Flags and Error Weight Parities: A Development of Fault-tolerant Quantum Computation with Few Ancillas

by

Theerapat Tansuwannont

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Doctor of Philosophy
in
Physics (Quantum Information)

Waterloo, Ontario, Canada, 2021

Examining Committee Membership

The following served on the Examining Committee for this thesis. The decision of the Examining Committee is by majority vote.

External Examiner:	Dan Browne Professor, Department of Physics and Astronomy, University College London
Supervisor:	Debbie Leung Professor, Department of Combinatorics and Optimization, University of Waterloo
Internal Member:	Raymond Laflamme Professor, Department of Physics and Astronomy, University of Waterloo
Internal-External Member:	David Gosset Associate Professor, Department of Combinatorics and Optimization, University of Waterloo
Other Member:	Beni Yoshida Faculty, Perimeter Institute for Theoretical Physics

Author's Declaration

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

In quantum computation, errors in a quantum circuit arising from its interaction with the environment is one of the biggest obstacles to building large-scale quantum computers. One way to deal with such errors is using fault-tolerant error correction (FTEC), a procedure which suppresses error propagation in a quantum circuit, together with other fault-tolerant gadgets for quantum computation to simulate a quantum circuit. However, for some platforms in which the number of physical qubits is limited, achieving a fault-tolerant simulation with very low logical error rate can be challenging since large overhead is required.

In this thesis, flag and weight parity techniques for FTEC which use only small number of ancillas are studied. The flag technique uses a few ancillas in circuits for syndrome measurement to detect high-weight errors arising from a few faults, while the weight parity technique uses weight parities and syndromes of errors to determine whether they are logically equivalent. The concepts of these two techniques can lead to the notion of distinguishable fault set, the central idea for the fault-tolerant protocol development in this thesis. In addition, fault-tolerant protocols for two families of codes are constructed: an FTEC protocol for the $[\![49, 1, 9]\!]$ concatenated Steane code which can correct up to 3 faults and uses 2 ancillas, and protocols for fault-tolerant quantum computation on capped color codes which require 1, 1, and 2 ancillas for the codes of distance 3, 5, and 7. The concept of distinguishable fault set also leads to a generalization of the definitions of fault-tolerant gadgets which give more flexibility when designing fault-tolerant protocols.

Acknowledgements

I would like to express my deepest gratitude to my supervisor, Prof. Debbie Leung, who has been an inspirational researcher as well as generous and understanding mentor. My research skills have been vastly improved in the past five years thanks to Debbie's insight and conscientiousness. I truly appreciate her helpful advice.

I am very grateful to Christopher Chamberland and Michael Vasmer for their suggestion of considering weight parity error correction on color codes. I would like to thank members of quantum error correction research group at Duke University, especially Prof. Ken Brown, Prof. Robert Calderbank, Rui Chao, Narayanan Rengaswamy, Arun Aloshious, Eric Sabo, and Shilin Huang for helpful discussions on the results, the manuscripts, and possible research directions. I would like to extend my gratitude to Prof. Dan Browne, Prof. David Gosset, Prof. Raymond Laflamme, Prof. Roger Melko, and Beni Yoshida for being parts of the advisory committee and the thesis defense committee, and for providing very helpful comments and suggestions. I would like to thank Junan Lin who has been a great friend and classmate at IQC.

I gratefully acknowledge Prof. Sujin Suwanna and Prof. Pruet Kalasuwan for helpful discussions on possible career paths in Thailand. I am also grateful to members of Thailand Quantum Initiative for interesting talks and discussions on recent works on quantum technologies. My Ph.D. studies are supported by The Queen Sirikit Scholarship under The Royal Patronage of Her Majesty Queen Sirikit of Thailand.

Last but not least, I would like to thank my beloved family and friends for always being there for me through good times and bad times. Your support means a lot to me.

Dedication

To my parents, Piyawadee and Wicharn, my elder brother, Pattarapong, and my little brothers and sisters, Owen, Beck, Ole, Ovi, Tigger, Sunny, and Rainy. Without you, I would not have made it this far.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction and Motivation

Computers developed on the current technology or *classical computers* are without doubt one of the most impactful innovations of the last century. Due to their impressive processing power, these tools have been used in almost every field of research, including quantum physics. However, there are several problems involving complicated quantum systems which cannot be solved efficiently by a classical computer. Although the performance of classical computers is still improving every year, the improvement seems slowing down because of the hardware limitation. Perhaps one solution to this problem is to build a computer on a different platform which is more compatible with the quantum systems.

In 1981, Richard Feynman proposed that a computer built on a quantum system, called *quantum computer*, might be able to solve a problem on other quantum systems efficiently because of the similarities in their nature [Fey82]. The notion of quantum computer later developed by David Deutsch in 1985 suggests that quantum computers can be viewed as a generalization of classical computers [Deu85]. In addition, several features of quantum systems which are not exhibited in classical systems may allow us to solve some problems unrelated to quantum physics on a quantum computer more efficiently than solving them on a classical computer. One of the most remarkable quantum algorithms that draws public attention to quantum computer is Shor's factoring algorithm in 1994 [Sho94]; the best known classical factoring algorithm requires sub-exponential time, while Shor's result predicts that solving a factoring problem on a quantum computer could be done in polynomial time.

Although quantum algorithms could be more powerful than classical algorithms for some types of problems, building a reliable quantum computer is not easy. One major obstacle is that a physical implementation of qubits and quantum gates in a quantum circuit can sometimes cause small errors, and such errors can propagate throughout the circuit and become intractable. One solution to this problem is to use quantum error correction (QEC), a process in which quantum data is encoded by a quantum error correcting code (QECC) so that small errors can be later detected and corrected. Unfortunately, the error correction procedure itself can sometimes be faulty and cause logical errors. For this reason, the procedure must be modified so that a few number of faults can be tolerated.

Fault-tolerant error correction (FTEC), a procedure which suppresses error propagation in a quantum circuit, is one of the most important components for building large-scale quantum computers. Given that the physical error rate is below some constant threshold value, an FTEC scheme along with other schemes for fault-tolerant quantum computation (FTQC) allow us to fault-tolerantly simulate any quantum circuit with arbitrarily low logical error rates [Sho96, ABO08, Kit97, KLZ96, Pre98, TB05, ND05, AL06, AGP06]. However, lower logical error rate requires more overhead (e.g., quantum gates and ancilla qubits) [Ste03, PR12, CJOL17, TYC17], making practical implementation on a platform in which the number of qubits is very limited more difficult. Therefore, fault-tolerant protocols which require a small number of ancillas and give high threshold value are very desirable.

Traditional FTEC schemes require substantial number of ancillas for error syndrome measurements. For example, the Shor error correction (EC) scheme [Sho96, DA07] which is applicable to any stabilizer code requires as many ancillas as the maximum weight of the stabilizer generators. The Knill EC scheme [Kni05], which is also applicable to any stabilizer code, requires two code block of ancillas. Meanwhile, the Steane EC scheme [Ste97, Ste02] which is applicable to any CSS code requires one code block of ancillas. (The Shor scheme also requires repeated syndrome measurement, while the Knill and the Steane schemes do not.) There are several recently proposed schemes which require fewer ancillas. Yoder and Kim proposed an FTEC scheme for the $[\![7,1,3]\!]$ code which requires only 2 ancillas [YK17], and their scheme is further developed into a well-known flag FTEC scheme for the $[\![5,1,3]\!]$ code and the $[\![7,1,3]\!]$ code which also require only 2 ancillas [CR18c] (where an $[\![n,k,d]\!]$ stabilizer code encodes k logical qubits into n physical qubits and has distance d). In general, a flag FTEC scheme for any stabilizer code requires as few as $d+1$

ancillas where d is the code distance [CR20], with further reduction known for certain families of codes [CR18c, CB18, TCL20, CKYZ20, CZY+20]. The flag technique can also be applied to other schemes for FTQC [CR18b, CC19, SCC19, BCC+19, BXG+19, Vui18, GMB19, LA20, CN20, DB20, RBBMS21].

How errors spread during the protocols depends on several factors such as the order of quantum gates in the circuits for syndrome measurement and the choice of stabilizer generators being measured. The idea behind the flag technique is that a few ancillas are added to the circuits in order to detect errors of high weight arising from a few faults, and the errors will be distinguished by their syndromes obtained from subsequent syndrome measurements. Note that some possible errors may be logically equivalent and need not be distinguished, and for some families of codes, we can tell whether the errors are logically equivalent using their syndromes and error weight parities. Moreover, the use of error weight parities can further reduce the number of required ancillas for some families of codes. In this thesis, fault-tolerant protocols for a family of concatenated codes and a family of capped color codes will be developed. These protocols are the main results of two papers: *Fault-tolerant quantum error correction using error weight parities* [TL21b] and *Achieving fault tolerance on capped color codes with few ancillas* [TL21a].

In [TL21b], we introduce a technique called weight parity error correction (WPEC) and construct an FTEC scheme for the $[\![49, 1, 9]\!]$ concatenated Steane code using only *two* ancilla qubits. The scheme relies on the fact that, for the $[\![7, 1, 3]\!]$ code, errors with the same syndrome and weight parity differ by the multiplication of some stabilizer; these errors are thus logically equivalent and need not be distinguished from one another. Therefore, error correction on each subblock of 7 qubits in the $[\![49, 1, 9]\!]$ code can be accomplished using only two ingredients: the error syndrome and the weight parity of error in each subblock. Most importantly, the weight parity for each subblock of 7 qubits in the $[\![49, 1, 9]\!]$ code can be obtained from the full syndrome measurement. Using this idea in conjunction with 2 ancilla qubits, our FTEC protocol for the $[\![49, 1, 9]\!]$ code can correct up to 3 faults. As a result, our protocol can suppress the error rate from p to $O(p^4)$ using 51 qubits in total.

The EC technique using weight parities introduced in [TL21b] which is originally developed for the $[\![7, 1, 3]\!]$ code can also be extended to some families of codes such as 2D color codes. In [TL21a], the weight parity of an error on a 2D color code is obtained by measuring stabilizer generators of a bigger code which contains the 2D color code as a subcode. In contrast to [TL21b], the bigger code presented in [TL21a] is not obtained from code

concatenation. Our development for FTEC protocols leads to a family of capped color codes which are CSS subsystem codes [Pou05, Bac06]. We study two stabilizer codes obtained from a (subsystem) capped color code through gauge fixing, namely capped color codes in H form and T form. The code in H form which contains a 2D color code as a subcode has transversal Clifford gates, while the code in T form has transversal CNOT and transversal T gates. In fact, our capped color codes bear similarities to the subsystem codes presented in [JOB16, BC15, JBH16], in which qubits can be arranged on a 2D plane.

Here is a summary of our contributions in [TL21a]. First, we develop a notion of distinguishable fault set which replaces the role of correctable errors in prior work. This facilitates the correction of high-weight errors using the flag and weight parity techniques. Second, we construct circuits for measuring generators of a capped color code in H form, and construct an FTEC scheme as well as other fault-tolerant schemes for measurement and state preparation. This requires the notion of distinguishable fault set, extension of weight parity techniques from [TL21b], suitable embedding of the 2D color code, and judicious ordering of gates in the measurement circuits. Our schemes for capped color codes in H form of distance 3, 5, and 7 require only 1, 1, and 2 ancillas, respectively. We also propose a theorem which can be helpful for finding the circuits for a capped color code of higher distance. Third, we extend the definitions of fault-tolerant gadgets that are compatible with the notion of distinguishable fault set, which can be viewed as a generalization of the definitions of fault-tolerant gadgets proposed by Aliferis, Gottesman, and Preskill [AGP06].

This thesis is organized as follows: In Chapter 2, the basics of quantum error correction in stabilizer formalism will be reviewed. Next, FTEC techniques using flags and error weight parities and the notion of distinguishable fault set will be discussed in Chapter 3. The concept of distinguishable fault set can lead to an alternate version of definitions of fault-tolerant gadgets, which will be presented alongside the conventional definitions by Aliferis, Gottesman, and Preskill in Chapter 4. Afterwards, fault-tolerant protocols developed for the $[\![49, 1, 9]\!]$ concatenated Steane code and capped color codes will be elaborated in Chapter 5 and Chapter 6, respectively. Finally, the results of [TL21b] and [TL21a] and possible directions for future work will be discussed in Chapter 7.

Chapter 2

Quantum Error Correction in Stabilizer Formalism

Physical noises on a quantum system arising from its interaction with the environment are one of the biggest obstacles to building large-scale quantum computers. In classical computation, one way to protect the logical information against noises is encoding the information using a classical error correcting code. Similar ideas can be developed for quantum computation. However, due to the differences in their natures, some techniques must be modified so that error correction on a quantum system is possible.

In this section, we will first review the basics of quantum error correction in Section 2.1. Afterwards, the stabilizer formalism will be described in Section 2.2. The concepts of syndrome measurement, logical operators, and error decoding problem will be later elaborated in Sections 2.3 and 2.4, and 2.5.

2.1 Basics of quantum error correction

In classical information processing, the smallest unit of information is called *bit*, which can be in either state 0 or 1. Any classical information can be represented by a bitstring of length n for some positive integer n, denoted by $\vec{x} \in \mathbb{Z}_2^n$. Similarly, the smallest unit used to represent quantum information is called quantum bit or *qubit*. However, one major

difference between the state of qubits and the state of classical bits is that a superposition of states is allowed in the quantum case. In particular, the state of a qubit can be represented by a vector in the 2-dimensional complex Euclidean space $\mathbb{C}^2$ of the form,

$$\alpha|0\rangle + \beta|1\rangle,$$

where $|0\rangle = \binom{1}{0}, |1\rangle = \binom{0}{1}$, and α, β are some complex numbers satisfying $|\alpha|^2 + |\beta|^2 = 1$. In addition, any quantum information can be represented by a state of n qubits for some positive interger n which, in general, can be written as,

$$\begin{aligned} |\psi\rangle &= c_{00\ldots0}|00\ldots0\rangle + c_{00\ldots1}|00\ldots1\rangle + \cdots + c_{11\ldots1}|11\ldots1\rangle \\ &= \sum_{\vec{x}\in\mathbb{Z}_2^n} c_{\vec{x}}|\vec{x}\rangle, \end{aligned} \tag{2.1}$$

where $\{|\vec{x}\rangle\}$ is an orthonormal basis of the 2^n-dimensional complex Euclidean space $\mathbb{C}^{2^n}$, and the complex numbers $c_{\vec{x}}$ satisfy $\sum_{\vec{x}\in\mathbb{Z}_2^n} |c_{\vec{x}}|^2 = 1$. If the quantum state in Eq. (2.1) is measured in the computational basis, the state in superposition will collapse to some state $|\vec{x}\rangle$ with probability $|c_{\vec{x}}|^2$.

When classical information is sent through a classical channel, some errors due to physical noises may occur, and the state of each classical bit may be flipped. One way to suppress the error rate is to encode each classical bit by a classical error correcting code and perform error correction afterwards. Let us consider the following encoding and decoding schemes of the 3-bit repetition code as an example:

$$\begin{aligned} \text{Encoding}: \quad & 0 \mapsto 000, \\ & 1 \mapsto 111, \\ \text{Decoding}: \quad & 000, 001, 010, \text{ or } 100 \mapsto 0, \\ & 111, 110, 101, \text{ or } 011 \mapsto 1. \end{aligned}$$

These schemes can correct a bit-flip error on up to 1 bit; that is, if at least two out of three bits of each encoded state remain unaffected after transmission, the original data (0 or 1) can be recovered. However, if bit-flip errors occur on two or more bits, the decoding scheme above will fail to retrieve the original data. Suppose that a single bit is flipped from 0 to 1 (or from 1 to 0) with probability p and remains unchanged with probability

$1-p$ after transmission. Then the probability that two or more bits in a bitstring of length 3 will flip after transmission is $3p^2(1-p)+p^3$, which is less than p if $p < \frac{1}{2}$. In other words, the 3-bit repetition code described above can suppress the error rate whenever $p < \frac{1}{2}$.

One may want to apply the aforementioned schemes directly to a quantum system in order to protect its quantum information. However, there are a few obstacles that prevent us to do so: First, it is impossible to construct a quantum state $|\psi\rangle \otimes |\psi\rangle \otimes |\psi\rangle$ from an unknown state $|\psi\rangle$; this fact is also known as the no-cloning theorem [NC00]. Second, the aforementioned decoding scheme requires the knowledge of the states of classical bits in order to recover the original data. In quantum settings, this method of decoding is equivalent to measuring the state of a quantum system, which can cause a superposition of quantum states to collapse. Third, possible errors in quantum settings are not limited to bit-flip errors, and they can be continuous in general. To overcome these issues, an error correction scheme for a quantum state must be improved.

A quantum error correcting code (QECC) Q can be defined as a subspace of the complex Euclidean space $\mathbb{C}^{2^n}$ (where n is the number of physical qubits). Let $\mathcal{E}$ be a set of errors that we aim to correct. In order to recover the original state from an erroneous state, we have to make sure that two orthogonal codewords remain orthogonal after some errors in $\mathcal{E}$ occur to the codewords. That is, the QECC Q and the set of errors $\mathcal{E}$ must satisfy the following condition [KL97]:

Theorem 2.1 *Let Q be a subspace of the complex Euclidean space $\mathbb{C}^{2^n}$, $\{|\bar{\psi}_i\rangle\}$ be an orthonormal basis of Q, and $\mathcal{E}$ be a set of errors. A quantum error correcting code Q can correct all errors in $\mathcal{E}$ if and only if for all pairs of basis vectors $|\bar{\psi}_i\rangle, |\bar{\psi}_j\rangle$, for all pairs of errors $E_a, E_b \in \mathcal{E}$,*

$$\langle\bar{\psi}_i|E_a^\dagger E_b|\bar{\psi}_j\rangle = C_{ab}\delta_{ij}, \tag{2.2}$$

where C_{ab} are some constants independent of the codewords, $\delta_{ij} = 1$ if $i = j$, and $\delta_{ij} = 0$ if $i \neq j$.

If the condition in Theorem 2.1 is satisfied, a QEC scheme which works for all basis states will also work for any superposition state. (To make this possible, we must obtain the information of the occurred error when performing QEC, not the information of the quantum state. An EC procedure will be further discussed in Section 2.3.) Moreover, if a QECC can correct two errors E and F, it can also correct any error of the form

$\alpha E + \beta F$, where α, β are some complex numbers. Let us consider the following Pauli operators defined on a single qubit:

$$I = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \quad X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \quad Y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}, \quad Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}. \tag{2.3}$$

Since any error on a single qubit can be written as a linear combination of I, X, Y, and Z, a QECC which can correct all Pauli errors (where I is the identity operator and need not be corrected) can also correct any error on a single qubit. This idea can also be generalized to the case of multi-qubit errors.

2.2 Stabilizer code

In the previous section, a QECC is described by the basis states $\{|\bar{\psi}_i\rangle\}$ of the coding subspace. However, this representation of a QECC may not be convenient in calculation since each basis state $|\bar{\psi}_i\rangle$ may have many terms. Fortunately, there is a family of quantum codes called *stabilizer codes* whose codewords can be described by $+1$ eigenvectors of some Pauli operators. In this section, the stabilizer formalism [Got96, Got97] for quantum codes will be elaborated.

To begin with, let us define a Pauli group as follows:

Definition 2.1 *Pauli group and the weight of Pauli operator*

The 1-qubit Pauli group $\mathcal{P}_1$ *is a group of* I, X, Y *and* Z *operators with a* ± 1 *or* $\pm i$ *phase factor; i.e.,*

$$\mathcal{P}_1 = \{\pm I, \pm iI, \pm X, \pm iX, \pm Y, \pm iY, \pm Z, \pm iZ\}. \tag{2.4}$$

The n-qubit Pauli group $\mathcal{P}_n$ *is a group of* n*-qubit tensor products of operators in* $\mathcal{P}_1$*; i.e.,*

$$\mathcal{P}_n = \{P_1 \otimes \cdots \otimes P_n : P_i \in \mathcal{P}_1 \ \textit{for all} \ i = 1, \ldots, n\}. \tag{2.5}$$

The weight $\mathrm{wt}(P)$ *of* $P \in \mathcal{P}_n$ *is the number of non-identity tensor factors of* P*.*

The Pauli group $\mathcal{P}_n$ has the following properties:

1. A product of any operators in $\mathcal{P}_n$ is also an operator in $\mathcal{P}_n$.
2. An operator P in $\mathcal{P}_n$ is either Hermitian ($P^\dagger = P$) or anti-Hermitian ($P^\dagger = -P$).
3. For all $P \in \mathcal{P}_n$, P^2 is $I^{\otimes n}$ (when P is Hermitian) or $-I^{\otimes n}$ (when P is anti-Hermitian).
4. Any two operators P_1, P_2 in $\mathcal{P}_n$ either commute ($[P_1, P_2] = P_1P_2 - P_2P_1 = 0$) or anticommute ($\{P_1, P_2\} = P_1P_2 + P_2P_1 = 0$).

A *stabilizer group* is an Abelian group of Pauli operators which does not contain $-I^{\otimes n}$. The group can be generated from some independent, commuting operators in $\mathcal{P}_n$. Operators in a stabilizer group are sometimes called *stabilizers*. For each stabilizer group S, we can define the coding subspace $T(S)$ corresponding to S as follows:

Definition 2.2 *Stabilizer group and stabilizer code*

Let stabilizer group $S \subset \mathcal{P}_n$ *be an Abelian group with* $-I^{\otimes n} \notin S$. *The* stabilizer code $T(S)$ *is the subspace of the complex Euclidean space* $\mathbb{C}^{2^n}$ *defined by,*

$$T(S) = \{|\bar{\psi}\rangle : M|\bar{\psi}\rangle = |\bar{\psi}\rangle \ \ \textit{for all} \ \ M \in S\}. \tag{2.6}$$

A stabilizer group S has the following properties:

1. For all $M, N \in S$, MN is also in S (the group is closed under multiplication). That is, for all $|\bar{\psi}\rangle \in T(S)$,
$$MN|\bar{\psi}\rangle = M|\bar{\psi}\rangle = |\bar{\psi}\rangle. \tag{2.7}$$
2. For all $M, N \in S$, $MN = NM$ (the group is Abelian). That is, for all $|\bar{\psi}\rangle \in T(S)$,
$$MN|\bar{\psi}\rangle = NM|\bar{\psi}\rangle = |\bar{\psi}\rangle. \tag{2.8}$$
3. Because $-I^{\otimes n} \notin S$ by the definition of stabilizer group, all stabilizers in S is Hermitian ($M^2 = I^{\otimes n}$ for all $M \in S$).

4. S can be generated by r independent, commuting Pauli operators $\{g_1, g_2, \ldots, g_r\}$ for some positive integer r. Here we will write $S = \langle g_1, g_2, \ldots, g_r \rangle$. In addition, any stabilizer $M \in S$ can be written as,

$$M = g_1^{a_1} g_2^{a_2} \cdots g_r^{a_r}, \tag{2.9}$$

where $a_i \in \{0, 1\}, i = 1, 2, \ldots, r$. Consequently, the size of S is $|S| = 2^r$.

5. Because $T(S)$ is the subspace defined by common $+1$ eigenvectors of r independent, commuting Pauli operators, the dimension in $T(S)$ is $\dim(T(S)) = 2^{n-r}$. We define $k = n - r$ to be the number of logical qubits encoded by a stabilizer code.

It should be noted that not all QECCs are stabilizer codes. Nevertheless, this thesis will consider only QECCs which are stabilizer codes. Because of simpler representations, the codes and related EC techniques in this thesis will be described in the stabilizer formalism.

One example of a QECC which is a stabilizer code is the Shor code [Sho95]. This code encodes 1 logical qubit into 9 physical qubits and can correct up to 1 error. A codeword (or an encoded state) of the Shor code can be expressed by the following basis states:

$$|\bar{0}\rangle = \left(\frac{|000\rangle + |111\rangle}{\sqrt{2}}\right) \otimes \left(\frac{|000\rangle + |111\rangle}{\sqrt{2}}\right) \otimes \left(\frac{|000\rangle + |111\rangle}{\sqrt{2}}\right), \tag{2.10}$$

$$|\bar{1}\rangle = \left(\frac{|000\rangle - |111\rangle}{\sqrt{2}}\right) \otimes \left(\frac{|000\rangle - |111\rangle}{\sqrt{2}}\right) \otimes \left(\frac{|000\rangle - |111\rangle}{\sqrt{2}}\right). \tag{2.11}$$

The Shor code can be considered as an extension of the classical 3-bit repetition code described in Section 2.1 to the quantum settings. In the stabilizer formalism, the Shor code can be described by the following generators:

$$\begin{array}{lccccccccc} g_1: & Z & Z & I & I & I & I & I & I & I \\ g_2: & I & Z & Z & I & I & I & I & I & I \\ g_3: & I & I & I & Z & Z & I & I & I & I \\ g_4: & I & I & I & I & Z & Z & I & I & I \\ g_5: & I & I & I & I & I & I & Z & Z & I \\ g_6: & I & I & I & I & I & I & I & Z & Z \\ g_7: & X & X & X & X & X & X & I & I & I \\ g_8: & I & I & I & X & X & X & X & X & X \end{array}. \tag{2.12}$$

Another example of a stabilizer code is the 7-qubit Steane code [Ste96], which encodes 1 logical qubit into 7 physical qubits and can correct up to 1 error. The Steane code can be constructed from the parity check matrix of the classical 7-bit Hamming code [MS77, CS96, Ste96]. In the stabilizer formalism, we can describe the Steane code by the following generators:

$$\begin{aligned} g_1^x &: I\ \ I\ \ I\ X\,X\,X\,X, \quad & g_1^z &: I\ I\ I\ Z\,Z\,Z\,Z, \\ g_2^x &: I\ X\,X\ I\ \ I\ X\,X, \quad & g_2^z &: I\ Z\,Z\ I\ \ I\ Z\,Z, \\ g_3^x &: X\ I\ X\ I\ X\ I\ X, \quad & g_3^z &: Z\ I\ Z\ I\ Z\ I\ Z. \end{aligned} \tag{2.13}$$

The Steane code plays an important role in the fault-tolerant protocols developed [TL21b] and [TL21a], which will be described later in Chapter 5 and Chapter 6.

2.3 Syndrome measurement

Whenever an error occurs to the encoded state, in order to perform error correction, we need to obtain some information about the error without measuring the encoded state directly (otherwise a superposition state will collapse). Observe that any encoded state $|\bar{\psi}\rangle$ is a $+1$ eigenvector of any generator of the stabilizer group. Whenever a Pauli error $E \in \mathcal{P}_n$ occurs to the encoded state, the corrupted state $E|\bar{\psi}\rangle$ may become a -1 eigenvector of some generators. In an EC procedure, whether the corrupted state is a $+1$ or -1 eigenvector of each generator will determined through a process called *syndrome measurement*, described in this section.

Let us consider a stabilizer $M \in S$ and a Pauli error $E \in \mathcal{P}_n$. From the properties of the Pauli group, we know that E and M either commute ($[E, M] = 0$) or anticommute ($\{E, M\} = 0$). If E commutes with M, we find that

$$ME|\bar{\psi}\rangle = EM|\bar{\psi}\rangle = E|\bar{\psi}\rangle, \tag{2.14}$$

for all $|\bar{\psi}\rangle \in T(S)$. That is, $E|\bar{\psi}\rangle$ is a $+1$ eigenvector of M. In contrast, if E anticommutes with M, then

$$ME|\bar{\psi}\rangle = -EM|\bar{\psi}\rangle = -E|\bar{\psi}\rangle, \tag{2.15}$$

for all $|\bar{\psi}\rangle \in T(S)$. That is, $E|\bar{\psi}\rangle$ is a -1 eigenvector of M.

Suppose that the stabilizer group can be generated by $\{g_1, g_2, \ldots, g_r\}$. We define the *syndrome* of E (denoted by $\vec{s}(E)$) to be an r-bit string, where the i-th bit is 0 (or 1) if $[E, g_i] = 0$ (or $\{E, g_i\} = 0$). If E anticommutes with at least one generator (i.e., $\vec{s}(E)$ is nontrivial), we say that E is *detectable*. In addition, if the weight of E is no more than the number of errors that a QECC can correct, then the syndrome $\vec{s}(E)$ can be used to find an error correction operator F such that $FE|\bar{\psi}\rangle = |\bar{\psi}\rangle$ for all $|\bar{\psi}\rangle \in T(S)$; that is, F maps a corrupted state back to the original encoded state.

The error syndrome of an error occurred to a codeword can be obtained by measuring eigenvalues of all generators of the stabilizer code. Let us consider the 7-qubit Steane code as an example. The eigenvalue of the generator $g_1^z = Z_4Z_5Z_6Z_7$ (or $g_1^x = X_4X_5X_6X_7$) can be obtained by measuring the ancilla qubit in the circuit displayed in Fig. 2.1a (or Fig. 2.1b), where the subscripts denote the qubits that the operators act on. In each circuit, the ancilla qubit is initially prepared in state $|0\rangle$ and measured in Z basis (the computational basis) at the end. The measurement results 0 and 1 correspond to $+1$ and -1 eigenvalues, respectively.

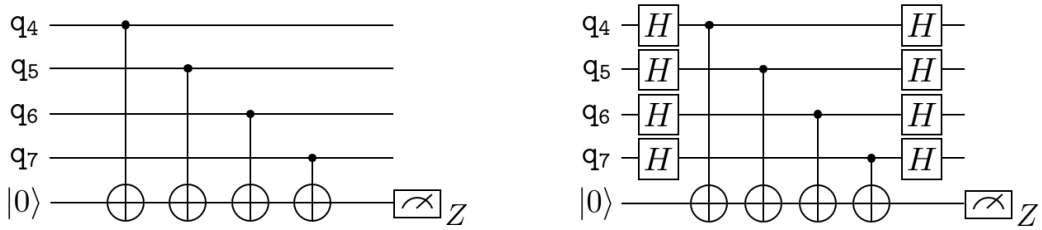


Figure 2.1: Eigenvalues of stabilizer generators $g_1^z = Z_4Z_5Z_6Z_7$ and $g_1^x = X_4X_5X_6X_7$ of the 7-qubit Steane code can be measured using the circuits illustrated in (a) and (b). Only qubits on which each operator acts nontrivially are displayed.

2.4 Normalizer group and logical operators

Some Pauli error may commute with all stabilizer generators. In that case, the syndrome of the error is trivial, and the error is *undetectable*. Let us consider the group of Pauli

operators that commute with all generators of a stabilizer group defined as follows:

Definition 2.3 *Normalizer group and code distance*

Let $S \subset \mathcal{P}_n$ be a stabilizer group. The normalizer group *$N(S)$ of S is defined by,*

$$N(S) = \{L \in \mathcal{P}_n : LM = ML \ \ \textit{for all} \ \ M \in S\}. \tag{2.16}$$

The distance *of the stabilizer code $T(S)$ is defined as the minimum weight of Pauli operators in $N(S) - S$.*

Obviously, any stabilizer $M \in S$ is also an element of $N(S)$, thus $S \subseteq N(S)$.

Let us consider the action of any operator $L \in N(S)$ on a codeword. Since L commutes with all stabilizers in S, we find that,

$$ML|\bar{\psi}\rangle = LM|\bar{\psi}\rangle = L|\bar{\psi}\rangle, \tag{2.17}$$

for all $|\bar{\psi}\rangle \in T(S)$, for all $M \in S$. Here we can see that $L|\bar{\psi}\rangle$ is a $+1$ eigenvector of all stabilizers, i.e., $L|\bar{\psi}\rangle$ is also a valid codeword. Since L maps a valid codeword to another valid codeword, we sometimes called operators in $N(S)$ *logical operators*. If L is a stabilizer in S, it is equivalent to the logical identity operator since it acts trivially on a codeword. However, if $L \in N(S) - S$, it is a nontrivial logical operator since it maps a codeword to a different codeword.

When an error $E \in S$ occurs to a codeword, error correction is not needed since E stabilizes the codeword ($E|\bar{\psi}\rangle = |\bar{\psi}\rangle$ for all $|\bar{\psi}\rangle \in T(S)$). However, if an error $E \in N(S) - S$ occurs to a codeword, the EC procedure fails to recover the original state. This is because $E|\bar{\psi}\rangle$ is different from $|\bar{\psi}\rangle$ but E cannot be detected since it commutes with all stabilizer generators. The distance of a stabilizer code is therefore the minimum weight of errors undetectable by the code. This means that a stabilizer code of distance d can detect any error of weight $\leq d - 1$.

Now, let us consider the relationship between the code distance and the weight of errors that a stabilizer code can correct. Suppose that $\mathcal{E}$ is a set of all Pauli errors of weight up to τ and a stabilizer code $T(S)$ can correct all errors in $\mathcal{E}$. Because $T(S)$ is a QECC, the

condition in Theorem 2.1 must hold, i.e.,

$$\langle \bar{\psi}_i | E_a^\dagger E_b | \bar{\psi}_j \rangle = C_{ab} \delta_{ij}, \tag{2.18}$$

for all pairs of basis vectors $|\bar{\psi}_i\rangle, |\bar{\psi}_j\rangle$, for all pairs of errors $E_a, E_b \in \mathcal{E}$. Here we find that $E_a^\dagger E_b$ cannot be an operator in $N(S) - S$ (otherwise, the inner product between $E_a^\dagger E_b |\bar{\psi}_j\rangle$ and $|\bar{\psi}_i\rangle$ is not zero for some $i \neq j$, and the condition will not be satisfied). That is, $E_a^\dagger E_b$ must be *detectable*, i.e., the syndrome of $E_a^\dagger E_b$ is nontrivial. Since $\mathcal{E}$ contains all Pauli errors of weight $\leq \tau$, the maximum weight of $E_a^\dagger E_b$ is 2τ. Therefore, the distance of the code must be at least $2\tau + 1$.

The relationship between the code distance, the weight of correctable errors, and the weight of detectable errors can be summarized as follows:

Proposition 2.1 *A stabilizer code of distance d can detect errors up to weight $d-1$ and correct errors up to weight $\tau = \lfloor (d-1)/2 \rfloor$.*

We sometimes use the following notation to describe a stabilizer code:

Definition 2.4 *An $[\![n, k, d]\!]$ code is a stabilizer code which encodes k logical qubits into n physical qubits and has distance d.*

2.5 Error decoding problem

As previously mentioned, whenever an error E occurs to a codeword, we will try to remove the error by measuring the error syndrome $\vec{s}(E)$ then find an EC operator F which has the same syndrome as $\vec{s}(E)$. In this thesis, we follow several prior works on QEC and adopt the terminology 'decoding' for the process to determine F from $\vec{s}(E)$. This should not be confused with the process of finding an unencoded state corresponding to each codeword as previously discussed in Section 2.1 (although these two processes are closely related). In this section, we will consider an issue that may occur when E has high weight.

If the weight of an error E occurred to a codeword is no more than the weight of errors τ that the code can correct, then an EC operator F in which $\vec{s}(F) = \vec{s}(E)$ are logically equivalent to E, and the original codeword $|\bar{\psi}\rangle$ can be obtained by applying F to the

corrupted codeword $E|\bar{\psi}\rangle$. However, if the weight of E is more than τ, there is no guarantee that $FE|\bar{\psi}\rangle$ will be the same state as $|\bar{\psi}\rangle$. Let us consider two Pauli operators E_1, E_2 which correspond to the same syndrome. From the fact that an operator $L \in N(S)$ commutes with all stabilizer generators, and because E_1 commutes (or anticommutes) with generator g_i if and only if E_2 commutes (or anticommutes) with the same generator g_i for all $i = 1, \ldots, r$, it is possible to write $E_2 = E_1 \cdot L$ for some $L \in N(S)$ (up to some phase factor).

Note that $L \in N(S)$ can be either trivial (such as $I^{\otimes n}$ or some stabilizer) or nontrivial (such as logical X or logical Z operator). From the aforementioned EC procedure where we choose an EC operator F such that $\vec{s}(F) = \vec{s}(E)$, if we pick F such that $F \cdot E \in S$, then the output state after error correction will be the same as the original state. However, if we pick F such that $F \cdot E \in N(S) - S$, then the output state will differ from the original state and we fail to perform error correction.

In an actual EC protocol, errors of weight greater than τ may arise depending on the noise model and other factors such as the structure of the circuits being used. Since we can only measure the error syndrome but do not know what error has occurred to the codeword, finding an appropriate EC operator for each syndrome is one of the main challenges when developing an EC protocol.

Chapter 3

Flags and Error Weight Parities in Error Correction

In general, a QEC procedure described in Chapter 2 may not be perfect, and errors arising from a few faults during the procedure can propagate throughout the circuit as the quantum state evolves. If not treated properly, these errors may become too large to correct in later parts of the computation. Fortunately, if the QEC procedure satisfies some conditions, fault-tolerant quantum computation can be done; i.e., the logical error rate can be made arbitrarily small when the physical error rate is smaller than some constant threshold value. This result is known as the *threshold theorem*. There are several works on the proofs of the threshold theorem which consider different sets of assumptions [Sho96, ABO08, Kit97, KLZ96, Pre98, TB05, ND05, AL06, AGP06]. In this thesis, a version of the threshold theorem proved by Aliferis, Gottesman, and Preskill [AGP06] will be considered; we will try to construct fault-tolerant protocols satisfying the definitions of fault-tolerant gadgets proposed in [AGP06] (these definitions will be later discussed in Chapter 4).

One common goal of the two papers [TL21b] and [TL21a] presented in this thesis is to construct FTEC protocols using the flag and weight parity EC techniques. In this chapter, we start by providing a brief review on the flag technique applied to the case of one fault in Section 3.1. Next, the idea will be extended to the case of multiple faults in Section 3.2, and the notion of distinguishable fault set will be introduced in Definition 3.3. Afterwards, how weight parities can be used in error correction will be explained in Section 3.3. The contents of this chapter follows Section II of [TL21a].

3.1 Flag error correction

Quantum computation is prone to noise, and an error on a few qubits can spread and cause a big problem in the computation if the error is not treated properly. As we have seen in Chapter 2, one way to protect quantum data against noise is to use a quantum error correcting code (QECC) to encode a small number of logical qubits into a larger number of physical qubits. A quantum $[\![n, k, d]\!]$ stabilizer code [Got96, Got97] encodes k logical qubits into n physical qubits and can correct errors up to weight $\tau = \lfloor (d-1)/2 \rfloor$. Quantum error correction (QEC) is a process that aims to undo the corruption that happens to a codeword.

A stabilizer code is a simultaneous $+1$ eigenspace of a list of commuting independent Pauli operators; they generates the stabilizer group for the code. For a stabilizer code, the error correction (EC) procedure involves measurements of stabilizer generators, which results in an error syndrome. The QEC is designed so that the more likely Pauli errors are either logically equivalent or have distinguishable syndrome. If the weight of the Pauli error E occurred to a codeword is no bigger than τ, E can be identified by the error syndrome $\vec{s}(E)$ obtained from the generator measurements, and be corrected by applying $E^\dagger$ to the codeword.

The above working principle for a stabilizer code assumes that the syndrome measurements are perfect. In practice, every step in a quantum computation, including those in the syndrome measurements, is subject to error. An initial error can lead to a complex overall effect in the circuit. We adhere to the following terminologies and noise model in our discussion.

Definition 3.1 *Location, noise model, and fault [AGP06]*

A circuit consists of a number of time steps and a number of qubits and is specified by operations to the qubits in each time step. The operations can be single qubit state preparation, 1- or 2-qubit gates, or single qubit measurement. (When nothing happens to a qubit, it goes through the 1-qubit gate of identity.) A location *is labeled by a time step and the index (or indices) of a qubit (or pair of qubits) involved in an operation.*

We consider the circuit-level noise in which every location is followed by depolarizing noise*: every one-qubit operation is followed by a single-qubit Pauli error I, X, Y, or Z, and every two-qubit operation is followed by a two-qubit Pauli error of the form $P_1 \otimes P_2$ where*

$P_1, P_2 \in \{I, X, Y, Z\}$.

A fault *is specified by a location and a nontrivial 1- or 2-qubit Pauli operation which describes a deviation from the ideal operation on the location. This Pauli operation is called the "Pauli error due to the fault".*

A small number of faults during the measurements can lead to an error of weight higher than τ which may cause the EC protocol to fail. To see this, first, we describe how an error of weight 1 or 2 arising from a faulty operation can propagate through a circuit and become an error of higher weight. Specifically, a Hadamard gate and a CNOT gate will transform X-type and Z-type errors as follows:

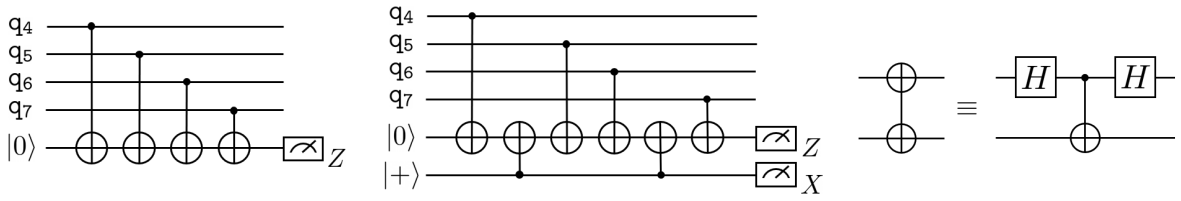
$$\begin{array}{rll} H: & X \mapsto Z, & Z \mapsto X, \\ \text{CNOT}: & XI \mapsto XX, & ZI \mapsto ZI, \\ & IX \mapsto IX, & IZ \mapsto ZZ. \end{array}$$

To see how errors from a few faults can cause an EC protocol to fail, let us consider a circuit for measuring a stabilizer generator of the Steane code as an example. Recall that the $[\![7, 1, 3]\!]$ Steane code [Ste96] is a stabilizer code which can be described by the following generators:

$$\begin{array}{ll} g_1^x : I\ I\ I\ X\ X\ X\ X, & g_1^z : I\ I\ I\ Z\ Z\ Z\ Z, \\ g_2^x : I\ X\ X\ I\ I\ X\ X, & g_2^z : I\ Z\ Z\ I\ I\ Z\ Z, \\ g_3^x : X\ I\ X\ I\ X\ I\ X, & g_3^z : Z\ I\ Z\ I\ Z\ I\ Z. \end{array} \tag{3.1}$$

Logical X and logical Z operators of the Steane code are $X^{\otimes 7}M$ and $Z^{\otimes 7}N$ for any stabilizers M, N. The syndrome is a 6-bit string of the form $(\vec{s}_x|\vec{s}_z)$, with the i-th bit being 0 (or 1) if measuring the i-th generator (ordered as g_1^x, g_2^x, g_3^x, then g_1^z, g_2^z, g_3^z) gives $+1$ (or -1) eigenvalue.

Suppose that during the syndrome measurement, all circuits for measuring stabilizer generators are perfect except for a circuit for measuring g_1^z which has at most 1 fault. Consider a circuit for measuring g_1^z and storing the syndrome using one ancilla qubit (called the *syndrome ancilla*) as in Fig. 3.1a. Also, assume that at most one CNOT gate causes either II, IZ, ZI, or ZZ error. Because of error propagation, a Z error occurred to the syndrome ancilla can propagate back to one or more data qubit(s). As a result, we find that possible



(a) (b) (c)

Figure 3.1: (a) An example of non-flag circuit for measuring generator g_1^z of the $[\![7, 1, 3]\!]$ code. Only qubits on which the operator acts are displayed. The measurement result 0 and 1 obtained from the syndrome ancilla correspond to the $+1$ and -1 eigenvalues of g_1^z. (b) An example of flag circuit for measuring g_1^z. The state of the flag ancilla can flip from $|+\rangle$ to $|-\rangle$ if some fault occurs in between two flag CNOT gates. A circuit for measuring X-type generator can be obtained by replacing each CNOT gate with the gate shown in (c).

errors on data qubits arising from at most 1 CNOT fault (up to multiplication of g_1^z) are,

$$I, Z_4, Z_5, Z_6, Z_7, Z_6Z_7. \tag{3.2}$$

A circuit fault may also cause the syndrome bit to flip. In order to obtain the syndrome exactly corresponding to the data error, one can perform full syndrome measurements until the outcomes are repeated two times in a row, then do the error correction using the repeated syndrome. However, note that the Steane code which can correct any error up to weight 1 must be able to correct the following errors as well:

$$I, Z_1, Z_2, Z_3, Z_4, Z_5, Z_6, Z_7. \tag{3.3}$$

Errors Z_1 and Z_6Z_7 have the same syndrome $(0, 0, 1|0, 0, 0)$ but are not logically equivalent, and subsequent syndrome measurements cannot distinguish between these two cases. This means that if a CNOT fault leads to the Z_6Z_7 error, a correction step for the syndrome $(0, 0, 1|0, 0, 0)$ that applies $Z_1^\dagger$ to the data qubits will result in a logical error $Z_1Z_6Z_7$ on the data qubits, causing the EC protocol to fail.

The goal of this work is to design an EC protocol which is *fault tolerant*; that is, we want to make sure that any subsequent error arising from a small number of faults will still be

correctable by the protocol regardless of its weight (the formal definitions of fault tolerance will be discussed in Chapter 4).

