\rangle$ sent by C_1. The server executes Algorithm 1 on $XZ(2\pi/3)|+\rangle$ sent by C_1 and $|+_{\pi/4}\rangle$ and $|+_{3\pi/4}\rangle$ from C_2 and C_3 to obtain the final result which should be $XZ(\delta)|+\rangle$, where $\delta = 2\pi/3 + (-1)^{(b_1 \oplus b_2)+1}\pi/4 + (-1)^{b_2+1}3\pi/4$, and b_1, b_2 represent the measurement results of the first two qubits in the circuit of Algorithm 1. The correctness of the result is verified by measuring the third qubit in the circuit. In order to be able to perform simulation on IBM's quantum platform, the

Table 1: Comparisons between the proposed protocols and other BQC protocols.

	The number of clients	The quantum capability of clients	The quantum capability of the server	Verifiability	The qubit cost
The KP protocol [40]	Multiple clients	Preparing single-qubit states	Full quantum	None	$O(N)$
The BFK protocol [15]	A client	Preparing single-qubit states	Full quantum	None	$O(N)$
The FK protocol [19]	A client	Preparing single-qubit states	Full quantum	Yes	$O(N^2)$
The MF protocol [16]	A client	Measuring single-qubit states	Full quantum	None	$O(N)$
The proposed protocol	Multiple clients	Preparing single-qubit states	Full quantum	Yes	$O(N)$

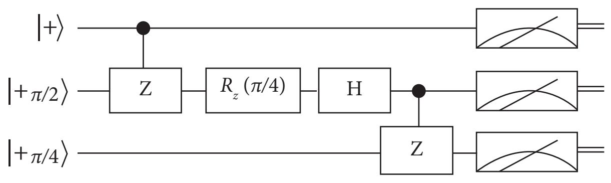


Figure 7: The target circuit for the three clients.

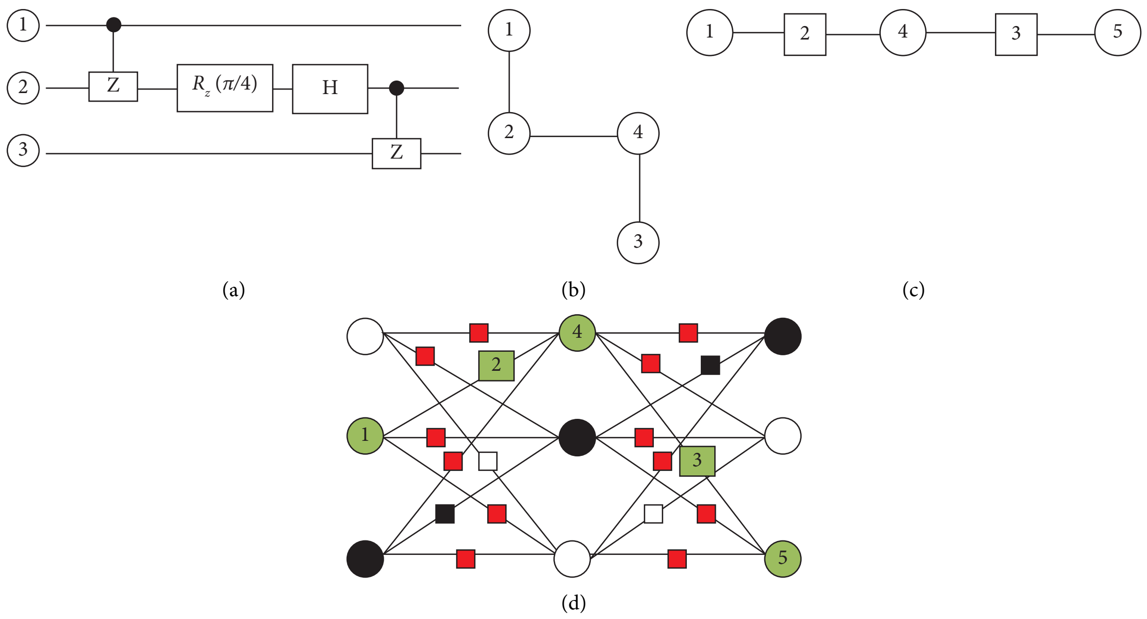


Figure 8: (a) The target circuit. (b) The base graph for the measurement-based quantum computation corresponding to the target circuit, where each edge of the base graph represents a CZ gate, and H gate and RZ gate in the circuit are realized by adding qubit 4. For example, if the state of qubit 2 is $|\psi\rangle$, the state of qubit 4 is $|+\rangle$. After measuring qubit 2 on the basis $|\pm_{-\pi/4}\rangle$ and obtaining measurement result s, the state of qubit 4 becomes $X^s HR_z(\pi/4)|\psi\rangle$. (c) A dotted base graph transformed from the base graph. The qubits 1, 2, 3, and 4 correspond to qubits 1, 2, 3, and 4 in the base graph. Qubit 5 is added to construct the DT (G) graph, which is a dummy qubit. (d) The DT (G) graph corresponding to the target circuit, where the green subgraph is used for computation and the black subgraph and white subgraph are used for verification. The qubits 1, 2, 3, 4, and 5 correspond to qubits 1, 2, 3, 4, and 5 in the dotted base graph.

