


Universal blind quantum computation with improved brickwork states

Shuquan Ma

*Optical Communication Research and Development Center, 54th Research Institute of China Electronics Technology Group Corporation, Shijiazhuang 050081, China
and State Key Laboratory of Integrated Services Networks, Xidian University, Xi'an, Shaanxi 710070, China*Changhua Zhu ^{*}*State Key Laboratory of Integrated Services Networks, Xidian University, Xi'an, Shaanxi 710071, China*

Xuchao Liu, Huagui Li, and Shaobo Li

Optical Communication Research and Development Center, 54th Research Institute of China Electronics Technology Group Corporation, Shijiazhuang 050081, China

(Received 4 July 2023; revised 18 October 2023; accepted 8 December 2023; published 4 January 2024)

Universal blind quantum computation allows a client who has limited quantum abilities to delegate his or her private computation to an untrusted quantum server. The first universal blind quantum computation protocol was proposed by Broadbent, Fitzsimons, and Kashefi. In their work the computation resource is the so-called brickwork state, which can be constructed by a number of specific single qubits. The number of single qubits is in theory linearly related to the size of the quantum circuit. However, due to the fixed structure of the brickwork states, the actual number of qubits is usually far more than the linear size. In this work we mainly construct three improved brickwork states whose structures are no longer fixed, thus they can efficiently reduce the client's qubit consumption. Using those improved brickwork states we propose a class of efficient universal blind quantum computation protocols. In our basic protocol, to implement a single-qubit basic gate the client needs to prepare only two ancillary qubits, while in the original protocol it will consume four ancillary qubits. Making use of our improved brickwork states the qubit consumption of implementing a nonadjacent two-qubit gate such as a control- X gate can be sharply reduced.

DOI: [10.1103/PhysRevA.109.012606](https://doi.org/10.1103/PhysRevA.109.012606)**I. INTRODUCTION**

Quantum computation has been extensively explored from theory to practice for a few decades [1–3]. Nowadays, it is widely believed that in the near future quantum computers will be similar to today's classical supercomputers; they will be used as specialized computation devices to accelerate the solution of certain difficult problems, which means quantum computers might be possessed by only a few organizations or companies and most people will have no access to them. To make quantum computers available to the general public, the idea of delegated quantum computation has been naturally proposed. In 2016 IBM first launched its cloud quantum computing service, then many companies followed closely, e.g., Google, Intel, and D-Wave. It is foreseeable that delegated quantum computation will be one of the main application forms of quantum computers during the noisy intermediate-scale quantum era [4].

Current delegated quantum computation is similar to the classical cloud computing. First, a classical client sends his or her quantum computation task, encoded as classical information, to a remote quantum server. Then, the server performs

the complete quantum computation for the client including preparation and measurement. Finally, the server sends the measurement outcome, i.e., the computation output, back to the client. Apparently the client's computation is not private since the server knows everything about the computation including input, output, and algorithm. Fortunately an unconditionally secure delegated quantum computation in theory can be achieved as long as we consider a half-classical client, i.e., a classical client with some specific quantum capacities [5]. Arrighi and Salvail first proposed the conception of blind quantum computation [6]. In their work they assumed that clients are able to generate arbitrary quantum states at least. However, their protocol considered only a class of *random verifiable functions* and thus was not universal. In fact, in 2005 Childs had proposed a universal secure delegated quantum computation protocol, called *secure assisted quantum computation* [7], which can guarantee that both input and output are secure but requires that clients have ability to generate specific qubits and perform Pauli X and Z gates and have polynomial-size quantum memory. Besides that, the protocol also requires a two-way classical and quantum communication during the computation. Clearly, Child's protocol is demanding for clients. In order to reduce the requirements, scholars put forward some improved protocols [8–11]. For example, in 2014 Broadbent proposed an improved protocol

^{*}chhzhu@xidian.edu.cn

[8], in which the client needs only to generate some specific qubits and perform Pauli X and Z gates and the phase gate S . Moreover, all qubits used for computation can be sent to the server before the computation, thus the protocol requires only a two-way classical communication during the computation.

In 2009 Broadbent, Fitzsimons, and Kashefi proposed the so-called *universal blind quantum computation* (UBQC) protocol [12]. Here the *blindness* means that the server can learn nothing about the computation except the upper bound of the computation size. It should be mentioned that Child's and related improved protocols can also satisfy the blindness, for example, making use of the *universal quantum circuit* [13]. Nevertheless, the work of Broadbent *et al.* opened a new direction toward the private quantum computation since it is the first secure delegated quantum computation protocol that considers working under the measurement-based quantum computation (MBQC) model [14–16] instead of the circuit-based quantum computation model [17]. The basic work flow of their UBQC protocol is as follows. First, the client randomly prepares a sequence of specific single qubits and then sends all them to the server. Once the server receives all qubits, it entangles them by the controlled- Z (CZ) gate into a specific multiqubit entangled state, i.e., a *cluster* state. After that, the client instructs the server to measure all nonoutput qubits step by step in an interactive manner. That is, the client tells the server how to measure each qubit orderly, and after each measurement the server sends the measurement outcome to the client. Finally, for a classical output, the server measures all output qubits and sends the measurement outcomes to the client, while for a quantum output, the server directly sends the output qubits to the client. Whether a classical or quantum output, only the client can acquire the correct computation result. After their pioneering work, a great number of blind quantum computation protocols and related work based on the MBQC model were put forward [18–26]. For example, Morimae considered the universal blind quantum computation under the continuous-variable MBQC model [18]. Later Morimae and Fujii proposed a fault-tolerant blind quantum computation protocol making use of the three-dimensional (3D) Raussendorf-Harrington-Goyal state [19]. In 2015 Morimae, Dunjko, and Kashefi proposed two blind quantum computation protocols that utilize the Affleck-Kennedy-Lieb-Tasaki state [22]. As well, Mantri, Delgado, and Fitzsimons developed a general framework to bound the resources of any possible blind quantum computation and showed that the UBQC protocol of Broadbent *et al.* comes within a factor $\frac{8}{3}$ of optimal when the client is restricted to preparing single qubits [27]. Dunjko and Kashefi examined the quantum capacity of the client and showed that, for the client, the capacity to prepare arbitrary two nonorthogonal qubits suffices for implementing any blind quantum computation [28]. Considering the measurement of a qubit is much easier than the generation of a qubit, Morimae and Fujii also proposed a measurement-only blind quantum computation protocol where the client needs only to perform single-qubit measurement [29].

In this paper we improve the original work of Broadbent *et al.* from the perspective of the client-side qubit consumption. Specifically we construct three improved brickwork states which can reduce the client's qubit overhead in different degrees. The main improved brickwork state, which we call

the square brickwork state, is able to reduce roughly half of qubit consumption comparing to the original brickwork state. While the other two improved brickwork states are variants of the square brickwork state and their graph structures are more complicated than the square brickwork state, the variant brickwork states can further reduce the qubit overhead of the client. Our main construction technique for the improved brickwork states is inspired by the excellent work of Fitzsimons and Kashefi for verifiable quantum computation [30], where they use the same technique to introduce an unconditional verification method in blind quantum computations.

The rest of this paper is organized as follows. In Sec. II we briefly review the basics of the UBQC protocol. In Sec. III we discuss the main deficiencies of the brickwork state proposed by Broadbent *et al.* Then in Sec. IV we present our first improved graph state in detail. In Sec. V using this improved graph state we propose an efficient UBQC protocol. In Sec. VI we construct two variants of the former improved graph states which can further reduce the qubit consumption. In the last section, we give a short conclusion on our work and point out some possible remaining problems.

II. UNIVERSAL BLIND QUANTUM COMPUTATION

As mentioned, the UBQC protocol proposed in [12] is built upon the MBQC model [14–16] where the computation is driven by a sequence of single-qubit measurements on a given cluster state, also known as a *graph* state. In this section we will briefly review the MBQC model. A comprehensive review of the MBQC model can be found in Ref. [31].

In the standard MBQC model, the graph state is usually described by a undirected graph $G = (V, E)$ where V is the vertex set while E is the edge set. For each vertex $i \in V$, it represents a qubit i prepared in state $|+\rangle$. For each edge $e_{i,j} \in E$, it represents a CZ gate performed on qubits i, j . The set V contains two nonoverlapping subsets I and O , which represent the input qubits¹ and the output qubits of the computation, respectively. Given a graph state denoted by $|G\rangle$, a computation in the MBQC model can be formally characterized by a measurement *pattern* $\mathbb{M}_{|G\rangle, \text{comp}} = (\{\theta_i\}_{i \in V \setminus O}, f_G)$, where $\{\theta_i\}_{i \in V \setminus O}$ specify all measurement angles on the nonoutput qubits while $f_G : V \setminus O \rightarrow V \setminus I$ is a bijective function (known as *flow*) that captures the ordering of measurements on $|G\rangle$. Hereafter, given the graph state $|G\rangle$ we will abbreviate the measurement pattern to $\mathbb{M}_{\text{comp}} = (\theta, f)$. Due to the nature of the MBQC model, the measurement angles $\{\theta_i\}_{i \in V \setminus O}$ are amended as

$$\theta'_i = (-1)^{s_{f^{-1}(i)}} \theta_i + \sum_{\substack{j: f(j) \in N_G(i) \\ j \neq i}} s_j \pi, \quad (1)$$

where $N_G(i)$ denotes the neighbor vertex set of the vertex i in the graph G and f^{-1} is the inverse of the flow f , while $s_{f^{-1}(i)}, s_j \in \{0, 1\}$ are the measurement outcomes of the qubits

¹In fact, the input qubits are not necessary to be prepared in state $|+\rangle$, they can be any desired state. Nevertheless, any state $|\psi\rangle$ can be obtained by applying an appropriate unitary operator U_ψ on a fixed input state. Of course, the size of U_ψ must be polynomial in n .

