SURFACE CODE ERROR CORRECTION DURING LOGICAL OPERATIONS

SURFACE CODE ERROR CORRECTION DURING LOGICAL OPERATIONS

Master's Thesis

To obtain the degree of Master's in Science
at the Delft University of Technology, the Netherlands

by

Zherui WANG

Student Number: 5570352

Thesis Committee: Dr. Barbara M. Terhal, Thesis Supervisor

Dr. David Elkouss Coronas

Dr. Maximilian Rimbach-Russ

ABSTRACT

Surface codes have become a leading method for quantum error correction (QEC) and play a pivotal role in the realization of fault-tolerant quantum computing. However, while the execution of more quantum error correction cycles can potentially protect the quantum information for longer time, an excess can potentially slow down computation and increase errors in idling logical qubits. This dissertation addresses this challenge by focusing on the optimization of rounds of QEC cycles for two important scenarios: lattice surgery and moving logical qubits. Theoretical upper bounds for decoding failure are derived for these settings, providing insights into the required number of QEC cycles for successful decoding. Numerical experiments are conducted to support and validate these theoretical findings, emphasizing the necessity of additional QEC cycles to ensure reliable stabilizer measurement outcomes, particularly in lattice surgery and logical qubit movement. Furthermore, the research investigates the fault-tolerant and proper order of syndrome measurements within each QEC cycle for square surface codes. The order of these measurements is shown to have a significant impact on logical error rates. Additionally, methods for constructing detectors under code and stabilizer deformation are introduced, facilitating accurate tracking of error syndromes.

ACKNOWLEDGEMENTS

I feel incredibly fortunate to have spent my years as a master's student at TU Delft surrounded by such remarkable individuals. I would like to express my sincere appreciation to all the wonderful people who have contributed to my learning and personal growth during this time, particularly those I mention here.

First and foremost, I would like to express my deepest gratitude to my supervisor, Barbara M. Terhal. Every discussion with her has been incredibly valuable and inspiring. I am grateful for her unwavering dedication to academia, her patience, and her insightful perspectives. I also appreciate the time and effort she spent in my thesis project, as well as her guidance in both academic and personal matters. I would also like to thank David Elkouss Coronas and Maximilian Rimbach-Russ for being my committee.

Being a part of the Terhal Group has been an extraordinary experience, and I am grateful to everyone in the group. Thank you, Yaroslav, Mac, Maarten, Boris, Yang, and Marc, for your help and stimulating discussions related to my thesis. I am also appreciative of the wonderful talks you delivered during our journal club meetings. Our memorable hiking adventures will always hold a special place in my heart.

I would like to express my gratitude to Jordi Tura i Brugués for being as my supervisor for the Casimir project. I am thankful for the guidance and support provided throughout the project. I would also like to thank Yash and David for the fruitful meetings we had. The discussions and insights shared during those meetings were valuable in shaping my research. My time in Leiden has brought me great joy, and I am grateful for the experiences and opportunities I had during that period. I would like to extend my thanks once again to David Elkouss Coronas for providing me with the opportunity to have an internship in Japan. I am now very excited about our future project.

I would like to express my gratitude to my friends from the *basically* group: Adria, Alvaro, Casper, Cornelius, Jane, Katerina, Ksusha, Liselotte, Marc, Mare, Matteo, Mick, Monika, Wouter, and Zarije. Your friendships have been a source of joy, and I cherish every moment we have spent together. I am grateful for the positive and energetic atmosphere you have all brought into my life, and I have learned a great deal from each of you.

I would also like to extend my thanks to my roommates at Prof. Schermerhornstraat 93: Haitao, Yitong, Yuxing, Ketong, and Yifei. Your accompany and the delicious homemade Chinese food we shared during every Chinese festival have been greatly appreciated.

To my fellow peers in the Applied Physics Master program at TU Delft, including Bokai, Dingshan, Duiquan, Jianyao, Jinlun, Luozhen, Mingshen, Siyu, Tianyin, Xiaoxian, Xiaoyu, Xinru, Yining, Yunzhe, Yuejie, Yudi, Yuning, Yaozu, and Zenghui, I extend my gratitude. Our time together, whether it was on trips or gathering at Chinese restaurants, has been memorable. I am thankful for the fruitful scientific discussions we have had throughout these two years.

Finally, I would like to express my heartfelt appreciation to my family. Your unwavering support and love have been the foundation of my journey. I am also deeply grateful to my girlfriend, Jinlun, for her love and encouragement. Without you, I would not have had the opportunity to accomplish what I have. I am at a loss for words to express my love and gratitude to all of you.

CONTENTS

1

INTRODUCTION

1.1. QUANTUM ERROR CORRECTION

Quantum computing holds great promise, but a key challenge stands in its way: quantum systems are remarkably sensitive to errors. These errors could be triggered by various factors, like environmental noise or inaccuracies in the execution of quantum operations. To protect quantum information and ensure accurate calculations, we need strategies for identifying and correcting these errors.

Quantum Error Correction (QEC) offers a solution. It is a set of techniques that safeguard quantum information by embedding it in a larger space, so that errors can be detected and corrected. As long as errors on the physical qubits of the code stay rare enough, we can maintain the integrity of our quantum information, despite the challenges presented by noise and imperfections.

1.1.1. MODELLING QUANTUM NOISE

The mechanisms that introduce noise in quantum information processing are manifold, ranging from imperfect classical control to interaction with the surrounding environment. Practically, any qubit is susceptible to an variety of errors. In order to properly characterize these qubit errors engendered by noise, we present several specific error models in this section, which serve to delineate the potential errors that may occur.

INDEPENDENT NOISE MODEL

Potential obstacle to the idea of error correction in the quantum computing is that noise can induce continuous state deformations. However, this problem can be readily addressed by recognizing that all quantum errors can actually be decomposed into merely two independent error types: bit-flip (X), phase-flip (Z) errors, which are independent with each other. The bit-flip and phase-flip channel can be mathematically described as follows:

$$\mathscr{E}_X(\rho, p_X) = \big(1 - p_X\big)\,\rho + p_X(X\rho X) \tag{1.1}$$

$$\mathscr{E}_Z(\rho, p_Z) = \big(1 - p_Z\big)\,\rho + p_Z(Z\rho Z) \tag{1.2}$$

1

In these equations, p_X and p_Z represent the respective probabilities of bit-flip or phase-flip error events occurring. We also note that the combination of a bit-flip and a phase-flip error is a bit-phase flip (Y) error, since $Y = iXZ$. Similarly, the bit-phase flip channel can be described as:

$$\mathscr{E}_Y(\rho, p_Y) = (1 - p_Y)\rho + p_Y(Y\rho Y) \tag{1.3}$$

DEPOLARIZING NOISE MODEL

The depolarizing noise channel is another model that inflicts random Pauli errors on the original state, distributing the probabilities equally. For a single qubit, the depolarizing model can be expressed as:

$$\mathscr{D}_1(\rho, p_1) = (1 - p_1)\rho + \frac{p_1}{3}(X\rho X + Y\rho Y + Z\rho Z), \tag{1.4}$$

where p_1 represents the probability of applying a Pauli X, Y, or Z operator to the qubit, respectively. For a two-qubit scenario, the model takes the following form:

$$\mathscr{D}_2(\rho, p_2) = (1 - p_2)\rho + \frac{p_2}{15}\sum_{i,j=0}^{3} K_{ij}\rho K_{ij}^{\dagger} \tag{1.5}$$

Here, each K_{ij} represents the tensor product of two Pauli matrices, defined as $K_{ij} = \sigma_i \otimes \sigma_j$. The Pauli matrices $\sigma_{0,1,2,3}$ correspond to the identity and the three Pauli matrices, specifically $\sigma_{0,1,2,3} = I, X, Y, Z$, respectively.

DECORRELATE THE DEPOLARIZING MODEL

In contrast to the independent noise model, depolarizing errors are defined using a set of disjoint individual error cases. This interdependence among various cases can complicate the analysis of depolarizing noise effects on quantum circuits. Consequently, it is crucial to redefine the depolarizing error in terms of independent error cases, that is, to decorrelate the depolarizing noise channel [1].

For the single-qubit scenario, we aim to model a depolarizing error that is going to to occur with a probability p_1. To achieve this, we employ independent random Pauli X, Y, and Z errors each occurring with probabilities $p_X = p_Y = p_Z = p$. More explicitly, we seek to express

$$\mathscr{D}_1(\rho, p_1) = \mathscr{E}_X \circ \mathscr{E}_Y \circ \mathscr{E}_Z(\rho, p), \tag{1.6}$$

which indicates that the effect of the depolarizing channel with parameter p_1 on ρ equates to the aggregate effect of the independent Pauli X, Z, Y noise channels each with parameter p on ρ. Upon examining Eq. (1.6), the equation holds if

$$p(1-p)^2 + p^2(1-p) = p_1/3 \tag{1.7}$$

From this, we can solve for p, yielding:

$$p = \frac{1}{2} \pm \frac{1}{2}\sqrt{1 - \frac{4}{3}p_1} \tag{1.8}$$

We can disregard the solution larger than $\frac{1}{2}$ as we assume that the error probability does not exceed 50%. It is important to observe that this solution is applicable only when

$p_1 \leqslant \frac{3}{4}$. This restriction aligns intuitively with the understanding that $p_1 = 3/4$ marks the threshold at which a depolarizing error maximally mixes a qubit.

When considering a two-qubit scenario, similarly, we can establish the decorrelation by setting:

$$\begin{aligned}\mathscr{D}_2(\rho, p_2) = &\mathscr{E}_{IX} \circ \mathscr{E}_{IY} \circ \mathscr{E}_{IZ} \circ \\ &\mathscr{E}_{XI} \circ \mathscr{E}_{XX} \circ \mathscr{E}_{XY} \circ \mathscr{E}_{XZ} \circ \\ &\mathscr{E}_{YI} \circ \mathscr{E}_{YX} \circ \mathscr{E}_{YY} \circ \mathscr{E}_{YZ} \circ \\ &\mathscr{E}_{ZI} \circ \mathscr{E}_{ZX} \circ \mathscr{E}_{ZY} \circ \mathscr{E}_{ZZ}(\rho, p'),\end{aligned} \tag{1.9}$$

One plausible solution of Eq. (1.9) is found to be [1]:

$$p' = \frac{1}{2} - \frac{1}{2}\left(1 - \frac{16}{15} p_2\right)^{1/8}, \tag{1.10}$$

which is only applicable if $p_2 \leqslant \frac{15}{16}$.

It is worth noting that in all simulations incorporated in this thesis, we always transformed depolarizing noise channels, as detailed in Section 1.1.1, into a series of independent error cases, by applying the method specified above.

CIRCUIT-LEVEL NOISE MODEL

In the circuit-level noise model, depolarizing channels are introduced following each physical gate to simulate imperfections in their implementation. Additionally, depolarizing channels are added to the qubits during idle periods to simulate their decoherence.

For a single qubit undergoing decoherence during an idle period of time $\Delta t \ll T_{1,2}$, its behavior can be phenomenologically characterized by a density matrix transformation model. In this model, the density matrix $\rho(t)$ at time t evolves to $\rho(t+\Delta t)$ as follows:

$$\rho(t) = \begin{bmatrix} a(t) & b(t) \\ b^*(t) & 1-a(t) \end{bmatrix} \rightarrow \rho(t+\Delta t) = \begin{bmatrix} (a(t)-a_0)\, e^{\frac{-\Delta t}{T_1}} + a_0 & b(t) e^{\frac{-\Delta t}{T_2}} \\ b^*(t) e^{\frac{-\Delta t}{T_2}} & (a_0 - a(t))\, e^{\frac{-\Delta t}{T_1}} + 1 - a_0 \end{bmatrix} \tag{1.11}$$

where T_1 and T_2 are the relaxation time and dephasing time, respectively, and a_0 characterizes the thermal equilibrium state. Assuming $a_0 = \frac{1}{2}$ as an approximation, which corresponds to the maximally mixed state, and considering $\Delta t \ll T_{1,2}$, we can simplify the expression of $\rho(t+\Delta t)$ as:

$$\rho(t+\Delta t) \approx \begin{bmatrix} a(t)\left(1 - \frac{\Delta t}{T_1}\right) + \frac{\Delta t}{2T_1} & b(t)\left(1 - \frac{\Delta t}{T_2}\right) \\ b^*(t)\left(1 - \frac{\Delta t}{T_2}\right) & -a(t)\left(1 - \frac{\Delta t}{T_1}\right) + 1 - \frac{\Delta t}{2T_1} \end{bmatrix} \tag{1.12}$$

On the other hand, the single-qubit depolarizing channel introduced in Section 1.1.1 can be alternatively expressed as:

$$\mathscr{D}_1(\rho, p_1) = \left(1 - \frac{4}{3} p_1\right)\rho + \frac{2p_1}{3} I \tag{1.13}$$

where we have used the fact that

$$I = \frac{\rho + X\rho X + Y\rho Y + Z\rho Z}{2} \tag{1.14}$$

for arbitrary ρ. Hence, the transformation of a single-qubit density matrix under a depolarizing channel $\mathscr{D}_1(\rho, p_1)$ can be written as:

$$\rho(t) = \begin{bmatrix} a(t) & b(t) \\ b^*(t) & 1-a(t) \end{bmatrix} \xrightarrow{\mathscr{D}_1} \begin{bmatrix} a(t)\left(1-\frac{4}{3}p_1\right)+\frac{2}{3}p_1 & b(t)\left(1-\frac{4}{3}p_1\right) \\ b^*(t)\left(1-\frac{4}{3}p_1\right) & -a(t)\left(1-\frac{4}{3}p_1\right)+1-\frac{2}{3}p_1 \end{bmatrix} \tag{1.15}$$

To approximate qubit decoherence using depolarizing channels, we compare Eq. (1.12) and Eq. (1.15), which yields:

$$p_1 = \frac{3}{4}\frac{\Delta t}{T_1} = \frac{3}{4}\frac{\Delta t}{T_2} \tag{1.16}$$

indicating that for a qubit with coherence time $T_1 = T_2$ idling for a duration Δt, we can use a single-qubit depolarizing channel with a parameter $p_1 = \frac{3}{4}\frac{\Delta t}{T_1} = \frac{3}{4}\frac{\Delta t}{T_2}$ to approximate its decoherence behavior.

The circuit-level noise model effectively approximates the imperfections inherent in real-world quantum computing processes. Throughout the simulations presented in this thesis, we adopt this noise model to more realistically represent the noise in the circuit.

PHENOMENOLOGICAL NOISE MODEL

The Phenomenological noise model combines the independent noise model, delineated in Section 1.1.1, with the effects of noisy measurements. This model is often employed in theoretical analysis and illustrative figures for the sake of simplicity and clarity.

1.1.2. STABILIZER FORMALISM

To describe the concept of stabilizer codes, we introduce the stabilizer formalism, a highly potent technique delineated in [2]. Stabilizer codes leverage parity check measurements to accumulate error information. Each stabilizer code can be characterized by an Abelian subgroup $\mathscr{S}$ generated by linearly independent generators $s_1, ..., s_m$ with $-I \notin \mathscr{S}$:

$$\mathscr{S} = \langle s_1, ..., s_m \rangle \tag{1.17}$$

The logical qubit states are the +1 eigenstates of each individual stabilizer $S \in \mathscr{S}$. The code space $\mathscr{T}(\mathscr{S})$ of a given stabilizer code corresponds to the shared +1 eigenspace of every element within $\mathscr{S}$. This is formally expressed as:

$$\mathscr{T}(\mathscr{S}) := \{|\psi\rangle \mid S|\psi\rangle = |\psi\rangle \ \ \forall S \in \mathscr{S}\} \tag{1.18}$$

This definition essentially means that any state $|\psi\rangle$ within the code space $\mathscr{T}(\mathscr{S})$ remains invariant under the application of any stabilizer S from the stabilizer group $\mathscr{S}$.

Let us denote $\mathbf{U}(2^n)$ as the group of unitary matrices acting on n qubits, and $\mathbf{C}_n$ as the Clifford group on n qubits, which is defined as:

$$\mathbf{C}_n = \left\{U \in \mathbf{U}(2^n) \mid U\mathscr{P}_n U^\dagger = \mathscr{P}_n\right\} \tag{1.19}$$

where $\mathscr{P}_n$ is the n-qubit Pauli group, representing the n-fold tensor products of Pauli operators found in $\mathscr{P}_1$. Here, $\mathscr{P}_1$ is the Pauli group on a single qubit, comprising 16 elements:

$$\mathscr{P}_1 =: \{\pm I, \pm iI, \pm X, \pm iX, \pm Y, \pm iY, \pm Z, \pm iZ\} \equiv \langle X, Y, Z\rangle \tag{1.20}$$

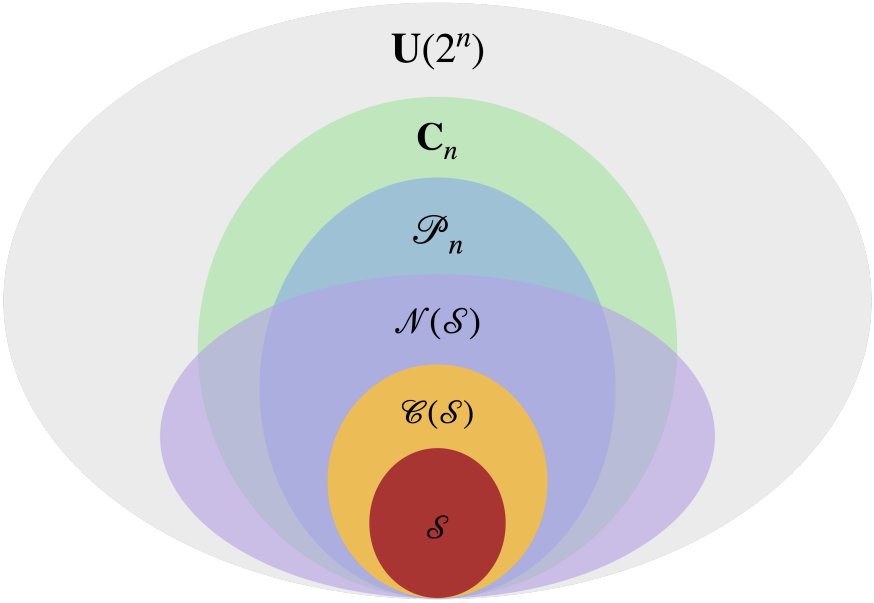

Figure 1.1: Venn diagram illustrating different groups of operators in quantum computation. Gray: Unitary operators $\mathbf{U}(2^n)$. Green: Clifford group $\mathbf{C}_n$. Blue: n-qubit Pauli group $\mathscr{P}_n$. Red: Stabilizer group $\mathscr{S}$. Yellow: Centralizer group $\mathscr{C}(\mathscr{S})$ of $\mathscr{S}$. Purple: Normalizer group $\mathscr{N}(\mathscr{S})$ of $\mathscr{S}$. This diagram depicts the relationships between these groups of operators.

The centralizer group $\mathscr{C}(\mathscr{S})$ of the stabilizer group $\mathscr{S}$ comprises all Pauli operators in $\mathscr{P}_n$ that commute with every element of $\mathscr{S}$, defined as:

$$\mathscr{C}(\mathscr{S}) = \{P \in \mathscr{P}_n \mid PS = SP \ \ \forall S \in \mathscr{S}\}. \tag{1.21}$$

On the other hand, the normalizer group $\mathscr{N}(\mathscr{S})$ of the stabilizer group $\mathscr{S}$ consists of unitary operators, which may not necessarily be Pauli operators, that normalize the stabilizer group $\mathscr{S}$, formally defined as:

$$\mathscr{N}(\mathscr{S}) = \{U \in \mathbf{U}(2^n) \mid U\mathscr{S}U^\dagger = \mathscr{S}\}. \tag{1.22}$$

It is worth noting that if the normalizer group $\mathscr{N}(\mathscr{S})$ of the stabilizer group $\mathscr{S}$ is defined as the group that consists of Pauli operators that normalize the stabilizer group $\mathscr{S}$, then we have:

$$\mathscr{N}(\mathscr{S}) = \{U \in \mathscr{P}_n \mid U\mathscr{S}U^\dagger = \mathscr{S}\} = \mathscr{C}(\mathscr{S}). \tag{1.23}$$

The definition given by Eq. (1.22) highlights the set of unitary operators U that, when applied to each element of $\mathscr{S}$, preserves the stabilizer group $\mathscr{S}$.

