이학박사학위논문

Universal Resource-efficient Topological Measurement-based Quantum Computing

유니버설하고 자원 효율적인 위상기하학적 측정 기반 양자 컴퓨팅

2023년 2월

서울대학교 대학원
물리천문학부
이석형

Universal Resource-efficient Topological Measurement-based Quantum Computing

유니버설하고 자원 효율적인 위상기하학적 측정 기반 양자 컴퓨팅

지도교수 정현석

이 논문을 이학박사 학위논문으로 제출함
2023년 1월

서울대학교 대학원
물리천문학부
이석형

이석형의 박사 학위논문을 인준함
2022년 12월

위 원 장	안 경 원	(인)
부위원장	정 현 석	(인)
위 원	신 용 일	(인)
위 원	김 도 헌	(인)
위 원	이 승 우	(인)

Universal Resource-efficient Topological Measurement-based Quantum Computing

Seok-Hyung Lee

Supervised by

Professor **Hyunseok Jeong**

A Dissertation

Submitted to the Faculty of

Seoul National University

in Partial Fulfillment of

the Requirements for the Degree of

Doctor of Philosophy

Jan. 2023

Department of Physics and Astronomy

The Graduate School

Seoul National University

Abstract

Universal Resource-efficient Topological Measurement-based Quantum Computing

Seok-Hyung Lee
Department of Physics and Astronomy
The Graduate School
Seoul National University

Measurement-based quantum computing (MBQC) is a methodology of quantum computing that is conducted with single-qubit measurements on large-scale entangled states called a cluster state, which is adequate in optical systems. In particular, MBQC can be tolerant to small faults by utilizing topological quantum error-correcting codes. This dissertation introduces two topological MBQC protocols that are advantageous over previous protocols in terms of fault tolerance and resource efficiency.

In the first part, we propose a topological MBQC protocol with a family of cluster states constructed based on two-dimensional color codes. The conventional topological MBQC protocol with Raussendorf's three-dimensional cluster states (RTCSs) has a drawback: The Hadamard and phase gates that are essential for building up arbitrary logic gates cannot be implemented

natively without additional techniques, which makes the protocol less feasible. We resolve this problem by altering RTCSs with color-code-based cluster states. Specifically, we show that the Hadamard and phase gates can be implemented natively in a fault-tolerant manner, which leads to about 26 times resource reduction compared to the protocol with RTCSs using state distillation.

In the second part, we suggest a linear-optical topological MBQC protocol employing multiphoton qubits based on the parity encoding. The nondeterministic nature of entangling operations and photon losses hinder the large-scale generation of cluster states and introduce logical errors in linear-optical MBQC. Our protocol turns out to be highly photon-loss tolerant and resource-efficient even under the effects of nonideal entangling operations that unavoidably corrupt nearby qubits. For the realistic error analysis, we introduce a Bayesian methodology to track errors caused by such detrimental effects. Notably, we show that our protocol is advantageous over several other existing protocols in terms of fault-tolerance, resource overhead, or feasibility of basic elements.

Keywords : Quantum Computing, Quantum Information, Quantum Error Correction, Measurement-based Quantum Computing

Student Number : 2017-27328

Contents

List of Figures

Figure 17. Construction of a Hadamard gate from a primal logical qubit to a dual one. Each colored single (double) line is the primal (dual) defect of that color. $S_{Z\mathsf{p}}$ is a dj-CS ending at the three primal defects and the (t_H+1)-layer. Similarly, $S_{X\mathsf{d}}$ is a pj-CS ending at the three dual defects and the t_H-layer. $S_{Z\mathsf{p}}$ and $S_{X\mathsf{d}}$ are chosen so that their supports overlap in the t_H- and (t_H+1)-layer between the defects. Next, $S_{X\mathsf{p}}$ is a pr-CS which surrounds the pg-D and ends at the t_H-layer. $S_{Z\mathsf{d}}$ is a dr-CS which surrounds the dg-D and reaches the (t_H-1)-layer. Note that $S_{Z\mathsf{d}}$ does not have a boundary in the (t_H-1)-layer; instead, its interior is penetrated by the pg-D. This is possible since $S_{Z\mathsf{d}}$ and the pg-D have different primalities. $S_{X\mathsf{p}}$ and $S_{Z\mathsf{d}}$ are chosen so that their supports overlap in the t_H-layer. Finally, $S_{ZX} := S_{Z\mathsf{p}}S_{X\mathsf{d}}$ and $S_{XZ} := S_{X\mathsf{p}}S_{Z\mathsf{d}}$ transform the logical Pauli operators as Eq. (3.10). 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. 56

List of Tables

Chapter 1

Introduction

Currently, we are in the nascent stage of quantum computing technology. Quantum computing utilizes quantum natures such as superposition or entanglement to handle computational tasks. It started to attract attention due to the discovery that several quantum computing algorithms such as Shor's factoring algorithm [4] and Grover's search algorithm [5] are known to have at most exponential speedup compared to their classical counterpart. Not only that, physics problems that are natively quantum, such as simulating many-body quantum systems [6] or finding the ground states of complicated Hamiltonians [7], are expected to be solved efficiently with quantum computing.

There is still a long way to go to get a decent quantum computing implementation. From a theoretical point of view, there are three major challenges: **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, if quantum gates in a **universal set of gates** are implementable, any unitary operation can be approximated to an arbitrary accuracy [8, 9]. One typical example of a universal set of gates is composed of the controlled-NOT (CNOT), Hadamard, phase, and T gates [9], which is respectively expressed in matrix

forms as

$$U_{\mathrm{CNOT}} := \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}, \; H := \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}, \tag{1.1}$$
$$S := \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix}, \qquad T := \begin{pmatrix} 1 & 0 \\ 0 & e^{\frac{\pi}{4}i} \end{pmatrix}.$$

Fault-tolerance means that adverse effects of errors during quantum computing are suppressed so that faithful computing results can be obtained under small enough faults. A representative way to achieve fault-tolerance is to use **quantum error-correcting (QEC) codes** in which a single logical qubit is composed of multiple physical qubits. QEC codes vary from simple codes with few physical qubits [10, 11, 12, 13, 14] to advanced **topological** stabilizer codes defined on lattice structures of qubits allowing only local interactions [15]. In particular, **surface codes** [16, 17, 18, 19, 20, 21, 1, 22] and **two-dimensional (2D) color codes** [23, 24, 15, 25] are two different families of topological codes defined on 2D lattices, which are promising due to their universality and relatively high fault-tolerance. Recently, alternative approaches for controlling errors in specific problems of estimating the expected values of operators have been suggested, which are collectively referred to as *error mitigation* [26, 27, 28]. Although these techniques are far less hardware-demanding than quantum error correction, which makes them suitable for noisy intermediate-scale quantum (NISQ) devices, their

performance is insufficient for application to complex quantum algorithms. In this dissertation, we do not cover error mitigation.

Lastly, fault-tolerant universal quantum computing typically requires enormous resources, 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 several logic gates to be fault-tolerant [29, 20, 1, 30]. For example, one round of a typical state distillation protocol to implement a logical T gate requires 15 ancillary logical qubits [29, 31, 1]. It is thus desirable to minimize the need for state distillation or find an efficient state distillation protocol. Not only that, there exist various platform-dependent problems that hinder the reduction of resource overheads, such as the non-deterministic nature of entangling operations in linear-optical systems.

Measurement-based quantum computing (MBQC) is a quantum computing methodology that is processed by single-qubit measurements on a large entangled state called a **cluster state** [32, 33, 34, 35, 31, 36]. The initial MBQC schemes via cluster states on 2D planes [32, 33] were universal but not fault-tolerant. To achieve fault-tolerance, the concept of topological codes is exquisitely combined with MBQC. Specifically, **Raussendorf's three-dimensional cluster states (RTCSs)** constructed based on surface codes allow universal and fault-tolerant MBQC with topologically-encoded logical qubits [34, 35, 31, 36, 2].

In this dissertation, we propose two different universal fault-tolerant MBQC protocols and delve into their performance and resource efficiency. We first briefly review essential background knowledge in Chapter 2, in-

cluding the stabilizer formalism, quantum error correcting codes, and the operational principle of MBQC. In Chapter 3, we suggest an MBQC protocol using a family of cluster states constructed based on 2D color codes, not surface codes which is the basis of RTCSs. We verify that this new protocol is advantageous over the protocol with RTCSs in terms of resource efficiency when implementing the Hadamard and phase gates, owing to the self-duality of 2D color codes. While Chapter 3 addresses platform-independent theory, in Chapter 4 we regard linear-optical systems. We propose a linear-optical MBQC protocol employing multiphoton qubits based on the parity encoding, which turns out to be highly photon-loss tolerant and resource-efficient compared to other existing protocols. For realistic error analysis, we introduce a Bayesian methodology to track errors caused by the detrimental effects of nonideal entangling operations during the construction of cluster states. We conclude with final remarks in Chapter 5.

Chapters 3 and 4 are respectively based on the following two papers:

- Seok-Hyung Lee and Hyunseok Jeong, "Universal hardware-efficient topological measurement-based quantum computation via color-code-based cluster states," Phys. Rev. Research **4**, 013010 (2022) [37].

- Seok-Hyung Lee, Srikrishna Omkar, Yong Siah Teo, and Hyunseok Jeong, "Parity-encoding-based quantum computing with Bayesian error tracking," arXiv:2207.06805 [quant-ph] (2022) [38].

Chapter 2

Preliminary

In this chapter, we briefly introduce essential background information. We first cover the stabilizer formalism, which is a fundamental mathematical language for describing quantum error correction (QEC) theory and measurement-based quantum computing (MBQC). Subsequently, we study how MBQC operates, including the structures of resource states, the implementation of quantum gates, and the error-correcting scheme.

The followings show notations used throughout the dissertation:

- For two operators O_1 and O_2,

$$\begin{cases} [O_1,\ O_2] := O_1O_2 - O_2O_1 & \text{(Commutator of } O_1 \text{ and } O_2\text{)} \\ \{O_1,\ O_2\} := O_1O_2 + O_2O_1 & \text{(Anticommutator of } O_1 \text{ and } O_2\text{)} \end{cases}$$

- $\langle \mathcal{G} \rangle = \langle G_1,\ G_2,\ \cdots,\ G_n \rangle$ for a set of operators $\mathcal{G} = \{G_1,\ G_2,\ \cdots,\ G_n\}$: Group generated by $\mathcal{G}$ under their multiplication. Namely, every element in the group can be expressed as the product of elements in $\mathcal{G}$.

- I, X, Y, Z: Single-qubit identity, Pauli-X, Y, and Z operators, respectively.

- $\mathcal{P}_n$: n-qubit **Pauli group** defined as

$$\mathcal{P}_n := \left\langle iI^{\otimes n},\, X_1,\, Z_1,\, X_2,\, Z_2,\, \cdots,\, X_n,\, Z_n \right\rangle,$$

  where O_j for a single-qubit operator O and $j \in \{1, \cdots, n\}$ is an operator in the n-qubit Hilbert space that is the tensor product of O on the j-th qubit and the identities on the other qubits.

- $\mathrm{supp}(O)$ for a multi-qubit operator O: Support of O, which means the set of qubits on which O acts non-trivially.

- $\mathrm{supp}_P(O)$ for a multi-qubit Pauli operator $O \in \mathcal{P}_n$ and a single-qubit Pauli operator $P \in \{X, Y, Z\}$: P-support of an operator O, which means the set of qubits on which O acts as P.

2.1 Stabilizer formalism

The stabilizer formalism, which was first developed by Daniel Gottesman [39], is a powerful methodology for describing certain multi-qubit quantum states and quantum operations on them. We here organize its basic concepts including the definitions of stabilizer groups and subspaces and the effects of unitary operations or measurements on them. See also the following references for comprehensive introductions of the formalism.

- M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information*, Chapter 6, Cambridge University Press (2010) [9].

- D. Bacon, *Quantum Error Correction* (edited by D. A. Lidar and T.

A. Brun), Chapter 2, Cambridge University Press (2013) [40].

2.1.1 Stabilizer group and subspace

We define a stabilizer group in an n-qubit Hilbert space $\mathcal{H}_n$ as follows:

Definition 2.1 (**Stabilizer group**). For an *abelian* subgroup $\mathcal{S}$ of $\mathcal{P}_n$ that satisfies $-I^{\otimes n} \notin \mathcal{S}$, the common eigenspace $\mathcal{H}_\mathcal{S}$ of the elements in $\mathcal{S}$ with eigenvalue $+1$, namely,

$$\mathcal{H}_\mathcal{S} := \left\{ |\Psi\rangle \in \mathcal{H}_n : \ \forall g \in \mathcal{S},\ g|\Psi\rangle = |\Psi\rangle \right\} \tag{2.1}$$

is called the **stabilizer subspace** stabilized by $\mathcal{S}$, and $\mathcal{S}$ is referred to as the **stabilizer group** of the space $\mathcal{H}_\mathcal{S}$. Each element in $\mathcal{S}$ is called a **stabilizer** of $\mathcal{H}_\mathcal{S}$.

Note that, for a generating set $\mathcal{G}_\mathcal{S}$ of $\mathcal{S}$, the condition in Eq. (2.1) can be simplified as $\forall g \in \mathcal{G}_\mathcal{S},\ g|\Psi\rangle = |\Psi\rangle$. The next proposition shows an important property of a stabilizer subspace that its dimension is determined by the number of generators of its stabilizer group:

Proposition 2.1 (**Dimension of stabilizer subspace**). *For a stabilizer subspace $\mathcal{H}_\mathcal{S} < \mathcal{H}_n$, if a minimum generating set of $\mathcal{S}$ is $\mathcal{G}$, $\mathcal{H}_\mathcal{S}$ is a 2^k-dimensional vector space where $k = n - |\mathcal{G}|$.*

This proposition can be intuitively understood as follows: For each stabilizer generator added, only one of the ± 1 eigenspaces that together generate the original space survives, thus the dimension of the stabilizer subspace

is cut in half. See Ref. [9] for rigorous proof. Due to this proposition, a stabilizer subspace can encode $k := n - |\mathcal{G}|$ **logical** qubits if $|\mathcal{G}| < n$, which are composed of n **physical** qubits. Such an encoding scheme is called a **stabilizer code** and the corresponding stabilizer subspace is called its **code space**. If $|\mathcal{G}| = n$, the stabilizer subspace contains only one quantum state $|\Psi_{\mathcal{S}}\rangle$ up to a global phase, which is simply called the **stabilizer state** stabilized by $\mathcal{S}$.

Example 2.1 (**Steane's 7-qubit code**)**.** Steane's 7-qubit code is defined on the stabilizer subspace of a seven-qubit Hilbert space stabilized by

$$\mathcal{S} = \langle g_1,\ g_2,\ g_3,\ g_4,\ g_5,\ g_6,\ g_7 \rangle ,$$

where

$$\begin{aligned} g_1 &:= X_1X_2X_3X_7, \quad g_2 := X_3X_4X_5X_7, \quad g_3 := X_1X_5X_6X_7, \\ g_4 &:= Z_1Z_2Z_3Z_7, \quad g_5 := Z_3Z_4Z_5Z_7, \quad g_6 := Z_1Z_5Z_6Z_7. \end{aligned} \tag{2.2}$$

The code encodes one logical qubit since it has six generators.

Example 2.2 (**Bell states**)**.** The four Bell states on two qubits which respectively have the basis of $\{|0\rangle, |1\rangle\}$ are defined as

$$\begin{aligned} |\Phi_\pm\rangle &:= |0\rangle \otimes |0\rangle \pm |1\rangle \otimes |1\rangle , \\ |\Psi_\pm\rangle &:= |0\rangle \otimes |1\rangle \pm |1\rangle \otimes |0\rangle . \end{aligned}$$

Each Bell state is stabilized by $\left\langle m_{\text{sign}} X \otimes X,\ m_{\text{lett}} Z \otimes Z \right\rangle$, where $m_{\text{sign}} = \pm 1$ for $|\Phi_\pm\rangle$ and $|\Psi_\pm\rangle$ and $m_{\text{lett}} = +1$ (-1) for $|\Phi_\pm\rangle$ $(|\Psi_\pm\rangle)$.

2.1.2 Unitary operation on a stabilizer subspace

We now study the effect of a unitary operation $U \in \mathcal{U}(\mathcal{H}_n)$ on a stabilizer subspace $\mathcal{H}_\mathcal{S} \subseteq \mathcal{H}_n$, where $\mathcal{U}(\mathcal{H}_n)$ is the set of unitary operations on $\mathcal{H}_n$. For every $|\Psi\rangle \in \mathcal{H}_\mathcal{S}$ and every $g \in \mathcal{S}$,

$$U|\Psi\rangle = Ug|\Psi\rangle = UgU^\dagger U|\Psi\rangle,$$

thus the transformed subspace $U\mathcal{H}_\mathcal{S}$ is the common eigenspace of the elements in $U\mathcal{S}U^\dagger := \{UgU^\dagger : g \in \mathcal{S}\}$ with eigenvalue $+1$. To say that $U\mathcal{H}_\mathcal{S}$ is the stabilizer subspace stabilized by $U\mathcal{S}U^\dagger$, $U\mathcal{S}U^\dagger$ should be a subgroup of $\mathcal{P}_n$, which holds if U transforms every Pauli operator into a Pauli operator under the Heisenberg picture (namely, $UPU^\dagger \in \mathcal{P}_n$ for every $P \in \mathcal{P}_n$). Note that such unitary operations form the n-qubit **Clifford group**, whose elements are termed **Clifford operations** or **gates**. It is known that any Clifford gate can be expressed as a combination of the Hadamard (H), phase (S), and CNOT (U_{CNOT}) gates on the n qubits, which are defined in Eq. (1.1). Each of these gates transforms the single- or two-qubit Pauli operators under the Heisenberg picture as follows:

$$\begin{aligned}
HXH^\dagger = Z, \quad SXS^\dagger = Y, \quad & U_{\text{CNOT}}(X \otimes I)U_{\text{CNOT}}^\dagger = X \otimes X, \\
HZH^\dagger = X, \quad SZS^\dagger = Z, \quad & U_{\text{CNOT}}(Z \otimes I)U_{\text{CNOT}}^\dagger = Z \otimes I, \\
& U_{\text{CNOT}}(I \otimes X)U_{\text{CNOT}}^\dagger = I \otimes X, \\
& U_{\text{CNOT}}(I \otimes Z)U_{\text{CNOT}}^\dagger = Z \otimes Z,
\end{aligned}$$

where the first (second) qubit is the control (target) for the CNOT gate.

The above method seems quite straightforward, but it implies the power of the stabilizer formalism. An n-qubit state generally requires 2^n amplitudes for its description and all of them should be tracked according to unitary operations applied to the qubits. However, provided that the state is a stabilized state, only n stabilizer generators are sufficient to describe the state and track its transformation. Such a reduction of computational cost is extremely useful when simulating quantum circuits. Additionally, it is directly related to an important theorem in quantum computational theory, the **Gottesman-Knill theorem** [41] that a quantum computation involving only state preparations in the computational basis, Clifford gates, measurements of Pauli operators, and classical controls conditioned on the outcomes of the measurements can be efficiently simulated on a classical computer.

Any unitary operation outside $\mathcal{S}$ that preserves the stabilizer group serves as a nontrivial logical operation in the code space. In other words, defining the **normalizer** $\mathcal{N}(\mathcal{S})$ of $\mathcal{S}$ by

$$\mathcal{N}(\mathcal{S}) := \left\{U \in \mathcal{U}(\mathcal{H}_n) : \; U\mathcal{S}U^\dagger = \mathcal{S}\right\},$$

any element in $\mathcal{N}(\mathcal{S}) \setminus \mathcal{S}$ changes the state of the logical qubits encoded in $\mathcal{H}_\mathcal{S}$ while preserving the code space. Note that such logical operations are equivalent up to the multiplication of an arbitrary stabilizer.

In particular, we can find a set of independent Hermitian operators

$$\overline{X}_1, \; \cdots, \overline{X}_k, \overline{Z}_1, \; \cdots, \overline{Z}_k \in \mathcal{P}_n \cap \mathcal{N}(\mathcal{S}) \setminus \mathcal{S}$$

that commute with each other except for each pair of $\overline{X}_j$ and $\overline{Z}_j$ for the same subscript j, which anticommutes. (Note that $\mathcal{P}_n \cap \mathcal{N}(\mathcal{S})$ is a group.) Here, the term "independent" means that no operator can be expressed as a product of others. $\overline{X}_j$ ($\overline{Z}_j$) represents the **logical Pauli-X (Z) operator** of the j-th logical qubit. We further define the **logical Pauli group** $\overline{\mathcal{P}}(\mathcal{S})$ by

$$\overline{\mathcal{P}}(\mathcal{S}) := \left\langle i,\, \overline{X}_1,\, \overline{Z}_1,\, \cdots,\, \overline{X}_k,\, \overline{Z}_k \right\rangle \leq \mathcal{P}_n \cap \mathcal{N}(\mathcal{S})$$

as a natural extension of the Pauli group to the logical space. We note the following two propositions regarding $\overline{\mathcal{P}}(\mathcal{S})$.

Proposition 2.2. *All the elements of $\mathcal{P}_n \cap \mathcal{N}(\mathcal{S})$, which include the elements of $\overline{\mathcal{P}}(\mathcal{S})$, commute with every element of $\mathcal{S}$.*

Proof. If $\overline{Q} \in \overline{\mathcal{P}}(\mathcal{S})$ anticommutes with $P \in \mathcal{S}$, $\overline{Q}P\overline{Q}^\dagger = -P \in \mathcal{S}$, which contradicts to $-I^{\otimes n} \notin \mathcal{S}$. □

Proposition 2.3. *$\mathcal{P}_n \cap \mathcal{N}(\mathcal{S})$ is the direct sum of $\mathcal{S}$ and $\overline{\mathcal{P}}(\mathcal{S})$; namely,*

$$\mathcal{P}_n \cap \mathcal{N}(\mathcal{S}) = \left\langle i,\, g_1,\, \cdots,\, g_{n-k},\, \overline{X}_1,\, \cdots,\, \overline{X}_k,\, \overline{Z}_1,\, \cdots,\, \overline{Z}_k \right\rangle, \tag{2.3}$$

where $\{g_1,\, \cdots,\, g_{n-k}\}$ is a minimum generating set of $\mathcal{S}$.

Proof. Let P be an arbitrary element of $\mathcal{P}_n \cap \mathcal{N}(\mathcal{S})$. For each $l \in \{1,\, \cdots,\, k\}$, P commutes at least one of $\overline{X}_l$, $\overline{Z}_l$, and $\overline{Y}_l := i\overline{X}_l\overline{Z}_l$, because

$$\left[\overline{X}_l\overline{Z}_l,\, P\right] = \overline{X}_l\left\{\overline{Z}_l,\, P\right\} - \left\{\overline{X}_l,\, P\right\}\overline{Z}_l = 0$$

if P does not commute with both $\overline{X}_l$ and $\overline{Z}_l$. Let $\overline{Q}_l$ denote one of $\overline{X}_l$, $\overline{Z}_l$, and

$\overline{Y}_l$ that commutes with P. Then

$$\mathcal{S}' := \langle g_1,\ \cdots,\ g_{n-k},\ \overline{Q}_1,\ \cdots,\ \overline{Q}_k \rangle$$

is an abelian subgroup of $\mathcal{P}_n$ that does not contain $-I^{\otimes n}$. Since $\mathcal{S}'$ has n independent generators, its stabilizer space $\mathcal{H}_{\mathcal{S}'}$ contains only one state $|\Psi_{\mathcal{S}'}\rangle$ up to a global phase. For every $g' \in \mathcal{S}'$, $g'P|\Psi_{\mathcal{S}'}\rangle = Pg'|\Psi_{\mathcal{S}'}\rangle = P|\Psi_{\mathcal{S}'}\rangle$, thus $P|\Psi_{\mathcal{S}'}\rangle \in \mathcal{H}_{\mathcal{S}'}$. Therefore, $P|\Psi_{\mathcal{S}'}\rangle = e^{i\theta}|\Psi_{\mathcal{S}'}\rangle$ for a real value θ, which means that $e^{-i\theta}P \in \mathcal{S}'$. Since $\mathcal{S}' < \mathcal{P}_n$ and $P \in \mathcal{P}_n$, $e^{-i\theta}$ is either ± 1 or $\pm i$. Hence, P can be expressed as a product of elements in $\{i,\ g_1,\ \cdots,\ g_{n-k},\ \overline{Q}_1,\ \cdots,\ \overline{Q}_k\}$, which proves the proposition. □

To reveal the effect of an arbitrary element U in $\mathcal{N}(\mathcal{S}) \setminus \mathcal{S}$ on the logical qubits, one can check the transformation of the logical Pauli operators upon U. For example, if a unitary operation $\overline{H}_j \in \mathcal{N}(\mathcal{S})$ satisfies $\overline{H}_j \overline{X}_j \overline{H}_j^\dagger = \overline{Z}_j$ and $\overline{H}_j \overline{Z}_j \overline{H}_j^\dagger = \overline{X}_j$ and commutes with all the other logical Pauli operators, $\overline{H}_j$ can be regarded as the logical Hadamard gate on the j-th logical qubit.

Example 2.3. For Steane's 7-qubit code defined in Example 2.1, the logical Pauli operators of the logical qubit are

$$X_L := \prod_{j=1}^{7} X_j, \qquad Z_L := \prod_{j=1}^{7} Z_j.$$

The operator $H_L = H^{\otimes 7}$, where H is the Hadamard gate on a physical qubit, serves as the logical Hadamard gate since $H_L \mathcal{S} H_L^\dagger = \mathcal{S}$, $H_L X_L H_L^\dagger = Z_L$, and $H_L Z_L H_L^\dagger = X_L$.

2.1.3 Measurement on a stabilizer subspace

We next address the effect of a measurement on a stabilizer subspace $\mathcal{H}_{\mathcal{S}}$. In particular, we consider the projective measurement of a Pauli observable $Q \in \mathcal{P}_n$ that is a tensor product of single-qubit identity or Pauli operators with no multiplicative factor of -1 or $\pm i$. When the initial state is $|\Psi\rangle \in \mathcal{H}_{\mathcal{S}}$ and the measurement outecome is $\lambda \in \{\pm 1\}$, the post-measurement state is

$$\frac{I+\lambda Q}{\sqrt{2(1+\lambda\langle\Psi|Q|\Psi\rangle)}}|\Psi\rangle\,. \tag{2.4}$$

There are three possibilities:

(i) Either Q or $-Q$ is in $\mathcal{S}$.

(ii) Neither Q nor $-Q$ is in $\mathcal{S}$ and Q commutes with all the elements in $\mathcal{S}$.

(iii) Q anticommutes with at least one element in $\mathcal{S}$.

The measurement may transform the stabilizer group $\mathcal{S}$ and require the redefinition of the logical Pauli operators $\overline{X}_1,\ \cdots,\ \overline{X}_k,\ \overline{Z}_1,\ \cdots,\ \overline{Z}_k$. Let $\mathcal{S}'$ denote the stabilizer group of the projected subspace. For all three cases, $\mathcal{S}'$ contains mQ.

Case (i): The measurement outcome λ is always ± 1 if $\pm Q \in \mathcal{S}$. Both the stabilizer group and the logical Pauli group are invariant under the measurement. This type of measurement is commonly called a **syndrome measurement** since wrong measurement outcomes mean the presence of errors on physical qubits, which is a core idea of **stabilizer quantum error-correcting (QEC) codes**.

Case (ii): This case corresponds to measuring logical qubits in a (possibly multi-qubit) Pauli basis. It is because, due to Proposition 2.3,

$$Q \in \mathcal{P}_n \cap \mathcal{N}(\mathcal{S}) = \left\langle \mathcal{G}_\mathcal{S} \cup \left\{ i,\, \overline{X}_1,\, \cdots,\, \overline{X_k},\, \overline{Z}_1,\, \cdots,\, \overline{Z}_k \right\} \right\rangle,$$

thus we can write $Q = cg \prod_{l=1}^{k} \overline{P}_l$, where $c \in \{\pm 1\}$, $g \in \mathcal{S}$, and $\overline{P}_l$ is either $I^{\otimes n}$, $\overline{X}_l$, $\overline{Z}_l$, or $\overline{Y}_l := i\overline{X}_l\overline{Z}_l$ for each l.

The stabilizer group of the projected space is $\mathcal{S}' = \langle \mathcal{G}_S \cup \{\lambda Q\} \rangle$, where $\mathcal{G}_S$ is a generating set of $\mathcal{S}$ and λ is the measurement outcome. λQ newly becomes a stabilizer since neither Q nor $-Q$ is in $\mathcal{S}$. The original stabilizers in $\mathcal{S}$ remain as stabilizers because

$$(I + \lambda Q)\,|\Psi\rangle = (I + \lambda Q) g\,|\Psi\rangle = g(I + \lambda Q)\,|\Psi\rangle$$

for every $g \in \mathcal{S}$.

The number of stabilizer generators increases by one, thereby the number of logical qubits decreases by one. The new logical Pauli operators can be found in the following way: Since neither Q nor $-Q$ is in $\mathcal{S}$, there exists at least one $l \in \{1,\, \cdots,\, k\}$ such that $\overline{P}_l \neq I^{\otimes n}$. We arbitrarily choose such an $l = l_0$ and define $\overline{P}'$ by either $\overline{X}_{l_0}$ or $\overline{Z}_{l_0}$ that is not $\overline{P}_{l_0}$. We then remove the l_0-th logical qubit and redefine the logical Pauli operators as $\overline{X}'_l,\ \overline{Z}'_l$ for

$l \in \{1, \cdots, k\} \setminus \{l_0\}$ where

$$\overline{X}_l' := \begin{cases} \overline{X}_l & \text{if } \overline{P}_l \in \left\{I^{\otimes n}, \overline{X}_l\right\}, \\ \overline{X}_l \overline{P}' & \text{otherwise,} \end{cases}$$

$$\overline{Z}_l' := \begin{cases} \overline{Z}_l & \text{if } \overline{P}_l \in \left\{I^{\otimes n}, \overline{Z}_l\right\}, \\ \overline{Z}_l \overline{P}' & \text{otherwise.} \end{cases}$$

One can easily check that the above $k-1$ pairs of operators satisfy the conditions for logical Pauli operators presented in Sec. 2.1.2.

Case (iii): We choose a generating set $\mathcal{G}_S$ of $\mathcal{S}$ that contains only one element g_{anti} anticommuting with Q. It is always possible since, if there are multiple stabilizer generators $g_{\text{anti}}, g_1, \cdots, g_l$ that anticommute with Q, we can redefine g_i to $g_i g_{\text{anti}}$ for each $i \in \{1, \cdots, l\}$ to leave only one element (g_{anti}) anticommuting with Q. After the projection, g_{anti} is removed from the generating set and λQ is newly added; namely,

$$\mathcal{S}' = \langle \mathcal{G}_S \cup \{\lambda Q\} \setminus \{g_{\text{anti}}\} \rangle .$$

Note that the two outcomes ($\lambda = \pm 1$) have the same probability regardless of the initial state $|\Psi\rangle \in \mathcal{H}_S$ since

$$\langle \Psi | Q | \Psi \rangle = \langle \Psi | Qg | \Psi \rangle = - \langle \Psi | gQ | \Psi \rangle = - \langle \Psi | Q | \Psi \rangle = 0.$$

Since the number of stabilizer generators does not change, the number

of logical qubits does not change either. For each logical Pauli operator, if it anticommutes with Q, we can make them commute by multiplying g_{anti}. The redefined k pairs of logical Pauli operators satisfy the conditions for logical Pauli operators.

Additionally, it is worth noting that the information of the logical qubits is invariant under the projection; in other words, $\langle\Psi|\overline{P}|\Psi\rangle = \langle\Psi'|\overline{P}|\Psi'\rangle$, where $|\Psi\rangle \in \mathcal{H}_S$ is the pre-measurement state, $|\Psi'\rangle$ is the post-measurement state in Eq. (2.4), and $\overline{P}$ is any logical Pauli operator commuting with Q. It is because

$$\begin{aligned}\left\langle\Psi'\middle|\overline{P}\middle|\Psi'\right\rangle &= \frac{1}{2(1+\lambda\langle\Psi|Q|\Psi\rangle)}\langle\Psi|(I+\lambda Q)^2\overline{P}|\Psi\rangle \\ &= \frac{1}{2}\langle\Psi|(I+\lambda Q)^2\overline{P}|\Psi\rangle \\ &= \langle\Psi|\overline{P}+\lambda\overline{P}Q|\Psi\rangle = \langle\Psi|\overline{P}|\Psi\rangle\,.\end{aligned}$$

The last equality holds since $\overline{P}$ commute with every stabilizer of $|\Psi\rangle$ while Q anticommutes with at least one stabilizer.

Example 2.4. The followings are examples of the above three cases regarding Steane's 7-qubit code (see Examples 2.1 and 2.3).

1. Measurement of each stabilizer generator in Eq. (2.2) always gives the outcome of $+1$ if there are no errors. Any single-qubit Pauli error can be detected by the code with a unique pattern of stabilizer measurements. For example, if a Z error occurs on the fifth qubit, only the two stabilizer generators $g_2 = X_3X_4X_5X_7$ and $g_3 = X_1X_5X_6X_7$ give the measurement outcomes of -1, from which the error is uniquely

identified provided that the error is a single-qubit Pauli one.

2. Suppose that we measure $Q = Z_4Z_5Z_6$ and get the outcome of λ, which corresponds to the case (ii). $g_0 := \lambda Q = \lambda Z_4Z_5Z_6$ is then added to the set of stabilizer generators. Since $g_4g_7 = \lambda Z_L$, the post-measurement stabilizer space, which is one-dimensional, is composed of the eigenstate of Z_L with eigenvalue λ.

3. Suppose that we measure $Q = Z_1$ and get the outcome of λ, which corresponds to the case (iii). Q anticommutes with two stabilizer generators: g_1 and g_3. To make only one of them anticommute with Q, we replace g_3 with $g_3' = g_1g_3 = X_2X_3X_5X_6$. After the measurement, the new stabilizer group is generated by $\{\lambda Z_1,\ g_2,\ g_3',\ g_4,\ g_5,\ g_6\}$. Since X_L anticommutes with Q, we redefine it as $X_L' := X_Lg_1 = X_4X_5X_6$. Z_L commutes with Q, thus it does not need to be redefined. Nevertheless, it can be simplified as $Z_L' := \lambda QZ_L = \lambda Z_2Z_3\cdots Z_7$.

2.2 Measurement-based quantum computing

In this section, we study what is **measurement-based quantum computing** (MBQC) and how it works. We first define **cluster states**, which are basic resource states of MBQC, and investigate their features. After that, we present the basic concept of MBQC and how elementary logic gates can be implemented using a two-dimensional cluster state. We then address the ways to achieve fault-tolerance using **topological** MBQC. See the following references for more comprehensive explanations of MBQC:

- Non-topological MBQC
  - R. Raussendorf and H. J. Briegel, "A one-way quantum computer," Phys. Rev. Lett. **86**, 5188–5191 (2001) [32].
  - R. Raussendorf, D. E. Browne, and H. J. Briegel, "Measurement-based quantum computation on cluster states," Phys. Rev. A **68**, 022312 (2003) [33].
- Topological MBQC
  - R. Raussendorf, J. Harrington, and K. Goyal, "A fault-tolerant one-way quantum computer," Ann. Phys. **321**, 2242–2270 (2006) [34].
  - R. Raussendorf, J. Harrington, and K. Goyal, "Topological fault-tolerance in cluster state quantum computation," New J. Phys. **9**, 199 (2007) [31].
  - R. Raussendorf and J. Harrington, "Fault-tolerant quantum computation with high threshold in two dimensions," Phys. Rev. Lett. **98**, 190504 (2007) [35].
  - A. G. Fowler and K. Goyal, "Topological cluster state quantum computing," Quantum Info. Comput. **9**, 721–738 (2009) [36].

2.2.1 Cluster state

Cluster states, which are also referred to as **graph states**, indicate a particular family of stabilizer states whose structures can be represented as graphs composed of multiple vertices and edges.

Definition 2.2 (**Cluster state**). For a set V of qubits and a given graph $G = (V,\ E)$ on V with an edge set $E \subset V \times V$, the corresponding **cluster state** or **graph state** $|\psi_G\rangle_V$ means a stabilizer state stabilized by

$$\left\langle \left\{ g_v = X_v \prod_{v' \in N(v)} Z_{v'} :\ v \in V \right\} \right\rangle =: \mathcal{S}_G, \tag{2.5}$$

where X_v (Z_v) is the Pauli-X (Z) operator on the qubit $v \in V$ and $N(v) := \{v' :\ (v,\ v') \in E\}$ is the set of vertices adjacent to v in G. Alternatively, $|\psi_G\rangle_V$ can be defined as

$$|\psi_G\rangle_V := \prod_{(v_1,v_2)\in E} U_{\mathrm{CZ}}(v_1,\ v_2) \bigotimes_{v\in V} |+\rangle_v,$$

where $U_{\mathrm{CZ}}(v_1,\ v_2)$ is the controlled-Z (CZ) gate on v_1 and v_2 and $|+\rangle_v$ is the state $(|0\rangle + |1\rangle)/\sqrt{2}$ on v. The CZ gate is the two-qubit gate that transforms the basis states as $U_{\mathrm{CZ}}|00\rangle = |00\rangle$, $U_{\mathrm{CZ}}|01\rangle = |01\rangle$, $U_{\mathrm{CZ}}|10\rangle = |10\rangle$, and $U_{\mathrm{CZ}}|11\rangle = -|11\rangle$.

Note that the Heisenberg picture of the CZ gate is

$$\begin{aligned} U_{\mathrm{CZ}}(X\otimes I)U_{\mathrm{CZ}}^\dagger = X\otimes Z, \quad U_{\mathrm{CZ}}(Z\otimes I)U_{\mathrm{CZ}}^\dagger = Z\otimes I, \\ U_{\mathrm{CZ}}(I\otimes X)U_{\mathrm{CZ}}^\dagger = Z\otimes X, \quad U_{\mathrm{CZ}}(I\otimes Z)U_{\mathrm{CZ}}^\dagger = I\otimes Z. \end{aligned}$$

In most cases, $\{g_v :\ v \in V\}$ is regarded as the standard choice of stabilizer generators for a cluster state. We say that g_v is the stabilizer generator **around** v and v is called the **interior** qubit or vertex of g_v.

Instead of employing the CZ gates to construct a cluster state, one may

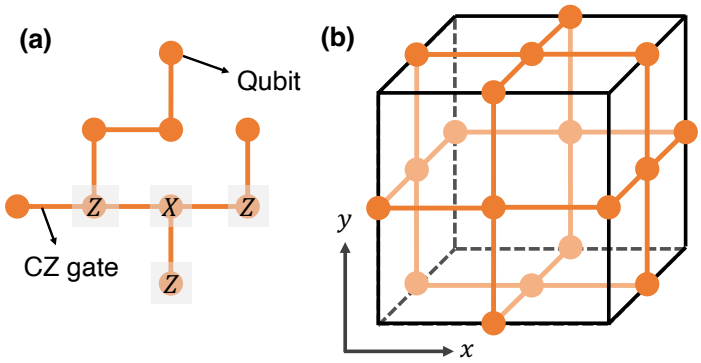

Figure 1: Examples of cluster states. Orange dots and lines indicate the vertices and edges of the graphs, respectively. **(a)** A cluster state on a simple graph. The presented "$XZZZ$" operator indicates an example of a stabilizer generator. **(b)** A unit cell of Rausssendorf's three-dimensional cluster states (RTCSs). A vertex is located on each edge and face of the cell.

take an approach to merge multiple small cluster states into a large one with so-called **fusion** operations [42]. A fusion operation consists of projecting two qubits in a particular entangled basis. By performing a fusion operation on two qubits in different cluster states, these two states can be merged up to several local operations. The initial small cluster states are typically three- or five-qubit states and are regarded as basic resource states for MBQC. We will cover this method in Chapter 4 in more detail.

Two examples of cluster states are visualized in Fig. 1. The cluster state in Fig. 1(b), which is a unit cell of Rausssendorf's three-dimensional cluster states (RTCSs) [34, 31], is particularly important for topological MBQC.

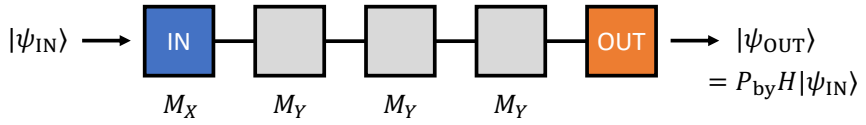

Figure 2: Implementation of the Hadamard gate in two-dimensional MBQC. M_X (M_Y) means the measurement in the X-basis (Y-basis).

2.2.2 How measurement-based quantum computing works

MBQC is a methodology to implement a quantum circuit by performing single-qubit measurements on a cluster state [32, 33]. It is determined by two factors: the structure of the cluster state and the **measurement pattern** (namely, the bases of the single-qubit measurements).