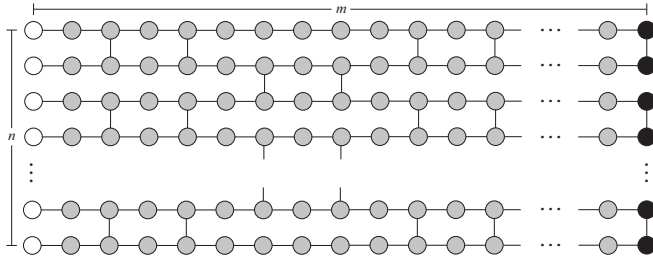


FIG. 1. Structure of an (nm) -qubit brickwork state defined in [12], where n, m satisfy that $n \equiv 0 \pmod{2}$ and $m \equiv 5 \pmod{8}$.

$f^{-1}(i)$ and j . In particular, we define $s_{f^{-1}(i)} = 0$ for all $i \in I$. For simplicity, in the following content we will abbreviate $f(i)$ and $f^{-1}(i)$ as f_i and f_i^{-1} , respectively.

In the UBQC protocol, the graph state used for computations is called the *brickwork state*; see Fig. 1 for a detailed construction. Each circle denotes a single qubit $|+\rangle$ while each edge connecting two circles denotes a two-qubit CZ gate. The white circles denote the input qubits while the black ones denote the output qubits; the rest denote the non-input-and-output qubits. The measurements are performed column by column from the leftmost side. In this paper we index the qubits of the brickwork state in the following way: for the qubit in the k th row and l th column, we use the number $i = (l - 1)n + (k - 1)$ to denote it. In other words, we index the first column of qubits (i.e., the input qubits) from top to bottom as $0, 1, \dots, n - 1$, and the second column of qubits as $n, n + 1, \dots, 2n - 1$, and so on. By this definition, it can be inferred that for any nonoutput qubit i , we always have $f_i = i + n$. The basic unit of the brickwork state can implement basic gates of the circuit-based quantum computation (see Fig. 2), such as H, T , and CX. Thus, the brickwork state is *universal*. We list in Fig. 3 the concrete measurement angles for the basic gates H, T, CX , and the identity gate I .

Obviously, according to Eq. (1), given all amended measurement angles and outcomes $\{\theta'_i, s_i\}_{i \in V \setminus \mathcal{O}}$, it is easy to determine the measurement angles $\{\theta_i\}_{i \in V \setminus \mathcal{O}}$. Thus, in order to guarantee that the server learns nothing about the computation, all qubits used to construct the brickwork state are prepared in state $|+\delta_i\rangle$ instead of $|+\rangle$, where $\delta_i \in_{\mathbb{R}} \{0, \frac{\pi}{4}, \dots, \frac{7\pi}{4}\}$. Note that up to an insignificant global phase $|+\delta_i\rangle \equiv R_z(\delta_i)|+\rangle$, and $R_z(\delta_i)$ commutes with CZ, thus we can always think that all qubits are first prepared in state $|+\rangle$ and then performed by a sequence of CZ gates according to the structure of the brickwork state, and finally each qubit i is rotated by $R_z(\delta_i)$. In the MBQC model, the single-qubit projective measurement $M(\theta)$ can be described by the operator

$$M(\theta) \equiv \sum_{s \in \{0,1\}} (-1)^s |+\theta+s\pi\rangle \langle +\theta+s\pi|, \quad (2)$$

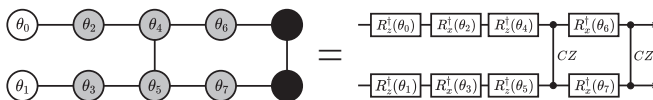


FIG. 2. Basic unit of the brickwork state, which implements a basic quantum circuit as shown in the right side.

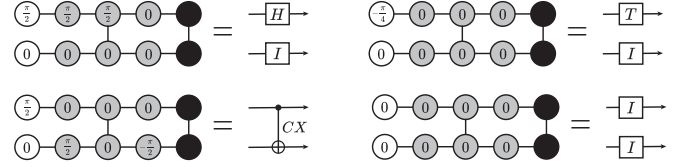


FIG. 3. Measurement angles for the basic gates H, T, CX , and the identity gate I .

where $\theta \in [0, 2\pi)$ is the measurement angle and $s \in \{0, 1\}$ denotes the measurement outcome. Traditionally, $M(\theta)$ is treated as a 'destructive measurement' that is, once a qubit is measured we can simply think that it will be discarded. In this case, it can be easily inferred that, for any multiqubit state $|\phi\rangle$, using $M(\theta_i)$ to measure a qubit i of $|\phi\rangle$ is equivalent to using $M(\theta_i + \delta_i)$ to measure the qubit i of $R_z(\delta_i)|\phi\rangle$, where $R_z(\delta_i)$ is acted on the qubit i . This *equivalence* refers to the postmeasurement state (not including the qubit i) and its occurring probability. Furthermore, according to Eq. (2), given a random bit $r \in \{0, 1\}$ we can obtain that

$$M(\theta + r\pi) = \sum_{s \in \{0,1\}} (-1)^{s \oplus r} |+\theta+s\pi\rangle \langle +\theta+s\pi|, \quad (3)$$

where $s \oplus r \in \{0, 1\}$ denotes the measurement outcome. Clearly, the measurement outcome of $M(\theta + r\pi)$ can be viewed as a one-time pad of the measurement outcome of $M(\theta)$, where r is the secure key. As a result, to correct the deviations caused by Z rotation and encrypt the measurement outcomes, the actual measurement angles $\{\phi_i\}_{i \in V \setminus \mathcal{O}}$ are adjusted as

$$\phi_i = (-1)^{s_{f_i^{-1}}} \theta_i + \sum_{\substack{j: f_j \in N_G(i) \\ j \neq i}} s_j \pi + \delta_i + r_i \pi, \quad (4)$$

where $\delta_i \in_{\mathbb{R}} \{0, \frac{\pi}{4}, \dots, \frac{7\pi}{4}\}$ and $r_i \in_{\mathbb{R}} \{0, 1\}$. Let $b_i \in \{0, 1\}$ be the measurement outcome of $M(\phi_i)$, then the measurement outcome of $M(\theta_i)$ will be $s_i = b_i \oplus r_i$. Note that the term $\delta_i + r_i \pi$ can be treated as a secure key to encrypt θ'_i . In summary, given the measurement angles and outcomes $\{\phi_i, b_i\}_{i \in V \setminus \mathcal{O}}$ the server can infer nothing about $\{\theta_i, s_i\}_{i \in V \setminus \mathcal{O}}$, which means the computation is blind to the server.

III. DEFICIENCIES OF THE BRICKWORK STATE

Although the brickwork state provides a nice framework for implementing the universal blind quantum computation, it suffers some intrinsic drawbacks. In this section we discuss those limits briefly.

In the following content, we consider an n -qubit circuit U with a *depth* of $O(n^c)$, which consists of basic gates such as H, T , and CX. We first estimate the number of qubits that are used to construct the brickwork state that implements the circuit U . On the face of it, the number of qubits is no more than $4nO(n^c) = O(n^{c+1})$ since each single-qubit basic gate (including the identity gate I) consumes four qubits (not including the output qubits). However, the real situation is more complicated due to the structure of the brickwork state. For example, consider a four-qubit circuit U_{swaps} which consists of two swap gates as shown in Fig. 4(a); this circuit can be

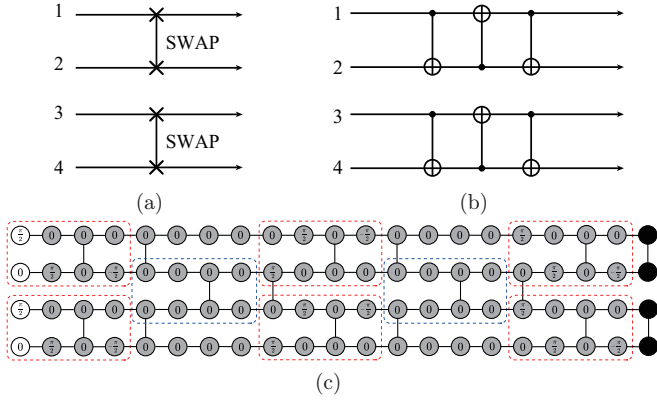


FIG. 4. (a) Example circuit U_{swaps} that implements two swap gates on qubits 1, 2 and 3, 4; (b) equivalent U_{swaps} consisting of six CX gates; (c) minimal brickwork state implementing U_{swaps} .

decomposed as six CX gates as shown in Fig. 4(b), where we can see the depth of U_{swaps} is 3. In theory, achieving this circuit demands $4 \times 4 \times 3 = 48$ qubits in total (not including the input qubits). But from Fig. 4(c) we can infer that the number of qubits is in fact $4 \times 4 \times 5 = 80$. The ultimate reason why it requires more ancillary qubits is that the structure of the brickwork state is fixed. Due to the fixed structure, the equivalent circuits implemented by the brickwork state also have a fixed layout as shown in Fig. 5.