One way to solve the error distinguishing issue is to use traditional FTEC schemes such as the ones proposed by Shor [Sho96, DA07], Steane [Ste97, Ste02], or Knill [Kni05]. However, these schemes require a large number of ancillas (as previously mentioned in Chapter 1). An alternative way to solve the problem is to add an additional ancilla qubit in a circuit for measuring g_1^z as shown in Fig. 3.1b. A circuit of this form is called *flag circuit* [CR18c] (in contrast to the circuit in Fig. 3.1a, which is called *non-flag circuit*). The additional ancilla qubit is called *flag ancilla*, which is initially prepared in the state $|+\rangle$. There are two types of CNOT gates in a flag circuit: a *data CNOT* which couples one of the data qubits and the syndrome ancilla, and a *flag CNOT* which couples the flag ancilla and the syndrome ancilla. Whenever a data CNOT in between two flag CNOTs causes either IZ or ZZ error, a Z error will propagate from the syndrome ancilla to the flag ancilla, causing the state of the flag ancilla to flip to $|-\rangle$. In general, a flag circuit may have more than one flag ancilla, and data and flag CNOTs may be arranged in a complicated way so that a certain number of faults can be caught by the flag ancillas.

By using the circuit in Fig. 3.1b for measuring g_1^z, we find that possible errors on the data qubits arising from at most 1 CNOT fault corresponding to each flag measurement outcome are,

$$\begin{aligned} 0: &\quad I, Z_4, Z_5, Z_6, Z_7, \\ 1: &\quad I, Z_4, Z_6Z_7, Z_7, \end{aligned} \tag{3.4}$$

where the outcome 0 and 1 correspond to $|+\rangle$ and $|-\rangle$ states, respectively. We can see that the flag measurement outcome is 1 whenever Z_6Z_7 occurs. In contrast, an input error Z_1 will not flip the state of the flag ancilla, so it always corresponds to the flag measurement outcome 0. Therefore, Z_1 and Z_6Z_7 can be distinguished using the flag measurement outcome, and an appropriate error correction for each case can be applied to correct such an error. The main advantage of the flag technique is that the number of ancillas required for the flag FTEC protocol is relatively small compared to that required for the traditional FTEC protocols (assuming that ancilla preparation and measurement are fast and the ancillas can be reused).

3.2 Distinguishable fault set

For a general stabilizer code which can correct errors up to weight $\tau = \lfloor (d-1)/2 \rfloor$, we would like to construct circuits for syndrome measurement in a way that all possible errors arising from up to t faults (where $t \leq \tau$) can be corrected, and t is as close to τ as possible. Note that these errors include any single-qubit errors and errors arising from any fault in any circuit involved in the syndrome measurement. For simplicity, this work will focus mainly on a stabilizer code in the Calderbank-Shor-Steane (CSS) code family [CS96, Ste96], in which X-type and Z-type errors can be detected and corrected separately.

For a given CSS code, a circuit for measuring Z-type generator will look similar to a circuit in Fig. 3.1a or Fig. 3.1b, except that there will be w data CNOT gates for a Z-type generator of weight w. A circuit can have any number of flag ancillas (or have no flag ancillas). There are several factors that can determine the ability to distinguish possible errors; for example, the number of flag ancillas, the ordering of data and flag CNOT gates, and the choice of generators being used for the syndrome measurement [CR18c]. A circuit for measuring X-type generator is similar to a circuit for measuring Z-type generator, except that each CNOT gate is replaced by the gate displayed in Fig. 3.1c.

For a given t, finding all possible combinations of faults up to t faults can be laborious since there are many circuits involved in the syndrome measurement, and each circuit have many gates. To simplify our analysis, we will first consider the case that there is only one CNOT fault in one of the circuits for measuring Z-type generators (similar to Fig. 3.1a or Fig. 3.1b). Suppose that there are a total of c flag ancillas involved in a single round of full syndrome measurement (counted from all circuits). We define a *flag vector* $\in \mathbb{Z}_2^c$ to be a bitstring wherein each bit is the measurement outcome of each flag ancilla. There are two mathematical objects associated with each fault: a data error arising from the fault, and a flag vector corresponding to the fault.

Recall that a faulty CNOT gate can cause a two-qubit error of the form $P_1 \otimes P_2$ where $P_1, P_2 \in \{I, X, Y, Z\}$. However, there are many cases of a single fault which are equivalent, meaning that they can give rise to the same data error and the same flag vector. We find that all possible cases in which a single fault can lead to a purely Z-type error on the data qubits can be obtained by considering only (1) the cases that a faulty CNOT gate in a circuit for measuring Z-type generator causes IZ error, and (2) the cases that a Z error occurs to any data qubit. This results from the following facts [TCL20]:

1. The case that a faulty CNOT gate causes ZZ error is equivalent to the case that the preceding CNOT gate causes error IZ.

2. The case that a faulty CNOT gate causes XZ, YZ error is equivalent to the case that an X error occurs to a data qubit and a faulty CNOT gate causes IZ or ZZ error.

3. The case that a faulty CNOT gate causes XI, YI, ZI, IX, XX, YX or ZX error can be considered as the case that a single-qubit error occurs to a data qubit since an X error occurred to the syndrome ancilla will not propagate back to any data qubit.

4. The case that a faulty CNOT gate causes IY, XY, YY or ZY error is similar to the case that a faulty CNOT gate causes IZ, XZ, YZ or ZZ error,

5. An ancilla preparation or measurement fault can be considered as the case that X or Z error occurred to an ancilla qubit (either syndrome or flag ancilla).

6. A CSS code can detect and correct X-type and Z-type errors separately, and a single fault in a circuit for measuring X-type generator cannot cause an Z-type error of weight greater than 1 (and vice versa).

Moreover, if X-type and Z-type generators have similar forms and the gate permutations in the measuring circuits are the same, then all possible faults that can lead to X-type errors on the data qubits are of similar form.

If there are many faults during the protocol, the data errors and the flag vectors caused by each fault can be combined [TL21b]. In particular, a fault combination can be defined as follows:

Definition 3.2 *Fault combination*

A fault combination $\Lambda = \{\lambda_1, \lambda_2, \ldots, \lambda_r\}$ *is a set of* r *faults* $\lambda_1, \lambda_2, \cdots, \lambda_r$. *Suppose that the Pauli error due to the fault* λ_i *can propagate through the circuit and lead to* data error E_i *and* flag vector $\vec{f_i}$. *The* combined data error $\mathbf{E}$ *and* cumulative flag vector $\vec{\mathbf{f}}$ *corresponding*

to Λ *are defined as follows:*

$$\mathbf{E} = \prod_{i=1}^{r} E_i, \tag{3.5}$$

$$\vec{\mathbf{f}} = \sum_{i=1}^{r} \vec{f}_i \,. \tag{3.6}$$

(Note that the error syndrome of the combined data error is $\vec{s}(\mathbf{E}) = \sum_{i=1}^{r} \vec{s}(E_i)$.) For example, suppose that a fault combination Λ arises from two faults λ_1 and λ_2 which can lead to data errors E_1 and E_2, and cumulative flag vectors $\vec{f}_1$ and $\vec{f}_2$. Then, the combined data error $\mathbf{E}$ and the cumulative flag vector $\vec{\mathbf{f}}$ of Λ are $\mathbf{E} = E_1 \cdot E_2$ and $\vec{\mathbf{f}} = \vec{f}_1 + \vec{f}_2 \pmod 2$.

When faults occur in an actual protocol, the faulty locations and the combined data error are not known. In order to determine the combined data error so that the error correction can be done, we will try to measure the error syndrome of the combined data error, and calculate the cumulative flag vector from the flag measurement results obtained since the beginning of the protocol. These measurements, in turn, are subject to errors. The full syndrome measurements will be performed until the syndromes and the cumulative flag vectors are repeated for a certain number of times (similar to the Shor FTEC scheme); examples of FTEC protocols for the $[\![49, 1, 9]\!]$ concatenated Steane code and a capped color code can be found in Section 5.2 and Section 6.3. (Note that by defining the cumulative flag vector as a sum of flag vectors, we lose the information of the ordering in which each fault occurs. However, we find that fault-tolerant protocols presented in this work can still be constructed without such information.)

As previously explained, error correction can fail if there are different faults that lead to non-equivalent errors but there is no way to distinguish them using their error syndromes or flag measurement results. To avoid this, all possible fault combinations must satisfy some conditions so that they can be distinguished. In particular, for a given set of circuits for measuring stabilizer generators, all possible fault combinations can be found, and their corresponding combined data error and cumulative flag vector can be calculated. Let the fault set $\mathcal{F}_t$ be the set of all possible fault combinations arising from up to t faults. We will be able to distinguish all fault combinations if the fault set satisfies the conditions in

the following definition:

Definition 3.3 *Distinguishable fault set*

Let the fault set $\mathcal{F}_t$ *denote the set of all possible fault combinations arising from up to* t *faults and let* S *be the stabilizer group of the quantum error correcting code used to encode the data. We say that* $\mathcal{F}_t$ *is* distinguishable *if for any pair of fault combinations* $\Lambda_p, \Lambda_q \in \mathcal{F}_t$*, at least one of the following conditions is satisfied:*

1. $\vec{s}(\mathbf{E}_p) \neq \vec{s}(\mathbf{E}_q)$*, or*
2. $\vec{\mathbf{f}}_p \neq \vec{\mathbf{f}}_q$*, or*
3. $\mathbf{E}_p = \mathbf{E}_q \cdot M$ *for some stabilizer* $M \in S$*,*

where $\mathbf{E}_p, \vec{\mathbf{f}}_p$ *correspond to* Λ_p*, and* $\mathbf{E}_q, \vec{\mathbf{f}}_q$ *correspond to* Λ_q*. Otherwise, we say that* $\mathcal{F}_t$ *is* indistinguishable.

An example of a distinguishable fault set with $t = 1$ is the fault set corresponding to Eq. (3.4). In that case, we can see that for any pair of faults, either the syndromes of the data errors or the flag measurement outcomes are different.

The following proposition states the relationship between 'correctable' and 'detectable' faults. This is similar to Proposition 2.1 which states that a stabilizer code of distance d can detect errors up to weight $d-1$ and can correct errors up to weight $\tau = \lfloor (d-1)/2 \rfloor$ [Got97].

Proposition 3.1 *$\mathcal{F}_t$ is distinguishable if and only if a fault combination corresponding to a nontrivial logical operator and the zero cumulative flag vector is not in $\mathcal{F}_{2t}$.*

Proof:
($\Rightarrow$) Let $\Lambda_p, \Lambda_q \in \mathcal{F}_t$ be fault combinations arising from up to t faults, let $\tilde{\Lambda}_r \in \mathcal{F}_{2t}$ be a fault combination arising from up to $2t$ faults, and let S be the stabilizer group. First, observe that for any $\tilde{\Lambda}_r \in \mathcal{F}_{2t}$, there exist $\Lambda_p, \Lambda_q \in \mathcal{F}_t$ such that $\tilde{\Lambda}_r = \Lambda_p \cup \Lambda_q$ (where the union of two fault combinations is similar to the union of two sets). Now suppose that $\mathcal{F}_t$ is distinguishable. Then, for each pair of Λ_p, Λ_q in $\mathcal{F}_t$, $\vec{s}(\mathbf{E}_p) \neq \vec{s}(\mathbf{E}_q)$ or $\vec{\mathbf{f}}_p \neq \vec{\mathbf{f}}_q$ or $\mathbf{E}_p = \mathbf{E}_q \cdot M$

for some stabilizer $M \in S$. We find that $\tilde{\Lambda}_r = \Lambda_p \cup \Lambda_q$ corresponds to $\mathbf{E}_r$ and $\vec{\mathbf{f}}_r$ such that $\vec{s}(\mathbf{E}_r) = \vec{s}(\mathbf{E}_p) + \vec{s}(\mathbf{E}_q) \neq 0$ or $\vec{\mathbf{f}}_r = \vec{\mathbf{f}}_p + \vec{\mathbf{f}}_q \neq 0$ or $\mathbf{E}_r = \mathbf{E}_p \cdot \mathbf{E}_q = M$ for some stabilizer $M \in S$. This is true for any $\tilde{\Lambda}_r \in \mathcal{F}_{2t}$, meaning that there is no fault combination in $\mathcal{F}_{2t}$ which corresponds to a nontrivial logical operator and the zero cumulative flag vector.

($\Leftarrow$) As before, we know that for any $\tilde{\Lambda}_r \in \mathcal{F}_{2t}$, there exist $\Lambda_p, \Lambda_q \in \mathcal{F}_t$ such that $\tilde{\Lambda}_r = \Lambda_p \cup \Lambda_q$. Now suppose that $\mathcal{F}_t$ is indistinguishable. Then, there are some pair of Λ_p, Λ_q in $\mathcal{F}_t$ such that $\vec{s}(\mathbf{E}_p) = \vec{s}(\mathbf{E}_q)$, $\vec{\mathbf{f}}_p = \vec{\mathbf{f}}_q$, and $\mathbf{E}_p \cdot \mathbf{E}_q$ is not a stabilizer in S. For such pair, we find that $\tilde{\Lambda}_r = \Lambda_p \cup \Lambda_q$ corresponds to $\mathbf{E}_r$ and $\vec{\mathbf{f}}_r$ such that $\vec{s}(\mathbf{E}_r) = \vec{s}(\mathbf{E}_p) + \vec{s}(\mathbf{E}_q) = 0$, $\vec{\mathbf{f}}_r = \vec{\mathbf{f}}_p + \vec{\mathbf{f}}_q = 0$, and $\mathbf{E}_r = \mathbf{E}_p \cdot \mathbf{E}_q$ is not a stabilizer in S. Therefore, there is a fault combination corresponding to a nontrivial logical operator and the zero cumulative flag vector in $\mathcal{F}_{2t}$. □

Finding a circuit configuration which gives a distinguishable fault set is one of the main goals of the protocol developments in Chapters 5 and 6. For a given set of circuits for measuring generators of a stabilizer code, if the fault set is distinguishable, an FTEC protocol for such a code can be constructed. However, we will defer the proof of this claim until Sections 6.3.1 and 6.4.

3.3 Finding equivalent errors using error weight parities

One common theme of the works presented in Chapters 5 and 6 is to find a good combination of stabilizer code and a set of circuits for measuring the code generators in which the corresponding fault set is distinguishable. As we see in Definition 3.3, whether each pair of fault combinations can be distinguished depends on the syndrome of the combined data error and the cumulative flag vector corresponding to each fault combination, and these features heavily depend on the structure of the circuits. However, we should note that there is no need to distinguish a pair of fault combinations whose combined data errors are logically equivalent. Therefore, if the circuits for a particular code are designed in a way that large portions of fault combinations can give equivalent errors, the fault set arising from the circuits will be more likely distinguishable.

For a general stabilizer code, it is not obvious to see whether two Pauli errors with the

same syndrome are logically equivalent or off by a multiplication of some nontrivial logical operator. Fortunately, for some CSS codes, it is possible to check whether two Pauli errors with the same syndrome are logically equivalent by comparing their weight parities, defined as follows:

Definition 3.4 *The* weight parity *of Pauli error E, denoted by* $\mathrm{wp}(E)$*, is 0 if E has even weight, or is 1 if E has odd weight.*

In [TL21b], we prove that for the $[\![7, 1, 3]\!]$ Steane code and the $[\![23, 1, 7]\!]$ Golay code, errors with the same syndrome and weight parity are logically equivalent. The idea is further extended in [TL21a] to a family of $[\![n, k, d]\!]$ CSS codes in which n is odd, k is 1, all stabilizer generators have even weight, and $X^{\otimes n}$ and $Z^{\otimes n}$ are logical X and logical Z operators, respectively. The lemma presented in [TL21a] (which is adapted from Claim 1 in [TL21b]) is as follows:

Lemma 3.1 *Let C be an $[\![n, k, d]\!]$ CSS code in which n is odd, $k = 1$, all stabilizer generators have even weight, and $X^{\otimes n}$ and $Z^{\otimes n}$ are logical X and logical Z operators. Also, let S_x, S_z be subgroups generated by X-type and Z-type generators of C, respectively. Suppose E_1, E_2 are Pauli errors of any weights with the same syndrome.*

1. *If E_1, E_2 are Z-type errors, then E_1, E_2 have the same weight parity if and only if $E_1 = E_2 \cdot M$ for some $M \in S_z$.*
2. *If E_1, E_2 are X-type errors, then E_1, E_2 have the same weight parity if and only if $E_1 = E_2 \cdot M$ for some $M \in S_x$.*

Proof:
We focus on the first case when E_1, E_2 are Z-type errors and omit the similar proof for the second case. First, recall that the normalizer group of the stabilizer group (the subgroup of Pauli operators that commute with all stabilizers) is generated by the stabilizer generators together with the logical X and the logical Z. Since E_1, E_2 have the same syndrome, their product $L = E_1 E_2$ has trivial syndrome, and is thus in the normalizer group. So we can express L as a product of the stabilizer generators and the logical X and Z's. But there is no X-type factors (since L is Z-type). Therefore, $L = M(Z^{\otimes n})^a$ where $M \in S_z$ and $a \in \{0, 1\}$.

Next, we make an observation. Let M_1, M_2 be two Z-type operators, with respective weights w_1, w_2. The weight of the product $M_1 M_2$ is $w_1 + w_2 - 2c$, where c is the number of qubits supported on both M_1 and M_2. From this observation, and the fact that all generators have even weight, we know M has even weight. Also, from the same observation, and the hypothesis that E_1, E_2 have the same weight parity, L also has even weight. If $a = 1$, $L = M(Z^{\otimes n})^a$ will contradict the observation, so, $a = 0$, $L = M$, and $E_1 E_2 = M \in S_z$ as claimed. On the other hand, if we assume that E_1, E_2 have different weight parities, then L has odd weight and $a = 1$, which implies that $E_1 E_2 = M(Z^{\otimes n})$ for some $M \in S_z$.

□

Lemma 3.1 provides a possible way to perform error correction using syndromes and weight parities, and it can help us find a good code and circuits in which the fault set is distinguishable. In particular, for a given CSS code satisfying Lemma 3.1, if the error syndrome and the weight parity of the data error can be measured perfectly, then an EC operator which can map the erroneous codeword back to the original codeword can be determined without failure. The EC operator can be any Pauli operator that has the same syndrome and the same weight parity as those of the data error. For example, if the $[\![7, 1, 3]\!]$ Steane code is being used and the data error is $Z_1 Z_3 Z_6 Z_7$, we can use $Z_1 Z_2$ as an EC operator to do the error correction.

However, measuring the weight parity should not be done directly on the codeword; measuring weight parities of Z-type and X-type errors correspond to measuring $X^{\otimes n}$ and $Z^{\otimes n}$, respectively, which may destroy the superposition of the encoded state. Moreover, $X^{\otimes n}$ and $Z^{\otimes n}$ do not commute. Fortunately, if we have two codes C_A, C_B such that C_A is a subcode of C_B, then the weight parity of an error on C_A can sometimes be determined by the measurement results of the generators of C_B.

In [TL21b] in which an FTEC protocol for a $[\![49, 1, 9]\!]$ concatenated Steane code is developed, we consider the case that C_A is the $[\![7, 1, 3]\!]$ Steane code and C_B is the $[\![49, 1, 9]\!]$ concatenated code. The error weight parities for each subblock of the 7-qubit code are determined by the syndrome obtained from the measurement of the $[\![49, 1, 9]\!]$ code generators. Afterwards, error correction is performed blockwisely using the weight parity of the error in each subblock, together with the syndrome obtained from the measurement of the 7-qubit code generators for such a subblock. We also find some evidences suggesting that a similar error correction technique may be applicable to other concatenated codes such as the concatenated Golay code and a concatenated Steane code with more than 2 levels of

concatenation. The technical details of [TL21b] can be found in Chapter 5.

In [TL21a], a different approach has been used; we consider a case that C_B is not constructed from concatenating C_A's. We first consider the 3D color code of distance 3 in the form that has the 2D color code of distance 3 as a subcode, and try to construct circuits for measuring generators of the 3D color code which give a distinguishable fault set. Afterwards, we extend the construction ideas to a capped color code of distance d which has the 2D color code of distance d on the center plane as a subcode. The work uses the fact that the biggest stabilizer generators of a capped color code (called `cap` generators) cover all qubits on the center plane. Therefore, any error occurred on the center plane can always be corrected using the syndrome obtained from measuring generators on the center plane (the generators of the 2D color code), together with the weight parity obtained from measuring the `cap` generator of X or Z type. Our schemes for capped color codes in H form of distance 3, 5, and 7 require only 1, 1, and 2 ancillas, respectively. In addition, we prove a theorem which can be helpful for finding the circuits for a capped color code of higher distance. The technical details of [TL21a] are discussed in Chapter 6.

Chapter 4

Redefining Fault Tolerance

In Chapter 3, we already discussed how flags and weight parities could be used in an FTEC protocol, and the notion of distinguishable fault set was introduced in Definition 3.3. When a fault set $\mathcal{F}_t$ is distinguishable, all possible errors of any weight arising from up to t faults can be accurately identified (up to a multiplication of some stabilizer) using their syndromes and cumulative flag vectors obtained from perfect subsequent syndrome measurements. Therefore, all possible errors arising from up to t faults are correctable. However, one should be aware that faults can happen anywhere in an EC protocol, including the locations in the subsequent syndrome measurements. Our goal is to construct a protocol which is *fault tolerant*; vaguely speaking, if an input state to an EC protocol has some error, we want to make sure that the output state is the same logical state as the input, and if output state has any error, the error must not be 'too large'.

What does it mean for the output error to be not too large? The general idea is that if an output error of a single round of the protocol becomes an input error of the next round of the protocol, the error should still be correctable by the latter round. In [AGP06], the authors proposed that the weight of the output error from a fault-tolerant protocol should be no more than the number of total faults occurred during the protocol. However, it should be noted that for an $[\![n, k, d]\!]$ code which can correct errors up to weight $\tau = \lfloor (d-1)/2 \rfloor$ and is not a perfect code (or not a perfect CSS code)[1], the idea of correctable errors can

[1] A perfect code is a quantum code which saturates the quantum Hamming bound; i.e., there is a one-to-one correspondence between correctable errors and all possible syndromes [Got96, Got97]. A perfect CSS code is defined similarly, except that the syndromes of X-type and Z-type errors are considered separately.

be extended to some errors of weight more than τ. For example, if the code being used is a non-perfect code of distance 3, there will be some error E of weight more than 1 whose syndrome $\vec{s}(E)$ is different from those of errors of weight 1. If no other error E' has the same syndrome as E in the set of correctable errors, then in this case, E is also correctable in the sense that we can perform an error correction by applying $E^\dagger$ every time we obtained the syndrome $\vec{s}(E)$.

In this chapter, we will 'refine' the idea of high-weight error correction and 'redefine' fault tolerance using the notion of distinguishable fault set. In Section 4.1, conventional definitions of fault-tolerant gadgets proposed by Aliferis, Gottesman, and Preskill [AGP06] will be described. Afterwards, the definitions will be revised using the notion of distinguishable fault set in Section 4.2. The contents of this chapter follow Section V A of [TL21a].

4.1 Conventional definitions of fault-tolerant gadgets

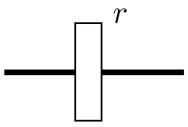
Recall that τ denotes the weight of errors that a stabilizer code can correct, and t denotes the number of faults. We will start by stating the definitions of an r-filter and an ideal decoder from [AGP06], which are the main tools for describing the properties of fault-tolerant gadgets. The definitions are as follows:

Definition 4.1 *r-filter (AGP version)*

Let $T(S)$ be the coding subspace defined by the stabilizer group S. An r-filter is the projector onto the subspace spanned by

$$\left\{ E \left|\bar{\psi}\right\rangle ; \left|\bar{\psi}\right\rangle \in T(S), \text{ the weight of } E \text{ is at most } r \right\}. \tag{4.1}$$

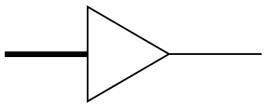
An r-filter in the circuit form is displayed below:



where a thick line represents a block of code.

Definition 4.2 *ideal decoder (AGP version)*

Let $\tau = \lfloor (d-1)/2 \rfloor$ where d is the code distance. An ideal decoder is a gadget which can correct any error of weight up to τ and map an encoded state $\left|\bar{\psi}\right\rangle$ on a code block to the corresponding (unencoded) state $|\psi\rangle$ on a single qubit without any fault. An ideal decoder in the circuit form is displayed below:



where a thick line represents a block of code, and a thin line represents a single qubit.

The intuition behind the definitions of these two gadgets are as follows: If an input state of an r-filter differs from a codeword by an error of weight $\leq r$, then the output state will also differ from the same codeword by an error of weight $\leq r$. However, if the input state has an error of weight $> r$, then the input and output states may correspond to different ideal codewords (i.e., they may be ideally decoded to different unencoded states). An ideal decoder is a gadget which guarantees that the output (unencoded) state and the input (encoded) state will be logically the same whenever the input state has an error of weight no more than τ.

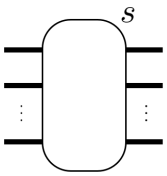
(Note that an r-filter is a linear, completely positive map but it is not trace-preserving; an r-filter cannot be physically implemented. In the definitions of fault-tolerant gadgets to be described, r-filters will be used as mathematical objects to express circuit identities that must be held when the weight of input or output errors and the number of faults are restricted. When each identity holds, both sides of the equation give the same output, including normalization, for the same input state, but the trace of the output might not be one.)

Using the definitions of r-filter and ideal decoder, fault-tolerant gate (FTG) gadget and fault-tolerant error correction (FTEC) gadget can be defined as follows:

Definition 4.3 *Fault-tolerant gate gadget (AGP version)*

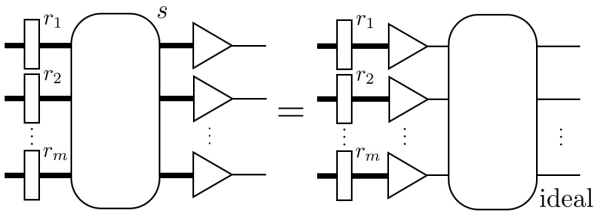
A gate gadget *with s faults simulating an ideal m-qubit gate is represented by the following*

picture:

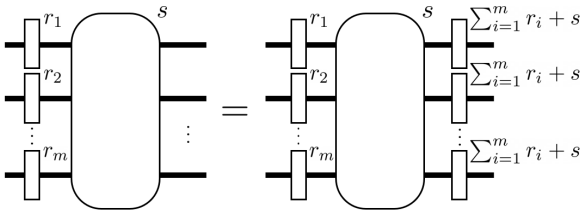


where each thick line represents a block of code. Let $t \leq \lfloor (d-1)/2 \rfloor$. *A gate gadget is* t-fault tolerant *if it satisfies both of the following properties:*

1. *Gate correctness property (GCP): whenever* $\sum_{i=1}^{m} r_i + s \leq t$,



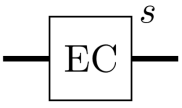
2. *Gate error propagation property (GPP): whenever* $\sum_{i=1}^{m} r_i + s \leq t$,



where the r*-filter and the ideal decoder are as defined in Definition 4.1 and Definition 4.2.*

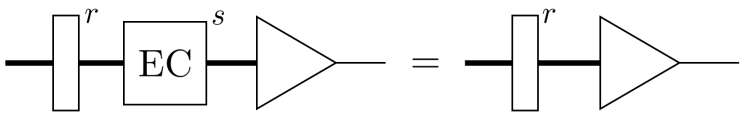
Definition 4.4 *Fault-tolerant error correction gadget (AGP version)*

An error correction gadget *with* s *faults is represented by the following picture:*

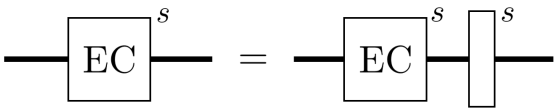


where a thick line represents a block of code. Let $t \leq \lfloor (d-1)/2 \rfloor$. *An error correction gadget is* t-fault tolerant *if it satisfies both of the following properties:*

1. *Error correction correctness property (ECCP): whenever* $r + s \leq t$,



2. *Error correction recovery property (ECRP): whenever* $s \leq t$,



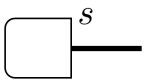
where the r*-filter and the ideal decoder are as defined in Definition 4.1 and Definition 4.2.*

When an FTG gadget satisfies both properties in Definition 4.3, it is guaranteed that whenever the weight of the input error plus the number of faults is no more than t, (1) the operation of an FTG gadget on an encoded state will be similar to the operation of its corresponding quantum gate on an unencoded state, and (2) an output state of an FTG gadget will have an error of weight no more than t (which is also $\leq \tau$). Meanwhile, the two properties of an FTEC gadget in Definition 4.4 guarantee that (1) the output and the input states of an FTEC gadget are logically the same whenever the weight of the input error plus the number of faults is no more than t, and (2) the weight of the output error of an FTEC gadget is no more than the number of faults whenever the number of faults is at most t, regardless of the weight of the input error.

Fault-tolerant state preparation (FTP) gadget and fault-tolerant (non-destructive) measurement (FTM) gadget, which are special cases of FTG gadget, can be defined as follows:

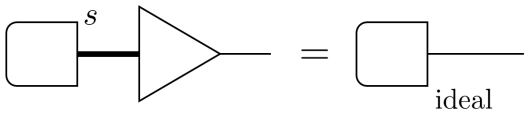
Definition 4.5 *Fault-tolerant state preparation gadget (AGP version)*

A state preparation gadget *with s faults is represented by the following picture:*

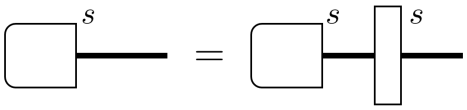


where a thick line represents a block of code. Let $t \leq \lfloor (d-1)/2 \rfloor$. A state preparation gadget is t-fault tolerant *if it satisfies both of the following properties:*

1. *Preparation correctness property (PCP): whenever $s \leq t$,*



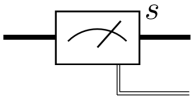
2. *Preparation error propagation property (PPP): whenever $s \leq t$,*



where the r-filter and the ideal decoder are defined as in Definition 4.1 and Definition 4.2.

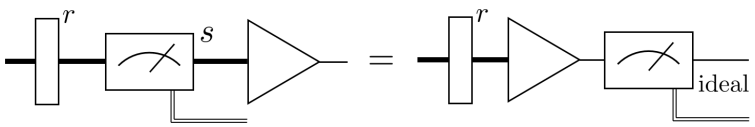
Definition 4.6 *Fault-tolerant (non-destructive) measurement gadget (AGP version)*

A (non-destructive) measurement gadget *with s faults is represented by the following picture:*

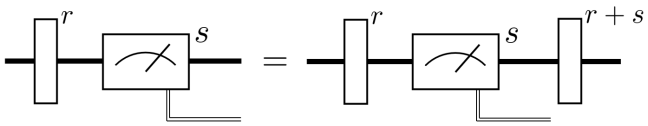


where a thick line represents a block of code. Let $t \leq \lfloor (d-1)/2 \rfloor$. A (non-destructive) measurement gadget is t-fault tolerant *if it satisfies both of the following properties:*

1. *Measurement correctness property (MCP): whenever* $r + s \leq t$,



2. *Measurement error propagation property (MPP): whenever* $r + s \leq t$,



where the r-filter and the ideal decoder are defined as in Definition 4.1 and Definition 4.2.

The meanings of the properties of FTP and FTM gadgets are similar to the meanings of the properties of an FTG gadgets as previously explained.

From Definitions 4.3 to 4.6, we can see that an action of a fault-tolerant gadget is guaranteed in the circumstance that the weight of the input error r and the number of faults occurred in the gadget s satisfy some condition. Now, a question arises: what will happen if the input error has weight greater than $\tau = \lfloor (d-1)/2 \rfloor$, which is the weight of errors that a code can correct? By Definition 3.3, we know that if a fault set $\mathcal{F}_t$ is distinguishable, possible errors arising from up to t faults in an EC protocol (where $t \leq \lfloor (d-1)/2 \rfloor$) can be distinguished using their corresponding syndromes or cumulative flag vectors, regardless of the error weights. Would it be more natural if the definitions of fault-tolerant gadgets depend on the *number of faults* related to an input error, instead of the *weight* of an input error? In the next section, we will try to modify the definitions of fault-tolerant gadgets and rewrite them using the notion of distinguishable fault set.

4.2 Revised definitions of fault-tolerant gadgets

In order to modify the definitions of fault-tolerant gadgets proposed in [AGP06], first, let us define distinguishable error set as follows:

Definition 4.7 *Distinguishable error set*

Let $\mathcal{F}_r$ be a distinguishable fault set, and let $\mathcal{F}_r|_{\vec{\mathbf{f}}=0}$ be a subset of $\mathcal{F}_r$ defined as follows:

$$\mathcal{F}_r|_{\vec{\mathbf{f}}=0} = \{\Lambda \in \mathcal{F}_r;\ \vec{\mathbf{f}} \textit{ of } \Lambda \textit{ is zero}\}. \tag{4.2}$$

A distinguishable error set $\mathcal{E}_r$ *corresponding to $\mathcal{F}_r$ is,*

$$\mathcal{E}_r = \{\mathbf{E} \textit{ of } \Lambda \in \mathcal{F}_r|_{\vec{\mathbf{f}}=0}\}. \tag{4.3}$$

If $\mathcal{F}_r$ is distinguishable, $\mathcal{F}_r|_{\vec{\mathbf{f}}=0}$ is also distinguishable since all pairs of fault combinations in $\mathcal{F}_r|_{\vec{\mathbf{f}}=0}$ also satisfy the conditions in Definition 3.3. Moreover, because all fault combinations in $\mathcal{F}_r|_{\vec{\mathbf{f}}=0}$ correspond to the zero cumulative flag vector, we find that for any pair of errors in $\mathcal{E}_r$, the errors either have different syndromes or are logically equivalent (up to a multiplication of a stabilizer). For this reason, we can safely say that $\mathcal{E}_r$ is a set of correctable errors.

Because the set of correctable errors is now expanded, the definitions of r-filter and ideal decoder can be revised as follows:

Definition 4.8 *r-filter (revised version)*

Let $T(S)$ be the coding subspace defined by the stabilizer group S, and let $\mathcal{E}_r$ be the distinguishable error set corresponding to a distinguishable fault set $\mathcal{F}_r$. An r-filter is the projector onto subspace spanned by

$$\left\{E\left|\bar{\psi}\right\rangle;\left|\bar{\psi}\right\rangle \in T(S), E \in \mathcal{E}_r\right\}. \tag{4.4}$$

An r-filter in the circuit form is similar to the one illustrated in Definition 4.1.

Definition 4.9 *ideal decoder (revised version)*

Let $\mathcal{E}_t$ be the distinguishable error set corresponding to a distinguishable fault set $\mathcal{F}_t$, where $t \leq \lfloor (d-1)/2 \rfloor$ and d is the code distance. An ideal decoder is a gadget which can correct any error in $\mathcal{E}_t$ and map an encoded state $\left|\bar{\psi}\right\rangle$ on a code block to the corresponding (unencoded) state $|\psi\rangle$ on a single qubit without any faults. An ideal decoder in the circuit form is similar to the one illustrated in Definition 4.2.

Using the revised definitions of r-filter and ideal decoder, fault-tolerant gadgets can be defined as follows:

Definition 4.10 *Fault-tolerant gadgets (revised version)*

Let $t \leq \lfloor (d-1)/2 \rfloor$. Fault-tolerant gadgets are defined as follows:

1. *A* gate gadget *is* t-fault tolerant *if it satisfies both of the properties in Definition 4.3, except that r-filter and ideal decoder are defined as in Definition 4.8 and Definition 4.9.*

2. *An* error correction gadget *is* t-fault tolerant *if it satisfies both of the properties in Definition 4.4, except that r-filter and ideal decoder are defined as in Definition 4.8 and Definition 4.9.*

3. *A* state preparation gadget *is* t-fault tolerant *if it satisfies both of the properties in Definition 4.5, except that r-filter and ideal decoder are defined as in Definition 4.8 and Definition 4.9.*

4. *A* (non-destructive) measurement gadget *is* t-fault tolerant *if it satisfies both of the properties in Definition 4.6, except that r-filter and ideal decoder are defined as in Definition 4.8 and Definition 4.9.*

The revised definitions of fault-tolerant gadgets in the circuit form may look very similar to the old definitions proposed in [AGP06], but the meanings are different: the conditions in the revised definitions depend on the number of faults which can cause an input or an output error, instead of the weight of an input or an output error. Roughly speaking, this means that (1) a fault-tolerant gadget is allowed to produce an output error of weight greater than τ (where $\tau = \lfloor (d-1)/2 \rfloor$), and (2) a fault-tolerant gadget can work perfectly even though the input error has weight greater than τ, as long as the input or the output error is similar to an error caused by no more than $t \leq \tau$ faults. Because the revised definitions of r-filter and ideal decoder are more general than the old definitions, we expect that a gadget that satisfies one of the old definitions of fault-tolerant gadgets (Definitions 4.3 to 4.6) will also satisfy the new definitions in Definition 4.10. Note that the revised definitions are based on the fact that a fault set relevant to a gadget is distinguishable, that is, whether the gadgets are fault tolerant depends on the way they are designed.

In a special case where the code being used is a CSS code and possible X-type and Z-type errors have the same form, the definition of distinguishable error set can be further extended as follows:

Definition 4.11 *Distinguishable error set (for a special family of CSS codes)*

Let $\mathcal{F}_r$ be a distinguishable fault set, and let $\mathcal{F}_r|_{\vec{\mathbf{f}}=0}$ be a subset of $\mathcal{F}_r$ defined as follows:

$$\mathcal{F}_r|_{\vec{\mathbf{f}}=0} = \{\Lambda \in \mathcal{F}_r;\ \vec{\mathbf{f}} \textit{ of } \Lambda \textit{ is zero}\}. \tag{4.5}$$

A distinguishable-X error set $\mathcal{E}_r^x$ *and a* distinguishable-Z error set $\mathcal{E}_r^z$ *corresponding to $\mathcal{F}_r$ are,*

$$\mathcal{E}_r^x = \{\mathbf{E} \textit{ of } \Lambda \in \mathcal{F}_r|_{\vec{\mathbf{f}}=0}; \mathbf{E} \textit{ is an } X\textit{-type error}\}, \tag{4.6}$$
$$\mathcal{E}_r^z = \{\mathbf{E} \textit{ of } \Lambda \in \mathcal{F}_r|_{\vec{\mathbf{f}}=0}; \mathbf{E} \textit{ is a } Z\textit{-type error}\}. \tag{4.7}$$

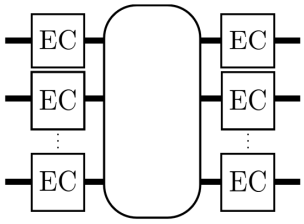
For a CSS code in which the elements of $\mathcal{E}_r^x$ and $\mathcal{E}_r^z$ have a similar form, a distinguishable error set $\mathcal{E}_r$ *corresponding to $\mathcal{F}_r$ is defined as follows:*

$$\mathcal{E}_r = \{E_x \cdot E_z; E_x \in \mathcal{E}_r^x, E_z \in \mathcal{E}_r^z\}. \tag{4.8}$$

Since a CSS code can detect and correct X-type and Z-type errors separately, here we modify the definition of distinguishable error set for a CSS code in which $\mathcal{E}_r^x$ and $\mathcal{E}_r^z$ are in the same form so that more Y-type errors are included in $\mathcal{E}_r$. For example, suppose that $t = 2$, each of $XXXX$ and $ZZZZ$ can be caused by 2 faults, and $YYYY$ can be caused by 4 faults. By the old definition (Definition 4.7), we will say that $XXXX$ and $ZZZZ$ are in $\mathcal{E}_2$, and $YYYY$ is in $\mathcal{E}_4$ but not in $\mathcal{E}_2$. In contrast, by Definition 4.11, we will say that $XXXX$, $YYYY$, and $ZZZZ$ are all in $\mathcal{E}_2$. This modification will give more flexibility when developing a fault-tolerant gadget for this special kind of CSS codes, e.g., a transversal S gate which produces an output error $YYYY$ from an input error $XXXX$ still satisfies the properties in Definition 4.10 when a distinguishable fault set is defined as in Definition 4.11.

When performing a fault-tolerant quantum computation, FTEC gadgets will be used repeatedly in order to reduce the error accumulation during the computation. Normally,

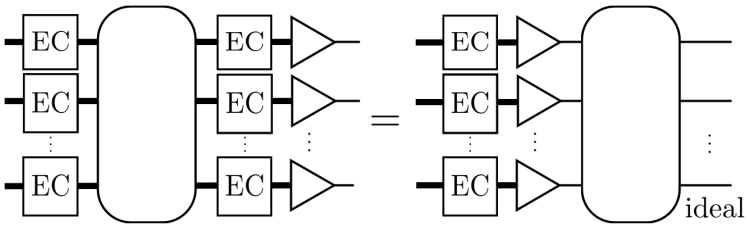
FTEC gadgets will be placed before and after other gadgets (FTG, FTP, or FTM gadgets). A group of gadgets including an FTG gadget, leading EC gadgets (the FTEC gadgets before the FTG gadget), and trailing EC gadgets (FTEC gadgets after the FTG gadget) as shown below is called an *extended rectangle at level 1* or *1-exRec*:



(A 1-exRec of an FTP or FTM gadget is defined similarly to a 1-exRec of an FTG gadget, except that there is no leading gadget in an FTP gadget.) We say that a 1-exRec is *good* if the total number of faults in a 1-exRec is no more than t. Using the revised definitions of fault-tolerant gadgets in Definition 4.10, a revised version of the exRec-Cor lemma at level 1, originally proposed in [AGP06], can be obtained:

Lemma 4.1 *ExRec-Cor lemma at level 1 (revised version)*

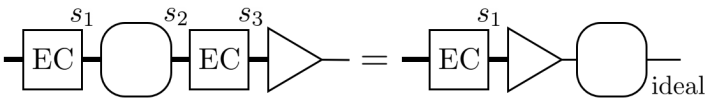
Suppose that all gadgets are t-fault tolerant according to Definition 4.10. If a 1-exRec is good *(i.e., a 1-exRec has no more than t faults), then the 1-exRec is* correct*; that is, the following condition is satisfied:*



where the r-filter and the ideal decoder are defined as in Definitions 4.8 and 4.9.

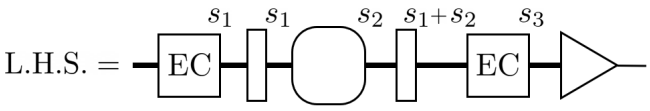
Proof:
Here we will focus only on the case that a gate gadget simulates a single-qubit gate. The proofs for the case of multiple-qubit gate and other gadgets are similar. Suppose that the leading EC gadget, the gate gadget, and the trailing EC gadget in an exRec have s_1, s_2,

and s_3 faults where $s_1 + s_2 + s_3 \leq t$. We will show that the following equation holds:

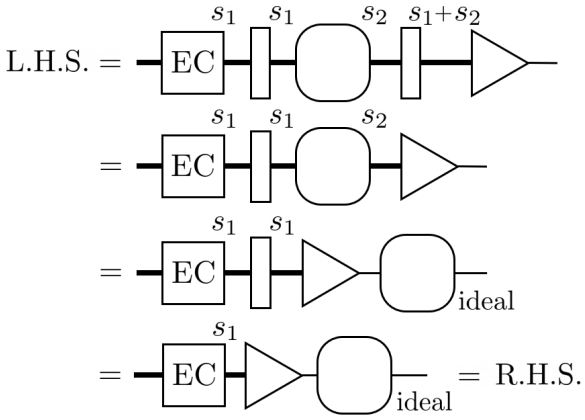


(4.9)

Because the gate gadget satisfies GPP and the EC gadgets satisfy ECRP, the left-hand side of Eq. (4.9) is



Using GCP, ECCP, and the fact that an ideal decoder can correct any error in $\mathcal{E}_t$, we obtain the following:



□

(Note that both sides of the equation in Lemma 4.1 are trace-preserving, completely positive maps, even though r-filters introduced during the proof are not trace-preserving. This is possible since the total number of faults in a 1-exRec is restricted and all gadgets satisfy Definition 4.10.)

The revised version of the exRec-cor lemma developed in this section is very similar to the original version in [AGP06], even though the r-filter, the ideal decoder, and the fault-tolerant gadgets are redefined. The exRec-Cor lemma is one of the main ingredients for

the proofs of other lemmas and theorems in [AGP06]. As a result, other lemmas and theorems developed in [AGP06] are also applicable to our case, including their version of the *threshold theorem* (the proofs of revised versions of the lemmas and theorems are similar to the proofs presented in [AGP06], except that Lemma 4.1 is used instead of the original exRec-Cor lemma). This means that fault-tolerant gadgets satisfying Definition 4.10 can be used to simulate any quantum circuit, and the logical error rate can be made arbitrarily small if the physical error rate is below some constant threshold value. The main advantage of the revised definitions of fault-tolerant gadgets over the conventional definitions is that high-weight errors are allowed as long as they arise from a small number of faults. These revised definitions can give us more flexibility when developing fault-tolerant protocols.