To see how MBQC works, we consider a simple setting shown in Fig. 2 with a five-qubit linear graph. For a given single-qubit input state $|\psi_{\mathrm{IN}}\rangle$, we prepare the state

$$|\Psi\rangle = U_{\mathrm{CZ}}(1,2)U_{\mathrm{CZ}}(2,3)U_{\mathrm{CZ}}(3,4)U_{\mathrm{CZ}}(4,5)\,|\psi_{\mathrm{IN}}\rangle \otimes |+\rangle^{\otimes 4},$$

where $U_{\mathrm{CZ}}(i,j)$ is the CZ gate between the i-th and j-th qubits. If we measure the first qubit in the X-basis and the second to fourth qubits in the Y-basis, the marginal state on the last qubit becomes $|\psi_{\mathrm{OUT}}\rangle = P_{\mathrm{by}} H\,|\psi_{\mathrm{IN}}\rangle$, where H is the Hadamard gate and P_{by}, which is called the **byproduct operator**, is one of I, X, Y, and Z determined by the measurement outcomes. In other words, this process implements the **Hadamard gate up to a Pauli operator**. It can be easily verified by using the basic quantum theory, but here we want to use the stabilizer formalism described in Sec. 2.1. The initial

state before applying the CZ gate is an element of the five-qubit stabilizer subspace stabilized by $\langle X_2,\ X_3,\ X_4,\ X_5\rangle$. The subspace has one logical qubit with the logical Pauli operators of $\overline{X} = X_1$ and $\overline{Z} = Z_1$. After applying the CZ gates, the stabilizer group is transformed into

$$\langle g_2 = Z_1X_2Z_3,\ g_3 = Z_2X_3Z_4,\ g_4 = Z_3X_4Z_5,\ g_5 = Z_4X_5\rangle,$$

and accordingly the logical Pauli operators are transformed as $\overline{X} = X_1Z_2$ and $\overline{Z} = Z_1$. Note that $g_1 = X_1Z_2$ is not a stabilizer unlike the cluster state of the same graph. We now measure X_1, Y_2, Y_3, and Y_4, which anticommute with at least one stabilizer. Since $\overline{X} = X_1Z_2$ anticommutes with Y_2, it should be redefined by multiplying g_3g_4 as $X_1Y_3Y_4Z_5$ that commutes with all the measurements. Similarly, $\overline{Z} = Z_1$ is redefined as $Y_2Y_3X_5$ by multiplying $g_2g_3g_5$. Following the instruction in Chapter 2.1.3, we get the new stabilizer group of $\langle \lambda_1X_1,\ \lambda_2Y_2,\ \lambda_3Y_3,\ \lambda_4Y_4\rangle$, where $\lambda_1,\ \lambda_2,\ \lambda_3,\ \lambda_4$ are the measurement outcomes in order. The logical Pauli operators are invariant but we can equivalently transform them as $\lambda_1\lambda_3\lambda_4Z_5$ and $\lambda_2\lambda_3X_5$ by multiplying stabilizers. To summarize, the logical Pauli operators are transformed as

$$\overline{X} = X_1 \to \lambda_1\lambda_3\lambda_4Z_5,$$
$$\overline{Z} = Z_1 \to \lambda_2\lambda_3X_5,$$

which corresponds to the Heisenberg picture of $P_{\text{by}}H$ where

$$P_{\text{by}} := X^{(1-\lambda_1\lambda_3\lambda_4)/2}Z^{(1-\lambda_2\lambda_3)/2}.$$

Other Clifford gates such as the phase and CNOT gates can be implemented in similar manners [33]. For the CNOT gate, two qubits are allocated for input and output respectively and a two-dimensional (2D) graph is used; because of it, this scheme is commonly called a **2D MBQC scheme**. Moreover, an arbitrary Pauli rotation including the T gate also can be implemented, but measurement in rotated bases selected adaptively by previous measurement outcomes is required [33]. These logic gates complete the universal set of gates, thus the 2D MBQC scheme is **universal**. See Ref. [33] for more details on the 2D MBQC scheme.

The above MBQC scheme is not fault-tolerant. In other words, any single-qubit errors may incur logical errors in the final results. To achieve fault-tolerance, we need to go to three-dimensional (3D) space and employ quantum error-correcting (QEC) codes. Particularly, **topological QEC codes**, which are families of stabilizer QEC codes defined on qubits in a lattice interacting only locally, make **topological MBQC** possible. In such schemes, errors can be corrected from the measurement outcomes of specific operators, called the **parity-check operators**, which are stabilizers that commute with the measurement bases and thus remain as stabilizers even after the measurement. A representative example is the scheme using RTCSs [34, 31, 36] in Fig. 1(b), which are constructed based on surface codes [17, 18, 1]. See Chapter 3 for detailed explanations of how topological MBQC works.

To sum up, MBQC to implement a quantum circuit with a given graph $G = (V, E)$ is generally processed through the following three steps:

1. **Preparation.** A qubit is attached to each vertex in V. V is divided into three subsets: the *input qubits* V_{IN}, the *output qubits* V_{OUT}, and the others. V_{IN} and V_{OUT} can be empty if the desired circuit does not require an input quantum state or produce an output quantum state, respectively. The input states for the circuit are prepared in V_{IN} and all the other qubits are initialized to the $|+\rangle$ state. A CZ gate is then applied on every pair of qubits connected by an edge.

2. **Measurement.** Each physical qubit in $V \setminus V_{\mathrm{OUT}}$ is measured by a basis selected according to the *measurement pattern*. The measurement pattern is determined by the desired circuit. If possible, errors in the outcomes are corrected by decoding the *parity-check* outcomes.

3. **Postprocessing.** The output logical state is obtained from Q_{OUT} up to logical Pauli operators called *byproduct operators* determined by the measurement outcomes. If $Q_{\mathrm{OUT}} = \emptyset$, the results of the final logical measurements are determined by the outcomes.

Note that, although we separate the preparation and measurement steps above, one does not have to complete the preparation step to begin the measurement step; they can be done simultaneously. Provided that a qubit q and its neighbors are prepared and CZ gates are applied on them, q can be measured before the other qubits are prepared. Therefore, we do not need to prepare a large entangled state at once, which is highly demanding for most environments.

Chapter 3

Color-code-based measurement-based quantum computing

The contents of this chapter are largely based on the following paper: Seok-Hyung Lee and Hyunseok Jeong, "Universal hardware-efficient topological measurement-based quantum computation via color-code-based cluster states," Phys. Rev. Research **4**, 013010 (2022) [37].

Measurement-based quantum computing using Rausssendorf's three-dimensional cluster states (RTCSs) has been widely studied since the scheme allows universal and fault-tolerant quantum computing with topologically-encoded logical qubits [34, 35, 31, 36, 2]. Nevertheless, it has a significant drawback: There are no ways to natively implement the topologically-protected logical Hadamard, phase, and T gates, unlike the CNOT gate. Several ways to circumvent this problem have been suggested. The conventional one is to use **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 $|A_L\rangle$ which are distilled from noisy ones [31]. However, this method requires at least seven ancillary logical qubits to implement, thus it can be highly costly. Alternatively, there have been proposals to map lattice surgery [43] onto MBQC models [2, 44]. With their methods, the Hadamard and phase gates can be fault-tolerantly implemented without distillation by "**dislocating**" the RTCS

structure (namely, 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. In other words, the hardware should be designed in a way that can create multiple types of lattice structures.

To solve the above problem, we propose a new 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 for state distillation and lattice dislocations, while keeping most of the advantages of MBQC via RTCSs. In this sense, our scheme is hardware-efficient.

This chapter is structured as follows: In Sec. 3.1, we construct CCCSs and describe their properties. In Sec. 3.2, we show that universal MBQC is possible via CCCSs by defining logical qubits and suggesting the schemes for their initializations and measurements, elementary logic gates, and state injection. In Sec. 3.3, we present the methods to correct physical-level errors. In Sec. 3.4, we calculate the error thresholds of MBQC via CCCSs and compare them with the results for RTCSs. In Sec. 3.5, we analyze and compare the resource overheads of placing logical qubits and implementing each logic gate in the two schemes. We conclude with final remarks in Sec. 3.6.

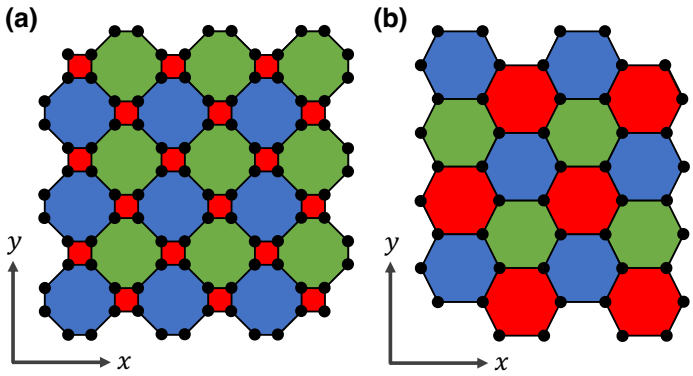

Figure 3: Two typical examples of color-code lattices: **(a)** 4-8-8 and **(b)** 6-6-6 lattices. The lattices are 3-valent and have 3-colorable faces.

3.1 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 [45], we consider a particular family of cluster states derived from 2D color-code lattices, called **color-code-based cluster states (CCCSs)**.

3.1.1 Two-dimensional color-code lattices

To define two-dimensional 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 **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. 3. In the original 2D color codes, a qubit is attached to each vertex and two stabilizer generators

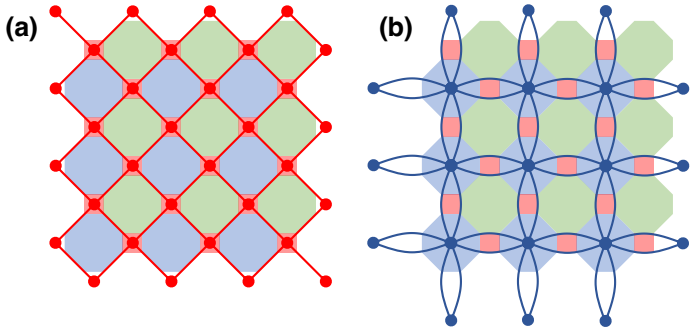

Figure 4: **(a)** Red and **(b)** blue shrunk lattices of the 4-8-8 color-code lattice. Red or blue dots (lines) indicate their vertices (edges), which correspond to red or blue faces (links) of the original lattices.

(X- and Z-type) correspond to each face; see Refs. [23, 15] for details.

Regarding a color-code lattice $\mathcal{L}_{2D}$, we define three **shrunk lattices**, one for each color by shrinking all the faces of that color, as shown in Fig. 4. 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.

3.1.2 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. 5 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

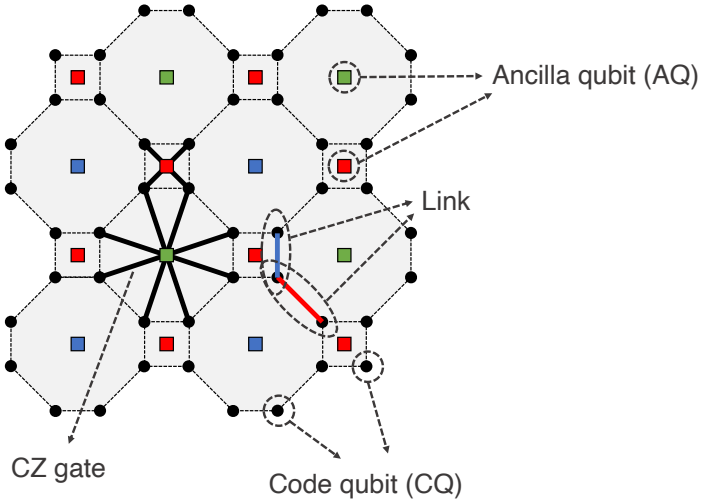

Figure 5: Structure of a single layer of a color-code-based cluster state (CCCS) based on the 4-8-8 color-code lattice $\mathcal{L}_{2D}$. Each black circle is a code qubit (CQ) located at a vertex of $\mathcal{L}_{2D}$. Each colored square is an ancilla qubit (AQ) with that color, located at the center 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.

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. 6. 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

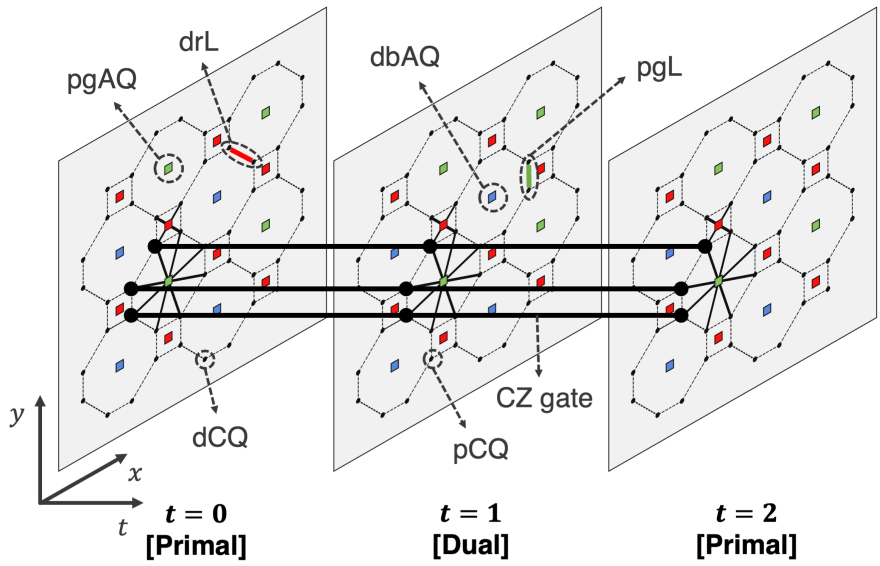

Figure 6: Stack of multiple identical layers along the simulating time axis for a CCCS. Each pair of two CQs adjacent along the time axis is connected by an edge, some of which are presented as black solid lines. One of the primalities ("primal" and "dual") is alternatively assigned to each layer. An 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. 3.1.2 are shown.

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.

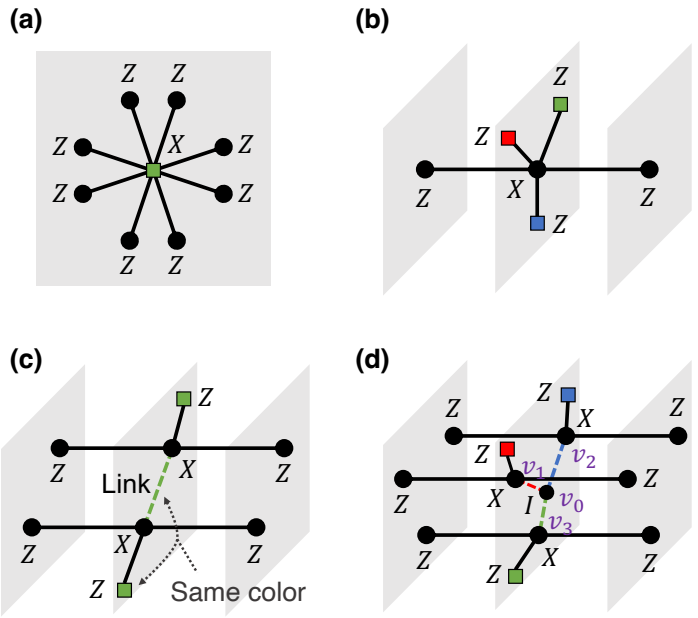

Figure 7: Four types of stabilizer generators in a CCCS defined in Definitions 3.1–3.3: **(a)** A-, **(b)** C-, **(c)** L-, and **(d)** J-type stabilizer generators. Each grey square indicates a layer. A stabilizer generator of each type is the tensor product of the marked X or Z operators on the qubits.

3.1.3 Stabilizer generators

We now present stabilizer generators of a CCCS. Remark that, for each vertex v in G, g_v given in Eq. (2.5) is a stabilizer generator if v (which is called the interior qubit of g_v) is initialized to $|+\rangle$.

We define **A-** and **C-type stabilizer generators** shown in Fig. 7(a) and (b) as follows.

Definition 3.1 (**A- and C-type stabilizer generators**)**.** For a CCCS, the operator $g_v := X_v \prod_{v' \in N(v)} Z_{v'}$ for an AQ (CQ) v is an *A-type (C-type) stabilizer generator* if v is initialized to $|+\rangle$.

The support of a C-type stabilizer generator is distributed in three adjacent layers, while that of an A-type stabilizer generator is contained in a layer.

Although these two types of stabilizer generators completely generate the stabilizer group, we define another two types of stabilizer generators: **L-** and **J-type stabilizer generators** in Fig. 7(c) and (d).

Definition 3.2 (**L-type stabilizer generator**)**.** For a CCCS, the product of two C-type stabilizer generators whose interior qubits constitute a link l is the *L-type stabilizer generator around a link l*.

Definition 3.3 (**J-type stabilizer generator**)**.** For a CCCS, let v_0, v_1, v_2, and v_3 be four CQs initialized to $|+\rangle$ such that (v_0, v_1), (v_0, v_2), and (v_0, v_3) are links with different colors. $S_J := S_{v_1} S_{v_2} S_{v_3}$ is then the *J-type stabilizer generator around the CQ* v_0.

A-, L-, and J-type stabilizer generators together generate the stabilizer group over-completely, if we do not care about qubits near boundaries. To see this, regarding a J-type stabilizer generator S_J, we consider an L-type stabilizer generator $S_{Li} := S_{v_0} S_{v_i}$ for the four CQs v_0, v_1, v_2, and v_3 used when defining S_J. Then $S_{v_0} = S_{L1} S_{L2} S_{L3} S_J$, thus any C-type stabilizer generator that is not very close to boundaries can be written as the product of L- and J-type stabilizer generators. Note also that, for every stabilizer generator regardless of its type, qubits in its X- and Z-support always have different primalities.

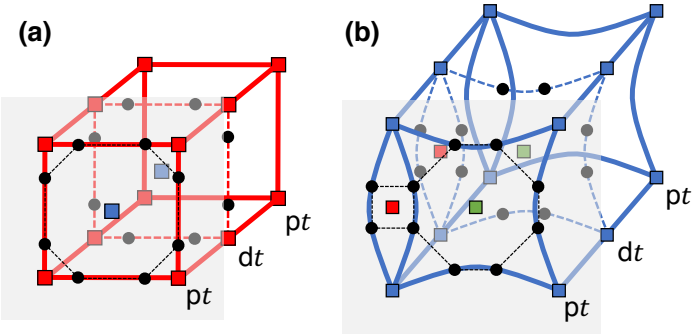

Figure 8: Unit cells of the primal shrunk lattices of a 4-8-8 CCCS: **(a)** a blue cell in the primal red shrunk lattice $\mathcal{L}^{\mathsf{pr}}$ (a green cell is identical except the colors of AQs) and **(b)** red and green cells in the primal blue shrunk lattice $\mathcal{L}^{\mathsf{pb}}$. p$t$s 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 $\mathcal{L}^{\mathsf{pc}}$. 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.

3.1.4 Shrunk lattices and correlation surfaces

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

We now construct the shrunk lattices of a CCCS, which are analogous to those of a 2D color code in Fig. 4. We then define correlation surfaces [34, 35, 31] within each shrunk lattice, through which logic gates are built for MBQC.

The primal c-colored shrunk lattice $\mathcal{L}^{\mathsf{pc}}$ 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. 8 for 4-8-8 CCCSs. Note that each primal layer in $\mathcal{L}^{\mathsf{pc}}$ 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 1. Here $Q(b)$ for an element b denotes the set of qubits corresponding to b. An element is colorable with the color of the AQ or link corresponding to it. In particular, cells and spacelike faces have colors different from the color of the shrunk lattice, e.g., $\mathcal{L}^{\mathsf{pr}}$ is composed of green and blue cells.

We now regard the shrunk lattices as chain complexes [34, 35, 31]. Let $\mathcal{B}_i^{\mathsf{pc}}$ for $i = 0$, 1, 2, or 3 be the set of vertices, edges, faces, or cells in $\mathcal{L}^{\mathsf{pc}}$, respectively. We then consider a vector space H_i^{pc} generated by $\mathcal{B}_i^{\mathsf{pc}}$ over $\mathbb{Z}_2$. Each primal shrunk lattice may be regarded as a chain complex:

Table 1: Qubits $Q(b)$ corresponding to each element (vertex, edge, face, or cell) b in $\mathcal{L}^{\mathsf{pc}}$. The results for $\mathcal{L}^{\mathsf{dc}}$ can be obtained by changing each p or d.

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

$\mathcal{L}^{\mathsf{pc}} = \left\{H_3^{\mathsf{pc}}, H_2^{\mathsf{pc}}, H_1^{\mathsf{pc}}, H_0^{\mathsf{pc}}\right\}$. Each element $h_i \in H_i^{\mathsf{pc}}$ is called an ***i*-chain** and corresponds to a set $B(h_i) \subseteq \mathcal{B}_i^{\mathsf{c}}$ 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 $\mathcal{L}_2^{\mathsf{pc}}$, $h_2 := f_1 + f_2 + f_3$ 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, thus we use h_i and $B(h_i)$ without distinction throughout the chapter for convenience. The chain complex $\mathcal{L}^{\mathsf{pc}}$ has a boundary map ∂ which maps $h_i \in H_i^{\mathsf{pc}}$ to $\partial h_i \in H_{i-1}^{\mathsf{pc}}$ corresponding to the geometrical boundary of h_i. Note that ∂ is a linear map and satisfies $\partial \circ \partial = 0$.

For an i-chain h_i and $P \in \{X, Y, Z\}$, we define a multi-qubit 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)**, which are essential elements for constructing logical operations through MBQC.

Definition 3.4 (Correlation surface). For each 2-chain $h_2 \in H_2^{\mathsf{p(d)c}}$, the operator

$$S_{\mathsf{CS}}(h_2) := X(h_2) Z(\partial h_2). \tag{3.1}$$

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, $S_{\mathsf{CS}}(f)$ is an A- or L-type stabilizer generator around the AQ or link corre-

sponding to f, respectively. The following theorem relates general 2-chains to stabilizers of the CCCS.

Theorem 3.1 (**CSs as stabilizers**). *For a 2-chain h_2, $S_{\mathsf{CS}}(h_2)$ 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. 2.2.2 which are not initialized to $|+\rangle$.*

Proof. **(If)** Since qubits outside Q_{IN} are initialized to the $|+\rangle$ states, there exists the A- or C-type stabilizer generator around each of them. Let $F := \left\{ f \in \mathcal{B}_2^{\mathsf{pc}} \mid Q(f) \cap Q_{\mathrm{IN}} = \emptyset \right\}$. For a face $f \in F$, $S_{\mathsf{CS}}(f)$ is a stabilizer before measuring any qubit in its support; it is an A- or L-type stabilizer generator. For a 2-chain $h_2 \in H_2^{\mathsf{pc}}$ 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 $P(h)P(h') = P(h+h')$ for any Pauli operator P, $S_{\mathsf{CS}}(h_2) = X(h_2)Z(\partial h_2) = \prod_i X(f_i)Z(\partial f_i) = \prod_i S_{\mathsf{CS}}(f_i)$ is a stabilizer before measuring any qubit in its support. The proof is analogous to dual 2-chains.

(Only if) Since qubits in Q_{IN} are not initialized to the $|+\rangle$ states, the A- and C-type stabilizer generators around each of them do not exist. Therefore, the X-support of any stabilizer cannot contain qubits in Q_{IN}. □

Regarding a primal CS $S := S_{\mathsf{CS}}(h_2)$, $Q(h_2)$ is called the **interior** of S (which is the natural extension of the interior of a stabilizer generator). The interior qubits of a primal CS are 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.

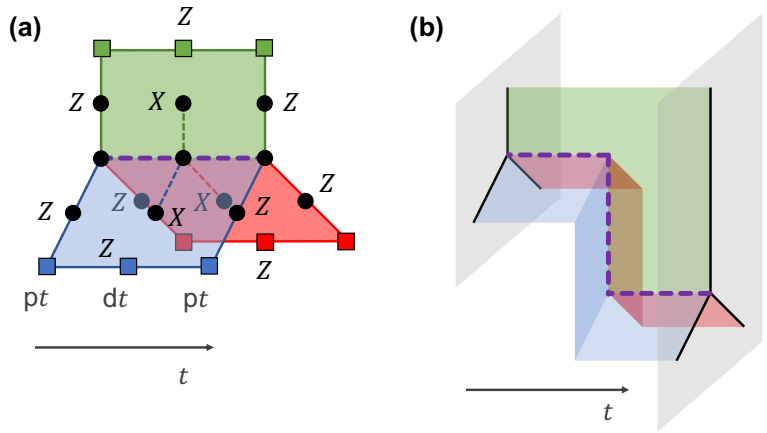

Figure 9: **(a)** Timelike joint of primal correlation surfaces (CSs) originated from a J-type stabilizer generator. The X or Z operators on the qubits indicate the support of the resulting CS. A series of CQs along which the three faces meet is marked as a purple dashed line. **(b)** Example of a general joint, obtained by multiplying a series of timelike and spacelike joints together with ordinary CSs.

CSs discussed above include all A- and L-type stabilizer generators, but not J-type stabilizer generators in Fig. 7(d). Each J-type stabilizer generator can be regarded as three primal timelike CSs with different colors "joined" along a timelike series of CQs as Fig. 9(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. 10. 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 stabilizer generators along a spacelike boundary of the timelike CS. Note that the ends of spacelike and timelike joints may fit perfectly with each

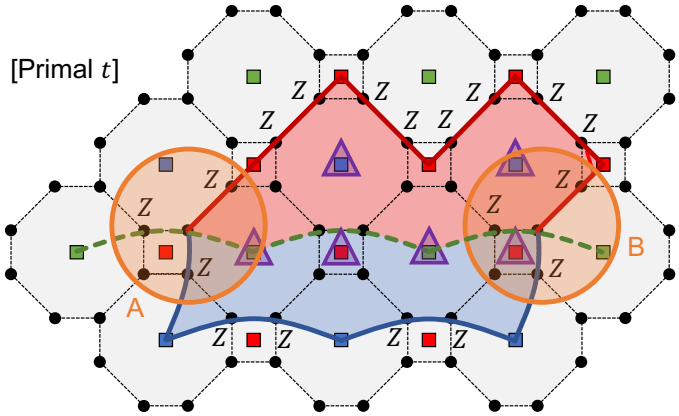

Figure 10: Example of the construction of a spacelike joint of three primal CSs. A primal layer of a 4-8-8 CCCS is presented. We first assume a timelike pg-CS S ending at the green dashed line. We then expand S by multiplying the A-type stabilizer generators around the pAQs marked with purple triangles. 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.

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. 9(b) 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 Figs. 9 and 10.

3.2 Measurement-based quantum computing via color-code-based cluster states

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 logic gates including the identity, CNOT, Hadamard, and phase gates, which together generate the Clifford group. We lastly 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 three-step process presented in Sec. 2.2.2: 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 $t_0 \leq t \leq t_1$. We first consider applying all the CZ gates between qubits of $t_0 \leq t \leq t_1$ again after measuring the qubits of $t > t_0$. Since the measurements commute with these CZ gates, the qubits of $t_0 < t \leq t_1$ simply return to the initial $|+\rangle$ states. The t_0-layer is then the only layer containing nontrivial information and is 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.

3.2.1 Measurement pattern

Remark 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. Lastly, each injection qubit is a special area for state injection and consists of a single qubit. Note that injection qubits are always input qubits.

Qubits in each area are measured as follows:

$$\text{A qubit } q \text{ is measured in the basis of } \begin{cases} X & \text{if } q \text{ is in the vacuum} \\ & \text{or is an injection qubit,} \\ Z & \text{if } q \text{ is in a defect,} \\ Y & \text{if } q \text{ is in a Y-plane.} \end{cases} \tag{3.2}$$

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

3.2.2 Defects and related correlation surfaces

We first define a defect as follows.

Definition 3.5 (**Defect**)**.** Consider a 2-chain $h_2 \in H_2^{\mathsf{d(p)c}}$ in the shape of a "pipe," as shown in Fig. 11(a). A *primal (dual)* c*-colored defect*, referred to as a *"*$\mathsf{p(d)c}$*-D,"* corresponding to h_2 is defined as

$$D(h_2) := \bigcup_{f \in h_2} Q(\partial f), \tag{3.3}$$

which consists of $\mathsf{p(d)cAQ}$s and $\mathsf{p(d)CQ}$s.

We say that a defect is **timelike** or **spacelike** if the "pipe" is extended along the time axis or a spatial axis, respectively. It is also possible that the direction of a defect is changed in the middle. Figure 11(c) and (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

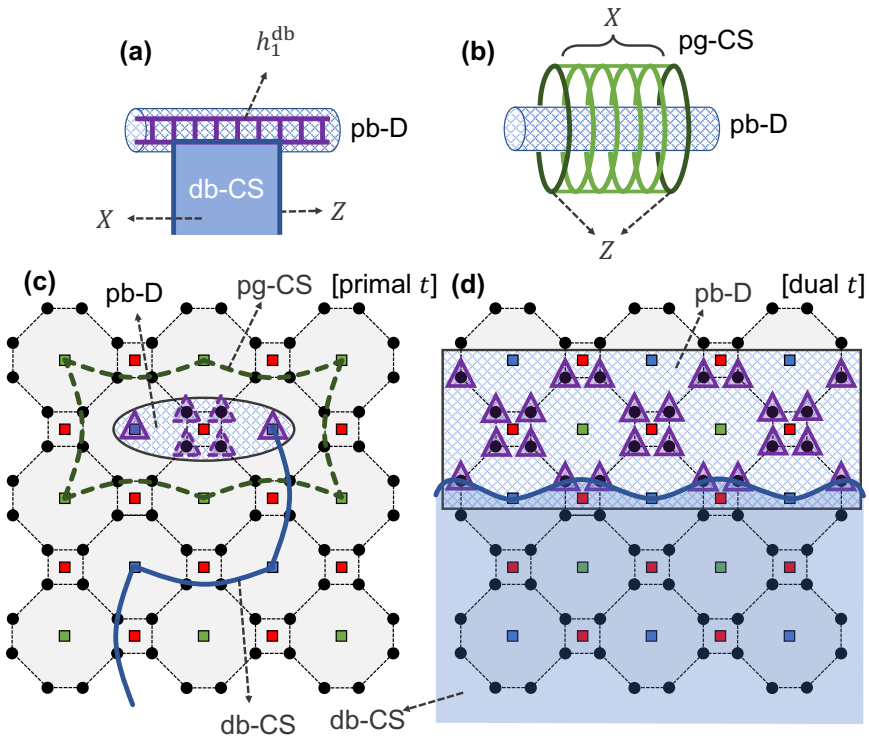

Figure 11: **(a)** Schematic diagram of a defect (pb-D) and a db-CS S ending at the defect. The defect is defined as Eq. (3.3) with a 2-chain $h_2^{\mathrm{db}} \in H_2^{\mathrm{db}}$ in the shape of a pipe. **(b)** Schematic diagram of a pg-CS surrounding a pb-D. **(c)** A primal layer in a 4-8-8 CCCS penetrated by a timelike pb-D $D\left(h_2^{\mathrm{db}}\right)$ for a 2-chain h_2^{db}. The cross-section of h_2^{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^{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.

a defect qubit located at the layer (adjacent layer).

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

Theorem 3.2 (**Compatible CSs**). *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 $\tilde{Q} \setminus Q_{\mathrm{OUT}}$) if and only if the followings hold:*

$$Q_{\mathrm{int}}(S) \cap \widetilde{Q} \setminus Q_{\mathrm{OUT}} \subseteq V \setminus Q_{\mathrm{IN}}, \tag{3.4a}$$

$$Q_{\mathrm{bnd}}(S) \cap \widetilde{Q} \setminus Q_{\mathrm{OUT}} \subseteq D, \tag{3.4b}$$

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

Proof. **(If)** Suppose that Eq. (3.4) holds. Let q be an arbitrary qubit in $\tilde{Q} \setminus Q_{\mathrm{OUT}}$. If $q \in Q_{\mathrm{int}}(S)$, q is in the vacuum, 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 $q \in Q_{\mathrm{bnd}}(S)$, 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 $q \in \tilde{Q} \setminus Q_{\mathrm{OUT}}$. Let q be an arbitrary qubit in $Q_{\mathrm{int}}(S) \cap \tilde{Q} \setminus Q_{\mathrm{OUT}} \subseteq \tilde{Q} \setminus Q_{\mathrm{OUT}}$. q cannot be an injection qubit, since $Q_{\mathrm{int}}(S) \cap Q_{\mathrm{IN}} = \emptyset$ according to Theorem 3.1 and injection qubits are always input qubits. Thus, if $q \notin V$, $M(q)$ is either $Y(q)$ or Z_q, thus

$M(q)$ and S anticommute, which contradicts to the assumption. Therefore, q is in V. Since $Q_{\mathrm{int}}(S) \cap Q_{\mathrm{IN}} = \emptyset$, $Q_{\mathrm{int}}(S) \cap \tilde{Q} \setminus Q_{\mathrm{OUT}} \subseteq V \setminus Q_{\mathrm{IN}}$ holds. $Q_{\mathrm{bnd}}(S) \cap \tilde{Q} \setminus Q_{\mathrm{OUT}} \subseteq D$ 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_{IN} is excluded in the right-hand side (RHS) of Eq. (3.4a) due to Theorem 3.1.

Table 2 shows allowed positional relations between a pc-D d and a compatible CS with each primality and color, derived from Theorem 3.2 and Table 1. Remark that d is composed of pcAQs and pCQs corresponding to edges in $\mathcal{L}^{\mathsf{dc}}$. Let us first check whether a compatible $\mathsf{pc}'-$CS S can be penetrated by d. The interior of S corresponds to faces in $\mathcal{L}^{\mathsf{pc}'}$, thus it consists of pCQs if S is timelike and pc''AQs ($\mathsf{c}'' \neq \mathsf{c}$) if S is spacelike, as shown in Table 1. According to Eq. (3.4a), the interior should not contain defect qubits for S to be compatible. Therefore, S can be penetrated by d only if $\mathsf{c} = \mathsf{c}'$ and S is spacelike (i.e., d is timelike). Additionally, S cannot

Table 2: Allowed positional relations between a primal defect d and a compatible CS. The relations for dual defects are analogous.

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

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 $\mathcal{L}^{\mathsf{dc}'}$. According to Eq. (3.4b), the boundary should contain defect qubits for S to be compatible. Since the defect qubits correspond to edges in $\mathcal{L}^{\mathsf{dc}}$, S can end at d if $\mathsf{c} = \mathsf{c}'$; otherwise, it is impossible in general. (There may be specific cases that it is possible even if $\mathsf{c} \neq \mathsf{c}'$, 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 Fig. 11(a) and (b) and explicitly in Fig. 11(c) and (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.

3.2.3 Defining a logical qubit

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

Definition 3.6 (**Connected 1-chain**). A 1-chain h_1 is *connected* if and only if it satisfies $|\partial h_1| \leq 2$. 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. 12(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 $(\mathsf{c}, \mathsf{c}')$, we

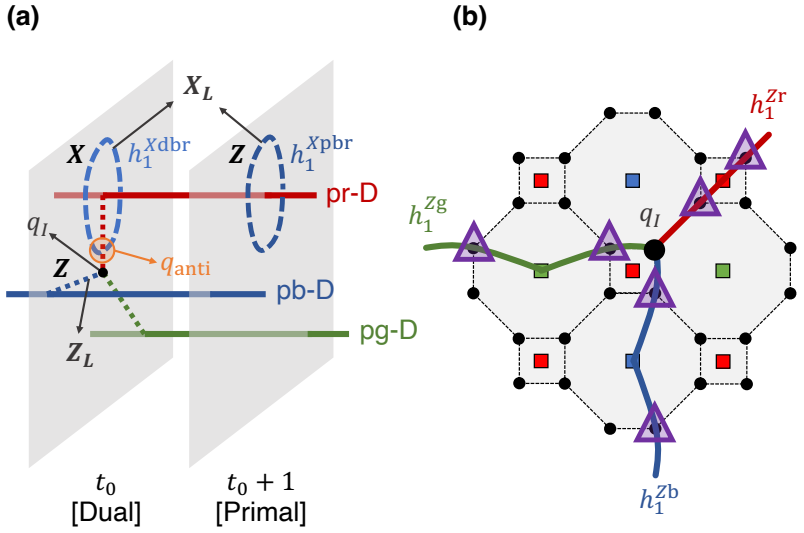

Figure 12: Definition of a primal logical qubit and its initialization and measurement. **(a)** Schematic diagram of a primal logical qubit composed of three parallel primal timelike defects with different colors. Blue dashed lines indicate 1-chains $h_1^{X\mathsf{dbr}}$ and $h_1^{X\mathsf{pbr}}$, which constitute $\mathrm{supp}_X(X_L)$ and $\mathrm{supp}_Z(X_L)$, respectively. Red, green, and blue dotted lines indicate 1-chains $h_1^{Z\mathsf{r}}$, $h_1^{Z\mathsf{g}}$, and $h_1^{Z\mathsf{b}}$, 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 $h_1^{Z\mathsf{r}}$, $h_1^{Z\mathsf{g}}$, and $h_1^{Z\mathsf{b}}$, respectively. Purple triangles indicate $\mathrm{supp}(Z_L)$.

consider two closed connected spacelike 1-chains $h_1^{X\mathsf{dcc'}} \in H_1^{\mathsf{dc}}$ and $h_1^{X\mathsf{pcc'}} \in H_1^{\mathsf{pc}}$: $h_1^{X\mathsf{dcc'}}$ is located in the t_0-layer and surrounding the $\mathsf{pc'}-\mathsf{D}$. $h_1^{X\mathsf{pcc'}}$ is defined as parallelly moving $h_1^{X\mathsf{dcc'}}$ one unit positively along the time axis. An example of X_L is shown in Fig. 12(a) for the case of $(\mathsf{c},\mathsf{c'}) = (\mathsf{b},\mathsf{r})$. Note that the two 1-chains consist of pcLs and dcLs, respectively. We then define

$$X_L := F_X^{\mathsf{cc'}}(t_0) := X\left(h_1^{X\mathsf{dcc'}}\right) Z\left(h_1^{X\mathsf{pcc'}}\right). \tag{3.5}$$

Note that $\mathrm{supp}_Z(X_L) = Q\left(h_1^{X\mathsf{pcc'}}\right)$ may be in the boundary of a $\mathsf{pc}-\mathsf{CS}$ 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. 3.2.5.

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. 12(a) schematically. Note that $h_1^{Z\mathsf{c}}$ is composed of pcLs. We define

$$Z_L := F_Z(t_0) := Z\left(h_1^{Z\mathsf{r}}\right)Z\left(h_1^{Z\mathsf{g}}\right)Z\left(h_1^{Z\mathsf{b}}\right)Z(q_I). \tag{3.6}$$

Note that q_I is out of $\mathrm{supp}\,(Z_L)$. The support of Z_L near q_I is explicitly shown in Fig. 12(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. 9(b).

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. 12(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.

3.2.4 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

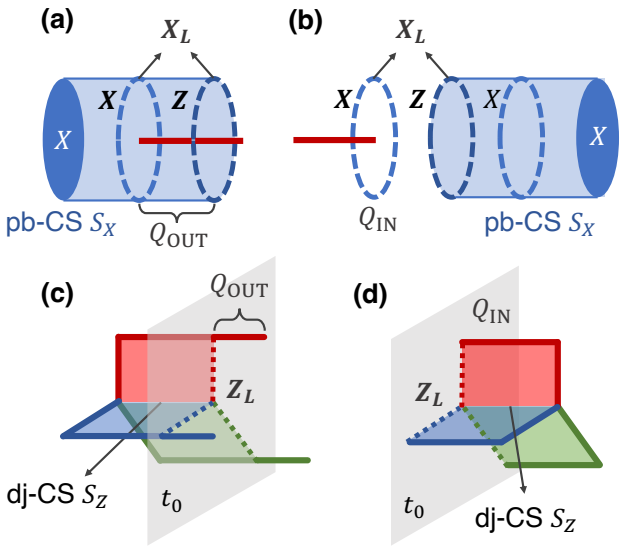

Figure 13: **(a)** X_L- and **(b)** Z_L-initialization. A logical qubit prepared in the output layers Q_{OUT} (t_0- and (t_0+1)-layer). For the X_L-initialization, the defects are made to start from the t_0-layer. For the Z_L-initialization, they are extended to meet at a point before the layer-t_0. 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 $X_L S_X$ ($Z_L S_Z$). **(c)** X_L- and **(d)** 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.

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. (3.5) is then a part of a "cup-shaped" pc-CS S_X as shown in Fig. 13(a). 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 post-measurement state is an eigenstate of $X_L S_X$.

S_X is a stabilizer both before and after the measurement step due to Theorem 3.2. Therefore, the post-measurement 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. 13(b). Z_L given in Eq. (3.6) is then a part of a dj-CS S_Z which is a stabilizer. From an analogous argument, the post-measurement 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 Fig. 13(c) and (d). 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. We redefine X_L as X_L' and 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.

3.2.5 Elementary logic gates

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. 14. 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. (3.5). We consider a pb-CS

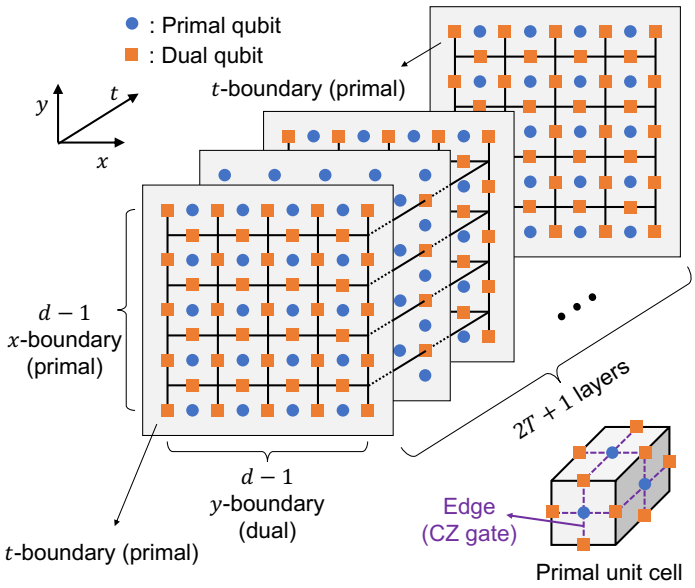

Figure 14: Logical identity gate of a primal logical qubit between the input layer Q_{IN} (t_0-layer) and the output layers Q_{OUT} (t_1- and (t_1+1)-layer). The gate is constructed by extending the defects from Q_{IN} to Q_{OUT}. The logical-X operator in Q_{IN} (Q_{OUT}) is X_L (X'_L), and Z_L and 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. 3.3.1.

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. 14(a). Since S_X is a stabilizer before the measurement step according to Theorem 3.1, X_L is equivalent to

$$\widetilde{X}_L := S_X X_L = \left(\bigotimes_{q\in V_X} X_q\right) X'_L, \tag{3.7}$$

where $V_X := \mathrm{supp}(S_X X_L X'_L) \subset V \setminus Q_{\mathrm{OUT}}$. Since $\tilde{X}_L$ commutes with every measurement basis, it is invariant under the measurement of qubits. [See

case (iii) in Sec. 2.1.3.] However, we can redefine it by multiplying the stabilizer $x_q X_q$ for each $q \in V_X$, where x_q is the measurement outcome of q, as

$$\left(\prod_{q\in V_X} x_q\right) X'_L := x_X X'_L.$$

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 Fig. 14(b). Z_L is then equivalent to

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

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

$$X_L \to x_X X'_L, \qquad Z_L \to x_Z z_Z Z'_L. \tag{3.8}$$

(Throughout this chapter, we use prime symbols to distinguish the output logical operators from the input ones.) Therefore, the input logical state $|\psi_L\rangle$ encoded in $|\psi\rangle$ with the logical Pauli operators $\{X_L, Z_L\}$ is transformed into

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

encoded in $|\psi'\rangle$ with the logical Pauli operators $\{X'_L, Z'_L\}$. This transformation corresponds to the identity gate up to some byproduct operators determined by the measurement results.