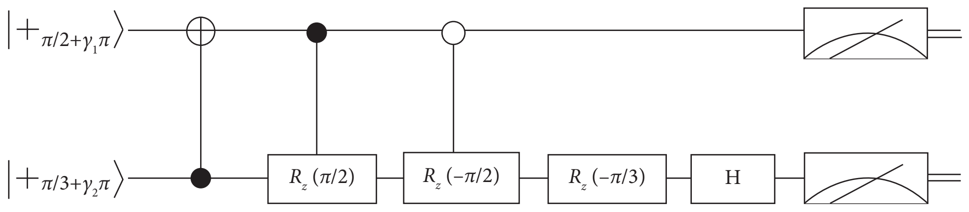

$|+_{\pi/2+\gamma_1\pi}\rangle$
$|+_{\pi/3+\gamma_2\pi}\rangle$
$R_z(\pi/2)$
$R_z(-\pi/2)$
$R_z(-\pi/3)$
H

Figure 9: The real circuit for implementing the logic circuit shown in Figure 5 on IBM's quantum platform.

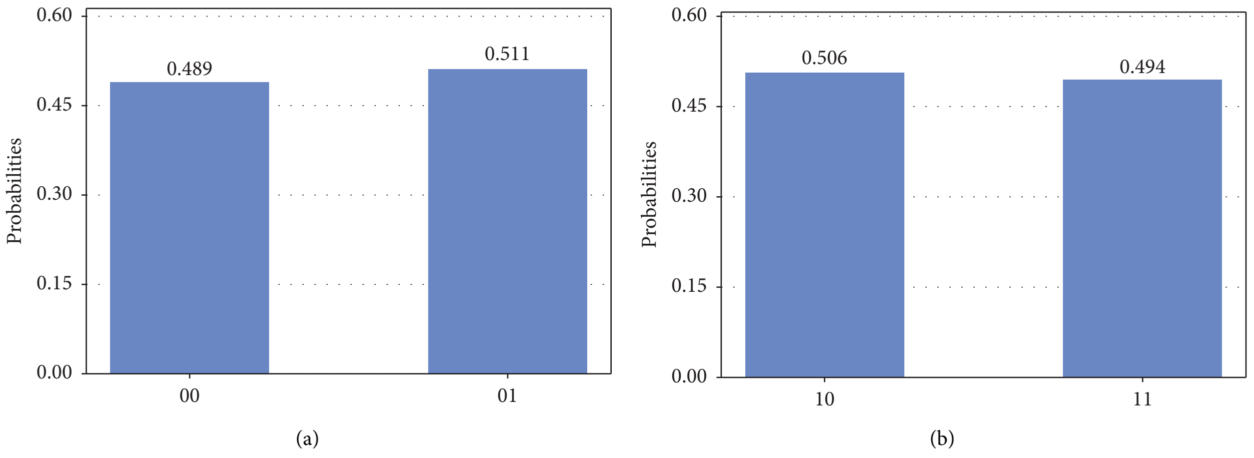


Figure 10: (a) The probabilities of the measurement results 00 and 10 after implementing the circuit of Algorithm 4 for a pair of qubits in states $|+_{\pi/2}\rangle$ and $|+_{\pi/3}\rangle$ are 48.9% and 51.1%, respectively. The first bit and second bit of 00 (and 10) represent the measurement result of the first qubit and that of the second qubit, respectively. So the measurement result of the second qubit is 0 with probability $100\% = 48.9\% + 51.1\%$, and it is equal to $\gamma_1 \oplus \gamma_2 = 0 \oplus 0$. (b) The probabilities of the measurement results 01 and 11 after implementing the circuit of Algorithm 4 for a pair of qubits $|+_{\pi/2+1\times\pi}\rangle$ and $|+_{\pi/3}\rangle$ are 50.6% and 49.4%, respectively. So the measurement result of the second qubit is 1 with probability $100\% = 49.4\% + 50.6\%$, and it is equal to $\gamma_1 \oplus \gamma_2 = 1 \oplus 0$. (a)$\{|+_{\pi/2}\rangle, |+_{\pi/3}\rangle\}(\gamma_1 = 0, \gamma_2 = 0)$. (b)$\{|+_{\pi/2+1\times\pi}\rangle, |+_{\pi/3}\rangle\}(\gamma_1 = 1, \gamma_2 = 0)$.

