Universal hardware-efficient topological measurement-based quantum computation via color-code-based cluster states

Seok-Hyung Lee and Hyunseok Jeong[*]

Department of Physics and Astronomy, Seoul National University, Seoul 08826, Republic of Korea

(Received 28 June 2021; accepted 23 November 2021; published 4 January 2022)

Topological measurement-based quantum computation (MBQC) enables one to carry out universal fault-tolerant quantum computation via single-qubit measurements with a family of large entangled states called cluster states as resources. Raussendorf's three-dimensional cluster states (RTCSs) based on the surface codes are mainly considered for topological MBQC. In such schemes, however, the logical Hadamard, phase ($Z^{1/2}$), and T ($Z^{1/4}$) gates which are essential for building up arbitrary logical gates are not implemented natively without using state distillation or lattice dislocations, to the best of our knowledge. In particular, state distillation generally consumes many ancillary logical qubits; thus it is a severe obstacle against practical quantum computing. To solve this problem, we suggest an MBQC scheme via a family of cluster states called *color-code-based cluster states* (CCCSs) based on the two-dimensional color codes instead of the surface codes. We define logical qubits, construct elementary logical gates, and describe error correction schemes. We show that all logical Clifford gates generated by the CNOT, Hadamard, and phase gates can be implemented natively in a fault-tolerant manner, although the T gate still requires state distillation to be fault-tolerant. The error thresholds of MBQC via CCCSs for logical-Z errors are calculated to be 2.7%–2.8%, which are comparable to the values for RTCSs, assuming a simple error model where physical qubits have nontrivial errors independently with the same probability. We analyze and compare the resource overheads of both the schemes. In particular, we show that the number of physical qubits required for implementing a phase gate with CCCSs is at least about 26 times smaller than with RTCSs using state distillation, for the same code distance.

DOI: 10.1103/PhysRevResearch.4.013010

I. INTRODUCTION

Three major theoretical challenges for quantum computation (QC) are *universality*, *fault tolerance*, and *resource efficiency*. Universality indicates the ability of a quantum computer to initialize logical qubits to the computational basis state, apply any unitary operations, and measure them in the computational basis. It is known that the controlled-NOT (CNOT), Hadamard, and phase ($Z^{1/2}$) gates completely generate the Clifford group, and together with the T ($Z^{1/4}$) gate, any unitary operation may be approximated to an arbitrary accuracy [1,2].

To achieve fault tolerance, various quantum error-correcting (QEC) codes have been proposed from simple codes with few physical qubits [3–7] to topological stabilizer codes defined on lattice structures of qubits allowing only local interactions [8]. Several simple codes also have been demonstrated experimentally in assorted systems for small code distances [9–17]. Particularly, the *surface codes* [18–25], a family of topological codes defined on two-dimensional (2D) lattices, have high *error thresholds* up to about 12% [24]; thus they are one of the most promising candidates for fault-tolerant QC. The *2D color codes* are another family of topological codes [8,26–28] which enable transversal [29] implementation of the logical CNOT, Hadamard, and phase gates due to their self-duality. Moreover, three-dimensional (3D) gauge color codes even allow transversal implementation of the logical T gate as well as the Clifford gates, and thus have been getting much attention recently [30–35].

Last, fault-tolerant universal QC typically requires enormous resource overheads, which makes it tough to realize it. It is not only because a single logical qubit is composed of multiple physical qubits, but also because state distillation, which generally demands many ancillary logical qubits, is required for non-Clifford gates and sometimes for several Clifford gates to be fault-tolerant [22,24,36,37]. It is thus desirable to find QC schemes minimizing the need for state distillation.

Measurement-based QC (MBQC) is an alternative of conventional circuit-based QC (CBQC), processed only by single-qubit measurements on a large entangled state called a *cluster state* [38–43]. The ingredients for generating a cluster state are physical qubits initialized to the X-basis states and appropriate controlled-Z (CZ) gates on them; thus MBQC requires fewer types of physical-level operations than typical CBQC. The initial MBQC schemes via cluster states on 2D planes [39,40] were universal but not fault-tolerant. To achieve fault tolerance, the space should be 3D; *Raussendorf's 3D cluster states* (RTCSs) based on the surface codes allow universal and fault-tolerant MBQC with

[*]jeongh@snu.ac.kr

topologically encoded logical qubits [38,41–44]. Additionally, it was shown that it can tolerate imperfect preparation of cluster states such as qubit losses or failed CZ gates [45–47]. Recently, it has been shown that it is possible to generalize the relation between RTCSs and the surfaces codes to any Calderbank-Steane-Shor (CSS) codes [48,49] and later to any stabilizer codes [50]. MBQC is now regarded as one of the promising candidates for practical fault-tolerant QC, especially in optical systems [47,51–62].

MBQC via RTCSs is powerful from the point of view of fault tolerance but has a significant drawback: There are no ways to natively implement the topologically protected logical Hadamard, phase, and T gates unlike the CNOT gate, to the best of our knowledge. Several ways to circumvent this problem have been suggested. The conventional one is to use costly state distillation; these gates can be realized with error-free ancilla logical states $|Y_L\rangle := (|0_L\rangle + i|1_L\rangle)/\sqrt{2}$ and $|A_L\rangle := (|0_L\rangle + e^{i\pi/4}|1_L\rangle)/\sqrt{2}$ which are distilled from noisy ones [38]. Alternatively, there have been proposals to map lattice surgery [63] onto MBQC models [44,50]. With their methods, the Hadamard and phase gates can be fault tolerantly implemented without distillation by “dislocating” the RTCS lattice structure (i.e., transforming the lattice locally) when the gates are applied. In other words, the lattice loses its translational symmetry when the gates are applied. However, such dislocations may be undesirable from a practical point of view since the hardware should be capable of applying extra CZ gates which are not in the original lattice. Namely, the hardware should be designed in a way that can create multiple types of lattice structures.

To solve the above problem, we propose a MBQC scheme via a family of cluster states based on the 2D color codes instead of the surface codes, called *color-code-based cluster states* (CCCSs). We show that MBQC via CCCSs natively implements the logical Hadamard and phase gates fault tolerantly without the need of state distillation and lattice dislocations, while keeping most of the advantages of MBQC via RTCSs. In this sense, our scheme is hardware efficient.

The paper is structured as follows: In Sec. II we review the concept of cluster states and the general process of MBQC. In Sec. III we construct CCCSs and describe their properties, especially their stabilizers called *correlation surfaces*. In Sec. IV we show that universal MBQC is possible via CCCSs by defining logical qubits and suggesting the schemes for their initializations and measurements, elementary logical gates, and state injection. In Sec. V we present the methods to correct physical-level errors. In Sec. VI we calculate the error thresholds of MBQC via CCCSs and compare them with the results for RTCSs. In Sec. VII we analyze and compare the resource overheads of placing logical qubits and implementing each logical gate in the two schemes. We conclude with final remarks in Sec. VIII.

II. CLUSTER STATES AND MEASUREMENT-BASED QUANTUM COMPUTATION

To define a *cluster state*, we consider a graph $G = (V, E)$, where V and E are the sets of vertices and edges, respectively. The cluster state $|G\rangle$ is constructed by attaching a qubit to every vertex in V, initializing them to the $|+\rangle$ states where

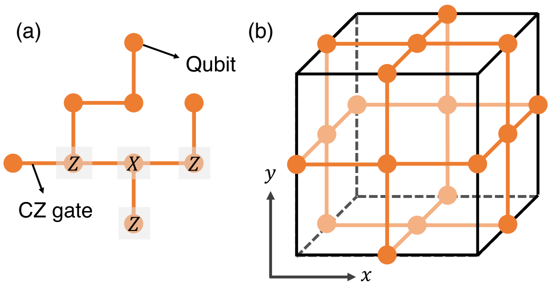


FIG. 1. Examples of cluster states. Orange dots and lines indicate vertices and edges of the graphs, respectively. To construct a cluster state, qubits initialized to the $|+\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$ states are placed on the vertices, then a CZ gate is applied on the qubits connected by each edge. (a) A cluster state on a simple graph. The presented “$XZZZ$” operator indicates an example of a stabilizer generator given in Eq. (1). (b) A unit cell of a Raussendorf’s 3D cluster state (RTCS). A vertex is located on each edge and face of the cell.

$|\pm\rangle := (|0\rangle \pm |1\rangle)/\sqrt{2}$, then applying a controlled-Z (CZ) gate on every pair of qubits connected by an edge.

The constructed cluster state has a stabilizer generator (SG) $S(v)$ for each vertex $v \in V$ defined as

$$S(v) := X(v) \prod_{v' \in \mathrm{adj}(v)} Z(v'), \tag{1}$$

where $\mathrm{adj}(v) := \{v' \in V \mid (v, v') \in E\}$ is the set of adjacent vertices of v and $X(v)$ and $Z(v)$ are the X and Z operators, respectively, on the qubit at the vertex v denoted by $Q(v)$. In other words, $g|G\rangle = |G\rangle$ for all $g \in \mathcal{S}$, where $\mathcal{S}$ is the stabilizer group generated by $\{S(v)|v \in V\}$. (See Chap. 10.5 of Ref. [2] for basic descriptions on the stabilizer formalism.) We say that $S(v)$ is *around* v or $Q(v)$, called its *center vertex* or *qubit*, respectively. Examples of cluster states are presented in Fig. 1.

For MBQC, the above definition of a cluster state is slightly modified: Some qubits (called *input qubits*) do not need to be initialized to the $|+\rangle$ states. $S(v)$ for each vertex v to which an input qubit is attached is then no longer a stabilizer, but others remain as SGs.

General MBQC via a cluster state to implement a quantum circuit is processed through the following three steps [38–40,42,43]:

(1) *Preparation.* For a given graph $G(V, E)$, a qubit is attached to each vertex. The set of all qubits $Q(V)$ is divided into three subsets: the input qubits Q_{IN}, the output qubits Q_{OUT}, and the others. $Q_{\mathrm{OUT}} = \emptyset$ if the desired circuit does not produce any output state or ends with measurements. The input logical states are prepared in Q_{IN}. All qubits except those in Q_{IN} are initialized to the $|+\rangle$ states. A CZ gate is then applied on every pair of qubits connected by an edge.

(2) *Measurement.* For each physical qubit except those in Q_{OUT}, a single-qubit measurement, selected by a *measurement pattern* with a classical computer, is performed. The measurement pattern is determined by the desired circuit. Let the measurement results be M. If possible, errors in M are corrected by decoding the *parity-check* outcomes.

(3) *Obtaining the results.* The output logical state is obtained from Q_{OUT} up to logical Pauli operators called *by-*

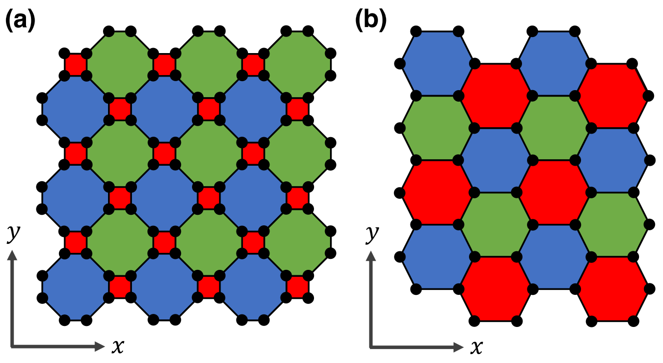


FIG. 2. Two typical examples of color-code lattices: the (a) 4-8-8 and (b) 6-6-6 lattices. The lattices are 3-valent and have 3-colorable faces.

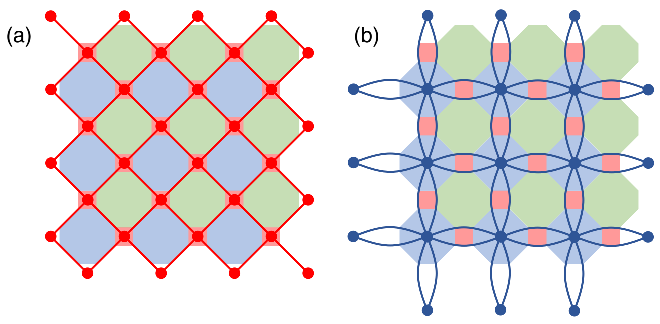


FIG. 3. (a) Red and (b) blue shrunk lattices of the 4-8-8 color-code lattice. Red or blue dots (lines) indicate their vertices (edges), which corresponds to red or blue faces (links) of the original lattices.

product operators determined by M. If $Q_{\mathrm{OUT}} = \emptyset$, the results of the final logical measurements are determined by M.

Note that the preparation and measurement steps may be performed simultaneously; after a qubit q and its neighbors are prepared and CZ gates are applied on them, it is allowed that q is measured before the other qubits are prepared. One of the spatial axes may be regarded as the *simulating time axis*, or simply the *time axis*, along which qubits are prepared and then measured in order. It is thus possible to minimize the number of unmeasured physical qubits in a moment by measuring each qubit as soon as possible after preparing it.

The cores of MBQC are the structure of the cluster state and the measurement pattern. We illustrate them one by one over the next two sections.

III. COLOR-CODE-BASED CLUSTER STATES

In this section, we define color-code-based cluster states and describe their properties. Based on the work on the foliation of CSS codes [48], we consider a particular family of cluster states derived from a 2D color-code lattice, called *color-code-based cluster states* (CCCSs).

A. Two-dimensional color-code lattices

We first present 2D color-code lattices on which CCCSs are based. We consider a lattice $\mathcal{L}_{2D}$ on a 2D plane which is 3-valent and has 3-colorable faces; namely, three edges meet at each vertex and one of the three colors (red, green, or blue) is assigned to each face in such a way that neighboring faces have different colors. Note that each edge, called a *link*, is also colorable with the color of the faces it connects. Two typical examples (4-8-8 and 6-6-6) of such lattices are shown in Fig. 2. In the original 2D color codes, a qubit is attached to each vertex and two SGs (X- and Z-type) correspond to each face. Details on the codes are described in Refs. [8,26].

Regarding a color-code lattice $\mathcal{L}_{2D}$, three *shrunk lattices* are defined, one for each color by shrinking all the faces of that color, as shown in Fig. 3. For example, in the red shrunk lattice, each vertex corresponds to a red face in $\mathcal{L}_{2D}$ and each face corresponds to a blue or green face in $\mathcal{L}_{2D}$. Edges of the red shrunk lattice then correspond to red links in $\mathcal{L}_{2D}$. The blue and green shrunk lattices are also defined analogously.

B. Construction of color-code-based cluster states

The graph G for a CCCS based on a color-code lattice $\mathcal{L}_{2D}$ has a 3D structure composed of multiple identical 2D layers stacked along the simulating time (t) axis. The layer of $t = t_0$ is referred to as the *t_0-layer*.

The structure of each layer is originated from $\mathcal{L}_{2D}$, as illustrated in Fig. 4(a) for the case of the 4-8-8 lattice. Each vertex in the layer is located at either a vertex of $\mathcal{L}_{2D}$ or the center of a face of $\mathcal{L}_{2D}$; the corresponding qubit is called a *code qubit* (CQ) or an *ancilla qubit* (AQ), respectively. Each AQ is colorable with the color of the corresponding face in $\mathcal{L}_{2D}$. For each face in $\mathcal{L}_{2D}$, the layer has an edge connecting the corresponding AQ and each surrounding CQ, on which a CZ gate is applied. Each pair of CQs connected by a link in $\mathcal{L}_{2D}$ is called *link* here as well. Note that links are not edges of G.

Next, we stack multiple identical layers along the time axis as shown in Fig. 4(b). Every pair of CQs adjacent along the time axis is connected by an edge in G. The vertices (CQs and AQs) and edges (between CQs and AQs in the same layer and between CQs in the adjacent layers) constructed above finally complete the graph G of the cluster state.

We assign each layer, qubit or link a "primality": either *primal* or *dual*. Each layer is primal (dual) if it has an even (odd) time. An AQ is primal (dual) if it is in a primal (dual) layer, while a CQ or link is primal (dual) if it is in a dual (primal) layer. We label each qubit or link in an abbreviated form with its primality ("p" for primal and "d" for dual), color ("r" for red, "g" for green, and "b" for blue; omitted for CQs), and type ("AQ," "CQ," and "L" for a link). For example, a pgAQ means a primal green ancilla qubit. We also frequently use "c" instead of a specific color (r, g, or b) for a variable on colors.

C. Stabilizer generators

We now present stabilizer generators (SGs) of a CCCS. Note that, for each vertex v in G, $S(v)$ given in Eq. (1) is a SG if $Q(v)$ is initialized to $|+\rangle$. We define *A*- and *C-type SGs* shown in Figs. 5(a) and 5(b) as follows.

Definition 1 (A- and C-type SGs). An A- or C-type SG is the SG given in Eq. (1) around an AQ or a CQ, respectively.

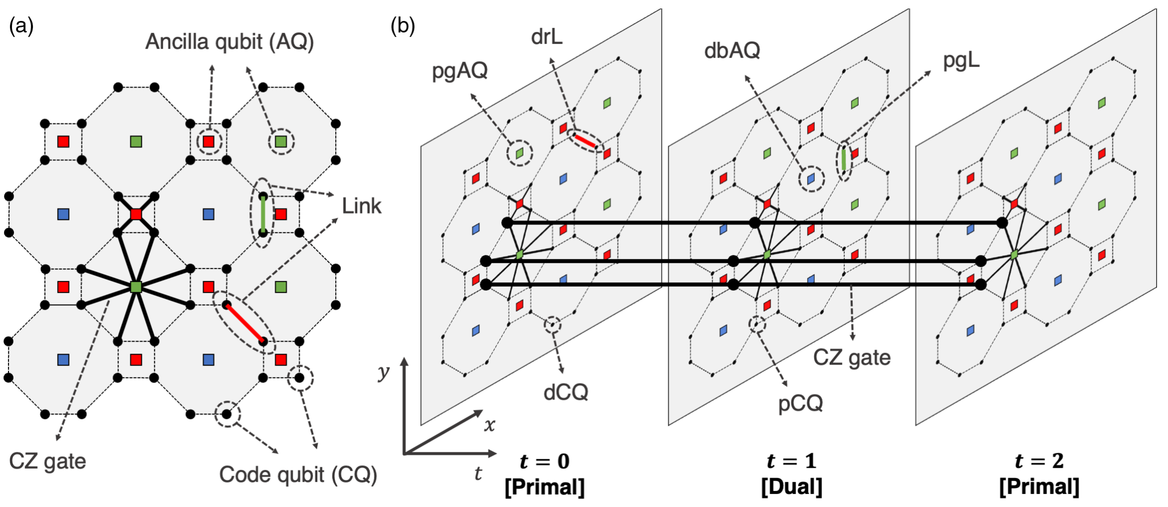


FIG. 4. Structure of a color-code-based cluster state (CCCS) based on the 4-8-8 color-code lattice $\mathcal{L}_{2D}$. (a) Structure of a single layer. Each black circle is a code qubit (CQ) located at a vertex of $\mathcal{L}_{2D}$. Each colored square is an ancilla qubit (AQ) with that color, located at the center of a face of $\mathcal{L}_{2D}$ with that color. Each AQ is connected with surrounding CQs by edges (CZ gates), some of which are drawn as black solid lines. Two adjacent CQs are connected by a link, some of which are drawn as colored lines. (b) Stack of multiple identical layers along the simulating time axis. Each pair of two CQs adjacent along the time axis is connected by an edge, some of which are presented as black solid lines. One of the primalities ("primal" and "dual") is alternatively assigned to each layer. An AQ (a CQ or link) is primal (dual) if it is in a primal layer, and vice versa for a dual layer. Labels of some elements defined in Sec. III B are shown.

The support of a C-type SG (namely, the set of qubits on which the SG acts nontrivially) is distributed in three adjacent layers, while that of an A-type SG is contained in a layer.

Although these two types of SGs completely generate the stabilizer group, we need another two types of SGs: *L-* and *J-type SGs* in Figs. 5(c) and 5(d).

Definition 2 (L-type SG). The L-type SG around a link l is the product of two C-type SGs whose center qubits constitute l.

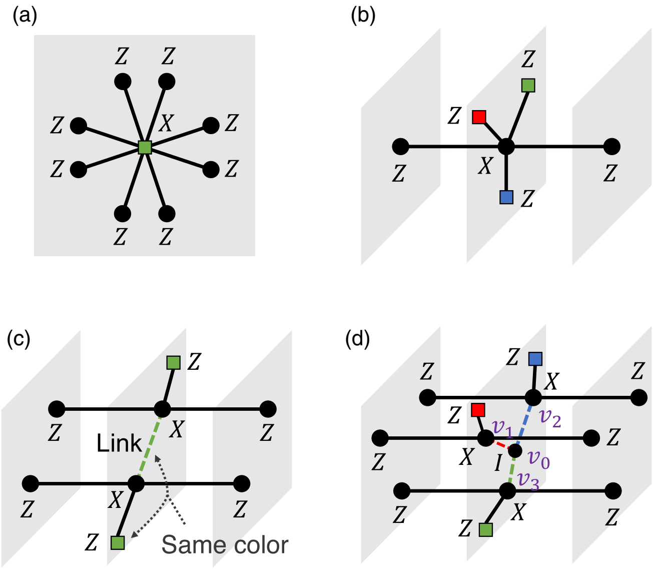


FIG. 5. Four types of stabilizer generators (SGs) in a CCCS defined in Definitions 1–3: (a) A-, (b) C-, (c) L-, and (d) J-type SGs. Each gray square indicates a layer. An SG of each type is the tensor product of the marked X or Z operators on the qubits.

Definition 3 (J-type SG). Let $S_i := S(v_i)$ for each $i \in \{0, 1, 2, 3\}$ be a C-type SG such that (v_0, v_1), (v_0, v_2), and (v_0, v_3) are links with different colors. $S_I := S_1S_2S_3$ is then the J-type SG around the CQ $Q(v_0)$.

A-, L-, and J-type SGs together generate the stabilizer group over-completely. To see this, regarding a J-type SG S_I, we consider an L-type SG $S_{Li} := S_0S_i$ for each $i \in \{0, 1, 2, 3\}$, where S_i's are defined in Definition 3. Then $S_0 = S_{L1}S_{L2}S_{L3}S_I$ holds; thus any C-type SG can be written as the product of L- and J-type SGs.

Let the *P-support* of a multiqubit Pauli operator O for $P \in \{X, Y, Z\}$ denoted by $\text{supp}_P(O)$ be the subset of $\text{supp}(O)$ (i.e., the support of O) corresponding to the P operators in O. Note that, for every SG regardless of its type, qubits in its X- and Z-support always have different primalities.

D. Shrunk lattices and correlation surfaces

Almost every discussion from now on is symmetric between the two primalities. Thus, throughout the rest of this paper, we frequently discuss only one of them, which implies that the other side can be treated similarly.

We now construct the shrunk lattices of a CCCS, which are analogous to those of the 2D color codes in Fig. 3. We then define correlation surfaces [38,41,42] within each shrunk lattice, through which logical gates are built for MBQC.

The primal c-colored shrunk lattice $\mathcal{L}^{\text{pc}}$ is a 3D lattice containing every pcAQ as a vertex. Note that the vertices are only in primal layers. There are two types of edges connecting them: "spacelike" and "timelike" edges. Each spacelike edge corresponds to a pcL and connects two vertices in a layer. Each timelike edge connects two vertices adjacent along the time axis and contains a dcAQ between them. Faces and cells are then naturally defined by the vertices and edges. Cells in each primal shrunk lattice are visualized in Fig. 6 for 4-8-8

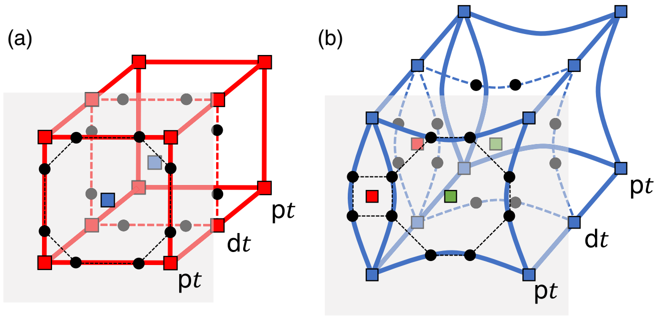


FIG. 6. Unit cells of the primal shrunk lattices of a 4-8-8 CCCS: (a) a blue cell in the primal red shrunk lattice $\mathcal{L}^{\mathrm{pr}}$ (a green cell is identical except the colors of AQs) and (b) red and green cells in the primal blue shrunk lattice $\mathcal{L}^{\mathrm{pb}}$. p$t$s and d$t$s indicate primal and dual layers, respectively. Some qubits on the last layer are not displayed. All the pcAQs are vertices of $\mathcal{L}^{\mathrm{pc}}$. Each spacelike (or timelike) edge, visualized as red or blue solid lines, connects two adjacent vertices in a layer (or different layers) and corresponds to a pcL (or dcAQ). Faces and cells are defined naturally with the edges.

CCCSs. Note that each primal layer in $\mathcal{L}^{\mathrm{pc}}$ is identical with the c-colored shrunk lattice of the 2D color code on which the CCCS is based.

Each element (vertex, edge, face, or cell) in a shrunk lattice corresponds to an AQ or a link, as presented in Table I. Here $Q(b)$ for an element b denotes the set of qubits corresponding to b. Note that an element is colorable with the color of AQ or link corresponding to it. In particular, cells and spacelike faces have colors different from the color of the shrunk lattice, e.g., $\mathcal{L}^{\mathrm{pr}}$ is composed of green and blue cells.

We now regard the shrunk lattices as chain complexes [38,41,42]. Let $\mathcal{B}_i^{\mathrm{pc}}$ for $i = 0$, 1, 2, or 3 be the set of vertices, edges, faces, or cells in $\mathcal{L}^{\mathrm{pc}}$, respectively. We then consider a vector space H_i^{pc} generated by $\mathcal{B}_i^{\mathrm{pc}}$ over $\mathbb{Z}_2$. Each primal shrunk lattice may be regarded as a chain complex: $\mathcal{L}^{\mathrm{pc}} = \{H_3^{\mathrm{pc}}, H_2^{\mathrm{pc}}, H_1^{\mathrm{pc}}, H_0^{\mathrm{pc}}\}$. Each element $h_i \in H_i^{\mathrm{pc}}$ is called an *i-chain* and corresponds to a set $B(h_i) \subseteq \mathcal{B}_i^{\mathrm{c}}$ where each $b \in B(h_i)$ has nonzero contribution in h_i. For example, if f_1, f_2, and f_3 are faces in $\mathcal{L}_2^{\mathrm{pc}}$, $h_2 := f_1 + f_2 + f_3$ is a 2-chain in H_2^{pc} and $B(h_2) = \{f_1, f_2, f_3\}$ holds. The correspondence is one-to-one, and thus we use h_i and $B(h_i)$ without distinction throughout the paper for convenience. The chain complex $\mathcal{L}^{\mathrm{pc}}$ has a boundary map ∂ which maps $h_i \in H_i^{\mathrm{pc}}$ to $\partial h_i \in H_{i-1}^{\mathrm{pc}}$ corresponding to the geometrical boundary of h_i. Note that ∂ is a linear map and satisfies $\partial \circ \partial = 0$.

TABLE I. Qubits $Q(b)$ corresponding to each element (vertex, edge, face, or cell) b in $\mathcal{L}^{\mathrm{pc}}$. The results for $\mathcal{L}^{\mathrm{dc}}$ can be obtained by changing each p or d.