Even worse, since the structure of the brickwork state is fixed, it can achieve a CX gate only on two adjacent qubits. Although a CX gate on two nonadjacent qubits can be simulated by a sequence of CX gates on adjacent qubits, it inevitably consumes massive qubits. To be specific, consider a CX gate on qubits 1 and $n (\geq 3)$, denoted by $CX_{1,n}$; it can be implemented by $(4n - 8)$ CX gates on adjacent qubits. The concrete circuit is shown in Fig. 6, from which we can see that the depth of the circuit is $4n - 8$. Since there are at most $n/2$ concurrent CX gates in each layer of circuits, thus it can be verified that in the worst case the number of qubits is $O(n^{c+3})$. Clearly, an increase of two orders of magnitudes comparing to the ideal estimation $O(n^{c+1})$.

IV. SQUARE BRICKWORK STATE

In order to reduce the expenditure of ancillary qubits, we first construct a universal graph state based on the original brickwork state in [12], and we simply name it a *square brickwork state*.

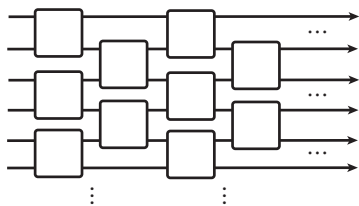


FIG. 5. Circuit layout implemented via the brickwork state, where each block denotes a unitary operator implemented by a basic unit of the brickwork state.

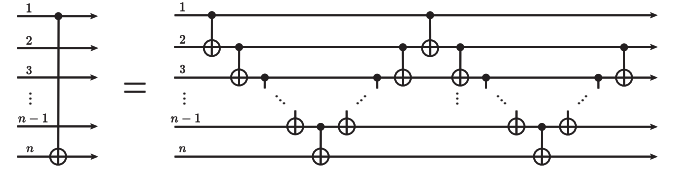


FIG. 6. $CX_{1,n}$ gate and its equivalent circuit consisting of CX gates on adjacent qubits.

Our main construction method is inspired by the work in [30], where the authors use the technique to introduce a verification mechanics in blind quantum computations. There are two kinds of operations in the procedure of the construction: *break* and *bridge* operations. Figuratively, the break operation is used to erase a vertex together with two edges connected to it, while the bridge operation is used to erase a vertex and then reconnect its two neighbor vertices; see Fig. 7. Such operations can be implemented by the following way: let $|\Phi\rangle$ be the state consisting of qubits k, l , for a break operation, first prepare the qubit i in state $|x_i\rangle$ where $x_i \in \{0, 1\}$, then measure it with any basis, and the postmeasurement state of qubits k, l will be $R_z(x_i\pi) \otimes R_z(x_i\pi)|\Phi\rangle$; for a bridge operation, first prepare the qubit i in state $|+_{x_i\pi}\rangle$ where $x_i \in \{0, 1\}$, then measure it with $M(\pi/2)$, and the postmeasurement state of qubits k, l will be $R_z[(-1)^{x_i \oplus s_i} \frac{\pi}{2}] \otimes R_z[(-1)^{x_i \oplus s_i} \frac{\pi}{2}] CZ|\Phi\rangle$, where $s_i \in \{0, 1\}$ is the measurement outcome of the qubit i . Apparently, ignoring the Z-rotation operators, the above state preparations and measurements exactly achieve the desired break and bridge operations.

A. Implementing the single-qubit basic gates

We first show that the three-qubit cluster depicted in Fig. 8 is universal for any single-qubit unitary operator. Specifically, it can be used to implement the basic gates H and T . From Fig. 2, we can infer that the equivalent circuit implemented by measuring the first two qubits of this three-qubit cluster is $R_x^\dagger(\theta_1)R_z^\dagger(\theta_0)$, where we ignore the redundant Pauli operators on the output qubit. Obviously, setting $\theta_0 = -\pi/4$ and $\theta_1 = 0$, we can obtain that the $\pi/8$ gate T . Note that up to a global phase $H \equiv R_z^\dagger(\frac{\pi}{2})R_x^\dagger(\frac{\pi}{2})R_z^\dagger(\frac{\pi}{2})$, so setting $\theta_0 = \theta_1 = \pi/2$, we can obtain the unitary operator such that $R_z(\pi/2)H$. As mentioned in Sec. II, any Z-rotation operator $R_z(\theta)$ can be corrected by the adaptive measurement. Thus, this three-qubit cluster is universal for any single-qubit unitary operator. Clearly, comparing with the original brickwork state shown in Fig. 3, the number of ancillary qubits for single-qubit gates shrinks by half.

We should mention that this three-qubit cluster state can be also used to implement more basic gates such as I, X, Y, Z, S, HSH, HTH , even their inverses (including T^\dagger).

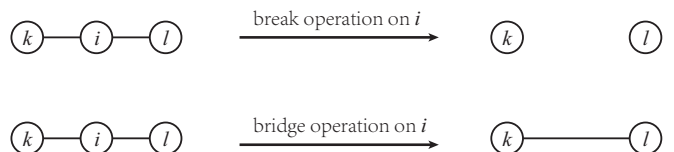


FIG. 7. Break operation and the bridge operation on qubit i .

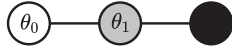


FIG. 8. Simple three-qubit cluster state.

In the view of computational complexity, introducing more basic gates can effectively reduce the depth of quantum circuits. For example, consider a unitary operator $U = HT^{\dagger}H$; if we use only H and T to implement it, then it will be $HTTTTTH$, which contains nine gates! As a result, implementing such U by the three-qubit cluster states requires $2 \times 9 = 18$ ancillary qubits. But, in fact, we can implement this unitary operator using only one three-qubit cluster, where we set $\theta_0 = 0$ and $\theta_1 = \pi/4$. Thus, introducing more basic gates in the gate set is indispensable. Finally, having inverses of the basic gates in the gate set is the basic requirement of the Solovay-Kitaev theorem [32].

B. Implementing the two-qubit basic gate

We next consider the case of two-qubit basic gates. Instead of the CX gate, we will use the CZ gate as the two-qubit basic gate. This can be done since we know that a CX gate can be decomposed into a CZ gate and two H gates. There are two reasons for this choice. First, from the point of view of the circuit design, the CZ gate is simpler than the CX gate since it is symmetric; that is, given a CZ gate on two qubits there is no need to specify the control and the target qubits. Second, the CZ gate in itself is one of basic operations in constructing a cluster state; thus using the CZ gate to describe a circuit is more fundamental than the CX gate. And, more importantly, unlike the single-qubit basic gates there is no need for ancillary qubits (and measurements) to implement a CZ gate, since it naturally exists in the structure of the cluster state! For example, suppose we want to execute a CX gate on two adjacent qubits; the circuit consists of four single-qubit gates and one CZ gate (see Fig. 9). Except the CZ gate, each single-qubit gate requires two ancillary qubits. As for the CZ gate, we can think it is embedded in the cluster state as a basic component before the measurements.

However, as we can see from the above example, the position of the CZ gate is exposed to both server and client. Thus, it cannot be directly used for the universal blind quantum computation. To plug this loophole, we make use of the aforementioned break and bridge operations. First, we construct a special seven-qubit cluster state, which consists of two three-qubit cluster states connected by an extra ancillary qubit between two output qubits (see Fig. 10). We have known that measuring the first two qubits on the top and bottom

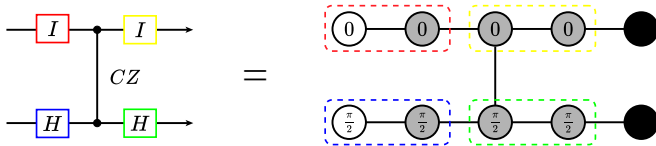


FIG. 9. CX gate decomposed as four single-qubit gates and one CZ gate, where we use different colors to denote those single-qubit gates. In the right-side cluster state the measurements implementing different single-qubit gates are highlighted with corresponding color.