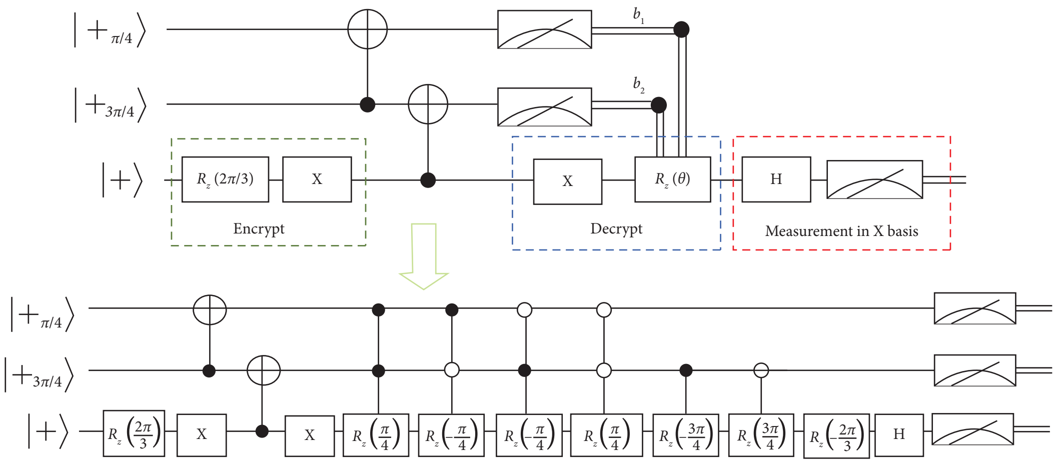

$|+_{\pi/4}\rangle$
$|+_{3\pi/4}\rangle$
$|+\rangle$
b_1
b_2
$R_z(2\pi/3)$
X
Encrypt
X
$R_z(\theta)$
Decrypt
H
Measurement in X basis
$R_z\left(\frac{2\pi}{3}\right)$
$R_z\left(\frac{\pi}{4}\right)$
$R_z\left(-\frac{\pi}{4}\right)$
$R_z\left(\frac{3\pi}{4}\right)$

Figure 11: The circuit for simulating Algorithm 1 by using the principle of deferred measurement, where the red box represents the measurement on the X basis and the green box and blue box represent encryption and decryption, respectively.

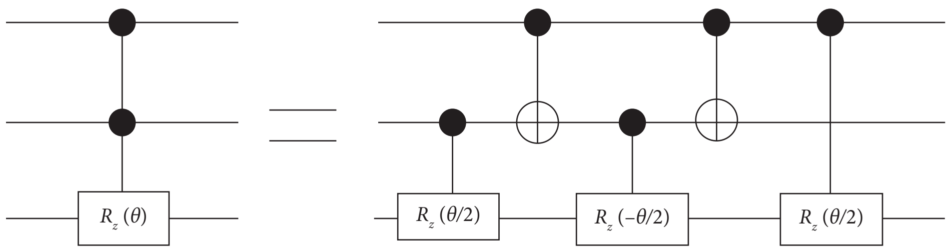


Figure 12: The realization of the double-controlled RZ gate by using the controlled RZ gates and CNOT gates.

circuit in Algorithm 1 is transformed by using the principle of deferred measurement, and the transformed circuit is shown in Figure 11. In addition, the double-controlled RZ gate can be realized by using a series of quantum gates composed of the controlled RZ gates and CNOT gates, as shown in Figure 12. The probabilities of the measurement results after 1024 runs of the circuit on the platform are shown in Figure 13, and the probability of obtaining result 0 by measuring the third qubit is 100%. Therefore, the correct result can be obtained by the circuit in Algorithm 1. Assuming that the obtained measurement

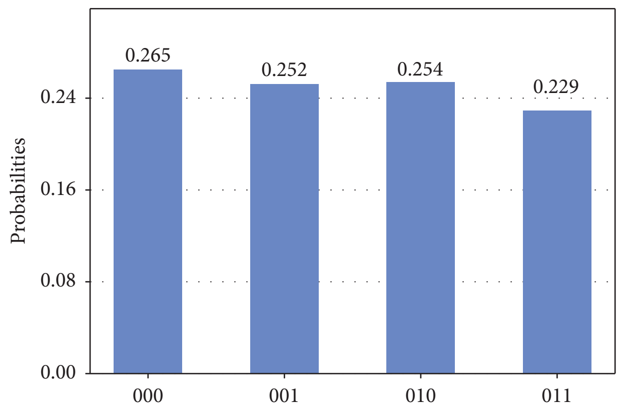


FIGURE 13: The probabilities of the measurement results 000, 100, 010, and 110 of all qubits in the circuit after implementing Algorithm 1 are 26.5%, 25.2%, 25.4%, and 22.9%, respectively. The third bit of these classical results represents the measurement result of the third qubit in the circuit, so the measurement result of the third qubit is 0 with probability 100% = 26.5% + 25.2% + 25.4% + 22.9%.