The above arguments show the basic ideas for implementing logic gates. Regarding n logical qubits, let P_{Li} for each $P \in \{X, Z\}$ and an integer $i \leq n$ denote the logical-P operator of the ith logical qubit. To construct a general logic 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 2 in that region.

If such CSs exist, the configuration implements the desired logic gate with some byproduct operators obtained from the measurement results.

CNOT and primality-switching gates

We first consider a CNOT gate between a primal logical qubit (target) and a dual one (control). Figure 15 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** [1].

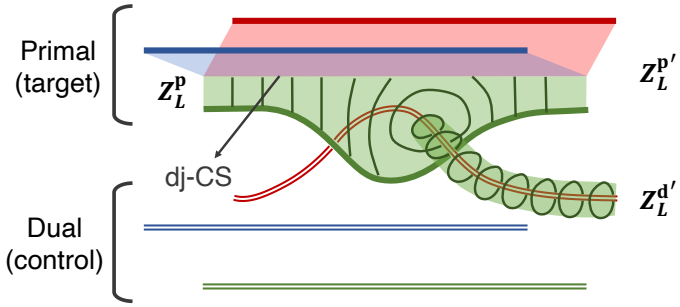

Figure 15: Construction of a CNOT gate between a primal logical qubit (target) and a dual one (control). Each colored single (double) line indicates the primal (dual) defect of the corresponding color. $Z_L^{\mathsf{p}} \otimes I_L^{\mathsf{d}}$ is transformed into $Z_L^{\mathsf{p}'} \otimes Z_L^{\mathsf{d}'}$ via the presented dj-CS.

The logical Pauli operators are transformed as

$$\begin{cases} X_L^{\mathsf{p}} I_L^{\mathsf{d}} & \to X_L^{\mathsf{p}'} I_L^{\mathsf{d}'}, \qquad I_L^{\mathsf{p}} X_L^{\mathsf{d}} \to X_L^{\mathsf{p}'} X_L^{\mathsf{d}'}, \\ Z_L^{\mathsf{p}} I_L^{\mathsf{d}} & \to Z_L^{\mathsf{p}'} Z_L^{\mathsf{d}'}, \qquad I_L^{\mathsf{p}} Z_L^{\mathsf{d}} \to I_L^{\mathsf{p}'} Z_L^{\mathsf{d}'}, \end{cases} \tag{3.9}$$

where the tensor product symbols and the sign terms such as x_X, x_Z, and z_Z in Eq. (3.8) 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. 3.2.5 to verify the transformations in Eq. (3.9). A dual CS for the transformation of $Z_L^{\mathsf{p}} \otimes I_L^{\mathsf{d}}$ is presented schematically in Fig. 15. 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 2). 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.

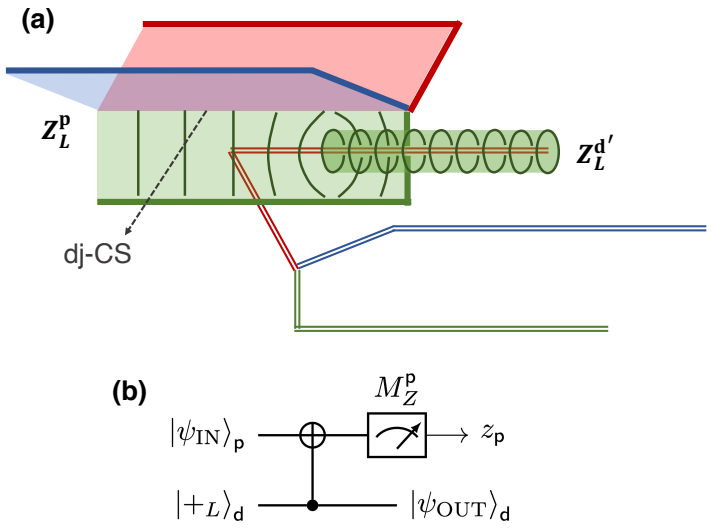

Figure 16: **(a)** Construction of the primality-switching gate changing a primal logical qubit to a dual one. Z_L^{p} is transformed into $Z_L^{\mathsf{d}'}$ via the presented dj-CS. **(b)** Circuit equivalent to the primality-switching gate. M_Z^{p} is the Z_L-measurement on the primal qubit, and the result is z_{p}.

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. 16(a). Remark 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. 16(b) up to byproduct 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. 16(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 the primalities of the input logical qubits, one can switch them to primal (target) or dual (control), and apply a CNOT gate in Fig. 15.

Note that the equivalence between the different definitions of the X_L operator, related to the choice of the color pair $(\mathsf{c}, \mathsf{c}')$ in Eq. (3.5), can be proven with the primality-switching gate. We consider a chain of two primality-switching gates: primal $\rightarrow$ dual $\rightarrow$ 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.

Hadamard gate

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

$$X_L \rightarrow Z'_L, \qquad Z_L \rightarrow X'_L. \tag{3.10}$$

It is simple if the gate is located just after a state injection block presented in the Sec. 3.2.6: 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

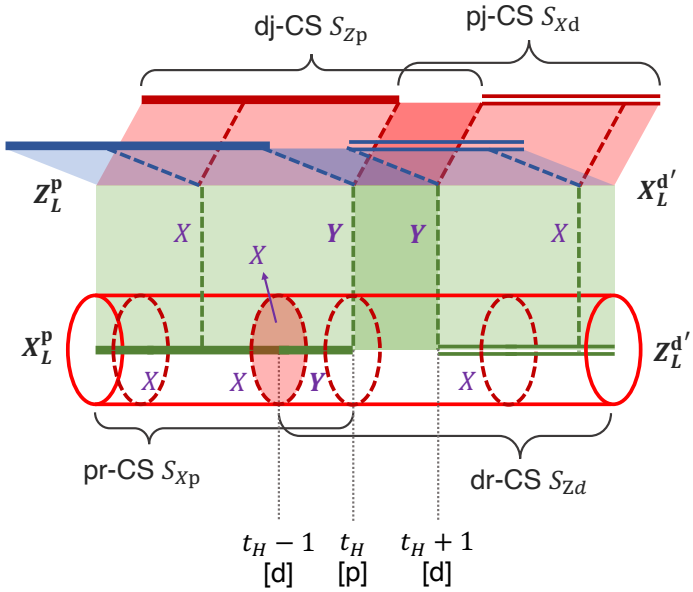

Figure 17: Construction of a Hadamard gate from a primal logical qubit to a dual one. Each colored single (double) line is the primal (dual) defect of that color. $S_{Z\mathrm{p}}$ is a dj-CS ending at the three primal defects and the (t_H+1)-layer. Similarly, $S_{X\mathrm{d}}$ is a pj-CS ending at the 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. (3.10). 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.

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. 17, 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 thanks 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: $(S_{Z\mathsf{p}}, S_{X\mathsf{d}})$ and $(S_{X\mathsf{p}}, S_{Z\mathsf{d}})$, where $S_{X\mathsf{p}}$, $S_{Z\mathsf{p}}$, $S_{X\mathsf{d}}$, and $S_{Z\mathsf{d}}$ are a pr-CS, dj-CS, pj-CS, and dr-CS defined in Fig. 17, respectively. $S_{ZX} := S_{Z\mathsf{p}}S_{X\mathsf{d}}$ then connects Z_L and X'_L. Similarly, $S_{XZ} := S_{X\mathsf{p}}S_{Z\mathsf{d}}$ connects X_L and Z'_L. Condition 1 in Sec. 3.2.5 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 3.7 (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. 3.3.3. 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. 3.3.3.

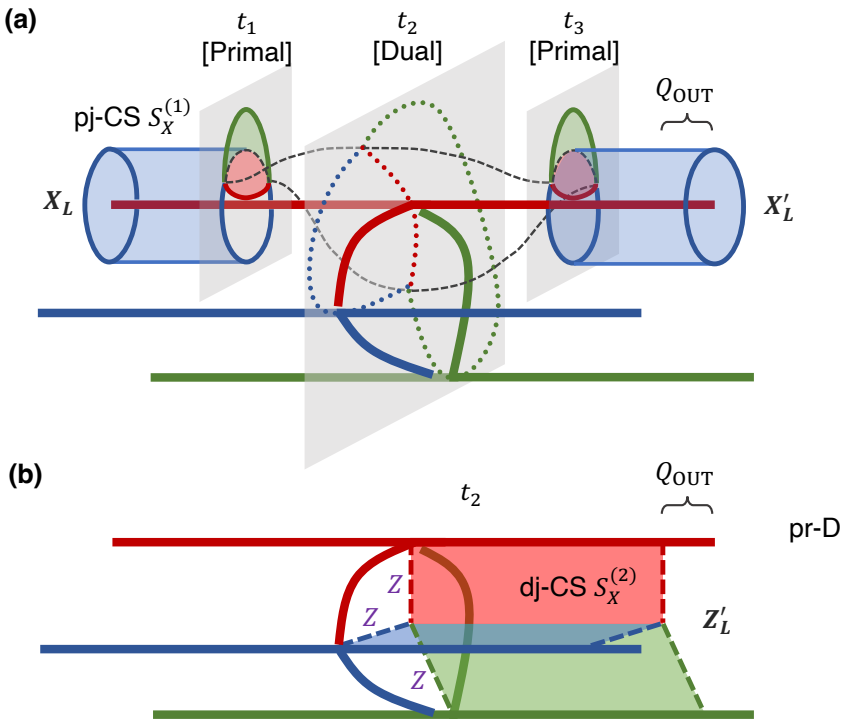

Figure 18: Construction of a logical phase gate on a primal logical qubit. The input logical-X operator (X_L) is transformed into the output logical-Y operator (Y'_L) via a stabilizer $S_X^{(1)}S_X^{(2)}$, where $S_X^{(1)}$ and $S_X^{(2)}$ are CSs shown in **(a)** and **(b)**, respectively. A pj-CS $S_X^{(1)}$ 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 $\mathrm{supp}_X\left(S_X^{(1)}\right)$ 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 spacelikely and $S_X^{(1)}S_X^{(2)}$ has X and Y operators as shown in Fig. 19.

Phase gate

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

$$X_L \to Y'_L, \qquad Z_L \to Z'_L.$$

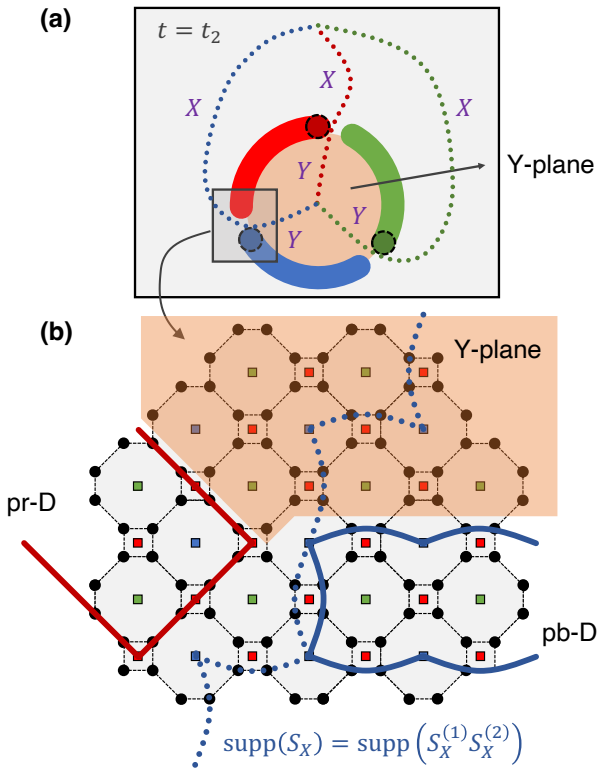

Figure 19: Placement of a Y-plane on the t_2-layer of Fig. 18. In **(a)**, the colored circles indicate the timelike defects penetrating the layer, and the thick colored lines indicate the spacelike defects. By placing a primal Y-plane in the area surrounded by the spacelike defects, Y operators in $S_X^{(1)} S_X^{(2)}$ can be measured. In **(b)**, the vicinity of the timelike pb-D is explicitly described. Here, the colored solid lines indicate the cross-sections of the spacelike defects, along which CQs are measured in the Z-basis.

It can be achieved with the defect structure in Fig. 18(a): The defects of a logical primal qubit are just extended from the input layer to the output layer as the logical identity gate in Fig. 14. The transformation of Z_L is straightforward; it is the same as that in an identity gate in Fig. 14(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. 18(a) and (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. 9 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. 19(a) 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 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 spacelikely to surround the Y-plane, as shown in Fig. 19(a) schematically and in Fig. 19(b) explicitly near where two defects meet. More details on it are presented in Sec. 3.3.3.

3.2.6 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. 13(a), where three defects meet at a point. First, a qubit q_{inj} in the pc-D for any color c is selected as an **injection qubit** which

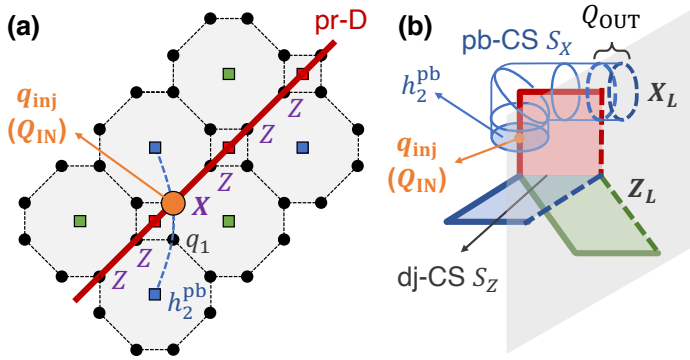

Figure 20: State injection procedure. **(a)** An unencoded state is injected into an injection qubit q_{inj}, which is the only input qubit, in the pr-D which is spacelike and thicknessless at q_{inj}. $Z(q_{\text{inj}})$ is invariant when the CZ gates associated with q_{inj} are applied. However, $X_{q_{\text{inj}}}$ is transformed into $S(q_{\text{inj}})$, where $S(q_{\text{inj}})$ is the C-type SG around q_{inj}. $S(q_{\text{inj}})$ is equivalent to $S_{\text{CS}}\left(h_2^{\text{pb}}\right)$ since $S_{\text{CS}}\left(h_2^{\text{pb}}\right) = S(q_{\text{inj}})S(q_1)$, where $h_2^{\text{pb}} \in H_2^{\text{pb}}$ is the timelike 2-chain marked as a blue dashed line and q_1 is the marked CQ adjacent to q_{inj}. q_{inj} is measured in the X-basis during the measurement step. **(b)** $S_{\text{CS}}\left(h_2^{\text{db}}\right)$ is transformed into X_L of the output logical qubit via the pb-CS S_X. $Z(q_{\text{inj}})$ is transformed into Z_L of the output logical qubit via the dj-CS S_Z.

is the only input qubit in Q_{IN}. We assume that the defect is "thicknessless" at q_{inj}; namely, its cross-section at q_{inj} contains at most one qubit as shown in Fig. 20(a). The desired initial state is injected into q_{inj} in an unencoded form $|\psi\rangle = a|0\rangle + b|1\rangle$, then the associated CZ gates are applied. Remark that q_{inj} is measured in the X-basis as stated in Eq. (3.2). The X (Z) operator on q_{inj} is transformed into X_L (Z_L) up to a sign factor as shown in Fig. 20, thus the logical state $|\psi_L\rangle = a|0_L\rangle + b|1_L\rangle$ is prepared up to byproduct operators.

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.

3.3 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.

3.3.1 Error correction in the vacuum

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

Definition 3.8 (Parity-check operator). For each cell c, the CS $S_{\mathsf{CS}}(\partial c) = X(\partial c)$ is a *parity-check operator (PC)*, where $S_{\mathsf{CS}}(\cdot)$ is given in Eq. (3.1).

PCs are classified into six groups according to primalities and cell colors. Here, the primality of a PC $S_{\mathsf{CS}}(\partial c)$ is that of the shrunk lattice $\mathcal{L}$ containing the cell c, and its cell color is the color of the AQ $Q(c)$. Remark that

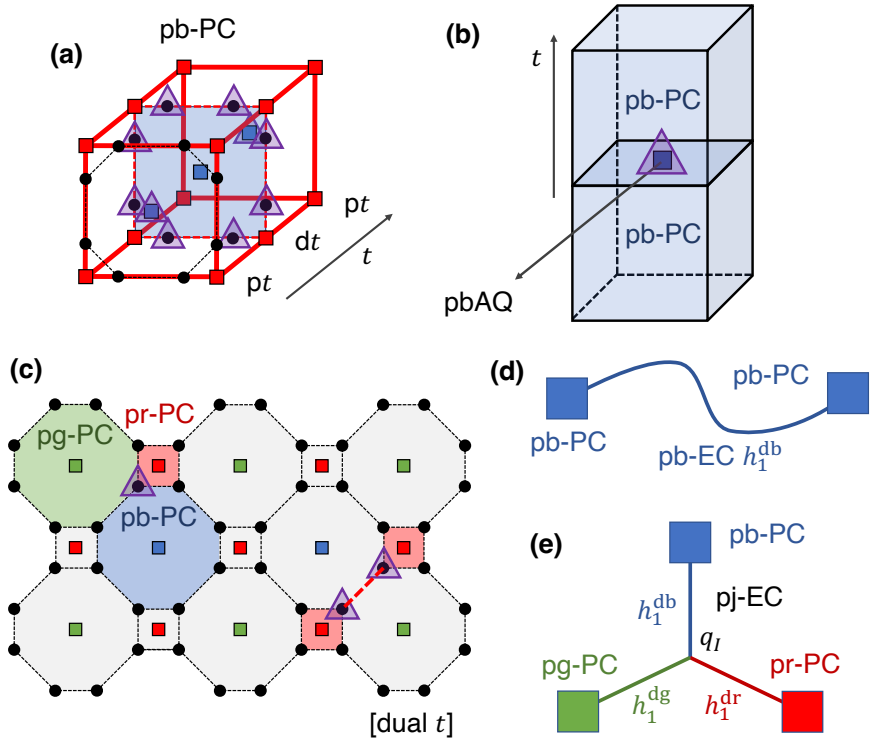

Figure 21: **(a)** Explicit structure of a parity-check operator (PC), specifically a pb-PC in a 4-8-8 CCCS. Purple triangles indicate its X-support qubits. **(b)** A Z or X-measurement error on a pcAQ (purple triangle) flips two pc-PCs sandwiching q. **(c)** A dual layer of a 4-8-8 CCCS is presented. 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.

the cell color is different from the color of $\mathcal{L}$, as shown in Table 1. We refer to a primal c-colored PC as a "pc-PC."

Remark that a given dcAQ q corresponds to two primal cells: one for

each of $\mathcal{L}^{\mathsf{pc}_1}$ and $\mathcal{L}^{\mathsf{pc}_2}$ where c, c_1, and c_2 are all different colors. However, the PCs corresponding to the cells are indeed the same, comparing Fig. 8(a) and (b) as an example. We can thus regard that one AQ (q) corresponds to one PC, and denote it as $S_{\mathsf{PC}}(q)$. The support of the pc-PC $S_{\mathsf{PC}}(q)$ for a dcAQ q contains two pcAQs and multiple pCQs around q as shown in Fig. 21(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. 21(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. 21(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 $Q(h_1^{\mathsf{dc}})$ for a connected dual 1-chain $h_1^{\mathsf{dc}} \in H_1^{\mathsf{dc}}$ has an error, the pc-PC $S_{\mathsf{PC}}(q)$ for each qubit $q \in Q(\partial h_1^{\mathsf{dc}})$ is flipped, as shown in Fig. 21(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 in 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.

3.3.2 Error correction near defects

We here investigate error correction near a pc-D $D_{\mathsf{pc}} = D(h_2)$ for a 2-chain h_2 where $D(\cdot)$ is given in Eq. (3.3). 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 $\mathsf{pc}'-$PCs ($\mathsf{c}' \neq \mathsf{c}$) 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, thus they are just removed. These include $\mathsf{pc}'-$PCs ($\mathsf{c}' \neq \mathsf{c}$) 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). Lastly, there are additional stabilizers that become compatible due to D_{pc}: dual CSs whose Z-supports are in the defect. These stabilizers include dc-CSs $\{S_{\mathsf{CS}}(f) \mid f \in h_2\}$ (e.g., the db-CSs in Fig. 22), and if D_{pc} is spacelike, they also include $\mathsf{dc}'-$CSs on the spacelike surfaces of D_{pc} (e.g., the dg-CS in Fig. 22(b)). Such CSs are called **defect PCs** 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

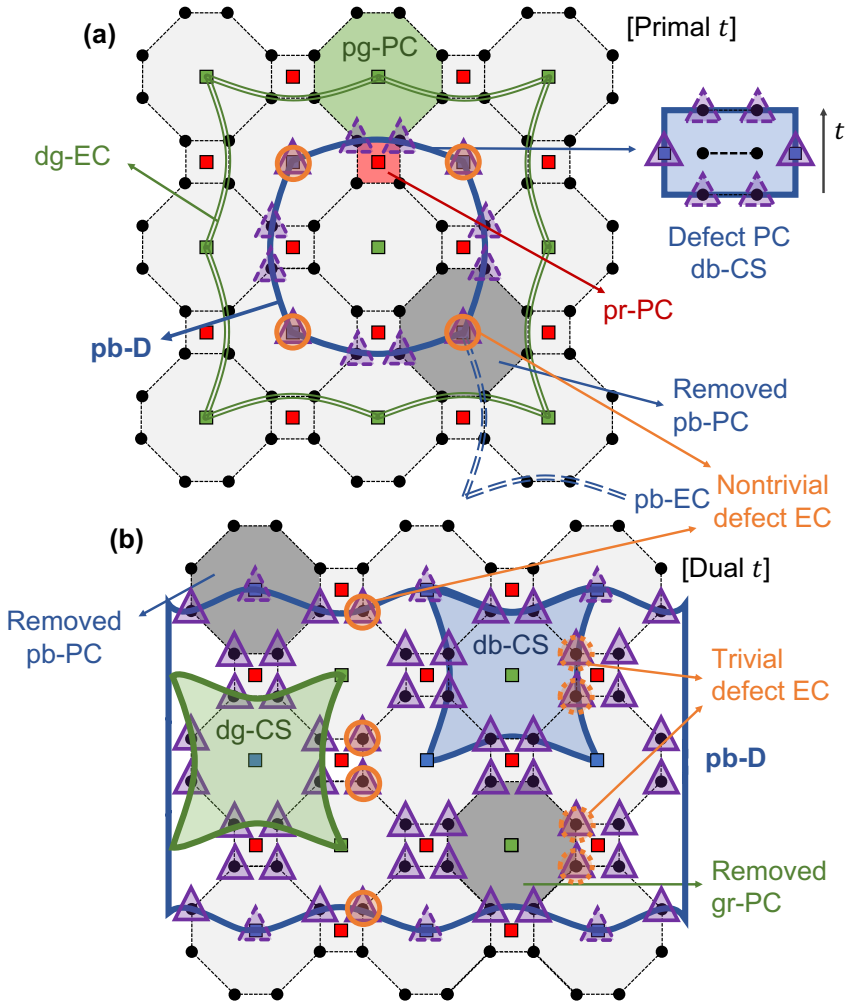

Figure 22: PCs deformed or created due to a **(a)** timelike or **(b)** spacelike pb-D in a 4-8-8 CCCS. Each purple triangle with a solid (or dashed) border indicates a defect qubit on the layer (or an adjacent dual layer). Examples of merged and removed primal PCs and shown as green, red, or grey faces. Examples of dual defect PCs are shown as faces with borders. In **(a)**, two types of nontrivial undetectable error chains are shown as a green double solid line (dg-EC) and a blue double dashed line (pb-EC). Specific nontrivial (trivial) defect error chains are partially shown as orange circles with solid (dotted) borders.

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 $\mathsf{pc}'-\mathsf{EC}$s ($\mathsf{c}' \neq \mathsf{c}$) 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 $\mathsf{pc}'-\mathsf{EC}$s can end at its spacelike surfaces. 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. 14(a).

Next, let us consider errors on dual vacuum qubits. All the dual PCs remain compatible even if a primal defect is placed, 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 $\mathsf{pc}'-\mathsf{D}$ is nontrivial if $\mathsf{c}' \neq \mathsf{c}$ (e.g., the dg-EC in Fig. 22(a)) since it may anticommute with $\mathsf{dc}'-\mathsf{CS}$s 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. 14(b).

Lastly, 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 Figure 22(a) where the defect error chain marked as orange circles may anticommute with db-CSs ending at the defect (see also Fig. 11). Otherwise, the defect error chain shares an even number of qubits with a db-CS, 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 $\mathrm{supp}(e_D)$.

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 is related to error chains ending at different defects, while the latter 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.

3.3.3 Error correction near Y-planes

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

Definition 3.9 (**Hybrid PC**)**.** For each d(p)cAQ q, the stabilizer $S_{\mathrm{PC}}(q)S_{\mathrm{A}}(q)$ is a *primal (dual) c-colored hybrid PC* 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. 23(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

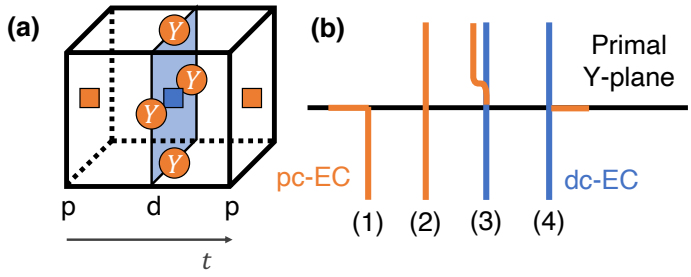

Figure 23: **(a)** Primal hybrid PC for error correction in a primal Y-plane, constructed by multiplying a primal PC and the dual A-type SG around its center qubit. Circles and squares indicate links and AQs, respectively, and their colors mean their primalities: orange (primal) and blue (dual). The 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.

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, 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. 23(b). On the other hand, primal error chains can penetrate a primal Y-plane or progress spacelikely in it without being detected, such as (1) and (2) in Fig. 23(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. 3.2.5, Y-planes are necessary to implement the log-

ical Hadamard and phase gates. We now verify that errors can be corrected well while implementing these gates; namely, local nontrivial undetectable error sets near the Y-planes do not exist. Here, an error set is said to be **local** if its size is unrelated to the distances between the defects or their thicknesses.

Error correction in a logical Hadamard gate

Two consecutive Y-planes are required for a Hadamard gate as shown in Fig. 17: 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. 24(a), each Y-plane completely covers 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 24(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. 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 timelikely can be merged to form a compatible stabilizer which can be used for error correction. Additionally, defect PCs (see Fig. 22) intersecting the Y-planes are also incompatible, thus each pair of them adjacent timelikely 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 subsection, it is assumed that the Y-planes are wide enough, thus their boundaries do

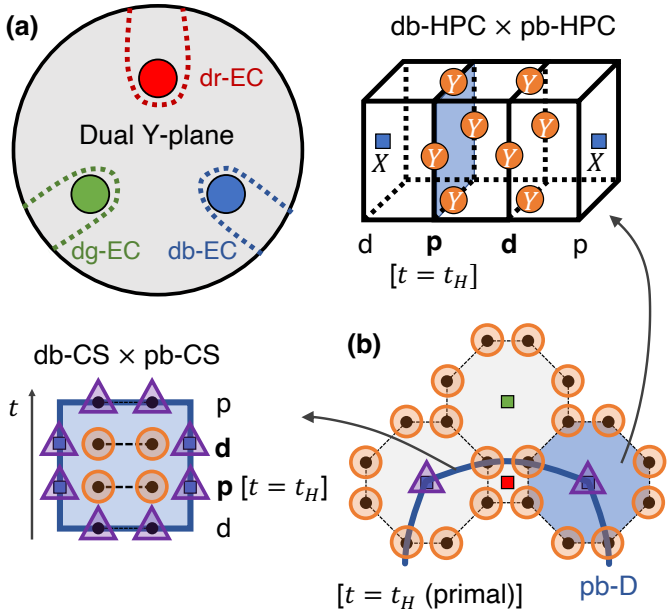

Figure 24: Error correction during the process for a Hadamard gate, particularly near the Y-planes. **(a)** The Y-planes should be wide enough since there are error chains connecting their boundaries and the defects. The dual Y-plane and the primal defects are shown as an example. **(b)** Dual Y-plane 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.

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. 17 directly, we introduce a simpler system $\mathcal{S}_I$ visualized in Fig. 25, where the primal defects are just extended straightly instead of changing to dual defects. Let $\mathcal{S}_H$ denote the original system for a Hadamard gate. In $\mathcal{S}_I$, a dj-CS

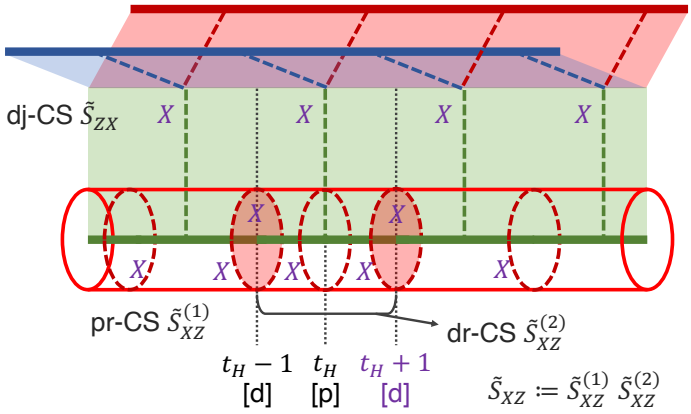

Figure 25: Configuration of the system $\mathcal{S}_I$ introduced to verify error correction in a Hadamard gate, where the primal defects are just extended straightly instead of changing to dual defects.

$\tilde{S}_{ZX}$ ending at the defects and a pr-CS $\tilde{S}_{XZ}^{(1)}$ surrounding the green defect are defined as usual. We additionally define a stabilizer $\tilde{S}_{XZ} := \tilde{S}_{XZ}^{(1)}\tilde{S}_{XZ}^{(2)}$, where $\tilde{S}_{XZ}^{(2)}$ is a closed dr-CS between the $(t_H - 1)$- and $(t_H + 1)$-layer as shown in Fig. 25.

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. 23 and 24), a pair of PCs in $\mathcal{S}_I$ are merged as shown in Fig. 26 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

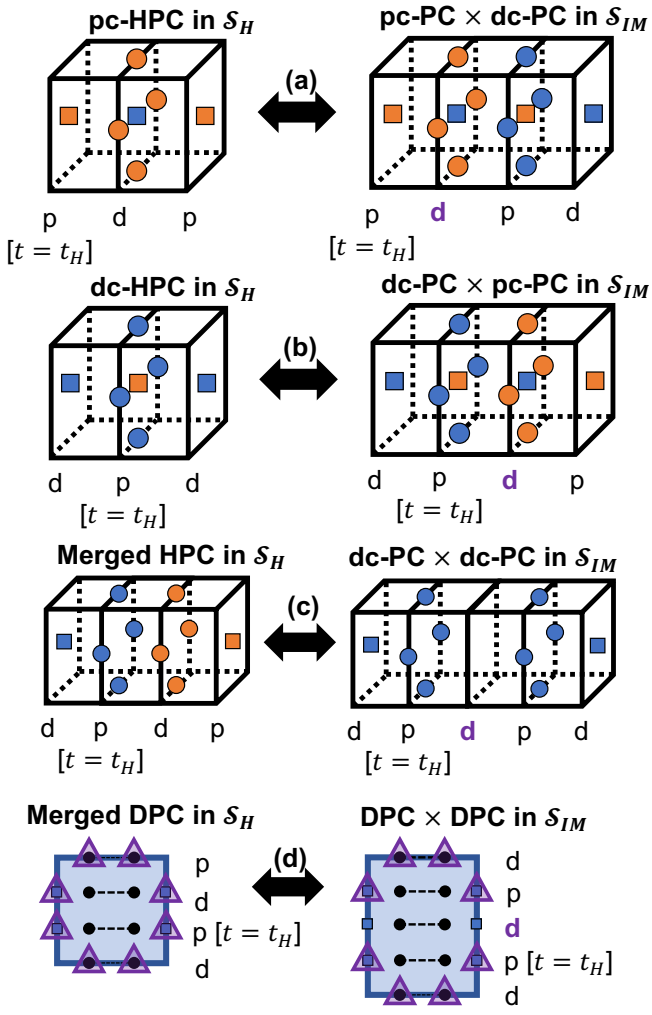

Figure 26: Correspondences of **(a)**, **(b)** hybrid PCs, **(c)** merged hybrid PCs, and **(d)** merged defect PCs in the original system $\mathcal{S}_H$ for a Hadamard gate and merged PCs in $\mathcal{S}_{IM}$, which is called type-1, -2, -3, and -4 merged PCs, respectively. The circles (squares) indicate 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 the PCs have the same support if the $(t_H + 1)$-layer in $\mathcal{S}_{IM}$ is omitted.

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 $\mathcal{S}_I$.

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. (Remark 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 $t \leq t_H$ 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 $\mathrm{supp}\,(\tilde{e}) = \{\tilde{q} \mid q \in \mathrm{supp}(e)\}$ 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. 17 and 25, $\mathrm{supp}\left(\tilde{S}_{ZX}\right) = \{\tilde{q} \mid q \in \mathrm{supp}\,(S_{ZX})\}$ can be checked. Similarly, $\mathrm{supp}\left(\tilde{S}_{XZ}\right)$ contains $\tilde{q}$ for each qubit $q \in \mathrm{supp}\,(S_{XZ})$, 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 $\mathrm{supp}\left(\tilde{S}\right) = \{\tilde{q} \mid q \in \mathrm{supp}(S)\}$ 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. 23 and 24). In such a case, $\tilde{S}$ is set to a merged PC in $\mathcal{S}_{IM}$ shown in

Fig. 26. Likewise, $\mathrm{supp}\left(\tilde{S}\right)$ is composed of $\tilde{q}$ for each qubit $q \in \mathrm{supp}\left(S\right)$ 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 $S_1, S_2, \cdots$ if and only if PCs flipped by $\tilde{e}$ are $\tilde{S}_1, \tilde{S}_2, \cdots$. Hence, e is detectable if and only if $\tilde{e}$ is detectable. Lastly, $\tilde{e}$ does not act on any qubit in the (t_H+1)-layer by definition.

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

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. Remark that $\mathcal{S}_I$ has some additional PCs compared to $\mathcal{S}_{IM}$. Therefore, e may be a local non-trivial **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. 26(c) can be written as $S_{\mathsf{d}}S'_{\mathsf{d}}$ for two dc-PCs S_{d} and S'_{d}, thus e may flip both of them. The same argument holds for defect PCs regarding type-4 merged PCs in Fig. 26(d). However, considering type-1 and -2 merged PCs in Fig. 26(a) and (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 $S_{i\mathsf{d}}$, $S'_{i\mathsf{d}}$ $(i=1,\cdots,n_{\mathsf{p}})$ of the same color for a pair of type-1 and -2 merged PCs, with dual PCs $S_{i\mathsf{d}}$ and $S'_{i\mathsf{d}}$ $(i=n_{\mathsf{p}}+1,\cdots,n_{\mathsf{d}}/2)$ of the same color for a type-3 merged PC, and with defect PCs S_{iD} and S'_{iD} $(i=1,\cdots,n_D)$ for a type-4 merged PC. Here, $S_{i\mathsf{d}}$ and $S'_{i\mathsf{d}}$ are set to act on qubits of $t \le t_H+1$ and $t \ge t_H+1$, 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. 11(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 only can be flipped by e_p. In order for e_p to be a proper error chain,

$$|P_\mathsf{pr}| \equiv |P_\mathsf{pg}| \equiv |P_\mathsf{pb}| \equiv x \pmod 2 \tag{3.11}$$

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 only can be flipped by e_d, thus

$$|P_\mathsf{dr}| \equiv |P_\mathsf{dg}| \equiv |P_\mathsf{db}| \equiv y \pmod 2 \tag{3.12}$$

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, $e_2 := e_1 \prod_i e_\mathsf{d}^{(i)} = e_\mathsf{p} e_{\mathsf{d}2} e_D$ only flips 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 only can be flipped by e_p, $e_{\mathsf{d}2} e_D$ is local and undetectable, thus it is trivial according to the second statement. Remark 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 only acts on vacuum dCQs and defect pAQs. Therefore, $e_1 = e_\mathsf{p} e_\mathsf{d} e_D = e_\mathsf{p} e_{\mathsf{d}2} e_D \prod_i e_d^{(i)}$ also commutes with $\tilde{S}_{ZX}$. We however know that e_1 is nontrivial, 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 $\tilde{S}_{XZ}$,

$$\left\{e_2, \tilde{S}_{XZ}\right\} = 0 \iff |P_{\mathsf{dg},I}| + |P_{\mathsf{db},I}| \equiv 0 \pmod 2 \tag{3.13}$$

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. Remark that merged c-colored hybrid PCs in $\mathcal{S}_H$ are placed inside the c-colored defects (see Fig. 24), 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 $i \in \{n_\mathsf{p}+1, \cdots, n_\mathsf{d}/2\}$ are green, thus

$$\begin{cases} |P_{\mathsf{pr}}| = |P_{\mathsf{dr}}|, & |P_{\mathsf{pb}}| = |P_{\mathsf{db}}|, \\ |P_{\mathsf{pr},I}| = |P_{\mathsf{dr},I}|, & |P_{\mathsf{pb},I}| = |P_{\mathsf{db},I}| \end{cases} \tag{3.14}$$

hold. As a consequence, we get

$$x \equiv |P_{\mathsf{pr}}| = |P_{\mathsf{dr}}| \equiv y \pmod 2, \tag{3.15}$$

considering Eq. (3.11) and Eq. (3.12). For green PCs, we can only say that

$$|P_{\mathsf{pg}}| - |P_{\mathsf{pg},I}| = |P_{\mathsf{dg}}| - |P_{\mathsf{dg},I}|, \tag{3.16}$$

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

Lastly, we try to make e_2 undetectable by multiplying appropriate primal error chains. Remark 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 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

$$\left\{e_3, \tilde{S}_{XZ}\right\} = 0 \iff |P_{\mathsf{dg},I}| + |P_{\mathsf{db},I}| + |P_{\mathsf{pg},I}| + |P_{\mathsf{pb},I}| \equiv 0 \pmod 2, \tag{3.17}$$

which is obtained by considering the above discussion and the proposition in Eq. (3.13). 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. (3.14), the second one comes from Eq. (3.16), the third one comes from Eqs. (3.11) and (3.12), and the last one comes from Eq. (3.15). 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.

Error correction in a logical phase gate

We now investigate error correction in a logical phase gate. As proposed in Sec. 3.2.5, 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. 18(c). Unlike the case of the Hadamard gate, the Y-plane cannot be made to cover a wide enough area, since $\mathrm{supp}(S_X)$ 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. 27 for $\mathsf{c} = \mathsf{b}$ where the orange circles (purple triangles) indicate Y-plane (defect) qubits in the layer. Although $\mathsf{pc}'-$PCs and $\mathsf{pc}'-$HPCs ($\mathsf{c}' \neq \mathsf{c}$) 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. 22. 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

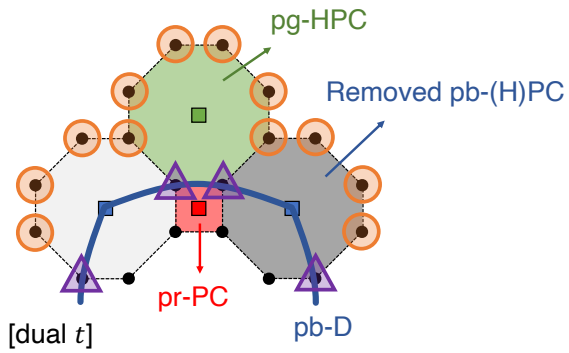

Figure 27: Error correction when the vacuum and a primal Y-plane on a dual layer are separated by a pb-D. Defect (Y-plane) qubits in the layer are marked as purple triangles (orange circles). pr-HPC, pg-HPC, pr-PC, and pg-PCs acting on defect qubits are incompatible, but they can be merged with each 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.

they do not overlap with it.

In summary, each PC near the defect either remains the same or is just replaced with 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 spacelikely to surround the Y-plane entirely as visualized in Fig. 18(c), so that the Y-plane only contacts with the defects. The shape of S_X and the paths of defects should be carefully chosen for the defects not to overlap with $\mathrm{supp}\,(S_X)$. In particular, the microscopic structures near where two defects are closest are important. As visualized in Fig. 28, 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

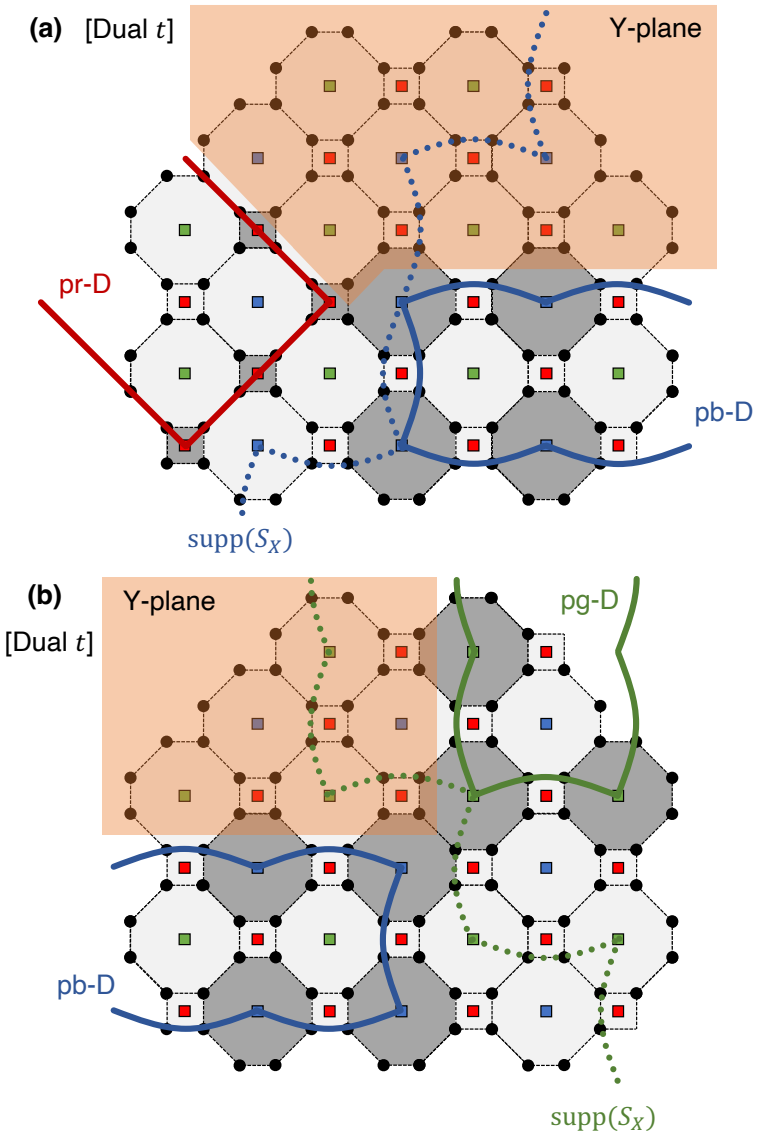

Figure 28: Microscopic structures near where **(a)** red and blue defects or **(b)** blue and green defects are closest for a logical phase gate. The colored solid (dotted) lines indicate the cross-sections of the defects (support of S_X) on the layer; CQs along the lines belong to the defects (support). Gray areas indicate removed PCs due to the defects.