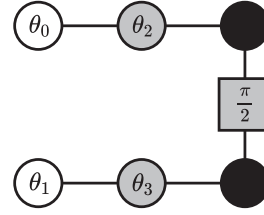


FIG. 10. Special seven-qubit cluster state, where the square denotes a non-input-and-output qubit whose state is taken from either $\{|0\rangle, |1\rangle\}$ or $\{|+\rangle, |-\rangle\}$ randomly.

lines amounts to applying a single-qubit basic gate on the top and bottom input qubits, respectively. In the case where we want to perform a CZ gate on the two output qubits, we perform a bridge operation on the ancillary qubit represented by the gray square. That is, we first initiate this qubit as $|+_{x\pi}\rangle$, then measure it with $M(\pi/2)$. Otherwise, we simply perform a break operation on that ancillary qubit. Specifically, we first initialize this qubit as $|x\rangle$ and then measure it also with $M(\pi/2)$. In the view of the server, the break and the bridge operations are indistinguishable, since the measurement bases in both operations are identical and more importantly $\{|0\rangle, |1\rangle\}$ and $\{|+\rangle, |-\rangle\}$ are completely indistinguishable, which is the security basis of most of quantum key distribution protocols [33]. Thus, the server cannot obtain any information about whether a CZ gate is performed on the output qubits. Note that the break and the bridge operations can be finished before the measurements on the three-qubit cluster states.

C. Construction of square brickwork states

We now use the previous seven-qubit cluster state as the basic unit to construct a square brickwork state for the universal blind quantum computation, whose structure is depicted in Fig. 11, where each circle denotes a qubit $|+\delta\rangle$, $\delta \in \{0, \frac{\pi}{4}, \frac{2\pi}{4}, \dots, \frac{7\pi}{4}\}$, while each square denotes a qubit $|x\rangle$ or $|+_{x\pi}\rangle$, $x \in \{0, 1\}$; the ancillary qubits represented by gray squares are used to perform the break and the bridge operations at the beginning of the computation.

In the following discussion, we use the same fashion as described in Sec. II to index each qubit represented by circles. On the basis of that, we denote the ancillary qubit represented by a square between qubits i and $i + 1$ as $(i, i + 1)$, whose state is written as $|x_{i+1}^i\rangle$ or $|+_{x_{i+1}^i\pi}\rangle$. It is easy to check

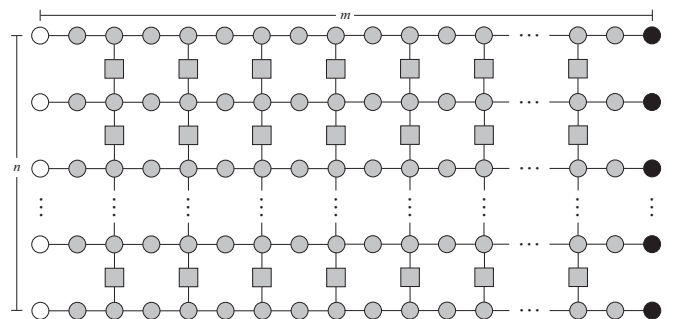


FIG. 11. Structure of the square brickwork state, where n is the number of input qubits and $m \equiv 1 \pmod{2}$ and $m \geq 5$.

Algorithm I. Constructing a square brickwork state.

Require: $\mathbb{M}_{\text{comp}} = (\theta, f, \mathcal{B})$
Ensure: A square brickwork state with n input qubits and m columns

- 1: **for** $i \leftarrow 0, mn - 1$ **do**
- 2: \mathcal{C} generates $|+\delta_i\rangle$ as the qubit i , where $\delta_i \in_R \{0, \frac{\pi}{4}, \dots, \frac{7\pi}{4}\}$
- 3: **end for**
- 4: **for** $k \leftarrow 1, n - 1$ **do**
- 5: **for** $l \leftarrow 1, \frac{m-3}{2}$ **do**
- 6: Let $i = 2ln + k - 1$
- 7: **if** $\mathcal{B}_{i+1}^i = Z$ **then**
- 8: \mathcal{C} generates $|x_{i+1}^i\rangle$ as the qubit $(i, i + 1)$, where $x_{i+1}^i \in_R \{0, 1\}$
- 9: **else**
- 10: \mathcal{C} generates $|+_{x_{i+1}^i}\rangle$ as the qubit $(i, i + 1)$, where $x_{i+1}^i \in_R \{0, 1\}$
- 11: **end if**
- 12: **end for**
- 13: **end for**
- 14: \mathcal{C} sends all qubits to \mathcal{S}
- 15: **for** $i \leftarrow 0, mn - n - 1$ **do**
- 16: \mathcal{S} performs a CZ gate on qubits i and $i + n$
- 17: **end for**
- 18: **for** $k \leftarrow 1, n - 1$ **do**
- 19: **for** $l \leftarrow 1, \frac{m-3}{2}$ **do**
- 20: Let $i = 2ln + k - 1$
- 21: \mathcal{S} performs $\text{CZ}_{i,(i+1)}$ and $\text{CZ}_{i+1,(i+1+n)}$
- 22: **end for**
- 23: **end for**

that the number i must satisfy $i = 2ln + k - 1$ where $k \in \{1, 2, \dots, n - 1\}$ and $l \in \{1, 2, \dots, \frac{m-3}{2}\}$ [hereafter, when we mention the qubit $(i, i + 1)$, it must satisfy this condition]. Furthermore, we define an indicator $\mathcal{B}_{i+1}^i \in \{Z, X\}$ for each qubit $(i, i + 1)$. If $\mathcal{B}_{i+1}^i = Z$, then the qubit $(i, i + 1)$ is prepared in state $|x_{i+1}^i\rangle$, otherwise $|+_{x_{i+1}^i}\rangle$. In this work we implicitly assume that there exists a polynomial algorithm which given a computation outputs a corresponding measurement pattern $\mathbb{M}_{\text{comp}} = (\theta, f, \mathcal{B})$ on the brickwork state, where the extra \mathcal{B} contains all the indicators defined above. Once the measurement pattern \mathbb{M}_{comp} is determined, one can easily construct the square brickwork state for the computation; see Algorithm I.

V. UNIVERSAL BLIND QUANTUM COMPUTATION USING SQUARE BRICKWORK STATE

Given a measurement pattern and the corresponding square brickwork state, the computation can be roughly divided into three steps.

Step 1: The server performs a measurement $M(\pi/2)$ on each ancillary qubit $(i, i + 1)$ and sends the result s_{i+1}^i to the client. We know that such an operation may result in a Z rotation on the qubits i and $i + 1$. Thus, the client needs to update the rotation angles δ_i and δ_{i+1} . The concrete updating rule can be expressed as

$$\delta_u := \delta_u + \delta_{Z, \mathcal{B}_{i+1}^i} x_{i+1}^i \pi + \delta_{X, \mathcal{B}_{i+1}^i} (-1)^{x_{i+1}^i \oplus s_{i+1}^i} \frac{\pi}{2}, \quad (5)$$

where $u \in \{i, i + 1\}$ and $\delta_{Z,Z} = \delta_{X,X} = 1$, otherwise $\delta_{Z,X} = \delta_{X,Z} = 0$.

Step 2: As with the original UBQC protocol, for each nonoutput qubit i the client first generates a random bit $r_i \in_R \{0, 1\}$, then computes the corrected measurement angle ϕ_i as

$$\phi_i = (-1)^{s_{i-n}} \theta_i + \sum_{\substack{j: f_j \in N_G(i) \\ j \neq i}} s_j \pi + \delta_i + r_i \pi + \Delta_i, \quad (6)$$

where the first two terms are used to correct the Pauli operators, the third term is used to compensate the Z rotation accumulated in Step 1, and the fourth term is used to encrypt the measurement outcome s_i , while the last term is also a compensation for the Z rotation caused by the Hadamard gate and defined as

$$\Delta_i = \text{odd}(i) \delta_{\frac{\pi}{2}, \theta_{i-2n}} \delta_{\frac{\pi}{2}, \theta_{i-n}} (-1)^{s_{i-n}} \frac{\pi}{2}, \quad (7)$$

where $\text{odd}(i) = 1$ if the qubit i is located in an odd-numbered column of the square brickwork state, otherwise $\text{odd}(i) = 0$. This product $\text{odd}(i) \delta_{\frac{\pi}{2}, \theta_{i-2n}} \delta_{\frac{\pi}{2}, \theta_{i-n}}$ captures a condition that whether or not the basic gate implemented by measuring the qubits $i - 2n$ and $i - n$ is a Hadamard gate, where i satisfies that $i = 2ln + k - 1$, i.e., $\text{odd}(i) = 1$. If so, we need to correct a Z rotation $R_z[(-1)^{s_{i-n}} \frac{\pi}{2}]$. In particular, we specify that $\Delta_i = 0$ for $i \in \{0, 1, \dots, n - 1\}$. Note that the angle Δ_i is related to the measurement outcome s_{i-n} , this is because measuring the qubit $i - n$ will introduce a random Pauli $X^{s_{i-n}}$ on qubit i . It is easy to check that for any $\theta \in [0, 2\pi)$ and $s \in \{0, 1\}$ we have $X^s R_z(\theta) X^s = R_z[(-1)^s \theta]$. Thus, the Z -rotation operator caused by implementing a Hadamard gate is $R_z[(-1)^{s_{i-n}} \frac{\pi}{2}]$. Once determining ϕ_i via Eq. (6), \mathcal{C} sends it to \mathcal{S} , then \mathcal{S} measures the qubit i with $M(\phi_i)$, obtaining the measurement outcome b_i and sending it to \mathcal{C} . Finally, \mathcal{C} computes the measurement outcome of $M(\theta_i)$, i.e., $s_i = b_i \oplus r_i$.