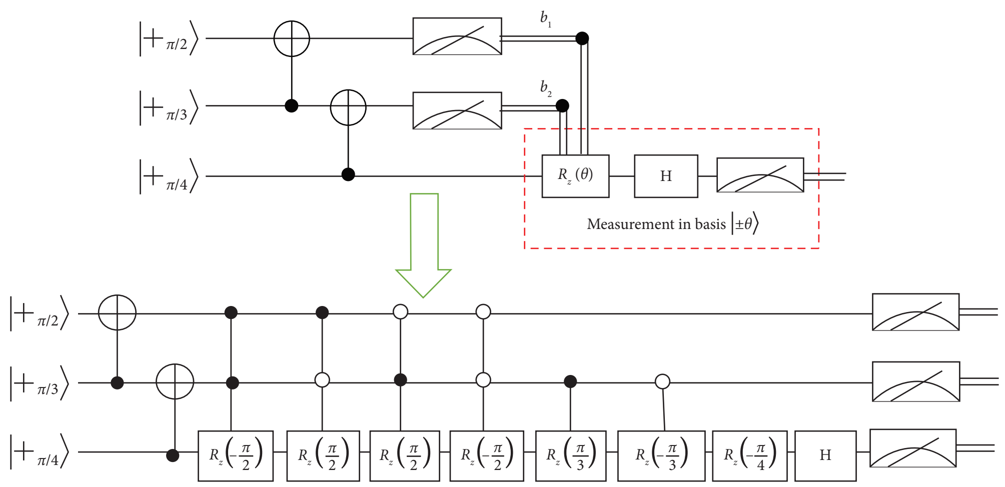

$|+_{\pi/2}\rangle$
$|+_{\pi/3}\rangle$
$|+_{\pi/4}\rangle$
b_1
b_2
$R_z(\theta)$
H
Measurement in basis $|\pm\theta\rangle$
$R_z\left(-\frac{\pi}{2}\right)$
$R_z\left(\frac{\pi}{2}\right)$
$R_z\left(\frac{\pi}{2}\right)$
$R_z\left(-\frac{\pi}{2}\right)$
$R_z\left(\frac{\pi}{3}\right)$
$R_z\left(-\frac{\pi}{3}\right)$
$R_z\left(-\frac{\pi}{4}\right)$
H

FIGURE 14: The circuit for implementing Algorithm 2 obtained by using the principle of deferred measurement. Note that the red box represents the measurement on the basis $|\pm_\theta\rangle$, where $\theta = \pi/4 + (-1)^{b_1 \oplus b_2}\pi/2 + (-1)^{b_2}\pi/3$.

results of the first two qubits are $b_1 = 1$ and $b_2 = 0$, the final encrypted input quantum state obtained by Algorithm 1 is $XZ(\pi/6)|+\rangle$, and the server returns b_1 and b_2 to all clients. The other final encrypted input quantum states will be produced by Algorithm 1 similarly.

For each noninput qubit $i \in I^c$

(1) C_1, C_2, and C_3 send $|+_{\pi/2}\rangle, |+_{\pi/3}\rangle$ and $|+_{\pi/4}\rangle$ to the server also through Algorithm 4, respectively.

(2) The server executes Algorithm 2 on these three qubits, and the final result of the calculation should be $|+_\theta\rangle$, where $\theta = \pi/4 + (-1)^{b_2}\pi/3 + (-1)^{b_1 \oplus b_2}\pi/2$ and b_1, b_2 represent the measurement results of the first two qubits in the circuit for implementing Algorithm 2 as shown in Figure 14. The correctness of the final result is verified by measuring the third qubit on the basis $|\pm_\theta\rangle$. After all the operations have been performed by using the quantum simulator provided by the platform, the result is 0 with probability 100% when measuring the third qubit 1024 times on the basis $|\pm_\theta\rangle$, as shown in Figure 15.

It implies that, by using Algorithm 2, the three clients are able to correctly prepare their jointly encrypted noninput qubits. For example, if the measurement results of the first two qubits in the circuit are $b_1 = 1$ and $b_2 = 1$, then the server obtains a noninput qubit $|+_{5\pi/12}\rangle$ and returns b_1 and b_2 to all clients.

Graph state: the server entangles the previously prepared qubits to form a DT (G) state.

