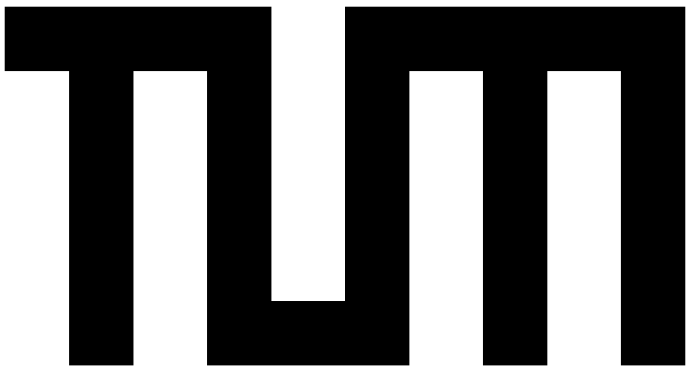
Localizable long-range encoded entanglement



Vincent Steffan

Master's thesis
Supervisor: Prof. Dr. Robert König
Advisor: Prof. Dr. Robert König
Submission Date: _____

Department of Mathematics
Technische Universität München

I assure the single handed composition of this Master's thesis only supported by the declared ressources.

Place, Date

Signature

Abstract

In this thesis we investigate possibilities to create long-range entanglement between encoded qubits. Our main focus is to restrict to constructions where all operations on physical qubits are local with respect to a 2 dimensional planar geometry.
We provide examples of such schemes for the repetition, Steane and surface codes and also a more general construction applicable for any CSS-Code. We also address the question of the reliability of this scheme when we assume local stochastic noise on the physical qubits.

Zusammenfassung

In dieser Arbeit beschäftigen wir uns mit der Generierung von enkodierten, verschränkten Quantenbits. Wir erkären eine Konstruktion hierfür, die nur Operationen verwendet, welche lokal bezüglich einer zweidimensional planaren Geometrie sind.
Neben einigen Beispielen für solche Konstruktionen für den 'repetition code', den 'Steane code' sowie den 'surface code' präsentieren wir ein allgemeines Schema, das auf jeden beliebigen CSS-Code angewendet werden kann. Wir untersuchen außerdem, wie verlässlich diese Konstruktion ist, wenn wir Fehler auf den Quantenbits zulassen. Als Fehlermodel wählen wir 'local stochastic noise' auf den physikalischen Quantenbits.

Contents

1 Introduction **6**

2 Quantum information theory **8**

2.1 Notation and conventions 9

2.2 Entanglement: Phenomena, applications and advantages 16

2.2.1 Entanglement 16

2.2.2 A Bell inequality and entanglement 16

2.2.3 Teleportation 21

3 Quantum codes and the stabilizer formalism **22**

3.1 Classical linear codes 23

3.1.1 Basic definitions and examples of classical codes 23

3.1.2 The Tanner graph of a code 28

3.2 Introduction to quantum codes 29

3.3 The stabilizer formalism 33

3.3.1 Stabilizer groups and stabilizer codes 33

3.3.2 A different representation of stabilizer codes 34

3.3.3 Evolution and the stabilizer formalism 35

3.3.4 Measurements and the stabilizer formalism 35

3.3.5 Stabilizers of CSS-Codes 36

3.4 The graph state 39

3.5 The surface code 43

3.6 Constructing CSS-Codes from graph states 47

4 Localizable long range encoded entanglement 54

4.1 The strategy 55

4.2 Examples of long range entanglement 56

4.2.1 The repetition code 56

4.2.2 The surface code 58

4.3 Long range entanglement for CSS-Codes 63

5 Error model and fault tolerance 69

5.1 Error models 70

5.1.1 Classical p-local stochastic noise 70

5.1.2 Noisy quantum circuits 71

5.2 Syndromes for measurements in graph states 73

1 Introduction

Quantum information theory promises a huge computational speed up. Already Feynman conjectured in 1982 that a computing device using the remarkable effects in quantum mechanics might improve our capability of solving computational problems [Fey82]. Since then, various computationally challenging problems like factoring have been proven to be solvable on a quantum device in polynomial time [Sho94]. In contrast to this appealing concept, it turns out that controlling real quantum systems is in practice intricate: Due to decoherence, quantum systems are very sensitive against noise introduced by the environment [Unr95, MPSE96, CLSZ95]. Fortunately, Shor demonstrated in 1995 that it is possible to encode the information carried by one 'qubit' into a larger system. With that, certain redundancies can be used to recover the original information even if errors have happened [Sho95].

Since then, the theory of encoding quantum information has developed a lot and promising quantum codes have been found, for example Kitaevs toric code [Kit03]. The great advantage of this code is that it can be implemented and corrected using operations which are local, that is, only act on a small number of qubits at once which are 'close to each other' with respect to a 3 dimensional geometry.

One remarkable difference between classical and quantum information theory is the usage of quantum non-locality: Using entangled states, that is, states which are not classically correlated, one can for example do some kind of 'precomputing' even before the actual problem is known. This is demonstrated in [RBB03], where the authors present a way of computing by first creating a large entangled state – more precisely, a graph state – followed by one qubit measurements. Only these measurements depend on the specific calculation, hence, we can interpret the implementation of the graph state as a precomputation.

Very recently, Bravyi et al. presented a scheme establishing quantum advantage

even assuming imperfect operations and noise on the physical qubits [BGKT19]. Here, the authors also used a graph state in order to create a Bell pair, that is, a maximally entangled state, between parties that are separated spatially by an arbitrarily large distance by acting with local measurements on the graph state. One of the key features is that it is possible to create such a Bell pair from a graph state which is already encoded in surface code, the planar analogue of Kitaev's toric code. In the setting of this publication, the Bell pair was created from a graph state on a 3D lattice, that is, the required operations for this scheme are local with respect to a 3D geometry. In order to make the whole construction faulttolerant, certain correlations between the measurement outcomes were established in order to control errors that happen during the preparation of the graph state and also faulty measurement devices. In this thesis we investigate to which extent we can create a Bell pair restricting ourselves to a 2D geometry using similar methods, that is, a planar graph state on which we may apply local measurements. A circuit local with respect to a 2D geometry is by far easier to realize than a 3-dimensional model.

Outline

In Chapter 2, we start by recalling some basic terms from classical information theory, quantum information theory as well as the theory about classical and quantum codes, which we introduce in Chapter 3. In particular, in Chapter 2, we give two examples for phenomena linked to entanglement – the violation of a Bell inequality and quantum teleportation. After that, we present graph states, the surface code and the so-called CSS-Codes and demonstrate the power of graph states by presenting a scheme for implementing an CSS-Code encoded qubit using a graph state. In Chapter 4, we present a scheme to create long range localizable encoded entanglement using graph states with a planar 'bulk' and investigate in Chapter 5 to which extent we can protect ourselves against noise in this particular scheme.

2

Quantum information theory

This chapter gives a quite short overview of the concepts in classical and quantum information theory we will use later on. The raison d'être of the chapter's first half is to demonstrate the notations and conventions we will use. In the second half, we highlight some phenomena linked to *entanglement*.
As reference for a thorough introduction to this topic we recommend [NC11].

2.1 Notation and conventions

Information theory is about storing, sending and processing information. Classically, a piece of *information* is a string of so-called bits, where every bit is either 0 or 1. In order to get an intuition for the challenges of information theory, let us introduce three important characters that will appear every once in a while in this thesis: Alice, Bob and Charlie. While they are trying to store, process and send information, we can discover some of the challenges an information theorist has to face. Let us see some examples:

(i) Suppose Alice wants to send to Bob the message **0011** consisting of four bits. For this, she can use a channel – some kind of device which transfers a bit to Bob.
Here, a typical challenge appears if the channel is not perfect. Say, the channel flips a bit with probability $\frac{1}{3}$. Then, the probability that Bob receives the correct message is $\left(\frac{2}{3}\right)^4$.

(ii) Now, Bob receives some four bit string. He does not want to forget this message, so he wants to save it on a hard drive. Again, we cannot expect that this will work perfectly and need to protect this against possible errors.

(iii) Charlie wants to identify with a bit string $(a_i)_{i=0}^n$ a natural number

$$a_0 \cdot 2^0 + a_1 \cdot 2^1 + a_2 \cdot 2^2 + \cdots + a_n \cdot 2^n.$$

Now, he wants to build a calculator which, whenever he inputs two bitstrings, outputs the sum of the two as a bitstring. For this, he can only use one bit *gates* like NOT, which flips a bit and two bit gates like CNOT, which flips the second bit if and only if the first bit is 1.
A natural question would be if we can do this efficiently or rather, how many gates we will need at least. Also in this setting, we should keep in mind that some gates may be faulty and prepare for errors to happen.

We can generalize example (iii): A collection of gates implementing some function

$$f\colon \{0,1\}^n \to \{0,1\}^m$$

by applying logical gates in some order is called *circuit*. More precisely, a classical circuit is a directed acyclic graph where the vertices of this graph are inputs, outputs or gates. We use the term *classical circuit* to distinguish from the quantum circuits which we will encounter later. Each gate applies a function

$$g\colon \{0,1\}^j \to \{0,1\}$$

where j is the in-degree of the corresponding vertex. This circuit implements the function f if – whenever we input an n-bit string x – it outputs the m-bit string $f(x)$.

Let us now turn to *quantum information theory*. Here, the building blocks of information are not bits but so-called *qubits*, which are nothing else but spin-$\frac{1}{2}$ particles. These particles have a state vector

$$|\psi\rangle \in \mathbb{C}^2.$$

The state vector is the information a qubit carries. The state space is spanned by two orthonormal basis vectors $|0\rangle$ and $|1\rangle$, to which we will refer to as the *computational basis states*.
If we not only consider an isolated particle but a system of N particles, we know that the state of the the system lives in the state space

$$|\psi\rangle \in \left(\mathbb{C}^2\right)^{\otimes N}.$$

Before we quickly highlight why the situation becomes so much richer than in the classical setting, let us specify the allowed operations:

(i) We allow ourselves to prepare a qubit in state $|0\rangle$.

(ii) Moreover, we allow ourselves to act on the qubits with gates from some finite universal gate set.

(iii) We may also measure qubits in the computational basis. Measuring a state $|\psi\rangle$ in computational basis will give the measurement outcome ± 1 with probability

$$\mathrm{P}(\pm 1) = \left\langle \psi \middle| \frac{\mathbb{1} \pm Z}{2} \middle| \psi \right\rangle.$$

When the outcome was ± 1, the state will be collapsed to

$$\frac{\frac{\mathbb{1}\pm Z}{2}|\psi\rangle}{\left\langle \psi \middle| \frac{\mathbb{1}\pm Z}{2} \middle| \psi \right\rangle}.$$

Here, Z is the Pauli Z-operator defined next.

Remark. The computational basis states may be characterized as the ± 1 eigenstates of the map

$$\sigma_Z : |0\rangle \mapsto |0\rangle, \qquad |1\rangle \mapsto -|1\rangle$$

We also define a unitary

$$\sigma_X : |0\rangle \mapsto |1\rangle, \qquad |1\rangle \mapsto |0\rangle$$

and the composition of these two $i \cdot \sigma_Z \circ \sigma_X = \sigma_Y$. These three operators are called *Pauli operators*, their matrix representation with respect to

$$\begin{pmatrix} 1 \\ 0 \end{pmatrix} = |0\rangle, \qquad \begin{pmatrix} 0 \\ 1 \end{pmatrix} = |1\rangle$$

Pauli matrices. Moreover, the ± 1 eigenstates of σ_X will be denoted $|+\rangle$ and $|-\rangle$. The Pauli matrices are given by

$$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \qquad Y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}, \qquad Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}.$$

For a realistic scenario we restrict to only one kind of measurements, namely measurements in the computational basis. Theoretically, we can use as an observable any Hermitian operator instead of σ_Z. The possible outcomes and after measurement states of such a measurement are specified by Born's rule:

Born's rule. Let $O = O^\dagger$ be a Hermitian operator on $(\mathbb{C}^2)^{\otimes n}$ with eigenvalue decomposition

$$O = \sum_k o_k E_k$$

where $E_k = E_k^2 = E_k^\dagger$ are the projectors onto the eigenspaces of O and let $|\psi\rangle$ be a state in $(\mathbb{C}^2)^{\otimes n}$.
Measuring O in $|\psi\rangle$ yields the outcome o_k with probability

$$\mathrm{P}(o_k) = \langle\psi \,|\, E_k \,|\, \psi\rangle\,.$$

When the outcome was o_k, the system will be afterwards in state

$$|\psi_k\rangle = \frac{E_k\,|\psi\rangle}{\langle\psi \,|\, E_k \,|\, \psi\rangle}.$$

Example 2.1.1. A famous – and often used – example of a finite universal gate set is the *Clifford+T* gate set. It consists of the Hadamard, the phase and the $\frac{\pi}{8}$-gate

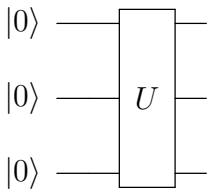
$$H = \frac{1}{\sqrt{2}}\begin{pmatrix}1 & 1\\ 1 & -1\end{pmatrix}, \qquad S = \begin{pmatrix}1 & 0\\ 0 & -i\end{pmatrix}, \qquad T = \begin{pmatrix}1 & 0\\ 0 & e^{i\frac{\pi}{4}}\end{pmatrix}$$

together with the controlled-Z acting on two qubits

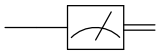
$$CZ = |0\rangle\langle 0| \otimes \mathbb{1} + |1\rangle\langle 1| \otimes Z.$$

Often, it is convenient to visualize circuits. For example, a circuit implementing the unitary $X \circ H$ can be visualized as

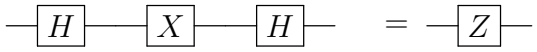
Note that we have to apply the H gate first. The qubit is represented by the 'wire'. Whenever we want to appy a gate to the qubit, we place it on the wire.
We can also visualize implementations on larger systems with this. Say, we want to apply the unitary U which acts on $(\mathbb{C}^2)^{\otimes 3}$ to a system of 3 qubits which are initially in state $|0\rangle^{\otimes 3}$. Visualized, the circuit looks like this:



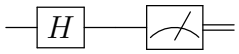
A measurement in computational basis will be visualized in the following way:



Here, the outcome wire stands for classical information, namely the measurement outcome.
Two different circuits may implement the same unitary:



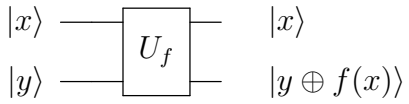
Using this, we can see that we can measure in the eigenbasis of the σ_X operator using the following implementation:



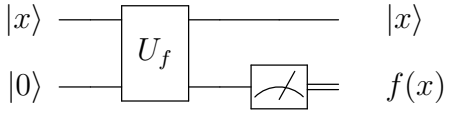
We refer to this as 'measuring in X-basis'.

Remark. What makes quantum information theory more rich than classical information theory?

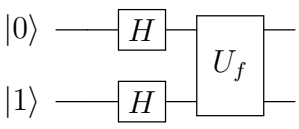
(i) One aspect is that we may not only work with the analogs of classical bits, $|0\rangle$ and $|1\rangle$, but also with so-called superpositions.
As an example we consider a function $f\colon \{0,1\} \to \{0,1\}$. We assume that we have a quantum circuit doing the following:



This can be interpreted as a reversible implementation of f. Indeed, we can calculate f with it by measuring the second qubit in computational basis:



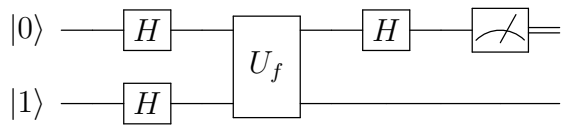
As the second system is in state $|f(x)\rangle$, the measurement outcome will be $f(x)$ with certainty.
Something interesting happens if we input superpositions. Consider the following circuit:



An elementary calculation shows that this circuit acts as

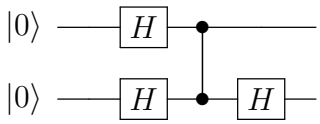
$$\begin{aligned}|0\rangle|1\rangle &\xmapsto{H\otimes H} \left(\frac{|0\rangle+|1\rangle}{\sqrt{2}}\right)\left(\frac{|0\rangle-|1\rangle}{\sqrt{2}}\right)\\ &\xmapsto{U_f} \frac{1}{\sqrt{2}}\left((-1)^{f(0)}|0\rangle+(-1)^{f(1)}|1\rangle\right)\left(\frac{|0\rangle-|1\rangle}{\sqrt{2}}\right)\end{aligned}$$

We notice that the second qubit remains unchanged. The first qubit is a multiple of $\frac{|0\rangle+|1\rangle}{\sqrt{2}}$ if $f(0)=f(1)$ and a multiple of $\frac{|0\rangle-|1\rangle}{\sqrt{2}}$ if $f(0)\neq f(1)$. Thus, the following circuit outputs $+1$ if and only if $f(0)=f(1)$:



This is remarkable: With only one application of U_f, we can find out with certainty if $f(0)$ is equal to $f(1)$. Classically, we would always need two applications of an implementation of f to figure this out. The described algorithm is known as the Deutsch algorithm [DP97]. Similar problems which can be solved more efficiently using superpositions are for example the Deutsch-Josza [DJ97] or the Bernstein-Vazirani problem [BV97].

(ii) Superposition is not the only aspect of quantum information theory that goes beyond classical information theory: Suppose Alice and Bob have one qubit each prepared in state $|0\rangle$. They now apply the following circuit to their qubits:



(2.1)

where the two qubit gate denotes the controlled Z gate. An easy calculation shows that the system of two qubits is now in state

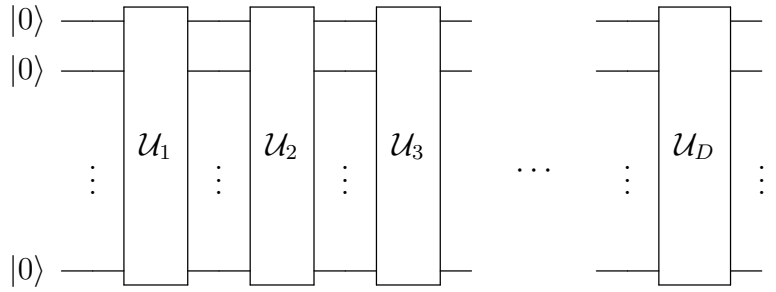
$$\left|\Phi^+\right\rangle = \frac{1}{\sqrt{2}}\left(|00\rangle+|11\rangle\right).$$

Now they go away from each other, but each of them keeps his qubit. If now Bob decides to measure his qubit in the computational basis, his qubit gets collapsed to either $|0\rangle$ or $|1\rangle$, which collapses the two qubit system to either $|00\rangle$ or $|11\rangle$. This therefore affects Alice's qubit immediately, even if they are far away from each other.
The next section gives a more precise picture of the consequences of this phenomenon called *entanglement*.

The last concept we introduce in this chapter is an important property of a specific implementation of a classical or quantum circuit – its depth.
We start with classical circuits. The depth of a classical circuit is the maximal length of a path connecting an input vertex with an output vertex. As we have to

apply the gates from each path of the graph step by step, this can be interpreted as the number of timesteps we need to run through the circuit once.
For quantum circuits, the situation is quite similar.

Definition 2.1.1. Consider the circuit



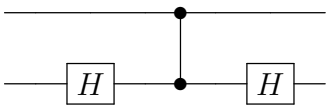
implementing a target unitary $\mathcal{U} = \mathcal{U}_D \cdot \mathcal{U}_{D-1} \cdot \cdots \cdot \mathcal{U}_1$. Moreover, assume that each $\mathcal{U}_i$ is a product of one and two qubit operations from the Clifford+T gate set acting on disjoint qubits.
Then, we say that this circuit has depth D.

Again, the depth of a circuit can be interpreted as the time it takes to run through it once: As $\mathcal{U}_i$ is a product of gates acting on disjoint sets of qubits, we can apply $\mathcal{U}_i$ 'in one time step'. As these factors are taken from the universal gate set, we assume that we can apply them immediately, that is, we do not need to split them up into simpler unitaries.
Therefore, the depth of the implementation of $\mathcal{U}$ is the number of time steps needed for this implementation.

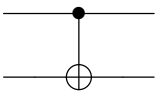
Example 2.1.2. (i) The circuit in Equation (2.1) has depth 3:

$$\mathcal{U}_1 = H^{\otimes 2}, \qquad \mathcal{U}_2 = CZ, \qquad \mathcal{U}_3 = \mathbb{1} \otimes H$$

(ii) The following circuit implements a controlled X. It has depth 3



We will denote the controlled X gate by



and use it in some of our constructions. Even though it is not an element of our chosen universal gate set, we will not run into trouble as it increases the depth of a circuit by a factor of not more than 3 if we allow circuits with such gates.

Remark. The restriction to one and two qubit gates for our universal gate set does not yet yield a realistic scenario. When we think of a 'quantum device' which consits of n qubits, it is not reasonable to allow controlled Z gate between any pair of qubits.

In a more realistic setting, the device could introduce the restriction to be *local* with respect to some *geometry*. As an example, assume that the qubits in the device are sitting on a line. We can act with any one qubit unitary from the Clifford + T gate set on each of the qubits. Moreover, we can act with controlled Z gates on every pair of neighbouring qubits.

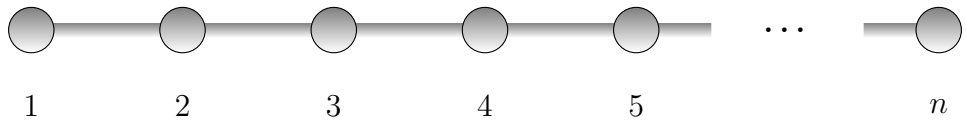


Figure 2.1: A quantum device consisting of n qubits: Controlled Z gates may only be applied betwenn qubits i and $i+1$.

This seems to be a severe restriction – at first sight this should increase the depth of many implementations to a great extent. We will see in the next section how *entanglement* helps us to circumvent this problem in some situations.

2.2 Entanglement: Phenomena, applications and advantages

The main focus of this thesis is to create entanglement between spatially separated qubits by local operations, that is, operations on single or constantly many qubits. To get motivated for this, we start by giving some examples of the power and beauty of entanglement. First, we will demonstrate how we can violate so-called *Bell inequalities* using entanglement. As a second example we will see a protocol to *teleport* qubits by local operations and the exchange of classical information. A good resource to learn much more on entanglement is for example [HHHH07].

2.2.1 Entanglement

We briefly introduce entanglement formally:

Definition 2.2.1. Consider a pure state $|\psi\rangle$ in a bipartite Hilbert space $\mathcal{H}_A \otimes \mathcal{H}_B$. If we find $|\psi_A\rangle$ in $\mathcal{H}_A$ and $|\psi_B\rangle$ in $\mathcal{H}_B$ with

$$|\psi\rangle = |\psi_A\rangle \otimes |\psi_B\rangle ,$$

then we call $|\psi\rangle$ a *product state*. We call all states that are not product states *entangled*.

The usual example is the Bell state:

Example 2.2.1. For $\mathcal{H}_A = \mathcal{H}_B = \mathbb{C}^2$, the state $\frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$ is entangled.

2.2.2 A Bell inequality and entanglement

The first phenomenon we present is the violation of a *Bell inequality* using an entangled pair of qubits. We start by defining the term *Bell inequality*. We do this similar as in [PV16]. More on quantum entanglement and Bell inequalities can be found for example in [WW01].

Consider two finite sets I and A. A conditional distribution from I to A is a tuple of probabilities

$$\mathrm{P}_{A|I} = (\mathrm{P}(a|i))_{a\in A, i\in I} \in \mathbb{R}_+^{|A|\times|I|}$$

satisfying

$$\sum_{a\in A} \mathrm{P}(a|i) = 1 \text{ for all } i \in I.$$

We denote the set of all conditional distributions from I to A by $\mathcal{P}(A|I)$.

For finite sets A, B, I, J we can of course also define the set $\mathcal{P}(A \times B|I \times J)$. Consider the following scenario: Two parties, Alice and Bob, live far from each

other and may not communicate with each other. Some third party – Charlie – hands Alice a box for each $i \in I$ and similarly Bob a box for each $j \in J$. The boxes are prepared in a way such that if Alice opens box i it will tell her in some way some outcome $a \in A$. Similarly, Bob will see some outcome $b \in B$ whenever he opens a box with some index $j \in J$. When Alice and Bob open the boxes i and j respectively, they will see the outcomes a and b with probability $\mathrm{P}(a, b|i, j)$ for a certain distribution in $\mathcal{P}(A \times B|I \times J)$. We call this the *classical scenario*. As an example, Charlie may give Alice and Bob three boxes each. The boxes contain one coin each that shows either heads (H) or tails (T). In other words, $I = J = \{1, 2, 3\}$ and $A = B = \{H, T\}$.