Step 3: For a quantum output, we can see that each output qubit i is encrypted by the operator $R_z(\delta_i + \Delta'_i) Z^{s_{i-2n}} X^{s_{i-n}}$, where $i \in \{(m - 1)n, \dots, mn - 1\}$ and Δ'_i is defined as

$$\Delta'_i = \delta_{\frac{\pi}{2}, \theta_{i-2n}} \delta_{\frac{\pi}{2}, \theta_{i-n}} (-1)^{s_{i-n}} \frac{\pi}{2}. \quad (8)$$

Note that the expression of Δ'_i is slightly different from the one of Δ_i , because all output qubits must be located in an odd-numbered layer [recall that $m \equiv 1 \pmod{2}$], which means the value of $\text{odd}(i)$ must be 1 for each output qubit i . According to Eq. (8), the right output state can be obtained by performing the inverse of $R_z(\delta_i + \Delta'_i) Z^{s_{i-2n}} X^{s_{i-n}}$ on each output qubit i .

Step 3: For a classical output, we assume that the all output qubits are measured in the Z basis. In this case the Z -rotation operator (including the Pauli Z gate) will not affect the distribution of the computation output. Thus, the correct output bit for the qubit i will be $s_{i-n} \oplus s_i$.

In summary, the complete procedure of the universal blind quantum computation protocol using a square brickwork state is described in Algorithm II.

Example

For comparison with the original UBQC protocol, we reconsider the example circuit U_{swaps} shown in Fig. 4(a). First, we express U_{swaps} using I, H , and CZ, then translate it into an equivalent measurement pattern, by which we can obtain the square brickwork state implementing U_{swaps} (see Fig. 12,

Algorithm II. Universal blind quantum computation via a square brickwork state.

Require: $\mathbb{M}_{\text{comp}} = (\theta, f, \mathcal{B})$ // θ and \mathcal{B} are blind to \mathcal{S}

Ensure: $y = y_1 y_2 \dots y_n \in \{0, 1\}^n$ // for a classical output $|out\rangle = U_{\text{comp}}|+\rangle^{\otimes n}$ // for a quantum output

- 1: Set up a square brickwork state for $\mathbb{M}_{\text{comp}} = (\theta, f, \mathcal{B})$ by calling Algorithm I.
- 2: For each qubit $(i, i + 1)$, \mathcal{S} measures it with $M(\frac{\pi}{2})$, obtaining the measurement outcome s_{i+1}^i and sending it to \mathcal{C} . Then \mathcal{C} updates the values of δ_i and δ_{i+1} according to Eq. (5).
- 3: For each nonoutput qubit i , \mathcal{C} generates $r_i \in_R \{0, 1\}$ and computes the measurement angle ϕ_i according to Eq. (6), then sends ϕ_i to \mathcal{S} . After that, \mathcal{S} measures the qubit i with $M(\phi_i)$, obtaining the measurement outcome, denoted by b_i , and sending to \mathcal{C} . Finally, \mathcal{C} computes $s_i = b_i \oplus r_i$.
- 4: For a quantum output, \mathcal{C} retrieves the encrypted output state from \mathcal{S} , then decrypts each qubit i by $X^{s_i-n} Z^{s_i-2n} R_z^{\Delta'_i}(\delta_i + \Delta'_i)$, where Δ'_i is obtained according to Eq. (8).
- 5: For a classical output, \mathcal{S} measures each output qubit i in Z basis and sends the measurement outcome s_i to \mathcal{C} , where $i = (m - 1)n + k - 1$ and $k \in \{1, 2, \dots, n\}$. Then \mathcal{C} computes $y_k = s_{i-n} \oplus s_i$.

where 0/1 denotes a random qubit taken from $\{|0\rangle, |1\rangle\}$ and $+/-$ denotes a random qubit taken from $\{|+\rangle, |-\rangle\}$. From the picture, we can see that the number of qubits in the square brickwork state is $4 \times 8 + 9 = 41$ (not including the input qubits), which is much smaller than the case in the original UBQC protocol [see Fig. 4(c)]. It should be mentioned that a swap gate can be more efficiently constructed via some certain graph state. For example, according to the original work of Raussendorf *et al.*, a swap gate can be implemented by the eight-qubit graph state as shown in Fig. 13, where all nonoutput qubits are measured in the X basis. This procedure can be viewed as a two-dimensional (2D) version of identity gate in the MBQC model [14]. Although this graph state is more efficient than the graph state shown in Fig. 12(b), it cannot be used to implement a swap gate in square brickwork states. The main reason is that this eight-qubit cluster state cannot be constructed by the sequence of seven-qubit cluster state shown in Fig. 10.

This simple example also reveals a good property of the square brickwork state. That is, in the square brickwork state the layout of basic gates is no longer fixed as shown in Fig. 5. In general, this will make it easier to translate U_{comp} into \mathbb{M}_{comp} . For example, if the computation circuit U_{comp} consists only of single-qubit basic gates and CZ gates on adjacent

qubits, then the measurement angles on each row can be trivially determined.

However, it should be mentioned that the square brickwork state cannot directly implement a CZ gate on nonadjacent qubits. As with the case in the original brickwork state, a CZ gate on two nonadjacent qubits must be converted into a sequence of CZ gates on adjacent qubits, which will inevitably consume plenty of ancillary qubits!

VI. MORE EFFICIENT BRICKWORK STATES

A. Hyper-brickwork state

In order to decrease the number of ancillary qubits further, we further improve the square brickwork state so that it can implement directly a CZ gate on two nonadjacent qubits. This basic idea is very simple: at each odd-numbered column (except the first column and the last column), we introduce more break and bridge operations among the rows. We name this improved graph state the *hyper-brickwork state*, whose construction can be described as follows.

First, we prepare an n -row and m -column qubits. For each row, we perform a CZ gate on two adjacent qubits. Then, for each $i = 2ln + k - 1$, where $l \in \{1, 2, \dots, \frac{m-3}{2}\}$ and $k \in \{1, 2, \dots, n - 1\}$, we define a generalized indicator $\mathcal{B}_j^i \in \{Z, X\}$, where $i + 1 \leq j \leq i + n - k$. For each \mathcal{B}_j^i , we prepare a qubit, labeled as (i, j) , in state $|x_j^i\rangle$ or $|+_{x_j^i}\pi\rangle$, where $x_j^i \in_R \{0, 1\}$. Specifically, if $\mathcal{B}_j^i = Z$, then the state of the qubit (i, j) is $|x_j^i\rangle$, otherwise $|+_{x_j^i}\pi\rangle$. Finally, we perform a CZ gate on qubits i and (i, j) , and a CZ gate on qubits j and (i, j) . The complete procedure of constructing a hyper-brickwork state is given in Algorithm III.

By the hyper-brickwork state, we can easily establish or eliminate a CZ gate on any two qubits in the same column. To make it more intuitive, we consider a simple example. Suppose $n = 4$ and we want to perform a CZ gate on qubits 0

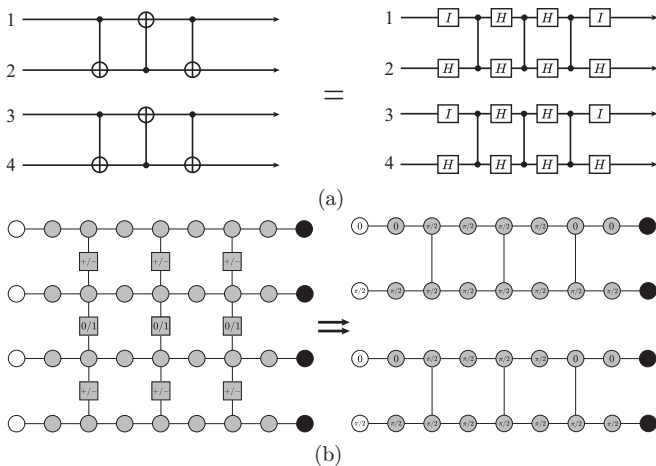


FIG. 12. (a) Two equivalent expressions of the circuit U_{swaps} ; (b) the square brickwork state implementing U_{swaps} and the corresponding measurement angles.

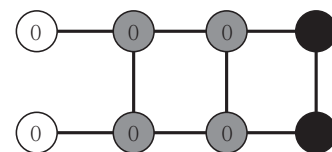


FIG. 13. Eight-qubit graph state that can be used to implement the swap gate by measuring all nonoutput qubits in X basis.