An error $E \in \mathscr{P}_n$ is classified as follows:

- It is detectable if $E \in \mathscr{P}_n \backslash \mathscr{C}(\mathscr{S})$ (meaning that E is in $\mathscr{P}_n$ but not $\mathscr{C}(\mathscr{S})$), implying that the error E anti-commutes with at least one element of $\mathscr{S}$.
- It is an undetectable logical error if $E \in \mathscr{C}(\mathscr{S}) \backslash \mathscr{S}$.

- It is considered a trivial error if $E \in \mathscr{S}$.

The code distance d is defined as the minimum weight $|P|$ of any operator $P \in \mathscr{C}(\mathscr{S}) \setminus \mathscr{S}$:

$$d := \min_{P \in \mathscr{C}(\mathscr{S}) \setminus \mathscr{S}} |P| \tag{1.24}$$

Here, the weight $|P|$ refers to the number of qubits on which P acts non-trivially. The code distance provides an important measure of the error resistance of a quantum code.

Logical operators, which can be created by single Pauli operators acting on data qubits, are contained within the centralizer group $\mathscr{C}(\mathscr{S})$. Since these logical operations commute with all stabilizers, the logical qubit remains within the code space under their transformation. However, commuting with all stabilizers is not a necessary condition for logical operators in a more general sense. Rather, logical operators reside in the normalizer group $\mathscr{N}(\mathscr{S})$, meaning a valid logical operation U must preserve the stabilizer group $\mathscr{S}$:

$$\mathscr{N}(\mathscr{S}) = \left\{ U \in \mathbf{U}(2^n) \mid \forall S \in \mathscr{S},\ \exists S' \in \mathscr{S} \text{ s.t. } US'U^\dagger = S \right\} \tag{1.25}$$

Thus, for $|\psi\rangle \in \mathscr{T}(\mathscr{S})$ and $U \in \mathscr{N}(\mathscr{S})$, it follows that $SU|\psi\rangle = US'|\psi\rangle = U|\psi\rangle \ \ \forall S \in \mathscr{S}$. This indicates that $U|\psi\rangle$ remains a $+1$ eigenstate of all stabilizers, thus verifying U as a legitimate logical operation that transforms an encoded codeword $|\psi\rangle$ to another valid codeword $U|\psi\rangle$.

1.2. SURFACE CODE

We now turn our attention to the surface code, a member of the stabilizer code family, first introduced by Bravyi and Kitaev [3].

This thesis primarily concerns itself with the planar code, a variant of the surface code that foregoes the periodic boundary conditions of toric code. This modification allows physical qubits to realistically interact within the system, enhancing the applicability of the planar code for practical fault-tolerant quantum computing implementations.

1.2.1. DEFINITION

In the construction of the planar code, physical qubits occupy positions along the edges of a two-dimensional lattice. The lattice is populated by two kinds of qubits: data qubits and ancilla qubits, distinguished solely by the distinct roles they play within the code.

Figure 1.2 depicts a representative lattice. It is evident that there are two varieties of stabilizer generators: the X-type and the Z-type. The X-type stabilizer generators (illustrated in red) are obtained from the product of X operators on the data qubits adjacent to X-type ancilla qubits. Similarly, the Z-type stabilizer generators (delineated in blue) are formed by the product of Z operators on the data qubits neighbouring the Z-type ancilla qubits. Thus, we can represent these stabilizer generators as follows:

$$S_x^v = \prod_{i \in Q(v)} X_i, \quad S_z^f = \prod_{k \in Q(f)} Z_k \tag{1.26}$$

In these equations, $Q(v), Q(f)$ denote the sets of qubits neighbouring the ancilla qubits v, f on the lattice grid, respectively. All these generators commute with each other, considering that either zero or two X and Z operators intersect between two different types

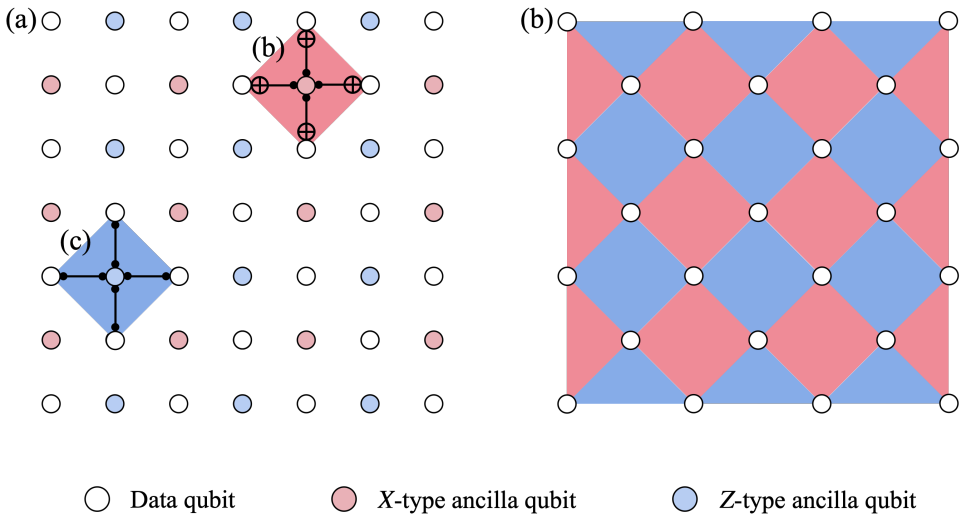

Figure 1.2: (a) A qubit lattice with boundary. Three types of qubits are shown, with data qubits in white, X-type ancilla qubits in red, and Z-type ancilla qubits in blue. The X-type and Z-type parity checks are shown in red and blue square tiles, respectively. (b) A distance-4 square surface code. Data qubits are depicted in white on the vertices, and the X-type and Z-type parity checks are represented by blue and red square tiles, respectively.

of stabilizers. Consequently, the data qubits can be put in a simultaneous eigenstate of all X and Z-type stabilizers.

In the operational cycle of the planar code, ancilla qubits are continuously interacted with neighbouring data qubits and their states repeatedly measured to probe for error occurrences. On the other hand, data qubits are measured primarily to carry out logical operations necessary for computation.

Throughout the remainder of this thesis, we will adopt the planar code representation as depicted in Figure 1.2 (b). In this notation, data qubits are designated by white dots, while red and blue squares represent X and Z-type stabilizers, respectively.

1.2.2. ERROR DETECTION

Let us now investigate the consequences of a Pauli error denoted as $E \in \mathscr{P}_n$ when it affects a state within the codespace with all stabilizers have eigenvalues of $+1$. The impact of such an error can be discerned by evaluating its commutation relation with a given stabilizer S. Specifically, if the error E commutes with S, we obtain

$$SE|\psi\rangle = ES|\psi\rangle = E|\psi\rangle\,, \tag{1.27}$$

indicating that the measurement outcome will correspond to a $+1$ eigenvalue. Conversely, if the error E anti-commutes with S, we find

$$SE|\psi\rangle = -ES|\psi\rangle = -E|\psi\rangle\,, \tag{1.28}$$

signifying that the measurement outcome will result in a -1 eigenvalue.

Consequently, errors impacting stabilizer codes are discernible by taking measurements of the stabilizers, which yield what is known as the syndrome, $\sigma(E)$. In the event that no errors have occurred, all stabilizer measurements will produce a $+1$ eigenvalue. Any deviation from this, manifested as a -1 outcome, alerts us to the presence of errors. The code distance d is defined as the minimum number of independent errors required to change the state of the logical qubits without being detected. To correct the detectable errors, the employment of a decoder becomes necessary, which is used to determine the location and nature of the errors, based on the given syndrome.

1.2.3. QUANTUM ERROR CORRECTION CYCLES

In the context of surface codes, quantum error correction cycles refer to a series of operations performed on the qubits of the code to detect errors. These cycles typically involve resetting the ancilla qubits, executing the stabilizer circuit, and measuring the ancilla qubits to obtain the syndrome. Throughout the rest of the thesis, the terms *QEC cycle* is used to refer to these error correction cycles.

1.2.4. CREATING LOGICAL QUBITS

Numerous approaches exist for constructing logically well-defined qubits on planar surface codes. One such technique entails encoding multiple qubits into a large surface code region using defects, and executing logical operations by braiding these defects [4]. However, this method imposes a substantial overhead in terms of the number of qubits and physical operations required. For comparison, a planar qubit of distance d is typically around three times smaller than a defect-based qubit with the same distance. Furthermore, it has been demonstrated that defects and braids in the surface code should be deprecated due to their significant overhead [5]. A more resource-efficient alternative is to encode logical qubits using a twist defect [6]. This method requires fewer physical qubits per logical qubit, while also facilitating the implementation of a full set of Clifford gates without the need for state distillation [7].

In this thesis, our focus is on the standard planar surface code due to its inherent simplicity and efficiency. The stabilizer group $\mathscr{S}$ leaves two unconstrained degrees of freedom in the Hilbert space of all data qubits, defined as the codespace $\mathscr{T}(\mathscr{S})$. The logical qubit resides within this codespace, and the orientations of the logical $|0\rangle$ and $|1\rangle$ states are determined by a pair of logical operators that anticommute with each other, defined as X_L and Z_L, respectively, see Figure 1.3.

The logical operator $Z_L = Z_1 Z_2 Z_3 Z_4$ forms a Z-chain connecting two Z-type boundaries. Similarly, the logical operator $X_L = X_1 X_5 X_6 X_7$ forms a X-chain connecting two X-type boundaries. It is important to note that any Z chain connecting two Z-type boundaries and commuting with all stabilizers can be considered a valid logical operator $\hat{Z}_L$, and the same applies for X chains in relation to X_L.

1.2.5. LOGICAL QUBIT INITIALIZATION AND MEASUREMENT

The logical qubits can be fault-tolerantly initialized and measured in the logical X and Z basis, respectively. However, initialization and measurement of the surface code in the Y basis, although achievable through the distillation of the logical $|+i\rangle$ state, demand considerable resources [8, 9].

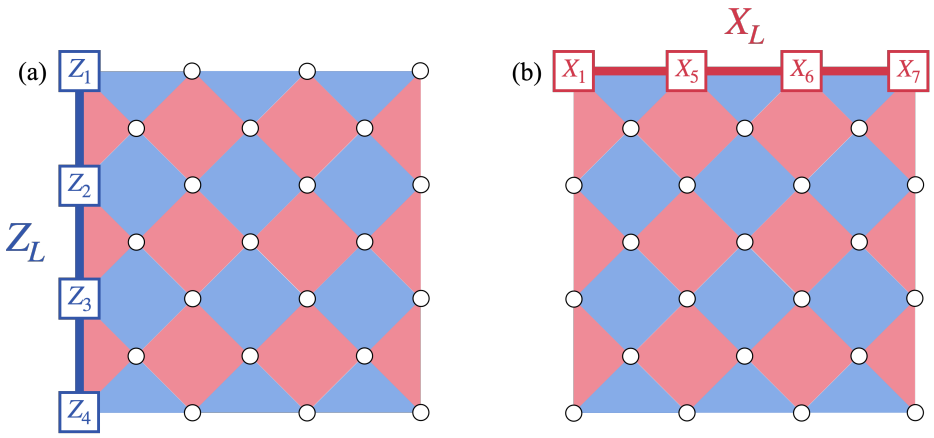

Figure 1.3: Logical operations of the square surface code. (a) An example of a logical operator $\hat{Z}_L$, which is implemented by applying physical Z gates on a chain of data qubits that connects two Z-type boundaries. (b) An example of a logical operator $\hat{X}_L$, which is implemented by applying physical X gates on a chain of data qubits that connects two X-type boundaries.

To initialize the logical qubit to $|\bar{0}\rangle$, we start by resetting all the data qubits to the state $|0\rangle$, and subsequently performing all parity checks. The measurement outcome for the Z stabilizers should remain trivial, i.e., $+1$, whereas the outcomes for the X stabilizers will be random. The resulting logical state is then defined as $|\bar{0}\rangle$, as it is an eigenstate of any Z_L operators. Similarly, to initialize the logical qubit to $|\bar{+}\rangle$, we begin by resetting all the data qubits to the state $|+\rangle$, followed by performing all parity checks. In this scenario, the measurement outcome for the X stabilizers should remain trivial, i.e., $+1$, while the outcomes for the Z stabilizers will be random. The resulting logical state is then defined as $|\bar{+}\rangle$, since it is an eigenstate of any X_L operators.

The process of measuring a logical qubit is essentially the converse of initialization. When conducting a measurement in the logical X (Z) basis, we perform physical X (Z) basis measurements on all data qubits. The outcome of the logical measurement is determined by the product of the measurement results of the data qubits on which a X_L (Z_L) gate operates. It is expected that the product of the measurements of each X (Z) stabilizer and its neighboring data qubits results in a trivial outcome. Any non-trivial product is a clear sign of error events, warranting the deployment of a decoder for rectification. This check is crucial to the assurance of fault-tolerant logical measurements.

1.3. RESEARCH MOTIVATION

Quantum computing has the potential for solving problems that are computationally infeasible for classical computers. However, the realization of this technology is hindered by significant obstacles, primarily the susceptibility of quantum systems to errors due to its unavoidable coupling with the environment and imperfect operations. Surface code quantum computation has emerged as one of the most promising paths to achieve fault-

tolerant quantum computing. However, several critical challenges need to be addressed to realize the full potential of this technology, which serves as the primary motivation for our research.

The primary aim of our research is to optimize the required number of QEC cycles to reach a desirable logical error rate . While QEC is crucial to maintaining the integrity of quantum information, excessive QEC cycles can slow down computation and increase the chances of errors in idle logical qubits. Therefore, finding when and how many rounds of QEC are necessary is important.

In addition, the order of syndrome measurements within each QEC cycle for the square surface code can also impact the logical error rates. This motivates our exploration into identifying the optimal fault-tolerant order of syndrome measurements for square surface code for different scenarios.

The road to practical quantum computing is fraught with challenges, but by addressing these key issues in surface code quantum computation, we aim to contribute to the advancement of this promising technology.

2

DECODING THE SURFACE CODE

2.1. DECODING PROBLEM

2.1.1. DETECTORS AND SYNDROME DEFECTS

In stabilizer circuits, errors are detected by monitoring deviations from the behavior of noiseless circuits. This means that any divergence from the ideal outcome of the corresponding noiseless circuit should be flagged as a potential error. Specifically, in circuits, there are often small sets of measurements that are expected to exhibit deterministic parity when executed without any noise. The parities of these sets of measurement outcomes are referred to as *detectors*. When the observed parity deviates from the expected parity, we denote the observed parity as nontrivial. Conversely, if the observed parity aligns with the expected one, it is labeled as trivial. The nontrivial parity triggers a "click" in the detector, leading to a *syndrome defect*. To represent this, detectors with nontrivial and trivial parities are respectively assigned values of 1 and 0. Upon examining the lattice of detectors, we observe that the 1 values appear as sparse anomalies amidst the 0 values, which is why these occurrences are referred to as defects.

The syndrome defects are determined by the module-2 addition of two consecutive ancilla measurement outcomes if we reset the ancilla qubits to $|0\rangle$ at the start of each QEC cycle. Let $m_{S,r}$ represent the measurement outcome of stabilizer generator S in the r-th round. Note that here we define the measurement outcome $m_{S,r} = 0$ if $S|\psi\rangle = |\psi\rangle$ and to $m_{S,r} = 1$ if $S|\psi\rangle = -|\psi\rangle$. The syndrome defect for stabilizer S at the r-th round can be expressed as:

$$\sigma_{S,r} = m_{S,r} \oplus m_{S,r-1} \tag{2.1}$$

This is because the measurement outcomes of a specific ancilla qubit remain consistent in the absence of noise. However, it is important to note that syndrome defects may not always be simply described by Eq. (2.1). For example, when code or stabilizer deformation is performed, care must be taken when configuring the detectors. We will discuss this in Chapter 4.

In the surface code, it is common to assume that qubit errors, which are typically represented by Pauli operators, can occur at various locations within the circuit, such

as single-qubit gates, two-qubit gates, idle locations, measurements, and resets. However, it is important to note that measurement errors are generally non-Pauli. A general measurement error can be thought of as a combination of a classical readout error and a after-measurement qubit error. The classical readout errors flip the measurement outcome, and we can disregard the after-measurement qubit error when the ancilla qubits are reset between QEC rounds. (It's worth noting that in DiCarlo Lab, they typically do not reset ancilla qubits between QEC rounds). We assume that errors can be represented as events with associated probabilities. In the following contexts, the probability p_{meas} associated with a measurement error refers specifically to the probability of the classical readout outcome being flipped.

In an ideal scenario without error events, no syndrome defects would be observed for all stabilizers, denoted as $\sigma_{S,r} = 0$ for all $\forall S \in \mathscr{S}$ and $r \in \mathbb{Z}^+$. This is due to the fact that $m_{S,r} = 0$ for all $S \in \mathscr{S}$ and $r \in \mathbb{Z}^+$, given that the logical qubit resides in the +1 eigenstate of all stabilizers.

Now, let's examine the situation when qubit errors occur. It has been demonstrated that a single Pauli qubit error event can generate at most two X-type defects and two Z-type defects for the surface code [10]. If the error takes place on one of the data qubits located at the boundary of the code, it is possible that the error leads to only one defect. In such cases, we can associate the defects generated by the same single error with an *error edge*, as depicted in Figure 2.1, where we display only X-type defects. For errors that result in just one defect, we can pair this defect with a virtual node.

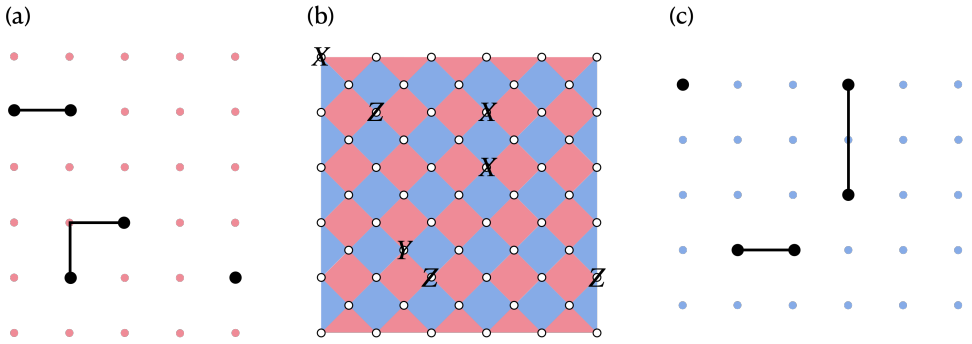

Figure 2.1: Pair of defects. (a) X-type of defects on the lattice of X-stabilizers (in red). (b) A distance-6 square surface code, with Pauli qubit errors occurring on data qubits. (c) Z-type of defects on the lattice of Z-stabilizers (in blue). When the probability of qubit errors is low, the error edges, which represent locations where errors occur, appear sparsely on the lattice. In such scenarios, it becomes relatively easy to pair the defects with error.