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 Sec. 3.3.2), we only need to consider primal and dual error chains.

Let us consider a local undetectable error chain $e := e_{\mathsf{p}} e_{\mathsf{d}}$ 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 $\mathrm{supp}_Z(S) = e_{\mathsf{d}}$. (If e_{d} is a closed dc(j)-EC, we can find a pc(j)-CS S whose boundary is $\mathrm{supp}(e_{\mathsf{d}})$. 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 $e_{\mathsf{d}} \sim e_{\mathsf{d}} S = X(\mathrm{supp}_X(S)) \sim X(D_S)X(Y_S)$, 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. Remark that a pc-EC e can end at the $\mathsf{pc}'-\mathsf{D}$ d only if $\mathsf{c} = \mathsf{c}'$ or the surface where e meets

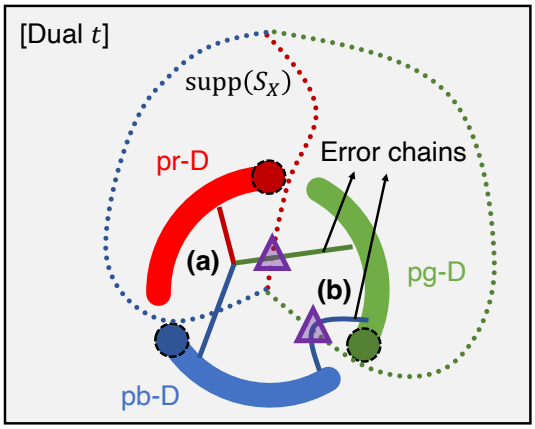

Figure 29: Nontrivial undetectable primal error chains regarding a logical phase gate. The colored circles indicate the timelike parts of the defects and the thick colored lines indicate their spacelike parts. supp(S_X) is presented as colored dotted lines. Each of such error chains can either **(a)** 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.

d is spacelike (see Sec. 3.3.2). Furthermore, since dual CSs S_Z and $S_X^{(2)}$ always commute with primal error chains, we only need to consider $S_X^{(1)}$ whose support in each dual layer is shaped as the dotted lines shown in Fig. 29. Therefore, two types of nontrivial undetectable primal error chains are possible as shown in Fig. 29: 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.)

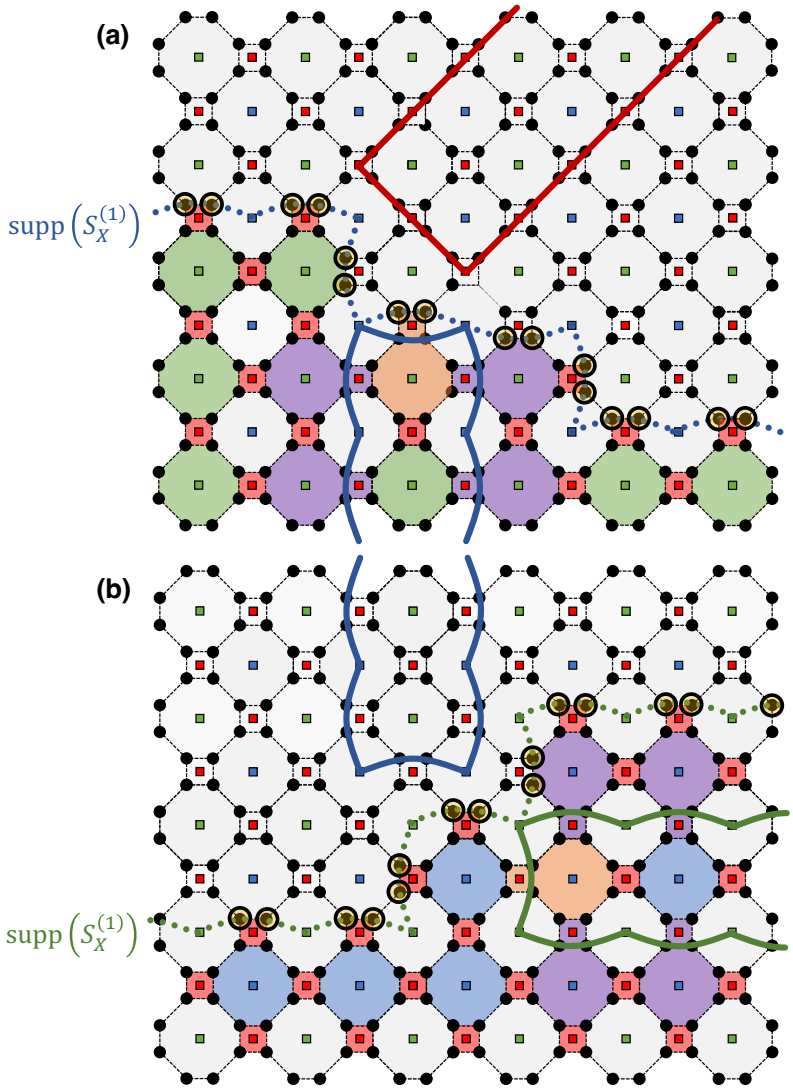

Figure 30: Removal of some primal PCs near where **(a)** red and blue defects or **(b)** green and blue defects are closest to verify that local nontrivial undetectable primal error chains in the area do not exist. Each red, green, or blue area indicates a survived pr-PC, pg-PC, or pb-PC, respectively. Each purple or orange area indicates a survived merged primal PC. Each qubit marked by a black circle is a terminable qubit that belongs to the support of one PC only.

One may wonder whether the above analysis on error chains works well even for such extreme cases where two defects are very close. We can approach the problem differently. 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 defect with respect to $\mathrm{supp}\left(S_X^{(1)}\right)$, as shown in Fig. 30(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 $\mathrm{supp}\,(S_X)$ and called a "terminable qubit." The region near where green and blue defects are closest can be considered analogously as shown in Fig. 30(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 the 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 $q_i^{\mathrm{sng}} \in e$ $(i = 1, 2, \cdots)$ which does not belong to the support of any PC, a single-qubit error on q_i^{sng} is undetectable by itself. Define $e' := e \setminus \left\{q_1^{\mathrm{sng}}, q_2^{\mathrm{sng}}, \cdots\right\}$.

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 $q_2, q_3, \cdots$ until we reach a terminable qubit and get an undetectable error set.

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

4. Pick a qubit $q_0 \in e'' := e' \setminus \left(e_1^{\mathrm{end}} \cup e_2^{\mathrm{end}} \cup \cdots\right)$. 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 $q_2, \cdots, q_i$ and the corresponding PCs $S_2, \cdots, S_i$ 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) $e_1^{\mathrm{cyc}} := \{q_0, \cdots, q_i\}$ is then an undetectable error set.

5. For each $i \geq 2$, repeat step 4 for $e'' \setminus \left(e_1^{\mathrm{cyc}} \cup \cdots \cup e_{i-1}^{\mathrm{cyc}}\right)$ 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: $\left\{q_1^{\mathrm{sng}}\right\}$, $\left\{q_2^{\mathrm{sng}}\right\}$, $\cdots$, e_1^{end}, e_2^{end}, $\cdots$, e_1^{cyc}, e_2^{cyc}, $\cdots$. For each i, e_i^{end} starts from a terminable qubit and ends at another, while $\left\{q_i^{\mathrm{sng}}\right\}$ and e_i^{cyc} only contain non-terminable qubits. Since all qubits in $\mathrm{supp}\,(S_X)$ in the area are terminable, e_i^{cyc} for each i meets $\mathrm{supp}\,(S_X)$ twice, while $\left\{q_i^{\mathrm{sng}}\right\}$ 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.

3.4 Error simulations

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

3.4.1 Error model

We assume a simple error model where vacuum qubits have nontrivial single-qubit errors independently with the same probability p_{phy}. Note that these nontrivial errors contain X-measurement, Y, and Z errors as discussed in Sec. 3.3; X errors on vacuum qubits cannot make logical errors.

3.4.2 Simulation methods

For each simulation with a code distance of d, we simulate the identity gate of a primal logical qubit covering consecutive $2T+1$ layers with $T = 4d+1$ starting from a primal layer. We used simplified 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 only take account 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 31 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

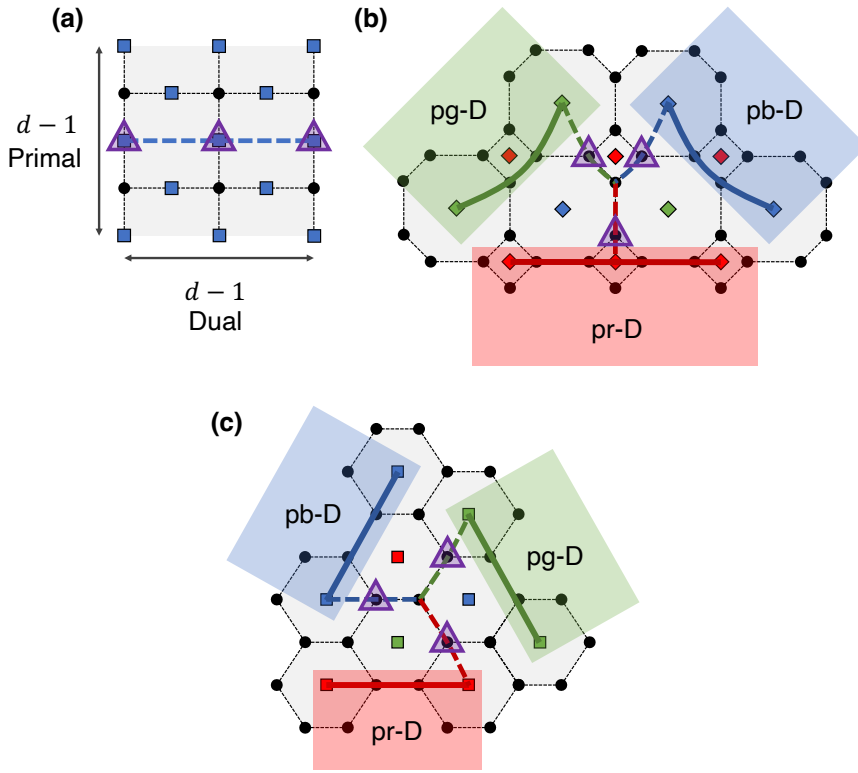

Figure 31: Structure of a layer in the simplified defect model for the simulation regarding **(a)** RTCSs, **(b)** 4-8-8 CCCSs, or **(c)** 6-6-6 CCCSs, particularly when the code distance is $d = 3$. In (a), blue squares (black circles) indicate primal (dual) qubits. In (b) and (c), a colored solid line 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. Purple 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.

connecting the three boundaries incurs a Z_L error.

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_{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 [46, 47, 48] via Blossom V software [49] is used for decoding (once for RTCSs and six times for CCCSs); see the following subsection for details. 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_{thrs} is obtained from the calculated P_{log} results for different values of d and p_{phy}; P_{log} decreases as d increases if $p_{\text{phy}} < p_{\text{thrs}}$ and vice versa if otherwise.

3.4.3 Decoding methods

RTCS: In an RTCS, the PC outcomes are decoded to locate errors at vacuum qubits via Edmonds' minimum-weight perfect matching algorithm (MWPM) [46, 47, 48], as frequently used in the literature [34, 50, 51, 52]. Remark 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 [49] 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.

CCCS: 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 $\mathcal{L}^{\mathsf{pr}}$ 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. Remark 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 lastly note the similarities and differences between our decoding method on CCCSs and the color-code decoders suggested in Refs. [53, 54, 55]. First, they have in common that the MWPM algorithm is first used in the ("shrunk" in our scheme and "restricted" in Refs. [54, 55]) 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.

3.4.4 Results

Figure 32 shows the results of the simulations: Z_L error rates ($P_{\log}$) against nontrivial physical-level error rates (p_{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.

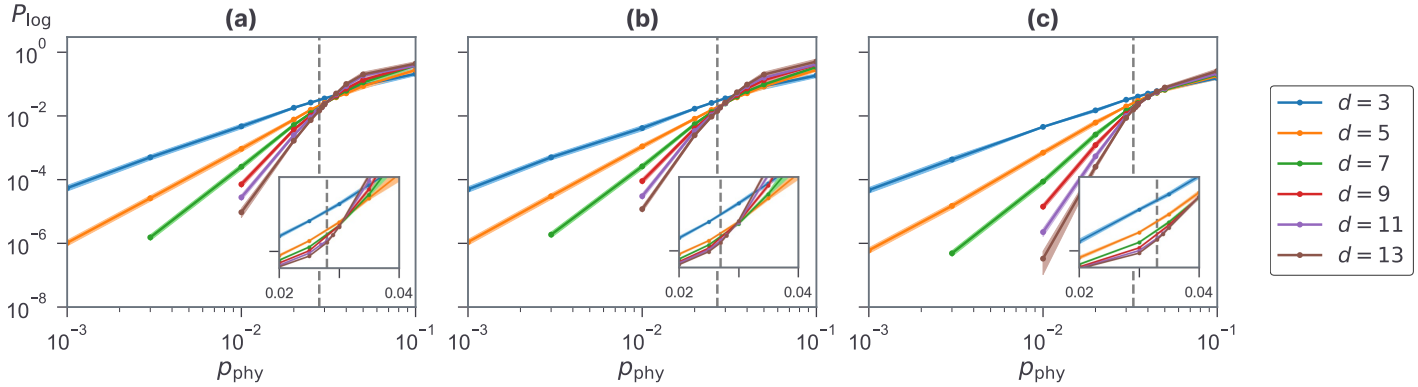

Figure 32: Z_L error rate per two layers $P_{\log}$ versus nontrivial physical-level error rate p_{phy}, for different code distances with respect to **(a)** 4-8-8 CCCSs, **(b)** 6-6-6 CCCSs, and **(c)** RTCSs. The small graphs show the results near the error thresholds. Pale areas around the lines indicate 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 grey dashed lines.

3.5 Resource analysis

3.5.1 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 [34, 35, 31, 36], 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 [2], 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, 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. We first optimize the arrangements while ignoring the implementation of nontrivial logic gates. We then investigate how the arrangements should be changed to make it possible to implement each logic 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 where arbitrary logic gates are applicable.

Note that we here do not consider implementing multiple adjacent gates at the same time. For a more detailed method, see Sec. 3.7.1.

Table 3 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

Table 3: Resource overheads of RTCS and CCCS computation for various sets of implementable logic gates, 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). 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, optimal hexagonal arrangements of parallel timelike primal defects are used. The arrangements are optimized while either ignoring all nontrivial logic gates, considering only one type of logic gate, or considering general gates. Only the leading-order terms on d are calculated.

Implementable logic 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	$29d^2$	$77d^2$

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 logic gates may need to be available for each qubit.

3.5.2 Resource overheads for nontrivial logic 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, remark that those results only show physical qubits per logical qubit in a layer, not the real numbers of physical qubits to implement each logic gate, which is investigated in this subsection. To calculate them, we need to know the number of layers required for each gate; see Secs. 3.7.2 and 3.7.3 for the analysis.

Table 4 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 logic gates are applicable. The numbers of physical qubits per layer are directly obtained from the results of Table 3. 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. 33(a). If $|Y_L\rangle$ has an error and the other parts of the circuit are perfect, the resulting phase gate also has an error. Seven noisy $|Y_L\rangle$ states can be distilled to obtain a less noisy $|Y_L\rangle$ state [31, 1] via the circuit shown in Fig. 33(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 $7\varepsilon^3$ of having an error. The success probability of the distillation process is $1-7\varepsilon$.

The numbers of physical qubits required for the logical phase gates are analyzed in Sec. 3.7.2 for RTCS computation and Sec. 3.7.3 for CCCS computation, and the results together with the residual errors ε_{res} in the distilled $|Y_L\rangle$'s are presented in Table 5. The results clearly show that a phase

Table 4: 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 logic gates are applicable. The results of Table 3 are used to obtain the numbers of physical qubits per layer. Note that, for patch-based RTCS, about $4d$ layers are additionally needed if the two logical qubits are not adjacent.

Type	# per layer	# of layers	Total #
Defect-based RTCS	$13d^2$	$4.5d$	$59d^3$
Patch-based RTCS	$12d^2$	$4d$	$48d^3$
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)	$56d^2$	$4.4d$	$250d^3$

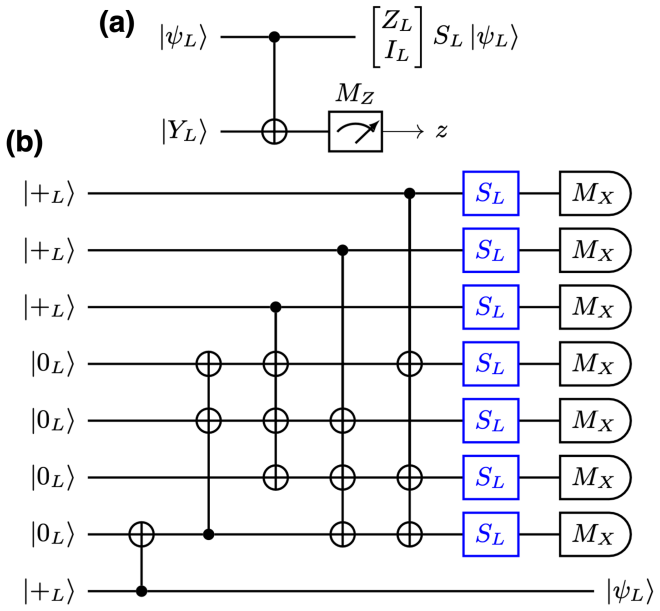

Figure 33: **(a)** Implementation of a logical phase gate S_L with an ancilla logical state $|Y_L\rangle := (|0_L\rangle + i|1_L\rangle)/\sqrt{2}$. $Z_L S_L$ or S_L is applied on the input state if the Z_L-measurement result z is $+1$ or -1, respectively. **(b)** Distillation circuit for a $|Y_L\rangle$ state [1]. 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 $|\psi_L\rangle$.

gate with CCCSs is significantly more resource-efficient (at least about 26 times) than with RTCSs. Note that the non-determinacy 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 $R_X(\pi/2) := \exp\left(i\frac{\pi}{4}X\right)$ is used in RTCS computation to complete a universal set of gates instead of the Hadamard gate [31, 2]. 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. 33(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 5. 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. 17.

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. 34, we illustrate the estimated numbers (n) of physical qubits required

Table 5: 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 is repeated once or twice. For CCCS computation, we assume that the defects are arranged so that all logic 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. ε is the error probability of the initial noisy $|Y_L\rangle$ obtained by state injection.

Type	Number of physical qubits	Residual error
DB RTCS (Distilled once)	$1000d^3$	$\geq 7\varepsilon^3$
DB RTCS (Distilled twice)	$7900d^3$	$\geq 7^4\varepsilon^9$
PB RTCS (Distilled once)	$840d^3$	$\geq 7\varepsilon^3$
PB RTCS (Distilled twice)	$6400d^3$	$\geq 7^4\varepsilon^9$
4-8-8 CCCS	$32d^3$	-
6-6-6 CCCS	$28d^3$	-

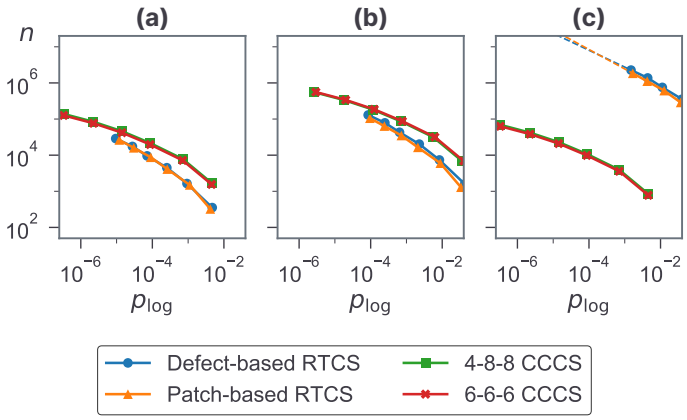

Figure 34: Estimated numbers of physical qubits required for **(a)** an identity gate, **(b)** a CNOT gate, or **(c)** a phase gate versus the logical error rate $P_{\log}$ for CCCS and RTCS computation, while fixing the physical-level error rate p_{phy} to 1%. For **(a)**, it is assumed that the total 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 dashed lines. Note that these results, particularly **(b)** and **(c)**, are rough estimations since we use the results in Sec. 3.4 which cover only Z_L errors in the identity gates.

for each logic gate against achievable logical error rates ($p_{\log}$) while fixing the physical-level error rate (p_{phy}) to 1%, considering the optimal arrangements allowing general logic 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. 3.4 which covers only Z_L errors in the identity gates. More precisely, for each logic 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. 3.4 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 lastly 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 [2, 44], 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.

3.6 Remarks

In this chapter, we have proposed a new **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) [34, 35, 31, 36], in the following three aspects:

1. **Universality:** Initializations and measurements of logical qubits and all the elementary logic 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. 3.2.

2. **Fault-tolerance:** We suggested the error correction scheme for each area of qubits in Sec. 3.3. We further verified in Sec. 3.4 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, thanks 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 [29, 31, 20]. In Sec. 3.5.2, we verified quantitatively that the phase gate in CCCS computation demands significantly fewer 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 [2, 44] 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 a more easy-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 [56] or exploiting redundant ancilla encoding and flag qubits [57]. Moreover, 3D gauge color codes [58, 59, 60, 61, 62, 63,

64, 65] enables non-Clifford gates without distillation. It may be possible to translate these protocols to be applicable to 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.

3.7 Appendix

3.7.1 Methods for analyzing resource overheads of placing logical qubits

We here describe the method to calculate the resource overheads for placing logical qubits, which gives the results in Sec. 3.5.1. 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.

We first define the coordinate systems for the analysis. The x and y axes are presented in Fig. 1(b) for RTCSs and Fig. 3 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. 35(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 35(b) shows the concerned hexagonal arrangement with five parameters $(\alpha,\gamma,\delta,\delta',\varepsilon)$ 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

$$l_{\mathsf{c}}(x,y) := \begin{cases} 2\max(x,y)+O(1) & \text{if } \mathsf{c}=\mathsf{r}, \\ |x|+|y|+O(1) & \text{otherwise.} \end{cases} \tag{3.18}$$

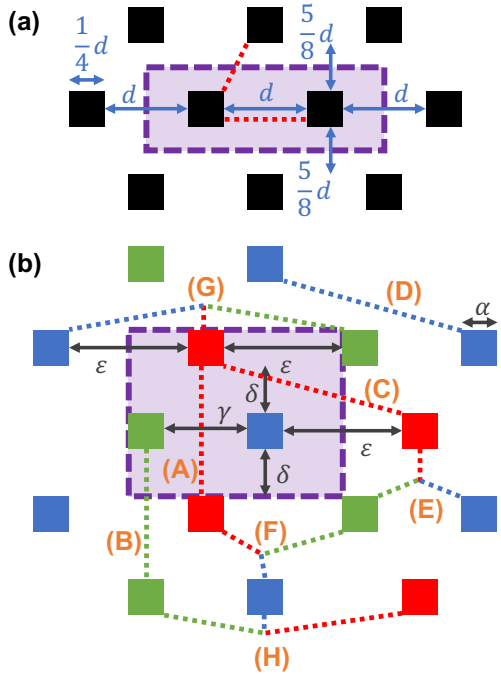

Figure 35: Arrangement of timelike primal defects for calculating the resource overheads of MBQC via **(a)** RTCSs or **(b)** CCCSs. Their projections on a plane perpendicular to the time axis are schematized. Each black, red, green, or blue square is a defect, where its color means the color of the 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 $(\alpha,\gamma,\delta,\delta',\varepsilon) = (\frac{1}{2}d, 0, \frac{1}{2}d, \frac{1}{2}d, \frac{1}{2}d)$ for 4-8-8 and $(\alpha,\gamma,\delta,\delta',\varepsilon) \approx (0.464d, 0.268d, 0.634d, 0.634d, 0.269d)$ 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. 5), and half the distance between two adjacent prAQs in 6-6-6 CCCSs [see Fig. 3(b)].

qubits. Also, the shortest defect error chain in a pc-D connecting $(0,0)$ and (x,y) contains $l_{\mathsf{c}}(x,y)/2$ qubits if the error chain is in a spacelike surface of the defect. If otherwise, it contains $l_{\mathsf{c}}(x,y)$ 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 for defects to 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 the error chain in Fig. 35(b):

$$\begin{cases} \text{(A)} \quad 2(\delta+\delta'+\alpha) \geq d, \\ \text{(B)} \quad \delta+\delta'+\alpha \geq d, \\ \text{(C)} \quad 2\max\left[\frac{\gamma+\alpha}{2}+\varepsilon, \min(\delta,\delta')\right] \geq d, \\ \text{(D)} \quad \frac{\alpha+\gamma}{2}+\varepsilon+\min(\delta,\delta') \geq d, \\ \text{(E)} \quad \gamma+2\min(\delta,\delta') \geq d, \\ \text{(F)} \quad \min(\delta,\delta')+\varepsilon+\frac{1}{2}\max(\gamma-\alpha,0) \geq d, \\ \text{(G)} \quad \alpha+2\varepsilon \geq d, \\ \text{(H)} \quad \alpha+\gamma+\varepsilon \geq d. \end{cases} \tag{3.19}$$

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

$$S \approx \left(\frac{3}{2}\alpha + \frac{\gamma}{2} + \varepsilon\right)\left(\delta + \delta' + 2\alpha\right). \tag{3.20}$$

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

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

$$l(x,y) := \max\left(x + \frac{1}{\sqrt{3}}y, \frac{2}{\sqrt{3}}y\right) + O(1) \tag{3.21}$$

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 $\alpha = \left(2\sqrt{3} - 3\right)d \approx 0.464d$, considering an error chain surrounding

a defect. The following inequalities are derived for each type of error chain:

$$\begin{cases} \text{(A), (B)} & \frac{2}{\sqrt{3}}(\alpha+\delta+\delta') \geq d, \\ \text{(C), (D)} & \max\left[\frac{\alpha+\gamma}{2}+\varepsilon+\frac{1}{\sqrt{3}}\min(\delta,\delta'), \right. \\ & \qquad \left. \frac{2}{\sqrt{3}}\min(\delta,\delta')\right] \geq d, \\ \text{(E)} & \gamma+\frac{2}{\sqrt{3}}\min(\delta,\delta') \geq d, \\ \text{(F)} & \varepsilon+\frac{2}{\sqrt{3}}\min(\delta,\delta') \geq d, \\ \text{(G)} & \alpha+2\varepsilon \geq d, \\ \text{(H)} & \alpha+\varepsilon+\gamma \geq d. \end{cases} \tag{3.22}$$

Minimizing S in Eq. (3.20) subject to the inequalities, we get $S \approx 2.41d^2$ where the corresponding spaces are $(\alpha,\gamma,\delta,\delta',\varepsilon) \approx (0.464, 0.268, 0.634, 0.634, 0.269)d$. We thus obtain $n/k \approx 6.27d^2$ and $N_{\mathrm{CZ}}/k \approx 16.7d^2$.

3.7.2 Methods for analyzing resource overheads of logic gates in RTCS computation

CNOT gate: A logical CNOT gate in patch-based RTCS computation [43, 2] 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 [66] visualized in Fig. 36(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

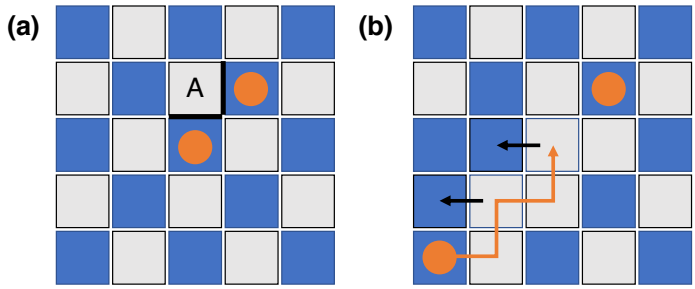

Figure 36: Checkerboard architecture in patch-based RTCS computation. Blue (grey) squares are patches for logical data (ancilla) qubits. **(a)** A CNOT gate between two data qubits (orange circles) is done with two "merge & split" operations [2] (black lines) between data qubits and the ancilla qubit A. The ancilla qubit is prepared just before the operation. **(b)** A CNOT gate between non-adjacent qubits is done by moving a logical qubit appropriately while setting aside qubits in the path.

path for a while as shown in Fig. 36(b). Therefore, we get $n/k \approx 6d^2$ and $N_{\mathrm{CZ}}/k \approx 12d^2$, 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 & split" operation) to keep the code distance at d since a time-like 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 [66], multiple SWAP gates are necessary to move logical qubits, which is very time-consuming; it is one of the advantages of MBQC that logical qubits can be moved quite flexibly.

A logical CNOT gate in defect-based RTCS computation is done by defect braiding [58]; 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 37(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. 37(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. 35(a) is spacious enough to satisfy this condition. Thus, the resource overheads remain the same: $n/k \approx 6.56d^2$ and $N_{\mathrm{CZ}}/k \approx 13.1d^2$.

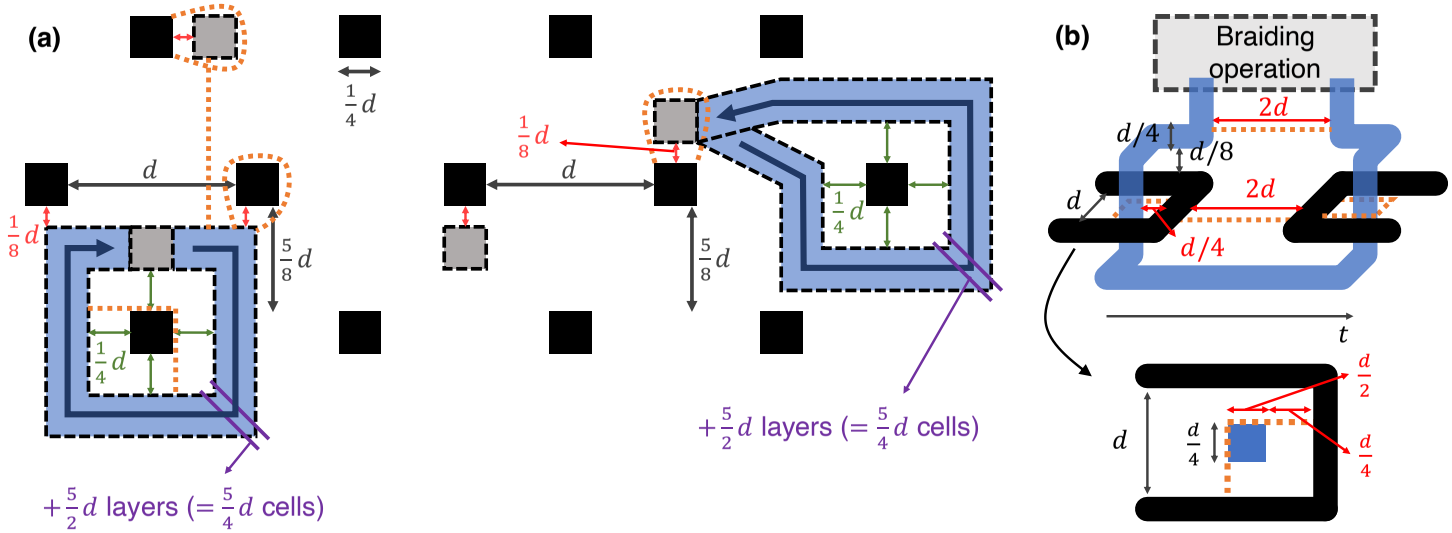

Figure 37: Arrangement of defects for the logical CNOT gate in defect-based RTCS computation. **(a)** The control primal logical qubit is first switched to a dual one (grey squares). Then one of the dual defects proceeds to wrap around a defect of the target qubit. During this process, the defect basically proceeds spacelikely 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.

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. 37(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 non-adjacent 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 spacelikely. 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.

Phase gate with state distillation: We consider using state distillation for the logical phase gate. As shown in Fig. 33(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. 33(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 $7\varepsilon^3$ [31, 1]. 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 [67].)

Using the circuit in Fig. 33(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 $2T_{\mathrm{R}}$ 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. 33(b). The fourth one requires $5T_{\mathrm{R}}$ layers, $4T_{\mathrm{R}}$ 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, 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 $(2+2+2+5+4+4+6+1+7)r_{\mathrm{R}}T_{\mathrm{R}} = 33r_{\mathrm{R}}T_{\mathrm{R}}$, 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 $35r_{\mathrm{R}}T_{\mathrm{R}}$ ($\approx 840d^3$ for patch-based and $\approx 1030d^3$ for defect-based) physical qubits are required since the S_L circuit in Fig. 33(a) additionally occupies $2r_{\mathrm{R}}T_{\mathrm{R}}$ qubits. If a $|Y_L\rangle$ state distilled twice is used, $(33\times 7+33+2)r_{\mathrm{R}}T_{\mathrm{R}} = 266r_{\mathrm{R}}T_{\mathrm{R}}$ ($\approx 6380d^3$ for patch-based and $\approx 7850d^3$ for defect-based) qubits are required.

3.7.3 Methods for analyzing resource overheads of logic gates in CCCS computation

For CCCS computation, we first investigate the optimal arrangement of defects to implement each nontrivial logic gate: the CNOT, Hadamard, or phase gate. Using these results, we obtain an arrangement allowing the

implementation of the universal set of gates. Lastly, we calculate the number of physical qubits required for each gate.

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 Sec. 3.5.1 ($\alpha = \frac{1}{2}d$ 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 38 shows some examples of 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. 16), as shown by purple circles in Fig. 38. 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 conditions in addition to Eq. (3.19) or (3.22):

$$\delta \geq \alpha, \quad \delta' \geq \alpha, \quad \varepsilon \geq \alpha. \tag{3.23}$$

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

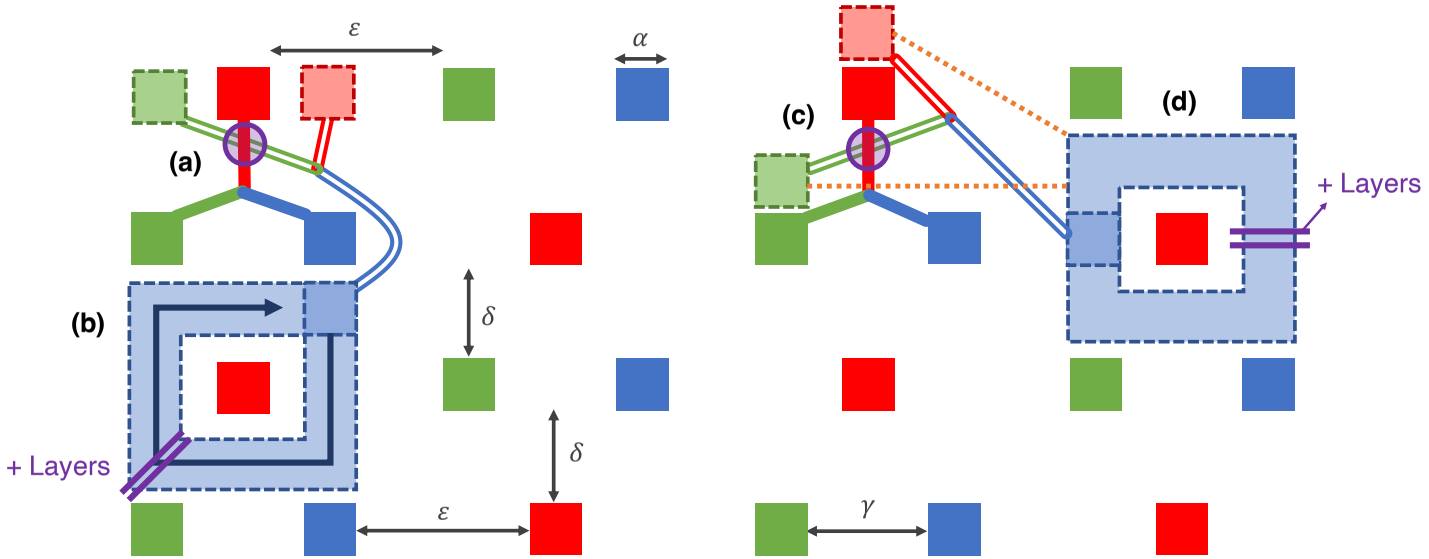

Figure 38: Arrangement of defects for the logical CNOT gate in CCCS computation. The control primal logical qubits are first switched to dual ones (colored squares with dashed boundaries) as shown in **(a)** and **(c)**, then the braiding operations are performed (blue arrows) as shown in **(b)** and **(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 purple circles.

We now count the required number of layers for a CNOT gate. Remark that a timelike error chain contains one qubit per two layers. The calculation is analogous to that for RTCS computation in Fig. 37(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 number of required layers is $T_{\mathrm{CNOT}} = 4t_{\mathrm{depth}} + 2t_{\mathrm{gap}} + 2d$. For both types of lattices, the conditions

$$\begin{aligned} &\frac{1}{2}t_{\mathrm{depth}} + 2\alpha \geq d, \\ &\frac{1}{2}\left(t_{\mathrm{gap}} + t_{\mathrm{depth}}\right) + \alpha \geq d \end{aligned}$$

are sufficient for error chains surrounding each defect (either surrounding it completely or ending at other defects of different primality) to be 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.

Hadamard gate: Remark 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 Sec. 3.3.3. We assume that the Y-planes are square in shape as visualized in Fig. 39 where possible error chains are also presented. For the 4-8-8 lattice, we get

$$\begin{cases} \alpha \geq \frac{d}{2}, \quad \delta' \geq \frac{3}{2}d, \\ \varepsilon - \frac{1}{2}(\alpha + \gamma) \geq 2d, \quad \gamma + 2\delta \geq d, \end{cases} \tag{3.24}$$

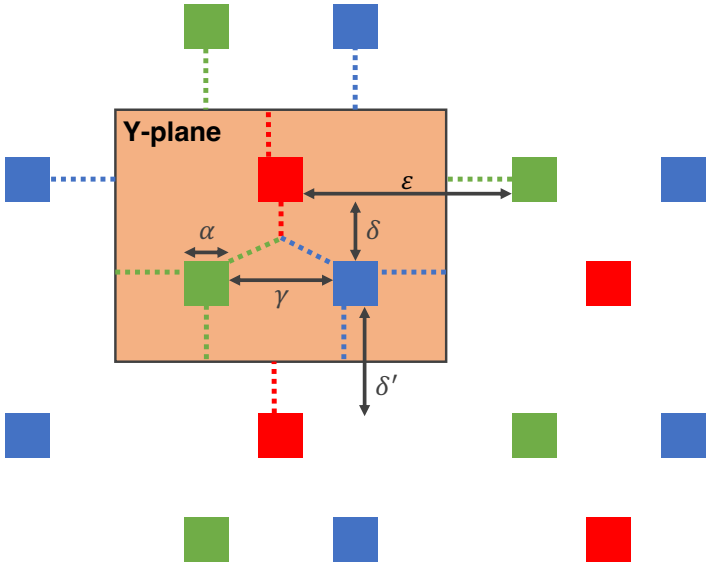

Figure 39: Arrangement of defects for a logical Hadamard gate. The Y-plane (orange square) covering the logical qubit should be wide enough so that error chains (colored dotted lines) connecting its boundary and the defects are longer than the code distance d.

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

$$\begin{cases} \alpha \geq \left(2\sqrt{3}-3\right)d, & \frac{2}{\sqrt{3}}\delta' \geq 2d, \\ \varepsilon - \frac{1}{2}(\alpha+\gamma) \geq 2d, & \gamma + \frac{2}{\sqrt{3}}\delta \geq d. \end{cases} \tag{3.25}$$

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 $(\alpha,\gamma,\delta,\delta',\varepsilon) = \left(\frac{1}{2},0,\frac{1}{2},\frac{3}{2},\frac{9}{4}\right)d$. 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 $(\alpha,\gamma,\delta,\delta',\varepsilon) \approx (0.464,0,0.866,1.73,2.23)d$.

Lastly, a logical Hadamard gate requires only three layers: the (t_H-1)-, t_H-, and (t_H+1)-layer in Fig. 17. Therefore, although the gate demands

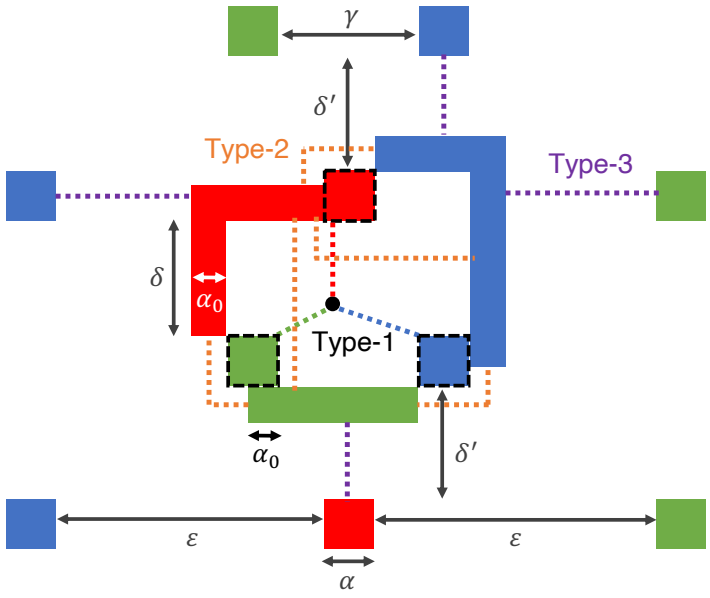

Figure 40: Arrangement of defects for a logical phase gate in CCCS computation. Defects of width α_0 are extended spacelikely to surround the Y-plane. Three types of error chains are considered: Type-1 is for those ending at the three defects in the concerning logical qubit and Type-2 (Type-3) is for those ending at two defects of different colors in the same logical qubit (different logical qubits).

relatively many physical qubits per layer, the total number of required qubits is rather small.