Element b in $\mathcal{L}^{\mathrm{pc}}$		Qubits $Q(b)$
Vertex ($\in \mathcal{B}_0^{\mathrm{c}}$)		pcAQ
Edge ($\in \mathcal{B}_1^{\mathrm{c}}$)	Timelike	dcAQ
	Spacelike	dcL (two dCQs)
Face ($\in \mathcal{B}_2^{\mathrm{c}}$)	Timelike	pcL (two pCQs)
	Spacelike	pc′AQ (c′ $\neq$ c)
Cell ($\in \mathcal{B}_3^{\mathrm{c}}$)		dc′AQ (c′ $\neq$ c)

For an i-chain h_i and $P \in \{X, Y, Z\}$, we define a multiqubit Pauli operator $P(h_i)$ by

$$P(h_i) := \prod_{q \in Q(h_i)} P(q),$$

where $Q(h_i) := \bigcup_{b_i \in h_i} Q(b_i)$ and $P(q)$ is the P operator on the qubit q tensored with identity on all other qubits. We now define correlation surfaces (CSs), essential elements for constructing logical operations through MBQC.

Definition 4 (Correlation surface). For each 2-chain $h_2 \in H_2^{\mathrm{p(d)c}}$, the operator

$$S_{\mathrm{CS}}(h_2) := X(h_2)Z(\partial h_2) \tag{2}$$

is a primal (dual) c-colored correlation surface, referred to as a "p(d)c-CS."

It is straightforward to see that, for a spacelike or timelike face f, $S_{\mathrm{CS}}(f)$ is an A- or L-type SG around the AQ or link corresponding to f, respectively. The following theorem relates general 2-chains to stabilizers of the CCCS.

Theorem 1 (Correlation surfaces as stabilizers). For a 2-chain h_2, $S_{\mathrm{CS}}(h_2)$ is a stabilizer before measuring any qubit in its support if and only if $Q(h_2) \cap Q_{\mathrm{IN}} = \emptyset$, where Q_{IN} is the set of input qubits defined in Sec. II which are not initialized to the $|+\rangle$ states.

Proof. (*If part*) Since qubits outside Q_{IN} is initialized to the $|+\rangle$ states, there exist the A- or C-type SG around each of them, as discussed in Sec. II. Let $F := \{f \in \mathcal{B}_2^{\mathrm{pc}} \mid Q(f) \cap Q_{\mathrm{IN}} = \emptyset\}$ be a set of faces. For a face $f \in F$, $S_{\mathrm{CS}}(f)$ is a stabilizer before measuring any qubit in its support; it is an A- or L-type SG. For a 2-chain $h_2 \in H_2^{\mathrm{pc}}$ where $Q(h_2) \cap Q_{\mathrm{IN}} = \emptyset$, h_2 can be written as a linear summation of elements in F: $\exists \{f_i\} \subseteq F$, $h_2 = \sum_i f_i$. Since the map ∂ is linear and $P(h)P(h') = P(h + h')$ holds for any Pauli operator P, $S_{\mathrm{CS}}(h_2) = X(h_2)Z(\partial h_2) = \prod_i X(f_i)Z(\partial f_i) = \prod_i S_{\mathrm{CS}}(f_i)$ is a stabilizer before measuring any qubit in its support. The proof is analogous for dual 2-chains. (*Only if part*) Since qubits in Q_{IN} are not initialized to the $|+\rangle$ states, the A- and C-type SGs around each of them do not exist. Therefore, the X-support of any stabilizer cannot contain qubits in Q_{IN}. ■

Regarding a primal CS $S := S_{\mathrm{CS}}(h_2)$, $Q(h_2)$ is called the *interior* of S, in which every qubit is primal and in $\mathrm{supp}_X(S)$. Similarly, $Q(\partial h_2)$ is called the *boundary* of S, in which every qubit is dual and in $\mathrm{supp}_Z(S)$. We say that S is *timelike* (*spacelike*) if h_2 is composed of timelike (spacelike) faces only.

CSs discussed above include all A- and L-type SGs, but not J-type SGs in Fig. 5(d). Each J-type SG can be regarded as three primal timelike CSs with different colors "joined" along a timelike series of CQs as Fig. 7(a), in the sense that each "wing" of a color c may be extended by multiplying ordinary pc-CSs. Note that the CQs along the joint are not included in the support.

A question arising naturally may be about "spacelike" joints, and those are also possible as presented in Fig. 7(b). A timelike pc-CS and two spacelike primal CSs with the other two colors may be joined along a spacelike series of pcLs. Such a joint can be obtained by multiplying several A-type SGs along a spacelike boundary of the timelike CS. Note that the ends of spacelike and timelike joints may fit perfectly with

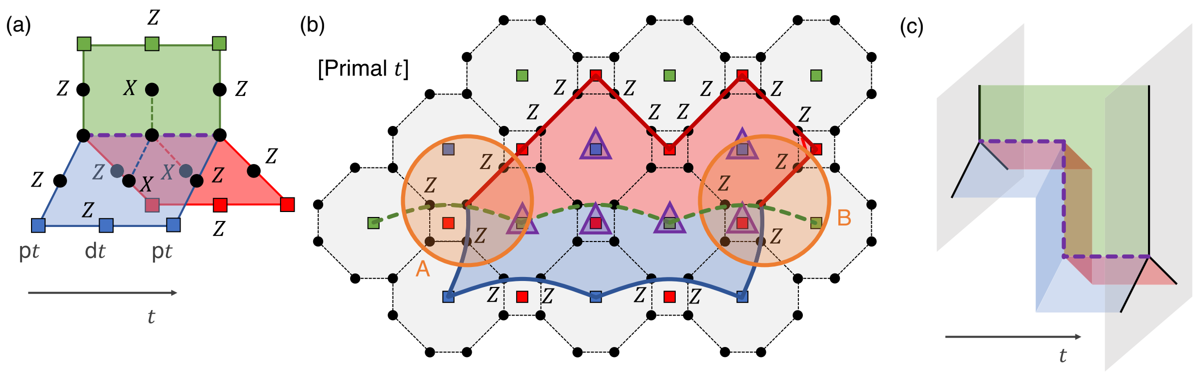


FIG. 7. (a) Timelike joint of primal correlation surfaces (CSs) originated from a J-type SG. The X or Z operators on the qubits indicate the support of the resulting CS. A series of CQs along which the three faces meet is marked as a purple dashed line. (b) An example of the construction of a spacelike joint of three primal CSs. A primal layer of a 4-8-8 CCCS is presented. We first assume a timelike pg-CS S ending at the green dashed line. We then expand S by multiplying the A-type SGs around the pAQs marked with purple triangles. After the expansion, $\mathrm{supp}_X(S)$ contains the marked pAQs, and $\mathrm{supp}_Z(S)$ contains the CQs along the red and blue solid lines. The red (blue) area above (below) the green line can be regarded as a pr(b)-CS, in the sense that it may be expanded by multiplying ordinary pr(b)-CSs. A joint of the three CSs is thus constructed, and S is the corresponding joined CS. The qubits in $\mathrm{supp}_Z(S)$ inside the area A or B exactly match with the final layer of a timelike joint; thus spacelike and timelike joints may be connected. (c) An example of a general joint, obtained by multiplying a series of timelike and spacelike joints together with ordinary CSs.

each other, in the sense that all the Z operators on the joint cancel out when multiplying them.

A general joint of CSs with different colors can be obtained as Fig. 7(c) by multiplying several spacelike and timelike joints together with ordinary CSs. We refer to such a primal CS with a joint as a "pj-CS." For consistency with ordinary CSs, we define the *interior* (*boundary*) of a pj-CS by its $X(Z)$-support, which is intuitive considering its visualization in Fig. 7.

IV. MEASUREMENT-BASED QUANTUM COMPUTATION VIA COLOR-CODE-BASED CLUSTER STATE

In this section, we describe the scheme for MBQC via CCCSs. We first introduce defects and define logical qubits using them. We then describe initializations and measurements of logical qubits and construct elementary logical gates including the identity, CNOT, Hadamard, and phase gates, which together generate the Clifford group. We last present the state injection scheme to prepare arbitrary logical states and implement the logical T gate.

Each logical initialization, measurement, gate, or state injection process can be regarded as an independent circuit "block" implemented by the process presented in Sec. II: In each block, the input logical state is first prepared in the input qubits Q_{IN}, then the output logical state is produced in the output qubits Q_{OUT} after the single-qubit measurements of all qubits except Q_{OUT}. Note that Q_{IN} (Q_{OUT}) is empty for the logical initializations (measurements). An arbitrary quantum circuit can be constructed by connecting multiple blocks in a way that the output qubits of each block are used as the input qubits of the next block.

We assume that the single-qubit measurements are performed layer by layer along the simulating time (t) axis. In that case, the output qubits of a (gate, initialization, or state injection) block are the last several layers in it, called the *output layers*. On the other hand, it is sufficient that the input qubits of a (gate or measurement) block contain only the first layer in it, called the *input layer*. Exceptionally, the input qubits of a state injection block contain only one qubit into which an unencoded state is injected.

Two subsequent blocks can be connected in a way that the input layer of the second block overlaps with the first output layer of the first block. To see this, let us assume that the output layers of the first block are the layers of $t_0 \leqslant t \leqslant t_1$. We first consider applying all the CZ gates between qubits of $t_0 \leqslant t \leqslant t_1$ again after measuring the qubits of $t > t_0$. Since the measurements commute with these CZ gates, the qubits of $t_0 < t \leqslant t_1$ simply return to the initial $|+\rangle$ states. The t_0-layer is then the only layer containing nontrivial information and used as the input layer of the second block. The CZ gates in the first $t_1 - t_0 + 1$ layers of the second block restore the output state of the first block to be used as the input state of the second block. Of course, the above argument is just a trick to connect two blocks conceptually; it is unnecessary to apply CZ gates multiple times in a real implementation.

A. Measurement pattern

Note that each qubit except the output qubits is measured in the basis determined by a predefined measurement pattern. In our scheme, a qubit is included in an area with one of the four types: *vacuum*, *defect*, *Y-plane*, and *injection qubit*. There may be multiple defects, Y-planes, and injection qubits, and the entire remaining area is the vacuum. We denote the set of all vacuum (defect) qubits as V (D).

Defects are key ingredients for the protocol; all the logical operations completely depend on how to place them. Y-planes are used in fault-tolerant Y-measurements on physical qubits for the logical Hadamard and phase gates. Last, each injection qubit is a special area for state injection and consists of a single qubit. Note that injection qubits are always input qubits.

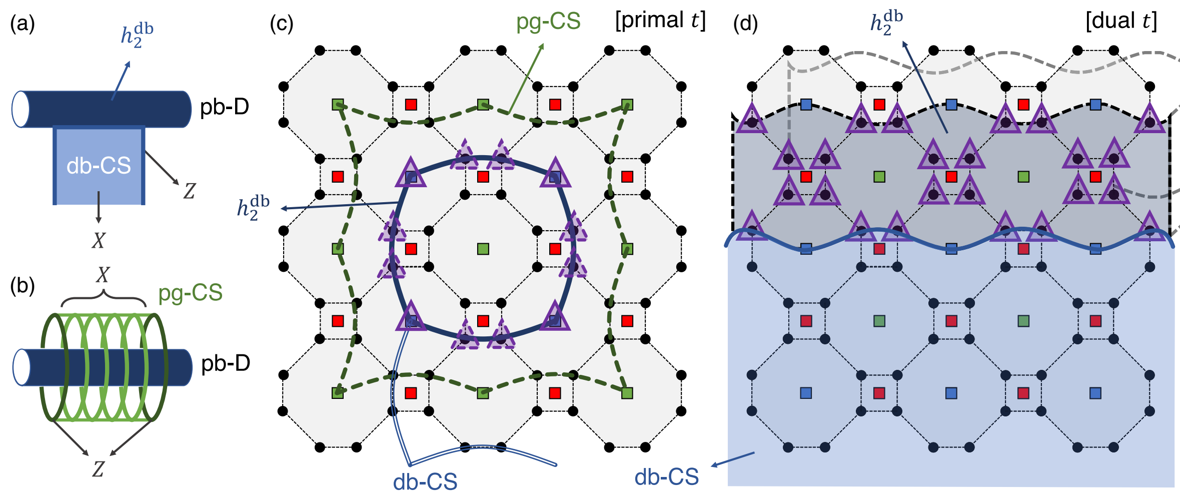


FIG. 8. (a) Schematic diagram of a defect (pb-D) and a db-CS S ending at the defect. The defect is defined by Eq. (4) with a 2-chain $h_2^{\rm db} \in H_2^{\rm db}$ in the shape of a pipe. (b) Schematic diagram of a pg-CS surrounding a pb-D. (c) A primal layer in a 4-8-8 CCCS penetrated by a timelike pb-D $D(h_2^{\rm db})$ for a 2-chain $h_2^{\rm db}$. The cross section of $h_2^{\rm db}$ is presented as a blue solid line. Each purple triangle with a solid (dashed) border indicates a defect pbAQ (pCQ) in the layer (adjacent layer) measured in the Z-basis. The cross sections of a timelike db-CS ending at the defect and a timelike pg-CS surrounding it are presented as a blue double line and a green dashed line, respectively. The double (or dashed) lines indicate faces bisected by the layer (or ending at the layer); that is, the corresponding qubits are on the layer (or an adjacent layer). (d) A dual layer in a 4-8-8 CCCS containing one side of a spacelike pb-D. Part of the 2-chain $h_2^{\rm db}$ corresponding to the defect is presented as a gray surface. A db-CS ending at the defect is visualized as a blue surface, where the blue line corresponds to its boundary.

Qubits in each area are measured as follows:

$$\begin{array}{c}\text{A qubit is measured}\\ \text{in the basis of}\end{array} \begin{cases} X & \text{if in the vacuum} \\ & \text{or an injection qubit,} \\ Z & \text{if in a defect,} \\ Y & \text{if in a Y-plane.}\end{cases} \tag{3}$$

Arranging these elements besides the vacuum properly is the key for implementing logical qubits and gates, which is what we cover in this section.

B. Defects and related correlation surfaces

We first define a defect as follows.

Definition 5 (Defect). Consider a 2-chain $h_2 \in H_2^{\rm d(p)c}$ in the shape of a "pipe," as shown in Fig. 8(a). A primal (dual) c-colored defect, referred to as a "p(d)c-D," corresponding to h_2 is defined by

$$D(h_2) := \bigcup_{f\in h_2} Q(\partial f), \tag{4}$$

which consists of p(d)cAQs and p(d)CQs.

We say that a defect is *timelike* or *spacelike* if the "pipe" is extended along the time axis or an spatial axis, respectively. It is also possible that the direction of a defect is changed in the middle. Figures 8(c) and 8(d) illustrate the explicit structures of timelike and spacelike defects, respectively, in a 4-8-8 CCCS. Here each purple triangle with a solid (dashed) border indicates a defect qubit located at the layer (adjacent layer).

We now get the following theorem regarding *compatible* CSs surviving after the measurement step.

Theorem 2 (Compatible correlation surfaces). For a set of qubits $\tilde{Q}$, a CS S is compatible with $\tilde{Q}$ (namely, S is a stabilizer both before measuring any qubit and after measuring all the qubits in $\tilde{Q} \setminus Q_{\rm OUT}$) if and only if the followings hold:

$$Q_{\rm int}(S) \cap \tilde{Q} \setminus Q_{\rm OUT} \subseteq V \setminus Q_{\rm IN}, \tag{5a}$$

$$Q_{\rm bnd}(S) \cap \tilde{Q} \setminus Q_{\rm OUT} \subseteq D, \tag{5b}$$

where $Q_{\rm int(bnd)}(S)$ is the interior (boundary) of S and $Q_{\rm IN(OUT)}$ is the set of input (output) qubits.

Proof. Note first that $S = X(Q_{\rm int}(S))Z(Q_{\rm bnd}(S))$. It is known [2] that a stabilizer S remains as a stabilizer after measuring a Pauli operator P if and only if S and P commute. (If part) Suppose that Eq. (5) holds. Let q be an arbitrary qubit in $\tilde{Q} \setminus Q_{\rm OUT}$. If $q \in Q_{\rm int}(S)$, q is in the vacuum, and thus $[M(q), S] = [X(q), S] = 0$, where $M(q)$ is the single-qubit Pauli operator on q corresponding to the measurement pattern. If $q \in Q_{\rm bnd}(S)$, q is in a defect, thus $[M(q), S] = [Z(q), S] = 0$. If otherwise, $q \notin \mathrm{supp}(S)$, thus $M(q)$ and S commute. Therefore, S is compatible with $\tilde{Q}$. (Only if part) Suppose that a CS S is compatible with $\tilde{Q}$. Then S should commute with $M(q)$ for each qubit $q \in \tilde{Q} \setminus Q_{\rm OUT}$. Let q be an arbitrary qubit in $Q_{\rm int}(S) \cap \tilde{Q} \setminus Q_{\rm OUT} \subseteq \tilde{Q} \setminus Q_{\rm OUT}$. q cannot be an injection qubit, since $Q_{\rm int}(S) \cap Q_{\rm IN} = \emptyset$ according to Theorem 1 and injection qubits are always input qubits. Thus, if $q \notin V$, $M(q)$ is either $Y(q)$ or $Z(q)$, and thus $M(q)$ and S anticommute, which contradicts the assumption. Therefore, q is in V. Since $Q_{\rm int}(S) \cap Q_{\rm IN} = \emptyset$, $Q_{\rm int}(S) \cap \tilde{Q} \setminus Q_{\rm OUT} \subseteq V \setminus Q_{\rm IN}$ holds. $Q_{\rm bnd}(S) \cap \tilde{Q} \setminus Q_{\rm OUT} \subseteq D$ can be shown analogously. ■

If a CS is compatible with all the qubits except the output qubits, we say that it is a compatible CS. We particularly want to emphasize that a compatible CS cannot end in the vacuum qubits. Note that $Q_{\rm IN}$ is excluded in the right-hand side (RHS) of Eq. (5a) due to Theorem 1.

TABLE II. Allowed positional relations between a primal defect d and a compatible CS. The relations for dual defects are analogous.

x \ y (With a pc-D d, a **xy**-CS...)	c	$c'(\neq c)$
p	Can be penetrated by d only if d is timelike.	Cannot be penetrated by d.
	Cannot end at d.	
d	Can be penetrated by d.	
	Can end at d.	Cannot end at d in general.

Table II shows allowed positional relations between a pc-D d and a compatible CS with each primality and color, derived from Theorem 2 and Table I. Note that d is composed of pcAQs and pCQs corresponding to edges in $\mathcal{L}^{\mathrm{dc}}$. Let us first check whether a compatible pc′-CS S can be penetrated by d. The interior of S corresponds to faces in $\mathcal{L}^{\mathrm{pc}'}$; thus it consists of pCQs if S is timelike and pc″AQs (c″ $\neq$ c) if S is spacelike, as shown in Table I. According to Eq. (5a), the interior should not contain defect qubits for S to be compatible. Therefore, S can be penetrated by d only if c $=$ c′ and S is spacelike (i.e., d is timelike). Additionally, S cannot end at d, since its boundary qubits are dual. Next, let us check whether a compatible dc′-CS S can end at d. The boundary of S corresponds to edges in $\mathcal{L}^{\mathrm{dc}'}$. According to Eq. (5b), the boundary should contain defect qubits for S to be compatible. Since the defect qubits correspond to edges in $\mathcal{L}^{\mathrm{dc}}$, S can end at d if c $=$ c′; otherwise, it is impossible in general. (There may be specific cases that it is possible even if c $\neq$ c′, but they are not utilized in our schemes.) Additionally, S can be penetrated by d since its interior qubits are dual.

We mainly concern two types of CSs with respect to a pc-D: pc-CSs surrounding the defect and dc-CSs ending at it, as shown schematically in Figs. 8(a) and 8(b) and explicitly in Figs. 8(c) and 8(d). Each of such CSs is compatible with all the qubits except the boundary qubits in the two ends about the direction of the defect.

C. Defining a logical qubit

We first define *connected 1-chains* as follows.

Definition 6 (Connected 1-chain). A 1-chain h_1 is connected if and only if it satisfies $|\partial h_1| \leqslant 2$. It is closed if $|\partial h_1| = 0$ and open if otherwise.

To define a logical qubit, we consider three parallel timelike defects with different colors passing through the t_0- and (t_0+1)-layer for a given integer t_0, as visualized schematically in Fig. 9(a). The constructed logical qubit is primal (dual) if the defects are primal (dual) and t_0 is odd (even).

We define a logical qubit by specifying the logical-X (X_L) and logical-Z (Z_L) operators. To define X_L, for a given pair of different colors (c, c′), we consider two closed connected spacelike 1-chains $h_1^{X\mathrm{dcc}'} \in H_1^{\mathrm{dc}}$ and $h_1^{X\mathrm{pcc}'} \in H_1^{\mathrm{pc}}$: $h_1^{X\mathrm{dcc}'}$ is located in the t_0-layer and surrounding the pc′-D. $h_1^{X\mathrm{pcc}'}$ is defined by parallelly moving $\mathrm{supp}_Z(X_L) = Q(h_1^{X\mathrm{pcc}'})$ one unit positively along the time axis. An example of X_L is shown in Fig. 9(a) for the case of (c, c′) $=$ (b, r). Note that the two 1-chains consist of pcLs and dcLs, respectively. We then define

$$X_L := F_X^{\mathrm{cc}'}(t_0) := X\big(h_1^{X\mathrm{dcc}'}\big)Z\big(h_1^{X\mathrm{pcc}'}\big). \tag{6}$$

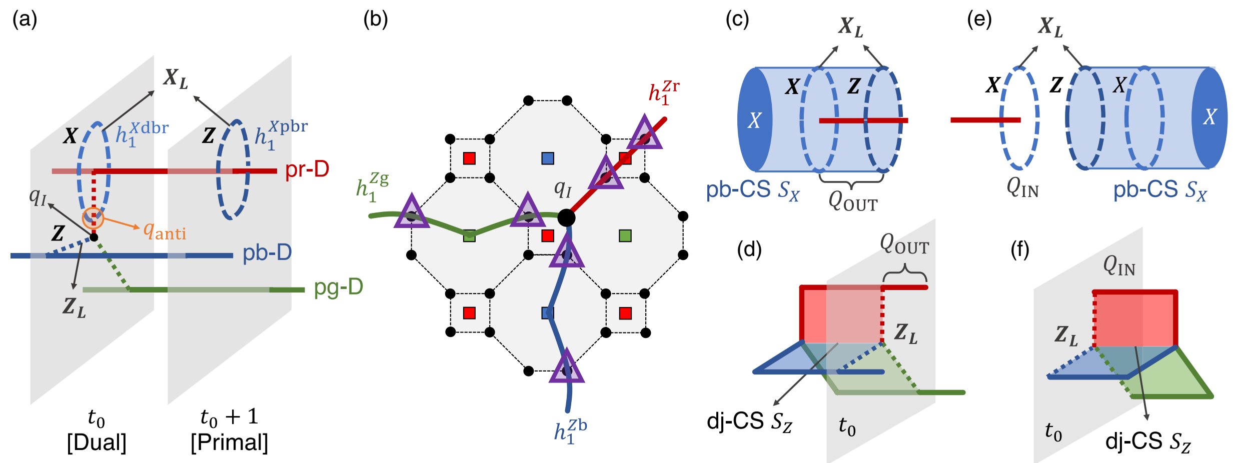


FIG. 9. Definition of a primal logical qubit and its initialization and measurement. (a) Schematic diagram of a primal logical qubit composed of three parallel primal timelike defects with different colors. Blue dashed lines indicate 1-chains $h_1^{X\mathrm{dbr}}$ and $h_1^{X\mathrm{pbr}}$, which constitute $\mathrm{supp}_X(X_L)$ and $\mathrm{supp}_Z(X_L)$, respectively. Red, green, and blue dotted lines indicate 1-chains h_1^{Zr}, h_1^{Zg}, and h_1^{Zb}, respectively, which constitute $\mathrm{supp}_Z(Z_L)$ except the pCQ q_I at which they end. $\mathrm{supp}_X(X_L)$ and $\mathrm{supp}_Z(Z_L)$ meet at a pCQ q_{anti}; thus they anticommute with each other. (b) Structure of Z_L near q_I in a 4-8-8 CCCS. Colored lines are h_1^{Zr}, h_1^{Zg}, and h_1^{Zb}, respectively. Purple triangles indicate $\mathrm{supp}(Z_L)$. (c) X_L- and (d) Z_L-initialization. A logical qubit prepared in the output layers Q_{OUT} (t_0- and (t_0+1)-layer). For the X_L-initialization, the defects are made to start from the t_0-layer. For the Z_L-initialization, they are extended to meet at a point before the layer-t_0. X_L (Z_L) is then a part of a pb-CS S_X (dj-CS S_Z), which is a stabilizer. After the measurement step, the logical qubit in Q_{OUT} is initialized to $|\pm_L\rangle$ ($|0_L\rangle$ or $|1_L\rangle$), depending on the measurement result of $X_L S_X$ ($Z_L S_Z$). (e) X_L- and (f) Z_L-measurement of a logical qubit inserted into the input layer (t_0-layer). Each of them is done by reversing the corresponding initialization process. There then exists a pb-CS S_X (dj-CS S_Z) which is a stabilizer, such that the measurement result of $S_X X_L$ ($S_Z Z_L$) determine the $X_L(Z_L)$-measurement result.

Note that $\mathrm{supp}_Z(X_L) = Q(h_1^{X\mathsf{pcc}'})$ may be in the boundary of a pc-CS since the boundary is a 1-chain in H_1^{pc} as well. The colors c and c′ can be any pair of different colors, and they are proven to be equivalent in Sec. IV E 2.

For the Z_L operator, we consider an open connected spacelike 1-chain $h_1^{Z\mathsf{c}} \in H_1^{\mathsf{dc}}$ for each color c, which is located in the t_0-layer and connects the pc-D and a common pCQ q_I, as shown in Fig. 9(a) schematically. Note that $h_1^{Z\mathsf{c}}$ is composed of pcLs. We define

$$\begin{aligned} Z_L &:= F_Z(t_0) \\ &:= Z(h_1^{Z\mathsf{r}})Z(h_1^{Z\mathsf{g}})Z(h_1^{Z\mathsf{b}})Z(q_I). \end{aligned} \tag{7}$$

Note that q_I is out of $\mathrm{supp}(Z_L)$. The support of Z_L near q_I is explicitly shown in Fig. 9(b), where purple triangles indicate the support qubits. It is worth noticing that $\mathrm{supp}(Z_L)$ may be in the boundary of a dj-CS, which is verifiable by comparing $\mathrm{supp}(Z_L)$ and the structure of a timelike joint of CSs shown in Fig. 7(c).

X_L and Z_L defined above anticommute with each other, considering that $\mathrm{supp}_Z(Z_L)$ and $\mathrm{supp}_X(X_L)$ meet at a pCQ q_{anti} in Fig. 9(a). A dual logical qubit is defined analogously, but now the logical operators are defined oppositely; $\mathrm{supp}(Z_L)$ surrounds a defect and $\mathrm{supp}(X_L)$ ends at each defect.

D. Initialization and measurement of a logical qubit

We first describe initializing a primal logical qubit to an eigenstate of X_L or Z_L. A dual logical qubit can be initialized analogously. As mentioned at the beginning of this section, in each initialization block, there is no input layer and the initialized state is prepared in the output layers Q_{OUT} (t_0- and (t_0+1)-layer) after the measurement step.

The X_L-initialization of a primal logical qubit is done by making the defects start from the t_0-layer. X_L given in Eq. (6) is then a part of a "cup-shaped" pc-CS S_X as shown in Fig. 9(c). Since $X_L S_X$ has the support out of the output qubits and commutes with each single-qubit measurement in the measurement step, the postmeasurement state is an eigenstate of $X_L S_X$. S_X is a stabilizer both before and after the measurement step due to Theorem 2. Therefore, the postmeasurement state is also an eigenstate of X_L and the eigenvalue is determined by the measurement result of $X_L S_X$.