Chapter 5

Fault-tolerant Error Correction for the 49-qubit Concatenated Steane Code

By simulating a quantum circuit using fault-tolerant gadgets defined in Section 4.1 or Section 4.2, it is possible to achieve an arbitrarily low error rate through code concatenation. However, doing so can be difficult in practice because very large overhead (such as gates and ancillas) is required if one desires to achieve a very small error rate. Traditional FTEC schemes require substantial number of ancillas. For example, Shor [Sho96, DA07], Steane [Ste97, Ste02], and Knill [Kni05] EC schemes applied to the $[\![7, 1, 3]\!]$ Steane code require 4, 7, and 14 ancillas, respectively. Fortunately, there are several schemes for the Steane code which require fewer; the FTEC scheme due to Yoder and Kim [YK17] and the flag FTEC scheme by Chao and Reichardt [CR18c] for the Steane code require only 2 ancillas. Meanwhile, the FTEC scheme for the Steane code proposed by Reichardt that extracts several syndrome bits at once requires no ancillas, provided that there are at least two code blocks (so at least 14 qubits are required in total) [Rei20].

In order to extend an FTEC scheme designed for a single-level code to a concatenated code, the scheme must be modified accordingly. One way to do so is replacing all physical qubits with code blocks and replacing all physical gates with corresponding logical gates [AGP06]. For the $[\![7, 1, 3]\!]$ Steane code, each qubit (including each ancilla qubit) required in

an FTEC scheme will become a block of 7 physical qubits in the modified scheme. Following this modification, the schemes in [YK17, CR18c] applied to the $[\![49, 1, 9]\!]$ concatenated Steane code will require 63 qubits in total. Meanwhile, the scheme in [Rei20] requires 98 qubits in total, encoding 2 logical qubits. Note that the maximum weight for the stabilizer generators increases quickly with concatenation. These difficulties motivate our main question in [TL21b]: how to reduce the number of ancillas required for an FTEC scheme for a concatenated code?

The ideas behind the FTEC scheme for the $[\![49, 1, 9]\!]$ concatenated Steane code proposed in [TL21b] will be elaborated in this chapter. In Section 5.1, we observe the equivalence between errors of any weight with the same syndrome and weight parity for the $[\![7, 1, 3]\!]$ Steane code (whose generalized version is given in Lemma 3.1), and introduce the weight parity error correction (WPEC) technique. In Section 5.2, we provide sufficient conditions for WPEC, then provide syndrome extraction circuits and an FTEC protocol for the $[\![49, 1, 9]\!]$ concatenated Steane code using only two ancilla qubits. In Section 5.3, WPEC is extended to the $[\![23, 1, 7]\!]$ Golay code and concatenated Steane codes with more than 2 levels of concatenation.

5.1 Weight parity error correction for the Steane code

The $[\![7, 1, 3]\!]$ Steane code [Ste96] is a quantum error correcting code that encodes 1 logical qubit into 7 physical qubits and can correct any error on up to 1 qubit. It has several desirable properties for fault-tolerant quantum computation, e.g., logical Clifford operations are transversal [Sho96]. The Steane code is a code in the Calderbank-Shor-Steane (CSS) code family [CS96, Ste96] where X-type and Z-type errors can be detected and corrected separately. The Steane code in the stabilizer formalism can be constructed from the parity check matrix of the classical $[7, 4, 3]$ Hamming code through the CSS construction [Got97]. In addition, it is known that any classical Hamming code can be rearranged into a cyclic code, a binary linear code in which any cyclic shift of a codeword is also a codeword [MS77]. We can describe the Steane code in cyclic form with the following stabilizer generators:

$$\begin{array}{ll} g_1^x : X\ I\ X\ X\ X\ I\ \ I, & g_1^z : Z\ I\ Z\ Z\ Z\ I\ I, \\ g_2^x : I\ X\ I\ X\ X\ X\ I, & g_2^z : I\ Z\ I\ Z\ Z\ Z\ I, \\ g_3^x : I\ \ I\ X\ I\ X\ X\ X, & g_3^z : I\ I\ Z\ I\ Z\ Z\ Z. \end{array} \tag{5.1}$$

(Note that the generators of the Steane code defined in this chapter and those defined in Chapters 2 and 3 are equivalent up to a permutation of qubits.)

The generators of a stabilizer code define not only the codespace, but also the measurements that give rise to the error syndrome. When these measurements are imperfect, different sets of generators for the same code can have different fault-tolerant properties. The use of the Steane code in cyclic form gives some advantages in distinguishing high-weight errors in consecutive form [TCL20] (see the discussion in Section 7.1 for more details). We can choose the logical X and logical Z operators to be $X^{\otimes 7}M$ and $Z^{\otimes 7}N$ for any stabilizer operators M, N. With this convention, we state the following crucial property of the Steane code that goes into our construction.

Fact 5.1 *Let L be any Z-type operator (a tensor product of Is and Zs) defined on 7 qubits. Suppose L commutes with all X-type generators of the $[\![7, 1, 3]\!]$ code. If L has even weight, then it is a logical I; otherwise, if L has odd weight, then it is a logical Z.*

Proof:
Because L is a Z-type operator that commutes with all X-type generators, L is either a stabilizer of Z type or a logical Z operator. Let E_1 and E_2 be Z-type operators with weights w_1 and w_2. Then E_1E_2 is an operator of weight $w_1 + w_2 - 2c$, where c is the number of qubits supported by both E_1 and E_2. Observe that all stabilizer generators of the Steane code have even weight, and a multiplication of two operators with even weight always gives an operator with even weight. Thus, all stabilizers of Z type (which are logical I operators) have even weight. Moreover, a Z-type operator which is a logical Z operator is of the form $Z^{\otimes 7}N$ where N is a stabilizer of Z type. Therefore, all logical Z operators of Z type have odd weight. □

For a Pauli error E on a block of 7 qubits, the syndrome is a 6-bit string denoted by $\vec{s}(E) = (\vec{s}_x|\vec{s}_z)$ where $\vec{s}_x, \vec{s}_z \in \mathbb{Z}_2^3$. The i-th bit of $\vec{s}_x$ (or $\vec{s}_z$) is 0 if E commutes with g_i^x (or g_i^z), and 1 if E anticommutes with g_i^x (g_i^z). If E occurs to a codeword of the Steane code, $\vec{s}(E)$ corresponds to the outcomes of measuring the six generators (0 and 1 correspond to $+1$ and -1 outcomes, respectively). The Steane code is a *perfect CSS code of distance* 3 meaning that for each $\vec{s}_x$, $(\vec{s}_x|000)$ is the syndrome of a *unique* weight-1 Z-type error, which we denote as $E^z_{wt\text{-}1}(\vec{s}_x)$, and similarly each $(000|\vec{s}_z)$ is the syndrome of a unique

X-type error.[1] For CSS codes, the X-type and Z-type error corrections are independent of one another. Furthermore, we focus on CSS codes in which X-type and Z-type generators have the same form, and the same method applies to both types of error correction. So we focus on Z errors for simplicity. Since Z-type errors have trivial $\vec{s}_z$, we focus on $\vec{s}_x$ from now on.

With the above notations, consider the following simple error correction procedure on the Steane code: if the syndrome is $(\vec{s}_x|000)$, do nothing if $\vec{s}_x$ is trivial, apply $E^z_{wt\text{-}1}(\vec{s}_x)$ otherwise. We observe that if the syndrome is caused by a Z-type error, then the procedure outputs the encoded data transformed by a logical I or logical Z. This is because the actual Z-type error combined with the correction remains Z-type and commutes with all of g^x_1, g^x_2, g^x_3, so the conclusion follows from Fact 5.1.

If a codeword is corrupted by an arbitrary Z-type error E, the above procedure always recovers the codeword, but sometimes with an undesirable logical Z error. The technique of weight parity error correction, to be developed next, is a revised procedure that will *always* correct the error E, but it requires knowing whether E has odd or even weight. Measuring the error weight parity should not be done on a single layer of Steane code since it measures a logical operator on the Steane code. Fortunately, the parity information can be safely learned for the constituent blocks when we concatenate the Steane code with itself. We will describe these ideas in detail in the rest of this section, and apply them for fault-tolerant error correction in Section 5.2.

First, we use Fact 5.1 to show that Z-type errors with the same syndrome *and* the same weight parity (whether odd or even) differ by the multiplication of some stabilizer. (Note that the following claim from [TL21b] is later developed into Lemma 3.1, which is the main ingredient of [TL21a].)

Claim 5.1 *Logical equivalence of errors with the same syndrome and weight parity for the* $[\![7,1,3]\!]$ *Steane code*

Let S_z *be the subgroup generated by* Z*-type generators of the* $[\![7,1,3]\!]$ *Steane code. Suppose* E_1, E_2 *are arbitrary* Z*-type errors (of any weights) on the* $[\![7,1,3]\!]$ *code with the same syndrome. Then,* E_1 *and* E_2 *have the same weight parity if and only if* $E_1 = E_2 \cdot M$ *for some stabilizer* $M \in S_z$*.*

[1] This is from the fact that the $[\![7,1,3]\!]$ Steane code can be constructed from the classical $[7,4,3]$ Hamming code which is a *classical perfect code*, a code saturating the classical Hamming bound [MS77].

Proof:
Let w_1, w_2 be the weights of E_1, E_2, respectively. Let $L = E_1 E_2$ (so $E_2 = E_1 L$ as $E_1 = E_1^\dagger$). The weight of L is equal to $w_1 + w_2 - 2c$ where c is the number of qubits supported by both E_1 and E_2. As L commutes with all of g_1^x, g_2^x, g_3^x, from Fact 5.1, L is a logical I (a stabilizer in S_z) if and only if $w_1 + w_2 - 2c$ is even (when E_1 and E_2 have the same weight parity). □

Second, we use Claim 5.1 to provide a method for error correction of Z-type error of arbitrary weight on the Steane code, *if the weight parity of the error is known*:

Definition 5.1 *Weight parity error correction (WPEC) for the* $[\![7, 1, 3]\!]$ *code*

Suppose a Z-type error E occurs to a codeword of the $[\![7, 1, 3]\!]$ code. Let $\vec{s}_x$ and w be the syndrome and the weight of E, $E^z_{wt\text{-}1}(\vec{s}_x)$ be the weight-1 Z-type operator with syndrome $\vec{s}_x$, and $E^z_{wt\text{-}2}(\vec{s}_x)$ be any weight-2 Z-type operator with syndrome $\vec{s}_x$, respectively. The following procedure is called weight parity error correction (WPEC):

1. If $\vec{s}_x$ is trivial, do nothing if w is even, or apply any logical Z if w is odd.
2. If $\vec{s}_x$ is nontrivial, apply $E^z_{wt\text{-}1}(\vec{s}_x)$ if w is odd, or apply $E^z_{wt\text{-}2}(\vec{s}_x)$ if w is even.

WPEC always returns the original codewords because in each case, the error E and the correction operation have the same syndrome and weight parity, so by Claim 5.1, the correction is logically equivalent to E.

WPEC allows us to correct high-weight errors in the Steane code, but we need to know the weight parity of the error. The weight parity of a Z-type error is the outcome of measuring $X^{\otimes 7}$, so learning the weight parity is equivalent to a logical X measurement, which destroys the superposition of the logical state. Fortunately, there is a setting in which the weight parity can be obtained without affecting the encoded data. Consider code concatenation in which each qubit of an error correcting code C_2 is encoded into another quantum error correcting code C_1. If C_1 is chosen to be the Steane code, the weight parity of each code block can potentially be learned from the syndrome of C_2. We will develop WPEC for the concatenated Steane code in the rest of this section and show the advantage in the context of fault tolerance in the next section.

Consider code concatenation using two Steane codes in cyclic form. The resulting code which is a $[\![49, 1, 9]\!]$ code can be described by 48 stabilizer generators. The first group of 42 generators, called 1st-level generators, have the form $g_i^x \otimes I^{\otimes 42}, g_i^z \otimes I^{\otimes 42}, I^{\otimes 7} \otimes g_i^x \otimes I^{\otimes 35}, I^{\otimes 7} \otimes g_i^z \otimes I^{\otimes 35}, \ldots, I^{\otimes 42} \otimes g_i^x, I^{\otimes 42} \otimes g_i^z$ for $i = 1, 2, 3$. The remaining 6 of these generators, called 2nd-level generators, have the form

$$\begin{array}{ll} \tilde{g}_1^x : \mathbf{X\ I\ X X X\ I\ \ I}, & \tilde{g}_1^z : \mathbf{Z\ I\ Z Z Z\ I\ I}, \\ \tilde{g}_2^x : \mathbf{I\ X\ I\ X X X\ I}, & \tilde{g}_2^z : \mathbf{I\ Z\ I\ Z Z Z\ I}, \\ \tilde{g}_3^x : \mathbf{I\ \ I\ X\ I\ X X X}, & \tilde{g}_3^z : \mathbf{I\ I\ Z\ I\ Z Z Z}, \end{array} \tag{5.2}$$

where $\mathbf{I} = I^{\otimes 7}, \mathbf{X} = X^{\otimes 7}$, and $\mathbf{Z} = Z^{\otimes 7}$ (note that $\mathbf{I}$, $\mathbf{X}$, and $\mathbf{Z}$ act as logical operators for the 1st-level Steane code C_1). The logical X and logical Z operators of the $[\![49, 1, 9]\!]$ code can be chosen to be $\bar{X} = X^{\otimes 49} M$ and $\bar{Z} = Z^{\otimes 49} N$ for any stabilizer operators M, N. Relevant parts of the error syndrome corresponding to the 1st-level and the 2nd-level generators will be called 1st-level and 2nd-level syndromes, respectively.

Let us consider error correction on the $[\![49, 1, 9]\!]$ concatenated Steane code and assume for now that error syndromes are reliable (which can be obtained from repetitive measurements). First, consider a simple motivating example, and suppose that a Z-type error E acts nontrivially on at most one subblock of 7-qubit code. In order to perform WPEC, the weight parity of E and the subblock in which E occurs must be known. Suppose that E has nontrivial 1st-level syndrome. The subblock in which E occurs is actually the subblock whose corresponding 1st-level syndrome is nontrivial, while the weight parity of E is a measurement result from a 2nd-level generator which acts nontrivially on that subblock (note that the 2nd-level generator must *act nontrivially on all qubits* in such a subblock, thus a choice of 2nd-level generators is important). Now, suppose that E has trivial 1st-level syndrome. The subblock in which E occurs can no longer be identified by the 1st-level syndrome. Fortunately, since the 2nd-level Steane code (C_2) is a distance-3 code, it can identify if any of the 7 subblocks of $[\![7, 1, 3]\!]$ code (the C_1 subblocks) has a Z-type error logically equivalent to $Z^{\otimes 7}$, thus providing the weight parity for each subblock of $[\![7, 1, 3]\!]$ code. That is, if E has trivial 1st-level syndrome and its weight is odd, the weight parity of E and the subblock in which E occurs can be determined using only the 2nd-level syndrome. (If E has trivial 1st-level syndrome and has even weight, it is a stabilizer and no error correction is needed.)

If the Z-type error is more general and may act on multiple subblocks, the 2nd-level

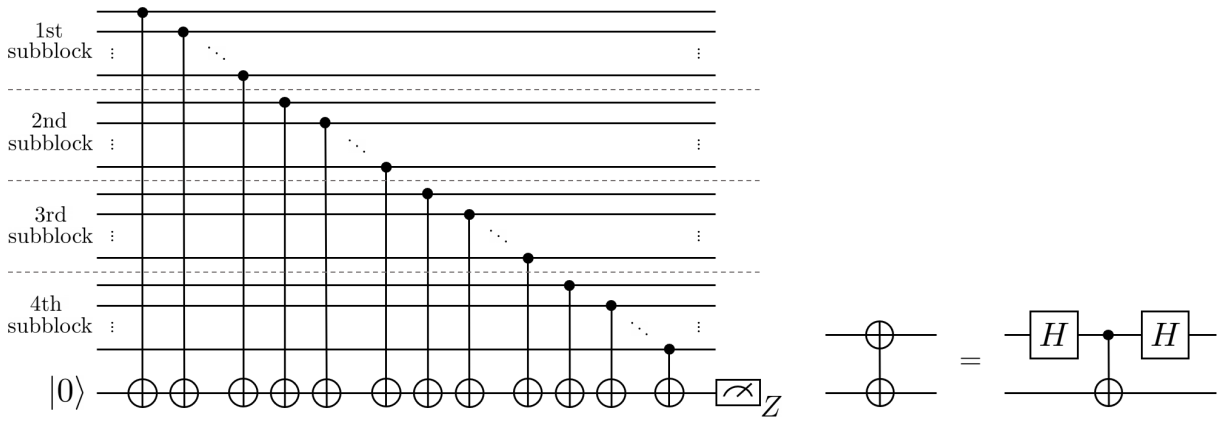


Figure 5.1: (a) An example of circuit for measuring generator $\tilde{g}_1^z$ = **ZIZZZII**. Here we display only the subblocks in which the operator acts nontrivially (the 1st, 2nd, 3rd, and 4th subblocks in the figure correspond to the 1st, 3rd, 4th, and 5th subblocks of $\tilde{g}_1^z$). A circuit for measuring X-type operator such as $\tilde{g}_1^x$ = **XIXXXII** can be obtained by replacing each CNOT gate in (a) with the gate illustrated in (b).

syndrome may not provide the weight parities of the subblocks. Instead, we consider only Z-type errors that arise from a small number of faults in specially designed generator measurements. We will show that for these errors, the weight parity for each subblock can be determined by the 2nd-level syndrome along with the information whether each subblock has trivial 1st-level syndrome or not.

In particular, let *block parity* $\vec{p}_x \in \mathbb{Z}_2^7$ be a bitstring, where each bit is the weight parity of the Z error in one subblock, and 0 and 1 represent even and odd weights, respectively. Also, define the *triviality* of a subblock to be 0 or 1 if the subblock has trivial or non-trivial 1st-level syndrome, and let *block triviality* $\vec{\eta}_x \in \mathbb{Z}_2^7$ be a 7-bit string in which the i-th bit represents the triviality of the i-th subblock. If the block parity can be accurately determined using the 2nd-level syndrome together with the block triviality (we will elaborate how this can be done later), then we can blockwisely perform WPEC as described in Definition 5.1 by using the 1st-level syndrome and the weight parity of each subblock.

In this work, we develop an FTEC protocol that uses WPEC to correct high-weight errors arising from up to 3 faults. As an example, consider the measurement of $\tilde{g}_1^z$ using the circuit depicted in Fig. 5.1a. Here we assume that a fault from any two-qubit gate can cause any

Form of error	m	2nd-level syndrome	Block triviality	Block parity
PIZZZII	7	(0,0,0)	(0,0,0,0,0,0,0)	(1,0,1,1,1,0,0)
	2,4,6	(1,0,0)	(1,0,0,0,0,0,0)	(0,0,1,1,1,0,0)
	1,3,5	(0,0,0)	(1,0,0,0,0,0,0)	(1,0,1,1,1,0,0)
IIPZZII	7	(1,0,0)	(0,0,0,0,0,0,0)	(0,0,1,1,1,0,0)
	2,4,6	(0,0,1)	(0,0,1,0,0,0,0)	(0,0,0,1,1,0,0)
	1,3,5	(1,0,0)	(0,0,1,0,0,0,0)	(0,0,1,1,1,0,0)
IIIPZII	7	(0,0,1)	(0,0,0,0,0,0,0)	(0,0,0,1,1,0,0)
	2,4,6	(1,1,1)	(0,0,0,1,0,0,0)	(0,0,0,0,1,0,0)
	1,3,5	(0,0,1)	(0,0,0,1,0,0,0)	(0,0,0,1,1,0,0)
IIIIPII	7	(1,1,1)	(0,0,0,0,0,0,0)	(0,0,0,0,1,0,0)
	2,4,6	(0,0,0)	(0,0,0,0,1,0,0)	(0,0,0,0,0,0,0)
	1,3,5	(1,1,1)	(0,0,0,0,1,0,0)	(0,0,0,0,1,0,0)
$\mathbf{I}^{\otimes 7}$	-	(0,0,0)	(0,0,0,0,0,0,0)	(0,0,0,0,0,0,0)

Table 5.1: All possible forms of data errors arising from a single fault occurred during syndrome measurement using a circuit in Fig. 5.1a (where $\mathbf{P} = I^{\otimes 7-m} \otimes Z^{\otimes m}$). The block parity corresponding to each form of errors can be determined by the 2nd-level syndrome and the block triviality obtained from a full syndrome measurement. By knowing the block parity, high-weight errors can be corrected using WPEC.

two-qubit Pauli errors on the qubits where the gate acts nontrivially (which corresponds to the noise model in Definition 3.1), and X-type and Z-type errors can be detected separately. Thus, we may assume that high-weight errors arising from a single CNOT fault is of the form **PIZZZII**, **IIPZZII**, **IIIPZII**, or **IIIIPII**, where $\mathbf{Z} = Z^{\otimes 7}$, $\mathbf{P} = I^{\otimes 7-m} \otimes Z^{\otimes m}$, and $m \in \{1, \ldots, 7\}$ (see the analysis of possible errors in Section 3.2 or [TCL20] for more details). It is not hard to find 2nd-level syndrome, block triviality, and block parity corresponding to each possible error. For example, error **PIZZZII** with $m = 6$ anticommutes with g_1^x and $\tilde{g}_1^x$, and commutes with the other generators. Thus, its corresponding 2nd-level syndrome, block triviality, and block parity are $(1,0,0), (1,0,0,0,0,0,0)$, and $(0,0,1,1,1,0,0)$, respectively. Table 5.1 displays all possible high-weight errors arising from a single fault during $\tilde{g}_1^z$ measurement and their corresponding 2nd-level syndrome, block triviality, and block parity. Note that except for the first and the last row (with errors differing by multiplication of a stabilizer), each row has a unique combination of

2nd-level syndrome and block triviality, so the block parity can be determined from the table. Since the 2nd-level syndrome and the block triviality can in turn be obtained from the generator measurements, all possible errors arising from a single CNOT fault during the measurement of $\tilde{g}_1^z$ can be corrected using WPEC. In addition, observe that **ZIZZZII** and $\mathbf{I}^{\otimes 7}$ are equivalent up to a multiplication of $\tilde{g}_1^z$ but their block parities are different. Here we can see that multiplying an error with some 2nd-level generators may change its block parity, but its 2nd-level syndrome and block triviality (which is deduced from its 1st-level syndrome) remain the same. In this case, WPEC is still applicable. We say that block parities are equivalent whenever they can be transformed to one another by multiplying the corresponding errors with some stabilizer.

In an actual fault-tolerant protocol, we want to distinguish all possible high-weight errors arising from various types of faults up to 3 faults, including any gate faults, faults during the preparation and measurement of ancilla qubits, and faults during wait time. The circuit construction in Fig. 5.1a, however, might not cause errors that can be distinguished. Note that possible errors arising from CNOT faults heavily depend on the ordering of CNOT gates being used in the measurement circuit. In Section 5.2, we will discuss conditions in which WPEC can be applied. We will also provide a family of circuits with specific CNOT ordering and an FTEC protocol for the $[\![49, 1, 9]\!]$ concatenated Steane code which can correct high-weight errors arising from up to 3 faults.

5.2 Fault-tolerant error correction protocol for the 49-qubit concatenated Steane code

As previously discussed in Chapter 4, FTEC gadget is one of the most essential gadgets for constructing large-scale quantum computers since it will be used repetitively in a fault-tolerant simulation of a quantum circuit. In [TL21b], an FTEC protocol which satisfies the following definition (which is equivalent to Definition 4.4, the conventional definition of FTEC gadget from [AGP06]) has been developed:

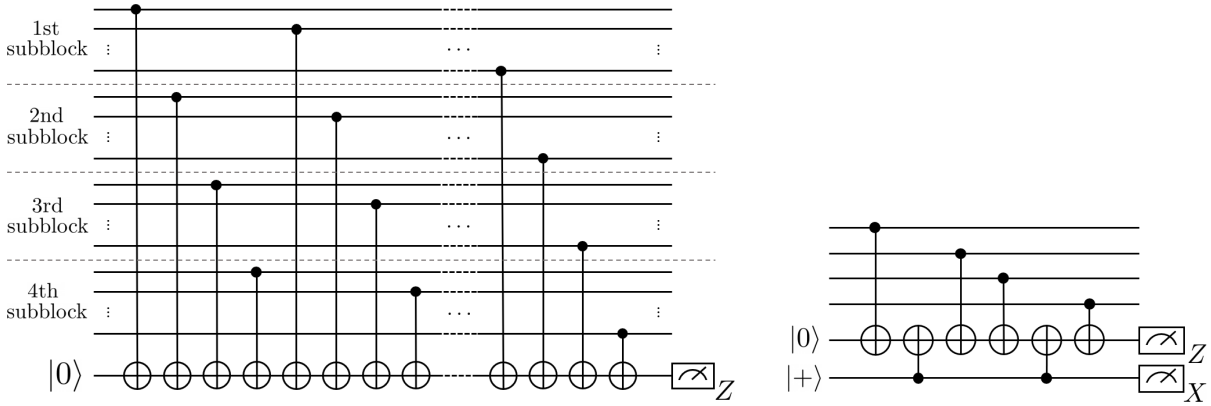
Definition 5.2 *Fault-tolerant error correction*

For $\tau = \lfloor (d-1)/2 \rfloor$ and $t \leq \tau$, an error correction protocol using a distance-d stabilizer code is t-fault tolerant *if the following two conditions are satisfied:*

1. *For any input codeword with error of weight r, if s faults occur during the protocol with $r + s \leq t$, ideally decoding the output state gives the same codeword as ideally decoding the input state.*

2. *If s faults happen during the protocol with $s \leq t$, no matter how many errors are present in the input state, the output state differs from any valid codeword by an error of at most weight s.*

The FTEC protocol for the $[\![49, 1, 9]\!]$ code developed in [TL21b] can correct up to 3 faults. The circuits for measuring 1st-level and 2nd-level generators are shown in Fig. 5.2. The types of faults being considered include faults that happen to the physical gates, faults during the preparation and measurement of ancilla qubits in the circuits, and faults during wait time.



(a) (b)

Figure 5.2: Circuits for measuring 2nd-level and 1st-level generators being used in [TL21b] are shown in (a) and (b), respectively. With this gate permutation, the block parity corresponding to every possible high-weight error arising from up to 3 faults can be accurately determined. As such, our protocol can correct up to 3 faults.

Let *fault combination* be a set of faults up to 3 faults (which may be of different types and can cause errors of weight much higher than 3 on the data qubits) as previously defined in Definition 3.2. Our goal is to distinguish all fault combinations that can be confusing and may cause WPEC to fail. Similar to an example of WPEC in Section 5.1, we can categorize

all possible fault combinations into subsets by their 2nd-level syndrome and block triviality. The following sufficient condition can determine when the WPEC technique can be applied:

Claim 5.2 *Sufficient condition for WPEC*

Let $\mathcal{F}_3$ be the set of all possible fault combinations arising from up to 3 faults during an FTEC protocol for the $[\![49, 1, 9]\!]$ code, and let $\mathcal{G}_j \subseteq \mathcal{F}_3$ be a subset of fault combinations with the same 2nd-level syndrome and the same block triviality (where $\bigcup_j \mathcal{G}_j = \mathcal{F}_3$). WPEC is applicable in the FTEC protocol if each $\mathcal{G}_j$ satisfies one of the following conditions:

1. *Data errors from all fault combinations in $\mathcal{G}_j$ give equivalent block parities.*
2. *Not every data error from a fault combination in $\mathcal{G}_j$ give the same block parity (or its equivalence), but for each pair of fault combinations in $\mathcal{G}_j$ whose block parities of their data errors are not equivalent, their 1st-level syndromes or flag measurement results (or both) are different.*

Proof:
Whenever subset $\mathcal{G}_j$ satisfies the first condition in Claim 5.2, we can find a block parity that works for all fault combinations in $\mathcal{G}_j$ using only the 2nd-level syndrome and the block triviality. A correction operator for each fault combination can be found following the definition of WPEC (Definition 5.1) using the 1st-level syndrome and the block parity. On the other hand, if $\mathcal{G}_j$ satisfies the second condition in Claim 5.2, a block parity cannot be accurately determined using only the 2nd-level syndrome and the block triviality. Fortunately, with the assistance of the 1st-level syndrome and the flag measurement result, fault combinations that correspond to non-equivalent block parities can be distinguished and the block parity of each fault combination can be found. Similarly, a correction operator for each fault combination can be determined following Definition 5.1. □

(Note that if $\mathcal{G}_j$ satisfies one of the conditions in Claim 5.2, any pair of errors in $\mathcal{G}_j$ either correspond to different syndromes or different flag measurement results, or are logically equivalent; that is, the subset $\mathcal{G}_j$ is distinguishable according to Definition 3.3. This means that $\mathcal{F}_3 = \bigcup_j \mathcal{G}_j$ is distinguishable whenever Claim 5.2 is satisfied for all $\mathcal{G}_j$'s.)

Whether a set of possible fault combinations satisfies Claim 5.2 or not depends heavily on the ordering of the CNOT gates and the use of flag ancillas in the circuits for syndrome

measurements. In our FTEC protocol for the $[\![49, 1, 9]\!]$ code, the CNOT gates being used in the circuits for measuring 2nd-level generator are applied in the following ordering:

$$(1, 8, 15, 22, 2, 9, 16, 23, \ldots, 7, 14, 21, 28), \tag{5.3}$$

where the numbers 1 to 28 represent the qubits in which $\tilde{g}_i^z$ acts nontrivially. That is, CNOT gates are applied on the first qubit in each subblock for all subblocks, then on the second qubit in each subblock for all subblocks, and so on. The circuit for measuring $\tilde{g}_1^z$ is shown in Fig. 5.2a. In addition, CNOT gates being used in the circuits for measuring 1st-level generator are in the normal ordering as shown in Fig. 5.2b. Note that there is no flag ancilla involved in the measurement of a 2nd-level generator, and there is one flag ancilla in the circuit for measuring a 1st-level generator.

Consider the case that there are some faults during Z-type generator measurements. Faulty circuits can produce nontrivial flag measurement results and cause error of any weight on the data qubits. Our goal is to detect and correct such an error using the flag measurement results from the faulty circuits, together with 1st-level and 2nd-level syndromes obtained from subsequent syndrome measurements. In particular, let the *flag vector* $\in \mathbb{Z}_2^{21}$ be a bitstring wherein each bit is the flag measurement result from each circuit for measuring g_i^z on each of the 7 subblocks. We define the *cumulative flag vector* $\vec{\mathbf{f}}_z \in \mathbb{Z}_2^{21}$ to be the entry-wise sum of flag vectors (modulo 2) obtained from g_i^z measurements accumulated from the first round up until the current round of measurements (see the protocol described below for the definition of a round of measurements). Cumulative flag vector $\vec{\mathbf{f}}_z$ together with 1st-level syndrome $\vec{\mathbf{s}}_{1x} \in \mathbb{Z}_2^{21}$, 2nd-level syndrome $\vec{\mathbf{s}}_{2x} \in \mathbb{Z}_2^3$, and block triviality $\vec{\eta}_x \in \mathbb{Z}_2^7$ from the latest round of measurements will be used for distinguishing all possible fault combinations that can occur during the syndrome measurements as described in Claim 5.2. Using the computer simulation described in Appendix A.1, we can verify that Claim 5.2 is satisfied when the number of input errors r and the number of faults s satisfy $r + s \leq 3$. A table of possible data errors and their corresponding $\vec{\mathbf{s}}_{1x}, \vec{\mathbf{s}}_{2x}, \vec{\eta}_x, \vec{\mathbf{f}}_z$, and block parity $\vec{p}_x$ (similar to Table 5.1) can also be obtained from the simulation. Moreover, the subsets $\mathcal{G}_j$ can be deduced from this table (see Appendix A.1 for more details).

Let *outcome bundle* $(\vec{\mathbf{s}}_1, \vec{\mathbf{s}}_2, \vec{\eta}, \vec{\mathbf{f}})$ be the collection of 1st-level syndrome $\vec{\mathbf{s}}_1 = (\vec{\mathbf{s}}_{1x}|\vec{\mathbf{s}}_{1z})$, 2nd-level syndrome $\vec{\mathbf{s}}_2 = (\vec{\mathbf{s}}_{2x}|\vec{\mathbf{s}}_{2z})$, block triviality $\vec{\eta} = (\vec{\eta}_x|\vec{\eta}_z)$, and cumulative flag vector $\vec{\mathbf{f}} = (\vec{\mathbf{f}}_x|\vec{\mathbf{f}}_z)$ obtained during a single round of full syndrome measurement, where subscripts x and z denote results corresponding to X-type and Z-type generator measurements.

Using the simulation result together with the fact that X-type and Z-type errors can be corrected separately, an FTEC protocol for the $[\![49, 1, 9]\!]$ code can be constructed as follows.

FTEC protocol for the $[\![49, 1, 9]\!]$ code

A full syndrome measurement, or a round of measurements, measure the generators in the following order: measure all $\tilde{g}_i^z$'s, then all $\tilde{g}_i^x$'s, then all g_i^z's, then all g_i^x's. Perform full syndrome measurements until the outcome bundles are repeated 4 times in a row. Afterwards, perform the following error correction procedure:

1. Find the subset $\mathcal{G}_j$ corresponding to $\vec{\mathbf{s}}_{2x}$ and $\vec{\eta}_x$ from the table of possible errors (obtained from the simulation in Appendix A.1).
   (a) If $\mathcal{G}_j$ satisfies Condition 1 in Claim 5.2, use a block parity of any fault combination in $\mathcal{G}_j$.
   (b) If $\mathcal{G}_j$ satisfies Condition 2 in Claim 5.2, use a block parity of any combination in $\mathcal{G}_j$ that corresponds to $\vec{\mathbf{s}}_{1x}$ and $\vec{\mathbf{f}}_z$.
   (c) If there is no $\mathcal{G}_j$ from the table of possible errors which corresponds to $\vec{\mathbf{s}}_{2x}$ and $\vec{\eta}_x$, use the block parity with all 1's.
2. Let $(\vec{\mathbf{s}}_{1x})_i$ be the 1st-level syndrome and wp_i be the weight parity of the i-th subblock. Apply Z-type error correction on each subblock as given by Definition 5.1. In particular:
   (a) If $(\vec{\mathbf{s}}_{1x})_i$ is trivial, apply $ZZIZIII$ (logically equivalent to $Z^{\otimes 7}$) to the i-th subblock when wp_i is odd, or do nothing when wp_i is even.
   (b) If $(\vec{\mathbf{s}}_{1x})_i$ is nontrivial, apply $E^z_{wt\text{-}1}((\vec{\mathbf{s}}_{1x})_i)$ to the i-th subblock when wp_i is odd, or apply $E^z_{wt\text{-}2}((\vec{\mathbf{s}}_{1x})_i)$ when wp_i is even.
3. If there is no $\mathcal{G}_j$ from the table of possible errors which corresponds to $\vec{\mathbf{s}}_{2x}$ and $\vec{\eta}_x$, further apply the following error correction procedure: find a Pauli operator from $\{\mathbf{ZIIIIII}, \mathbf{IZIIIII}, \ldots, \mathbf{IIIIIIZ}\}$ which corresponds to $\vec{\mathbf{s}}_{2x}$, then apply such an operator (or its logically equivalent operator) to the data qubits.
4. Repeat steps 1–3 but use $\vec{\mathbf{s}}_{1z}$, $\vec{\mathbf{s}}_{2z}$, $\vec{\eta}_z$, and $\vec{\mathbf{f}}_x$ to deduce the X-type error correction ($E^x_{wt\text{-}1}((\vec{\mathbf{s}}_{1z})_i)$, $E^x_{wt\text{-}2}((\vec{\mathbf{s}}_{1z})_i)$, or $XXIXIII$) for each subblock.

Here we will assume that there are at most 3 faults during the protocol and the error is of Z type. The assumption on the number of faults guarantees that the outcome bundles must be repeated 4 times in a row within 16 rounds (the outcome bundle cannot keep changing forever since the number of faults is limited). To verify that the protocol above is 3-fault tolerant, i.e., it satisfies the FTEC conditions in Definition 5.2 with $t = 3$ (the $[\![49, 1, 9]\!]$ code acts as a distance-7 code), first let us consider the case that there are no faults during the last round of full syndrome measurement. In this case, the outcome bundle corresponds to the actual data error. From the simulation in Appendix A.1, we know that whenever $r + s \leq 3$, one of the conditions in Claim 5.2 is satisfied and the block parity can be accurately determined. The operation in Step 2 will give the correct output state, thus both of the FTEC conditions are satisfied. On the other hand, if $r + s > 3$ but $s \leq 3$, $\vec{\mathbf{s}}_{2x}$ and $\vec{\eta}_x$ may not correspond to any error in the table of possible errors. By using the block parity with all 1's, the operation in Step 2 will project the state in each subblock back to the subspace of the $[\![7, 1, 3]\!]$ code, where each subblock has an error equivalent to either $\mathbf{I}$ or $\mathbf{Z}$ after the operation. Afterwards, the operation in Step 3 will project the output state back to the subspace of the $[\![49, 1, 9]\!]$ code. Thus, the second condition in Definition 5.2 is satisfied.

Now, let us consider the case that there are some faults during the last round of full syndrome measurement. The outcome bundle we obtained from the last round may not correspond to the data error since some errors arising during the last round may be undetectable. Since we perform full syndrome measurements until the outcome bundles are repeated 4 times in a row and there are at most 3 faults during the whole protocol, at least one of the last 4 rounds of full syndrome measurement must be correct. From the simulation result in Appendix A.1, the outcome bundle obtained from the last round (which is equal to that obtained from any correct round in the last 4 rounds) can definitely correct the error occurred before the last correct round. Here we want to verify that whenever the last 4 rounds have s faults (where $s \leq 3$), after the last round, the weight of the data error is increased by no more than s. This can be verified using the computer simulation described in Appendix A.2. By applying operation in Step 2 (and possibly Step 3) as previously discussed, the output state differs from a valid codeword by an error of weight at most s, regardless of the number of input errors. Thus, the second condition in Definition 5.2 is satisfied. Furthermore, whenever $r + s \leq 3$, we will obtain an output state which differs from a correct output state by an error of weight at most 3. Therefore, the first condition in Definition 5.2 is also satisfied.

The analysis for X-type errors is similar to that of Z-type errors. Note that during the measurement of Z-type generators, a single gate fault can cause an X-type error of weight 1 on the data qubits. This error can be considered as an input error for the measurement of X-type generators, thus the same analysis is applicable.

It should be noted that the FTEC protocol for the $[\![49, 1, 9]\!]$ concatenated Steane code proposed in [TL21b] was designed with the conventional definition of FTEC gadget (Definition 5.2) in mind. Nevertheless, it is also possible to construct an FTEC protocol satisfying the revised definition of FTEC gadget (Definition 4.10) since the circuit construction presented in [TL21b] gives a distinguishable fault set $\mathcal{F}_3$. One applicable protocol is the FTEC protocol for a capped color code in Section 6.3.1 (which is also applicable to any CSS code whose fault set is distinguishable and satisfies Definition 4.11).

5.3 Weight parity error correction for other codes

Besides the Steane code, we find that the WPEC technique is applicable to the $[\![23, 1, 7]\!]$ Golay code [Ste03], which is a perfect CSS code of distance 7. The $[\![23, 1, 7]\!]$ Golay code can correct up to 3 errors and can be constructed from the parity check matrix of the classical $[23, 12, 7]$ Golay code [MS77]. In cyclic form, the $[\![23, 1, 7]\!]$ Golay code can be constructed from the parity check matrix,

$$\begin{pmatrix}
1&1&1&1&1&0&0&1&0&0&1&0&1&0&0&0&0&0&0&0&0&0&0\\
0&1&1&1&1&1&0&0&1&0&0&1&0&1&0&0&0&0&0&0&0&0&0\\
0&0&1&1&1&1&1&0&0&1&0&0&1&0&1&0&0&0&0&0&0&0&0\\
0&0&0&1&1&1&1&1&0&0&1&0&0&1&0&1&0&0&0&0&0&0&0\\
0&0&0&0&1&1&1&1&1&0&0&1&0&0&1&0&1&0&0&0&0&0&0\\
0&0&0&0&0&1&1&1&1&1&0&0&1&0&0&1&0&1&0&0&0&0&0\\
0&0&0&0&0&0&1&1&1&1&1&0&0&1&0&0&1&0&1&0&0&0&0\\
0&0&0&0&0&0&0&1&1&1&1&1&0&0&1&0&0&1&0&1&0&0&0\\
0&0&0&0&0&0&0&0&1&1&1&1&1&0&0&1&0&0&1&0&1&0&0\\
0&0&0&0&0&0&0&0&0&1&1&1&1&1&0&0&1&0&0&1&0&1&0\\
0&0&0&0&0&0&0&0&0&0&1&1&1&1&1&0&0&1&0&0&1&0&1
\end{pmatrix},$$

which can be generated from the check polynomial $h(x) = x^{12}+x^{10}+x^7+x^4+x^3+x^2+x+1$ [MS77]. The i-th Z-type (or X-type) generator of this code will be denoted as g_i^z (or g_i^x) where $i = 1, \ldots, 11$. The logical X and logical Z operators of this code can be chosen to be $\bar{X} = X^{\otimes 23}M$ and $\bar{Z} = Z^{\otimes 23}N$ for any stabilizer operators M, N.

Similar to the $[\![7, 1, 3]\!]$ code, we can prove the equivalence of errors with the same syndrome and the same weight parity as follows:

Claim 5.3 *Logical equivalence of errors with the same syndrome for the* $[\![23, 1, 7]\!]$ *Golay code*

Let S_z *be the subgroup generated by* Z*-type generators of the* $[\![23, 1, 7]\!]$ *Golay code. Suppose* E_1, E_2 *are arbitrary* Z*-type errors (of any weights) on the* $[\![23, 1, 7]\!]$ *code with the same syndrome. Then,* E_1 *and* E_2 *have the same weight parity if and only if* $E_1 = E_2 \cdot M$ *for some stabilizer* $M \in S_z$*.*

Proof:
We can verify that every Z-type stabilizer in S_z has even weight, and every logical Z operator has odd weight. The rest of the proof follows the proof of Claim 5.1. □

Let us consider Z-type error correction for the $[\![23, 1, 7]\!]$ code. Since the code is a perfect CSS code of distance 7, for each $\vec{s}_x \in \mathbb{Z}_2^{11}$, $(\vec{s}_x|0...0)$ is the syndrome of a unique Z-type error of weight ≤ 3. Suppose that a codeword is corrupted by a Z-type error with syndrome $\vec{s}_x$. If we apply the minimal weight error correction corresponding to $\vec{s}_x$, we sometimes obtain the codeword with undesirable logical Z operator. Fortunately, by knowing the weight parity of the error, the WPEC technique can be applied. The error correction procedure for the $[\![23, 1, 7]\!]$ code is defined as follows:

Definition 5.3 *Weight parity error correction for the* $[\![23, 1, 7]\!]$ *Golay code*

Suppose a Z*-type error* E *occurs to a codeword of the* $[\![23, 1, 7]\!]$ *code. Let* $\vec{s}_x$ *and* w *be the syndrome and the weight of* E*, and let* $E^z_{min}(\vec{s}_x)$ *be the unique minimal weight error correction corresponding to the syndrome* $\vec{s}_x$*. The following procedure is called* weight parity error correction (WPEC)*:*

1. *If* $E^z_{min}(\vec{s}_x)$ *has even weight (0 or 2), apply* $E^z_{min}(\vec{s}_x)$ *to the data qubits whenever* w *is even, or apply any* Z*-type operator* P *that has odd weight and corresponds to* $\vec{s}_x$ *to the data qubits whenever* w *is odd.*

2. *If $E^z_{min}(\vec{s}_x)$ has odd weight (1 or 3), apply $E^z_{min}(\vec{s}_x)$ to the data qubits whenever w is odd, or apply any Z-type operator P that has even weight and corresponds to $\vec{s}_x$ to the data qubits whenever w is even.*

Note that the $[\![23, 1, 7]\!]$ Golay code can be made cyclic, thus it can distinguish high-weight errors in consecutive form [TCL20]. Claim 5.3 together with the cyclic property give us some possibilities to construct an FTEC protocol for the $[\![529, 1, 49]\!]$ concatenated Golay code in the same way as what we have done for the $[\![49, 1, 9]\!]$ code. We expect that our technique can lead to a protocol which can correct a large number of faults and will compare well with other FTEC schemes. To reach this goal, syndrome extraction circuits with appropriate permutation of gates (and possibly with flag ancillas) must be found so that conditions similar to those in Claim 5.2 are satisfied. The search for such circuits with careful numerical verification of fault tolerance is a challenging and interesting future research direction.

The WPEC technique may also apply to the code obtained from concatenating the Steane code to the k-th level, e.g., the $[\![7^k, 1, 3^k]\!]$ code. Since the k-th-level Steane code is a distance-3 code, we expect that a block of errors in the $(k-1)$-th level can be determined and corrected using the syndrome and the block parity defined at the k-th level. Again, however, appropriate syndrome extraction circuits must be found, which is beyond the scope of this work.

Chapter 6

Fault-tolerant Error Correction and Quantum Computation for Capped Color Codes

In Chapter 5, we describe how an FTEC protocol for the $[\![49, 1, 9]\!]$ concatenated Steane code can be constructed using flag and weight parity techniques, and we discuss possibilities of applying such techniques to other concatenated codes such as the $[\![529, 1, 49]\!]$ concatenated Golay code and concatenated Steane codes with more than 2 levels of concatenation. Nevertheless, there are families of codes that attain high distance without code concatenation. Topological codes in which the code distance can be made arbitrarily large by increasing the lattice size are good candidates for practical implementation of quantum computers since fault-tolerant protocols for these codes typically give very high accuracy thresholds [DKLP02, DCP10, BH13, DCP13, ABCB14, BSV14, Del14, BLP+16, BNB16, CR18a, DBT18, DP18, KD19, KP19, MKJO19, NB19, VBK21]. Examples of two-dimensional (2D) topological stabilizer codes are 2D toric codes [Kit97, BK98] and 2D color codes [BMD06]. These codes are suitable for physical implementations using superconducting qubits [FMMC12, CZY+20, CKYZ20] and qubits realized by Majorana zero modes [KKL+17, CBDH20] since qubits can be arranged on a 2D plane and only quantum gates involving neighboring qubits are required. Toric codes and color codes can be transformed to one another using the techniques developed in [KYP15] (see also [VB19]).

