

# **Fault-Tolerant Quantum Computation with Realistic Error Models**

*James Michael Auger*

A dissertation submitted in partial fulfillment  
of the requirements for the degree of  
**Doctor of Philosophy**  
of  
**University College London.**

Department of Physics and Astronomy  
University College London

I, James Michael Auger, confirm that the work presented in this thesis is my own. Where information has been derived from other sources, I confirm that this has been indicated in the work.

# Abstract

In this thesis, we consider steps towards building a useful quantum computer in the presence of realistic errors.

We start by presenting a novel scheme for fault-tolerant quantum computation with Rydberg atoms. The scheme uses topological error correction codes to provide fault-tolerance and utilises electromagnetically induced transparency to perform entangling gates between qubits. We examine the benefits of this scheme compared to alternative approaches, and we investigate the errors that are likely to occur and how these can be managed.

We then consider how to perform quantum computation with a surface code based quantum computer that has permanently faulty components. We study both faulty links between qubits, which prevent qubits from interacting, and faulty qubits themselves; such faults could affect a wide variety of physical implementations, including schemes that use superconducting qubits and those that use trapped ions. The approach we propose for dealing with these faulty components requires minimal modification to the quantum operations required by the surface code and should therefore be implementable by almost all current and future proposals for surface code based quantum devices.

Finally, we consider how to perform fault-tolerant quantum computation with components that fail probabilistically. This section is largely motivated by quantum computation with linear optics and therefore focuses on quantum computation with cluster states. We show that these probabilistic failures can be tolerated providing the failure rate is suitably low, and we present methods for increasing error thresholds in the presence of such errors.

# List of publications and preprints

Much of the work contained within this thesis is presented in the following papers:

- Auger, J. M., Bergamini, S., and Browne, D. E. (2017). Blueprint for fault-tolerant quantum computation with Rydberg atoms. *Phys. Rev. A*, 96(5):052320.
- Auger, J. M., Anwar, H., Gimeno-Segovia, M., Stace, T. M., and Browne, D. E. (2017). Fault-tolerance thresholds for the surface code with fabrication errors. *Phys. Rev. A*, 96(4):042316.
- Auger, J. M., Anwar, H., Gimeno-Segovia, M., Stace, T. M., and Browne, D. E. (2017). Fault-tolerant quantum computation with non-deterministic entangling gates. arXiv:1708.05627 [quant-ph].

# Acknowledgements

Firstly, I'd like to say a huge thank-you to my supervisor Dan Browne. His seemingly limitless patience and enthusiasm during my eight years at UCL have been indispensable and have always spurred me to go one step further in my research.

I'm also extremely grateful to Hussain Anwar and Fern Watson, who began sharing their knowledge with me on the day I started my PhD, and I'd like to thank my collaborators Mercedes Gimeno-Segovia, Tom Stace and Silvia Bergamini, who have each made vital contributions to my research. There are many more people that I've had the pleasure of working with during my PhD, but in particular I'd like to thank Chris Perry, Naomi Nickerson, Sam Morley-Short, Terry Rudolph, Ben Brown and Pete Shadbolt for numerous interesting and fruitful discussions over the years.

None of this would have happened without the continued support of my parents, Kathryn and David, who have always encouraged and enabled me to pursue my interests, so I am immeasurably thankful to them. Finally, my biggest thanks are to my amazing wife Amy, who has been behind me all the way, and my son Eddie, whose contagious smile has served as a wonderful distraction from writing this thesis.

# Contents

<b>I</b>	<b>Introduction</b>	<b>12</b>
<b>1</b>	<b>Quantum computation</b>	<b>13</b>
1.1	Quantum bits and gates . . . . .	14
1.2	The circuit model of quantum computation . . . . .	15
1.3	Clifford gates . . . . .	18
<b>2</b>	<b>Quantum error correction</b>	<b>20</b>
2.1	Stabiliser codes . . . . .	21
2.2	The surface code . . . . .	24
2.2.1	Detecting and correcting errors . . . . .	26
2.2.2	Multiple logical qubits and computation . . . . .	30
<b>II</b>	<b>Fault-tolerant quantum computation with realistic error models</b>	<b>33</b>
<b>3</b>	<b>A blueprint for fault-tolerant quantum computation with Rydberg atoms</b>	<b>34</b>
3.1	Quantum computation with Rydberg atoms . . . . .	35
3.2	Proposed scheme . . . . .	37
3.2.1	EIT gates . . . . .	37
3.2.2	Trapping atoms . . . . .	40
3.2.3	Error correction and computation . . . . .	41
3.2.4	Leakage and loss . . . . .	44

3.3	Threshold simulations . . . . .	45
3.3.1	Error model . . . . .	45
3.3.2	Simulation methods . . . . .	46
3.4	Results and discussion . . . . .	46
3.4.1	Correlated errors . . . . .	49
3.5	Conclusion . . . . .	49
<b>4</b>	<b>Fault-tolerance thresholds for the surface code with fabrication errors</b>	<b>51</b>
4.1	Fabrication errors . . . . .	53
4.1.1	Measuring supercheck operators and gauge qubits . . . . .	55
4.1.2	Mapping fabrication errors to faulty data qubits . . . . .	57
4.1.3	Percolation thresholds and effective code distance . . . . .	57
4.1.4	Complications . . . . .	62
4.2	Threshold simulations . . . . .	64
4.2.1	Error model . . . . .	64
4.2.2	Simulation methods . . . . .	66
4.3	Results . . . . .	68
4.4	Discussion . . . . .	70
4.4.1	Scaling of supercheck operator weight . . . . .	71
4.5	Conclusion . . . . .	73
<b>5</b>	<b>Fault-tolerant quantum computation with non-deterministic entangling gates</b>	<b>75</b>
5.1	Topological cluster states . . . . .	77
5.1.1	Detecting and correcting errors . . . . .	78
5.1.2	Logical qubits and computation . . . . .	80
5.2	Bond loss . . . . .	82
5.2.1	Non-adaptive method . . . . .	83
5.2.2	Adaptive method . . . . .	84
5.3	Threshold simulations . . . . .	84
5.3.1	Error model . . . . .	84

<i>Contents</i>	8
5.3.2 Simulation methods . . . . .	85
5.4 Results . . . . .	87
5.5 Discussion . . . . .	88
5.5.1 Fault-tolerant linear optical quantum computation . . . . .	89
5.6 Conclusion . . . . .	90
<b>6 Summary and outlook</b>	<b>92</b>
<b>Bibliography</b>	<b>95</b>

# List of Figures

1.1	Examples of quantum circuit diagrams . . . . .	18
2.1	Planar code lattices with dimension $L = 5$ . . . . .	25
2.2	Circuit for performing a vertex check on the surface code . . . . .	26
2.3	Two distinct error strings that result in the same excitations on the surface code . . . . .	27
2.4	Examples of successful and unsuccessful error correction strings on the planar code . . . . .	28
2.5	Circuit for implementing a $T$ gate . . . . .	31
3.1	The Rydberg atom dipole blockade . . . . .	36
3.2	An overview of our proposed scheme for quantum computation using Rydberg atoms . . . . .	38
3.3	Using EIT to perform a CNOT gate . . . . .	39
3.4	Measuring a star operator using a multi-target EIT CNOT gate . . . . .	39
3.5	Arrangement of Rydberg atoms for a planar code . . . . .	40
3.6	Stabiliser measurement pattern . . . . .	42
3.7	Leakage detection circuit . . . . .	45
3.8	Steps for simulating error correction with Rydberg atoms . . . . .	47
3.9	Logical error rates from error simulations . . . . .	48
4.1	The impact of fabrication errors on the topology of the surface code . . . . .	52
4.2	Syndrome extraction circuits for star and plaquette stabiliser generators . . . . .	54
4.3	Forming supercheck operators in the presence of data qubit fabrication errors . . . . .	56

4.4 Mapping link fabrication errors to disabled data qubits . . . . .	57
4.5 Mapping syndrome fabrication errors to disabled data qubits . . . . .	58
4.6 Percolation rates for link fabrication errors only . . . . .	60
4.7 Percolation rates for qubit fabrication errors only . . . . .	60
4.8 Average effective code distance for link fabrication errors only . . . . .	61
4.9 Average effective code distance for qubit fabrication errors only . . . . .	62
4.10 Ratio between average effective code distance and intended code distance for link fabrication errors only only . . . . .	63
4.11 Ratio between average effective code distance and intended code distance for qubit fabrication errors only . . . . .	63
4.12 Effective order of gauge operator measurement with damaged checks	64
4.13 Dealing with fabrication errors at the edge of the lattice . . . . .	65
4.14 Steps for simulating surface code error correction with fabrication errors . . . . .	67
4.15 Logical error rates with qubit fabrication error rate $p_{\text{qubit}} = 4\%$ . . . . .	69
4.16 Pauli error thresholds for fabrication errors . . . . .	69
4.17 Logical error rates for codes with fabrication errors with intended distance $L = 13$ and actual distance $L' = 9$ compared to native distance 9 codes with no fabrication errors . . . . .	71
4.18 Rate at which highest-weight supercheck operators occur for link fabrication error rate $p_{\text{link}} = 10\%$ . . . . .	72
5.1 The three-dimensional topological cluster state . . . . .	76
5.2 Abstract representation of correlation surfaces and logical errors for defect-based TCS . . . . .	81
5.3 TCS supercheck operators . . . . .	83
5.4 Steps for simulating TCS with bond failures . . . . .	86
5.5 Thresholds in the presence of failed bonds. . . . .	88
5.6 Simulation results showing the relationship between fusion-gate suc- cess rates and bond failure rates when building a linear optics device	90

# List of Tables

- 3.1 Crosstalk interaction strength . . . . . 43
- 3.2 Error model for Rydberg atom scheme simulations . . . . . 45
- 4.1 Error model for surface code simulations with fabrication errors . . . 65
- 5.1 Error model for TCS simulations with bond failures . . . . . 85

# **Part I**

## **Introduction**

## Chapter 1

# Quantum computation

Quantum computers have the potential to revolutionise the technological world, opening the door to an exponential increase in processing power compared to current-day ‘classical’ computers. They will be able to perform tasks such as efficiently simulating quantum systems and breaking the encryption that underpins secure communication over the internet. Although quantum computers will initially be limited to solving very specific problems, the evolution of classical computers from room-sized code-breaking machines, such as *Colossus*, to multi-purpose devices that fit in our pockets serves to show that the greatest benefits offered by quantum computers are likely yet to be discovered.

The idea of a quantum computer was first proposed during a 1981 talk by Richard Feynman [Feynman, 1982], where he discussed the concept of a *universal quantum simulator*. It is practically impossible to simulate large quantum systems on current-day classical (non-quantum) computers, even when using the most powerful supercomputers, with the difficulty being that the resources required to simulate a quantum system scale exponentially with the size of the system being simulated. It is therefore believed that a computer that directly exploits quantum phenomena will be capable of efficiently solving exact quantum simulation problems that cannot be solved efficiently on a classical computer.

In addition to quantum simulation, quantum computers are expected to solve other problems for which no efficient classical algorithms are known. For example, Shor’s algorithm [Shor, 1997] allows quantum computers to efficiently factor

numbers and therefore potentially break RSA encryption, while Grover's unordered-search algorithm [Grover, 1996] is quadratically faster than the fastest known classical equivalent. There are also quantum algorithms that outperform the fastest known classical algorithms for tasks such as Monte Carlo simulation [Montanaro, 2015] and solving certain problems involving linear systems of equations [Harrow et al., 2009].

Quantum computation is being pursued with a variety of physical systems including linear optics [Knill et al., 2001, Kok et al., 2007, Gimeno-Segovia et al., 2015], trapped ions [Cirac and Zoller, 1995, Häffner et al., 2008], quantum dots [Loss and DiVincenzo, 1998], Rydberg atoms [Saffman et al., 2010] and superconducting Josephson junctions [Devoret et al., 2004, Clarke and Wilhelm, 2008, Barends et al., 2014]. Hybrid schemes that combine different systems are also in development [Walquist et al., 2009], as are distributed schemes that use fibre-optic communication links to combine small, physically separated quantum units to build a full quantum computer [Nickerson et al., 2014].

The delicate nature of quantum systems means that all approaches to building a quantum computer will be vulnerable to errors; a quantum computer that can still perform reliable quantum computation in the presence of errors is said to be *fault-tolerant*.

## 1.1 Quantum bits and gates

While most classical computers use bits in zero or one states, it is likely that quantum computers will use quantum bits, or qubits, that exist in a superposition of zero and one states simultaneously, such as

$$|\psi\rangle = \alpha_0 |0\rangle + \alpha_1 |1\rangle. \quad (1.1)$$

Here  $|\psi\rangle$  labels the quantum state, and  $|0\rangle$  and  $|1\rangle$  are the quantum analogues of the 0 and 1 binary states used in classical computation. The  $\alpha$  coefficients are complex numbers that represent the mix between the  $|0\rangle$  and  $|1\rangle$  states and satisfy

$|\alpha_0|^2 + |\alpha_1|^2 = 1$ . A state involving two qubits is written in the form

$$|\psi\rangle = \sum_{i,j=0}^1 \alpha_{ij} |i\rangle \otimes |j\rangle = \alpha_{00} |0\rangle \otimes |0\rangle + \alpha_{01} |0\rangle \otimes |1\rangle + \dots, \quad (1.2)$$

where  $\otimes$  is the tensor product operation, although this will be abbreviated to

$$|\psi\rangle = \sum_{i,j=0}^1 \alpha_{ij} |ij\rangle. \quad (1.3)$$

A quantum state can involve any number of qubits, with more qubits generally being required for more complex calculations.

Several schemes have been suggested for quantum computation, and these can be broadly divided into a number of equivalent *universal* schemes [Deutsch, 1985] (capable of performing arbitrary quantum computations) and *non-universal* schemes (which cannot perform arbitrary quantum computations but are still believed to outperform classical computers for certain tasks).

Examples of universal schemes include the circuit model [Feynman, 1986, Barenco et al., 1995], adiabatic quantum computation [Farhi et al., 2000], topological quantum computation [Kitaev, 2003], measurement-based quantum computation [Raussendorf and Briegel, 2001] and quantum walks [Aharonov et al., 1993]. Examples of non-universal schemes include quantum annealing [Finnila et al., 1994, Kadowaki and Nishimori, 1998] (used by the *D-Wave* commercial quantum computers), boson sampling [Aaronson and Arkhipov, 2011] and the one clean qubit model [Knill and Laflamme, 1998]. The ideas from different universal schemes can be combined — for example, topological quantum computation can be performed using the circuit model, which is the focus of much of this thesis.

## 1.2 The circuit model of quantum computation

The circuit model of quantum computation is the scheme that bears the most similarity to classical computation: in analogy to the logic gates used by classical computers, one defines a set of quantum gates that is universal for quantum compu-

tation. A set of gates is universal for quantum computation if that gate set can be used to perform an arbitrary quantum computation to arbitrary precision [Deutsch, 1985, Barenco et al., 1995].

An example of such a gate set for two or more qubits is the Hadamard, or  $H$ , gate, the  $T$  gate (sometimes called the  $\pi/8$  gate) and the two-qubit controlled-NOT, or CNOT, gate [Boykin et al., 1999]. When writing the state of a qubit in column vector notation,

$$\alpha_0 |0\rangle + \alpha_1 |1\rangle = \begin{bmatrix} \alpha_0 \\ \alpha_1 \end{bmatrix}, \quad (1.4)$$

these gates are represented in matrix form as:

$$H = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \quad T = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\pi/4} \end{bmatrix} \quad \text{CNOT} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}.$$

It is possible to efficiently find a sequence of these gates to perform any quantum computation efficiently and to arbitrary accuracy by using the Solovay-Kitaev algorithm [Kitaev, 1997].

The Pauli  $X$ ,  $Y$  and  $Z$  gates are extremely useful in quantum computation, so although they can be generated from products of  $H$  and  $T$  gates, it is often desirable to implement them directly. The identity gate,  $I$ , is also conceptually useful — it is the gate that leaves a quantum state unchanged. The  $I$ ,  $X$ ,  $Y$  and  $Z$  gates are:

$$I = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \quad X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \quad Y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix} \quad Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}.$$

The group generated by  $\langle X, Y, Z \rangle$  is referred to as the single-qubit Pauli group, and the group generated by all  $n$ -qubit tensor products of Pauli operators is referred to as the  $n$ -qubit Pauli group  $\mathcal{P}_n$ . Note that  $XYZ = iI$ , meaning that  $\pm I$  and  $\pm iI$  are elements of every Pauli group.

Two other useful gates that can be generated from products of  $H$ ,  $T$  and CNOT gates are the phase, or  $S$ , gate, and the controlled- $Z$ , or CPHASE, gate:

$$S = \begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix} \quad \text{CPHASE} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix}.$$

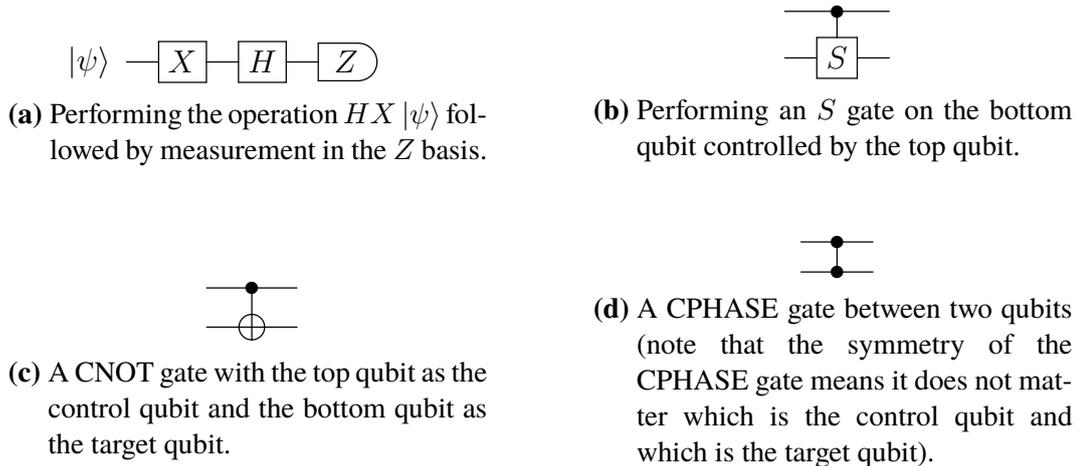
The Pauli  $X$  gate acts much like a classical NOT gate, swapping the  $|0\rangle$  and  $|1\rangle$  states of a qubit. The CNOT and CPHASE gates perform  $X$  and  $Z$  gates respectively on a target qubit depending on the state of a control qubit. If the control qubit is in the state  $|1\rangle$  then the respective gate is performed on the target qubit, otherwise the identity gate is performed on the target qubit.

The eigenvectors of the single qubit Pauli operators each form an orthonormal basis that can be used to represent the state of a single qubit. For example, the eigenvectors of the  $Z$  operator are  $|0\rangle$  and  $|1\rangle$ , and the eigenvectors of the  $X$  operator are  $|+\rangle = (|0\rangle + |1\rangle)/\sqrt{2}$  and  $|-\rangle = (|0\rangle - |1\rangle)/\sqrt{2}$ .

Qubits can be measured in any of these bases to obtain an outcome corresponding to the eigenvalue associated with the eigenvector; this is always  $\pm 1$  for Pauli operators. If a qubit is in a superposition of basis states when it is measured then it will *collapse* into one of the basis states with a probability given by the square of the coefficients in equation 1.1. For example, performing a  $Z$  measurement on the state  $|0\rangle$  will always give the outcome  $+1$ , but performing a  $Z$  measurement on the state  $(1/2)|0\rangle + (\sqrt{3}/2)|1\rangle$  will give the outcome  $+1$  with probability  $(1/2)^2 = 1/4$ , or the outcome  $-1$  with probability  $(\sqrt{3}/2)^2 = 3/4$ ; the quantum state then will be either  $|0\rangle$  or  $|1\rangle$  respectively.

Quantum circuit diagrams are used to represent computations using these quantum gates. In these diagrams, a horizontal line is used to represent a qubit, and most quantum gates are represented by a box intersecting the qubits upon which the gate acts, with a letter inside the box labelling the gate. Time runs from left to right in circuit diagrams. For example, Fig. 1.1a shows the circuit diagram for  $HX|\psi\rangle$

(applying an  $X$  gate and then an  $H$  gate to the state  $|\psi\rangle$ ) followed by measurement in the computational  $Z$  basis). Controlled gates are represented by a line from the control qubit to the gate, as shown in Fig. 1.1b. The CNOT and CPHASE gates are represented slightly differently to most other gates, as shown in Fig. 1.1c and Fig. 1.1d.