However, a measurement error on an ancilla qubit can flip the measurement outcome. Therefore, a measurement error will indicate a defect where there shouldn't have been one, and eliminate a defect where there should have been one. To conceptually distinguish the defects caused by qubit errors and measurement errors, we adopt the terminology "genuine defects" and "ghost defects" introduced in [11]. Genuine defects are caused by qubit errors and have strongly correlated positions on the lattice, while

ghost defects are caused by measurement errors and generally appear randomly if we assume measurements are independent, as shown in Figure 2.2.

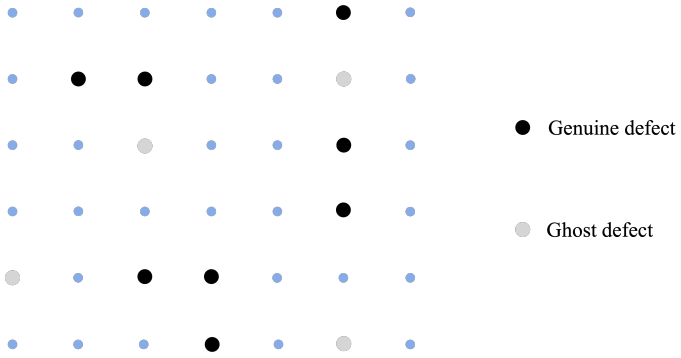

Figure 2.2: Genuine defects and ghost defects. The light blue dots represent the lattice of the Z-stabilizer graph. Darkly shaded circles denote genuine defects caused by single qubit errors, while lightly shaded circles indicate ghost defects resulting from measurement errors. Typically, genuine defects appear in pairs, except at boundaries where they can appear singly. In these instances, we pair the solitary genuine defect with a virtual defect, as depicted in Figure 2.5 (a). Conversely, ghost defects usually appear individually on the lattice.

Ghost defects causes a challenge in decoding for quantum memory, as they cannot be distinguished from genuine defects in real-world experiments when one round of quantum error correction is performed. This complicates the process of accurately determining the location and nature of the qubit errors experienced by the data qubits within the code.

2.1.2. DEFECTS AND ERROR EDGES IN SPACETIME

One approach to addressing faulty measurements is to repeat the measurement process, thereby increasing our confidence in the outcome. However, this method can introduce new errors during the repeated measurements. To address this issue, one approach is to perform multiple rounds of quantum error correction (QEC), effectively providing more syndrome information for decoding. This issue will be discussed in more detail in Chapter 4.

In a standard QEC experiment, a logical state encoded by the code is prepared, and QEC rounds are performed repeatedly for r rounds. The stabilizers that are measured repeatedly, and the final measurements on data qubits provide information about where and when errors occurred. This information is used by the decoder to determine whether the logical measurement outcome should be flipped or not. To effectively analyze the syndrome information, it is helpful to introduce a three-dimensional spacetime lattice of the stabilizers, with the third dimension representing integer-valued time and syndrome measurements occurring between each t and $t+1$, as shown in Figure 2.3. We introduce two fundamental error edges on the lattice:

- Space-like edges: error edges that connect defects that occur at the same time but at different spatial locations on the lattice. Space-like edges correspond to single errors on qubits and are typically horizontal connections on the lattice.

- Time-like edges: error edges that connect defects that occur at different times but at the same spatial location. Time-like edges correspond to measurement errors, and are typically represented as vertical connections on the lattice.

The spacetime lattice depicted in Figure 2.3 is plotted under the effects of the phenomenological noise model, as discussed in Chapter 1.1.1. It is important to note that under a more realistic circuit-level noise model, additional spacetime-like diagonal edges can be observed on the spacetime lattice.

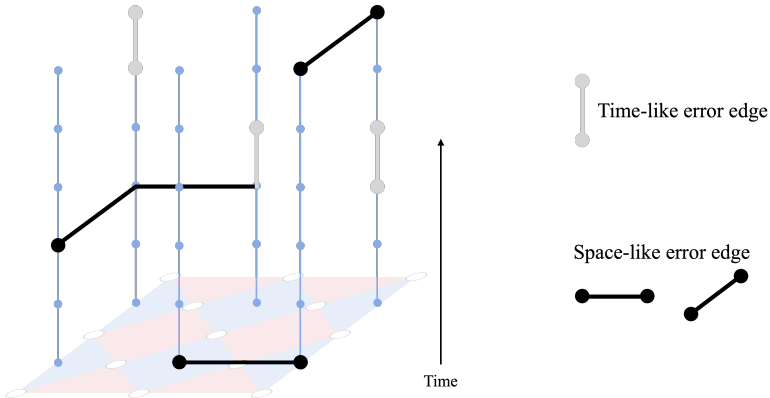

Figure 2.3: Defects and error edges in a three-dimensional spacetime volume of Z-type stabilizers under phenomenological noise model (see Chapter 1.1.1). A distance-3 square surface code undergoes repeated quantum error correction for several rounds. Each blue node represents a specific X stabilizer at a particular time. Vertical gray lines connecting ghost defects represent time-like error edges, while horizontal dark lines connecting genuine defects represent space-like error edges.

Consider a qubit error series $\mathbf{E} = (E_1, E_2, \ldots, E_t, \ldots, E_T)$, where each element $E_t \in \mathscr{P}_n$ represents qubit errors occurring at time t. Furthermore, let us define a measurement error series $\mathbf{M} = (M_1, M_2, \ldots, M_t, \ldots, M_T)$, with each element M_t being a binary vector, whose i-th element set to 1 if a measurement error occurs in the i-th stabilizer and 0 otherwise. These qubit and measurement error series collectively result in syndrome defects $\sigma(\mathbf{E}, \mathbf{M})$ throughout the spacetime volume.

2.2. DECODING ALGORITHMS

The aim of decoding quantum error correction codes is the restoration of the logical state of the code, predicated on a provided syndrome $\sigma(\mathbf{E}, \mathbf{M})$, via the implementation of specific operators on the spacetime lattice. To illustrate the fundamentals of decoding, consider Figure 2.4. Within (a), a Pauli error, denoted as $E \in \mathscr{P}_n$, with its corresponding syndrome $\sigma(E)$, leads to the corruption of the logical state from $|\phi\rangle$ to $E|\phi\rangle$. Subsequently, a decoding algorithm leverages the syndrome $\sigma(E)$ to select an appropriate correction operator, $E_c \in \mathscr{P}_n$, as shown in (b). This operator transforms the state from $E|\phi\rangle$ to $E_c E|\phi\rangle$. It is important to notice that applying a correction on a different string is permissible, as long as the resulting loop is homologically trivial (as demonstrated by

the right loop in Figure 2.4 (c)). This implies that $E_c E \in \mathcal{S}$, meaning the overall operation of the correction and the error is equivalent to the product of the encircled stabilizer operators. However, if the error and its correction collectively form a logical operator (as illustrated by the left path in Figure 2.4 (c)), represented as $E_c E \in \mathcal{C}(\mathcal{S}) \backslash \mathcal{S}$, an undetectable logical error may potentially be introduced.

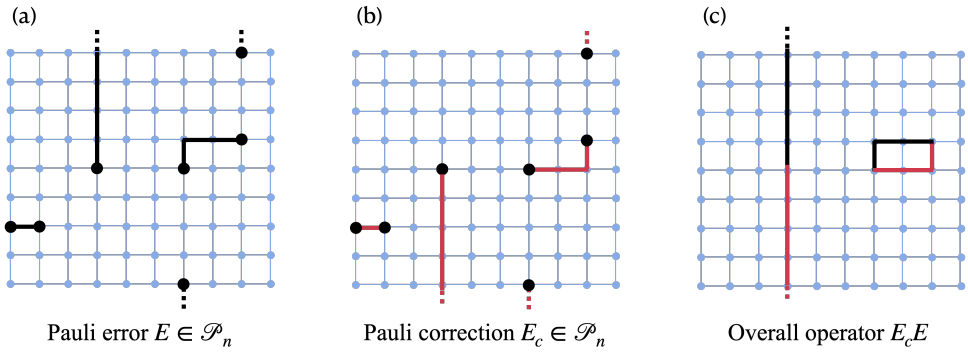

Figure 2.4: A decoding algorithm receives a syndrome $\sigma(E)$ as input and outputs a correction operator E_c on the Z-stabilizer lattice. (a) presents the syndrome defects induced by a qubit Pauli error $E \in \mathcal{P}_n$ on the Z-graph of a distance-10 square surface code. Error edges connecting to the boundary are indicated by dotted lines. (b) showcases the correction operator $E_c \in \mathcal{P}_n$ determined by the decoder, which is represented by red edges. (c) The correction operator is combined with the errors, yielding two types of outcomes: a trivial loop and a path that connects two boundaries. The path connecting the boundaries can potentially lead to a logical error.

In real-world scenarios, the process is considerably more complex. The presence of multi-qubit correlated errors can give rise to space-like error hyperedges (a hyperedge is a generalization of an edge that can connect more than two vertices), which is in contrast to the space-like error edge produced by a single qubit error. Additionally, a given pair of defects can potentially be generated by multiple error strings, further complicating the decoding process. While an optimal algorithm, the Maximum Likelihood Decoder (MLD), does exists to simply correct the most-likely errors, its application is not straightforward. The MLD operates on the principle of identifying and finding the error E compatible with the syndrome σ that holds the highest probability of occurrence. However, the required computational resource of the MLD grows exponentially with respect to the number of qubits, making it impractical for large-scale systems [12, 13]. Therefore, the heart of designing decoding algorithms for quantum error correction is to find a balance between the its accuracy and computational complexity.

2.2.1. MINIMUM-WEIGHT PERFECT MATCHING

The Minimum-Weight Perfect Matching (MWPM) formulation of decoding addresses the task of pairing defects and has been developed based on two fundamental assumptions [11, 4]:

- A single error event results in at most two defects on the X-graph or Z-graph, respectively (We note that a single error event can be also be a two-qubit error). These errors are uncorrelated, and thus they can be treated as independent events.

- The optimal pairing is the one that minimizes the total error weight.

In light of these assumptions, the MWPM decoding problem can be seen as a graph problem. Specifically, we construct a matching graph $G = (V, E)$, as depicted in Figure 2.5 (a), where each vertex $v \in V$ represents a detector and each edge $e \in E$ signifies a single Pauli error event. On the boundary, where single errors anti-commute with only one stabilizer, we incorporate additional boundary nodes (illustrated as hollow squares). All boundary nodes are connected by zero-weight edges, and for simplicity, they can be merged into a single node, as shown on the left side of the lattice in Figure 2.5 (a). This setup ensures that the single defects on the boundary can be paired with a virtual defect located on the boundary node.

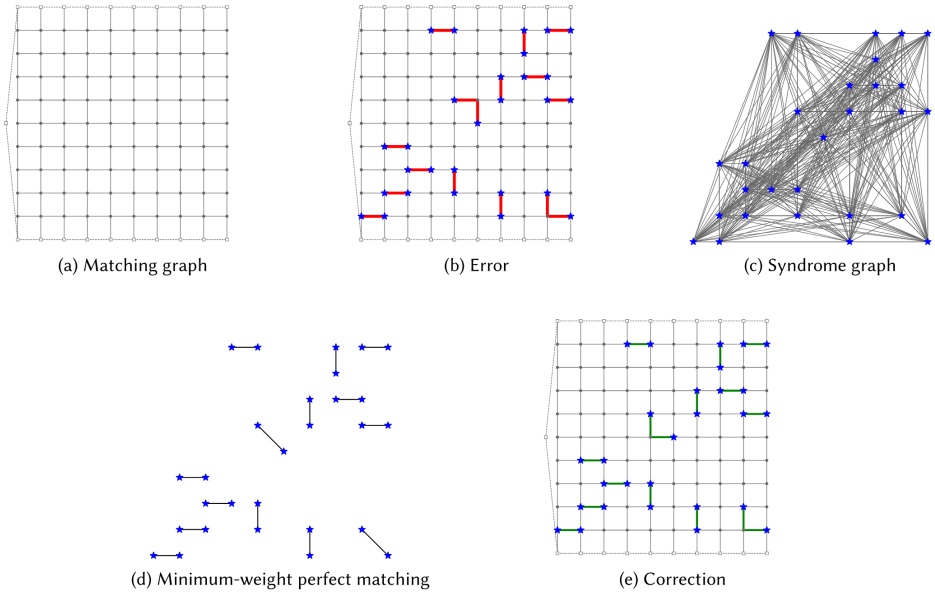

Figure 2.5: This figure is reproduced from [14]. Stages of the MWPM decoder for a distance-10 surface code. (a) Matching graph, with each dark node denotes a detector and boundary nodes denoted with hollow squares. Each edge represent a single error event. Note that a single error event can also be a two-qubit error. (b) Errors are denoted by red edges, and blue stars represents the corresponding defects caused by the error. (c) Syndrome graph which has a node for each defect. (d) The MWPM of the syndrome graph. (e) The correction output by the MWPM decoder.

Each edge $e \in E$ is assigned a weight $w_e = \log\left(\frac{1-p_e}{p_e}\right)$, where $p_e < \frac{1}{2}$ represents the probability of a single error event. As p_e diminishes, the weight w_e increases significantly, and as p_e approaches $\frac{1}{2}$, the weight w_e converges to zero. This therefore indicates that errors with higher probabilities are attributed lower weights.

Figure 2.5 (b) shows an example of an error (red edges) and the corresponding defects (blue stars). After the syndrome measurements, a syndrome graph S can be constructed (see 2.5 (c)), where each vertex corresponds to a defect. For a given qubit error $E_q \in \mathcal{P}_n$, which is associated with a subset of edges $R \in E$, its probability of occurrence can be

represented as:

$$p(E_q) = \prod_{e \in E \setminus R} (1 - p_e) \prod_{e \in R} p_e = \prod_{e \in E} (1 - p_e) \prod_{e \in R} \left(\frac{p_e}{1 - p_e} \right) \tag{2.2}$$

The weight of E_q can then be derived as follows:

$$w(E_q) = \log\big(p(E_q)\big) = \sum_{e \in E} \log(1 - p_e) - \sum_{e \in R} w_e \tag{2.3}$$

Within the syndrome graph S, if p_e is same for all edges, the weight assigned to each edge between two defect nodes is determined by the length of the shortest path between the associated detectors in the original matching graph G, which can be identified using Dijkstra's algorithm [15].

The subsequent stage involves perfectly matching all the nodes in S, such that the sum of the weights of all edges is minimized. This optimization task, referred to as the minimum-weight perfect matching problem (MWPM), can be resolved using the Blossom algorithm [16]. The worst-case computational complexity of this algorithm is $\mathcal{O}(N^3)$, where N denotes the number of physical qubits.

2.3. IMPROVED STABILIZER MEASUREMENT CIRCUITS

2.3.1. PROPER STABILIZER MEASUREMENT CIRCUITS

The optimization of QEC cycle times can be approached through the simultaneous execution of X-type and Z-type stabilizer measurements. However, this methodology demands meticulous attention to the sequence in which the CZ gates are applied within each parity check. This is because the outcomes of the two types of ancilla qubits can be randomized, when the measured X-stabilizer does not commute with the measured Z-stabilizers anymore. Such non-commutation undermines the properness of this specific stabilizer measurement circuit. In this context, a stabilizer measurement circuit is considered *proper* when the measurement of the ancilla qubits is in fact equivalent to the measurement of the corresponding stabilizer.

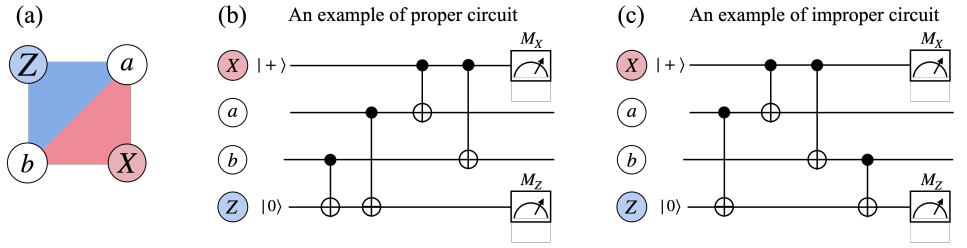

Figure 2.6: (a) depicts a pair of data qubits a and b (represented by white circles) along with an X-type ancilla qubit (blue circle) and a Z-type ancilla qubit (red circle). These ancilla qubits are utilized to measure the X_aX_b and Z_aZ_b stabilizers, as indicated by the red and blue patches, respectively. (b) An example of a proper stabilizer measurement circuit. The data qubits a and b interact first with the Z-type ancilla qubit, respectively. Subsequently, the data qubits a and b interact with the X-type ancilla qubit, respectively. (c) An example of an improper stabilizer measurement circuit.

Let's consider an example of proper and improper stabilizer measurement circuits shown in Figure 2.6 (b) and (c), respectively. These circuits are designed to measure the stabilizers X_aX_b and Z_aZ_b in Figure 2.6 (a). To illustrate this concept, we are going to use the stabilizers of a 4-qubit system to identify the state of the qubits in Figure 2.6 (a), and we track the evolution of these qubit state by examining the evolution of their stabilizers in the Heisenberg picture [17].

AN EXAMPLE OF PROPER STABILIZER MEASUREMENT CIRCUIT

For the proper stabilizer measurement circuit shown in Figure 2.6 (b), let's examine the evolution of the stabilizers: First, we focus on the X-type ancilla qubit. It is initially prepared in the state $|+\rangle$, implying that the initial state of the system is an eigenstate of the stabilizer $X_xI_aI_bI_z$. The evolution of this stabilizer is as follows:

$$X_xI_aI_bI_z \xrightarrow{CNOT_{b,z}} X_xI_aI_bI_z \xrightarrow{CNOT_{a,z}} X_xI_aI_bI_z \xrightarrow{CNOT_{x,a}} X_xX_aI_bI_z \xrightarrow{CNOT_{x,b}} X_xX_aX_bI_z \tag{2.4}$$

Next, the X-type ancilla qubit is measured in the X basis. Since the X-type ancilla qubit was prepared in the state $|+\rangle$, its measurement is equivalent to measuring $X_aX_bI_z$.

Similarly, let's analyze the Z-type ancilla qubit in the proper stabilizer measurement circuit. It is prepared in the state $|0\rangle$, indicating that the initial state of the system is an eigenstate of the stabilizer $I_xI_aI_bZ_z$. The evolution of this stabilizer is as follows:

$$I_xI_aI_bZ_z \xrightarrow{CNOT_{b,z}} I_xI_aZ_bZ_z \xrightarrow{CNOT_{a,z}} I_xZ_aZ_bZ_z \xrightarrow{CNOT_{x,a}} I_xZ_aZ_bZ_z \xrightarrow{CNOT_{x,b}} I_xZ_aZ_bZ_z \tag{2.5}$$

The Z-type ancilla qubit is then measured in the Z basis. Since the Z-type ancilla qubit was prepared in the state $|0\rangle$, its measurement is equivalent to measuring $I_xZ_aZ_b$.

AN EXAMPLE OF IMPROPER STABILIZER MEASUREMENT CIRCUIT

However, for the improper stabilizer measurement circuit shown in Figure 2.6 (c), we observe that the final measurement outcomes of the ancilla qubits can be random.

Let's examine the evolution of the stabilizers. For the X-type ancilla qubit:

$$X_xI_aI_bI_z \xrightarrow{CNOT_{a,z}} X_xI_aI_bI_z \xrightarrow{CNOT_{x,a}} X_xX_aI_bI_z \xrightarrow{CNOT_{x,b}} X_xX_aX_bI_z \xrightarrow{CNOT_{b,z}} X_xX_aX_bX_z \tag{2.6}$$

The X-type ancilla qubit is then measured in the X basis. Since the X-type ancilla qubit is prepared in the state $|+\rangle$, its measurement is equivalent to measuring $X_aX_bX_z$. This measurement outcome can be interpreted as: the expectation value of the X-type ancilla qubit's measurement depends on the expectation value of X on the Z-type ancilla qubit, which is prepared in the state $|0\rangle$.