The simplest way to perform FTQC on a topological stabilizer code is to implement logical gates by applying physical gates transversally since doing so does not spread errors (therefore fault tolerant). Unfortunately, it is known by the Eastin-Knill theorem that a universal set of quantum operations cannot be achieved using only transversal gates [EK09]. Moreover, logical gates which can be implemented transversally on a 2D topological stabilizer code are in the Clifford group [BK13] (see also [PY15]). The Clifford group can be generated by the Hadamard gate (H), the $\frac{\pi}{4}$-gate (S), and the CNOT gate [CRSS97, Got98]. A transversal CNOT gate is achievable by both 2D toric codes and 2D color codes since these codes are in the CSS code family [CS96, Ste96]. In addition, the 2D color codes have transversal H and S gates [BMD06], so, any Clifford operation can be implemented transversally on any 2D color code.

Implementing only Clifford gates on a 2D color code is not particularly interesting since Clifford operation can be efficiently simulated by a classical computer (the result is known as Gottesman-Knill theorem) [Got97, NC00]. However, universality can be achieved by Clifford gates together with any gate not in the Clifford group [NRS01]. There are two compelling approaches for implementing a non-Clifford gate on a 2D color code: magic state distillation [BK05] and code switching [PR13, ADCP14, Bom15a, KB15]. The former approach focuses on producing high-fidelity T states from noisy T states and Clifford operations, where $|T\rangle = (|0\rangle + \sqrt{i}|1\rangle)/\sqrt{2}$ is the state that can be used to implement non-Clifford $T = \left(\begin{smallmatrix} 1 & 0 \\ 0 & \sqrt{i} \end{smallmatrix}\right)$ operation. By replacing any physical gates and qubits with logical gates and blocks of code, a logical T gate can be implemented using a method similar to that proposed in [BK05]. The latter approach uses the gauge fixing method to switch between a 2D color code (in which Clifford gates are transversal) and a 3D color code (in which the T gate is transversal). A recent study [BKS21] which compares the overhead required for these two approaches shows that code switching does not outperform magic state distillation when certain FT schemes are used, except for some small values of physical error rate. Nevertheless, their results do not rule out the possibilities of FT schemes have yet to be discovered, in which the authors are hopeful that such schemes could reduce the overhead required for either of the aforementioned approaches.

In this chapter, we will extend the ideas of FTEC using flags and weight parities from Chapter 5 to the 2D color codes. The major difference between the constructions in this chapter and the previous chapter is that the bigger code being used in this chapter is not obtained from concatenating smaller codes. Our study leads to a family of capped color

codes, which are CSS subsystem codes [Pou05, Bac06]. Two stabilizer codes obtained from a subsystem capped color code, namely capped color codes in H form and T form, will be studied in this chapter. The code in H form possesses transversal H, S, and CNOT gates, while the code in T form possesses transversal CNOT and T gates. Similar to 2D and 3D color codes, one can transform between the capped color codes in H form and T form through code switching.

This chapter is organized as follows: In Section 6.1, we review basic properties of the 3D color code of distance 3 (which is defined as a subsystem code). We then provide a construction of circuits for measuring the stabilizer generators of the 3D color code in H form which give a distinguishable fault set. In Section 6.2, we define a family of capped color codes, whose properties are very similar to those of the 3D color code of distance 3. Afterwards, circuits for measuring the stabilizer generators of the capped code in H form are constructed using ideas from the previous section. We also prove Theorem 6.1 which states sufficient conditions for the circuits that can give a distinguishable fault set, then provide circuits for the capped color codes of distance 5 and 7 which satisfy the sufficient conditions. In Section 6.3, we construct fault-tolerant protocols for a capped color code in H form, including FTEC, FTM, and FTP protocols. In addition, an FTEC protocol for a stabilizer code whose syndrome measurement circuits give a distinguishable fault set can be developed using similar ideas. The protocol for such a code is provided in Section 6.4. All protocols provided in this chapter satisfy the revised definitions of fault-tolerant gadgets previously discussed in Section 4.2. The FTEC techniques and protocols presented in this chapter follow Sections III, IV, and V of [TL21a].

6.1 Syndrome measurement circuits for the 3D color code of distance 3

In this section, we will try to find circuits for measuring generators of the 3D color code of distance 3 which gives a distinguishable fault set. We will first define a 3D color code of distance 3 as a CSS subsystem code and observe some of its properties which is useful for FTQC. Afterwards, we will give the CNOT orderings for the circuits which can make the fault set become distinguishable.

6.1.1 The 3D color code of distance 3

First, let us consider the qubit arrangement as displayed in Fig. 6.1a. A 3D color code of distance 3 [Bom15a] is a $[\![15, 1, 3]\!]$ CSS subsystem code [Pou05, Bac06] which can be described by the stabilizer group $S_{3D} = \langle v_i^x, v_i^z \rangle$ and the gauge group $G_{3D} = \langle v_i^x, v_i^z, f_j^x, f_j^z \rangle$, $i = 0, 1, 2, 3$ and $j = 1, 2, ..., 6$, where v_i^x's and f_j^x's (or v_i^z's and f_j^z's) are X-type (or Z-type) operators defined on the following set of qubits:

- v_0^x (or v_0^z) is defined on $\mathtt{q}_0, \mathtt{q}_1, \mathtt{q}_2, \mathtt{q}_3, \mathtt{q}_4, \mathtt{q}_5, \mathtt{q}_6, \mathtt{q}_7$
- v_1^x (or v_1^z) is defined on $\mathtt{q}_1, \mathtt{q}_2, \mathtt{q}_3, \mathtt{q}_5, \mathtt{q}_8, \mathtt{q}_9, \mathtt{q}_{10}, \mathtt{q}_{12}$
- v_2^x (or v_2^z) is defined on $\mathtt{q}_1, \mathtt{q}_3, \mathtt{q}_4, \mathtt{q}_6, \mathtt{q}_8, \mathtt{q}_{10}, \mathtt{q}_{11}, \mathtt{q}_{13}$
- v_3^x (or v_3^z) is defined on $\mathtt{q}_1, \mathtt{q}_2, \mathtt{q}_4, \mathtt{q}_7, \mathtt{q}_8, \mathtt{q}_9, \mathtt{q}_{11}, \mathtt{q}_{14}$
- f_1^x (or f_4^z) is defined on $\mathtt{q}_1, \mathtt{q}_2, \mathtt{q}_3, \mathtt{q}_5$
- f_2^x (or f_5^z) is defined on $\mathtt{q}_1, \mathtt{q}_3, \mathtt{q}_4, \mathtt{q}_6$
- f_3^x (or f_6^z) is defined on $\mathtt{q}_1, \mathtt{q}_2, \mathtt{q}_4, \mathtt{q}_7$
- f_4^x (or f_1^z) is defined on $\mathtt{q}_1, \mathtt{q}_4, \mathtt{q}_8, \mathtt{q}_{11}$
- f_5^x (or f_2^z) is defined on $\mathtt{q}_1, \mathtt{q}_2, \mathtt{q}_8, \mathtt{q}_9$
- f_6^x (or f_3^z) is defined on $\mathtt{q}_1, \mathtt{q}_3, \mathtt{q}_8, \mathtt{q}_{10}$

where qubit i in Fig. 6.1a is denoted by $\mathtt{q}_i$. Graphically, v_i^x's and v_i^z's are 8-body volumes shown in Fig. 6.1b, and f_j^x's and f_j^z's are 4-body faces shown in Fig. 6.1c. Note that f_j^x and f_k^z anticommute when $j = k$, and they commute when $j \neq k$. The dual lattice of the 3D color code of distance 3 is illustrated in Fig. 6.1d, where each vertex represents each stabilizer generator.

The 3D color code of distance 3 can be viewed as the $[\![15, 7, 3]\!]$ Hamming code in which 6 out of 7 logical qubits become gauge qubits. From the subsystem code previously described, a $[\![15, 1, 3]\!]$ stabilizer code can be constructed by fixing some gauge qubits; i.e., choosing some gauge operators which commute with one another and including them in the stabilizer group. In this work, we will discuss two possible ways to construct a stabilizer code from

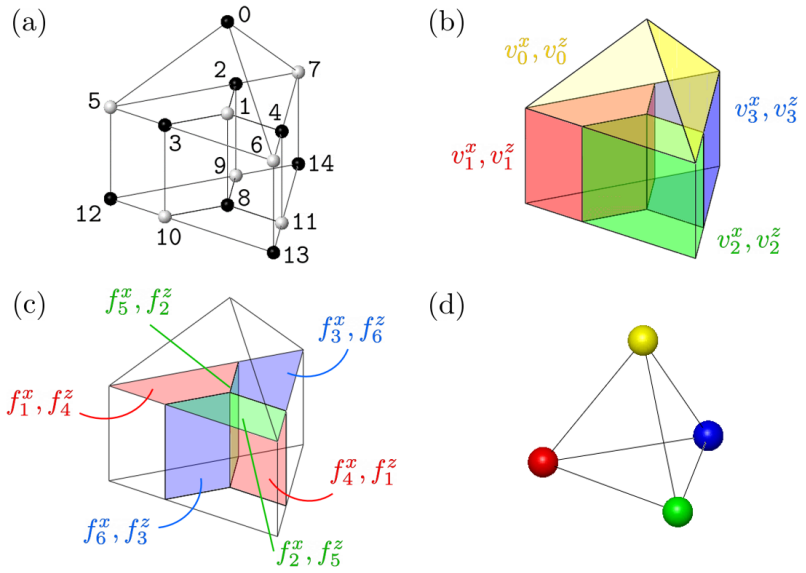


Figure 6.1: The 3D color code of distance 3. In (a), qubits are represented by vertices. Note that the set of qubits are bipartite, as displayed by black and white colors. Stabilizer generators and gauge generators of the code are illustrated by volume operators in (b) and face operators in (c), respectively. The dual lattice of the code is shown in (d).

the 3D color code of distance 3. The resulting codes will be called the 3D color code in H form and the 3D color code in T form.

The 3D color code of distance 3 in H form

Let us consider the center plane of the code shown in Fig. 6.1a which covers $\mathtt{q}_1$ to $\mathtt{q}_7$. We can see that the plane looks exactly like the 2D color code of distance 3 [BMD06], whose stabilizer group is $S_{\mathrm{2D}} = \langle f_1^x, f_2^x, f_3^x, f_4^z, f_5^z, f_6^z \rangle$ (the 2D color code of distance 3 is equivalent to the $[\![7,1,3]\!]$ Steane code). The 3D color code in H form is constructed by adding the stabilizer generators of the 2D color code to the old generating set of the 3D color code; the stabilizer group of the 3D color code of distance 3 in H form is

$$S_{\mathrm{H}} = \langle v_0^x, v_1^x, v_2^x, v_3^x, v_0^z, v_1^z, v_2^z, v_3^z, f_1^x, f_2^x, f_3^x, f_4^z, f_5^z, f_6^z \rangle. \tag{6.1}$$

We can choose logical X and logical Z operators of this code to be $X^{\otimes n}M$ and $Z^{\otimes n}N$ for some stabilizers $M, N \in S_{\mathrm{H}}$. One important property of the code in H form for fault-tolerant quantum computation is that the logical Hadamard, S, and CNOT gates are

transversal; i.e., $\bar{H} = H^{\otimes n}$ is a logical Hadamard gate, $\bar{S} = (S^\dagger)^{\otimes n}$ is a logical S gate, and $\overline{\text{CNOT}} = \text{CNOT}^{\otimes n}$ is a logical CNOT gate, where $H = \frac{1}{\sqrt{2}}\left(\begin{smallmatrix} 1 & 1 \\ 1 & -1 \end{smallmatrix}\right)$ and $S = \left(\begin{smallmatrix} 1 & 0 \\ 0 & i \end{smallmatrix}\right)$.

Note that the choice of stabilizer generators for S_{H} is not unique. However, the choice of generators determines how the error syndrome will be measured, and different choices of generators can give different fault sets. The circuits for measuring generators discussed later in Section 6.1.2 only correspond to the choice of generators in Eq. (6.1).

The 3D color code of distance 3 in T form

Compared to the code in H form, the 3D color code of distance 3 in T form is constructed from different gauge operators of the $[\![15, 1, 3]\!]$ subsystem code. In particular, the generators of the code in T form consist of the generators of the $[\![15, 1, 3]\!]$ subsystem code and all Z-type 4-body face generators; i.e., the stabilizer group of the code in T form is

$$S_{\text{T}} = \langle v_0^x, v_1^x, v_2^x, v_3^x, v_0^z, v_1^z, v_2^z, v_3^z, f_1^z, f_2^z, f_3^z, f_4^z, f_5^z, f_6^z \rangle. \tag{6.2}$$

Similar to the code in H form, we can choose logical X and logical Z operators of this code to be $X^{\otimes n}M$ and $Z^{\otimes n}N$ for some stabilizers $M, N \in S_{\text{T}}$. Also, CNOT gate is transversal in the code of T form. However, one major difference from the code in H form is that Hadamard and S gates are not transversal in this code. Instead, a T gate is transversal; a logical T gate can be implemented by applying T gates on all qubits represented by black vertices in Fig. 6.1a and applying $T^\dagger$ gates on all qubits represented by white vertices, where $T = \left(\begin{smallmatrix} 1 & 0 \\ 0 & \sqrt{i} \end{smallmatrix}\right)$.

In fact, the code in T form is equivalent to the $[\![15, 1, 3]\!]$ quantum Reed-Muller code. Note that Lemma 3.1 is applicable to both codes in H form and T form since they have all code properties required by the lemma, even though X-type and Z-type generators are not similar in the case of the code in T form.

Code switching

It is possible to transform between the code in H form and the code in T form using the technique called *code switching* [PR13, ADCP14, Bom15a, KB15]. The process involves measurements of gauge operators of the $[\![15, 1, 3]\!]$ subsystem code, which can be done as follows: Suppose that we start from the code in H form. We can switch to the code in

T form by first measuring f_1^z, f_2^z and f_3^z. Afterwards, we must apply an X-type Pauli operator that

1. commutes with all v_i^x's and v_i^z's $(i = 0, 1, 2, 3)$, and
2. commutes with f_4^z, f_5^z, f_6^z, and
3. for each $j = 1, 2, 3$, commutes with f_j^z if the outcome from measuring such an operator is 0 (the eigenvalue is $+1$) or anticommutes with f_j^z if the outcome is 1 (the eigenvalue is -1).

Switching from the code in T form to the code in H form can be done similarly, except that f_1^x, f_2^x and f_3^x will be measured and the operator to be applied must be a Z-type Pauli operator that commutes or anticommutes with f_1^x, f_2^x and f_3^x (depending on the measurement outcomes).

Transversal gates satisfy the conditions for fault-tolerant gate gadgets proposed in [AGP06] (see Definition 4.3), thus they are very useful for fault-tolerant quantum computation. It is known that universal quantum computation can be performed using only H, S, CNOT, and T gates [CRSS97, Got98, NRS01]. However, for any QECC, universal quantum computation cannot be achieved using only transversal gates due to the Eastin-Knill theorem [EK09]. Fortunately, the code switching technique allows us to perform universal quantum computation using both codes in H form and T form; any logical Clifford gate can be performed transversally on the code in H form since the Clifford group can be generated by $\{H, S, \text{CNOT}\}$, and a logical T gate can be performed transversally on the code in T form. For the 3D color code of distance 3, code switching can be done fault-tolerantly using the above method [Bom15a, KB15] or a method presented in [BKS21] which involves a logical Einstein-Podolsky-Rosen (EPR) state.

6.1.2 Circuit configuration for the 3D color code of distance 3

In this section, circuits for measuring the generators of the 3D color code of distance 3 in H form will be developed. Here we will try to find CNOT orderings for the circuits which make fault set $\mathcal{F}_1$ distinguishable (where $\mathcal{F}_1$ is the set of all fault combinations arising from up to 1 fault as defined in Definition 3.3). The ideas used for the circuit construction

in this section will be later adapted to the circuits for measuring generators of a capped color code (the code will be defined in Section 6.2.1 and the circuit construction will be discussed in Section 6.2.2). Fault-tolerant protocols for the 3D color code is distance 3 and capped color codes will be later discussed in Section 6.3.

For simplicity, since X-type and Z-type data errors can be corrected separately and X-type and Z-type generators of our choice have the same form, we will only discussed the case that a single fault can give rise to a Z-type data error. Similar analysis will also be applicable to the case of X-type errors. We start by observing that the 2D color code of distance 3 is a subcode of the the 3D color code of distance 3 in H form, where the 2D color code lies on the center plane of the code illustrated in Fig. 6.1a. The 2D color code is a code to which Lemma 3.1 is applicable, meaning that if we can measure the syndrome and the weight parity of any Z-type Pauli error occurred on the center plane, we can always find a Pauli operator logically equivalent to such an error. Moreover, we can see that the generator v_0^x has support on all qubits on the center plane ($\mathtt{q}_1$ to $\mathtt{q}_7$). This means that the weight parity of a Z-type error on the center plane can be obtained by measuring v_0^x. For these reasons, we can always find an error correction operator for any Z-type error occurred on the center plane using the measurement outcomes of f_1^x, f_2^x, f_3^x (which give the syndrome of the error evaluated on the 2D color code) and the measurement outcome of v_0^x (which gives the weight parity of the error).

All circuits for measuring generators of the 3D color code in H form used in this section are non-flag circuits. Each circuit has w data CNOTs where w is the weight of the operator being measured. The circuit for each generator looks similar a circuit in Fig. 6.2, but the ordering of data CNOTs has yet to be determined.

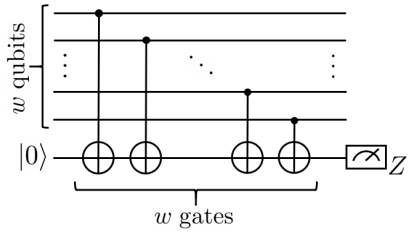


Figure 6.2: A non-flag circuit for measuring a Z-type generator of weight w for the 3D color code of distance 3. The ordering of the CNOT gates for each generator has yet to be determined.

Our goal is to find CNOT orderings for all circuits involved in the syndrome measurement so that $\mathcal{F}_1$ is distinguishable. Thus, we have to consider all possible errors arising from a single fault, not only the errors occurred on the center plane. Let us first consider an arbitrary single fault which can lead to a purely Z-type error. Since the 3D color code in H form has distance 3, all Z-type errors of weight 1 correspond to different syndromes. All we have to worry about are single faults which can lead to a Z-type error of weight > 1 that has the same syndrome as some error of weight 1 but is not logically equivalent to such an error. Note that a Z-type error of weight > 1 arising from a single fault can only be caused by a faulty CNOT gate in some circuit for measuring a Z-type generator.

We can divide the generators of the 3D color code in H form into 3 categories:

1. `cap` generators, consisting of v_0^x and v_0^z,
2. `f` generators, consisting of $f_1^x, f_2^x, f_3^x, f_4^z, f_5^z, f_6^z$,
3. `v` generators, consisting of $v_1^x, v_2^x, v_3^x, v_1^z, v_2^z, v_3^z$.

(v_0^x and v_0^z are considered separately from other `v` generators because they cover all qubits on the center plane.) Here we will analyze the pattern of Z-type errors arising from the measurement of Z-type generators of each category. The syndrome of each Z-type error will be represented in the form $(u, \vec{v}, \vec{w})$, where $u, \vec{v}, \vec{w}$ are syndromes obtained from the measurement of `cap`, `f`, and `v` generators of X type, respectively. Note that for each `v` generator, there will be only one `f` generator such that the set of supporting qubits of the `v` generator contains all supporting qubits of the `f` generator (for example, v_1^x and f_1^x, or v_1^z and f_4^z).

Let us start by observing the syndromes of any Z-type error of weight 1. An error on the following qubits gives the syndrome of the following form:

- An error on $\mathtt{q}_0$ gives syndrome $(1, \vec{0}, \vec{0})$.
- An error on $\mathtt{q}_i$ $(i = 1, \ldots, 7)$ gives syndrome of the form $(1, \vec{q_i}, \vec{q_i})$.
- An error on $\mathtt{q}_{7+i}$ $(i = 1, \ldots, 7)$ gives syndrome of the form $(0, \vec{0}, \vec{q_i})$.

where $\vec{q_i} \in \mathbb{Z}_2^3$ is not zero (see Table 6.1 as an example). We can see that all Z-type errors of weight 1 give different syndromes as expected. Next, let us consider a Z-type error E of

any weight which occurs only on the center plane. Suppose that the weight parity of E is wp (wp is 0 or 1), and the syndrome of E obtained from measuring f_1^x, f_2^x, f_3^x is $\vec{p}$. Then, the syndrome of E obtained from measuring all X-type generators is as follows:

- An error E on the center plane gives syndrome of the form $(\text{wp}, \vec{p}, \vec{p})$.

We find that:

1. E and the error on $\mathtt{q}_0$ will have the same syndrome if E has odd weight and $\vec{p}$ is trivial, which means that E is equivalent to $Z^{\otimes 7}$ on the center plane. In this case, E and Z_0 are logically equivalent up to a multiplication of v_0^z and some stabilizer.
2. E and an error on $\mathtt{q}_i$ $(i = 1, 2, \ldots, 7)$ will have the same syndrome if E has odd weight and $\vec{p} = \vec{q}_i$ for some i. In this case, E and Z_i have the same weight parity and the same syndrome (evaluated by the generators of the 2D color code), meaning that E and Z_i are logically equivalent by Lemma 3.1.
3. E and an error on $\mathtt{q}_i$ $(i = 7, 8, \ldots, 14)$ cannot have the same syndrome since $\vec{q}_i \neq \vec{0}$.

Therefore, a Z-type error of any weight occurred only on the center plane either has syndrome different from those of Z-type errors of weight 1, or is logically equivalent to some Z-type error of weight 1.

Because of the aforementioned properties of a Z-type error on the center plane, we will try to design circuits for measuring Z-type generators so that most of the possible Z-type errors arising from a single fault are on the center plane. Finding a circuit for any `f` generator is easy since for the 3D color code in H form, any `f` generator lies on the center plane, so any CNOT ordering will work. Finding a circuit for a `cap` generator is also easy; if the first data CNOT in the circuit is the one that couples $\mathtt{q}_0$ with the syndrome ancilla, we can make sure that all possible Z-type errors arising from a faulty CNOT in this circuit are on the center plane (up to a multiplication of v_0^z or v_0^x).

Finding a circuit for measuring a `v` generator is not obvious. Since some parts of any `v` generator of Z type are on the center plane and some parts are off the plane, some Z-type errors from a faulty data CNOT have support on some qubits which are not on the center plane. We want to make sure that in such cases, the error will not cause any problem; i.e., its syndrome must be different from those of other Z-type errors, or it must be logically

equivalent to some Z-type error. In particular, we will try to avoid the case that a CNOT fault can cause a Z error of weight > 1 which is totally off-plane. This is because such a high-weight error and some Z_i with $i = 8, 9, ..., 14$ may have the same syndrome but they are not logically equivalent (for example, $Z_{10}Z_{12}$ and Z_{13} have the same syndrome but they are not logically equivalent).

One possible way to avoid such an error is to arrange the data CNOTs so that the qubits on which they act are alternated between on-plane and off-plane qubits. An ordering of data CNOTs used in the circuit for any `v` generator will be referenced by the ordering of data CNOTs used in the circuit for its corresponding `f` generator. For example, if the ordering of data CNOTs used for f_4^z is (2,5,3,1), then the ordering of data CNOTs used for v_1^z will be (2,9,5,12,3,10,1,8). A configuration of data CNOTs for a `v` generator similar to this setting will be called *sawtooth configuration*. Using this configuration for every `v` generator, we find that there exists a CNOT ordering for each generator such that all possible (non-equivalent) Z-type errors from all circuits can be distinguished.

An example of the CNOT orderings which give a distinguishable fault set can be represented by the diagram in Fig. 6.3. The diagram looks similar to the 2D color code on the center plane, thus all `f` generators are displayed. The meanings of the diagram are as follows:

1. Each arrow represents the ordering of data CNOTs for each `f` generator: the qubits on which data CNOTs act start from the qubit at the tail of an arrow, then proceed counterclockwise.

2. The ordering of data CNOTs for each `v` generator can be obtained from its corresponding `f` generator using the sawtooth configuration.

3. The ordering of data CNOTs for the `cap` generator is in numerical order.

From the diagram, the exact orderings of data CNOTs for `f`, `v`, and `cap` generators are,

1. `f` generators: (2,5,3,1), (3,6,4,1), and (4,7,2,1).

2. `v` generators: (2,9,5,12,3,10,1,8), (3,10,6,13,4,11,1,8), and (4,11,7,14,2,9,1,8).

3. `cap` generator: (0,1,2,3,4,5,6,7).

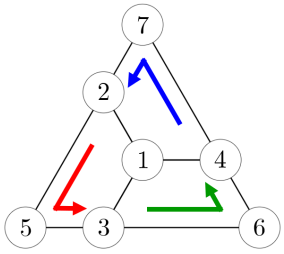


Figure 6.3: An example of the orderings of CNOT gates for the 3D color code of distance 3 in H form which give a distinguishable fault set $\mathcal{F}_1$. For each `f` generator, the qubits on which data CNOT gates act start from the tail of each arrow, then proceed counter-clockwise. The ordering of CNOT gates for the `cap` generator is determined by the qubit numbering.

(Please note that the CNOT orderings displayed here are not the only orderings which give a distinguishable fault set.)

Possible Z-type errors of weight greater than 1 depend heavily on the ordering of CNOT gates in the circuits for measuring Z-type generators. The exhaustive list of all possible Z-type errors arising from 1 fault and their syndrome corresponding to the CNOT orderings in Fig. 6.3 is given in Table 6.1. From the list, we find that any pair of possible Z-type errors either have different syndromes or are logically equivalent.

Since X-type and Z-type generators have the same form, this result is also applicable to the case of X-type errors. In general, a single fault in any circuit can cause an error of mixed types. However, note that a single fault in a circuit for measuring a Z-type generator cannot cause an X-type error of weight > 1 (and vice versa), and X-type and Z-type errors can be detected and corrected separately. Therefore, our results for X-type and Z-type errors implies that all fault combinations arising from up to 1 fault satisfy the condition in Definition 3.3. This means that $\mathcal{F}_1$ is distinguishable, and the protocols in Section 6.3 will be applicable. Since the circuits for measuring generators of the 3D color code are non-flag circuits, only one ancilla is required in each protocol (assuming that the qubit preparation and measurement are fast and the ancilla can be reused).

In the next section, we will generalize our technique to a family of capped color codes. A capped color code whose properties are similar to the 3D color code of distance 3 will be defined in Section 6.2.1, and the construction of circuits for measuring its generators will be discussed in Section 6.2.2.

Fault origin	Error	Syndrome $(u, \vec{v}, \vec{w})$			Fault origin	Error	Syndrome $(u, \vec{v}, \vec{w})$		
		u	$\vec{v}$	$\vec{w}$			u	$\vec{v}$	$\vec{w}$
q_0	Z_0	1	(0,0,0)	(0,0,0)	v_0^z	Z_0	1	(0,0,0)	(0,0,0)
q_1	Z_1	1	(1,1,1)	(1,1,1)		Z_0Z_1	0	(1,1,1)	(1,1,1)
q_2	Z_2	1	(1,0,1)	(1,0,1)		$Z_0Z_1Z_2$	1	(0,1,0)	(0,1,0)
q_3	Z_3	1	(1,1,0)	(1,1,0)		$Z_0Z_1Z_2Z_3$	0	(1,0,0)	(1,0,0)
q_4	Z_4	1	(0,1,1)	(0,1,1)		$Z_5Z_6Z_7$	1	(1,1,1)	(1,1,1)
q_5	Z_5	1	(1,0,0)	(1,0,0)		Z_6Z_7	0	(0,1,1)	(0,1,1)
q_6	Z_6	1	(0,1,0)	(0,1,0)		Z_7	1	(0,0,1)	(0,0,1)
q_7	Z_7	1	(0,0,1)	(0,0,1)	v_1^z	Z_2	1	(1,0,1)	(1,0,1)
q_8	Z_8	0	(0,0,0)	(1,1,1)		Z_2Z_9	1	(1,0,1)	(0,0,0)
q_9	Z_9	0	(0,0,0)	(1,0,1)		$Z_2Z_9Z_5$	0	(0,0,1)	(1,0,0)
q_{10}	Z_{10}	0	(0,0,0)	(1,1,0)		$Z_2Z_9Z_5Z_{12}$	0	(0,0,1)	(0,0,0)
q_{11}	Z_{11}	0	(0,0,0)	(0,1,1)		$Z_{10}Z_1Z_8$	1	(1,1,1)	(1,1,0)
q_{12}	Z_{12}	0	(0,0,0)	(1,0,0)		Z_1Z_8	1	(1,1,1)	(0,0,0)
q_{13}	Z_{13}	0	(0,0,0)	(0,1,0)		Z_8	0	(0,0,0)	(1,1,1)
q_{14}	Z_{14}	0	(0,0,0)	(0,0,1)	v_2^z	Z_3	1	(1,1,0)	(1,1,0)
f_4^z	Z_2	1	(1,0,1)	(1,0,1)		Z_3Z_{10}	1	(1,1,0)	(0,0,0)
	Z_2Z_5	0	(0,0,1)	(0,0,1)		$Z_3Z_{10}Z_6$	0	(1,0,0)	(0,1,0)
	Z_1	1	(1,1,1)	(1,1,1)		$Z_3Z_{10}Z_6Z_{13}$	0	(1,0,0)	(0,0,0)
f_5^z	Z_3	1	(1,1,0)	(1,1,0)		$Z_{11}Z_1Z_8$	1	(1,1,1)	(0,1,1)
	Z_3Z_6	0	(1,0,0)	(1,0,0)		Z_1Z_8	1	(1,1,1)	(0,0,0)
	Z_1	1	(1,1,1)	(1,1,1)		Z_8	0	(0,0,0)	(1,1,1)
f_6^z	Z_4	1	(0,1,1)	(0,1,1)	v_3^z	Z_4	1	(0,1,1)	(0,1,1)
	Z_4Z_7	0	(0,1,0)	(0,1,0)		Z_4Z_{11}	1	(0,1,1)	(0,0,0)
	Z_1	1	(1,1,1)	(1,1,1)		$Z_4Z_{11}Z_7$	0	(0,1,0)	(0,0,1)
						$Z_4Z_{11}Z_7Z_{14}$	0	(0,1,0)	(0,0,0)
						$Z_9Z_1Z_8$	1	(1,1,1)	(1,0,1)
						Z_1Z_8	1	(1,1,1)	(0,0,0)
						Z_8	0	(0,0,0)	(1,1,1)

Table 6.1: The exhaustive list of all possible Z-type errors arising from 1 fault and their syndrome corresponding to the CNOT orderings in Fig. 6.3. Any pair of possible Z-type errors on the list either have different syndromes or are logically equivalent.

6.2 Syndrome measurement circuits for a capped color code

In the previous section, we have seen that it is possible to construct circuits for the 3D color code of distance 3 such that the fault set is distinguishable. In this section, we will extend our construction ideas to quantum codes of higher distance. First, we will introduce a new family of codes called capped color codes, whose properties are similar to those of the 3D color codes, but the structures of the capped color codes of higher distance are more suitable for our construction rather than those of the 3D color codes of higher distance (as defined in [Bom15a]). Afterwards, we will apply the error correction ideas using weight parities from the previous section and develop the main theorem of this work, which can help us find proper CNOT orderings for a capped color code of any distance.

6.2.1 Capped color codes

We begin by defining some notations for the 2D color codes [BMD06] and stating some code properties. A 2D color code of distance d ($d = 3, 5, 7, \dots$) is an $[\![n_{2\mathrm{D}}, 1, d]\!]$ CSS code where $n_{2\mathrm{D}} = (3d^2 + 1)/4$. The number of stabilizer generators of each type is $r = (n_{2\mathrm{D}} - 1)/2$ (note that the total number of generators is $2r$). For any 2D color code, it is possible to choose generators so that those of each type (X or Z) are 3-colorable. The three smallest 2D color codes are shown in Fig. 6.4.

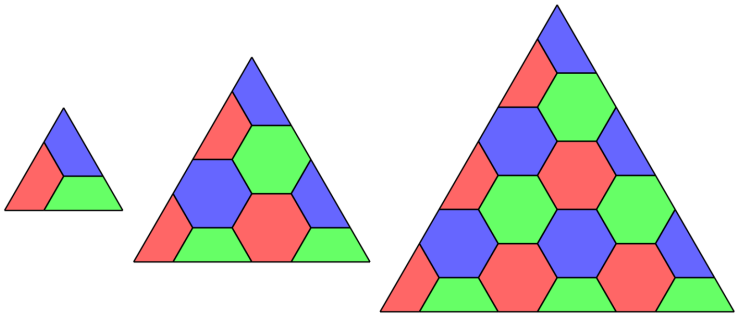


Figure 6.4: 2D color codes of distance 3, 5, and 7.

A 2D color code of any distance has the following properties [KB15]:

1. the number of qubits $n_{2\mathrm{D}}$ is odd,
2. every generator has even weight,
3. the code encodes 1 logical qubit,
4. logical X and logical Z operators are of the form $X^{\otimes n_{2\mathrm{D}}}M$ and $Z^{\otimes n_{2\mathrm{D}}}N$, where M, N are some stabilizers,
5. the set of physical qubits of a 2D color code is bipartite.

With properties 1-4, we can see that Lemma 3.1 is applicable to a 2D color code of any distance.

A *capped color code* $CCC(d)$ is constructed from 2 layers of the 2D color code of distance d plus one qubit. Thus, the number of qubits of $CCC(d)$ is $2n_{2\mathrm{D}}+1 = 3(d^2+1)/2$. Examples of capped color codes with $d = 5$ and 7 are displayed in Fig. 6.5a, and their dual lattices are shown in Fig. 6.5b. Let $\mathtt{q}_i$ denote qubit i. For convenience, we will divide each code into 3 areas: the *top qubit* (consisting of $\mathtt{q}_0$), the *center plane* (consisting of $\mathtt{q}_1$ to $\mathtt{q}_{n_{2\mathrm{D}}}$), and the *bottom plane* (consisting of $\mathtt{q}_{n_{2\mathrm{D}}+1}$ to $\mathtt{q}_{2n_{2\mathrm{D}}}$). We will primarily use the center plane as a reference, and sometimes call the qubits on the center plane *on-plane qubits* and call the qubits on the bottom plane *off-plane qubits*. Note that the set of physical qubits of $CCC(d)$ is also bipartite (as colored in black and white in Fig. 6.5a) since the set of physical qubits of any 2D color code is bipartite.

A capped color code $CCC(d)$ is a CSS subsystem code [Pou05, Bac06]. Its stabilizer generators are volume operators which can be defined as follows:

1. v_0^x and v_0^z are X-type and Z-type operators that cover $\mathtt{q}_0$ and all qubits on the center plane. These operators are called `cap` generators; and
2. $v_1^x, \ldots, v_r^x$ and $v_1^z, \ldots, v_r^z$ are X-type and Z-type operators in which each v_i^x (or v_i^z) acts as an X-type (or a Z-type) generator of the 2D color code on both center and bottom planes. These operators are called `v` generators.

The stabilizer generators of a capped color code are illustrated in Fig. 6.5c. Using these notations, the stabilizer group of the code is

$$S_{\mathrm{CCC}} = \langle v_0^x, v_1^x, \ldots, v_r^x, v_0^z, v_1^z, \ldots, v_r^z \rangle. \tag{6.3}$$

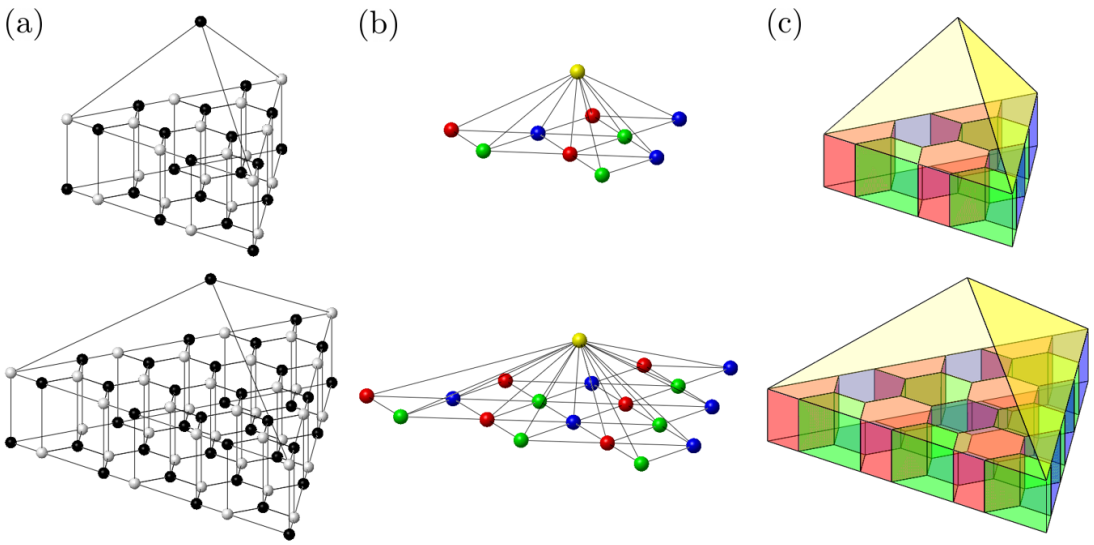


Figure 6.5: Capped color codes $CCC(d)$ with $d = 5$ (top) and $d = 7$ (bottom). (a) The set of qubits of any capped color code is bipartite, as displayed by black and white vertices. (b) The dual lattice of each capped color code. (c) Stabilizer generators of each code can be illustrated by volume operators.

For each $CCC(d)$, the generators of the gauge group are face operators which can be defined as follows:

1. $f_1^x, \ldots, f_r^x$ are X-type operators in which each operator acts as an X-type generator of the 2D color code on the center plane.

2. $f_{r+1}^z, \ldots, f_{2r}^z$ are Z-type operators in which each operator acts as a Z-type generator of the 2D color code on the center plane, and f_i^x and f_{r+i}^z $(i = 1, \ldots, r)$ act on the same set of qubits.

3. $f_1^z, \ldots, f_r^z$ and $f_{r+1}^x, \ldots, f_{2r}^x$ are Z-type and X-type operators that satisfy the following conditions:

   (a) f_i^x and f_j^z anticommute when $i = j$ $(i, j = 1, \ldots, 2r)$,

   (b) f_i^x and f_j^z commute when $i \neq j$ $(i, j = 1, \ldots, 2r)$,

   (c) f_i^z and f_{r+i}^x $(i = 1, \ldots, r)$ act on the same set of qubits.

With these notations, the gauge group of each $CCC(d)$ is,

$$G_{\text{CCC}} = \langle v_i^x, v_i^z, f_j^x, f_j^z \rangle, \tag{6.4}$$

where $i = 0, 1, \ldots, r$ and $j = 1, \ldots, 2r$.

It should be noted that in this work, the term "color code" is used to describe a subsystem code satisfying two conditions proposed in [KB15]. This may be different from common usages in other literature in which the term refers to a stabilizer code. A capped color code is actually a color code in 3 dimensions since the dual lattice of the code (see Fig. 6.5b for examples) is 4-colorable and can be constructed by attaching tetrahedra together (see [KB15] for more details). However, the capped color code and the 3D color code defined in [Bom15a] are different codes.

A capped color code is a subsystem code which encodes 1 logical qubit, meaning that there are n_{2D} gauge qubits for each $CCC(d)$. We can clearly see that $CCC(3)$ is exactly the 3D color code of distance 3 discussed in Section 6.1.1. Similarly, a stabilizer code encoding 1 logical qubit can be obtained from $CCC(d)$ by choosing n_{2D} independent, commuting gauge operators and including them in the stabilizer group. This work will discuss two possible ways to do so, and the resulting codes will be called the code in H form and the code in T form (similar to the case of the 3D color code of distance 3).

Capped color codes in H form

Observe that the center plane of $CCC(d)$ which covers qubits 1 to n_{2D} looks exactly like the 2D color code of distance d. The stabilizer group of the 2D color code is $S_{\text{2D}} = \langle f_1^x, \ldots, f_r^x, f_{r+1}^z, \ldots, f_{2r}^z \rangle$. A capped color code in H form constructed from $CCC(d)$ can be obtained by adding the stabilizer generators of the 2D color code to the original generating set of $CCC(d)$. Thus, the stabilizer group of the code in H form is,

$$S_{\text{H}} = \langle v_i^x, f_j^x, v_i^z, f_k^z \rangle, \tag{6.5}$$

where $i = 0, 1, \ldots, r$, $j = 1, 2, \ldots, r$, and $k = r+1, r+2, \ldots, 2r$. Logical X and logical Z operators of this code are of the form $X^{\otimes n}M$ and $Z^{\otimes n}N$, where M, N are some stabilizers in S_{H}. Note that Lemma 3.1 is applicable to the code in H form constructed from any $CCC(d)$.

The code in H form is a code of distance d. This can be proved as follows:

Proposition 6.1 *The capped color code in H form constructed from $CCC(d)$ has distance d.*

Proof:
In order to prove that the distance of a stabilizer code is d, we will show that the weight of a nontrivial logical operator is at least d; that is, any Pauli error of weight $< d$ is either a stabilizer or an error with a nontrivial syndrome, and there exists a nontrivial logical operator of weight exactly d. Since the capped color code in H form is a CSS code and X-type and Z-type generators have the same form, we can consider X-type and Z-type errors separately. For an error occurred on the code in H form, we will represent its weight by a triple (a, b, c) where a, b, c are the weights of the errors occurred on the top qubit, the center plane, and the bottom plane, respectively.

Suppose that a Z-type error has weight $k < d$. The weight of such an error will be of the form (a, b, c) with $a = 0$ and $b + c = k$, or with $a = 1$ and $b + c = k - 1$. Observe that the stabilizer generators of the 2D color code on the center plane (which is a subcode of the capped color code in H form) are $f_1^x, \ldots, f_r^x$ and $f_{r+1}^z, \ldots, f_{2r}^z$. Moreover, the 2D color code on the bottom plane is also a subcode of the capped color code in H form, whose stabilizer generators are $f_1^x \cdot v_1^x, \ldots, f_r^x \cdot v_r^x$ and $f_{r+1}^z \cdot v_1^z, \ldots, f_{2r}^z \cdot v_r^z$ (the syndrome obtained by measuring $\mathtt{v}$ generators is the sum of the syndromes obtained from the 2D color codes on both planes). Since both 2D color codes on the center and the bottom planes have distance d, any Z-type error of weight $< d$ which occurs solely on the center or the bottom plane either has nontrivial syndrome or acts as a stabilizer on such a plane. From the possible forms of error, a Z-type error of weight $< d$ on the capped color code in H form corresponding to the trivial syndrome must act as a stabilizer on both planes and commute with v_0^x. Using Lemma 3.1, the weight of such an operator must be in the form $(0, b, c)$ where b, c are even numbers. So the total weight of the error is even, and it cannot be a logical Z operator (by Lemma 3.1). Therefore, any Z-type error of weight $< d$ is either a stabilizer or an error with a nontrivial syndrome. The same analysis is applicable to X-type errors of weight $< d$.

Next, we will show that there exists a logical Z operator of weight exactly d. Consider a Z-type operator whose weight is of the form $(0, 0, d)$ and acts as a logical Z operator on the 2D color code on bottom plane (the operator exists because the 2D color code has

distance d). Such an operator commutes with all generators of the capped color code in H form and has odd weight. By Lemma 3.1, this operator is a logical Z operator. The proof is now completed. □

The capped color code in H form constructed from $CCC(d)$ is an $[\![n, 1, d]\!]$ code where $n = 2n_{2\mathrm{D}} + 1$. Similar to the 3D color code of distance 3 in H form, it is not hard to verify that Hadamard, S, and CNOT gates are transversal; their logical gates are $\bar{H} = H^{\otimes n}$, $\bar{S} = (S^\dagger)^{\otimes n}$, and $\overline{\mathrm{CNOT}} = \mathrm{CNOT}^{\otimes n}$.

It should be noted that there are many other choices of stabilizer generators that can give the same code as what is constructed here. However, different choices of generators can give different fault sets, which may or may not be distinguishable. In Section 6.2.2, we will only discuss circuits for measuring generators corresponding to Eq. (6.5).

Capped color codes in T form

A capped color code in T form is constructed from $CCC(d)$ by adding all Z-type 4-body face generators to the old generating set of $CCC(d)$. That is, the stabilizer group of the code in T form is

$$S_{\mathrm{T}} = \langle v_i^x, v_i^z, f_j^z \rangle, \tag{6.6}$$

where $i = 0, 1, \ldots, r$ and $j = 1, 2, \ldots, 2r$. Similar to the code in H form, logical X and logical Z operators of this code are of the form $X^{\otimes n}M$ and $Z^{\otimes n}N$, where M, N are some stabilizers in S_{T}. Note that Lemma 3.1 is also applicable to the code in T form constructed for any $CCC(d)$.