**Figure 1.1:** Examples of quantum circuit diagrams.

### 1.3 Clifford gates

The group of gates generated by the  $H$ ,  $S$  and CNOT gates acting on  $n$  qubits is a special group in quantum computation known as the  $n$ -qubit Clifford group,  $\mathcal{C}_n$  [Gottesman, 1998]. Gates in this group, called Clifford gates, leave the Pauli group,  $\mathcal{P}_n$ , fixed under conjugation, i.e.

$$C\mathcal{P}_nC^\dagger = \mathcal{P}_n \quad \forall \quad C \in \mathcal{C}_n. \quad (1.5)$$

Note that the  $T$  gate is not a member of the Clifford group.

The Gottesman-Knill theorem [Gottesman, 1998] states that computations consisting only of Clifford group gates and Pauli group measurements can be efficiently simulated on a classical computer. This means that at least one non-Clifford gate must be implementable to perform efficient computations that are not possible on a classical computer. As will be seen in later sections, this theorem has implications

for the resources required by quantum error correction: it is often the case that Clifford gates are easier to perform than non-Clifford gates.

## Chapter 2

# Quantum error correction

Building a working universal quantum computer will almost certainly require the use of some form of error correction to circumvent the effects of decoherence (qubits deviating from their intended states). For classical computations, a simple error correction procedure may involve making multiple copies of the data, performing the same operations on each copy and comparing the state of the classical bits in each copy at various stages throughout the computation. Providing the error rate is low enough, a majority vote can be used to infer the correct state of each bit.

Such a duplication and majority voting scheme cannot be used for quantum computation for two reasons. Firstly, it is impossible to copy a general quantum state due to the no-cloning theorem [Wootters and Zurek, 1982]; this means that qubits cannot be copied during a computation. Secondly, quantum states cannot be compared during a computation because doing so would require measurement of the qubits, causing them to collapse into the basis in which they are measured (see section 1.2); this measurement collapse effectively leaves the qubits in a classical state and no longer useful for quantum computation. More sophisticated methods are therefore required to detect and correct errors.

Like classical error correction codes, quantum error correction codes use multiple physical qubits per *logical qubit*. A computation fails if there is an uncorrectable *logical error* on a logical qubit. Quantum error correction also uses the concept of an error threshold, which is defined as the maximum physical qubit error rate at which a quantum computation can be performed reliably. The definition of the error

rate depends on the error model under consideration, but in general it is a parameter used to quantify the frequency with which errors occur for the chosen model. The existence of a threshold for quantum error correction was shown in [Aharonov and Ben-Or, 2008].

If the physical qubit error rate is lower than the threshold, increasing the number of physical qubits will reduce the logical error rate. If the physical qubit error rate is higher than the threshold, increasing the number of physical qubits will not, in general, reduce the logical error rate. This means, in theory, that below the physical qubit error threshold an arbitrary quantum computation can be performed if there are enough physical qubits, although this number may be enormous. The exact relationship between logical error rates and the number of physical qubits per logical qubit depends on the particular quantum error correction code in question — as an example, surface code quantum error correction codes are likely to require  $10^3$ - $10^4$  physical qubits per logical qubit [Fowler et al., 2012], and Shor’s algorithm [Shor, 1997] requires  $O(n)$  qubits to factorise an  $n$  bit number, so factorising a 2048-bit RSA key would likely require in excess of a million qubits.

## 2.1 Stabiliser codes

Stabiliser codes are an important class of quantum error correction codes, and they can be fully described using Pauli operators. A state  $|\psi\rangle$  is said to be stabilised by an operator  $S$  if

$$S|\psi\rangle = |\psi\rangle. \quad (2.1)$$

For example, the two-qubit state

$$|\Phi^+\rangle = \frac{1}{\sqrt{2}} (|00\rangle + |11\rangle) \quad (2.2)$$

is stabilised by both  $X_1 \otimes X_2$  ( $X$  acting simultaneously on both qubits) and  $Z_1 \otimes Z_2$  ( $Z$  acting simultaneously on both qubits).

Stabiliser codes are defined as follows [Gottesman, 1997]. Suppose that  $S$  is

a subgroup of the  $n$ -qubit Pauli group,  $\mathcal{P}_n$ , and  $V_S$  is the set of  $n$ -qubit states that satisfies

$$S|v\rangle = |v\rangle \quad \forall S \in \mathcal{S}, |v\rangle \in V_S. \quad (2.3)$$

The group  $\mathcal{S}$  is defined to be the stabiliser of the states in the set  $V_S$ , and the states in the set  $V_S$  define logical qubit states;  $V_S$  is known as the code space. It can be seen that  $-I$  cannot be in  $\mathcal{S}$  (because  $-I|\psi\rangle = -|\psi\rangle$ ), which in turn requires that  $\pm iI$  is not in  $\mathcal{S}$  to satisfy group axioms.

In addition to the definition in equation 2.3, the stabiliser  $\mathcal{S}$  can also be defined using the group elements that generate it. In general, a minimum generating set  $\langle g_1, \dots, g_l \rangle$  of  $\mathcal{S}$  is chosen such that all  $g_i$  are independent Pauli operators (no element  $g_i$  is a product of the others). Errors can then be detected by measuring these *stabiliser generators*.

As Pauli operators have eigenvalues  $\pm 1$ , the outcome of any stabiliser generator measurement will be  $\pm 1$ . Obtaining a ‘ $-1$ ’ outcome when measuring a stabiliser generator  $g_i$  indicates that the qubits are no longer in a superposition of states in  $V_S$  because equation 2.3 is not satisfied; this means an error that anticommutes with  $g_i$  has occurred. Measuring all stabiliser generators gives a *syndrome* of outcomes from which an attempt can be made to determine which error, if any, has occurred, and a suitable correction can then be applied. Detecting and correcting Pauli errors alone is sufficient to correct more general errors with stabiliser codes, as arbitrary errors collapse into Pauli errors when the stabiliser generators are measured [Nielsen and Chuang, 2000]; this is known as *error discretisation*.

Any Pauli operator that commutes with all stabiliser generators but is not itself in the stabiliser will not result in any ‘ $-1$ ’ outcomes. Such operations are logical operations that change the state of one or more logical qubits, and logical operations occurring as a result of errors cannot be corrected and will cause the computation to fail.

A quantum error correction code that uses  $n$  physical qubits to represent  $k$  logical qubits and can reliably correct  $\lfloor (d-1)/2 \rfloor$  errors is referred to as an  $[[n, k, d]]$

code.  $d$  is called the distance of the code, and is equal to the weight of the lowest-weight logical operator, where the weight of an operator is the number of qubits upon which the operator acts non-trivially (i.e. with any operator other than the identity gate). For example, the operator  $X_1 I_2 Z_3 I_4 X_5 Y_6 X_7$  has weight five<sup>1</sup>. A larger code distance means a greater number of physical qubit errors are required to cause a logical error. The number of stabiliser generators required by an  $[[n, k, d]]$  stabiliser code is given by  $n - k$ .

Two of the most common approaches to implementing fault-tolerant quantum computation with stabiliser codes are code concatenation and topological codes. Code concatenation [Knill and Laflamme, 1996] involves replacing each physical qubit in a stabiliser code with a logical qubit of another stabiliser code (potentially the same or a different code), and topological codes encode logical qubits in such a way that logical states are protected by global topological properties of the code, with the most common examples being the surface code [Kitaev, 2003] and colour codes [Bombin and Martin-Delgado, 2006]. It is even possible to combine topological codes with code concatenation, e.g. [Criger and Terhal, 2016].

## Gauge operators

Stabiliser codes can be modified by the introduction of *gauge operators*, which are operators that don't commute with the stabiliser generators but don't affect the encoded logical state: they effectively act as logical operators for *gauge qubits* that are not used to store useful information [Poulin, 2005]. Gauge operators are useful as their measurement outcomes can be used to obtain stabiliser operator outcomes by multiplying them together, in which case the individual outcomes are random but the product is deterministic; these codes are known as *subsystem codes*. Using gauge operators means that lower-weight measurements can be used to effectively perform higher-weight stabiliser operations, such as in [Bacon, 2006].

---

<sup>1</sup>Here  $X_1 I_2 Z_3 I_4 X_5 Y_6 X_7 = X \otimes I \otimes Z \otimes I \otimes X \otimes Y \otimes X$ , and this will generally be further abbreviated to  $X_1 I_2 Z_3 I_4 X_5 Y_6 X_7 = X I Z I X Y X$ . The distinction between operator multiplication and tensor product will be clear from the context.

## 2.2 The surface code

The surface code is a topological stabiliser code, with the prototypical example being the toric code due to Kitaev [Kitaev, 2003]. The surface code is one of the most promising quantum error correction codes due to its relatively high thresholds [Wang et al., 2003, Stephens, 2014] and attractive scaling qualities. It is ideally suited to two-dimensional chip-based implementations, with superconducting qubits currently at the forefront of progress towards building a surface code quantum computer [Kelly et al., 2015, Córcoles et al., 2015].

The simplest form of the surface code is the planar code [Bravyi and Yu. Kitaev, 1998], which is defined using a square lattice of *data qubits* arranged on a planar surface (or manifold) with dimension  $L \times L$ . We will consider the planar code as a quantum memory to explain the key features of the surface code.

Fig. 2.1a shows an example of a planar code lattice with  $L = 5$ , where each edge of the lattice corresponds to a physical data qubit. The boundaries at the left and right of the code are referred to as *rough* boundaries and the boundaries at the top and bottom of the code as *smooth* boundaries. Each vertex,  $s$ , is associated with a *star* stabiliser generator,  $A_s$ , on the qubits surrounding the vertex

$$A_s = \bigotimes_{i \in s} X_i, \quad (2.4)$$

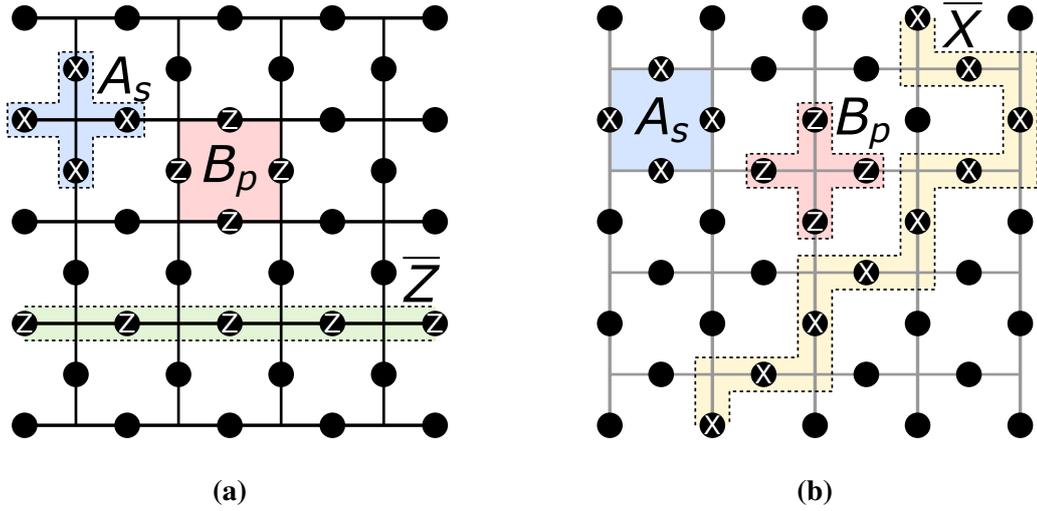
and each square,  $p$ , is associated with a *plaquette* stabiliser generator,  $B_p$ , on the qubits surrounding the square

$$B_p = \bigotimes_{i \in p} Z_i. \quad (2.5)$$

All stabiliser generators involve only neighbouring qubits.

In addition to the lattice configuration in Fig. 2.1a, which we refer to as the *primal* lattice, we can also define a *dual* lattice where the roles of vertices and plaquettes are swapped but the physical data qubits are unchanged, as shown in Fig. 2.1b. The concept of primal and dual lattices is useful for describing logical Pauli operators and for error correction.

Logical  $Z$  and  $X$  operators,  $\bar{Z}$  and  $\bar{X}$ , are strings of single-qubit  $Z$  ( $X$ ) operators



**Figure 2.1:** Planar code lattices with dimension  $L = 5$ . (a) Primal planar code lattice showing a vertex check  $A_s$ , a plaquette check  $B_p$  and a logical  $Z$  Pauli operator  $\bar{Z}$ . (b) Dual lattice showing the same checks as (a) and a logical  $X$  Pauli operator  $\bar{X}$  that anticommutes with the logical  $Z$  operator in (a).

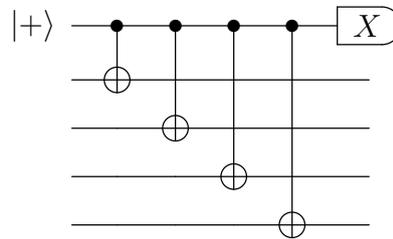
from the left to right (top to bottom) boundaries on the primal (dual) lattice, as shown in Fig. 2.1; these logical operators commute with all stabiliser generators and anticommute with each other. The choice of which operator is logical  $X$  and which is logical  $Z$  is arbitrary, although we will stick to the convention of having the operator consisting of only  $X$  operations as the  $X$  logical operator, and the operator consisting of only  $Z$  operations as the logical  $Z$  operator. Note that the logical operators are not unique: each logical operator can be modified by multiplication with an element of the stabiliser to form a new logical operator that has the same effect on the logical state of the code in the absence of errors, i.e.

$$\left. \begin{aligned} \bar{X} |\bar{\psi}\rangle &= S\bar{X} |\bar{\psi}\rangle \\ \bar{Z} |\bar{\psi}\rangle &= S\bar{Z} |\bar{\psi}\rangle \end{aligned} \right\} \quad \forall S \in \mathcal{S}, \quad (2.6)$$

where  $|\bar{\psi}\rangle$  represents the encoded logical state of the code. This means each logical operator can be redefined by combining it with stabiliser generators such that they need not be straight paths across the lattice, such as the logical  $X$  operator in Fig. 2.1b. The lowest-weight logical operator on the planar code is one that has a straight path across the lattice, so the code distance of the planar code is  $d = L$ . A

distance  $d$  planar code requires  $\sim 2d^2$  data qubits.

The measurement of a stabiliser generator, for example using the circuit shown in Fig. 2.2, gives the outcomes  $\pm 1$  depending on the state of the data qubits. Each stabiliser generator measurement is referred to as a *check*, with the ancilla (helper) qubit used for the measurement in Fig. 2.2 referred to as a *syndrome qubit*. Syndrome qubits are not strictly part of the definition of the planar code, as it is possible to measure the stabiliser generators using different methods, but we will generally assume there is a syndrome qubit associated with each star and plaquette check. A ‘ $-1$ ’ outcome from a check is referred to as an *excitation*, and the set of excitations obtained by performing all checks is referred to as the *syndrome*.



**Figure 2.2:** Circuit for performing a vertex check on the surface code. The top qubit is an ancilla qubit referred to as the *syndrome qubit*, and the measurement outcome of this qubit gives the outcome of the check.

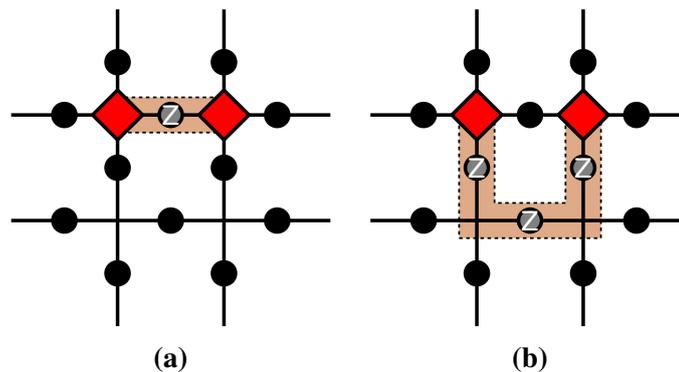
### 2.2.1 Detecting and correcting errors

$X$  and  $Z$  errors can be detected and corrected in an independent and analogous manner, so this section will now focus on the vertex checks on the primal lattice, which are used to detect and correct  $Z$  errors; the phrase *syndrome* will therefore refer only to the excitations detected by the set of vertex checks.

Error detection proceeds by measuring all the stabiliser generators. When there are no errors, the outcome of all measurements is ‘ $+1$ ’, and if an error occurs that anti-commutes with a stabiliser generator, then the outcome of that stabiliser generator is flipped to ‘ $-1$ ’. If a string of a single-type of Pauli error occurs, only the ends of the string are detectable, and different error strings may lead to the same detection pattern.

As an example, assume that the planar code initially has no errors and is in the ‘ $+1$ ’ state for all vertex checks, and consider a single  $Z$  error occurring on a data

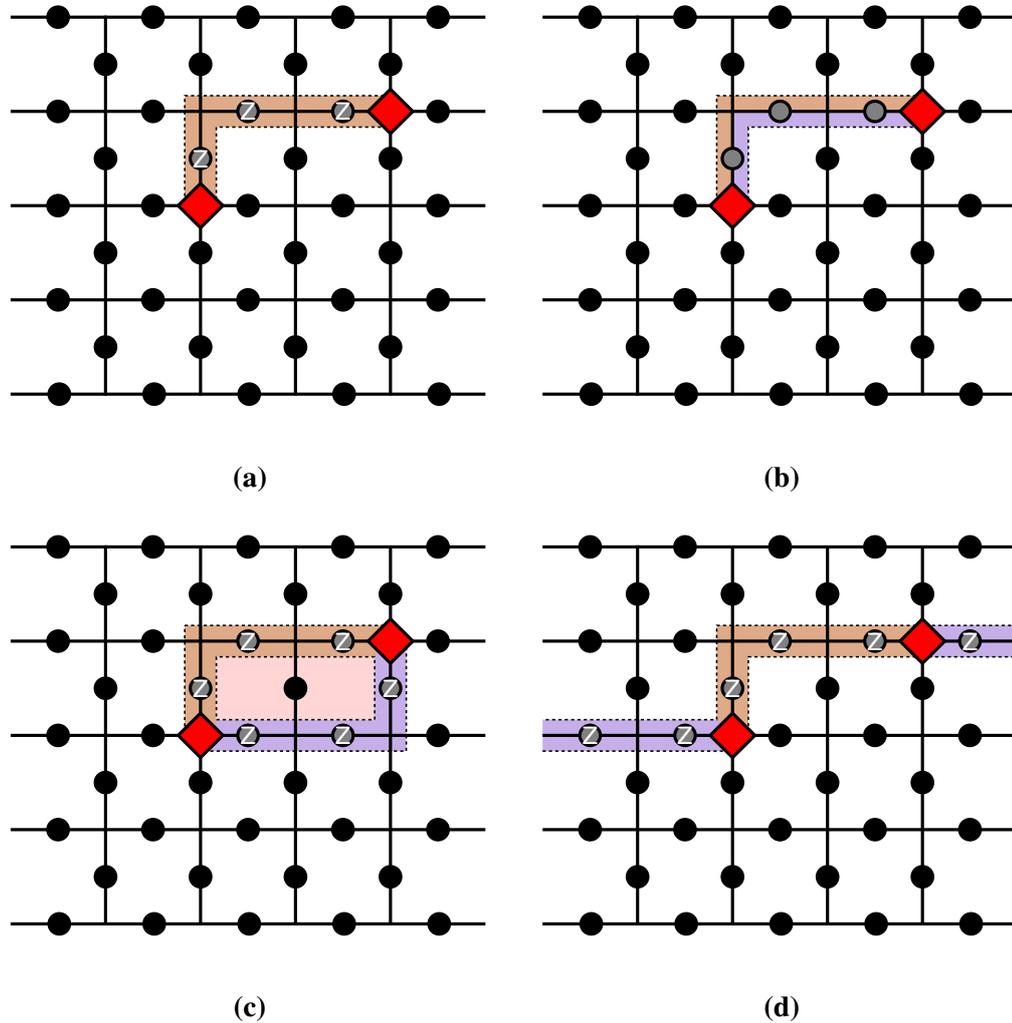
qubit. This  $Z$  error will anticommute with the stabiliser generators of the two vertex checks it is adjacent to, therefore causing an excitation at each of these vertices, as shown in Fig. 2.3a. However, if we have a string of adjacent errors such as the three  $Z$  errors shown in Fig. 2.3b, we will only be able to detect the ends of that string as the errors will commute with all stabiliser generators except those acting on the qubits at the ends of the string. Therefore, the error string in Fig. 2.3b will cause excitations in the same locations as the error in Fig. 2.3a. This feature of the code, where syndromes are not uniquely associated with a particular set of errors, is referred to as *degeneracy* — different errors can have the same excitations, so it is impossible to know for certain which error has occurred. In the case of Fig. 2.3, applying a Pauli  $Z$  to the location of the error in Fig. 2.3a is sufficient to correct either of the errors, because in the case of Fig. 2.3a the correction will cancel the error (as  $Z^2 = I$ ), and in the case of Fig. 2.3b the error string and correction will be equivalent to a plaquette stabiliser generator, so will have no impact on our logical state.