Algorithm III. Constructing a hyper-brickwork state.

Require: $\mathbb{M}_{\text{comp}} = (\theta, f, \mathcal{B}_{\text{hyper}})$ // $\mathcal{B}_{\text{hyper}}$ contains all indicators $\{\mathcal{B}_j^i\}$
Ensure: A hyper-brickwork state with n input qubits and m columns

- 1: **for** $i \leftarrow 0, mn - 1$ **do**
- 2: \mathcal{C} generates $|+\delta_i\rangle$ as the qubit i , where $\delta_i \in_{\mathcal{R}} \{0, \frac{\pi}{4}, \dots, \frac{7\pi}{4}\}$
- 3: **end for**
- 4: **for** $k \leftarrow 1, n - 1$ **do**
- 5: **for** $l \leftarrow 1, \frac{m-3}{2}$ **do**
- 6: Let $i = 2ln + k - 1$
- 7: **for** $j \leftarrow i + 1, i + n - k$ **do**
- 8: **if** $\mathcal{B}_j^i = Z$ **then**
- 9: \mathcal{C} generates $|x_j^i\rangle$ as the qubit (i, j) , where $x_j^i \in_{\mathcal{R}} \{0, 1\}$
- 10: **else**
- 11: \mathcal{C} generates $|+x_j^i\rangle$ as the qubit (i, j) , where $x_j^i \in_{\mathcal{R}} \{0, 1\}$
- 12: **end if**
- 13: **end for**
- 14: **end for**
- 15: **end for**
- 16: \mathcal{C} sends all qubits to \mathcal{S}
- 17: **for** $i \leftarrow 0, mn - n - 1$ **do**
- 18: \mathcal{S} performs a CZ gate on qubits i and $i + n$
- 19: **end for**
- 20: **for** $k \leftarrow 1, n - 1$ **do**
- 21: **for** $l \leftarrow 1, \frac{m-3}{2}$ **do**
- 22: Let $i = 2ln + k - 1$
- 23: **for** $j \leftarrow i + 1, i + n - k$ **do**
- 24: \mathcal{S} performs $\text{CZ}_{i,(i,j)}$ and $\text{CZ}_{j,(i,j)}$
- 25: **end for**
- 26: **end for**
- 27: **end for**

and 3. Clearly, we need to prepare six ancillary qubits, where the qubit (0,3) is taken from $\{|+\rangle, |-\rangle\}$, while the others are taken from $\{|0\rangle, |1\rangle\}$. Then, measuring each ancillary qubit with $M(\pi/2)$, it will create a CZ gate only on qubits 0 and 3 (see Fig. 14).

From this example, we can infer that for an n -qubit state, in order to make sure that a break operation or a bridge operation can be implemented on any two qubits, we need only $\frac{n(n-1)}{2}$ ancillary qubits in total. This makes a remarkable decrease in the number of ancillary qubits, since we saw in Sec. III that a CX gate on qubits 1 and n will consume $4n(4n-8) = 16n^2 - 32n$ ancillary qubits! More importantly, the hyper-brickwork state can implement at most $n(n-1)/2$ concurrent CZ gates in each column, which will also save many ancillary qubits. To see that, we use the syndrome circuit

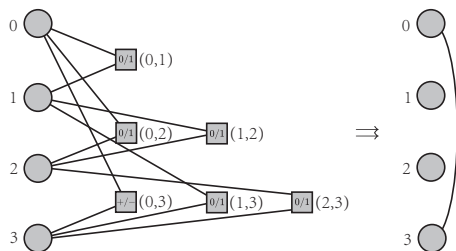


FIG. 14. Implementing a CZ gate on qubits 0 and 3 by a sequence of break and bridge operations on ancillary qubits.

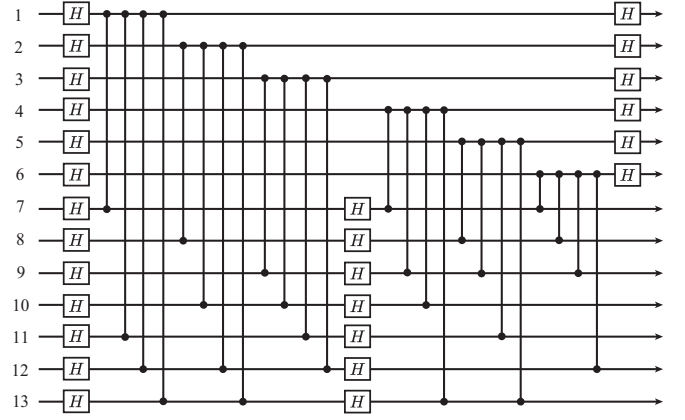


FIG. 15. Quantum circuit for measuring the generators of the seven-qubit Steane code [34]. This circuit can be implemented by an $n = 13$ and $m = 7$ hyper-brickwork state.

of the Steane code as a qualitative illustration (see Fig. 15). It is not hard to check that to implement this circuit by the original brickwork state requires around 28 496 ancillary qubits, but the hyper-brickwork state needs only 234 ancillary qubits. According to the circuit in Fig. 15, constructing the hyper-brickwork state for a seven-qubit Steane code measurement circuit is simple. However, expressing this hyper-brickwork state in the 2D plane is somewhat difficult since the qubits in each odd-numbered column (except the first and the last column) form a local dotted-complete graph state. Given an arbitrary graph G , the dotted-complete graph \tilde{G} is defined by replacing every edge in G with a new vertex connected to the two vertices originally joined by that edge [30]. Inspired by the work [30], we can illustrate the hyper-brickwork state for the seven-qubit Steane code measurement circuit. The illustration consists of two parts. The first part is the backbone of the hyper-brickwork state (see Fig. 16), which is basically the same with the square brickwork state, where the qubits within red-dotted boxes consist of 13-qubit dotted-complete graph states. The second part is the illustration of the 13-qubit dotted-complete graph cluster (see Fig. 17). Determining the information of the indicator $\mathcal{B}_{\text{hyper}}$ is easy but tedious, and for the sake of brevity we will not explain it here. According to Fig. 16 and Fig. 17, we can conclude that the number of qubits except the output qubits is 234. In summary, we conclude that for an n -qubit circuit with a depth of $O(n^c)$, the number of qubits for the hyper-brickwork state is no more than $2nO(n^c) + \frac{n(n-1)}{2}O(n^c) = O(n^{c+2})$.

The universal blind quantum computation using a hyper-brickwork state is similar to II. We need to adjust only the first two steps. In the first step, we construct the hyper-brickwork state by calling Algorithm III. In the second step, the server measures each qubit (i, j) and sends the measurement outcome s_j^i to the client, then the client updates the rotation angle δ_u where $u \in \{i, j\}$ by the following rule:

$$\delta_u := \delta_u + \delta_{Z, \mathcal{B}_j^i} x_j^i \pi + \delta_{X, \mathcal{B}_j^i} (-1)^{x_j^i \oplus s_j^i} \frac{\pi}{2}. \quad (9)$$

The rest steps are identical with Algorithm II.

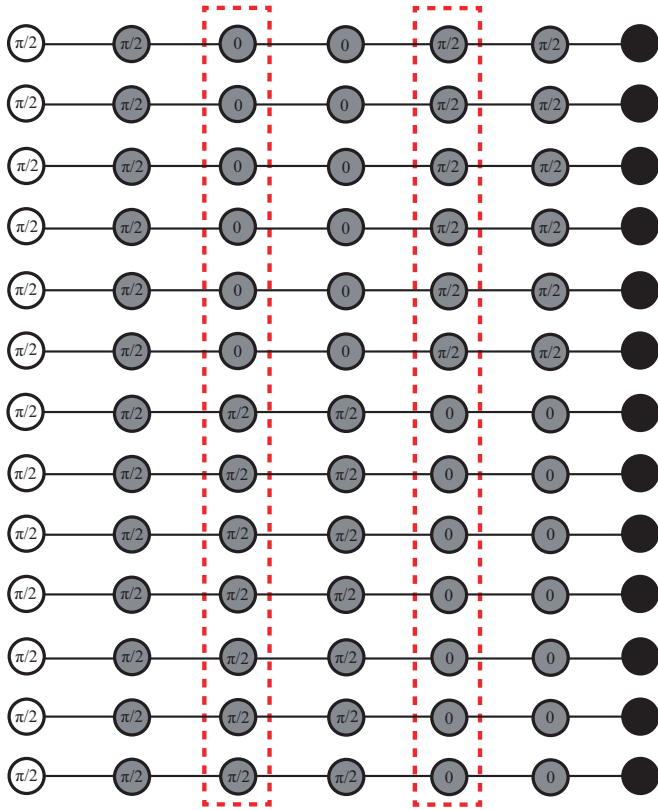


FIG. 16. Backbone of the hyper-brickwork state for seven-qubit Steane code measurement circuit, where the qubits within dotted-line boxes denote 13-qubit dotted-complete graph states.