The Z_L-initialization of a primal logical qubit is done by extending the defects to meet at a qubit before the t_0-layer, as shown in Fig. 9(d). Z_L given in Eq. (7) is then a part of a dj-CS S_Z which is a stabilizer. From an analogous argument, the postmeasurement state is an eigenstate of Z_L and the eigenvalue is determined by the measurement result of $Z_L S_Z$. The X_L- or Z_L-measurement is done by reversing the time order from the corresponding initialization process, as shown in Figs. 9(e) and 9(f). This time, Q_{IN} is the t_0-layer and Q_{OUT} is empty. Regarding the X_L-measurement, there exists a pb-CS S_X which is a stabilizer before the measurement step such that $X_L' := X_L S_X$ commutes with each single-qubit measurement in the measurement step. X_L is *equivalent* to X_L'; namely, $\langle\psi|X_L|\psi\rangle = \langle\psi|X_L'|\psi\rangle$ for every stabilized state $|\psi\rangle$ before the measurement step. Therefore, redefining X_L to X_L' does not change the logical state encoded in $|\psi\rangle$. The measurement result of X_L' can be directly obtained from the results of the measurement step. The Z_L-measurement process can be verified analogously.

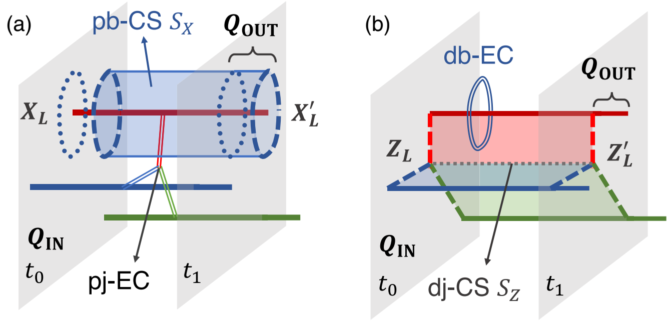


FIG. 10. Construction of the logical identity gate of a primal logical qubit between the input layer Q_{IN} (t_0-layer) and the output layers Q_{OUT} [t_1- and (t_1+1)-layer]. The gate is constructed by extending the defects from Q_{IN} to Q_{OUT}. The logical-X operator in Q_{IN} (Q_{OUT}) is X_L (X_L'), and Z_L and Z_L' are defined similarly. (a) X_L is transformed into X_L' via a pb-CS S_X surrounding the red defect, and (b) Z_L is transformed into Z_L' via a dj-CS S_Z ending at the three defects. Double lines indicate error chains causing logical errors covered in Sec. V B.

E. Elementary logical gates

1. Identity gate

The identity gate of a primal logical qubit is constructed just by extending the defects along the time axis between Q_{IN} (t_0-layer) and Q_{OUT} [t_1- and (t_1+1)-layer] as shown in Fig. 10. Let X_L and X_L' be the logical-X operators of the input and output logical qubits, respectively: $X_L := F_X^{\mathsf{br}}(t_0)$ and $X_L' := F_X^{\mathsf{br}}(t_1)$, where $F_X^{\mathsf{br}}(\cdot)$ is given in Eq. (6). We consider a pb-CS S_X which surrounds the red defect and ends at $\mathrm{supp}_Z(X_L)$ and $\mathrm{supp}_Z(X_L')$, as shown in Fig. 10(a). Since S_X is a stabilizer before the measurement step according to Theorem 1, X_L is equivalent to

$$\tilde{X}_L := S_X X_L = \left(\bigotimes_{q\in V_X} X(q)\right) X_L', \tag{8}$$

where $V_X := \mathrm{supp}(S_X X_L X_L') \subset V \setminus Q_{\mathrm{OUT}}$. After measuring qubits excluding output qubits, $\tilde{X}_L$ is *transformed* into

$$\left(\prod_{q\in V_X} x_q\right) X_L' := x_X X_L',$$

where x_q (z_q) is the $X(Z)$-measurement result of the qubit q. In other words,

$$\langle\psi|\tilde{X}_L|\psi\rangle = \langle\psi'|x_X X_L'|\psi'\rangle \tag{9}$$

holds, where $|\psi\rangle$ and $|\psi'\rangle$ are the states before and after the measurements, respectively. (See Appendix A for the proof.)

We do a similar thing on the Z_L operators. Denoting those of the input and output logical qubits as Z_L and Z_L', respectively, we consider a dj-CS S_Z ending at $\mathrm{supp}(Z_L)$, $\mathrm{supp}(Z_L')$, and the defects, as in Fig. 10(b). Z_L is then equivalent to

$$\tilde{Z}_L := S_Z Z_L = \left(\bigotimes_{q\in V_Z} X(q)\right)\left(\bigotimes_{q\in D_Z} Z(q)\right) Z_L',$$

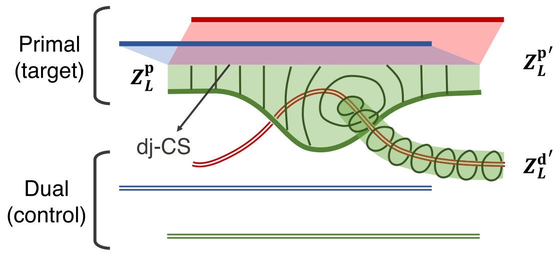


FIG. 11. Construction of the logical CNOT gate between a primal logical qubit (target) and a dual one (control). Each colored single (double) line indicates the primal (dual) defect of the corresponding color. $Z_L^{\mathsf{p}} \otimes I_L^{\mathsf{d}}$ is transformed into $Z_L^{\mathsf{p}'} \otimes Z_L^{\mathsf{d}'}$ via the presented dj-CS.

where $V_Z := \mathrm{supp}_X(S_Z Z_L Z_L') \subset V \setminus Q_{\mathrm{OUT}}$ and $D_Z := \mathrm{supp}_Z(S_Z Z_L Z_L') \subseteq D \setminus Q_{\mathrm{OUT}}$. After the measurement step, $\widetilde{Z}_L$ transforms into $x_Z z_Z Z_L'$ where $x_Z := \prod_{q\in V_Z} x_q$ and $z_Z := \prod_{q\in D_Z} z_q$.

The transformations of the logical operators are summarized as

$$X_L \to x_X X_L', \qquad Z_L \to x_Z z_Z Z_L'. \tag{10}$$

(Throughout this paper, we use prime symbols to distinguish the output logical operators from the input ones.) More explicitly, they are written as

$$\langle\psi|X_L|\psi\rangle = \langle\psi'|x_X X_L'|\psi'\rangle,$$
$$\langle\psi|Z_L|\psi\rangle = \langle\psi'|x_Z z_Z Z_L'|\psi'\rangle.$$

Therefore, the input logical state $|\psi_L\rangle$ encoded in $|\psi\rangle$ with the logical Pauli operators $\{X_L, Z_L\}$ is transformed into

$$|\psi_L'\rangle = X^{(1-x_Z z_Z)/2} Z^{(1-x_X)/2} |\psi_L\rangle$$

encoded in $|\psi'\rangle$ with the logical Pauli operators $\{X_L', Z_L'\}$. This transformation corresponds to the identity gate up to some by-product operators determined by the measurement results. The by-product operators can be handled by a software to be delayed to the end of the entire circuit and finally merged with the logical measurements [24].

The above arguments show the basic ideas for implementing logical gates. Regarding n logical qubits, let P_{Li} for each $P \in \{X, Z\}$ and integer $i \leqslant n$ denote the logical-P operator of the ith logical qubit. To construct a general logical gate U for n logical qubits, one should find a configuration of defects (and Y-planes for some gates) where a CS S_{Pi} exists for each P_{Li} satisfying the following conditions:

Condition 1. S_{Pi} should connect P_{Li} of the input logical qubits and $UP_{Li}U^\dagger$ of the output logical qubits. X_L (Z_L) of a logical qubit can be connected with primal (dual) CSs.

Condition 2. S_{Pi} should be compatible with all qubits except $\mathrm{supp}(P_{Li})$; it satisfies the relationships shown in Table II in that region.

If such CSs exist, the configuration implements the desired logical gate with some by-product operators obtained from the measurement results.

2. CNOT and primality-switching gates

We first consider a CNOT gate between a primal logical qubit (target) and a dual one (control). Figure 11 illustrates the defect configuration, where the pg-D of the primal logical qubit and the dr-D of the dual one are twisted one round with each other, which is commonly called *defect braiding*. The logical Pauli operators are transformed as

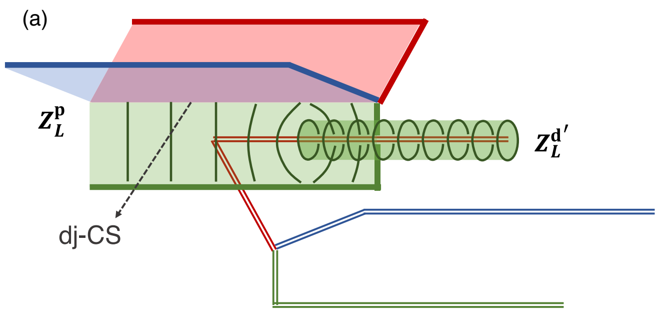
$$\begin{cases} X_L^{\mathsf{p}} I_L^{\mathsf{d}} \to X_L^{\mathsf{p}'} I_L^{\mathsf{d}'}, & I_L^{\mathsf{p}} X_L^{\mathsf{d}} \to X_L^{\mathsf{p}'} X_L^{\mathsf{d}'}, \\ Z_L^{\mathsf{p}} I_L^{\mathsf{d}} \to Z_L^{\mathsf{p}'} Z_L^{\mathsf{d}'}, & I_L^{\mathsf{p}} Z_L^{\mathsf{d}} \to I_L^{\mathsf{p}'} Z_L^{\mathsf{d}'}, \end{cases} \tag{11}$$

where the tensor product symbols and the sign terms such as x_X, x_Z, and z_Z in Eq. (10) are omitted, and each superscript p or d indicates the primality of the logical qubit. The above transformation is exactly the Heisenberg picture of a CNOT gate where the primal logical qubit is the target.

We need to find CSs satisfying two Conditions presented in Sec. IV E 1 to verify the transformations in Eq. (11). A dual CS for the transformation of $Z_L^{\mathsf{p}} \otimes I_L^{\mathsf{d}}$ is presented schematically in Fig. 11. Note that the "tunnel" of the CS along the dr-D must be formed since the dr-D cannot overlap with a dg-CS (see Table II). A CS for $I_L^{\mathsf{p}} \otimes X_L^{\mathsf{d}}$ can be constructed analogously; now a tunnel of a pr-CS is made along the pg-D. The other two transformations are straightforward.

Exploiting a CNOT gate discussed above, it is possible to make a *primality-switching* gate which changes a primal logical qubit to a dual one, by "closing" the input part of the dual one and the output part of the primal one, as shown in Fig. 12(a). Note that these closures indicate the Z_L-measurement of the primal one and the X_L-initialization of the dual one. The modified configuration is thus equivalent to the circuit in Fig. 12(b) up to by-product operators, which implements the identity or X_L gate while changing the primality. Alternatively, this result is directly obtainable by finding appropriate CSs; for example, the dj-CS in Fig. 12(a) verify the transformation of Z_L^{p} to $Z_L^{\mathsf{d}'}$. The primality-switching gate from a dual logical qubit to a primal one can be made similarly.

The primality-switching gate enables the CNOT gate between logical qubits with arbitrary primalities. Regardless of



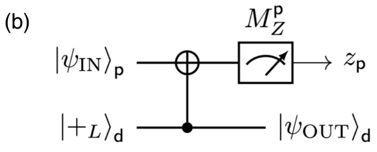

(b)
M_Z^{p}
$|\psi_{\mathrm{IN}}\rangle_{\mathsf{p}}$
z_{p}
$|+_L\rangle_{\mathsf{d}}$
$|\psi_{\mathrm{OUT}}\rangle_{\mathsf{d}}$

FIG. 12. (a) Construction of the primality-switching gate changing a primal logical qubit to a dual one. Z_L^{p} is transformed into $Z_L^{\mathsf{d}'}$ via the presented dj-CS. (b) A circuit equivalent to the primality-switching gate. M_Z^{p} is the Z_L-measurement on the primal qubit, and the result is z_{p}.

the primalities of the input logical qubits, one can switch them to primal (target) or dual (control), and apply a CNOT gate in Fig. 11.

Note that the equivalence between the different definitions of the X_L operator, related to the choice of the color pair (c, c') in Eq. (6), can be proven with the primality-switching gate. We consider a chain of two primality-switching gates: primal → dual → primal. No matter how X_L is defined in the first primal logical qubit, it becomes symmetric about the color in the dual one. We can thus transform it into any definition of X_L in the final primal one.

3. Hadamard gate

To construct a logical Hadamard gate, the logical Pauli operators should be transformed as

$$X_L \rightarrow Z_L', \qquad Z_L \rightarrow X_L'. \tag{12}$$

It is simple if the gate is located just after a state injection block presented in the Sec. IV F: injecting the unencoded state to a dual logical qubit instead of a primal one. This method is valid since the definitions of X_L and Z_L are opposite for primal and dual logical qubits.

If the Hadamard gate is located in the middle of the circuit, it is a bit tricky. Since X_L (Z_L) of a logical qubit can be connected only with primal (dual) CSs regardless of the primality of the logical qubit, there should be a CS having different primalities near the input and output layers, to achieve the transformation. To solve this problem, we construct a defect structure starting with a primal logical qubit and ending with a dual one as shown in Fig. 13, where the primal one stops at the primal t_H-layer and the dual one starts from the dual $(t_H + 1)$-layer. Each pair of defects with the same color must have the same spatial structure at $t = t_H$ and $t = t_H + 1$. Note that such a configuration is possible due to the self-duality of the 2D color codes, which makes primal and dual layers have the same structure.

We consider two pairs of overlapping primal and dual CSs: $(S_{Z\mathrm{p}}, S_{X\mathrm{d}})$ and $(S_{X\mathrm{p}}, S_{Z\mathrm{d}})$, where $S_{X\mathrm{p}}$, $S_{Z\mathrm{p}}$, $S_{X\mathrm{d}}$, and $S_{Z\mathrm{d}}$ are a pr-CS, dj-CS, pj-CS, and dr-CS defined in Fig. 13, respectively. $S_{ZX} := S_{Z\mathrm{p}}S_{X\mathrm{d}}$ then connects Z_L and X_L'. Similarly, $S_{XZ} := S_{X\mathrm{p}}S_{Z\mathrm{d}}$ connects X_L and Z_L'. Condition 1 in Sec. IV E 1 is thus satisfied with these two "hybrid" CSs. What remains is Condition 2. Since S_{ZX} and S_{XZ} contain Y operators on some CQs in the overlapping regions, the qubits should be measured in the Y-basis for the CSs to be compatible.

To make the Y-measurements fault-tolerant, we introduce *Y-planes*:

Definition 7 (Y-plane). A primal (dual) Y-plane is the set of p(d)CQs in a continuous area contained in a dual (primal) layer. CQs in Y-planes are measured in the Y-basis.

Errors in Y-planes can be corrected by an error correction procedure presented in Sec. V C. Therefore, the Y-measurements for the Hadamard gate can be fault tolerantly done by placing wide enough Y-planes to cover $\mathrm{supp}_Y(S_{ZX})$ and $\mathrm{supp}_Y(S_{XZ})$ completely. More details including microscopic pictures are presented in Sec. V C.

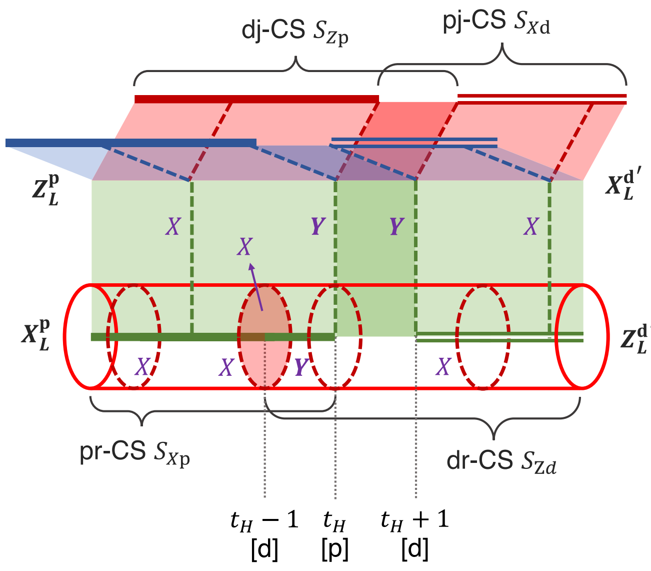


FIG. 13. Construction of the logical Hadamard gate from a primal logical qubit to a dual one. Each colored single (double) line is the primal (dual) defect of that color. $S_{Z\mathrm{p}}$ is a dj-CS ending at the three primal defects and the $(t_H + 1)$-layer. Similarly, $S_{X\mathrm{d}}$ is a pj-CS ending at the three dual defects and the t_H-layer. $S_{Z\mathrm{p}}$ and $S_{X\mathrm{d}}$ are chosen so that their supports overlap in the t_H- and $(t_H + 1)$-layer between the defects. Next, $S_{X\mathrm{p}}$ is a pr-CS which surrounds the pg-D and ends at the t_H-layer. $S_{Z\mathrm{d}}$ is a dr-CS which surrounds the dg-D and reaches the $(t_H - 1)$-layer. Note that $S_{Z\mathrm{d}}$ does not have a boundary in the $(t_H - 1)$-layer; instead, its interior is penetrated by the pg-D. This is possible since $S_{Z\mathrm{d}}$ and the pg-D have different primalities. $S_{X\mathrm{p}}$ and $S_{Z\mathrm{d}}$ are chosen so that their supports overlap in the t_H-layer. Finally, $S_{ZX} := S_{Z\mathrm{p}}S_{X\mathrm{d}}$ and $S_{XZ} := S_{X\mathrm{p}}S_{Z\mathrm{d}}$ transform the logical Pauli operators as Eq. (12). The supports of S_{ZX} and S_{XZ} are marked as colored dashed lines and a circle filled in red. In particular, their Y-support qubits are in the t_H- and $(t_H + 1)$-layer and measured in the Y-basis. For these Y-measurements to be fault-tolerant, dual and primal Y-planes are placed on the t_H- and $(t_H + 1)$-layer, respectively.

4. Phase gate

To construct a logical phase gate, the logical Pauli operators should be transformed as

$$X_L \rightarrow Y_L', \qquad Z_L \rightarrow Z_L'.$$

It can be achieved with the defect structure in Fig. 14(a): The defects of a logical primal qubits are just extended from the input layer to the output layer as the logical identity gate in Fig. 10. The transformation of Z_L is straightforward; it is the same as that in an identity gate in Fig. 10(b). X_L is transformed into Y_L' by a stabilizer $S_X := S_X^{(1)}S_X^{(2)}$, where $S_X^{(1)}$ and $S_X^{(2)}$ are CSs defined in Fig. 14(a) and 14(b) as follows: $S_X^{(1)}$ is a pj-CS connecting X_L and X_L'. It has the form of a pb-CS surrounding the pr-D near the input and output layers, but is deformed appropriately through joints in between. (See Fig. 7 for more details on joints.) $S_X^{(2)}$ is a dj-CS connecting a dual layer ($t = t_2$) and Z_L'. These two CSs can be chosen such that S_X contains Y operators in the t_2-layer as shown in Fig. 14(c) schematically as dotted lines.

To make S_X compatible between the input and output layers, a primal Y-plane is placed on the t_2-layer to cover the

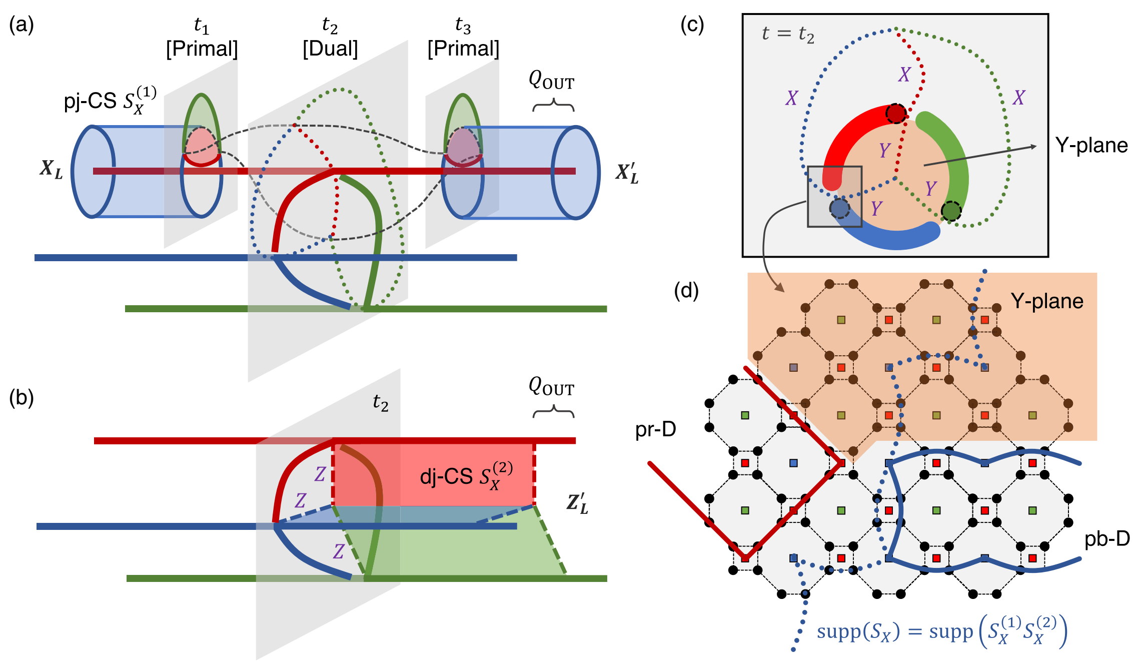


FIG. 14. Construction of the logical phase gate on a primal logical qubit. The input logical-X operator (X_L) is transformed into the output logical-Y operator (Y'_L) via a stabilizer $S_X^{(1)}S_X^{(2)}$, where $S_X^{(1)}$ and $S_X^{(2)}$ are CSs given as follows: A pj-CS $S_X^{(1)}$ presented in (a) connects X_L and X'_L. Near the input layer, $S_X^{(1)}$ has the form of a pb-CS surrounding the red defect. On the t_1-layer, it is divided into three CSs with different colors through a spacelike joint. Each CS is then deformed appropriately so that the joint is extended along the black dashed line and $\text{supp}_X(S_X^{(1)})$ contains the 1-chains on the t_2-layer (colored dotted lines). On the t_3-layer, the joint becomes spacelike again. After that, $S_X^{(1)}$ returns to the form of a pb-CS and is connected to X'_L. A dj-CS $S_X^{(2)}$ presented in (b) connects Z'_L and the 1-chains on the t_2-layer (colored dashed lines). In the t_2-layer, the defects are extended in a spacelike manner, and $S_X^{(1)}S_X^{(2)}$ has X and Y operators as shown in (c) and (d). In (c), the colored circles indicate the timelike defects penetrating the layer and the thick colored lines indicate the spacelike defects. By placing a primal Y-plane in the area surrounded by the spacelike defects, Y operators in $S_X^{(1)}S_X^{(2)}$ can be measured. In (d), the vicinity of the timelike pb-D is explicitly described. The colored solid lines indicate the cross sections of the spacelike defects, along which CQs are measured in the Z-basis.

Y operators. (The Y-plane does not affect the transformation of Z_L, since the CS used for the transformation is dual.) However, this is not enough; because of an issue regarding error correction near the boundary of the Y-plane, the defects need to be extended in a spacelike manner to surround the Y-plane, as shown in Fig. 14(c) schematically and in Fig. 14(d) explicitly near where two defects meet. More details on it are presented in Sec. V C and Appendix D.

F. State injection

We finish this section by introducing a state injection scheme. Preparation of an arbitrary logical qubit $a|0_L\rangle + b|1_L\rangle$ is essential for implementing a logical T gate as well as quantum computation with arbitrary input states. This is done in our scheme by injecting the corresponding unencoded state into a physical qubit.

We start from the configuration for the Z_L-initialization of a primal logical qubit shown in Fig. 9(c), where three defects meet at a point. First, a qubit $q_{\rm inj}$ in the pc-D for any color c is selected as an *injection qubit* which is the only input qubit in $Q_{\rm IN}$. We assume that the defect is "thicknessless" at $q_{\rm inj}$; namely, its cross section at $q_{\rm inj}$ contains at most one qubit as shown in Fig. 15(a). The desired initial state is injected into $q_{\rm inj}$ in an unencoded form $|\psi\rangle = a|0\rangle + b|1\rangle$, then the associated CZ gates are applied. Note that $q_{\rm inj}$ is measured in the X-basis as stated in Eq. (3). The X (Z) operator on $q_{\rm inj}$ is transformed into X_L (Z_L) up to a sign factor as shown in Fig. 15; thus the logical state $|\psi_L\rangle = a|0_L\rangle + b|1_L\rangle$ is prepared up to by-product operators.

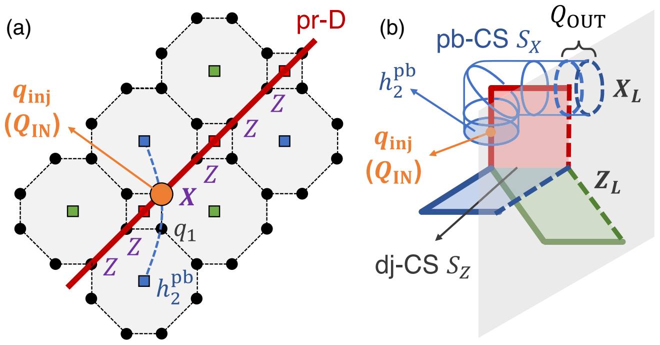


FIG. 15. State injection procedure. (a) An unencoded state is injected into an injection qubit $q_{\rm inj}$, which is the only input qubit, in the pr-D which is spacelike and thicknessless at $q_{\rm inj}$. $Z(q_{\rm inj})$ is invariant when the CZ gates associated with $q_{\rm inj}$ are applied. However, $X(q_{\rm inj})$ is transformed into $S(q_{\rm inj})$, where $S(q_{\rm inj})$ is the C-type SG around $q_{\rm inj}$. $S(q_{\rm inj})$ is equivalent to $S_{\rm CS}(h_2^{\rm pb})$ since $S_{\rm CS}(h_2^{\rm pb}) = S(q_{\rm inj})S(q_1)$, where $h_2^{\rm pb} \in H_2^{\rm pb}$ is the timelike 2-chain marked as a blue dashed line and q_1 is the marked CQ adjacent to $q_{\rm inj}$. $q_{\rm inj}$ is measured in the X-basis during the measurement step. (b) $S_{\rm CS}(h_2^{\rm db})$ is transformed into X_L of the output logical qubit via the pb-CS S_X. $Z(q_{\rm inj})$ is transformed into Z_L of the output logical qubit via the dj-CS S_Z.