Only certain conditional distributions can arise in the classical scenario:
There, we assume that for each box given to Alice or Bob the probability of an outcome $a \in A$ or $b \in B$ is already determined. More formally, there is a 'hidden' variable λ in some space Λ determining probability functions

$$\lambda \mapsto \mathrm{p}_i(a, \lambda), \qquad \lambda \mapsto \mathrm{p}_j(b, \lambda)$$

for each of the boxes $i \in I$ and $j \in J$ and each $a \in A$ and $b \in B$. In the following, we assume Λ to be finite. We denote the probability measure on Λ according to which the λ's are distributed μ.
In the example with the three boxes containing coins, the hidden variable fixes for each box the probability that the coin shows heads or tails respectively.

Our second assumption was what is ofen called 'no-signaling condition': There is no exchange of information between one party – Alice and her boxes – and the other party – Bob and his boxes. This can be expressed by the condition that Alice' outcome probabilities are independent of which box Bob chooses and vice versa, that is,

$$\sum_{b \in B} \mathrm{P}((a, b)|(i, j)) \text{ is independent on } j,$$
$$\sum_{a \in A} \mathrm{P}((a, b)|(i, j)) \text{ is independent on } i.$$

Together, these two 'classicality conditions' imply that the conditional probabilities must be of the form

$$\mathrm{P}((a, b)|(i, j)) = \int_\Lambda \mathrm{p}_i(a, \lambda)\mathrm{p}_j(b, \lambda) d\mu(\lambda) = \sum_{\lambda \in \Lambda} \mu(\lambda)\mathrm{p}_i(a, \lambda)\mathrm{p}_j(b, \lambda).$$

As $\sum_{\lambda \in \Lambda} \mu(\lambda) = 1$, we see that the set of possible distributions is the convex hull of all product distributions. We call them the *classical conditional distributions*:

$$\mathcal{P}_{\mathrm{cl}}(A \times B|I \times J) = \mathrm{conv}(\{\mathrm{P}_{A|I} \times \mathrm{P}_{B|J} : \mathrm{P}(A|I) \in \mathcal{P}(A|I), \mathrm{P}(B|J) \in \mathcal{P}(B|J)\})$$

For finite sets A, B, I, J we call a linear map

$$\mathcal{M} \colon \mathbb{R}^{A \times B \times I \times J} \to \mathbb{R}$$

a *Bell functional*. It can be specified by coefficients

$$(M_{i,j,a,b})_{i\in I,j\in J,a\in A,b\in B} \in \mathbb{R}^{|A|\times|B|\times|I|\times|J|}$$

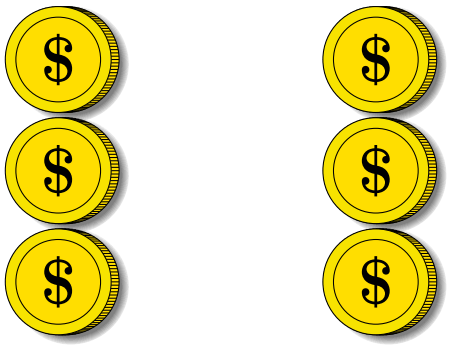
and acts on an element of $\mathcal{P}(A\times B|I\times J)$ as

$$(\mathrm{P}((a,b)|(i,j)))_{(a,b)\in A\times B,(i,j)\in I\times J} \mapsto \sum_{i,j,a,b} M_{i,j,a,b}\mathrm{P}((a,b)|(i,j)).$$

For a Bell functional $\mathcal{M}$ and a subset $\mathcal{U}$ of $\mathcal{P}_{\text{classical}}(A\times B|I\times J)$ a *Bell inequality* is an inequality bounding $\mathcal{M}(\mathrm{P})$ for all $P\in\mathcal{U}$.

Using an entangled pair of qubits we can violate certain Bell inequalities. Let us see an example.

Assume that Alice and Bob get three coins each from Charlie.



Alice Bob

Figure 2.2: Our setting for the Bell inequality: Alice and Bob get three coins each.

Each single coin is in a box and Alice and Bob can only see if it shows heads or tails if they open the corresponding box. Now, Alice and Bob may open one box each. We denote the index of the box by $i\in I=\{1,2,3\}$ and $j\in J=\{1,2,3\}$ respectively.
We denote their 'outcome' – heads or tails – by a and b. That is, we take $A=B=\{H,T\}$.
Depending on their choice, they will see the results a and b with probability

$$\mathrm{P}((a,b)|(i,j)). \tag{2.2}$$

We only assume that their choices, in other words, the distributions of i and j, are independent. Hence, the conditional distribution in Equation (2.2) is a classical distribution.
The 'coins in the boxes' are of course only a visualization. The whole argument holds for any two party scenario where both parties have three different measurement devices – in our case, boxes – which have two different possible outcomes – in our case, heads or tails.

For our Bell inequality, we restrict to the set $\mathcal{U}$ of conditional distributions P with

$$\mathrm{P}(a=b|i=j)=1,$$

that is, whenever Alice and Bob open the box with the same index, they will either see both heads or both tails.

As two of the three coins need to show the same, we see that

$$\mathrm{P}(a = b|i = 1, j = 2) + \mathrm{P}(a = b|i = 2, j = 3) + \mathrm{P}(a = b|i = 3, j = 1) \geq 1 \qquad (2.3)$$

This is a Bell inequality.

We can make a quantum version of this scenario similar as in [Pre01]: Alice and Bob share some bipartite state $|\psi_{AB}\rangle \in \mathcal{H}_A \otimes \mathcal{H}_B$. They can each 'open a box' by measuring their part along some axis $\overrightarrow{a}^i$ (resp. $\overrightarrow{b}^j$) with i and j between 1 and 3, that is, measuring the observable

$$\overrightarrow{a}^i \cdot \overrightarrow{\sigma} = a_1^i\sigma_1 + a_2^i\sigma_2 + a_3^i\sigma_3,$$

where we write $\sigma_1 = \sigma_X, \sigma_2 = \sigma_Y$ and $\sigma_3 = \sigma_Z$.
In our setting Alice and Bob shall share a so-called *Bell state*, namely

$$|\psi_{AB}^-\rangle = \frac{1}{\sqrt{2}}\left(|01\rangle - |10\rangle\right).$$

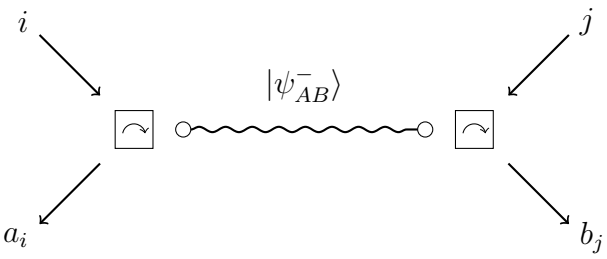


Figure 2.3: The quantum setting of our game: Alice and Bob share the maximally entangled state $|\psi_{AB}^-\rangle$. Depending on the input i, j they perform measurements on their part of the state and receive outcomes a_i and b_j.

It will turn out that – if we choose the axes wisely – we are in the above scenario but violate the Bell inequality. In order to achieve this, the following equation will ease the subsequent calculations:

$$\left(\sigma_i^A + \sigma_i^B\right)\left|\psi_{AB}^-\right\rangle = 0 \qquad (2.4)$$

for all $i = 1, 2, 3$, where we mean $\sigma_i \otimes \mathbb{1}$ when writing σ_i^A, in other words, we omit the identities. It is straightforward to verify Equation (2.4) for the three Pauli matrices. Using this we get

$$\begin{aligned}
&\left\langle\psi_{AB}^-\right|\left(\overrightarrow{\sigma}^A \cdot \overrightarrow{n}\right)\cdot\left(\overrightarrow{\sigma}^B \cdot \overrightarrow{m}\right)\left|\psi_{AB}^-\right\rangle = \\
&-\left\langle\psi_{AB}^-\right|\left(\overrightarrow{\sigma}^A \cdot \overrightarrow{n}\right)\cdot\left(\overrightarrow{\sigma}^A \cdot \overrightarrow{m}\right)\left|\psi_{AB}^-\right\rangle = \\
&-\sum_{k,l=1,2,3} n_k m_l \left\langle\psi_{AB}^-\right|\sigma_k^A\sigma_l^A\left|\psi_{AB}^-\right\rangle \overset{*}{=} \\
&-\sum_{k=1,2,3} n_k m_k = -\overrightarrow{n}\cdot\overrightarrow{m} = -\cos(\theta),
\end{aligned}$$

where θ is the angle between $\overrightarrow{n}$ and $\overrightarrow{m}$. Also note that to obtain $*$ we used

$$\left\langle \psi_{AB}^{-} \right| \sigma_k^A \sigma_l^A \left| \psi_{AB}^{-} \right\rangle = \delta_{kl},$$

which can be verified by a straightforward calculation.
Measuring

$$\left(\overrightarrow{\sigma}^A \cdot \overrightarrow{n}\right) \cdot \left(\overrightarrow{\sigma}^B \cdot \overrightarrow{m}\right),$$

with outcomes ± 1 leads to projections

$$E_{\pm 1}\left(\overrightarrow{n}\right) = \frac{1}{2}\left(\mathbb{1} \pm \overrightarrow{\sigma} \cdot \overrightarrow{n}\right), \; E_{\pm 1}\left(\overrightarrow{m}\right) = \frac{1}{2}\left(\mathbb{1} \pm \overrightarrow{\sigma} \cdot \overrightarrow{m}\right).$$

The probability of the outcomes is given by

$$\begin{aligned} P(\pm 1, \pm 1) &= \left\langle \psi_{AB}^{-} \right| E_{\pm 1}\left(\overrightarrow{n}\right) \otimes E_{\pm 1}\left(\overrightarrow{m}\right) \left| \psi_{AB}^{-} \right\rangle \\ &= \frac{1}{4} \left\langle \psi_{AB}^{-} \right| \left(\mathbb{1} + \overrightarrow{n} \cdot \overrightarrow{\sigma}^A + \overrightarrow{m} \cdot \overrightarrow{\sigma}^B + \overrightarrow{n} \cdot \overrightarrow{\sigma}^A \overrightarrow{m} \cdot \overrightarrow{\sigma}^B\right) \left| \psi_{AB}^{-} \right\rangle \\ &= \frac{1}{4}\left(1 - \cos(\theta)\right). \end{aligned}$$

An analogous calculation yields the probability for different outcomes:

$$P(\pm 1, \mp 1) = \frac{1}{4}\left(1 + \cos(\theta)\right)$$

Therefore, the probability of equal and different outcome when measuring system A along n and B along m is given by

$$\begin{aligned} P(\text{equal outcome}) &= \frac{1}{2}(1 - \cos\theta), \\ P(\text{different outcome}) &= \frac{1}{2}(1 + \cos\theta). \end{aligned}$$

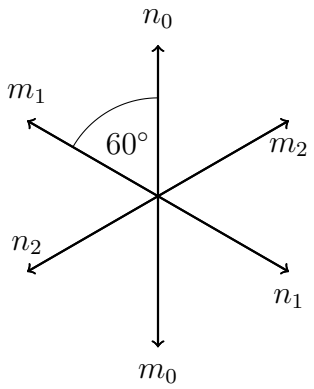


Figure 2.4: Choosing these as axes to measure along provides a violation of the Bell inequality.

Choosing axes $\overrightarrow{m_1}, \overrightarrow{m_2}, \overrightarrow{m_3}$ and $\overrightarrow{n_1}, \overrightarrow{n_2}, \overrightarrow{n_3}$ such that $\sphericalangle(\overrightarrow{n}_i, \overrightarrow{m}_i) = 180°$ and $\sphericalangle(\overrightarrow{n}_i, \overrightarrow{m}_j) = 60°$ for $i \neq j$ yields

$$\begin{aligned} P(\text{equal}) &= 1 \text{ for } i = j \\ P(\text{equal}) &= \frac{1}{4} \text{ for } i \neq j. \end{aligned}$$

Therefore, the setting of our scenario is satisfied: Whenever Alice and Bob 'open the same box', that is, measure along axes with coinciding label, they will have the same outcome ± 1. Moreover, we have

$$\mathrm{P}(a=b|i=1,j=2)+\mathrm{P}(a=b|i=2,j=3)+\mathrm{P}(a=b|i=3,j=1)=\frac{3}{4}<1,$$

therefore, the Bell inequality (2.3) is violated. That is, the conditional distribution arrising from this is non-classical.

2.2.3 Teleportation

Another useful application of entanglement is *teleportation*. Consider the following setting: Alice and Bob share one entangled state, say,

$$\left|\phi^+_{AB}\right\rangle = \frac{1}{\sqrt{2}}\left(|00\rangle + |11\rangle\right).$$

Moreover, Alice possesses an unknown state $|\chi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle$ which she wants to send to Bob. We will now demonstrate how Alice can *teleport* this state to Bob using the shared entanglement and the transmission of classical information. Note that this is impressive: On the one hand it is impossible for Alice to detect the state $|\chi\rangle$ as measurement destroys all the information in the state. On the other hand, it would be impossible for Alice to send a precise description of $|\chi\rangle$ to Bob using (a finite amount of) classical information.

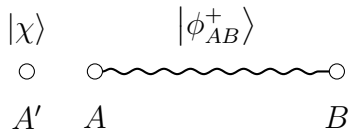


Figure 2.5: The setting of the Teleportation protocol.

In the first step of the protocol, Alice measures her system in Bell basis given by

$$\left|\phi^+\right\rangle = \frac{1}{\sqrt{2}}\left(|00\rangle + |11\rangle\right) \qquad \left|\phi^-\right\rangle = \frac{1}{\sqrt{2}}\left(|00\rangle - |11\rangle\right)$$
$$\left|\psi^+\right\rangle = \frac{1}{\sqrt{2}}\left(|01\rangle + |10\rangle\right) \qquad \left|\psi^-\right\rangle = \frac{1}{\sqrt{2}}\left(|01\rangle - |10\rangle\right)$$

Note, that each of the four Bell states can be written in a unique way as

$$|\varphi_{ab}\rangle = \left(Z^a X^b \otimes \mathbb{1}\right)\left|\phi^+\right\rangle = \left(\mathbb{1}\otimes X^b Z^a\right)\left|\phi^+\right\rangle$$

with $a, b \in \{0,1\}$. Now, a straight forward calculation shows that if Alice's measurement yields outcome a, b, her system gets collapsed to $|\varphi_{ab}\rangle$ and Bob's system is now in state $X^b Z^a\,|\chi\rangle$. In the next step of the protocol, Alice sends a and b to Bob. With this information and he can easily recover the state $|\chi\rangle$ by applying

$$\left(X^b Z^a\right)^\dagger.$$

3

Quantum codes and the stabilizer formalism

Let us now turn to the theory of codes. We first recapitulate some facts about classical linear codes. On the one hand, classical codes provide a pleasant starting point for this chapter, on the other hand they will become useful later on: By some modifications we will be able to construct quantum codes – so-called CSS-Codes – from classical linear codes. The main chapter of this thesis will demonstrate how we can obtain Bell pairs encoded in CSS codes only using local operations.
A more thorough introduction to classical coding theory may be found in one of the many textbooks on this topic, for Example [vL99]. More details about quantum codes may be found in [NC11].

3.1 Classical linear codes

3.1.1 Basic definitions and examples of classical codes

The theory of encoding information is motivated by a scenario we have seen earlier on: Alice wants to send information to Bob over some kind of channel. Again, a piece of information is just a string of bits. Let us consider the scenario where Alice wants to send only one bit. It is convenient to interpret a bit as an element of the field with two elements $\mathbb{F}_2$. Of course, we cannot assume that this channel is perfect, so, with some probability p, Bob receives $a \oplus 1$ when Alice sent a. Here, we denote by $\oplus$ the addition in $\mathbb{F}_2$.
There is an obvious way to deal with this obstacle: Whenever Alice wants to send $a \in \mathbb{F}_2$, she repeats the process of sending a, say, three times. Bob will receive three bits of information and predicts what Alice wanted to send by majority vote. The probability that Bob predicts incorrectly, in other words, the probability that at least two out of three bits are flipped, is then

$$p^3 + 3p^2(1-p) = p\left(3p - 3p^2\right),$$

which is an improvement to sending only one bit for any error probability $p < \frac{1}{2}$. This way of encoding information is known as 'repetition code'.

We want to generalize this: We notice that Alice sends an element of $\mathbb{F}_2^3$ to Bob to tell him about an element of $\mathbb{F}_2^1$, so a code should be some map

$$\mathcal{C} : \mathbb{F}_2^k \to \mathbb{F}_2^n,\ k \leq n.$$

We restrict to the case where this map $\mathcal{C}$ is linear. In order to be useful as code, we require the map $\mathcal{C}$ to be injective.
If Bob receives $(0,1,1)$ in the initial example, he immediately notices that something must have gone wrong. Why? There is no element of $\mathbb{F}_2$ that gets encoded to $(0,1,1)$. In the language of linear codes, $(0,1,1)$ is not an element of the *code space*, which is defined as

$$\left\{\mathcal{C}(v) : v \in \mathbb{F}_2^k\right\}.$$

In the example, this is just $\{(0,0,0),(1,1,1)\}$, which is a linear subspace of $\mathbb{F}_2^3$.

What is the systematic way of recognizing if some vector is in the code space? When Bob receives some vector v, he knows that it is in the code space if and only if all of its entries are equal. Bob can check this by checking if the first and the second entry are equal and same for the second and third. In other words, he checks if $(1,1,0)$ and $(0,1,1)$, two vectors orthogonal to all vectors in the code space, are orthogonal to v. We also notice that the code space contains all vectors which are orthogonal to both of the two.
Again, we can generalize this: The linear code $\mathcal{C} : \mathbb{F}_2^k \to \mathbb{F}_2^k$ can be represented by a matrix $M = (M_{ij}) \in \mathbb{F}_2^{n\times k}$, its matrix representation in the standard basis. Then the –linearly independent – columns of M span the code space of $\mathcal{C}$, so we can find $n-k$ vectors in $\mathbb{F}_2^n$ which are orthogonal to the codespace. This is equivalent to finding a matrix $H \in \mathbb{F}_2^{k\times n}$ such that

$$H \cdot M = 0.$$

We call H the parity check matrix of the code $\mathcal{C}$. Bob's parity check matrix could look like this:

$$H = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \end{pmatrix}$$

The next natural question is: How *good* is a code? This can be quantified by investigating under which conditions a code fails. Bob fails to decode Alice's message correctly if more than one bit is flipped. Say, Alice wants to send 0, so she puts $(0, 0, 0)$ in the channel, which flips the last two bits, Bob receives $(0, 1, 1)$. He decodes this as 1, because $(0, 1, 1)$ is in some way *closer* to $(1, 1, 1)$ than to $(0, 0, 0)$. To give a general notion of *closeness*, we define the *Hamming distance* of two vectors in $\mathbb{F}_2^n$ as the number of entries, where they differ. In Bob's case, he chooses $(1, 1, 1)$ instead of $(0, 0, 0)$, because the Hamming distance to $(0, 1, 1)$ of the former is 1, of the latter 2.
Let us generalize this. For a code $\mathcal{C}$, we define its *distance* as the minimal Hamming distance of two different code words, that is, two different vectors in the code space. If Alice encodes k bits in n bits using a distance d code and less than $\lfloor \frac{d-1}{2} \rfloor$ errors happen, the correct code word is still the unique element in the codespace closest to the word Bob receives. In other words, a distance d code can correct $\lfloor \frac{d-1}{2} \rfloor$ errors correctly. In the example, Bob can correct 1 error but not 2.
We summarize the concepts from this section:

Definition 3.1.1. A *classical linear code* is a linear map

$$\mathcal{C} : \mathbb{F}_2^k \to \mathbb{F}_2^n.$$

The image $\mathrm{im}(\mathcal{C})$ is called the *code space*. From now on we will denote both the encoding map and the code space by $\mathcal{C}$ – it will always be clear from context which one we mean. We call a matrix $H \in \mathbb{F}_2^{k \times n}$ whose rows are orthogonal to all vectors in the code space *parity check matrix*. The *distance* of a code is given my

$$\min \left(\sum_{i=1}^{n} (v_i \oplus w_i) | v \neq w \in \mathcal{C} \right).$$

We denote a classical linear code encoding k bits in n bits with code distance d by $[n, k, d]$.

Example 3.1.1 (Cycle codes)**.** There is a way of constructing codes with graphs: Let $G = (V, E)$ be an undirected, connected graph without loops or multiple edges. Denote $|V| = r$ and $|E| = n$. Without loss of generality denote $E = \{1, ..., n\}$ and $V = \{1, ..., r\}$.
We can identify a subset of E with a vector $v \in \{0, 1\}^n$ via

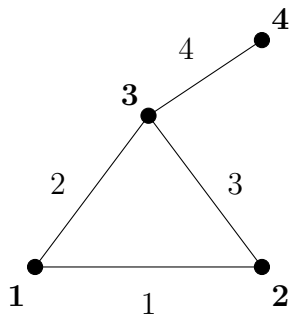
$$i \in E \text{ if and only if } v_i = 1.$$

A *cycle* in G is a subset $x \in \{0, 1\}^n$ of E such that any vertex of G is incident to an even number of edges of x. It is easy to see that the set of cycles is stable under addition modulo 2, in other words, a linear subspace of $\mathbb{F}_2^n$. It is therefore a linear code.
Define the *incidence matrix* of G

$$H = (h_{ij}) \in \{0, 1\}^{r \times n}$$

with $h_{ij} = 1$ if and only if the vertex i belongs to j. By construction, H has exactly two non-zero entries per column since an edge connects exactly two vertices. One can check that the rows of the incidence matrix are orthogonal to the code space. Therefore, we can construct a parity check matrix from it by taking a maximal linearly independent subset of the rows.

Since the set of cycles, that is, the code space, is stable under addition, the distance



Cycle code: $\mathrm{span}\left(\begin{pmatrix}1\\1\\1\\0\end{pmatrix}\right)$

$$H = \begin{pmatrix}1 & 1 & 0 & 0\\ 1 & 0 & 1 & 0\\ 0 & 1 & 1 & 1\\ 0 & 0 & 0 & 1\end{pmatrix}$$

Figure 3.1: An example of a graph with one cycle, its cycle code and incidence matrix.

of a cycle code is the minimal length of a cycle in G. This value is known as the *girth* of the graph G.

The theory of cycle codes can lead to interesting results. As an example, for a family of codes the *rate* $\frac{k}{n}$, which describes how many logical bits can be encoded per physical bit, is an interesting number. By constructing a suitable family of graphs, the *Cayley graphs* for $\mathrm{SL}(\mathbb{F}_p)$, one gets a family of codes with constant rate and distance growing logarithmically in n. More details can be found in the article of Gilles Zemor on page 259 in [CGL+09].

Example 3.1.2 (Dual codes)**.** Let $\mathcal{C}$ be a classical linear $[n, k]$ code with generating matrix M and parity check matrix H. We call the $[n, n-k]$ code with generating matrix H^T the *dual code* of $\mathcal{C}$ which we denote as $\mathcal{C}^\perp$.

Clearly, a parity check matrix of $\mathcal{C}^\perp$ is given by M^T as

$$M^T \cdot H^T = (H \cdot M)^T = 0$$

and $n - (n-k) = k$.

Let us construct a *self-dual* code, that is, a code $\mathcal{C}$ such that for all $v, w \in \mathcal{C}$ we have

$$v \cdot w = 0.$$

As this must be true for $v = w$, we see that every vector in the code needs an even number of non-zero entries. In other words, the number of 1's in each column of the generating matrix must be even.

Moreover, the codespace needs to be $\frac{n}{2}$-dimensional and the number of common non-zero entries of two different vectors in $\mathcal{C}$ must be even, too.

An example of a self-dual code could be represented like this:

$$M = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \end{pmatrix}$$

We easily convice ourself that M^T serves as parity check matrix.