Unlike the code in H form, the capped color code in T form constructed from $CCC(d)$ is a code of distance 3 regardless of the parameter d, i.e., it is an $[\![n, 1, 3]\!]$ code where $n = 2n_{2\mathrm{D}} + 1$. The proof of the code distance is as follows:

Proposition 6.2 *The capped color code in T form constructed from $CCC(d)$ has distance 3.*

Proof:
Similar to the proof of Proposition 6.1, we will show that (1) any Pauli error of weight < 3 is either a stabilizer or an error with a nontrivial syndrome, and (2) there exists a nontrivial logical operator of weight exactly 3. However, for the capped color code in T form, X-type

and Z-type generators have different forms, so we have to analyze both types of errors. Observe that all of the Z-type generators of the code in H form are also Z-type generators of the code in T form, thus we can use the analysis in the proof of Proposition 6.1 to show that any X-type error of weight $< d$ is either a stabilizer or an error with a nontrivial syndrome. Thus, we only have to show that any Z-type error of weight < 3 is either a stabilizer or an error with a nontrivial syndrome, and there exists a logical Z operator of weight exactly 3. Similar to the proof of Proposition 6.1, we will represent its weight by a triple (a, b, c) where a, b, c are the weights of the errors occurred on the top qubit, the center plane, and the bottom plane, respectively.

The X-type generators of the capped color code in T form are $v_0^x, v_1^x, \ldots, v_r^x$. First, let us consider any Z-type error of weight 1. We can easily verify that the error anticommutes with at least one X-type generator, so its syndrome is nontrivial. Next, consider a Z-type error of weight 2. The weight of the error will have one of the following forms: $(0, 2, 0), (0, 1, 1), (0, 0, 2), (1, 1, 0)$, or $(1, 0, 1)$. We find that (1) a Z-type error of the form $(0, 1, 1)$ or $(1, 0, 1)$ anticommutes with v_0^x, and (2) a Z-type errors of the form $(0, 2, 0), (0, 0, 2)$, or $(1, 1, 0)$ anticommutes with at least one `v` generator (since `v` generators act as generators of the 2D color code on both planes simultaneously, and the 2D color code has distance d). Therefore, the syndrome of any Z-type error of weight 2 is nontrivial.

Next, we will show that there exists a logical Z operator of distance exactly 3. Consider a Z-type operator of weight 3 of the form $Z_0 Z_i Z_{r+i}$, where $i = 1, 2, \ldots, r$. We can verify that such an operator commutes with all X-type generators. Since the operator has odd weight, it is a logical Z operator by Lemma 3.1. □

CNOT and T gates are transversal for the code in T form, while Hadamard and S gates are not. In order to prove the transversalily of the T gate, we will use the following lemma [KB15]:

Lemma 6.1 *Let C be an $[\![n, k, d]\!]$ CSS subsystem code in which n is odd, k is 1, and $X^{\otimes n}$ and $Z^{\otimes n}$ are bare logical X and Z operators*[1]*. Also, let Q be the set of all physical qubits of C, and let p be any positive integer. Suppose there exists $V \subset Q$ such that for any $m = 1, \ldots, p$, for every subset $\{g_1^x, \ldots, g_m^x\}$ of the X-type gauge generators of the code, the*

[1] A bare logical operator is a logical operator that acts on the logical qubit(s) of a subsystem code and does not affect the gauge qubit(s); see [Pou05, Bac06, Bra11].

following holds:

$$\left|V \cap \bigcap_{i=1}^{m} G_i\right| = \left|V^c \cap \bigcap_{i=1}^{m} G_i\right| \mod 2^{p-m+1}, \tag{6.7}$$

where G_i is the set of physical qubits that support g_i^x. Then, a logical R_p gate (denoted by $\bar{R}_p$) can be implemented by applying R_p^q to all qubits in V and applying R_p^{-q} to all qubits in V^c, where $R_p = \mathrm{diag}\,(1, \exp\,(2\pi i/p))$, q is a solution to $q(|V| - |V^c|) = 1 \mod 2^p$, and $V^c = Q \backslash V$.

The proof of the transversality of the T gate is as follows:

Proposition 6.3 *A T gate is transversal for the capped color code in T form constructed from any $CCC(d)$.*

Proof:
Let C be the capped color code in T form constructed from any $CCC(d)$ (C is a stabilizer code, i.e., it is a subsystem code in which the stabilizer group and the gauge group are the same). Note that the X-type stabilizer generators of the code are $v_0^x, v_1^x, \ldots, v_r^x$, which are also the X-type gauge generators. Also, let $p = 3$ (since $T = R_3$), $q = 1$, and let V and V^c be the sets of qubits similar to those represented by black and white vertices in Fig. 6.5a (this kind of representation is always possible for any $CCC(d)$ since the set of physical qubits of $CCC(d)$ is bipartite). We will use Lemma 6.1 and show that Eq. (6.7) is satisfied for $m = 1, 2, 3$.

Let G_i be the set of qubits that support X-type generator g_i^x. If $m = 1$, we can easily verify that $|V \cap G_1| = |V^c \cap G_1| \mod 8$ for every $g_1^x \in \{v_0^x, v_1^x, \ldots, v_r^x\}$ since half of supporting qubits of any X-type generator is in V and the other half is in V^c.

In the case when $m = 2$, let $\{g_1^x, g_2^x\}$ be a subset of $\{v_0^x, v_1^x, \ldots, v_r^x\}$. If g_1^x is a `cap` generator v_0^x and g_2^x is a `v` generator $v_i^x, i = 1, \ldots, r$, then $G_1 \cap G_2$ are the qubits that support the face generator f_i^x. Since half of qubits in $G_1 \cap G_2$ is in V and the other half is in V^c, we have that $|V \cap G_1 \cap G_2| = |V^c \cap G_1 \cap G_2|$ (equal to 2 or 3, depending on v_i^x). If g_1^x and g_2^x are adjacent `v` generators, then $G_1 \cap G_2$ have 4 qubits, two of them are in V and the other two are in V^c. So $|V \cap G_1 \cap G_2| = |V^c \cap G_1 \cap G_2| = 2$. If g_1^x and g_2^x are non-adjacent `v` generators, then $|V \cap G_1 \cap G_2| = |V^c \cap G_1 \cap G_2| = 0$. Therefore, Eq. (6.7) is satisfied for any subset $\{g_1^x, g_2^x\}$.

In the case when $m = 3$, let $\{g_1^x, g_2^x, g_3^x\}$ be a subset of $\{v_0^x, v_1^x, \ldots, v_r^x\}$. If g_1^x is a `cap` generator v_0^x and g_2^x, g_3^x are adjacent `v` generators, or g_1^x, g_2^x, g_3^x are `v` generators in which any two of them are adjacent, then $G_1 \cap G_2 \cap G_3$ have 2 qubits, one of them is in V and the other one is in V^c. Thus, $|V \cap G_1 \cap G_2 \cap G_3| = |V^c \cap G_1 \cap G_2 \cap G_3| = 1$. If g_1^x is a `cap` generator v_0^x and g_2^x, g_3^x are non-adjacent `v` generators, or g_1^x, g_2^x, g_3^x are `v` generators in which some pair of them are not adjacent, then $G_1 \cap G_2 \cap G_3$ is the empty set. So $|V \cap G_1 \cap G_2 \cap G_3| = |V^c \cap G_1 \cap G_2 \cap G_3| = 0$. Therefore, Eq. (6.7) is satisfied for any subset $\{g_1^x, g_2^x, g_3^x\}$.

Since the sufficient condition in Lemma 6.1 is satisfied, a transversal T gate can be implemented by applying T gates to all qubits in V (represented by black vertices) and applying $T^\dagger$ gates to all qubits in V^c (represented by white vertices). □

Incidentally, the capped color codes in T form presented here are similar to some codes that appear in other literature. In fact, the capped color codes in T form is the same as the stacked codes with distance 3 protection defined in [JOB16] (where alternative proofs of Propositions 6.2 and 6.3 are also presented). Such a code is the basis for the construction of the $(d-1)+1$ stacked code defined in the same work, whose code distance is d (see also [BC15, JBH16] for other subsystem codes with similar construction).

Code switching

Similar to the 3D color code of distance 3, one can transform between the capped color code in H form and the code in T form derived from the same $CCC(d)$ using the code switching technique [PR13, ADCP14, Bom15a, KB15]. Suppose that we start from the code in H form. The code switching can be done by first measuring $f_1^z, \ldots, f_r^z$, then applying an X-type Pauli operator that

1. commutes with all v_i^x's and v_i^z's $(i = 0, 1, \ldots, r)$, and
2. commutes with $f_{r+1}^z, \ldots, f_{2r}^z$, and
3. for each $j = 1, \ldots, r$, commutes with f_j^z if the outcome from measuring such an operator is 0 (the eigenvalue is $+1$) or anticommutes with f_j^z if the outcome is 1 (the eigenvalue is -1).

We can use a similar process to switch from the code in T form to the code in H form,

except that $f_1^x, \ldots, f_r^x$ will be measured and the operator to be applied is a Z-type Pauli operator that commutes or anticommutes with $f_1^x, \ldots, f_r^x$ (depending on the outcomes).

We have not yet discussed whether the procedure above is fault tolerant when we switch between the capped color codes in H form and T form. However, the discussion of the fault-tolerant implementation of T gate will be deferred until Section 6.3.3.

6.2.2 Circuit configuration for a capped color code

One of the main goals of [TL21a] is to find circuits for measuring generators of a capped color code in H form in which the corresponding fault set $\mathcal{F}_t$ is distinguishable (where $t = \tau = (d-1)/2$ and $d = 3, 5, 7, ...$ is the code distance). As discussed before, the CNOT orderings and the number of flag ancillas are crucial for the circuit design. Finding such circuits for a capped code of any distance using a random approach can be very challenging because of a few reasons: (1) the number of stabilizer generators of a capped color code increases quadratically as the distance increases. This means that the number of possible single faults in the circuits grow quadratically as well. (2) for a code with larger distance, a fault set $\mathcal{F}_t$ with larger t will be considered. Since it concerns all possible fault combinations arising from up to t faults, the size of $\mathcal{F}_t$ grows dramatically (perhaps exponentially) as t and the number of possible single faults increase. For these reasons, verifying whether $\mathcal{F}_t$ is distinguishable using the conditions in Definition 3.3 requires a lot of computational resources, and exhaustive search for appropriate CNOT orderings may turn intractable.

Fortunately, there is a way to simplify the search for the CNOT orderings. From the structure of the capped color code in H form, it is possible to relate CNOT orderings for the 3D-like generators to those for the 2D-like generators, as we have seen in the circuit construction in Section 6.1.2. Instead of finding CNOT orderings directly for all generators, we will simplify the problem and develop sufficient conditions for the CNOT orderings of the 2D-like generators which, if satisfied, can guarantee that the fault set $\mathcal{F}_t$ (which concerns both 3D-like and 2D-like generators) is distinguishable. Although we still need to check whether the sufficient conditions are satisfied for given CNOT orderings, the process is much simpler than checking the conditions in Definition 3.3 directly when the size of $\mathcal{F}_t$ is large.

We begin by dividing the stabilizer generators of the capped color code in H form constructed from $CCC(d)$ into 3 categories (similar to the discussion in Section 6.1.2):

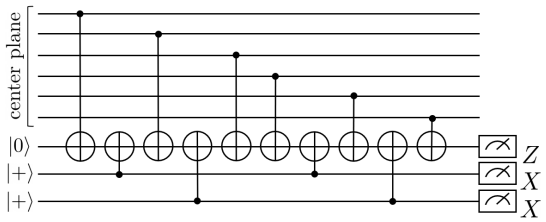
1. `cap` generators consisting of v_0^x and v_0^z,
2. `v` generators consisting of $v_1^x, \ldots, v_r^x$, and $v_1^z, \ldots, v_r^z$,
3. `f` generators consisting of $f_1^x, \ldots, f_r^x$, and $f_{r+1}^z, \ldots, f_{2r}^z$.

Here we will only consider fault combinations arising from circuits for measuring Z-type generators which can lead to purely Z-type data errors of any weight. This is because i faults in circuits for measuring X-type generators cannot cause Z-type data error of weight greater than i (and vice versa). Similar analysis will be applicable to the case of purely X-type errors, and also the case of mixed-type errors. We will first consider a Z-type data error and a flag vector arising from each single fault. Afterwards, fault combinations constructed from multiple faults will be considered, where the combined data error and the cumulative flag vector for each fault combination can be calculated using Eqs. (3.5) and (3.6).

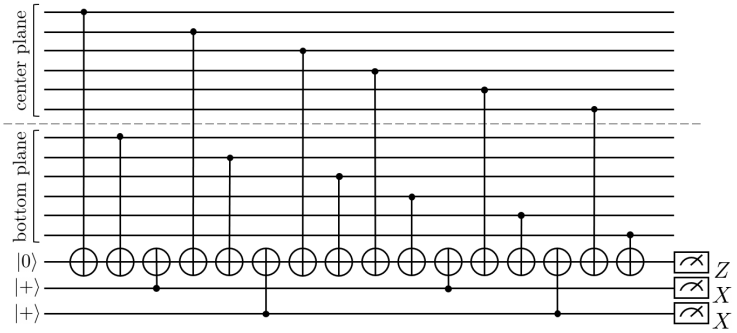
Observe that the center plane of a capped color code behaves like a 2D color code, and the weight of a Z-type error occurred on the center plane can be measured by the `cap` generator v_0^x. In order to find CNOT orderings for generators of each category, we will use an idea similar to that presented in Section 6.1.2; we will try to design circuits for measuring Z-type generators so that most of possible Z-type errors arising from a single fault are on the center plane. In this work, we will start by imposing general configurations of data and flag CNOT gates; these general configurations will facilitate finding CNOT orderings. Then, exact configurations of CNOT gates which can make $\mathcal{F}_t$ distinguishable will be found using the theorem developed later in this section. The general configurations of data CNOT gates, which depend on the category of the generator, are as follows:

General configurations of data CNOT gates

1. `f` generator: there is no constraint for the ordering of data CNOTs since each `f` generator lies on the center plane, but the ordering for f_{r+i}^z (or f_i^x) must be related to the ordering for v_i^z (or v_i^x) where $i = 1, \ldots, r$.
2. `v` generator: The *sawtooth configuration* will be used; the qubits on which the data CNOTs act must be alternated between on-plane and off-plane qubits. The ordering of data CNOTs for v_i^z (or v_i^x) is referenced by the ordering of data CNOTs for f_{r+i}^z (or f_i^x) where $i = 1, \ldots, r$ (see examples in Fig. 6.6 and Section 6.1.2).



(a)



(b)

Figure 6.6: (a) An example of flag circuit for measuring f generator with two flag ancillas. (b) A flag circuit for measuring the corresponding v generator. The circuit is obtained by replacing each data CNOT which couples $\mathtt{q}_j$ with the syndrome ancilla by two data CNOTs which couple $\mathtt{q}_j$ and $\mathtt{q}_{n_{2\mathrm{D}}+j}$ with the syndrome ancilla.

3. `cap` generator: The first data CNOT must always be the one that couples $\mathtt{q}_0$ with the syndrome ancilla. The ordering of the other data CNOTs has yet to be fixed.

In general, some flag ancillas may be added to the circuits for measuring a generator to help distinguish some possible errors and make $\mathcal{F}_t$ distinguishable. In that case, the general configurations for data CNOT gates will also be applied to the data CNOTs in each flag circuit. Moreover, additional configurations for flag CNOT gates will be required.

General configurations of flag CNOT gates

1. For each flag circuit, the first and the last data CNOTs must not be in between any pair of flag CNOT gates.

2. The arrangements of flag CNOTs in the circuits for each pair of `f` and `v` generators must be similar; Suppose that a flag circuit for f^z_{r+i} (or f^x_i) where $i = 1, \ldots, r$ is

given. A flag circuit for v_i^z (or v_i^x) is obtained by replacing each data CNOT which couples $\mathtt{q}_j$ with the syndrome ancilla ($j = 1, \ldots, n_{2\mathrm{D}}$) by two data CNOTs which couple $\mathtt{q}_j$ and $\mathtt{q}_{n_{2\mathrm{D}}+j}$ with the syndrome ancilla; see an example in Fig. 6.6.

By imposing the general configurations for data and flag CNOTs, what have yet to be determined before $\mathcal{F}_t$ is specified are the ordering of data CNOTs for each f generator, the ordering of data CNOTs after the first data CNOT for each `cap` generator, and the number of flag ancillas and the ordering of their relevant flag CNOTs. (Note that having more flag ancillas can make fault distinguishing become easier, but more resources such as qubits and gates are also required.)

In this work, possible single faults which can give Z-type errors will be divided into 7 types (based on relevant faulty locations) as follows:

1. Type $\mathtt{q}_0$: a fault causing a Z-type error on $\mathtt{q}_0$ which does not arise from any Z-type generator measurement. The total number of $\mathtt{q}_0$ faults is n_0 (which is 0 or 1).

2. Type $\mathtt{q}_{\mathtt{on}}$: a fault causing a single-qubit Z-type error on the center plane which does not arise from any Z-type generator measurement. The syndrome of an error is denoted by $\vec{q}_{\mathtt{on}}$. The total number of $\mathtt{q}_{\mathtt{on}}$ faults is $n_{\mathtt{on}}$.

3. Type $\mathtt{q}_{\mathtt{off}}$: a fault causing a single-qubit Z-type error on the bottom plane which does not arise from any Z-type generator measurement. The syndrome of an error is denoted by $\vec{q}_{\mathtt{off}}$. The total number of $\mathtt{q}_{\mathtt{off}}$ faults is $n_{\mathtt{off}}$.

4. Type `f`: a fault occurred during a measurement of a `f` generator of Z type. A Z-type error from each fault of this type and its syndrome are denoted by $\sigma_{\mathtt{f}}$ and $\vec{p}_{\mathtt{f}}$. A flag vector corresponding to each fault of this type is denoted by $\vec{f}_{\mathtt{f}}$. The total number of `f` faults is $n_{\mathtt{f}}$.

5. Type `v`: a fault occurred during a measurement of a `v` generator of Z type which give errors of the same form on both center and bottom planes (see an example in Fig. 6.7). A part of a Z-type error from each fault of this type occurred on the center plane only (or the bottom plane only) and its syndrome are denoted by $\sigma_{\mathtt{v}}$ and $\vec{p}_{\mathtt{v}}$. A flag vector corresponding to each fault of this type is denoted by $\vec{f}_{\mathtt{v}}$. The total number of `v` faults is $n_{\mathtt{v}}$.

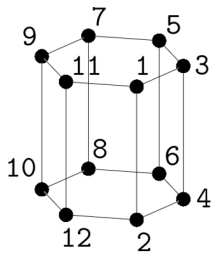


Figure 6.7: Consider a circuit for measuring a $\mathtt{v}$ generator of Z type in which its supporting qubits are labeled as displayed above and the ordering of data CNOT gates is $(1, 2, \ldots, 12)$. A single fault in the circuit is either $\mathtt{v}$ type or $\mathtt{v}^*$ type, depending on whether the data errors on the center and the bottom planes have the same form. For example, an IZ fault on the 7th data CNOT is a $\mathtt{v}^*$ fault since the data error arising from the fault is $Z_9 Z_{11} \otimes Z_8 Z_{10} Z_{12}$, while an IZ fault on the 8th data CNOT is a $\mathtt{v}$ fault since the data error arising from the fault is $Z_9 Z_{11} \otimes Z_{10} Z_{12}$.

6. Type $\mathtt{v}^*$: a fault occurred during a measurement of a $\mathtt{v}$ generator of Z-type in which an error occurred on the center plane and an error on the bottom plane are different (see an example in Fig. 6.7). A part of a Z-type error from each fault of this type occurred on the center plane only and its syndrome are denoted by $\sigma_{\mathtt{v}^*,\mathtt{cen}}$ and $\vec{p}_{\mathtt{v}^*,\mathtt{cen}}$. The other part of the Z-type error that occurred on the bottom plane only and its syndrome are denoted by $\sigma_{\mathtt{v}^*,\mathtt{bot}}$ and $\vec{p}_{\mathtt{v}^*,\mathtt{bot}}$. A flag vector corresponding to each fault of this type is denoted by $\vec{f}_{\mathtt{v}^*}$. The total number of $\mathtt{v}^*$ faults is $n_{\mathtt{v}^*}$.

7. Type $\mathtt{cap}$: a fault occurred during a measurement of a $\mathtt{cap}$ generator of Z type. A Z-type error from each fault of this type and its syndrome are denoted by $\sigma_{\mathtt{cap}}$ and $\vec{p}_{\mathtt{cap}}$ ($\sigma_{\mathtt{cap}}$ is always on the center plane up to a multiplication of the $\mathtt{cap}$ generator being measured). A flag vector corresponding to each fault of this type is denoted by $\vec{f}_{\mathtt{cap}}$. The total number of $\mathtt{cap}$ faults is $n_{\mathtt{cap}}$.

Examples of faults of each type on the 3D structure are illustrated in Fig. 6.8a on page 90. Note that a fault of $\mathtt{q}_0$, $\mathtt{q}_{\mathtt{on}}$, or $\mathtt{q}_{\mathtt{off}}$ type can be a Z-type input error, a single-qubit error from phase flip, or a single fault during any X-type generator measurement which gives a Z-type error.

Suppose that a single fault causes a Z-type data error E and a flag vector $\vec{f}$. The syndrome of E evaluated by X-type generators can be written as $(s_a, \vec{s}_b, \vec{s}_c)$, where $s_a, \vec{s}_b, \vec{s}_c$ are

syndromes obtained from measuring $\texttt{cap}, \texttt{f}$, and $\texttt{v}$ generators of X type. In addition, the flag vector can be written as $(\vec{f}_a, \vec{f}_b, \vec{f}_c)$, where $\vec{f}_a, \vec{f}_b, \vec{f}_c$ are flag outcomes obtained from circuits for measuring $\texttt{cap}, \texttt{f}$, and $\texttt{v}$ generators of Z type, respectively. (The lengths of $s_a, \vec{s}_b, \vec{s}_c$ are equal to the number of generators of each category, while the lengths of $\vec{f}_a, \vec{f}_b, \vec{f}_c$ are equal to the number of generators of each category times the number of flag ancillas in each flag circuit, assuming that all flag circuits have equal number of flag ancillas.) Let $\mathrm{wp}(\sigma)$ denote the weight parity of error σ. Due to the general configurations of CNOT gates being used, the weight parity and the syndromes of a Z-type error (evaluated by X-type generators) and a flag vector arising from each type of faults can be summarized as in Table 6.2. Note that for a $\texttt{v}^*$ fault, $\sigma_{\texttt{v}^*,\texttt{cen}}$ and $\sigma_{\texttt{v}^*,\texttt{bot}}$ differ by a Z error on a single qubit; i.e., $\mathrm{wp}(\sigma_{\texttt{v}^*,\texttt{cen}}) + \mathrm{wp}(\sigma_{\texttt{v}^*,\texttt{bot}}) = 1$. Sometimes we will write $\vec{p}_{\texttt{v}^*,\texttt{cen}} + \vec{p}_{\texttt{v}^*,\texttt{bot}} = \vec{q}_{\texttt{v}^*}$ to emphasize its similarity to the syndrome of a single-qubit error.

		Type of fault						
		$\texttt{q}_0$	$\texttt{q}_{\texttt{on}}$	$\texttt{q}_{\texttt{off}}$	$\texttt{f}$	$\texttt{v}$	$\texttt{v}^*$	$\texttt{cap}$
Syndrome	s_a ($\texttt{cap}$)	1	1	0	$\mathrm{wp}(\sigma_{\texttt{f}})$	$\mathrm{wp}(\sigma_{\texttt{v}})$	$\mathrm{wp}(\sigma_{\texttt{v}^*,\texttt{cen}})$	$\mathrm{wp}(\sigma_{\texttt{cap}})$
	$\vec{s}_b$ ($\texttt{f}$)	0	$\vec{q}_{\texttt{on}}$	0	$\vec{p}_{\texttt{f}}$	$\vec{p}_{\texttt{v}}$	$\vec{p}_{\texttt{v}^*,\texttt{cen}}$	$\vec{p}_{\texttt{cap}}$
	$\vec{s}_c$ ($\texttt{v}$)	0	$\vec{q}_{\texttt{on}}$	$\vec{q}_{\texttt{off}}$	$\vec{p}_{\texttt{f}}$	0	$\vec{p}_{\texttt{v}^*,\texttt{cen}} + \vec{p}_{\texttt{v}^*,\texttt{bot}}$ (or $\vec{q}_{\texttt{v}^*}$)	$\vec{p}_{\texttt{cap}}$
Weight parity		1	1	1	$\mathrm{wp}(\sigma_{\texttt{f}})$	0	1	$\mathrm{wp}(\sigma_{\texttt{cap}})$
Flag	$\vec{f}_a$ ($\texttt{cap}$)	0	0	0	0	0	0	$\vec{f}_{\texttt{cap}}$
	$\vec{f}_b$ ($\texttt{f}$)	0	0	0	$\vec{f}_{\texttt{f}}$	0	0	0
	$\vec{f}_c$ ($\texttt{v}$)	0	0	0	0	$\vec{f}_{\texttt{v}}$	$\vec{f}_{\texttt{v}^*}$	0

Table 6.2: Syndrome $\vec{s} = (s_a, \vec{s}_b, \vec{s}_c)$, weight parity, and flag vector $\vec{f} = (\vec{f}_a, \vec{f}_b, \vec{f}_c)$ corresponding to a single fault of each type which leads to a Z-type error. $s_a, \vec{s}_b, \vec{s}_c$ are syndromes evaluated by $\texttt{cap}, \texttt{f}$ and $\texttt{v}$ generators of X type, while $\vec{f}_a, \vec{f}_b, \vec{f}_c$ are flag outcomes obtained from circuits for measuring $\texttt{cap}, \texttt{f}$ and $\texttt{v}$ generators of Z type. Note that in some cases, a syndrome bit is equal to the weight parity of an error.

Now, let us consider the case that a fault combination arises from multiple faults. The syndrome and the weight parity of the combined error, and the cumulative flag vector of a fault combination can be calculated by adding the syndromes and the flag outcomes of all faults in the fault combination (the addition is modulo 2). For example, suppose that a fault combination consists of 2 faults which are of $\texttt{q}_{\texttt{on}}$ type and $\texttt{v}$ type. The syndrome

$\vec{s}(\mathbf{E})$ and the weight parity $\mathrm{wp}(\mathbf{E})$ of the combined error $\mathbf{E}$, and the cumulative flag vector $\vec{\mathbf{f}}$ correspond to such a fault combination are,

$$\begin{aligned}\vec{s}(\mathbf{E}) &= (1+\mathrm{wp}(\sigma_{\mathtt{v}}), \vec{q}_{\mathtt{on}}+\vec{p}_{\mathtt{v}}, \vec{q}_{\mathtt{on}}),\\ \mathrm{wp}(\mathbf{E}) &= 1,\\ \vec{\mathbf{f}} &= (\vec{0}, \vec{0}, \vec{f}_{\mathtt{v}}).\end{aligned}$$

For a general fault combination composed of multiple faults, the corresponding syndrome, weight parity, and cumulative flag vector can be calculated as follows: let $s_{\mathtt{cap}}, \vec{s}_{\mathtt{f}}, \vec{s}_{\mathtt{v}}$ denote syndromes of the combined error evaluated by $\mathtt{cap}, \mathtt{f}$, and $\mathtt{v}$ generators of X type, let $\mathrm{wp}_{\mathrm{tot}}$ denote the weight parity, and let $\vec{\mathbf{f}}_{\mathtt{cap}}, \vec{\mathbf{f}}_{\mathtt{f}}, \vec{\mathbf{f}}_{\mathtt{v}}$ denote parts of the cumulative flag vector obtained from circuits for measuring $\mathtt{cap}, \mathtt{f}$, and $\mathtt{v}$ generators of Z type. From Table 6.2, we find that for each fault combination,

$$s_{\mathtt{cap}} = n_0 + n_{\mathtt{on}} + \sum \mathrm{wp}(\sigma_{\mathtt{f}}) + \sum \mathrm{wp}(\sigma_{\mathtt{v}}) + \sum \mathrm{wp}(\sigma_{\mathtt{v}^*,\mathtt{cen}}) + \sum \mathrm{wp}(\sigma_{\mathtt{cap}}), \tag{6.8}$$

$$\vec{s}_{\mathtt{f}} = \sum \vec{q}_{\mathtt{on}} + \sum \vec{p}_{\mathtt{f}} + \sum \vec{p}_{\mathtt{v}} + \sum \vec{p}_{\mathtt{v}^*,\mathtt{cen}} + \sum \vec{p}_{\mathtt{cap}}, \tag{6.9}$$

$$\vec{s}_{\mathtt{v}} = \sum \vec{q}_{\mathtt{on}} + \sum \vec{q}_{\mathtt{off}} + \sum \vec{p}_{\mathtt{f}} + \sum \vec{q}_{\mathtt{v}^*} + \sum \vec{p}_{\mathtt{cap}}, \tag{6.10}$$

$$\mathrm{wp}_{\mathtt{tot}} = n_0 + n_{\mathtt{on}} + n_{\mathtt{off}} + \sum \mathrm{wp}(\sigma_{\mathtt{f}}) + n_{\mathtt{v}^*} + \sum \mathrm{wp}(\sigma_{\mathtt{cap}}), \tag{6.11}$$

$$\vec{\mathbf{f}}_{\mathtt{cap}} = \sum \vec{f}_{\mathtt{cap}}, \tag{6.12}$$

$$\vec{\mathbf{f}}_{\mathtt{f}} = \sum \vec{f}_{\mathtt{f}}, \tag{6.13}$$

$$\vec{\mathbf{f}}_{\mathtt{v}} = \sum \vec{f}_{\mathtt{v}} + \sum \vec{f}_{\mathtt{v}^*}, \tag{6.14}$$

where each sum is over the same type of faults (the equations are modulo 2). In addition, adding Eq. (6.8) to Eq. (6.11) and adding Eq. (6.9) to Eq. (6.10) give the following equations:

$$\mathrm{wp}_{\mathtt{bot}} = n_{\mathtt{off}} + \sum \mathrm{wp}(\sigma_{\mathtt{v}}) + \sum \mathrm{wp}(\sigma_{\mathtt{v}^*,\mathtt{bot}}), \tag{6.15}$$

$$\vec{s}_{\mathtt{bot}} = \sum \vec{q}_{\mathtt{off}} + \sum \vec{p}_{\mathtt{v}} + \sum \vec{p}_{\mathtt{v}^*,\mathtt{bot}}, \tag{6.16}$$

where $\mathrm{wp}_{\mathtt{bot}} = s_{\mathtt{cap}} + \mathrm{wp}_{\mathtt{tot}}$ and $\vec{s}_{\mathtt{bot}} = \vec{s}_{\mathtt{f}} + \vec{s}_{\mathtt{v}}$.

Eqs. (6.8) to (6.16) are the main ingredients for the proof of the main theorem to be

developed. One may notice that Eqs. (6.8) and (6.9), Eqs. (6.10) and (6.11), and Eqs. (6.15) and (6.16) come in pairs. They have the following physical meanings: suppose that the combined error $\mathbf{E}$ is $\mathbf{E}_0 \cdot \mathbf{E}_{\mathtt{on}} \cdot \mathbf{E}_{\mathtt{off}}$ where $\mathbf{E}_0, \mathbf{E}_{\mathtt{on}}, \mathbf{E}_{\mathtt{off}}$ are the error on $\mathtt{q}_0$, the error on the center plane, and the error on the bottom plane. Then,

1. Eq. (6.9) is the syndrome of $\mathbf{E}_{\mathtt{on}}$, while Eq. (6.8) is the weight parity $\mathbf{E}_{\mathtt{on}}$ plus the weight parity of $\mathbf{E}_0$.

2. Eq. (6.10) is the syndrome of $\mathbf{E}_{\mathtt{on}} \cdot \mathbf{E}_{\mathtt{off}}$, while Eq. (6.11) is the weight parity of $\mathbf{E}_{\mathtt{on}} \cdot \mathbf{E}_{\mathtt{off}}$ plus the weight parity of $\mathbf{E}_0$. (Since `v` generators capture errors on both planes simultaneously, $\mathbf{E}_{\mathtt{on}} \cdot \mathbf{E}_{\mathtt{off}}$ can be viewed as a remaining error when $\mathbf{E}_{\mathtt{on}}$ and $\mathbf{E}_{\mathtt{off}}$ are 'projected' on the same plane.)

3. Eq. (6.16) is the syndrome of $\mathbf{E}_{\mathtt{off}}$, while Eq. (6.15) is the weight parity of $\mathbf{E}_{\mathtt{off}}$.

From these pairs of equations, and from the fact that now we only have to specify the ordering of data CNOTs for each `f` generator, the ordering of data CNOTs after the first gate for the `cap` generator, and the ordering of flag CNOTs for each flag circuit, we can now simplify the CNOT ordering finding problem for a 3D structure to the problem of finding CNOT orderings on a 2D plane (which is similar to the 2D color code of distance d). In particular, each pair of equations concern errors on a 2D plane (the center, the bottom, or the projected plane). We will try to find conditions for the CNOT orderings on a 2D plane such that if satisfied, a bad case which makes $\mathcal{F}_t$ indistinguishable cannot happen.

Some types of faults on the 3D structure can be considered as the same type of faults when the problem is simplified. The followings are types of possible single faults on the 2D plane and their correspondence on the 3D structure:

1. Type $\mathtt{q}_{\mathtt{2D}}$: a fault causing a single-qubit Z-type error on the 2D plane which does not arise from any Z-type generator measurement. The syndrome of an error is denoted by $\vec{q}_{\mathtt{2D}}$. The total number of $\mathtt{q}_{\mathtt{2D}}$ faults is $n_{\mathtt{q2D}}$. The combined error from only $\mathtt{q}_{\mathtt{2D}}$ faults is denoted by $\mathbf{E}_{\mathtt{q2D}}$. This type of faults corresponds to $\mathtt{q}_{\mathtt{on}}$ and $\mathtt{q}_{\mathtt{off}}$ faults on the 3D structure.

2. Type $\mathtt{f}_{\mathtt{2D}}$: a fault occurred during a measurement of a `f` generator of Z type. A Z-type error from each fault of this type and its syndrome are denoted by $\sigma_{\mathtt{f2D}}$ and

$\vec{p}_{\mathtt{f}_{2D}}$. A flag vector corresponding to each fault of this type is denoted by $\vec{f}_{\mathtt{f}_{2D}}$. The total number of $\mathtt{f}_{2D}$ faults is $n_{\mathtt{f}_{2D}}$. The combined error from only $\mathtt{f}_{2D}$ faults is denoted by $\mathbf{E}_{\mathtt{f}_{2D}}$. This type of faults corresponds to $\mathtt{f}$ and $\mathtt{v}$ faults on the 3D structure (since an error on the center plane and an error on the bottom plane from a $\mathtt{v}$ fault have the same form; see an example in Fig. 6.7).

3. Type $\mathtt{v}^*_{2D}$: a fault occurred during a measurement of a $\mathtt{v}$ generator of Z type in which an error occurred on the center plane and an error on the bottom plane are different (see an example in Fig. 6.7). A part of a Z-type error from each fault of this type occurred on the center plane only and its syndrome are denoted by $\sigma_{\mathtt{v}^*_{2D},\mathtt{cen}}$ and $\vec{p}_{\mathtt{v}^*_{2D},\mathtt{cen}}$. The other part of the Z-type error that occurred on the bottom plane only and its syndrome are denoted by $\sigma_{\mathtt{v}^*_{2D},\mathtt{bot}}$ and $\vec{p}_{\mathtt{v}^*_{2D},\mathtt{bot}}$. A flag vector corresponding to each fault of this type is denoted by $\vec{f}_{\mathtt{v}^*_{2D}}$. The total number of $\mathtt{v}^*_{2D}$ faults is $n_{\mathtt{v}^*_{2D}}$. The part of the combined error from only $\mathtt{v}^*_{2D}$ faults on the center plane and the part on the bottom plane are denoted by $\mathbf{E}_{\mathtt{v}^*_{2D},\mathtt{cen}}$ and $\mathbf{E}_{\mathtt{v}^*_{2D},\mathtt{bot}}$. This type of faults corresponds to $\mathtt{v}^*$ faults on the 3D structure. (Note that this is the only type of faults which cannot be represented completely on the 2D plane since the error on the center plane and the error on the bottom plane are different. However, when running a computer simulation, we can treat a fault of $\mathtt{v}^*_{2D}$ type similarly to a fault of $\mathtt{f}_{2D}$ type except that two values of errors will be assigned to each fault.)

4. Type $\mathtt{cap}_{2D}$: a fault occurred during a measurement of a $\mathtt{cap}$ generator of Z type. A Z-type error from each fault of this type and its syndrome are denoted by $\sigma_{\mathtt{cap}_{2D}}$ and $\vec{p}_{\mathtt{cap}_{2D}}$ ($\sigma_{\mathtt{cap}_{2D}}$ is always on the center plane up to a multiplication of the $\mathtt{cap}$ generator being measured). A flag vector corresponding to each fault of this type is denoted by $\vec{f}_{\mathtt{cap}_{2D}}$. The total number of $\mathtt{cap}_{2D}$ faults is $n_{\mathtt{cap}_{2D}}$. The combined error from only $\mathtt{cap}_{2D}$ faults is denoted by $\mathbf{E}_{\mathtt{cap}_{2D}}$. This type of faults corresponds to $\mathtt{cap}$ faults on the 3D structure.

Examples of faults of each type on the 2D plane are illustrated in Fig. 6.8b on page 90. The correspondence between the notations for types of faults on the 2D plane and the 3D structure can be summarized in Table 6.3 on page 91.

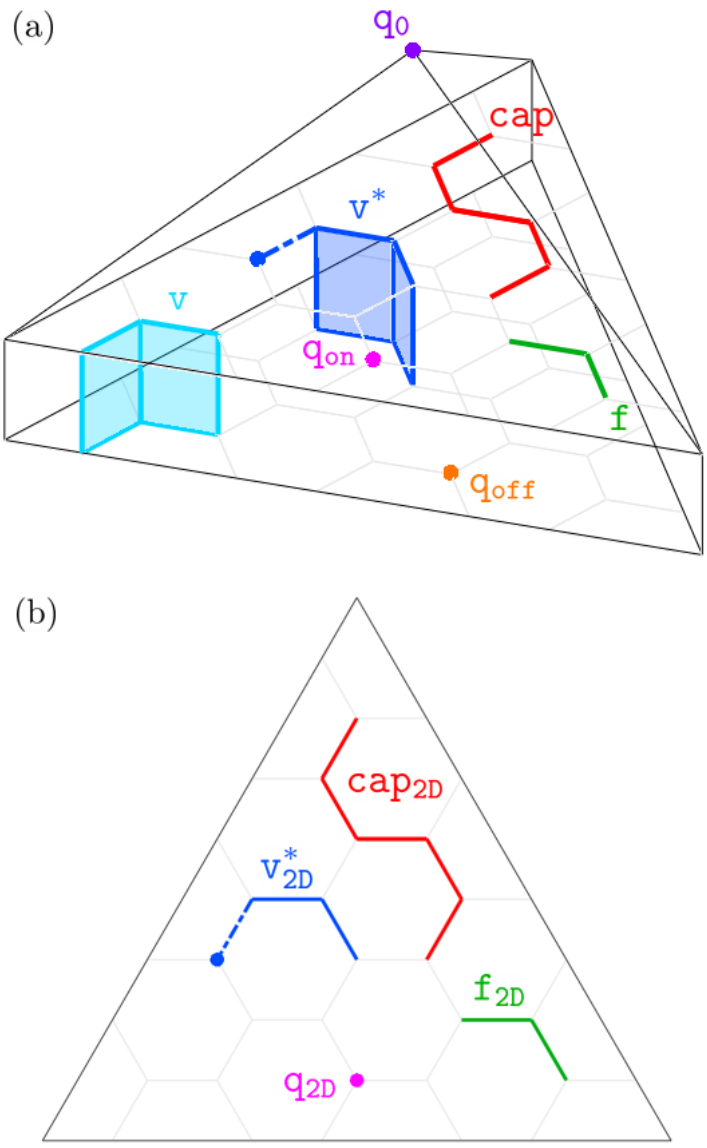


Figure 6.8: (a) Examples of faults of each type on the 3D structure. (b) Examples of faults of each type on the 2D plane.

2D plane				3D structure			
Fault type	Syndrome	Weight parity	Flag vector	Fault type	Syndrome	Weight parity	Flag vector
$\mathtt{q}_{\mathtt{2D}}$	$\vec{q}_{\mathtt{2D}}$	1	-	$\mathtt{q}_{\mathtt{on}}$, $\mathtt{q}_{\mathtt{off}}$, or $\mathtt{q}_{\mathtt{v}^*}$	$\vec{q}_{\mathtt{on}}$, $\vec{q}_{\mathtt{off}}$, or $\vec{q}_{\mathtt{v}^*}$	1	-
$\mathtt{f}_{\mathtt{2D}}$	$\vec{p}_{\mathtt{f}_{\mathtt{2D}}}$	$\mathrm{wp}(\sigma_{\mathtt{f}_{\mathtt{2D}}})$	$\vec{f}_{\mathtt{f}_{\mathtt{2D}}}$	$\mathtt{f}$, $\mathtt{v}$, or $\mathtt{v}^*$	$\vec{p}_{\mathtt{f}}$, $\vec{p}_{\mathtt{v}}$, $\vec{p}_{\mathtt{v}^*,\mathtt{cen}}$, or $\vec{p}_{\mathtt{v}^*,\mathtt{bot}}$	$\mathrm{wp}(\sigma_{\mathtt{f}})$, $\mathrm{wp}(\sigma_{\mathtt{v}})$, $\mathrm{wp}(\sigma_{\mathtt{v}^*,\mathtt{cen}})$, or $\mathrm{wp}(\sigma_{\mathtt{v}^*,\mathtt{bot}})$	$\vec{f}_{\mathtt{f}}$, $\vec{f}_{\mathtt{v}}$, or $\vec{f}_{\mathtt{v}^*}$
$\mathtt{v}^*_{\mathtt{2D}}$	$\vec{p}_{\mathtt{v}^*_{\mathtt{2D}},\mathtt{cen}}$	$\mathrm{wp}(\sigma_{\mathtt{v}^*_{\mathtt{2D}},\mathtt{cen}})$	$\vec{f}_{\mathtt{v}^*_{\mathtt{2D}}}$	$\mathtt{v}^*$	$\vec{p}_{\mathtt{v}^*,\mathtt{cen}}$	$\mathrm{wp}(\sigma_{\mathtt{v}^*,\mathtt{cen}})$	$\vec{f}_{\mathtt{v}^*}$
	$\vec{p}_{\mathtt{v}^*_{\mathtt{2D}},\mathtt{bot}}$	$\mathrm{wp}(\sigma_{\mathtt{v}^*_{\mathtt{2D}},\mathtt{bot}})$			$\vec{p}_{\mathtt{v}^*,\mathtt{bot}}$	$\mathrm{wp}(\sigma_{\mathtt{v}^*,\mathtt{bot}})$	
$\mathtt{cap}_{\mathtt{2D}}$	$\vec{p}_{\mathtt{cap}_{\mathtt{2D}}}$	$\mathrm{wp}(\sigma_{\mathtt{cap}_{\mathtt{2D}}})$	$\vec{f}_{\mathtt{cap}_{\mathtt{2D}}}$	$\mathtt{cap}$	$\vec{p}_{\mathtt{cap}}$	$\mathrm{wp}(\sigma_{\mathtt{cap}})$	$\vec{f}_{\mathtt{cap}}$

Table 6.3: The correspondence between the notations for types of faults on the 2D plane and the 3D structure.

We can see that possible Z-type errors on the 2D plane depend on the CNOT orderings for measuring `f` and `cap` generators of Z type. Next, we will state the sufficient conditions for the CNOT orderings on the 2D plane which will make $\mathcal{F}_t$ (which concerns fault combinations from the 3D structure) distinguishable. These sufficient conditions are introduced in order to prevent the case that can lead to an 'indistinguishable' pair (a pair of fault combinations from the 3D structure which does not satisfy any condition in Definition 3.3).

First, we will state a condition which is automatically satisfied if a code being considered on the 2D plane is a code of distance d to which Lemma 3.1 is applicable:

Condition 0 *For any fault combination on the 2D plane which satisfies $n_{\mathtt{q}_{2\mathtt{D}}} \leq d-1$, $\mathbf{E}_{\mathtt{q}_{2\mathtt{D}}}$ is not a nontrivial logical operator; equivalently, at least one of the followings is satisfied:*

1. $\sum \vec{q}_{2\mathtt{D}} \neq 0 \mod 2$, *or*
2. $n_{\mathtt{q}_{2\mathtt{D}}} \neq 1 \mod 2$.

Note that a nontrivial logical operator is an error corresponding to the trivial syndrome whose weight parity is odd (from Lemma 3.1). Condition 0 is equivalent to the fact that an error of weight $\leq d-1$ is detectable by a code of distance d; i.e., it either has a nontrivial syndrome or is a stabilizer. We state Condition 0 explicitly (although it is automatically satisfied) because the condition in this form looks similar to other conditions, which will simplify the proof of the main theorem.