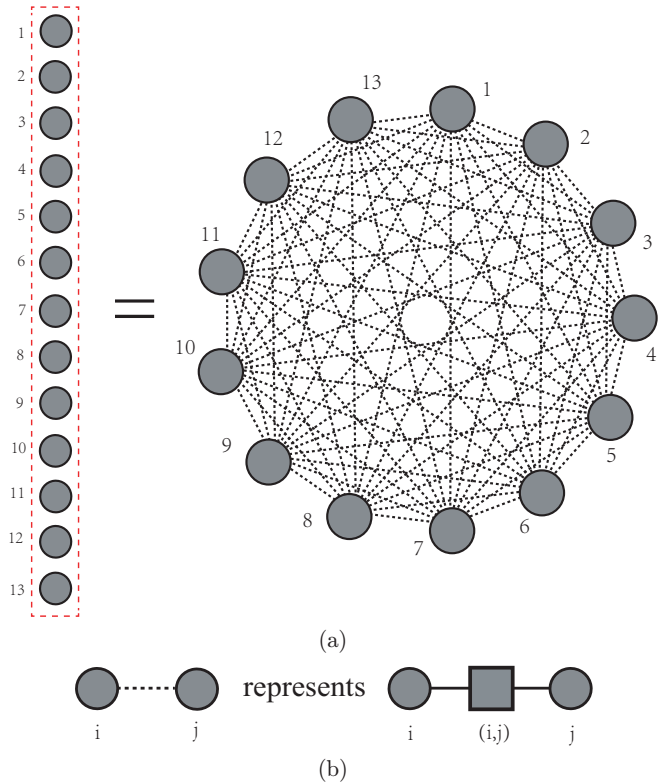


FIG. 17. (a) Thirteen-qubit dotted-complete graph state expressing in the 2D plane. (b) Dotted line between two vertices i and j denotes that they are connected via an extra vertex (i, j) .

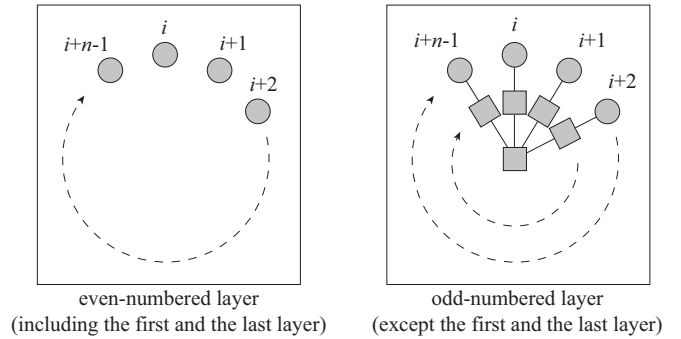


FIG. 18. Structures of each layer in circular brickwork states.

B. Circular brickwork state

Although the hyper-brickwork state is a more efficient universal graph state, from Fig. 17 we can see that its structure is somewhat complicated. We know that cluster states can be created efficiently in any system with a quantum Ising-type interaction (at very low temperatures) between two-state particles in a lattice configuration [35]. If we restrict the quantum Ising-type interactions on two neighbor qubits, then it is impossible to construct a hyper-brickwork state in 2D lattices.

In the rest of this section, we consider a 3D brickwork state named by *circular brickwork state*, which can avoid the difficulty of implementing the quantum Ising-type interactions on nonadjacent qubits. Meanwhile, it can also decrease the number of ancillary qubits into the level of $O(n^{c+1})$. However, in order to reach this goal, we need to assume that each layer of a quantum circuit contains at most one CZ gate. Fortunately, in most cases practical quantum circuits satisfy or approximately satisfy this condition, for example, the quantum Fourier transform.

The procedure of constructing a circular brickwork state is as follows. First, we prepare nm single qubits, each of which is in state $|+\delta\rangle$ and $\delta \in_R \{0, \pi/4, \dots, 7\pi/4\}$. Then we group those qubits into m layers, where each layer contains n qubits arranged in a circular manner. For each two qubits that are located in neighboring layers and the same radial direction, we perform a CZ gate on them. Next, for each odd-numbered layer (except the input and the output layers), we prepare additional $n + 1$ ancillary qubits, where one qubit is located in the center of the circle while the other n qubits encircle it. Those qubits are used for break and bridge operations, thus they are prepared as $|x\rangle$ or $|+\pi\rangle$. Finally, for each additional ancillary qubit, in the radial direction we perform a CZ gate on it and its each neighboring qubit. See Fig. 18 for the concrete structure of each layer.

Finally, we give a concrete example to make the structure more intuitive. Suppose we are given eight qubits labeled as $1, 2, \dots, 8$ and we want to implement a CZ gate on qubits 2 and 6. According to the configuration of the odd-numbered layers shown in Fig. 18, we need to prepare only nine ancillary qubits and entangle all qubits by the quantum Ising-type interactions on part of the adjacent qubits. The concrete structure of this 17-qubit cluster state is shown in Fig. 19, where the ancillary qubits connecting qubits 2 and 6 are prepared in $|+\rangle$ or $|-\rangle$ randomly, while the others are prepared in $|0\rangle$

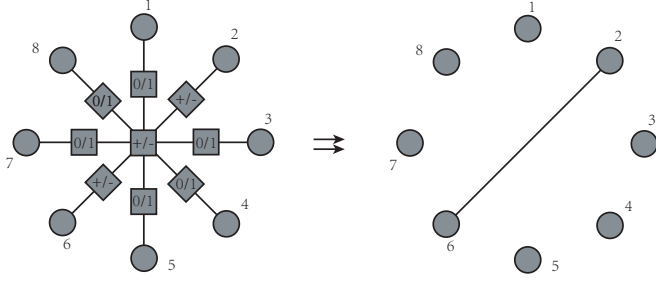


FIG. 19. Implementing a CZ gate on qubits 2 and 6 using nine ancillary qubits.

or $|1\rangle$ randomly. Clearly, it follows from the picture that measuring all ancillary qubits with $M(\pi/2)$ will result in a CZ gate on qubits 2 and 6 only. Of course, it should be mentioned that those measurements will introduce random Z rotations on qubits 1, 2, \dots , 8. Nevertheless, those Z rotations can be easily corrected by the adaptive measurements on those qubits. Determining the specific rotation angle of Z rotation on each qubit is tedious, but the basic computation procedure is straightforward as we described above, and here we will not deal with it any further. According to this example, we can see that given an n -qubit state, we need to prepare $n + 1$ ancillary qubits in total. More generally, for an n -qubit circuit with a depth of $O(n^c)$, the number of ancillary qubits for constructing the circular brickwork state is no more than $2nO(n^c) + (n + 1)O(n^c) = O(n^{c+1})$. Again, we emphasize that the quantum circuit must satisfy the condition that each layer of the circuit contains no more than one CZ gate. If not, we need to first convert it into the expected form. The procedure is simple, and it may cause the depth of the circuit increases to the level of $O(n^{c+1})$. Thus, in the worst case the total number of qubits is no more than $O(n^{c+2})$, which is exactly same with the upper bound of the hyper-brickwork state.

VII. CONCLUSION

In this work we investigated the UBQC protocol proposed by Broadbent *et al.* and constructed three improved brickwork states. With those improved brickwork states, we propose a class of efficient UBQC protocols which can reduce the client-side qubit consumption in different degrees. For the first improved brickwork state, the square brickwork state, to implement a single-qubit gate requires only that the client prepares two ancillary qubits. For the second improved brickwork state, the hyper-brickwork state, which is a variant of the square brickwork state, it can efficiently implement a nonadjacent two-qubit basic gate. The last improved brickwork state, called the circular brickwork state, is a further variant of the square brickwork state, which utilizes the property of the 3D space and can efficiently reduce the difficulty of construction of the brickwork states.

Our work considers only a universal blind quantum computation protocol where the server is *honest*, which means the server follows the protocol. However, in real situations the untrusted server may not follow the protocol honestly. To detect such a malicious server, we need to introduce a verifi-

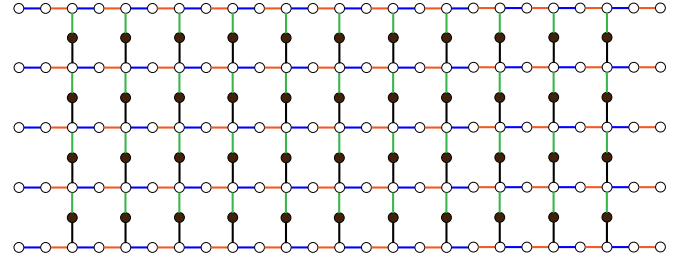


FIG. 20. Square brickwork architecture, where blue lines represent the first layer of gates to act, orange lines represent the second layer, green lines represent the third layer, and black lines represent the final layer.

cation mechanics in our protocol. Indeed, verifiable quantum computation is one research focus in quantum computation theory [36], and there exist many related works [30,37–41]. A simple method is that we can prepare some extra ancillary qubits, then perform a random quantum circuit on them, and finally verify the quantum computation by checking the measurement outcomes of those extra ancillary qubits. The extra ancillary qubits should be randomly placed in the input qubits. Due to the blindness, the server cannot learn the information about the position of the extra ancillary qubits and the random quantum circuit. The random quantum circuit needs to be easily verified by a classical computer, for example, permutation circuits.