Example 3.1.3 (Hamming code)**.** In this example we want to get our hands dirty by constructing the so-called *Hamming code*. We set the following task: Construct a classical linear code which encodes 4 bits of information and is capable of correcting any 1 bit error. That is, we look for

$$\mathcal{C}\colon \{0,1\}^4 \to \{0,1\}^n$$

with a parity check matrix

$$H = \begin{pmatrix} - & h_1 & - \\ & \vdots & \\ - & h_{(n-4)} & - \end{pmatrix} \in \{0,1\}^{n\times(n-4)}.$$

We want to be able to correct any bit flip. Suppose, our 4 bits of information get encoded as a vector v. Then, a single bit flip error can be modeled by adding e_i to v, where e_i is the i'th standard basis vector in $\{0,1\}^n$.
In order to detect errors, we then check for errors using the parity check matrix and calculate

$$H \cdot (v + e_i) = H \cdot v + H \cdot e_i = H \cdot e_i.$$

If we know, where the error has happend, that is, which e_i has been added, we can simply correct it and recover the original information by

$$v + e_i \mapsto (v + e_i) + e_i = v.$$

We know where the error has happened if and only if all of the $H \cdot e_i$ are different, so we need to be able to find n different vectors in the space $\{0,1\}^{(n-4)}$. This gives us a lower bound for n: Obviously, we cannot find 5 different vectors in

$$\{0,1\}^{(5-4)} = \{0,1\}^1.$$

Even 6 physical bits are not enough, as there are no 6 different vectors in

$$\{0,1\}^{(6-4)} = \{0,1\}^2.$$

Seven bits are enough, as we have 9 different elements in $\{0,1\}^3$ and only need 7. We can just take 7 of them and write them as columns in the parity check matrix. For example,

$$H = \begin{pmatrix} 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{pmatrix}$$

is the parity check matrix for a code that meets our desire. As it can correct any one bit error, it has distance greater or equal than 3. We can find two codewords

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

which have Hamming distance three, so the code distance is exactly 3.
We will use this code later on to construct a quantum code from it, the so-called *Steane code*. The following observation (which is nice by itself) will be crucial for this construction:
We easily see that all rows of H are pairwise orthogonal and, moreover, as every row has exactly 4 non zero entries, every row is orthogonal to itself. But this implies that the rows of H are valid code words, in other words, elements of the code space. Therefore, the dual code of the Hamming code is – as it is generated by the rows of H – a subcode of the Hamming code.

3.1.2 The Tanner graph of a code

A good way to visualize a code $\mathcal{C}$ and to recognize structure behind it lies in the so called *Tanner graph* of $\mathcal{C}$. Let H be the parity check matrix of $\mathcal{C}$ and denote by $h_1, \ldots, h_{n-k}$ the rows of H. Moreover, we denote by $h_i(j)$ the j'th entry of the i'th row of H. We start by defining the vertex set of the Tanner graph $G = (V, E)$.

$$V = \{1, \ldots, n\} \cup \{h_1, \ldots, h_{n-k}\},$$

that is, we have a vertex for every physical bit and also for every row of H.
The set of edges is given by

$$\{(i, h_j) : h_j(i) = 1\}.$$

Clearly, this graph is bipartite, there are no edges between two bits or between two rows of H. The vertices associated with the rows of H are sometimes called *check vertices*. So far, we do not need more than this definition. To make it more intuitive, we present the Tanner graphs of both the repetition and the Hamming code.

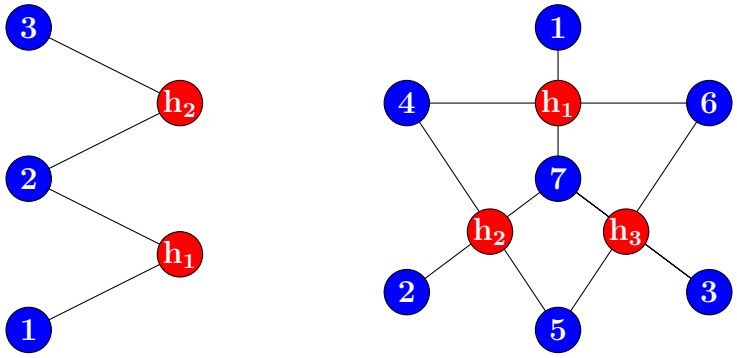


Figure 3.2: On the left the Tanner graph for the repetition code, on the right the Tanner graph of the Hamming code. The nodes for the physical bits are blue while the check vertices are red.

3.2 Introduction to quantum codes

In the quantum setting, Alice wants to send qubits to Bob, which are, say, states in some space $(\mathbb{C}^2)^{\otimes n}$. To increase readability we will sometimes write $\mathcal{H}$, $\mathcal{H}_A$ or something similar for this space.
Unfortunately, we immediately run into trouble if we try to apply the classical procedure by simply duplicating our information:

Theorem 3.2.1 (No-cloning theorem)**.** There is no physical map

$$\mathcal{E} : \mathcal{H}_A \otimes \mathcal{H}_B \to \mathcal{H}_A \otimes \mathcal{H}_B,$$

which takes $|\psi\rangle_A \otimes |0\rangle_B$ to $|\psi\rangle_A \otimes |\psi\rangle_B$ for all $|\psi\rangle_A \in \mathcal{H}_A$. In particular, we cannot 'copy' a given qubit.

Proof. A map with this property would obey

$$(|\psi\rangle_A + |\varphi\rangle_A) \otimes |0\rangle_B \mapsto (|\psi\rangle_A + |\varphi\rangle_A) \otimes (|\psi\rangle_B + |\varphi\rangle_B)\,. \tag{3.1}$$

But by linearity

$$(|\psi\rangle_A + |\varphi\rangle_A) \otimes |0\rangle_B = (|\psi\rangle_A \otimes |0\rangle_B) + (|\varphi\rangle_A \otimes |0\rangle_B)$$

must map to

$$(|\psi\rangle_A \otimes |\psi\rangle_B) + (|\varphi\rangle_A \otimes |\varphi\rangle_B)$$

which is not equal to the outcome in (3.1) if we choose $|\psi\rangle = |0\rangle$ and $|\varphi\rangle$ orthogonal to it. □

This is not the only additional obstacle when working with quantum bits instead of classical bits:

(i) In classical information theory the only possible error is the bit flip. In quantum information theory a noisy channel can introduce a variety of errors. In particular, the set of errors is a continuum.

(ii) Also, in classical information theory, Bob could just 'look' at the bits he received. In quantum information theory, measuring the qubits would possibly destroy valuable information.

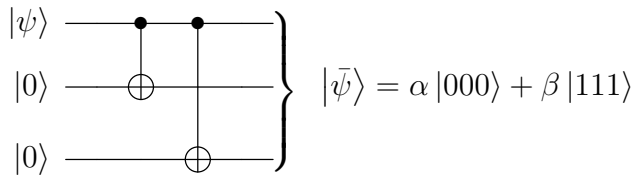
Let us try to generalize the classical repetition code: We define a linear map via

$$|0\rangle \mapsto |\bar{0}\rangle = |000\rangle, \;\; |1\rangle \mapsto |\bar{1}\rangle = |111\rangle,$$

which – by linearity – yields

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle \mapsto |\bar{\psi}\rangle = \alpha|000\rangle + \beta|111\rangle.$$

How can we realize this? The following circuit does exactly what we want:



The classical repetition code could protect against single bit flips. What happens if Alice encodes $|\psi\rangle$ as $|\bar{\psi}\rangle$ and the error channel introduces a Pauli-X at one position, which is analogous to a classical bit flip? Bob should definitely not measure in the computational basis since this collapses the state to a computational basis state and destroys any information about α and β.
Instead, Bob measures the two observables

$$Z_1Z_2 = Z \otimes Z \otimes \mathbb{1}, \quad Z_2Z_3 = \mathbb{1} \otimes Z \otimes Z$$

It is easy to see that such a measurement collapses $|\bar{\psi}\rangle$ to a state proportional to $|\bar{\psi}\rangle$, hence, no information is destroyed. It is also clear that the two measurement outcomes tell Bob exactly where the error has occured: Assume for example that the channel has flipped the second qubit. Then both measurements yield the outcome -1 with probability 1. If the first qubit would have been flipped, the measurement outcomes would be -1 and $+1$, for a flip on the third qubit $+1$ and -1. In other words, every single qubit flip yields a different *syndrome* which determines the position of the error. We will talk about *syndromes* in more detail in the next section when we introduce the stabilizer formalism.
Before we define the notion of a quantum code, let us investigate what happens with continuous errors in this situation. Say, the channel introduces the error

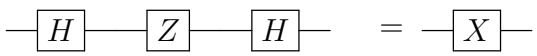
$$|\bar{\psi}\rangle \mapsto e^{i\theta X_1}|\bar{\psi}\rangle = \cos\theta\,(\alpha|000\rangle + \beta|111\rangle) + i\sin\theta\,(\alpha|100\rangle + \beta|011\rangle)\,.$$

Measuring Z_1Z_2 yields outcome $+1$ with probability $(\cos\theta)^2$ and -1 with probability $(\sin\theta)^2$. In the first case, the state gets collapsed to the error-free version, in the second case to $X_1|\bar{\psi}\rangle$, thus we can correct this kind of error by the above procedure.
Unfortunately this code has a severe weakness: If the channel introduces a single Pauli-Z error on any qubit, the state Bob receives is a valid – but different – code word:

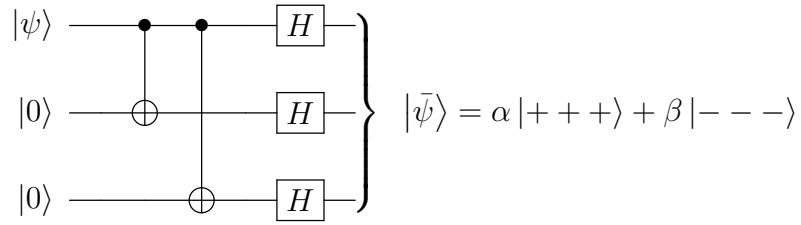
$$|\bar{\psi}\rangle = \alpha|000\rangle + \beta|111\rangle \mapsto Z_i|\bar{\psi}\rangle = \alpha|000\rangle - \beta|111\rangle,\ i = 1, 2, 3$$

which is the code word for $|\psi'\rangle = \alpha|0\rangle - \beta|1\rangle$.

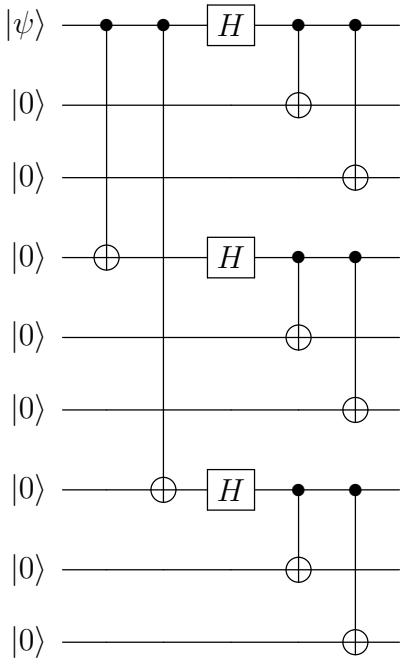
There is an analogous way of encoding information to protect against Z-errors. We have



In other words, applying a Hadamard gate switches between X and Z eigenstates.



This circuit implements the *sign flip code*. It can correct any one qubit Z error: By measuring the observables X_1X_2 and X_2X_3, Bob can detect in a similiar manner where the Z-error has happened and reconstruct the correct code word.
It is possible to *concatenate* these codes which results in the *9 qubit Shore code* which is able to correct any one qubit error. We will not think about it in detail, but let us mention that it can be implemented by the following circuit:



Now that we have gained some intuition on quantum codes we finalize this section by giving a formal definition of a quantum code:

Definition 3.2.1. A quantum code encoding k logical qubits in n physical qubits is a 2^k-dimensional subspace of the complex Hilbert space

$$\mathcal{H} = \left(\mathbb{C}^2\right)^{\otimes n}.$$

The distance d of a quantum code is the minimum weight – in other words the number of non-identity components – of a Pauli operator $P \in \mathcal{P}^{\otimes n}$ such that we can find two different states in the code space $|v\rangle$ and $|w\rangle$ such that

$$P|v\rangle \propto |w\rangle.$$

This is exactly the generalization of the distance of a classical linear code. We also denote a quantum code encoding k logical qubits in n physical qubits with distance d by $[n, k, d]$.

Example 3.2.1 (CSS-Codes)**.** We have seen how to generalize our simple example for classical linear codes to quantum codes. In fact, there is a general construction to turn certain pairs of classical linear codes into quantum codes, so-called *CSS-Codes*: We start with two classical linear codes $\mathcal{C}_1$ and $\mathcal{C}_2$ which are $[n, k_1, d_1]$ and $[n, k_2, d_2]$ codes. Moreover, suppose that $\mathcal{C}_2 \subset \mathcal{C}_1$, so in particular $k_2 \leq k_1$. We can associate a state $|\psi\rangle \in (\mathbb{C}^2)^{\otimes n}$ with a vector $(x_1, \ldots, x_n) \in \mathcal{C}_1$ via $|\psi\rangle = |x_1 \ldots x_n\rangle$.
Consider the state

$$|x + \mathcal{C}_2\rangle \coloneqq \frac{1}{\sqrt{|\mathcal{C}_2|}} \sum_{y \in \mathcal{C}_2} |x \oplus y\rangle.$$

Clearly, we have for $x + y = x' \in \mathcal{C}_1$ with $y \in \mathcal{C}_2$ that

$$|x + \mathcal{C}_2\rangle = |x' + \mathcal{C}_2\rangle.$$

Therefore, the state $|x + \mathcal{C}_2\rangle$ only depends on the *coset* $x + \mathcal{C}_2$ in $\mathcal{C}_1$.
Now, let $|x + \mathcal{C}_2\rangle$ and $|x' + \mathcal{C}_2\rangle$ be states for x, x' belonging to different cosets. As the cosets of x and x' are disjoint, we have

$$x + y \neq x' + y'$$

for all y, y' in $\mathcal{C}_2$. In other words, all the summands of $|x + \mathcal{C}_2\rangle$ and $|x' + \mathcal{C}_2\rangle$ are pairwise orthogonal (as they are different computational basis states), so $|x + \mathcal{C}_2\rangle$ and $|x' + \mathcal{C}_2\rangle$ are orthogonal.
We define the *CSS-Code* of $\mathcal{C}_1$ over $\mathcal{C}_2$ as

$$CSS\left(\mathcal{C}_1, \mathcal{C}_2\right) \coloneqq \operatorname{span}\left(|x + \mathcal{C}_2\rangle : x \in \mathcal{C}_1\right).$$

By the previous discussion, its dimension is equal to the dimension of the quotient space

$$\dim\left(\mathcal{C}_1 / \mathcal{C}_2\right) = \frac{2^{k_1}}{2^{k_2}} = 2^{k_1 - k_2}.$$

One can see that if both $\mathcal{C}_1$ and $\mathcal{C}_2^{\perp}$ can correct up to t errors, $CSS(\mathcal{C}_1, \mathcal{C}_2)$ is also capable of correcting errors on t qubits. A detailed proof of this may be found in Chapter 10 of [NC11].

3.3 The stabilizer formalism

3.3.1 Stabilizer groups and stabilizer codes

Recall that we defined a classical linear code as a linear map

$$\mathcal{C}\colon \{0,1\}^k \to \{0,1\}^n$$

where the code words are the images of the vectors in $\{0,1\}^k$.
We also generalized this by defining quantum codes as isometric maps

$$\mathcal{C}\colon \left(\mathbb{C}^2\right)^{\otimes k} \to \left(\mathbb{C}^2\right)^{\otimes n}.$$

In Chapter 2, we saw that in many situations it is more convenient to work with the parity check matrix of a classical linear code and characterize the code space via

$$\mathcal{C} = \mathcal{C}\left(\{0,1\}^k\right) = \{v \in \{0,1\}^n \colon Hv = 0\}.$$

The aim of this section is to find a similar formalism for quantum codes.
Notice that the condition

$$v \in \mathcal{C} \text{ if and only if } Hv = 0$$

is equivalent to

$$v \in \mathcal{C} \text{ if and only if } \left(\mathbb{1} - ww^T\right) v = v \text{ for all rows } w \text{ of } H.$$

In other words, every $[n,k]$ code can be specified by $n-k$ linear operators that fix the code space.
This is something we can easily generalize to quantum codes: The *stabilizer codes* are codes whose code space is the common $+1$ eigenspace of a group of Pauli operators. Here, we write $\mathcal{P}(n)$ for the n-qubit Pauli group $\{\pm 1, \pm i\} \cdot \{\mathbb{1}, X, Y, Z\}^{\otimes n}$.

Definition 3.3.1. Let $\mathcal{S}$ be an abelian subgroup of the n-qubit Pauli group $\mathcal{P}(n)$. The set

$$\left\{|\psi\rangle \in \left(\mathbb{C}^2\right)^{\otimes n} : S|\psi\rangle = |\psi\rangle \text{ for all } S \in \mathcal{S}\right\}$$

is called *stabilizer code* (or *code space of the stabilizer code*) associated with $\mathcal{S}$. The group $\mathcal{S}$ is called the *stabilizer group*, its elements *stabilizer operators* or *stabilizers*. Note that the stabilizer operators must be abelian in order to have non-trivial common $+1$ eigenstates. We also notice that this definition only makes sense if $-\mathbb{1}$ is not an element of $\mathcal{S}$. In the following we assume that this is true.

Remark. Of course, neither a parity check matrix nor a stabilizer group fixes one specific encoding map, it only fixes the code space. But, as for the properties of a code the particular way of encoding is irrelevant, we do not need to worry about that.

Fortunately, we do not need to work with the full group of stabilizer operators. Let $\mathcal{S}$ be a stabilizer group and choose generators of this group $S_1, \ldots, S_m$, that is, any element of $\mathcal{S}$ is a product of certain generators. We easily recognize that

$$S\,|\psi\rangle = |\psi\rangle \text{ for all stabilizer operators } S$$

is equivalent to

$$S_i |\psi\rangle = |\psi\rangle \text{ for all stabilizer generators } S_i.$$

Therefore, we can always work with a generating set of the stabilizer group.

Let us once again look at the classical case. Encoding k bits into n bits gave $n-k$ *linearly independent* rows w in the parity check matrix, so $n-k$ linearly independent 'stabilizers' $\mathbb{1} - ww^T$.
A generalization of this is also true for stabilizer codes:

Theorem 3.3.1. [NC11, Chapter 10] Let $\mathcal{S} = \langle S_1, ..., S_{n-k} \rangle \subset \mathcal{P}(n)$ be the subgroup generated by the S_i. We assume that the S_i's commute and are independent in the sense that no generator is a product of the other generators. Then $\mathcal{S}$ defines a stabilizer code whose code space has dimension 2^k.

3.3.2 A different representation of stabilizer codes

Consider a stabilizer group $\mathcal{S} = \langle S_1, \ldots, S_{n-k} \rangle \subset \mathcal{P}(n)$ given by independent stabilizer generators. As the S_i's are elements of $\mathcal{P}(n)$, they can be written – up to a factor in $\{\pm 1, \pm i\}$ – as

$$S_i = \bigotimes_{j=1,\ldots,n} X^{\alpha_{ij}} Z^{\beta_{ij}},$$

where $\alpha_{ij}, \beta_{ij} \in \{0, 1\}$. All the information about the stabilizer S_i is contained in the binary vectors $\alpha_i = (\alpha_{ij})_{j=1,\ldots,n}$ and $\beta_i = (\beta_{ij})_{j=1,\ldots,n}$. Therefore, we can specify a stabilizer code as

$$\left(\begin{array}{c|c} & \\ M_X & M_Z \\ & \end{array} \right) \in \{0,1\}^{(n-k)\times(n+n)}$$

where the rows of M_X are the vectors α_i, the rows of M_Z the vectors β_i.
As an example, the stabilizer code of the repetition code can be specified as

$$\left(\begin{array}{ccc|ccc} 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 \end{array} \right),$$

where we chose $Z_1 Z_2$ and $Z_2 Z_3$ as generators for the stabilizer group.
To be a reasonable stabilizer group, the generators have to commute. Let S_i and S_j be two generators of a stabilizer group, that is, $[S_i, S_j] = 0$. Write $\alpha_i|\beta_i, \alpha_j|\beta_j$ for the corresponding binary vectors. The S_i commute if and only if an even number of X's and Z's interact, in other words

$$\alpha_i \cdot \beta_j \oplus \beta_i \cdot \alpha_j = 0.$$

It is often more comfortable to work with this formalism than with the definition of the stabilizer code.

3.3.3 Evolution and the stabilizer formalism

Our next natural question is: What happens to the code space and its stabilizers under quantum operations? Say we apply a unitary U to the code space $\mathcal{C}$ of some stabilizer code $\mathcal{S}$. The new code space is now

$$\{U|\psi\rangle : |\psi\rangle \in \mathcal{C}\}.$$

It is easy to see that

$$\left(USU^\dagger\right) U|\psi\rangle = U|\psi\rangle.$$

Clearly, $U\mathcal{S}U^\dagger$ is commutative if $\mathcal{S}$ is, so the evolution U changes the code space stabilized by $\mathcal{S}$ to the space stabilized by

$$U\mathcal{S}U^\dagger = \left\{USU^\dagger : S \in \mathcal{S}\right\}. \tag{3.2}$$

Note that $U\mathcal{S}U^\dagger$ is not necessarily a subset of the Pauli group $\mathcal{P}(n)$. Therefore, U transforms $\mathcal{S}$ to a proper stabilizer code only if $USU^\dagger$ is a Pauli operator for all S in $\mathcal{S}$.

Example 3.3.1. Consider again the repetition code. As we have seen, it has stabilizer $\mathcal{S} = \{Z_1Z_2, Z_2Z_3\}$ and these two operators are obviously independent.
Let us apply the unitary

$$H^{\otimes 3} = H \otimes H \otimes H.$$

As seen in the last chaper, we have

$$H^{\otimes 3} (Z \otimes Z \otimes \mathbb{1}) \left(H^{\otimes 3}\right)^\dagger = X \otimes X \otimes \mathbb{1}$$

$$H^{\otimes 3} (\mathbb{1} \otimes Z \otimes Z) \left(H^{\otimes 3}\right)^\dagger = \mathbb{1} \otimes X \otimes X$$

Hence, applying a Hadamard gate to every physical qubit switches between repetition and sign flip code.

3.3.4 Measurements and the stabilizer formalism

Instead of applying a unitary we can measure the qubits. For simplicity, we restrict to the following example which is sufficient for our results in the following chapters: Let us consider a stabilizer code $\mathcal{S} = \{S_1, \ldots, S_n\}$ with n independent generators on $\mathcal{H} = (\mathbb{C}^2)^{\otimes n}$, which gives us a $2^0 = 1$ dimensional code space, that is, it fixes one state $|\psi\rangle$ up to a phase. We also assume that the observable M is an element of the n-qubit Pauli group $\mathcal{P}(n)$.
By the latter assumption, we know that M either commutes or anticommutes with each element of the stabilizer.
We start with the simplest case where M commutes with all generators of the stabilizer and therefore with all elements of the stabilizer. This yields

$$M|\psi\rangle = MS|\psi\rangle = SM|\psi\rangle$$

which tells us that $M|\psi\rangle$ is in the code space. Since the code space is the ray through ψ, it is therefore a multiple of $|\psi\rangle$. M is an element of $\mathcal{P}(n)$, which implies $M^2 = \mathbb{1}$,

so $M|\psi\rangle = \pm|\psi\rangle$. Hence, the updated codespace has the same stabilizers as the old one, and we conclude:

$$\text{Measuring } M \text{ with } [M, S_i] = 0 \text{ for all } i = 1, \ldots, n \text{ does } \textit{not} \text{ change the stabilizer group.} \tag{3.3}$$

In the second case, M anticommutes with certain stabilizer generators, without loss we assume it anticommutes with $S_1, \ldots, S_l$ and commutes with $S_{l+1}, \ldots, S_n$. Clearly, we have

$$\mathcal{S} = \langle S_1, \ldots, S_n \rangle = \langle S_1, S_2 S_1, \ldots, S_l S_1, S_{l+1}, \ldots S_n \rangle ,$$

where all generators except S_1 commute with M. Independent on the measurement outcome, $S_2 S_1, \ldots, S_l S_1, S_{l+1}, \ldots S_n$ are still stabilizers. Moreover, we have a new stabilizer M, which is independent of the former $n-1$ stabilizers. If the measurement outcome is -1, it is easy to see that $-M$ becomes a stabilizer. Summarized we have the new stabilizer

$$\mathcal{S} = \langle \pm M, S_2 S_1, \ldots S_l S_1, S_{l+1}, \ldots, S_n \rangle \tag{3.4}$$

depending on the measurement outcome of M.

Example 3.3.2. Consider again the stabilizer code for the repetition code with stabilizer $\mathcal{S} = \langle Z_1 Z_2, Z_2 Z_3 \rangle$.
If we measure the observable $X_1 = X \otimes \mathbb{1} \otimes \mathbb{1}$, we get a new stabilizer, depending on the measurement outcome, $\tilde{\mathcal{S}} = \langle \pm X_1, Z_2 Z_3 \rangle$.
If we instead measure $X_1 X_3$, both of the former stabilizers anticommute with the observable, so by the update rule the stabilizer changes to $\tilde{\mathcal{S}} = \langle \pm X_1 X_3, Z_1 Z_3 \rangle$.