Phase gate: We lastly consider the logical phase gate. Remark that defects are extended spacelikely to surround the Y-plane for a phase gate (see Secs. 3.2.5 and 3.3.3). We assume that these extensions are done as shown in Fig. 40, 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 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 Sec. 3.3.2). Such error chains may cause difficulties because some of them have lengths that only depend on the width of each timelike defect as shown in Fig. 40, 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 (≥ 10000) 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. (3.19):

$$\begin{cases} \text{Type-1 (numerical):} & \gamma \gtrapprox 0.4d, \\ & 0.3\gamma + \delta \gtrapprox 0.45d \\ \text{Type-2:} & 2\alpha \geq d, \\ & \frac{3}{2}\alpha + \frac{1}{2}\gamma \geq d, \\ & \alpha + \delta \geq d, \\ \text{Type-3:} & \delta' \geq d, \\ & \varepsilon - \frac{1}{2}(\alpha + \gamma) \geq d. \end{cases} \tag{3.26}$$

The conditions for Type-1 error chains are valid when $\alpha = \frac{1}{2}d$. By minimizing S in Eq. (3.20) subject to the above conditions, we get $n/k = 18.75d^2$ and $N_{\mathrm{CZ}}/k = 50d^2$ for $(\alpha, \gamma, \delta, \delta', \varepsilon) = \left(\frac{1}{2}, \frac{1}{2}, \frac{1}{2}, 1, \frac{3}{2}\right)d$.

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

$$\begin{cases} \text{Type-1 (numerical):} & \gamma \gtrapprox 0.5, \quad \delta \gtrapprox 0.4 \\ \text{Type-2:} & \left(1 + \frac{2}{\sqrt{3}}\right)\alpha \geq d, \\ & \frac{3}{2}\alpha + \frac{1}{2}\gamma \geq d, \\ & \frac{2}{\sqrt{3}}(\alpha + \delta) \geq d, \\ \text{Type-3:} & \frac{2}{\sqrt{3}}\delta' \geq d, \\ & \varepsilon - \frac{1}{2}(\alpha + \gamma) \geq d. \end{cases} \tag{3.27}$$

The conditions on Type-1 error chains are valid when $\alpha = \left(2\sqrt{3} - 3\right)d$. By

minimizing S, we get $n/k \approx 14.5d^2$ and $N_{\mathrm{CZ}}/k \approx 38.6d^2$ for $(\alpha,\gamma,\delta,\delta',\varepsilon) \approx (0.464, 0.608, 0.402, 0.866, 1.537)d$.

Lastly, 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, $T_{\mathrm{phase}} = d$ holds for both types of lattices.

Optimal arrangement for general logic 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 logic gates are applicable. For the 4-8-8 lattice, considering the conditions in Eqs. (3.19), (3.23), (3.26), and (3.24), we get $n/k = 31.5d^2$ and $N_{\mathrm{CZ}}/k = 84d^2$ for $(\alpha,\gamma,\delta,\delta',\varepsilon) = (\frac{1}{2},\frac{1}{2},\frac{1}{2},\frac{3}{2},\frac{5}{2})d$. For the 6-6-6 lattice, considering the conditions in Eqs. (3.21), (3.23), and (3.27), and (3.25), we get $n/k \approx 28.7d^2$ and $N_{\mathrm{CZ}}/k \approx 76.5d^2$ for $(\alpha,\gamma,\delta,\delta',\varepsilon) = (0.464, 0.608, 0.464, 1.73, 2.54)d$.

Chapter 4

Linear-optical measurement-based quantum computing with parity-encoded multiphoton qubits

The contents of this chapter are largely based on the following manuscript: Seok-Hyung Lee, Srikrishna Omkar, Yong Siah Teo, and Hyunseok Jeong, "Parity-encoding-based quantum computing with Bayesian error tracking," arXiv:2207.06805 [quant-ph] (2022) [38].

Photonic qubits are a promising candidate for quantum computing with advantages such as long decoherence time even at room temperature. Among different encoding schemes, those of dual-rail allow one to detect photon losses by counting the total photon number and manipulate and measure single qubits *via* linear optical elements and photodetectors [68]. Measurement-based quantum computing (MBQC) is a representative way to achieve universal quantum computing in linear optical systems.

The generation of cluster states, which is a significant challenge for realizing fault-tolerant optical MBQC, can be done by entangling multiple small resource states with **fusions of types I and/or II** [42]. Both types of fusions are **nonideal** in linear optics because of theoretical limitations and environmental factors such as photon losses. Fusion success rates cannot exceed 50% without additional resources [69] for single-photon qubits, which

is far too insufficient to implement MBQC [70]. There exist several types of approaches to overcome this shortcoming such as the following examples:

1. Different types of encoding strategies with coherent states [71, 72], hybrid qubits [73, 74], and multiphoton qubits [75, 3] that significantly improve error thresholds and resource overheads [3].

2. Adding ancillary photons to boost the success rate of a type-II fusion to 75% [76, 77], which enables MBQC with the renormalization method [78].

3. Redundant structures added to resource states to replace a single fusion by multiple fusion attempts [79, 80, 81].

4. Use of squeezing for teleportation channels [82] or inline-processes [83, 84].

Previous studies frequently treated fusion failures with bond disconnection [85, 86, 87] or qubit removals [70, 78, 73, 3]. However, to accurately evaluate the performance of computing protocols, the detrimental effects of nonideal fusions affecting nearby qubits should be analyzed more rigorously. In this chapter, we study how nonideal fusions corrupt stabilizers and how errors arising from such corruption can be tracked during the generation of graph states. Using a Bayesian approach and the stabilizer formalism, we can now assign error rates with strong posterior evidence from measurement data on certain qubits in the final lattice, thereby enabling much more realistic error simulations and adaptive decoding of syndromes.

We then propose a linear-optical fault-tolerant MBQC protocol termed

a **parity-encoding-based topological quantum computing** (PTQC), which employs a **Raussendorf's 3D cluster state (RTCS)** constructed using the **parity encoding** [88] and **concatenated Bell-state measurement (CBSM)** [89]. We use the polarization of photons as the degree of freedom to encode quantum information and denote the horizontally (vertically) polarized single-photon state by $|\mathrm{H}\rangle$ ($|\mathrm{V}\rangle$). The protocol requires on-off or single-photon resolving detectors, optical switches, delay lines, and three-photon Greenberger-Horne-Zeilinger (GHZ-3) states that can be generated with linear optics. Here, a single-photon resolving detector discriminates between zero, one, and more than one photon entering the detector. We analyze the loss-tolerance of the protocol while exhaustively tracking the detrimental effects of nonideal fusions. The resource overhead in terms of the number of required GHZ-3 states is also investigated. To minimize it, we introduce a graph-theoretical method for optimizing the process of constructing resource states, which is generalizable for other MBQC schemes. By comparing PTQC with three other known approaches using single-photon qubits with fusions assisted by ancillary photons, using simple repetition codes, and using redundant tree graphs, we show that our protocol is advantageous over these protocols in terms of fault-tolerance, resource overheads, or feasibility of basic elements.

This chapter is structured as follows. In Sec. 4.1, we describe the type-II fusion process, introduce a Bayesian methodology for tracking errors caused by nonideal fusions, and present a method to construct an RTCS through fusions. In Sec. 4.2, we propose our new PTQC protocol and analyze its performance. In Sec. 4.3, we depict the modified concatenated

Bell-state measurement scheme used for the PTQC protocol. In Sec. 4.4, we compare the PTQC protocol with three other known approaches and show in what aspects PTQC is advantageous over them. We finish this chapter with final remarks in Sec. 4.5.

4.1 Constructing Raussendorf's 3D cluster states through fusions

4.1.1 Type-II fusion

We denote the four Bell states by

$$\left|\phi^{\pm}\right\rangle := |0\rangle|0\rangle \pm |1\rangle|1\rangle,$$
$$\left|\psi^{\pm}\right\rangle := |0\rangle|1\rangle \pm |1\rangle|0\rangle$$

(normalization coefficients are omitted) and call "$\pm$" its **sign** and "ϕ" or "ψ" its **letter**. An ideal Bell-state measurement (BSM) entails the measurements of $X\otimes X$ and $Z\otimes Z$ on two qubits, whose outcomes are addressed as its **sign and letter outcomes**, respectively.

Since the direct implementation of a controlled-Z gate for photonic qubits demands multi-photon interaction, linear optical MBQC typically takes an approach to construct a graph state by merging multiple small resource graph states via fusion operations [42, 90, 79, 80, 85, 81, 86, 78, 87, 73, 3]. Among the two types of fusions [42], we only consider type II since type I may convert photon losses into unheralded errors [81]. A type-II fusion is done by measuring $X\otimes Z$ and $Z\otimes X$ on two qubits. In prac-

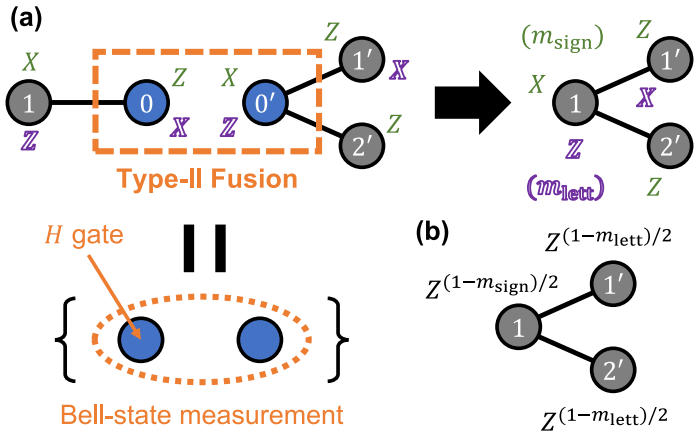

Figure 41: Example of a type-II fusion. A type-II fusion is done by measuring $Z_0X_{0'}$ and $X_0Z_{0'}$ on the two graph states. In **(a)**, two stabilizers (green and purple operators) become those of the resulting graph state up to sign factors (the sign or letter outcome m_{sign}, m_{lett} of the BSM) after the fusion. The final state is the graph state shown in **(b)**, where the presented Pauli-Z operators are applied.

tice, it is realized by applying the Hadamard gate on one of the qubits and then performing a BSM on them. For two qubits (v_1, v_2), if $\{v_1\} \cup N(v_1)$ and $\{v_2\} \cup N(v_2)$ are disjoint, the effect of a fusion on the qubits is to connect (disconnect) every possible pair of disconnected (connected) qubits, one from $N(v_1)$ and the other from $N(v_2)$, up to several Pauli-Z operators determined by the BSM outcome. These Pauli-Z operators are compensated by updating the Pauli frame [91] classically. This effect can be checked by tracking stabilizers, as shown in the example of Fig. 41(a). Here, the stabilizer $X_1Z_0X_{0'}Z_{1'}Z_{2'}$ (colored in green) before the fusion is transformed into $m_{\text{sign}}X_1Z_{1'}Z_{2'}$ after the fusion, where $m_{\text{sign}} \in \{\pm 1\}$ is the sign outcome of the BSM if the Hadamard gate is applied on qubit 0. The other two stabilizers

$Z_1X_0Z_{0'}X_{1'}$ (colored in purple) and $Z_1X_0Z_{0'}X_{2'}$ that commute with the fusion can be transformed in similar ways. Consequently, the marginal state on the unmeasured qubits is equal to the merged graph state up to several Pauli-Z operators, as presented in Fig. 41(b).

4.1.2 Bayesian error tracking for nonideal fusions

We consider errors of qubits in the "vacuum" measured in the X-basis, which occupies most of the area in the RTCS [34]; thus, X-errors do not affect the results. Henceforth, every error mentioned is a Z-error.

We first verify that the marginal state of the qubits participating in a fusion is maximally mixed. More strictly, we prove the proposition:

Proposition 4.1 (**Maximally mixed marginal states in a cluster state**). *For a cluster state $|\psi_G\rangle_V$ with a graph $G=(V,E)$ and given two vertices $a,b\in V$, if $\{a\}\cup N(a)$ and $\{b\}\cup N(b)$ are disjoint and neither $N(a)$ nor $N(b)$ is empty where $N(v)$ for a vertex $v\in V$ is the set of vertices adjacent to v, the marginal state $\mathrm{Tr}_{V\setminus\{a,b\}}|G\rangle\langle G|_V =: \rho_{ab}$ is maximally mixed.*

Proof.. Let $\mathcal{S}$ denote the stabilizer group of the zero-dimensional Hilbert space $\{|G\rangle_V\}$. First, any stabilizer $g\in\mathcal{S}$ can be written as the product of stabilizer generators: $g=\prod_{v\in V_0}g_v$ where $V_0\subseteq V$ and $g_v := X_v\prod_{v'\in N(v)}Z_{v'}$. If V_0 contains a vertex $c\neq a,b$, S must contain X_c or Y_c since no stabilizer generators besides g_c contain X_c. If otherwise, V_0 is one of $\emptyset$, $\{a\}$, $\{b\}$, and $\{a,b\}$. Except when V_0 is empty (namely, g is identity), there exists a vertex $c\neq a,b$ such that g contains Z_c, since $N(a)$ and $N(b)$ are not empty, $b\notin N(a)$, $a\notin N(b)$, and $N(a)\neq N(b)$. Therefore, every single- or two-qubit

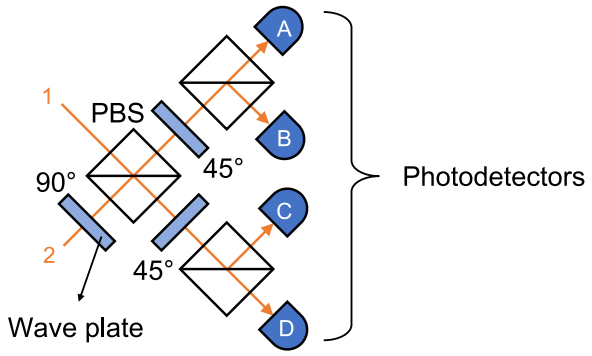

Figure 42: BSM scheme for single-photon polarization qubits. It uses three polarizing Beam splitters (PBSs), 90° and 45° wave plates, and four (A–D) photodetectors (single-photon resolving or on-off detectors). A PBS transmits (reflects) photons polarized horizontally (vertically). The scheme distinguishes $|\psi^{\pm}\rangle$: $|\psi^{+}\rangle$ if detectors (A,C) or (B,D) detect one photon respectively and $|\psi^{-}\rangle$ if detectors (A,D) or (B,C) detect one photon respectively. If otherwise, it fails or detects a loss, which can be distinguished by the total number of detected photons if single-photon resolving detectors are used. Two distinguishable Bell states can be chosen by putting or removing wave plates appropriately before the first PBS.

Pauli operator on a and b that is not identity cannot be a stabilizer, thus it anticommutes with at least one stabilizer. (If such an operator P_aP_b commutes with all stabilizers, $P_aP_b|\psi_G\rangle_V$ is also stabilized by $\mathcal{S}$, which means that $P_aP_b|\psi_G\rangle_V = |\psi_G\rangle_V$ since $\mathcal{S}$ stabilizes the zero-dimensional Hilbert space.) Consequently, $\mathrm{Tr}(P_aP_b\rho_{ab}) = \langle\psi_G|P_aP_b|\psi_G\rangle = 0$ for every single- or two-qubit Pauli operator P_aP_b that is not identity. The state ρ_{ab} satisfying this condition is unique and maximally mixed. □

We now introduce the methodology to track the errors caused by non-ideal fusions. Let us revisit the example in Fig. 41, supposing that the qubits are single-photon polarization ones and there are no photon losses. Then

a BSM can discriminate between only two Bell states (say, $|\psi^{\pm}\rangle$) among the four without additional resources [92]; see Fig. 42 for the scheme. The intact final state $|C_{\rm f}\rangle$ is obtained only when the BSM succeeds. When the BSM fails (which is heralded), $m_{\rm lett}$ is determined while $m_{\rm sign}$ is left completely ambiguous. In other words, the posterior probability that the input state is $|\phi^{\pm}\rangle$ for the obtained photodetector outcomes is equal for both signs ($\pm$), assuming that the four Bell states have the same prior probability. This assumption can be justified by Proposition 4.1. Therefore, we fix the value of $m_{\rm lett}$ while randomly assigning that of $m_{\rm sign}$. Then, the operator $m_{\rm sign}X_1Z_{1'}Z_{2'}$, which is originally a stabilizer of $|C_{\rm f}\rangle$, gives ± 1 randomly when it is measured after the failed BSM. Whereas, the other two stabilizers $m_{\rm lett}Z_1X_{1'}$ and $m_{\rm lett}Z_1X_{2'}$ are left undamaged. The key point is that this situation is equivalent to a 50% chance of an erroneous qubit 1 in $|C_{\rm f}\rangle$ in terms of stabilizer statistics. In other words, both situations give the same statistics if the stabilizers of $|C_{\rm f}\rangle$ are measured; thus, every process in MBQC described with the stabilizer formalism works in the same way.

Generally, a nonideal BSM gives one of the possible outcomes and the posterior probability of each Bell state for the outcome can be calculated with the Bayesian theorem, assuming the equal prior probabilities of the Bell states. Accordingly, the Bell state with the highest posterior probability is selected as the result of the BSM, and the probability $q_{\rm sign}$ ($q_{\rm lett}$) that the selected sign (letter) is wrong can be obtained as well. These error probabilities are "propagated" into nearby qubits in a way that the stabilizer statistics are preserved. For example, if the fusion in Fig. 41 is nonideal in such a way, it is equivalent to qubit 1 having an error with probability $q_{\rm sign}$ and qubits $1'$

and $2'$ having correlated errors with probability q_{lett}. We term a qubit with a nonzero error rate **deficient**.

Additionally, if a qubit participating in a fusion is erroneous, this error is propagated to the qubits on the opposite side. For example, an erroneous qubit 0 in Fig. 41 induces an error in the $X_0Z_{0'}$ measurement, which is equivalent to erroneous qubits $1'$ and $2'$.

The above error tracking methodology can be utilized for accurate and effective error simulations. The method can precisely locate qubits affected by unsuccessful fusions, which is closer to reality than simple bond disconnection or qubit removal. Since unsuccessful fusions are now regarded as Pauli error sources, we no longer need lattice deformation and the construction of supercheck operators [50, 70]. Instead, the error probabilities on individual qubits are employed for decoding syndromes in an adaptive manner (with decoders such as the *weighted* minimum-weight perfect matching one), which may be particularly effective if the probabilities are between 0 and $1/2$ since regarding such errors as just removal of qubits is a loss of information.

4.1.3 Building a lattice

An RTCS can be built with two types of linear three-qubit graph states called **central and side microclusters** [85, 78]. The process is composed of two steps (see Fig. 43): In step 1, a central microcluster and two side microclusters are merged by two fusions to form a five-qubit graph state named a **star cluster** composed of one **central qubit** and four **side qubits**. In step 2, the side qubits of star clusters are fused to form an RTCS. Even-

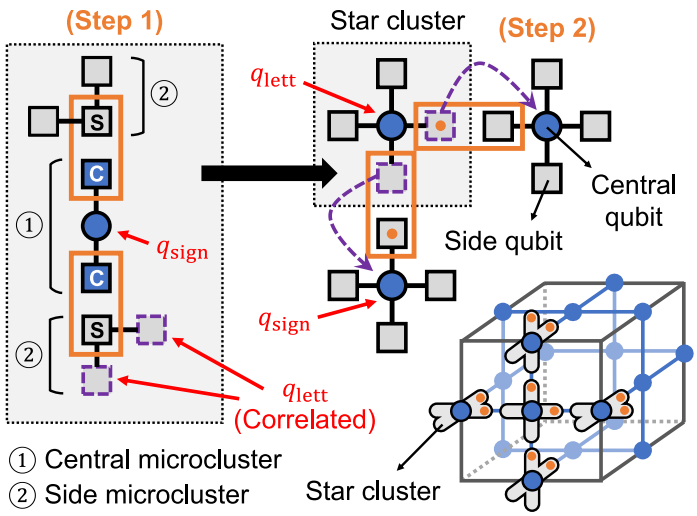

Figure 43: Lattice building process with microclusters. The orange boxes indicate fusions. In step 1, side and central microclusters are fused to form a star cluster. The locations of the Hadamard gates are marked as "C" ("S") for the HIC (HIS) configuration. In step 2, multiple star clusters are fused to form an RTCS. The macroscopic picture of step 2 in a unit cell of the lattice is depicted in the lower right. The locations of the Hadamard gates are marked as orange dots. The error probabilities of qubits assigned by one fusion in each step for the HIC configuration are written in red, where q_{sign} (q_{lett}) is the sign (letter) error probability of the BSM. Errors in the side qubits remaining after step 1 (purple dashed squares) are propagated to central qubits during step 2 (purple dashed arrows).

tually, the lattice includes only the central qubits, which are measured in appropriate bases for MBQC. For step 2, we consider two options: (i) Star clusters with successful step-1 fusions may be post-selected, or (ii) all generated star clusters are used regardless of the fusion results. The locations of the Hadamard gates during fusions (called H**-configuration**) may be chosen arbitrarily. Here, we define two specific H-configurations: **Hadamard-in-center (HIC)** and **Hadamard-in-side (HIS)**. In the HIC (HIS) configura-

tion, the Hadamard gates in step 1 are applied on qubits in the central (side) microclusters, as shown in Fig. 43. Whereas the Hadamard gates in step 2 are arranged in the same pattern for both configurations.

Nonideal fusions during lattice building render some central qubits in the final lattice deficient, as shown in Fig. 43 when the HIC configuration is used. When the HIS configuration is used, the positions of q_{sign} and q_{lett} in the figure are swapped. Note that errors in the side qubits are propagated to the nearest central qubits after step 2. Correlation between the sign and letter errors of a fusion, if any, can be neglected if the primal and dual lattices are considered separately since these errors respectively affect primal and dual [34] qubits (or vice versa).

4.2 Parity-encoding-based topological quantum computing

We introduce the new linear-optical **parity-encoding-based topological quantum computing (PTQC) protocol**, where fusion success rates are boosted by using multiphoton qubits for all qubits that participate in fusions and single-photon polarization encoding is used for central qubits. The **parity encoding** [88] is employed for the multiphoton qubits, which are fused by **concatenated Bell-state measurement (CBSM)** [89]. On-off or single-photon resolving detectors are used as photodetectors, and GHZ-3 states, which can be generated linear-optically [93], are regarded as basic resource

states. The (n,m) parity encoding defines a basis as

$$|0_L\rangle := \left|+^{(m)}\right\rangle^{\otimes n}, \qquad |1_L\rangle := \left|-^{(m)}\right\rangle^{\otimes n}, \tag{4.1}$$

where

$$\left|\pm^{(m)}\right\rangle := (|\mathrm{H}\rangle + |\mathrm{V}\rangle)^{\otimes m} \pm (|\mathrm{H}\rangle - |\mathrm{V}\rangle)^{\otimes m}. \tag{4.2}$$

The Hilbert space has a hierarchical structure composed of three levels: the lattice, block, and physical levels with respective bases $\{|0_L\rangle, |1_L\rangle\}$, $\left\{\left|\pm^{(m)}\right\rangle\right\}$, and $\{|\mathrm{H}\rangle, |\mathrm{V}\rangle\}$. In the original CBSM scheme [89], a BSM of a certain level is decomposed into multiple BSMs of one level below. Our current CBSM scheme slightly differs from the original one in the following two areas: (i) We consider two types of photodetectors: single-photon resolving and on-off detectors. A physical-level BSM can discriminate between a photon loss and failure only if single-photon resolving detectors are used. (ii) The letter outcome of a lattice-level BSM is obtained by a weighted majority vote of block-level letter outcomes. See Sec. 4.3 for the details of the CBSM scheme and its error rates.

4.2.1 Noise model

We consider a noise model where each photon suffers an independent loss with probability η, which arises from imperfections throughout the protocol: GHZ-3 states (which are initial resource states), delay lines, optical switches, and photodetectors. We assume that noise that cannot be modeled

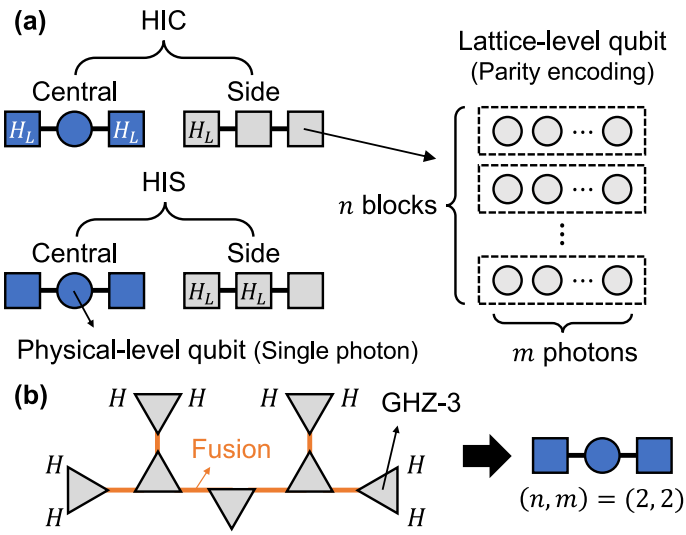

Figure 44: Structure and generation of post-H microclusters for PTQC. **(a)** Schematic of central and side post-H microclusters used in PTQC for the two H-configurations, HIC and HIS. The marks "H_L" indicate the locations of the lattice-level Hadamard gates. **(b)** Example of a process generating a post-H microcluster from GHZ-3 states. Each GHZ-3 state is represented by a triangle whose vertices indicate its three photons. An orange line connecting two vertices and a mark "H" next to a vertex respectively mean a fusion and Hadamard gate performed on the photon(s). The graph of the triangles connected with the orange lines is called a merging graph.

with photon losses such as dark counts is negligible. Note that not only non-ideal fusions but also photon losses in central qubits, which are detectable by on-off detectors, may incur deficiency. If the measurement outcome of a central qubit cannot be determined due to photon losses, we select the outcome randomly and assign an error rate of 50% to the qubit.

4.2.2 Generation of microclusters

For practical reasons, we consider generating **post-H microclusters** (that is, the states obtained by applying several lattice-level Hadamard gates on microclusters) directly from GHZ-3 states, instead of generating microclusters first and then applying the lattice-level Hadamard gates for the fusions. Figure 44(a) depicts the central and side post-H microclusters for the HIC and HIS configurations. A post-H microcluster can be generated up to several physical-level Hadamard gates by performing physical-level BSMs or fusions (referred to as **merging operations**) between multiple GHZ-3 states according to a predetermined **merging graph**, as shown in the example of Fig. 44(b). Note that the merging graph may be not unique for a post-H microcluster. However, each merging operation has a low success rate of less than or equal to 50%, which may lead to extensive usage of GHZ-3 states for generating a post-H microcluster successfully. Thus, the generation process, which is determined by the merging graph and the order of the merging operations, should be adjusted carefully to minimize the resource overhead. To optimize the merging order, our protocol utilizes a graph edge coloring algorithm, based on the idea that merging operations for non-adjacent edges can be performed simultaneously.

We now address the generation of post-H microclusters and the optimization problem in detail.

Physical-level graphs of post-H microclusters: We first present the physical-level graphs of post-H microclusters for PTQC. A post-H microcluster, which is composed of three lattice-level qubits or two of them and one

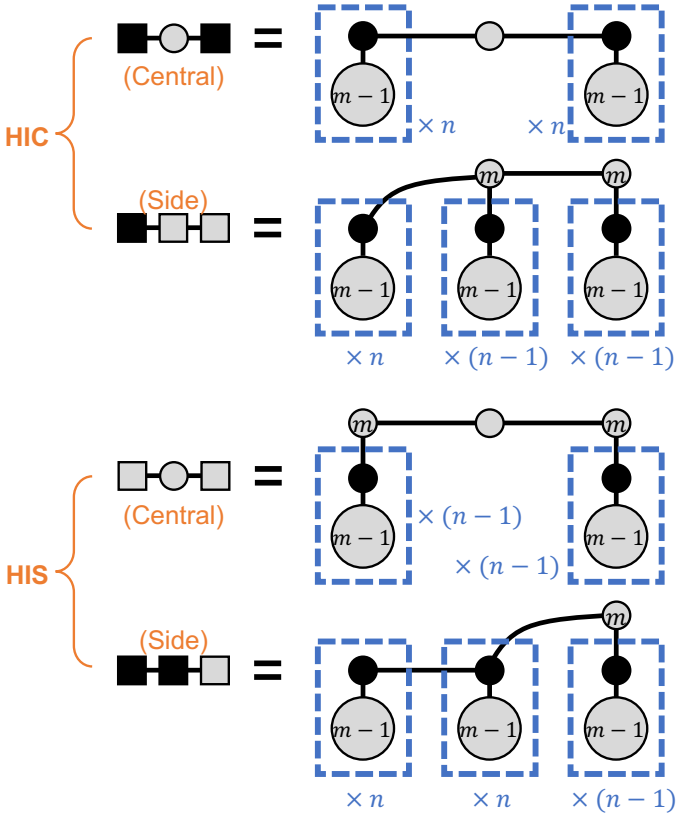

Figure 45: Physical-level graphs of post-H microclusters for the HIC and HIS configurations when the (n,m) parity encoding is used for PTQC. The squares (circles) correspond to lattice-level (physical-level) qubits, among which black ones indicate that the lattice-level (physical-level) Hadamard gates are applied to the qubits on the graph state. A blue dashed box indicates a group of recurrent subgraphs; that is, the structure in the box is repeated as many times as indicated, and if there is an edge across the border of the box, it means that edges of the same pattern exist in each of the repeated structures. A number inside a circle means a blue dashed box surrounding only the circle with the indicated repetition number. If there is an edge between two blue dashed boxes or circles containing numbers, the full graph can be recovered just by expanding them one by one. See Fig. 46 for examples.

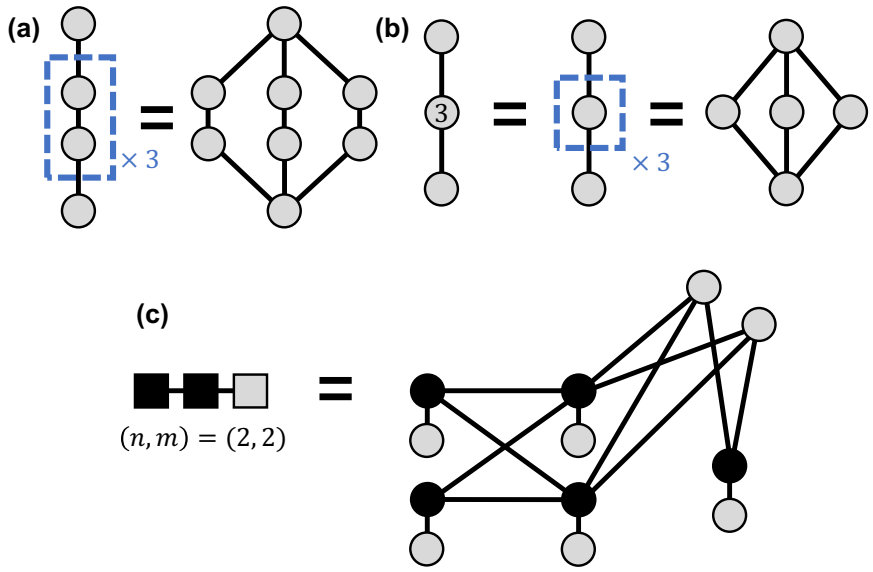

Figure 46: Examples of graph notations used in Fig. 45. **(a)** and **(b)** respectively show examples of a blue dashed box and a number inside a circle, which indicate groups of recurrent subgraphs. **(c)** shows the full graph of the side microcluster of the HIS configuration when $n = m = 2$.

photon (physical-level qubit), can be regarded as a graph state of photons up to several physical-level Hadamard gates. The graph of this graph state, called the **physical-level graph** of the post-H microcluster, is visualized in Fig. 45 for each post-H microcluster; see Sec. 4.7 for their derivation. Here, the squares (circles) indicate lattice-level (physical-level) qubits. If a square (circle) is filled with black, it means that the lattice-level (physical-level) Hadamard gate is applied on the qubit after the involved edges are connected. Recurrent subgraphs are abbreviated as blue dashed squares or circles with numbers; see Fig. 46 for the detailed interpretation of these notations.

Generating post-H microclusters: We now depict the ways to generate a specific post-H microcluster from GHZ-3 states. We first describe a straightforward method and then adjust or generalize it. The final method can be summarized as follows:

1. Determine a **merging graph** G for the post-H microcluster that we want to create by the algorithm presented below. Each edge of G is labeled as either "internal" or "external."

2. For each vertex v in G, Prepare a GHZ-3 state $|\mathrm{GHZ}_3\rangle_v$.

3. For each edge e in G that connects v_1 and v_2, perform a BSM (fusion) on two photons selected respectively from $|\mathrm{GHZ}_3\rangle_{v_1}$ and $|\mathrm{GHZ}_3\rangle_{v_2}$ if e is an internal (external) edge. The order of the operations does not matter.

We define the **GHZ-l state** for an integer $l \geq 3$ by the state $|\mathrm{GHZ}_l\rangle := |\mathrm{H}\rangle^{\otimes l} + |\mathrm{V}\rangle^{\otimes l}$. Note that it is a state obtained from a graph state with a star graph (where the number of vertices is l) by applying Hadamard gates on all the leaves of the graph; namely,

$$|\mathrm{GHZ}_l\rangle = H_2 \cdots H_l C_{12}^Z \cdots C_{1l}^Z |+\rangle^{\otimes l}.$$

We refer to the first photon of the above expression as the **root photon** of the state (which can be chosen arbitrarily) and the other photons as its **leaf photons**.

If a *BSM* is performed on the root photon of a GHZ-l_1 state and a leaf photon of a GHZ-l_2 state, the resulting state on the remaining photons

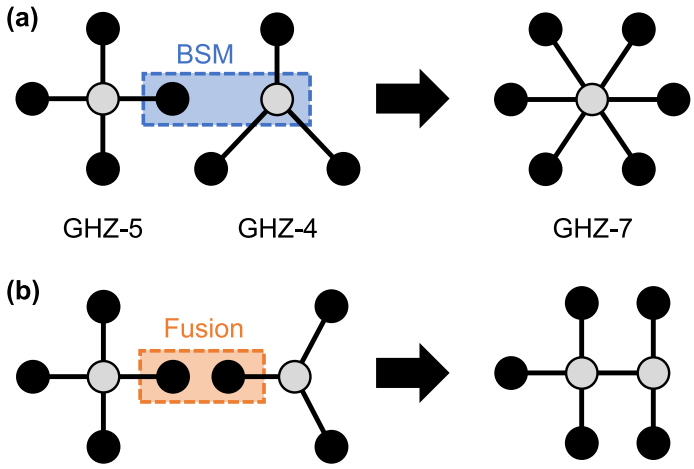

Figure 47: Examples of the two types of merging operations on two GHZ states: **(a)** a BSM on the root photon of one state and a leaf photon of the other and **(b)** a fusion on two leaf photons.

is a GHZ-$(l_1 + l_2 - 2)$ state; see Fig. 47(a) for an example. Thus, an arbitrary GHZ state can be constructed by performing BSMs on multiple GHZ-3 states appropriately. On the other hand, if a *fusion* is performed on two leaf photons selected respectively from GHZ-l_1 and GHZ-l_2 states, the resulting state is no longer a GHZ state, but it is a graph state (up to some Hadamard gates) with a graph containing a vertex with degree $l_1 - 1$, a vertex with degree $l_2 - 1$, and multiple vertices with degree one; see Fig. 47(b) for an example. (The degree d_v of a vertex v means the number of edges connected to v.)

Combining the above facts, a post-H microcluster (or an arbitrary graph state) with the physical-level graph G can be generated from GHZ-3 states up to physical-level Hadamard gates in the following way: For each vertex v of G with a degree larger than one, prepare a state $|\mathrm{GHZ}_{d_v+1}\rangle_v$ through

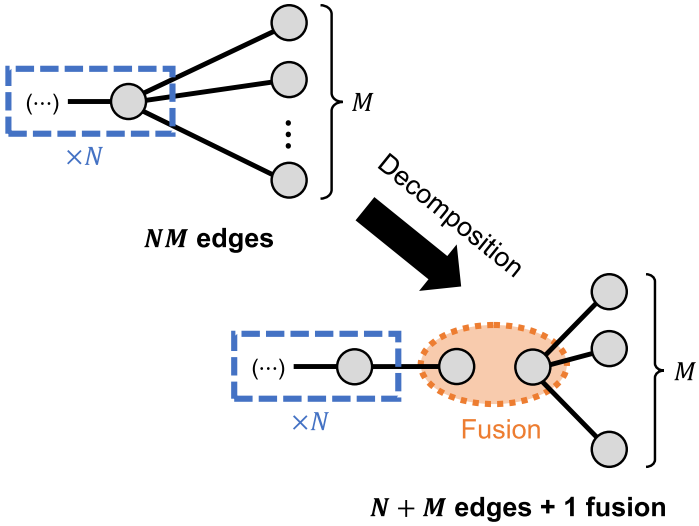

Figure 48: Decomposition of a graph state done by separating recurrent subgraphs that are connected with multiple vertices.

BSMs on GHZ-3 states. Then, for each edge (v_1, v_2) of G, perform a fusion on two photons selected respectively from $\left|\mathrm{GHZ}_{d_{v_1}+1}\right\rangle_{v_1}$ and $\left|\mathrm{GHZ}_{d_{v_2}+1}\right\rangle_{v_2}$. We refer to each BSM or fusion during this process as a **merging operation**.

However, the above method still has room for improvement. The physical-level graphs in Fig. 45 can be decomposed into multiple components that are combined by fusions through the process shown in Fig. 48. Here, each recurrent subgraph connected with multiple vertices is separated and connected with only one vertex. The decomposition of various post-H microclusters is explicitly presented in Figs. 49 and 50 for the HIC and HIS configurations, respectively. To generate a post-H microcluster, we prepare the individual components first by the aforementioned method, then merge them through fusions. This process may greatly reduce the number of required merging

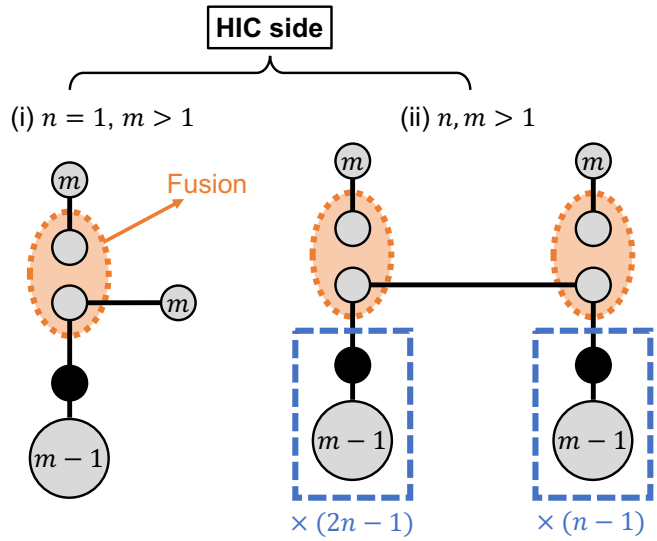

Figure 49: Decomposition of post-H microclusters for the HIC configuration. Different types of post-H microclusters are decomposed by the method shown in Fig. 48. Only the side microclusters are considered since the central microclusters do not have connected pairs of recurrent subgraphs, thus their physical-level graphs are single components by themselves.

operations since the number of edges decreases as shown in Fig. 48.

Furthermore, we can generalize the method using the fact that every merging operation commutes with each other. That is, even if all the fusions and BSMs in the above process are performed in an arbitrary order, the final state does not vary (up to the change of the Pauli frame). To systematically address this feature, we define a **merging graph** of a post-H microcluster or one of its components by a graph in which the vertices correspond to initial GHZ-3 states and the edges indicate the merging operations between them required to generate the state. Each edge of a merging graph is either **internal** or **external** that corresponds to BSMs or fusions, respectively.

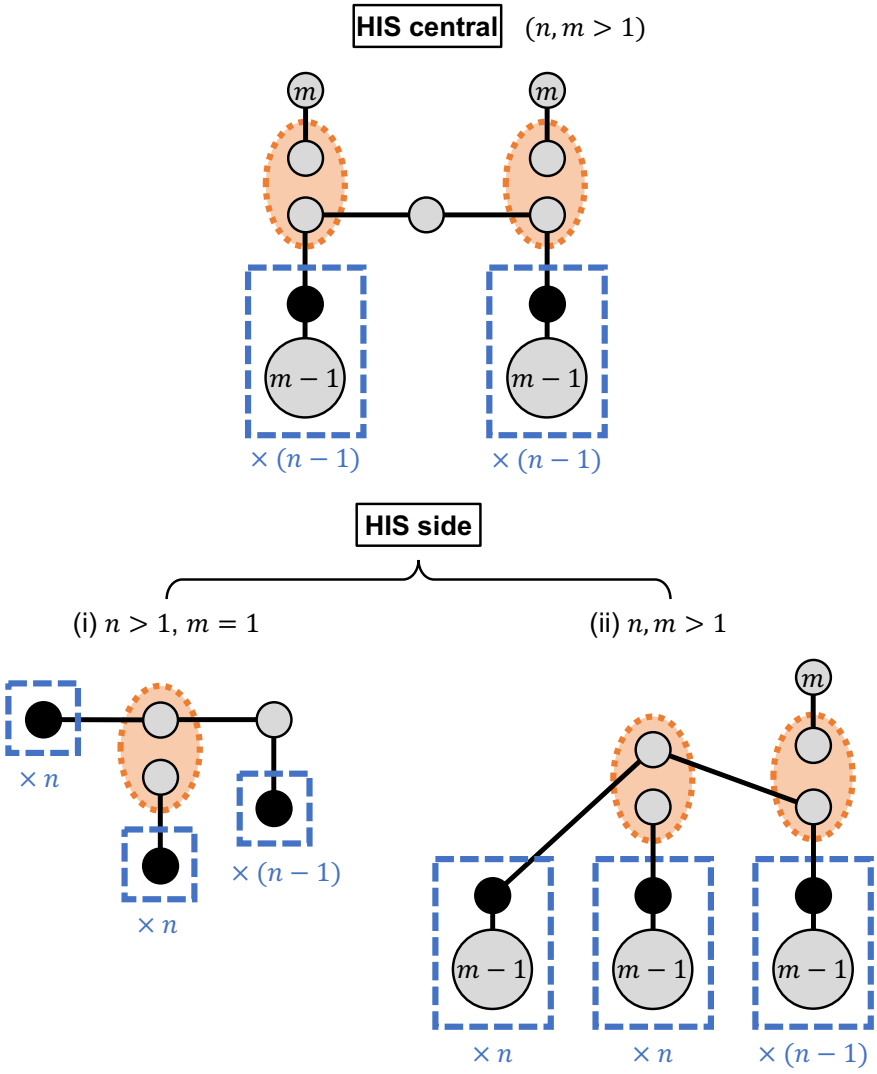

Figure 50: Decomposition of post-H microclusters for the HIS configuration. Different types of post-H microclusters are decomposed by the method shown in Fig. 48. Post-H microclusters that are not presented here do not have connected pairs of recurrent subgraphs, thus their physical-level graphs are single components by themselves.