Finally, we note that a recent work proposed by Napp *et al.* indicated that random shallow 2D quantum circuits can be efficiently simulated on classical computers [42]. Obviously, any cluster state in the 2D plane can be viewed as a 2D quantum circuit with a certain depth. For example, a square brickwork state with any scale can be viewed as a 2D quantum circuit with a depth 4, as shown in Fig. 20, where white circles represent the qubits that are used to implement the quantum circuit while black circles represent the qubits that are used to implement break and bridge operations. Napp *et al.* [42] proposed a classical algorithm, the so-called space-evolving block decimation (SEBD), which can be efficiently simulated the extended brickwork states with a depth 3. We present rigorous evidence that the cluster states with the extended brickwork structure are classically simulable. Although the square brickwork states are not compatible with the extended brickwork structure defined in [42], they can be converted into the extended brickwork structure efficiently, since we know that the square brickwork states have a variable structure after performing break and bridge operations. Thus, we believe our improved brickwork states in this paper will bring insights for the SEBD algorithm.

ACKNOWLEDGMENTS

This project was supported by the National Natural Science Foundation of China (Grants No. 61372076, No. 61971348, and No. 62001351) and the Foundation of the Shaanxi Key Laboratory of Information Communication Network and Security (ICNS201802).

- [1] D. Deutsch, Quantum theory, the Church-Turing principle and the universal quantum computer, *Proc. R. Soc. London A* **400**, 97 (1985).
- [2] D. P. DiVincenzo, Quantum computation, *Science* **270**, 255 (1995).
- [3] J. Preskill, Quantum computing 40 years later, *Feynman Lectures on Computation* (CRC Press, New York, 2023), pp. 193–244.
- [4] D. Leichle, L. Music, E. Kashefi, and H. Ollivier, Verifying BQP computations on noisy devices with minimal overhead, *PRX Quantum* **2**, 040302 (2021).
- [5] J. F. Fitzsimons, Private quantum computation: An introduction to blind quantum computing and related protocols, *npj Quantum Inf.* **3**, 23 (2017).
- [6] P. Arrighi and L. Salvail, Blind quantum computation, *Int. J. Quantum. Inform.* **04**, 883 (2006).
- [7] A. M. Childs, Secure assisted quantum computation, *Quantum Inf. Comput.* **5**, 456 (2005).
- [8] A. Broadbent, Delegating private quantum computations, *Can. J. Phys.* **93**, 941 (2015).
- [9] X. Tan and X. Zhou, Universal half-blind quantum computation, *Ann. Telecommun.* **72**, 589 (2017).
- [10] W. Liu, Z. Chen, J. Liu, Z. Su, and L. Chi, Full-blind delegating private quantum computation, *Comput. Mater. Continua* **56**, 211 (2018).
- [11] S. Ma, C. Zhu, M. Nie, D. Quan, and C. Pei, Secure delegated quantum computation based on Z-rotation encryption, *Europhys. Lett.* **137**, 38001 (2022).
- [12] A. Broadbent, J. Fitzsimons, and E. Kashefi, Universal blind quantum computation, in *2009 50th Annual IEEE Symposium on Foundations of Computer Science* (IEEE, 2009), pp. 517–526.
- [13] A. Chi-Chih Yao, Quantum circuit complexity, in *Proceedings of 1993 IEEE 34th Annual Foundations of Computer Science* (1993), pp. 352–361.
- [14] R. Raussendorf and H. J. Briegel, A one-way quantum computer, *Phys. Rev. Lett.* **86**, 5188 (2001).
- [15] R. Raussendorf, D. E. Browne, and H. J. Briegel, Measurement-based quantum computation on cluster states, *Phys. Rev. A* **68**, 022312 (2003).
- [16] H. J. Briegel, D. E. Browne, W. Dür, R. Raussendorf, and M. Van den Nest, Measurement-based quantum computation, *Nat. Phys.* **5**, 19 (2009).
- [17] D. E. Deutsch, Quantum computational networks, *Proc. R. Soc. London A* **425**, 73 (1989).
- [18] T. Morimae, Continuous-variable blind quantum computation, *Phys. Rev. Lett.* **109**, 230502 (2012).
- [19] T. Morimae and K. Fujii, Blind topological measurement-based quantum computation, *Nat. Commun.* **3**, 1036 (2012).
- [20] T. Sueki, T. Koshihara, and T. Morimae, Ancilla-driven universal blind quantum computation, *Phys. Rev. A* **87**, 060301(R) (2013).
- [21] V. Giovannetti, L. Maccone, T. Morimae, and T. G. Rudolph, Efficient universal blind quantum computation, *Phys. Rev. Lett.* **111**, 230501 (2013).
- [22] T. Morimae, V. Dunjko, and E. Kashefi, Ground state blind quantum computation on AKLT state, *Quantum Inf. Comput.* **15**, 200 (2015).
- [23] A. Gheorghiu, E. Kashefi, and P. Wallden, Robustness and device independence of verifiable blind quantum computing, *New J. Phys.* **17**, 083040 (2015).
- [24] X. Zhang, W. Luo, G. Zeng, J. Weng, Y. Yang, M. Chen, and X. Tan, A hybrid universal blind quantum computation, *Info. Sci.* **498**, 135 (2019).
- [25] Q. Xu, X. Tan, and R. Huang, Improved resource state for verifiable blind quantum computation, *Entropy* **22**, 996 (2020).
- [26] Y.-F. Jiang, K. Wei, L. Huang, K. Xu, Q.-C. Sun, Y.-Z. Zhang, W. Zhang, H. Li, L. You, Z. Wang, H.-K. Lo, F. Xu, Q. Zhang, and J.-W. Pan, Remote blind state preparation with weak coherent pulses in the field, *Phys. Rev. Lett.* **123**, 100503 (2019).
- [27] A. Mantri, C. A. Pérez-Delgado, and J. F. Fitzsimons, Optimal blind quantum computation, *Phys. Rev. Lett.* **111**, 230502 (2013).
- [28] V. Dunjko and E. Kashefi, Blind quantum computing with two almost identical states, [arXiv:1604.01586](https://arxiv.org/abs/1604.01586) [quant-ph].
- [29] T. Morimae and K. Fujii, Blind quantum computation protocol in which Alice only makes measurements, *Phys. Rev. A* **87**, 050301(R) (2013).
- [30] J. F. Fitzsimons and E. Kashefi, Unconditionally verifiable blind quantum computation, *Phys. Rev. A* **96**, 012303 (2017).
- [31] R. Jozsa, An introduction to measurement based quantum computation, NATO Science Series, III: Computer and Systems Sciences. Quantum Information Processing-From Theory to Experiment **199**, 137 (2006).
- [32] C. M. Dawson and M. A. Nielsen, The Solovay-Kitaev algorithm, [arXiv:quant-ph/0505030](https://arxiv.org/abs/quant-ph/0505030) [quant-ph].
- [33] H.-K. Lo and H. F. Chau, Unconditional security of quantum key distribution over arbitrarily long distances, *Science* **283**, 2050 (1999).
- [34] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information: 10th Anniversary Edition* (Cambridge University Press, Cambridge, 2010).
- [35] H. J. Briegel and R. Raussendorf, Persistent entanglement in arrays of interacting particles, *Phys. Rev. Lett.* **86**, 910 (2001).
- [36] A. Gheorghiu, T. Kapourniotis, and E. Kashefi, Verification of quantum computation: An overview of existing approaches, *Theory Comput. Syst.* **63**, 715 (2019).
- [37] M. Hayashi and T. Morimae, Verifiable measurement-only blind quantum computing with stabilizer testing, *Phys. Rev. Lett.* **115**, 220502 (2015).
- [38] D. Aharonov, M. Ben-Or, E. Eban, and U. Mahadev, Interactive proofs for quantum computations, [arXiv:1704.04487](https://arxiv.org/abs/1704.04487) [quant-ph].
- [39] J. F. Fitzsimons, M. Hajdušek, and T. Morimae, *Post hoc* verification of quantum computation, *Phys. Rev. Lett.* **120**, 040501 (2018).
- [40] A. Broadbent, How to verify a quantum computation, *Theory Comput.* **14**, 1 (2018).
- [41] Q. Xu, X. Tan, R. Huang, and M. Li, Verification of blind quantum computation with entanglement witnesses, *Phys. Rev. A* **104**, 042412 (2021).
- [42] J. C. Napp, R. L. La Placa, A. M. Dalzell, F. G. S. L. Brandão, and A. W. Harrow, Efficient classical simulation of random shallow 2D quantum circuits, *Phys. Rev. X* **12**, 021021 (2022).