Next, we will state five sufficient conditions for the CNOT orderings on the 2D plane which will make $\mathcal{F}_t$ distinguishable. The conditions are as follows:

Condition 1 *For any fault combination on the 2D plane which satisfies $n_{\mathtt{f}_{2\mathtt{D}}} \leq d-2$, $\mathbf{E}_{\mathtt{f}_{2\mathtt{D}}}$ is not a nontrivial logical operator or the cumulative flag vector is not zero; equivalently, at least one of the followings is satisfied:*

1. $\sum \vec{p}_{\mathtt{f}_{2\mathtt{D}}} \neq 0 \mod 2$, *or*
2. $\sum \mathrm{wp}(\sigma_{\mathtt{f}_{2\mathtt{D}}}) \neq 1 \mod 2$, *or*
3. $\sum \vec{f}_{\mathtt{f}_{2\mathtt{D}}} \neq 0 \mod 2$.

Condition 2 *For any fault combination on the 2D plane which satisfies* $n_{\mathtt{q}_{2\mathtt{D}}}+n_{\mathtt{f}_{2\mathtt{D}}} \leq d-3$, $\mathbf{E}_{\mathtt{q}_{2\mathtt{D}}} \cdot \mathbf{E}_{\mathtt{f}_{2\mathtt{D}}}$ *is not a nontrivial logical operator or the cumulative flag vector is not zero; equivalently, at least one of the followings is satisfied:*

1. $\sum \vec{q}_{2\mathtt{D}} + \sum \vec{p}_{\mathtt{f}_{2\mathtt{D}}} \neq 0 \mod 2$, *or*
2. $n_{\mathtt{q}_{2\mathtt{D}}} + \sum \mathrm{wp}(\sigma_{\mathtt{f}_{2\mathtt{D}}}) \neq 1 \mod 2$, *or*
3. $\sum \vec{f}_{\mathtt{f}_{2\mathtt{D}}} \neq 0 \mod 2$.

Condition 3 *For any fault combination on the 2D plane which satisfies* $n_{\mathtt{f}_{2\mathtt{D}}} = 1$ *and* $n_{\mathtt{q}_{2\mathtt{D}}} + n_{\mathtt{f}_{2\mathtt{D}}} \leq d-2$, $\mathbf{E}_{\mathtt{q}_{2\mathtt{D}}} \cdot \mathbf{E}_{\mathtt{f}_{2\mathtt{D}}}$ *is not a nontrivial logical operator or the cumulative flag vector is not zero; equivalently, at least one of the followings is satisfied:*

1. $\sum \vec{q}_{2\mathtt{D}} + \sum \vec{p}_{\mathtt{f}_{2\mathtt{D}}} \neq 0 \mod 2$, *or*
2. $n_{\mathtt{q}_{2\mathtt{D}}} + \sum \mathrm{wp}(\sigma_{\mathtt{f}_{2\mathtt{D}}}) \neq 1 \mod 2$, *or*
3. $\sum \vec{f}_{\mathtt{f}_{2\mathtt{D}}} \neq 0 \mod 2$.

Condition 4 *For any fault combination on the 2D plane which satisfies* $n_{\mathtt{f}_{2\mathtt{D}}} = 1$, $n_{\mathtt{q}_{2\mathtt{D}}} \geq 1$, $n_{\mathtt{v}^*_{2\mathtt{D}}} \geq 2$, *and* $n_{\mathtt{q}_{2\mathtt{D}}} + n_{\mathtt{f}_{2\mathtt{D}}} + n_{\mathtt{v}^*_{2\mathtt{D}}} = d-1$, *the following does not happen:* $\mathbf{E}_{\mathtt{f}_{2\mathtt{D}}} \cdot \mathbf{E}_{\mathtt{v}^*_{2\mathtt{D}},\mathtt{cen}}$ *is a stabilizer, and* $\mathbf{E}_{\mathtt{q}_{2\mathtt{D}}} \cdot \mathbf{E}_{\mathtt{v}^*_{2\mathtt{D}},\mathtt{bot}}$ *is a nontrivial logical operator, and the cumulative flag vector is zero; equivalently, at least one of the followings is satisfied:*

1. $\sum \vec{p}_{\mathtt{f}_{2\mathtt{D}}} + \sum \vec{p}_{\mathtt{v}^*_{2\mathtt{D}},\mathtt{cen}} \neq 0 \mod 2$, *or*
2. $\sum \mathrm{wp}(\sigma_{\mathtt{f}_{2\mathtt{D}}}) + \sum \mathrm{wp}(\sigma_{\mathtt{v}^*_{2\mathtt{D}},\mathtt{cen}}) \neq 0 \mod 2$, *or*
3. $\sum \vec{q}_{2\mathtt{D}} + \sum \vec{p}_{\mathtt{v}^*_{2\mathtt{D}},\mathtt{bot}} \neq 0 \mod 2$, *or*
4. $n_{\mathtt{q}_{2\mathtt{D}}} + \sum \mathrm{wp}(\sigma_{\mathtt{v}^*_{2\mathtt{D}},\mathtt{bot}}) \neq 1 \mod 2$, *or*
5. $\sum \vec{f}_{\mathtt{f}_{2\mathtt{D}}} \neq 0 \mod 2$, *or*
6. $\sum \vec{f}_{\mathtt{v}^*_{2\mathtt{D}}} \neq 0 \mod 2$.

Condition 5 *For any fault combination on the 2D plane which satisfies* $n_{\mathtt{cap}_{2D}} = 1$, $n_{\mathtt{q}_{2D}} \geq 1$, $n_{\mathtt{f}_{2D}} + n_{\mathtt{v}^*_{2D}} \geq 2$, *and* $n_{\mathtt{q}_{2D}} + n_{\mathtt{f}_{2D}} + n_{\mathtt{v}^*_{2D}} + n_{\mathtt{cap}_{2D}} = d - 1$, *the following does not happen:* $\mathbf{E}_{\mathtt{f}_{2D}} \cdot \mathbf{E}_{\mathtt{v}^*_{2D},\mathtt{cen}} \cdot \mathbf{E}_{\mathtt{cap}_{2D}}$ *is a stabilizer, and* $\mathbf{E}_{\mathtt{q}_{2D}} \cdot \mathbf{E}_{\mathtt{f}_{2D}} \cdot \mathbf{E}_{\mathtt{v}^*_{2D},\mathtt{bot}}$ *is a nontrivial logical operator, and the cumulative flag vector is zero; equivalently, at least one of the followings is satisfied:*

1. $\sum \vec{p}_{\mathtt{f}_{2D}} + \sum \vec{p}_{\mathtt{v}^*_{2D},\mathtt{cen}} + \sum \vec{p}_{\mathtt{cap}_{2D}} \neq 0 \mod 2$, *or*
2. $\sum \mathrm{wp}(\sigma_{\mathtt{f}_{2D}}) + \sum \mathrm{wp}(\sigma_{\mathtt{v}^*_{2D},\mathtt{cen}}) + \sum \mathrm{wp}(\sigma_{\mathtt{cap}_{2D}}) \neq 0 \mod 2$, *or*
3. $\sum \vec{q}_{2D} + \sum \vec{p}_{\mathtt{f}_{2D}} + \sum \vec{p}_{\mathtt{v}^*_{2D},\mathtt{bot}} \neq 0 \mod 2$, *or*
4. $n_{\mathtt{q}_{2D}} + \sum \mathrm{wp}(\sigma_{\mathtt{f}_{2D}}) + \sum \mathrm{wp}(\sigma_{\mathtt{v}^*_{2D},\mathtt{bot}}) \neq 1 \mod 2$, *or*
5. $\sum \vec{f}_{\mathtt{f}_{2D}} + \vec{f}_{\mathtt{v}^*_{2D}} \neq 0 \mod 2$, *or*
6. $\sum \vec{f}_{\mathtt{cap}_{2D}} \neq 0 \mod 2$.

Conditions 1 to 5 prevent fault combinations of some form from occurring on the 2D plane (such fault combinations can lead to an indistinguishable fault set). If we arrange the CNOT gates in the circuits for `f` and `cap` generators so that all conditions are satisfied, then a fault set $\mathcal{F}_t$ (which considers the 3D structure) will be distinguishable. The main theorem of [TL21a] is as follows:

Theorem 6.1 *Let* $\mathcal{F}_t$ *be the fault set corresponding to circuits for measuring* `f`, `v`, *and* `cap` *generators of the capped color code in H form constructed from* $CCC(d)$ *(where* $t = (d-1)/2$, $d = 3, 5, 7, ...$*), and suppose that the general configurations of CNOT gates for* `f`, `v`, *and* `cap` *generators are imposed, and the circuits for each pair of* X*-type and* Z*-type generators use the same CNOT ordering. Let the code on the (simplified) 2D plane be the 2D color code of distance* d*. If all possible fault combinations on the 2D plane arising from the circuits for measuring* `f` *and* `cap` *generators satisfy Conditions 1 to 5, then* $\mathcal{F}_t$ *is distinguishable.*

Proof ideas: The proof is organized as follows: First, we try to show that if Conditions 1 to 5 are satisfied, then for any fault combination arising from up to $d-1$ faults whose combined error is purely Z type, the fault combination cannot lead to a logical Z operator and the zero cumulative flag vector. The same analysis is also applicable to fault combinations

whose combined error is purely X type since the circuits for measuring each pair of X-type and Z-type generators are of the same form. Afterwards, we use the fact that i faults during the measurements of Z-type generators cannot cause an X-type error of weight more than i (and vice versa), and show that there is no fault combination arising from up to $d-1$ faults which leads to a nontrivial logical operator and the zero cumulative flag vector. By Proposition 3.1, this implies that $\mathcal{F}_t$ is distinguishable.

In order to prove the first part, we will assume that Conditions 1 to 5 are satisfied and there exists a fault combination arising from $< d$ faults whose combined error is a logical Z operator and its cumulative flag vector is zero, then show that some contradiction will happen. From Lemma 3.1, a logical Z operator is a Z-type error with trivial syndrome and odd weight parity. Therefore, such a fault combination will give $s_{\mathtt{cap}} = 0$, $\vec{s}_{\mathtt{f}} = \vec{0}$, $\vec{s}_{\mathtt{v}} = \vec{0}$, $\mathrm{wp}_{\mathtt{tot}} = 1$, $\vec{\mathbf{f}}_{\mathtt{cap}} = \vec{0}$, $\vec{\mathbf{f}}_{\mathtt{f}} = \vec{0}$, $\vec{\mathbf{f}}_{\mathtt{v}} = \vec{0}$, $\mathrm{wp}_{\mathtt{bot}} = 1$, and $\vec{s}_{\mathtt{bot}} = 0$ in the main equations (Eqs. (6.8) to (6.16)). A proof for this part will be divided into 4 cases: (1) $n_{\mathtt{f}} = 0$ and $n_{\mathtt{cap}} = 0$, (2) $n_{\mathtt{f}} \geq 1$ and $n_{\mathtt{cap}} = 0$, (3) $n_{\mathtt{f}} = 0$ and $n_{\mathtt{cap}} \geq 1$, and (4) $n_{\mathtt{f}} \geq 1$ and $n_{\mathtt{cap}} \geq 1$. In each case, the main equations will be simplified by eliminating the terms which are equal to zero. Afterwards, We will consider the following pairs of equations: Eq. (6.8) and Eq. (6.9), Eq. (6.10) and Eq. (6.11), Eq. (6.15) and Eq. (6.16). For each pair, the types of faults on the 3D structure will be translated to their corresponding types of faults on the 2D plane in order to find matching conditions from Conditions 1 to 5. Note that the total number of faults of each type will also help in finding the matching conditions, and the total number of faults of all types is at most $d-1$. When the matching conditions are found, we will find that some contradictions will happen (assuming that all conditions are satisfied), and this is true for all possible cases.

Proof:
In the first part of the proof, we will assume that data errors arising from all faults are purely Z type, and show that if Conditions 1 to 5 are satisfied, then there is no fault combination arising from up to $d-1$ faults whose combined error is a logical Z operator and its cumulative flag vector is zero. Because i faults during the measurements of X-type generators cannot cause a Z-type error of weight more than i, we can assume that each fault is either a qubit fault causing a Z-type error (which is $\mathtt{q}_0$, $\mathtt{q}_{\mathtt{on}}$, or $\mathtt{q}_{\mathtt{off}}$ fault), or a fault during a measurement of some Z-type generator (which is $\mathtt{f}$, $\mathtt{v}$, $\mathtt{v}^*$, or $\mathtt{cap}$ fault).

First, recall the main equations (in mod 2):

$$s_{\mathtt{cap}} = n_0 + n_{\mathtt{on}} + \sum \mathrm{wp}(\sigma_{\mathtt{f}}) + \sum \mathrm{wp}(\sigma_{\mathtt{v}}) + \sum \mathrm{wp}(\sigma_{\mathtt{v}^*,\mathtt{cen}}) + \sum \mathrm{wp}(\sigma_{\mathtt{cap}}), \tag{6.8}$$

$$\vec{s}_{\mathtt{f}} = \sum \vec{q}_{\mathtt{on}} + \sum \vec{p}_{\mathtt{f}} + \sum \vec{p}_{\mathtt{v}} + \sum \vec{p}_{\mathtt{v}^*,\mathtt{cen}} + \sum \vec{p}_{\mathtt{cap}}, \tag{6.9}$$

$$\vec{s}_{\mathtt{v}} = \sum \vec{q}_{\mathtt{on}} + \sum \vec{q}_{\mathtt{off}} + \sum \vec{p}_{\mathtt{f}} + \sum \vec{q}_{\mathtt{v}^*} + \sum \vec{p}_{\mathtt{cap}}, \tag{6.10}$$

$$\mathrm{wp}_{\mathtt{tot}} = n_0 + n_{\mathtt{on}} + n_{\mathtt{off}} + \sum \mathrm{wp}(\sigma_{\mathtt{f}}) + n_{\mathtt{v}^*} + \sum \mathrm{wp}(\sigma_{\mathtt{cap}}), \tag{6.11}$$

$$\vec{\mathbf{f}}_{\mathtt{cap}} = \sum \vec{f}_{\mathtt{cap}}, \tag{6.12}$$

$$\vec{\mathbf{f}}_{\mathtt{f}} = \sum \vec{f}_{\mathtt{f}}, \tag{6.13}$$

$$\vec{\mathbf{f}}_{\mathtt{v}} = \sum \vec{f}_{\mathtt{v}} + \sum \vec{f}_{\mathtt{v}^*}, \tag{6.14}$$

$$\mathrm{wp}_{\mathtt{bot}} = n_{\mathtt{off}} + \sum \mathrm{wp}(\sigma_{\mathtt{v}}) + \sum \mathrm{wp}(\sigma_{\mathtt{v}^*,\mathtt{bot}}), \tag{6.15}$$

$$\vec{s}_{\mathtt{bot}} = \sum \vec{q}_{\mathtt{off}} + \sum \vec{p}_{\mathtt{v}} + \sum \vec{p}_{\mathtt{v}^*,\mathtt{bot}}. \tag{6.16}$$

Note that the types of faults involved in the main equations and the types of faults involved in the conditions are related by the correspondence in Table 6.3. Here we will show that if Conditions 1 to 5 are satisfied and there exists a fault combination arising from up to $d-1$ faults which corresponds to a logical Z operator and the zero cumulative flag vector, some contradictions will happen (also note that Condition 0 is automatically satisfied). By Lemma 3.1, a fault combination corresponding to a logical Z operator and the zero cumulative flag vector gives $s_{\mathtt{cap}} = 0$, $\vec{s}_{\mathtt{f}} = \vec{0}$, $\vec{s}_{\mathtt{v}} = \vec{0}$, $\mathrm{wp}_{\mathtt{tot}} = 1$, $\vec{\mathbf{f}}_{\mathtt{cap}} = \vec{0}$, $\vec{\mathbf{f}}_{\mathtt{f}} = \vec{0}$, $\vec{\mathbf{f}}_{\mathtt{v}} = \vec{0}$, $\mathrm{wp}_{\mathtt{bot}} = 1$, and $\vec{s}_{\mathtt{bot}} = 0$. We will divide the proof into 4 cases: (1) $n_{\mathtt{f}} = 0$ and $n_{\mathtt{cap}} = 0$, (2) $n_{\mathtt{f}} \geq 1$ and $n_{\mathtt{cap}} = 0$, (3) $n_{\mathtt{f}} = 0$ and $n_{\mathtt{cap}} \geq 1$, and (4) $n_{\mathtt{f}} \geq 1$ and $n_{\mathtt{cap}} \geq 1$.

Case 1: $n_{\mathtt{f}} = 0$ and $n_{\mathtt{cap}} = 0$. The main equations can be simplified as follows (trivial equations are neglected):

$$0 = n_0 + n_{\mathtt{on}} + \sum \mathrm{wp}(\sigma_{\mathtt{v}}) + \sum \mathrm{wp}(\sigma_{\mathtt{v}^*,\mathtt{cen}}), \tag{6.8}$$

$$\vec{0} = \sum \vec{q}_{\mathtt{on}} + \sum \vec{p}_{\mathtt{v}} + \sum \vec{p}_{\mathtt{v}^*,\mathtt{cen}}, \tag{6.9}$$

$$\vec{0} = \sum \vec{q}_{\mathtt{on}} + \sum \vec{q}_{\mathtt{off}} + \sum \vec{q}_{\mathtt{v}^*}, \tag{6.10}$$

$$1 = n_0 + n_{\mathtt{on}} + n_{\mathtt{off}} + n_{\mathtt{v}^*}, \tag{6.11}$$

$$\vec{0} = \sum \vec{f}_{\mathtt{v}} + \sum \vec{f}_{\mathtt{v}^*}, \tag{6.14}$$

$$1 = n_{\mathtt{off}} + \sum \mathrm{wp}(\sigma_{\mathtt{v}}) + \sum \mathrm{wp}(\sigma_{\mathtt{v}^*,\mathtt{bot}}), \tag{6.15}$$

$$\vec{0} = \sum \vec{q}_{\mathtt{off}} + \sum \vec{p}_{\mathtt{v}} + \sum \vec{p}_{\mathtt{v}^*,\mathtt{bot}}. \tag{6.16}$$

All faults involved in Eqs. (6.10) and (6.11) correspond to $\mathtt{q}_{\mathtt{2D}}$ faults on the 2D code and the total number of faults are at most $d-1$. Because Condition 0 is satisfied, from Eqs. (6.10) and (6.11), we must have that $n_{\mathtt{on}} + n_{\mathtt{off}} + n_{\mathtt{v}^*} = 0 \pmod 2$ which implies that $n_0 = 1$. Thus, Eq. (6.8) becomes,

$$1 = n_{\mathtt{on}} + \sum \mathrm{wp}(\sigma_{\mathtt{v}}) + \sum \mathrm{wp}(\sigma_{\mathtt{v}^*,\mathtt{cen}}). \tag{6.8}$$

Since the total number of faults are $n_0 + n_{\mathtt{on}} + n_{\mathtt{off}} + n_{\mathtt{v}} + n_{\mathtt{v}^*} \leq d-1$, we find that $n_{\mathtt{on}} + n_{\mathtt{off}} + n_{\mathtt{v}} + n_{\mathtt{v}^*} \leq d-2$. Let us consider the following cases:

(1.a) If $n_{\mathtt{off}} = 0$, we have $n_{\mathtt{v}} + n_{\mathtt{v}^*} \leq d-2-n_{\mathtt{on}} \leq d-2$. In this case, Eqs. (6.14) to (6.16) contradict Condition 1 (where $\mathtt{v}$ and $\mathtt{v}^*$ faults correspond to $\mathtt{f}_{\mathtt{2D}}$ fault).

(1.b) If $n_{\mathtt{off}} \geq 1$, we have $n_{\mathtt{on}} + n_{\mathtt{v}} + n_{\mathtt{v}^*} \leq d-2-n_{\mathtt{off}} \leq d-3$. In this case, Eqs. (6.8), (6.9) and (6.14) contradict Condition 2 (where $\mathtt{q}_{\mathtt{on}}$ fault corresponds to $\mathtt{q}_{\mathtt{2D}}$ fault, and $\mathtt{v}$ and $\mathtt{v}^*$ faults correspond to $\mathtt{f}_{\mathtt{2D}}$ fault).

Case 2: $n_{\mathtt{f}} \geq 1$ and $n_{\mathtt{cap}} = 0$. The main equations can be simplified as follows:

$$0 = n_0 + n_{\mathtt{on}} + \sum \mathrm{wp}(\sigma_{\mathtt{f}}) + \sum \mathrm{wp}(\sigma_{\mathtt{v}}) + \sum \mathrm{wp}(\sigma_{\mathtt{v}^*,\mathtt{cen}}), \tag{6.8}$$

$$\vec{0} = \sum \vec{q}_{\mathtt{on}} + \sum \vec{p}_{\mathtt{f}} + \sum \vec{p}_{\mathtt{v}} + \sum \vec{p}_{\mathtt{v}^*,\mathtt{cen}}, \tag{6.9}$$

$$\vec{0} = \sum \vec{q}_{\mathtt{on}} + \sum \vec{q}_{\mathtt{off}} + \sum \vec{p}_{\mathtt{f}} + \sum \vec{q}_{\mathtt{v}^*}, \tag{6.10}$$

$$1 = n_0 + n_{\mathtt{on}} + n_{\mathtt{off}} + \sum \mathrm{wp}(\sigma_{\mathtt{f}}) + n_{\mathtt{v}^*}, \tag{6.11}$$

$$\vec{0} = \sum \vec{f}_{\mathtt{f}}, \tag{6.13}$$

$$\vec{0} = \sum \vec{f}_{\mathtt{v}} + \sum \vec{f}_{\mathtt{v}^*}, \tag{6.14}$$

$$1 = n_{\mathtt{off}} + \sum \mathrm{wp}(\sigma_{\mathtt{v}}) + \sum \mathrm{wp}(\sigma_{\mathtt{v}^*,\mathtt{bot}}), \tag{6.15}$$

$$\vec{0} = \sum \vec{q}_{\mathtt{off}} + \sum \vec{p}_{\mathtt{v}} + \sum \vec{p}_{\mathtt{v}^*,\mathtt{bot}}. \tag{6.16}$$

The total number of faults are $n_0 + n_{\mathtt{on}} + n_{\mathtt{off}} + n_{\mathtt{f}} + n_{\mathtt{v}} + n_{\mathtt{v}^*} \leq d-1$, which means that $n_{\mathtt{off}} + n_{\mathtt{v}} + n_{\mathtt{v}^*} \leq d - 1 - n_0 - n_{\mathtt{on}} - n_{\mathtt{f}}$ (where $n_{\mathtt{f}} \geq 1$). Consider the following cases:

(2.a) If $n_0 = 1$ or $n_{\mathtt{on}} \geq 1$ or $n_{\mathtt{f}} \geq 2$, we have $n_{\mathtt{off}} + n_{\mathtt{v}} + n_{\mathtt{v}^*} \leq d-3$. In this case, Eqs. (6.14) to (6.16) contradict Condition 2 (where $\mathtt{q}_{\mathtt{off}}$ fault corresponds to $\mathtt{q}_{\mathtt{2D}}$ fault, and $\mathtt{v}$ and $\mathtt{v}^*$ faults correspond to $\mathtt{f}_{\mathtt{2D}}$ fault).

(2.b) If $n_0 = 0, n_{\mathtt{on}} = 0$, and $n_{\mathtt{f}} = 1$, we find that $n_{\mathtt{off}} + n_{\mathtt{f}} + n_{\mathtt{v}} + n_{\mathtt{v}^*} \leq d-1$ and $n_{\mathtt{off}} + n_{\mathtt{v}} + n_{\mathtt{v}^*} \leq d-2$. Let us divide this case into the following subcases (where some subcases may overlap):

(i) If $n_{\mathtt{v}} \geq 1$, then $n_{\mathtt{off}} + n_{\mathtt{f}} + n_{\mathtt{v}^*} \leq d-2$. In this case, Eqs. (6.10), (6.11) and (6.13) contradict Condition 3 (where $\mathtt{q}_{\mathtt{off}}$ and $\mathtt{q}_{\mathtt{v}^*}$ faults correspond to $\mathtt{q}_{\mathtt{2D}}$ fault, and $\mathtt{f}$ fault corresponds to $\mathtt{f}_{\mathtt{2D}}$ fault).

(ii) If $n_{\mathtt{v}} = 0$ and $n_{\mathtt{v}^*} = 0$, then Eqs. (6.15) and (6.16) contradict Condition 0 (where $\mathtt{q}_{\mathtt{off}}$ fault corresponds to $\mathtt{q}_{\mathtt{2D}}$ fault).

(iii) If $n_{\mathtt{off}} = 0$, then $n_{\mathtt{v}} + n_{\mathtt{v}^*} \leq d-2$ and Eqs. (6.14) to (6.16) contradict Condition 1 (where $\mathtt{v}$ and $\mathtt{v}^*$ faults correspond to $\mathtt{f}_{\mathtt{2D}}$ fault).

(iv) If $n_{\mathtt{off}} \geq 1$, $n_{\mathtt{v}} = 0$, and $n_{\mathtt{v}^*} = 1$, then $n_{\mathtt{off}} + n_{\mathtt{v}^*} \leq d-2$ and Eqs. (6.14) to (6.16) contradict Condition 3 (where $\mathtt{q}_{\mathtt{off}}$ fault correspond to $\mathtt{q}_{\mathtt{2D}}$ fault, and $\mathtt{v}^*$ fault corresponds to $\mathtt{f}_{\mathtt{2D}}$ fault).

(v) If $n_{\mathtt{off}} \geq 1$, $n_{\mathtt{v}} = 0$, $n_{\mathtt{v}^*} \geq 2$, and $n_{\mathtt{off}} + n_{\mathtt{f}} + n_{\mathtt{v}^*} \leq d-2$, then Eqs. (6.10), (6.11) and (6.13) contradict Condition 3 (where $\mathtt{q}_{\mathtt{off}}$ and $\mathtt{q}_{\mathtt{v}^*}$ faults correspond to $\mathtt{q}_{\mathtt{2D}}$ fault, and $\mathtt{f}$ fault corresponds to $\mathtt{f}_{\mathtt{2D}}$ fault).

(vi) If $n_{\mathtt{off}} \geq 1$, $n_{\mathtt{v}} = 0$, $n_{\mathtt{v}^*} \geq 2$, and $n_{\mathtt{off}} + n_{\mathtt{f}} + n_{\mathtt{v}^*} = d-1$, then Eqs. (6.8), (6.9) and (6.13) to (6.16) contradict Condition 4 (where $\mathtt{q}_{\mathtt{off}}$, $\mathtt{q}_{\mathtt{f}}$, and $\mathtt{q}_{\mathtt{v}^*}$ faults correspond to $\mathtt{q}_{\mathtt{2D}}$, $\mathtt{f}_{\mathtt{2D}}$, and $\mathtt{v}^*_{\mathtt{2D}}$ faults, respectively).

Case 3: $n_{\mathtt{f}} = 0$ and $n_{\mathtt{cap}} \geq 1$. The main equations can be simplified as follows:

$$0 = n_0 + n_{\mathtt{on}} + \sum \mathrm{wp}(\sigma_{\mathtt{v}}) + \sum \mathrm{wp}(\sigma_{\mathtt{v}^*,\mathtt{cen}}) + \sum \mathrm{wp}(\sigma_{\mathtt{cap}}), \tag{6.8}$$

$$\vec{0} = \sum \vec{q}_{\mathtt{on}} + \sum \vec{p}_{\mathtt{v}} + \sum \vec{p}_{\mathtt{v}^*,\mathtt{cen}} + \sum \vec{p}_{\mathtt{cap}}, \tag{6.9}$$

$$\vec{0} = \sum \vec{q}_{\mathtt{on}} + \sum \vec{q}_{\mathtt{off}} + \sum \vec{q}_{\mathtt{v}^*} + \sum \vec{p}_{\mathtt{cap}}, \tag{6.10}$$

$$1 = n_0 + n_{\mathtt{on}} + n_{\mathtt{off}} + n_{\mathtt{v}^*} + \sum \mathrm{wp}(\sigma_{\mathtt{cap}}), \tag{6.11}$$

$$\vec{0} = \sum \vec{f}_{\mathtt{cap}}, \tag{6.12}$$

$$\vec{0} = \sum \vec{f}_{\mathtt{v}} + \sum \vec{f}_{\mathtt{v}^*}, \tag{6.14}$$

$$1 = n_{\mathtt{off}} + \sum \mathrm{wp}(\sigma_{\mathtt{v}}) + \sum \mathrm{wp}(\sigma_{\mathtt{v}^*,\mathtt{bot}}), \tag{6.15}$$

$$\vec{0} = \sum \vec{q}_{\mathtt{off}} + \sum \vec{p}_{\mathtt{v}} + \sum \vec{p}_{\mathtt{v}^*,\mathtt{bot}}. \tag{6.16}$$

The total number of faults are $n_0 + n_{\mathtt{on}} + n_{\mathtt{off}} + n_{\mathtt{v}} + n_{\mathtt{v}^*} + n_{\mathtt{cap}} \leq d-1$, which means that $n_{\mathtt{off}} + n_{\mathtt{v}} + n_{\mathtt{v}^*} \leq d - 1 - n_0 - n_{\mathtt{on}} - n_{\mathtt{cap}}$ (where $n_{\mathtt{cap}} \geq 1$). Consider the following cases:

(3.a) If $n_0 \geq 1$ or $n_{\mathtt{on}} \geq 1$ or $n_{\mathtt{cap}} \geq 2$, then $n_{\mathtt{off}} + n_{\mathtt{v}} + n_{\mathtt{v}^*} \leq d-3$. In this case, Eqs. (6.14) to (6.16) contradict Condition 2 (where $\mathtt{q}_{\mathtt{off}}$ fault corresponds to $\mathtt{q}_{\mathtt{2D}}$ fault, and $\mathtt{v}$ and $\mathtt{v}^*$ faults correspond to $\mathtt{f}_{\mathtt{2D}}$ fault).

(3.b) If $n_0 = 0, n_{\mathtt{on}} = 0$, and $n_{\mathtt{cap}} = 1$, we find that $n_{\mathtt{off}} + n_{\mathtt{v}} + n_{\mathtt{v}^*} + n_{\mathtt{cap}} \leq d-1$ and $n_{\mathtt{off}} + n_{\mathtt{v}} + n_{\mathtt{v}^*} \leq d-2$. Let us divide the proof into the following subcases (where some subcases may overlap):

(i) If $n_{\mathtt{v}} + n_{\mathtt{v}^*} = 0$, then Eqs. (6.15) and (6.16) contradict Condition 0 (where $\mathtt{q}_{\mathtt{off}}$ fault corresponds to $\mathtt{q}_{\mathtt{2D}}$ fault).

(ii) If $n_{\mathtt{v}} + n_{\mathtt{v}^*} = 1$, then Eqs. (6.14) to (6.16) contradict Condition 3 (where $\mathtt{q}_{\mathtt{off}}$ fault corresponds to $\mathtt{q}_{\mathtt{2D}}$ fault, and $\mathtt{v}$ and $\mathtt{v}^*$ faults correspond to $\mathtt{f}_{\mathtt{2D}}$ fault).

(iii) If $n_{\mathtt{off}} = 0$, then $n_{\mathtt{v}} + n_{\mathtt{v}^*} \leq d-2$. In this case, Eqs. (6.14) to (6.16) contradict Condition 1 (where $\mathtt{v}$ and $\mathtt{v}^*$ faults correspond to $\mathtt{f}_{\mathtt{2D}}$ fault).

(iv) If $n_{\mathtt{off}} + n_{\mathtt{v}} + n_{\mathtt{v}^*} + n_{\mathtt{cap}} \leq d-2$ (or equivalently, $n_{\mathtt{off}} + n_{\mathtt{v}} + n_{\mathtt{v}^*} \leq d-3$), then Eqs. (6.14) to (6.16) contradict Condition 2 (where $\mathtt{q}_{\mathtt{off}}$ fault corresponds to $\mathtt{q}_{\mathtt{2D}}$ fault, and $\mathtt{v}$ and $\mathtt{v}^*$ faults correspond to $\mathtt{f}_{\mathtt{2D}}$ fault).

(v) If $n_{\mathtt{off}} \geq 1$, $n_{\mathtt{v}} + n_{\mathtt{v}^*} \geq 2$, and $n_{\mathtt{off}} + n_{\mathtt{v}} + n_{\mathtt{v}^*} + n_{\mathtt{cap}} = d - 1$, then Eqs. (6.8), (6.9), (6.12) and (6.14) to (6.16) contradict Condition 5 (where $\mathtt{q}_{\mathtt{off}}$, $\mathtt{v}$, $\mathtt{v}^*$, $\mathtt{cap}$ faults correspond to $\mathtt{q}_{\mathtt{2D}}$, $\mathtt{f}_{\mathtt{2D}}$, $\mathtt{v}^*_{\mathtt{2D}}$, and $\mathtt{cap}_{\mathtt{2D}}$ faults, respectively).

Case 4: $n_{\mathtt{f}} \geq 1$ and $n_{\mathtt{cap}} \geq 1$ (the main equations cannot be simplified in this case). From the fact that the total number of faults is at most $d-1$, we have $n_{\mathtt{off}} + n_{\mathtt{v}} + n_{\mathtt{v}^*} \leq d - 3$. In this case, we find that Eqs. (6.14) to (6.16) contradict Condition 2 (where $\mathtt{q}_{\mathtt{off}}$ fault corresponds to $\mathtt{q}_{\mathtt{2D}}$ fault, and $\mathtt{v}$ and $\mathtt{v}^*$ faults correspond to $\mathtt{f}_{\mathtt{2D}}$ fault).

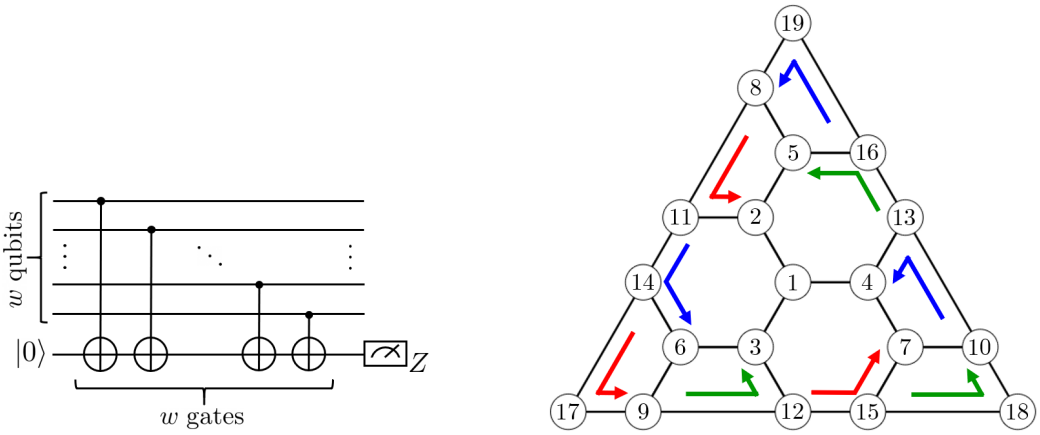
So far, we have shown that if Conditions 1 to 5 are satisfied and all faults give rise to purely Z-type errors, then there is no fault combination arising from up to $d-1$ faults whose combined error is a logical Z operator and its cumulative flag vector is zero. Because the circuits for each pair of X-type and Z-type generators use the same CNOT ordering, the same analysis is also applicable to the case of purely X-type errors; i.e., if Conditions 1 to 5 are satisfied and all faults give rise to purely X-type errors, then there is no fault combination arising from up to $d-1$ faults whose combined error is a logical X operator and its cumulative flag vector is zero. In the next part of the proof, we will use these results to show that $\mathcal{F}_t$ is distinguishable.

Let us consider a fault combination whose combined error is of mixed type. Let t_x and t_z denote the total number of faults during the measurements of X-type and Z-type generators, and let u_x, u_y, u_z denote the number of qubit faults which give X-type, Y-type, and Z-type errors, respectively. Suppose that the fault combination arises from no more than $d-1$ faults, we have $t_x + t_z + u_x + u_y + u_z \leq d-1$. Next, observe that t_x faults during the measurement of X-type generators cannot cause a Z-type error of weight more than t_x, and t_z faults during the measurement of Z-type generators cannot cause a X-type error of weight more than t_z. Thus, the Z-part of the combined error and the cumulative flag vector corresponding to Z-type generators can be considered as an error and a cumulative flag vector arising from $t_z + t_x + u_z + u_y \leq d-1$ faults which give rise to purely Z-type errors. Similarly, the X-part of the combined error and the cumulative flag vector corresponding to X-type generators can be considered as an error and a cumulative flag vector arising from $t_x + t_z + u_x + u_y \leq d-1$ faults which give rise to purely X-type errors. Recall that there is no fault combination arising from up to $d-1$ faults whose combined error is a logical X (or a logical Z) operator and its cumulative flag vector is zero when all faults give rise to purely X-type (or purely Z-type) errors. Using this, we find that for any fault combination arising from $d-1$ faults, it cannot correspond to a nontrivial logical operator

and the zero cumulative flag vector. That is, there is no fault combination corresponding to a nontrivial logical operator and the zero cumulative flag vector in $\mathcal{F}_{2t}$ where $2t = d-1$. By Proposition 3.1, this implies that $\mathcal{F}_t$ is distinguishable. □

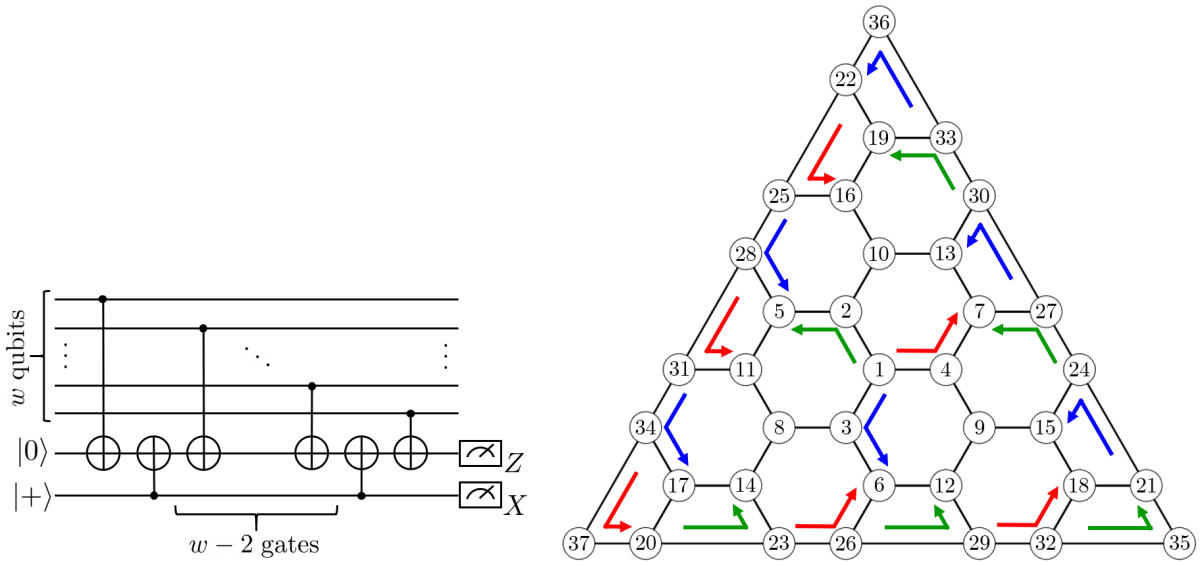
Theorem 6.1 can make the process of finding CNOT orderings which give a distinguishable fault set less laborious; instead of finding all possible fault combinations arising from the circuits for `f`, `v`, and `cap` generators and check whether any condition in Definition 3.3 is satisfied, we just have to check whether all possible fault combinations arising from the circuits for `f` and `cap` generators satisfy Conditions 1 to 5. Note that number of possible fault combinations of the latter task is much smaller than that of the prior task because the total number of generators involved in the latter calculation roughly decreases by half, and the weight of an `f` generator is half of the weight of its corresponding `v` generator. After good CNOT orderings for `f` and `cap` generators are found, we can find the CNOT orderings of `v` generators by the constraints imposed by the general configurations for data and flag CNOTs.

Using Theorem 6.1, we can find circuits for capped color codes in H form of distance 5 and 7, which give distinguishable fault sets with $t = 2$ and $t = 3$, respectively. The circuit for measuring a generator of weight w of the code of distance 5 is a non-flag circuit as shown in Fig. 6.9a, and the orderings of data CNOTs for `f` and `cap` generators are presented by the diagram in Fig. 6.9b. The circuit for measuring a generator of weight w of the code of distance 7 is a flag circuit with one flag ancilla as shown in Fig. 6.10a, and the orderings of data CNOTs for `f` and `cap` generators are presented by the diagram in Fig. 6.10b. (For the meanings of these diagrams, please refer to the description of the diagram presented in Section 6.1.2.) Since the fault sets in both cases are distinguishable, the fault-tolerant protocols in Section 6.3 are applicable. Note that the protocols for the capped color code of distance 5 and 7 only need one and two ancillas, respectively. This means that our protocols for the codes of distance 5 and 7 require 40 and 77 qubits in total.



(a) (b)

Figure 6.9: (a) A non-flag circuit for measuring a generator of the capped color code of distance 5 in H form, where w is the weight of the generator. (b) The orderings of data CNOT gates which give a distinguishable fault set $\mathcal{F}_2$.



(a) (b)

Figure 6.10: (a) A flag circuit for measuring a generator the capped color code of distance 7 in H form, where w is the weight of the generator. (b) The orderings of data CNOT gates which give a distinguishable fault set $\mathcal{F}_3$.

6.3 Fault-tolerant protocols for a capped color code

Previously in Section 4.2, we have shown that it is possible to redefine r-filter and ideal decoder as in Definitions 4.8 and 4.9 using the notions of distinguishable fault set (Definition 3.3) and distinguishable error set (Definition 4.7 or Definition 4.11), and redefine fault-tolerant gadgets as in Definition 4.10. These revised definitions give us more flexibility when designing fault-tolerant protocols, while ensuring that the simulated circuit constructed from these protocols still work fault-tolerantly. In Section 6.3.1, we will construct an FTEC protocol for a capped color code in H form of any distance in which its fault set is distinguishable. Note that having only the FTEC protocol is not enough for general fault-tolerant quantum computation, so we will also construct other fault-tolerant protocols which share the same distinguishable fault set with the FTEC protocol for a particular code in Sections 6.3.2 and 6.3.3.

6.3.1 Fault-tolerant error correction protocol

To construct an FTEC protocol for a capped color code in H form obtained from $CCC(d)$, we will first assume that the fault set $\mathcal{F}_t$ (where $t = (d-1)/2$) corresponding to the circuits for measuring the generators of the code is distinguishable, and the orderings of gates in the circuits for each pair of X-type and Z-type generators are the same. From the fact that $\mathcal{F}_t$ is distinguishable, we can build a list of all possible fault combinations and their corresponding combined error, syndrome of the combined error, and cumulative flag vector. Note that if several fault combinations have the same syndrome and cumulative flag vector, their combined errors are all logically equivalent (from Definition 3.3).

Let $\vec{\mathbf{s}} = (\vec{\mathbf{s}}_x|\vec{\mathbf{s}}_z)$ be the syndrome obtained from the measurements of X-type and Z-type generators, and let $\vec{\mathbf{f}} = (\vec{\mathbf{f}}_x|\vec{\mathbf{f}}_z)$ be the cumulative flag vector corresponding to the flag outcomes from the circuits for measuring X-type and Z-type generators, where $\vec{\mathbf{f}}$ is accumulated from the first round until the current round. We define the *outcome bundle* $(\vec{\mathbf{s}}, \vec{\mathbf{f}})$ to be the collection of $\vec{\mathbf{s}}$ and $\vec{\mathbf{f}}$ obtained during a single round of full syndrome measurement. An FTEC protocol for the capped color code in H form is as follows:

FTEC protocol for a capped color code in H form

During a single round of full syndrome measurement, measure the generators in the following order: measure v_i^x's, then f_i^x's, then v_i^z's, then f_i^z's. Perform full syndrome measurements until the outcome bundles $(\vec{\mathbf{s}}, \vec{\mathbf{f}})$ are repeated $t+1$ times in a row. Afterwards, do the following:

1. Determine an EC operator F_x using the list of possible fault combinations as follows:
   (a) If there is a fault combination on the list whose syndrome and cumulative flag vector are $(\vec{0}|\vec{\mathbf{s}}_z)$ and $(\vec{\mathbf{f}}_x|\vec{0})$, then F_x is the combined error of such a fault combination. (If there are more than one fault combination corresponding to $(\vec{0}|\vec{\mathbf{s}}_z)$ and $(\vec{\mathbf{f}}_x|\vec{0})$, a combined error of any of such fault combinations will work.)
   (b) If none of the fault combinations on the list corresponds to $(\vec{0}|\vec{\mathbf{s}}_z)$ and $(\vec{\mathbf{f}}_x|\vec{0})$, then F_x can be any Pauli X operator whose syndrome is $(\vec{0}|\vec{\mathbf{s}}_z)$.
2. Determine an EC operator F_z using the list of possible fault combinations:
   (a) If there is a fault combination on the list whose syndrome and cumulative flag vector are $(\vec{\mathbf{s}}_x|\vec{0})$ and $(\vec{0}|\vec{\mathbf{f}}_z)$, then F_z is the combined error of such a fault combination. (If there are more than one fault combination corresponding to $(\vec{\mathbf{s}}_x|\vec{0})$ and $(\vec{0}|\vec{\mathbf{f}}_z)$, a combined error of any of such fault combinations will work.)
   (b) If none of the fault combinations on the list corresponds to $(\vec{\mathbf{s}}_x|\vec{0})$ and $(\vec{0}|\vec{\mathbf{f}}_z)$, then F_z can be any Pauli Z operator whose syndrome is $(\vec{\mathbf{s}}_x|\vec{0})$.
3. Apply $F_x \cdot F_z$ to the data qubits to perform error correction.