7.2. *The Computation Phase*

For each nonoutput qubit $i \in O^c$

(1) Three clients calculate the corresponding measurement angle of the nonoutput qubit by using classical multiparty computation and send it to the server. Suppose that the actual measurement angle of a nonoutput qubit $|+_{5\pi/12}\rangle$ is $-\pi/4$. C_1 randomly chooses $r_1 = 0$ and shares it with the other two clients. C_2 and C_3 also choose $r_2 = 1$ and $r_3 = 0$, respectively, and make similar operations like C_1 does. The dummy qubits adjacent to this nonoutput

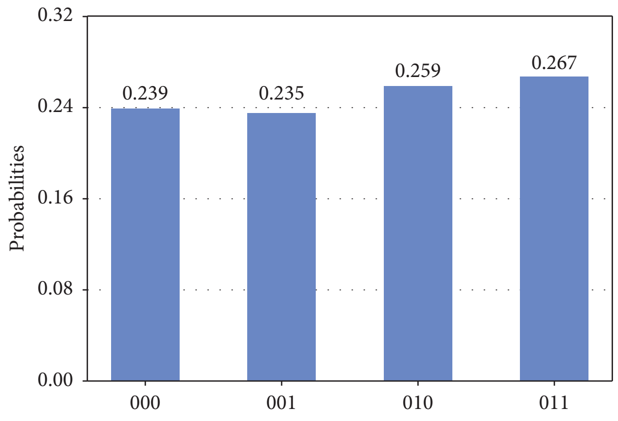


Figure 15: The probabilities of the measurement results 000, 100, 010, and 110 of all qubits in the circuit after implementing Algorithm 2 are 23.9%, 23.5%, 25.9%, and 26.7%, respectively. The third bit of these classical results represents the measurement result of the third qubit in the circuit, so the measurement result of the third qubit is 0 with probability 100% = 23.9% + 23.5% + 25.9% + 26.7%.

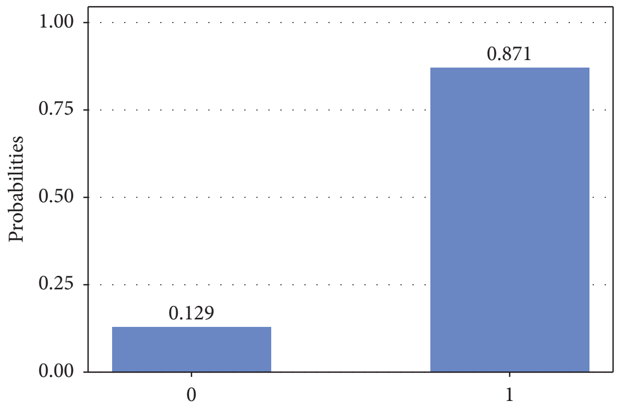


Figure 16: The probabilities of two measurement results obtained after the trap qubit $|+_{\pi/2}\rangle$ are measured on the basis $|\pm_{7\pi/4}\rangle$. Since an incorrect measurement basis is used for the measurement, the probability of getting the result 0 is small.

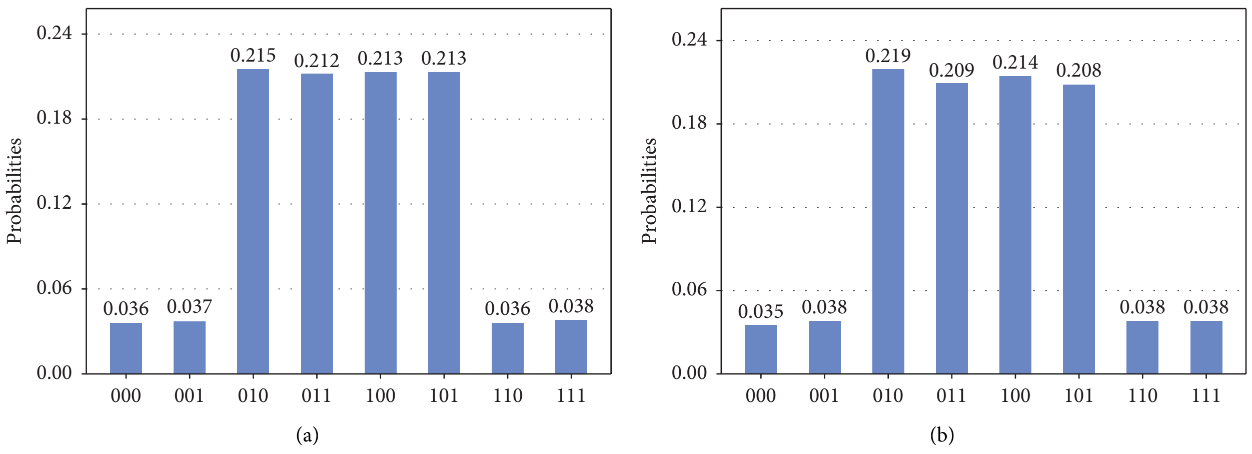


Figure 17: The measurement results of the target circuit and that of the given example for the proposed verifiable MDQC protocol. (a) The measurement results of the target circuit. (b) The measurement results of the given three-party verifiable MDQC protocol.

qubit are $|0\rangle$. Then, by using classical multiparty computation, the measurement angle should be $7\pi/6 = -\pi/4 + \pi(r_1 \oplus r_2 \oplus r_3) + 5\pi/12$ which is sent to the server.