Similarly, let's examine the evolution of the stabilizers for the Z-type ancilla qubit in the improper stabilizer measurement circuit:

$$I_xI_aI_bZ_z \xrightarrow{CNOT_{a,z}} I_xZ_aI_bZ_z \xrightarrow{CNOT_{x,a}} Z_xZ_aI_bZ_z \xrightarrow{CNOT_{x,b}} Z_xZ_aI_bZ_z \xrightarrow{CNOT_{b,z}} Z_xZ_aZ_bZ_z \tag{2.7}$$

The Z-type ancilla qubit is then measured in the Z basis. Since the Z-type ancilla qubit is prepared in the state $|0\rangle$, its measurement is equivalent to measuring $Z_xZ_aZ_b$. This

measurement outcome can be interpreted as: the expectation value of the Z-type ancilla qubit's measurement depends on the expectation value of Z on the X-type ancilla qubit, which is prepared in the state $|+\rangle$.

Therefore, in this case the measurement outcomes of the two ancilla qubits can be random, hence the measurements are improper.

2.3.2. FAULT-TOLERANT STABILIZER MEASUREMENT CIRCUITS

The issue of proper fault-tolerant stabilizer measurement has been rigorously explored for the rotated surface code [18, 19], both theoretically and experimentally. Nonetheless, there is a lack of research addressing this problem in the context of the square surface code, which, unlike the rotate-symmetric rotated surface code, is characterized by mirror-dual symmetry. That is, by folding the surface code along the diagonal of the square code, we can pair an X-type stabilizer with a corresponding Z-type stabilizer. The disparate symmetry properties inherent to the rotated and square surface codes make it imperative to extend investigations to elucidate the nature of proper stabilizer measurement for the square surface code.

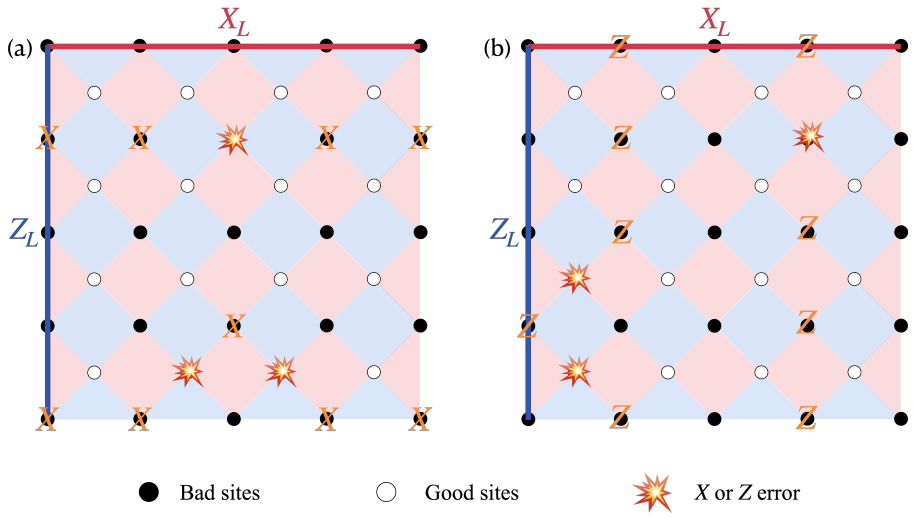

Figure 2.7: Illustration of the lattice of data qubits in a distance-5 square surface code, highlighting the distinction between good and bad sites. Bad sites are represented by black circles, while good sites are depicted as white circles. The blow-up symbol denotes X or Z errors. (a) X errors on bad sites have a more detrimental impact than those on good sites in terms of inducing a logical X error. When an X error occurs on a bad site (as shown in the blow-up symbol above), it necessitates the presence of $d-1$ additional X errors or correction operators to form a logical X chain. Conversely, for X errors occurring on good sites (as shown in the blow-up symbol below), there will always be a requirement for d additional X errors or correction operators to form a logical X chain. (b) Similarly, Z errors on bad sites have a greater detrimental effect than those on good sites in terms of resulting in a logical Z error. When a Z error occurs on a bad site (as indicated by the right blow-up symbol), it necessitates the presence of $d-1$ additional Z errors or correction operators to form a logical Z chain. In contrast, for Z errors occurring on good sites (as indicated by the left blow-up symbols), there will always be a need for d additional Z errors or correction operators to form a logical Z chain.

As shown in Figure 2.7, data qubits of square surface code can sit on either *good*

site (white circles) or *bad site* (black circles) of the lattice. In the context of inducing a logical X error, physical X errors on bad sites have a more detrimental impact compared to those on good sites (see Figure 2.7 (a)). Specifically, when an X error occurs on a bad site, it requires the presence of $d-1$ additional X errors or correction operators to form a logical X chain. On the other hand, for X errors occurring on good sites, there will always be a requirement for d additional X errors or correction operators to form a logical X chain. Similarly, in terms of inducing a logical Z error, physical Z errors on bad sites have a greater detrimental effect compared to those on good sites (see Figure 2.7 (b)). When a Z error occurs on a bad site, it necessitates the presence of $d-1$ additional Z errors or correction operators to form a logical Z chain. Conversely, for Z errors occurring on good sites, there will always be a need for d additional Z errors or correction operators to form a logical Z chain.

Hence, when designing the fault-tolerant order of stabilizer measurement, it is important to minimize the probability of encountering X or Z errors on data qubits located at bad sites. This can help mitigate the detrimental effects associated with these errors and improve the overall fault-tolerance of the code.

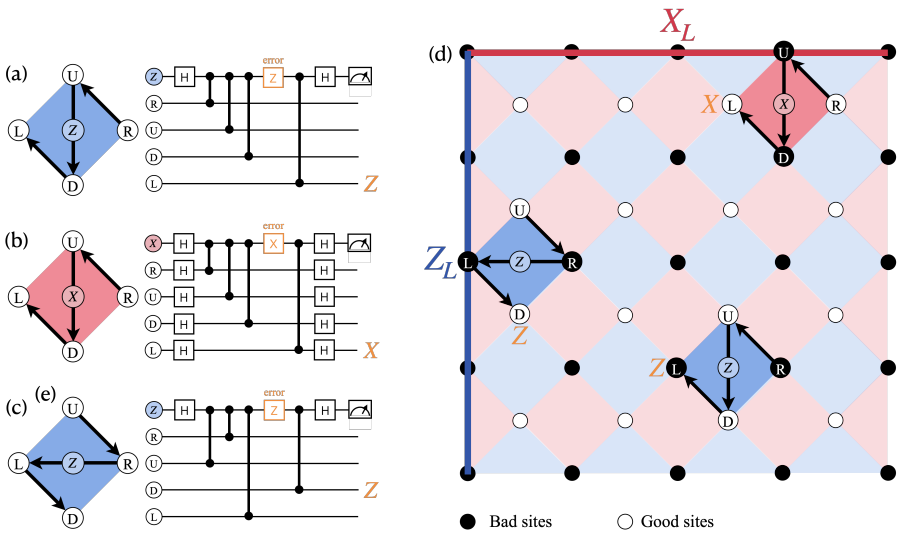

Figure 2.8: (a) The sequence of CZ gates for Z-type stabilizers, resulting in a Z error on a bad site. (b) The sequence of CZ gates for X-type stabilizers, resulting in an X error on a good site. (c) The sequence of CZ gates for Z-type stabilizers, resulting in a Z error on a good site.

The prescribed order of CZ gates for each type of stabilizer, represented as N-like in Figure 2.8 (a-c), ensures that these circuits maintain fault-tolerance while only inducing at most two defects on the Z or X graph [18]. In Figure 2.8 (a), we observe that a Z error on a Z-type ancilla qubit, following the third CZ gates, propagates onto one data qubit that lied on a bad site. Similarly, as shown in Figure 2.8 (b) (and (c)), a Z (X) error on a Z-type (X-type) ancilla qubit post the third CZ gates impacts one data qubit that lies on a good site.

In an attempt to avert the occurrence of logical errors, we incline towards adopting the order of CZ gates as displayed in Figure 2.8 (b) and (c) for the X-type and Z-type syndrome measurements, respectively. However, the circuit is improper if the CZ gates of X-stabilizer and Z-stabilizer are executed in parallel since the measured X-stabilizer does not commute with the measured Z-stabilizers in this case. Conversely, parallel operation of the CZ gates of X-stabilizer and Z-stabilizer is feasible since the resulting circuit is proper if we adhere to the CZ gates' order presented in Figure 2.8 (a) and (b) for the X-type and Z-type syndrome measurements, respectively. It should be noted that this methodology, although facilitating parallel execution, renders the code more susceptible to logical Z errors. This is because data qubits on bad sites are more likely undergoing Z errors as shown in Figure 2.8.

We categorize these two distinct choices as Type-I and Type-II syndrome measurements, illustrated in Figure 2.9.

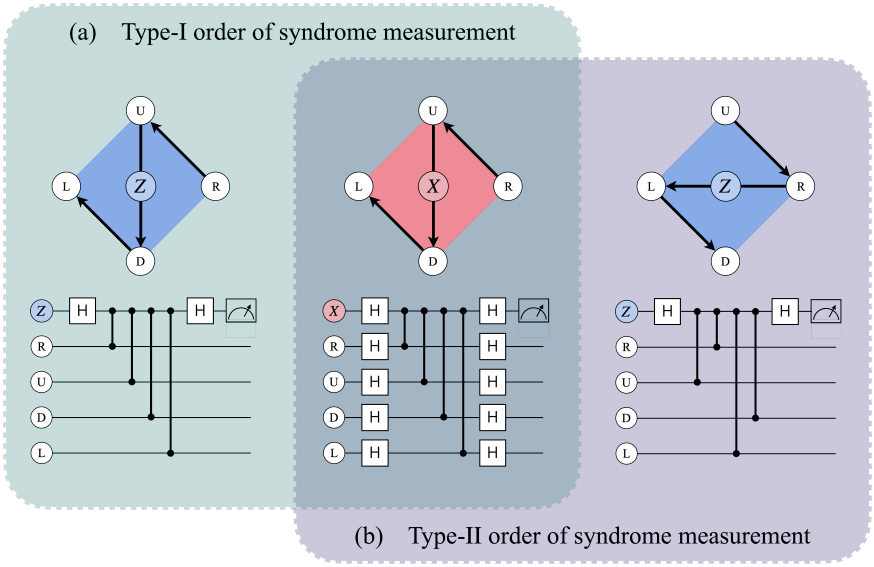

Figure 2.9: Two types of order of syndrome measurement. (a) Type-I: Z-type parity checks could potentially lead to Z errors along the direction of the Z_L chain, but X-type parity checks do not induce X errors parallel with the direction of X_L chain. (b) Type-II: X-type (Z-type) parity checks do not induce X (Z) errors parallel with the direction of X_L (Z_L) chain.

The temporal profiles of each QEC cycle utilizing Type-I and Type-II syndrome measurements are depicted in Figure 2.10 and 2.11, respectively. Under the Type-I scheme, the CZ gates for both X- and Z-stabilizers can be executed simultaneously, culminating in a depth-9 circuit for each QEC cycle. On the other hand, to make sure the circuit is proper, the Type-II approach mandates separate execution of CZ gates for X- and Z-stabilizers, which extends the circuit to a depth of 12 for each QEC cycle. Although the Type-II syndrome measurement confers an advantage in terms of fault-tolerance, it imposes more potential error events on physical qubits due to the extended time each QEC cycle takes and the increased number of required H gates.

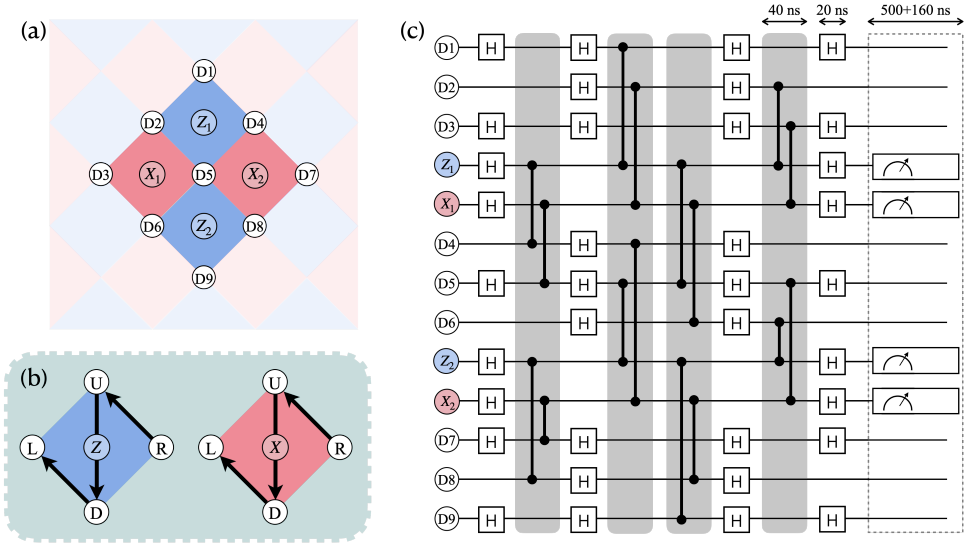

Figure 2.10: Depiction of a depth-9 quantum circuit for Type-I stabilizer measurement in the surface code employing CZ gates. The CZ gates inside each gray box are executed simultaneously. Typical durations for gate operation, readout, and reset procedures are indicated at the top.

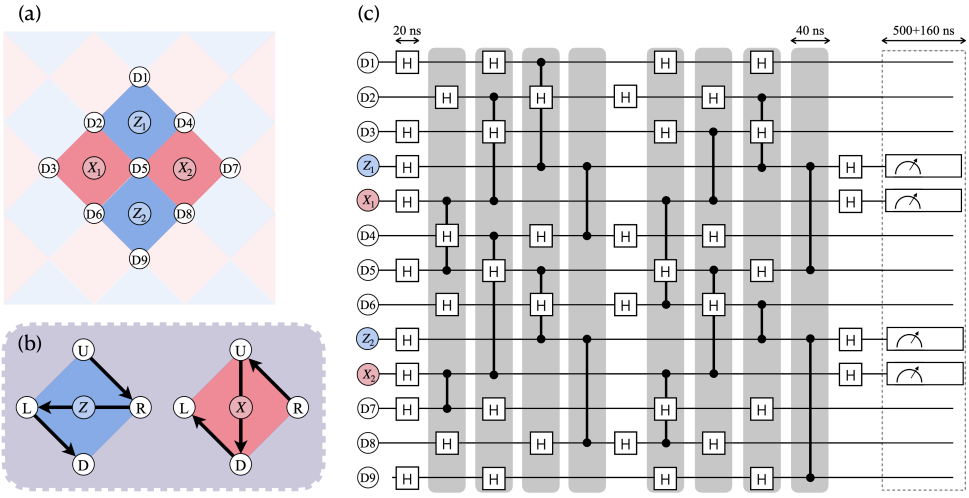

Figure 2.11: Depiction of a depth-12 quantum circuit for Type-II stabilizer measurement in the surface code employing CZ gates. The CZ gates inside each gray box are executed simultaneously. Typical durations for gate operation, readout, and reset procedures are indicated at the top.

In each QEC cycle, we adhere to typical parameters consistent with scalable superconducting quantum hardware [20, 19, 21, 22, 23]. The measurement duration is set to $500ns$ while the reset duration takes $160ns$. We further adopt a CZ gate duration of $40ns$

and a single-qubit gate duration of $20ns$.

2.3.3. COMPARATIVE ANALYSIS OF TWO SYNDROME MEASUREMENT TYPES

Two distinct quantum memory experiments are conducted, simulating both Type-I and Type-II orderings of the CZ gates within parity checks, with the results presented in Figure 2.12. The code for the simulation can be found in Chapter 6 (Appendix).

The surface code is initialized into perfect logical states $|0\rangle$ and $|+\rangle$, represented by the orange and blue curves, respectively. In each experiment, the codes underwent d rounds of quantum error correction, followed by logical measurements in the logical Z and X basis, respectively. Python package *Stim* [1] is used to generate the circuits. We decode the syndrome information with the MWPM decoder [14] introduced in Section 2.2.1. In our simulations, the single-qubit gate error rate is assumed to be a tenth of the CZ gate error rate. Furthermore, we integrated the consideration of circuit-level noise (see Chapter 1.1.1). The idle qubits within each QEC cycle were associated with a depolarization channel whose parameter p reflects a coherence time of $T_1 = T_2 = 60\mu s$. An assignment error rate of 1% for the readout of physical qubits is also taken into account.

Figure 2.12 (a) illustrates that the logical error rate for the logical $|+\rangle$ state exceeds that of the logical $|0\rangle$ state in standard quantum memory experiments, signifying a greater likelihood for logical Z errors. This phenomenon is attributable to the order of CZ gates for Z-type stabilizers, which can create Z errors on data qubits along the Z_L chain. Furthermore, the observed thresholds of the code are approximately 0.78% and 0.90% for initialization to logical $|+\rangle$ and $|0\rangle$ states, respectively. These findings elucidate the asymmetric robustness against X and Z errors when employing the Type-I order. Conversely,

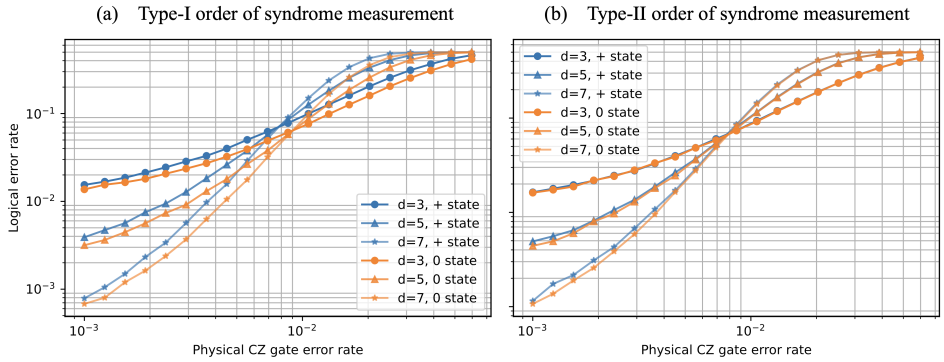

Figure 2.12: Numerical simulations of quantum memory error rates for the square surface code under two different scenarios: Type-I (a) and Type-II (b) orders of CZ gates within parity checks. The code is initialized in logical $|0\rangle$ and $|+\rangle$ states, respectively.

Figure 2.12 (b) demonstrates a comparable logical error rate for both logical $|+\rangle$ and $|0\rangle$ states under the Type-II order of syndrome measurements. Correspondingly, the threshold of the code for both initial logical states, $|+\rangle$ and $|0\rangle$, is approximately at 0.80%.

To evaluate the relative performance of Type-I and Type-II syndrome measurements, we calculate the average of the logical X and Z error rates, denoted as r_{I} and r_{II}, respec-

tively. The relative difference in average logical error rates is then computed as:

$$\frac{r_{\mathrm{II}} - r_{\mathrm{I}}}{r_{\mathrm{I}}} \tag{2.8}$$

Both circuits are simulated using a circuit-level noise model (see Chapter 1.1.1), maintaining a CZ gate error rate of 0.5% and a measurement error rate of 1%. The results are depicted in Figure 2.13.