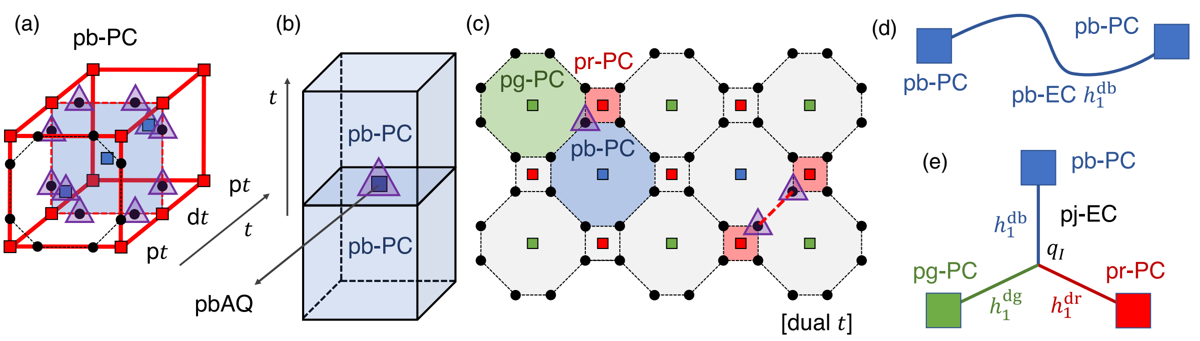


FIG. 16. (a) Explicit structure of a parity-check operator (PC), specifically a pb-PC in a 4-8-8 CCCS. Purple triangles indicate its X-support qubits. (b) A Z or X-measurement error on a pcAQ (purple triangle) flips two pc-PCs sandwiching q. (c) A dual layer of a 4-8-8 CCCS is presented. Purple triangles indicate the pCQs with errors. Each c-colored face corresponds to a flipped pc-PC, where an example is shown in (a) as a blue face on the dual layer. (d) A primal blue error chain (pb-EC), where every qubit along a connected dual 1-chain h_1^{db} has an error, flips two pb-PCs located at its two ends. (e) Starting from an error on a pCQ q_I, a pj-EC is constructed by multiplying a pc-EC ending at the flipped pc-PC for each color c to the error operator. A pj-EC flips three primal PCs located at its ends.

Note that the state injection procedure is inherently not fault-tolerant, since it uses an unprotected single-qubit state and the defect is thicknessless at q_{inj}. Therefore, magic state distillation is essential for the faithful T gate.

V. ERROR CORRECTION

Now we describe error correction schemes in CCCSs. We first consider the cases without defects and Y-planes, then investigate how they affect the scheme.

We consider four types of single-qubit errors: X, Y, and Z errors before the measurement step and measurement errors. We say that two sets of (single-qubit) errors are *equivalent* if they incur the same logical error. We also say that an error set is *trivial* if it does not incur any logical errors (i.e., it is equivalent to the identity). Note that a measurement error is equivalent to a Pauli error before the measurement; for example, an X-measurement error on a vacuum qubit is equivalent to a Z error before the measurement. Therefore, we can write any error set as a tensor product of Pauli operators. Additionally, regarding an error set e, the error set obtained by multiplying e, arbitrary stabilizers, and arbitrary Pauli operators commuting with the measurement pattern is equivalent to e.

A. Error correction in the vacuum

For error correction in the vacuum, we exploit *parity-check operators* (PCs) defined as follows:

Definition 8 (Parity-check operator). For each cell c, the CS

$$S_{\text{CS}}(\partial c) = X(\partial c)$$

is a parity-check operator (PC), where $S_{\text{CS}}(\cdot)$ is given in Eq. (2).

PCs are classified into six groups according to primalities and cell colors. Here the primality of a PC $S_{\text{CS}}(\partial c)$ is that of the shrunk lattice $\mathcal{L}$ containing the cell c, and its cell color is the color of the AQ $Q(c)$. Note that the cell color is different from the color of $\mathcal{L}$, as shown in Table I. We refer to a primal c-colored PC as a "pc-PC."

Note that a given dcAQ q corresponds to two primal cells: one for each of $\mathcal{L}^{\text{pc}_1}$ and $\mathcal{L}^{\text{pc}_2}$ where c, c_1, and c_2 are all different colors. However, the PCs corresponding to the cells are indeed the same, considering the structure of each cell shown in Fig. 6. We can thus regard that one AQ (q) corresponds to one PC, and denote it as $S_{\text{PC}}(q)$. The support of the pc-PC $S_{\text{PC}}(q)$ for a dcAQ q contains two pcAQs and multiple pCQs around q as shown in Fig. 16(a), where the purple triangles indicate the support qubits.

We now assume that there are no defects and Y-planes. Since vacuum qubits are measured in the X-basis, all PCs survive as stabilizers after the measurement step and thus can be used to detect Z errors on vacuum qubits. Note that X errors on them are trivial. The final step for error correction is to decode errors from the PC measurement results and correct the errors.

An error may occur on either an AQ or a CQ. An error on a pcAQ flips two pc-PCs sandwiching the qubit along the time axis as shown in Fig. 16(b), where the purple triangle indicates the qubit with an error. An error on a pCQ flips pr-PC, pg-PC, and pb-PC surrounding the qubit spatially, as shown in Fig. 16(c). If both the pCQs constituting a pcL have errors, the two pc-PCs connected by the link are flipped.

Combining the above facts, we conclude that, if every qubit in $Q(h_1^{\text{dc}})$ for a connected dual 1-chain $h_1^{\text{dc}} \in H_1^{\text{dc}}$ has an error, the pc-PC $S_{\text{PC}}(q)$ for each qubit $q \in Q(\partial h_1^{\text{dc}})$ is flipped, as shown in Fig. 16(d). Such an error set in the vacuum is called a primal c-colored *error chain*, referred to as a "pc-EC." Furthermore, starting from an error on a pCQ, each flipped PC may be "moved" by multiplying a primal error chain of the corresponding color ending at the PC. An error set constructed by this way flips three primal PCs located at its ends and is referred to as a "pj-EC." (A single-qubit error separated from other error sets is also regarded as a pj-EC by itself.) General error chains are obtained by connecting multiple pc-ECs for each color c and pj-ECs.

B. Error correction near defects

We now investigate the effects of a timelike pc-D on the above error correction scheme. More details on this subsection are presented in Appendix B.

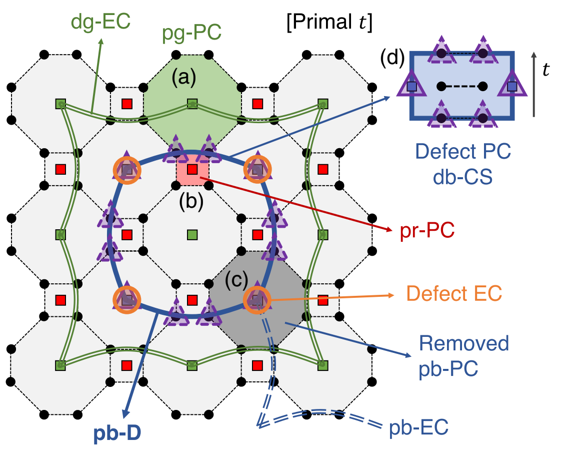


FIG. 17. PCs deformed or created due to a timelike pb-D in a 4-8-8 CCCS. Each purple triangle with a solid (or dashed) border indicates a defect qubit on the layer (or an adjacent dual layer). Incompatible pr-PCs and pg-CSs as shown in (a) and (b) are merged with each other properly to form a compatible stabilizer, whereas incompatible pb-PCs as shown in (c) are just removed. db-CSs whose boundaries are in the defect as shown in (d), called *defect* PC*s*, become compatible and are exploited for the error correction of defect qubits. There are two types of nontrivial undetectable error chains: dr-ECs and dg-ECs surrounding the defect (green double solid line) and pb-ECs ending at the defect (blue double dashed line). Additionally, an error set on defect qubits, called *defect error chains*, may be nontrivial and undetectable as well (orange circles).

Due to the existence of the defect, some PCs are merged, removed, or created, as shown in Fig. 17: First, all primal PCs whose supports contain defect qubits are no longer compatible, while dual PCs are unaffected. Incompatible pc′-PCs (c′ ≠ c) may be multiplied with each others to form larger compatible stabilizers, as the ones shown in Figs. 17(a) and 17(b). It is worth noting that such merged PCs are still local like ordinary PCs; i.e., their sizes are independent of the thickness of the defect. On the other hand, such processes are impossible for pc-PCs; thus they are just removed, as the one shown in Fig. 17(c). Therefore, undetectable pc-ECs may end at the defect. Last, dc-CSs whose Z-supports are in the defect (called *defect* PC*s*) are now compatible as the one shown in Fig. 17(d); thus they can be used for error correction. An error on a defect qubit flips adjacent defect PCs. As in the case of vacuum qubits, a series of errors on defect qubits (called *defect error chains*) flips defect PCs located at its ends. We now identify nontrivial undetectable error sets which may cause logical errors for primal logical qubits. Undetectable error chains in the vacuum are closed or end at defects. Considering the identity gate of a primal logical qubit, the shortest error chain inducing a Z_L error is a pj-EC ending at the three defects, and the shortest one inducing an X_L error is a closed dc-EC surrounding the pc′-D (c′ ≠ c), as shown schematically in Fig. 10. Note that a dc-EC surrounding the pc-D is trivial since every dc-CS ending at the defect commutes with it. Defect error chains also can be nontrivial and undetectable, as presented in Fig. 17. They may cause an X_L error in a logical identity gate since they may anticommute with pj-CSs ending at the defects. The *code distance* of a logical qubit is defined by the size of the smallest undetectable error set causing a logical error, and it increases as the defects get thicker or get farther from each other. Note that we have considered only timelike defects here. For spacelike defects, the arguments are slightly different; for example, an undetectable error chain may end at a spacelike defect of a different color. However, it still holds that the code distance depends on the thicknesses of defects and distances between them. See Appendix B for more details.

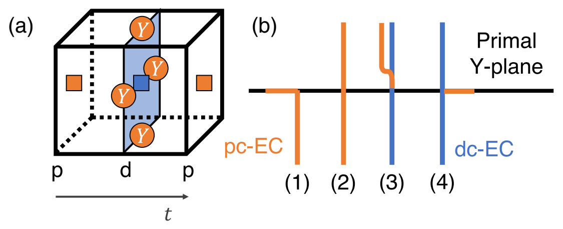


FIG. 18. (a) A primal hybrid PC for error correction in a primal Y-plane, constructed by multiplying a primal PC and the dual A-type SG around its center qubit. Circles and squares indicate links and AQs, respectively, and their colors mean their primalities: orange (primal) and blue (dual). The hybrid PC contains Y operators on CQs in the dual layer. (b) Undetectable error chains near a primal Y-plane. Orange (blue) lines are primal (dual) error chains. Undetectable primal error chains can behave as if there are no Y-planes, such as (1) and (2). However, if a dual error chain passes through the Y-plane, there should be a primal error chain of the same color ending at the intersection point such as (3) and (4), for a total error set to be undetectable.

C. Error correction near Y-planes

To correct errors in Y-planes, we use *hybrid* PC*s* defined as follows.

Definition 9 (Hybrid parity-check operator). For each d(p)cAQ q, the stabilizer $S_{\mathrm{PC}}(q)S_{\mathrm{A}}(q)$ is a primal (dual) c-colored hybrid parity-check operator denoted by p(d)c-HPC, where $S_{\mathrm{PC}}(q)$ is the p(d)c-PC corresponding to q and $S_{\mathrm{A}}(q)$ is the A-type SG around q.

As visualized in Fig. 18(a), a primal hybrid PC contains Y operators on CQs in a dual layer. Ordinary primal PCs intersecting a primal Y-plane are no longer compatible; thus primal hybrid PCs are used instead. A notable thing is that a hybrid PC contains both primal and dual qubits in its support. Therefore, a pc-HPC detects not only pc-ECs ending at it but also timelike dc-ECs penetrating it. As a consequence, in order for a dc-EC passing through a primal Y-plane to be undetectable, there should be a pc-EC ending at the pc-HPC located at the intersection point, such as (3) and (4) in Fig. 18(b). On the other hand, primal error chains can penetrate a primal Y-plane or progress in a spacelike manner in it without being detected, such as (1) and (2) in Fig. 18(b). However, they may end at the boundary of the Y-plane in contact with the vacuum since neither hybrid PCs nor ordinary PCs cannot be defined along the boundary.

As proposed in Sec. IV E, Y-planes are necessary to implement the logical Hadamard and phase gates. In Appendixes C and D, we verify that errors can be corrected well (namely, local nontrivial undetectable error sets near the Y-planes do not exist) while implementing these gates. Here an error set is

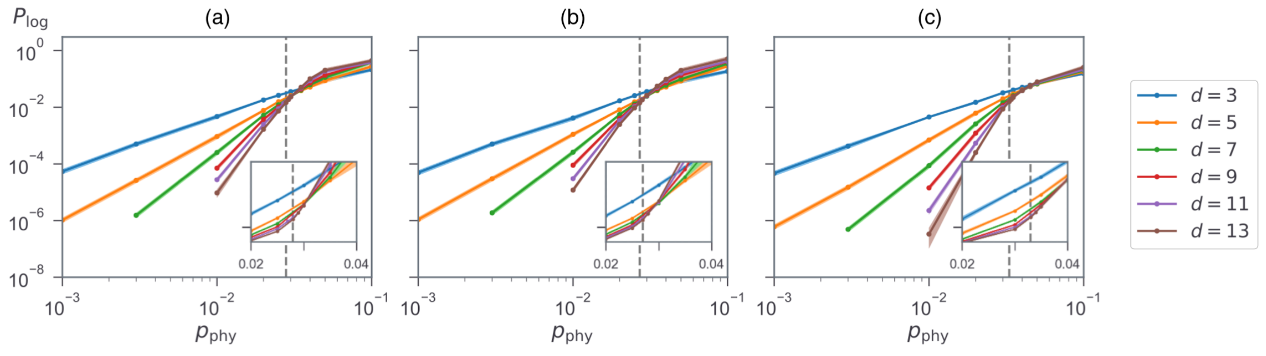


FIG. 19. Z_L error rate per two layers $P_{\log}$ vs nontrivial physical-level error rate $p_{\rm phy}$, for different code distances with respect to (a) 4-8-8 CCCSs, (b) 6-6-6 CCCSs, and (c) RTCSs. The small graphs show the results near the error thresholds. Pale areas around the lines indicate the 99% confidence intervals of $P_{\log}$. The error thresholds are obtained by using the results of the two largest code distances ($d = 11,\ 13$) and the values are 2.8% for 4-8-8 CCCSs, 2.7% for 6-6-6 CCCSs, and 3.3% for RTCSs, which are shown as gray dashed lines.

said to be *local* if its size is unrelated to the distances between the defects or their thicknesses.

VI. ERROR SIMULATIONS

We here numerically simulate correction of physical-level errors in MBQC via RTCSs and CCCSs and compare their *error thresholds*.

A. Error model

We assume a simple error model where vacuum qubits have nontrivial single-qubit errors independently with the same probability $p_{\rm phy}$. Note that these nontrivial errors contain X-measurement, Y, and Z errors as discussed in Sec. V; X errors on vacuum qubits cannot make logical errors.

B. Simulation methods

For each simulation with a code distance of d, we consider the identity gate of a primal logical qubit covering consecutive $2T + 1$ layers with $T = 4d + 1$ starting from a primal layer. Simplified defect models presented in Appendix H 1 are used, instead of considering big areas containing the entire defects. We calculate the Z_L error rate per two layers with the Monte Carlo method; we repeat a sampling cycle many times enough to obtain a desired confidence interval of the Z_L error rate. Each cycle is structured as follows.

We first prepare a cluster state whose shape and size are determined by d and T. Here we assume perfect preparation, namely, no qubit losses or failures of CZ gates. Errors are then randomly assigned to primal qubits with a given probability $p_{\rm phy}$, except those in the first and final layers to prevent error chains ending at these layers. After that, the outcomes of primal PCs are calculated, then decoded to locate errors. Edmonds' minimum-weight perfect matching (MWPM) algorithm [64–66] via Blossom V software [67] is used for decoding (once for RTCSs and six times for CCCSs); more details on this are presented in Appendix H 2. We then identify primal error chains connecting different defects which incur Z_L errors by comparing the assigned and decoded errors. We count such error chains while repeating the cycles and obtain the Z_L error rate per two layers $P_{\log}$. The error threshold $p_{\rm thrs}$ is obtained from the calculated $P_{\log}$ results for different values of d and $p_{\rm phy}$; $P_{\log}$ decreases as d increases if $p_{\rm phy} < p_{\rm thrs}$ and vice versa if otherwise.

C. Results

Figure 19 shows the results of the simulations: Z_L error rates ($P_{\log}$) against nontrivial physical-level error rates ($p_{\rm phy}$) for different MBQC schemes and code distances (d). The obtained error thresholds are about 2.8% for 4-8-8 CCCSs, 2.7% for 6-6-6 CCCSs, and 3.3% for RTCSs. The values for CCCSs are slightly lower than the value for RTCSs, but they have similar orders of magnitude.

VII. RESOURCE ANALYSIS

A. Resource overheads for placing logical qubits

We first analyze the minimal resource overheads required to place logical qubits in RTCSs or CCCSs. We consider two schemes for RTCS computation: defect-based and patch-based ones. In the defect-based scheme [38,41–43], each logical qubit is encoded in a pair of defects and logical operations are done by defect braiding or state distillation. In the patch-based scheme [44], logical qubits are encoded in square "patches" separated from each other and logical operations are done by lattice surgery or state distillation. For CCCS computation, we consider two types of lattices: 4-8-8 and 6-6-6.

Except for the patch-based RTCS scheme, we consider a periodic hexagonal arrangement of parallel timelike primal defects as shown in Appendix E, where primal logical qubits with the code distance of d are compactly packed in the space. In other words, the spaces between defects are determined to minimize the number of physical qubits per logical qubit while keeping all the possible nontrivial undetectable error chains to contain d or more qubits. In Appendix E, we first optimize the arrangements while ignoring the implementation of nontrivial logical gates. We then investigate how the arrangements should be changed to make it possible to implement each logical gate on an arbitrary logical qubit (or an arbitrary pair of logical qubits) while keeping the code distance the same. In particular, for CCCS computation, we first get the arrangements for implementing each one of the gates (the CNOT, Hadamard, and phase gates) and then the arrangements

TABLE III. Resource overheads of RTCS and CCCS computation, evaluated by the numbers of physical qubits (n) and CZ gates (N_{CZ}) per layer in terms of the code distance (d) and the number of logical qubits (k). Only the leading-order terms on d are presented. For RTCS computation, the patch-based and defect-based schemes are considered. For CCCS computation, the 4-8-8 and 6-6-6 lattices are considered. Except for the patch-based RTCS scheme, we regard optimal hexagonal arrangements of parallel timelike primal defects shown in Appendix E. The arrangements are optimized while either ignoring all nontrivial logical gates, considering only one type of logical gate, or considering general gates.

Considered logical gates	n/k	N_{CZ}/k
(a) Defect-based RTCS computation		
—	$6.6d^2$	$13d^2$
CNOT	$6.6d^2$	$13d^2$
(b) Patch-based RTCS computation		
—	$3d^2$	$6d^2$
CNOT	$6d^2$	$12d^2$
(c) 4-8-8 CCCS computation		
—	$7.5d^2$	$20d^2$
CNOT	$7.5d^2$	$20d^2$
Hadamard	$25d^2$	$66d^2$
Phase	$19d^2$	$50d^2$
General	$32d^2$	$84d^2$
(d) 6-6-6 CCCS computation		
—	$6.3d^2$	$17d^2$
CNOT	$6.7d^2$	$18d^2$
Hadamard	$27d^2$	$72d^2$
Phase	$15d^2$	$39d^2$
General	$29d^2$	$77d^2$

where arbitrary logical gates are applicable. Note that we here do not consider implementing multiple adjacent gates at the same time.

Table III shows the calculated values of n/k and N_{CZ}/k in terms of the code distance d, where k is the number of logical qubits and n (N_{CZ}) is the number of required physical qubits (CZ gates) per layer. Note that N_{CZ}/n is 2 for RTCSs and 8/3 for CCCSs. It is observed that the values of n/k are not significantly different from scheme to scheme if only the CNOT gates are considered. However, the Hadamard and phase gates with CCCSs require relatively large values of n/k. Note that, in a real implementation, the arrangement of logical qubits does not always have to be the most general one which is the most costly; thus n/k lies somewhere between these values. Depending on its purpose, not all types of logical gates may need to be available for each qubit.

B. Resource overheads for nontrivial logical gates

Considering the results of the previous subsection, our CCCS scheme seems to be worse than the RTCS schemes in terms of resource efficiency. However, we remark that those results show only physical qubits per logical qubit in a layer, not the real numbers of physical qubits to implement each logical gate, which is investigated in this subsection. To calculate them, we need to know the number of layers required for each gate, which is analyzed in Appendix F.

TABLE IV. Numbers of physical qubits required for the logical CNOT gates with RTCSs or CCCSs. Only the leading-order terms on d are presented. We consider the two cases for CCCS computation: Defects are arranged so that (a) only the CNOT gate or (b) all logical gates are applicable. The results of Table III are used to obtain the numbers of physical qubits per layer.

Type	No. per layer	No. of layers	Total No.
Defect-based RTCS	$13d^2$	$4.5d$	$59d^3$
Patch-based RTCS[a]	$12d^2$	$4d$	$48d^3$
4-8-8 CCCS (a)	$15d^2$	$4d$	$60d^3$
6-6-6 CCCS (a)	$13d^2$	$4.4d$	$60d^3$
4-8-8 CCCS (b)	$63d^2$	$4d$	$250d^3$
6-6-6 CCCS (b)	$56d^2$	$4.4d$	$250d^3$

[a]If the two logical qubits are not adjacent, about $4d$ layers are additionally needed.

Table IV presents the number of physical qubits required to implement a CNOT gate for each MBQC scheme. For CCCS computation, we consider the two cases: Defects are arranged so that only the CNOT gate or all logical gates are applicable. The numbers of physical qubits per layer are directly obtained from the results of Table III. Considering the most general arrangement of defects in each scheme, a CNOT gate requires about five times more physical qubits in CCCS computation than in RTCS computation, for the same code distances.

We now evaluate resource overheads for the logical phase gates, assuming that the gates are implemented by state distillation in RTCS computation. A logical ancilla state $|Y_L\rangle := |0_L\rangle + i|1_L\rangle$ is used to implement a phase gate in RTCS computation through the circuit in Fig. 20(a). If $|Y_L\rangle$ has an error and the other parts of the circuit are perfect, the resulting

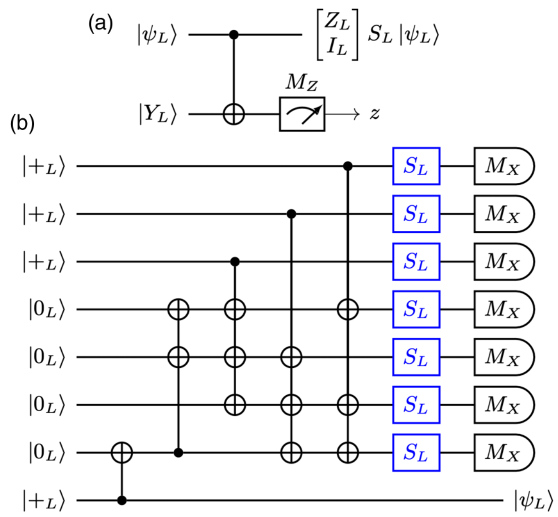


FIG. 20. (a) Implementation of a logical phase gate S_L with an ancilla logical state $|Y_L\rangle := (|0_L\rangle + i|1_L\rangle)/\sqrt{2}$. $Z_L S_L$ or S_L is applied on the input state if the Z_L-measurement result z is $+1$ or -1, respectively. (b) Distillation circuit for a $|Y_L\rangle$ state [24]. Each S_L gate is implemented with a noisy $|Y_L\rangle$ state by the circuit in (a). The X_L-measurement (M_X) results determine whether the distillation succeeds or not. If it succeeds, the distilled state is obtained from $|\psi_L\rangle$.

TABLE V. Numbers of physical qubits required for the logical phase gates in defect-based (DB) RTCS, patch-based (PB) RTCS, or CCCS computation. Only the leading-order terms on d are presented. For each RTCS scheme, we consider the two cases: The distillation cycle ("Dist.") is repeated once or twice. For CCCS computation, we assume that the defects are arranged so that all logical gates are applicable. Lower bounds of residual errors in the output $|Y_L\rangle$ states are calculated for the cases of RTCS computation. The bounds are achieved when logical errors do not occur during the distillation processes.

Type		No. of physical qubits	Residual error[a]
DB RTCS	Dist. once	$1000d^3$	$\geqslant 7\epsilon^3$
	Dist. twice	$7900d^3$	$\geqslant 7^4\epsilon^9$
PB RTCS	Dist. once	$840d^3$	$\geqslant 7\epsilon^3$
	Dist. twice	$6400d^3$	$\geqslant 7^4\epsilon^9$
4-8-8 CCCS		$32d^3$	-
6-6-6 CCCS		$28d^3$	-

[a] ϵ is the error probability of the initial noisy $|Y_L\rangle$ obtained by state injection.

phase gate also has an error. Seven noisy $|Y_L\rangle$ states can be distilled to obtain a less noisy $|Y_L\rangle$ state [24,38] via the circuit shown in Fig. 20(b); if each input state has an X_L or Z_L error with a probability of ϵ and the distillation process is perfect, the output state has a probability $7\epsilon^3$ of having an error. The success probability of the distillation process is $1 - 7\epsilon$.

The numbers of physical qubits required for the logical phase gates are analyzed in Appendix F (for CCCS computation) and Appendix G (for RTCS computation), and the results together with the residual errors $\epsilon_{\rm res}$ in the distilled $|Y_L\rangle$'s are presented in Table V. For the calculations, we assume that logical qubits are arranged in the way discussed in Appendix F. The results clearly show that a phase gate with CCCSs is significantly more resource-efficient (at least about 26 times) than with RTCSs. Note that the nondeterminacy of distillation is not considered here; if considering it, the difference in resource overheads gets even bigger.

It is inappropriate to directly compare the Hadamard gates in the two schemes since $R_X(\pi/2) := \exp(i\frac{\pi}{4}X)$ is used in RTCS computation to complete a universal set of gates instead of the Hadamard gate [38,44]. Since $R_X(\pi/2)$ is also implemented by state distillation with the state $|A_L\rangle$, we can at least say that the Hadamard gate in CCCS computation is a more resource-efficient element to complete a universal set of gates than $R_X(\pi/2)$ in RTCS computation. In detail, the circuit to implement $R_X(\pi/2)$ is the same as the one shown in Fig. 20(a) except that the target and control of the CNOT gate are swapped and the ancilla logical qubit is measured in the X-basis, not the Z-basis. Therefore, it requires almost the same number of physical qubits as the logical phase gate shown in Table V. On the other hand, a Hadamard gate in CCCS computation requires only about $90d^2$ physical qubits for both the 4-8-8 and 6-6-6 lattice since it needs three consecutive layers: the $(t_H - 1)$-, t_H-, and $(t_H + 1)$-layer in Fig. 13.