**Figure 2.3:** Two distinct error strings that result in the same excitations on the surface code. Errored qubits are grey with the error string highlighted in brown, and red diamonds represent excitations.

To give a further example, consider the error string in Fig. 2.4a. This can be corrected by performing  $Z$  gates on the exact error locations, as shown in Fig. 2.4b. Another valid correction string would be to apply  $Z$  gates to the locations shown in Fig. 2.4c. In this case, the error and correction strings combine to leave a closed loop of  $Z$  operations that is equivalent to the product of the shaded plaquette stabiliser generators. However, Fig. 2.4d shows a correction that combines with the error in

a way that is equivalent to a logical operation — applying such a correction would cause an undetectable logical error and likely result in a failed computation.



**Figure 2.4:** Examples of successful and unsuccessful error correction strings on the planar code. (a) An example error string (brown) (errored qubits are grey, excitations are red diamonds). (b) The error in (a) can be corrected by applying  $Z$  gates to each of the errored qubits (purple). (c) The error can also be corrected by forming a string of errors and corrections that is equivalent to a product of stabiliser generators (shaded in pink). (d) shows a correction string that removes the excitations but is not equivalent to a product of stabiliser generators — it is equivalent to a logical operator and would therefore cause a logical error.

## Homology classes

The error and correction strings in Fig. 2.4 demonstrate the concept of *homology classes*. With respect to the surface code, a homology class is a set of operations

that are equivalent up to multiplication by stabiliser elements, i.e. they all have the same effect on the encoded logical state.

Fig. 2.4c gives a combined error and correction string that is a product of stabiliser generators and does not cause a logical error — this string is *contractible* and is in the same homology class as the logical identity operator. Any product of stabiliser generators (including simultaneous products of both plaquette and vertex generators) is in the same homology class as the identity operator and therefore causes no excitations or logical errors.

The string in Fig. 2.4d is an *uncontractible* string in a different homology class to the identity operator and therefore leads to no excitations but causes an undetectable logical error, in this case a logical  $Z$  error. Any undetectable string of errors from one side of the lattice to the opposing side is in a different homology class to the identity operator and acts as a logical operator. There are four distinct homology classes on the planar lattice in Fig. 2.1: the logical identity (consisting of contractible strings of Pauli operations that are equivalent to products of stabiliser generators), logical  $Z$  (consisting of strings of Pauli operations from left to right on the primal lattice), logical  $X$  (consisting of strings of Pauli operations from top to bottom on the dual lattice) and logical  $Y$  (strings that are a combination of logical  $X$  and  $Z$  operations).

## Decoding and correction

After obtaining the syndrome measurements, the values are passed to a *decoder*, which is a classical algorithm that finds a correction. When correcting errors, we want to find the correction that is most likely to result in strings in the same homology class as the identity operator — a decoder that performs this task perfectly is known as a maximum-likelihood decoder. However, efficient algorithms for maximum-likelihood decoding are only known for specific circumstances [Bravyi et al., 2014], so alternative algorithms must be used to instead find a good approximation to this optimal correction.

Numerous approaches have been suggested for efficient surface code decoders, including minimum weight perfect matching decoders [Dennis et al., 2002], renor-

malization decoders [Duclos-Cianci and Poulin, 2010] and cellular automaton decoders [Herold et al., 2015]. Decoders based on minimum weight perfect matching [Dennis et al., 2002] are the most widely used for the surface code as they achieve relatively high error thresholds. Exact error thresholds vary depending on the error model and assumptions — [Stephens, 2014] provides thresholds for several scenarios not specific to any particular implementation, with values varying from around 0.5% to around 1%. Minimum weight perfect matching decoders work by pairing the ends of error strings in a way that minimises the total weight of the correction strings, with each edge of the surface code having a weight assigned to it related to the probability of an error occurring at that location.

In addition to errors occurring on data qubits, the syndrome measurement process itself is prone to error, which could lead to an incorrect (false) error detection event. Therefore, multiple rounds of syndrome measurement, of order  $O(L)$  [Wang et al., 2003], are needed to improve confidence in the syndrome outcomes. The repeated rounds of syndrome measurements create a three-dimensional syndrome history that can be processed by the decoder to return a correction on the physical lattice.

### Other types of Pauli error

We have focused on detecting and correcting Pauli  $Z$  errors on the primal lattice in this section. Pauli  $X$  errors are detected and corrected using an analogous procedure on the dual lattice. Pauli  $Y$  errors are equivalent to an  $X$  and a  $Z$  error occurring on the same qubit, so these affect both the primal and dual lattices; decoding methods that take account of correlations between  $X$  and  $Z$  errors due to  $Y$  errors can be used to maximise error thresholds [Fowler, 2013b].

### 2.2.2 Multiple logical qubits and computation

An attractive feature of the surface code is that error correction only requires fixed-weight local operators, i.e. checks only require interactions between four neighbouring data qubits regardless of the size of the code. However, the planar code only encodes one logical qubit, so performing computations involving more than two

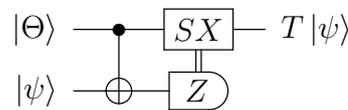
logical qubits requires interactions between multiple planes, which will either require a three-dimensional arrangement or interactions between far-separated qubits. These problems can be overcome by using codes similar to the planar code but with distinct *rough* and *smooth* defects 'punched' into them: these are regions where stabiliser generators are not measured or enforced. Single and multiple qubit logical operations, such as the logical Pauli operators, logical Hadamard and logical CNOT, can then be performed fault-tolerantly by combinations of string operators, transversal gates and braiding of defects [Raussendorf and Harrington, 2007, Fowler et al., 2009].

Non-Clifford gates, such as the  $T$  gate, cannot be performed directly in the surface code, however, it is possible to perform a  $T$  gate indirectly by preparing many noisy copies of the state

$$|\Theta\rangle = \frac{1}{\sqrt{2}} (|0\rangle + e^{i\pi/4} |1\rangle), \quad (2.7)$$

each with some error probability  $p$ , and then using a process known as *magic state distillation* [Bravyi and Kitaev, 2005] to produce fewer copies of  $|\Theta\rangle$ , but each with a reduced probability of error [Raussendorf and Harrington, 2007]. Note that the state  $|\Theta\rangle$  cannot be prepared using Clifford gates alone, so it must be prepared in a non-fault-tolerant manner, hence the use of 'noisy' copies of the state.

Once the probability of error has been sufficiently reduced, a  $T$  gate can be implemented using the circuit shown in Fig. 2.5. This process therefore requires more time and qubits than Clifford gates.



**Figure 2.5:** Circuit for implementing a  $T$  gate. This process uses only Clifford gates and a pre-prepared state  $|\Theta\rangle = (|0\rangle + e^{i\pi/4} |1\rangle)/\sqrt{2}$ .  $|\psi\rangle$  is an arbitrary logical state and  $S$  is the logical  $S$  gate (not an element of the stabiliser group). The gate  $SX$  is classically controlled by the outcome of the  $Z$  measurement.

In addition to defect-based computation, two other surface code proposals for fault-tolerant universal quantum computation are lattice surgery [Horsman et al.,

2012], which involves interacting logical qubits encoded in separate planar codes by fusing their boundaries, and twists [Bombin, 2010, Hastings and Geller, 2015], which are similar to defects but deform pairs of stabiliser generators to create logical qubits.

## **Part II**

# **Fault-tolerant quantum computation with realistic error models**

## Chapter 3

# A blueprint for fault-tolerant quantum computation with Rydberg atoms

Rydberg atoms are a promising candidate for quantum computation [Saffman et al., 2010], having desirable properties such as relatively simple entangling gates between many qubits and the ability to fit thousands of qubits into a very small footprint.

Although there has been much interest in using Rydberg atoms for quantum computation, including a recent 51-qubit quantum simulator [Bernien et al., 2017], very little work has considered the steps required to build a fault-tolerant quantum computer with Rydberg atoms. Previous work on error correction has been limited to [Brion et al., 2008], which considered error correction within an ensemble of atoms representing a single qubit, [Crow et al., 2016], which focused on using Rydberg atoms in measurement-free error correction schemes, and [Isenhower et al., 2011], which investigated error rates for multiple-controlled CNOT gates using Rydberg atoms. Additionally, methods for building a universal quantum computer with Rydberg atoms using a decoherence-free subspace to mitigate the effects of errors were suggested in [Brion et al., 2007].

In this chapter, we propose a Rydberg atom scheme for performing fault-tolerant quantum computation with the surface code. Ideas for a Rydberg atom based quantum simulator using the toric code were considered in [Weimer et al.,

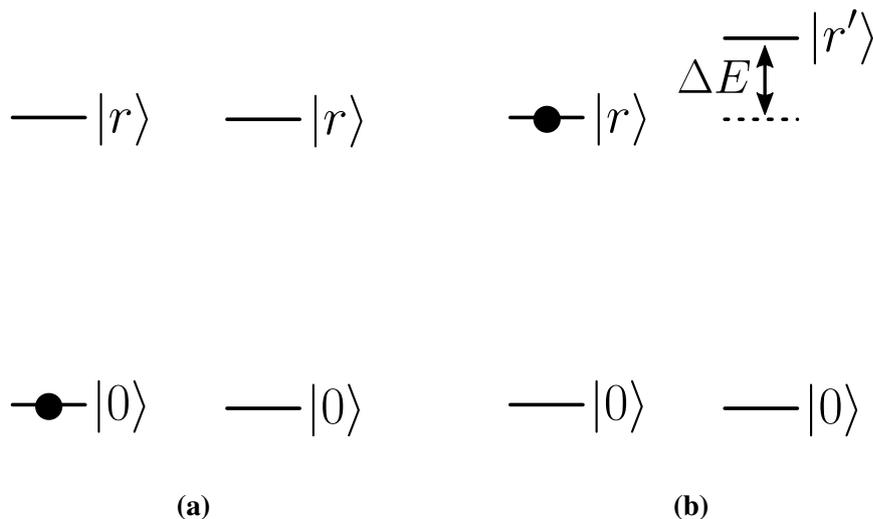
2010, Weimer et al., 2011], but the work in this chapter goes beyond this to consider some of the steps required to build a fully-fledged universal Rydberg atom quantum computer with active error correction. Section 3.1 provides a brief introduction to Rydberg atoms, and Sections 3.2 and 3.3 provide details of the scheme and error threshold simulations respectively.

### 3.1 Quantum computation with Rydberg atoms

Rydberg atoms are neutral atoms with one or more electrons in a highly-excited state, i.e. with principal quantum number  $n \gg 1$  — the alkali metals, particularly rubidium and caesium, are the species of atoms most commonly used for Rydberg atom experiments due to their single valence electrons. One of their most useful features for quantum computation is the dipole blockade, which facilitates the implementation of entangling gates between multiple atoms.

The dipole blockade effect is shown in Fig. 3.1. When two neighbouring neutral atoms are in their ground states with separation  $R$ , the energy required to excite one of them to a particular Rydberg state  $|r\rangle$  is  $E_r$ . However, once one atom is in its Rydberg state, the energy required to excite the neighbouring atom to the state  $|r\rangle$  is increased to  $E_r + \Delta E(R)$ ; exciting one atom to its Rydberg state effectively *blockades* the other. This effect can therefore be used to implement entangling gates by utilising Rydberg states to mediate interactions between two or more atoms; one method for this, based around electromagnetically induced transparency, is introduced in Section 3.2.

The size of this energy shift  $\Delta E(R)$  generally falls into one of two regimes: when the atoms are sufficiently close, the dominant interaction is due to the resonant dipole-dipole interaction, which scales as  $\Delta E(R) \sim 1/R^3$ . When the atoms are sufficiently distant from each other, the dominant interaction becomes the van der Waals interaction, which scales as  $\Delta E(R) \sim 1/R^6$ . We will favour the van der Waals regime for our scheme due to the faster decay in interaction strength, which helps to reduce unwanted interactions between distant atoms. The van der Waals regime occurs for atomic separations of around 5-50  $\mu m$  [Saffman et al., 2010] for



**Figure 3.1:** The Rydberg atom dipole blockade. (a) shows the energy levels of a pair of atoms where neither occupies a Rydberg state, and (b) shows the effective energy levels when the atom on the left occupies the Rydberg state  $|r\rangle$ : the Rydberg level of the atom on the right is shifted to  $|r'\rangle$ .

rubidium 87, although exact values depend upon atom species and which Rydberg state is used.

The interaction strength in the van der Waals regime is given by

$$\Delta E(R) = \frac{C_6}{R^6}, \quad (3.1)$$

where the coefficient  $C_6$  scales with  $n^{11}$  and depends on the types and states of the atoms involved — values were calculated for a variety of configurations in [Singer et al., 2005].

Experimentally, single-qubit gate fidelities in excess of 99% have been demonstrated [Xia et al., 2015, Wang et al., 2016], but two-qubit gates are languishing behind, with the best experiments achieving fidelities of around 80% when post-selecting for qubit loss [Jau et al., 2015, Maller et al., 2015]; it is to be noted that this is due to technical limitations rather than a fundamental limit. For a recent summary of the state of Rydberg atom experiments, we direct the reader to [Saffman, 2016].

## 3.2 Proposed scheme

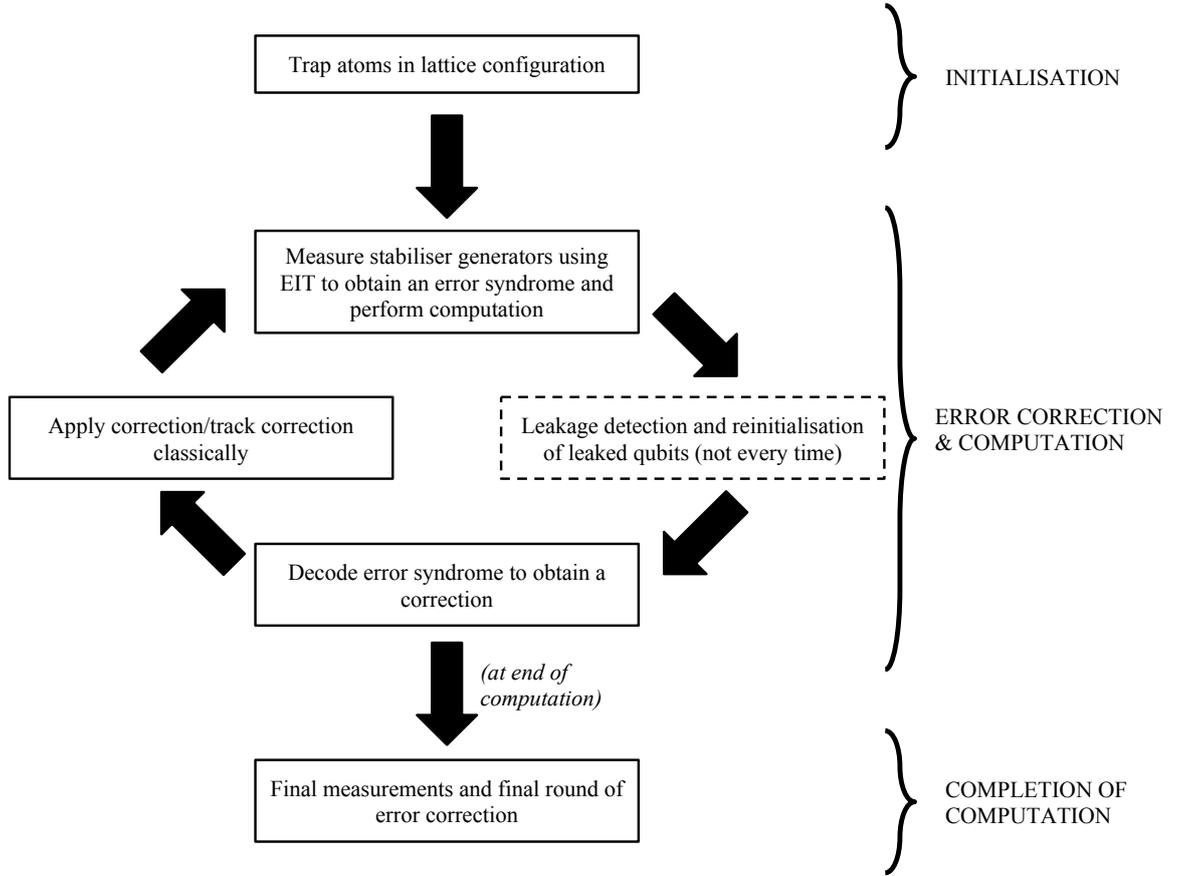
Our scheme involves using individually-addressable optically-trapped neutral atoms to represent qubits in a planar code, with quantum computation achieved either by braiding defects or lattice surgery (see Section 2.2.2). The planar code has been chosen as it has a comparatively high error threshold and requires qubits to be trapped in a relatively simple lattice geometry with interactions between a small number of neighbouring physical qubits regardless of code size, which is a desirable property for scalability and is not generally the case for concatenated codes. The  $|0\rangle$  and  $|1\rangle$  states of each physical qubit are represented by hyperfine ground states of the atoms, and Rydberg states, labelled  $|r\rangle$ , are used to mediate interactions. Note that atoms involved in an interaction may utilise different Rydberg states such that  $|r\rangle_i$  and  $|r\rangle_j$  are not necessarily the same states for atoms  $i$  and  $j$ .

Multi-qubit gates performed by exploiting electromagnetically induced transparency (EIT) using the methods in [Müller et al., 2009]. This approach has several desirable features, including parallel operation, the ability to activate local interactions with large contrast via laser addressing as needed, and robustness towards interactions between target atoms, provided that Rabi frequencies and interaction strengths involved (i.e. Rydberg states) are chosen appropriately [Müller et al., 2009].

An overview of the scheme is shown in Fig. 3.2.

### 3.2.1 EIT gates

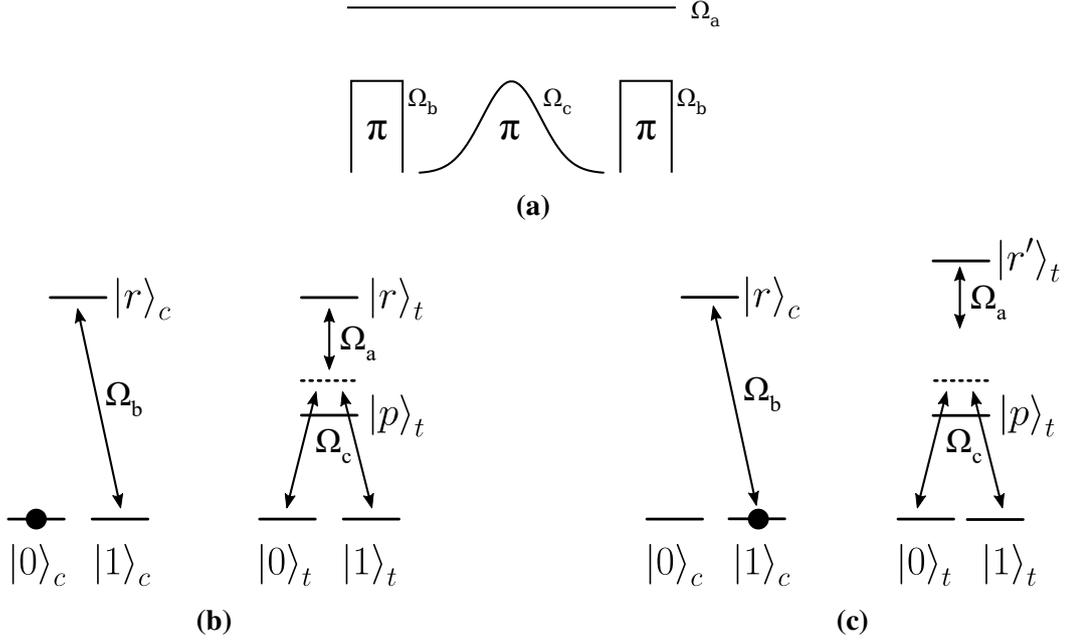
Fig. 3.3 shows the process for using EIT to perform a CNOT gate between a control and a target qubit, as proposed in [Müller et al., 2009]. Initially, the  $|1\rangle_c$  state of the control atom is resonantly coupled to the  $|r\rangle_c$  state using a  $\pi$  laser pulse with Rabi frequency  $\Omega_b$ . A second  $\pi$  pulse with Rabi frequency  $\Omega_c$  is then used to off-resonantly couple the  $|0\rangle_t$  and  $|1\rangle_t$  states of the target atom via an off-resonantly coupled intermediate state  $|p\rangle_t$ , before another  $\pi$  pulse with frequency  $\Omega_b$  is applied to again couple the  $|1\rangle_c$  and  $|r\rangle_c$  states of the control atom. Throughout this process, a strong laser with Rabi frequency  $\Omega_a$ , where  $\Omega_a \gg \Omega_c$ , is used to off-resonantly couple the Rydberg state  $|r\rangle_t$  of the target to the intermediate state  $|p\rangle_t$  and achieve



**Figure 3.2:** An overview of our proposed scheme for quantum computation using Rydberg atoms. Error correction and computation are performed simultaneously, as computation by lattice surgery or braiding defects are achieved by disabling specific check operators at set times during the error correction procedure, so there is no explicit computation step.