(2) The server measures the qubit in the basis $|\pm_{7\pi/6}\rangle$ and returns the measurement result $t = 1$ to all clients. These clients get the actual result $s = t \oplus r_1 \oplus r_2 \oplus r_3 = 0$ by using classical multiparty computation.

(3) If the qubit is a trap qubit, it can be used to verify whether the server performed the measurement honestly. If the state of the trap qubit is $|+_{\pi/2}\rangle$ and measured in the basis $|+_{\pi/2}\rangle$, the correct result 0 can be obtained. But if it is measured on a wrong basis such as $|\pm_{7\pi/4}\rangle$, the probability of getting the correct result 0 is small as shown in Figure 16.

For each output qubit $i \in O$

(1) The server sends the output qubit to the corresponding client.

(2) If the output qubit $|+_{3\pi/4}\rangle$ is a trap qubit, the measurement angle $3\pi/4$ can be obtained by using classical multiparty computation, and the corresponding client measures the trap qubit on the basis $|+_{3\pi/4}\rangle$. If the measurement result is incorrect, the client can know that the server returned the wrong output qubit.

(3) Three clients decrypt their real output qubits to get the actual results. The decrypted output qubits can be measured to make comparisons with the measurement results of the target circuit as shown in Figure 17. According to Figure 17, the measurement results are almost the same as expected.

From the above example and the results of the simulation, it can be concluded that the proposed MDQC protocol can be correctly simulated on IBM's quantum platform. However, once the real quantum system is used to realize this protocol, the error rate will increase due to the use of large number of real imperfect quantum gates.

8. Conclusion

We have proposed an algorithm that enforces the clients to send qubits honestly to the server without revealing the positions of trap qubits in the graph state and a verifiable MDQC protocol based on the given algorithm and DT (G), in which each client can verify the correctness of the computation. Furthermore, the proposed MDQC protocol is secure against a dishonest server and against a coalition of dishonest clients. It can be applied in quantum networks to solve the problem of multiple clients cooperating together to complete a calculation on an untrusted server. In addition, the proposed protocol has been compared with similar other BQC protocols to show its advantages. A specific example for three parties also has been given and implemented on IBM's quantum platform to show the feasibility of the proposed verifiable MDQC protocol. However, in order to satisfy the property of verification, clients need the ability to prepare dummy qubits and trap qubits in the presented protocol. Therefore, whether a verifiable MDQC protocol can be implemented by reducing quantum capabilities of clients or using less quantum resources deserves further investigation.

Data Availability

The data used to support the findings of this study are available upon request to the authors.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Authors' Contributions

The manuscript was written by Qin Li and Can Wang. Jiang Zhu, Lingling Xu, and Zhiwei Sun checked the algorithms and protocols for errors. All the authors have read and approved the manuscript.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (Grant nos. 62271436 and U1736113), the Science and Technology Innovation Program of Hunan Province (Grant no. 2022RC1187), the Natural Science Foundation of Hunan Province (Grant no. 2018JJ2403), and the Shenzhen Science and Technology Program (Grant no. JCYJ20210324100813034).

References

[1] P. W. Shor, "Algorithms for quantum computation: discrete logarithms and factoring," in *Proceedings of the 35th Annual Symposium on Foundations of Computer Science*, pp. 124–134, Santa Fe, NM, USA, November 1994.

[2] L. K. Grover, "Quantum mechanics helps in searching for a needle in a haystack," *Physical Review Letters*, vol. 79, no. 2, pp. 325–328, 1997.

[3] Q. Li, J. Wu, J. Quan, J. Shi, and S. Zhang, "Efficient quantum blockchain with a consensus mechanism QDPoS," *IEEE Transactions on Information Forensics and Security*, vol. 17, pp. 3264–3276, 2022.

[4] P. Wang, W. Chen, S. Lin, L. Liu, Z. Sun, and F. Zhang, "Consensus algorithm based on verifiable quantum random numbers," *International Journal of Intelligent Systems*, vol. 37, no. 10, pp. 6857–6876, 2022.

[5] A. Malossini, E. Blanzieri, and T. Calarco, "Quantum genetic optimization," *IEEE Transactions on Evolutionary Computation*, vol. 12, no. 2, pp. 231–241, 2008.