To verify that the above EC protocol is fault tolerant according to the revised definition (Definition 4.10), we have to show that the two properties in Definition 4.4 are satisfied when the r-filter and the ideal decoder are defined as in Definitions 4.8 and 4.9 (instead of Definitions 4.1 and 4.2) and the distinguishable error set is defined as in Definition 4.11 (the circuits for X-type and Z-type generators of the capped color code in H form use similar gate orderings). Here we will assume that there are no more than t faults during the whole protocol. Therefore, the condition that the outcome bundles are repeated $t+1$ times in a row will be satisfied within $(t+1)^2$ rounds. We will divide the analysis into two cases: (1) the case that the last round of the full syndrome measurement has no faults, and (2) the case that the last round has some faults.

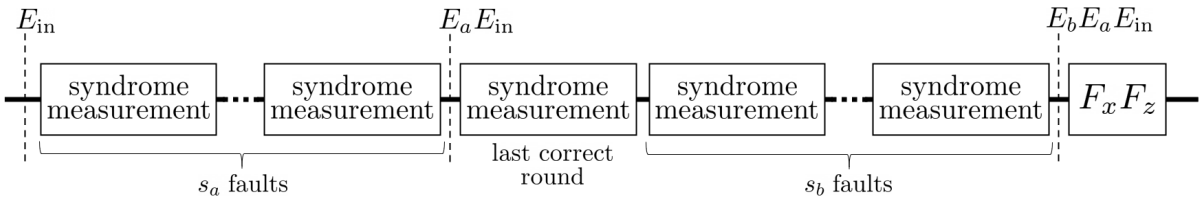


Figure 6.11: Fault-tolerant error correction protocol for a capped color code.

(1) Because the outcome bundles are repeated $t+1$ times and the last round of the full syndrome measurement has no faults, we know that the outcome bundle of the last round is correct and corresponds to the data error before the error correction in Step 3. Let E_{in} be the input error and E_a be the combined error of a fault combination arising from the s_a faults where $s_a \leq t$. The error on the data qubits before Step 3 is $E_a \cdot E_{\text{in}}$. First, consider the case that E_{in} is in $\mathcal{E}_r$ (defined in Definition 4.11) where $r + s_a \leq t$. Both E_{in} and E_a can be separated into X and Z parts. We find that the X part of E_{in} is in $\mathcal{E}_r^x$ (which is derived from $\mathcal{F}_r|_{\vec{\mathbf{f}}=0}$). Thus, the X part of $E_a \cdot E_{\text{in}}$ is the combined error of X type of some fault combination in $\mathcal{F}_{r+s_a}$. Similarly, the Z part of E_{in} is in $\mathcal{E}_r^z$, and the Z part of $E_a \cdot E_{\text{in}}$ is the combined error of Z type of some fault combination in $\mathcal{F}_{r+s_a}$. By picking EC operators F_x and F_z as in Steps 1a and 2a, Step 3 can completely remove the data error. Thus, both ECCP and ECRP in Definition 4.4 are satisfied. On the other hand, if E_{in} is not in $\mathcal{E}_r$ where $r + s_a \leq t$, the X part or the Z part of $E_a \cdot E_{\text{in}}$ might not correspond to any fault combination in $\mathcal{F}_t$. In this case, F_x or F_z will be picked as in Step 1b or 2b. Because the X part (or the Z part) of $E_a \cdot E_{\text{in}}$ and F_x (or F_z) have the same syndrome no matter how we pick F_x (or F_z), the output state after Step 3 is a valid codeword, but it may or may not be logically the same as the input state. In any cases, the output state can pass the s_a-filter, so the ECRP in Definition 4.4 is satisfied.

(2) In the case that the last round of the full syndrome measurement has some faults, the outcome bundle of the last round may not correspond to the data error before the error correction in Step 3. Fortunately, since the outcome bundles are repeated $t+1$ times in a row and there are no more than t faults during the whole protocol, we know that at least one round in the last $t+1$ rounds must be correct, and the outcome bundle of the last round must correspond to the data error right before the last correct round. Let E_{in} be the input error, E_a be the combined error arising from s_a faults which happen before the last correct round, and E_b be the combined error arising from s_b faults which happen after

the last correct round, where the total number of faults is $s = s_a + s_b \leq t$ (see Fig. 6.11). First, consider the case that E_{in} is in $\mathcal{E}_r$ where $r + s \leq t$. By an analysis similar to that presented in (1), we find that both X and Z parts of $E_a \cdot E_{\text{in}}$ are the combined errors of some fault combinations in $\mathcal{F}_{r+s_a}$, and F_x and F_z from Steps 1a and 2a can completely remove $E_a \cdot E_{\text{in}}$. Thus, the output data error after Step 3 is E_b. Since $s_b \leq t$ and the cumulative flag vectors do not change after the last correct round, we find that E_b is the combined error of some fault combination arising from s_b faults whose cumulative flag vector is zero; that is, E_b is in $\mathcal{E}_{s_b}$ where $s_b \leq t$. For this reason, E_b can pass the s-filter and can be corrected by the ideal decoder, meaning that both ECCP and ECRP in Definition 4.4 are satisfied. In contrast, if E_{in} is not in $\mathcal{E}_r$ where $r + s \leq t$, E_{in} may not correspond to any fault combination in $\mathcal{F}_t$, and F_x or F_z may be picked as in Step 1b or 2b. Similar to the previous analysis, $F_x \cdot F_z$ will have the same syndrome as that of $E_a \cdot E_{\text{in}}$. By an operation in Step 3, the output state will be a valid codeword with error E_b, which can pass the the s-filter. Therefore, the ECRP in Definition 4.4 is satisfied in this case.

In addition to the capped color code in H form, the FTEC protocol above is also applicable to any CSS code in which $\mathcal{F}_t$ is distinguishable and the possible X-type and Z-type errors are of the same form (i.e., a code to which Definition 4.11 is applicable for all $r \in \{1, \ldots, t\}$, $t \leq \lfloor (d-1)/2 \rfloor$). Besides this, we can also construct an FTEC protocol for a general stabilizer code whose circuits for the syndrome measurement give a distinguishable fault set (a code in which $\mathcal{E}_r$ is defined by Definition 4.7 instead of Definition 4.11) using similar ideas. An FTEC protocol for such a code is provided in Section 6.4.

6.3.2 Fault-tolerant measurement and state preparation protocols

Besides FTEC protocols, we also need other gadgets such as FTM, FTP, and FTG gadgets in order to perform fault-tolerant quantum computation. Note that the definitions of the r-filter (Definition 4.8) and the ideal decoder (Definition 4.9) depend on how the distinguishable error set is defined. Therefore, in order to utilize the new definitions of fault-tolerant gadgets in Definition 4.10, all protocols used in the computation must share the same definition of distinguishable error set. In this section, we will construct an FTM protocol for a capped color code in H form, which is also applicable to other CSS codes with similar properties. The distinguishable error set being used in the construction of

the FTM protocol will be similar to the distinguishable error set defined for the FTEC protocol for the same code. In addition, an FTP protocol can also be obtained from the FTM protocol.

We will start by constructing an FTM protocol for a capped color code in H form obtained from $CCC(d)$. The FTM protocol discussed below can be used to fault-tolerantly measure any logical X or logical Z operator of the form $X^{\otimes n}M$ or $Z^{\otimes n}N$, where M, N are some stabilizers. Let L be the logical operator being measured. We will assume that the circuits for measuring X-type and Z-type generators are similar to the ones used in the FTEC protocol for a capped color code, which give a distinguishable fault set $\mathcal{F}_t$ with $t = (d-1)/2$ (the list of possible fault combinations for the FTM protocol is the same as the list used in the FTEC protocol). In addition, we can always use a non-flag circuit with an arbitrary gate ordering for measuring L (since any error arising from the circuit faults can always be corrected as we will see later in the protocol analysis). For the FTM protocol, the outcome bundle will be defined as $(m, \vec{\mathbf{s}}, \vec{\mathbf{f}})$, where m is the measurement outcome of the logical operator L ($m = 0$ and $m = 1$ correspond to $+1$ and -1 eigenvalues of L), and $\vec{\mathbf{s}} = (\vec{\mathbf{s}}_x|\vec{\mathbf{s}}_z)$ and $\vec{\mathbf{f}} = (\vec{\mathbf{f}}_x|\vec{\mathbf{f}}_z)$ are the syndrome and the cumulative flag vector obtained from the measurements of X-type and Z-type generators ($\vec{\mathbf{f}}$ is accumulated from the first round until the current round). An FTM protocol is as follows:

FTM protocol for a capped color code in H form

During a single round of logical operator and full syndrome measurements, measure the operators in the following order: measure L, then v_i^x's, then f_i^x's, then v_i^z's, then f_i^z's. Perform logical operator and full syndrome measurements until the outcome bundles $(m, \vec{\mathbf{s}}, \vec{\mathbf{f}})$ are repeated $t+1$ times in a row. Afterwards, do the following:

1. Determine an EC operator F_x using the list of possible fault combinations as follows:
   (a) If there is a fault combination on the list whose syndrome and cumulative flag vector are $(\vec{0}|\vec{\mathbf{s}}_z)$ and $(\vec{\mathbf{f}}_x|\vec{0})$, then F_x is the combined error of such a fault combination. (If there are more than one fault combination corresponding to $(\vec{0}|\vec{\mathbf{s}}_z)$ and $(\vec{\mathbf{f}}_x|\vec{0})$, a combined error of any of such fault combinations will work.)
   (b) If none of the fault combinations on the list corresponds to $(\vec{0}|\vec{\mathbf{s}}_z)$ and $(\vec{\mathbf{f}}_x|\vec{0})$, then F_x can be any Pauli X operator whose syndrome is $(\vec{0}|\vec{\mathbf{s}}_z)$.

2. Determine an EC operator F_z using the list of possible fault combinations as follows:

   (a) If there is a fault combination on the list whose syndrome and cumulative flag vector are $(\vec{\mathbf{s}}_x|\vec{0})$ and $(\vec{0}|\vec{\mathbf{f}}_z)$, then F_z is the combined error of such a fault combination. (If there are more than one fault combination corresponding to $(\vec{\mathbf{s}}_x|\vec{0})$ and $(\vec{0}|\vec{\mathbf{f}}_z)$, a combined error of any of such fault combinations will work.)

   (b) If none of the fault combinations on the list corresponds to $(\vec{\mathbf{s}}_x|\vec{0})$ and $(\vec{0}|\vec{\mathbf{f}}_z)$, then F_z can be any Pauli Z operator whose syndrome is $(\vec{\mathbf{s}}_x|\vec{0})$.

3. Apply $F_x \cdot F_z$ to the data qubits to perform error correction.

4. If L and $F_x \cdot F_z$ anticommute, modify m from 0 to 1 or from 1 to 0. If L and $F_x \cdot F_z$ commute, do nothing.

5. Output m as the operator measurement outcome, where $m = 0$ and $m = 1$ correspond to $+1$ and -1 eigenvalues of L. If L is a logical Z operator, the output state is the logical $|0\rangle$ or logical $|1\rangle$ state for $m = 0$ or 1. If L is a logical X operator, the output state is the logical $|+\rangle$ or logical $|-\rangle$ state for $m = 0$ or 1.

To verify that the FTM protocol for a capped color code is fault tolerant according to the revised definition (Definition 4.10), we will show that both of the properties in Definition 4.6 is satisfied when the r-filter, the ideal decoder, and the distinguishable fault set $\mathcal{E}_r$ are defined as in Definitions 4.8, 4.9 and 4.11. The distinguishable fault set $\mathcal{F}_t$ for this protocol is the same fault set as the one defined for the FTEC protocol (i.e., $\mathcal{F}_t$ concerns the circuits for measuring X-type and Z-type generators, and does not concern the circuit for measuring L). We will also assume that there are no more than t faults during the whole protocol, so the outcome bundles must be repeated $t + 1$ times in a row within $(t + 1)^2$ rounds. First, suppose that the operator being measured L is a logical Z operator. The analysis will be divided into two cases: (1) the case that the last round of operator and full syndrome measurements has no faults, and (2) the case that the last round of operator and full syndrome measurements has some faults.

(1) Because the last round is correct and the outcome bundles are repeated $(t + 1)$ times in a row, $m, \vec{\mathbf{s}}$, and $\vec{\mathbf{f}}$ exactly correspond to the error on the state before Step 3. Let $E_{\text{in}} \in \mathcal{E}_r$ be the input error, E_a be the combined error arising from s_a faults in the circuits for measuring L, and E_b be the combined error arising from s_b faults in the syndrome

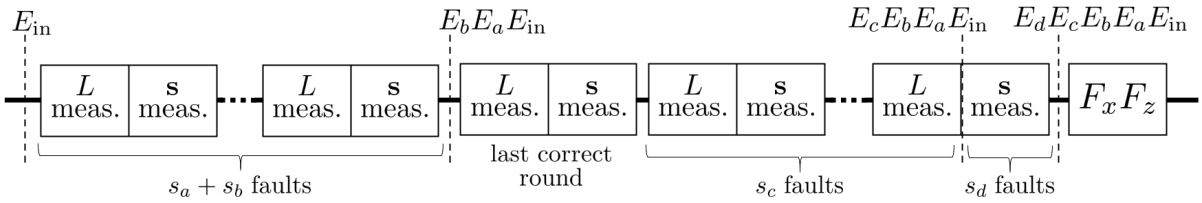


Figure 6.12: Fault-tolerant measurement protocol for a capped color code.

measurement circuits, where $r + s_a + s_b \leq t$. Also, assume that the (uncorrupted) input state is $|\bar{m}_{\text{in}}\rangle$ where $m_{\text{in}} = 0$ or 1. The data error on the state before the last round is $E_bE_aE_{\text{in}}$. Since L is of the form $Z^{\otimes n}N$ where N is some stabilizer, the X part of E_a has weight no more than s_a, while the Z part of E_a can be any Z-type error. We find that the X part of $E_bE_aE_{\text{in}}$, denoted as $(E_bE_aE_{\text{in}})_x$, is similar to a combined error of X type of some fault combination in $\mathcal{F}_{r+s_a+s_b}$. However, the Z part of $E_bE_aE_{\text{in}}$, denoted as $(E_bE_aE_{\text{in}})_z$, may or may not correspond to a Z-type error of some fault combination in $\mathcal{F}_t$. By picking F_x and F_z as in Steps 1 and 2, F_x is logically equivalent to $(E_bE_aE_{\text{in}})_x$ and F_z is logically equivalent to $(E_bE_aE_{\text{in}})_z$ or $(E_bE_aE_{\text{in}})_zZ^{\otimes n}$. So after the error correction in Step 3, the output state is $|\bar{m}_{\text{in}}\rangle$ or $Z^{\otimes n}|\bar{m}_{\text{in}}\rangle$. Note that $|\bar{m}_{\text{in}}\rangle$ and $Z^{\otimes n}|\bar{m}_{\text{in}}\rangle$ are the same state for both $m_{\text{in}} = 0$ and $m_{\text{in}} = 1$ cases (the -1 global phase can be neglected in the case of $m_{\text{in}} = 1$).

Next, let us consider the result m obtained from the last round, which tell us whether the state before the measurement of L during the last round is $+1$ or -1 eigenstate of L. We find that if $m_{\text{in}} = 0$, $m = 0$ whenever $E_bE_aE_{\text{in}}$ commutes with L, and $m = 1$ whenever $E_bE_aE_{\text{in}}$ anticommutes with L. On the other hand, if $m_{\text{in}} = 1$, $m = 1$ whenever $E_bE_aE_{\text{in}}$ commutes with L, and $m = 0$ whenever $E_bE_aE_{\text{in}}$ anticommutes with L. Also, note that $F_x \cdot F_z$ is either $E_bE_aE_{\text{in}}$ or $E_bE_aE_{\text{in}}Z^{\otimes n}$ and L is a logical Z operator, so $E_bE_aE_{\text{in}}$ commutes (or anticommutes) with L if and only if $F_x \cdot F_z$ commutes (or anticommutes) with L. Thus, we need to flip the output as in Step 4 whenever $F_x \cdot F_z$ anticommutes with L so that $m = m_{\text{in}}$. As a result, the measurement protocol gives an output state $|\bar{m}_{\text{in}}\rangle$ and its corresponding measurement outcome $m = m_{\text{in}}$ which reflect the uncorrupted input state.

Now, let us consider the case that the uncorrupted input state is of the form $\alpha|\bar{0}\rangle + \beta|\bar{1}\rangle$. If there is at least one round before the last correct round in which the measurement of L is correct, then the superposition state collapses and the state before the last correct

round is either $E_b E_a E_{\text{in}}|\bar{0}\rangle$ or $E_b E_a E_{\text{in}}|\bar{1}\rangle$, so the analysis above is applicable. However, if the measurements of L before the last correct round are all incorrect, it is possible that the superposition state may not collapse and the state before the last correct round is of the form $E_b E_a E_{\text{in}}(\alpha|\bar{0}\rangle + \beta|\bar{1}\rangle)$. Suppose that the measurement of L in the last correct round gives $m = 0$. Then the output state from the last correct round is a $+1$ eigenstate of L, which is $E_b E_a E_{\text{in}}|\bar{0}\rangle$ if $E_b E_a E_{\text{in}}$ commutes with L, or $E_b E_a E_{\text{in}}|\bar{1}\rangle$ if $E_b E_a E_{\text{in}}$ anticommutes with L. In constrast, if the measurement of L in the last correct round gives $m = 1$, then the output state from the last correct round is a -1 eigenstate of L. This state is $E_b E_a E_{\text{in}}|\bar{1}\rangle$ if $E_b E_a E_{\text{in}}$ commutes with L, or $E_b E_a E_{\text{in}}|\bar{0}\rangle$ if $E_b E_a E_{\text{in}}$ anticommutes with L. By applying $F_x \cdot F_z$ as in Step 3 and modifying m whenever $F_x \cdot F_z$ anticommutes with L as in Step 4, the outputs from the protocol are either $m = 0$ and $|\bar{0}\rangle$, or $m = 1$ and $|\bar{1}\rangle$ (up to some global phase). Therefore, both MCP and MPP in Definition 4.6 are satisfied.

(2) In the case that the last round has some faults, because the outcome bundles are repeated $(t + 1)$ times in a row and there are no more than t faults in the protocol, there must be at least one correct round in the last $t + 1$ rounds, and the outcome bundles correspond to the error on the state before the last correct round. Let $E_{\text{in}} \in \mathcal{E}_r$ be the input error, E_a be the combined error arising from s_a faults in the circuits for measuring L before the last correct round, E_b be the combined error arising from s_b faults in the syndrome measurement circuits before the last correct round, and E_c be the combined error arising from s_c faults in any circuits after the last correct round but before the syndrome measurement circuits of the very last round, and E_d be the combined error arising from s_d faults in the syndrome measurement circuits of the very last round, where $r + s_a + s_b + s_c + s_d \leq t$ (see Fig. 6.12). By an analysis similar to (1), we find that F_x from Step 1 is logically equivalent to $(E_b E_a E_{\text{in}})_x$, and F_z from Step 2 is logically equivalent to $(E_b E_a E_{\text{in}})_z$ or $(E_b E_a E_{\text{in}})_z Z^{\otimes n}$.

Now, let us consider E_c which can arise from the circuits for measuring L or the syndrome measurement circuits, and E_d which can arise from the syndrome measurement circuits. Because the syndromes and the cumulative flag vectors do not change after the last correct round, and because i faults in the circuits for measuring L cannot cause X-type error of weight more than i, the X part of E_c (denoted as $(E_c)_x$) is similar to the combined error of X type of a fault combination arising from s_c faults whose cumulative flag vector is zero, i.e., $(E_c)_x$ is an error in $\mathcal{E}^x_{s_c}$. In contrast, because the circuits for measuring L can cause Z-type error of any weight but the syndromes and the cumulative flag vectors do not

change after the last correct round, the Z part of E_c (denoted as $(E_c)_z$) can be written as $(\tilde{E}_c)_z$ or $(\tilde{E}_c)_z Z^{\otimes n}$, where $(\tilde{E}_c)_z \in \mathcal{E}^z_{s_c}$. That is, E_c is either $\tilde{E}_c$ or $\tilde{E}_c Z^{\otimes n}$ where $\tilde{E}_c \in \mathcal{E}_{s_c}$. For E_d which arising from s_d the syndrome measurement circuits in the very last round, we find that it is an error in $\mathcal{E}_{s_d}$ since the cumulative flag vector from the very last round remains the same.

Let the (uncorrupted) input state be of the form $\alpha|\bar{0}\rangle+\beta|\bar{1}\rangle$. Suppose that the measurement outcome of L from the last correct round is $m = 0$. From the argument on a superposition state in (1), we find that the output state from the last correct round is $E_b E_a E_{\text{in}}|\bar{0}\rangle$ if $E_b E_a E_{\text{in}}$ commutes with L, or $E_b E_a E_{\text{in}}|\bar{1}\rangle$ if $E_b E_a E_{\text{in}}$ anticommutes with L. Thus, the state before Step 3 is $E_d \tilde{E}_c E_b E_a E_{\text{in}}|\bar{0}\rangle$ or $E_d \tilde{E}_c Z^{\otimes n} E_b E_a E_{\text{in}}|\bar{0}\rangle$ if $E_b E_a E_{\text{in}}$ commutes with L, or $E_d \tilde{E}_c E_b E_a E_{\text{in}}|\bar{1}\rangle$ or $E_d \tilde{E}_c Z^{\otimes n} E_b E_a E_{\text{in}}|\bar{1}\rangle$ if $E_b E_a E_{\text{in}}$ anticommutes with L. Recall that $F_x \cdot F_z$ is either $E_b E_a E_{\text{in}}$ or $E_b E_a E_{\text{in}} Z^{\otimes n}$, and $E_b E_a E_{\text{in}}$ commutes (or anticommutes) with L if and only if $F_x \cdot F_z$ commutes (or anticommutes) with L. By applying $F_x \cdot F_z$ as in Step 3 and modifying m whenever $F_x \cdot F_z$ anticommutes with L as in Step 4, the protocol either outputs $m = 0$ with the output state $E_d \tilde{E}_c|\bar{0}\rangle$ (up to some global phase), or outputs $m = 1$ with the output state $E_d \tilde{E}_c|\bar{1}\rangle$ (up to some global phase). Similar results will be obtained in the case that the measurement outcome of L from the last correct round is $m = 1$.

Since $E_d \tilde{E}_c \in \mathcal{E}_s$ where $s = s_a + s_b + s_c + s_d$ and $r + s \leq t$, the output bit corresponds to the logical qubit of the output state in every case, and the output bit is 0 (or 1) if the (uncorrupted) input state is $|\bar{0}\rangle$ (or $|\bar{1}\rangle$), both of MCP and MPP in Definition 4.6 are satisfied. Similar analysis can be made for the case that L is a logical X operator. In that case, we will let $m = 0$ and $m = 1$ correspond to $|\bar{+}\rangle$ and $|\bar{-}\rangle$, and the analysis similar to (1) and (2) can be applied.

In addition, it is possible to construct an FTP protocol from the FTM protocol described above. For example, if we want to prepare the state $|\bar{0}\rangle$, we can do so by applying the FTM protocol for a logical Z operator to any state, then applying a logical X operator on the output state if $m = 1$ or do nothing if $m = 0$.

The FTM and the FTP protocols presented in this section is also applicable to any CSS code in which the number of encoded qubit is 1, $\mathcal{F}_t$ is distinguishable (where $\mathcal{F}_t$ corresponds to the circuits for measuring code generators), and the errors in $\mathcal{E}^x_r$ and $\mathcal{E}^z_r$ have the same form for all $r = 1, \ldots, t$, $t \leq \lfloor (d-1)/2 \rfloor$.

6.3.3 Transversal gates and other gate gadgets

From the properties of a capped color code in H form discussed in Section 6.2.1, we know that H, S, and CNOT gates are transversal. These gates can play an important role in fault-tolerant quantum computation because transversal gates satisfy both properties of fault-tolerant gate gadgets originally proposed in [AGP06] (Definition 4.3). However, since the definition of fault-tolerant gadgets being used in this work is revised as in Definition 4.10, transversal gates which satisfy the old definition may or may not satisfy the new one. In this section, we will show that transversal H, S, and CNOT gates are still fault tolerant according to the new definition of fault-tolerant gadgets when the distinguishable error set $\mathcal{E}_r$ of a capped color code in H form is defined as in Definition 4.11. Afterwards, we will also discuss some possibilities of building other fault-tolerant gate gadgets in order to achieve the universal set of quantum gates.

We start by observing the operations of H, S, and CNOT gates. These gates can transform Pauli operators as follows:

$$\begin{array}{rlll} H: & X \mapsto Z, & Y \mapsto -Y, & Z \mapsto X, \\ S: & X \mapsto Y, & Y \mapsto -X, & Z \mapsto Z, \\ \text{CNOT}: & XI \mapsto XX, & ZI \mapsto ZI, & \\ & IX \mapsto IX, & IZ \mapsto ZZ. & \end{array}$$

Meanwhile, the transversal H, S, and CNOT gates can map logical operators $\bar{X} = X^{\otimes n}$ and $\bar{Z} = Z^{\otimes n}$ as follows:

$$\begin{array}{rll} H^{\otimes n}: & \bar{X} \mapsto \bar{Z}, & \bar{Z} \mapsto \bar{X}, \\ S^{\otimes n}: & \bar{X} \mapsto -\bar{Y}, & \bar{Z} \mapsto \bar{Z}, \\ \text{CNOT}^{\otimes n}: & \bar{X} \otimes \bar{I} \mapsto \bar{X} \otimes \bar{X}, & \bar{Z} \otimes \bar{I} \mapsto \bar{Z} \otimes \bar{I}, \\ & \bar{I} \otimes \bar{X} \mapsto \bar{I} \otimes \bar{X}, & \bar{I} \otimes \bar{Z} \mapsto \bar{Z} \otimes \bar{Z}, \end{array}$$

where $\bar{I} = I^{\otimes n}$, $\bar{Y} = i\bar{X}\bar{Z} = -Y^{\otimes n}$, and $n = 3(d^2+1)/2$ is the total number of qubits for each $CCC(d)$ (since $d = 3, 5, 7, ...$, we find that $n = 3 \pmod 4$ and $\bar{Y} = -Y^{\otimes n}$ for any $CCC(d)$). In addition, the coding subspace is preserved under the operation of $H^{\otimes n}$, $S^{\otimes n}$, or $\text{CNOT}^{\otimes n}$ (i.e., each stabilizer is mapped to another stabilizer). Therefore, $H^{\otimes n}$, $S^{\otimes n}$, and $\text{CNOT}^{\otimes n}$ are logical H, logical $S^\dagger$, and logical CNOT gates, respectively.

Next, we will verify whether the new definition of fault-tolerant gate gadgets in Defini-

tion 4.10 is satisfied. Let the distinguishable error set $\mathcal{E}_r$ $(r = 1, \ldots, t)$ be defined as in Definition 4.11, where the distinguishable fault set $\mathcal{F}_t$ is the same fault set as the one defined for the FTEC protocol for a capped color code in H form. Suppose that the operation of $H^{\otimes n}$, $S^{\otimes n}$ or $\text{CNOT}^{\otimes n}$ has s faults, the input error of $H^{\otimes n}$ or $S^{\otimes n}$ is an error in $\mathcal{E}_r$ where $r + s \leq t$, and the input error of $\text{CNOT}^{\otimes n}$ is an error in $\mathcal{E}_{r_1} \times \mathcal{E}_{r_2}$ where $r_1 + r_2 + s \leq t$. The input error for $H^{\otimes n}$ and $S^{\otimes n}$ can be written as $E_1^x \cdot E_2^z$ where $E_1^x \in \mathcal{E}_r^x$ and $E_2^z \in \mathcal{E}_r^z$, and the input error for $\text{CNOT}^{\otimes n}$ can be written as $(E_3^x \otimes E_4^x) \cdot (E_5^z \otimes E_6^z)$ where $E_3^x \in \mathcal{E}_{r_1}^x$, $E_4^x \in \mathcal{E}_{r_2}^x$, $E_5^z \in \mathcal{E}_{r_1}^z$, $E_6^z \in \mathcal{E}_{r_2}^z$. Let E_i^x and E_i^z be X-type and Z-type operators which act on the same qubits. We find that,

1. $H^{\otimes n}$ maps $E_1^x \cdot E_2^z$ to $E_1^z \cdot E_2^x$, which is an error in $\mathcal{E}_r$.
2. $S^{\otimes n}$ maps $E_1^x \cdot E_2^z$ to $e^{i\theta} E_1^x \cdot E_1^z \cdot E_2^z$ for some phase $e^{i\theta}$ (which is ± 1 or $\pm i$), where $E_1^x \cdot E_1^z \cdot E_2^z$ is an error in $\mathcal{E}_r$.
3. $\text{CNOT}^{\otimes n}$ maps $(E_3^x \otimes E_4^x) \cdot (E_5^z \otimes E_6^z)$ to $(E_3^x \otimes E_3^x E_4^x) \cdot (E_5^z E_6^z \otimes E_6^z)$, which is an error in $\mathcal{E}_{r_1+r_2} \times \mathcal{E}_{r_1+r_2}$.

In addition, s faults during the application of $H^{\otimes n}$ or $S^{\otimes n}$ can cause an error in $\mathcal{E}_s$, and s faults during the application of $\text{CNOT}^{\otimes n}$ can cause an error in $\mathcal{E}_s \times \mathcal{E}_s$. Combining the input error and the error from faults, we find that an output error from $H^{\otimes n}$ or $S^{\otimes n}$ is an error in $\mathcal{E}_{r+s}$, while an output error from $\text{CNOT}^{\otimes n}$ is an error in $\mathcal{E}_{r_1+r_2+s} \times \mathcal{E}_{r_1+r_2+s}$. As a result, $H^{\otimes n}$, $S^{\otimes n}$, and $\text{CNOT}^{\otimes n}$ satisfy both GCP and GPP in Definition 4.3 when the r-filter, the ideal decoder, and the distinguishable error set are defined in Definitions 4.8, 4.9 and 4.11. That is, transversal H, S, and CNOT gates are fault tolerant according to the revised definition.

(Note that whether a transversal gate satisfies the revised definition of fault-tolerant gate gadgets in Definition 4.10 depends on how the distinguishable error set is defined (as in either Definition 4.7 or Definition 4.11). For example, if the input error E_{in} can arise from t faults (E_{in} is in $\mathcal{E}_t$) and a transversal gate transforms such an error to another error E_{out} which cannot arise from $\leq t$ faults (E_{out} is not in $\mathcal{E}_t$), then this transversal gate is not considered fault tolerant.)

Since the Clifford group can be generated by H, S, and CNOT [CRSS97, Got98], any Clifford gate can be fault-tolerantly implemented on the capped code in H form using transversal H, S, and CNOT gates. In order to achieve a universal set of quantum gates, we

also need a fault-tolerant implementation of some gate outside the Clifford group [NRS01]. One possible way to implement a non-Clifford gate on the capped color code in H form is to use magic state distillation [BK05], but large overhead might be required [FMMC12]. Another possible way is to perform code switching; since the code in H form possesses transversal H, S, and CNOT gates, and the code in T form possesses a transversal T gate, we can apply transversal H, S, or CNOT gates and perform FTEC on the code in H form, and switch to code in T form to apply a transversal T gate when necessary. However, a careful analysis to verify whether the traditional code switching method described in Section 6.2.1 is fault tolerant has yet to be done.

In [JOB16], the implementation of a transversal T gate is done on the $(d-1)+1$ stacked code while the code switching is done by the traditional procedure, and the whole process is fault tolerant since the code has distance d. However, the capped color code in T form has only distance 3 (even though it looks very similar to the stacked code), so the traditional method might not be fault tolerant in our case. Another way to perform code switching is to use the method involving a logical EPR state proposed in [BKS21], which works perfectly on the capped color code in the case of no faults. However, their error analysis for the case that some faults happen may not be directly applicable to our case since the capped color code and the traditional 3D color code have different structure. Nevertheless, it should be noted that possible errors from circuit faults depend heavily on the structure of the circuits involved in the code switching, and the FTEC protocol the capped color code developed in Section 6.3.1 can correct some errors of weight more than $\tau = \lfloor (d-1)/2 \rfloor$. Therefore, we are hopeful that circuits for measuring gauge operators could be carefully designed so that any possible errors from the circuit faults arising during the code switching are correctable by the FTEC protocol.

Another possible way to achieve universality is to design a gate gadget which can implement a logical T gate (or another non-Clifford gate) without using magic state distillation or code switching. To do so, we have to make sure that the newly-obtained error set (which includes possible errors arising from the FTEC protocol and the gadget implementing a non-Clifford gate) is still distinguishable, so that any errors arising during the computation can be corrected by the FTEC protocol.

6.4 Fault-tolerant error correction protocol for a general stabilizer code

In Section 6.3.1, we construct an FTEC protocol for a capped color code in H form of any distance in which its fault set is distinguishable. We also show that such a protocol is fault tolerant when the r-filter, the ideal decoder, and the distinguishable error set are defined as in Definitions 4.8, 4.9 and 4.11. Using similar ideas, we can also construct an FTEC protocol for a general stabilizer code whose circuits for the syndrome measurement give a distinguishable fault set $\mathcal{F}_t$, i.e., a code in which $\mathcal{E}_r$ is defined by Definition 4.7 instead of Definition 4.11. The outcome bundle defined for the protocol in this section is similar to the outcome bundle defined for the FTEC protocol for a capped color code, except that the syndrome $\vec{\mathbf{s}}$ and the cumulative flag vector $\vec{\mathbf{f}}$ are not separated into X and Z parts. We can also build a list of all possible fault combinations and their corresponding combined error and cumulative vector from the distinguishable fault set $\mathcal{F}_t$. The FTEC protocol for a general stabilizer code is as follows:

FTEC protocol for a stabilizer code whose syndrome measurement circuits give a distinguishable fault set

During a single round of full syndrome measurement, measure the all generators in any order. Perform full syndrome measurements until the outcome bundles $(\vec{\mathbf{s}}, \vec{\mathbf{f}})$ are repeated $t+1$ times in a row. Afterwards, do the following:

1. Determine an EC operator F using the list of possible fault combinations as follows:
   (a) If there is a fault combination on the list whose syndrome and cumulative flag vector are $\vec{\mathbf{s}}$ and $\vec{\mathbf{f}}$, then F is the combined error of such a fault combination. (If there are more than one fault combination corresponding to $\vec{\mathbf{s}}$ and $\vec{\mathbf{f}}$, a combined error of any of such fault combinations will work since they are logically equivalent.)
   (b) If none of the fault combinations on the list corresponds to $\vec{\mathbf{s}}$ and $\vec{\mathbf{f}}$, then F can be any Pauli operator whose syndrome is $\vec{\mathbf{s}}$.
2. Apply F to the data qubits to perform error correction.

To verify that the FTEC protocol for a general stabilizer code satisfies both properties of an FTEC gadget according to the revised definition (Definition 4.10), we can use an analysis similar to that presented in Section 6.3.1, except that $\mathcal{E}_r$ is defined by Definition 4.7 instead of Definition 4.11 and the errors in the analysis (E_{in}, E_a, and E_b) need not be separated into X and Z parts.

Chapter 7

Discussion and Conclusions

So far, the FTEC techniques for two families of codes developed in [TL21b] and [TL21a] have been elaborated in this thesis. In particular, the FTEC protocol for the $[\![49, 1, 9]\!]$ concatenated Steane code is presented in Chapter 5, while the notion of distinguishable fault set, the revised definitions of fault-tolerant gadgets, and the fault-tolerant protocols for a capped color code are provided in Chapters 3, 4 and 6. In this chapter, we will discuss the main results of [TL21b] and [TL21a], and provide possible directions for future work.

7.1 Fault-tolerant error correction for the 49-qubit concatenated Steane code

In [TL21b], we prove the logical equivalence between errors of any weight on 7 qubits which have the same weight parity and correspond to the same error syndrome when error detection is performed by the $[\![7, 1, 3]\!]$ Steane code in Claim 5.1. From this result, we introduce the WPEC technique in Definition 5.1, which can correct errors of any weight on 7 qubits whenever their weight parity is known. We show that the WPEC technique can be extended to error correction in subblocks of the $[\![49, 1, 9]\!]$ concatenated Steane code, and we prove the sufficient condition for WPEC in Claim 5.2. Afterwards, we provide a family of circuits and an FTEC protocol for the $[\![49, 1, 9]\!]$ code which can correct up to 3 faults. We also point out that the WPEC technique seems applicable to FTEC schemes

for other codes such as the concatenated Golay code and concatenated Steane code with more than 2 levels of concatenation.

Since the FTEC protocol provided in this work satisfies the definition of FTEC in Definition 5.2 with $t = 3$, we can guarantee that the logical error rate is suppressed to $O(p^4)$ whenever the physical error rate is p under the random Pauli noise model. Note that we did not use the full ability of a code with distance 9 which, in principle, can correct up to 4 errors. In terms of error suppression, our FTEC protocol is as good as typical FTEC protocols for a concatenated code which are constructed by replacing each physical qubit with a code block and replacing each physical gate with the corresponding logical gate [AGP06].

One major advantage of our FTEC protocol in [TL21b] is that only 2 ancillas are needed: one ancilla for a syndrome measurement result and another ancilla for a flag measurement result (assuming that the qubit preparation and measurement are fast compared to the gate operation time). As a result, our protocol requires 51 qubits in total. The number of required qubits is very low compared to other FTEC protocols for the $[\![49, 1, 9]\!]$ code; the FTEC schemes in [YK17, CR18c] extended to the $[\![49, 1, 9]\!]$ code require 63 qubits in total (the minimum number of required ancillas is 14 assuming that they are recyclable). Meanwhile, the FTEC protocol in [Rei20] which extracts multiple syndromes at once encodes 2 logical qubits and requires no ancilla, but needs to work on two code blocks of 98 qubits in total. Our protocol might not have the fewest total number of qubits compared with other protocols for a different code which can correct up to 3 faults; for example, the flag FTEC protocol in [CR20] applying to the $[\![37, 1, 7]\!]$ 2D color code requires 45 qubits in total. Nevertheless, our work provides a substantial improvement over other FTEC protocols for a *concatenated* code, an approach that is still advantageous in some circumstances. Furthermore, the use of weight parities in error correction may be extended to other families of codes as discussed in [TL21a]. We believe that if the protocol requires fewer ancillas, the number of total locations will decrease, which can result in higher accuracy threshold. However, a simulation with careful analysis is required for the accuracy threshold calculation, thus we leave this for future work.

The protocol in Section 5.2 which can correct up to 3 faults exploits two techniques; the flag technique which partitions set of possible errors using flag measurement results, and the WPEC technique which corrects errors of any weight using their syndromes and weight parities. It should be emphasized that flag ancillas are not necessarily required for

a protocol exploiting WPEC technique; we find that a protocol which uses circuits similar to a circuit in Fig. 5.2a for 2nd-level syndrome measurements and uses non-flag circuits for 1st-level measurements can correct up to 2 faults.

We point out that the permutation of CNOT gates in the syndrome extraction circuits that make the protocol satisfies Claim 5.2 is not unique. We choose the permutation in Eq. (5.3) by using the fact that a CSS code constructed from classical cyclic codes can distinguish high-weight errors in the consecutive form [TCL20]. In particular, the circuit is designed in the way that high-weight errors arising in each subblock can be determined by the underlying $[\![7, 1, 3]\!]$ code in cyclic form. We did not prove the optimality of the choice of gate permutation in our protocol, so an FTEC protocol for the $[\![49, 1, 9]\!]$ code with only one ancilla or a protocol that can correct up to 4 faults might be possible.

Last, we note that the WPEC technique introduced in [TL21b] is not limited to the $[\![49, 1, 9]\!]$ concatenated Steane code. In Section 5.3, we prove the logical equivalence of errors with the same syndrome and weight parity for the $[\![23, 1, 7]\!]$ Golay code in Claim 5.3 and provide a WPEC scheme in Definition 5.3, which shows that WPEC can correct some high-weight errors in a subblock of the $[\![529, 1, 49]\!]$ concatenated Golay code. In addition, we expect that WPEC can be applied to any concatenated Steane code with more than 2 levels of concatenation in a similar fashion. However, circuits and a protocol must be carefully designed so that the full error correction ability of the code can be achieved. Another interesting future direction would be extending the WPEC technique to other families of quantum codes (such as the families of codes discussed in [TL21a]).

7.2 Fault-tolerant error correction and quantum computation for capped color codes

In [TL21a], we observe that errors arising from a few faults depend on the structure of the circuits chosen for syndrome measurement, and develop an FTEC protocol accordingly. A fault set which includes all possible fault combinations arising from at most a certain number of faults is said to be distinguishable if any pair of fault combinations in the set either lead to logically equivalent data errors, or lead to different syndromes or cumulative flag vectors (as defined in Definition 3.3). Distinguishability may depend on the number of flag ancillas being used in the circuits, the ordering of gates in the circuits, and the choice

of stabilizer generators being measured. If we can find a set of circuits for a stabilizer code which leads to a distinguishable fault set, we can construct an FTEC protocol, as shown in Sections 6.3.1 and 6.4.

We prove in Lemma 3.1 that if an $[\![n, k, d]\!]$ CSS code has odd n, $k = 1$, even weight stabilizer generators, and logical X and Z being $X^{\otimes n}$ and $Z^{\otimes n}$, then two Pauli errors of X type (or Z type) with the same syndrome are logically equivalent if and only if they have the same weight parity. One may notice that the weight parity of a Pauli operator and the anticommutation between the Pauli operator and a logical operator are closely related. In fact, for a given stabilizer code, the normalizer group can be generated by the stabilizer generators of the code and all independent logical Pauli operators; for example, the normalizer group of the Steane code is $N(S) = \langle g_i^x, g_i^z, X^{\otimes 7}, Z^{\otimes 7} \rangle_{i=1,2,3}$. If the anticommutation between a Pauli error E and each of the generators of $N(S)$ can be found, then a Pauli error logically equivalent to E can be determined with certainty. The EC techniques presented in Chapters 5 and 6 use the fact that the weight parity of an error on a smaller code (or the anticommutation between the error and a logical operator of a smaller code) can be inferred by the measurement results of the stabilizer generators of a bigger code. We are hopeful that the relationship between the weight parity and the anticommutation can lead to EC techniques similar to the weight parity technique for a general stabilizer code in which the number of logical qubits can be greater than 1.

With Lemma 3.1 in mind, we present the 3D color code of distance 3 in Section 6.1 and construct a family of capped color codes in Section 6.2, which are good candidates for our protocol construction (the 3D color code of distance 3 is the smallest capped color code). A capped color code is a subsystem code; it can be transformed to stabilizer codes, namely capped color codes in H form and T form, by the gauge fixing method. The code in H form has transversal H, S, and CNOT gates, while the code in T form has transversal CNOT and T gates. One interesting property of a capped color code in H form is that the code contains a 2D color code as a subcode lying on the center plane. Since a `cap` generator of X type (or Z type) has support on all qubits on the center plane, the weight parity of an error of Z type (or X type) occurred on the center plane can be obtained from the measurement result of the `cap` generator. The syndrome of the error on the center plane corresponding to the measurements of the 2D code generators together with the error weight parity can lead to an EC operator for such an error by Lemma 3.1. Exploiting these facts, we design circuits for measuring generators of a capped color code such that most of the possible

errors are on the center plane. We prove in Theorem 6.1 that if the circuits satisfy some conditions, the fault set corresponding to all possible fault combinations arising from up to $t = (d-1)/2$ faults is distinguishable, where $d = 3, 5, 7, ...$ is the distance of the code. We also provide examples of circuit construction for capped color codes in H form of distance 3, 5, and 7 which can give distinguishable fault sets. The circuits for these codes require only 1, 1, and 2 ancillas, respectively (including syndrome and flag ancillas). This means that the FTEC protocols for capped color codes of distance 3, 5, and 7 (which can correct up to 1, 2, and 3 faults) require 16, 40, and 77 qubits in total.

The sufficient conditions of Theorem 6.1 give us some possibilities to construct good syndrome measurement circuits for a capped color code in H form of any distance. The intuition behind the proof of Theorem 6.1 is that several conditions are introduced in order to prevent indistinguishable pair of fault combinations from happening. It should be noted that (1) some conditions may imply one another, (2) some conditions might be automatically satisfied from the structure of the 2D color code (which is a subcode of a capped color code in H form), and (3) some conditions can be satisfied with the use of only few flag ancillas. We are hopeful that the conditions in Theorem 6.1 will lead to FTEC protocols for capped color codes of any distance in which the number of required ancillas does not grow rapidly as the distance grows. However, careful analysis of the relationship between these conditions and the number of required ancillas is required, thus we leave this for future work.