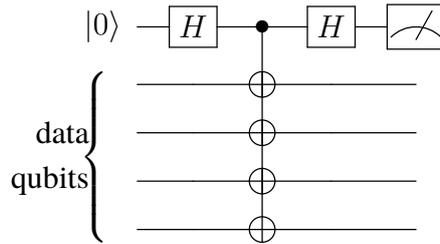
EIT. When the control atom starts in the  $|0\rangle_c$  state, the initial  $\Omega_b$  pulse has no effect and the beam  $\Omega_a$  prevents Raman transfer between the  $|0\rangle_t$  and  $|1\rangle_t$  states on the target due to EIT. When the control atom starts in the  $|1\rangle_c$  state, the  $|r\rangle_c$  state of the control atom becomes populated after the first  $\Omega_b$  pulse, which in turn shifts the Rydberg state  $|r\rangle_t \mapsto |r'\rangle_t$  of the target atom, which takes the  $\Omega_c$  beam out of resonance and removes the EIT condition on the target, leading to an effective coupling between the  $|0\rangle_t$  and  $|1\rangle_t$  states.

This method can be used to perform simultaneous CNOT gates between a single control qubit and multiple target qubits, making it ideal for syndrome measurement in the surface code using the measurement circuit shown in Fig. 3.4 — every star and



**Figure 3.3:** Using EIT to perform a CNOT gate with the method in [Müller et al., 2009] (control qubit always on the left, target qubit always on the right). (a) shows the order of the pulses, (b) shows EIT blocking the  $|0\rangle_t \leftrightarrow |1\rangle_t$  transition on the target qubit and (c) shows the dipole blockade shifting the EIT out of resonance and allowing the  $|0\rangle_t \leftrightarrow |1\rangle_t$  transition on the target qubit.

plaquette has an associated syndrome qubit used to measure the stabiliser generators. The  $\Omega_c$  pulse is of the order of a few 10s of MHz such that the multi-qubit interaction can be performed in under a millisecond [Müller et al., 2009].



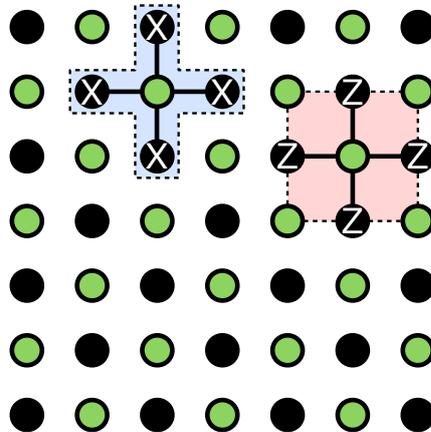
**Figure 3.4:** Measuring a star operator using a multi-target EIT CNOT gate. The top qubit is the ancilla syndrome qubit and is used only for syndrome measurement. The method for measuring a plaquette stabiliser generator involves applying Hadamard gates to each data qubit before and after the entangling operation but is otherwise identical.

The fidelity of the process is dependent upon the chosen atom species, Rabi frequencies and Rydberg states, but to give an indication, [Müller et al., 2009] calculated that EIT can be used to perform the operation  $|+000\rangle \mapsto 1/\sqrt{2}(|0000\rangle +$

$|1111\rangle$ ) with a fidelity in excess of 97% with  $^{87}\text{Rb}$ . Higher fidelities may be achieved by an appropriate choice of the laser parameters and Rydberg states, as discussed in [MacCormick et al., 2016] and [Mansell and Bergamini, 2014], where the gating parameters were optimised for different spatial arrangements of the target qubits. Fidelities alone don't provide details of the underlying error channels so cannot be mapped to error correction thresholds: leakage errors, for example, can be less harmful than Pauli errors [Suchara et al., 2015].

### 3.2.2 Trapping atoms

Our proposal requires that atoms be trapped in a lattice configuration like that shown in Fig. 3.5. Deterministic loading of traps remains a major hurdle for Rydberg atom quantum computation, but methods to overcome this have been suggested, including starting with a partially loaded lattice and rearranging the qubits [Weiss et al., 2004] — this approach has been successfully used to construct 2D lattice geometries of  $\sim 50$  qubits [Barredo et al., 2016] with atomic separations of a few  $\mu\text{m}$  using optical tweezers, which would be sufficient for a prototype device.



**Figure 3.5:** Arrangement of Rydberg atoms for a planar code. Green shaded qubits denote ancilla syndrome qubits used to measure stabiliser generators, and solid black qubits denote data qubits of the planar code. Using different species of atoms for syndrome and data qubits may help to reduce crosstalk during measurement [Beterov and Saffman, 2015].

It should be noted that it is not necessary to construct a perfect lattice of qubits: we will show in Chapter 4 that low rates of missing syndrome and data qubits can be tolerated without requiring any additional quantum processing, at the cost of a

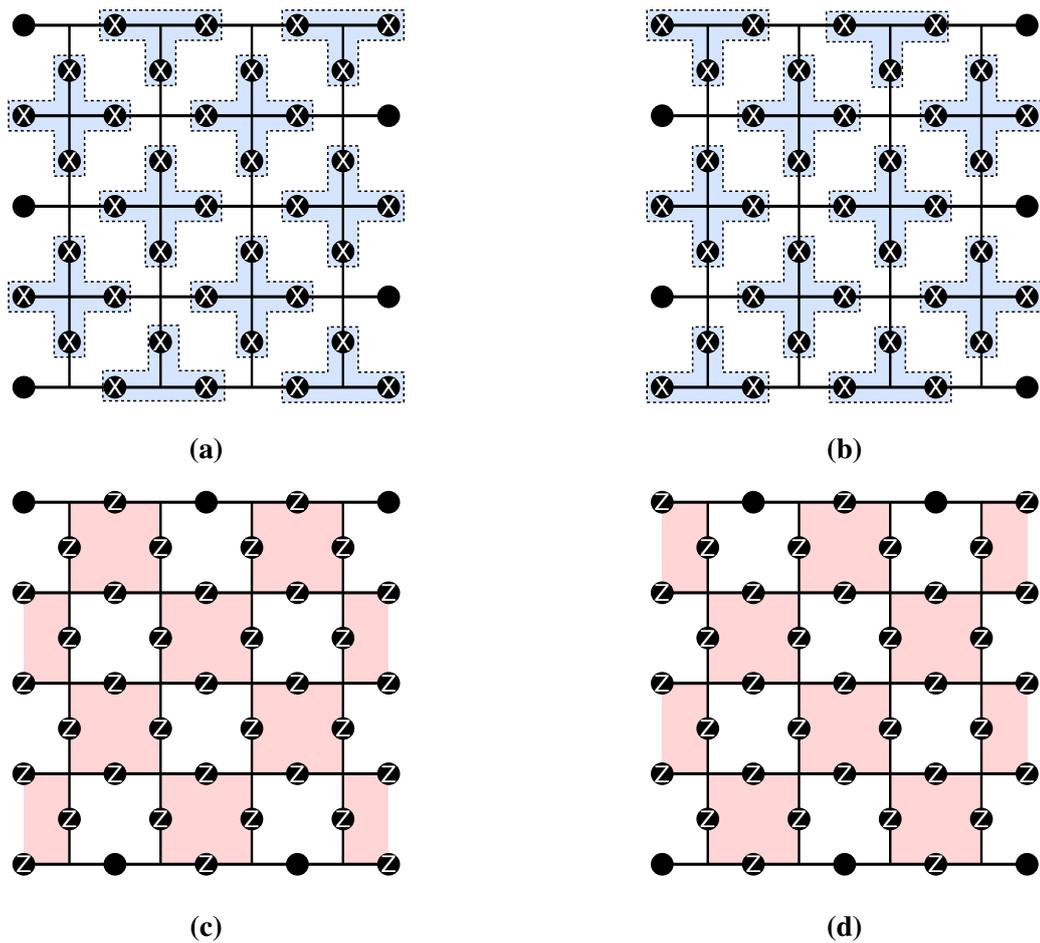
reduction in the Pauli error threshold.

### 3.2.3 Error correction and computation

Once the atoms are trapped, error correction proceeds by repeatedly measuring the stabiliser generators of the surface code. For example, to measure a star stabiliser generator, the syndrome qubit is prepared in the  $|+\rangle$  state, and then the EIT gate method is used to apply simultaneous CNOT gates controlled by the associated syndrome qubit, with the four surrounding data qubits as targets. A Hadamard gate is then applied to the syndrome qubit before it is measured in the computational basis. This process is shown in Fig. 3.4. Measurement of plaquette stabiliser generators is performed in the same manner, but with Hadamard gates applied to each data qubit before and after the CNOT gates.

Each data qubit can only be involved in star or plaquette checks at any time, so measuring a full set of checks will require at least two stages of measurement: one for stars and one for plaquettes. However, measuring all star or plaquette operators simultaneously is likely to lead to strong unwanted crosstalk interactions between qubits involved in neighbouring checks due to the  $1/r^6$  nature of the van der Waals interaction, which would lead to unreliable measurement outcomes and could potentially propagate errors across the lattice. We therefore propose measuring the check operators in at least four separate stages, as shown in Fig. 3.6. It should be noted that because of the scaling of the van der Waals interaction with distance, the number of staggered measurement stages may need to be increased further to avoid crosstalk. This staggered measurement pattern should not impact the overall speed of the computation significantly, as the readout stage is several orders of magnitude slower than the interaction stage, and the actual readout from the syndrome qubits can be performed simultaneously. Should measurement speed be increased, then the choice of measurement pattern will depend upon the trade-off between errors accumulating due to the delay between stabiliser generator measurements and errors occurring due to crosstalk during the EIT gates.

Table 3.1 shows how the strength of crosstalk interactions scales with the number of measurement stages. The strength of crosstalk interactions depends on



**Figure 3.6:** Stabiliser measurement pattern. Stabiliser generators are measured in at least four stages to ensure that each data qubit is only involved in a single interaction at any time in order to reduce crosstalk. Each of the subfigures represents one stage of measurement, with all four stages required for one complete round of syndrome measurement. (a) and (b) show measurement of star stabiliser generators, and (c) and (d) show measurement of plaquette stabiliser generators.

the shortest distance between syndrome qubits and data qubits of different stabiliser generators that are being measured, as this will be the strongest unwanted van der Waals interaction. Measuring in four stages using the measurement pattern in Fig. 3.6 results in the crosstalk interactions being  $\sim 10^2$  times weaker than the desired interactions within stabiliser generators (i.e. the CNOT or CPHASE gates). Increasing the number of measurement stages to 8 and 16 steps reduces the strength of the crosstalk interactions to  $\sim 10^3$  times and  $\sim 10^4$  times weaker than the stabiliser generator interactions respectively; this rapid decay in crosstalk interaction strength is due to the  $1/r^6$  scaling of the van der Waals interaction strength. This analysis is

only a rough indicator of crosstalk strength as it doesn't take account of additional factors such as how many qubits crosstalk is occurring between or the strength of the next-strongest crosstalk interaction. However, it indicates that increasing the number of stages of staggered measurements should be effective at suppressing errors caused by crosstalk if necessary.

Number of measurement stages	Shortest distance, $d_{min}$ , between syndrome and data qubits of different active stabiliser generators	Ratio between check operation strength and crosstalk strength ( $d_{min}^6/r^6$ )
4	$\sqrt{5}r$	125
8	$3r$	729
16	$5r$	15625

**Table 3.1:** Crosstalk interaction strength. Here  $r$  is the distance between syndrome and data qubits that are in the same stabiliser generator, i.e. the interaction distance for the desired van der Waals interactions.

The time taken to perform a single EIT gate is on the order of  $1 \mu s$  [Müller et al., 2009], whereas the lifetime of the Rydberg state of a rubidium or caesium atom is on the order of  $100 \mu s$  [Beterov et al., 2009]. The control atom only needs to be in the Rydberg state while the gate is being performed and not while it is being measured, so spontaneous decay from the Rydberg state during the interaction will have a small contribution to the overall error rate.

Fast, high-fidelity measurement is another outstanding challenge for Rydberg atom devices. Quantum nondemolition measurements with arrays of qubits have only been performed using relatively noisy electron-multiplying CCDs [Alberti et al., 2016] rather than discrete photon detectors. Such measurements take around  $20 \text{ ms}$  [Martinez Dorantes, 2016], so this is currently the limiting factor for the clock speed of our scheme and will limit the computation speed to frequencies on the order of  $10 \text{ Hz}$  until improvements are made. As a comparison, the networked trapping scheme in [Nickerson et al., 2014] would be capable of kilohertz frequencies, so this is an important bottleneck to overcome if this Rydberg atom scheme is to provide a competitive clock speed. Crosstalk during measurement poses an additional problem, although suggestions for reducing crosstalk by using a two-

species architecture [Beterov and Saffman, 2015] would be ideally suited to a surface code quantum computer, where rubidium atoms could be used for the frequently-measured syndrome qubits and caesium atoms could be used for the data qubits.

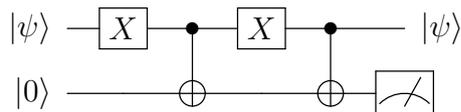
In addition to syndrome measurement, performing full quantum computation by braiding defects or lattice surgery will require the ability to measure individual data qubits occasionally. This could be achieved by using CNOT gates and measuring syndrome qubits, therefore removing the requirement to be able to directly measure the data qubits.

### 3.2.4 Leakage and loss

Atoms are non-binary systems and we are making extensive use of non-qubit Rydberg states of atoms, so there is a risk of leakage errors occurring, where the atom leaves the computational basis. Atom traps are also imperfect and prone to interference from external sources, so in addition to leakage, atoms may be lost. It is therefore prudent to include some form of leakage and loss detection and reduction. As the effects of leakage and loss are similar, we will use the term leakage to refer to both types of error.

Methods for dealing with leakage in the surface code were considered in [Suchara et al., 2015], which showed that low levels of leakage can be tolerated at the cost of a lower Pauli error threshold. When a qubit leaks, this qubit can be reset to a known state, e.g.  $|0\rangle$ , and the error correction can proceed as normal, with the decoder taking account of the increased probability of an error occurring on the leaked qubit.

Leakage can be detected by using the circuit in Fig. 3.7 periodically [Preskill, 1997]; the syndrome qubits can play the role of the ancilla qubits required by this method. The frequency with which leakage detection needs to be performed will depend on the rate at which leakage errors occur; leakage detection will introduce additional errors so will ideally be performed as infrequently as possible.



**Figure 3.7:** Leakage detection circuit from [Preskill, 1997]. If the top qubit is in the computational basis, then the bottom qubit (an ancilla) will be observed in the  $|1\rangle$  state. If the top qubit has leaked or been lost, the bottom qubit will be observed in the  $|0\rangle$  state. The top qubit can be reinitialised if leakage or loss occur. In our scheme, the syndrome qubits can be used as the ancilla qubits.

### 3.3 Threshold simulations

We have performed a simulation of this scheme to obtain an error correction threshold. The simulation uses the planar code as a quantum memory, and simulates stabiliser generator measurements, as outlined in Section 3.2, in the presence of errors.

#### 3.3.1 Error model

The error model used in the simulation is based around a single error parameter  $p$  and is summarised in Table 3.2. State preparation and measurement are assumed to result in the preparation or detection of orthogonal states respectively with probability  $p$ . Each multi-qubit EIT gate is modelled to act perfectly followed by depolarising noise with probability  $p$ , i.e. for an  $n$  qubit gate, each of the possible  $4^n - 1$  non-identity Pauli operations will occur with probability  $p/(4^n - 1)$ . Single qubit gates, such as identity gates and Hadamard gates, are assumed to be free from error on the basis that such operations will generally have much lower error rates than other operations.

Operation	Error model
State preparation	Orthogonal state prepared with probability $p$
Single-qubit gate	Free from error
$n$ -qubit gate, $n > 1$	One of the $4^n - 1$ non-identity Pauli operators is applied with probability $p$ (each Pauli operator is equally likely)
State measurement	Incorrect measurement outcome with probability $p$

**Table 3.2:** Error model for Rydberg atom scheme simulations.

As mentioned in Section 2.1, measuring the stabiliser generators leads to a discretisation of errors: a more general error will collapse into a combination of

Pauli errors. It is non-trivial to perform a simulation with physical errors or to directly relate physical errors to a simulatable error model. This error model therefore has been chosen in lieu of knowledge of the exact error channels and associated Pauli error rates, as is standard when obtaining quantum error correction thresholds; this allows for a comparison with thresholds obtained for other approaches, such as those in [Stephens, 2014].

Leakage errors, such as atom loss or excitation of unintended energy levels, were not considered in the simulation — such a simulation is left for future work.

### 3.3.2 Simulation methods

Planar codes with code distance  $d = 8, 10, 12$  and  $14$  were simulated for  $2d$  rounds of syndrome measurement using the scheme outlined in Sec. 3.2 and the above error model, and a minimum weight perfect matching algorithm based on Blossom V [Kolmogorov, 2009] was used for decoding. Fig. 3.8 details the steps performed by the simulation.

After preliminary simulations to determine an approximate threshold, final results were obtained by varying Pauli error rates from 1% to 1.5% in steps of 0.05%, and each combination of code distance and Pauli error rate was simulated  $10^5$  times — this number was found to be sufficient to give clean results.

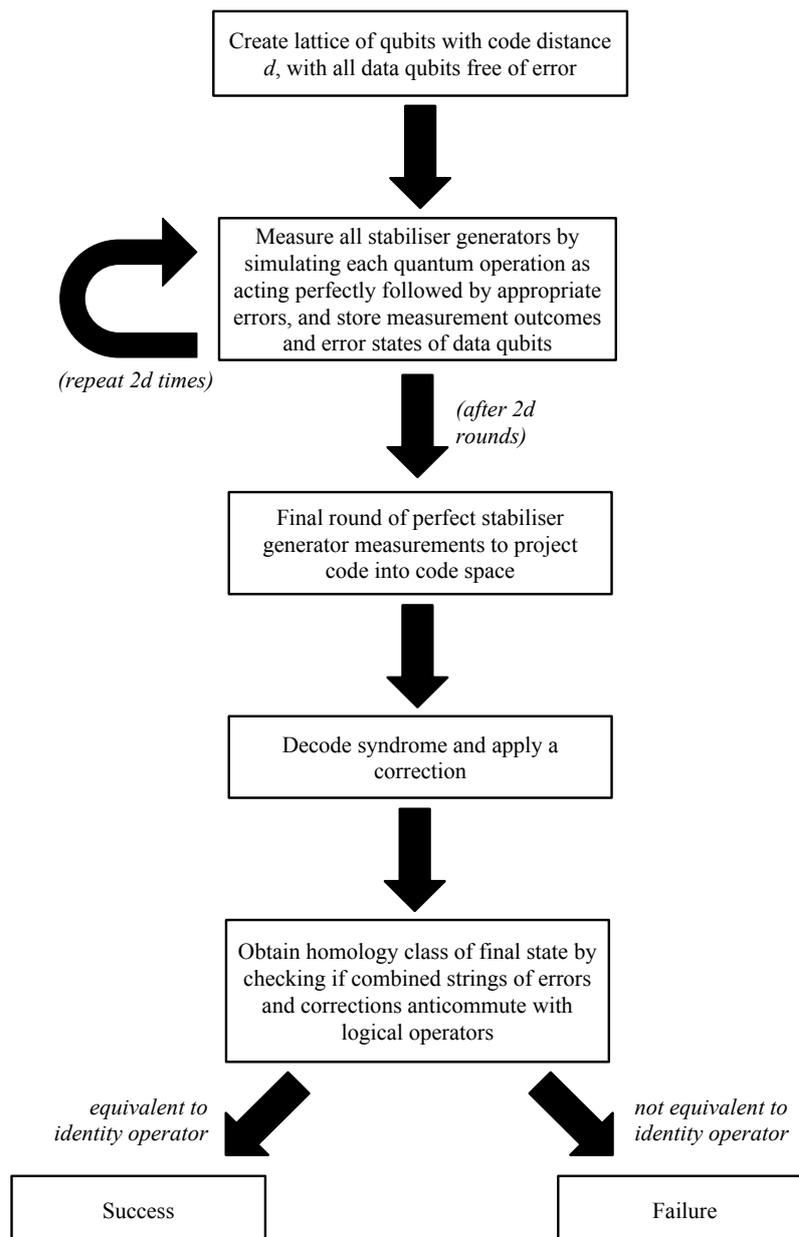
## 3.4 Results and discussion

The results obtained from the simulations are shown in Fig. 3.9. This figure shows the logical error rates obtained for each configuration of code distance and Pauli error rate, calculated by

$$p_{\text{logical}}(d, p) = \frac{\#_{\text{failures}}(d, p)}{\#_{\text{runs}}(d, p)}, \quad (3.2)$$

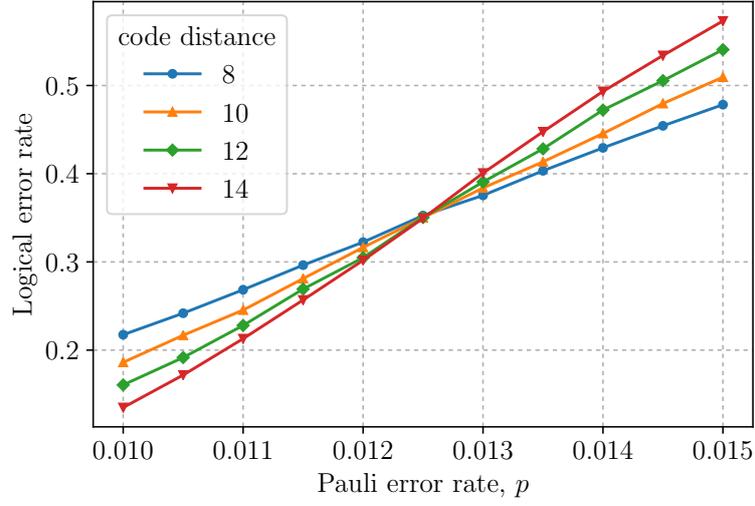
where  $\#_{\text{failures}}$  is the number of failures and  $\#_{\text{runs}}$  is the number of runs.