The above analyses may be not fair comparisons, since the same code distance does not mean the same level of protection against errors. Thus, in Fig. 21 we illustrate the estimated numbers (n) of physical qubits required for each logical gate

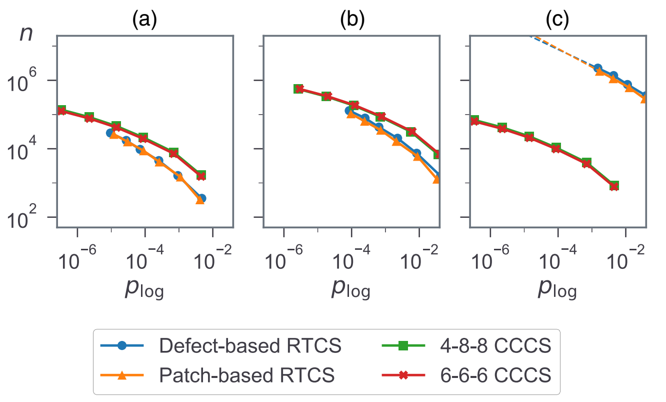


FIG. 21. Estimated numbers of physical qubits required for (a) an identity gate, (b) a CNOT gate, or (c) a phase gate vs the logical error rate $P_{\log}$ for CCCS and RTCS computation, while fixing the physical-level error rate $p_{\rm phy}$ to 1%. Optimal arrangements of logical qubits allowing general logical gates presented in Appendix F are considered. For (a), it is assumed that the total numbers of layers are equal to twice the code distances. For RTCS computation in (c), we consider using the state distillation cycle once to implement the phase gate. Extrapolated values for RTCS computation are shown as the dashed lines. Note that these results, particularly (b) and (c), are rough estimations since we use the results in Sec. VI which cover only Z_L errors in the identity gates.

against achievable logical error rates ($p_{\log}$) while fixing the physical-level error rate ($p_{\rm phy}$) to 1%, considering the optimal arrangements allowing general logical gates. For the identity gate, we assume that the number of layers is equal to twice the code distance d. It apparently shows that, although the identity and CNOT gates with CCCSs are slightly more costly than those with RTCSs, the phase gate with CCCSs is significantly more resource-efficient than that with RTCSs. Note that these results are rough estimations since they are obtained by using the logical error rates calculated in Sec. VI which covers only Z_L errors in the identity gates. More precisely, for each logical gate, scheme, and code distance, we here assume that the logical error rate is equal to $p_{Z_L} \sum_i T_i$, where p_{Z_L} is the corresponding Z_L error rate per two layers calculated in Sec. VI and T_i is the number of layers demanded by the ith logical qubit participating in the gate (which can be an ancillary logical qubit for distillation).

We last remark that state distillation is not the only method to implement the Hadamard and phase gates with RTCSs. These gates can be implemented by lattice dislocations [44,50] as mentioned in Sec. I, which may lead to small resource overheads comparable to those in CCCS computation. However, we then need to sacrifice the regularity of the lattices, which is another obstacle to realization.

VIII. REMARKS

In this paper, we have proposed a topological measurement-based quantum computation (MBQC) scheme via color-code-based cluster states (CCCSs). We have shown that our scheme is comparable with or even better than the conventional scheme via Raussendorf's 3D cluster states (RTCSs) [38,41–43], in the following three aspects:

(1) *Universality.* Initializations and measurements of logical qubits and all the elementary logical gates constituting a universal set of gates (the CNOT, Hadamard, phase, and T gates) can be implemented via appropriate placement of defects and Y-planes. We described each one of them explicitly in Sec. IV.

(2) *Fault tolerance.* We suggested the error correction scheme for each area of qubits in Sec. V. We further verified in Sec. VI that the error thresholds for errors in the vacuum have a similar order of magnitude with the values for RTCSs.

(3) *Hardware efficiency.* Contrary to the case of using RTCSs, the Hadamard and phase gates are implemented natively with CCCSs, due to the nature of the self-duality of the 2D color codes. One way to implement these gates using RTCSs is to use state distillation, but it typically consumes many ancillary logical qubits [22,36,38]. In Sec. VII B we verified quantitatively that the phase gate in CCCS computation demands significantly less physical qubits (at least about 26 times) than that the gate in RTCS computation implemented by state distillation. Other known methods to implement these gates with RTCSs require lattice dislocations [44,50] to the best of our knowledge. Although they are more resource-efficient than using distillation, the regularity of the lattices should be sacrificed, which may be undesirable from a practical point of view. Our protocol with CCCSs does not have such a problem as well; it always uses strictly regular lattices.

We particularly emphasize the last aspect on hardware efficiency as a definite improvement from the previous schemes, which makes our scheme an easier-to-implement alternative to those.

Our work has several limitations. First, logical T gates still need costly state distillation. Some methods to significantly reduce the cost of distillation have been proposed, such as using logical qubits with low code distances as ancilla qubits [68] or exploiting redundant ancilla encoding and flag qubits [69]. Moreover, 3D gauge color codes [30–35,70,71] enables non-Clifford gates without distillation. It may be possible to translate these protocols to be applicable for our MBQC scheme. We also assume the perfect preparation of cluster states, which is unrealistic. It is unclear how much the fault tolerance gets weaker if we consider qubits losses or failures of CZ gates, which is particularly related to photon losses in optical systems. It will be interesting future works to further investigate and resolve these problems.

ACKNOWLEDGMENTS

This work was supported by the National Research Foundation of Korea (NRF-2019M3E4A1080074, NRF-2020R1A2C1008609, NRF-2020K2A9A1A06102946) via the Institute of Applied Physics at Seoul National University and by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (IITP-2021-2020-0-01606 and IITP-2021-0-01059).

APPENDIX A: VERIFICATION OF EQ. (9)

Here we verify Eq. (9):

$$\langle\psi|\widetilde{X}_L|\psi\rangle = \langle\psi'|x_X X'_L|\psi'\rangle. \tag{A1}$$

We first assume $V_X = \{q_0\}$ for a qubit q_0. Since there exists a stabilizer S_0 anticommuting with $X(q_0)$ before the measurements, $\langle\psi|X(q_0)|\psi\rangle = 0$. Thus,

$$\begin{aligned}|\psi'\rangle &= \left|\left|\frac{I + x_{q_0}X(q_0)}{2}|\psi\rangle\right|\right|^{-1}\frac{I + x_{q_0}X(q_0)}{2}|\psi\rangle\\ &= \frac{I + x_{q_0}X(q_0)}{\sqrt{2}}|\psi\rangle\end{aligned}$$

holds, which gives

$$\begin{aligned}\langle\psi'|x_X X'_L|\psi'\rangle &= \langle\psi|[I + x_{q_0}X(q_0)]x_{q_0}X'_L|\psi\rangle\\ &= x_{q_0}\langle\psi|X'_L|\psi\rangle + \langle\psi|\widetilde{X}_L|\psi\rangle.\end{aligned}$$

Since $\widetilde{X}_L$ commutes with all stabilizers before the measurements, S_0 also anticommutes with $X(q_0)\widetilde{X}_L = X'_L$, and thus $\langle\psi|X'_L|\psi\rangle$ vanishes. Hence, we get Eq. (A1). For an arbitrary V_X with $|V_X| > 1$, we can show Eq. (A1) by simply repeating this process for every qubit in V_X.

APPENDIX B: DETAILS ON ERROR CORRECTION NEAR DEFECTS

We here investigate error correction near defects in more detail than Sec. V B. We introduce a pc-D $D_{\mathsf{pc}} = D(h_2)$ for a 2-chain h_2 where $D(\cdot)$ is given in Eq. (4). First, all primal PCs whose supports contain defect qubits are no longer compatible, while dual PCs are unaffected. Incompatible PCs may be multiplied with each other to form larger compatible stabilizers. Such processes are possible for pc′-PCs (c′ ≠ c) contacting with timelike surfaces of D_{pc} (namely, timelike areas of h_2), as shown in Fig. 22(a) where a pr-PC and pg-PC adjacent to a pb-D are merged. It is worth noting that merged PCs are still local like ordinary PCs; i.e., their sizes are independent of the thicknesses of D_{pc}. Other types of incompatible PCs cannot be merged in such a way, and thus they are just removed. These include pc′-PCs (c′ ≠ c) contacting with spacelike surfaces of D_{pc} [e.g., the pr-PC in Fig. 22(b)] and pc-PCs (e.g., the pb-PCs in Fig. 22). Last, there are additional stabilizers which become compatible due to D_{pc}: dual CSs whose Z-supports are in the defect. These stabilizers include dc-CSs $\{S_{\mathrm{CS}}(f) \mid f \in h_2\}$ (e.g., the db-CSs in Fig. 22), and if D_{pc} is spacelike, they also include dc′-CSs on the spacelike surfaces of D_{pc} [e.g., the dg-CS in Fig. 22(b)]. Such CSs are called *defect* PC*s* and used for error correction in defect qubits.

We now identify nontrivial undetectable error sets regarding primal logical qubits. Let us first consider errors on primal vacuum qubits. Since pc-PCs adjacent to a pc-D are removed, undetectable pc-ECs can end at the defect as shown in Fig. 22(a). On the other hand, undetectable pc′-ECs (c′ ≠ c) cannot end at the defect in general, but they may change their colors while passing through the defect due to merged PCs. As an exception, if the defect is spacelike, undetectable pc′-ECs can end at its spacelike surfaces. In order for a primal error chain to be nontrivial and undetectable, it should end at multiple different primal defects. For example, for a logical identity gate of a primal logical qubit, a pj-EC ending at the three defects incurs a Z_L error as shown in Fig. 10(a).

Next, let us consider errors on dual vacuum qubits. All the dual PCs remain compatible even if a primal defect is placed,

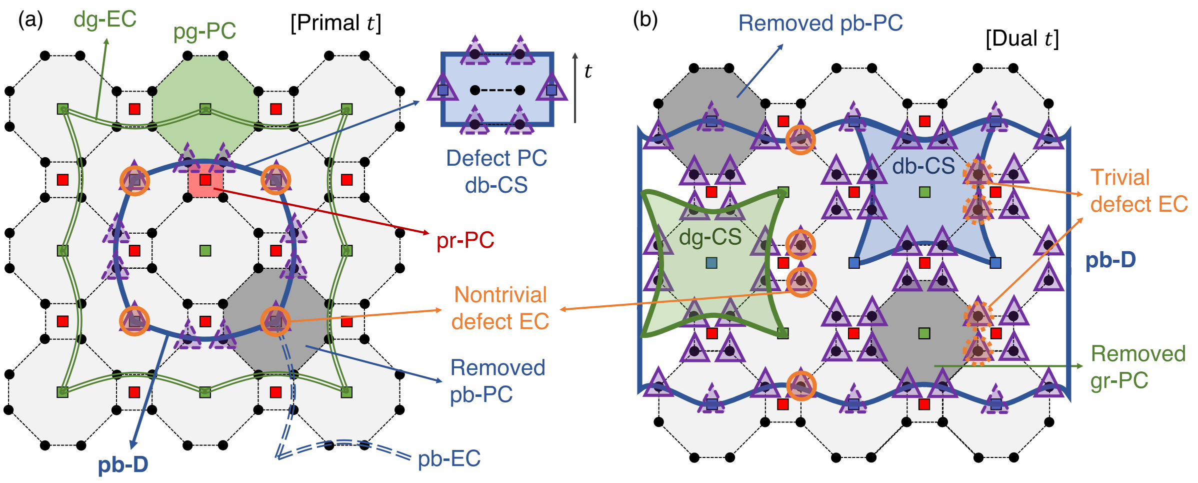


FIG. 22. PCs deformed or created due to a (a) timelike or (b) spacelike pb-D in a 4-8-8 CCCS. Each purple triangle with a solid (or dashed) border indicates a defect qubit on the layer (or an adjacent dual layer). In (a), incompatible pr-PCs and pg-PCs such as the ones colored in red and green are merged with each other properly to form a compatible stabilizer. However, incompatible pb-PCs such as the one colored in gray cannot be merged in such a way, and thus they are just removed. db-CSs whose boundaries are in the defect (blue square), called *defect* PC*s*, become compatible and are exploited for error correction in defect qubits. There are two types of nontrivial undetectable error chains: dr-ECs and dg-ECs surrounding the defect (green double solid line) and pb-ECs ending at the defect (blue double dashed line). Additionally, an error set on defect qubits, called *defect error chains*, may be nontrivial and undetectable as well (orange circles). In (b), it is the same as (a) that incompatible pb-PCs are removed and db-CSs in the defect become compatible. However, incompatible pr-PCs and pg-PCs are also removed, not merged with each other, which makes it possible that pr-ECs and pg-ECs end at the spacelike surfaces of the defect. Additionally, dg-CSs in the surfaces also become compatible so can be used as defect PCs. The orange circles with solid borders indicate part of a nontrivial defect error chain which goes around the surface of the defect. Note that there are undetectable defect error chains which do not go around the defect but just traverse the spacelike surface of the defect, such as the orange circles with dotted borders. Such defect error chains are trivial since they commute with db-CSs ending at the defect.

and thus a dual error chain cannot end at the defect. Therefore, every undetectable dual error chain is closed. A closed dc-EC E surrounding a pc′-D is nontrivial if $c' \neq c$ [e.g., the dg-EC in Fig. 22(a)] since it may anticommute with dc′-CSs ending at the defect. For example, for a logical identity gate of a primal logical qubit, a db-EC surrounding the pr-D incurs an X_L error as shown in Fig. 10(b).

Finally, errors on defect qubits also may incur logical errors. An error on a defect qubit flips adjacent two or three defect PCs. Similar to the case of vacuum qubits, a series of errors on defect qubits (called *defect error chains*) flips defect PCs located at its ends. For a defect error chain to be nontrivial and undetectable, it should go around the surface of the defect once, as shown in Fig. 22(a) where the defect error chain marked as orange circles may anticommute with db-CSs ending at the defect (see also Fig. 8). Otherwise, the defect error chain shares an even number of qubits with a db-CS and thus does not incur a logical error. Furthermore, since defect PCs contain X operators on dual vacuum qubits, there exist undetectable error sets containing both dual and defect error chains. For example, a dc-EC penetrating a dc′-D flips defect PCs; thus there should be defect error chains ending at these defect PCs for the total error set to be undetectable. Note that we can always obtain a dual error chain equivalent to a defect error chain e_D by multiplying A- or C-type SGs around the qubits in supp(e_D).

The *code distance* is determined by the size of the smallest nontrivial undetectable error set. Two factors are determining the code distance: distances between defects and their thicknesses. The former one is related to error chains ending at different defects, while the latter one is related to those surrounding the defects and defect error chains. Note that the shortest nontrivial undetectable defect error chain is generally shorter than the shortest nontrivial undetectable dual error chain, as shown visually in Fig. 22(a), although comparing them directly may be unfair if the error model used is biased.

APPENDIX C: ERROR CORRECTION IN A LOGICAL HADAMARD GATE

We here investigate error correction in a Hadamard gate shown in Sec. IV E 3, especially near the Y-planes. Two consecutive Y-planes are required for a Hadamard gate as shown in Fig. 13: a dual Y-plane at the end of the primal defects and a primal one at the end of the dual defects. As shown in Fig. 23(a), each Y-plane completely cover the three defects. Since there are undetectable error chains connecting the defects and the boundary of the Y-planes, the Y-planes should be wide enough so that such error chains are longer than the code distance. Figure 23(b) shows the dual Y-plane near the pb-D, where the orange circles indicate Y-plane qubits. All dual (primal) PCs intersecting the dual (primal) Y-plane are replaced with the corresponding hybrid PCs suggested in Sec. V C. Exceptionally, dc-HPCs and pc-HPCs overlapping with the pc-D or dc-D are incompatible since their supports contain defect qubits. Instead, each pair of them adjacent in a timelike manner can be merged to form a compatible stabilizer which can be used for error correction. Additionally, defect PCs (see Fig. 17) intersecting the Y-planes are also incompatible; thus

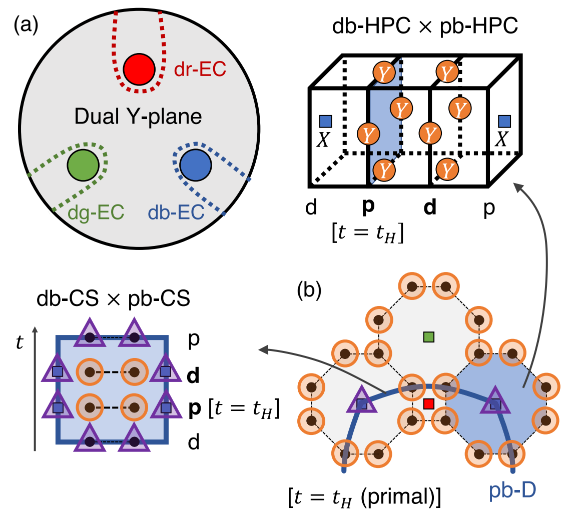


FIG. 23. Error correction during the process for a Hadamard gate, particularly near the Y-planes. (a) The Y-planes should be wide enough since there are error chains connecting their boundaries and the defects. The dual Y-plane and the primal defects are shown as an example. (b) Dual Y-plane in the t_H-layer near the pb-D. Defect (Y-plane) qubits are marked as purple triangles (orange circles). The same structure is repeated in the next layer for the primal Y-plane and the db-D. A db-HPC around the defect pbAQ is no longer compatible and so is the next pb-HPC; thus they are merged to be compatible. Similarly, a defect PC in the pb-D intersecting the Y-plane is merged with the adjacent defect PC in the db-D to be compatible.

each pair of them adjacent in a timelike manner should be merged to form a compatible stabilizer.

We now verify that local nontrivial undetectable error sets do not occur during the implementation of a Hadamard gate. Throughout this Appendix, it is assumed that the Y-planes are wide enough; thus their boundaries do not need to be considered. The problems then may happen near where the Y-planes and defects meet.

Instead of investigating the configuration for a Hadamard gate in Fig. 13 directly, we introduce a simpler system $\mathcal{S}_I$ visualized in Fig. 24, where the primal defects are just extended straightly instead of changing to dual defects. Let $\mathcal{S}_H$ denote

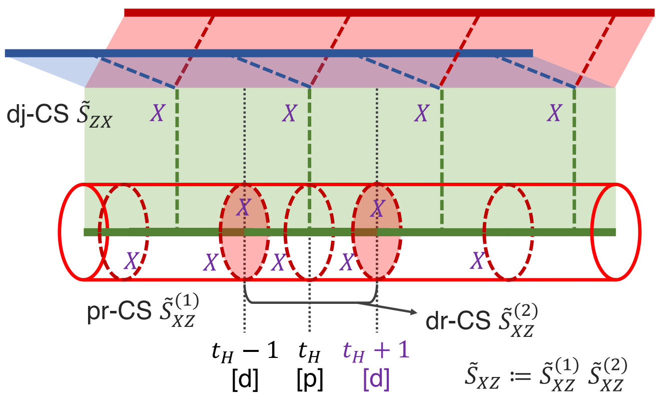

dj-CS $\tilde{S}_{ZX}$
X
pr-CS $\tilde{S}^{(1)}_{XZ}$
dr-CS $\tilde{S}^{(2)}_{XZ}$
$t_H - 1$ t_H $t_H + 1$
[d] [p] [d]
$\tilde{S}_{XZ} := \tilde{S}^{(1)}_{XZ}\, \tilde{S}^{(2)}_{XZ}$

FIG. 24. Configuration of the system $\mathcal{S}_I$ introduced to verify error correction in a Hadamard gate, where the primal defects are just extended straightly instead of changing to dual defects.

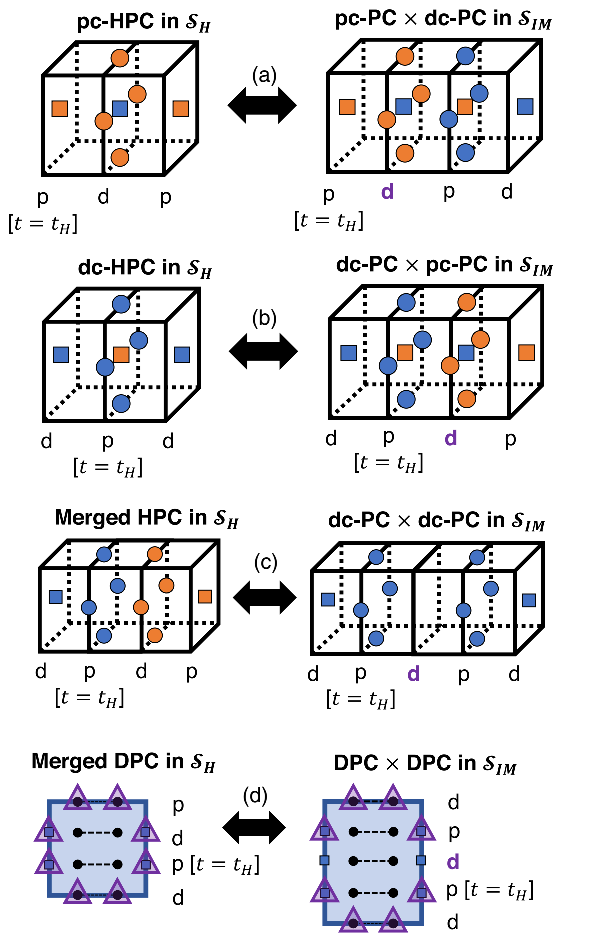

pc-HPC in $\mathcal{S}_H$
pc-PC × dc-PC in $\mathcal{S}_{IM}$
(a)
dc-HPC in $\mathcal{S}_H$
dc-PC × pc-PC in $\mathcal{S}_{IM}$
(b)
Merged HPC in $\mathcal{S}_H$
dc-PC × dc-PC in $\mathcal{S}_{IM}$
(c)
Merged DPC in $\mathcal{S}_H$
DPC × DPC in $\mathcal{S}_{IM}$
(d)
[$t = t_H$]

FIG. 25. Correspondences of (a), (b) hybrid PCs, (c) merged hybrid PCs, and (d) merged defect PCs in the original system $\mathcal{S}_H$ for a Hadamard gate and merged PCs in $\mathcal{S}_{IM}$, which is called type-1, -2, -3, and -4 merged PCs, respectively. The circles (squares) indicate links (AQs). In (a)–(c), the primalities of the qubits are presented as colors: orange (primal) and blue (dual). In (d), Z-support qubits are marked as purple triangles. Note that, for each correspondence, both PCs have the same support if the $(t_H + 1)$-layer in $\mathcal{S}_{IM}$ is omitted.

the original system for a Hadamard gate. In $\mathcal{S}_I$, a dj-CS $\tilde{S}_{ZX}$ ending at the defects and a pr-CS $\tilde{S}^{(1)}_{XZ}$ surrounding the green defect are defined as usual. We additionally define a stabilizer $\tilde{S}_{XZ} := \tilde{S}^{(1)}_{XZ}\tilde{S}^{(2)}_{XZ}$, where $\tilde{S}^{(2)}_{XZ}$ is a closed dr-CS between the $(t_H - 1)$- and $(t_H + 1)$-layer as shown in Fig. 24.

We also consider another system $\mathcal{S}_{IM}$ which is identical with $\mathcal{S}_I$ except for one difference: For each of hybrid PCs, merged hybrid PCs, and merged defect PCs in $\mathcal{S}_H$ (see Figs. 18 and 23), a pair of PCs in $\mathcal{S}_I$ are merged as shown in Fig. 25 and form *type-1, -2, -3, and -4 merged* PCs, respectively. These merged PCs are used in $\mathcal{S}_{IM}$ instead of original PCs involved in the $(t_H + 1)$-layer.

We now prove that $\mathcal{S}_H$ does not allow local nontrivial undetectable error sets (LNUEs) by showing the following three statements:

(1) For each error set e in $\mathcal{S}_H$, there exists an error set in $\mathcal{S}_{IM}$ which has the same properties (i.e., whether it is local, trivial, or detectable) as e and does not act on any qubit in the $(t_H + 1)$-layer. Here we say that an error set in $\mathcal{S}_{IM}$ is nontrivial if it anticommutes with $\tilde{S}_{ZX}$ or $\tilde{S}_{XZ}$.

(2) $\mathcal{S}_I$ does not allow LNUEs.

(3) If there exists an LNUE in $\mathcal{S}_{IM}$ which does not act on any qubit in the (t_H+1)-layer, there also exists an LNUE in $\mathcal{S}_I$.

1. Proof of the first statement

To show the first statement, we should notice that $\mathcal{S}_{IM}$ is just a variation of $\mathcal{S}_H$ where an extra layer is inserted between the t_H- and (t_H+1)-layer. (Note that these two layers contain Y-planes in $\mathcal{S}_H$.) For a qubit q in $\mathcal{S}_H$ at (x,y,t), let $\tilde{q}$ denote a qubit in $\mathcal{S}_{IM}$ at (x,y,t) if $t \leqslant t_H$ and at $(x,y,t+1)$ if otherwise. Note that $\tilde{q}$ cannot be in the (t_H+1)-layer. Then for an error set e in $\mathcal{S}_H$, there is an error set $\tilde{e}$ in $\mathcal{S}_{IM}$ such that $\text{supp}(\tilde{e}) = \{\tilde{q} \mid q \in \text{supp}(e)\}$ holds. Note that we here consider only their supports, not their actual operators.

Furthermore, we can find similar correspondences for PCs and stabilizers for transforming logical operators (S_{XZ} and S_{ZX}) in $\mathcal{S}_H$. [However, this time the supports of their counterparts in $\mathcal{S}_{IM}$ may contain qubits in the (t_H+1)-layer.] Comparing Figs. 13 and 24, $\text{supp}(\tilde{S}_{ZX}) = \{\tilde{q} \mid q \in \text{supp}(S_{ZX})\}$ can be checked. Similarly, $\text{supp}(\tilde{S}_{XZ})$ contains $\tilde{q}$ for each qubit $q \in \text{supp}(S_{XZ})$, but this time it additionally contains some qubits in the (t_H+1)-layer. For a PC S in $\mathcal{S}_H$, it is straightforward to obtain a unique PC $\tilde{S}$ in $\mathcal{S}_{IM}$ such that $\text{supp}(\tilde{S}) = \{\tilde{q} \mid q \in \text{supp}(S)\}$ holds, if S is an ordinary PC or a defect PC. Otherwise, S is a hybrid or merged PC involved in Y-plane qubits (see Figs. 18 and 23). In such a case, $\tilde{S}$ is set to a merged PC in $\mathcal{S}_{IM}$ shown in Fig. 25. Likewise, $\text{supp}(\tilde{S})$ is composed of $\tilde{q}$ for each qubit $q \in \text{supp}(S)$ and some additional qubits in the (t_H+1)-layer. Note that this correspondence for PCs is bijective since we remove original PCs involved in the (t_H+1)-layer which do not have their counterparts in $\mathcal{S}_H$.

We can now notice that e and $\tilde{e}$ have the same properties. Since they have the same size, e is local if and only if $\tilde{e}$ is local. If e is trivial, e and each of S_{XZ} and S_{ZX} share an even number of qubits; thus so do $\tilde{e}$ and each of $\tilde{S}_{XZ}$ and $\tilde{S}_{ZX}$, which means that $\tilde{e}$ is trivial. [It does not matter that the support of $\tilde{S}_{XZ}$ contains additional qubits in the (t_H+1)-layer, since it is guaranteed that $\tilde{e}$ does not act on those qubits.] It is straightforward to see that PCs flipped by e are $S_1, S_2, \ldots$ if and only if PCs flipped by $\tilde{e}$ are $\tilde{S}_1, \tilde{S}_2, \ldots$. Hence, e is detectable if and only if $\tilde{e}$ is detectable. Finally, $\tilde{e}$ does not act on any qubit in the (t_H+1)-layer by definition.

2. Proof of the second statement

Noticing that $\mathcal{S}_I$ is just the simple extensions of defects, we can show that $\mathcal{S}_I$ does not allow local undetectable error sets anticommuting with $\tilde{S}_{ZX}$, $\tilde{S}_{XZ}^{(1)}$, $\tilde{S}_{XZ}^{(2)}$, or $\tilde{S}_{XZ} = \tilde{S}_{XZ}^{(1)}\tilde{S}_{XZ}^{(2)}$. $\tilde{S}_{ZX}$ and $\tilde{S}_{XZ}^{(1)}$ are CSs used for a logical identity gate; thus we already know that this statement is true for them. Since $\tilde{S}_{XZ}^{(2)}$ is a dual CS, only dual error chains passing through it an odd number of times can anticommute with it. However, since $\tilde{S}_{XZ}^{(2)}$ is closed and dual error chains cannot end at primal defects, there are no undetectable error sets anticommuting with $\tilde{S}_{XZ}^{(2)}$. The statement for $\tilde{S}_{XZ}$ then automatically holds.