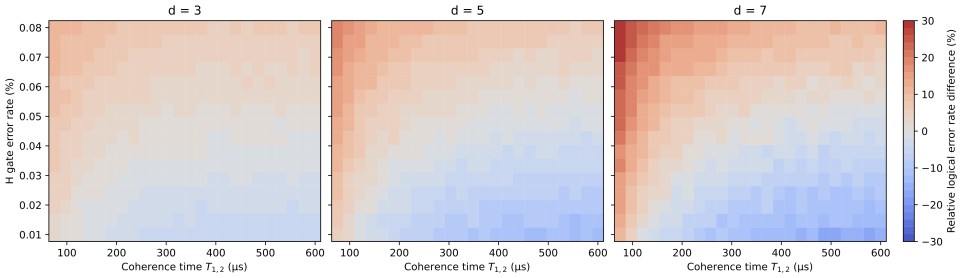

Figure 2.13: Heatmap displaying the relative average logical error rate difference, $\frac{r_{\mathrm{II}} - r_{\mathrm{I}}}{r_{\mathrm{I}}}$, in relation to H gate error and coherence time for square surface codes of distance 3,5, and 7. The CZ gate error rate is set at 0.5%, and the measurement error rate at 1%.

The performance disparity between Type-I and Type-II syndrome measurements manifests in different areas of the heat map. Type-II yields superior performance in regions characterized by larger coherence times and smaller H gate error rates, depicted as blue regions on the heat map. Conversely, Type-I performs better in areas indicated in red.

The superior performance of Type-II in these conditions can be attributed to its inclusion of more H gates within its more complex QEC circuit. Consequently, it experiences less decoherence and fewer single qubit errors in the lower-right corner of the heat map. Further, as the code distance increases, the advantage of Type-II over Type-I becomes increasingly pronounced.

3

FAULT TOLERANT UNIVERSAL QUANTUM COMPUTING

Practical universal quantum computation demands the fault-tolerant execution of a comprehensive set of gates on encoded qubits. An example of a universal gate set is the Clifford+T set, encompassing the Hadamard H, $CNOT$, S, and T gates. Notably, the Hadamard (H), $CNOT$, and S gates comprise the subset known as Clifford gates. The combination of logical qubit initialization, logical measurement, and this universal gate set enables the representation of any arbitrary quantum gate via a finite sequence of operations. Transversal logical gates provide a straightforward construction of fault-tolerant logical gates. However, it has been shown that no quantum stabilizer code that permits the transversal realization of a universal set of gates exists [24, 25].

This chapter concentrates on universal computation using patches of the planar code. Logical Pauli gates are managed entirely by classical control software, thereby eliminating the need for physical quantum operations. Execution of the logical H gate is achievable by applying a transversal H gate to data qubits, succeeded by a lattice transformation and error syndrome measurements to restore the code to its original orientation [26]. Section 3.1 introduces lattice surgery, a technique for performing joint logical measurements, aiding in the implementation of logical $CNOT$ gates. Section 3.2 highlights the utility of magic states for implementing logical S gates and non-Clifford $T = \text{diag}(1, e^{i\pi/4})$ gates. Lastly, Section 3.3 presents a transversal method for implementing logical $S = \text{diag}(1, e^{i\pi/2})$ gates [27], employing long-range CZ physical gates.

3.1. LATTICE SURGERY

Lattice surgery is a method employed for non-destructive XX and ZZ measurements on logical qubits encoded by the planar code. This technique facilitates the execution of two-qubit gates between separate sheets of the surface code. Remarkably, lattice surgery only requires standard nearest-neighbour physical interactions while preserving complete fault tolerance [28].

3.1.1. STANDARD LATTICE SURGERY

Lattice surgery essentially consists of the *merging* and *splitting* operations applied to patches of code, as illustrated in Figure 3.1. The merging operation combines two code surfaces into a single, larger code surface, while the splitting operation partitions a single code surface into two smaller surfaces.

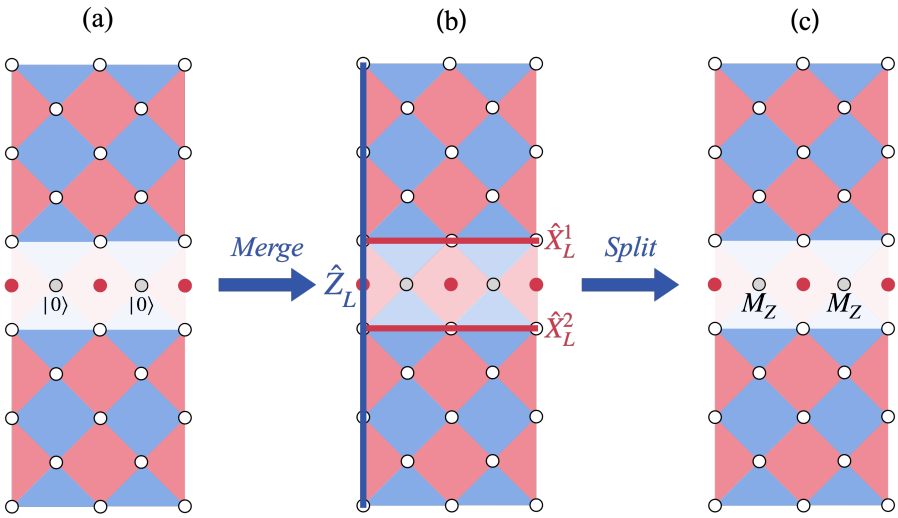

Figure 3.1: Lattice surgery for measurement of $X_L^1 X_L^2$. The procedure leverages merging and splitting operations between two sheets of code. The blue and red tiles are Z- and X-type parity checks, respectively. (a) Two distance-3 square surface codes, facing each other with X boundaries. The intermediate data qubits (grey) are initialized to $|0\rangle$. (b) The merging operation introduces intermediate data qubits (grey) and X-type ancilla qubits (red), facilitating the fusion of two smaller codes into one larger entity. (c) The large code is subsequently divided back into two smaller codes through the measurement of intermediate data qubits (grey).

MERGING OPERATION

The square surface code encompasses two distinctive types of boundaries, namely, X and Z boundaries. The X boundaries are defined as the boundaries where Z-string can terminate, and the Z boundaries are defined as the boundaries where X-string can terminate. Consequently, two different procedures for merging code surfaces arise: one involves merging the X boundaries of two sheets with the introduction of new X stabilizer generators, and the other merges Z boundaries of two sheets by introducing new Z stabilizer generators. In this discussion, we primarily focus on the X merge method.

The merge procedure is initiated by preparing the intermediate data qubits, denoted in grey, in the $|0\rangle$ state, as illustrated in Figure 3.1 (a). Following this, several rounds of quantum error correction cycles are executed on both the original and the newly introduced X stabilizers in Figure 3.1 (b). Here, three X-type stabilizers are newly introduced, represented as light pink tiles. In the merge procedure, the original Z-type stabilizer on the X boundary undergoes a transformation from a weight-3 stabilizer in Figure 3.1 (a)

to a corresponding weight-4 stabilizer in Figure (b) 3.1 by incorporating one intermediate data qubit. The values of the weight-4 Z-type stabilizer will be the same as the values of the weight-3 stabilizer, as the intermediate data qubits are initialized to the $|0\rangle$ state, which is the $+1$ eigenstate of the Z operator. Notably, initializing the intermediate data qubits to $|0\rangle$ ensures the construction of detectors for the Z stabilizer during the merging operation. This is because it guarantees that the values of the weight-4 Z-type stabilizer are determined within a noiseless circuit. In contrast, if the intermediate data qubits were not initialized to the eigenstates of the Z operator, the values would be random. Throughout this process, the entire system is treated as a unified surface code. The specific number of QEC rounds required for the merge procedure will be discussed in more detail in Chapter 4. Interestingly, the three new X stabilizers, represented in light red and spanning the boundary, generate random measurement outcomes. However, the product of these outcomes is the eigenvalue of the product of these X stabilizers, which corresponds to the operator $\hat{X}_L^1 \hat{X}_L^2$. This means that measuring these three new X stabilizers and computing their parity is equivalent to measuring $\hat{X}_L^1 \otimes \hat{X}_L^2$, thus performing non-destructive XX measurements.

Conversely, in a Z merge scenario, the intermediate qubits are initially prepared in the $|+\rangle$ state before measuring the new joint operator. Owing to symmetry, a Z merge is equivalent to the measurement of $\hat{Z}_L^1 \otimes \hat{Z}_L^2$.

SPLITTING OPERATION

Similarly, the process of splitting a code surface comes in two distinct forms: the X split and the Z split. The X split involves dividing a single logical qubit surface in two by conducting a series of measurements on intermediate data qubits in Z basis. This method effectively removes data qubits from the lattice. Alternatively, the Z split slices a single logical qubit surface in half by carrying out a sequence of measurements on intermediate data qubits in X basis, also leading to the elimination of data qubits from the lattice. After executing the splitting operation, quantum error correction cycles are performed on the two separate surfaces, focusing on the original stabilizers. In this study, we primarily focus on the X splitting operation.

The outcomes of measurements on intermediate data qubits will come out as 0 or 1 at random. Each of these measurements subsequently leaves weight-3 Z-type stabilizers on either side of the split. The parity of these weight-3 Z-type stabilizers, in tandem with the measurement outcome of the fourth data qubit, is expected to be 0. Specifically, we anticipate a 1 (or 0) for the left weight-3 Z-type stabilizers if the corresponding data qubit's measurement outcome is 1 (or 0).

We now analyze the transformation of the individual states of the two surfaces following the measurement of the row of intermediate data qubits. Following the splitting operation, the two logical states will be in an eigenstate of the joint logical operators $\hat{X}_L^1 \otimes \hat{X}_L^2$. Consequently, the joint state should inhabit the Hilbert space spanned by either $|++\rangle_L, |--\rangle_L$ or $|+-\rangle_L, |-+\rangle_L$, depending on the measurement result M_{XX} from the non-destructive XX measurement. Additionally, the logical $\hat{Z}_L$ operator is preserved by the X split, namely it possesses the capability to flip logical qubits 1 and 2 as well as the entirety of a single logical qubit. Therefore, we can conclude that the splitting oper-

ation transforms logical states in the following manner:

$$\alpha|0\rangle_L + \beta|1\rangle_L \longrightarrow \alpha|00\rangle_L + \beta|11\rangle_L, \tag{3.1}$$

$$a|+\rangle_L + b|-\rangle_L \longrightarrow a|++\rangle_L + b|--\rangle_L. \tag{3.2}$$

It is important to note that unlike the merging process, logical level information is not lost or gained during a split. The original state of a single qubit can be logically retrieved by performing a reverse merge operation following the split.

3

3.1.2. UNIVERSAL GATE WITH LATTICE SURGERY

In the context of defect-based surface codes, logical two-qubit gates, predominantly the $CNOT$ operation, are commonly implemented via braiding. However, this method has more qubit overhead. This is where the concept of lattice surgery comes into play. As indicated in Ref.[5], lattice surgery offers a significant reduction in storage overhead, making it a more efficient alternative for performing two-qubit gate operations in surface codes.

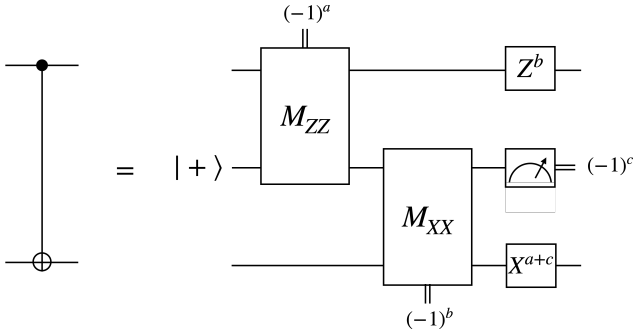

Figure 3.2: Implementation of the $CNOT$ operation employing lattice surgery: Utilization of non-destructive XX and ZZ measurements on the control and target qubits, with the inclusion of an ancillary qubit. The measurement box denotes a measurement in Z basis.

As depicted in Figure 3.2, a $CNOT$ gate can be realized on control and target qubits through the employment of an ancillary qubit, utilizing non-destructive XX and ZZ measurements. The method of lattice surgery facilitates the execution of this measurement-based $CNOT$ gate within a 2D layout, leveraging only local operations, as described in Ref.[28]. Furthermore, significant research has been conducted on the architectural considerations of the surface code, focusing particularly on the balance between space and time requirements, as detailed in works such as Ref.[29, 30]. These studies provide valuable insights into optimizing the surface code setup for quantum computations.

3.1.3. PLAIN SURGERY

Plain surgery provides an alternative approach for performing joint measurements of logical operators [31], offering a trade-off between time overhead and space (qubit) overhead compared to standard lattice surgery. It requires more space overhead but results in reduced time overhead. In plain surgery, the measurement results of the joint logical observable are determined directly by the measurement outcomes of a set of data

qubits. On the other hand, in standard lattice surgery, the measurement results of the joint logical observable depend on the measurement outcomes of a set of ancilla qubits. Consequently, as we will discuss in Chapter 4.2, standard lattice surgery necessitates multiple rounds of quantum error correction to ensure accurate measurements.

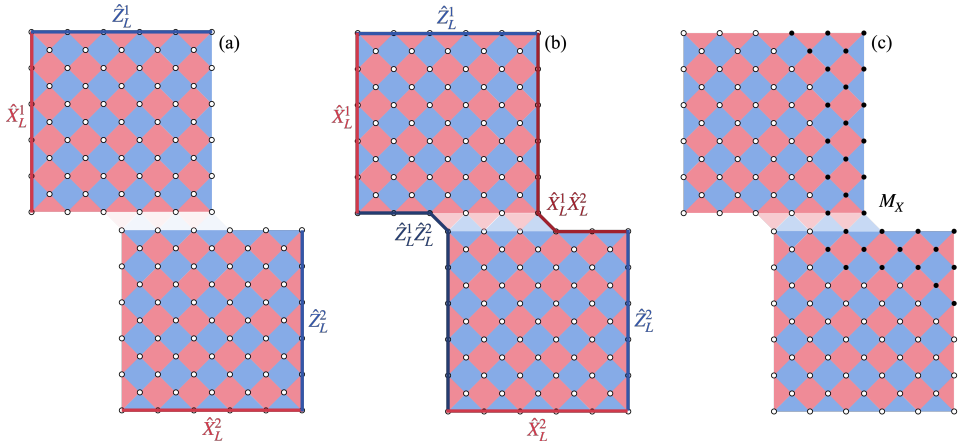

Figure 3.3: Procedures of lattice surgery for measurements of $Z_L^1 Z_L^2$ or $X_L^1 X_L^2$. (a) Two distance-6 square surface code patches are positioned with an overlap of approximately $\frac{2}{3}d$, the X-boundary of one facing the Z-boundary of the other. (b) The qubit layout after the merging operation. An example of the joint logical $Z_L^1 Z_L^2$ and $X_L^1 X_L^2$ operators are depicted. (c) Data qubits marked in black are measured fault-tolerantly in the X basis to evaluate $X_L^1 X_L^2$ instead of individual X_L^1 or X_L^2.

The plain surgery procedure encompasses two distinct steps: plain merge and measurement. The plain merge step begins with the arrangement of two patches, overlapping approximately by $\frac{2}{3}d$. One patch's X-boundary is juxtaposed with the Z-boundary of the other, as depicted in Figure 3.3 (a). This operation deforms the two separate sheets into a single code block, characterized by three X-boundaries and three Z-boundaries, effectively encapsulating two logical qubits. In the unified code, the logical operator $X_L^1 X_L^2$ is defined by a string originating from the top boundary of the upper patch and concluding at the right boundary of the lower patch, as illustrated in Figure 3.3 (b). Thus, by performing measurements on qubits in the X basis in a region distanced from the third X-boundary as shown in Figure 3.3 (c), the $X_L^1 X_L^2$ can be determined, which closely resembles the conventional logical measurement in a surface code block.

3.2. MAGIC STATE

While the surface code offers a relatively efficient fault-tolerant implementation of logical Clifford gates, it is worth noting that a quantum circuit composed solely of these gates is not universal and presents no quantum computational advantage over classical computing [17, 32, 33, 2]. In order to expand this limited set of computations to a universal set, the non-Clifford gate $T = \mathrm{diag}(1, e^{i\pi/4})$ is necessary. However, this gate cannot be implemented transversally for the surface code.

The solution for achieving universality with the surface code is to utilize ancillary

magic states. Logical T operators can be implemented by consuming the T magic state $|A\rangle = TH|0\rangle$, as illustrated in Figure 3.4 (a). Similarly, logical S operators can be realized using the $|Y\rangle = SH|0\rangle$ magic state, as shown in Figure 3.4 (b). It should be noted that the logical $CNOT$ gate in the circuits can be implemented using the lattice surgery technique. Magic states are typically prepared using state injection. However, existing state

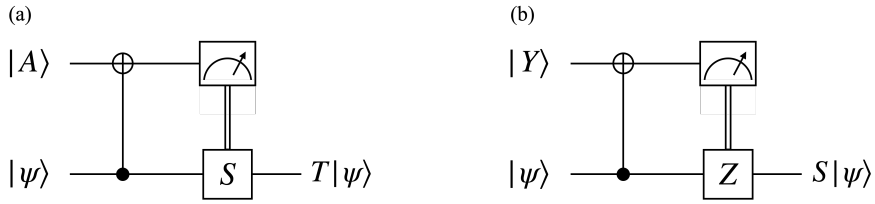

Figure 3.4: (a) Implementation of a T gate via magic state $|A\rangle = TH|0\rangle$. (b) Implementation of a S gate via magic state $|Y\rangle = SH|0\rangle$. The measurement box denotes a measurement in Z basis.

injection protocols, such as those mentioned in Ref. [4, 28], are not fault-tolerant, and hence the prepared states are often noisy. When a new magic state is initially injected, its fidelity sometimes can be so low that it can not be used in computation. As a solution, magic state distillation [34, 35] is used to transform numerous low-fidelity magic states into fewer, higher-fidelity states. This process continues until the magic states achieve acceptable fidelity. Notably, experiments demonstrating the preparation of logical magic states with fidelity beyond the distillation threshold have been conducted using superconducting devices [36].

Minimizing the significant overhead associated with preparing a large number of magic states necessitates the creation of magic states with high initial fidelity. Ref.[37] introduces a novel protocol using post-selection that demonstrates the infidelity in the encoded magic state can be less than half of the infidelity of a single $CNOT$ gate, assuming the error rate for single-qubit gates is considerably lower than for two-qubit gates. The protocol begins by initializing a single physical qubit to a magic state and encoding it into a surface code. If no error syndrome is found in this first phase, the protocol proceeds to the second phase. During the second phase, the code distance is increased to the target code distance to complete the encoding.

3.3. TRANSVERSAL LOGICAL S GATE

Implementing logical S gates on surface code can be accomplished through the use of magic states, as outlined in Section 3.2. However, this approach can be costly due to the need for both the preparation and distillation of magic states. An alternative approach for implementing logical S gates on surface code involves the use of non-local CZ gates between physical data qubits [27]. The key advantage of this method is its ability to be implemented transversally, which simplifies the process and reduces the associated overhead.