The error threshold is given by the point where the logical error rates for different code distances intersect [Dennis et al., 2002, Wang et al., 2003], resulting in a threshold of  $p_{th} \approx 1.25\%$  for our scheme with the chosen error model.



**Figure 3.8:** Steps for simulating error correction with Rydberg atoms.

As mentioned in the previous section, it is non-trivial to directly link Pauli error thresholds and gate fidelities — to do so requires further investigation, such as performing process tomography experiments to determine the underlying error channel and associated error rates. However, we can use the threshold we have obtained to calculate the fidelity for a particular quantum state in the idealistic scenario where our error model corresponds to the actual error channel of the gate.



**Figure 3.9:** Logical error rates from error simulations for code distances 8, 10, 12 and 14. The crossing point gives the resulting error threshold of 1.25%.

In this case, our chosen error channel on a pure quantum state  $|\psi\rangle$ ,  $\varepsilon(|\psi\rangle)$ , is [Nielsen and Chuang, 2000]

$$\varepsilon(|\psi\rangle) = (1 - p)I |\psi\rangle \langle\psi| I^\dagger + \frac{p}{4^n - 1} \sum_{P_i \in \mathcal{P}_n, P_i \neq I} P_i |\psi\rangle \langle\psi| P_i^\dagger, \quad (3.3)$$

where  $n$  is the number of qubits.

We can then calculate the fidelity,  $F(|\psi\rangle, \varepsilon(|\psi\rangle))$  using [Nielsen and Chuang, 2000]

$$F(|\psi\rangle, \varepsilon(|\psi\rangle)) = \sqrt{\langle\psi| \varepsilon(|\psi\rangle) |\psi\rangle}. \quad (3.4)$$

Working through this equation using the state  $|\psi\rangle = |++++\rangle$  as an example, and noting that  $P_i^\dagger = P_i$ , we have

$$F(|\psi\rangle, \varepsilon(|\psi\rangle))^2 = (1 - p) \langle\psi| I |\psi\rangle^2 + \frac{p}{4^5 - 1} \sum_{P_i \in \mathcal{P}_5, P_i \neq I} \langle\psi| P_i |\psi\rangle^2 \quad (3.5)$$

$$= 1 - p + \frac{p}{4^5 - 1} (2^5 - 1) \quad (3.6)$$

$$= 1 - \frac{32}{33}p, \quad (3.7)$$

where we arrived at the second line by noticing that  $|\psi\rangle$  is stabilised by  $2^5 - 1$  five-qubit Pauli operators (excluding the identity), where  $\langle\psi|P_i|\psi\rangle^2 = 1$ , and the remaining Pauli operators are not stabiliser operators of  $|\psi\rangle$  and therefore result in  $\langle\psi|P_i|\psi\rangle^2 = 0$ .

This means that our threshold value of  $p \approx 1.25\%$  corresponds to a fidelity of  $F \approx 99.4\%$  for a five-qubit acting gate on the state  $|\psi\rangle$ . Caution must be taken when drawing conclusions from this figure, but such fidelities are currently well beyond what has been achieved experimentally with multi-qubit Rydberg atom gates. However, as mentioned in Section 3.2.1, other types of error, such as leakage, can be less damaging than depolarising Pauli noise, so the required fidelity may be lower for processes involving such errors.

### 3.4.1 Correlated errors

The van der Waals interaction between atoms scales with  $1/R^6$ , where  $R$  is the separation between atoms. This polynomial decay means that there may be non-negligible crosstalk between distant qubits during multi-qubit gates, which could cause correlated errors between non-neighbouring qubits. Similar errors were considered for the surface code in [Fowler and Martinis, 2014], which found that logical error suppression could be achieved even in the extreme case of quadratically decaying interactions.

## 3.5 Conclusion

In this chapter, we have proposed a new scheme for fault-tolerant quantum computation with Rydberg atoms. Our proposal uses electromagnetically induced transparency to perform multi-qubit gates for syndrome extraction, and we have suggested methods to mitigate the effects of leakage and qubit loss. We have found a threshold of 1.25% for an error model based on this scheme, which we hope will provide an initial target for experimentalists looking to build a prototype quantum computer with Rydberg atoms. Prospects for initial scalability are good, with arrays with on the order of  $10^4$  atoms being realistically achievable [Saffman, 2016]. Larger numbers of qubits would be desirable in the long run, but this should be satisfactory

for early devices attempting to demonstrate quantum speedup.

Experimental results from a recent 51-qubit quantum simulator based on Rydberg atoms [Bernien et al., 2017] give cause for optimism, but this device falls short of being a fault-tolerant quantum computer. The atoms in this simulator were arranged in a one-dimensional array, and interactions were performed using an Ising-type quantum spin model rather than sequences of circuit model quantum gates. Achieving quantum operations with sufficiently high fidelities and low loss rates remains a challenge for the field, but this is mostly due to engineering obstacles rather than physical limitations, so we are optimistic that large improvements will be made. While quantum gates can be performed at MHz frequencies, slow measurement for arrays of atoms currently limits the potential clock speed of our scheme to the order of a few tens of Hz, so this is a key area for improvement.

A thorough error analysis of multi-qubit gates based around EIT is required to determine whether sufficiently low error rates can be achieved — it is likely that improvements will be needed to achieve the error rates below 1.25% required for reliable quantum computation. If necessary, the EIT gates in our proposal can easily be replaced by another multi-qubit interaction without significantly affecting the rest of the scheme. It should be noted that a threshold alone cannot be used to verify that a scheme will work for surface code based quantum computation — a more convincing analysis is to experimentally demonstrate that a larger system has superior error suppression compared to a smaller system, as has been accomplished with bit-flip errors on superconducting qubits [Kelly et al., 2015].

Our findings suggest that while there are advantageous features of Rydberg atoms, gate fidelities need to be improved before fault-tolerant universal quantum computation can be achieved — experiments based on other implementations of quantum computation, such as superconducting qubits [Barends et al., 2014] and trapped ions [Ballance et al., 2016] are currently ahead of Rydberg atoms. We nonetheless believe Rydberg atoms are a promising candidate for building a scalable fault-tolerant quantum computer.

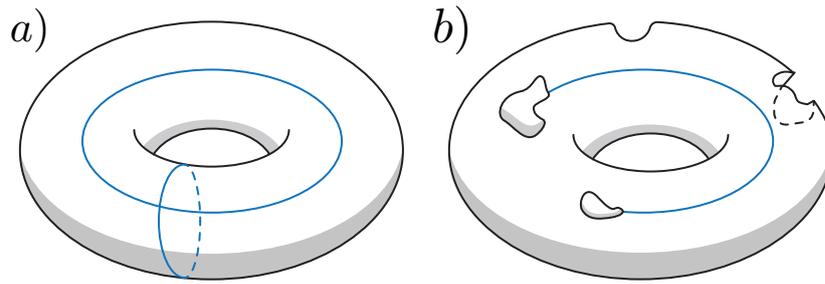
## Chapter 4

# Fault-tolerance thresholds for the surface code with fabrication errors

Implementing topological quantum error correction codes has become the focus of many current experiments, with recent advances made in building prototype devices consisting of a small number of physical qubits with fine-tuned local interactions [Cory et al., 1998, Kelly et al., 2015, Reed et al., 2012, Chiaverini et al., 2004, Jin et al., 2012, Córcoles et al., 2015]. However, it is expected that a universal fault-tolerant topological architecture will have a very large number of physical components, and for such large-scale machines the production and fine-tuning of each individual component will undoubtedly suffer from permanent faults resulting from imperfect manufacturing processes — we refer to such faults as *fabrication errors*. As an example, the latest non-universal D-Wave 2X machine has qubit manufacturing defects (typically fewer than 5%) [King et al., 2015]. Therefore, it is important that the performance of current schemes is studied against such a static error.

The construction of topological codes relies on utilising the non-trivial topology to encode a logical state, such that the encoded state can only be corrupted by global errors. This construction is vulnerable to the effects of fabrication errors by design, as fabrication errors directly damage the topology by introducing new unwanted logical qubits and lowering the code distance for the encoded logical states, see Fig. 4.1.

The threshold performance of the surface code has been extensively studied



**Figure 4.1:** The impact of fabrication errors on the topology of the surface code. The toric code (a surface code on a torus) is used here to depict such errors. (a) The construction of topological codes relies on the fabrication of a perfect topology in order for the encoded logical state to be globally protected. (b) Fabrication errors have the effect of damaging the topology by introducing new degrees of freedom and shortening the distance of the code.

against many noise models, such as for Pauli errors [Wang et al., 2003, Wang et al., 2011, Stephens, 2014], stochastic qubit loss [Stace et al., 2009, Stace and Barrett, 2010, Fujii and Tokunaga, 2012] and leakage [Suchara et al., 2015, Fowler, 2013a, Whiteside and Fowler, 2014, Ghosh and Fowler, 2015, Wood and Gambetta, 2017], but much less focus has been given to errors resulting from imperfect manufacturing processes.

In this chapter, we investigate the threshold performance of the surface code in the presence of fabrication errors, and we show that the ability to disable a qubit or an entangling gate (*link*) is sufficient to map any fabrication error into disabled qubits, hence allowing us to always form larger stabiliser operators, the so-called *supercheck operators* [Stace et al., 2009]. The scheme we present does not require anything in addition to what is already necessary to perform surface code quantum computation by code deformation or lattice surgery. The techniques presented here are most relevant to chip-based topological schemes, such as those using superconducting qubits or quantum dots.

Two recent approaches have been proposed to mitigate the effects of fabrication errors; the first is to construct a robust topology that tolerates sparse fabrication errors using additional sacrificial qubits [Tang and Miao, 2016], and the second is to use primitive SWAP gates in the construction of the syndrome read-out circuit [Nagayama et al., 2017]. Our approach differs from these by keeping the construction

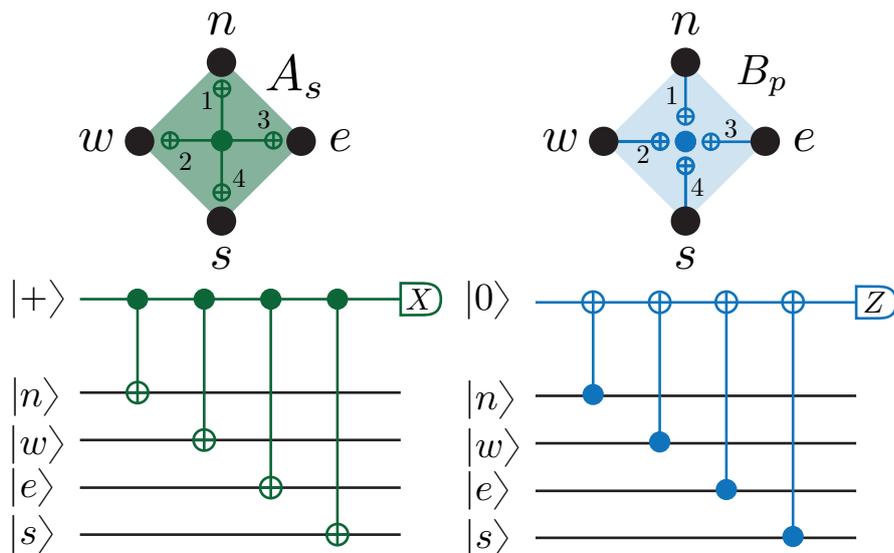
of the surface code unaltered with no additional quantum processing. We show that by directly measuring the defective stabiliser generators as gauge operators, the outcome of the supercheck operators can be obtained deterministically.

This chapter is structured as follows. In Section 4.1 we define different types of fabrication errors and show how to perform the syndrome extraction on a defective lattice by constructing the outcome of supercheck operators from the gauge qubit operators. We outline our noise model and simulation parameters in Section 4.2, and present the fault-tolerant thresholds we obtain in Section 4.3. In the final two sections, we discuss our approach in comparison to other schemes and conclude.

## 4.1 Fabrication errors

The stabiliser generators of the surface code are multi-qubit operators that can be difficult to measure directly, so an ancilla qubit (the syndrome qubit) is associated with each star and each plaquette to assist with the measurement process; each stabiliser generator can then be measured by performing a sequence of two-qubit gates between the ancilla qubits and data qubits, as discussed in Section 2.2. By performing the two-qubit gates in a *z-shaped* order, as shown in Fig. 4.2, all the stabiliser generators can be measured simultaneously in a total of six time-steps: one for syndrome qubit preparation, four for two-qubit gates and one for syndrome qubit measurement (it may be possible to combine measurement and initialisation for some implementations, but we consider them to be separate processes in our simulations). These six time steps constitute one round of syndrome measurement.

We define fabrication errors as permanent faults to components caused during the initial chip-manufacturing process of the surface code, or due to failed components arising during the lifetime of the chip. It is important to emphasise that the locations of the fabrication errors are known before the surface code device is used for computation (i.e. the fabrication errors are known deterministic failed components). In addition, we assume that the user of the surface code chip can turn off any of the components. We consider two types of fabrication error: *qubit fabrication* errors and *link fabrication* errors. A qubit fabrication error is considered to be a



**Figure 4.2:** Syndrome extraction circuits for star (left) and plaquette (right) stabiliser generators. Each syndrome extraction circuit involves six temporal steps: preparation of the syndrome qubit, four CNOT gates, and syndrome qubit measurement. The data qubits are idle during the preparation and measurement steps. The CNOT gates are always applied in a specific order, in this case north ( $n$ ), west ( $w$ ), east ( $e$ ) then south ( $s$ ) (forming a zigzag shape), to ensure all stabiliser generators commute when measured simultaneously.

qubit (either a data qubit or syndrome qubit) that is permanently faulty and cannot be used to store quantum information reliably. A link fabrication error is considered to be an error that prevents two qubits from interacting, i.e. it prevents a CNOT or CPHASE gate from being performed between a syndrome qubit and a data qubit.

Before proceeding, it is useful to introduce some additional terminology to describe the different types of failed components we will encounter in our analysis. We use the term *faulty* to strictly refer to a component with a permanent fabrication error, and the term *disabled* to refer to a component that we have chosen to disable. Moreover, we call a check operator *damaged* if at least one of its four links or data qubits suffers a fabrication error.

Both types of fabrication error are potentially detrimental for the surface code construction and, left unchecked, can introduce new logical qubits and reduce the code distance. For example, if a syndrome qubit is faulty, one might be tempted to simply disable the associated stabiliser generator. But a disabled star or plaquette creates a new logical qubit that can interact with our encoded logical state, therefore

reducing the code distance. A reduction in code distance alone may not be a problem in itself, but if we assume fabrication errors are randomly spaced throughout the code, then the code distance will start to shrink with increasing  $L$ , leading to a pseudo-threshold behaviour for smaller lattice sizes that disappears for larger lattices.

We will now show how the detrimental effect of qubit and link fabrication errors can be mitigated at no additional hardware cost by disabling data qubits.

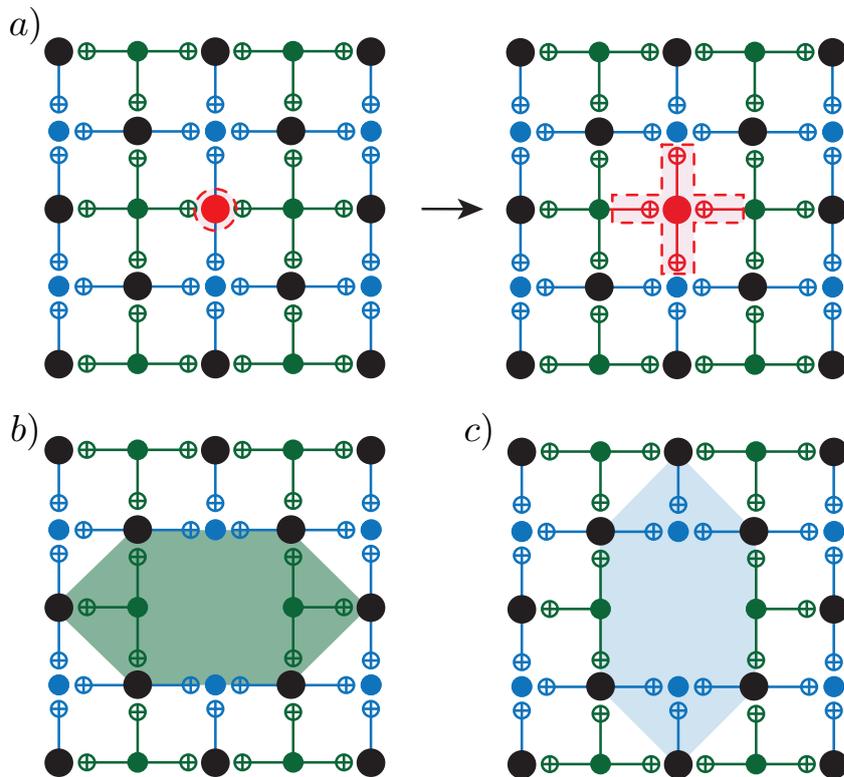
#### 4.1.1 Measuring supercheck operators and gauge qubits

The idea of using supercheck operators was first introduced to combat lost data qubits in the toric code [Stace et al., 2009]. This approach works on the basis that the product of two stabiliser generators is also in the code stabiliser, so when a data qubit is lost (i.e. an edge is removed from the lattice) the two adjacent damaged stabiliser generators can be jointly measured — forming a larger *supercheck* operator — to avoid the lost data qubit, hence preserving the stabiliser structure of the code.

This same approach can in theory be used for data qubit fabrication errors. However, measuring a supercheck operator directly is often a non-trivial task as it may require interaction between arbitrarily separated qubits or involve many SWAP gates, as was shown in [Nagayama et al., 2017], which can affect measurement of nearby stabiliser generators. Our approach for handling fabrication errors is based on the supercheck operator approach, but makes use of gauge qubits. Instead of measuring the supercheck operators themselves, we use the gauge qubits to construct the outcomes of the supercheck operators from the direct measurement outcomes of damaged stabiliser generators, such that all interactions remain as nearest-neighbour qubit interactions and no SWAP gates are required; any additional processing required is performed classically.

Consider the simple case of disabling a data qubit as shown in Fig. 4.3; the adjacent damaged generators will anti-commute, but their supercheck operator product remains deterministic. Each disabled data qubit in the surface code (except data qubits on the edge of the lattice, see Sec. 4.1.4) introduces one degree of freedom, or gauge qubit, similar to when a stabiliser is turned-off to perform defect-based computation. The logical Pauli  $X$  and  $Z$  operators of these gauge qubits are the

damaged star and plaquette stabilisers — we will refer to them as the *gauge operators*. When these anti-commuting gauge operators are measured, the logical state of the gauge qubit is randomised, but the state of the gauge qubit is unimportant, so this randomisation is not a problem. Importantly, strings of  $X$  or  $Z$  operators cannot terminate undetectably in this region, unlike when a stabiliser generator is turned off — these gauge operators reduce the code distance slightly, but code distance still scales with physical lattice size.



**Figure 4.3:** Forming supercheck operators in the presence of data qubit fabrication errors. When a data qubit is disabled or faulty (shown in red with a dashed border), the associated links are disabled (a), resulting in four adjacent damaged check operators. The product of two stabiliser generators is used to form a supercheck operator, effectively removing this qubit from the code. This process occurs in both the primal (b) and dual (c) lattices. Note that the CNOT gates shown in this figure are those used when measuring the supercheck operators as products of gauge operators.