In Section 6.3, we construct several fault-tolerant protocols using the fact that the fault set corresponding to the protocols being used is distinguishable. Our definitions of fault-tolerant gadgets in Definition 4.10 also take the fact that some errors can be distinguished by their relevant flag information, so they can be viewed as a generalization of the definitions of fault-tolerant gadgets proposed in [AGP06] (Definitions 4.3 to 4.6). Our protocols are not limited to the capped color codes; some of the protocols are also applicable to other families of stabilizer codes if their syndrome measurement circuits give a distinguishable fault set. Since possible errors depend on every fault-tolerant gadget being used, all protocols for quantum computation (including error correction, gate, measurement, and state preparation gadgets) must be designed in tandem in order to achieve fault tolerance. Note that in this work we only use the cumulative flag vector within the FTEC protocol; all fault-tolerant gadgets defined in Definition 4.10 does not output any flag information, and the ideal decoder defined in Definition 4.9 does not take any flag information as an

input. One interesting future direction would be studying how fault-tolerant protocols can be further improved by exploiting the flag information outside of the FTEC protocol.

One should note that it is possible to use fault-tolerant protocols satisfying the old definitions of fault-tolerant gadgets (Definitions 4.3 to 4.6) in conjunction with fault-tolerant protocols satisfying our definitions of fault-tolerant gadgets (Definition 4.10). In particular, observe that for any Pauli error of weight $w \leq t$, we can always find a fault combination arising from w faults whose combined error is such an error; i.e., any Pauli error of weight up to t is contained in a distinguishable fault set $\mathcal{F}_t$. Therefore, an FTEC protocol satisfying Definition 4.10 can be used to correct an output error of any fault-tolerant protocol satisfying one of the old definitions (assuming that both protocols can tolerate the same number of faults). However, the converse might not be true since an FTEC protocol satisfying the old definition of FTEC gadget might not be able to correct errors of high weight arising from a small number of faults in the protocol satisfying Definition 4.10.

In Section 6.3, we provide FTEC, FTM, and FTP protocols for a capped color code in H form which are applicable to the code of any distance as long as the fault set is distinguishable. We also show that transversal H, S, and CNOT gates are fault tolerant (according to our definitions of fault-tolerant gadgets in Definition 4.10). These gates and protocols are sufficient for implementing any Clifford operation on the capped color code fault-tolerantly. In order to achieve the universal set of quantum gates, we still need an implementation of a non-Clifford gate. One possible approach is to use magic state distillation [BK05], which can be done fault-tolerantly on the capped color code. Another approach is to perform code switching [PR13, ADCP14, Bom15a, KB15] and apply a transversal T gate on the code in T form if necessary. Further study is required to show whether the traditional code switching method (as in Section 6.2.1) or an alternative method involving a logical EPR state (as proposed in [BKS21]) is fault tolerant when applied to the capped color code. Nevertheless, we hope that the circuits for measuring gauge operators could be designed so that any possible errors are correctable by our FTEC protocol (i.e., the fault set is still distinguishable).

Compared with other fault-tolerant protocols for other codes with the same distance, our protocols for a capped color code may not have the fewest total number of qubits (including data qubits and all ancillas). For example, the flag FTEC protocols from [CR20] applied to the 2D color codes of distance 5 and 7 require 25 and 45 total qubits, while the FTEC protocol using error weight parities from [TL21b] applied to the $[\![49, 1, 9]\!]$ concatenated

Steane code can correct up to 3 faults and requires 51 total qubits. However, one advantage that the capped color codes have over the aforementioned codes is that a transversal T gate can be operated on the capped color codes through code switching. Recently, Beverland, Kubica, and Svore [BKS21] compare the overhead required for T gate implementation with two methods: using a 2D color code via magic state distillation versus using a (traditional) 3D color code via code switching. They found that magic state distillation outperforms code switching except at some low physical error rate and when certain fault-tolerant schemes are used in the simulation. Since our protocols require only a few ancillas and the data block of a capped color code is much smaller than that of a 3D color code of the same distance, we are hopeful that the range of physical error rate in which code switching beats magic state distillation could be improved by our protocols. A careful simulation on the overhead is required, thus we leave this for future work.

Last, we point out that our fault-tolerant protocols using the flag and the weight parity techniques are specially designed for the *circuit-level noise* so that all possible data errors arising from a few faults (including any 1- and 2-qubit gate faults, faults during the ancilla preparation and measurement, and faults during wait time) can be corrected. However, our protocols require repeated syndrome measurements in order to avoid syndrome bit flips which may occur during the protocols, and the processes can increase the number of gate operations. The single-shot error correction [Bom15b] is one technique that can deal with the syndrome bit flips without using repeated syndrome measurements. We hope that the flag, the weight parity, and the single-shot error correction techniques could be used together to build fault-tolerant protocols which can protect the data against the circuit-level noise and require only small numbers of gates and ancillas.

Bibliography

[ABCB14] Hussain Anwar, Benjamin J Brown, Earl T Campbell, and Dan E Browne. Fast decoders for qudit topological codes. *New Journal of Physics*, 16(6):063038, 2014.

[ABO08] Dorit Aharonov and Michael Ben-Or. Fault-tolerant quantum computation with constant error rate. *SIAM Journal on Computing*, 2008.

[ADCP14] Jonas T Anderson, Guillaume Duclos-Cianci, and David Poulin. Fault-tolerant conversion between the Steane and Reed-Muller quantum codes. *Physical review letters*, 113(8):080501, 2014.

[AGP06] Panos Aliferis, Daniel Gottesman, and John Preskill. Quantum accuracy threshold for concatenated distance-3 codes. *Quantum Information and Computation*, 6(2):97–165, 2006.

[AL06] Panos Aliferis and Debbie W Leung. Simple proof of fault tolerance in the graph-state model. *Physical Review A*, 73(3):032308, 2006.

[Bac06] Dave Bacon. Operator quantum error-correcting subsystems for self-correcting quantum memories. *Physical Review A*, 73(1):012340, 2006.

[BC15] Sergey Bravyi and Andrew Cross. Doubled color codes. *arXiv preprint arXiv:1509.03239*, 2015.

[BCC+19] Paul Baireuther, MD Caio, B Criger, Carlo WJ Beenakker, and Thomas E O'Brien. Neural network decoder for topological color codes with circuit level noise. *New Journal of Physics*, 21(1):013003, 2019.

[BH13] Sergey Bravyi and Jeongwan Haah. Quantum self-correction in the 3d cubic code model. *Physical review letters*, 111(20):200501, 2013.

[BK98] Sergey B Bravyi and A Yu Kitaev. Quantum codes on a lattice with boundary. *arXiv preprint quant-ph/9811052*, 1998.

[BK05] Sergey Bravyi and Alexei Kitaev. Universal quantum computation with ideal Clifford gates and noisy ancillas. *Physical Review A*, 71(2):022316, 2005.

[BK13] Sergey Bravyi and Robert König. Classification of topologically protected gates for local stabilizer codes. *Physical review letters*, 110(17):170503, 2013.

[BKS21] Michael E Beverland, Aleksander Kubica, and Krysta M Svore. Cost of universality: A comparative study of the overhead of state distillation and code switching with color codes. *PRX Quantum*, 2(2):020341, 2021.

[BLP+16] Benjamin J Brown, Daniel Loss, Jiannis K Pachos, Chris N Self, and James R Wootton. Quantum memories at finite temperature. *Reviews of Modern Physics*, 88(4):045005, 2016.

[BMD06] Hector Bombin and Miguel Angel Martin-Delgado. Topological quantum distillation. *Physical review letters*, 97(18):180501, 2006.

[BNB16] Benjamin J Brown, Naomi H Nickerson, and Dan E Browne. Fault-tolerant error correction with the gauge color code. *Nature communications*, 7(1):1–8, 2016.

[Bom15a] Héctor Bombín. Gauge color codes: optimal transversal gates and gauge fixing in topological stabilizer codes. *New Journal of Physics*, 17(8):083002, 2015.

[Bom15b] Héctor Bombín. Single-shot fault-tolerant quantum error correction. *Physical Review X*, 5(3):031043, 2015.

[Bra11] Sergey Bravyi. Subsystem codes with spatially local generators. *Physical Review A*, 83(1):012320, 2011.

[BSV14] Sergey Bravyi, Martin Suchara, and Alexander Vargo. Efficient algorithms for maximum likelihood decoding in the surface code. *Physical Review A*, 90(3):032326, 2014.

[BXG+19] A Bermudez, X Xu, M Gutiérrez, SC Benjamin, and M Müller. Fault-tolerant protection of near-term trapped-ion topological qubits under realistic noise sources. *Physical Review A*, 100(6):062307, 2019.

[CB18] Christopher Chamberland and Michael E. Beverland. Flag fault-tolerant error correction with arbitrary distance codes. *Quantum*, 2:53, February 2018.

[CBDH20] Rui Chao, Michael E Beverland, Nicolas Delfosse, and Jeongwan Haah. Optimization of the surface code design for Majorana-based qubits. *Quantum*, 4:352, 2020.

[CC19] Christopher Chamberland and Andrew W Cross. Fault-tolerant magic state preparation with flag qubits. *Quantum*, 3:143, 2019.

[CJOL17] Christopher Chamberland, Tomas Jochym-O'Connor, and Raymond Laflamme. Overhead analysis of universal concatenated quantum codes. *Physical Review A*, 95(2):022313, 2017.

[CKYZ20] Christopher Chamberland, Aleksander Kubica, Theodore J Yoder, and Guanyu Zhu. Triangular color codes on trivalent graphs with flag qubits. *New Journal of Physics*, 22(2):023019, 2020.

[CN20] Christopher Chamberland and Kyungjoo Noh. Very low overhead fault-tolerant magic state preparation using redundant ancilla encoding and flag qubits. *npj Quantum Information*, 6(1):1–12, 2020.

[CR18a] Christopher Chamberland and Pooya Ronagh. Deep neural decoders for near term fault-tolerant experiments. *Quantum Science and Technology*, 3(4):044002, 2018.

[CR18b] Rui Chao and Ben W. Reichardt. Fault-tolerant quantum computation with few qubits. *npj Quantum Information*, 4(1):42, 2018.

[CR18c] Rui Chao and Ben W Reichardt. Quantum error correction with only two extra qubits. *Physical review letters*, 121(5):050502, 2018.

[CR20] Rui Chao and Ben W Reichardt. Flag fault-tolerant error correction for any stabilizer code. *PRX Quantum*, 1(1):010302, 2020.

[CRSS97] A Robert Calderbank, Eric M Rains, Peter W Shor, and Neil JA Sloane. Quantum error correction and orthogonal geometry. *Physical Review Letters*, 78(3):405, 1997.

[CS96] A Robert Calderbank and Peter W Shor. Good quantum error-correcting codes exist. *Physical Review A*, 54(2):1098, 1996.

[CZY+20] Christopher Chamberland, Guanyu Zhu, Theodore J Yoder, Jared B Hertzberg, and Andrew W Cross. Topological and subsystem codes on low-degree graphs with flag qubits. *Physical Review X*, 10(1):011022, 2020.

[DA07] David P DiVincenzo and Panos Aliferis. Effective fault-tolerant quantum computation with slow measurements. *Physical review letters*, 98(2):020501, 2007.

[DB20] Dripto M Debroy and Kenneth R Brown. Extended flag gadgets for low-overhead circuit verification. *Physical Review A*, 102(5):052409, 2020.

[DBT18] Kasper Duivenvoorden, Nikolas P Breuckmann, and Barbara M Terhal. Renormalization group decoder for a four-dimensional toric code. *IEEE Transactions on Information Theory*, 65(4):2545–2562, 2018.

[DCP10] Guillaume Duclos-Cianci and David Poulin. Fast decoders for topological quantum codes. *Physical review letters*, 104(5):050504, 2010.

[DCP13] Guillaume Duclos-Cianci and David Poulin. Kitaev's Z d-code threshold estimates. *Physical Review A*, 87(6):062338, 2013.

[Del14] Nicolas Delfosse. Decoding color codes by projection onto surface codes. *Physical Review A*, 89(1):012317, 2014.

[Deu85] David Deutsch. Quantum theory, the Church–Turing principle and the universal quantum computer. *Proceedings of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences*, 400(1818):97–117, 1985.

[DKLP02] Eric Dennis, Alexei Kitaev, Andrew Landahl, and John Preskill. Topological quantum memory. *Journal of Mathematical Physics*, 43(9):4452–4505, 2002.

[DP18] Andrew S Darmawan and David Poulin. Linear-time general decoding algorithm for the surface code. *Physical Review E*, 97(5):051302, 2018.

[EK09] Bryan Eastin and Emanuel Knill. Restrictions on transversal encoded quantum gate sets. *Physical review letters*, 102(11):110502, 2009.

[Fey82] R. P. Feynman. Simulating physics with computers. *International Journal of Theoretical Physics*, 21(6):467–488, 1982.

[FMMC12] Austin G Fowler, Matteo Mariantoni, John M Martinis, and Andrew N Cleland. Surface codes: Towards practical large-scale quantum computation. *Physical Review A*, 86(3):032324, 2012.

[GMB19] M Gutiérrez, M Müller, and Alejandro Bermúdez. Transversality and lattice surgery: Exploring realistic routes toward coupled logical qubits with trapped-ion quantum processors. *Physical Review A*, 99(2):022330, 2019.

[Got96] Daniel Gottesman. Class of quantum error-correcting codes saturating the quantum Hamming bound. *Physical Review A*, 54(3):1862, 1996.

[Got97] Daniel Gottesman. *Stabilizer Codes and Quantum Error Correction*. PhD thesis, California Institute of Technology, 1997.

[Got98] Daniel Gottesman. Theory of fault-tolerant quantum computation. *Physical Review A*, 57(1):127, 1998.

[JBH16] Cody Jones, Peter Brooks, and Jim Harrington. Gauge color codes in two dimensions. *Physical Review A*, 93(5):052332, 2016.

[JOB16] Tomas Jochym-O'Connor and Stephen D Bartlett. Stacked codes: Universal fault-tolerant quantum computation in a two-dimensional layout. *Physical Review A*, 93(2):022323, 2016.

[KB15] Aleksander Kubica and Michael E Beverland. Universal transversal gates with color codes: A simplified approach. *Physical Review A*, 91(3):032330, 2015.

[KD19] Aleksander Kubica and Nicolas Delfosse. Efficient color code decoders in $d \geq 2$ dimensions from toric code decoders. *arXiv preprint arXiv:1905.07393*, 2019.

[Kit97] Alexei Y. Kitaev. Quantum computations: algorithms and error correction. *Russian Mathematical Surveys*, 52(6):1191–1249, 1997.

[KKL+17] Torsten Karzig, Christina Knapp, Roman M Lutchyn, Parsa Bonderson, Matthew B Hastings, Chetan Nayak, Jason Alicea, Karsten Flensberg, Stephan Plugge, Yuval Oreg, et al. Scalable designs for quasiparticle-poisoning-protected topological quantum computation with Majorana zero modes. *Physical Review B*, 95(23):235305, 2017.

[KL97] Emanuel Knill and Raymond Laflamme. Theory of quantum error-correcting codes. *Physical Review A*, 55(2):900, 1997.

[KLZ96] Emanuel Knill, Raymond Laflamme, and Wojciech H. Zurek. Threshold accuracy for quantum computation. *arXiv: quant-ph/9610011*, 1996.

[Kni05] Emanuel Knill. Scalable quantum computing in the presence of large detected-error rates. *Physical Review A*, 71(4):042322, 2005.

[KP19] Aleksander Kubica and John Preskill. Cellular-automaton decoders with provable thresholds for topological codes. *Physical review letters*, 123(2):020501, 2019.

[KYP15] Aleksander Kubica, Beni Yoshida, and Fernando Pastawski. Unfolding the color code. *New Journal of Physics*, 17(8):083026, 2015.

[LA20] Lingling Lao and Carmen G Almudever. Fault-tolerant quantum error correction on near-term quantum processors using flag and bridge qubits. *Physical Review A*, 101(3):032333, 2020.

[MKJO19] Nishad Maskara, Aleksander Kubica, and Tomas Jochym-O'Connor. Advantages of versatile neural-network decoding for topological codes. *Physical Review A*, 99(5):052351, 2019.

[MS77] F. J. MacWilliams and N. J. A. Sloane. *The theory of error-correcting codes*. North-Holland, New York, 1977.

[NB19] Naomi H Nickerson and Benjamin J Brown. Analysing correlated noise on the surface code using adaptive decoding algorithms. *Quantum*, 3:131, 2019.

[NC00] Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information*. Cambridge University Press, Cambridge, England, 2000.

[ND05] Michael A Nielsen and Christopher M Dawson. Fault-tolerant quantum computation with cluster states. *Physical Review A*, 71(4):042323, 2005.

[NRS01] Gabriele Nebe, Eric M. Rains, and Neil JA Sloane. The invariants of the Clifford groups. *Designs, Codes and Cryptography*, 24(1):99–122, 2001.

[Pou05] David Poulin. Stabilizer formalism for operator quantum error correction. *Physical review letters*, 95(23):230504, 2005.

[PR12] Adam Paetznick and Ben W. Reichardt. Fault-tolerant ancilla preparation and noise threshold lower bounds for the 23-qubit Golay code. *Quantum Information and Computation*, 12(11-12):1034–1080, November 2012.

[PR13] Adam Paetznick and Ben W Reichardt. Universal fault-tolerant quantum computation with only transversal gates and error correction. *Physical review letters*, 111(9):090505, 2013.

[Pre98] John Preskill. Reliable quantum computers. *Proceedings of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences*, 454(1969):385–410, 1998.

[PY15] Fernando Pastawski and Beni Yoshida. Fault-tolerant logical gates in quantum error-correcting codes. *Physical Review A*, 91(1):012305, 2015.

[RBBMS21] Andrea Rodriguez-Blanco, Alejandro Bermudez, Markus Müller, and Farid Shahandeh. Efficient and robust certification of genuine multipartite entanglement in noisy quantum error correction circuits. *PRX Quantum*, 2(2):020304, 2021.

[Rei20] Ben W Reichardt. Fault-tolerant quantum error correction for Steane's seven-qubit color code with few or no extra qubits. *Quantum Science and Technology*, 6(1):015007, 2020.

[SCC19] Yunong Shi, Christopher Chamberland, and Andrew Cross. Fault-tolerant preparation of approximate GKP states. *New Journal of Physics*, 21(9):093007, 2019.

[Sho94] P.W. Shor. Algorithms for quantum computation: Discrete logarithms and factoring. *Proceedings., 35th Annual Symposium on Foundations of Computer Science*, pages 124–134, 1994.

[Sho95] Peter W Shor. Scheme for reducing decoherence in quantum computer memory. *Physical review A*, 52(4):R2493, 1995.

[Sho96] Peter W. Shor. Fault-tolerant quantum computation. *Proceedings., 37th Annual Symposium on Foundations of Computer Science*, pages 56–65, 1996.

[Ste96] Andrew Steane. Multiple-particle interference and quantum error correction. *Proceedings of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences*, 452(1954):2551–2577, 1996.

[Ste97] Andrew M Steane. Active stabilization, quantum computation, and quantum state synthesis. *Physical Review Letters*, 78(11):2252, 1997.

[Ste02] Andrew M Steane. Fast fault-tolerant filtering of quantum codewords. *arXiv preprint quant-ph/0202036*, 2002.

[Ste03] Andrew M Steane. Overhead and noise threshold of fault-tolerant quantum error correction. *Physical Review A*, 68(4):042322, 2003.

[TB05] Barbara M Terhal and Guido Burkard. Fault-tolerant quantum computation for local non-Markovian noise. *Physical Review A*, 71(1):012336, 2005.

[TCL20] Theerapat Tansuwannont, Christopher Chamberland, and Debbie Leung. Flag fault-tolerant error correction, measurement, and quantum computation for cyclic Calderbank-Shor-Steane codes. *Physical Review A*, 101(1):012342, 2020.

[TL21a] Theerapat Tansuwannont and Debbie Leung. Achieving fault tolerance on capped color codes with few ancillas. *arXiv preprint arXiv:2106.02649*, 2021.

[TL21b] Theerapat Tansuwannont and Debbie Leung. Fault-tolerant quantum error correction using error weight parities. *Physical Review A*, 104(4):042410, 2021. "Copyright (2021) by the American Physical Society".

[TYC17] Ryuji Takagi, Theodore J Yoder, and Isaac L Chuang. Error rates and resource overheads of encoded three-qubit gates. *Physical Review A*, 96(4):042302, 2017.

[VB19] Michael Vasmer and Dan E Browne. Three-dimensional surface codes: Transversal gates and fault-tolerant architectures. *Physical Review A*, 100(1):012312, 2019.

[VBK21] Michael Vasmer, Dan E Browne, and Aleksander Kubica. Cellular automaton decoders for topological quantum codes with noisy measurements and beyond. *Scientific reports*, 11(1):1–14, 2021.

[Vui18] Christophe Vuillot. Is error detection helpful on IBM 5q chips? *Quantum Information and Computation*, 18(11&12):0949–0964, 2018.

[YK17] Theodore J. Yoder and Isaac H. Kim. The surface code with a twist. *Quantum*, 1:2, April 2017.

APPENDIX

Appendix A

Simulations of possible faults during the FTEC protocol for the 49-qubit concatenated Steane code

A.1 Simulation of possible faults assuming that the last round of full syndrome measurement has no faults

As discussed in Section 5.2, in order to verify that the FTEC protocol for the $[\![49, 1, 9]\!]$ code satisfies the FTEC conditions in Definition 5.2, we consider two separate cases: the case that there are some faults during the last round of full syndrome measurement, and the case that there are not. In this section, we provide details of a simulation to show that whenever the number of faults is at most 3 and none of the faults occurs during the last round, all possible fault combinations satisfy Claim 5.2 and our protocol can correct errors on the data qubits.

In our protocol, we will perform full syndrome measurements until the outcome bundles are repeated 4 times in a row. Since there are at most 3 faults, the repetition condition will be satisfied within 16 rounds of full syndrome measurement. In this simulation, we assume that the last round of measurement has no faults, thus the high-weight error on the data

qubits arising from at most 3 faults is accumulated from up to 15 rounds. We will use the outcome bundle (syndromes and flag vector) obtained from the last round to determine the fault combination that cause the error so that the corresponding weight parity can be found and the WPEC can be done.

We first define mathematical objects being used in our simulation. Let *fault* be an object with two associated variables: *Pauli error* defined on the code block of 49 qubits arising from the fault, and *flag vector* $\in \mathbb{Z}_2^{21}$ which indicates the flag measurement results associated with the fault. There are 4 types of possible faults: faults during wait time (denoted by W), faults arising from the measurement of 1st-level and 2nd-level generators (denoted by G_1 and G_2, respectively), and flag measurement faults (denoted by F). A *fault combination* can be constructed by combining faults of same or different types up to 3 faults, i.e., multiplying their Pauli errors and adding their flag vectors. The errors on the input codeword can be considered as wait time faults in which associated Pauli errors do not propagate to other data qubits during the FTEC protocol. In addition, the X-type errors on the data qubits arising from the faults during the measurement of Z-type generators can be considered as wait time faults during the measurement of subsequent X-type generators, in which our simulation is also applicable. (Since the last round of measurement has no faults, we can assume that the syndromes obtained from the last round are correct and the syndrome measurement faults can be neglected.)

Next, we define *fault set* as follows: for faults of type G_1 (or type G_2), we denote $F_{i,j}^{G_1}$ (or $F_{i,j'}^{G_2}$) to be sets of possible G_1 (or G_2) faults arising from a circuit for measuring g_j^z, $j = 1, \ldots, 21$ (or $\tilde{g}_{j'}^z$, $j' = 1, 2, 3$) where the number of faults is $i \in \{0, 1, 2, 3\}$ (g_j^z refers to the generator $g_{(j-1) \bmod 3+1}^z$ on the $\lceil j/3 \rceil$-th subblock). Also, we denote F_i^W and F_i^F to be sets of possible faults of type W and F, respectively, where the number of faults is $i \in \{0, 1, 2, 3\}$. In addition, we define *fault set combination* to be a set of fault sets up to 3 sets.

Last, let $v_{G_1}, v_{G_2}, v_W, v_F$ be the number of faults of type G_1, G_2, W, and F, respectively. $(v_{G_1}, v_{G_2}, v_W, v_F)$ that satisfies $v_{G_1} + v_{G_2} + v_W + v_F \leq 3$ is called *fault number combination*.

With the definitions of fault, fault combination, fault set, fault set combination, and fault number combination, now we are ready to describe the simulation.

Pseudocode for a simulation of possible faults assuming that the last round of full syndrome measurement has no faults

1. Construct fault sets $F_{i,j}^{G_1}, F_{i,j'}^{G_2}, F_i^W$, and F_i^F for all $i = 0, 1, 2, 3$, $j = 1, \ldots, 21$, $j' = 1, 2, 3$.

2. Construct all possible fault number combinations that satisfy $v_{G_1} + v_{G_2} + v_W + v_F \leq 3$.

3. For each $(v_{G_1}, v_{G_2}, v_W, v_F)$, find all possible fault set combinations from $v_{G_1}, v_{G_2}, v_W, v_F$. Note that if v_{G_1} is 2, the fault set combination can have $F_{i,j}^{G_1}$ and $F_{i',j'}^{G_1}$ with $i = i' = 1$, or have $F_{i,j}^{G_1}$ with $i = 2$. Also, if v_{G_1} is 3, the fault set combination can have $F_{i,j}^{G_1}$, $F_{i',j'}^{G_1}$, and $F_{i'',j''}^{G_1}$ with $i = i' = i'' = 1$, or have $F_{i,j}^{G_1}$ and $F_{i',j'}^{G_1}$ with $i = 2, i' = 1$, or have $F_{i,j}^{G_1}$ with $i = 3$. The same goes for v_{G_2}.

   (a) For each fault set combination, find all possible fault combinations. Each fault combination can be found by picking one fault from each fault set (up to 3 sets) in the fault set combination, then combine the faults to get the combined Pauli error $\mathbf{E}$ and the cumulative flag vector $\vec{\mathbf{f}}_z$ associated with the fault combination.

   (b) For each fault combination, find 1st-level syndrome $\vec{\mathbf{s}}_{1x}$, 2nd-level syndrome $\vec{\mathbf{s}}_{2x}$, block triviality $\vec{\eta}_x$, and block parity $\vec{p}_x$ from the associated combined data error $\mathbf{E}$. Store $(\vec{\mathbf{s}}_{1x}, \vec{\mathbf{s}}_{2x}, \vec{\eta}_x, \vec{\mathbf{f}}_z, \vec{p}_x)$ for each fault combination in a lookup table.

4. After the lookup table is complete, categorize fault combinations by their 2nd-level syndromes and block trivialities in order to get $\mathcal{G}_j$'s as in Claim 5.2.

5. For each $\mathcal{G}_j$, verify whether Condition 1 or 2 in Claim 5.2 is satisfied.

From the simulation above, we find that all possible fault combinations satisfy Claim 5.2. That is, for each fault combination, we can determine the weight parity from the outcome bundles obtained from the last round of full syndrome measurement by looking at the table constructed in Step 3b. The weight parity can be later used to perform WPEC on the code block. With this simulation result, we can verify our FTEC protocol for the $[\![49, 1, 9]\!]$ code satisfies FTEC conditions as previously discussed in Section 5.2.

A.2 Simulation of possible faults assuming that the last round of full syndrome measurement has some faults

In Appendix A.1, we describe the simulation of possible faults during the FTEC protocol for the $[\![49, 1, 9]\!]$ code which is applicable to the case that there are no faults during the last round of full syndrome measurement. In this appendix, we will extend the ideas and construct a simulation of possible faults for the case that some faults occur during the last round.

As previously described, we will perform full syndrome measurements in the protocol until the outcome bundles are repeated 4 times in a row. Now, suppose that the last round of full syndrome measurement has some faults. In this case, we cannot be sure whether the outcome bundle from the last round exactly corresponds to the error in the data qubits. Fortunately, since there are at most 3 faults during the whole protocol, at least one outcome bundle obtained from the last 4 rounds must be correct. Note that the outcome bundles from the last 4 rounds are identical. From the simulation result discussed in Appendix A.1, the outcome bundle from the last round can be used to correct the data error occurred before *any* correct round using the WPEC technique (see Fig. A.1 for more details). The goal of the simulation in this section is to verify that all possible fault combinations which can happen after the last correct round give data error of weight no more than 3.

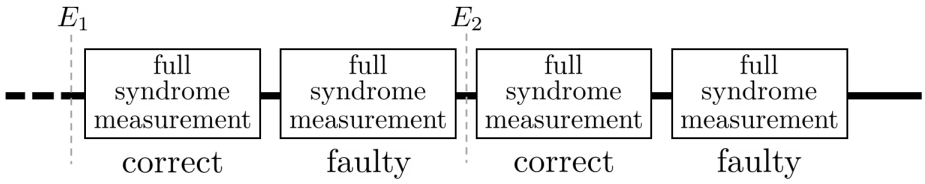


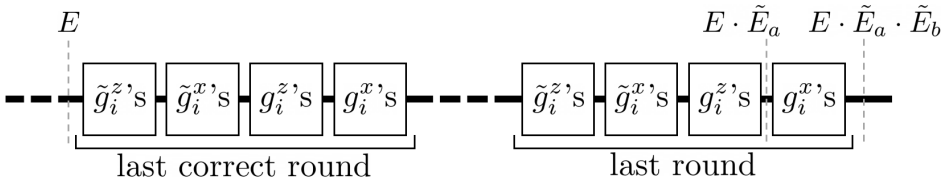
Figure A.1: At least one of the last 4 rounds of full syndrome measurement is correct since there are at most 3 faults. Because the outcome bundles from the last 4 rounds are identical, the outcome bundle from the last round can be used in WPEC to correct both errors E_1 and E_2 (even though E_1 and E_2 may not be equal).

A straightforward way to verify the claim above is to find all possible fault combinations and check the weight of their combined data errors. Unfortunately, this process requires many

computational resources. Thus, we will use "relaxed conditions" for the verification instead; for each fault combination, if the corresponding combined data error and cumulative flag vector satisfy all relaxed conditions, the fault combination will be marked (indicating that the fault combination might cause the protocol to fail). We want to make sure that for all fault combination that can cause the protocol to fail (i.e., its combined data error has weight more than 3), the fault combination will be marked. Note that some fault combinations may be marked by the relaxed conditions but will not cause the protocol to fail. For this reason, all of the marked fault combinations must be examined after the simulation is done.

We should note that the order of generator measurements is important for the fault tolerance of our FTEC protocol. Consider the protocol description in Section 5.2 in which we measure generator measurements in the following order during a single round of full syndrome measurement: measuring all $\tilde{g}_i^z$'s, then all $\tilde{g}_i^x$'s, then all g_i^z's, then all g_i^x's. Let us first consider the errors that can be caught by the g_i^x measurements of the last round. Observe that all Z-type data errors that arise before the g_i^x measurements of the last round will be evaluated by the 1st-level syndrome $\vec{\mathbf{s}}_{1x}$. However, some faults during g_i^x measurements of the last round may cause X-type or Z-type errors that will not be caught by any syndrome. Without loss of generality, we will construct a simulation using an assumption that faults before the g_i^x measurements of the last round can cause only Z-type errors, and faults during or after the g_i^x measurements can cause X-type or Z-type errors. The simulation is also applicable to the case of g_i^z measurements.

Let E, $\tilde{E}_a$, and $\tilde{E}_b$ be data errors arising from faults occurred before the last correct round among the last 4 rounds, faults occurred after the last correct round but before the g_i^x measurements of the last round, and faults occurred during or after the g_i^x measurements of the last round. The errors can be illustrated as follows:



The outcome bundle obtained from the last round is equal to the outcome bundle obtained from the correct round and can be used to correct E. Thus, we would like to mark

every fault combination that can occur after the correct round, corresponds to the trivial outcome bundle (since the outcome bundle obtained from the last round is the same as that obtained from the correct round), and its combined data error has weight more than 3. In particular, our relaxed conditions will examine 3 objects for each fault combination: the 1st-level syndrome, the cumulative flag vector, and the weight of the combined data error.

The mathematical objects being used in this simulation are similar to those defined in Appendix A.1. In addition, we will consider syndrome measurement faults (denoted by S) as another type of faults in this simulation since we will assume that the syndrome measurement during the last 4 rounds can be faulty. Also, let $v_{G_{1a}}$ be the number of G_1 faults that occur before the g_i^x measurements of the last round, and let $v_{G_{1b}}$ be the number of G_1 faults that occur during or after the g_i^x measurements of the last round. *Fault number combination* is a tuple $(v_{G_{1a}}, v_{G_{1b}}, v_{G_2}, v_W, v_F, v_S)$ that satisfies $v_{G_{1a}} + v_{G_{1b}} + v_{G_2} + v_W + v_F + v_S \leq 3$.

For the first relaxed condition, let us first assume that none of the faults of type W occurs before or during the g_i^x measurements of the last round. For each $(v_{G_{1a}}, v_{G_{1b}}, v_{G_2}, v_W, v_F, v_S)$, error $\tilde{E}_a$ will be constructed from possible fault combinations that correspond to $v_{G_{1a}}$ and v_{G_2}. We will mark every fault combination whose associated $\tilde{E}_a$ gives a 1st-level syndrome that has Hamming weight no more than v_S (where the Hamming weight is the number of 1's in a bitstring). This is because each fault of type S can alter at most 1 syndrome bit. Now let us consider the case that some faults of type W occurs before or during the g_i^x measurements. Each W fault (which corresponds to error of weight 1) can change at most 3 bits of $\vec{\mathbf{s}}_{1x}$, but the change will affect only the subblock in which the fault acts nontrivially. We will define function $\sigma(\tilde{E}_a, v_W)$ by the following calculation:

1. Find the 1st-level syndrome of $\tilde{E}_a$ and calculate the Hamming weight of the syndrome for each subblock.

2. Sort the Hamming weights from all subblocks. The function value is the the sum of the $7 - v_W$ smallest Hamming weights.

The value of $\sigma(\tilde{E}_a, v_W)$ is the minimum Hamming weight of the 1st-level syndrome when v_W faults of type W occur. Taking all fault types into account, our first relaxed condition becomes

$$\sigma(\tilde{E}_a, v_W) \leq v_S. \tag{A.1}$$

For the second relaxed condition, we will consider the cumulative flag vector associated with each fault combination. Note that a flag measurement result will be obtained during any g_i^x or g_i^z measurement. Let $\vec{\mathbf{f}} = (\vec{\mathbf{f}}_x|\vec{\mathbf{f}}_z)$ denote the cumulative flag vector associated with each fault combination, and let $h(\vec{\mathbf{f}})$ denote the Hamming weight of $\vec{\mathbf{f}}$. Since each fault of type F can alter at most 1 bit of $\vec{\mathbf{f}}$, our second relaxed condition becomes,

$$h(\vec{\mathbf{f}}) \leq v_F. \tag{A.2}$$

For the third relaxed condition, we will consider the weight of the combined data error associated with each fault combination. The weight is evaluated at the end of the protocol where the resulting error is caused by all faults of type G_1, G_2, and W (errors arising during or after the g_i^x measurements of the last round can be X-type or Z-type). If W faults do not occur before or during the g_i^x measurements at the last round, the weight of the resulting error is the weight of $\tilde{E}_a \cdot \tilde{E}_b$. If they do, each W fault can increase the total weight by at most 1. Hence, our third condition becomes,

$$\mathrm{wt}(\tilde{E}_a \cdot \tilde{E}_b) + v_W > 3. \tag{A.3}$$

Note that the weight of $\tilde{E}_a \cdot \tilde{E}_b$ can be reduced by multiplication of some stabilizer, and the fault combination will not be marked unless Eq. (A.3) is satisfied for all choice of stabilizer.

Using the relaxed conditions in Eqs. (A.1) to (A.3), our simulation to verify that all possible data errors arising after the correct round have weight no more than 3 can be constructed as follows:

Pseudocode for a simulation of possible faults assuming that the last round of full syndrome measurement has some faults

1. Construct fault sets $F_{i,j}^{G_1}, F_{i,j'}^{G_2}$ for all $i = 0, 1, 2, 3$, $j = 1, \ldots, 21$, $j' = 1, 2, 3$.

2. Construct all possible fault number combinations that satisfies $v_{G_{1a}} + v_{G_{1b}} + v_{G_2} + v_W + v_F + v_S \leq 3$.

3. For each $(v_{G_{1a}}, v_{G_{1b}}, v_{G_2}, v_W, v_F, v_S)$, construct all possible fault set combinations from only $v_{G_{1a}}$, $v_{G_{1b}}$, and v_{G_2}. During the construction of each fault set combination, label fault sets that come from $v_{G_{1a}}$ or v_{G_2} with letter a, and label fault sets that come from $v_{G_{1b}}$ with letter b. Note that if $v_{G_{1a}}$ is 2, the fault set combination can have

$F^{G_1}_{i,j}$ and $F^{G_1}_{i',j'}$ with $i = i' = 1$, or have $F^{G_1}_{i,j}$ with $i = 2$. Also, if $v_{G_{1a}}$ is 3, the fault set combination can have $F^{G_1}_{i,j}$, $F^{G_1}_{i',j'}$, and $F^{G_1}_{i'',j''}$ with $i = i' = i'' = 1$, or have $F^{G_1}_{i,j}$ and $F^{G_1}_{i',j'}$ with $i = 2, i' = 1$, or have $F^{G_1}_{i,j}$ with $i = 3$. The same goes for $v_{G_{1b}}$ and v_{G_2}.

(a) For each fault set combination, find all possible fault combinations. Each fault combination can be found by picking one fault from each fault set (up to 3 sets) in the fault set combination. $\tilde{E}_a$ associated with each fault combination can be found by combining only faults from fault sets with label a, while $\vec{\mathbf{f}}$ and $\tilde{E}_a \cdot \tilde{E}_b$ can be found by combining faults from all fault sets.

i. For each fault combination, if Eqs. (A.1) to (A.3) are all satisfied, the fault combination will be marked. Note that for Eq. (A.3), the weight of $\tilde{E}_a \cdot \tilde{E}_b$ must be minimized by stabilizer multiplication.

From the simulation above, we find that there are 6 fault combinations which are marked by the relaxed conditions in Eqs. (A.1) to (A.3). All of them correspond to the case that $v_{G_2} = 1$, $v_W = 2$, $v_{G_{1a}}, v_{G_{1b}}, v_F, v_S = 0$, and their combined data errors are trivial on 5 subblocks and have either $IIIIIIZ$ or $ZIIIIII$ on 2 subblocks. We find that $IIIIIIZ$ and $ZIIIIII$ correspond to 1st-level syndrome (001) and (100), respectively. Since $v_S = 0$, the associated 1st-level syndrome must be trivial whenever errors from W faults are taken into account. This can happen only when errors from W faults cancel with the aforementioned Pauli error, which means that the resulting error has weight 0. As a result, we find that all of the marked fault combinations cannot cause data error of weight higher than 3. Similar simulations can be done to show that whenever s faults occur where $s = 0, 1, 2$, the weight of the output error is at most s. This result verifies that the FTEC protocol for the $[\![49, 1, 9]\!]$ code satisfies FTEC conditions as discussed in Section 5.2.

Glossary

Abelian group A group in which all operators commute with one another.

anticommute We say that two operators A and B anticommute if $AB + BA = 0$.

bit The smallest unit of classical information. A bit can be in either state 0 or 1.

block parity A bitstring in which each bit is the weight parity of an error in each subblock of a concatenated code.

block triviality A bitstring in which each bit is the triviality of an error in each subblock of a concatenated code.

Clifford operation An operation which maps a Pauli operator to another Pauli operator.

code switching A procedure involving gauge operator measurements which can be used to switch between two stabilizer codes with similar structures.

combined data error A product of data errors of all faults in a fault combination.

commute We say that two operators A and B commute if $AB - BA = 0$.

concatenated code A quantum error correcting code obtained from code concatenation (in which the quantum data is encoded repeatedly).

CSS code A stabilizer code whose stabilizer generators can be chosen to be purely X type or purely Z type.

cumulative flag vector A sum of flag vectors of all faults in a fault combination.

cyclic code A code in which any cyclic shift of a codeword is also a codeword.

data CNOT gate A CNOT gate which couples one of the data qubits and the syndrome ancilla.

data error Pauli errors on the data qubits caused by a Pauli error due to the fault propagating through the circuit for measuring a stabilizer generator.

data qubits Physical qubits in a code block which are used to encode the quantum data.

distance The distance of a stabilizer code is the minimum weight of Pauli operators which are in the normalizer group but not in the stabilizer group.

distinguishable error set A set of combined data errors of all fault combinations with trivial cumulative flag vectors in a distinguishable fault set.

distinguishable fault set We say that a fault set is distinguishable if for any pair of fault combinations in the fault set, the syndromes of their combined data errors are different, or their cumulative flag vectors are different, or their combined data errors are logically equivalent. Otherwise, we say that the fault set is indistinguishable.

error decoding A procedure to find an error correction operator corresponding to the error syndrome obtained from syndrome measurements.

extended rectangle A group of gadgets including leading error correction gadget(s), a gate (or a measurement or a state preparation) gadget, and trailing error correction gadget(s).

fault A location and a 1- or 2-qubit Pauli operation describing a deviation of an actual quantum operation from its ideal operation.

fault combination A set of faults. There are combined data error and cumulative flag vector associated with each fault combination.

fault set A fault set $\mathcal{F}_t$ is a set of all fault combinations arising from up to t faults.

fault tolerance The property of a quantum operation or a gadget which does not produce errors beyond the error correction ability of a quantum error correcting code when the input error is small.

flag ancilla An ancilla qubit used to keep the flag measurement outcome in a circuit for measuring a stabilizer generator.

flag circuit A circuit for measuring a stabilizer generator which has at least one flag ancilla.

flag CNOT gate A CNOT gate which couples one of the flag ancillas and the syndrome ancilla.

flag vector A bitstring of the measurement outcomes of all flag ancillas. A flag vector is caused by a Pauli error due to the fault propagating through the flag circuit for measuring a stabilizer generator.

gauge operator An operator which commutes with all stabilizers of a subsystem code. In general, gauge operators may not commute with one another. Gauge operators can be viewed as logical operators corresponding to the logical qubits which are not used to encode any quantum data.

location A time step and the index (or indices) of a qubit (or pair of qubits) involved in a quantum operation.

logical operator An operator in the normalizer group. A logical operator maps a valid codeword of the stabilizer code to another valid codeword.

logically equivalent errors Two Pauli operators are logically equivalent if they are off by a multiplication of some stabilizer.

magic state distillation A procedure to produce high-fidelity quantum states (which can be used to perform a non-Clifford operation) from noisy quantum states.

non-flag circuit A circuit for measuring a stabilizer generator which has no flag ancillas.

normalizer group A group of Pauli operators which commute with all operators in the stabilizer group.

outcome bundle The collection of measurement outcomes obtained during each single round of operator measurements in a fault-tolerant protocol.

Pauli error due to the fault A 1- or 2-qubit Pauli operation describing a deviation of an actual quantum operation from its ideal operation.

Pauli group A group of tensor products of Pauli operators (I, X, Y or Z) with a ± 1 or $\pm i$ phase factor.

perfect code A quantum error correcing code which saturates the quantum Hamming bound; i.e., there is a one-to-one correspondence between correctable errors and all possible syndromes.

perfect CSS code A CSS code constructed from classical error correcting codes which saturate the classical Hamming bound; i.e., there is a one-to-one correspondence between correctable X-type (or Z-type) errors and all possible syndromes obtained from measuring Z-type (or X-type) stabilizer generators.

quantum error correcting code A subspace of the complex Euclidean space which can be used to protect quantum data against noise.

quantum error correction A procedure to recover the original quantum state after some errors occur.

qubit The smallest unit of quantum information. A qubit can be in state 0, 1, or a superposition of states 0 and 1.

stabilizer An operator in the stabilizer group.

stabilizer code A $+1$ simulteneous eigenspace of all operators in a stabilizer group. A quantum state in this subspace is called codeword.

stabilizer group An Abelian group of Pauli operators which does not contain $-I^{\otimes n}$. A stabilizer group can be generated by stabilizer generators (which are independent, commuting Pauli operators).

subcode A quantum error correcting code defined on a subset of physical qubits of a bigger code.

subsystem code A quantum error correcting code which can be described by stabilizers and gauge operators.

syndrome The syndrome of an error E is a bitstring where the i-th bit is 0 if E commutes with stabilizer generator g_i, or is 1 if E anticommutes with g_i.

syndrome ancilla An ancilla qubit used to keep the syndrome measurement outcome in a circuit for measuring a stabilizer generator.

threshold An error rate in which if the physical error rate is below this value, the logical error rate can be made arbitrarily small.

transversal gate A logical gate which can be implemented on a code block by applying physical gates transversally.

triviality The triviality of an error E is 0 if E has the trivial syndrome, or is 1 if E has a non-trivial syndrome.

weight The number of non-identity factors of a multi-qubit Pauli operator.

weight parity The weight parity of an error E is 0 (or even) if E has even weight, or is 1 (or odd) if E has odd weight.

weight parity error correction An error correction procedure which uses the fact that for some stabilizer codes, Pauli errors with the same syndrome and weight parity are logically equivalent.