In this study, we focus on square surface codes. Figure 3.5 presents a code defined by a stabilizer group $\mathscr{S}$, which is generated by the following set of linearly independent

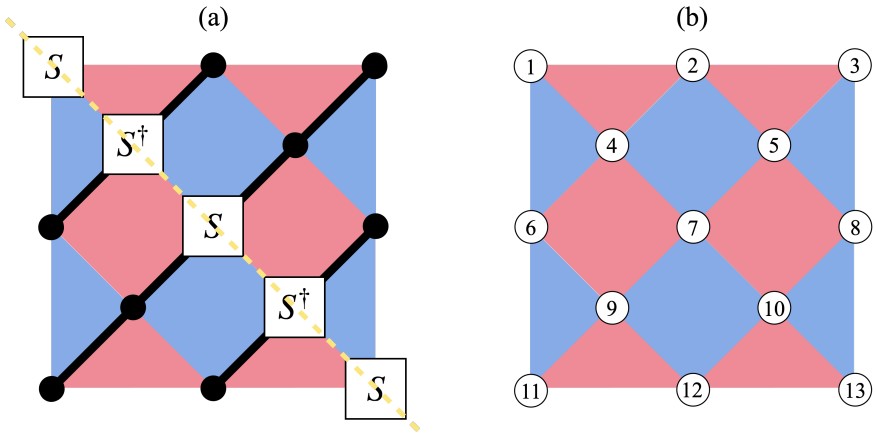

Figure 3.5: Transversal implementation of logical S gates for folded square surface codes. (a) To execute a logical S gate, data qubits situated on the yellow dashed diagonal line are interactively subjected to S or $S_\dagger$ operations. Furthermore, each pair of data qubits, symmetric in relation to the diagonal line, undergo a CZ gate. (b) Assign each data qubit on the lattice a unique number.

generators:

$$\begin{aligned}\mathscr{S} := \langle & X_1X_2X_4,\ X_2X_3X_5,\ X_4X_6X_7X_9,\ X_5X_7X_8X_{10},\ X_9X_{11}X_{12},\ X_{10}X_{12}X_{13},\\ & Z_1Z_4Z_6,\ Z_6Z_9Z_{11},\ Z_2Z_4Z_5Z_7,\ Z_7Z_9Z_{10}Z_{12},\ Z_3Z_5Z_8,\ Z_8Z_{10}Z_{13}\rangle.\end{aligned}$$

The logical S operation, denoted as $\bar{S}$, is expressed as:

$$\bar{S} = S_1S_4^\dagger S_7S_{10}^\dagger S_{13}CZ_{2,6}CZ_{3,11}CZ_{5,9}CZ_{8,12} \tag{3.3}$$

In this logical operation, S and $S^\dagger$ gates are applied to the data qubits situated on the yellow dashed diagonal line, while CZ gates are performed on pairs of data qubits. The pairs of qubits are symmetrically positioned with respect to the diagonal line, indicated in yellow in Figure 3.5(a).

To affirm that this construction of $\bar{S}$ is indeed a legitimate logical operation, we must demonstrate that this operation $\bar{S}$ preserves the stabilizer group $\mathscr{S}$. This requires checking whether the statement $\forall G \in \mathscr{S},\ \exists G' \in \mathscr{S}$ s.t. $U^\dagger GU = G'$ is valid or not. Through verification, we find that:

$$\begin{aligned}\bar{S}^\dagger(X_1X_2X_4)\bar{S} &= X_1X_2X_4 \otimes Z_1Z_4Z_6\\ \bar{S}^\dagger(X_2X_3X_5)\bar{S} &= X_2X_3X_5 \otimes Z_6Z_9Z_{11}\\ \bar{S}^\dagger(X_4X_6X_7X_9)\bar{S} &= X_4X_6X_7X_9 \otimes Z_2Z_4Z_5Z_7\\ \bar{S}^\dagger(X_5X_7X_8X_{10})\bar{S} &= X_5X_7X_8X_{10} \otimes Z_7Z_9Z_{10}Z_{12}\\ \bar{S}^\dagger(X_9X_{11}X_{12})\bar{S} &= X_9X_{11}X_{12} \otimes Z_3Z_5Z_8\\ \bar{S}^\dagger(X_{10}X_{12}X_{13})\bar{S} &= X_{10}X_{12}X_{13} \otimes Z_8Z_{10}Z_{13},\end{aligned} \tag{3.4}$$

and

$$\begin{aligned}
\bar{S}^\dagger(Z_1Z_4Z_6)\bar{S} &= Z_1Z_4Z_6\\
\bar{S}^\dagger(Z_6Z_9Z_{11})\bar{S} &= Z_6Z_9Z_{11}\\
\bar{S}^\dagger(Z_2Z_4Z_5Z_7)\bar{S} &= Z_2Z_4Z_5Z_7\\
\bar{S}^\dagger(Z_7Z_9Z_{10}Z_{12})\bar{S} &= Z_7Z_9Z_{10}Z_{12}\\
\bar{S}^\dagger(Z_3Z_5Z_8)\bar{S} &= Z_3Z_5Z_8\\
\bar{S}^\dagger(Z_8Z_{10}Z_{13})\bar{S} &= Z_8Z_{10}Z_{13}
\end{aligned} \tag{3.5}$$

Hence, we can confidently conclude that $\bar{S}$ is a valid logical operation.

To ensure that $\bar{S}$ effectively performs the logical equivalent of the S gate operation, we need to confirm that the conditions $\bar{S}^\dagger\bar{X}\bar{S} = -\bar{Y}$, $\bar{S}^\dagger\bar{Y}\bar{S} = \bar{X}$, $\bar{S}^\dagger\bar{Z}\bar{S} = \bar{Z}$ are all satisfied. Since $SXS = iX$, $SYS = iY$, $SZS = I$ for the standard quantum gate operations, we would expect that $\bar{S}\bar{X}\bar{S} = i\bar{X}$, $\bar{S}\bar{Y}\bar{S} = i\bar{Y}$, $\bar{S}\bar{Z}\bar{S} = \bar{I}$ for their logical counterparts. Indeed, we find that these relations hold:

$$\bar{S}\bar{X}\bar{S} = \bar{S}(X_2X_7X_{12})\bar{S} = i\bar{X}\otimes(Z_1Z_4Z_6)\otimes(Z_8Z_{10}Z_{13}), \tag{3.6}$$

$$\bar{S}\bar{Z}\bar{S} = \bar{S}(Z_1Z_2Z_3)\bar{S} = (Z_2Z_4Z_5Z_7)\otimes(Z_3Z_5Z_8)\otimes(Z_8Z_{10}Z_{13}) \tag{3.7}$$

Thus, it's validated that $\bar{S}$ successfully performs the S gate operation at the logical level.

We note that the existence of mirror-dual symmetry with respect to the diagonal line in square surface codes enables the transversal construction of the logical S gate. That is, by folding the surface code along the diagonal line, we can pair an X-type stabilizer with a corresponding Z-type stabilizer. However, the logical S gate transforms the generators, which necessitates a careful design of detectors, which we will discuss in more depth in Chapter 4.1.2.

4

OPTIMIZING ROUNDS OF QUANTUM ERROR CORRECTION CYCLES

4.1. CONSTRUCTING DETECTORS UNDER CODE AND STABILIZER DEFORMATION

In Chapter 2, we introduced the concept of constructing detectors when there are no stabilizers deformations or code deformations in play. However, it is crucial to carefully identify syndrome detectors under these deformations.

4.1.1. CODE DEFORMATION

The technique of code deformation is a tool often employed in the fault-tolerant execution of logical gates. Code deformation involves transforming one quantum error-correcting code into another, and this process can mainly be undertaken in two ways:

1. By making a series of changes on the set of stabilizer generators that need to be measured

2. By carrying out unitary transformations that change the code space

Examples of the first method include expanding or shrinking code patches, and merging or splitting them. These transformations involve adding or deleting stabilizer generators to alter the stabilizer group. The second method involves transforming the existing stabilizer group into a new one via unitary operations, that is, $\mathcal{S}_{\text{new}} = \bar{U}^{\dagger}\mathcal{S}_{\text{old}}\bar{U}$. An exemplary demonstration of this transformation process occurs during the implementation of a logical Hadamard gate. In this scenario, physical H gates are applied to each individual data qubit, resulting in a transformation of the stabilizer group. Specifically, the operation $\bar{H}$ transforms the initial stabilizer group, denoted $\mathcal{S}_{\text{old}}$, into a new configuration, expressed as $\mathcal{S}_{\text{new}} = \bar{H}^{\dagger}\mathcal{S}_{\text{old}}\bar{H}$. In this instance, the code lattice specified by $\mathcal{S}$new is the dual of the lattice defined by $\mathcal{S}$old, that is X and Z checks have interchanged with

each other. This duality emerges due to the inherent properties of Hadamard transformations, i.e., $H^\dagger X H = Z$ and $H^\dagger Z H = X$. Thus, we observe a systematic swap in the nature of stabilizers: each X-type stabilizer in the old lattice transitions into a Z-type stabilizer in the new lattice, and reciprocally, every Z-type stabilizer in the old lattice transforms into an X-type stabilizer in the new lattice. This leaves the planar surface at a different orientation from the original.

The first way of code deformations entails alterations to the original set of stabilizer generators, specifically through their addition or removal. Figure 4.1 provides an illustrative example of this process, showcasing the addition or deletion of stabilizer generators along the X boundary.

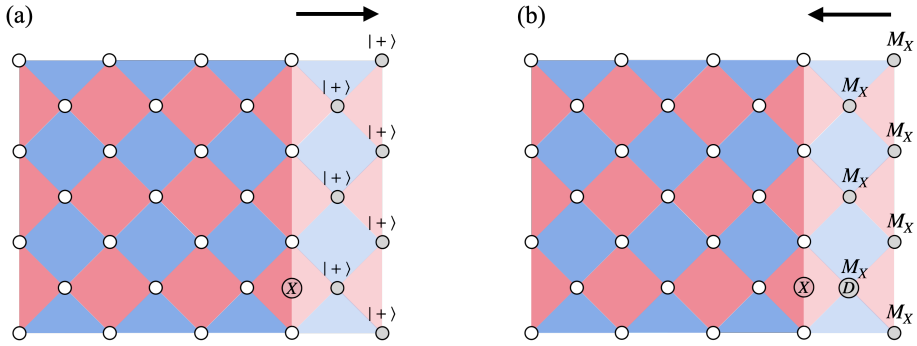

Figure 4.1: Code deformation by adding or deleting stabilizer generators. (a) Expansion of the original distance-4 surface code is achieved through the introduction of new stabilizer generators, which are lightly drawn in the figure. Data qubits, depicted in gray, are initialized in the $|+\rangle$ state, after which quantum error correction rounds are performed for all stabilizers. (b) The process of stabilizer deletion is achieved by measuring the gray-highlighted data qubits in the X basis.

When new stabilizer generators are added, as illustrated in Figure 4.1 (a), the data qubits introduced are in the $|+\rangle$ state. As such, the measurement outcome of the X stabilizers on the boundary remains unchanged in the absence of noise, even though each of them transitions from a weight-3 stabilizer to a weight-4 stabilizer during the addition process. Consequently, the syndrome detector for all X-type stabilizers S_X after the expansion is defined as:

$$\sigma_{S_X}^{\text{after}} = m_{S_X}^{\text{before}} \oplus m_{S_X}^{\text{after}} \tag{4.1}$$

Here, $m_{S_X}^{\text{before}}$ and $m_{S_X}^{\text{after}}$ denote the measurement results of the X-type stabilizer of interest in the QEC cycle prior to and following the expansion, respectively. Specifically, on the boundary being expanded, $m_{S_X}^{\text{before}}$ corresponds to the measurement result of the weight-3 X-type stabilizer generators before the expansion, while $m_{S_X}^{\text{after}}$ corresponds to the measurement outcome of the weight-4 X-type stabilizer generators after the expansion. The measurement results of the newly introduced Z-type stabilizers are random, meaning that their values cannot be determined beforehand. As a result, the construction of syndrome detectors for these new Z-type stabilizers can only be constructed after the first round of QEC following the expansion. For the original Z-type stabilizers S_Z,

however, the syndrome detector after the expansion can be constructed as:

$$\sigma_{S_Z}^{\text{after}} = m_{S_Z}^{\text{before}} \oplus m_{S_Z}^{\text{after}} \tag{4.2}$$

On the other hand, constructing the detectors when deleting stabilizer generators is not as straightforward as when adding them. The process of contracting or extracting a surface code, as depicted in Figure 4.1 (b), is carried out by measuring the gray-highlighted data qubits in the X basis. Each of these measurements subsequently leaves behind weight-3 X-type stabilizers on the X boundary. It is crucial to note that the outcomes of these data qubit measurements are random. Consequently, in an ideal scenario, the remaining data qubits are randomly in the $+1$ or -1 eigenstate of each weight-3 X-type stabilizer on the contracting boundary, depending on the measurement result of the data qubit which the weight-3 X-type stabilizer touches (denoted as data qubit D and represented in gray in the figure). The detector for the weight-3 X-type stabilizer post-contraction is thus defined as:

$$\sigma_{S_X}^{\text{after}} = m_{S_X}^{\text{before}} \oplus m_{S_X}^{\text{after}} \oplus m_D \tag{4.3}$$

where $m_{S_X}^{\text{before}}$ and $m_{S_X}^{\text{after}}$ denote the measurement results of the X-type stabilizer on the contracting boundary in the QEC cycle prior to and following the contraction, respectively. m_D represents the measurement result of the data qubit D. The construction of syndrome detectors for the X-type stabilizer generators that are not on the boundary to be constructed, as well as all Z-type stabilizer generators after the contraction, follows a straightforward process. These detectors can be constructed using the same principles described in Eq. (4.1) for X-type stabilizers and Eq. (4.2) for Z-type stabilizers.

4.1.2. STABILIZER DEFORMATION

Logical operations $\bar{U} \in \mathcal{N}(\mathcal{S}) =: \{\bar{U} \in \mathbf{U}(2^n) \mid \forall S \in \mathcal{S},\ \exists S' \in \mathcal{S} \text{ s.t.} S' = \bar{U}^\dagger S \bar{U}\}$ can sometimes transform one stabilizer generator to another one in Heisenberg picture, but always keep the stabilizer group unchanged. In this case, the construction of syndrome detectors has to be carefully designed, too. For a surface code defined by $\mathcal{S} = \langle G_1, G_2, ..., G_K \rangle$ with K the number of stabilizer generators of the code, let the measurement results of G in the QEC cycle before and after the application of $\bar{U}$ be:

$$m_{G,\text{before}} = \langle\psi| G |\psi\rangle \rightarrow \langle\psi| \bar{U}^\dagger G \bar{U} |\psi\rangle = m_{G,\text{after}} \tag{4.4}$$

Note that here we define the measurement outcome $\langle\psi| G |\psi\rangle = 0$ if $G|\psi\rangle = |\psi\rangle$ and to 1 if $G|\psi\rangle = -|\psi\rangle$. Since $\bar{U}^\dagger G \bar{U} \in \mathcal{S}$, we can always write it as a product of generators:

$$\bar{U}^\dagger G \bar{U} = \bigotimes_{i=1}^{K} G_i^{p_i} \tag{4.5}$$

Here, p_i are integers, which are either 0 or 1. The power p_i is 0 if the generator G_i is not part of the product, and it is 1 if the generator G_i is part of the product. Therefore, $m_{G,\text{after}}$ can be rewritten as:

$$m_{G,\text{after}} = \bigotimes_{i=1}^{K} \langle\psi| G_i^{p_i} |\psi\rangle = \bigoplus_{i=1}^{K} p_i m_{G_i,\text{before}} \tag{4.6}$$

Therefore, once a logical operation, denoted as $\bar{U}$, has been performed, we can track the syndrome (error information) on stabilizer generator G during the subsequent QEC cycle using the following formula:

$$\sigma_G^{\text{after}} = \left(\bigoplus_{i=1}^{K} p_i m_{G_i,\text{before}} \right) \oplus m_G^{\text{after}} \tag{4.7}$$

This equation defines the syndrome detector to accurately identify and track error syndromes across stabilizer generator transformations.

As an example, we consider the construction of syndrome detectors under stabilizer deformation due to the implementation of the transversal $\bar{S}$ operation. As discussed in Chapter 3, the application of a transversal logical S gate transforms each X-type stabilizer into the product of itself and the Z-type stabilizer that is mirror-symmetric to it with respect to the diagonal line, which is depicted in Eq. (3.4) and . We represent this transformation as:

$$\bar{S}^\dagger S_X^v \bar{S} = S_X^v \otimes S_Z^{\text{sym}(v)} \tag{4.8}$$

In this expression, S_X^v refers to the X-type stabilizer at site v on the lattice, while $\text{sym}(v)$ indicates the site mirror-symmetrical to v with respect to the diagonal line, as shown in Figure 3.5. Consequently, $S_Z^{\text{sym}(v)}$ signifies the Z-type stabilizer at the site $\text{sym}(v)$ on the lattice.

Given this transformation, the syndrome detector on the stabilizer S_X^v in the QEC cycle following the execution of the logical operation $\bar{S}$ becomes:

$$\sigma_{S_X^v}^{\text{after}} = m_{S_X^v,\text{before}} \oplus m_{S_Z^{\text{sym}(v)},\text{before}} \oplus m_{S_X^v,\text{after}} \tag{4.9}$$

Here, $m_{S_X^v,\text{before}}$ and $m_{S_X^v,\text{after}}$ denote the measurement outcomes of the X-type stabilizer at site v, before and after the implementation of the $\bar{S}$ operation, respectively. Similarly, $m_{S_Z^{\text{sym}(v)},\text{before}}$ stands for the measurement result of the Z-type stabilizer at the site $\text{sym}(v)$.

4.2. DECODING LATTICE SURGERY

As we introduced in Chapter 3.1, lattice surgery serves as a critical technique in surface code quantum computation. It is used to execute logical two-qubit gates and the non-Clifford logical T gate with a logical magic state. Numerous resources [28, 30, 29] propose that d rounds of QEC must be undertaken to achieve sufficiently low fidelity in the logical joint measurement during the merging operation of lattice surgery. While this supposition aligns intuitively with our understanding, it demands a more rigorous theoretical backing. It is equally important to consider additional parameters, such as the qubit error probability and the qubit measurement probability. In this section, we aim to provide both theoretical and simulation-based evidence to elucidate this issue. We endeavor to create a comprehensive and precise understanding of the relationship between rounds of QEC and the resulting fidelity in lattice surgery operations.

4.2.1. THEORY

In this section, we provide a theoretical determination of the requisite number of QEC cycles needed for lattice surgery when the circuit is under phenomenological noise model (see Chapter 1.1.1). This is to ensure a sufficiently low probability of a fault occurring during a logical joint measurement.

IDENTIFY HARMFUL LOOPS FOR LATTICE SURGERY

In quantum memory experiments, the combination of errors with the correction can result in nontrivial paths across the whole spacetime lattice or trivial loops in the spacetime lattice. Similarly, in the context of decoding the outcome of logical joint measurements using lattice surgery, two types of overall operations (errors and corrections) emerge as potentially harmful:

1. paths originating from the newly introduced stabilizers at the past-time boundary and ending at the future-time boundary,

2. paths starting from the newly introduced stabilizers at the past-time boundary and terminating at the spatial boundary.

Examples for these paths when measuring logical XX of two distance-5 square surface codes are illustrated in Figure 4.2. The X boundaries of two patches of code are merged by introducing new X stabilizer generators in the middle. The purple solid lines denote error edges within the spacetime volume (shown in yellow), and the dashed purple lines indicate error edges that across either the temporal or spatial boundary of the spacetime volume.

It is important to recognize that all such harmful overall operations begin from the newly introduced X-type stabilizer generators at the past-time boundary. This is because the result of the joint measurement is determined by the parity of the outcomes of the newly-introduced stabilizer generators during the first round of QEC after the merging operation, as introduced in Chapter 3.1.1. As a consequence, even if there are nontrivial loops of overall operation present, as long as they do not interact with the newly added ancilla qubits, these overall operations will not affect the value of the logical joint measurement outcome.

Figure 4.2 illustrates two fundamentally distinct categories of homologically nontrivial overall operation (error and correction) chains within the three-dimensional spacetime lattice. Each of these chains originates from the past-time boundary, and ends either at a spatial boundary or at the future-time boundary. Of these chains, one is purely space-like, another purely time-like, and the third one is comprised of both space-like and time-like error edges.