Remark. Notice that in the preceding example, in both cases, the probability of outcome ± 1 was $\frac{1}{2}$ each. In fact, this always holds in the case of stabilizer codes as the following calculation shows:

$$\begin{aligned} \mathrm{P}(+1) &= \langle\psi| \left(\frac{\mathbb{1} + M}{2}\right) |\psi\rangle = \langle\psi| \left(\frac{\mathbb{1} + M}{2}\right) S_1 |\psi\rangle = \langle\psi| S_1 \left(\frac{\mathbb{1} - M}{2}\right) |\psi\rangle \\ &= \langle\psi| \left(\frac{\mathbb{1} - M}{2}\right) |\psi\rangle = \mathrm{P}(-1) \end{aligned}$$

where we assumed that S_1 anticommutes with M and used the fact, that S_1 is Hermitian and stabilizes $|\psi\rangle$.

3.3.5 Stabilizers of CSS-Codes

In the previous chapter we saw how to construct so-called CSS-Codes from classical codes. We now want to see that all CSS-Codes are stabilizer codes where the stabilizer generators can be constructed from the parity check matrices of the corresponding classical codes.

Let again $\mathcal{C}_2 \subset \mathcal{C}_1$ be classical linear $[n, k_2]$ and $[n, k_1]$ codes. Recall that the code space of the CSS-Code was defined in Section 3.2.1 as

$$\mathcal{C} = \left\{ |x + \mathcal{C}_2\rangle = \sum_{y \in \mathcal{C}_2} |x + y\rangle : x \in \mathcal{C}_1 \right\}$$

Moreover, we now need the dual code of $\mathcal{C}_2$, a concept we introduced in example 3.1.2. We can specify a stabilizer state as in Section 3.3.2 by

$$\left(\begin{array}{c|c} H(\mathcal{C}_2^{\perp}) & 0 \\ 0 & H(\mathcal{C}_1) \end{array} \right), \tag{3.5}$$

where a 1 on the lefthand side characterizes an X in the stabilizer generator, one on the righthand side a Z. In particular, all stabilizer generators are elements of $\{\mathbb{1}, X\}^{\otimes n}$ or $\{\mathbb{1}, Z\}^{\otimes n}$. We also refer to them as X-type or Z-type stabilizers.

First of all we want to see that the generators commute. But this simply follows from $\mathcal{C}_2 \subset \mathcal{C}_1$: A row v of $H(\mathcal{C}_2^{\perp})$ is an element of the codespace $\mathcal{C}_2$. Therefore, every row w of the parity check matrix $H(\mathcal{C}_1)$ is orthogonal to v. This implies that there is an even number of positions where both v and w are 1. Translated to the product of the stabilizers arising from these rows this means that there is an even number of X's that get multiplied with a Z. This clearly shows that they commute.

We can identify this as a stabilizer of $\mathcal{C}$: Let v be one of the X-type rows of Equation (3.5). Applying the corresponding operator X_v to a state yields

$$X_v \colon |x\rangle \mapsto |x + v\rangle \,, \; x \in \{0, 1\}^n.$$

So, for an element of the code space of the CSS-Code this operator acts as

$$X_v \colon |x + \mathcal{C}_2\rangle = \sum_{y \in \mathcal{C}_2} |x + y\rangle \mapsto \sum_{y \in \mathcal{C}_2} |x + y + v\rangle = |x + \mathcal{C}_2\rangle \,,$$

that is, it stabilizes the state.
Next, we look at the Z-type operators. Let w be the right part of a corresponding row of Equation (3.5), and let $x \in \mathcal{C}_1$ and $y \in \mathcal{C}_2$. As w is the row of the parity check matrix of $\mathcal{C}_1$ and $\mathcal{C}_2 \subset \mathcal{C}_1$, we see that w is orthogonal to both x and y. Therefore it is also orthogonal to $x + y$. In other words, the number of positions m_{x+y} where both w and $x + y$ are 1 is even. So the operator Z_w associated with w acts as

$$Z_w \colon |x + \mathcal{C}_2\rangle = \sum_{y \in \mathcal{C}_2} |x + y\rangle \mapsto \sum_{y \in \mathcal{C}_2} (-1)^{m_{x+y}} |x + y\rangle = |x + \mathcal{C}_2\rangle \,,$$

as we get a factor of -1 whenever a Z hits a 1 and all m_{x+y} are even.
This shows that the stabilizer generators defined in Equation (3.5) in fact stabilize the code space of the CSS-Code. We still have to convince ourselves that there are no states outside of $\mathcal{C}$ stabilized by this stabilizer.

In order to do so, let $|\psi\rangle = \sum_{i=1}^{2^n} \alpha_i |e_i\rangle$ be a state written as its *unique* linear combination of the standard basis vectors. Assume that $|\psi\rangle$ is stabilized by all

operators associated with the rows of Equation (3.5).
Since it is stabilized by the X-type operators, we have

$$\sum_{i=1}^{2^n} \alpha_i |e_i\rangle = \sum_{i=1}^{2^n} \alpha_i |e_i + y\rangle$$

for all $y \in \mathcal{C}_2$. By the uniqueness of the linear coefficients this gives $\alpha_i = \alpha_j$ whenever $e_i = e_j + y$ for some $y \in \mathcal{C}_2$. Pick some basis vectors $|e_1\rangle, \ldots, |e_m\rangle$ such that we have a basis

$$\{|e_i + y\rangle : i = 1, \ldots, m, y \in \mathcal{C}_2\}$$

of the whole space $(\mathbb{C}^2)^{\otimes n}$.
By the above argument we can write

$$|\psi\rangle = \sum_{i=1}^{m} \alpha_i \left(\sum_{y \in \mathcal{C}_2} |e_i + y\rangle \right),$$

where the coefficients α_i are still unique. Now we want to use that the Z-type stabilizers act trivially on $|\psi\rangle$.
Let e_i not in $\mathcal{C}_1$ be a binary n-dimensional vector. As it is not an element of $\mathcal{C}_1$, it cannot be in the kernel of the parity check matrix, in other words there is a row w of $H(\mathcal{C}_1)$ such that the number of positions where both w and e_i are 1 is odd. Translating this vector w to the corresponding stabilizer generator Z_w and bearing in mind that $\mathcal{C}_2 \subset \mathcal{C}_1$ gives us

$$Z_w \colon \sum_{y \in \mathcal{C}_2} |e_i + y\rangle \mapsto - \sum_{y \in \mathcal{C}_2} |e_i + y\rangle .$$

Therefore, $Z_w |\psi\rangle$ has coefficients $-\alpha_i$ where $|\psi\rangle$ has coefficients α_i. Since we assume that Z_w stabilizes $|\psi\rangle$ and the state determines the coefficients uniquely, this implies that $\alpha_i = 0$ whenever $e_i \notin \mathcal{C}_1$. Hence, every state stabilized by all operators associated with rows of Equation (3.5) is an element of the code space of the CSS-Code space $\mathcal{C}$.

Example 3.3.3 (Steane code)**.** As announced in example 3.1.3, we want to build a quantum code from the classical code in this example. Remember that we had a classical linear code encoding 4 bits of information in 7 physical bits, the code space was specified by the parity check matrix

$$H = \begin{pmatrix} 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{pmatrix}.$$

We denote the code space of the Hamming code by $\mathcal{C}_1$, the code space of the dual code by $\mathcal{C}_2$. Recall that $\mathcal{C}_2 \subset \mathcal{C}_1$. Therefore, we can construct a CSS-Code from $\mathcal{C}_1$ and $\mathcal{C}_2$. As $\mathcal{C}_2$ is the dual code of $\mathcal{C}_1$, we easily see that the dual code of $\mathcal{C}_2$ is $\mathcal{C}_1$, so Equation 3.5 looks like this:

$$\left(\begin{array}{c|c} H & 0 \\ 0 & H \end{array} \right)$$

Writing out what this means, the stabilizer of the Steane code is given by

$$\mathcal{S} = \langle X_1X_4X_6X_7, X_2X_4X_5X_7, X_3X_5X_6X_7, \\ Z_1Z_4Z_6Z_7, Z_2Z_4Z_5Z_7, Z_3Z_5Z_6Z_7 \rangle$$

3.4 The graph state

Equipped with the stabilizer formalism, we can now do something similar as we did while considering classical codes: We now show two ways to construct stabilizer codes from graphs which yields a plethora of examples for stabilizer codes. Later on we will define the surface code on a lattice, but let us start with the so-called *graph states*.
Let $G = (V, E)$ be an undirected, connected graph without loops or multiple edges. With every vertex of G, we associate a qubit. For each $i \in V$ we define

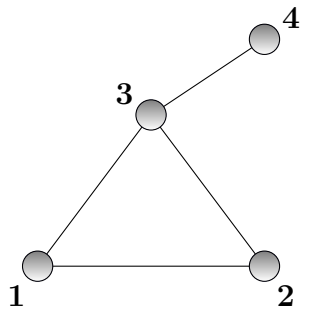
$$S_i = X_i \prod_{j \in \text{neigh}(i)} Z_j \tag{3.6}$$

where neigh(i) denotes the neighbourhood of the vertex i. Since the neighbourhood relation is symmetric, the operators S_i commute pairwise. They are also independent:
The product of certain S_i's with $i \neq j$ can only have $\pm\mathbb{1}$ or $\pm Z$ as its j'th component, so it cannot be equal to S_j which has an X in the j'th entry.
Therefore, these operators define a stabilizer code fixing one qubit as the code space has dimension

$$2^{(|V|-|V|)} = 1.$$

Note that this is not suitable for encoding one qubit. Later, we will use the graph state more as a 'resource' than as a code.

Definition 3.4.1. For a graph $G = (V, E)$, the code space of the stabilizer code with stabilizers as in Equation (3.6) is called *graph state*.



$$\mathcal{S} = \langle X_1 Z_2 Z_3, Z_1 X_2 Z_3, Z_1 Z_2 X_3 Z_4, Z_3 X_4 \rangle$$

Figure 3.3: A simple graph and the stabilizer for the corresponding graph state.

Later on, we want to create encoded long range entanglement by performing certain measurements on graph states. Therefore, we should know how to prepare a graph state. For this we need the controlled-Z

$$U_{ij} = (|0\rangle \langle 0|_i \otimes \mathbb{1}_j + |1\rangle \langle 1|_i \otimes Z_j)$$

between the qubits i and j.
It can be represented by the matrix

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix} \tag{3.7}$$

Note that the controlled-Z gate is symmetric, that is, $U_{ij} = U_{ji}$. Moreover, we see that $U_{ij} = U_{ij}^\dagger$.
The following theorem tells us how we can construct any graph state from a product state using controlled-Z gates:

Theorem 3.4.1. Let $G = (V, E)$ be a graph. Denote the stabilizers for the graph state for each vertex v as S_v. Then

$$|G\rangle = \left(\prod_{(i,j)\in E} U_{ij} \right) |+\rangle^{\otimes |V|}$$

is an element of the code space.

Proof. We need to show that

$$\left(X_v \prod_{u\in \mathrm{neigh}(v)} Z_u \right) \left(\prod_{(i,j)\in E} U_{ij} \right) |+\rangle^{\otimes |V|} = \left(\prod_{(i,j)\in E} U_{ij} \right) |+\rangle^{\otimes |V|}$$

For convenience, we will omit normalization. With this we can write

$$|+\rangle^{\otimes |V|} = \sum_{\underline{a}\in\{0,1\}^{|V|}} |\underline{a}\rangle \,. \tag{3.8}$$

The next thing we observe is that U_{ij} commutes with S_v whenever v is not incident to (i, j), and of course, all of the U_{ij} commute pairwise. Therefore we have

$$S_v \left(\prod_{(i,j)\in E} U_{ij} \right) |+\rangle^{\otimes |V|} = \left(\prod_{(i,j)\in E, v\notin(i,j)} U_{ij} \right) S_v \left(\prod_{(i,j)\in E, v\in(i,j)} U_{ij} \right) |+\rangle^{\otimes |V|}$$

and it suffices to show

$$S_v \left(\prod_{(i,j)\in E, v\in(i,j)} U_{ij} \right) |+\rangle^{\otimes |V|} = \left(\prod_{(i,j)\in E, v\in(i,j)} U_{ij} \right) |+\rangle^{\otimes |V|} \,. \tag{3.9}$$

Let us do this by investigating the impact of both sides of 3.9 to the summands in 3.8. Define

$$L = S_v \left(\prod_{(i,j)\in E, v\in(i,j)} U_{ij} \right) |+\rangle^{\otimes |V|} \,, \quad R = \left(\prod_{(i,j)\in E, v\in(i,j)} U_{ij} \right) |+\rangle^{\otimes |V|}$$

and

$$|\tilde{\underline{a}}\rangle = \left| a_1 \dots (a_v \oplus 1) \dots a_{|V|} \right\rangle = X_v \, |\underline{a}\rangle$$

We distinguish four cases:

Case 1: $a_v = 0$ and $|\{w \in \mathrm{neigh}(v) : a_w = 1\}|$ even. Then

$$L \, |\underline{a}\rangle = |\tilde{\underline{a}}\rangle \,, \quad R \, |\underline{a}\rangle = |\underline{a}\rangle \,.$$

Case 2: $a_v = 0$ and $|\{w \in \text{neigh}(v) : a_w = 1\}|$ odd. Then

$$L\,|\underline{a}\rangle = -\,|\underline{\tilde{a}}\rangle\,,\quad R\,|\underline{a}\rangle = |\underline{a}\rangle\,.$$

Case 3 $a_v = 1$ and $|\{w \in \text{neigh}(v) : a_w = 1\}|$ even. Then

$$L\,|\underline{a}\rangle = |\underline{\tilde{a}}\rangle\,,\quad R\,|\underline{a}\rangle = |\underline{a}\rangle\,.$$

Case 4: $a_v = 1$ and $|\{w \in \text{neigh}(v) : a_w = 1\}|$ odd. Then

$$L\,|\underline{a}\rangle = |\underline{\tilde{a}}\rangle\,,\quad R\,|\underline{a}\rangle = -\,|\underline{a}\rangle\,.$$

We observe that the summands of $L\,|+\rangle^{\otimes|V|}$ and $R\,|+\rangle^{\otimes|V|}$ are equal up to permutation, as the summands from the cases 1 and 3 as well as the summands from 2 and 4 are in one to one correspondence, which yields Equation (3.9). This concludes the proof. □

There is another way of seeing that U_{ij} – the controlled-Z gate between the qubits i and j – corresponds to adding or deleting the edge (i, j) using the update rule for stabilizer codes from Equation (3.2). Suppose we have a graph state $|G\rangle$ specified by a stabilizer

$$\mathcal{S} = \langle S_1, \ldots S_n\rangle\,,$$

to which we apply U_{ij} for two nodes i and j. By Equation (3.2), this acts on the stabilizers as

$$S_k \mapsto U_{ij} S_k (U_{ij})^\dagger = U_{ij} S_k U_{ij}.$$

It is easy to see that this does not change the stabilizers S_k for $k \neq i, j$ as U_{ij} commutes with them. Moreover, we have

$$U_{ij} S_k U_{ij} = U_{ij} X_k \prod_{l \in \text{neigh}(k)} Z_l U_{ij} = U_{ij} X_k U_{ij} \prod_{l \in \text{neigh}(k)} Z_l$$

as the controlled-Z gate also commutes with all Z terms. An easy calculation shows

$$U_{ij} X_i U_{ij} = Z_j X_i$$
$$U_{ij} X_j U_{ij} = X_j Z_i$$

which yields that applying the controlled-Z gate corresponds to adding or deleting the edge (i, j).
Note that this yields an alternative proof of Theorem 3.4.1 by applying this succesively for every edge of the graph starting from the graph without edges.

Example 3.4.1. As an example, we want to apply our thoughts from the previous chapter and see what happens with the graph state under measurements. We only demonstrate the simplest example – measuring one qubit in the Z basis. Consider a graph state for a graph $G = (V, E)$ with $V = \{1, \ldots, n\}$. Suppose, we measure the qubit sitting at vertex i in the Z-basis. Obviously, all stabilizers except

$$X_i \prod_{j \in \text{neigh}(i)} Z_j$$

commute with the observable Z_i. Therefore we know from Equation (3.4) that the resulting state has stabilizer

$$\mathcal{S} = \langle \pm Z_i, S_j, j \neq i \rangle$$

where the S_j stay unchanged. We easily see that the resulting state is a graph state on the new graph $G' = G - \{i\}$ and an isolated qubit which is a Z-eigenstate. Up to a local unitary – namely a Hadamard gate on the qubit i – this is again a graph state.

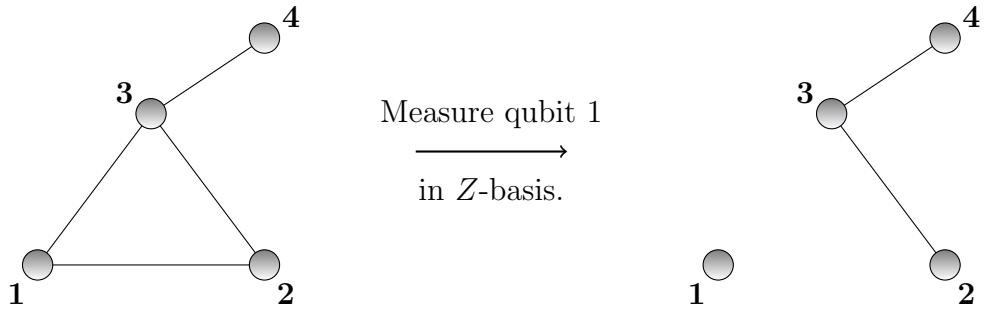


Figure 3.4: Measuring the qubit sitting at the vertice 1 in Z-basis: The new state is again a graph state.

There are similar – but far more complicated – update rules for X and Y measurements. They are explained in detail in [HEB04].

3.5 The surface code

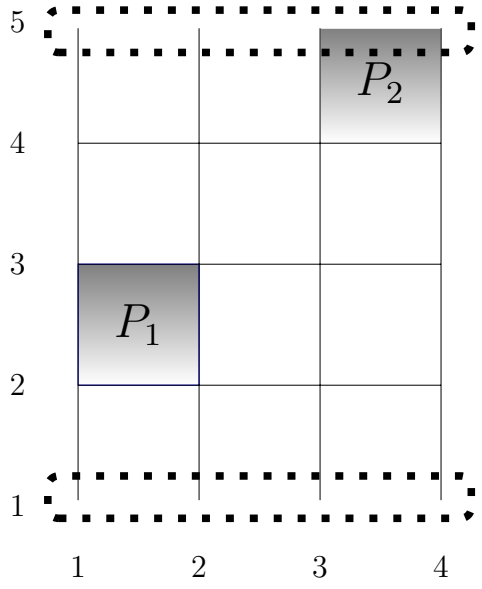
In this section we will see a promising quantum code – the surface code. It is defined on physical qubits sitting on the edges of a 2D lattice. As we will see, it has a distance growing linearly in the linear size of the lattice. The surface code is a planar variant of Kitaev's toric code introduced in [Kit03].

In order to simplify the following arguments, we first give some terminology about 2D lattices.

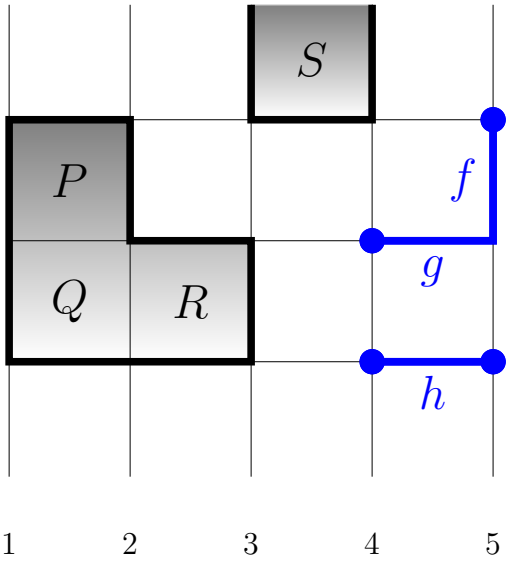
Definition 3.5.1. A 2D $k \times m$ lattice Λ (with rough and smooth boundaries) has

- vertices (i, j) for $1 \leq i \leq k$ and $1 \leq j \leq m$,
- (vertical) edges $((i, j), (i, j + 1))$ for $1 \leq i \leq k$ and $1 \leq j \leq m - 1$,
- (horizontal) edges $((i, j), (i + 1, j))$ for $1 \leq i \leq k - 1$ and $2 \leq j \leq m - 1$,
- plaquettes $((i, j), (i + 1, j), (i + 1, j + 1), (i, j + 1))$ for $1 \leq i \leq k - 1$ and $1 \leq j \leq m - 1$.

A look at Figure 3.5a explains the terms *rough* and *smooth* boundary.



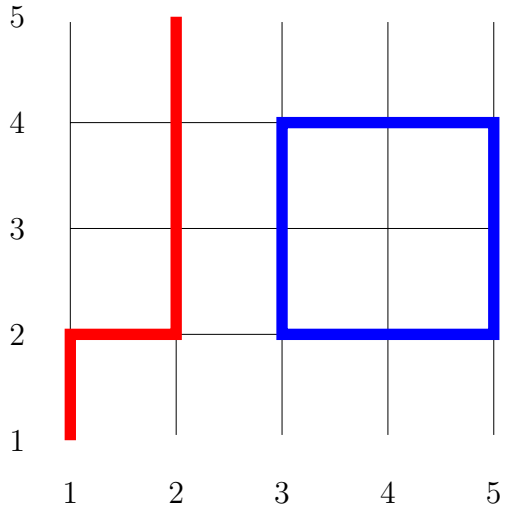
(a) A 4×5 lattice. P_1 and P_2 are examples for plaquettes. The dashed areas highlight the rough boundaries. On the left and right hand side are the smooth boundaries.



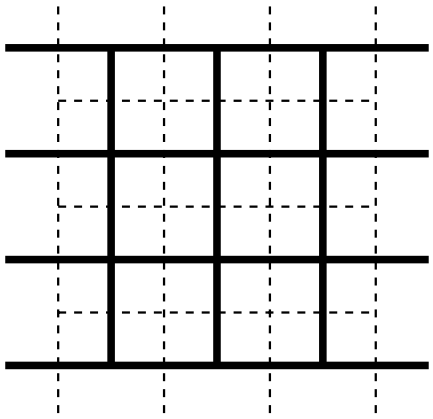
(b) In gray the 2-chain $P+Q+R+S$. Its boundary $\partial_2(P + Q + R + S)$ is highlighted boldly. In blue the 1-chain $f + g + h$. Its boundary are the blue circles.

Figure 3.5: Lattices with some plaquettes (a) and some chains (b).

We call the free $\mathbb{Z}_2$ vector space $H_2(\Lambda)$ generated by the plaquettes of Λ the set of 2-chains. It consists of formal sums of plaquettes with coefficients 0 or 1.



(a) A homologically trivial cycle (blue) and a homologically non-trivial cycle (red). Note that the red chain is a cycle since the vertices in the rough boundary are ignored by the boundary operator ∂_1.



(b) The lattice Λ (dashed) and its dual lattice Λ^* (bold).

Figure 3.6: Cycles and the dual lattice.

Analogously, we define $H_1(\Lambda)$ – the set of 1-chains – as the free $\mathbb{Z}_2$ vector space generated by the edges of the lattice.
The 0-chains – $H_0(\Lambda)$ – are formal linear combinations of vertices *not lying in the rough boundaries*. Later, it will become clear why this is reasonable.
As usual, we can define 'boundary operators'

$$\partial_2 \colon H_2(\Lambda) \to H_1(\Lambda), \qquad \partial_1 \colon H_1(\Lambda) \to H_0(\Lambda)$$

mapping a plaquette to the sum of the edges surrounding it and an edge to the sum of vertices at its end (and not contained in the rough boundary).
Note that our choice to neglect vertices in the rough boundary implies that

$$\partial_1 \circ \partial_2 = 0.$$

This is true as for any formal sum

$$\sum_i \lambda_i P_i, \qquad \lambda_i \in \mathbb{Z}_2$$

any vertex of Λ lies in an even number of edges belonging to $\partial_2(\sum_i \lambda_i P_i)$. See also Figure 3.5b.

We call a chain whose boundary is 0 a *cycle*. A 1-cycle is said to be *homologically trivial* if it is in the image of ∂_2, that is, the boundary of a 2-chain. Examples are depicted in Figure 3.6a.

For later use we also need to define the *dual lattice* Λ^* of Λ. It has a vertex for each plaquette of Λ. We draw an edge between two such lattices whenever the associated plaquettes share a surrounding edge.
Also, we add horizontal edges for each vertical edge of Λ lying in the smooth boundary. These additional edges end in vertices from the rough boundary of Λ^* (See Figure 3.6b).
If Λ is a $k \times m$ lattice, Λ^* is a $(m+1) \times (k-1)$ lattice, so we can use all concepts defined before also for Λ^*. Also, we have $(\Lambda^*)^* = \Lambda$.