A merging graph G_{mrg} of a component can be constructed by the following algorithm starting from its physical-level graph (see Fig. 51 for two examples):

1. Initialize the graph $G_{\mathrm{mrg}} = (V, E)$ by the graph $G = (V_0, E_0)$ of the component; that is, $V \leftarrow V_0, E \leftarrow E_0$.

2. Let us define $V_{\deg \geq 3} := \{v \in V | d_v \geq 3\}$. This set is fixed and not updated during the entire process. For each vertex $v \in V_{\deg \geq 3}$, perform the follows:

   (a) Remove v from G_{mrg} and add $d_v - 1$ new vertices. Let $V_{\mathrm{new}} = \left(v_{\mathrm{new}}^{(1)}, \cdots, v_{\mathrm{new}}^{(d_v-1)}\right)$ denote the series of the new vertices and $V_{\mathrm{ngh}} = \left(v_{\mathrm{ngh}}^{(1)}, \cdots, v_{\mathrm{ngh}}^{(d_v)}\right)$ denote the series of the vertices that were adjacent to v before removing it. The order of the vertices in V_{ngh} can be arbitrarily chosen.

   (b) Connect the vertices in V_{new} linearly with *internal* edges; namely, connect $\left(v_{\mathrm{new}}^{(1)}, v_{\mathrm{new}}^{(2)}\right)$, $\left(v_{\mathrm{new}}^{(2)}, v_{\mathrm{new}}^{(3)}\right)$, and so on.

   (c) Choose one of the vertices in V_{new} arbitrarily and term it the seed vertex v_{seed} of v_0, where v_0 is the vertex in G from which v originates.

   (d) Let us define a series V'_{new} by omitting v_{seed} from V_{new} while keeping the order of the other vertices. For each $i \in \{1, \cdots, d_v - 2\}$, connect $v_{\mathrm{ngh}}^{(i)}$ and the ith element of V'_{new} with an *external* edge.

   (e) Connect $\left(v_{\mathrm{new}}^{(1)}, v_{\mathrm{ngh}}^{(d_v-1)}\right)$ and $\left(v_{\mathrm{new}}^{(d_v-1)}, v_{\mathrm{ngh}}^{(d_v)}\right)$ with *external* edges.

3. Remove all vertices with degree 1 from the graph. For each vertex $v_0 \in V_0$ with degree 2, if v is the vertex in V originating from v_0 (which is not removed in the previous steps), define the seed vertex of v_0 as v.

It is worth noting that there are two degrees of freedom in the above algorithm for each vertex with a degree larger than 2: (i) the order of the series V_{ngh} and (ii) the selection of the seed vertex. Different merging graphs can be constructed depending on their selection, which may severely affect the resource overheads.

The merging graph of a post-H microcluster is constructed by combining the merging graphs of its components. That is, for each fusion between different components, the corresponding seed vertices in the merging graphs are connected by an external edge.

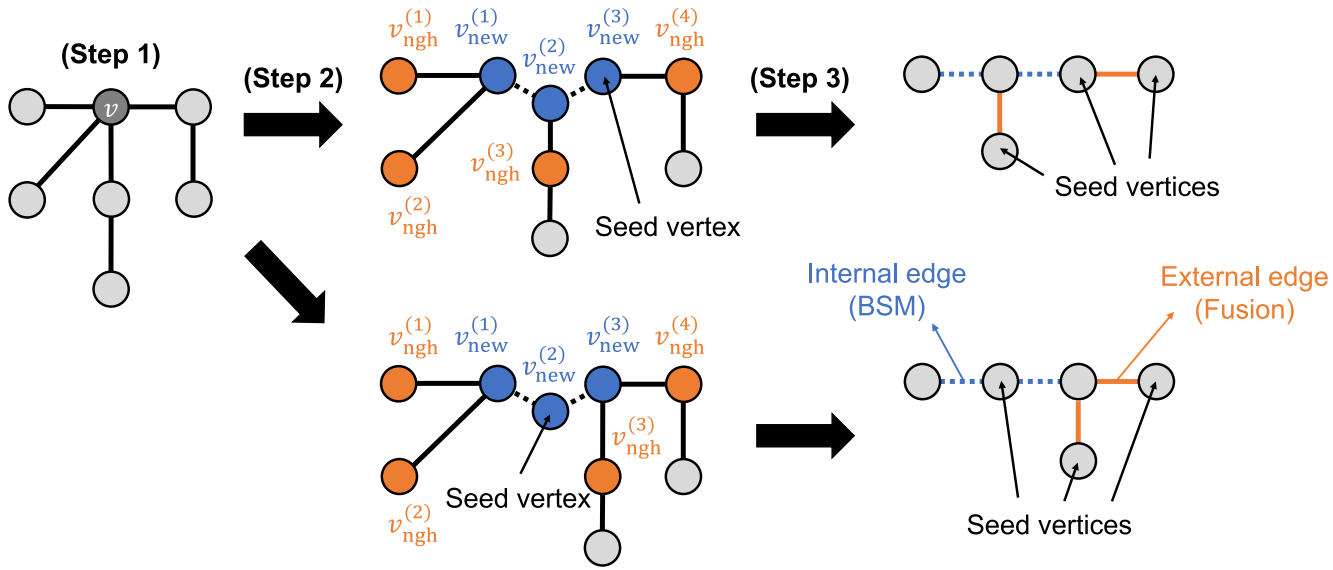

Figure 51: Construction of merging graphs from a physical-level graph. v is the only vertex with a degree larger than two in the original graph. The upper and lower processes differ in the selection of the seed vertex for the decomposition of v.

Optimization of resource overheads: The process of generating a post-H microcluster described above is determined by two factors: the merging graph and the order of the merging operations. Here, we discuss their optimization for minimizing resource overhead. The merging graph is selected randomly by the algorithm presented above. Based on it, we determine the order of the merging operations through an algorithm found heuristically and calculate the expected number $N_{\mathrm{GHZ}}^{\mathrm{MC}}$ of GHZ-3 states required to generate the state. We repeat this process for a large enough number to obtain as low resource overhead as possible.

During the generation process, performing each merging operation can be regarded as contracting the corresponding edge, which means removing the edge, merging the two vertices (v_1, v_2) that it previously joined into a new vertex w, and reconnecting all the edges that were connected to v_1 and v_2 with w. Here, each vertex indicates a connected subgraph (a group of entangled photons) of the intermediate graph state. We assign a "weight" N_v (which is initialized to 1) on each vertex v, which is the average number of GHZ-3 states required to generate the connected subgraph. If the edge between two vertices v_1 and v_2 are contracted, the new vertex w has the weight of

$$N_w = \frac{2}{(1-\eta)^2}(N_{v_1} + N_{v_2}) =: N_{v_1} +_m N_{v_2}, \tag{4.3}$$

where the factor $2/(1-\eta)^2$ is the inverse of the success probability of the merging operation. By repeating this process, the post-H microcluster is obtained when there is only one vertex left, whose weight is equal to $N_{\mathrm{GHZ}}^{\mathrm{MC}}$.

To find an optimal order of merging operations, we use the following strategy:

1. Find the set $E_{\text{min.wgt}}$ of edges with the smallest weight, where the weight of an edge (v_1, v_2) is defined as $N_{v_1} +_m N_{v_2}$.

2. Using an edge coloring algorithm, allocate "colors" to all edges so that different edges sharing a vertex have different colors and as few colors as possible are used.

3. Partition $E_{\text{min.wgt}}$ into disjoint subsets by the colors of the edges. Find the largest subset E_{mrg} among them. If such a subset is not unique, choose one randomly.

4. Contract each edge in E_{mrg} in an arbitrary order.

5. Repeat all the above steps until only one vertex is left.

The strategy is based on the following two intuitions: First, it is better to merge vertices with small weights first, since $(N_1 +_m N_2) +_m N_3 < N_1 +_m (N_2 +_m N_3)$ if $N_1 < N_2 < N_3$. Secondly, it is better to perform merging operations in parallel as much as possible. Such a set of edges can be found by the edge coloring algorithm. For our results, we have used the function `coloring.greedy_color` in NetworkX package [94] with the strategy `largest_first`. (Since the function performs vertex coloring, we input the line graph of G_{mrg} into the function.)

4.2.3 Performance analysis

For error simulations, we consider the logical identity gate with the length T of $4d+1$ unit cells along the simulated time axis, where d is the code distance. All the fusion outcomes are sampled from appropriate probability distributions, and the corresponding error rates are assigned to individual central qubits according to the process described earlier. These error rates are exploited when decoding syndromes by the *weighted* minimum-weight perfect matching in the PyMatching package [95]. The loss thresholds are calculated by finding the intersections of logical error rates for $d = 9$ and $d = 11$. The resource overhead of PTQC is quantified by the average total number $\mathcal{N}_{p_L}$ of GHZ-3 states to achieve a target logical error rate of p_L for the logical identity gate of $T = d - 1$, which depends on the photon loss rate η. See Secs. 4.7.1 and 4.7.2 for the detailed methods of error simulations and resource calculations, respectively.

The simulation results of the loss thresholds and the resource overheads (quantified by $\mathcal{N}_{10^{-6}}$) are respectively presented in Figs. 52 and 53 for the two types of photodetectors, the two options for the post-selection of star clusters, and the two H-configurations.

Figure 52 shows that, if single-photon resolving detectors are used, η_{th} reaches up to 8.5% ($n = 5,\ m = 4,\ j = 2$) when star clusters are post-selected and up to 6.3% ($n = m = 5,\ j = 3$, HIC) when they are not. If on-off detectors are used, η_{th} reaches up to 4.4% ($n = 5,\ m = 4,\ j = 1$) when star clusters are post-selected and up to 3.3% ($n = 5,\ m = 4,\ j = 1$, HIS) when they are not. The post-selection of star clusters increases the photon loss

thresholds by about 1–2%p.

From Fig. 53, it is observed that the protocol using single-photon resolving detectors is most resource-efficient with $\mathcal{N}_{10^{-6}} \approx 5 \times 10^5$ ($n = 4,\ m = 3,\ j = 1$, HIC) when star clusters are post-selected and with $\mathcal{N}_{10^{-6}} \approx 1 \times 10^6$ ($n = m = 4,\ j = 2$, HIS) when they are not. If on-off detectors are used, the protocol is most resource-efficient with $\mathcal{N}_{10^{-6}} \approx 2 \times 10^7$ ($n = m = 4,\ j = 2$, HIC) when star clusters are post-selected and with $\mathcal{N}_{10^{-6}} \approx 3 \times 10^7$ ($n = m = 5,\ j = 2$, HIC) when they are not. It is worth noting that, compared to the protocol without the post-selection, the protocol with it requires fewer GHZ-3 states to achieve a target logical error rate. In other words, further fault-tolerance obtained by using only successfully-generated star clusters leads to a positive overall effect that surpasses the negative effect caused by the increase in the number of required GHZ-3 states for one central qubit in the final lattice.

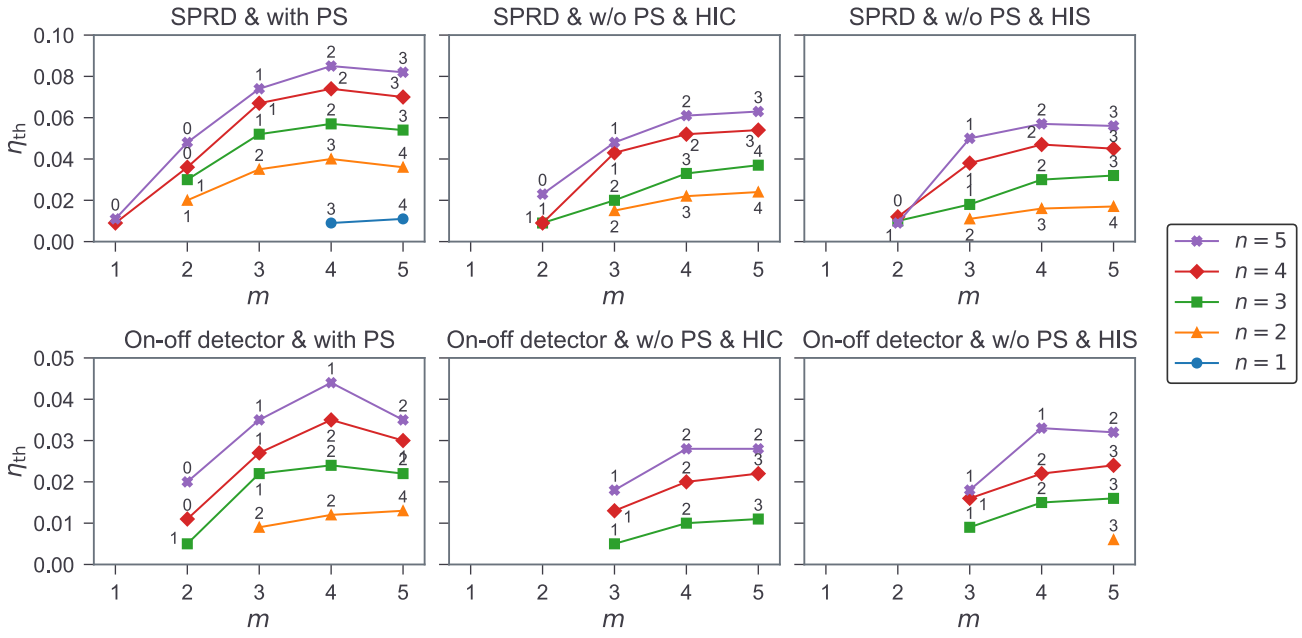

Figure 52: Loss threshold η_{th} for various parameters on the encoding size (n,m), the type of detectors, the post-selection (PS) of star clusters, and the H-configuration. "SPRD" stands for single-photon resolving detector. The values of j are chosen to maximize η_{th} and shown next to the data points. The H-configuration does not affect the results when star clusters are post-selected.

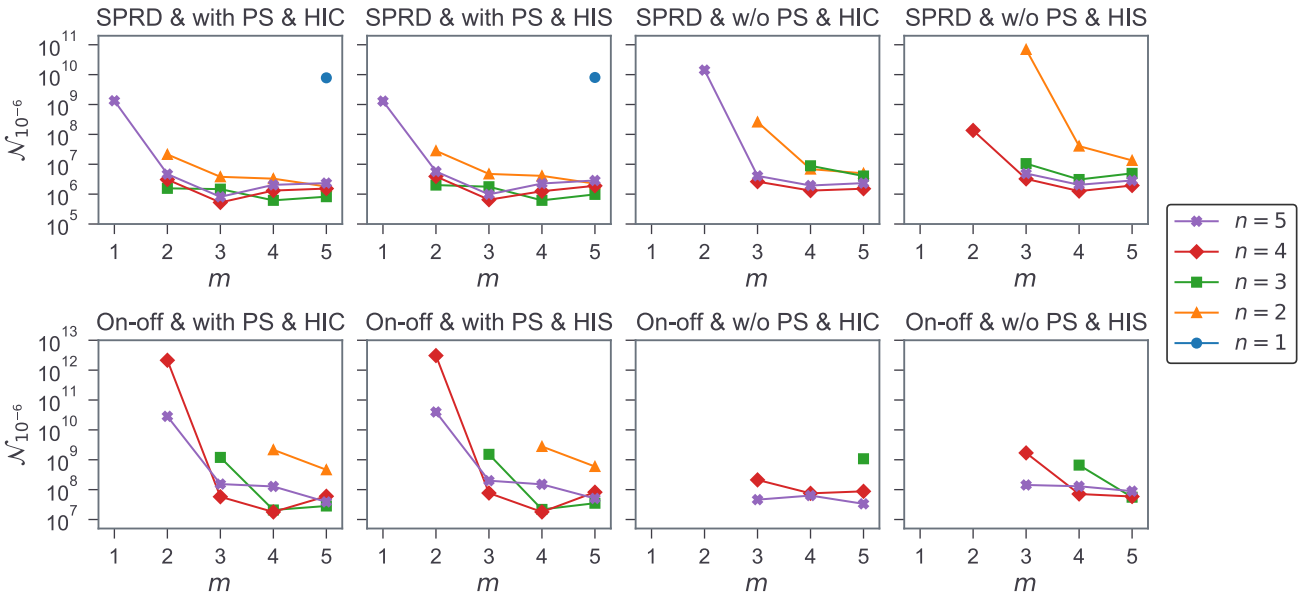

Figure 53: Resource overhead $\mathcal{N}_{10^{-6}}$ (calculated at $\eta = 0.01$) for various parameters on the encoding size (n,m), the type of detectors, the post-selection (PS) of star clusters, and the H-configuration. "SPRD" stands for single-photon resolving detector. The values of j are chosen to be the same as the ones in Fig. 52.

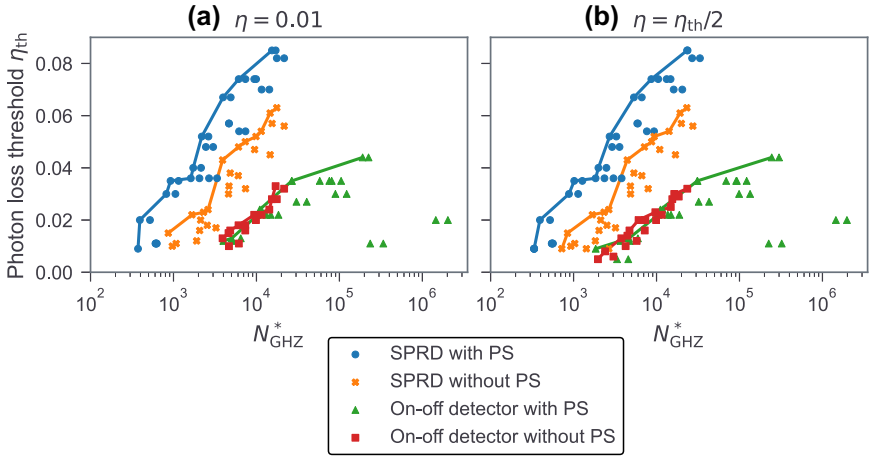

Figure 54: Photon loss thresholds η_{th} as a function of the number N^*_{GHZ} of GHZ-3 states required per central qubit. N^*_{GHZ} is calculated at $\eta = 0.01$ or $\eta = \eta_{\text{th}}/2$. "SPRD" stands for a single-photon resolving detector. The data points correspond to different parameter settings on the type of detectors, the post-selection (PS) of star clusters, the encoding size, and the H-configuration, which are grouped by the first two factors. The upper envelope for each of the groups is presented as a line. The values of j are chosen to maximize η_{th}.

Additionally, Fig. 54 presents the photon loss thresholds as a function of N^*_{GHZ} when η is fixed to 0.01 or variable as $\eta = \eta_{\text{th}}/2$, which is used to calculate $\mathcal{N}_{10^{-6}}$. aIt shows that at least about 400 GHZ-3 states are required per central qubit for PTQC to work. The explicit information of the data points along the upper envelope lines in the figure is presented in Tables 6 and 7 for single-photon resolving and on-off detectors, respectively.

Lastly, we show evidence that our optimizing strategy for microcluster generation described in Sec. 4.2.2 is indeed highly effective in terms of both the optimality of the calculated overhead and searching time, by comparing its performance with those of its variants constructed by omitting or altering

specific steps. We consider four of such variants that are the same as the original strategy except for the following differences:

Variant 1 The original physical-level graph is directly used as a single component without decomposition.

Table 6: Information of the data points along the upper envelope lines in Fig. 54(b) when single-photon resolving detectors are used. N^*_{GHZ}, $\mathcal{N}_{10^{-7}}$, and $d_{10^{-7}}$ at $\eta = 1\%$ are not calculated when $\eta_{\mathrm{th}} < 1\%$.

η_{th}	N^*_{GHZ} ($\eta = \eta_{\mathrm{th}}/2$)	N^*_{GHZ} ($\eta = 1\%$)	$\mathcal{N}_{10^{-7}}$ ($\eta = 1\%$)	$d_{10^{-7}}$	n	m	j	H-config.
(a) Single-photon resolving detector with post-selection								
0.009	3.3×10^2	3.7×10^2			1	4	3	HIC
0.02	3.9×10^2	3.9×10^2	2.1×10^7	21	2	2	1	HIC
0.03	8.8×10^2	8.2×10^2	1.5×10^6	7	3	2	1	HIC
0.035	1.0×10^3	9.2×10^2	3.8×10^6	9	2	3	2	HIC
0.036	1.8×10^3	1.6×10^3	3.0×10^6	7	4	2	1	HIC
0.04	1.9×10^3	1.7×10^3	3.2×10^6	7	2	4	3	HIC
0.052	2.7×10^3	2.1×10^3	1.4×10^6	5	3	3	1	HIC
0.067	5.3×10^3	3.9×10^3	5.2×10^5	3	4	3	1	HIC
0.074	8.6×10^3	6.1×10^3	8.1×10^5	3	5	3	1	HIC
0.085	2.3×10^4	1.5×10^4	2.0×10^6	3	5	4	2	HIC
(b) Single-photon resolving detector without post-selection								
0.009	7.2×10^2	-	-	-	3	2	1	HIC
0.015	8.4×10^2	8.6×10^2	2.5×10^8	37	2	3	2	HIC
0.022	1.6×10^3	1.6×10^3	6.8×10^6	9	2	4	3	HIC
0.023	2.3×10^3	2.3×10^3	1.4×10^{10}	101	5	2	1	HIC
0.024	2.6×10^3	2.6×10^3	4.9×10^6	7	2	5	4	HIC
0.043	4.4×10^3	3.9×10^3	2.6×10^6	5	4	3	1	HIC
0.048	7.1×10^3	6.0×10^3	4.0×10^6	5	5	3	1	HIC
0.05	8.7×10^3	7.3×10^3	4.9×10^6	5	5	3	1	HIS
0.052	9.3×10^3	9.8×10^3	1.3×10^6	3	4	4	2	HIC
0.054	1.4×10^4	1.1×10^4	1.5×10^6	3	4	5	3	HIC
0.061	1.9×10^4	1.4×10^4	1.9×10^6	3	5	4	2	HIC
0.063	2.3×10^4	1.7×10^4	2.3×10^6	3	5	5	3	HIC

Variant 2 GHZ-N states for $N \geq 3$ are first constructed and then they are merged by fusions to construct a post-H microcluster, which is the process described before introducing merging graphs in Sec. 4.2.2. A GHZ-N state is generated by merging two GHZ-$(N/2+1)$ states if N is even, or by merging a GHZ-$[(N+1)/2]$ state and a GHZ-$[(N+3)/2]$ state if N is odd. The order of the fusions is determined by the same strategy as the original one, regarding the merging graph where the vertices

Table 7: Information of the data points along the upper envelope lines in Fig. 54(b) when on-off detectors are used. N^*_{GHZ}, $\mathcal{N}_{10^{-7}}$, and $d_{10^{-7}}$ at $\eta = 1\%$ are not calculated when $\eta_{\mathrm{th}} < 1\%$.

η_{th}	N^*_{GHZ} $(\eta = \eta_{\mathrm{th}}/2)$	N^*_{GHZ} $(\eta = 1\%)$	$\mathcal{N}_{10^{-7}}$ $(\eta = 1\%)$	$d_{10^{-7}}$	n	m	j	H-config.
(a) On-off detector with post-selection								
0.009	1.8×10^3	-	-	-	2	3	2	HIC
0.012	3.6×10^3	3.9×10^3	2.1×10^9	45	2	4	2	HIC
0.013	4.6×10^3	5.0×10^3	4.6×10^8	25	2	5	4	HIC
0.022	1.0×10^4	1.0×10^4	1.1×10^9	27	3	3	1	HIC
0.024	1.1×10^4	1.0×10^4	2.0×10^7	7	3	4	2	HIC
0.035	3.1×10^4	2.6×10^4	1.7×10^7	5	4	4	2	HIS
0.044	2.4×10^5	1.9×10^5	1.2×10^8	5	5	4	1	HIC
(b) On-off detector without post-selection								
0.005	1.9×10^3	-	-	-	3	3	2	HIC
0.008	2.3×10^3	-	-	-	3	3	1	HIS
0.013	3.7×10^3	3.9×10^3	2.1×10^8	21	4	3	1	HIC
0.014	4.4×10^3	4.6×10^3	6.6×10^8	29	3	4	2	HIS
0.016	4.7×10^3	4.8×10^3	1.6×10^9	39	4	3	1	HIS
0.02	6.0×10^3	6.0×10^3	4.6×10^7	11	5	3	1	HIC
0.023	9.6×10^3	9.3×10^3	7.1×10^7	11	4	4	2	HIS
0.025	1.4×10^4	1.4×10^4	5.8×10^7	9	4	5	3	HIS
0.028	1.5×10^4	1.5×10^4	6.3×10^7	9	5	4	2	HIC
0.03	1.6×10^4	1.7×10^4	1.2×10^8	11	5	4	1	HIS
0.032	2.3×10^4	2.1×10^4	8.8×10^7	9	5	5	2	HIS

correspond to general GHZ states and the edges indicate the fusions to perform. Accordingly, the weight of each vertex is not initialized to 1, but to the expected number of GHZ-3 states used to generate the corresponding GHZ state.

Variant 3 The merging order is just randomly chosen without using a specific strategy.

Variant 4 The merging order is chosen by considering only the weights of the vertices without using the edge coloring algorithm. Namely, steps 2, 3, and 4 in the original strategy in Sec. 4.2.2 are replaced with contracting a random edge in $E_{\text{min.wgt}}$.

Figure 55 displays how the overhead of a side microcluster (namely, the expected number $N_{\text{GHZ}}^{\text{side}}$ of GHZ-3 states required to generate it) varies depending on the used strategy for several settings. Since the strategies contain randomness, the distributions of the outcomes are visualized as box plots. It is clearly shown that the original strategy is the most optimal in general, although Variant 2 or 4 is also as effective as the original strategy for some cases. Moreover, the original strategy gives the least variance on the calculated overhead (except for Variant 2), which means that the optimal point can be found quickly.

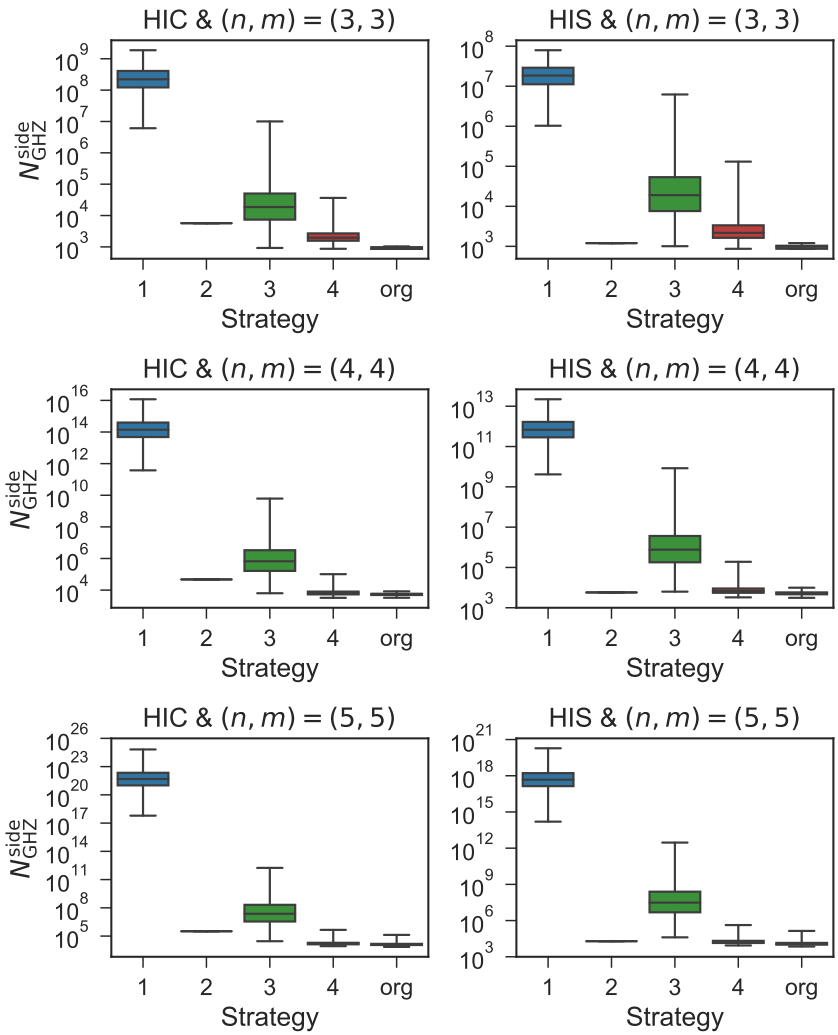

Figure 55: Comparison of different strategies for generating a post-H microcluster. It shows the distribution of the calculated overhead $N_{\mathrm{GHZ}}^{\mathrm{side}}$ of a side microcluster depending on the used strategy (among the original strategy and its four variants) for different H-configurations and values of n and m. We considered 9,600 samples for each box plot. Each box shows the range between the first and third quartile and the line crossing represents the median. The minimum and maximum values are indicated by whiskers.

4.3 Modified concatenated Bell-state measurement scheme

In this section, we describe the modified CBSM scheme used for PTQC. For the lattice, block, and physical levels of the (n,m) parity encoding, the Bell states are respectively defined as

$$\begin{cases} |\Phi^{\pm}\rangle := |0_L\rangle|0_L\rangle \pm |1_L\rangle|1_L\rangle, \\ |\Psi^{\pm}\rangle := |0_L\rangle|1_L\rangle \pm |1_L\rangle|0_L\rangle, \end{cases}$$

$$\begin{cases} \left|\phi^{\pm}_{(m)}\right\rangle := \left|+^{(m)}\right\rangle\left|+^{(m)}\right\rangle \pm \left|-^{(m)}\right\rangle\left|-^{(m)}\right\rangle, \\ \left|\psi^{\pm}_{(m)}\right\rangle := \left|+^{(m)}\right\rangle\left|-^{(m)}\right\rangle \pm \left|-^{(m)}\right\rangle\left|+^{(m)}\right\rangle, \end{cases}$$

$$\begin{cases} |\phi^{\pm}\rangle := |\mathrm{H}\rangle|\mathrm{H}\rangle \pm |\mathrm{V}\rangle|\mathrm{V}\rangle, \\ |\psi^{\pm}\rangle := |\mathrm{H}\rangle|\mathrm{V}\rangle \pm |\mathrm{V}\rangle|\mathrm{V}\rangle, \end{cases}$$

where $|0_L\rangle$, $|1_L\rangle$, and $\left|\pm^{(m)}\right\rangle$ are defined in Eqs. (4.1) and (4.2). The Bell states of each level can be decomposed into those of one level below as follows:

$$\left|\Phi^{\pm}\right\rangle = 2^{-\frac{n-1}{2}} \sum_{l:\mathrm{even(odd)}\le n} \mathcal{P}\left[\left|\phi^{-}_{(m)}\right\rangle^{\otimes l}\left|\phi^{+}_{(m)}\right\rangle^{\otimes n-l}\right], \tag{4.4a}$$

$$\left|\Psi^{\pm}\right\rangle = 2^{-\frac{n-1}{2}} \sum_{l:\mathrm{even(odd)}\le n} \mathcal{P}\left[\left|\psi^{-}_{(m)}\right\rangle^{\otimes l}\left|\psi^{+}_{(m)}\right\rangle^{\otimes n-l}\right], \tag{4.4b}$$

$$\left|\phi^{\pm}_{(m)}\right\rangle = 2^{-\frac{m-1}{2}} \sum_{k:\mathrm{even}\le m} \mathcal{P}\left[\left|\psi^{\pm}\right\rangle^{\otimes k}\left|\phi^{\pm}\right\rangle^{\otimes m-k}\right], \tag{4.4c}$$

$$\left|\psi^{\pm}_{(m)}\right\rangle = 2^{-\frac{m-1}{2}} \sum_{k:\mathrm{odd}\le m} \mathcal{P}\left[\left|\psi^{\pm}\right\rangle^{\otimes k}\left|\phi^{\pm}\right\rangle^{\otimes m-k}\right], \tag{4.4d}$$

where $\mathcal{P}[\cdot]$ means the summation of all the permutations of the tensor products inside the bracket. Therefore, a BSM can be performed in a concatenated manner: A lattice-level BSM ($\mathrm{BSM_{lat}}$) is done by n block-level BSMs ($\mathrm{BSM_{blc}}$'s), each of which is again done by m physical-level BSMs ($\mathrm{BSM_{phy}}$'s). We refer to the sign (letter) result obtained from a lattice-, block-, or physical-level BSM as the **lattice-, block-, or physical-level sign (letter)**, respectively.

4.3.1 Original CBSM scheme

We review the original CBSM scheme of the parity encoding in Ref. [89]. A $\mathrm{BSM_{phy}}$ can discriminate between only two among the four Bell states. Three types of $\mathrm{BSM_{phy}}$'s (B_ψ, B_+, and B_-) are considered, which discriminate between $\{|\psi^+\rangle, |\psi^-\rangle\}$, $\{|\phi^+\rangle, |\psi^+\rangle\}$, and $\{|\phi^-\rangle, |\psi^-\rangle\}$, respectively. B_ψ can be implemented by the process in Fig. 42, which can be modified to implement B_+ instead by adding a 45° wave plate on each input line just before the first PBS. If the 90° wave plate on the second input line is removed in the setting for B_+, B_- is executed alternatively. A $\mathrm{BSM_{phy}}$ has four possible outcomes: two successful cases (e.g., for B_ψ, $|\psi^+\rangle$ and $|\psi^-\rangle$), "failure," and "detecting a photon loss." Failure and loss can be distinguished by the number of total photons detected by the photon detectors. Since two photons may enter a single detector, it is assumed that single-photon resolving detectors are used. Note that, even in the failure cases, either sign or letter still can be determined. (For example, even if a B_ψ fails, we can still learn that the letter is ϕ.) On the other hand, if it detects a loss, we can get neither a sign nor a letter.

A $\mathrm{BSM}_{\mathrm{blc}}$ is done by m-times of $\mathrm{BSM}_{\mathrm{phy}}$'s. Each block is composed of m photons, thus we consider m pairs of photons selected respectively in the two blocks. The types of the $\mathrm{BSM}_{\mathrm{phy}}$'s are selected as follows: First, B_ψ is performed on each pair of photons in order until it either succeeds, detects a loss, or consecutively fails j times, where $j \leq m-1$ is a predetermined number. Then a sign $s = \pm$ is selected by the sign of the last B_ψ outcome if it succeeds or selected randomly if it fails or detects a loss. After that, B_s's are performed for all the left pairs of photons.

The block-level sign (letter) is determined by the physical-level signs (letters) of the m $\mathrm{BSM}_{\mathrm{phy}}$'s. In detail, the block-level sign is chosen (i) to be the same as s if the last B_ψ succeeds or any B_s succeeds, and (ii) to be the opposite of s if the last B_ψ does not succeed and any B_s fails. (iii) Otherwise (namely, if the last B_ψ does not succeed and all the B_s's detect losses), the block-level sign is not determined. The block-level letter is determined only when all the physical-level letters are determined, namely, when no losses are detected and all B_s's succeed. For such cases, the block-level letter is ϕ (ψ) if the number of ψ in the $\mathrm{BSM}_{\mathrm{phy}}$ results is even (odd).

Next, a $\mathrm{BSM}_{\mathrm{lat}}$ is done by n-times of $\mathrm{BSM}_{\mathrm{blc}}$'s. The lattice-level sign is determined only when all the block-level signs are determined; it is $(+)$ if the number of $(-)$ in the $\mathrm{BSM}_{\mathrm{blc}}$ results is even and it is $(-)$ if the number is odd. The lattice-level letter is equal to any determined block-level letter. Thus, if all $\mathrm{BSM}_{\mathrm{blc}}$'s cannot determine letters, the lattice-level letter is not determined as well.

4.3.2 Modified CBSM scheme for PTQC

In our PTQC protocol, we consider using either single-photon resolving or on-off detectors. The CBSM scheme should be slightly modified for this case.

Since failure and loss cannot be distinguished, a $\mathrm{BSM}_{\mathrm{phy}}$ now has three possible outcomes: two successful cases and failure. Consequently, in a $\mathrm{BSM}_{\mathrm{blc}}$, B_ψ's are performed until it either succeeds or consecutively fails j times. The way to determine the block-level sign and letter is the same as the original scheme, except that case (iii) when determining the sign no longer occurs. The biggest difference from the original scheme is that the determined sign and letter may be wrong. These error probabilities are presented in the next subsection.

In a $\mathrm{BSM}_{\mathrm{lat}}$, the lattice-level sign is determined from the block-level signs by the same method as the original scheme, although it may be wrong with a nonzero probability as well. On the other hand, the lattice-level letter is not determined by a single block-level letter unlike the original scheme; instead, we use a weighted majority vote of block-level letters. The weight of each block-level letter is given as $w := \log\left[(1-q_{\mathrm{lett}}^{\mathrm{blc}})/q_{\mathrm{lett}}^{\mathrm{blc}}\right]$, where $q_{\mathrm{lett}}^{\mathrm{blc}}$ is the probability that the block-level letter is wrong. This weight factor is justified as follows: Let I_ϕ (I_ψ) denote the set of the indices of block pairs where the block-level letters are ϕ (ψ). Assuming that the two lattice-level

letters (Φ and Ψ) have the same prior probability, we get

$$\begin{aligned}\frac{\Pr\left(\Phi\middle|I_\phi,I_\psi\right)}{\Pr\left(\Psi\middle|I_\phi,I_\psi\right)} &= \frac{\Pr\left(I_\phi,I_\psi\middle|\Phi\right)\Pr\left(\Phi\right)}{\Pr\left(I_\phi,I_\psi\middle|\Psi\right)\Pr\left(\Psi\right)} = \frac{\Pr\left(I_\phi,I_\psi\middle|\Phi\right)}{\Pr\left(I_\phi,I_\psi\middle|\Psi\right)} \\ &= \frac{\prod_{i\in I_\phi}\left(1-q_{\text{lett}}^{(i)}\right)\prod_{i\in I_\psi}q_{\text{lett}}^{(i)}}{\prod_{i\in I_\phi}q_{\text{lett}}^{(i)}\prod_{i\in I_\psi}\left(1-q_{\text{lett}}^{(i)}\right)} \\ &= \prod_{i\in I_\phi}\frac{1-q_{\text{lett}}^{(i)}}{q_{\text{lett}}^{(i)}}\Bigg/\prod_{i\in I_\psi}\frac{1-q_{\text{lett}}^{(i)}}{q_{\text{lett}}^{(i)}} \\ &= \exp\left(\sum_{i=1}^{n}w^{(i)}\right),\end{aligned}$$

where $q_{\text{lett}}^{(i)}$ and $w^{(i)}$ are respectively the letter error probability and the weight of the ith block. Note that the third equality comes from the fact that a lattice-level Bell state is decomposed into block-level Bell states of the same letter, as shown in Eqs. (4.4a) and (4.4b).

4.3.3 Error probabilities of a CBSM under a lossy environment

We now present the possible outcomes of a CBSM using either single-photon resolving or on-off detectors and the corresponding error probabilities $\left(q_{\text{sign}},q_{\text{lett}}\right)$. We denote $x := (1-\eta)^2$, which is the probability that a BSM_{phy} does not detect photon losses. It is assumed that the four Bell states have the same prior probabilities; namely, the initial marginal state on qubits 1 and 2 before suffering losses is the equal mixture of four lattice-level Bell states, which is justified by Proposition 4.1. For a BSM_{blc} or BSM_{lat}, to avoid confusion, we use the term "outcome" to indicate the tuple of the out-

comes of the $\mathrm{BSM}_{\mathrm{phy}}$'s constituting the $\mathrm{BSM}_{\mathrm{blc}}$ or $\mathrm{BSM}_{\mathrm{lat}}$, and use the term "result" to indicate one of the four Bell states that gives the largest posterior probability under its outcome. Note that the result of a BSM may be not deterministically determined by its outcome; if multiple Bell states have the same posterior probability, one of them is randomly selected as the result.