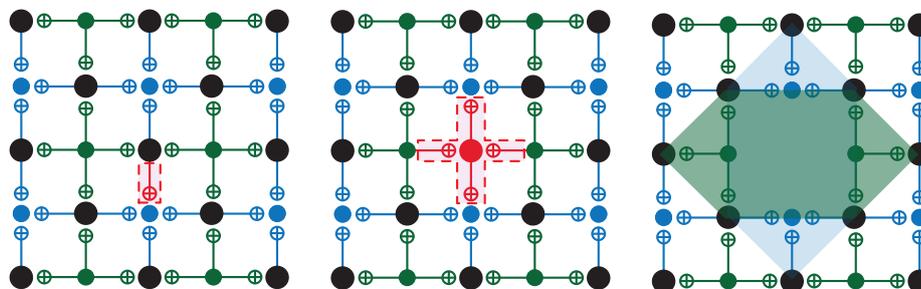
The supercheck operator product of damaged stabiliser generators commutes with every damaged stabiliser generator, so the supercheck operators remain in the stabiliser group and can be used for error correction during the classical processing

stage by treating the products of the damaged stars and plaquettes as supercheck operators.

### 4.1.2 Mapping fabrication errors to faulty data qubits

We saw in the previous section how in the presence of faulty (or disabled) data qubits the outcome of a supercheck operator can still deterministically be obtained by taking the product of the outcomes of the damaged operators. We will exploit this fact to map both link fabrication errors and syndrome qubit fabrication errors to disabled data qubits such that supercheck operators can always be formed.

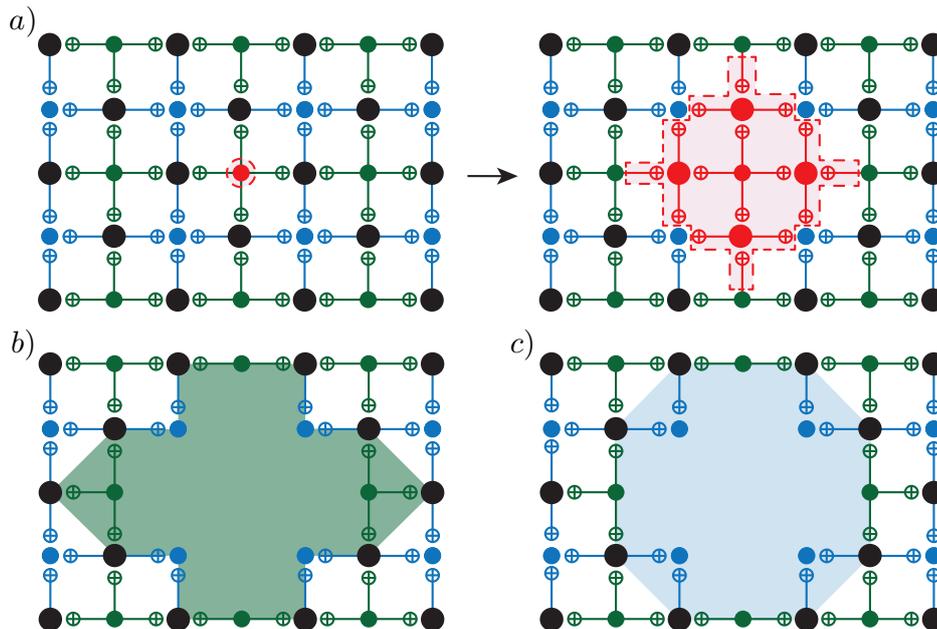
The mapping works as follows. If a link fabrication error occurs, it is mapped to a disabled qubit on the data qubit to which it connects, so that the data qubit is disabled along with its associated links, as shown in Fig. 4.4. If a syndrome qubit fabrication error occurs, it is mapped to disabled qubits on all of the surrounding data qubits, so that all these data qubits along with their associated links are disabled, as shown in Fig. 4.5. As a result, we see that syndrome qubit fabrication errors have the most destructive effect on the lattice in our approach, which highlights an important bias between data and syndrome qubits.



**Figure 4.4:** Mapping link fabrication errors to disabled data qubits; faulty and disabled components are shown in red with a dashed border. When a link fabrication error occurs (left), the data qubit associated with the link is disabled (middle) and superstars and superplaquettes are formed on the primal and dual lattices respectively (right).

### 4.1.3 Percolation thresholds and effective code distance

The use of supercheck operators is limited by percolation — if a string of faulty or disabled data qubits percolates the lattice, a logical qubit cannot be encoded as it is



**Figure 4.5:** Mapping syndrome fabrication errors to disabled data qubits; faulty and disabled components are shown in red with a dashed border. When a syndrome qubit fabrication error occurs, all the data qubits involved in the associated stabiliser generator are disabled (a). Large superstars (b) and superplaquettes (c) are formed around these disabled data qubits.

impossible to form consistent spanning logical operators and supercheck operators. Note that it might be possible to use part of the lattice as a smaller code, but we will consider percolating fabrication defects to be a manufacturing failure as the device cannot be used in its intended manner, and hence the surface code is discarded.

In percolation theory, one generally considers graphs formed of *sites* (nodes) and *bonds* (edges between nodes), and *percolation thresholds* can be defined with regards to each of these. If each site (bond) has a probability  $p$  of existing for a given graph structure with a size parameter  $n$ , the threshold for site (bond) percolation is the minimum value of  $p$  required to guarantee the existence of a path from one side of the graph to the other as  $n$  tends to infinity. Above the percolation threshold, increasing  $n$  increases the probability of a percolating path, and below the percolation threshold, increasing  $n$  decreases the probability of a percolating path.

The square lattice structure of the surface code implies that the qubit percolation threshold for the surface code is equivalent to the bond percolation threshold for the square lattice, which is a known analytic value of 50% [Sykes and Essam, 1964].

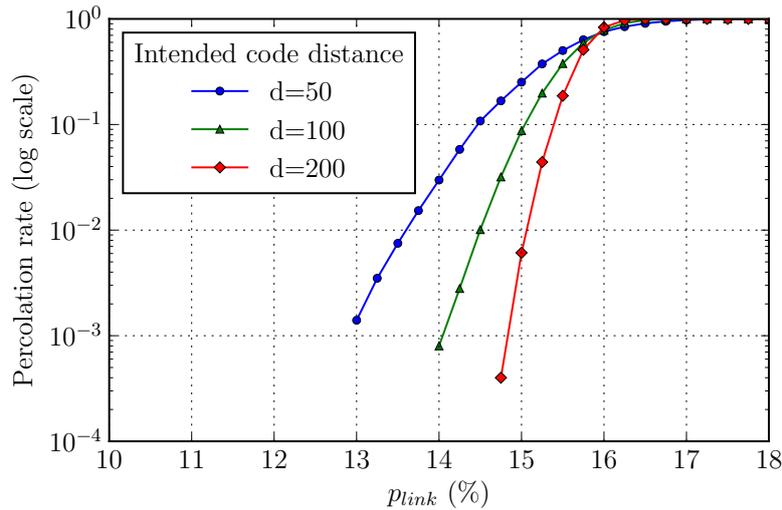
By using our mapping to disabled data qubits, we can derive approximate analytic percolation thresholds in the presence of qubit and link fabrication errors. These percolation thresholds provide upper bounds for tolerable fabrication error rates when building surface code devices because increasing the code distance when fabrication error rates are beyond these threshold values increases the probability that percolating defects occur, so logical error rates can no longer be reduced by increasing the code distance.

In the bulk of the lattice, each data qubit has four links. With a link fabrication error rate of  $p_{\text{link}}$ , the probability of each data qubit being disabled due to faulty links is  $1 - (1 - p_{\text{link}})^4$ , i.e. one minus the probability of no faulty links occurring. This implies that 50% of data qubits will be disabled when  $1 - (1 - p_{\text{link}})^4 = 0.5$ , or equivalently when  $p_{\text{link}} = 1 - \sqrt[4]{0.5} \approx 0.159$ . This analysis does not account for qubits at the edges of the planar code having fewer than four links, but the percolation threshold is an asymptotic behaviour, so this effect can be neglected for large lattices.

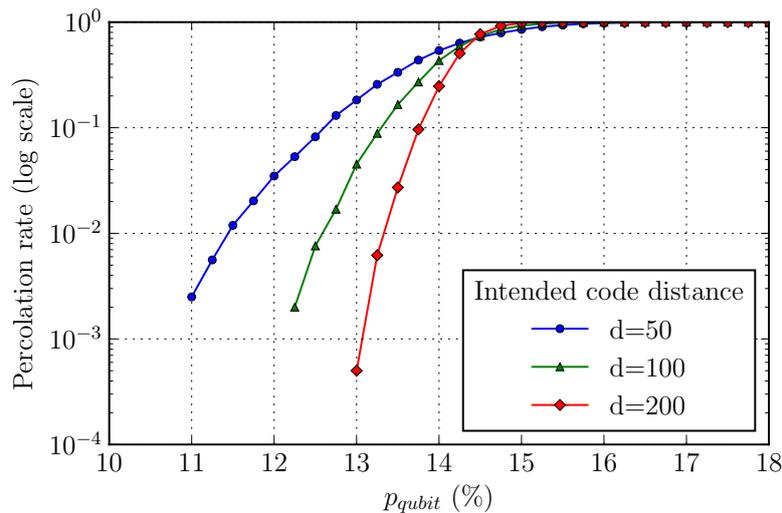
An analogous argument can be used to calculate an approximate threshold for qubit fabrication errors. A data qubit will be disabled when it is either faulty or one or more of the four syndrome qubits it is linked to are faulty. The probability that a particular qubit is disabled when the qubit fabrication error rate is  $p_{\text{qubit}}$  is  $1 - (1 - p_{\text{qubit}})^5$ . Therefore, the qubit fabrication error percolation threshold occurs approximately when  $1 - (1 - p_{\text{qubit}})^5 = 0.5$ , or  $p_{\text{qubit}} = 1 - \sqrt[5]{0.5} \approx 0.129$ . This analysis is less accurate than that for link fabrication errors as it does not account for the possible correlations between disabled data qubits due to faulty syndrome qubits. However, localised clusters of disabled qubits are generally less likely to percolate than data qubits that are disabled at random, so we expect the actual threshold to be slightly higher.

Results from our numerical simulations for planar code percolation, shown in Fig. 4.6 and Fig. 4.7 for link and qubit fabrication errors respectively, are in strong agreement with our above approximations. These simulations modelled the creation of imperfect planar code lattices with intended code distances of 50, 100 and 200

and tested for the occurrence of percolation errors, with each code distance and error rate combination performed for  $10^4$  runs. We find the link fabrication threshold, Fig. 4.6, to be just under 16% and the qubit fabrication threshold, Fig. 4.7, to be between 14% and 15% (higher than the analytic estimate of 12.9% due to data qubit loss occurring in localised clusters, as expected).



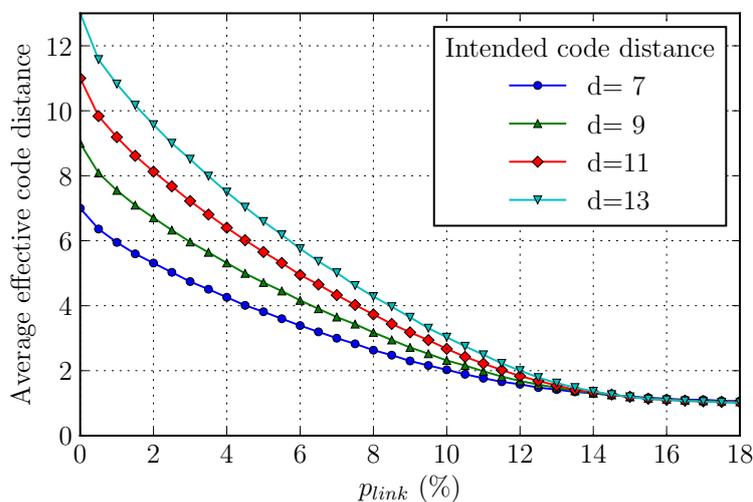
**Figure 4.6:** Percolation rates for link fabrication errors only. The crossing point gives a threshold of just under 16%.



**Figure 4.7:** Percolation rates for qubit fabrication errors only. The crossing point gives a threshold of around 14.5%.

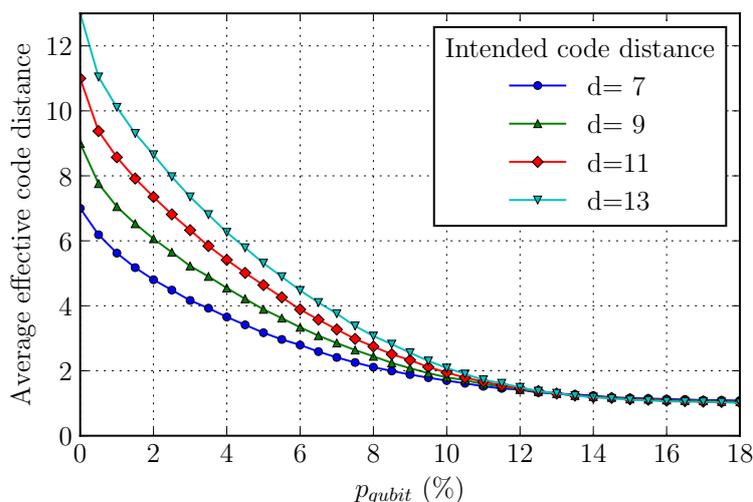
### Effective code distance

Forming supercheck operators leads to a reduction in the *effective* code distance compared to the intended code distance as it reduces the length of the shortest logical operator. We have analysed how the average effective code distance varies with link and qubit fabrication errors, as shown in Fig. 4.8 and Fig. 4.9, respectively. These graphs were produced by simulating the imperfect fabrication of planar code lattices with intended code distances 7, 9, 11 and 13, with each type of fabrication error varied independently, and then finding the effective code distance of each lattice by identifying the weight of the lowest-weight logical operator. The lowest-weight logical operator is found by using a path finding routine to identify the shortest path across each of the primal and dual lattices, with the lowest-weight logical operator being the shortest of these. Each code distance and error rate combination was repeated  $10^4$  times, and the effective code distance was then averaged over all runs, with the average being calculated using only non-percolated lattices to ensure that the average effective code distance does not fall below 1.



**Figure 4.8:** Average effective code distance for link fabrication errors only.

For an undamaged surface code, the code distance scales linearly with the dimension  $L$  of the lattice. If this relationship holds for lattices with fabrication errors, we would expect a relationship between effective distance and intended



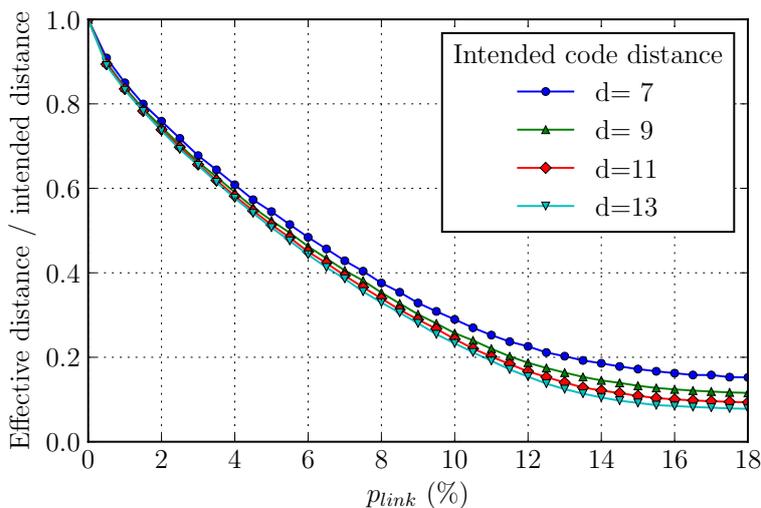
**Figure 4.9:** Average effective code distance for qubit fabrication errors only.

distance to be of the form  $d_{\text{effective}} = c(p)d_{\text{intended}}$ , where  $c(p)$  is a number that varies with the fabrication error rate  $p$ , such that the ratio between effective code distance and intended code distance should be constant for a given fabrication error rate.

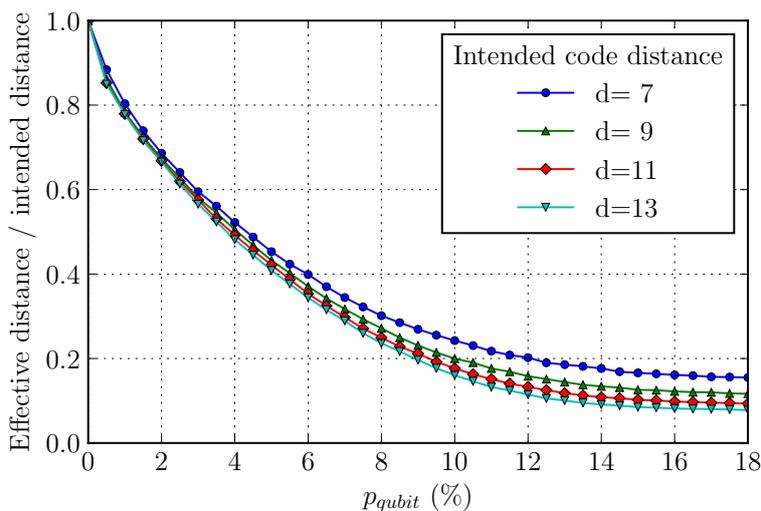
Fig. 4.10 and Fig. 4.11 show the how the value of  $d_{\text{effective}}/d_{\text{intended}}$  scales with link and qubit fabrication error rates respectively; if the effective code distance scales linearly with the dimension of the lattice then we expect the value of  $d_{\text{effective}}/d_{\text{intended}}$  to be independent of the intended code distance. As can be seen in the figures, the value initially seems to be almost independent of intended code distance at low fabrication error rates. However, as the fabrication error rates increase, the relationship weakens (this is expected as the effective code distance is asymptotic to 1 in Fig. 4.8 and Fig. 4.9). The dependence of  $d_{\text{effective}}/d_{\text{intended}}$  on intended code distance appears to be weaker for the larger lattice sizes, which suggests this may be a finite-size effect.

#### 4.1.4 Complications

There are two complications that occur when using gauge operators to measure supercheck operators. The first is that supercheck operators can only be measured during alternating rounds of syndrome measurement, and the second involves faulty data qubits at the edge of the lattice.



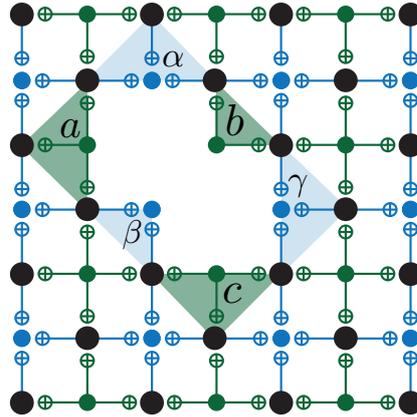
**Figure 4.10:** Ratio between average effective code distance and intended code distance for link fabrication errors only.



**Figure 4.11:** Ratio between average effective code distance and intended code distance for qubit fabrication errors only.

The interleaved  $z$ -shaped measurement pattern that allows all stabiliser generators to be measured simultaneously on a perfect lattice can no longer be used for supercheck operators. Ensuring that the product of damaged stabilisers is deterministic requires that no anti-commuting operations are performed while the constituent gauge operators of the supercheck operators are being measured; this is not possible with the normal measurement pattern, as the example in Fig. 4.12 shows. This prob-

lem is mitigated by measuring star and plaquette type supercheck operators during alternating rounds. All undamaged stabiliser generators are measured every round as normal; this means that supercheck operators are measured half as frequently as undamaged stabiliser generators, and these measurements therefore have a higher effective error rate than undamaged stabiliser generator measurements.



**Figure 4.12:** Effective order of gauge operator measurement with damaged checks. If the normal *z-shaped* measurement pattern is used, the effective order in which the gauge operators are measured is  $c$ ,  $\gamma$  &  $\beta$ ,  $a$  &  $b$ ,  $\alpha$ . The anti-commutation randomises the supercheck operator products, so star and plaquette gauge operators are instead measured in alternating rounds to ensure the supercheck operator outcome is deterministic.

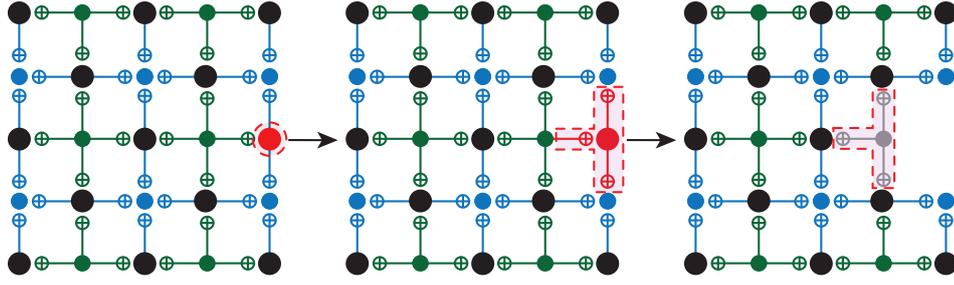
The second issue occurs when data qubits are faulty at the edges of the lattice. If there is a data qubit fault such as that shown in Fig. 4.13, then there is no corresponding stabiliser generator to pair it with. Therefore, the edge of the lattice must be redefined by completely disabling the damaged stabiliser generator. This process must then be repeated if any of the qubits in this new edge are faulty — the process effectively results in a supercheck operator being disabled.