[6] H. Zhenxue, W. Xiaoqian, W. Chao, H. Zhisheng, X. Limin, and W. Xiang, "Delay optimization for ternary fixed polarity Reed–Muller circuits based on multilevel adaptive quantum genetic algorithm," *International Journal of Intelligent Systems*, vol. 36, no. 10, pp. 5981–6006, 2021.

[7] S. Jaques, M. Naehrig, M. Roetteler, and F. Virdia, "Implementing Grover oracles for quantum key search on AES and LowMC," *Advances in Cryptology – Eurocrypt 2020, LNCS*, vol. 12106, pp. 280–310, 2020.

[8] J. Zou, Z. Wei, S. Sun, X. Liu, and W. Wu, "Quantum circuit implementations of AES with fewer qubits," *Advances in*

Cryptology – Asiacrypt 2020, LNCS, vol. 12492, pp. 697–726, 2020.

[9] X. B. Chen, Y. R. Sun, G. Xu, and Y. X. Yang, "Quantum homomorphic encryption scheme with flexible number of evaluator based on (k, n)-threshold quantum state sharingk, n)-threshold quantum state sharing," *Information Sciences*, vol. 501, pp. 172–181, 2019.

[10] J. Liu, Q. Li, J. Quan, C. Wang, J. Shi, and H. Situ, "Efficient quantum homomorphic encryption scheme with flexible evaluators and its simulation," *Designs, Codes and Cryptography*, vol. 90, no. 3, pp. 577–591, 2022.

[11] P. Li, J. Li, Z. Huang et al., "Multi-key privacy-preserving deep learning in cloud computing," *Future Generation Computer Systems*, vol. 74, pp. 76–85, 2017.

[12] P. Li, J. Li, Z. Huang, C. Z. Gao, W. B. Chen, and K. Chen, "Privacy-preserving outsourced classification in cloud computing," *Cluster Computing*, vol. 21, no. 1, pp. 277–286, 2018.

[13] A. M. Childs, "Secure assisted quantum computation," *Quantum Information and Computation*, vol. 5, no. 6, pp. 456–466, 2005.

[14] P. Arrighi and L. Salvail, "Blind quantum computation," *International Journal of Quantum Information*, vol. 04, no. 05, pp. 883–898, 2006.

[15] A. Broadbent, J. Fitzsimons, and E. Kashefi, "Universal blind quantum computation," in *Proceedings of the 50th Annual IEEE Symposium on Foundations of Computer Science*, pp. 517–526, Atlanta, GA, USA, October 2009.

[16] T. Morimae and K. Fujii, "Blind quantum computation protocol in which Alice only makes measurements," *Physical Review A*, vol. 87, no. 5, Article ID 050301, 2013.

[17] T. Morimae and K. Fujii, "Secure entanglement distillation for double-server blind quantum computation," *Physical Review Letters*, vol. 111, no. 2, Article ID 020502, 2013.

[18] Q. Li, W. H. Chan, C. Wu, and Z. Wen, "Triple-server blind quantum computation using entanglement swapping," *Physical Review A*, vol. 89, no. 4, Article ID 040302, 2014.

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

[20] T. Morimae, "Verification for measurement-only blind quantum computing," *Physical Review A*, vol. 89, no. 6, Article ID 060302, 2014.

[21] T. Morimae, "Measurement-only verifiable blind quantum computing with quantum input verification," *Physical Review A*, vol. 94, no. 4, Article ID 042301, 2016.

[22] A. Broadbent, "How to verify a quantum computation," *Theory of Computing*, vol. 14, no. 1, pp. 1–37, 2018.

[23] G. Sato, T. Koshiba, and T. Morimae, "Arbitrable blind quantum computation," *Quantum Information Processing*, vol. 18, no. 12, pp. 370–378, 2019.

[24] M. Hayashi and T. Morimae, "Verifiable measurement-only blind quantum computing with stabilizer testing," *Physical Review Letters*, vol. 115, no. 22, Article ID 220502, 2015.

[25] E. Kashefi and P. Wallden, "Optimised resource construction for verifiable quantum computation," *Journal of Physics A: Mathematical and Theoretical*, vol. 50, no. 14, Article ID 145306, 2017.

[26] J. Quan, Q. Li, C. Liu, J. Shi, and Y. Peng, "A simplified verifiable blind quantum computing protocol with quantum input verification," *Quantum Engineering*, vol. 3, no. 1, p. e58, 2021.

[27] Q. Li, Z. Li, W. H. Chan, S. Zhang, and C. Liu, "Blind quantum computation with identity authentication," *Physics Letters A*, vol. 382, no. 14, pp. 938–941, 2018.

[28] R. T. Shan, X. Chen, and K. G. Yuan, "Multi-party blind quantum computation protocol with mutual authentication in network," *Science China Information Sciences*, vol. 64, no. 6, pp. 162302–162314, 2021.