Now, we are ready to define and investigate the surface code. Let Λ be a $k \times m$ lattice as before. We associate with each edge of the lattice a physical qubit.
To define a stabilizer code on these qubits, we need generators for the stabilizer group. For each vertex v of the lattice (except the ones in the rough boundary) we define an operator

$$S_v = \prod_{v \in \partial_1(e)} X_e$$

where we write $v \in \partial_1(e)$ whenever v is a non-trivial summand of $\partial_1(e)$. These operators will be called 'star stabilizers'.
In the same manner, we define

$$S_P = \prod_{e \in \partial_2(P)} Z_P$$

for each plaquette P. These are the 'plaquette stabilizers'. Some examples of these stabilizers are depicted in Figure 3.7.

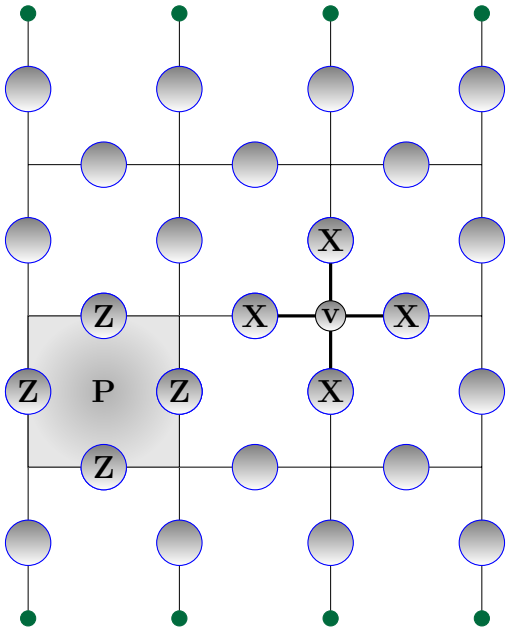


Figure 3.7: The stabilizers coming from the plaquette P and the node v. There are no stabilizers coming from the nodes at the rough boundary, which are marked in green.

We easily convince ourselves that these operators commute and are independent. Therefore, they define a stabilizer code on the physical qubits. We need to do some counting in order to figure out how many logical qubits one can encode in the surface code:
We easily see that there are $mk - 2k$ 'stars' yielding stabilizer generators. Also, we have $(k-1)(m-1)$ plaquettes and $(m-1)k + (k-1)(m-2)$ edges. Therefore, the surface code encodes

$$(m-1)k + (k-1)(m-2) - (mk - 2k) - (m-1)(k-1) = 1$$

logical qubits.
Write $n = (m-1)k + (k-1)(m-2)$ for the number of physical qubits. We can find the logical $\bar{X}$ and $\bar{Z}$ operators on the encoded qubit which generate the normalizer of the stabilizer group. As the surface code is a CSS-Code, they can be taken as elements of the n-fold Pauli group.
Let $P \in \mathcal{P}(n)$ be any Pauli operator on the physical qubits. Up to a global phase in $\{\pm 1, \pm i\}$ it is the product of an operator only consisting of X's and another only consisting of Z's:

$$P = P_X \cdot P_Z, \qquad P_X \in \{\mathbb{1}, X\}^{\otimes n}, P_Z \in \{\mathbb{1}, Z\}^{\otimes n}.$$

Clearly, P preserves the codespace if and only if P_Z commutes with all star stabilizers and P_X commutes with all plaquette stabilizers.
We associate with P_Z a 1-chain c, the sum of all edges where P_Z acts non-trivially. Then, P_Z commutes with all star stabilizers if an only if

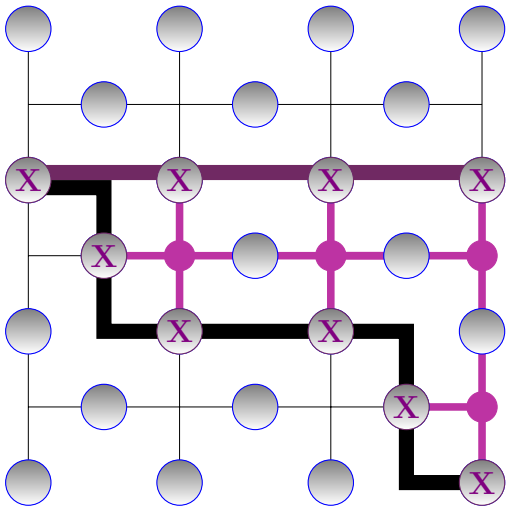
$$\partial_1(c) = 0$$

that is, c is a cycle. This follows from the fact that if v is a non-trivial summand of $\partial_1(c)$, then P_Z must anticommute with the star stabilizer at v.
If $c = \partial_2(\sum_i P_i)$ is homologically trivial, then of course

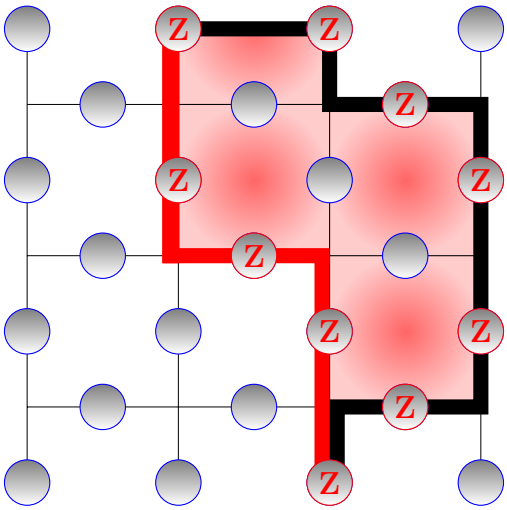
$$P_Z = \prod_i S_{P_i}$$

acts trivially on the code space. The only non-trivial cycles (modulo $\partial_2(H_2(\Lambda))$ are paths connecting both rough boundaries. Also, it is easy to see that all those paths are equivalent modulo $\partial_2(H_2(\Lambda))$.
In the stabilizer setting, this means that P_Z is either a product of plaquette stabilizers and therefore acting trivially on the codespace or a product of Z operators on qubits on a path connecting the rough boundaries times some plaquette stabilizers. Noticing that the plaquettes of the dual lattice correspond to the stars of Λ and vice versa, we see that – modulo star stabilizers – P_X is either trivial, that is corresponds to a homologically trivial cycle on the dual lattice, or a path of X operators connecting the two rough boundaries of the dual lattice.
The preceding discussion shows that we may choose $\bar{X}$ as the operator acting with X on the qubits on a path on the dual lattice connecting the rough boundaries. For $\bar{Z}$ we may take the product of Z operators on a path on the original lattice connecting the rough boundaries. See Figure 3.8 for examples.

In particular, the above discussion shows that any non-trivial logical operator is supported on at least $\min\{k, m\}$ qubits, which shows that the distance is $\min\{k, m\}$. For a square lattice, the distance therefore scales linearly in the lattice size.



(a) Candidates for $\bar{X}$. The violet operator is the black operator times the highlighted star stabilizers.



(b) Candidates for $\bar{Z}$. The red one is the black one times the highlighted plaquette operators.

Figure 3.8: The operators $\bar{X}$ and $\bar{Z}$. We give two possible candidates for each. Multiplying one of the two candidates by the stabilizer generators associates with the highlighted stars/plaquettes ends in the other candidate.

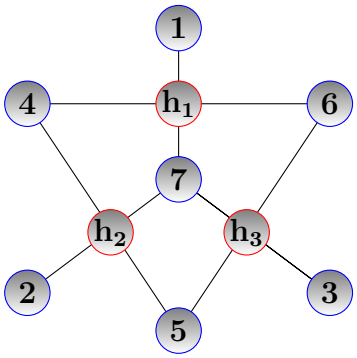
3.6 Constructing CSS-Codes from graph states

In this section we give a first taste of the power of graph states. In fact, our next goal is to see how we can construct *any* CSS-Code up to a Pauli correction from a related graph state by acting with local measurements on it. The resulting code is the desired CSS-Code up to a Pauli correction which is determined by the measurement results. This construction was introduced in [BDCPS16].

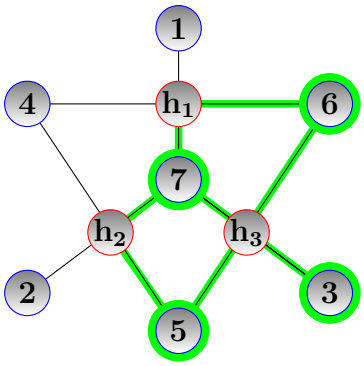
We start with a CSS-Code on n physical qubits specified by sets of Z- and X-type stabilizer generators $\mathcal{S}_Z$ and $\mathcal{S}_X$. We can think of the generators in $\mathcal{S}_Z$ as rows of a parity check matrix and construct from it the Tanner graph of the corresponding classical linear code. In other words, this bipartite graph has vertices

$$V = \{1, \dots, n\} \cup \{v_{P_Z} : P_Z \in \mathcal{S}_Z\},$$

and an edge (i, v_{P_Z}) if and only if $i \in \mathrm{supp}(P_Z)$.
We now claim that the qubits sitting on the vertices $\{1, \dots, n\}$ – whose joint state we denote by $|\phi\rangle$ – are invariant under stabilizers, which are up to a factor ± 1 equivalent to the stabilizers of the initial CSS-Code, when we measure all auxiliary qubits $\{v_{P_Z} : P_Z \in \mathcal{S}_Z\}$ in X basis.
To see this, take any Z-type stabilizer $P_Z \in \mathcal{S}_Z$. By construction of the graph, the



(a) Measuring the qubits h_1, h_2 and h_3 in X basis leaves the remaining qubits 1 – 7 in the code space of the Steane code.



(b) The Z operators of the graph state stabilizers associated with the highlighted vertices 3,5,6 and 7 cancel on each of the check vertices h_1, h_2 and h_3.

Figure 3.9: The Steane code constructed from a graph state.

stabilizer for the vertex v_{P_Z} is given by

$$M_{P_Z} = X_{P_Z}\left(\prod_{i\in \text{neigh}(v_{P_Z})} Z_i\right) = X_{P_Z}\left(\prod_{i\in \text{supp}(P_Z)} Z_i\right) = X_{P_Z}P_Z. \tag{3.10}$$

Denote the measurement outcome at v_{P_Z} by x_{P_z}. Then Equation (3.10) yields

$$x_{P_Z}P_Z\left|\phi\right\rangle = \left|\phi\right\rangle.$$

We will see later how we can erase the factors x_{P_Z} with a Pauli operator acting on the physical qubits.
Next, take any X-type stabilizer generator $P_X \in \mathcal{S}_X$. Recovering P_X from the graph stabilizers relies on the fact that P_X commutes with every Z-type stabilizer P_Z. We notice that this is equivalent to the fact that $|\text{supp}(P_X) \cap \text{supp}(P_Z)|$ is even for each $P_Z \in \mathcal{S}_Z$. This means that

$$\prod_{i\in \text{supp}(P_X)} M_i$$

has an even number of factors Z on the each vertex v_{P_Z}, which implies that these factors cancel out and we have

$$\prod_{i\in \text{supp}(P_X)} M_i = \prod_{i\in \text{supp}(P_X)} X_i = P_X$$

invariantly of the X measurements on the auxiliary qubits. Therefore we can also recover the X-type stabilizers from the graph state, which now implies that the arising state $|\phi\rangle$ is a valid code word in the code space of the CSS-Code.
To make this more intuitive we close this section by constructing both the Steane code and a surface code from graph states.

Example 3.6.1 (Steane code)**.** We start with the Steane code. Recall that it was defined on 7 physical qubits and specified by the following stabilizer generators:

$$\mathcal{S} = \langle X_1X_4X_6X_7, X_2X_4X_5X_7, X_3X_5X_6X_7,$$
$$Z_1Z_4Z_6Z_7, Z_2Z_4Z_5Z_7, Z_3Z_5Z_6Z_7\rangle.$$

By the construction, we therefore need $7 + 3 = 10$ vertices and edges according to the parity check matrix

$$H = \begin{pmatrix} 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{pmatrix}.$$

In Figure 3.9a we depict the Tanner graph. In Figure 3.9b, we can see how the Z parts of the graph state stabilizers

$$S_3 \cdot S_5 \cdot S_6 \cdot S_7$$

cancel out as they give two factors of Z on each h_1 and h_2 and four factors on h_3.

Example 3.6.2. Next, we turn to the surface code. Say, we are interested in the surface code on a 4×4 lattice:

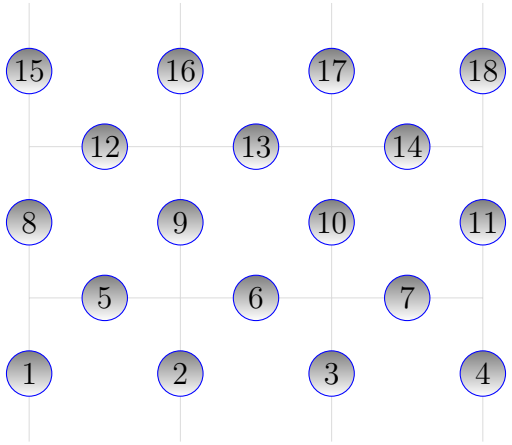


Figure 3.10

These are only the vertices for the physical qubits. We also need a vertex for each plaquette stabilizer. They need to be connected to each qubit on the edges surrounding the plaquette. If we place these vertices in the associated plaquettes, we can therefore draw the graph state planarly, see Figure 3.11.

In fact, this example is more than just a gimmick, we can easily draw an important consequence from it:

Theorem 3.6.1. We can implement the surface code up to Pauli corrections determined by the measurement outcomes from the measurements in the circuit on n qubits with a circuit using $\mathcal{O}(1)$ one and two qubit gates and single qubit measurements. We can achieve this using $\mathcal{O}(n)$ ancilla qubits.

Proof. Starting with a state $|+\rangle^{\otimes N}$, we can implement a graph state associated with a two dimensional lattice on the qubits in 4 operations as depicted here:

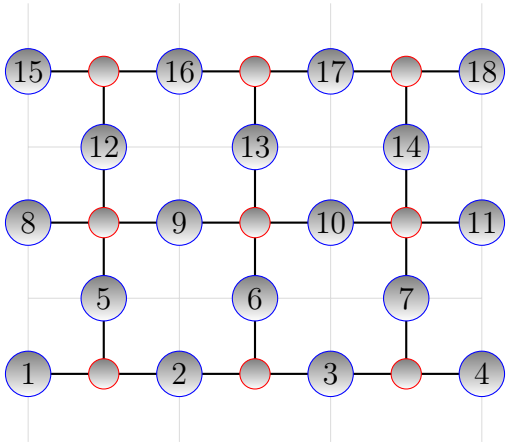
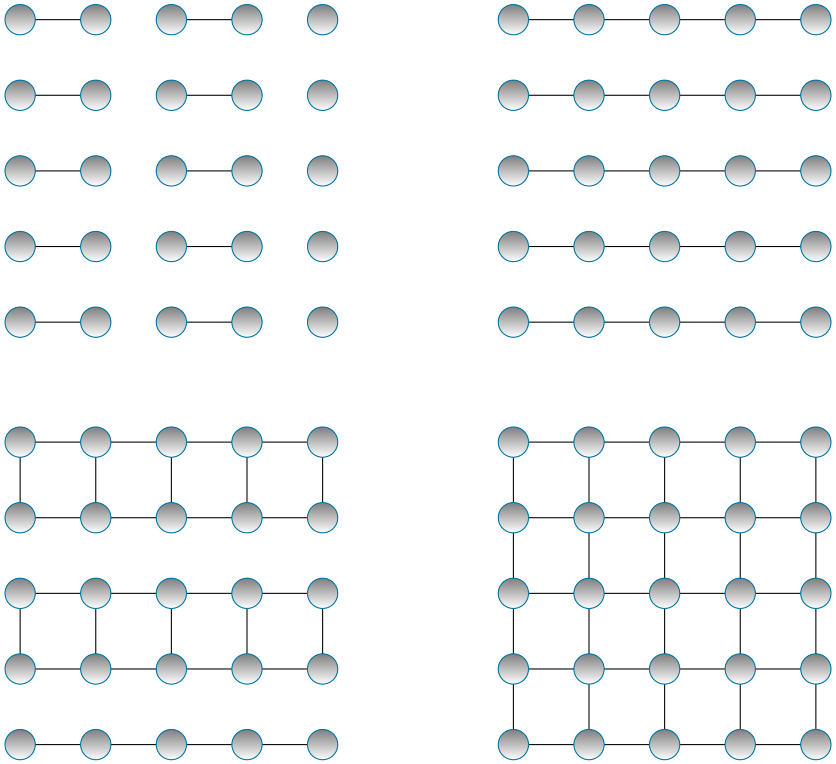
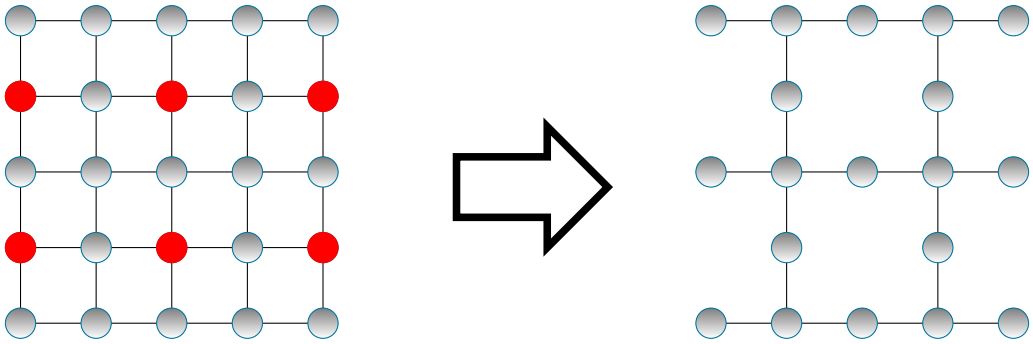


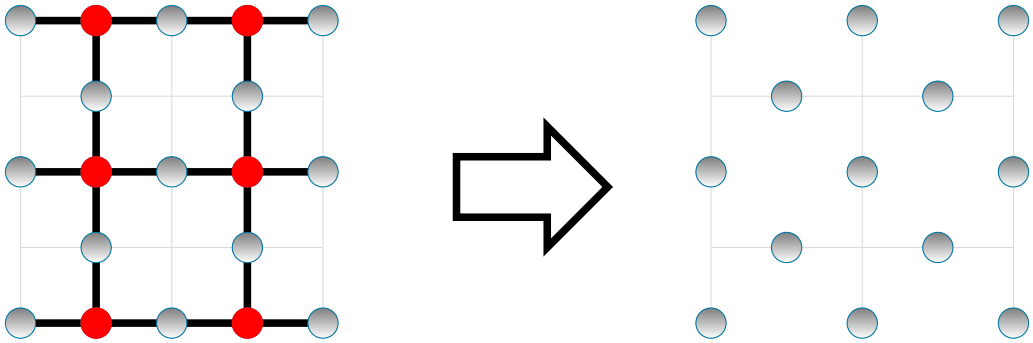
Figure 3.11: Measuring the qubits at the red vertices in X basis leaves the remaining qubits in the code space of the surface code.



Each of the four steps applies controlled Z gates to disjoint pairs of qubits, therefore this is a circuit of depth 4.
In the next step, we measure the following qubits (highlighted in red) in Z basis to delete them from the lattice:



As a last step we need to measure the qubits that will sit in the plaquettes in X basis. As we have seen previously, this leaves the remaining qubits in surface code defined on the light gray lattice.



This is an element in the desired code space. It is clear that this algorithm works for any desired size of the lattice. □

Using exactly the same argument, we can generalize Theorem 3.6.1 to CSS-Codes:

Theorem 3.6.2. Let $(n_i)_{i\in\mathbb{N}}$ be a sequence of natural numbers and C_{n_i} be instances of a code with n_i physical qubits. Moreover, let G_i be graphs such that C_{n_i} can be constructed from G_i as described previously. Denote by d_i the maximal degree of a vertex in G_i. Then, we can implement a circuit which outputs a qubit in a code which is C_{n_i} up to a Pauli correction determined by the measurement outcomes using $\mathcal{O}(d_i)$ one and two qubit gates and local measurements. In particular, this circuit is of constant depth if the sequence of d_i's is bounded.

It still remains to justify that the additional signs of the stabilizer generators introduced by the measurements do not become an obstacle. We demonstrate that they can be corrected by a Pauli operator determined by the measurement outcomes. Due to the comfortable graphical description we only explain a procedure for the surface code, the generalization to CSS-Codes is straight forward but tedious.

While the stabilizer group of a surface code is given by

$$\mathcal{S} = \langle S_P, S_v | P \text{ plaquette}, v \text{ star} \rangle,$$

the above construction will lead to a state $|\psi\rangle$ encoded in a stabilizer code given by

$$\bar{\mathcal{S}} = \langle (-1)^{a_P} S_P, (-1)^{b_v} S_v | P \text{ plaquette}, v \text{ star} \rangle,$$

where a_P and b_v are elements of $\{0, 1\}$ and depend on the measurement outcomes at the ancilla qubits.

Let $v_1, \ldots, v_r$ be the stars with $b_{v_i} = 1$. Choose paths, that is, tuples of edges, connecting v_1 with v_2, v_3 with v_4 and so forth. If r is odd, we choose a vertex w in the rough boundary and a path from v_r to w. Call these paths π_i for $i \in \{1, \ldots, r\}$ or $i \in \{1, \ldots, r-1\}$ respectively.

Now, apply to each qubit sitting on an edge of one of these paths a Z operator, that is,

$$|\psi\rangle \mapsto \underbrace{\prod_i \left(\prod_{e \in \pi_i} Z_e \right)}_{:=\mathcal{O}} |\psi\rangle .$$

As $\mathcal{O}$ commutes with all plaquette stabilizers and is self inverse, it does not change them. Also, we easily see that $\mathcal{O}$ commutes with all star stabilizers except $S_{v_1}, \ldots, S_{v_r}$. Hence, this operation changes the stabilizer to

$$\bar{\mathcal{S}} \mapsto \mathcal{O}\mathcal{S}\mathcal{O}^\dagger = \langle (-1)^{a_P} S_P, S_v | P, v \rangle$$

An analogous procedure acting with X's on edges on paths on the dual lattice deletes the signs on the plaquette stabilizers. See Figure 3.12 for an example.
We note that this procedure is in general not achievable by a classical circuit of constant depth.

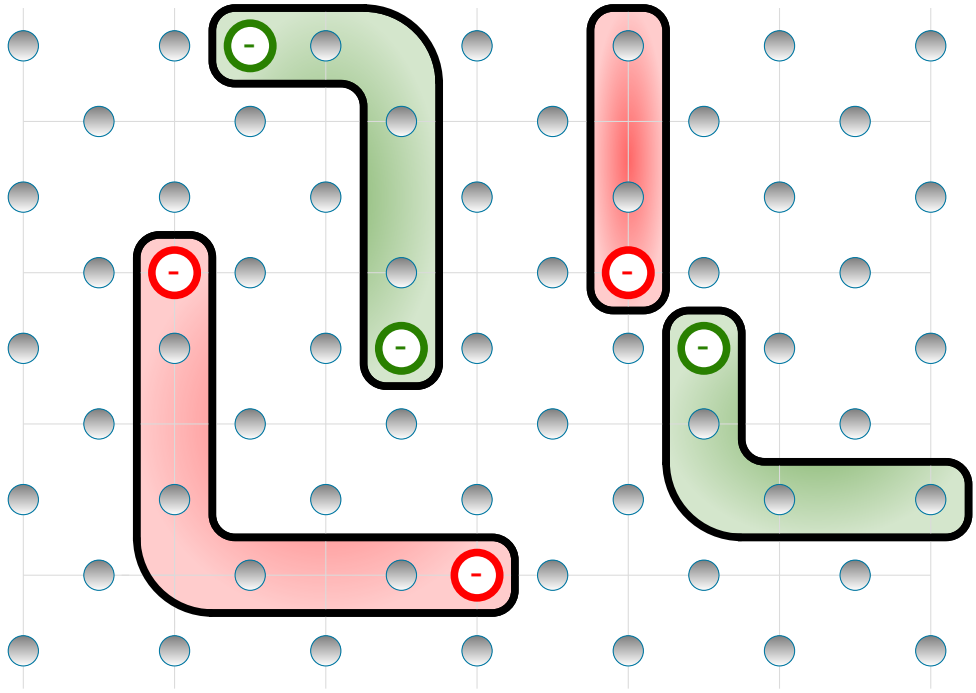


Figure 3.12: A possibility to correct signs of the stabilizers of the surface code. Say, the stabilizer generators associated with the vertices highlighted in red have an additional sign of -1. Acting with Z operators on all qubits lying on the highlighted paths will delete the sign and leave the other stabilizers unchanged. Similarly, the stabilizers for the highlighted plaquettes (green) shall have an additional sign of -1. Acting on all qubits on the paths with an X operator also will delete these signs and leave all other stabilizers unchanged.