## 4.2 Threshold simulations

### 4.2.1 Error model

The error model used in the simulations is outlined below, and summarised in Table 4.1.

Every quantum operation is modelled to experience computational errors with a probability denoted by the parameter  $p_{\text{comp}}$ . Each two-qubit gate is assumed to



**Figure 4.13:** Dealing with fabrication errors at the edge of the lattice; faulty and disabled components are shown in red with a dashed border. When data qubits at the edge of the lattice are disabled (left, middle), the product of gauge operators is not in the stabiliser, so such stabiliser generators must be disabled rather than forming supercheck operators (right).

Operation	Error model
Qubit fabrication	Each qubit is permanently faulty with probability $p_{\text{qubit}}$
Link fabrication	Each link is permanently faulty with probability $p_{\text{link}}$
State-preparation	Orthogonal state prepared with probability $p_{\text{comp}}$
Single-qubit gate	One of $X$ , $Y$ and $Z$ is applied with probability $\frac{4}{5}p_{\text{comp}}$ (each is equally likely)
Two-qubit gate	One of the 15 non-identity Pauli operators is applied with probability $p_{\text{comp}}$ (each is equally likely)
State measurement	Incorrect measurement outcome with probability $p_{\text{comp}}$

**Table 4.1:** Error model for surface code simulations with fabrication errors.

act perfectly followed by depolarising Pauli noise with probability  $p_{\text{comp}}$ . Single qubit gates (only the identity gate in our simulations) are assumed to act perfectly followed by depolarising noise with probability  $4p_{\text{comp}}/5$ . The justification for this follows that of [Knill, 2005]:  $4p/5$  is the marginal error rate on each qubit involved in a two-qubit gate experiencing depolarising noise with probability  $p$ . If we were to choose a single-qubit error rate of  $p_{\text{comp}}$ , this would imply that single-qubit gates are more prone to errors than two-qubit gates, which is unlikely to be the case in a real device. Preparation is considered to have probability  $p_{\text{comp}}$  of preparing the state in an orthonormal basis, and measurement is considered to have a probability  $p_{\text{comp}}$  of giving the incorrect outcome.

All fabrication errors are considered to occur independently before error correction is initiated, and the locations of all fabrication errors are assumed to be known. A qubit fabrication error occurs with probability  $p_{\text{qubit}}$  for each qubit (syndrome

and data qubits) and link fabrication errors occur with probability  $p_{\text{link}}$  for each link. The parameters  $p_{\text{comp}}$ ,  $p_{\text{qubit}}$  and  $p_{\text{link}}$  are varied independently.

### 4.2.2 Simulation methods

An overview of the simulation is shown in Fig. 4.14. Each simulation starts by generating a lattice of qubits and links with the appropriate fabrication error rates. The fabrication errors are then mapped to data qubit fabrication errors using the mapping described in Sec. 4.1.2, and suitable logical operators are found by using a path finding routine on the primal and dual lattices. If a logical operator cannot be found, then the lattice is percolated by faulty (or disabled) data qubits and the simulation is terminated.

When a logical operator is found,  $2 \times L$  rounds of syndrome measurement are performed. Each round consists of syndrome qubit initialisation, four stages of two-qubit gates and then syndrome qubit measurement. Each of these is considered to take one unit of time, and any qubit that is not involved in a two-qubit gate, measurement or preparation during a particular time step undergoes an identity gate.

The code is initialised with a round of perfect star and plaquette measurements to get the error-free outcomes for each star and plaquette measurement, and the simulations are capped with a final round of perfect measurement to project the final state into the code space. All other rounds of stabiliser measurement use the Pauli error model given above.

We use the CHP stabiliser routine [Aaronson and Gottesman, 2004] to simulate the quantum state of the lattice during all gates and measurements to ensure that the gauge operators give the correct outcomes. Once all measurements have been obtained, a minimum weight perfect matching routine involving Blossom V [Kolmogorov, 2009] is used to find a correction based on the obtained syndrome. Edge weights for the perfect matching routine are set using the same methods as [Bravyi and Vargo, 2013, Suchara et al., 2015] to optimise the matching, where each edge weight is inversely proportional to probability of a single error occurring at that location. Syndrome measurement is simulated on both the primal and dual lattices

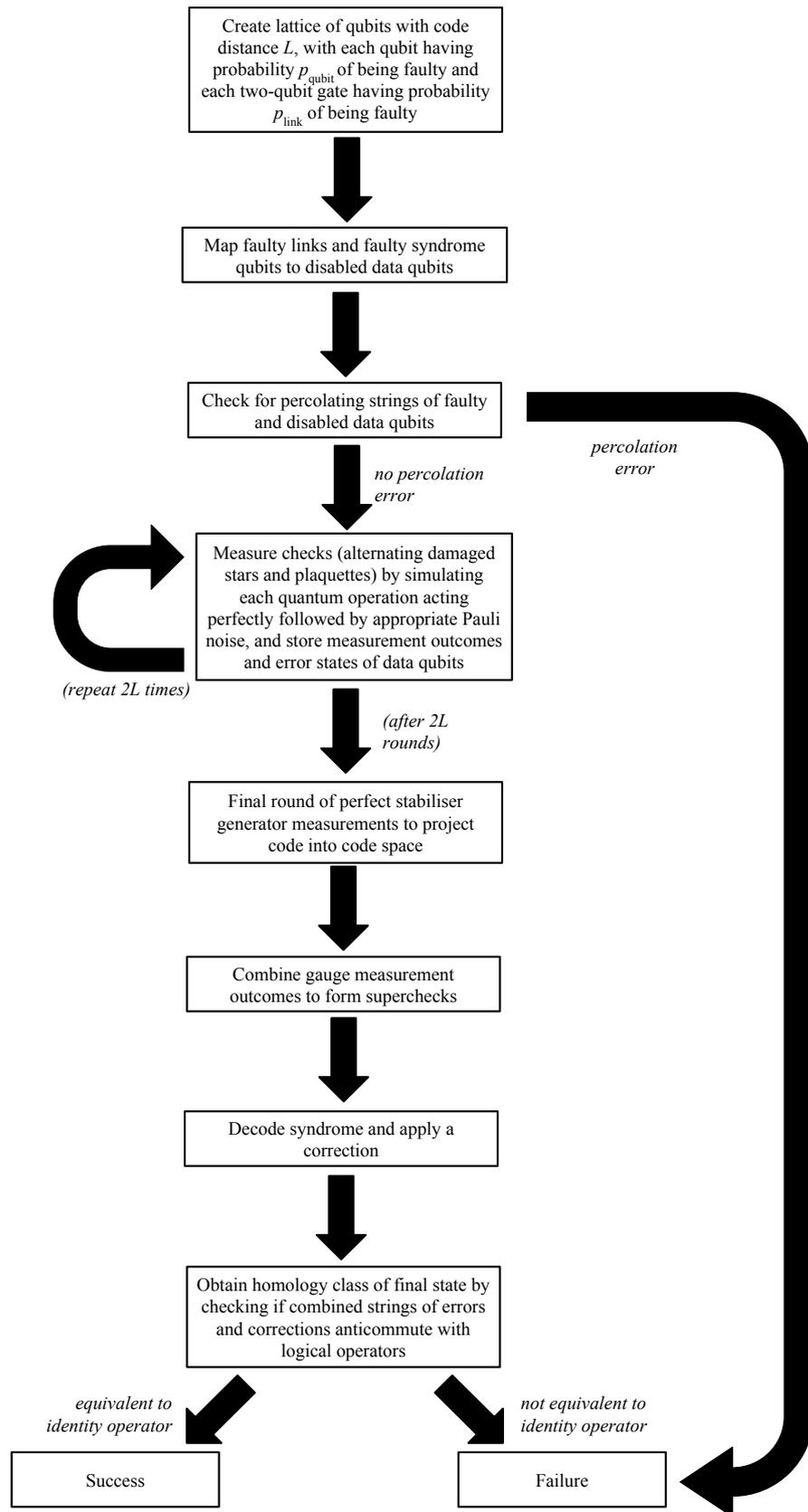


Figure 4.14: Steps for simulating surface code error correction with fabrication errors.

to allow for accurate error propagation, but error correction is only performed on one lattice to reduce computation time (the symmetry between primal and dual lattices means that logical error rates are almost identical). Once a correction has been found and applied, we test for the occurrence of a logical  $X$  error by checking if the combined error and correction string commutes with the logical  $Z$  operator.

Pauli error rates were varied from  $p_{\text{comp}} = 0.05\%$  to  $p_{\text{comp}} = 1.00\%$  in steps of  $0.05\%$ , with additional values of  $0.001\% < p_{\text{comp}} < 0.010\%$  used when the Pauli error threshold became very small. Each fabrication error rate was separately varied from  $0\%$  in steps of  $2\%$  until no threshold could be obtained, with an additional simulation performed at  $5\%$  fabrication error rate to allow for a direct comparison with [Nagayama et al., 2017]. Each combination of error rates and code distance was simulated for a minimum of  $5 \times 10^4$  runs, as this was number found to be sufficient to obtain the required thresholds.

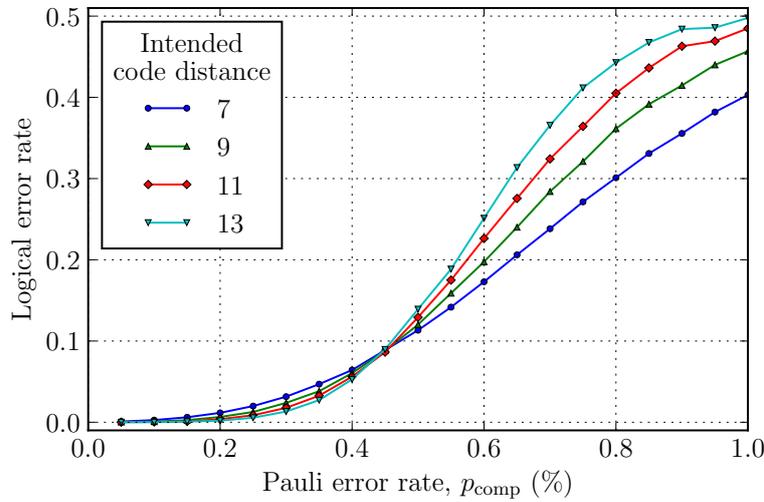
### 4.3 Results

Our main result is summarised in Fig. 4.16, which shows how the planar code Pauli error threshold varies with each type of fabrication error. Thresholds for each fabrication error rate were obtained by finding the intersection of logical error rates when results for each code distance are plotted on the same graph [Wang et al., 2003]; an example is shown in Fig. 4.15 for the results obtained when  $p_{\text{qubit}} = 4\%$ , which results in a threshold  $p_{\text{th}} \approx 0.45\%$ . Instances in which faulty or disabled data qubits percolate the lattice, such that a logical qubit cannot be encoded, have been discounted when calculating the threshold, such that the logical error rate is given by

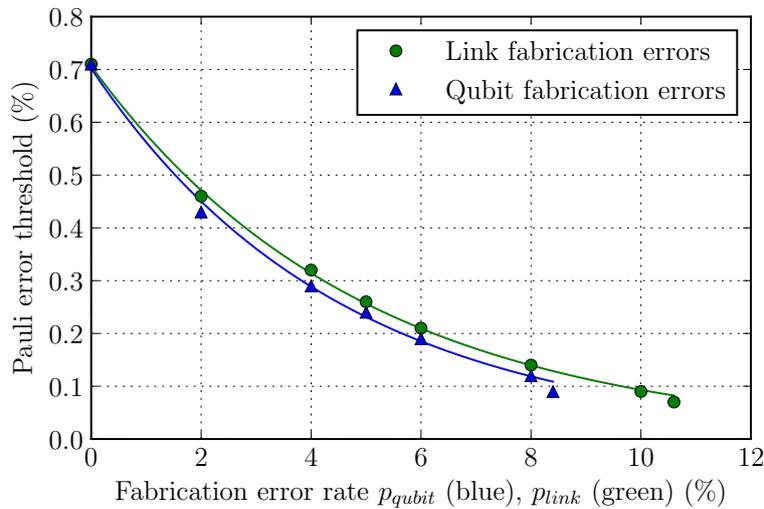
$$p_{\text{log}} = \frac{\#_{\text{logical errors}}}{\#_{\text{total runs}} - \#_{\text{percolation errors}}}, \quad (4.1)$$

where  $\#_i$  is used to signify ‘number of  $i$ ’. This allows us to investigate the computational performance of viable devices only without thresholds being affected by percolation errors. The percolation graphs in Fig. 4.6 and Fig. 4.7 give an indication of the frequency of such percolation errors (i.e. the proportion of devices

manufactured that are not viable to use for computation).



**Figure 4.15:** Logical error rates with qubit fabrication error rate  $p_{\text{qubit}} = 4\%$ . The error threshold is given by the intersection point, in this case  $p_{\text{th}} \approx 0.45\%$ .



**Figure 4.16:** Pauli error thresholds for link,  $p_{\text{link}}$ , and qubit,  $p_{\text{qubit}}$ , fabrication errors. An exponential is fit applied to each dataset:  $0.71e^{-20p_{\text{link}}}$  for link fabrication errors and  $0.70e^{-22p_{\text{qubit}}}$  for qubit fabrication errors.

As expected, qubit fabrication errors have a slightly more damaging effect on the threshold than link fabrication errors. With no fabrication errors, the chosen error model results in a Pauli error threshold of  $p_{\text{comp}} \approx 0.71\%$ . As  $p_{\text{qubit}}$  and  $p_{\text{link}}$  increase, the thresholds of  $p_{\text{comp}}$  decrease, with the respective thresholds dropping

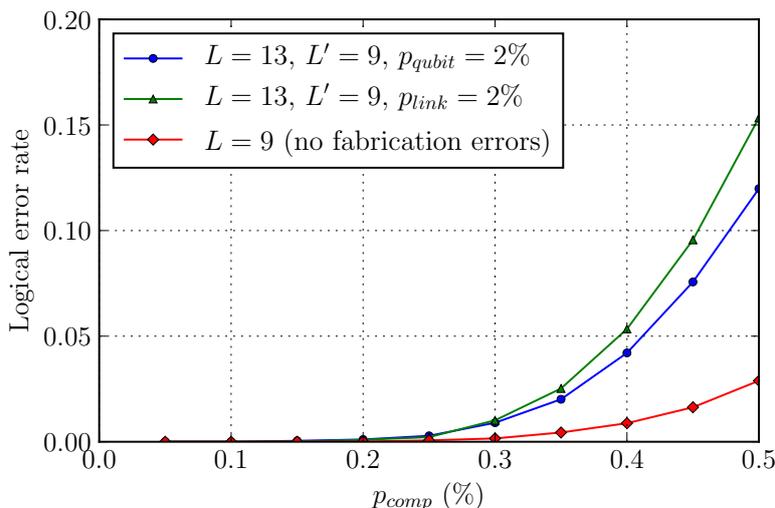
below 0.1% when  $p_{\text{qubit}} \gtrsim 8\%$  and  $p_{\text{link}} \gtrsim 10\%$ . It has not been possible to find clean thresholds to determine the behaviour of the thresholds beyond  $p_{\text{link}} = 10.6\%$  and  $p_{\text{qubit}} = 8.4\%$ ; this is due to effective code distances converging as fabrication error rates increase — finding thresholds requires a range of code distances, but as shown in Fig. 4.8 and Fig. 4.9, the average effective code distances for link fabrication error rates of 12% qubit fabrication error rates of 10% are all  $\lesssim 2$  for the lattice sizes simulated.

An exponential fit of the form  $\alpha \exp(\beta p_{\text{fab}})$  to each data set results in fits of  $\alpha = 0.70$ ,  $\beta = -22$  for qubit fabrication errors and  $\alpha = 0.71$ ,  $\beta = -20$  for link fabrication errors. These fits fail close to the percolation threshold but are in good agreement with the simulation results for  $p_{\text{qubit}} \leq 0.08$  and  $p_{\text{link}} \leq 0.1$ .

## 4.4 Discussion

The logical error rate,  $p_{\text{log}}$ , depends on both the intended code distance  $L$  and the actual distance  $L'$ , such that  $p_{\text{log}} = p_{\text{log}}(L, L')$ . Fabrication errors mean that  $L' \leq L$ . We find that  $p_{\text{log}}(L', L') < p_{\text{log}}(L, L')$ , i.e. fabrication errors degrade the code performance beyond simply reducing the code distance, as shown in Fig. 4.17 for distance 9 codes. The larger codes perform worse because the higher-weight supercheck operators are more prone to error and give less-specific information about the location of an error compared to a *native* code. This suggests that building a larger surface code is generally only beneficial if the fabrication error rate does not increase substantially.

Our thresholds for qubit fabrication errors are slightly lower than that of [Nagayama et al., 2017]. This is because a faulty syndrome qubit is more damaging in our scheme as each syndrome qubit fabrication error is mapped to multiple disabled data qubits; this results in lower percolation error thresholds and lower effective code distances for qubit fabrication errors. However, the gauge operator approach we present may be better for handling data qubit fabrication errors as it requires fewer logic gates. It is also worth noting that from the point of view of the implementation, our code does not require the performance of any extra gates, as the approach pre-



**Figure 4.17:** Logical error rates for codes with fabrication errors with intended distance  $L = 13$  and actual distance  $L' = 9$  compared to native distance 9 codes with no fabrication errors. The native codes have lower logical error rates.

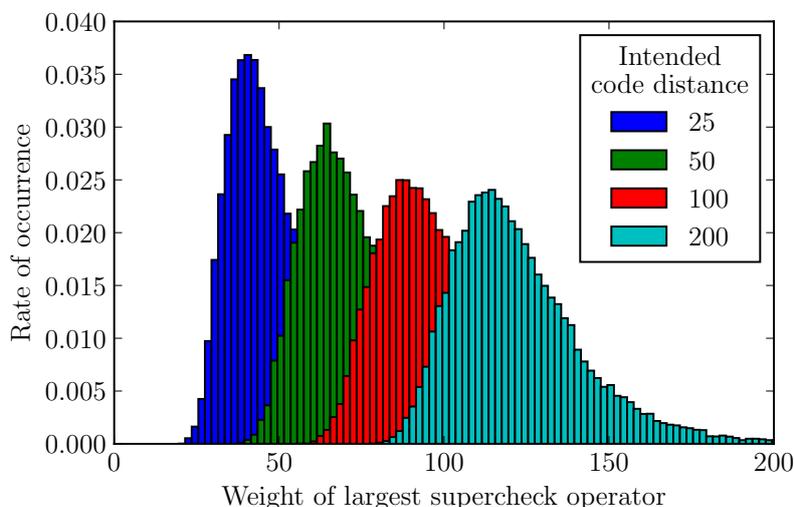
sented in [Nagayama et al., 2017] does; all the adaptations required by our scheme involve a change in the measurement order and disabling particular measurements, which may be a more amenable adaptation for some physical systems. It should be noted that thresholds are dependent upon the chosen error model, so one cannot claim that a given approach is definitively better than the other in all circumstances. Link fabrication errors were not considered in [Nagayama et al., 2017].

#### 4.4.1 Scaling of supercheck operator weight

Additional simulations were performed to investigate the scaling of the weight of the largest supercheck operator as code distance increases. These simulations modelled the creation of lattice sizes with code distances 25, 50, 100 and 200 with the same fabrication error rates used in the main simulations, and for each run the weight of the largest supercheck operator was determined (no error correction was performed). Each combination of error rates and code distance was performed for  $10^4$  runs.

We find that larger lattice sizes have higher-weight supercheck operators — Fig. 4.18 shows the results for  $p_{link} = 10\%$ . This phenomenon can be understood by considering the similarity between clustering in subcritical percolation, where the size of the largest cluster scales logarithmically with the size of the system [Bazant,

2000], and forming supercheck operators on a surface code.



**Figure 4.18:** Rate at which highest-weight supercheck operators occur for link fabrication error rate  $p_{\text{link}} = 10\%$ . The weight of the highest-weight supercheck operator appears to increase logarithmically with code distance as expected [Bazant, 2000]. In this case, the peaks are at weights of 40, 65, 88 and 115 for code distance 25, 50, 100 and 200 respectively (a spacing of approximately 25 each time the code distance doubles), which results in a modal supercheck operator weight fitting of  $\log_{1.028}(d) - 76.10$ .