BOUND ON NONTRIVIAL LOOP PROBABILITIES

Having identified the homologically nontrivial chains formed by errors and corrections in lattice surgery, we may inquire about their probability of occurrence, which is the probability of failure of the logical joint measurement by lattice surgery. To approach this question, we first consider the probability $\mathrm{Prob}(H,V)$ that a given path, with (H,V) horizontal (time-like) and vertical (space-like) edges, is contained within the overall operation (errors and corrections) $\mathbf{E_e}+\mathbf{E_c}$. As we discussed in Chapter 2.1.2, horizontal and

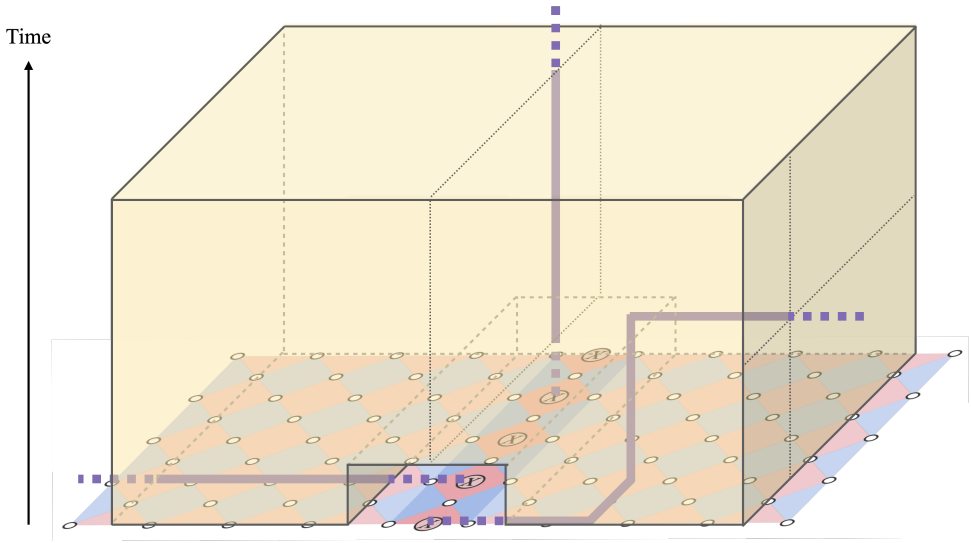

Figure 4.2: Homologically nontrivial chains on the lattice formed by errors and corrections for lattice surgery decoding in the spacetime volume. Two distance-5 square surface codes with X boundaries are merged through the introduction of new data and X-type ancilla qubits in between them. The spacetime volume is represented by the yellow cuboid. Error edges within the spacetime volume are depicted as solid purple lines. Error edges that cross the boundaries of the spacetime volume connect one defect on the boundary with a virtual defect (recall Figure 2.4). These error edges are represented by dashed purple lines.

vertical edges correspond to single errors on qubits and measurement errors, respectively. For simplicity, diagonal (space-time-like) edges are not taken into consideration since we are considering phenomenological noise model.

Consider a specific path (which could be homologically trivial) within the spacetime volume, composed of H horizontal edges and V vertical edges. Let (H_e, V_e) denote the number of those edges that are contained by the errors $\mathbf{E_e}$, and (H_c, V_c) represent the number of those links that are contained by the correction $\mathbf{E_c}$. That is to say:

$$H = H_e + H_c, \quad V = V_e + V_c \tag{4.10}$$

There are 2^{H+V} ways to distribute error edges (H_e, V_e) along the specified path, where each link could either contain an error or not. Once the locations for these errors have been determined, the probability $p_0(H,V)$ of errors appearing at those particular sites is given by:

$$\begin{aligned} p_0(H,V) =& p_{\mathrm{q}}^{H_e}(1-p_{\mathrm{q}})^{H_c} p_{\mathrm{meas}}^{V_e}(1-p_{\mathrm{meas}})^{V_c} \\ =& (1-p_{\mathrm{q}})^{H}(1-p_{\mathrm{meas}})^{V}\left(\frac{p_{\mathrm{q}}}{1-p_{\mathrm{q}}}\right)^{H_e}\left(\frac{p_{\mathrm{meas}}}{1-p_{\mathrm{meas}}}\right)^{V_e} \end{aligned} \tag{4.11}$$

The minimum-weight decoder is designed to determine $\mathbf{E_c}$ in a way such that

$$\left(\frac{p_{\mathrm{q}}}{1-p_{\mathrm{q}}}\right)^{H_e}\left(\frac{p_{\mathrm{meas}}}{1-p_{\mathrm{meas}}}\right)^{V_e} \leq \left(\frac{p_{\mathrm{q}}}{1-p_{\mathrm{q}}}\right)^{H_c}\left(\frac{p_{\mathrm{meas}}}{1-p_{\mathrm{meas}}}\right)^{V_c}. \tag{4.12}$$

Hence, combining Eq. (4.10), (4.11), and (4.12), we conclude that $p_0(H,V) \leq \left(\tilde{p}_{\mathrm{q}}^{H}\tilde{p}_{\mathrm{meas}}^{V}\right)^{1/2}$ if the path is contained in $\mathbf{E_e}+\mathbf{E_c}$ where

$$\tilde{p}_{\mathrm{q}} = p_{\mathrm{q}}(1-p_{\mathrm{q}}), \quad \tilde{p}_{\mathrm{meas}} = p_{\mathrm{meas}}(1-p_{\mathrm{meas}}) \tag{4.13}$$

As a result, the probability $\mathrm{Prob}(H,V)$ that a particular path with (H,V) horizontal and vertical edges is contained in $\mathbf{E_e}+\mathbf{E_c}$ can be bounded as follows:

$$\mathrm{Prob}(H,V) \leq 2^{H+V}\left(\tilde{p}_{\mathrm{q}}^{H}\tilde{p}_{\mathrm{meas}}^{V}\right)^{1/2} \tag{4.14}$$

We can set a boundary on the probability that $\mathbf{E_e}+\mathbf{E_c}$ encompasses any connected path with (H,V) edges by enumerating such paths. Taking a cue from Ref. [11], we treat the path as a walk on the three-dimensional lattice. Our primary interest pertains to self-avoiding walks (SAWs) — those that visit any given node in the spacetime volume no more than once (or in the context of a loop, only return to the start and end point). This is because for any open error walk connecting two nodes, a SAW can always be derived by eliminating certain closed loops of links from the original walk.

To assess the number of QEC cycles necessary for lattice surgery decoding, we need to consider Self-Avoiding Walks (SAWs) lying between the past-time boundary at $t=0$ and the future-time boundary at $t=T$. As discussed in Section 4.2.1, such an SAW could commence at any one of the d newly-introduced X stabilizers at $t=0$. If $n_{\mathrm{SAW}}(H,V)$ represents the number of SAWs with (H,V) edges and a specified starting site, then the probability $\mathrm{Prob}_{\mathrm{SAW}}(H,V)$ that $\mathbf{E_e}+\mathbf{E_c}$ encompasses at least one SAW with (H,V) edges can be expressed as:

$$\mathrm{Prob}_{\mathrm{SAW}}(H,V) \leq d\cdot n_{\mathrm{SAW}}(H,V)\cdot 2^{H+V}\left(\tilde{p}_{\mathrm{q}}^{H}\tilde{p}_{\mathrm{meas}}^{V}\right)^{1/2} \tag{4.15}$$

This upper bound, which is obtained from Ref. [11], forms the basis for the ensuing results.

BOUND ON FAILURE PROBABILITY OF LATTICE SURGERY

The encoded information relating to the joint measurement outcome of two logical qubits will be lost if $\mathbf{E_e}+\mathbf{E_c}$ incorporates homologically nontrivial paths. For lattice surgery, the first type of homologically nontrivial path must contain at a minimum, T vertical edges, while the second type of homologically nontrivial path needs to encompass at least d horizontal edges. Therefore, we can express the failure probability as:

$$\mathrm{Prob}_{\mathrm{fail}} \leq \sum_{V\geq 0}\sum_{H\geq d}\mathrm{Prob}_{\mathrm{SAW}}(H,V) + \sum_{V\geq T}\sum_{H\geq 0}\mathrm{Prob}_{\mathrm{SAW}}(H,V) \tag{4.16}$$

The number of self-avoiding walks (SAWs) in a three-dimensional spacetime volume, given a specific number of horizontal and vertical edges, can be upper bounded as follows [38]:

$$n_{\mathrm{SAW}}^{(3)}(\ell) \leq P_3(\ell)\left(\mu_3\right)^{\ell} \tag{4.17}$$

where $\ell = H + V$, $P_3(\ell)$ is a polynomial, and μ_3 is a constant. Therefore, by combining equations (4.15), (4.16) and (4.17), we can derive an upper bound for $\text{Prob}_{\text{fail}}$

$$\begin{aligned}\text{Prob}_{\text{fail}} \leq & d\left(\sum_{V\geq 0}\sum_{H\geq d} P_3(H+V)\cdot\left(4\mu_3^2\tilde{p}_{\text{q}}\right)^{H/2}\left(4\mu_3^2\tilde{p}_{\text{meas}}\right)^{V/2}\right) \\ & + d\left(\sum_{V\geq T}\sum_{H\geq 0} P_3(H+V)\cdot\left(4\mu_3^2\tilde{p}_{\text{q}}\right)^{H/2}\left(4\mu_3^2\tilde{p}_{\text{meas}}\right)^{V/2}\right).\end{aligned} \tag{4.18}$$

Assuming that

$$\tilde{p}_{\text{q}} < \left(4\mu_3^2\right)^{-1}, \quad \tilde{p}_{\text{meas}} < \left(4\mu_3^2\right)^{-1} \tag{4.19}$$

we obtain

$$\left(4\mu_3^2\tilde{p}_{\text{q}}\right)^{H/2}\cdot\left(4\mu_3^2\tilde{p}_{\text{meas}}\right)^{V/2} \leq \left(4\mu_3^2\tilde{p}_{\text{q}}\right)^{d/2} \tag{4.20}$$

for every term in the first sum, and

$$\left(4\mu_3^2\bar{p}\right)^{H/2}\cdot\left(4\mu_3^2\tilde{p}_{\text{meas}}\right)^{V/2} \leq \left(4\mu_3^2\tilde{p}_{\text{meas}}\right)^{T/2} \tag{4.21}$$

for every term in the second sum. With a total of $4d^2T$ horizontal links and $2d^2T$ vertical links, we can deduce that:

$$\text{Prob}_{\text{fail}} \leq Q_1(d,T)\cdot(4\mu_3^2\tilde{p}_{\text{q}})^{d/2} + Q_2(d,T)\cdot(4\mu_3^2\tilde{p}_{\text{meas}})^{T/2} \tag{4.22}$$

where $Q_{1,2}(d,T)$ are polynomials of d and T.

Since this upper bound is analyzed under a phenomenological noise model, it is important to note that the constant μ_3 may differ in real-world experiments due to variations in nontrivial path counting. Thus, in a more general context, we can state the following:

$$\text{Prob}_{\text{fail}} \leq Q_1(d,T)\cdot(C_1\tilde{p}_{\text{q}})^{d/2} + Q_2(d,T)\cdot(C_2\tilde{p}_{\text{meas}})^{T/2} \tag{4.23}$$

where C_1 and C_2 are constants.

Increasing T will initially cause the logical error to decrease exponentially relative to T, but ultimately, the logical error rate will reach a plateau that scales exponentially with the code distance d (we will discuss this again in Figure 4.4). To achieve optimal performance given the code distance d, the value of the second term should be much smaller than that of the first term. This can be ensured when:

$$T \gg d\frac{\log\left(C_1\tilde{p}_{\text{q}}\right)^{-1}}{\log\left(C_2\tilde{p}_{\text{meas}}\right)^{-1}} \equiv d\cdot C(\tilde{p}_{\text{q}},\tilde{p}_{\text{meas}}) \tag{4.24}$$

4.2.2. SIMULATION

To verify our theoretical findings, we perform a simulation of a simplified lattice surgery experiment, as depicted in Figure 4.3. We initialize two logical qubits, encoded by the square surface code, to the state $|\bar{+}\rangle$. Subsequently, we perform lattice surgery to merge their X boundaries together, enabling the measurement of the logical observable $\bar{X}\bar{X}$. We note that splitting operation is not performed in this experiment as it does not affect the joint measurement outcome. After the merging operation, we conducted T rounds of

QEC on the combined patch of code. After these rounds, defect information is collected and the MWPM decoder [14] introduced in Chapter 2.2.1 is utilized for decoding. Python package *Stim* [1] is used to generate the circuits. The code for the simulation can be found in Chapter 6 (Appendix).

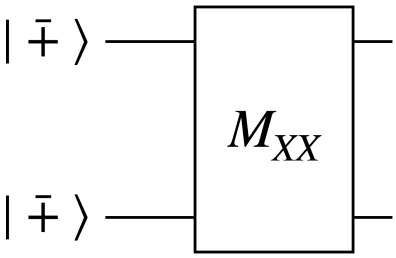

Figure 4.3: The setup for the lattice surgery experiments. Two logical qubits, encoded by the square surface code, are each initialized to the state $|\bar{+}\rangle$. Using lattice surgery techniques, a joint logical measurement, denoted as M_{XX}, is carried out. Following the merging operation, QEC is performed for a total of T rounds.

In all the simulations conducted for this work, we use a circuit-level noise model, as described in Chapter 1. This model simulates physical gate errors by incorporating a one-qubit depolarization channel following each physical single-qubit gate operation, and a two-qubit depolarization channel after each physical two-qubit gate execution. Moreover, a depolarization channel is also added for each idling qubit to account for decoherence, using the method introduced in Chapter 1.1.1. The coherence times for the qubits are established at $T_1 = T_2 = 60\mu s$. The other parameters used in this model are as follows: the error rate for the two-qubit gate is set to 0.5%, while the error rate for the single-qubit gate is kept at one-tenth of this value, that is, 0.05%. The measurement error rate varies and is set differently for individual experiments.

Figure 4.4 displays the results of an experiment examining the logical error rate of M_{XX} measurement outcomes in relation to the number of QEC cycles (T) following the merging operation for lattice surgery. The experiment involved the merging of two logical qubits encoded by square surface codes with varying code distances (d), specifically $d = 3,5,7,9,11,13,15$. The classical read-out error rate was set to $p_{\text{meas}} = 1.0\%$. The results demonstrate a distinct trend: as the number of QEC cycles T increases, the logical M_{XX} error rate initially decreases exponentially before eventually reaching a plateau. Moreover, the value of the plateau decreases exponentially with respect to the code distance d. We note that the logical error of M_{XX} should not increase with T since our primary concern is not the preservation of the qubit state over time, but rather the error probability associated with the joint measurement.

This observation aligns with our theoretical findings. The second term in the upper bound expression for $\text{Prob}_{\text{fail}}$ (see Eq. (4.23)), denoted as $Q_2(d,T)\cdot(C_2\tilde{p}_{\text{meas}})^{T/2}$, decreases exponentially as T increases, explaining the initial exponential decrease in the error rate observed in Figure 4.4. However, as T continues to increase, $Q_2(d,T)\cdot(C_2\tilde{p}_{\text{meas}})^{T/2}$ becomes sufficiently small and negligible compared to the first term $Q_1(d,T)\cdot(C_1\tilde{p}_{\text{q}})^{d/2}$, which is dominated by the code distance d. Consequently, every curve eventually reaches a plateau whose value is exponentially dependent on the code distance d, as the contribution from the second term becomes negligible.

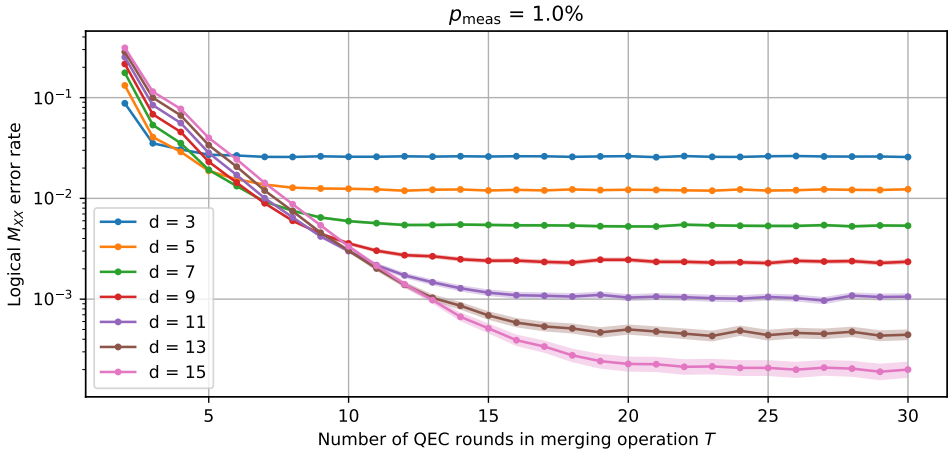

Figure 4.4: The logical joint measurement M_{XX} as a function of the number of QEC cycles, denoted by T, carried out during the merging operation. The simulations are executed for square surface codes of distance $d = 3,5,7,9,11,13,15$, respectively. Under a circuit-level noise model, the error rate is fixed at 0.5% for the two-qubit gate, while the single-qubit gate has an error rate of one tenth of this. The measurement error rate, p_{meas}, is set at 1%, and the qubit coherence times are $T_1 = T_2 = 60\mu s$.

We also observe that the starting point of each plateau, where the logical error rate levels off and remains relatively constant, exhibits a proportional relationship with the code distance d. This finding aligns with the relationship described in Eq. (4.24). Furthermore, it is important to note that performing only d rounds of QEC is insufficient to reach the plateau of the logical error rate, as one can see in Figure 4.4 .

As highlighted in Eq. (4.24), the number of QEC cycles necessary to determine the results of M_{XX} with optimal confidence depends not only on the code distance d, but also on the qubit error probability p_{q} and measurement error probability p_{meas}.

To illustrate this more explicitly, we conduct an additional experiment examining the logical error rate of M_{XX} and T for two logical qubits encoded by a distance-7 square surface code. We varied the measurement error probability p_{meas}, setting it to 0.5%, 1.0%, 1.5%, 2.0%, 2.5%, and 3.0% respectively, as illustrated in Figure 4.5.

In Figure 4.5, the curves representing different values of p_{meas} clearly demonstrate that a higher measurement error probability p_{meas} requires a larger number of QEC cycles to reach the plateau of the logical error rate for M_{XX}. This observation aligns with the relationship described in Eq. (4.24), that is, the larger the measurement error rate, the more rounds of QEC required.

In summary, the numerical results presented in this study offer empirical evidence that supports our theoretical understanding of the necessary number of QEC cycles for effective decoding. One crucial aspect to note is that the joint logical observable of interest can be expressed as a product of stabilizer measurement results. To confidently determine the sign of these stabilizer measurement results, it is imperative to perform decoding over a larger spacetime volume, which corresponds to longer time intervals. This minimizes the probability of nontrivial overall operation paths occurring, which in

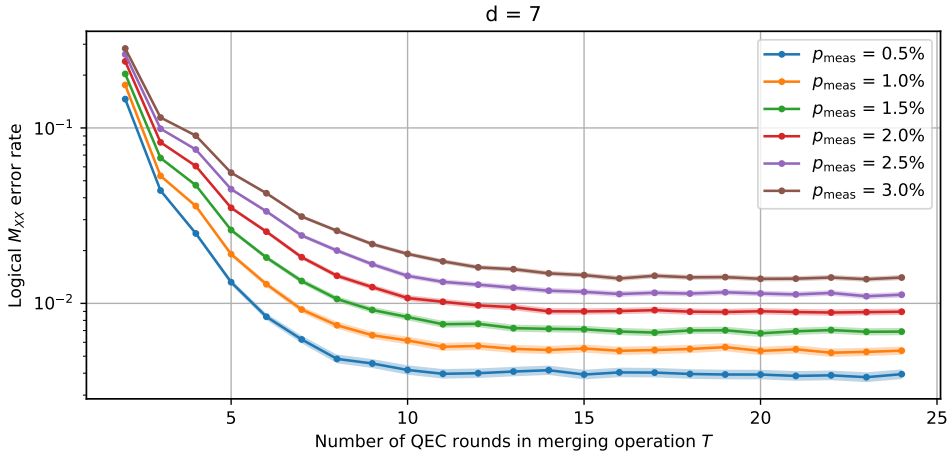

Figure 4.5: The logical joint measurement M_{XX} in relation to the number of QEC cycles, represented as T, implemented during the merging operation. The simulations are carried out for square surface codes of distance $d = 7$ under different measurement error rates p_{meas}—specifically, 0.5%, 1.0%, 1.5%, 2.0%, 2.5%, 3.0%. In this circuit-level noise model, the two-qubit gate has an error rate of 0.5%, while the single-qubit gate has a reduced error rate of just one tenth of this. Additionally, the qubit coherence times are fixed at $T_1 = T_2 = 60\mu s$.

turn improves the accuracy of the decoding process.