3. Proof of the third statement

Let us assume that there exists an LNUE e in $\mathcal{S}_{IM}$ which does not act on any qubit in the (t_H+1)-layer. Note that $\mathcal{S}_I$ has some additional PCs compared to $\mathcal{S}_{IM}$. Therefore, e may be a local nontrivial *detectable* error set in $\mathcal{S}_I$. In detail, if two PCs S_1 and S_2 are merged in $\mathcal{S}_{IM}$, e may flip both S_1 and S_2 in $\mathcal{S}_I$. For example, a type-3 merged PC in Fig. 25(c) can be written as $S_\mathsf{d}S'_\mathsf{d}$ for two dc-PCs S_d and S'_d, and thus e may flip both of them. The same argument holds for defect PCs regarding type-4 merged PCs in Fig. 25(d). However, considering type-1 and -2 merged PCs in Figs. 25(a) and 25(b), a pc-PC is involved in two merged PCs with two dc-PCs. Hence, e either commutes or anticommutes with all these three PCs.

Suppose that e flips n_p primal PCs, n_d dual PCs, and n_D defect PCs. In other words, e anticommutes with a primal PC $S_{i\mathsf{p}}$ and two dual PCs $S_{i\mathsf{d}}$, $S'_{i\mathsf{d}}$ $(i=1,\ldots,n_\mathsf{p})$ of the same color for a pair of type-1 and -2 merged PCs, with dual PCs $S_{i\mathsf{d}}$ and $S'_{i\mathsf{d}}$ $(i=n_\mathsf{p}+1,\ldots,n_\mathsf{d}/2)$ of the same color for a type-3 merged PC, and with defect PCs S_{iD} and S'_{iD} $(i=1,\ldots,n_D)$ for a type-4 merged PC. Here $S_{i\mathsf{d}}$ and $S'_{i\mathsf{d}}$ are set to act on qubits of $t \leqslant t_H+1$ and $t \geqslant t_H+1$, respectively. We define P_p, P_d, P'_d, and P_D by the sets of $S_{i\mathsf{p}}$'s, $S_{i\mathsf{d}}$'s, $S'_{i\mathsf{d}}$'s, and S_{iD}'s, respectively.

Let $e_D^{(i)}$ denote the length-1 defect error chain consisting of a qubit shared by S_{iD} and S'_{iD} for each i. $e_D^{(i)}$ is trivial since the corresponding qubit is a defect CQ which neither $\tilde{S}_{ZX}$ nor $\tilde{S}_{XZ}$ contains in its support. ($\tilde{S}_{ZX}$ contains defect qubits, but they are AQs as shown in Fig. 8(c).) Therefore, $e_1 := e\prod_i e_D^{(i)}$ is local, nontrivial, and detected by PCs in $P_\mathsf{p} \cup P_\mathsf{d} \cup P'_\mathsf{d}$.

Let us write $e_1 := e_\mathsf{p}e_\mathsf{d}e'_\mathsf{d}e_D$, where e_p is a primal error chain, e_d is a dual error chain in the layers of $t < t_H+1$, e'_d is a dual error chain in the layers of $t > t_H+1$, and e_D is a defect error chain. Note that the dual error chain in e_1 can be divided by two in such a way since e does not contain qubits in the (t_H+1)-layer. Note that primal PCs can be flipped only by e_p. In order for e_p to be a proper error chain,

$$|P_\mathsf{pr}| \equiv |P_\mathsf{pg}| \equiv |P_\mathsf{pb}| \equiv x \pmod 2 \tag{C1}$$

should hold for $x \in \{0,1\}$, where P_pc for each color c is the set of pc-PCs in P_p, because of the fusion rule: A unit error chain (pr-PC, pg-PC, pb-PC, or pj-PC) always flips either two PCs of the same color or three PCs of different colors. Similarly, dual PCs in P_d can be flipped only by e_d; thus

$$|P_\mathsf{dr}| \equiv |P_\mathsf{dg}| \equiv |P_\mathsf{db}| \equiv y \pmod 2 \tag{C2}$$

holds for $y \in \{0,1\}$, where P_dc for each color c is the set of dc-PCs in P_d.

Now, let $e_\mathsf{d}^{(i)}$ denote the length-1 dual error chain consisting of the qubit shared by $S_{i\mathsf{d}}$ and $S'_{i\mathsf{d}}$. Since $e_\mathsf{d}^{(i)}$ flips only these two PCs, $e_2 := e_1\prod_i e_\mathsf{d}^{(i)} = e_\mathsf{p}e_{\mathsf{d}2}e_D$ flips only primal PCs in P_p, where $e_{\mathsf{d}2} := e_\mathsf{d}\prod_i e_\mathsf{d}^{(i)}$ is a new dual error chain. Since the primal PCs can be flipped only by e_p, $e_{\mathsf{d}2}e_D$ is local and undetectable, and thus it is trivial according to the second statement. Note that the qubit in $e_\mathsf{d}^{(i)}$ is a dcAQ in the (t_H+1)-layer if $S_{i\mathsf{d}}$ and $S'_{i\mathsf{d}}$ have the color of c. Thus, $e_\mathsf{d}^{(i)}$ (for every i), $e_{\mathsf{d}2}e_D$, and e_p all commute with $\tilde{S}_{ZX}$ which acts only on vacuum dCQs and defect pAQs. Therefore, $e_1 = e_\mathsf{p}e_\mathsf{d}e_D = e_\mathsf{p}e_{\mathsf{d}2}e_D\prod_i e_\mathsf{d}^{(i)}$ also commutes with $\tilde{S}_{ZX}$. We however know that e_1 is nontrivial, and thus it should anticommute with $\tilde{S}_{XZ}$. Moreover, $e_\mathsf{d}^{(i)}$ anticommutes with $\tilde{S}_{XZ}$ if and only if it is either green or blue and located inside the area enclosed by $\tilde{S}_{XZ}$. Therefore, since e_1 anticommutes with $\tilde{S}_{XZ}$,

$$e_2, \tilde{S}_{XZ} = 0 \iff |P_{\mathsf{dg},I}| + |P_{\mathsf{db},I}| \equiv 0 \pmod 2 \tag{C3}$$

holds, where $P_{\mathsf{p(d)c},I}$ is the set of PCs in $P_{\mathsf{p(d)c}}$ to which the qubits corresponding are located inside the area enclosed by $\tilde{S}_{XZ}$.

We can get another equation regarding the locality condition. Note that merged c-colored hybrid PCs in $\mathcal{S}_H$ are placed inside the c-colored defects (see Fig. 23); thus so are type-3 c-colored merged PCs in $\mathcal{S}_{IM}$. Since e_1 is local and anticommutes with $\tilde{S}_{XZ}$, it can be assumed that its support is near the green defect, which indicates that neither dr-PCs nor db-PCs in $\mathcal{S}_I$ corresponding to type-3 merged PCs in $\mathcal{S}_{IM}$ are not flipped by e_1. Therefore, $S_{i\mathsf{d}}$ and $S'_{i\mathsf{d}}$ for every $i \in \{n_\mathsf{p}+1, \dots, n_\mathsf{d}/2\}$ are green, and thus

$$\begin{cases} |P_{\mathsf{pr}}| = |P_{\mathsf{dr}}|, & |P_{\mathsf{pb}}| = |P_{\mathsf{db}}|, \\ |P_{\mathsf{pr},I}| = |P_{\mathsf{dr},I}|, & |P_{\mathsf{pb},I}| = |P_{\mathsf{db},I}| \end{cases} \tag{C4}$$

hold. As a consequence, we get

$$x \equiv |P_{\mathsf{pr}}| = |P_{\mathsf{dr}}| \equiv y \pmod 2, \tag{C5}$$

considering Eq. (C1) and Eq. (C2). For green PCs, we can say only that

$$|P_{\mathsf{pg}}| - |P_{\mathsf{pg},I}| = |P_{\mathsf{dg}}| - |P_{\mathsf{dg},I}|, \tag{C6}$$

which is about PCs outside the area enclosed by $\tilde{S}_{XZ}$.

Last, we try to make e_2 undetectable by multiplying appropriate primal error chains. Note that e_2 flips only primal PCs. First, let us consider moving all flipped PCs in $P_{\mathsf{pc},I}$ outside the area enclosed by $\tilde{S}_{XZ}$. If a moved PC is green or blue, the corresponding multiplied error chain anticommutes with $\tilde{S}_{XZ}$. After that, since every flipped PC is outside the area, they can move properly and completely annihilate with each other without touching $\tilde{S}_{XZ}$, then a local undetectable error chain e_3 is finally obtained. To see whether e_3 is nontrivial or not, we use

$$\begin{aligned} &e_3, \tilde{S}_{XZ} = 0 \\ &\iff |P_{\mathsf{dg},I}| + |P_{\mathsf{db},I}| + |P_{\mathsf{pg},I}| + |P_{\mathsf{pb},I}| \equiv 0 \pmod 2, \end{aligned} \tag{C7}$$

which is obtained by considering the above discussion and the proposition in Eq. (C3). We get

$$\begin{aligned} |P_{\mathsf{dg},I}| + |P_{\mathsf{db},I}| + |P_{\mathsf{pg},I}| + |P_{\mathsf{pb},I}| &\equiv |P_{\mathsf{dg},I}| + |P_{\mathsf{pg},I}| \\ &\equiv |P_{\mathsf{pg}}| + |P_{\mathsf{dg}}| \\ &\equiv x + y \equiv 0 \pmod 2, \end{aligned}$$

where the first equivalence comes from Eq. (C4), the second one comes from Eq. (C6), the third one comes from Eqs. (C1) and (C2), and the last one comes from Eq. (C5). Therefore, e_3 is indeed nontrivial. In summary, if there exists an LNUE e in $\mathcal{S}_{IM}$ which does not act on any qubit in the (t_H+1)-layer, there also exists an LNUE e_3 in $\mathcal{S}_I$, which contradicts to the second statement that $\mathcal{S}_I$ does not allow LNUEs.

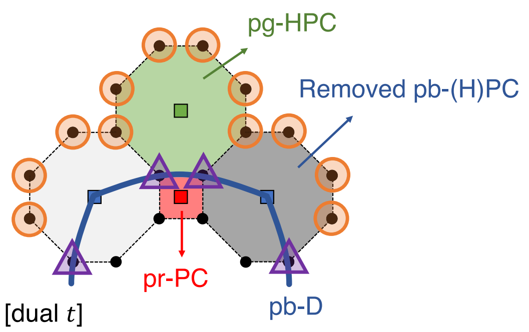


FIG. 26. Error correction when the vacuum and a primal Y-plane on a dual layer are separated by a pb-D. Defect (Y-plane) qubits in the layer are marked as purple triangles (orange circles). pr-HPC, pg-HPC, pr-PC, and pg-PCs acting on defect qubits are incompatible, but they can be merged with each other appropriately to form compatible stabilizers. However, pb-PCs and pb-HPCs overlapping with the defect cannot be merged in such a way; thus they are just removed.

APPENDIX D: ERROR CORRECTION IN A LOGICAL PHASE GATE

Here we investigate error correction in a logical phase gate. As proposed in Sec. IV E 4, a primal Y-plane is placed in the middle of the timelike defects of a primal logical qubit. X_L is transformed into Y'_L via $S_X := S_X^{(1)} S_X^{(2)}$ and Z_L is transformed into Z'_L via S_Z, where $S_X^{(1)}$ and a primal CS while $S_X^{(2)}$ and S_Z are dual CSs. It is important that S_X has X and Y operators in the t_2-layer on which the Y-plane is placed, as shown in Fig. 14(c). Unlike the case of the Hadamard gate (see Appendix C), the Y-plane cannot be made to cover a wide enough area, since supp(S_X) has X operators on qubits just near the defects. However, since neither hybrid PCs nor ordinary PCs are not compatible along the interface between the Y-plane and vacuum, there may be short nontrivial undetectable error chains near the interface.

On the other hand, if the Y-plane and the vacuum are separated by defect qubits, compatible PCs can be appropriately defined along the interface. To see this, let us suppose that the Y-plane and the vacuum are separated by the pc-D, as shown in Fig. 26 for c = b where the orange circles (purple triangles) indicate Y-plane (defect) qubits in the layer. Although pc′-PCs and pc′-HPCs (c′ ≠ c) acting on defect qubits are incompatible, they can be merged with each other appropriately to form larger compatible stabilizers. It is worth noticing that such "hybrid merged PCs" can be analogously used instead of original merged PCs near defects in Fig. 17. Note that, unlike original merged stabilizers, hybrid merged PCs have Z operators on several defect qubits. Additionally, defect PCs are not affected by the Y-plane, since they do not overlap with it.

In summary, each PC near the defect either remains the same or is just replaced to its "hybrid version" when the Y-plane is placed. This contrasts with the fact that PCs along the interface between the Y-plane and vacuum cannot be compatible. Therefore, we need to extend the defects in a spacelike manner to surround the Y-plane entirely as visualized in Fig. 14(c), so that the Y-plane contacts only with the defects. The shape of S_X and the paths of defects should be

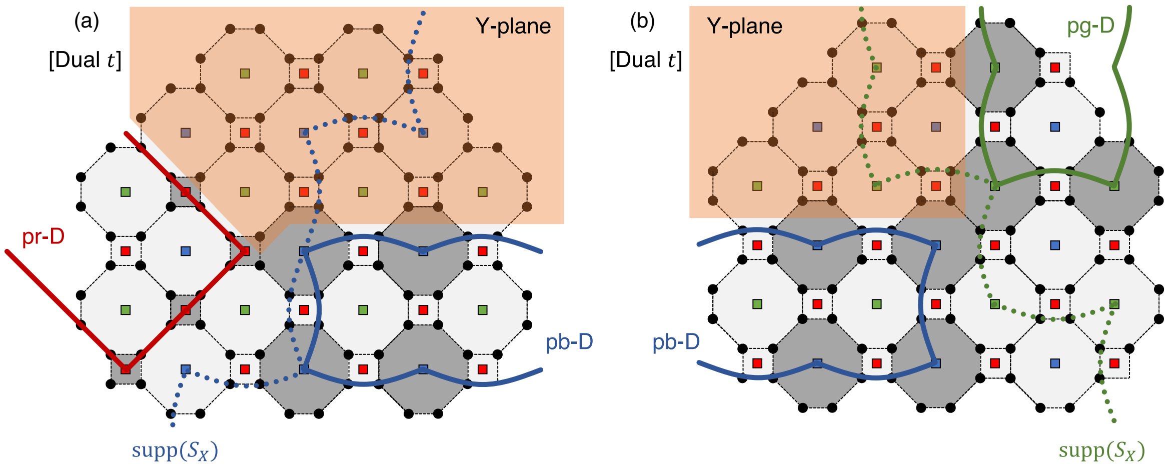


FIG. 27. Microscopic structures near where (a) red and blue defects or (a) blue and green defects are closest for a logical phase gate. The colored solid (dotted) lines indicate the cross sections of the defects (support of S_X) on the layer; CQs along the lines belong to the defects (support). Gray areas indicate removed PCs due to the defects.

carefully chosen for the defects not to overlap with supp(S_X). In particular, the microscopic structures near where two defects are closest are important. As visualized in Fig. 27, the Y-plane should not contact directly with ordinary PCs for the vacuum; they always meet separated by a defect.

We now verify that local nontrivial undetectable error sets do not exist in the above configuration. We first show that it is enough to consider only primal error chains. It is observed that each hybrid merged PC contains various types of qubits: primal vacuum, dual vacuum, defect, and Y-plane qubits. Therefore, primal error chains may end at the Y-plane and connect with defect or dual error chains without being detected. However, since there always exists a dual error chain equivalent to each defect error chain (see Appendix B), we need to consider only primal and dual error chains.

Let us consider a local undetectable error chain $e := e_\mathrm{p} e_\mathrm{d}$ where e_p (e_d) is a primal (dual) error chain. Since there are neither flipped dual PCs nor dual defects, e_d should be closed; thus there exists a stabilizer S such that $\mathrm{supp}_Z(S) = e_\mathrm{d}$. [If e_d is a closed dc(j)-EC, we can find a pc(j)-CS S whose boundary is supp(e_d). e_d generally can be written as the product of multiple closed error chains; thus S is also the product of the corresponding CSs.] Since $\mathrm{supp}_X(S)$ is composed of primal qubits, we get $e_\mathrm{d} \sim e_\mathrm{d} S = X(\mathrm{supp}_X(S)) \sim X(D_S)X(Y_S)$, where the symbol "$\sim$" means the equivalence relation and D_S (Y_S) is defect (Y-plane) qubits in $\mathrm{supp}_X(S)$. Here D_S can be assumed to be empty, because e_d neither goes around a defect (since e is local) nor penetrates it (since e should not be detected by defect PCs). Therefore, e_d is equivalent to a primal error chain in the Y-plane, which means that e is equivalent to a primal error chain.

We find that local undetectable primal error chains are trivial. Each of such error chains behaves as if the primal Y-plane does not exist since each hybrid PC or hybrid merged PC and the corresponding original one contain exactly the same primal qubits in their supports. Note that a pc-EC e can end at the pc′-D d only if c = c′ or the surface where e meets d is spacelike (see Appendix B). Furthermore, since dual CSs S_Z and $S_X^{(2)}$ always commutes with primal error chains, we need to consider only $S_X^{(1)}$, whose support in each dual layer is shaped as the dotted lines shown in Fig. 28. Therefore, two types of nontrivial undetectable primal error chains are possible as shown in Fig. 28: error chains ending at three or two defects. For an error chain of the second type, at least one of the surfaces where it meets the defects should be spacelike. We can check that both of them are nonlocal. (It may seem unclear that an error chain of the second type is also nonlocal. However, since it should pass by the timelike part of a defect to reach its spacelike surface, its size depends on the circumference of the defect; thus it is nonlocal.)

One may wonder whether the above analysis on error chains works well even for such extreme cases that two defects are very close. We can approach the problem in a different way. Let us consider removing all primal PCs near where red and blue defects are closest, except red and green PCs (including their merged ones) in a region close to the blue

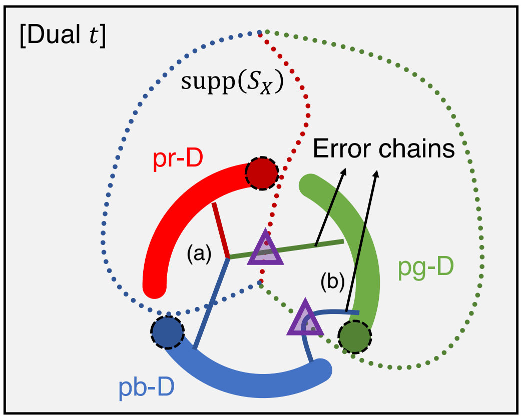


FIG. 28. Nontrivial undetectable primal error chains regarding a logical phase gate. The colored circles indicate the timelike parts of the defects and the thick colored lines indicate their spacelike parts. supp(S_X) is presented as colored dotted lines. Each of such error chains can either (a) end at the three defects or (b) end at two defects, as shown in colored solid lines. In the case of (b), at least one of the surfaces where it meets the defects should be spacelike. The intersection points of the error chains and supp(S_X) are marked as triangles.

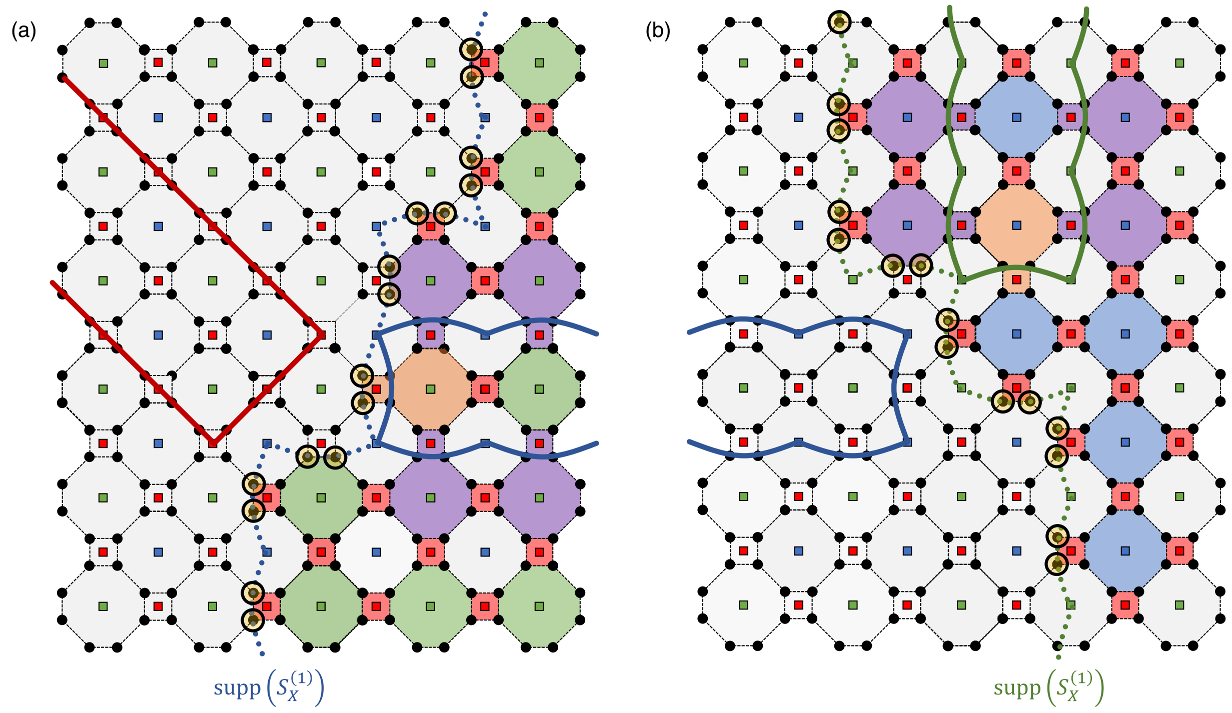


FIG. 29. Removal of some primal PCs near where (a) red and blue defects or (b) green and blue defects are closest to verify that local nontrivial undetectable primal error chains in the area do not exist. Each red, green, or blue area indicates a survived pr-PC, pg-PC, or pb-PC, respectively. Each purple or orange area indicates a survived merged primal PC. Each qubit marked by a black circle is a terminable qubit which belongs to the support of one PC only.

defect with respect to supp$(S_X^{(1)})$, as shown in Fig. 29(a). We can observe that each vacuum qubit in this area either belongs to the supports of one or two PCs or does not belong to the support of any PC at all. In particular, each vacuum qubit belonging to the support of one PC is contained in supp(S_X) and called a "terminable qubit." The region near where green and blue defects are closest can be considered analogously as shown in Fig. 29(b).

We now show that, if some PCs are removed as suggested above, any undetectable primal error chain in this area can be decomposed of multiple undetectable primal error chains by following steps. For simplicity, we regard an error set and a PC as its support; i.e., we omit the corresponding operators. Let e be an undetectable error set.

(1) For each qubit $q_i^{\mathrm{sng}} \in e$ $(i = 1, 2, \dots)$ which does not belong to the support of any PC, a single-qubit error on q_i^{sng} is undetectable by itself. Define $e' := e \setminus \{q^{\mathrm{sng}\}_1, q_2^{\mathrm{sng}}, \cdots}$.

(2) Pick a terminable qubit q_0 from e'. If there are no such qubits, skip this and the following step.

(a) Let S_0 be the unique PC flipped by an error on q_0. Pick a qubit q_1 from $e'_0 \cap S_0$ where $e'_0 := e' \setminus \{q_0\}$ is an error set. This is possible since e'_0 flips S_0 only.

(b) If q_1 is terminable, we get an undetectable error set $e_1^{\mathrm{end}} := \{q_0, q_1\}$.

(c) If otherwise,

(i) Let S_1 be the unique PC flipped by an error on q_1 such that $S_1 \neq S_0$. Pick a qubit q_2 from $e'_1 \cap S_1$ where $e'_1 := e'_0 \setminus \{q_1\}$ is an error set. This is possible since e'_1 flips S_1 only.

(ii) If q_2 is terminable, we get an undetectable error set $e_1^{\mathrm{end}} := \{q_0, q_1, q_2\}$.

(iii) If otherwise, repeat step 2(c) analogously for $q_2, q_3, \dots$ until we reach a terminable qubit and get an undetectable error set.

(3) For each $i \geqslant 2$, repeat step 2 for $e' \setminus (e_1^{\mathrm{end}} \cup \cdots \cup e_{i-1}^{\mathrm{end}})$ instead of e' to get an undetectable error set e_i^{end} until there are no terminable qubits in $e' \setminus (e_1^{\mathrm{end}} \cup \cdots \cup e_i^{\mathrm{end}})$.

(4) Pick a qubit $q_0 \in e'' := e' \setminus (e_1^{\mathrm{end}} \cup e_2^{\mathrm{end}} \cup \cdots)$. If there are no such qubits, skip this and the following step.

(a) Let S_{-1} and S_0 be PCs flipped by q_0. Pick a qubit $q_1 \in e''_0 \cap S_0$ where $e''_0 := e'' \setminus \{q_0\}$. This is possible since e''_0 flips S_{-1} and S_0. Let S_1 be the unique PC flipped by an error on q_1 such that $S_1 \neq S_0$.

(b) Through the same method as step 2, obtain qubits $q_2, \dots, q_i$ and the corresponding PCs $S_2, \dots, S_i$ where $S_i = S_{-1}$. The only difference from step 2 is that e''_j for $j < i$ flips not only S_j but also S_{-1}. Such a qubit q_i always can be reached since the number of PCs is limited.

(c) $e_1^{\mathrm{cyc}} := \{q_0, \dots, q_i\}$ is then an undetectable error set.

(5) For each $i \geqslant 2$, repeat step 4 for $e'' \setminus (e_1^{\mathrm{cyc}} \cup \cdots \cup e_{i-1}^{\mathrm{cyc}})$ instead of e'' to get an undetectable error set e_i^{cyc} until $e'' = e_1^{\mathrm{cyc}} \cup \cdots \cup e_i^{\mathrm{cyc}}$ holds.

Through the above process, we can decompose an undetectable primal error chain e into multiple mutually disjoint undetectable primal error chains: $\{q_1^{\mathrm{sng}}\}, \{q_2^{\mathrm{sng}}\}, \dots, e_1^{\mathrm{end}}, e_2^{\mathrm{end}}, \dots, e_1^{\mathrm{cyc}}, e_2^{\mathrm{cyc}}, \dots$. For each i, e_i^{end} starts from a terminable qubit and end at another, while $\{q_i^{\mathrm{sng}}\}$ and e_i^{cyc} contain only nonterminable qubits. Since all qubits in supp(S_X) in the area are terminable, e_i^{cyc} for each i meets supp(S_X) twice, while $\{q_i^{\mathrm{sng}}\}$ and e_i^{cyc} for each i does not meet it at all. Therefore, e commutes with S_X; thus it is trivial. In other words, every local undetectable primal error chain near where two defects are closest is trivial if some PCs are removed. This statement also holds for the original setting where PCs are not removed since a local nontrivial undetectable error chain retains these properties even if some PCs are removed.