4

Localizable long range encoded entanglement

This chapter combines the two main concepts introduced so far: Entanglement and quantum codes. In particular we are interested in the following question:
How can we create a Bell pair which is already encoded in some quantum code?
For this, we will use a strategy used in a similar way in [RBH05]. While they used a graph state defined on a three dimensional lattice, we will focus on only using gates that are local with respect to a *planar 2D geometry*.
We start by describing the strategy formally. After we have gained some intuition by seeing schemes for repetition and surface codes, we will provide a general scheme creating a Bell pair encoded in a general CSS-Code.
A generalization of [RBH05] was already done in [BDCPS16] also using 3 dimensional cluster states. Here, we present a construction where the part connecting the two codes will always be a planar 2 dimensional graph.

4.1 The strategy

We give a very informal description of the method we will use in the following paragraphs.
Suppose, we are interested in getting a Bell pair encoded in a specific CSS-Code $\mathcal{C}$ encoding 1 qubit. Let G be the graph state from which we can construct it as in paragraph 3.6. We are looking for a graph 'containing' two copies of G which are 'far from each other'. The part 'between' them is what we call the *bulk*. Call the whole graph $\bar{G}$. Schematically, this might look like Figure 4.1.

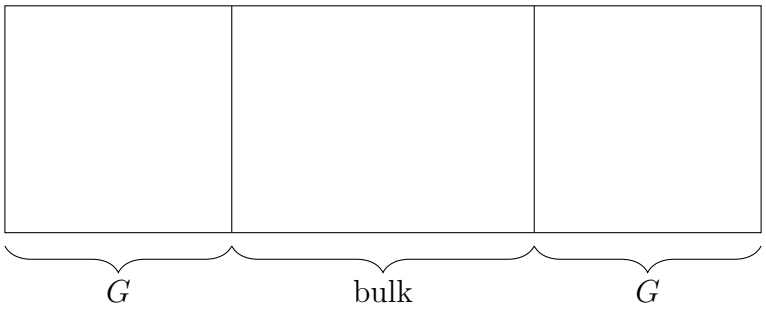


Figure 4.1: Two copies of G connected by some bulk.

The qubits are now in the graph state

$$\left|\bar{G}\right\rangle .$$

Now, we want to apply operations to the physical qubits which are local with respect to the 2D geometry given by the graph $\bar{G}$ such that the qubits are afterwards in the state

$$\left|\bar{\Phi}\right\rangle \otimes |\psi\rangle$$

where $\left|\bar{\Phi}\right\rangle$ is a Bell state encoded in the two CSS-Codes whose physical qubits are contained in the two copies of G and $|\psi\rangle$ is the state of the ancilla qubits.
It is desirable that the construction satisfies these requirements:

(i) We can 'extend' the bulk to an arbitrary size. With this, we can in principle create entanglement between parties which are arbitrarily far apart from each other.

(ii) We can implement the whole procedure with a constant depth circuit independent of the size of the bulk.

(iii) The 2D geometry should not be 'too complicated'. A possible way to formalize this is to require that the whole graph $\bar{G}$, or at least the bulk, is planar.

4.2 Examples of long range entanglement

Before we provide a general scheme for CSS-Codes in the following chapter, we want to gain some intuition by investigating the procedure for examples of codes.

4.2.1 The repetition code

Let us start with a simple example, the repetition code on three physical qubits. Recall that the code space of the repetition code can be specified by the stabilizer generators

$$Z_1Z_2, \qquad Z_2Z_3.$$

By the procedure from Chapter 3.6 we know that we obtain a $|+\rangle$ state encoded in the repetition code from a graph state on the graph $G = (V, E)$ with

$$V = \{q_1, q_2, q_3, h_1, h_2\}, \qquad E = \{\{q_1, h_1\}, \{q_2, h_1\}, \{q_2, h_2\}, \{q_3, h_2\}\}$$

by measuring h_1 and h_2 in X basis, see Figure 4.2.

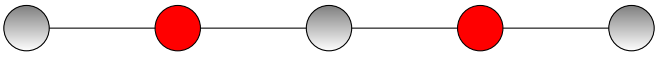


Figure 4.2: The graph for the repetition code: Measuring the red qubits in X basis leaves the remaining in repetition code.

Consider the graph $G'_k = (V, E)$ with vertices

$$V = \{q_1, q_2, q_3, q'_1, q'_2, q'_3, h_1, h_2, h'_1, h'_2, b_1, \ldots, b_k\}$$

and edges

$$\begin{aligned} E = \{&\{q_1, h_1\}, \{q_2, h_1\}, \{q_2, h_2\}, \{q_3, h_2\}, \\ &\{q'_1, h'_1\}, \{q'_2, h'_1\}, \{q'_2, h'_2\}, \{q'_3, h'_2\}, \\ &\{q_3, b_1\}, \{b_k, q'_1\}\} \cup \{\{b_i, b_{i+1}\} | i = 1, \ldots, k-1\} \end{aligned}$$

where k is an odd number. An example for this is shown in Figure 4.3.

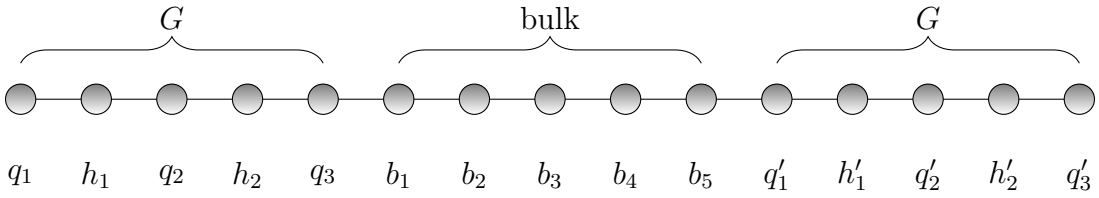


Figure 4.3: The graph G'_5.

The following claim finishes the construction:

Lemma 4.2.1. Let k be odd and G and G'_k as before. Measuring the qubits sitting at the vertices h_i, h'_i and b_i for all possible i in Pauli-X basis leaves the qubits $q_1, q_2, q_3, q'_1, q'_2$ and q'_3 in a Bell state encoded in repetition code up to a Pauli correction determined by the measurement outcomes. Moreover, the measurement outcomes determine the Bell state.

Proof. Abusing notation we also denote the measurement outcomes by h_i, h'_i and b_i for all possible i. We denote the joint state of the qubits at $q_1, q_2, q_3, q'_1, q'_2$ and q'_3 by $|\phi\rangle$.
The graph state stabilizers at h_i and h'_i respectively commute with every single measurement. Thus, we obtain

$$Z_{q_i} Z_{q_{i+1}} |\phi\rangle = h_i |\phi\rangle \text{ and } Z_{q'_i} Z_{q'_{i+1}} |\phi\rangle = h'_i |\phi\rangle .$$

If $h_1 = h_2 = h'_1 = h'_2 = 1$, we have two qubits encoded in repetition code. Else, we can simply apply a Pauli correction equalizing the signs h_i and h'_i:
Say, for example, $h_1 = h_2 = -1$ and $h'_1 = -h'_2 = -1$. We easily see that applying $X_{q_2} X_{q'_2} X_{q'_3}$ will leave the qubits at $q_1, q_2, q_3, q'_1, q'_2$ and q'_3 as physical qubits of a repetition code encoding two qubits. Similarly, we can easily find Pauli corrections for any other possible measurement outcomes h_1, h_2, h'_1 and h'_2.
It remains to show that the two qubits encoded in these repetition codes are in fact a Bell pair. We call the code on the physical qubits q_1, q_2 and q_3 the *left code*, the other one *right code*. Recall the logical $\bar{X}$ and $\bar{Z}$ operators on the repetition code:

$$\begin{aligned} \bar{X}_L &= X_{q_1} X_{q_2} X_{q_3}, & \bar{X}_R &= X_{q'_1} X_{q'_2} X_{q'_3} \\ \bar{Z}_L &= Z_{q_3}, & \bar{Z}_R &= Z_{q'_1} \end{aligned}$$

are possible representatives of the logical operators where the indices L and R indicate the left and right code respectively.
To see that the encoded state is a Bell pair, it suffices to see that it is an eigenstate of both

$$\bar{X}_L \bar{X}_R \text{ and } \bar{Z}_L \bar{Z}_R.$$

Denote the graph state stabilizer at the vertex i by S_i. We easily see that the product

$$\prod_{i=1,2,3} S_{q_i} S_{q'_i} \prod_{j \text{ even}} S_{b_j} = \prod_{i=1,2,3} X_{q_i} X_{q'_i} \prod_{j \text{ even}} X_{b_j} = \bar{X}_L \bar{X}_R \prod_{j \text{ even}} X_{b_j}$$

commutes with every X measurement as all Z factors cancel out, see Figure 4.4.

Therefore, it still stabilizes the whole state. As all qubits at positions b_j are measured in the X basis and therefore X eigenstates, this implies that – in fact – $\bar{X}_L \bar{X}_R$ has the encoded qubits as an $b_2 \cdot b_4$ eigenstate.

A similar argument works for the logical Z operators. Consider the product

$$\prod_{i \text{ odd}} S_{b_i} = Z_{q_3} Z_{q'_1} \prod_{i \text{ odd}} X_{b_i} = \bar{Z}_L \bar{Z}_R \prod_{i \text{ odd}} X_{b_i}$$

which again commutes with all performed measurements, see also Figure 4.5. By the same argument, we see that also $\bar{Z}_L \bar{Z}_R$ has $|\phi\rangle$ as a $b_1 \cdot b_3 \cdot b_5$ eigenstate.
This concludes the proof. □

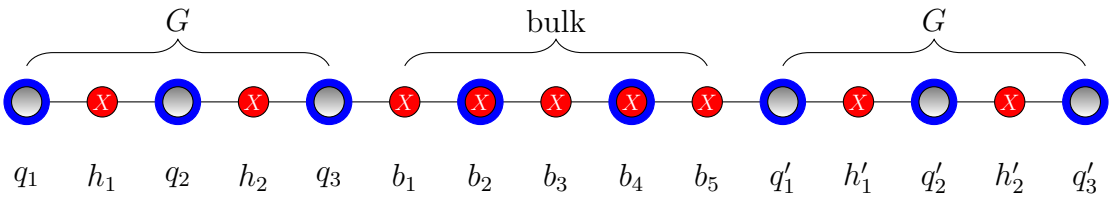


Figure 4.4: Taking the product of the graph state stabilizers at the vertices highlighted in blue all Z factors cancel out. As the bulk qubits are measured in the X basis, this product acts as $b_2 b_4 \bar{X}_L \bar{X}_R$.

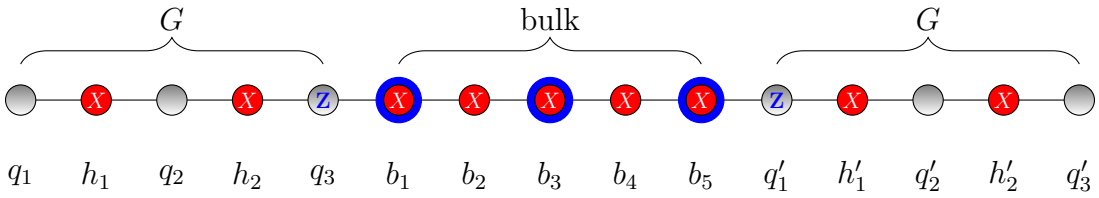


Figure 4.5: The product of the graph state stabilizers at the highlighted vertices commutes with every measurement. This acts on the system as $\pm \bar{Z}_L \bar{Z}_R$.

Hence we see that we can create a Bell state encoded in repetition code up to a Pauli correction. Clearly, this procedure satisfies all requirements stated in the preceding paragraph:

(i) We can add an arbitrary (but odd) number of qubits between the two codes and therefore make the spatial distance as large as we want.

(ii) We can implement the graph state on this one dimensional lattice using a constant depth circuit independent of its length.

(iii) This is not only a planar 2D graph, but a simple 1 dimensional graph.

4.2.2 The surface code

We can also find an analogous scheme to prepare a Bell state encoded in the surface code. For this, we consider a $(2k+l) \times k$ lattice where both k and l are odd. It has vertices

$$V = \{(i,j) | 1 \leq i \leq (2k+l), 1 \leq j \leq k\},$$

horizontal edges of the form $\{(i,j),(i+1,j)\}$, and vertical edges $\{(i,j),(i,j+1)\}$ for $i \leq 2k+l-1$ and $j \leq k-q$ respectively. We call this lattice Λ_{tot}.
For our scheme, we do not need the whole lattice. Let us delete the vertices (and all incident edges) (i,j) for i odd and j even. An example of the resulting graph can be seen in Figure 4.6, we will refer to this 'lattice' as Λ.

As indicated in Figure 4.6, the qubits

$$\{(i,j) | 1 \leq i \leq k\} \text{ and } \{(i,j) | k+l+1 \leq i \leq 2k+l\}$$

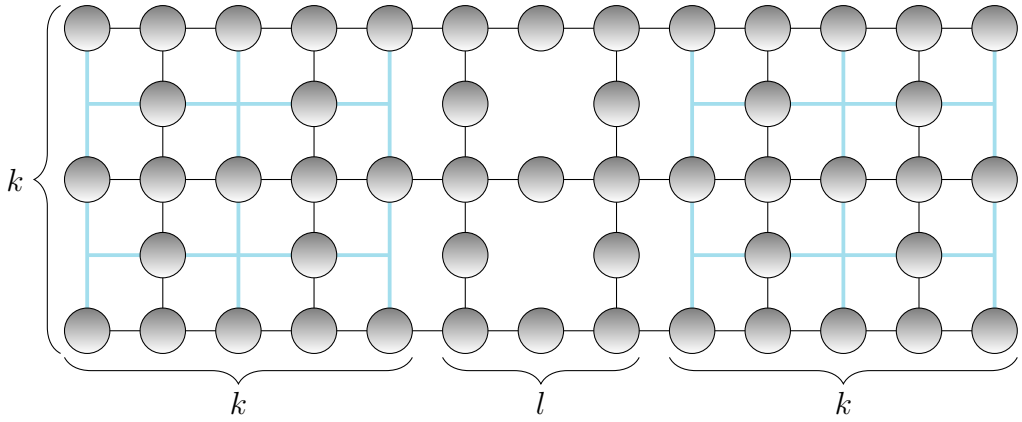


Figure 4.6: A 13×5-lattice where we have deleted vertices of the form (i, j) for i odd and j even. The light blue lattices Λ_1 and Λ_2 indicate where both the surface codes will be. Notice that the vertices from $\Lambda_{\text{tot}} \setminus \Lambda$ are the vertices of the lattices Λ_1 and Λ_2 together with some deleted vertices from the boundary.

will be used to construct surface codes on a subset of qubits, the part in between will be the bulk.
We also mention that the graph state in this lattice can be implemented with a constant depth circuit similar as in the proof of Theorem 3.6.1.
Again, it is now enough to measure all qubits that will not be physical qubits of one of the two surface codes in the X basis. The measurement pattern thus looks like in Figure 4.7.

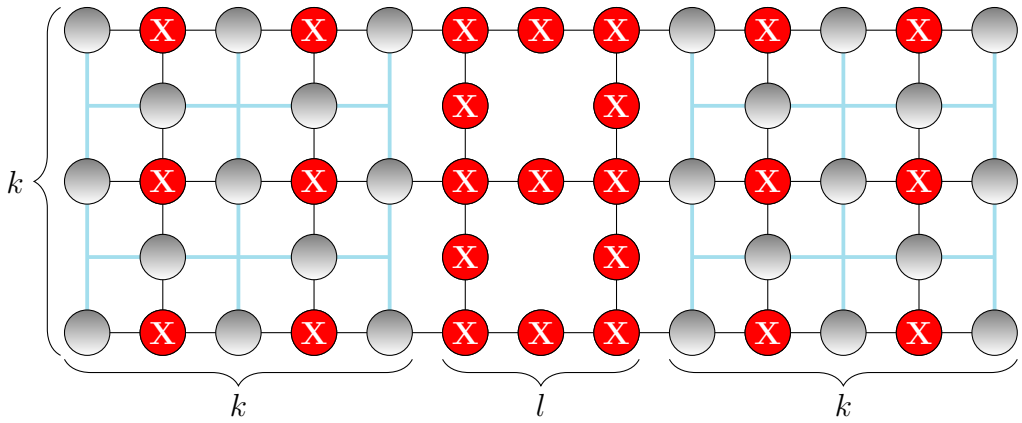


Figure 4.7: The measurement pattern.

Formally, we measure all qubits in the bulk as well as the qubits in the set M defined as

$$\begin{aligned} M_1 =& \{(i,j) | 1 \leq i \leq k, 1 \leq j \leq k, i \text{ even }, j \text{ odd}\}, \\ M_2 =& \{(i,j) | k+l \leq i \leq 2k+l, 1 \leq j \leq k, i \text{ even }, j \text{ odd}\}, \\ M =& M_1 \cup M_2. \end{aligned}$$

Also define the subsets of $\Lambda_{\text{tot}} \setminus \Lambda$

$$N_1 = \{(i,j) | 1 \leq i \leq k, 1 \leq j \leq k, i \text{ odd }, j \text{ even}\},$$
$$N_2 = \{(i,j) | k+l \leq i \leq 2k+l, 1 \leq j \leq k, i \text{ odd }, j \text{ even}\}.$$

These are the stars of the lattices which define our surface codes.
We can define lattices Λ_1 and Λ_2 on the vertex sets N_1 and N_2 by drawing edges between vertices which had distance 2 in the original lattice Λ_{tot}, see Figure 4.7 for an example. We notice that the edges of these lattices correspond to the unmeasured qubits. Also, in each plaquette of the lattices Λ_1 and Λ_2, there is a qubit connected (in Λ) to the qubits on the edges surrounding the plaquette which is measured in X basis.
We claim that the unmeasured qubits encode a Bell pair in two surface codes on Λ_1 and Λ_2 up to a Pauli correction determined by the measurement outcomes. We denote the joint state of the unmeasured qubits $|\phi\rangle$.
As with the repetition code we first need to reconstruct all the code stabilizers. Recall that the stabilizer group of the surface code is generated by plaquette and star stabilizers. For this, we again denote the graph state stabilizer at vertex (i,j) of Λ as $S_{(i,j)}$. Moreover, we denote the measurement outcomes $m_{(i,j)}$ for all measured qubits (i,j). For a visualization of the following arguments see Figure 4.8.
We start with the plaquette stabilizers. Let P be a plaquette of Λ_1 which corresponds to a vertex (i,j) in M_1. We see that the corresponding graph state stabilizer $S_{(i,j)}$ commutes with every single measurement. Together with the fact that the qubit (i,j) is measured in the X basis, we see that

$$m_{(i,j)} Z_{(i-1,j)} Z_{(i+1,j)} Z_{(i,j-1)} Z_{(i,j+1)} |\phi\rangle = |\phi\rangle . \tag{4.1}$$

As described in Section 3.6, we can find a Pauli correction transforming the stabilizers in (4.1) into proper plaquette stabilizers. Of course, the same argument works for Λ_2.
Let us now construct the star stabilizers. For this, let $s = (i,j)$ be a star, without loss we can assume that $1 \leq i, j \leq k$, that is, s is a star of Λ_1. If s is as vertex of Λ_{tot} not connected to a qubit in the bulk, in other words, $j \neq k$, the argument from example 3.6.2 works: Taking the product of the graph state stabilizers at the qubits surrounding s, all Z factors cancel out and we obtain the star stabilizer.
Now let $s = (k,j)$. As the vertex lies in the smooth boundary of Λ_1, the star stabilizer has the form

$$X_{k-1,j} X_{k,j+1} X_{k,j-1}.$$

We notice that the product of graph state stabilizer generators

$$S_{(k-1,j)} S_{(k,j+1)} S_{(k,j-1)} S_{(k+1,j)} = X_{k-1,j} X_{k,j+1} X_{k,j-1} X_{k+1,j}$$

commutes with every X measurement as the Z factors cancel out. Also, as the qubit at the vertex at $(k+1,j)$ is measured in X basis, this implies, that the $|\phi\rangle$ is a $m_{(k+1,j)} = \pm 1$ eigenstate of the star stabilizer. Again, we can transform this to proper star stabilizers using Pauli corrections as in Section 3.6.

Therefore, the unmeasured qubits encode a logical two qubit state. Again, denote by $\bar{Z}_L, \bar{Z}_R, \bar{X}_L$ and $\bar{X}_R$ the logical X and Z operators on the code spaces of the

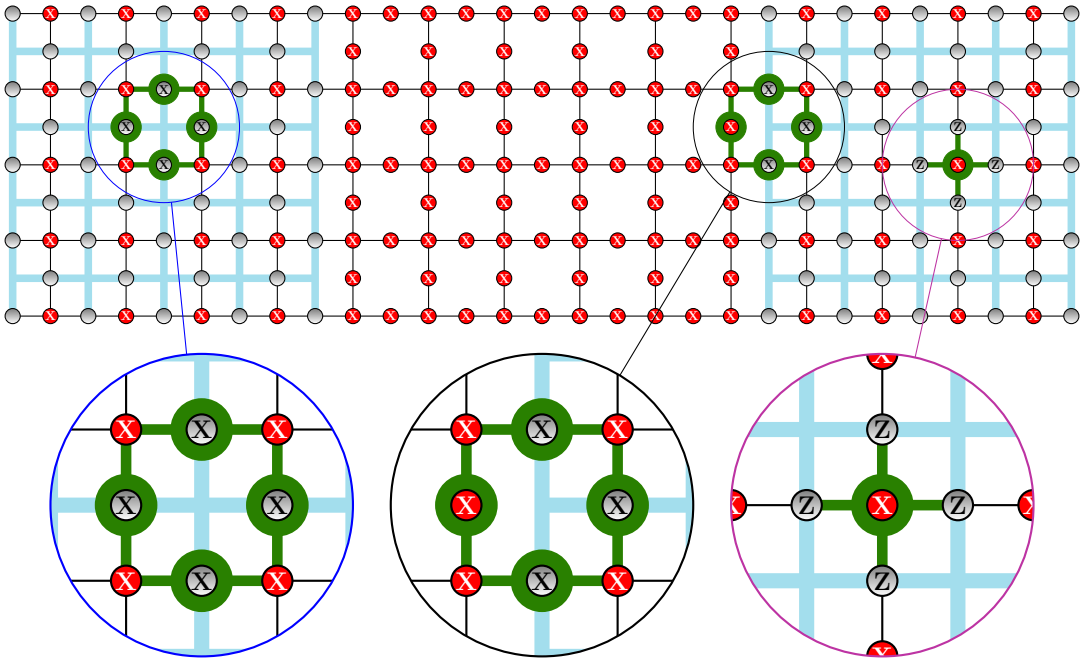


Figure 4.8: The stabilizer of the surface codes on Λ_1 and Λ_2. (a) The blue lens shows how a star stabilizer arises. Taking the product of the graph state stabilizers at the highlighted vertices, we end up with the corresponding star stabilizer. (b) In the black lens, also a star stabilizer is constructed, but now using a graph state stabilizer at a vertex in the bulk. (c) The purple lens magnifies a plaquette. The graph state stabilizer of the vertex sitting in the plaquette together with the X measurement yields the plaquette stabilizer.

surface code on Λ_1 and Λ_2 respectively. We will again argue that the encoded state is a Bell pair by showing that it is an eigenstate of both $\bar{Z}_L\bar{Z}_R$ and $\bar{X}_L\bar{X}_R$.
We start with the logical Z operators. Consider the product of graph state stabilizers of the vertices

$$V_Z = \{(i,j) | k+1 \leq i \leq k+l-1, 1 \leq j \leq k, i \text{ even}, j \text{ odd}\}.$$

An example can be seen in Figure 4.9. It is not hard to see that the following holds as again most of the Z factors occur twice and therefore cancel out:

$$\prod_{(i,j)\in V_Z} S_{(i,j)} = \bar{Z}_L\bar{Z}_R \prod_{(i,j)\in V_Z} X_{(i,j)}. \tag{4.2}$$

Here, we choose the representations

$$\bar{Z}_L = \prod_{j \text{ odd}} Z_{k,j}, \qquad \bar{Z}_R = \prod_{j \text{ odd}} Z_{k-l,j}.$$

After measuring the bulk qubits in X basis, this implies indeed that the encoded qubits are a

$$\pm 1 = \prod_{(i,j)\in V_Z} m_{(i,j)}$$

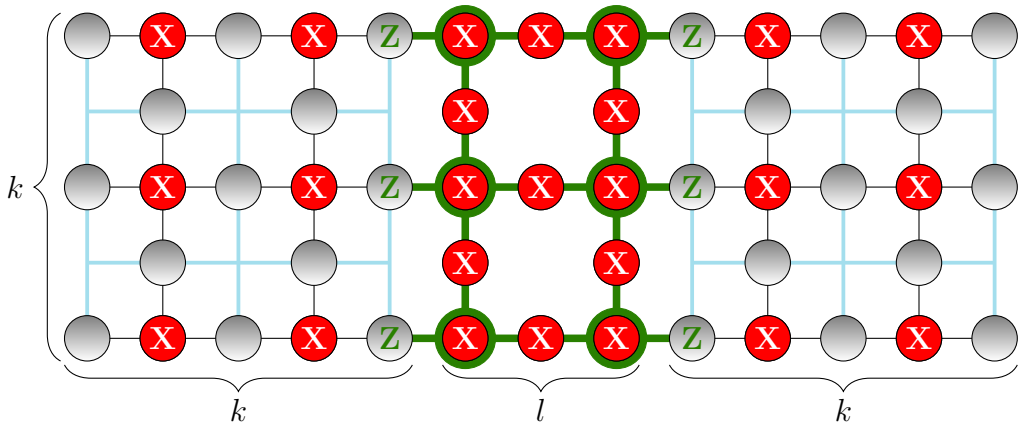


Figure 4.9: The product of the graph state stabilizers at the highlighted vertices commutes with every single qubit measurement. After the measurement, it acts as $\pm\bar{Z}_L\bar{Z}_R$ where the sign is determined by the measurement outcomes in the bulk.

eigenstate of $\bar{Z}_L\bar{Z}_R$.