4.3. DECODING LOGICAL QUBIT MOVEMENT

Moving logical qubits on the lattice is a crucial technique in surface code quantum computation. It serves various purposes, such as adjusting the orientation of the code and enabling the entanglement of remote logical qubits.

The essence of moving an encoded logical qubit involves relocating its logical observables through space. Figure 4.6 illustrates an example of moving a distance-4 square surface code along the direction of its logical X chain. Specifically, the logical observables $\hat{Z}_L^{\text{old}}$ and $\hat{X}_L^{\text{old}}$ are transformed into $\hat{Z}_L^{\text{new}}$ and $\hat{X}_L^{\text{new}}$, respectively.

In Figure 4.6 (a), we begin with a distance-4 square surface code on the left, associated with $\hat{Z}_L^{\text{old}}$ and $\hat{X}_L^{\text{old}}$. The data qubits on the right-hand side of the code are initialized to the $|+\rangle$ state. In Figure 4.6 (b), quantum error correction cycles are performed on all the stabilizers. The measurement outcomes of the newly introduced Z-type stabilizers $\{S_{Z,i}^{\text{new}}\}_{i=1}^{K}$ (shown in the legend) are random. It is crucial to accurately infer their measurement outcomes since they determine the relationship between $\hat{Z}_L^{\text{old}}$ and $\hat{Z}_L^{\text{new}}$ as follows:

$$\hat{Z}_L^{\text{new}} = \hat{Z}_L^{\text{old}} \left(\bigotimes_{i=1}^{K} S_{Z,i}^{\text{new}} \right) \tag{4.25}$$

Therefore, if any odd number of these stabilizer measurements are incorrect, it will lead to an incorrect sign for the moved logical observable. For example, if the product of the measurement outcomes of all the newly introduced stabilizers is $+1$, we know that $\hat{Z}_L^{\text{new}} = \hat{Z}_L^{\text{old}}$. Conversely, if the product is -1, we know that $\hat{Z}_L^{\text{new}} = -\hat{Z}_L^{\text{old}}$. In other

words, the value of $\bigotimes_{i=1}^{K} S_{Z,i}^{\text{new}}$ indicates how to express the logical observable $\hat{Z}_L^{\text{new}}$ in terms of $\hat{Z}_L^{\text{old}}$ and $\{S_{Z,i}^{\text{new}}\}_{i=1}^{K}$. Consequently, faulty decoding of the measurement outcomes of these newly introduced Z-type stabilizers generators can potentially lead to an incorrect observable $\hat{Z}_L^{\text{new}}$. To complete the movement, the data qubits, represented by the lightly shaded area, are measured in the X basis, as depicted in Figure 4.6 (c). During this process, the logical observable $\hat{Z}_L^{\text{new}}$ remains unchanged, preserving its original state.

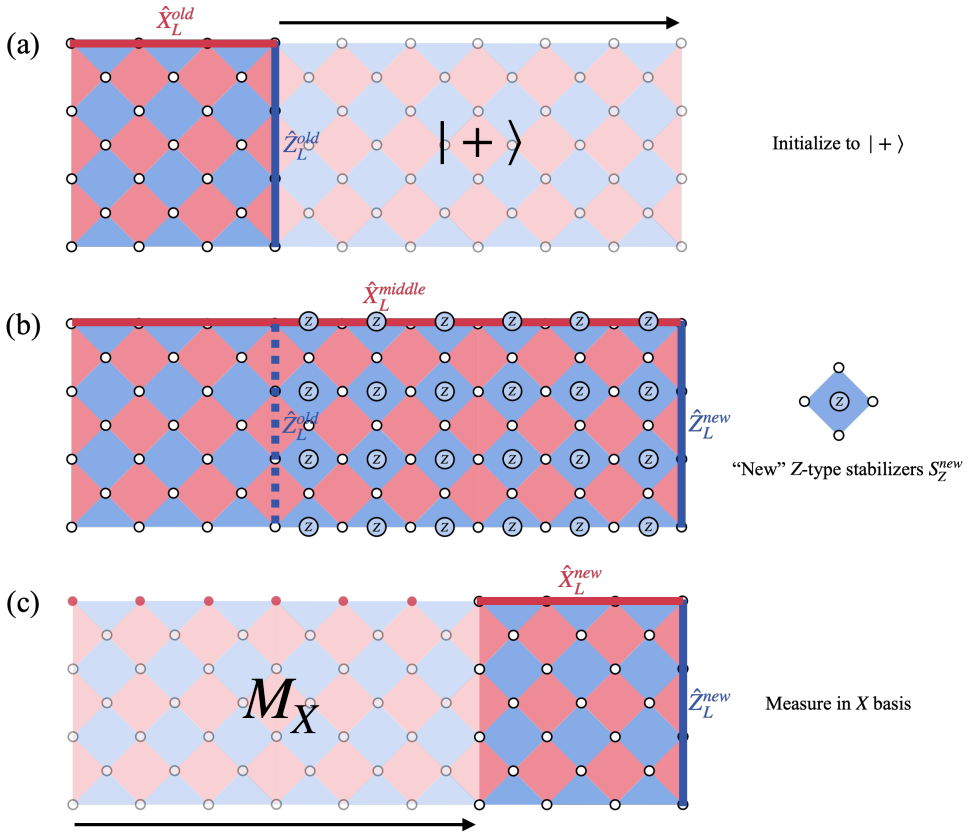

Figure 4.6: Transition of one logical qubit on the lattice. (a) The original distance-4 square surface code, illustrated with Z_L^{old} and X_L^{old} operations. New data qubits are initialized to the $|+\rangle$ state. (b) QEC cycles are executed on all stabilizers of the expanded code, with operations Z_L^{new} and X_L^{middle}. The newly incorporated Z-type stabilizers are depicted in the legend. (c) Data qubits to be removed are measured in the X basis. This results in a new distance-4 square surface code with operations Z_L^{new} and X_L^{new}.

In contrast to the movement of the logical Z observable, the movement of the logical X observable is straightforward. This is due to the fact that all new data qubits were initialized to the $|+\rangle$ state, resulting in a product of all new X-type stabilizers being $+1$. Consequently, we can confidently assert that $\hat{X}_L^{\text{middle}} = \hat{X}_L^{\text{old}}$ prior to decoding. When

contracting, however, it turns out that:

$$\hat{X}_L^{\text{middle}} = \hat{X}_L^{\text{new}} \cdot M \tag{4.26}$$

where M is the product of the measurement results of the data qubits (shown in red dots in Figure 4.6 (c)). This line of data qubits are removed from $\hat{X}_L^{\text{middle}}$ and thus we are left with $\hat{X}_L^{\text{new}}$. The measurements results of these data qubits can be fault-tolerantly determined, similarly as how we measure a logical observable of a logical qubit (as described in Chapter 1.2.5).

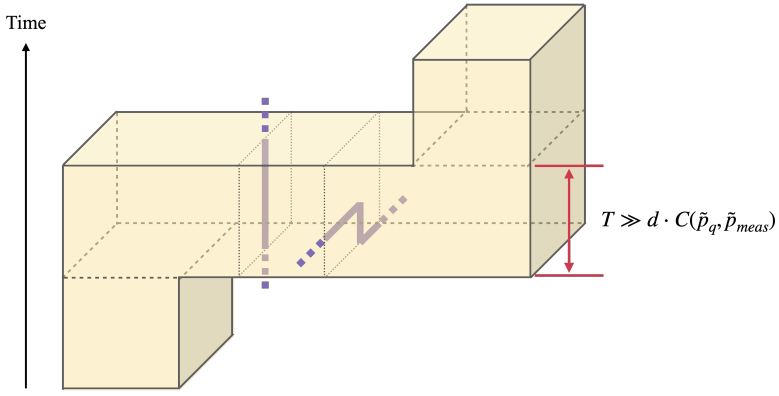

Figure 4.7: Spacetime volume and nontrivial chains of overall operations (errors and corrections) in the decoding process of moving a logical qubit. The error edges, indicated in purple, can lead to an incorrect sign for the moved logical observable. Solid lines represent edges connecting two nodes in the decoding graph, while dashed lines represent edges connecting a node to the boundary. The required number of QEC cycles needed for decoding the movement of a logical qubit after expanding the original code is shown on the right of the figure.

The figure presented in Figure 4.7 illustrates the spacetime volume and examples of nontrivial chains of overall operations (errors and correction) involved in the decoding process of moving a logical qubit. The purple lines indicate chains of overall operations that can result in an incorrect sign for the moved logical observable. Solid lines represent edges connecting two nodes in the spacetime lattice, while dashed lines connect a node to the boundary. Additionally, the figure demonstrates the required number of QEC cycles necessary for decoding the movement of a logical qubit after expanding the original code.

To accurately determine the number of QEC cycles needed to achieve a desired level of certainty in the correct movement of a logical qubit, it is crucial to ascertain the value of every $S_{Z,i}^{\text{new}}$ in this case. Examples of nontrivial chains of overall operations formed by errors and corrections that can lead to faulty decoding is highlighted in purple within the spacetime volume.

Similar to lattice surgery decoding, two types of harmful paths can result in failure during the movement of the logical qubit. The first type involves a path of overall operations, including errors and corrections, that connects the past- and future-time boundaries. This path contains at least T vertical (time-like) edges within the spacetime vol-

ume, where T corresponds to the number of QEC cycles performed after expanding the original code. The second type of path connects spatial boundaries and represents a potential logical error. This path contains at least d horizontal (space-like) edges within the spacetime volume. Both types of paths can lead to the failure of the logical qubit movement, resulting in an incorrect sign for the new logical observable or a logical error on the logical qubit. Hence, we obtain a similar result as with lattice surgery decoding. The probability of failure can be bounded by the expression:

$$\text{Prob}_{\text{fail}} \leq Q_1'(d,T) \cdot (C_1' \tilde{p}_{\text{q}})^{d/2} + Q_2'(d,T) \cdot (C_2' \tilde{p}_{\text{meas}})^{T/2} \tag{4.27}$$

where $Q_{1,2}'(d,T)$ are polynomials of d and T, and C_1' and C_2' are constants.

Furthermore, we observe a similar result as with lattice surgery decoding, stating that the required number of QEC cycles:

$$T \gg d \cdot C'(\tilde{p}_{\text{q}}, \tilde{p}_{\text{meas}}) \tag{4.28}$$

5

CONCLUSION

In the field of surface code quantum computation, repeatedly performing quantum error correction (QEC) cycles is crucial due to the high susceptibility of our quantum systems to errors. Conversely, excessive rounds of QEC cycles following one logical operation can impede the computation speed and subject idling logical qubits to additional errors. Hence, determining when and how many rounds of QEC cycles needed to attain a desirable decoding success rate is significant. This thesis has yielded several noteworthy results towards this goal.

Firstly, we elucidated the fault-tolerant and proper orders of syndrome measurement within each QEC cycle for the square surface code. We assessed two types of syndrome measurement orders. The Type-I order, despite necessitating fewer physical single-qubit H gates and reducing implementation duration, displays asymmetric resistance to logical X and Z errors due to the mirror-dual symmetry of the square surface code. In contrast, the Type-II order not only maintains a same level of logical X and Z error rates in quantum memory experiments, but also presents an advantage in terms of the averaged logical X and Z error rates over Type-I, particularly when the qubit coherence times and the single-qubit gate error rate fall below a specific threshold. These findings offer crucial guidance for the selection of syndrome measurement orders in future square surface code quantum computation experiments.

Secondly, this thesis provides a method for constructing detectors under code and stabilizer deformation. This method is instrumental for the precise identification and tracking of error syndromes, ensuring that decoders are provided with accurate information for decoding.

Thirdly, we have provided a theoretical explanation for the need to conduct more QEC cycles to achieve a longer spacetime volume for decoding when reliable stabilizer measurement outcomes are desired. We demonstrated two situations in which the knowledge of stabilizer measurement outcomes is crucial to proceed computation: lattice surgery and the movement of a logical qubit. Numerical experiments are conducted for measuring a joint logical observable of two surface codes using lattice surgery, and the results are consistent with our theoretical findings.

While these findings contribute significantly to the field, there are still areas requiring further exploration. The optimization of QEC cycles remains an open issue. Factors such as the type of code, methods of constructing the universal set of logical operations, hardware control, and noise in real-world experiments, among others, complicate this problem. However, addressing this issue promises to greatly benefit universal fault-tolerant quantum computation. This thesis stands as a valuable step towards resolving these challenges and advancing the field of surface code quantum computation.

BIBLIOGRAPHY

[1] Craig Gidney. “Stim: a fast stabilizer circuit simulator”. In: *Quantum* 5 (2021), p. 497.

[2] Daniel Gottesman. *Stabilizer codes and quantum error correction*. California Institute of Technology, 1997.

[3] Sergey B Bravyi and A Yu Kitaev. “Quantum codes on a lattice with boundary”. In: *arXiv preprint quant-ph/9811052* (1998).

[4] Austin G Fowler, Adam C Whiteside, and Lloyd CL Hollenberg. “Towards practical classical processing for the surface code”. In: *Physical review letters* 108.18 (2012), p. 180501.

[5] Austin G Fowler and Craig Gidney. “Low overhead quantum computation using lattice surgery”. In: *arXiv preprint arXiv:1808.06709* (2018).

[6] Héctor Bombın. “Topological order with a twist: Ising anyons from an abelian model”. In: *Physical review letters* 105.3 (2010), p. 030403.

[7] Theodore J Yoder and Isaac H Kim. “The surface code with a twist”. In: *Quantum* 1 (2017), p. 2.

[8] Panos Aliferis, Daniel Gottesman, and John Preskill. “Quantum accuracy threshold for concatenated distance-3 codes”. In: *arXiv preprint quant-ph/0504218* (2005).

[9] Austin G Fowler and Simon J Devitt. “A bridge to lower overhead quantum computation”. In: *arXiv preprint arXiv:1209.0510* (2012).

[10] David S Wang, Austin G Fowler, and Lloyd CL Hollenberg. “Surface code quantum computing with error rates over 1%”. In: *Physical Review A* 83.2 (2011), p. 020302.

[11] Eric Dennis et al. “Topological quantum memory”. In: *Journal of Mathematical Physics* 43.9 (2002), pp. 4452–4505.

[12] Kao-Yueh Kuo and Chung-Chin Lu. “On the hardnesses of several quantum decoding problems”. In: *Quantum Information Processing* 19 (2020), pp. 1–17.

[13] Elwyn Berlekamp, Robert McEliece, and Henk Van Tilborg. “On the inherent intractability of certain coding problems (corresp.)” In: *IEEE Transactions on Information Theory* 24.3 (1978), pp. 384–386.

[14] Oscar Higgott. “PyMatching: A Python package for decoding quantum codes with minimum-weight perfect matching”. In: *ACM Transactions on Quantum Computing* 3.3 (2022), pp. 1–16.

[15] Edsger W Dijkstra. “A note on two problems in connexion with graphs”. In: *Edsger Wybe Dijkstra: His Life, Work, and Legacy*. 2022, pp. 287–290.

[16] Jack Edmonds. “Paths, trees, and flowers”. In: *Canadian Journal of mathematics* 17 (1965), pp. 449–467.

[17] Daniel Gottesman. “The Heisenberg representation of quantum computers”. In: *arXiv preprint quant-ph/9807006* (1998).

[18] Yu Tomita and Krysta M Svore. “Low-distance surface codes under realistic quantum noise”. In: *Physical Review A* 90.6 (2014), p. 062320.

[19] Richard Versluis et al. “Scalable quantum circuit and control for a superconducting surface code”. In: *Physical Review Applied* 8.3 (2017), p. 034021.

[20] “Suppressing quantum errors by scaling a surface code logical qubit”. In: *Nature* 614.7949 (2023), pp. 676–681.

[21] Youwei Zhao et al. “Realization of an error-correcting surface code with superconducting qubits”. In: *Physical Review Letters* 129.3 (2022), p. 030501.

[22] JF Marques et al. “Logical-qubit operations in an error-detecting surface code”. In: *Nature Physics* 18.1 (2022), pp. 80–86.

[23] Sebastian Krinner et al. “Realizing repeated quantum error correction in a distance-three surface code”. In: *Nature* 605.7911 (2022), pp. 669–674.

[24] Xie Chen et al. “Subsystem stabilizer codes cannot have a universal set of transversal gates for even one encoded qudit”. In: *Physical Review A* 78.1 (2008), p. 012353.

[25] Bryan Eastin and Emanuel Knill. “Restrictions on transversal encoded quantum gate sets”. In: *Physical review letters* 102.11 (2009), p. 110502.

[26] Lingling Lao et al. “Mapping of lattice surgery-based quantum circuits on surface code architectures”. In: *Quantum Science and Technology* 4.1 (2018), p. 015005.

[27] Jonathan E Moussa. “Transversal Clifford gates on folded surface codes”. In: *Physical Review A* 94.4 (2016), p. 042316.

[28] Clare Horsman et al. “Surface code quantum computing by lattice surgery”. In: *New Journal of Physics* 14.12 (2012), p. 123011.

[29] Daniel Litinski. “A game of surface codes: Large-scale quantum computing with lattice surgery”. In: *Quantum* 3 (2019), p. 128.

[30] Jonghyun Lee et al. “Lattice surgery-based Surface Code architecture using remote logical CNOT operation”. In: *Quantum Information Processing* 21.6 (2022), p. 217.

[31] Christophe Vuillot et al. “Code deformation and lattice surgery are gauge fixing”. In: *New Journal of Physics* 21.3 (2019), p. 033028.

[32] Scott Aaronson and Daniel Gottesman. “Improved simulation of stabilizer circuits”. In: *Physical Review A* 70.5 (2004), p. 052328.

[33] Kaifeng Bu and Dax Enshan Koh. “Efficient classical simulation of Clifford circuits with nonstabilizer input states”. In: *Physical review letters* 123.17 (2019), p. 170502.

[34] Sergey Bravyi and Alexei Kitaev. “Universal quantum computation with ideal Clifford gates and noisy ancillas”. In: *Physical Review A* 71.2 (2005), p. 022316.

[35] Sergey Bravyi and Jeongwan Haah. “Magic-state distillation with low overhead”. In: *Physical Review A* 86.5 (2012), p. 052329.

[36] Yangsen Ye et al. “Logical Magic State Preparation with Fidelity Beyond the Distillation Threshold on a Superconducting Quantum Processor”. In: *arXiv preprint arXiv:2305.15972* (2023).

[37] Ying Li. “A magic state’s fidelity can be superior to the operations that created it”. In: *New Journal of Physics* 17.2 (2015), p. 023037.

[38] Neal Madras and Gordon Slade. *The self-avoiding walk*. Springer Science & Business Media, 2013.

6

APPENDIX

The code for the numerical simulation in this thesis can be found here.