APPENDIX E: RESOURCE ANALYSIS WITHOUT CONSIDERING NONTRIVIAL LOGICAL GATES

Here we analyze the resource overheads of logical qubits in MBQC via RTCSs or CCCSs without considering nontrivial logical gates; namely, we calculate n/k and N_{CZ}/k in terms of the code distance d, where k is the number of logical qubits and n (N_{CZ}) is the number of required physical qubits (CZ gates) per layer. The results on the calculation are presented in Table III and Sec. VII. We consider two schemes for RTCS computation: defect-based and patch-based ones. In the defect-based scheme [38,41–43], logical qubits are encoded in pairs of defects and logical operations are done by defect braiding or state distillation. In the patch-based schemes [44], logical qubits are encoded in square "patches" separated from each other and fault-tolerant logical operations are done by lattice surgery or state distillation. For CCCS computation, we consider two types of lattices: 4-8-8 and 6-6-6.

To make the code distance equal to d, we should find arrangements of defects or patches where all the possible nontrivial undetectable error chains contain d or more qubits. Throughout this Appendix, only the leading-order term on d is calculated for each result.

We first define the coordinate systems for the analysis. The x and y axes are presented in Fig. 1(b) for RTCSs and Fig. 2 for the two types of CCCSs. The unit length is the length of a side of a unit cell for RTCSs, the distance between adjacent prAQ and pgAQ for 4-8-8 CCCSs, and half the distance between two adjacent AQs with the same color for 6-6-6 CCCSs. A unit area contains three qubits and six CZ gates for RTCSs, three qubits and eight CZ gates for 4-8-8 CCCSs, and $3\sqrt{3}/2$ qubits and $4\sqrt{3}$ CZ gates for 6-6-6 CCCSs. Note that, when counting CZ gates, we regard that each CZ gate connecting different layers belongs to these layers divided in half.

The analysis for the patch-based RTCS scheme is straightforward. Each patch is a square with side length d and the gaps between patches are sufficient to be $O(1)$. We therefore get $n/k \approx 3d^2$ and $N_{\mathrm{CZ}}/k \approx 6d^2$.

For the other three schemes, we consider hexagonal arrangements of parallel timelike primal defects, where every error chain connecting different defects or surrounding a defect has d or more qubits. We need to find the optimal distances between defects minimizing n/k.

The optimal arrangement for the defect-based RTCS scheme is shown in Fig. 30(a) where each black square indicates a primal defect and the purple area indicates a region occupied by a logical qubit. It is straightforward to obtain the distances, considering that the shortest error chain connecting $(0,0)$ and (x,y) contains $|x|+|y|+O(1)$ qubits. The area occupied by a logical qubit is thus about $\frac{35}{16}d^2 \approx 2.19d^2$, and since a unit area contains three qubits and six CZ gates, we get $n/k \approx 6.56d^2$ and $N_{\mathrm{CZ}}/k \approx 13.1d^2$.

It is more tricky to obtain the optimal arrangements in 4-8-8 or 6-6-6 CCCSs. Figure 30(b) shows the concerned

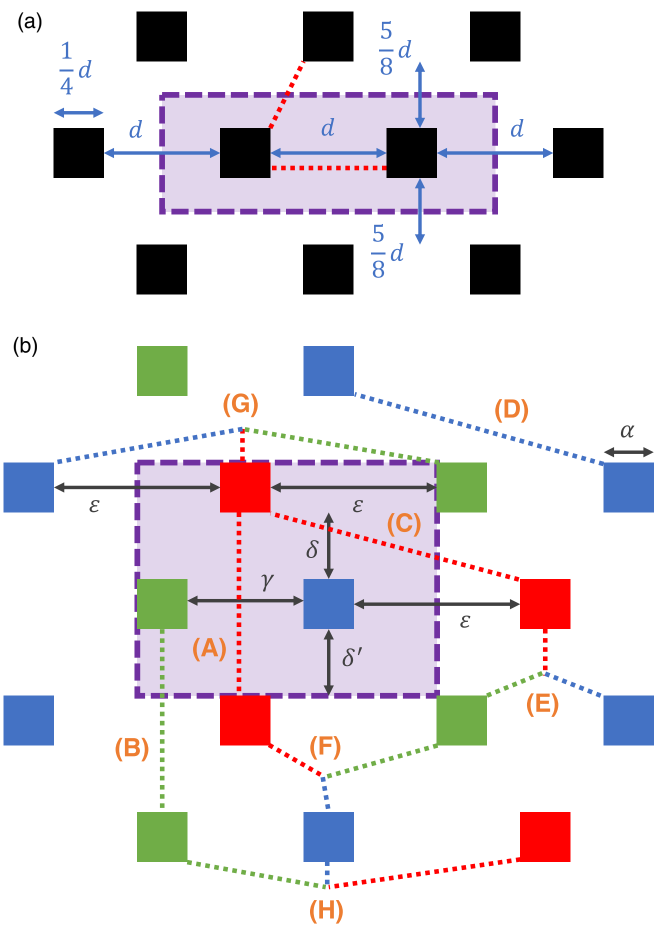


FIG. 30. Arrangement of timelike primal defects for calculating the resource overheads of MBQC via (a) RTCSs or (b) CCCSs. Their projections on a plane perpendicular to the time axis are schematized. Each black, red, green, or blue square is a defect, where its color means the color of the defect in CCCS computation. Each purple rectangle surrounded by dashed lines is an area occupied by a logical qubit. Dotted lines indicate all the possible types of error chains which may be the shortest ones, which are used for obtaining the values of the marked spaces minimizing the area of a logical qubit. Note that, in (b), counterparts of some error chains regarding the exchange of blue and green defects are omitted, since the two lattices (4-8-8 and 6-6-6) which we concern have symmetry on those defects. The optimal spaces for RTCSs are directly presented in (a). For CCCSs, they are $(\alpha,\gamma,\delta,\delta',\epsilon) = (\frac{1}{2}d, 0, \frac{1}{2}d, \frac{1}{2}d, \frac{1}{2}d)$ for 4-8-8 and $(\alpha,\gamma,\delta,\delta',\epsilon) \approx (0.464d, 0.268d, 0.634d, 0.634d, 0.269d)$ for 6-6-6. Here the unit length is a side of a unit cell in RTCSs [see Fig. 1(b)], the distance between adjacent prAQ and pgAQ in 4-8-8 CCCSs [see Fig. 4(a)], and half the distance between two adjacent prAQs in 6-6-6 CCCSs [see Fig. 2(b)].

hexagonal arrangement with five parameters $(\alpha,\gamma,\delta,\delta',\epsilon)$ considering the symmetry, where each colored square indicates a defect of the color.

We first consider 4-8-8 CCCSs. The shortest pc-EC connecting $(0,0)$ and (x,y) contains

$$l_{\mathrm{c}}(x,y) := \begin{cases} 2\max(x,y)+O(1) & \text{if } \mathrm{c}=\mathrm{r}, \\ |x|+|y|+O(1) & \text{otherwise} \end{cases} \tag{E1}$$

qubits. Also, the shortest defect error chain in a pc-D connecting (0, 0) and (x, y) contains $l_c(x, y)/2$ qubits if the error chain is in a spacelike surface of the defect. If otherwise, it contains $l_c(x, y)$ qubits (see Fig. 22). The width α of each defect can be derived from the shortest defect error chain surrounding it: $\alpha = \frac{1}{2}d$. (It may be more optimal that defects have different widths for different colors. We however constrain the widths to be equal for ease of calculation.) The following eight inequalities are derived from the eight possible types (A)–(H) of error chain in Fig. 30(b):

$$\begin{cases} \text{(A)} & 2(\delta + \delta' + \alpha) \geqslant d, \\ \text{(B)} & \delta + \delta' + \alpha \geqslant d, \\ \text{(C)} & 2\max\left[\frac{\gamma+\alpha}{2} + \epsilon, \min(\delta, \delta')\right] \geqslant d, \\ \text{(D)} & \frac{\alpha+\gamma}{2} + \epsilon + \min(\delta, \delta') \geqslant d, \\ \text{(E)} & \gamma + 2\min(\delta, \delta') \geqslant d, \\ \text{(F)} & \min(\delta, \delta') + \epsilon + \frac{1}{2}\max(\gamma - \alpha, 0) \geqslant d, \\ \text{(G)} & \alpha + 2\epsilon \geqslant d, \\ \text{(H)} & \alpha + \gamma + \epsilon \geqslant d. \end{cases} \tag{E2}$$

Note that, to get the inequalities corresponding to (E)–(H), the points at which three error chains meet should be placed carefully. It is straightforward to see that placing each point just next to the red defect minimizes the length of the error chain. The area S occupied by a logical qubit is written as

$$S \approx \left(\frac{3}{2}\alpha + \frac{\gamma}{2} + \epsilon\right)(\delta + \delta' + 2\alpha). \tag{E3}$$

Minimizing S subject to the above inequalities, we get $S \approx 2.5d^2$ where the corresponding spaces are $(\alpha, \gamma, \delta, \delta', \epsilon) = (\frac{1}{2}, 0, \frac{1}{2}, \frac{1}{2}, \frac{1}{2})d$. We thus obtain $n/k \approx 7.5d^2$ and $N_{CZ}/k \approx 20d^2$.

The optimal arrangement for 6-6-6 CCCSs also can be derived similarly. The shortest error chain connecting (0,0) and (x, y) for $x, y \geqslant 0$ contains

$$l(x, y) := \max\left(x + \frac{1}{\sqrt{3}}y, \frac{2}{\sqrt{3}}y\right) + O(1) \tag{E4}$$

qubits. The length of the corresponding shortest defect error chain is half of it if the error chain is in a timelike surface and the same as it if otherwise. We thus get $\alpha = (2\sqrt{3} - 3)d \approx 0.464d$, considering an error chain surrounding a defect. The following inequalities are derived for each type of error chain:

$$\begin{cases} \text{(A), (B)} & \frac{2}{\sqrt{3}}(\alpha + \delta + \delta') \geqslant d, \\ \text{(C), (D)} & \max\left[\frac{\alpha+\gamma}{2} + \epsilon + \frac{1}{\sqrt{3}}\min(\delta, \delta'), \right. \\ & \left. \frac{2}{\sqrt{3}}\min(\delta, \delta')\right] \geqslant d, \\ \text{(E)} & \gamma + \frac{2}{\sqrt{3}}\min(\delta, \delta') \geqslant d, \\ \text{(F)} & \epsilon + \frac{2}{\sqrt{3}}\min(\delta, \delta') \geqslant d, \\ \text{(G)} & \alpha + 2\epsilon \geqslant d, \\ \text{(H)} & \alpha + \epsilon + \gamma \geqslant d. \end{cases} \tag{E5}$$

Minimizing S in Eq. (E3) subject to the inequalities, we get $S \approx 2.41d^2$ where the corresponding spaces are $(\alpha, \gamma, \delta, \delta', \epsilon) \approx (0.464, 0.268, 0.634, 0.634, 0.269)d$. We thus obtain $n/k \approx 6.27d^2$ and $N_{CZ}/k \approx 16.7d^2$.

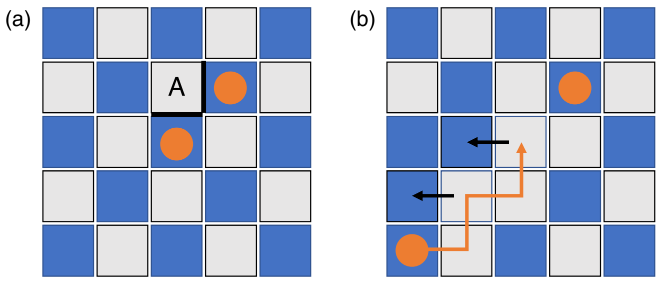


FIG. 31. Checkerboard architecture in patch-based RTCS computation. Blue (gray) squares are patches for logical data (ancilla) qubits. (a) A CNOT gate between two data qubits (orange circles) is done with two "merge and split" operations [44] (black lines) between data qubits and the ancilla qubit A. The ancilla qubit is prepared just before the operation. (b) A CNOT gate between nonadjacent qubits is done by moving a logical qubit appropriately while setting aside qubits in the path.

APPENDIX F: RESOURCE ANALYSIS WHILE CONSIDERING LOGICAL GATES

We here analyze the resource overheads of MBQC via RTCSs or CCCSs, considering nontrivial logical gates. In other words, we investigate the optimal arrangements of defects under the constraint that every elementary logical gate is applicable while keeping the code distance the same. Furthermore, we calculate the total number of physical qubits required for each logical gate. As in Appendix E, we consider two types of RTCS schemes (defect-based and patch-based) and two types of CCCS schemes (4-8-8 and 6-6-6). For the RTCS computation, we need to consider only the logical CNOT gate since the other gates are implemented by state distillation. On the other hand, the logical Hadamard and phase gates also should be considered for CCCS computation. Note that we here do not consider applying gates simultaneously on adjacent logical qubits.

1. Patch-based RTCS computation

A logical CNOT gate in patch-based RTCS computation [44,63] can be done with lattice surgery between logical qubits in diagonally adjacent patches, which requires an ancillary logical qubit adjacent to both of them. Therefore, there should be spaces for such ancillary qubits to be defined. The checkerboard architecture [72] visualized in Fig. 31(a) allows a CNOT gate between an arbitrary pair of qubits (orange circles). The gate can be directly done if the qubits are diagonally adjacent; otherwise, one of them should be moved appropriately while setting aside qubits in the path for a while as shown in Fig. 31(b). Therefore, we get $n/k \approx 6d^2$ and $N_{CZ}/k \approx 12d^2$, which are twice the values obtained without considering the CNOT gate.

A CNOT gate between two adjacent qubits requires $4d$ layers ($2d$ for each "merge and split" operation) to keep the code distance at d since a timelike error chain contains one qubit per two layers. Therefore, at least $48d^3$ physical qubits are required for a CNOT gate. If the two qubits are not adjacent, $4d$ layers are additionally needed to set aside qubits in the path and put them back. Note that, in the original scheme with the surface codes [72], multiple SWAP gates are necessary to

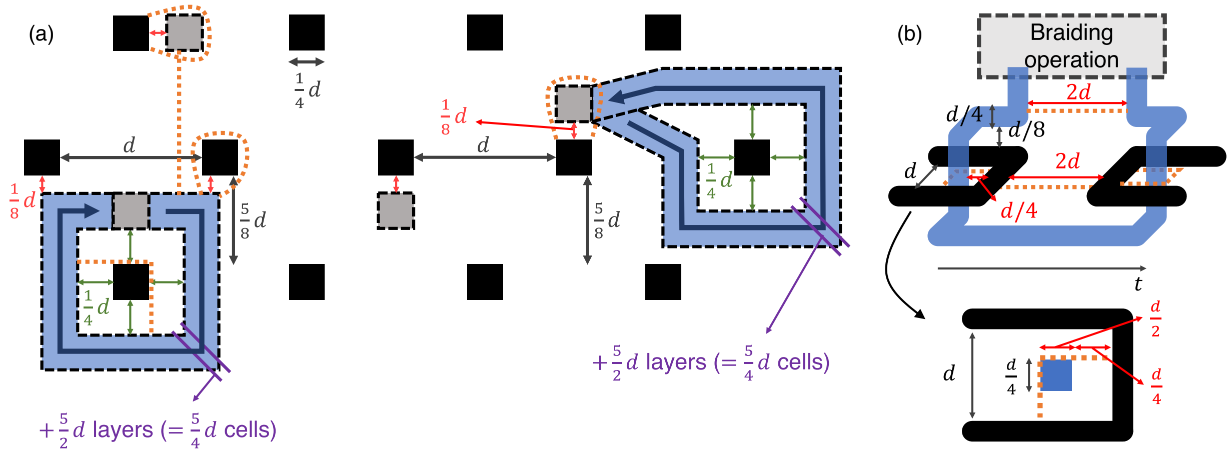


FIG. 32. Arrangement of defects for the logical CNOT gate in defect-based RTCS computation. (a) The control primal logical qubit is first switched to a dual one (gray squares). Then one of the dual defects proceeds to wrap around a defect of the target qubit. During this process, the defect basically proceeds in a spacelike manner, but proceeds by $\frac{5}{2}d$ layers along the positive time axis at a specific location (marked as purple "⫽"). The blue paths indicate two examples of such braiding operations. Primal and dual defects should be more than a certain distance apart, due to the existence of the two types of nontrivial undetectable error chains (orange dotted lines). (b) 3D picture of a CNOT gate. The black and blue lines are primal and dual defects, respectively. The orange dotted lines indicate possible types of error chains, from which the number of layers between defects (highlighted in red) is obtained. Note that a timelike error chain contains one qubit per two layers.

move logical qubits, which is very time-consuming; it is one of the advantages of MBQC that logical qubits can be moved quite flexibly.

2. Defect-based RTCS computation

A logical CNOT gate in defect-based RTCS computation is done by defect braiding [30]; the control logical qubit is first switched to a dual qubit, one of the defects constituting it proceeds to surround a defect of the target qubit (called a *braiding operation*), and finally the control qubit returns to a primal qubit. Figure 32(a) shows examples of such operations. New types of nontrivial undetectable error chains arise from the coexistence of primal and dual defects as shown in Fig. 32(a): the error chains ending at primal defects and surrounding dual defects (or vice versa). These give restrictions that primal and dual defects must be more than a certain distance apart ($d/8$ or $d/4$). Fortunately, the optimal arrangement in Fig. 30(a) is spacious enough to satisfy this condition. Thus, the resource overheads remain the same: $n/k \approx 6.56d^2$ and $N_{CZ}/k \approx 13.1d^2$.

The minimal number of layers required for a CNOT gate is $\frac{9}{2}d$, which can be obtained by considering possible error chains shown in Fig. 32(b). Hence, the number of physical qubits for a CNOT gate is $\sim 59.1d^3$.

We note two things regarding the CNOT gate in defect-based RTCS computation. First, a CNOT gate between any pair of nonadjacent logical qubits is also possible without modifying the arrangement. The entire process discussed above including the number of required layers remains the same, except that one of the dual defects should proceed further in a spacelike manner. Second, multiple CNOT gates with the same control qubit can be done simultaneously by braiding a defect of the control qubit in a way that its path surrounds one of the defects of every target qubit. However, additional layers may be needed during the braiding operation, depending on the shape of the path. These two statements also hold for CCCS computation.

3. CCCS computation

For CCCS computation, we first investigate the optimal arrangement of defects to implement each nontrivial logical gate: the CNOT, Hadamard, or phase gate. Using these results, we obtain an arrangement allowing the implementation of the universal set of gates. Finally, we calculate the number of physical qubits required for each gate.

a. CNOT gate

The analysis for the CNOT gate in CCCS computation is analogous to that in defect-based RTCS computation. Similarly, there exist undetectable nontrivial error chains involved in both primal and dual defects, which may constrain the minimal distances between them. However, if the width α of each defect is equal to or larger than that obtained in Appendix E ($\alpha = \frac{1}{2}d$ for the 4-8-8 lattice and $\alpha = 0.464d$ for the 6-6-6 lattice), such error chains are always longer than the code distance d. We therefore do not need to consider spacelike gaps between primal and dual defects unless they overlap. (If they overlap, defect PCs are no longer compatible.)

Figure 33 shows some examples on CNOT gates in CCCS computation. For a CNOT gate, the control logical qubit is first switched to a dual logical qubit (squares with dashed borders) through a primality-switching gate. It is important to make the spacelike part of one of the dual defects penetrate a primal CS of a different color (see Fig. 12), as shown by purple circles in Fig. 33. Additionally, the dual defects should be sufficiently far away from each other to keep the code distance. As long as these conditions meet, the dual defects can be placed quite freely. After the primality-switching gate, the dg-D or db-D circles around the pr-D of the target logical qubit. In order for these operations to be possible, we need three simple

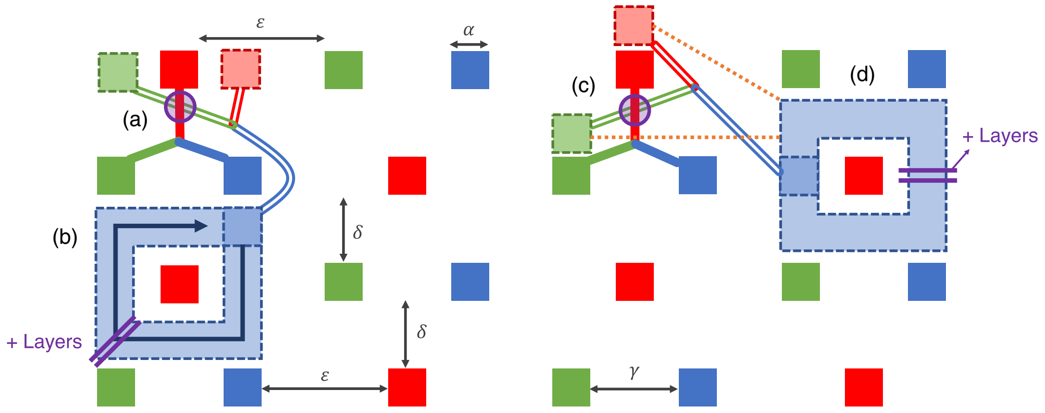


FIG. 33. Arrangement of defects for the logical CNOT gate in CCCS computation. The control primal logical qubits are first switched to dual ones (colored squares with dashed boundaries) as shown in (a) and (c), then the braiding operations are performed (blue arrows) as shown in (b) and (d). The spacelike extensions of primal (dual) defects for primality-switching gates are shown as the colored solid (double) lines. The dual defects penetrate primal CSs at the points marked as the purple circles.

conditions in addition to Eq. (E2) or (E5):

$$\delta \geqslant \alpha, \quad \delta' \geqslant \alpha, \quad \epsilon \geqslant \alpha. \tag{F1}$$

The previous result for the 4-8-8 lattice satisfies these conditions: $(\alpha, \gamma, \delta, \delta', \epsilon) = (\frac{1}{2}, 0, \frac{1}{2}, \frac{1}{2}, \frac{1}{2})d$, $n/k \approx 7.5d^2$, and $N_{\mathrm{CZ}}/k \approx 20d^2$. However, for the 6-6-6 lattice, the parameters should be modified: $\alpha = \gamma = \delta = \delta' = \epsilon \approx 0.464d$, $n/k \approx 6.72d^2$, and $N_{\mathrm{CZ}}/k \approx 17.9d^2$.

We now count the required number of layers for a CNOT gate. Note that a timelike error chain contains one qubit per two layers. The calculation is analogous to that for RTCS computation in Fig. 32(b); denoting the depth (i.e., thickness along the time axis) of each spacelike defect by t_{depth} and the gap between each pair of adjacent primal and dual defects along the time axis by t_{gap} (which are in the units of layers), the total required number of layers is $T_{\mathrm{CNOT}} = 4t_{\mathrm{depth}} + 2t_{\mathrm{gap}} + 2d$. For both types of lattices, the conditions

$$\tfrac{1}{2}t_{\mathrm{depth}} + 2\alpha \geqslant d,$$
$$\tfrac{1}{2}(t_{\mathrm{gap}} + t_{\mathrm{depth}}) + \alpha \geqslant d$$

are sufficient for error chains surrounding each defect (either surrounding it completely or ending at other defects of different primality) are longer than d. Therefore, we get $t_{\mathrm{CNOT}} = 4d$ for the 4-8-8 lattice and $t_{\mathrm{CNOT}} \approx 4.43d$ for the 6-6-6 lattice.

b. Hadamard gate

We next consider the Hadamard gate. Note that every error chain connecting the defects and the boundaries of the Y-planes for the gate should be longer than the code distance d, as discussed in Appendix C. We assume that the Y-planes are square in shape as visualized in Fig. 34, where possible error chains are also presented. For the 4-8-8 lattice, we get

$$\begin{cases} \alpha \geqslant \frac{d}{2}, \quad \delta' \geqslant \frac{3}{2}d, \\ \epsilon - \frac{1}{2}(\alpha + \gamma) \geqslant 2d, \quad \gamma + 2\delta \geqslant d, \end{cases} \tag{F2}$$

and for the 6-6-6 lattice, we get

$$\begin{cases} \alpha \geqslant (2\sqrt{3} - 3)d, \quad \frac{2}{\sqrt{3}}\delta' \geqslant 2d, \\ \epsilon - \frac{1}{2}(\alpha + \gamma) \geqslant 2d, \quad \gamma + \frac{2}{\sqrt{3}}\delta \geqslant d. \end{cases} \tag{F3}$$

Minimizing S subject to the above conditions, for the 4-8-8 lattice, we get $n/k = 24.75d^2$ and $N_{\mathrm{CZ}}/k = 66d^2$ for $(\alpha, \gamma, \delta, \delta', \epsilon) = (\frac{1}{2}, 0, \frac{1}{2}, \frac{3}{2}, \frac{9}{4})d$. Similarly, for the 6-6-6 lattice, we get $n/k \approx 26.8d^2$ and $N_{\mathrm{CZ}}/k \approx 71.5d^2$ for $(\alpha, \gamma, \delta, \delta', \epsilon) \approx (0.464, 0, 0.866, 1.73, 2.23)d$.

Last, a logical Hadamard gate requires only three layers: the $(t_H - 1)$-, t_H-, and $(t_H + 1)$-layer in Fig. 13. Therefore, although the gate demands relatively many physical qubits per layer, the total number of required qubits is rather small.

c. Phase gate

We consider last the logical phase gate. Note that defects are extended in a spacelike manner to surround the Y-plane for a phase gate (see Sec. IV E 4 and Appendix D). We assume that these extensions are done as shown in Fig. 35, where possible nontrivial undetectable error chains are also visualized. We classify the error chains into Type-1, -2, and -3: Type-1 is for those ending at the three defects in the concerning

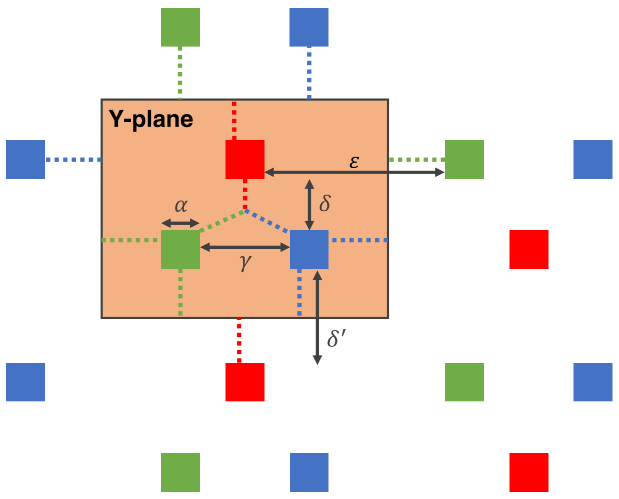


FIG. 34. Arrangement of defects for a logical Hadamard gate. The Y-plane (orange square) covering the logical qubit should be wide enough so that error chains (colored dotted lines) connecting its boundary and the defects are longer than the code distance d.

logical qubit, and Type-2 (Type-3) is for those ending at two defects of different colors in the same logical qubit (different logical qubits). Note that Type-2 and Type-3 error chains are possible since they can end at the spacelike surface of a defect regardless of their colors (see Appendix B). Such error chains may cause difficulties, because some of them have lengths which depend on only the width of each timelike defect as shown in Fig. 35, but the previous values of the width α ($0.5d$ for the 4-8-8 lattice and $0.464d$ for the 6-6-6 lattice) may be not enough for them to be longer than d. Nevertheless, the width α_0 of each spacelike defect does not need to be α; we can set it to $O(1)$ and make it deep enough along the time axis. Another problem is that it is not straightforward to analytically find conditions regarding Type-1 error chains. We thus numerically estimate the condition on γ and δ regarding them. In detail, we randomly sample Type 1 error chains by choosing their end and joint points for a sufficiently large number ($\geqslant 10\,000$) of times, then check whether there are error chains shorter than d.