For the logical X operators, we consider the set of vertices of Λ

$$V_X = \{(i,1)|1 \leq i \leq 2k+l, i \text{ odd}\}$$

and the subset of V_X of qubits that are in the bulk, $V_{X,\text{bulk}}$.
Again, we easily see that

$$\prod_{(i,j)\in V_X} S_{(i,j)} = \bar{X}_L\bar{X}_R \prod_{(i,j)\in V_{X,\text{bulk}}} X_{(i,j)},$$

see Figure 4.10 for a visualization. This obviously commutes with all measurements and therefore still stabilizes the whole state. As all qubits in the bulk are measured in the X basis, this shows that the encoded qubits are a

$$\pm 1 = \prod_{(i,j)\in V_{X,\text{bulk}}} m_{(i,j)}$$

eigenstate of $\bar{X}_L\bar{X}_R$.

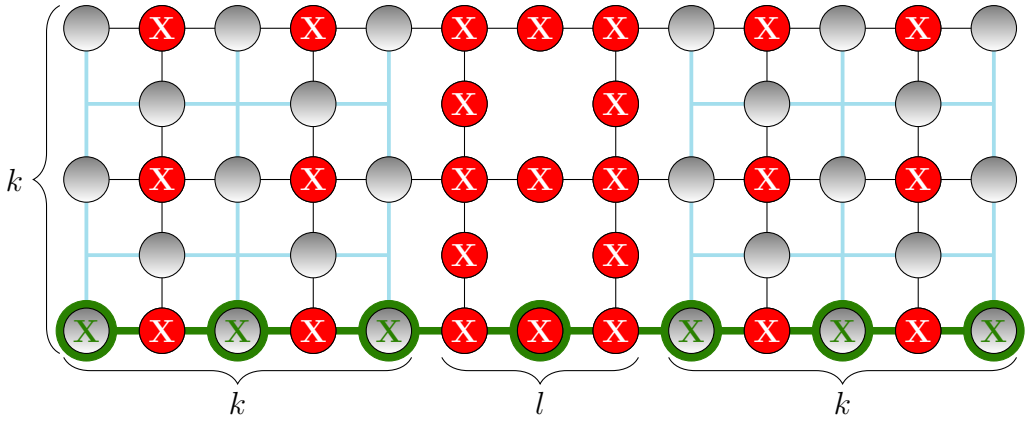


Figure 4.10: The product of the graph state stabilizers at the highlighted vertices commutes with every measurement. After the measurements are performed, it acts as $\pm\bar{X}_L\bar{X}_R$ where the sign is determined by the measurement outcomes.

4.3 Long range entanglement for CSS-Codes

It turns out that we can generalize this construction. Let $\mathcal{C}$ be an $[n, 1]$ CSS-Code and denote $\mathcal{C}_Z$ the classical linear code corresponding to the Z type stabilizers of $\mathcal{C}$, see Section 3.3.5.
We place N qubits in $|+\rangle$ state on the left- and righthand side each, where N is the number of vertices of the Tanner graph of $\mathcal{C}_Z$ as defined in Section 3.1.2.

$$|+\rangle^{\otimes N} \qquad\qquad |+\rangle^{\otimes N}$$

From both sets of qubits, we choose n qubits which shall represent the physical qubits of the code, the others are the check qubits and will be measured later on. By using Hadamard and controlled Z gates, we can implement a graph state corresponding to the Tanner graph of $\mathcal{C}_Z$ on both sides as in Section 3.6. Denote the state of these physical qubits on both the left and the right side by

$$|\phi_{LR}\rangle \in \left(\mathbb{C}^2\right)^{2n}.$$

Also, choose some odd number $m \in \mathbb{N}$.
As we have seen before, the logical $\bar{Z}$-operator of the CSS-Code is given by a product of Z-operators on the physical qubits. Let M_L and M_R be sets of qubits such that acting with a Pauli Z on the qubits from M_L respectively M_R implements the logical Z-operator on the left respectively right code.

- *Step 1:* Place layers of $|M_L| = |M_R|$ qubits in state $|+\rangle$ 'next to' the left and the right code each arranged on a vertical line. We call the set of qubits next to the left code L_1 (the first layer) and the set of qubits next to the right code L_m (the m-th layer). See Figure 4.11.

- *Step 2:* Choose bijections

$$f_L \colon M_L \to L_1, \qquad f_R \colon M_R \to L_m$$

and perform controlled Z gates on all pairs $(i, f_L(i))$, $(j, f_R(j))$ for all $i \in M_L$ and $j \in M_R$.
After this, the whole system of qubits is in a graph state, every vertex in M_L respectively M_R is adjacent to exactly one qubit in L_1 respectively L_m. See Figure 4.13.

- *Step 3:* Add $m-2$ layers of qubits in $|+\rangle$ state, where a layer consists of $|M_L|$ qubits arranged on a vertical ray. We also denote these layers by $L_2, \ldots, L_{m-1}$. Again using controlled Z gates, we connect the layers with edges such that the j-th qubit in layer i is connected to the j-th qubit in layer $i+1$ for all $j = 1, \ldots, |M_L|$ and all $i = 1, \ldots, m-1$. See Figure 4.12.
- *Step 4:* Between two neighbouring qubits in each of the layers L_1 and L_m – here, neighbouring is meant with respect to the arrangement on the vertical ray – we place another qubit and connect it using two controlled Z gates with the 'upper' and the 'lower' qubit. See Figure 4.14.
- *Step 5:* Measure all the qubits in the bulk as well as the check qubits on the left- and righthand side in X basis.

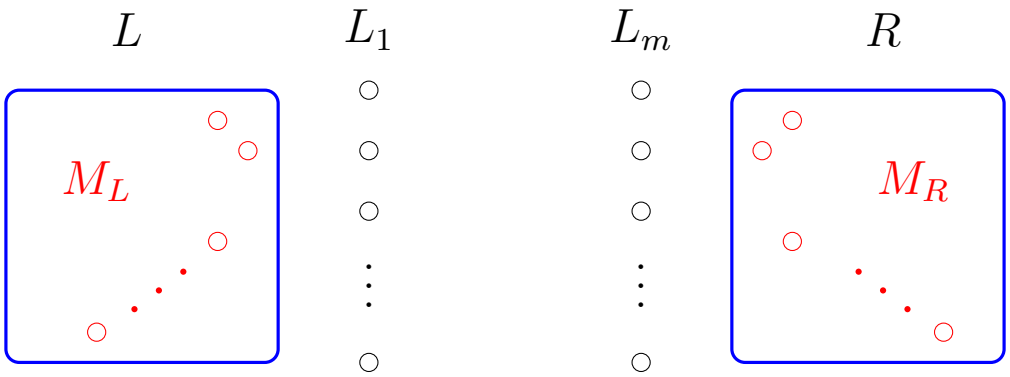


Figure 4.11: Step 1. We consider the boxes as some Tanner graphs of a CSS-Code. Applying Z operators to the red qubits implements a logical Z operator on the CSS-Code.

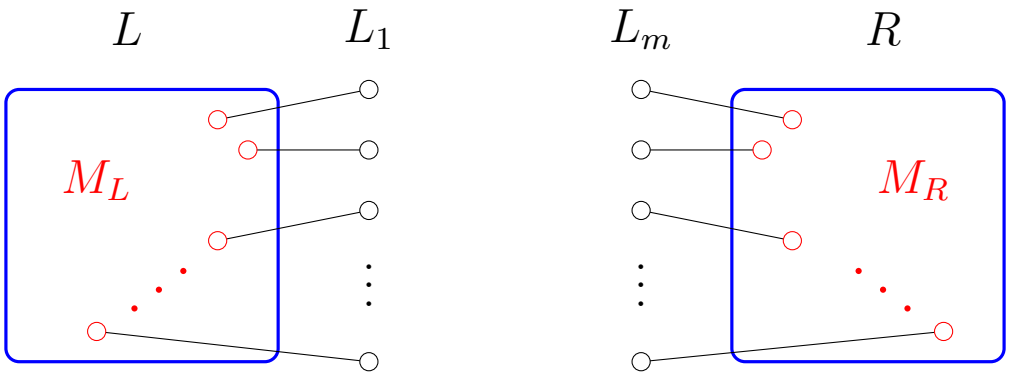


Figure 4.12: Step 2.

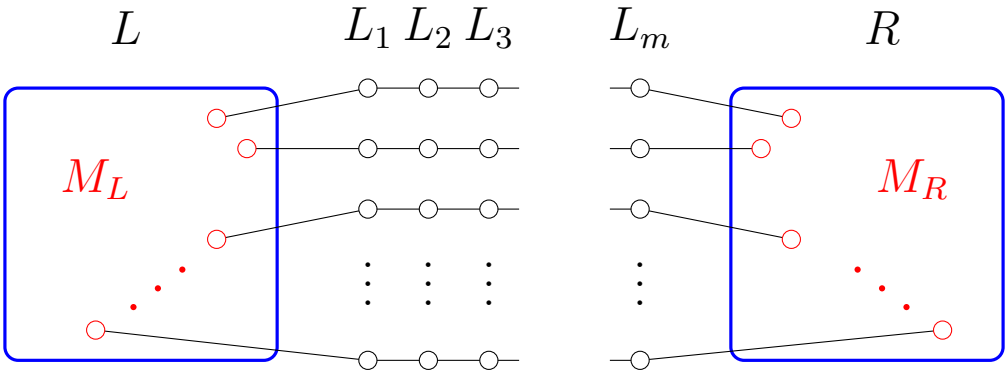


Figure 4.13: Step 3.

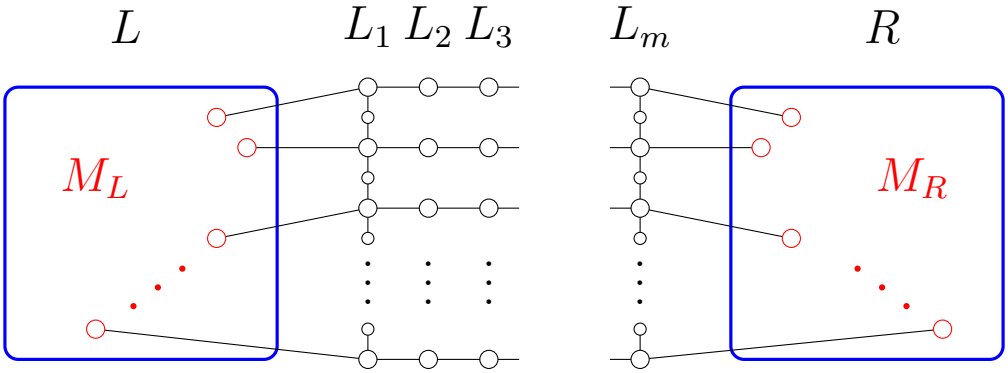


Figure 4.14: Step 4.

Theorem 4.3.1. Applying the above procedure leaves the unmeasured qubits as ± 1-eigenstates of the stabilizers of the left and the right CSS code. Moreover, the remaining qubits obey

$$\bar{X}_L \bar{X}_R \left|\phi_{LR}\right\rangle = \pm \left|\phi_L R\right\rangle , \qquad \bar{Z}_L \bar{Z}_R \left|\phi_{LR}\right\rangle = \pm \left|\phi_{LR}\right\rangle ,$$

where $\bar{X}_{L/R}$ and $\bar{Z}_{L/R}$ denote the logical operators on the corresponding code.

Proof. We start with the stabilizers of the codes:

1. Z-type stabilizers:
   This follows by the exact same argument as before: All neighbours of check vertices are unmeasured qubits. Therefore, the graph state stabilizers associated with these commute with every single qubit measurement. These stabilizers together with the X measurements at the check vertices yield exactly the Z-type stabilizers.

2. X-type stabilizers:
   Here, the qubits added in step four of the construction become crucial. As every consideration in the following holds for both codes, we only deal with the code on the left hand side.
   Let $S \subset \{1, \ldots, n\}$ be a subset such that

$$\prod_{i \in S} X_i$$

is the X-type stabilizer we want to reproduce.
As in Section 3.6, we notice that all Z factors coming from the product

$$\prod_{i \in S} G_i \tag{4.3}$$

of graph stabilizers cancel out on the check vertices. But now the additional qubits introduce a new challenge: The vertices from S may be adjacent to vertices in the first layer L_1, so the product in 4.3 does not necessarily commute with the X measurements there.
We know that $|M_L \cap S|$ is even, where M_L is the set of vertices adjacent to vertices in the first layer, in other words the support of a realization of the logical $\bar{Z}$ operator on the code space. This follows easily from the fact that this X type stabilizer has to commute with the logical operator.
This implies that the number of vertices in the first layer L_1 adjacent to some vertex in S is even.
Denote the vertices in the first layer $\{l_1, l_{1.5}, l_2, l_{2.5}, \ldots, l_{(k-1)+0.5}, l_k\}$ where we order them from the top to the bottom. The l_i's with integer indices are therefore the vertices added in step 1, half-integer indexed vertices the ones added in step 4. The set $\mathcal{A}$ of vertices in the first layer adjacent to a vertex from our X-type stabilizer has an even number of elements and consists of l_i's with integer-valued indices. Hence, we may split them into disjoint pairs

$$\mathcal{A} = \{l_{i_1}, l_{i_2}\} \cup \{l_{i_3}, l_{i_4}\} \cup \cdots \cup \{l_{i_{|\mathcal{A}|-1}}, l_{i_{|\mathcal{A}|}}\},$$

where $i_1 < i_2 < \cdots < i_{|\mathcal{A}|}$.
Now, consider the following product of graph state stabilizers:

$$\begin{aligned}\left(\prod_{i \in S} G_i\right) &\cdot G_{i_1+0.5} \cdot G_{i_1+1.5} \ldots G_{(i_2-1)+0.5} \\ &\cdot G_{i_3+0.5} \ldots G_{(i_4-1)+0.5} \ldots G_{i_{|\mathcal{A}|}-0.5}\end{aligned} \tag{4.4}$$

We can see that the term from Equation (4.3) and the remaining factors of (4.4) give factors Z on the exact same qubits. Therefore, these factors cancel out and Equation (4.4) becomes

$$\begin{aligned}\left(\prod_{i \in S} X_i\right) &\cdot X_{i_1+0.5} \cdot X_{i_1+1.5} \ldots X_{(i_2-1)+0.5} \\ &\cdot X_{i_3+0.5} \ldots X_{(i_4-1)+0.5} \ldots X_{i_{|\mathcal{A}|}-0.5},\end{aligned} \tag{4.5}$$

which certainly commutes with every single qubit measurement performed in the construction.
As we measure all qubits in the bulk in the X basis, Equation (4.5) translates to

$$\prod_{i \in S} X_i \left|\phi_{LR}\right\rangle = \pm \left|\phi_{LR}\right\rangle$$

where the sign depending on the measurement outcomes in the bulk is given by the product of the measurement outcomes at the qubits

$$i_1 + 0.5, i_1 + 1.5, \ldots, (i_2 - 1) + 0.5, i_3 + 0.5, \ldots, (i_4 - 1) + 0.5, \ldots, i_{|\mathcal{A}|} - 0.5.$$

3. $\bar{X}_L\bar{X}_R\,|\phi_{LR}\rangle = \pm\,|\phi_{LR}\rangle$:
This argument is similar to the one we used to create localizable long range entanglement encoded in repetition code. Let M_L^X and M_R^X be the sets of qubits in the left and right code on which we need to act with Pauli X's in order to implement the logical X operators. Taking the product of graph stabilizers

$$\prod_{i\in M_L^X} G_i \prod_{j\in M_R^X} G_j$$

may introduce Z factors on L_1 and L_m – following the notation from the last argument – on qubits with integer indices. As the constructions were analogous, the Z factors appear in L_1 and L_m both at the same heights $\{i_1,\ldots,i_p\}$. If we multiply the graph state stabilizers associated with the vertices at these heights in the layers $L_2,\ldots,L_{m-1}$, we again see that all Z factors cancel out and we have

$$\bar{X}_L\bar{X}_R\,|\phi_{LR}\rangle = \pm\,|\phi_{LR}\rangle$$

depending on the measurement outcomes in the bulk.

4. $\bar{Z}_L\bar{Z}_R\,|\phi_{LR}\rangle = \pm\,|\phi_{LR}\rangle$:
This is again quite similar to the repetition code example. Consider the product

$$\prod_{i=1,3,5,\ldots,m} \left(\prod_{j\in L_i} G_j\right).$$

We easily see that all Z terms cancel except the ones on the sets M_L and M_R. After the X measurements in the bulk are performed, this results in

$$\bar{Z}_L\bar{Z}_R\,|\phi_{LR}\rangle = \pm\,|\phi_{LR}\rangle\,,$$

depending on the measurement outcomes in the bulk. This concludes the proof.

□

Example 4.3.1. Let us finish this chapter by seeing yet another example for localizable long range encoded entanglement, this time for two copies of the Steane code. By Theorem 4.3.1, a suitable measurement pattern looks like this:

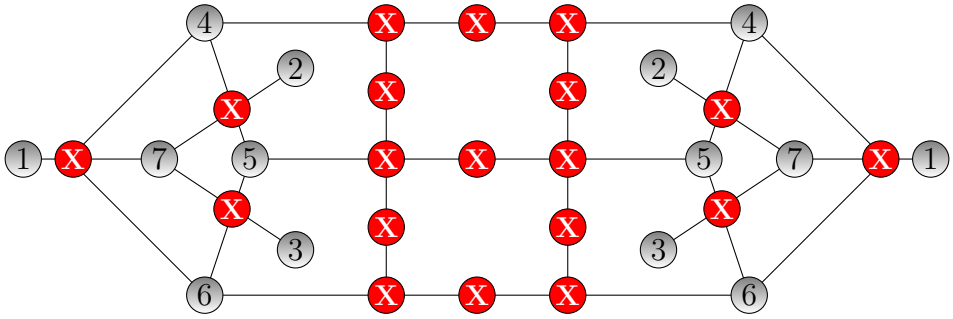


Figure 4.15: The measuring pattern for two copies of the Steane code.

Remark. We easily convince ourselves that we can add additional qubits in the layers $L_3, L_5, \ldots, L_{m-1}$ similar to the qubits with half-integer index in L_1 and L_m and still generate an encoded Bell pair. With this, we get for any CSS-Code correlations between the measurement outcomes as described in the next chapter for the surface code. We already did this in the introductory example of the toric code, see for example Figure 4.8.

5

Error model and fault tolerance

The procedure from Chapter 4 seems to provide a way of creating a Bell-pair which is stable against certain errors: As the Bell-pair is already encoded we can apply some error correcting procedure. If we choose a suitable code, this should protect our Bell-pair.
But so far we have not taken care of the following: Errors on physical qubits are not the only kind of problems we have to face. We should be prepared for the following obstacles:

(i) Gates can only be implemented up to some precision. We call gates that might be faulty 'noisy gates'.

(ii) Measurements might also be faulty. On the one hand, the measurement might not collapse the state to what we would expect from the outcome. Also, we might just receive the wrong outcome from our 'noisy measurement'.

(iii) Of course there might happen errors on the qubits at each time.

In the following, we will start by thinking about how a suitable error model might look like. Afterwards, we investigate how our scheme for localizable long range encoded entanglement performs with respect to errors.

5.1 Error models

5.1.1 Classical p-local stochastic noise

We start by considering an error model on classical information. Say, we want to store, process and read out an n-bit string. We define an error E as an element of $\{0,1\}^n$.
When we initially wanted to store the bit string $x \in \{0,1\}^n$ and the error E happens, our bits are afterwards in the configuration

$$x \oplus E$$

where $\oplus$ denotes the bitwise addition modulo 2.
The simplest non-trivial error model is the following: In each time step, each bit is flipped with probability $p \in (0,1)$.
This model assumes that errors on the bits are independent and that it is equally likely for each bit to suffer from an error.
Choose a subset $\mathcal{I} \subset \{1,\dots,n\}$. The probability that the error flips the bits in $\mathcal{I}$ is

$$\begin{aligned}
\mathrm{p}(\text{"error on "}\mathcal{I}) &= \left(\sum_{i=1}^{n-|\mathcal{I}|} \binom{n-|\mathcal{I}|}{i} (1-p)^i p^{n-i}\right) \\
&= \left(\sum_{i=1}^{n-|\mathcal{I}|} \binom{n-|\mathcal{I}|}{i} (1-p)^i p^{n-|\mathcal{I}|-i}\right) p^{|\mathcal{I}|} \\
&= p^{|\mathcal{I}|}(1-p)^{n-|\mathcal{I}|}.
\end{aligned}$$

Of course, it would be too restrictive to work with this error model only. The following slightly generalizes this simple model and also covers the possibility of some local correlations between errors.

Definition 5.1.1. Let an error E be drawn from some probability distribution on $\{0,1\}^n$. We call the error *classical p-local stochastic noise* if we have

$$\mathrm{p}(\mathcal{I} \subset \mathrm{supp}(E)) \leq p^{|\mathcal{I}|}$$

for all $\mathcal{I} \subset \{1,\dots,n\}$.

What about errors when processing the bits?
Say we want to process our bit string x to another bit string x' only using gates which can be applied all in the same time step. If something in this procedure goes wrong, we end up with the bit string $x' + E$ for some error E. Hence a noisy gate is equivalent to a perfect gate followed by an error E.
The same holds for reading out information: Say, after we have processed our information, we want to know the bit at the i-th position. It may happen, that our device does not tell us the actual value x_i but $x_i \oplus 1$. Again, this is equivalent to a correct measurement followed by the error flipping the i-th bit.
Therefore, we have developed the following error model: In each time step we may apply operations on the bits which can be carried out at once. After each time step, we apply some error to the bits which is drawn from some distribution.

5.1.2 Noisy quantum circuits

For quantum codes, we restrict to errors in $\{\mathbb{1}, X, Y, Z\}^{\otimes n}$. With the same motivation as with classical codes, we define *p-local stochastic noise* as in [BGKT19].

Definition 5.1.2. Let $\mathcal{E}$ drawn from some distribution on $\{\mathbb{1}, X, Y, Z\}^{\otimes n}$ be a Pauli error on n qubits and $p \in [0, 1]$. The error $\mathcal{E}$ is called *p-local stochastic noise* if for all $F \subset \{1, \ldots, n\}$ we have

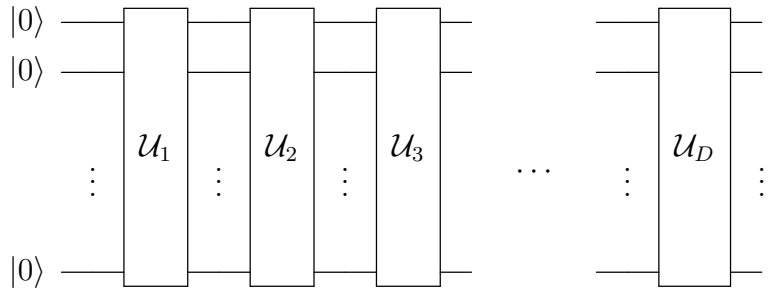
$$P\left(F \subset \mathrm{Supp}(\mathcal{E})\right) \leq p^{|F|} \tag{5.1}$$

where $\mathrm{Supp}(\mathcal{E})$ is the number of components of $\mathcal{E}$ which are not the identity.
We write $\mathcal{E} \sim \mathcal{N}(p)$ whenever $\mathcal{E}$ satisfies Equation (5.1).