[29] A. Gheorghiu, E. Kashefi, and P. Wallden, "Robustness and device independence of verifiable blind quantum computing," *New Journal of Physics*, vol. 17, no. 8, Article ID 083040, 2015.

[30] M. Hayashi and M. Hajdušek, "Self-guaranteed measurement-based quantum computation," *Physical Review A*, vol. 97, no. 5, Article ID 052308, 2018.

[31] Q. Xu, X. Tan, R. Huang, and X. Zeng, "Parallel self-testing for device-independent verifiable blind quantum computation," *Quantum Engineering*, vol. 2, no. 3, p. e51, 2020.

[32] C. Gustiani and D. P. DiVincenzo, "Blind oracular quantum computation," *Quantum Science and Technology*, vol. 6, no. 4, Article ID 045022, 2021.

[33] A. C. C. Yao, "How to generate and exchange secrets," in *Proceedings of the 27th Annual Symposium on Foundations of Computer Science*, pp. 162–167, Washington, DC, USA, October 1986.

[34] C. Crépeau, D. Gottesman, and A. Smith, "Secure multi-party quantum computation," in *Proceedings of the 34th Annual ACM Symposium on Theory of Computing*, pp. 643–652, Chicago, IL, USA, June 2002.

[35] M. Ben-Or, C. Crépeau, D. Gottesman, A. Hassidim, and A. Smith, "Secure multiparty quantum computation with (only) a strict honest majority," in *Proceedings of the 47th Annual IEEE Symposium on Foundations of Computer Science*, pp. 249–260, Berkeley, CA, USA, October 2006.

[36] B. Alon, H. Chung, K. M. Chung, M. Y. Huang, Y. Lee, and Y. C. Shen, "Round efficient secure multiparty quantum computation with identifiable abort," *Advances in Cryptology–Crypto 2021, LNCS*, vol. 12825, pp. 436–466, 2021.

[37] V. Lipinska, J. Ribeiro, and S. Wehner, "Secure multiparty quantum computation with few qubits," *Physical Review A*, vol. 102, no. 2, Article ID 022405, 2020.

[38] Z. Deng, Y. Zhang, X. Zhang, and L. Li, "Privacy-preserving quantum multi-party computation based on circular structure," *Journal of Information Security and Applications*, vol. 47, pp. 120–124, 2019.

[39] Y. Dulek, A. B. Grilo, S. Jeffery, C. Majenz, and C. Schaffner, "Secure multi-party quantum computation with a dishonest majority," *Advances In Cryptology – Eurocrypt 2020, LNCS*, vol. 12107, pp. 729–758, 2020.

[40] E. Kashefi and A. Pappa, "Multiparty delegated quantum computing," *Cryptography*, vol. 1, no. 2, p. 12, 2017.

[41] R. Raussendorf and H. J. Briegel, "A one-way quantum computer," *Physical Review Letters*, vol. 86, no. 22, pp. 5188–5191, 2001.

[42] D. E. Deutsch, “Quantum computational networks,” *Proceedings of the Royal Society of London, Series A: Mathematical and Physical Sciences*, vol. 425, no. 1868, pp. 73–90, 1989.

[43] R. Raußendorf, “Measurement-based quantum computation with cluster states,” *International Journal of Quantum Information*, vol. 07, no. 06, pp. 1053–1203, 2009.

[44] V. Danos and E. Kashefi, “Determinism in the one-way model,” *Physical Review A*, vol. 74, no. 5, Article ID 052310, 2006.

[45] R. Bendlin, I. Damgård, C. Orlandi, and S. Zakarias, “Semi-homomorphic encryption and multiparty Computation,” *Advances in Cryptology – Eurocrypt 2011, LNCS*, vol. 6632, pp. 169–188, 2011.

[46] I. Damgård, V. Pastro, N. Smart, and S. Zakarias, “Multiparty computation from somewhat homomorphic encryption,” *Advances in Cryptology – Crypto 2012, LNCS*, vol. 7417, pp. 643–662, 2012.

[47] R. Kumaresan, A. Patra, and C. P. Rangan, “The round complexity of verifiable secret sharing: the statistical case,” *Advances in Cryptology - Asiacrypt 2010, LNCS*, vol. 6477, pp. 431–447, 2010.

[48] P. Laud and A. Pankova, “Verifiable computation in multiparty protocols with honest majority,” in *Proceedings of the Provable Security (ProvSec 2014)*, pp. 146–161, Hong Kong, China, October 2014.

[49] S. Kandar and B. C. Dhara, “A verifiable secret sharing scheme with combiner verification and cheater identification,” *Journal of Information Security and Applications*, vol. 51, Article ID 102430, 2020.