Using single-photon resolving detectors: The case using single-photon resolving detectors is analyzed in Ref. [89] and we here review the contents to be self-contained. The outcome of a $\mathrm{BSM}_{\mathrm{blc}}$ is included in one of the following three cases: (Success) Both the sign and letter are identified if no losses are detected and all the $B_{\pm}$'s succeed. (Failure) Neither sign nor letter is identified if no B_{ψ}'s succeed and all $B_{\pm}$'s detect losses. (Sign discrimination) Only the sign is identified if otherwise. The block-level sign (or letter) is selected randomly if it is not identified. The probabilities of these cases are respectively

$$\begin{cases} \text{Success :} & p_{\mathrm{s}} = \left[1 - 2^{-(j+1)}\right] x^m, \\ \text{Failure :} & p_{\mathrm{f}} = \sum_{l=0}^{j} \left(\frac{x}{2}\right)^l (1-x)^{m-l}, \\ \text{Sign discrimination :} & p_{\mathrm{sd}} = 1 - p_{\mathrm{s}} - p_{\mathrm{f}}. \end{cases}$$

For a $\mathrm{BSM}_{\mathrm{lat}}$, let N_{s} (N_{f}) denote the number of successful (failed) $\mathrm{BSM}_{\mathrm{blc}}$'s. The lattice-level letter is identified if $N_{\mathrm{s}} \geq 1$ (namely, if at least one block-level letter is identified) and the sign is identified if $N_{\mathrm{f}} = 0$ (namely, if all block-level signs are identified). Hence, the outcome of a $\mathrm{BSM}_{\mathrm{lat}}$ is included

in one of the following four events:

$$\begin{cases} S \text{ (Success)}: & N_{\rm s} \geq 1 \wedge N_{\rm f} = 0, \\ D_L \text{ (Letter discrimination)}: & N_{\rm s}, N_{\rm f} \geq 1, \\ D_S \text{ (Sign discrimination)}: & N_{\rm s} = N_{\rm f} = 0, \\ F \text{ (Failure)}: & N_{\rm s} = 0 \wedge N_{\rm f} \geq 1. \end{cases}$$

The sign and letter error probabilities $(q_{\rm sign}, q_{\rm lett})$ of the $\text{BSM}_{\rm lat}$ for each event are $(0,0)$ for S, $(1/2,0)$ for D_L, $(0,1/2)$ for D_S, and $(1/2,1/2)$ for F. The probabilities of the events are respectively given as

$$\begin{aligned} P_S &= (1-p_{\rm f})^n - p_{\rm sd}^n, \\ P_{D_L} &= 1-(1-p_{\rm s})^n + (1-p_{\rm f})^n - p_{\rm sd}^n, \\ P_{D_S} &= p_{\rm sd}^n, \\ P_F &= (1-p_{\rm s})^n - p_{\rm sd}^n. \end{aligned} \tag{4.5}$$

Using on-off detectors: We now consider using on-off detectors for fusions. Each outcome of a $\text{BSM}_{\rm blc}$ is uniquely identified by a triple $O = (r, s, \mathbf{U})$, where $r \in \mathbb{Z}_{j+1} := \{0, \cdots, j\}$ is the number of failed B_ψ's, $s = \pm$ is the sign chosen by the successful $(r+1)$th B_ψ (if $r < j$) or randomly (if $r = j$), and $\mathbf{U}$ is an $(m-r)$-element tuple composed of "ϕ," "ψ," and "f" (failure) indicating the outcomes of the $\text{BSM}_{\rm phy}$'s from the $(r+1)$th to the the last. (If $r < j$, the first component of $\mathbf{U}$ is always ψ, and the other components are determined by the B_s's. If $r = j$, all the components are determined by the B_s's.) Let $N_e(\mathbf{U})$ for $e \in \{\phi, \psi, f\}$ denote the number of e in $\mathbf{U}$. Then

a $\mathrm{BSM}_{\mathrm{blc}}$ outcome O is included in one of the following $j+3$ events:

$$\begin{aligned}
\mathcal{S}_r &:= \big\{(r,s,\mathbf{U})\big|N_f(\mathbf{U})=0\big\} \quad (0\le r\le j),\\
\mathcal{F} &:= \big\{(j,s,\mathbf{U})\big|N_f(\mathbf{U})=m-j\big\},\\
\mathcal{D} &:= O\setminus\left[\mathcal{F}\cup\bigcup_{r=0}^{j}\mathcal{S}_r\right],
\end{aligned} \tag{4.6}$$

where O is the set of all possible outcomes. Note that the events $\mathcal{S}_r$, $\mathcal{F}$, and $\mathcal{D}$ correspond to success, failure, and sign discrimination when $\eta=0$. For each event $\mathcal{E}$ in Eq. (4.6), its sign and letter error probabilities $q^{\mathrm{blc}}_{\mathrm{sign/lett}}(\mathcal{E})$ and the probability $p_{\mathcal{E}}$ that the event occurs are given as follows (see Sec. 4.6.1 for the proof):

$$\begin{cases} q^{\mathrm{blc}}_{\mathrm{sign}}(\mathcal{S}_r)=0, \qquad q^{\mathrm{blc}}_{\mathrm{lett}}(\mathcal{S}_r)=\frac{1}{2}-\frac{1}{2}\big(\frac{x}{2-x}\big)^r,\\ p_{\mathcal{S}_r}=\frac{1}{2}\big(1-\frac{x}{2}\big)^r x^{m-r}, \end{cases}$$
$$\begin{cases} q^{\mathrm{blc}}_{\mathrm{sign}}(\mathcal{F})=\frac{(1-x)^{m-j}}{1+(1-x)^{m-j}}, \qquad q^{\mathrm{blc}}_{\mathrm{lett}}(\mathcal{F})=\frac{1}{2},\\ p_{\mathcal{F}}=\frac{1}{2}\big(1-\frac{x}{2}\big)^j\big[1+(1-x)^{m-j}\big], \end{cases} \tag{4.7}$$
$$\begin{cases} q^{\mathrm{blc}}_{\mathrm{sign}}(\mathcal{D})=0, \qquad q^{\mathrm{blc}}_{\mathrm{lett}}(\mathcal{D})=\frac{1}{2},\\ p_{\mathcal{F}}=1-\sum_r p_{\mathcal{S}_r}-p_{\mathcal{F}}. \end{cases}$$

A possible outcome of a $\mathrm{BSM}_{\mathrm{lat}}$ corresponds to an n-tuple of events composed of $\mathcal{S}_r$ $(0\le r\le j)$, $\mathcal{F}$, and $\mathcal{D}$, which can be regarded as an independent event for the outcomes of the $\mathrm{BSM}_{\mathrm{lat}}$. The probability that an event

$\mathbf{E} = (\mathcal{E}_1, \cdots, \mathcal{E}_n)$ occurs is

$$p_{\mathbf{E}} = \prod_{i=1}^{n} p_{\mathcal{E}_i} \tag{4.8}$$

and the sign and letter error probabilities of $\mathbf{E} = (\mathcal{E}_1, \cdots, \mathcal{E}_n)$ are respectively

$$\begin{aligned} q_{\text{sign}}(\mathbf{E}) &= \frac{1}{2} - \frac{1}{2}\left[1 - 2q_{\text{sign}}^{\text{blc}}(\mathcal{F})\right]^{N_{\mathcal{F}}}, \\ q_{\text{lett}}(\mathbf{E}) &= \frac{1}{2} + \frac{1}{2} \sum_{(\lambda_1, \cdots, \lambda_n) \in \mathbb{Z}_2^n} \prod_{i=1}^{n} \left[q_i^{\lambda_i}(1-q_i)^{1-\lambda_i}\right] \\ &\qquad \times \text{sgn}\left(\sum_{i=1}^{n} (2\lambda_i - 1) \log \frac{1-q_i}{q_i}\right), \end{aligned}$$

where $N_{\mathcal{F}}$ is the number of $\mathcal{F}$'s in $\mathbf{E}$, $q_i := q_{\text{lett}}^{\text{blc}}(\mathcal{E}_i)$, and $\text{sgn}(a)$ is $a/|a|$ if $a \neq 0$ and 0 if $a = 0$. See Sec. 4.6.1 for the proof.

4.4 Comparison with other approaches

We now compare the PTQC protocol with three other known approaches for linear optical quantum computing:

(i) Using single photons for all qubits with fusions assisted by ancillary photons.

(ii) Using simple repetition codes.

(iii) Attaching redundant tree structures to replace a single fusion by multiple fusion attempts.

We show evidence that PTQC is more efficient than these approaches.

4.4.1 Comparison with approach (i)

The first approach uses single photons for all qubits with fusions assisted by ancillary photons [77], which has been widely studied in the context of ballistic quantum computing [85, 86, 78, 87]. In these works, cluster states that differ from RTCSs are used as resources except for Ref. [78]; however, RTCSs should be used to enable a solid error correction, as also mentioned in Refs. [85, 87]. Moreover, in these works, the detrimental effects of failed fusions corrupting nearby qubits are not treated comprehensively; instead, they (except Ref. [78]) regard a fusion failure as removing the corresponding edge and mainly focus on finding percolation thresholds.

Under the noise model described in Sec. 4.2.1, a fusion detects a loss with probability $1-(1-\eta)^2$, if losses in ancillary photons are neglected. Since a marginal state of every Bell state is maximally mixed, detection of a photon loss means complete loss of information; thus, $q_{\mathrm{lett}} = q_{\mathrm{sign}} = 1/2$ in such a case. If losses are not detected, the fusion fails with probability p_{f}, where the letter information of the Bell state still can be obtained [77]; namely, $q_{\mathrm{lett}} = 0$ and $q_{\mathrm{sign}} = 1/2$. These two cases make some central qubits deficient, which can be tracked using the methodology of analyzing nonideal fusions presented in Sec. 4.1.2. HIC is used for the H-configuration to make the failure of a step-1 fusion affects only one central qubit; see Fig. 43.

The photon loss thresholds calculated numerically are plotted in Fig. 56 with theoretical estimations for various values of p_{f}. It shows that p_{f} should be less than about 10% (1%) even if η is only 1% when star clusters are

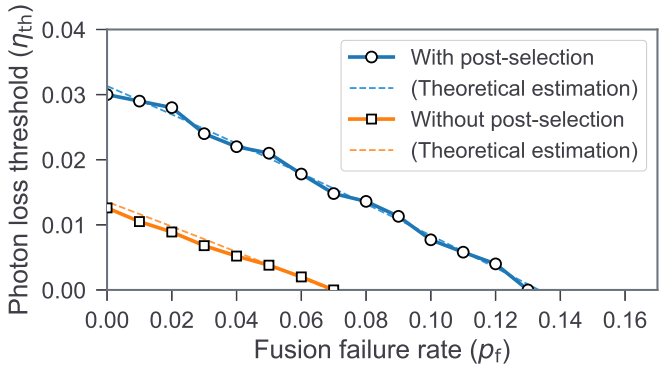

Figure 56: Simulation results for the approach using single-photon qubits with fusions assisted by ancillary photons. It shows the photon loss thresholds η_{th} obtained from simulations or estimated theoretically as a function of the fusion failure rate p_{f}.

(are not) post-selected. Such low fusion failure rates are highly demanding to implement with linear optics due to the requirements of photon-number resolving detectors (PNRDs) that can resolve many photons and ancillary states hard to generate. The failure rate of 10% can be achieved by using the BSM scheme of $N = 3$ in Ref. [77] where $p_{\text{f}} = 6.25\%$. BSM with $N = 3$ requires PNRDs resolving up to 16 photons and the ancillary states $|\Upsilon_1\rangle, |\Upsilon_2\rangle, and\, |\Upsilon_3\rangle$. ($|\Upsilon_j\rangle$ is a 2^j-mode state defined as $|\Upsilon_j\rangle := |2,0,2,0,\cdots,2,0\rangle + |0,2,0,2,\cdots,0,2\rangle$.) It is probably impossible to obtain $|\Upsilon_j\rangle$'s for $j \geq 2$ from single photons with linear optics [77]. Moreover, our simulation does not consider the imperfectness of ancillary states and additional PNRDs; if they are considered, the requirements will be even stricter.

We note that there is a possibility that the lattice renormalization method in Ref. [78] makes the protocol less demanding, which is worth investigating in future works. However, the method has a shortcoming that the renormal-

ized lattice may be significantly smaller than the original lattice; namely, about 20^3 photons are consumed to generate one node [78].

The theoretical estimation in Fig. 56 is done by the following methods: We first assume that star clusters are not post-selected. For a central qubit q to be not deficient, the following conditions should be satisfied simultaneously:

1. Two step-1 fusions in the star cluster containing q succeed.

2. Four step-1 fusions in the four adjacent star clusters (one for each) do not detect losses.

3. Four step-2 fusions involved in the star cluster containing q do not detect losses. Two among them (that make q deficient if they fail) succeed.

4. q itself does not suffer a loss.

From above, we obtain the probability that a central qubit in the final lattice is intact: $p_{\rm int}(\eta, p_{\rm f}) = (1-p_{\rm f})^4(1-\eta)^{21}$. If star clusters are post-selected, the first and second conditions are no longer needed, thus we get $p_{\rm int}(\eta, p_{\rm f}) = (1-p_{\rm f})^2(1-\eta)^9$. Regarding a 50% chance of a Z-error as erasing the qubit by measuring it in the Z-basis (while ignoring the correlation of errors), a photon loss threshold $\eta_{\rm th}$ can be estimated by solving $1-p_{\rm prc} = p_{\rm int}(\eta_{\rm th}, p_{\rm f})$, where $p_{\rm prc} = 0.249$ is the known cubic-lattice bond percolation threshold [50, 96].

4.4.2 Comparison with approach (ii)

We next investigate the approach using simple repetition codes, which is covered in our previous work [3]. In this protocol (called "MTQC"), side qubits are n-photon ones encoded in the basis of $\left\{|\mathrm{H}\rangle^{\otimes n}, |\mathrm{V}\rangle^{\otimes n}\right\}$, where n is a natural number. For central qubits, we first consider using m-photon qubits and then concatenate them with the N-repetition code. That is, we use the basis of $\left\{\left(|\mathrm{H}\rangle^{\otimes m} \pm |\mathrm{V}\rangle^{\otimes m}\right)^{\otimes N}\right\}$ for the central qubits. In Ref. [3], the photon loss thresholds and resource overheads are analyzed in detail, but a rigorous analysis of the effects of nonideal fusions like that done for PTQC is lacking.

Since the n-photon encoding for side qubits is equal to the $(n,1)$ parity encoding, the effects of nonideal fusions can be analyzed in the same way as done for PTQC with the $(n,1)$ parity encoding. The difference between the two is the way that central qubits become deficient due to photon losses in themselves. In PTQC, central qubits are single photons, thus a central qubit becomes deficient with probability η. In MTQC, however, the deficiency rate due to photon losses in central qubits is $[1-(1-\eta)^m]^N$, which decreases exponentially as N increases. The photon loss thresholds recalculated based on these facts are presented in Fig. 57 with the previous values reported in Ref. [3], which shows that the recalculated photon loss thresholds are smaller than the reported values. In particular, it is observed that the central qubit encoding strategy does not improve the thresholds significantly. This discrepancy is because the detrimental effects of nonideal fusion affecting nearby qubits have not been sufficiently rigorously addressed.

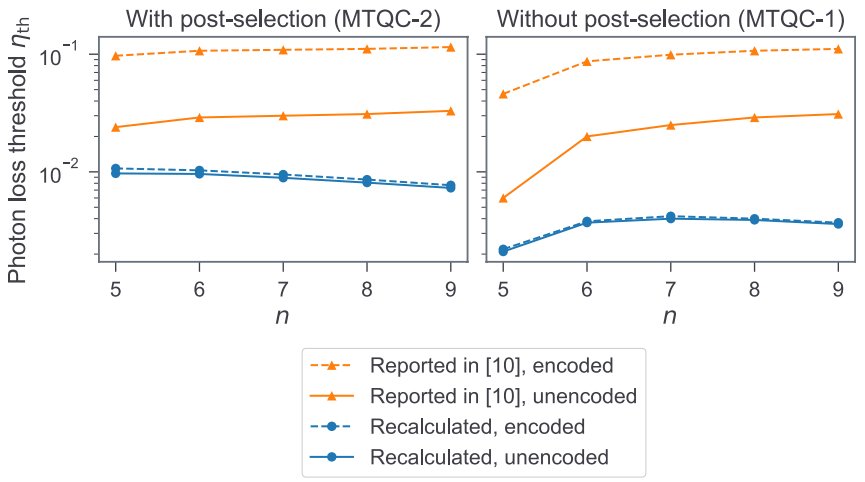

Figure 57: Simulation results for the approach using the simple repetition codes. It shows the photon loss thresholds η_{th} as a function of n for MTQC, which are obtained from Ref. [3] and the recalculation using the methodology for analyzing nonideal fusions. Other parameters are $(m,N) = (2,1)$ and $(m,N) = (2,3)$ for the unencoded and encoded cases, respectively. Two subvariants of MTQC, one with the post-selection of star clusters and the other without it, are considered, which are respectively termed MTQC-2 and MTQC-1 in Ref. [3].

4.4.3 Comparison with approach (iii)

Lastly, we compare PTQC with the approach of (iii) that utilizes redundant tree structures on graph states. Such an approach also has been actively investigated [79, 80, 81], among which Ref. [81] presents the current most advanced version of the protocol where an RTCS is constructed by entangling multiple GHZ-3 states like PTQC. There, at least $\sim 2 \times 10^5$ photodetectors are required per data qubit to achieve a positive photon loss threshold with single-photon resolving detectors, while PTQC requires at least $\sim 7 \times 10^4$ photodetectors per data qubit (see below for the calcula-

tion). Hence, PTQC shows a twofold improvement in resource efficiency compared to the protocol in Ref. [81]. Furthermore, we have shown that PTQC also operates with on-off detectors, while the protocol in Ref. [81] is currently unclear whether it is possible. Nevertheless, further work will be required to compare their performance (especially their fault-tolerance) rigorously and comprehensively.

Conversion of resource measures: In Ref. [81], resource overheads are quantified by the number of photodetectors required per central qubit, not the number of GHZ-3 states we have used, thus conversion between them is necessary for a fair comparison. In PTQC, detectors are used when generating GHZ-3 states, applying physical-level BSMs, and measuring central qubits. We suppose that GHZ-3 states are generated by the scheme proposed in Ref. [93] like the protocol in Ref. [81]. The scheme uses six detectors to generate a single GHZ-3 state and succeeds with probability $1/32$; thus, generating one GHZ-3 state requires 192 detectors. (If it is allowed to use photodetectors repeatedly during the generation of each GHZ-3 state, only six detectors are required per GHZ-3 state. However, we ignore this option to be consistent with Ref. [81].) Next, four detectors are used for one physical-level BSM (see Fig. 42). Counting the number of physical-level BSMs per central qubit is not simple, but we can get its upper bound as $(3N^*_{\mathrm{GHZ}} - 1)/2$, which is half the number of total photons in all GHZ-3 states except one photon in the central qubit. Lastly, two detectors are used for the two polarization modes when measuring a central qubit. In total, $N_{\mathrm{det}} = 198N^*_{\mathrm{GHZ}}$ detectors are required per data qubit in PTQC. Since

$N^*_{\mathrm{GHZ}} \gtrapprox 330$ is required for a positive photon loss threshold (see Fig. 54), N_{det} should be at least about 7×10^4.

4.5 Remarks

In this chapter, we address the problem of overcoming the negative effects of nonideal fusions and photon losses during linear-optical measurement-based quantum computing (MBQC). We first introduced a Bayesian methodology for tracking errors caused by nonideal fusions during the construction of graph states, which enables accurate and effective error simulations. We then proposed the **parity-encoding-based topological quantum computing** (PTQC) protocol that uses the parity encoding and concatenated Bell-state measurement, which turns out to have a high loss threshold of at most $\sim 8.5\%$. Moreover, logical error rates near 10^{-6} can be achieved using about 10^6 or fewer three-photon Greenberger-Horne-Zeilinger states (GHZ-3) states in total when the photon loss rate is 1%, which outperforms other known linear optical computing protocols [3]. We presented comprehensive and systematic methods to construct a graph state from GHZ-3 states, including the graph-theoretical algorithm that can minimize the resource overhead efficiently.

Additionally, we investigated three other known approaches that respectively use single-photon qubits with fusions assisted by ancillary photons, simple repetition codes, and redundant tree graphs. We verified that the first two are highly demanding compared to PTQC due to low photon loss thresholds or hard-to-implement requirements such as photodetectors

that can resolve many photons. Compared to the third approach, we showed that PTQC has a twofold improvement in terms of the resource overhead required for the loss threshold to be positive, although additional work will be necessary to compare their fault-tolerance as well.

One may apply the Bayesian error tracking method to other encoding schemes or decoding algorithms (such as the union-find decoder [97]) to improve fault-tolerance or resource overheads. More careful consideration of component-wise errors, including both heralded photon losses and unheralded errors (such as dark counts on photodetectors), shall give rise to more realistic analyses. Resource analysis will be more comprehensive if other factors such as the number of optical switches or the lengths of delay lines are considered. In particular, one trial of CBSM may require optical switches to change the types (B_Ψ, B_+, and B_-) of the physical-level BSMs. Our graph-theoretical optimization scheme for generating graph states can be applied to arbitrary graph states as well as microclusters for PTQC. It will be interesting future work to investigate the resource reduction effect of this scheme for various MBQC protocols or other applications of graph states such as quantum repeaters. Lastly, our methods may be generalized to fusion-based quantum computing [98] that is attracting attention recently, or other MBQC protocols such as the color-code-based one [37].

4.6 Appendix

4.6.1 Calculation of the error probabilities of a CBSM when on-off detectors are used

We here derive the error probabilities of a CBSM on two qubits (say, qubits 1 and 2) encoded with the parity encoding when on-off detectors are used for fusions. We denote $x := (1-\eta)^2$, which is the probability that a $\mathrm{BSM}_{\mathrm{phy}}$ does not detect photon losses.

Block-level BSM ($\mathrm{BSM}_{\mathrm{blc}}$)

We note that every positive operator-valued measure (POVM) element of a lossy $\mathrm{BSM}_{\mathrm{phy}}$ has vanishing off-diagonal entries in the Bell basis; see Sec. 4.6.2 for the proof. Also, each POVM element of a lossy $\mathrm{BSM}_{\mathrm{blc}}$, denoted by M_O^{blc} for each outcome $O = (r, s, \mathbf{U})$, is the tensor product of particular POVM elements of the lossy $\mathrm{BSM}_{\mathrm{phy}}$'s constituting the $\mathrm{BSM}_{\mathrm{blc}}$. Thus, the conditional probability of getting O from a block-level Bell state $|B\rangle$ is

$$\Pr(O|B) = \langle B| M_O^{\mathrm{blc}} |B\rangle = \frac{1}{2^{m-1}} \sum_i \langle B_i| M_O^{\mathrm{blc}} |B_i\rangle = \frac{1}{2^{m-1}} \sum_i \Pr(O|B_i),$$

where $|B_i\rangle$'s are the terms constituting the summation in Eqs. (4.4c) and (4.4d), namely, $|B\rangle = \frac{1}{\sqrt{2^{m-1}}} \sum_i |B_i\rangle$. In other words, when calculating $\Pr(O|B)$, it is enough to find $\Pr(O|B_i)$'s and then take their average.

The posterior probability of a block-level Bell state $|B\rangle$ under a given

outcome O is

$$\Pr(B|O) = \frac{\Pr(O|B)}{\sum_{|B'\rangle\in\mathcal{B}_{\rm blc}} \Pr(O|B')}, \tag{4.9}$$

where $\mathcal{B}_{\rm blc}$ is the set of the four block-level Bell states. Thus, the result of the $\mathrm{BSM}_{\rm blc}$ is selected randomly in the set $R(O) := \mathrm{argmax}_B \Pr(O|B)$. The sign (letter) error probability as a function of O is

$$q^{\rm blc}_{\rm sign(lett)}(O) = \frac{1}{|R(O)|} \sum_{|B\rangle\in R(O)} \left[\Pr\left(F_{\rm sign(lett)}(B)\middle|O\right) + \Pr\left(F_{\rm sign}\circ F_{\rm lett}(B)\middle|O\right)\right], \tag{4.10}$$

where $\left|F_{\rm sign(lett)}(B)\right\rangle$ is the Bell state obtained by flipping the sign (letter) from $|B\rangle$ (e.g., $F_{\rm sign}(\phi^{\pm}) = \phi^{\mp}$).

Block-level outcomes can be grouped by the $j+3$ events $\mathcal{S}_r$ $(r = 0, \cdots, j)$, $\mathcal{F}$, and $\mathcal{D}$, as defined in Eq. (4.6). We now calculate the probability that each event occurs and the corresponding sign and letter error probabilities. Let us first consider an outcome $O = (r, s, \mathbf{U}) \in \mathcal{S}_r$, where $N_{\rm f}(\mathbf{U}) = 0$. Regarding a single term in the decomposition of $\left|\phi^{\pm}_{(m)}\right\rangle$ [see Eq. (4.4c)], if there are total k of $|\psi^{\pm}\rangle$'s, the first r physical levels contain $k - N_{\psi}(\mathbf{U})$ of $|\psi^{\pm}\rangle$'s, which should suffer photon losses by the definition of r. If $r < j$, s selected by the successful $(r+1)$th B_{ψ} is certainly $\pm$, the sign of $\left|\phi^{\pm}_{(m)}\right\rangle$. If $r = j$, the randomly selected s may or may not be corrected; however, the latter case is out of $\mathcal{S}_r$ since all the following $B_{\mp}$'s must fail. The remaining $m - r$ $\mathrm{BSM}_{\rm phy}$'s should not suffer photon losses since $N_f(\mathbf{U}) = 0$. Hence, for all $\mathbf{U}$ satisfying

$N_{\rm f}(\mathbf{U})=0$, we get

$$\begin{aligned}
\Pr\left(r,\pm,\mathbf{U}\middle|\phi^{\pm}_{(m)}\right) &= \frac{1}{2^{m-1}}\sum_{k:\text{even}\le r+N_\psi}\binom{r}{k-N_\psi}(1-x)^{k-N_\psi}\frac{1}{2^{\delta_{rj}}}x^{m-r}\\
&= \frac{1}{2^{\delta_{rj}}}\left[\left(1-\frac{x}{2}\right)^r\left(\frac{x}{2}\right)^{m-r}+(-1)^{N_\psi}\left(\frac{x}{2}\right)^m\right],\\
\Pr\left(r,\mp,\mathbf{U}\middle|\phi^{\pm}_{(m)}\right) &= 0,
\end{aligned} \tag{4.11}$$

where $N_\psi = N_\psi(\mathbf{U})$. Similarly, we get

$$\begin{aligned}
\Pr\left(r,\pm,\mathbf{U}\middle|\psi^{\pm}_{(m)}\right) &= \frac{1}{2^{\delta_{rj}}}\left[\left(1-\frac{x}{2}\right)^r\left(\frac{x}{2}\right)^{m-r}-(-1)^{N_\psi}\left(\frac{x}{2}\right)^m\right],\\
\Pr\left(r,\mp,\mathbf{U}\middle|\psi^{\pm}_{(m)}\right) &= 0.
\end{aligned} \tag{4.12}$$

From Eqs. (4.9)–(4.12), we obtain

$$\begin{aligned}
&q^{\rm blc}_{\rm sign}(O)=0=:q^{\rm blc}_{\rm sign}(\mathcal{S}_r),\\
&q^{\rm blc}_{\rm lett}(O)=\frac{\left(1-\frac{x}{2}\right)^r\left(\frac{x}{2}\right)^{m-r}-\left(\frac{x}{2}\right)^m}{2\left(1-\frac{x}{2}\right)^r\left(\frac{x}{2}\right)^{m-r}}=\frac{1}{2}\left[1-\left(\frac{x}{2-x}\right)^r\right]=:q^{\rm blc}_{\rm lett}(\mathcal{S}_r).
\end{aligned} \tag{4.13}$$

Note that the error probabilities are the same for all $O\in\mathcal{S}_r$. The total probability that the event $\mathcal{S}_r$ occurs is

$$\begin{aligned}
p_{\mathcal{S}_r} :=& \frac{1}{4}\sum_{O\in\mathcal{S}_r}\sum_{|B\rangle\in\mathcal{B}_{\rm blc}}\Pr(O|B)\\
=& \frac{1}{2}\frac{1}{2^{\delta_{rj}}}\left(1-\frac{x}{2}\right)^r\left(\frac{x}{2}\right)^{m-r}2^{m-r-1+\delta_{rj}}\\
=& \frac{1}{2}\left(1-\frac{x}{2}\right)^r x^{m-r},
\end{aligned}$$

where the factor $2^{m-r-1+\delta_{rj}}$ is the number of possible $\mathbf{U}$'s for a given value of r.

Next, we consider $O=(j,s,\mathbf{U})\in\mathcal{F}$, where $N_f(\mathbf{U})=m-j$, namely, all the B_s's fail. Regarding a single term in the decomposition of $\left|\phi^{\pm}_{(m)}\right\rangle$, all the $|\psi^{\pm}\rangle$'s in the first j physical levels should suffer photon losses. If $s=\pm$, all the following $B_{\pm}$'s should suffer photon losses as well. If $s=\mp$, all the following $B_{\mp}$'s fail regardless of photon losses. We thus get

$$\begin{aligned}
\Pr\left(j,\pm,\mathbf{U}=(f,\cdots,f)\middle|\phi^{\pm}_{(m)}\right) &= \frac{1}{2^{m-1}} \sum_{\substack{0\le k_1\le j\\ 0\le k_2\le m-j\\ k_1+k_2\text{: even}}} \binom{j}{k_1}\binom{m-j}{k_2}(1-x)^{k_1+m-j}\cdot\frac{1}{2} \\
&= \frac{1}{2}\left(1-\frac{x}{2}\right)^{j}(1-x)^{m-j}, \\
\Pr\left(j,\mp,\mathbf{U}=(f,\cdots,f)\middle|\phi^{\pm}_{(m)}\right) &= \frac{1}{2}\left(1-\frac{x}{2}\right)^{j},
\end{aligned}$$

where k_1 (k_2) in the summation indicates the number of $|\psi^{\pm}\rangle$'s in the first j (last $m-j$) physical levels. Similarly, the same results are obtained for $\left|\psi^{\pm}_{(m)}\right\rangle$:

$$\begin{aligned}
\Pr\left(j,\pm,\mathbf{U}=(f,\cdots,f)\middle|\psi^{\pm}_{(m)}\right) &= \frac{1}{2}\left(1-\frac{x}{2}\right)^{j}(1-x)^{m-j}, \\
\Pr\left(j,\mp,\mathbf{U}=(f,\cdots,f)\middle|\psi^{\pm}_{(m)}\right) &= \frac{1}{2}\left(1-\frac{x}{2}\right)^{j}.
\end{aligned}$$

The corresponding error probabilities are

$$q^{\mathrm{blc}}_{\mathrm{sign}}(O)=\frac{(1-x)^{m-j}}{1+(1-x)^{m-j}}=:q^{\mathrm{blc}}_{\mathrm{sign}}(\mathcal{F}),\qquad q^{\mathrm{blc}}_{\mathrm{lett}}(O)=\frac{1}{2}=:q^{\mathrm{blc}}_{\mathrm{lett}}(\mathcal{F})$$

and the total probability of the event $\mathcal{F}$ is

$$p_{\mathcal{F}} = \frac{1}{4}\sum_{s=\pm}\sum_{|B\rangle\in\mathcal{B}_{\mathrm{blc}}} \Pr(j,s,\mathbf{U}=(f,\cdots,f)|B) = \frac{1}{2}\left(1-\frac{x}{2}\right)^{j}\left[1+(1-x)^{m-j}\right].$$

Lastly, we consider $O=(r,s,\mathbf{U})\in\mathcal{D}$. If $r<j$, $N_f(\mathbf{U})>0$ by the definition of $\mathcal{D}$ and $N_f(\mathbf{U})<m-r$ since the first component of $\mathbf{U}$ is always ψ. If $r=j$, $0<N_f(\mathbf{U})<m-j$ by the definition of $\mathcal{D}$. Therefore, regardless of r, $\mathbf{U}$ contains at least one failure and one success (ψ or ϕ). Thanks to the successful $\mathrm{BSM}_{\mathrm{phy}}$'s, the sign of the result is identified without an error. On the other hand, the letter is not identified because of the failures. We can see intuitively without calculation that the letter error probability is $1/2$: Even if there is only one failure in $\mathbf{U}$, the letter information of the corresponding physical-level Bell state is completely lost, considering that the marginal state of a block-level Bell state on a single physical level is $|\phi^{\pm}\rangle\langle\phi^{\pm}|+|\psi^{\pm}\rangle\langle\psi^{\pm}|$. Thus, the block-level letter information (determined by the parity of the number of $\mathrm{BSM}_{\mathrm{phy}}$ outcomes with ψ) is completely lost as well. To rewrite the results, we get

$$q^{\mathrm{blc}}_{\mathrm{sign}}(\mathcal{D})=0, \qquad q^{\mathrm{blc}}_{\mathrm{lett}}(\mathcal{D})=\frac{1}{2}, \qquad p_{\mathcal{D}}=1-\sum_{r=0}^{j}p_{\mathcal{S}_r}-p_{\mathcal{F}}.$$

Lattice-level BSM ($\mathrm{BSM}_{\mathrm{lat}}$)

Each n-tuple of events composed of $\mathcal{S}_r$ ($0\le r\le j$), $\mathcal{F}$, and $\mathcal{D}$ corresponds to a set of possible outcomes of a $\mathrm{BSM}_{\mathrm{lat}}$. Let us consider such an n-tuple $\mathbf{E}=(\mathcal{E}_1,\cdots,\mathcal{E}_n)$. A lattice-level sign error occurs when there is an odd number of block-level sign errors and $\mathcal{F}$ is the only event where a

block-level sign error may occur; thus, the sign error probability is

$$q_{\text{sign}} = \sum_{i:\text{odd}\le N_{\mathcal{F}}} \binom{N_{\mathcal{F}}}{i} q_{\text{sign}}^{\text{blc}}(\mathcal{F})^i \left[1 - q_{\text{sign}}^{\text{blc}}(\mathcal{F})\right]^{N_{\mathcal{F}}-i} = \frac{1}{2} - \frac{1}{2}\left[1 - 2q_{\text{sign}}^{\text{blc}}(\mathcal{F})\right]^{N_{\mathcal{F}}},$$

where $N_{\mathcal{F}}$ is the number of $\mathcal{F}$'s in $\mathbf{E}$.

A lattice-level letter error occurs when the weighted majority vote of the block-level letters gives a wrong answer. We consider i.i.d. random variables $\Lambda_1, \cdots, \Lambda_n$ such that $\Lambda_i \sim \text{Bernoulli}(q_i)$ for each i where $q_i := q_{\text{lett}}^{\text{blc}}(\mathcal{E}_i)$, which indicates whether a letter error occurs in the ith block. A lattice-level letter error occurs if

$$\sum_i (2\Lambda_i - 1) \log \frac{1-q_i}{q_i} =: V(\Lambda_1, \cdots, \Lambda_n)$$

is larger than zero or if it is equal to zero and the randomly selected letter is wrong. Therefore, we get

$$\begin{aligned}
q_{\text{lett}} &= \Pr\left(V(\Lambda_1, \cdots, \Lambda_n) > 0\right) + \frac{1}{2}\Pr\left(V(\Lambda_1, \cdots, \Lambda_n) = 0\right) \\
&= \sum_{(\lambda_1, \cdots, \lambda_n) \in \mathbb{Z}_2^n} \prod_{i=1}^n \Pr(\Lambda_i = \lambda_i) \left\{ \Theta[V(\lambda_1, \cdots, \lambda_n) > 0] + \frac{1}{2}\Theta[V(\lambda_1, \cdots, \lambda_n) = 0] \right\} \\
&= \sum_{(\lambda_1, \cdots, \lambda_n) \in \mathbb{Z}_2^n} \prod_{i=1}^n \left[q_i^{\lambda_i} (1-q_i)^{1-\lambda_i} \right] \left[\frac{1}{2} \operatorname{sgn}\left(V(\lambda_1, \cdots, \lambda_n)\right) + \frac{1}{2} \right], \\
&= \frac{1}{2} + \frac{1}{2} \sum_{(\lambda_1, \cdots, \lambda_n) \in \mathbb{Z}_2^n} \prod_{i=1}^n \left[q_i^{\lambda_i} (1-q_i)^{1-\lambda_i} \right] \operatorname{sgn}\left(\sum_{i=1}^n (2\lambda_i - 1) \log \frac{1-q_i}{q_i} \right),
\end{aligned}$$

where $\Theta[C]$ for a condition C is equal to 1 if C is true and 0 if it is false, and $\operatorname{sgn}(a)$ is $a/|a|$ if $a \neq 0$ and 0 if $a = 0$.

4.6.2 Proof of vanishing off-diagonal entries of the POVM elements of a lossy physical-level BSM

We here prove that every POVM element of a lossy $\mathrm{BSM}_{\mathrm{phy}}$ in a CBSM with on-off detectors has vanishing off-diagonal entries in the Bell basis. Let Λ_η denote the photon loss channel of a loss rate η defined as $\Lambda_\eta(\sigma) := (1-\eta)\sigma + \eta\,|0\rangle\langle 0|$ for a single-photon state σ and the vacuum state $|0\rangle$. By substituting $\sigma = |\psi\rangle\langle\psi|$ for an arbitrary pure state $|\psi\rangle = \alpha\,|\mathrm{H}\rangle + \beta\,|\mathrm{V}\rangle$, we get $\Lambda_\eta(|H\rangle\langle V|) = (1-\eta)\,|H\rangle\langle V|$. Thus,

$$
\begin{aligned}
(\Lambda_\eta \otimes \Lambda_\eta)(|\phi^\pm\rangle\langle\psi^\pm|) =& (1-\eta)^2\,|\phi^\pm\rangle\langle\psi^\pm| \\
&+ \eta(1-\eta)(|H0\rangle\langle V0| + |V0\rangle\langle H0| + |0H\rangle\langle 0V| + |0V\rangle\langle 0H|), \\
(\Lambda_\eta \otimes \Lambda_\eta)(|\phi^\pm\rangle\langle\psi^\mp|) =& (1-\eta)^2\,|\phi^\pm\rangle\langle\psi^\mp| \\
&+ \eta(1-\eta)(|H0\rangle\langle V0| - |V0\rangle\langle H0| - |0H\rangle\langle 0V| + |0V\rangle\langle 0H|), \\
(\Lambda_\eta \otimes \Lambda_\eta)(|\phi^+\rangle\langle\phi^-|) =& (1-\eta)^2\,|\phi^+\rangle\langle\phi^-| \\
&+ \eta(1-\eta)(|H0\rangle\langle H0| - |V0\rangle\langle V0| + |0H\rangle\langle 0H| - |0V\rangle\langle 0V|), \\
(\Lambda_\eta \otimes \Lambda_\eta)(|\psi^+\rangle\langle\psi^-|) =& (1-\eta)^2\,|\psi^+\rangle\langle\psi^-| \\
&+ \eta(1-\eta)(|H0\rangle\langle H0| - |V0\rangle\langle V0| - |0H\rangle\langle 0H| + |0V\rangle\langle 0V|).
\end{aligned}
\tag{4.14}
$$

Now, let $M_\pm$ and M_f denote the POVM elements of a lossy B_ψ corresponding to the outcomes $|\psi^\pm\rangle$ and failure, respectively. By modelling a lossy B_ψ as a photon loss channel followed by an ideal B_ψ, M_i for $i \in \{+,-,f\}$

satisfies

$$\mathrm{Tr}[M_i\rho] = \mathrm{Tr}[\Pi_i(\Lambda_\eta \otimes \Lambda_\eta)(\rho)], \tag{4.15}$$

for any two-qubit state ρ, where

$$\Pi_+ := \left|\psi^+\right\rangle\!\left\langle\psi^+\right|, \qquad \Pi_- := \left|\psi^-\right\rangle\!\left\langle\psi^-\right|,$$
$$\Pi_f := |\phi_+\rangle\langle\phi_+| + |\phi_-\rangle\langle\phi_-| + |H0\rangle\langle H0| + |V0\rangle\langle V0| + |0H\rangle\langle 0H| + |0V\rangle\langle 0V| + |00\rangle\langle 00|$$

are the projectors of an ideal B_ψ with a lossy input. From Eqs. (4.14) and (4.15), we obtain $\langle\psi^\pm|M_i|\phi^\pm\rangle = \langle\psi^\pm|M_i|\phi^\mp\rangle = \langle\phi^+|M_i|\phi^-\rangle = \langle\psi^+|M_i|\psi^-\rangle = 0$ for every $i \in \{+,-,f\}$. Similar arguments can be done for a lossy B_+ and B_- as well.

4.7 Derivation of the physical-level graphs of post-H microclusters

Here we derive the physical-level graph structures of the central and side post-H microclusters for the two H-configurations, which are shown in Fig. 45. The first step of the derivation is to investigate how a graph state is transformed if a Hadamard gate (H_1) is applied on one of the qubits (say, qubit 1) and then a CZ gate (C_{12}^Z) is applied on qubit 1 and another qubit (say, qubit 2) that is not adjacent to qubit 1. Note that, in the Heisenberg picture, the CZ gate transforms the Pauli-X operators of the qubits as $X_1 \to X_1Z_2$ and $X_2 \to Z_1X_2$, while it leaves the Pauli-Z operators the same. For a qubit i, $g_i := X_i\prod_{j\in N(i)} Z_j$, where $N(i)$ is the set of qubits adjacent to

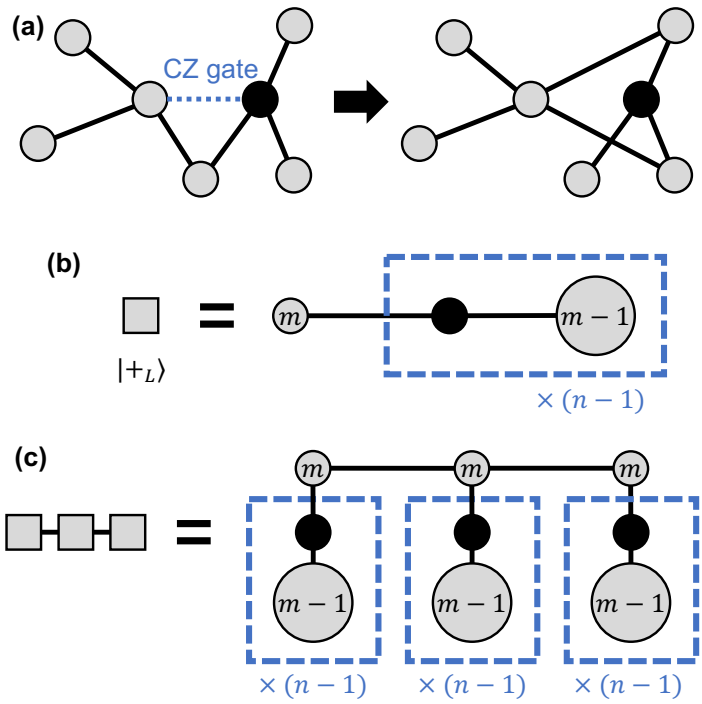

Figure 58: **(a)** Transformation of a graph state by applying a Hadamard gate followed by applying a CZ gate. **(b)** Physical-level graph structure of the state $|+_L\rangle = |0_L\rangle + |1_L\rangle$. **(c)** Physical-level graph structure of a lattice-level three-qubit linear graph state.

qubit i, is a stabilizer of the initial graph state. The stabilizers S_1 and S_1S_2 are transformed by $C_{12}^Z H_1$ as

$$g_1 = X_1 \prod_{j\in N(1)} Z_j \longrightarrow Z_1 \prod_{j\in N(1)} Z_j = H_1 \left(X_1 \prod_{j\in N(1)} Z_j \right) H_1,$$

$$g_1 g_2 = X_1 X_2 \prod_{j\in N(1)\triangle N(2)} Z_j \longrightarrow X_2 \prod_{j\in N(1)\triangle N(2)} Z_j = H_1 \left(X_2 \prod_{j\in N(1)\triangle N(2)} Z_j \right) H_1,$$

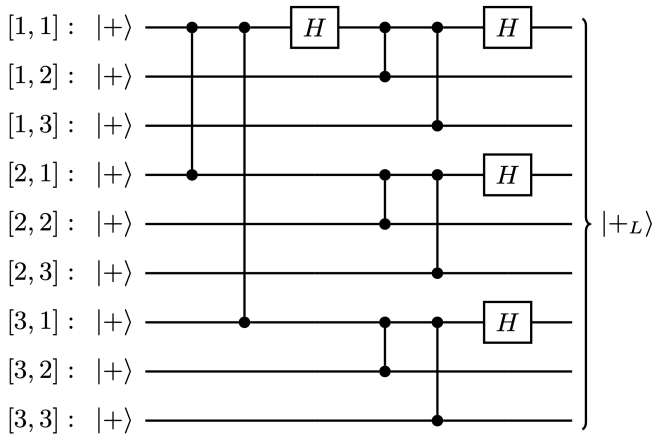

Figure 59: Encoding circuit of the state $|+_L\rangle := |0_L\rangle + |1_L\rangle$ in the $(3,3)$ parity encoding. It employs multiple copies of the state $|+\rangle := |\mathrm{H}\rangle + |\mathrm{V}\rangle$, CZ gates, and Hadamard gates. The label $[i,j]$ for each physical-level qubit indicates the index i of the block and the index j of the photon in the block.

where $A \triangle B := A \cup B \setminus (A \cap B)$ for two sets A and B. Also, for each qubit $i \in N(1)$,

$$g_i = X_i \prod_{j \in N(i)} Z_j \longrightarrow X_i X_1 Z_2 \prod_{j \in N(i) \setminus \{1\}} Z_j = H_1 \left(X_i \prod_{j \in N(i) \triangle \{2\}} Z_j \right) H_1.$$

Therefore, the overall effect of the process is, for each qubit i adjacent to qubit 1, to flip the connectivity of the qubits 2 and i (namely, connect them if they are disconnected and disconnect them if they are already connected) and then apply H_1. An example of this transformation is presented in Fig. 58(a).