For the 4-8-8 lattice, we get the following conditions in addition to Eq. (E2):

$$\begin{cases} \text{Type-1 (numerical):} & \gamma \gtrsim 0.4d, \\ & 0.3\gamma + \delta \gtrsim 0.45d \\ \text{Type-2:} & 2\alpha \geqslant d, \\ & \frac{3}{2}\alpha + \frac{1}{2}\gamma \geqslant d, \\ & \alpha + \delta \geqslant d, \\ \text{Type-3:} & \delta' \geqslant d, \\ & \epsilon - \frac{1}{2}(\alpha+\gamma) \geqslant d. \end{cases} \tag{F4}$$

The conditions for Type-1 error chains are valid when $\alpha = \frac{1}{2}d$. By minimizing S in Eq. (E3) subject to the above conditions, we get $n/k = 18.75d^2$ and $N_{CZ}/k = 50d^2$ for $(\alpha, \gamma, \delta, \delta', \epsilon) = (\frac{1}{2}, \frac{1}{2}, \frac{1}{2}, 1, \frac{3}{2})d$.

For the 6-6-6 lattice, we get the conditions:

$$\begin{cases} \text{Type-1 (numerical):} & \gamma \gtrsim 0.5, \quad \delta \gtrsim 0.4 \\ \text{Type-2:} & (1+\frac{2}{\sqrt{3}})\alpha \geqslant d, \\ & \frac{3}{2}\alpha + \frac{1}{2}\gamma \geqslant d, \\ & \frac{2}{\sqrt{3}}(\alpha+\delta) \geqslant d, \\ \text{Type-3:} & \frac{2}{\sqrt{3}}\delta' \geqslant d, \\ & \epsilon - \frac{1}{2}(\alpha+\gamma) \geqslant d. \end{cases} \tag{F5}$$

The conditions on Type-1 error chains are valid when $\alpha = (2\sqrt{3}-3)d$. By minimizing S, we get $n/k \approx 14.5d^2$ and $N_{CZ}/k \approx 38.6d^2$ for $(\alpha, \gamma, \delta, \delta', \epsilon) \approx (0.464, 0.608, 0.402, 0.866, 1.537)d$.

Last, the number of layers T_{phase} required for a phase gate is determined by the widths of the spacelike defects surrounding the Y-plane. Since $\alpha_0 = O(1)$, the depth of each spacelike defect should be at least d layers. Therefore, $T_{\text{phase}} = d$ holds for both types of lattices.

4. Optimal arrangement for general logical gates

Until now, we have investigated the arrangements of defects to implement each of the logical CNOT, phase, and Hadamard gates. We next find the optimal arrangements where all the logical gates are applicable. For the 4-8-8 lattice, considering the conditions in Eqs. (E2), (F1), (F4), and (F2), we get $n/k = 31.5d^2$ and $N_{CZ}/k = 84d^2$ for $(\alpha, \gamma, \delta, \delta', \epsilon) = (\frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{3}{2}, \frac{5}{2})d$. For the 6-6-6 lattice, considering the conditions in Eqs. (E4), (F1), and (F5), and (F3), we get $n/k \approx 28.7d^2$ and $N_{CZ}/k \approx 76.5d^2$ for $(\alpha, \gamma, \delta, \delta', \epsilon) = (0.464, 0.608, 0.464, 1.73, 2.54)d$.

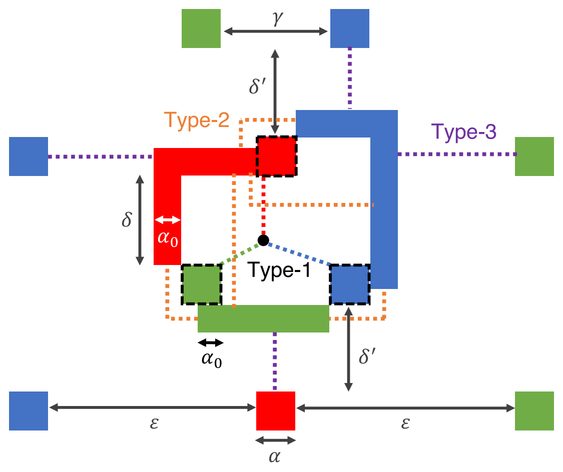


FIG. 35. Arrangement of defects for a logical phase gate in CCCS computation. Defects of width α_0 are extended in a spacelike manner to surround the Y-plane. Three types of error chains are considered: Type-1 is for those ending at the three defects in the concerning logical qubit and Type-2 (Type-3) is for those ending at two defects of different colors in the same logical qubit (different logical qubits).

APPENDIX G: RESOURCE ANALYSIS ON THE PHASE GATE USING STATE DISTILLATION IN RTCS COMPUTATION

For RTCS computation, we consider using state distillation for the logical phase gate. We assume that logical qubits are arranged in the same way as described in Appendix F. Throughout this Appendix, only the leading-order term on d is calculated for each value.

As shown in Fig. 20(a), a phase gate is implemented with an ancilla logical state $|Y_L\rangle := (|0_L\rangle + i|1_L\rangle)/\sqrt{2}$. A noisy $|Y_L\rangle$ is first prepared by state injection and then distilled with the circuit in Fig. 20(b). If the initial $|Y_L\rangle$ has an X_L or Z_L error with a probability of ϵ, the distilled state has an error rate of $7\epsilon^3$ [24,38]. The distillation circuit can be repeated multiple times to achieve a low enough error rate.

To obtain the resource overheads, we count physical qubits used for CNOT gates. We assume that multiple CNOT gates with the same control qubit can be implemented simultaneously, although it is uncertain for patch-based RTCS computation to the best of our knowledge. (It is known that such processes are possible if the rotated surface codes are used [73].)

Using the circuit in Fig. 20(b), we can find out a lower bound of required layers for each logical qubit. For example, denoting the number of layers used for a CNOT gate by T_R, the second one requires $2T_R$ layers, T_R for each of the groups of CNOT gates with the same control qubit in the distillation circuit and in the S_L circuit of Fig. 20(b). The fourth one requires $5T_R$ layers, $4T_R$ for the CNOT gates in the distillation and S_L circuits and T_R for waiting until the fourth group of single-control CNOT gates ends. Additionally,

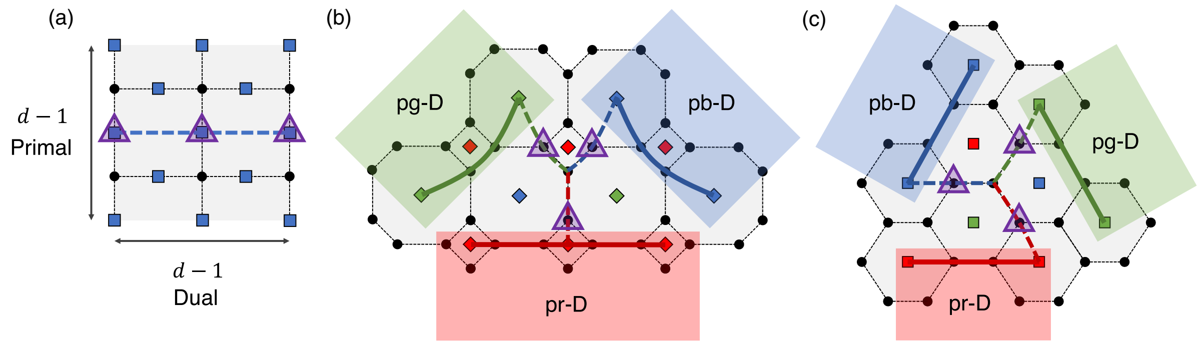


FIG. 36. Structure of a layer in the simplified defect model for the simulation regarding (a) RTCSs, (b) 4-8-8 CCCSs, or (c) 6-6-6 CCCSs, particularly when the code distance is $d = 3$. In (a), blue squares (black circles) indicate primal (dual) qubits. In (b) and (c), a colored solid line is a boundary corresponding to that color, which can be regarded as a part of a defect. For all of them, dashed lines are examples of primal error chains incurring Z_L errors. Purples triangles indicate the qubits in the error chains, which show that the code distances are three. Defect models for $d > 3$ can be constructed analogously by increasing the distances between the boundaries while keeping their shapes.

we need seven logical qubits for noisy $|Y_L\rangle$ states, each of which occupies T_R layers. The number of total physical qubits required for a distillation circuit is then lower-bounded by $(2+2+2+5+4+4+6+1+7)r_R T_R = 33 r_R T_R$, where r_R is the number of physical qubits per logical qubit in a layer.

If a $|Y_L\rangle$ state distilled once is used for a phase gate, total $35 r_R T_R$ ($\approx 840d^3$ for patch-based and $\approx 1030d^3$ for defect-based) physical qubits are required since the S_L circuit in Fig. 20(a) additionally occupies $2r_R T_R$ qubits. If a $|Y_L\rangle$ state distilled twice is used, $(33\times 7+33+2)r_R T_R = 266 r_R T_R$ ($\approx 6380d^3$ for patch-based and $\approx 7850d^3$ for defect-based) qubits are required.

APPENDIX H: DETAILS ON THE CALCULATIONS OF ERROR THRESHOLDS

We here present some details on the calculation of error thresholds presented in Sec. VI.

1. Simplified defect models

As mentioned in the main text, we simplify the defect models for efficient simulations. Instead of considering big regions containing the entire defects, we consider only regions surrounded by boundaries corresponding to the defects; that is, we take account only of error chains located in the "inner" regions surrounded by the defects. Since those error chains are strictly shorter than error chains passing outside the regions, we conjecture that this assumption does not affect the resulting Z_L error rates much.

Figure 36 shows single layers of the three simplified defect models for the simulations regarding RTCSs, 4-8-8 CCCSs, and 6-6-6 CCCSs, respectively. Each layer of the concerned RTCSs has the shape of a square with a side length of $d-1$ in the units of cells for the code distance d, where the boundaries are of different types (primal and dual). Any error chain connecting the two primal boundaries incurs a Z_L error. For CCCSs, we consider a region surrounded by three boundaries of different colors, where each boundary can be regarded as a part of a defect. Any error chain connecting the three boundaries incurs a Z_L error.

2. Decoding methods

a. Raussendorf's 3D cluster states

In an RTCS, the PC outcomes are decoded to locate errors at vacuum qubits via Edmonds' minimum-weight perfect matching algorithm (MWPM) [64–66], as frequently used in the literature [41,45,46,74]. Note that an error chain flips at most two PCs located at its ends, and if it flips one PC, it ends at the boundary. Hence, our goal is to figure out the most probable set of error chains based on the PC outcomes.

The decoding procedure is briefly summarized as follows. First, a graph is constructed from the PC outcomes. The vertex set of the graph contains two vertices for each flipped PC: One is the PC itself and the other is the "boundary vertex." An edge is connected between each pair of different PCs, each pair of a PC and the corresponding boundary vertex, and each pair of different boundary vertices. A "weight" value is assigned to each edge as follows: If both the vertices are PCs, the weight is the number of qubits in the shortest error chain between them. If only one of them is a PC, the weight is the number of qubits in the shortest error chain between the PC and the closest boundary. If both of them are boundary vertices, the weight is zero.

We use the MWPM algorithm via Blossom V software [67] to search for a set of edges of the graph constructed above which covers all the vertices, does not contain duplicated vertices, and minimizes the total weight. Each edge in the resulting set corresponds to a pair of PCs flipped by an error chain or a PC flipped by an error chain ending at the boundary, unless the edge connects two boundary vertices, which is ignored. We can thus locate errors from the error chain along the shortest path for each edge.

b. Color-code-based cluster states

The decoding method for RTCSs is not directly applicable to CCCSs, since an error in a CCCS flips at most three PCs, unlike the case of an RTCS. The decoding for each sample requires the application of the MWPM algorithm six times.

First, the outcomes of pb-PCs and pg-PCs are decoded to find the faces in $\mathcal{L}^{\mathrm{pr}}$ containing only one qubit with an error, via the method analogous to that for RTCSs. This is possible since each of such faces flips at most two (blue or green) PCs like an error in an RTCS. Note that each face in $\mathcal{L}^{\mathsf{pr}}$ corresponds to a pbAQ, pgAQ, or prL. Errors on pbAQs and pgAQs are thus obtained from this process, while errors on prLs are left ambiguous (since a prL is composed of two pCQs). Next, the left results for prLs obtained above and the outcomes of pr-PCs are decoded to locate errors on prAQs and pCQs, treating the parity of the number of errors in each prL like a PC. This is possible since an error on a prAQ or pCQ flips at most two among pr-PCs and the error parities of prLs.

All the errors are finally located by the above process. However, to make the decoding more accurate, we repeat it for $\mathcal{L}^{\mathsf{pb}}$ and $\mathcal{L}^{\mathsf{pg}}$ analogously and select the smallest set of decoded errors among the three results.

We last note the similarities and differences between our decoding method on CCCSs and the color-code decoders suggested in Refs. [75–77]. First, they have in common that the MWPM algorithm is first used in the (“shrunk” in our scheme and “restricted” in Refs. [76,77]) lattice corresponding to each color derived from the original lattice. However, the processes after that are different: The MPWM algorithm is used one more time in our method, while a “local lifting” procedure is applied in the other decoders. It is not straightforward to convert these color-code decoders to suit our scheme, since the lattice structures of CCCSs are in 3D and contain not only code qubits arranged on color-code lattices but also ancilla qubits. If such conversions are possible, it will be worth investigating which one performs better.

[1] A. Galindo and M. A. Martín-Delgado, Information and computation: Classical and quantum aspects, Rev. Mod. Phys. **74**, 347 (2002).

[2] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information* (Cambridge University Press, Cambridge, 2010).

[3] P. W. Shor, Scheme for reducing decoherence in quantum computer memory, Phys. Rev. A **52**, R2493(R) (1995).

[4] C. H. Bennett, D. P. DiVincenzo, J. A. Smolin, and W. K. Wootters, Mixed-state entanglement and quantum error correction, Phys. Rev. A **54**, 3824 (1996).

[5] R. Laflamme, C. Miquel, J. P. Paz, and W. H. Zurek, Perfect Quantum Error Correcting Code, Phys. Rev. Lett. **77**, 198 (1996).

[6] A. R. Calderbank and P. W. Shor, Good quantum error-correcting codes exist, Phys. Rev. A **54**, 1098 (1996).

[7] A. Steane, Multiple-particle interference and quantum error correction, Proc. R. Soc. London **452**, 2551 (1996).

[8] H. Bombín, Topological codes, in *Quantum Error Correction*, edited by D. A. Lidar and T. A. Brun (Cambridge University Press, Cambridge, 2013), pp. 455–481.

[9] D. G. Cory, M. D. Price, W. Maas, E. Knill, R. Laflamme, W. H. Zurek, T. F. Havel, and S. S. Somaroo, Experimental Quantum Error Correction, Phys. Rev. Lett. **81**, 2152 (1998).

[10] J. Chiaverini, D. Leibfried, T. Schaetz, M. D. Barrett, R. B. Blakestad, J. Britton, W. M. Itano, J. D. Jost, E. Knill, C. Langer, R. Ozeri, and D. J. Wineland, Realization of quantum error correction, Nature (London) **432**, 602 (2004).

[11] P. Schindler, J. T. Barreiro, T. Monz, V. Nebendahl, D. Nigg, M. Chwalla, M. Hennrich, and R. Blatt, Experimental repetitive quantum error correction, Science **332**, 1059 (2011).

[12] M. D. Reed, L. DiCarlo, S. E. Nigg, L. Sun, L. Frunzio, S. M. Girvin, and R. J. Schoelkopf, Realization of three-qubit quantum error correction with superconducting circuits, Nature (London) **482**, 382 (2012).

[13] D. Nigg, M. Müller, E. A. Martinez, P. Schindler, M. Hennrich, T. Monz, M. A. Martin-Delgado, and R. Blatt, Quantum computations on a topologically encoded qubit, Science **345**, 302 (2014).

[14] A. D. Córcoles, E. Magesan, S. J. Srinivasan, A. W. Cross, M. Steffen, J. M. Gambetta, and J. M. Chow, Demonstration of a quantum error detection code using a square lattice of four superconducting qubits, Nat. Commun. **6**, 6979 (2015).

[15] J. Kelly, R. Barends, A. G. Fowler, A. Megrant, E. Jeffrey, T. C. White, D. Sank, J. Y. Mutus, B. Campbell, Y. Chen, B. Chiaro, A. Dunsworth, I.-C. Hoi, C. Neill, P. J. J. O'Malley, C. Quintana, P. Roushan, A. Vainsencher, J. Wenner, A. N. Cleland *et al.*, State preservation by repetitive error detection in a superconducting quantum circuit, Nature (London) **519**, 66 (2015).

[16] N. Ofek, A. Petrenko, R. Heeres, P. Reinhold, Z. Leghtas, B. Vlastakis, Y. Liu, L. Frunzio, S. M. Girvin, L. Jiang, M. Mirrahimi, M. H. Devoret, and R. J. Schoelkopf, Extending the lifetime of a quantum bit with error correction in superconducting circuits, Nature (London) **536**, 441 (2016).

[17] C. K. Andersen, A. Remm, S. Lazar, S. Krinner, N. Lacroix, G. J. Norris, M. Gabureac, C. Eichler, and A. Wallraff, Repeated quantum error detection in a surface code, Nat. Phys. **16**, 875 (2020).

[18] A. Y. Kitaev, Quantum computations: Algorithms and error correction, Russ. Math. Surv. **52**, 1191 (1997).

[19] S. B. Bravyi and A. Y. Kitaev, Quantum codes on a lattice with boundary, arXiv:quant-ph/9811052.

[20] E. Dennis, A. Kitaev, A. Landahl, and J. Preskill, Topological quantum memory, J. Math. Phys. **43**, 4452 (2002).

[21] A. Y. Kitaev, Fault-tolerant quantum computation by anyons, Ann. Phys. **303**, 2 (2003).

[22] A. G. Fowler, A. M. Stephens, and P. Groszkowski, High-threshold universal quantum computation on the surface code, Phys. Rev. A **80**, 052312 (2009).

[23] H. Bombin and M. A. Martin-Delgado, Quantum measurements and gates by code deformation, J. Phys. A: Math. Theor. **42**, 095302 (2009).

[24] A. G. Fowler, M. Mariantoni, J. M. Martinis, and A. N. Cleland, Surface codes: Towards practical large-scale quantum computation, Phys. Rev. A **86**, 032324 (2012).

[25] B. M. Terhal, Quantum error correction for quantum memories, Rev. Mod. Phys. **87**, 307 (2015).

[26] H. Bombin and M. A. Martin-Delgado, Topological Quantum Distillation, Phys. Rev. Lett. **97**, 180501 (2006).

[27] A. G. Fowler, Two-dimensional color-code quantum computation, Phys. Rev. A **83**, 042310 (2011).

[28] M. S. Kesselring, F. Pastawski, J. Eisert, and B. J. Brown, The boundaries and twist defects of the color code and their applications to topological quantum computation, Quantum **2**, 101 (2018).

[29] That a logical gate U is transversal means that U can be expressed as $U = U_1 U_2 \cdots$ such that a unitary operator U_i acts only on the ith physical qubit for each logical qubit for all i's. For example, if $X_L := X_1 \cdots X_n$ for $[[n, 1, d]]$ code where X_i is the X operator on the ith physical qubit, X_L is transversal. Specific 2D color codes implement a logical Hadamard gate by the combination of a Hadamard gate on every physical qubit, and similarly for a logical phase gate [26,27].
[30] H. Bombin and M. A. Martin-Delgado, Topological Computation Without Braiding, Phys. Rev. Lett. **98**, 160502 (2007).
[31] H. Bombin and M. A. Martin-Delgado, Exact topological quantum order in $d = 3$ and beyond: Branyons and brane-net condensates, Phys. Rev. B **75**, 075103 (2007).
[32] H. Bombín, Gauge color codes: Optimal transversal gates and gauge fixing in topological stabilizer codes, New J. Phys. **17**, 083002 (2015).
[33] A. Kubica and M. E. Beverland, Universal transversal gates with color codes: A simplified approach, Phys. Rev. A **91**, 032330 (2015).
[34] F. H. E. Watson, E. T. Campbell, H. Anwar, and D. E. Browne, Qudit color codes and gauge color codes in all spatial dimensions, Phys. Rev. A **92**, 022312 (2015).
[35] A. Kubica, M. E. Beverland, F. Brandão, J. Preskill, and K. M. Svore, Three-Dimensional Color Code Thresholds Via Statistical-Mechanical Mapping, Phys. Rev. Lett. **120**, 180501 (2018).
[36] S. Bravyi and A. Kitaev, Universal quantum computation with ideal Clifford gates and noisy ancillas, Phys. Rev. A **71**, 022316 (2005).
[37] C. Jones, Multilevel distillation of magic states for quantum computing, Phys. Rev. A **87**, 042305 (2013).
[38] R. Raussendorf, J. Harrington, and K. Goyal, Topological fault-tolerance in cluster state quantum computation, New J. Phys. **9**, 199 (2007).
[39] R. Raussendorf and H. J. Briegel, A One-Way Quantum Computer, Phys. Rev. Lett. **86**, 5188 (2001).
[40] R. Raussendorf, D. E. Browne, and H. J. Briegel, Measurement-based quantum computation on cluster states, Phys. Rev. A **68**, 022312 (2003).
[41] R. Raussendorf, J. Harrington, and K. Goyal, A fault-tolerant one-way quantum computer, Ann. Phys. **321**, 2242 (2006).
[42] R. Raussendorf and J. Harrington, Fault-Tolerant Quantum Computation with High Threshold in Two Dimensions, Phys. Rev. Lett. **98**, 190504 (2007).
[43] A. G. Fowler and K. Goyal, Topological cluster state quantum computing, Quantum Inf. Comput. **9**, 721 (2009).
[44] D. Herr, A. Paler, S. J. Devitt, and F. Nori, Lattice surgery on the Raussendorf lattice, Quantum Sci. Technol. **3**, 035011 (2018).
[45] S. D. Barrett and T. M. Stace, Fault Tolerant Quantum Computation with Very High Threshold for Loss Errors, Phys. Rev. Lett. **105**, 200502 (2010).
[46] A. C. Whiteside and A. G. Fowler, Upper bound for loss in practical topological-cluster-state quantum computing, Phys. Rev. A **90**, 052316 (2014).
[47] Y. Li, S. D. Barrett, T. M. Stace, and S. C. Benjamin, Fault Tolerant Quantum Computation with Nondeterministic Gates, Phys. Rev. Lett. **105**, 250502 (2010).
[48] A. Bolt, G. Duclos-Cianci, D. Poulin, and T. M. Stace, Foliated Quantum Error-Correcting Codes, Phys. Rev. Lett. **117**, 070501 (2016).
[49] A. Bolt, D. Poulin, and T. M. Stace, Decoding schemes for foliated sparse quantum error-correcting codes, Phys. Rev. A **98**, 062302 (2018).
[50] B. J. Brown and S. Roberts, Universal fault-tolerant measurement-based quantum computation, Phys. Rev. Research **2**, 033305 (2020).
[51] M. A. Nielsen, Optical Quantum Computation Using Cluster States, Phys. Rev. Lett. **93**, 040503 (2004).
[52] C. M. Dawson, H. L. Haselgrove, and M. A. Nielsen, Noise thresholds for optical cluster-state quantum computation, Phys. Rev. A **73**, 052306 (2006).
[53] N. C. Menicucci, P. van Loock, M. Gu, C. Weedbrook, T. C. Ralph, and M. A. Nielsen, Universal Quantum Computation with Continuous-Variable Cluster States, Phys. Rev. Lett. **97**, 110501 (2006).
[54] S. J. Devitt, A. G. Fowler, A. M. Stephens, A. D. Greentree, L. C. L. Hollenberg, W. J. Munro, and K. Nemoto, Architectural design for a topological cluster state quantum computer, New J. Phys. **11**, 083032 (2009).
[55] D. A. Herrera-Martí, A. G. Fowler, D. Jennings, and T. Rudolph, Photonic implementation for the topological cluster-state quantum computer, Phys. Rev. A **82**, 032332 (2010).
[56] K. Fujii and Y. Tokunaga, Fault-Tolerant Topological One-Way Quantum Computation with Probabilistic Two-Qubit Gates, Phys. Rev. Lett. **105**, 250503 (2010).
[57] C. R. Myers and T. C. Ralph, Coherent state topological cluster state production, New J. Phys. **13**, 115015 (2011).
[58] X.-C. Yao, T.-X. Wang, H.-Z. Chen, W.-B. Gao, A. G. Fowler, R. Raussendorf, Z.-B. Chen, N.-L. Liu, C.-Y. Lu, Y.-J. Deng, Y.-A. Chen, and J.-W. Pan, Experimental demonstration of topological error correction, Nature (London) **482**, 489 (2012).
[59] M. Gimeno-Segovia, P. Shadbolt, D. E. Browne, and T. Rudolph, From Three-Photon Greenberger-Horne-Zeilinger States to Ballistic Universal Quantum Computation, Phys. Rev. Lett. **115**, 020502 (2015).
[60] Y. Li, P. C. Humphreys, G. J. Mendoza, and S. C. Benjamin, Resource Costs for Fault-Tolerant Linear Optical Quantum Computing, Phys. Rev. X **5**, 041007 (2015).
[61] S. Omkar, Y. S. Teo, and H. Jeong, Resource-Efficient Topological Fault-Tolerant Quantum Computation with Hybrid Entanglement of Light, Phys. Rev. Lett. **125**, 060501 (2020).
[62] S. Omkar, Y. S. Teo, S.-W. Lee, and H. Jeong, Highly photon-loss-tolerant quantum computing using hybrid qubits, Phys. Rev. A **103**, 032602 (2021).
[63] C. Horsman, A. G. Fowler, S. Devitt, and R. Van Meter, Surface code quantum computing by lattice surgery, New J. Phys. **14**, 123011 (2012).
[64] J. Edmonds, Paths, trees, and flowers, Can. J. Math. **17**, 449 (1965).
[65] J. Edmonds, Maximum matching and a polyhedron with 0, 1-vertices, J. Res. Natl. Bur. Stand. **69B**, 125 (1965).
[66] A. G. Fowler, Minimum weight perfect matching of fault-tolerant topological quantum error correction in average $O(1)$ parallel time, Quantum Inf. Comput. **15**, 145 (2015).
[67] V. Kolmogorov, Blossom V: A new implementation of a minimum cost perfect matching algorithm, Math. Program. Comput. **1**, 43 (2009).
[68] D. Litinski, Magic State distillation: Not as costly as you think, Quantum **3**, 205 (2019).

[69] C. Chamberland and K. Noh, Very low overhead fault-tolerant magic state preparation using redundant ancilla encoding and flag qubits, npj Quant. Inf. **6**, 91 (2020).

[70] H. Bombin, 2D quantum computation with 3D topological codes, arXiv:1810.09571.

[71] H. Bombin, Transversal gates and error propagation in 3D topological codes, arXiv:1810.09575.

[72] L. Lao, B. van Wee, I. Ashraf, J. van Someren, N. Khammassi, K. Bertels, and C. G. Almudever, Mapping of lattice surgery-based quantum circuits on surface code architectures, Quantum Sci. Tech. **4**, 015005 (2018).

[73] D. Litinski and F. v. Oppen, Lattice surgery with a twist: Simplifying Clifford gates of surface codes, Quantum **2**, 62 (2018).

[74] A. G. Fowler, A. C. Whiteside, A. L. McInnes, and A. Rabbani, Topological Code Autotune, Phys. Rev. X **2**, 041003 (2012).

[75] N. Delfosse, Decoding color codes by projection onto surface codes, Phys. Rev. A **89**, 012317 (2014).

[76] A. Kubica and N. Delfosse, Efficient color code decoders in $d \geqslant 2$ dimensions from toric code decoders, arXiv:1905.07393.

[77] C. Chamberland, A. Kubica, T. J. Yoder, and G. Zhu, Triangular color codes on trivalent graphs with flag qubits, New J. Phys. **22**, 023019 (2020).