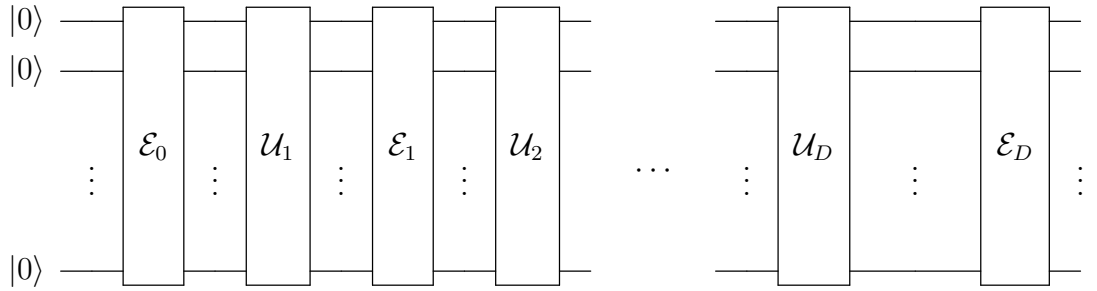
Let $\mathcal{G}$ be some set of quantum gates, for example, the Clifford+T gate set introduced in example 2.1.1. Recall that a circuit of depth S over $\mathcal{G}$ is of the form

$$\mathcal{U} = \mathcal{U}_D \cdot \cdots \cdot \mathcal{U}_1$$

where each $\mathcal{U}_i$ is a product of gates from $\mathcal{G}$ acting on disjoint sets of qubits. This ensures that we can apply the unitary $\mathcal{U}_i$ in one time step. This implementation therefore enables us to apply the unitary $\mathcal{U}$ in D time steps.



We again notice that a noisy implementation of the gate $\mathcal{U}_i$ is equivalent to a perfect implementation followed by an error. Moreover, we assume that there is an error before $\mathcal{U}_1$ corresponding to a 'noisy preparation' of the input $|0\rangle^{\otimes n}$.
Hence, a noisy implementation of $\mathcal{U}$ might look like



where the $\mathcal{E}_i$ are Pauli errors.

What about noisy measurements? For simplicity, we restrict to measuring single qubits in the computational basis. Say, we measure the i-th qubit of the system in

state $|\psi\rangle$ in Z basis. Depending on the measurement outcome ± 1, this will collapse the state to

$$\mathbb{1}^{i-1} \otimes (\mathbb{1} \pm Z) \otimes \mathbb{1}^{n-i}.$$

Here the noise can appear in two ways: On the one hand, our measurement device might only measure approximately in the computational basis, that is, the collaps is noisy. It is clear that this can be thought of as a perfect measurement followed by an error.
On the other hand, the device might just return the wrong measurement outcome. But this is obviously equivalent to a perfect measurement followed by a Pauli X error on the measured qubit.
We will assume that all errors are p-local stochastic noise for some probability $p \in (0, 1)$. We summarize the previous discussion in the following definition.

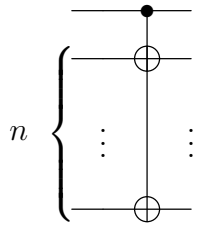
Definition 5.1.3. Let $\mathcal{U} = \mathcal{U}_D \dots \mathcal{U}_1$ be a depth-D implementation of $\mathcal{U}$ with input state $|0\rangle^{\otimes n}$ followed by a measurement in the computational basis. A *noisy implementation* of $\mathcal{U}$ with noise rate p will output a bit string z according to the conditional distribution

$$\mathrm{P}(z|\mathcal{E}_0, \dots, \mathcal{E}_{D+1}) = |\langle z | \mathcal{E}_{D+1}\mathcal{E}_D\mathcal{U}_D \dots \mathcal{E}_1\mathcal{U}_1\mathcal{E}_0 | 0^n\rangle|^2,$$

where the $\mathcal{E}_i$'s are drawn from some joint distribution such that $\mathcal{E}_i \sim \mathcal{N}(p)$.

It highly depends on the gates we use how severely errors may propagate.

Example 5.1.1. Suppose we work with a gate set containing the gate



which acts on n qubits as Pauli X controlled by the first qubit.
Now consider the error model where at each time step there is a Pauli X error on each qubit independently with probability p. If there is an error occuring on the first qubit before the gate is applied, this error will propagate and introduce an error on all other qubits.

Using multi qubit gates, single errors can cause a lot of damage. This is another reason why it is reasonable to work with 1 and 2 qubit gates only.

Lemma 5.1.1 ([BGKT19], Lemma 11 and 12)**.** (a) Let $\mathcal{E} \sim \mathcal{N}(p)$ and $\mathcal{E}' \sim \mathcal{N}(p')$ be random Pauli errors which may depend on each other. Then $\mathcal{E} \cdot \mathcal{E}' \sim \mathcal{N}(q)$ where $q = 2\max\{\sqrt{p}, \sqrt{p'}\}$.

(b) Let $\mathcal{E} \sim \mathcal{N}(p)$ and $\mathcal{U}$ be a product of 1 and 2 qubit Clifford gates acting on disjoint sets of qubits. Then $\mathcal{U}\mathcal{E}\mathcal{U}^\dagger \sim \mathcal{N}(\sqrt{2p})$.

(c) Let $\mathcal{U} = \mathcal{U}_D \dots \mathcal{U}_1$ be a depth-D circuit composed of depth-1 Clifford circuits. Then, the probability of getting the outcome $z \in \{0,1\}^n$ from a noisy implementation of $\mathcal{U}$ with noise rate p is given by

$$\mathrm{P}(z|E) = |\langle z|\mathcal{E}\mathcal{U}|0^n\rangle|^2$$

We want to apply this to our scheme for creating long-range entanglement. As we have seen, we can implement this scheme by a constant depth circuit followed by X measurements on single qubits for suitable codes like the surface code.
Therefore, the noisy version of this can be treated as a perfect implementation followed by a Pauli error and the measurements, which may be faulty as well.
We notice that X factors of this error on qubits that will be measured can be ignored.
The factors of the error acting on the qubits that are not to be measured commute with the measurements as they act on disjoint sets of qubits. Thus, these can be interpreted as errors that happened after the measurements. As the result of our scheme will be a quantum code, these errors may be corrected later on by these codes. Therefore, we will not care about them here.
With that, we only have to worry about Z-errors on measured qubits and about the faulty measurements. But in fact, these two are equivalent for us: Both of them lead to a wrong interpretation of the measurement outcome as a Z-error on the qubit sitting at vertex i only transforms the graph state stabilizer S_i to $-S_i$.
Therefore, it now remains to see to which extent we can detect these faulty measurements.

5.2 Syndromes for measurements in graph states

We want to detect faulty measurements. For this, we use that the measurement outcomes may be correlated due to the fact that qubits at connected vertices are entangled. A first simple lemma demonstrates how *syndromes* for measurement outcomes may arise.
In this paragraph, we always consider some graph $G = (V, E)$ and denote the corresponding graph state $|G\rangle$. Each vertex $i \in V$ gives rise to a stabilizer generator which we denote by S_i.

Lemma 5.2.1. Let $\mathcal{I} \subset V$ be a set of vertices. Suppose we measure all qubits from $\mathcal{I}$ in the X basis and for $i \in \mathcal{I}$ denote $x_i \in \{-1, 1\}$ the measurement outcome of the measurement of the qubit at i.
If a subset $\mathcal{J} \subset \mathcal{I}$ obeys

$$\prod_{j\in\mathcal{J}} S_j = \prod_{j\in\mathcal{J}} X_j$$

then it holds that

$$\mathrm{P}\left(\prod_{j\in\mathcal{J}} x_j = 1\right) = 1.$$

Proof. We notice that

$$\prod_{j\in\mathcal{J}} S_j$$

commutes with every single measurement. Therefore, it still stabilizes $|G\rangle$ after the measurements have been performed. But also, each of the qubits sitting at vertices $j \in \mathcal{J}$ is a $\pm 1 = x_j$ eigenstate of X_j. That gives

$$|G\rangle = \prod_{j\in\mathcal{J}} S_j |G\rangle = \prod_{j\in\mathcal{J}} x_j |G\rangle$$

□

The next lemma proves that these are essentially all correlations between the measurement outcomes.

Lemma 5.2.2. Let $\mathcal{I} \subset V$ be a set of vertices. Suppose we measure all qubits from $\mathcal{I}$ in X basis and for $i \in \mathcal{I}$ denote $x_i \in \{-1, 1\}$ the measurement outcome of the measurement of the qubit at i.
Let $\mathcal{J} \subset \mathcal{I}$ and assume

$$\mathrm{P}\left(\prod_{j\in\mathcal{J}} x_j = 1\right) = 1. \tag{5.2}$$

then it holds that

$$\prod_{j\in\mathcal{J}} S_j = \prod_{j\in\mathcal{J}} X_j.$$

Proof. For simplicity we assume $\mathcal{J} = \{1, \ldots, n\}$ and define A as the set of n-tupels in $\{+,-\}^n$ with an even number of +'s as entries.
We use the shorthands

$$\mathcal{O}_{\text{even}} = \left(\sum_{\underline{x}\in A} |x_1\rangle\langle x_1| \otimes \cdots \otimes |x_n\rangle\langle x_n|\right) \otimes \mathbb{1}_{V\setminus[n]}$$

$$\mathcal{O}_{\text{odd}} = \left(\sum_{\underline{x}\in\{+,-\}^n\setminus A} |x_1\rangle\langle x_1| \otimes \cdots \otimes |x_n\rangle\langle x_n|\right) \otimes \mathbb{1}_{V\setminus[n]}$$

By Born's rule we know that the probability in Equation (5.2) is given by

$$\mathrm{tr}\left[\mathrm{tr}_{V\setminus\mathcal{J}}[|G\rangle\langle G|] \sum_{\underline{x}\in A} |x_1\rangle\langle x_1| \otimes \cdots \otimes |x_n\rangle\langle x_n|\right].$$

which is equal to

$$\left\langle G \middle| \sum_{\underline{x}\in A} |x_1\rangle\langle x_1| \otimes \cdots \otimes |x_n\rangle\langle x_n| \otimes \mathbb{1}_{V\setminus[n]} \middle| G \right\rangle.$$

This implies that we have

$$\mathcal{O}_{\text{even}} |G\rangle = |G\rangle.$$

This, together with the fact

$$\mathcal{O}_{\text{even}} + \mathcal{O}_{\text{odd}} = \mathbb{1}$$

implies that

$$\mathcal{O}_{\text{odd}} |G\rangle = 0.$$

Remembering $X = |+\rangle\langle+| - |-\rangle\langle-|$ we conclude

$$X^{\otimes n} \otimes \mathbb{1}_{V\setminus[n]} |G\rangle = \mathcal{O}_{\text{even}} - \mathcal{O}_{\text{odd}} |G\rangle$$

which all together shows that $X^{\otimes n} \otimes \mathbb{1}_{V\setminus[n]}$ stabilizes $|G\rangle$.
Hence, it is an element of the stabilizer group of the graph state stabilizer, that is, a product of the S_i. Clearly, this is only possible if

$$X^{\otimes n} \otimes \mathbb{1}_{V\setminus[n]} = \prod_{i=1}^{n} S_i \otimes \mathbb{1}_{V\setminus[n]}.$$

This shows the claim. □

With this, we can reduce the problem of detecting faulty measurements to a graph theoretical problem: Consider a graph $G = (V, E)$ which gives rise to a graph state. We want to measure qubits from a certain set in the X basis. We denote the set of corresponding vertices $\mathcal{I}$. We call a subset $\mathcal{J} \subset \mathcal{I}$ of vertices *neighbourhood free* if each vertex in V is adjacent to an even number of vertices from $\mathcal{J}$ which is equivalent to

$$\prod_{j\in\mathcal{J}} S_j = \prod_{j\in\mathcal{J}} X_j$$

for the graph state stabilizers S_j. That is, the neighbourhood free sets correspond to the syndromes of measurement outcomes.
Correlations as in Equation (5.2) help us to detect faulty measurements: If our measurement outcomes violate an equality as in (5.2), we know that an involved measurement must be faulty.
Unfortunately, these syndromes do not always give sufficient information where the errors have happened. An example will come up later when we classify the neighbourhood free sets in our long range entanglement scheme.

We have the following:

Lemma 5.2.3. Let $G = (V, E)$ and $\mathcal{I} \subset V$ as above. Moreover, let $\mathcal{J}_1, \mathcal{J}_2 \subset \mathcal{I}$ be neighbourhood free. Then the symmetric difference $(\mathcal{J}_1 \cup \mathcal{J}_2) \setminus (\mathcal{J}_1 \cap \mathcal{J}_2)$ is neighbourhood free.

Proof. Choose some vertex $v \in V$. Let

$$\{\underbrace{j_1^1, \ldots, j_\lambda^1}_{\in\mathcal{J}_1\setminus\mathcal{J}_2}, \underbrace{j_1^2, \ldots, j_\mu^2}_{\in\mathcal{J}_2\setminus\mathcal{J}_1}, \underbrace{j_1, \ldots, j_\rho}_{\in\mathcal{J}_1\cap\mathcal{J}_2}\}$$

be the set of vertices in $\mathcal{J}_1 \cup \mathcal{J}_2$ adjacent to v. As $\mathcal{J}_1$ and $\mathcal{J}_2$ are neighbourhood free, we have that $\lambda + \rho$ and $\mu + \rho$ are even. Therefore, $\lambda + \mu$ is even which is exactly the number of vertices in $(\mathcal{J}_1 \cup \mathcal{J}_2) \setminus (\mathcal{J}_1 \cap \mathcal{J}_2)$ adjacent to v. □

This somehow reminds us of classical linear codes over $\mathcal{F}_2$: Say, $\mathcal{I} = \{1, \ldots, k\}$ and denote x_1 the measurement outcome of the X measurement at the vertex $i \in \mathcal{I}$. Let $m_i = \frac{1}{2}(x_i + 1) \in \{0, 1\}$ and $M = (m_i)_{i=1}^k \in \{0, 1\}^k$. Then, each neighbourhood free subset $\mathcal{J}$ gives rise to an equation

$$\sum_{i \in \mathcal{J}} m_i = 0,$$

that is, the neighbourhood free sets yield rows of a parity check matrix H. Each valid k-tuple of measurement outcomes M obeys

$$H \cdot M = 0.$$

In this context, Lemma 5.2.3 just states that the sum of two parity check rows is still a parity check row.
Let us now turn to our scheme for localizable long range encoded entanglement of two surface codes and see to which extent we can detect faulty measurements. Recall that we started with a graph state on a subset of a lattice: For a $(2k+l) \times k$ lattice $\tilde{\Lambda} = (\tilde{V}, \tilde{E})$ we consider the subgraph induced by

$$V = \tilde{V} \setminus \{(i, j) \in \tilde{V} | i \text{ even}, j \text{ odd}\}.$$

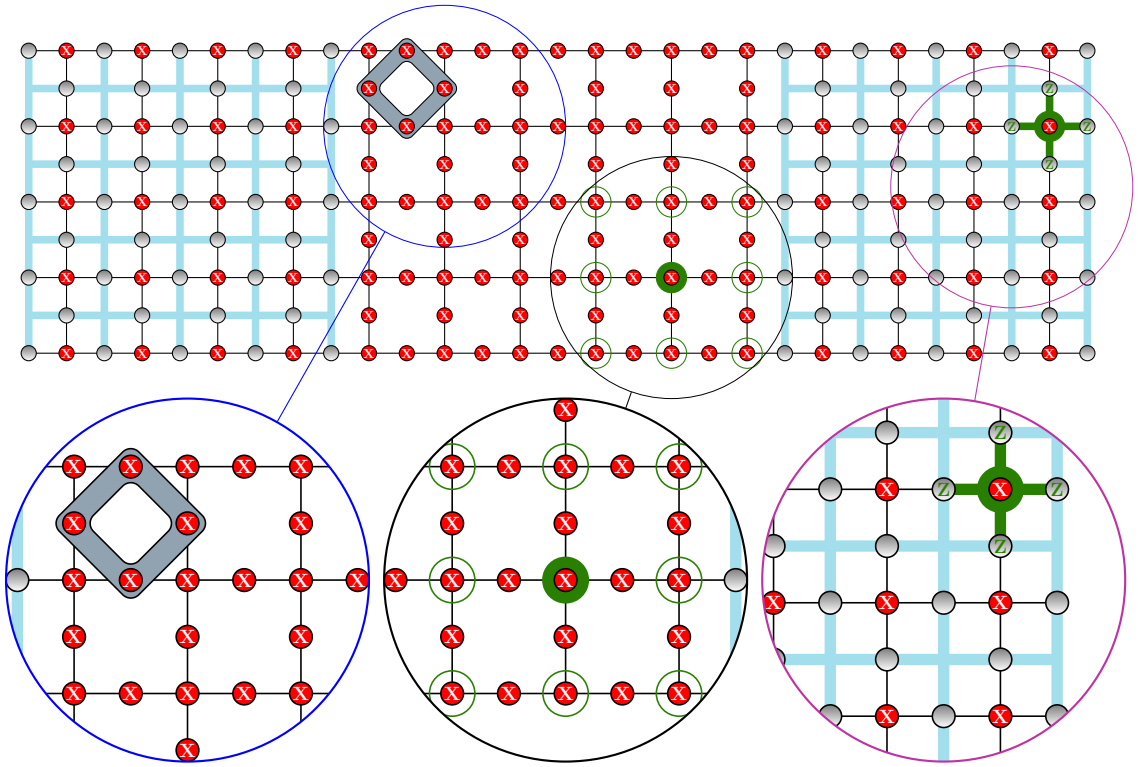
We call this graph $\Lambda = (V, E)$.
Also, recall the partition of the lattices in two $k \times k$ and one $l \times k$ lattice corresponding to the two copies of the surface code and the bulk and denote their vertex sets R, B and L in the natural way, that is, B is the vertex set of the bulk.
Recall that the set $\mathcal{I}$ of vertices that are measured in X basis is given by

$$\mathcal{I} = \{(i, j) \in L \cup R | i \text{ odd}, j \text{ even}\} \cup B.$$

We want to determine the parity constraints for the measurements, that is, the neighbourhood free subsets of $\mathcal{I}$. The following arguments are depicted in Figure 5.1.
First, we consider a vertex (i, j) with both i and j odd and assume we find a neighbourhood free set $\mathcal{J}$ containing it. Without loss, we assume that it does not lie on the boundary of the lattice, that is, it has neighbours $(i-1, j)$, $(i+1, j)$, $(i, j+1)$ and $(i, j-1)$. All these vertices have degree 2, that is, $(i-2, j)$, $(i+2, j)$, $(i, j+2)$ and $(i, j-2)$ must be in $\mathcal{J}$.
In particular, $\mathcal{J}$ must contain vertices of the form $(2k+l-1, j)$ for $j = 1, 3, \ldots, l-2, l$. But these are neighbours of the vertices $(2k + l, j)$ which have degree 1. This clearly contradicts the fact that $\mathcal{J}$ is neighbourhood free. Hence, there can not be a neighbourhood free set containing any vertex of the form (i, j) with both i and j odd. In other words, this tells us that the measurement outcomes at these qubits are not correlated with the other outcomes.
Now, let (i, j) be a vertex in the bulk with i even and j odd and – for simplicity – assume that $i < l + k$. We easily verify that

$$\mathcal{J} = \{(i, j), (i+2, j), (i+1, j+1), (i+1, j-1)\} \tag{5.3}$$

is neighbourhood free. With what we have done before, it is now easy to see that every neighbourhood free set must be a symmetric difference of neighbourhood free sets of the form in (5.3).



(a) (b) (c)

Figure 5.1: (a) For each bulk vertex of the form (i, j) with i even and j odd, we find the neighbourhood free set $\{(i, j), (i + 2, j), (i + 1, j + 1), (i + 1, j - 1)\}$. It is also obvious that every neighbourhood free set containing it is a symmetric difference of such sets. (b) If a neighbourhood free set contains a vertex (i, j) with both i and j even, it must contain all vertices with distance 2. With this, we see that such a set must contain all vertices of this form. (c) As a vertex adjacent to a vertex in the rough boundary of the resulting surface code is the only neighbour of it, we see that no neighbourhood free set can contain it. This together with (b) implies that no neighbourhood free set can contain a vertex of the form (i, j) with i and j even.

This analysis tells us that

(i) we have no correlations of the form (5.2) for qubits sitting on vertices (i, j) for both i and j odd. This is a severe restriction: An error on these qubits in L or R causes a wrong interpretation of the sign of the plaquette stabilizers. An error in these qubits in the bulk may lead to an encoded eigenstate for the wrong eigenvalue.

(ii) On the remaining measured qubits we indeed get a code which lets us detect and correct errors to a certain extent.

Bibliography

[BDCPS16] A. Bolt, G. Duclos-Cianci, D. Poulin, and T. Stace. Foliated quantum error-correcting codes. *physical review letters*, 117(7), Aug 2016.

[BGKT19] Sergey Bravyi, David Gosset, Robert Koenig, and Marco Tomamichel. Quantum advantage with noisy shallow circuits in 3D. *arXiv e-prints*, page arXiv:1904.01502, Apr 2019.

[BV97] E. Bernstein and U. Vazirani. Quantum complexity theory. *SIAM Journal of Computing*, 26:1411 – 1473, 1997.

[CGL+09] Yeow Meng Chee, Zhenbo Guo, San Ling, Fengjing Shao, Yuansheng Tang, Huaxiong Wang, and Chaoping Xing. *Coding and Cryptology*. Springer, Berlin Heidelberg, 1st edition, 2009.

[CLSZ95] I. L. Chuang, R. Laflamme, P. W. Shor, and W. H. Zurek. Quantum computers, factoring, and decoherence. *Science*, 270(5242):1633–1635, 1995.

[DJ97] D. Deutsch and R. Jozsa. Rapid solution of problems by quantum computation. *Proc. R. Soc. Lond. A.*, 439, Jan 1997.

[DP97] D. Deutsch and R. Penrose. Quantum theory, the church-turing principle and the universal quantum computer. *Proc. R. Soc. Lond. A.*, 400, Jan 1997.

[Fey82] R.P. Feynman. Simulating physics with computers. *Int. J. Theor. Phys.*, 21, 1982.

[HEB04] M. Hein, J. Eisert, and H. J. Briegel. Multiparty entanglement in graph states. *physical review letters*, 69(6):062311, Jun 2004.

[HHHH07] R Horodecki, P Horodecki, M Horodecki, and K Horodecki. Quantum entanglement. *Reviews of Modern Physics*, 81:865–942, Apr 2007.

[Kit03] A. Yu. Kitaev. Fault-tolerant quantum computation by anyons. *Annals of Physics*, 303(1):2–30, Jan 2003.

[MPSE96] G. Massimo Palma, K.A. Suominen, and A.K. Ekert. Quantum Computers and Dissipation. *Proceedings of the Royal Society of London Series A*, 452(1946):567–584, Mar 1996.

[NC11] Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information: 10th Anniversary Edition*. Cambridge University Press, New York, NY, USA, 10th edition, 2011.

[Pre01] J. Preskill. Chapter 4. quantum entanglement, 2001.

[PV16] C. Palazuelos and T. Vidick. Survey on nonlocal games and operator space theory. *Journal of Mathematical Physics*, 57(1):015220, Jan 2016.

[RBB03] R. Raussendorf, D. E. Browne, and H. J. Briegel. Measurement-based quantum computation on cluster states. *Phys. Rev. A*, 68(2), Aug 2003.

[RBH05] R. Raussendorf, S. Bravyi, and J. Harrington. Long-range quantum entanglement in noisy cluster states. *physical review letters*, 69(6), Jun 2005.

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

[Sho95] P. Shor. Scheme for reducing decoherence in quantum memory. *Phys. Rev. A*, 52, 1995.

[Unr95] W.G. Unruh. Maintaining coherence in quantum computers. *Phys. Rev. A*, 51:992–997, Feb 1995.

[vL99] J.H. van Lint. *Introduction to Coding Theory*. Springer, Berlin Heidelberg, 3rd edition, 1999.

[WW01] R.F. Werner and M.M. Wolf. Bell inequalities and entanglement. *arXiv e-prints*, pages quant–ph/0107093, Jul 2001.