Next, we obtain the graph structure of the state $|+_L\rangle := |0_L\rangle + |1_L\rangle$. Fig. 59 shows the encoding circuit of the state for the $(3,3)$ parity encoding, which employs multiple copies of the state $|+\rangle := |\mathrm{H}\rangle + |\mathrm{V}\rangle$, CZ gates,

and Hadamard gates. Here, we label the jth physical qubit of the ith block by $[i, j]$. It is straightforward to generalize it for any pair of (n, m) The graph structure of $|+_L\rangle$ shown in Fig. 58(b) is obtained by preparing nm isolated vertices and tracking the transformation of the graph via the CZ and Hadamard gates in the circuit.

A lattice-level CZ gate C_L^Z is done by m^2 physical CZ gates:

$$C_L^Z = \prod_{i,j \le m} C^Z_{1i,1j}, \tag{4.16}$$

where $C^Z_{ij,kl}$ is the CZ gate between the $[i, j]$ qubit of the first lattice-level qubit and the $[k, l]$ qubit of the second lattice-level qubit. It can be verified as follows: The stabilizer generators of the (n, m) parity encoding are

$$\left\{ X_{ij}X_{i(j+1)} \ \ (\forall i \le n,\ \forall j \le m-1), \quad \prod_{j=1}^{m} Z_{ij}Z_{(i+1)j} \ \ (\forall i \le n-1) \right\}$$

and the lattice-level Pauli operators are

$$X_L = X_{11} \cdots X_{n1}, \qquad Z_L = Z_{11} \cdots Z_{1m},$$

where X_{ij} (Z_{ij}) is the Pauli-X (-Z) operator on the $[i, j]$ qubit. It is straightforward to see that the RHS of Eq. (4.16) commutes with all the stabilizers and transforms the lattice-level Pauli operators correctly.

Combining the above results on the $|+_L\rangle$ state and the lattice-level CZ gate, we attain the graph structure of a lattice-level three-qubit linear graph state shown in Fig. 58(c). The only left ingredient is the lattice-level

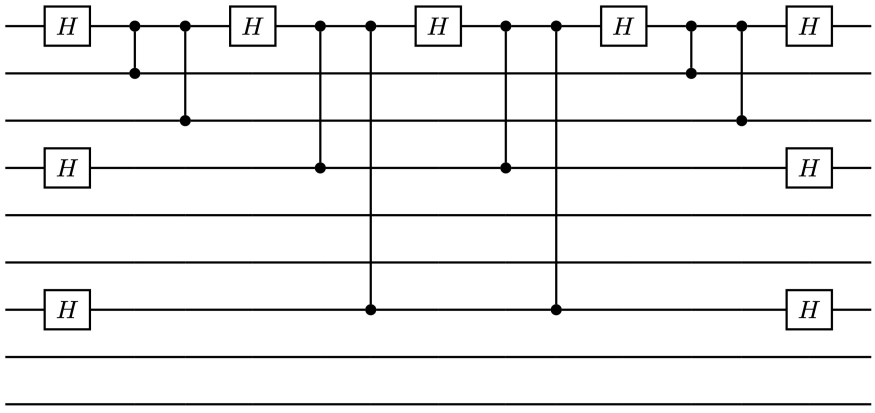

Figure 60: Circuit to implement the lattice-level Hadamard gate of the (3, 3) parity encoding.

Hadamard gate (H_L). The circuit for H_L is obtained by simply connecting the decoding circuit, the physical Hadamard gate, and the encoding circuit, which is explicitly shown in Fig. 60 for the (3, 3) parity encoding. By transforming the graph in Fig. 58(c) with appropriate lattice-level Hadamard gates, we finally get the desired graph structures of the post-H microclusters shown in Fig. 45. Note that, for central microclusters, the middle lattice-level qubits are replaced with unencoded physical-level qubits.

4.7.1 Details of error simulations

Here we describe the error simulation method in detail. We first introduce the parameters that determine the details of PTQC:

- `pssl`: If `True`, star clusters generated by successful step-1 fusions are post-selected for step 2. If `False`, all generated star clusters are used regardless of the fusion results.

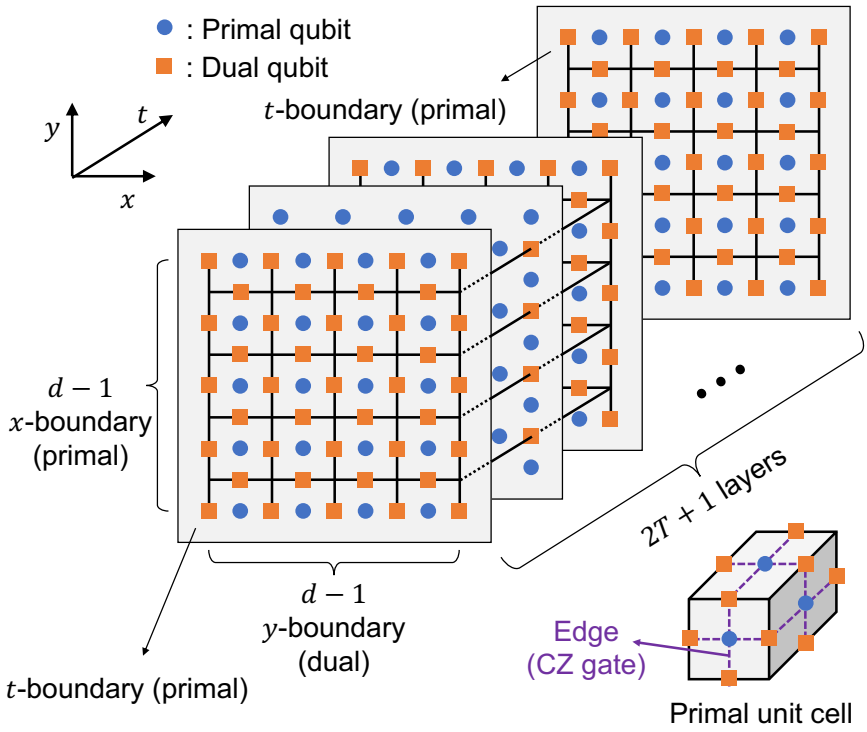

Figure 61: Structure of a logical identity gate for simulations where the code distance is $d = 5$ and the length along the simulated time (t) axis is T in the unit of a cell.

- `hic`: If `True`, the H-configuration is HIC. If `False`, it is HIS.
- `sprd`: If `True`, single-photon resolving detectors are used. If `False`, on-off detectors are used.
- n, m: The (n,m) parity encoding is used to encode side qubits.
- j: The maximal number of B_ψ's in a $\mathrm{BSM}_{\mathrm{blc}}$. (See the CBSM scheme in Sec. 4.3)

For a fixed parameter setting, we consider an RTCS lattice whose bound-

aries are in the form of a cuboid as visualized in Fig. 61, which implements a logical identity gate. Let us term the three axes of the cuboid as the x-, y-, and t-axis and the corresponding boundaries as the x-, y-, and t-boundaries. The t-axis is also referred to as the **simulated time axis**. The cuboid has the widths of $d-1$ unit cells along the x- and y-axis, where d is the code distance, and the width of $T = 4d+1$ unit cells along the t-axis. The value of T is arbitrarily set to be larger enough than d for reducing the effects of errors near the t-boundaries. The x- and t-boundaries are set to be primal, while the y-boundaries are set to be dual. In other words, the x- and t-boundaries adjoin normally on primal unit cells, while the y-boundaries cross the middle of primal unit cells. For error simulations, we count error chains connecting the opposite x-boundaries, thus we assume that the qubits on the t-boundaries do not have errors.

We use a Monte-Carlo method for the simulations. Each trial proceeds as follows:

1. Sample the outcomes of all fusions in steps 1 and 2 (only step 2 if `pssl` is `True`) by the probabilities shown in Sec. 4.3, which depend on the values of n, m, j, and η.

2. For each fusion outcome, the corresponding error probabilities (q_{sign}, q_{lett}) are obtained and whether the fusion has a sign or letter error is randomly determined by the probabilities. These error probabilities and errors are then propagated to appropriate central qubits determined by the value of `hic`. For each central qubit i, the presence or absence of an error and its probability are assigned to a boolean

variable $\texttt{error}_i$ and a floating-point variable $q_{\mathrm{err},i}$, respectively.

3. For each central qubit i, a photon loss is sampled with probability η. If it has a loss, $q_{\mathrm{err},i}$ is updated to 0.5 and $\texttt{error}_i$ is flipped with probability 50%.

4. The syndrome of each parity-check operator (which corresponds to a primal unit cell) is determined by the values of $\texttt{error}_i$'s of the qubits in the support of the operator.

5. The syndromes are decoded to infer the locations of the errors. We use the weighted minimum-weight perfect matching decoder via PyMatching package [95] where the weight for each qubit i is $\log[(1-q_{\mathrm{err},i})/q_{\mathrm{err},i}]$. (If $q_{\mathrm{err},i}=0$, the weight is infinity, which is handled by ignoring the qubit from the input of the decoder.) Exceptionally, if every value of $q_{\mathrm{err},i}$ is either 0 or $1/2$, the qubits with $q_{\mathrm{err},i}=1/2$ are given the weight of one, not zero, for a technical reason.

6. The remaining errors are obtained by comparing the original and estimated errors. If the number of the remaining errors on one side of the x-boundaries is odd, we regard that this trial has a logical error.

The logical error rate p_L for a given parameter setting is obtained by repeating the above process a sufficient number of times. In detail, we repeat the process until $\Delta p_L/p_L \leq 0.1$ is reached where Δp_L is half the width of the 99% confidence interval. The logical error rates $p_L^{(9)}(\eta)$, $p_L^{(11)}(\eta)$ are calculated while varying η for two code distances $d=9,\ 11$ and the loss threshold η_{th} is obtained by finding the largest η satisfying $p_L^{(11)}(\eta)+\Delta p_L^{(11)}(\eta) <$

$p_L^{(9)}(\eta) - \Delta p_L^{(9)}(\eta)$.

4.7.2 Details of resource analysis

Here, we describe the details of resource analysis on PTQC. We first investigate calculating N^*_{GHZ}, the expected number of required GHZ-3 states to generate one star cluster. N^*_{GHZ} is used to obtain the expected total number $\mathcal{N}_{p_L^{\mathrm{targ}}}$ of GHZ-3 states to achieve the target logical error rate of p_L^{targ} for the logical identity gate with the length of $d-1$ unit cells.

By using the optimization method presented in Sec. 4.2.2, we determine the merging graphs and the orders of the merging operations for center and side post-H microclusters and calculate their resource overheads $N^{\mathrm{central}}_{\mathrm{GHZ}}$ and $N^{\mathrm{side}}_{\mathrm{GHZ}}$. Then we get

$$N^*_{\mathrm{GHZ}} = \begin{cases} \left[\left(N^{\mathrm{central}}_{\mathrm{GHZ}} + N^{\mathrm{side}}_{\mathrm{GHZ}}\right)/p_{\mathrm{succ,step1}} + N^{\mathrm{side}}_{\mathrm{GHZ}}\right]/p_{\mathrm{succ,step1}} & \text{if } \texttt{pssl} \text{ is } \texttt{True}, \\ N^{\mathrm{central}}_{\mathrm{GHZ}} + 2N^{\mathrm{side}}_{\mathrm{GHZ}} & \text{if } \texttt{pssl} \text{ is } \texttt{False}, \end{cases}$$

where $p_{\mathrm{succ,step1}}$ is the average success probability of step-1 fusions and `pssl` is defined in Sec. 4.7.1

To obtain the simulation results in Sec. 4.2.3, we sample 1200 values of $N^{\mathrm{MC}}_{\mathrm{GHZ}}$ through the aforementioned process. Let N_1 (N_2) be the minimal values of $N^{\mathrm{MC}}_{\mathrm{GHZ}}$ for the first 600 (total 1200) samples. If $N_1 = N_2$, the value is returned. If otherwise, we sample 1200 values of $N^{\mathrm{MC}}_{\mathrm{GHZ}}$ again and denote the minimal $N^{\mathrm{MC}}_{\mathrm{GHZ}}$ for the total 2400 samples by N_3. If $N_2 = N_3$, the value is returned. If otherwise, we sample 2400 merging graphs again and so on. By varying the total number of samples in this way, it is possible to increase the

odds that we reach close to the real optimal value.

After obtaining $N^*_{\rm GHZ}$ (at $\eta = \eta_0$), we consider the logical identity gate with $T = d-1$ (see Fig. 61) to calculate $\mathcal{N}_{p_L^{\rm targ}}$. $\mathcal{N}_{p_L^{\rm targ}}$ is determined by the following equality:

$$\mathcal{N}_{p_L^{\rm targ}} = N^*_{\rm GHZ}(2d_{p_L^{\rm targ}}+1)(3d^2_{p_L^{\rm targ}} - 3d_{p_L^{\rm targ}}+1),$$

where $d_{p_L^{\rm targ}}$ is the minimal code distance to achieve the target logical error rate of $p_L^{\rm targ}$ for the identity gate when $\eta = \eta_0$. $d_{p_L^{\rm targ}}$ is obtained by employing the error simulation method in Fig. 4.7.1. However, this method simulates the logical identity gate with $T = 4d+1$, while our current interest is that with $T = d-1$. From an obtained logical error rate p'_L from the method with $T = 4d+1$, we estimate the **logical error rate $p_L^{(1)}$ per two layers (one unit cell)** from the relation

$$p'_L = \sum_{t \le T:{\rm odd}} \binom{T}{t} \left(p_L^{(1)}\right)^t \left(1-p_L^{(1)}\right)^{T-t} = \frac{1}{2}\left[1-\left(1-2p_L^{(1)}\right)^T\right],$$

where $T = 4d+1$. Using a similar relation for $T = d-1$, we can convert p'_L into the logical error rate p_L of the gate with $T = d-1$. We then obtain $d_{p_L^{\rm targ}}$ by calculating the logical error rate $p_L^{(d)}$ at $\eta = \eta_0$ for each code distance $d \le 11$ and finding the smallest d satisfying $p_L^{(d)} < p_L^{\rm targ}$. If $p_L^{(11)} \ge p_L^{\rm targ}$, $d_{p_L^{\rm targ}}$ is estimated from the linear extrapolation of the points $(9,\ \log p_L^{(9)})$ and $(11,\ \log p_L^{(11)})$.

Chapter 5

Conclusion

Measurement-based quantum computing (MBQC) is an attractive methodology for conducting quantum computing, thanks to its nature that it is processed with only single-qubit measurements provided that entangled resource states are supplied. Optical systems are particularly suitable to implement MBQC since single-qubit measurement can be made with high accuracy and entangled resource states can be generated with linear optical circuits such as beam splitters. However, the detrimental effects of various error sources during computation must be sufficiently suppressed by error-correcting techniques to obtain reliable outcomes. Not only that, a large number of resources demanded to generate cluster states makes its realization technically difficult.

In this dissertation, we have explored two different schemes for universal fault-tolerant MBQC with topologically-encoded logical qubits, which have several significant advantages over previous approaches including resource efficiency. The first scheme investigated in Chapter 3, which is platform-independent, uses color-code-based cluster states as resource states instead of conventional Raussendorf's three-dimensional cluster states (RTCSs). We elaborately designed the methods to perform elementary logic gates and correct errors that can occur in different regions of a cluster state. Notably, we showed that it is significantly more hardware-efficient when conducting

the phase and Hadamard gates than the scheme with RTCSs, which is due to the discovery that these gates can be implemented natively without additional techniques or resources. The second scheme addressed in Chapter 4, which operates in linear optical systems, exploits parity-encoded multiphoton qubits to boost the success probability of entangling operations for efficient construction of cluster states. We verified that this new protocol can tolerate high photon loss rates near 8% and has advantages over previous approaches in terms of loss-tolerance, resource overheads, or feasibility of basic elements. We further proposed a graph-theoretical algorithm to optimize the resources required to generate large-scale cluster states. We anticipate that our suggestions can contribute to lowering the technical barriers to accomplishing topological MBQC.

We finish this dissertation by presenting several related open questions:

1. We can construct cluster states based on various quantum error-correcting codes besides surface and color codes. For example, one may consider the $XZZX$ surface code, which has been recently getting attention due to its exceptional thresholds [99]. Are there any benefits to using such cluster states for MBQC?

2. We presented a color-code-based scheme in Chapter 3, but the linear-optical scheme in Chapter 4 is based on RTCSs. It is because we made use of the existing scheme for constructing an RTCS from GHZ-3 states. If we apply the parity encoding and the Bayesian error tracking method to the color-code-based scheme, how will the performance be compared to the previous results?

3. Are there any other encoding schemes better than the parity encoding for boosting the success probability of the type-II fusion?

4. How can we reduce the number of required switching circuits? Our parity-encoding-based protocol requires many of them for various parts such as adaptive lattice-level Bell-state measurement and post-selection of GHZ-3 states. They may heavily affect the overall photon loss rate.

We hope that these questions will be resolved in the near future.

Bibliography

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

[2] D. Herr, A. Paler, S. J. Devitt, and F. Nori, "Lattice surgery on the Raussendorf lattice," *Quantum Sci. Technol.*, vol. 3, no. 3, p. 035011, 2018.

[3] S. Omkar, S.-H. Lee, Y. S. Teo, S.-W. Lee, and H. Jeong, "All-photonic architecture for scalable quantum computing with Greenberger-Horne-Zeilinger states," *PRX Quantum*, vol. 3, p. 030309, 2022.

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

[5] L. K. Grover, "A fast quantum mechanical algorithm for database search," in *Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing*, STOC '96, (New York, NY, USA), p. 212–219, Association for Computing Machinery, 1996.

[6] I. M. Georgescu, S. Ashhab, and F. Nori, "Quantum simulation," *Rev. Mod. Phys.*, vol. 86, pp. 153–185, 2014.

[7] A. Das and B. K. Chakrabarti, "Colloquium: Quantum annealing and analog quantum computation," *Rev. Mod. Phys.*, vol. 80, pp. 1061–1081, 2008.

[8] A. Galindo and M. A. Martín-Delgado, "Information and computation: Classical and quantum aspects," *Rev. Mod. Phys.*, vol. 74, pp. 347–423, 2002.

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

[10] P. W. Shor, "Scheme for reducing decoherence in quantum computer memory," *Phys. Rev. A*, vol. 52, pp. R2493–R2496, 1995.

[11] C. H. Bennett, D. P. DiVincenzo, J. A. Smolin, and W. K. Wootters, "Mixed-state entanglement and quantum error correction," *Phys. Rev. A*, vol. 54, pp. 3824–3851, 1996.

[12] R. Laflamme, C. Miquel, J. P. Paz, and W. H. Zurek, "Perfect quantum error correcting code," *Phys. Rev. Lett.*, vol. 77, pp. 198–201, 1996.

[13] A. R. Calderbank and P. W. Shor, "Good quantum error-correcting codes exist," *Phys. Rev. A*, vol. 54, pp. 1098–1105, 1996.

[14] A. Steane, "Multiple-particle interference and quantum error correction," *P. Roy. Soc. Lond. A Mat.*, vol. 452, no. 1954, pp. 2551–2577, 1996.

[15] H. Bombín, "Topological codes," in *Quantum Error Correction* (D. A. Lidar and T. A. Brun, eds.), ch. 19, pp. 455–481, Cambridge, 2013.

[16] A. Y. Kitaev, "Quantum computations: algorithms and error correction," *Russ. Math. Surv.*, vol. 52, no. 6, pp. 1191–1249, 1997.

[17] S. B. Bravyi and A. Y. Kitaev, "Quantum codes on a lattice with boundary," *arXiv:quant-ph/9811052*, 1998.

[18] E. Dennis, A. Kitaev, A. Landahl, and J. Preskill, "Topological quantum memory," *J. Math. Phys.*, vol. 43, no. 9, pp. 4452–4505, 2002.

[19] A. Y. Kitaev, "Fault-tolerant quantum computation by anyons," *Ann. Phys.*, vol. 303, no. 1, pp. 2–30, 2003.

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

[21] H. Bombin and M. A. Martin-Delgado, "Quantum measurements and gates by code deformation," *J. Phys. A: Math. Theor.*, vol. 42, no. 9, p. 095302, 2009.

[22] B. M. Terhal, "Quantum error correction for quantum memories," *Rev. Mod. Phys.*, vol. 87, pp. 307–346, 2015.

[23] H. Bombin and M. A. Martin-Delgado, "Topological quantum distillation," *Phys. Rev. Lett.*, vol. 97, p. 180501, 2006.

[24] A. G. Fowler, "Two-dimensional color-code quantum computation," *Phys. Rev. A*, vol. 83, p. 042310, 2011.

[25] 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*, vol. 2, p. 101, 2018.

[26] B. Koczor, "Exponential error suppression for near-term quantum devices," *Phys. Rev. X*, vol. 11, p. 031057, 2021.

[27] W. J. Huggins, S. McArdle, T. E. O'Brien, J. Lee, N. C. Rubin, S. Boixo, K. B. Whaley, R. Babbush, and J. R. McClean, "Virtual distillation for quantum error mitigation," *Phys. Rev. X*, vol. 11, p. 041036, 2021.

[28] K. Bharti, A. Cervera-Lierta, T. H. Kyaw, T. Haug, S. Alperin-Lea, A. Anand, M. Degroote, H. Heimonen, J. S. Kottmann, T. Menke, W.-K. Mok, S. Sim, L.-C. Kwek, and A. Aspuru-Guzik, "Noisy intermediate-scale quantum algorithms," *Rev. Mod. Phys.*, vol. 94, p. 015004, 2022.

[29] S. Bravyi and A. Kitaev, "Universal quantum computation with ideal Clifford gates and noisy ancillas," *Phys. Rev. A*, vol. 71, p. 022316, 2005.

[30] C. Jones, "Multilevel distillation of magic states for quantum computing," *Phys. Rev. A*, vol. 87, p. 042305, 2013.

[31] R. Raussendorf, J. Harrington, and K. Goyal, "Topological fault-tolerance in cluster state quantum computation," *New J. Phys.*, vol. 9, no. 6, p. 199, 2007.

[32] R. Raussendorf and H. J. Briegel, "A one-way quantum computer," *Phys. Rev. Lett.*, vol. 86, pp. 5188–5191, 2001.

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

[34] R. Raussendorf, J. Harrington, and K. Goyal, "A fault-tolerant one-way quantum computer," *Ann. Phys.*, vol. 321, no. 9, pp. 2242–2270, 2006.

[35] R. Raussendorf and J. Harrington, "Fault-tolerant quantum computation with high threshold in two dimensions," *Phys. Rev. Lett.*, vol. 98, p. 190504, 2007.

[36] A. G. Fowler and K. Goyal, "Topological cluster state quantum computing," *Quantum Info. Comput.*, vol. 9, no. 9, p. 721–738, 2009.

[37] S.-H. Lee and H. Jeong, "Universal hardware-efficient topological measurement-based quantum computation via color-code-based cluster states," *Phys. Rev. Research*, vol. 4, p. 013010, 2022.

[38] S.-H. Lee, S. Omkar, Y. S. Teo, and H. Jeong, "Parity-encoding-based quantum computing with Bayesian error tracking," *arXiv:2207.06805 [quant-ph]*, 2022.

[39] D. Gottesman, "Class of quantum error-correcting codes saturating the quantum Hamming bound," *Phys. Rev. A*, vol. 54, pp. 1862–1868, 1996.

[40] D. A. Lidar and T. A. Brun, *Quantum Error Correction*. Cambridge University Press, 2013.

[41] D. Gottesman, *Stabilizer codes and quantum error correction*. PhD thesis, California Institute of Technology, Pasadena, CA, 1997.

[42] D. E. Browne and T. Rudolph, "Resource-efficient linear optical quantum computation," *Phys. Rev. Lett.*, vol. 95, p. 010501, 2005.

[43] C. Horsman, A. G. Fowler, S. Devitt, and R. Van Meter, "Surface code quantum computing by lattice surgery," *New J. Phys.*, vol. 14, no. 12, p. 123011, 2012.

[44] B. J. Brown and S. Roberts, "Universal fault-tolerant measurement-based quantum computation," *Phys. Rev. Research*, vol. 2, p. 033305, 2020.

[45] A. Bolt, G. Duclos-Cianci, D. Poulin, and T. M. Stace, "Foliated quantum error-correcting codes," *Phys. Rev. Lett.*, vol. 117, p. 070501, 2016.

[46] J. Edmonds, "Paths, trees, and flowers," *Can. J. Math.*, vol. 17, pp. 449–467, 1965.

[47] J. Edmonds, "Maximum matching and a polyhedron with 0, 1-vertices," *J. Res. Nat. Bur. Stand. B*, vol. 69, no. 125-130, pp. 55–56, 1965.

[48] A. G. Fowler, "Minimum weight perfect matching of fault-tolerant topological quantum error correction in average $o(1)$ parallel time," *Quantum Info. Comput.*, vol. 15, no. 1–2, p. 145–158, 2015.

[49] V. Kolmogorov, "Blossom V: a new implementation of a minimum cost perfect matching algorithm," *Math. Program. Comput.*, vol. 1, no. 1, pp. 43–67, 2009.

[50] S. D. Barrett and T. M. Stace, "Fault tolerant quantum computation with very high threshold for loss errors," *Phys. Rev. Lett.*, vol. 105, p. 200502, 2010.

[51] A. G. Fowler, A. C. Whiteside, A. L. McInnes, and A. Rabbani, "Topological code autotune," *Phys. Rev. X*, vol. 2, p. 041003, 2012.

[52] A. C. Whiteside and A. G. Fowler, "Upper bound for loss in practical topological-cluster-state quantum computing," *Phys. Rev. A*, vol. 90, p. 052316, 2014.

[53] N. Delfosse, "Decoding color codes by projection onto surface codes," *Phys. Rev. A*, vol. 89, p. 012317, 2014.

[54] A. Kubica and N. Delfosse, "Efficient color code decoders in $d \geq 2$ dimensions from toric code decoders," *arXiv:1905.07393 [quant-ph]*, 2019.

[55] C. Chamberland, A. Kubica, T. J. Yoder, and G. Zhu, "Triangular color codes on trivalent graphs with flag qubits," *New J. Phys.*, vol. 22, no. 2, p. 023019, 2020.

[56] D. Litinski, "Magic state distillation: Not as costly as you think," *Quantum*, vol. 3, p. 205, 2019.

[57] C. Chamberland and K. Noh, "Very low overhead fault-tolerant magic state preparation using redundant ancilla encoding and flag qubits," *npj Quant. Inf.*, vol. 6, no. 1, p. 91, 2020.

[58] H. Bombin and M. A. Martin-Delgado, "Topological computation without braiding," *Phys. Rev. Lett.*, vol. 98, p. 160502, 2007.

[59] H. Bombin and M. A. Martin-Delgado, "Exact topological quantum order in $d = 3$ and beyond: Branyons and brane-net condensates," *Phys. Rev. B*, vol. 75, p. 075103, 2007.

[60] H. Bombín, "Gauge color codes: optimal transversal gates and gauge fixing in topological stabilizer codes," *New J. Phys.*, vol. 17, no. 8, p. 083002, 2015.

[61] A. Kubica and M. E. Beverland, "Universal transversal gates with color codes: A simplified approach," *Phys. Rev. A*, vol. 91, p. 032330, 2015.

[62] 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*, vol. 92, p. 022312, 2015.

[63] 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.*, vol. 120, p. 180501, 2018.

[64] H. Bombin, "2D quantum computation with 3D topological codes," *arXiv:1810.09571 [quant-ph]*, 2018.

[65] H. Bombin, "Transversal gates and error propagation in 3D topological codes," *arXiv:1810.09575 [quant-ph]*, 2018.

[66] 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.*, vol. 4, no. 1, p. 015005, 2018.

[67] D. Litinski and F. v. Oppen, "Lattice surgery with a twist: Simplifying Clifford gates of surface codes," *Quantum*, vol. 2, p. 62, 2018.

[68] T. C. Ralph and G. J. Pryde, "Chapter 4 - Optical Quantum Computation," vol. 54 of *Progress in Optics*, pp. 209–269, Elsevier, 2010.

[69] S. L. Braunstein and A. Mann, "Measurement of the Bell operator and quantum teleportation," *Phys. Rev. A*, vol. 51, p. R1727(R), 1995.

[70] J. M. Auger, H. Anwar, M. Gimeno-Segovia, T. M. Stace, and D. E. Browne, "Fault-tolerant quantum computation with nondeterministic entangling gates," *Phys. Rev. A*, vol. 97, p. 030301(R), 2018.

[71] H. Jeong, M. S. Kim, and J. Lee, "Quantum-information processing for a coherent superposition state via a mixedentangled coherent channel," *Phys. Rev. A*, vol. 64, p. 052308, 2001.

[72] H. Jeong and M. S. Kim, "Efficient quantum computation using coherent states," *Phys. Rev. A*, vol. 65, p. 042305, 2002.

[73] S. Omkar, Y. S. Teo, and H. Jeong, "Resource-efficient topological fault-tolerant quantum computation with hybrid entanglement of light," *Phys. Rev. Lett.*, vol. 125, p. 060501, 2020.

[74] S. Omkar, Y. S. Teo, S.-W. Lee, and H. Jeong, "Highly photon-loss-tolerant quantum computing using hybrid qubits," *Phys. Rev. A*, vol. 103, p. 032602, 2021.

[75] S.-W. Lee, K. Park, T. C. Ralph, and H. Jeong, "Nearly deterministic Bell measurement with multiphoton entanglement for efficient quantum-information processing," *Phys. Rev. A*, vol. 92, p. 052324, 2015.

[76] W. P. Grice, "Arbitrarily complete Bell-state measurement using only linear optical elements," *Phys. Rev. A*, vol. 84, p. 042331, 2011.

[77] F. Ewert and P. van Loock, "$3/4$-Efficient Bell measurement with passive linear optics and unentangled ancillae," *Phys. Rev. Lett.*, vol. 113, p. 140403, 2014.

[78] D. Herr, A. Paler, S. J. Devitt, and F. Nori, "A local and scalable lattice renormalization method for ballistic quantum computation," *npj Quantum Information*, vol. 4, no. 1, p. 27, 2018.

[79] K. Fujii and Y. Tokunaga, "Fault-tolerant topological one-way quantum computation with probabilistic two-qubit gates," *Phys. Rev. Lett.*, vol. 105, p. 250503, 2010.

[80] Y. Li, S. D. Barrett, T. M. Stace, and S. C. Benjamin, "Fault tolerant quantum computation with nondeterministic gates," *Phys. Rev. Lett.*, vol. 105, p. 250502, 2010.

[81] Y. Li, P. C. Humphreys, G. J. Mendoza, and S. C. Benjamin, "Resource costs for fault-tolerant linear optical quantum computing," *Phys. Rev. X*, vol. 5, p. 041007, 2015.

[82] S. Takeda, T. Mizuta, M. Fuwa, P. Van Loock, and A. Furusawa, "Deterministic quantum teleportation of photonic quantum bits by a hybrid technique," *Nature*, vol. 500, no. 7462, pp. 315–318, 2013.

[83] H. A. Zaidi and P. van Loock, "Beating the one-half limit of ancilla-free linear optics Bell measurements," *Phys. Rev. Lett.*, vol. 110, p. 260501, 2013.

[84] T. Kilmer and S. Guha, "Boosting linear-optical Bell measurement success probability with predetection squeezing and imperfect photon-number-resolving detectors," *Phys. Rev. A*, vol. 99, p. 032302, 2019.

[85] 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.*, vol. 115, p. 020502, 2015.

[86] H. A. Zaidi, C. Dawson, P. van Loock, and T. Rudolph, "Near-deterministic creation of universal cluster states with probabilistic Bell measurements and three-qubit resource states," *Phys. Rev. A*, vol. 91, p. 042301, 2015.

[87] M. Pant, D. Towsley, D. Englund, and S. Guha, "Percolation thresholds for photonic quantum computing," *Nat. Commun.*, vol. 10, no. 1, p. 1070, 2019.

[88] T. C. Ralph, A. J. F. Hayes, and A. Gilchrist, "Loss-tolerant optical qubits," *Phys. Rev. Lett.*, vol. 95, p. 100501, 2005.

[89] S.-W. Lee, T. C. Ralph, and H. Jeong, "Fundamental building block for all-optical scalable quantum networks," *Phys. Rev. A*, vol. 100, p. 052303, 2019.

[90] K. Kieling, T. Rudolph, and J. Eisert, "Percolation, renormalization, and quantum computing with nondeterministic gates," *Phys. Rev. Lett.*, vol. 99, p. 130501, 2007.

[91] E. Knill, "Quantum computing with realistically noisy devices," *Nature*, vol. 434, no. 7029, pp. 39–44, 2005.

[92] N. Lütkenhaus, J. Calsamiglia, and K.-A. Suominen, "Bell measurements for teleportation," *Phys. Rev. A*, vol. 59, pp. 3295–3300, 1999.

[93] M. Varnava, D. E. Browne, and T. Rudolph, "How good must single photon sources and detectors be for efficient linear optical quantum computation?," *Phys. Rev. Lett.*, vol. 100, p. 060502, 2008.

[94] A. A. Hagberg, D. A. Schult, and P. J. Swart, "Exploring network structure, dynamics, and function using NetworkX," in *Proceedings of the 7th Python in Science Conference* (G. Varoquaux, T. Vaught, and J. Millman, eds.), (Pasadena, CA USA), pp. 11 – 15, 2008.

[95] O. Higgott, "PyMatching: A Python package for decoding quantum codes with minimum-weight perfect matching," *ACM Trans. Quantum Comput.*, 2021.

[96] C. D. Lorenz and R. M. Ziff, "Precise determination of the bond percolation thresholds and finite-size scaling corrections for the sc, fcc, and bcc lattices," *Phys. Rev. E*, vol. 57, pp. 230–236, 1998.

[97] N. Delfosse and N. H. Nickerson, "Almost-linear time decoding algorithm for topological codes," *Quantum*, vol. 5, p. 595, 2021.

[98] S. Bartolucci, P. Birchall, H. Bombin, H. Cable, C. Dawson, M. Gimeno-Segovia, E. Johnston, K. Kieling, N. Nickerson, M. Pant, *et al.*, "Fusion-based quantum computation," *arXiv:2101.09310 [quant-ph]*, 2021.

[99] J. P. Bonilla Ataides, D. K. Tuckett, S. D. Bartlett, S. T. Flammia, and B. J. Brown, "The XZZX surface code," *Nature communications*, vol. 12, no. 1, p. 2172, 2021.

국문초록

측정 기반 양자 컴퓨팅(MBQC)은 양자 컴퓨팅의 한 방법론으로, 클러스터 상태라고 하는 큰 얽힌 상태에 단일 큐비트 측정을 함으로써 이루어지며, 특히 광학 시스템에 적합하다. 위상기하학적 양자 오류 정정 부호를 활용하면 MBQC가 작은 오류들에 내성을 가지도록 할 수 있음이 알려져 있다. 이 학위논문에서는 기존의 프로토콜들에 대해 오류에 대한 내성과 자원 효율성의 측면에서 장점을 가지는 두 종류의 위상기하학적 MBQC 프로토콜을 소개한다.

첫 번째 프로토콜로서 2차원 색 부호를 기반으로 구성된 클러스터 상태를 이용하는 위상기하학적 MBQC 프로토콜을 제안한다. 라우젠도르프(Raussendorf)의 3차원 클러스터 상태(RTCS)를 기반으로 하는 기존의 위상기하학적 MBQC 프로토콜은 한 가지 문제점이 있는데, 임의의 논리적 게이트를 구축하기 위해 필수적인 하다마드(Hadamard) 게이트와 위상 게이트가 추가적인 기술 없이 구현될 수 없다는 것이다. 이러한 단점은 RTCS 프로토콜의 실현을 방해하는 기술적인 장벽이 된다. 우리는 이러한 문제를 해결하기 위해 RTCS 대신 색 부호 기반의 클러스터 상태를 사용하는 프로토콜을 제시한다. 그 결과 하다마드 게이트와 위상 게이트가 오류 정정과 함께 추가적인 기술 없이 구현될 수 있으며, 이는 RTCS 프로토콜에서 상태 증류(state distillation) 방법을 활용하는 것보다 약 26배의 필요 자원량 감소를 가져온다는 것을 보인다.

두 번째 프로토콜로서 패리티(parity) 부호 기반의 다중광자 큐비트를 활용하는 선형광학적 위상기하학 MBQC 프로토콜을 제시한다. 선형광학 시스템에서의 MBQC에서 얽힘 작용의 비결정론적인 특성과 광자 손실은

거대한 클러스터 상태의 생성을 방해하고 논리적 오류를 유발한다. 비이상적(nonideal) 얽힘 작용이 주위 큐비트들을 불가피하게 손상시킴에도 불구하고 제시된 프로토콜이 광자 손실에 높은 내성을 가지고 자원 효율적이라는 사실을 밝힌다. 현실적인 오류 분석을 위해, 이러한 악영향들에 의한 오류들을 추적하는 베이지안(Bayesian)적 방법론을 소개한다. 오류에 대한 내성, 필요 자원량, 기본 요소들의 실현 가능성 등의 측면에서 기존에 존재하는 방법들보다 우리의 프로토콜이 우위를 가진다는 것을 보인다.

주요어 : 양자 컴퓨팅, 양자 정보, 양자 오류 정정, 측정 기반 양자 컴퓨팅
학번 : 2017-27328

감사의 글

6년간의 석박통합과정 생활을 보내면서 많은 분들의 도움이 있었습니다. 이 자리를 통해서 그 분들께 감사의 인사를 남깁니다.

가장 먼저, 그동안 저를 지도해주신 정현석 교수님께 감사하다는 말씀을 드리고 싶습니다. 지덕을 겸비하신 교수님의 연구실에 들어올 수 있었던 것은 크나큰 행운이었습니다. 연구실에 처음 들어와서 한동안 주제를 잡지 못하고 방황하던 저에게 나아가야 할 방향을 조언해주시고 헌신적으로 지도해주신 덕분에 무탈하게 졸업할 수 있었습니다.

박사논문 심사위원을 맡아주시고 귀중한 시간을 내주신 안경원 교수님, 신용일 교수님, 김도헌 교수님, 이승우 박사님께도 감사의 말씀을 드립니다. 많은 유익한 질문들과 조언들이 학위논문을 작성할 때 큰 도움이 되었습니다. 특히 이승우 박사님과 박사님의 연구실 분들과 함께 공동연구를 진행하면서 유익한 학술 논의를 할 수 있어서 즐거웠습니다. 앞으로도 같이 연구할 기회가 있으면 좋겠습니다.

연구실의 선후배님들 덕분에 즐거운 연구실 생활을 보낼 수 있었습니다. 앞서 졸업하신 선배님들인 권혁준 교수님, 오창훈 박사님, 안대건 박사님, 전인우 박사님, 최성전 박사님, 이석형(Lie) 박사님께 연구와 대학원 생활 관련해서 많은 도움들과 조언을 주셔서 감사하다는 말씀을 드리고 싶습니다. 후배들인 혁건이형, 성욱이형, 규동이, 병선이, 성주에게도 함께 지낼 수 있어서 즐거웠고 앞으로 남은 대학원 생활을 응원한다는 말을 전하고 싶습니다. Dr. Omkar와 Dr. Yong Siah께도 공동연구를 같이 하면서 많은 도움을 주시고 해외 포스닥 생활에 대해 유익한 조언들을 해주신 점에 대해 감사를 표합니다. 여러분 덕분에 연구실 생활이 정말 그리울 것

같습니다.

학부시절 같이 밴드를 했고 그 후에도 가깝게 지내온 재준이형, 준규형, 다나누나, 다혜누나, 가은누나, 연재, 그리고 고등학교 동창인 동현이와 중학교 때부터 알고 지내던 송현이에게도 그동안 편안한 안식처가 되어줘서 고맙다는 말을 전하고 싶습니다. 앞으로도 계속 연락하면서 친하게 지낼 수 있으면 좋겠습니다.

마지막으로, 사랑하는 가족들의 든든한 지원이 있었기 때문에 박사학위를 무사히 마칠 수 있었습니다. 저를 훌륭하게 키워주시고 아낌없는 사랑을 주신 부모님과, 찾아뵐때마다 항상 반겨주시고 자랑스러워해 주셨던 할머니, 할아버지, 외할머니께 마음 속 깊이 감사드립니다. 받은 은혜만큼 베풀 수 있는 사람이 되겠습니다. 이제 학부 졸업이 얼마 남지 않은 동생 태형이에게는 앞날을 응원한다는 말을 전하고 싶습니다.