Higher-weight supercheck operators are more prone to error, so this effect means that the thresholds observed for smaller lattice sizes may not be stable and could lead to pseudo-threshold behaviour, where the threshold disappears with increasing lattice size. This is not unique to our approach; it is a feature of any surface code or topological cluster state code based around the supercheck operators introduced in [Stace et al., 2009], including [Barrett and Stace, 2010] and [Nagayama et al., 2017].

We note that existing threshold proofs for the surface code are unlikely to hold when measuring supercheck operators as products of gauge operators. Indeed, the scheme in this chapter shares many features with Bacon-Shor codes [Bacon, 2006], which are subsystem codes where stabiliser operators are formed by multiplying the outcomes of gauge operators. Bacon-Shor codes do not have true thresholds [Napp and Preskill, 2013], although they do exhibit error suppression at suitably low code distances, so it is possible the same is true when measuring supercheck operators of

the surface code as products of gauge operators.

However, concatenated codes [Knill and Laflamme, 1996], which do have thresholds, also have check operators that increase in weight with increasing code distance, so the increase in the weight of the supercheck operators alone does not rule out the possibility that thresholds exist. Proving or disproving the existence of a threshold, and finding a way to overcome this problem if necessary, remains an open problem.

## 4.5 Conclusion

We have presented a full analysis for the threshold performance of the surface code in the presence of fabrication errors and have showed that the ability to disable qubits or links (two-qubit gates) is sufficient to map the fabrication errors into lost qubits, such that the syndrome extraction can be performed without the need of any additional hardware components. Our method combines the supercheck operator approach and the concept of gauge qubit operators to deterministically obtain the outcome of supercheck operators. Interestingly, our approach shows that syndrome qubit fabrication errors have a more drastic damaging effect on the lattice in comparison to data qubit fabrication errors, showing that more care is required to fabricate high quality syndrome qubits.

One advantage of the scheme presented here compared to alternative methods is that it should be applicable to non-gate-based implementations where SWAP gates may not be feasible, such as [O’Gorman et al., 2016]. However, in schemes where the SWAP gate is readily available, the gauge operator scheme presented in this work and the SWAP gate scheme of [Nagayama et al., 2017] can be complementary schemes. For data qubit fabrication errors, the scheme presented here requires fewer quantum operations to obtain syndrome data so would be preferable. However, the scheme in [Nagayama et al., 2017] may be preferable to deal with syndrome qubit fabrication errors, as it still allows for the measurement of stabiliser generators associated with faulty syndrome qubits. Therefore, a hybrid of both approaches could lead to an improvement in the overall performance.

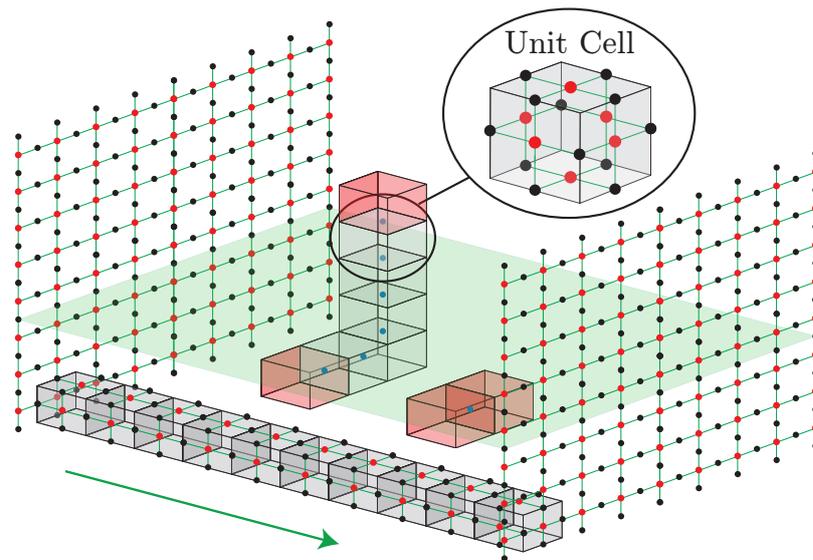
The fabrication error model presented here can be expanded to include cases where the fabrication process results in components of different quality, such that some components are fabricated with higher chance of being prone to computational Pauli errors. Such an asymmetry in the quality of fabricated components is to be expected for large-scale systems, but we leave such analysis for future investigations.

## Chapter 5

# Fault-tolerant quantum computation with non-deterministic entangling gates

There are many current experimental proposals for building a universal quantum computer, and all of these suffer from the accumulation of errors that arise from the decoherence of physical quantum operations; these errors can be handled using standard quantum error correction codes such as the surface code. Some implementations — such as those that utilise optical components in constructing large-scale linear optical architectures [Knill et al., 2001, Browne and Rudolph, 2005, Rudolph, 2017, Gimeno-Segovia et al., 2015] or networks of trapped ions [Häffner et al., 2008, Nickerson et al., 2014] — suffer from an additional problem in the form of non-deterministic entangling operations, a problem that has not been widely studied.

In this chapter, we show that it is possible to perform fault-tolerant quantum computation with probabilistic entangling gates using the well-established topological cluster state scheme due to Raussendorf [Raussendorf et al., 2006, Raussendorf et al., 2007] — a three-dimensional measurement-based scheme that supports topological error correction. We propose two approaches for handling non-deterministic entanglement generation in Raussendorf’s scheme: a non-adaptive approach, which involves the same measurement pattern as the original scheme with no additional quantum processing, and an adaptive approach, which involves changing the basis



**Figure 5.1:** The three-dimensional topological cluster state, constructed from cubic cells (inset); bulk qubits are hidden for clarity. Only the ends of strings of  $Z$  errors (blue) are detected by check operators (highlighted in red). The correlation surface (green surface) spans the lattice in the time-like direction (green arrow).

in which some qubits are measured.

The work in this chapter is primarily motivated by linear optical architectures [Knill et al., 2001, Browne and Rudolph, 2005, Rudolph, 2017, Gimeno-Segovia et al., 2015], but the analysis is sufficiently general that the qualitative results are relevant to other implementations with non-deterministic entanglement. The approach we describe relaxes the need for deterministic or repeat-until-success [Lim et al., 2006] entanglement generation, and we show that Raussendorf’s scheme can tolerate a degree of failure in the construction of the underlying cluster state bonds.

Previous work along similar lines of research include [Barrett and Stace, 2010, Whiteside and Fowler, 2014], which considered qubit loss and leakage in topological cluster states, [Li et al., 2010], which considered the construction of topological codes with non-deterministic entanglement between multi-qubit resource states, and [Fowler et al., 2010], which considered surface code based quantum repeaters with non-deterministic entanglement between nodes but deterministic entanglement within nodes. The work presented in this chapter differs from that of [Li et al., 2010] in that we are considering non-deterministic entanglement between all qubits, rather

than between networks of multi-qubit nodes.

## 5.1 Topological cluster states

In this chapter, we discuss the cubic topological cluster state (TCS) [Raussendorf et al., 2006, Raussendorf et al., 2007], which consists of qubits in a lattice configuration based around the unit cell shown in Fig. 5.1; the edges between qubits in this figure are referred to as *bonds*. Entanglement is created between carefully-chosen neighbouring qubits during initialisation to form a highly-entangled *cluster state* [Briegel and Raussendorf, 2001], and the error correction and quantum computation then proceed by single-qubit operations alone with no further multi-qubit measurements or gates required — this scheme is an example of measurement-based quantum computation [Raussendorf and Briegel, 2001]. Topological protection is achieved by having the surface code as a substrate at each layer of the cluster state, such that the two-dimensional logical operators of each surface code are expanded into to the third dimension to form *correlation surfaces* that encode logical information globally.

The quantum state of the lattice is equivalent to that obtained by preparing every qubit in the  $|+\rangle$  state and performing CPHASE gates between qubits linked by bonds, but the lattice can be created in different but equivalent ways without explicit  $|+\rangle$  state preparation and CPHASE gates, such as the use of fusion gates [Browne and Rudolph, 2005] in the linear optics scheme in [Gimeno-Segovia et al., 2015], where GHZ states are used to form the cluster states and CPHASE gates are not explicitly performed. TCS quantum computation is qualitatively similar to a defect-based surface code scheme on a three-dimensional lattice of qubits where one physical direction acts as a time-like dimension; we will therefore refer to the dimensions of the lattice as *space-like* or *time-like*. This three-dimensional nature means TCS schemes are less suitable for chip-based quantum computation, but well suited to schemes based on linear optics. As with the surface code, we will focus mostly on using topological cluster states as a quantum memory to explain their key features.

The structure of the TCS lattice gives rise to primal and dual lattices that are

used for error correction — two interleaved cubic lattices, one with the black qubits from Fig. 5.1 on the centre of each cube face and the other with the red qubits on the centre of each face. For each qubit,  $i$ , in a cluster state, there is an associated stabiliser operator,  $S_i$ , of the form

$$S_i = X_i \bigotimes_{j \in N(i)} Z_j, \quad (5.1)$$

where  $N(i)$  is the neighbourhood of qubit  $i$  (the adjacent qubits). Therefore, for each cube face,  $f_i$ , centred on qubit  $i$ , there is an associated stabiliser generator  $S_i$  with an  $X$  operator acting on qubit  $i$  and  $Z$  operators acting on the adjacent qubits.

By multiplying the six face operators of each cube together, the  $Z$  contributions cancel, leaving a six-body operator with  $X$  operators acting on the qubit at the centre of each face. Stabiliser measurements can therefore be performed by measuring each face qubit in  $X$  and multiplying the outcomes to form a parity check associated with that cube; the term *check operator* will be used to describe these parity checks.

### 5.1.1 Detecting and correcting errors

In the absence of errors, all check operators have parity ‘+1’. If a  $Z$  error or a measurement error (incorrect measurement outcome) occurs on a single qubit on one face of a cube, this flips the parity of the check operator associated with that cube from ‘+1’ to ‘−1’ and also flips the parity of the corresponding adjacent cube. Errors on qubits on two faces of a cube will not change the parity of that cube’s check operator but will be detected by the two adjacent cubes, such that the check operators detect only the ends of error strings (see Fig. 5.1), much like in the surface code. Homologically trivial error strings — those that form closed loops — are equivalent to logical identity operations. Strings of errors spanning the space-like directions of the lattice result in undetectable and uncorrectable logical errors, whereas undetectable strings of errors spanning the time-like dimension of the lattice from the first to the final layer do not affect the logical state of the encoded qubits so are harmless. The length of the shortest undetectable space-like string across the lattice gives the code distance, so a lattice with a shortest dimension of  $n$

unit cells has code distance  $n + 1$ .

Once all qubits have been measured in the  $X$  basis, an error syndrome is obtained by collating the check operator outcomes. This syndrome provides the locations of the ends of error strings on the primal and dual lattices, and the decoder attempts to pair these to find a correction that minimises the probability of a logical error occurring. Much like with the surface code, TCS syndromes are degenerate, so the syndrome does not uniquely identify the errors that occur: different strings of errors can result in the same syndrome. However, again like the surface code, it is only important that the combined error and correction strings are homologically equivalent to the identity operation, not that the error is exactly inverted. As this is a measurement-based scheme, the correction is not physically applied to the qubits — information about the correction is used to update the measurement outcomes retrospectively, and these corrected measurement outcomes are then used to infer the logical state of the encoded qubits using *correlation surfaces*, which will be introduced shortly.

Syndrome information is obtained for both the primal and dual lattices, but unlike the surface code, both of these detect only  $Z$  errors, as  $X$  errors commute with the measurements used to obtain the syndrome; this is not a problem, as  $X$  errors are harmless to TCS<sup>1</sup>. This requirement to correct only  $Z$  errors can be understood by considering the equivalence between TCS and a surface code with repeated syndrome measurements.

In this picture, each TCS layer is equivalent to a surface code, with the black qubits in Fig. 5.1 representing data qubits and the red qubits representing syndrome qubits or vice-versa on alternating layers. The structure of the cluster state means that performing the  $X$  measurements on the data-like qubits in a given layer has the effect of teleporting the surface code onto the next layer and applying transversal Hadamard gates (i.e. applying a Hadamard gate to every data qubit), and measuring the syndrome-like qubits has the effect of measuring a surface code plaquette.

---

<sup>1</sup>If CPHASE gates are used to create the cluster states then  $X$  errors during initialisation may cause  $Z$  errors, so  $X$  errors are only harmless once the lattice has been constructed. It is still only necessary to detect  $Z$  errors, however.

Hadamard gates interchange  $X$  and  $Z$  operations, so measuring the syndrome-like qubits in this next layer is now equivalent to measuring the star check operators of the surface code, and measuring the data qubits is equivalent to a second round of teleportation and Hadamard gates, therefore restoring the surface code to its original state (as Hadamard gates are self-inverting); this process is then repeated for each TCS layer. The similarity between the surface code and TCS means that, in general, surface code decoders, such as those based on minimum weight perfect matching, can be readily adapted for decoding TCS syndromes.

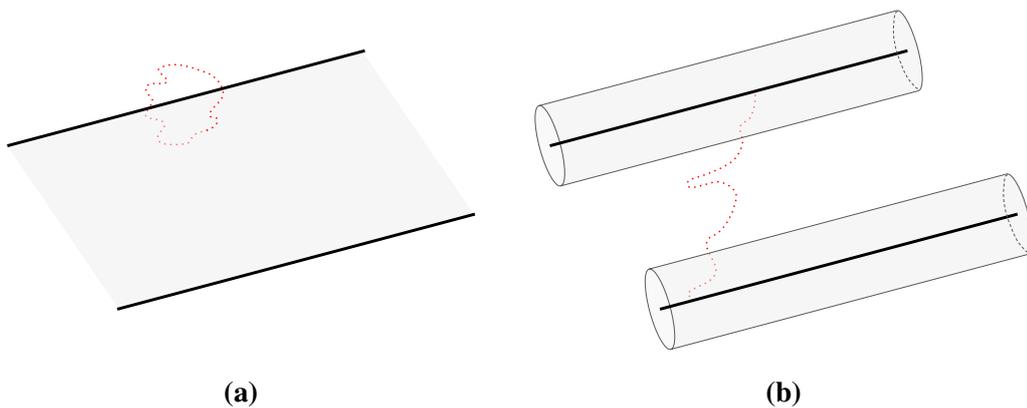
One can consider a lattice where a surface code state is input at one end, the measurements are performed, and a corresponding surface code state is output at the other end [Raussendorf et al., 2005] — this is essentially a quantum teleportation scheme for qubits encoded using the surface code. The primal and dual lattices of the TCS then each have a *correlation surface* linking the input and output surfaces, as shown in Fig. 5.1. The combined parity of the error-corrected measurement outcomes of qubits in each of the correlation surfaces indicates whether a logical Pauli correction is required on the output state to mitigate the effect of errors. Each correlation surface can be deformed to a logically equivalent operator by multiplication with an element of the stabiliser group (i.e. a cube) such that the correlation surface is not unique.

### 5.1.2 Logical qubits and computation

Logical qubits in TCS are created in a similar manner to logical qubits in a defect-based surface code: pairs of one-dimensional defects are carved into the lattice [Raussendorf et al., 2006, Raussendorf et al., 2007]. Defects are created by measuring certain qubits in the  $Z$  basis to disentangle them from the surrounding qubits and remove them from the TCS lattice, with the outcome of each  $Z$  measurement being used to perform a local Pauli correction on the surrounding physical qubits. It is possible to prepare logical qubits in eigenstates of either the logical  $X$  or logical  $Z$  operators, and measurement in either basis can be performed by inverting the preparation procedure.

Correlation surfaces are associated with each pair of defects: there is a corre-

lation surface given by those surfaces which are topologically equivalent to a planar surface between the two defects, such as that shown in Fig. 5.2a, and correlation surfaces topologically equivalent to a cylinder surrounding each defect, such as that shown in Fig. 5.2b. Logical errors occur when an undetectable string of  $Z$  errors runs between two defects, as shown in Fig. 5.2b, or an undetectable, uncontractible string of  $Z$  errors loops around a defect, as shown in Fig. 5.2a (which of these causes a logical  $X$  error and which of these causes a logical  $Z$  error depends on how the logical basis is defined when logical qubits are created). The code distance is therefore dependent on the size of the defects and their separation, with larger defects and greater defect separation required to achieve a larger code distance.



**Figure 5.2:** Abstract representation of correlation surfaces and logical errors for defect-based TCS. Defects are represented by parallel one-dimensional thick black lines, and correlation surfaces are represented by shaded grey regions. Logical error strings are represented by red dotted lines. (a) shows an example of a planar correlation surface between the two defects and a logical error caused by a string of  $Z$  errors surrounding one of the defects. (b) shows an example of the cylindrical correlation surfaces surrounding each defect and shows a logical error caused by a string of  $Z$  errors between the two defects.

As with the surface code, CNOT operations are performed by braiding defects, but in three-dimensional space rather than two-dimensional space plus time. To form a complete universal gate set, logical qubit preparation and measurement and CNOT gates are complemented by magic state distillation [Raussendorf et al., 2007].

## 5.2 Bond loss

This chapter considers the impact of bond failures in TCS schemes, i.e. when certain bonds between qubits are never created. Such errors are relevant to any TCS scheme where entangling operations can fail, particularly linear optics schemes using fusion gates [Gimeno-Segovia et al., 2015]. Our results also provide qualitative insights for surface code schemes with non-deterministic two-qubit gates due to the similarity between TCS and surface codes.

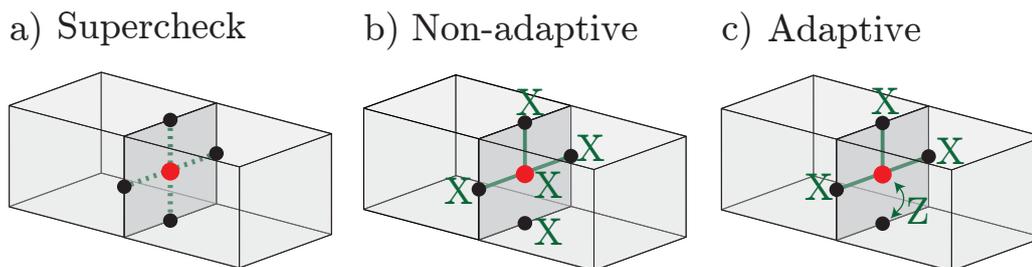
A failed bond has a similar effect to losing the qubits at either end of a successful bond. Qubit loss in TCS was considered in [Barrett and Stace, 2010], which looked at TCS schemes in which all bonds were successful but some qubits were lost before and after bond creation. Lost qubits are handled during the error correction procedure by combining multiple cubes to form *supercheck* operators made up of more than six measurement outcomes: whenever a qubit is lost, the two check operators associated with the adjacent cubes are multiplied together to remove the effect of the lost qubit from the parity check. An example of a supercheck is shown in Fig. 5.3(a). This procedure restores the error correcting properties of the code at the cost of reduced code distance, and tolerates qubit loss rates up to 24.9%.

Later work by [Whiteside and Fowler, 2014] expanded this analysis by considering a gate-based TCS scheme experiencing dynamic loss during all stages of the computation, not just initialisation and measurement. This analysis resulted in a higher effective loss rate per qubit, and correspondingly lower loss threshold of  $2 - 5\%$  *per operation* (rather than per qubit).

Our work mitigates the effect of failed bonds using a similar procedure to [Barrett and Stace, 2010]. To isolate the impact of failed bonds, it is assumed that qubit loss does not occur, and it is assumed that the locations of all failed bonds are known — we refer to this as *heralded* bond loss. Bond failures are heralded in the fusion gate scheme of [Gimeno-Segovia et al., 2015], although it is likely that this scheme would also suffer from qubit loss in realistic scenarios.

We propose two approaches for dealing with failed bonds. In the first method, called the non-adaptive method, every bulk qubit is measured in  $X$  as normal (see

Fig. 5.3(b)). In the second method, called the adaptive method, certain qubits are measured in the  $Z$  basis to remove them from the lattice (see Fig. 5.3(c)). It should be noted that both approaches can also handle qubit loss, in which case there will be a trade-off between tolerable qubit loss rates and bond failure rates; finding the quantitative details of this trade-off will require additional numerical investigations and is left for future work.



**Figure 5.3:** TCS supercheck operators. (a) Multiplying two cubes to form a supercheck removes the face qubit shared between them and results in a parity check involving the  $X$  measurements associated with the ten remaining face qubits. (b) In the non-adaptive approach, all bulk qubits are measured in the  $X$  basis in the presence of a failed bond. (c) In the adaptive approach, one of the qubits incident on the missing bond is randomly chosen to be measured in the  $Z$  basis while the other is measured in the  $X$  basis.

### 5.2.1 Non-adaptive method

In the non-adaptive method, bond failures are mapped onto the qubits by treating the qubit at each end of the bond as a lost qubit in the picture of [Barrett and Stace, 2010]; this means that all additional processing is performed classically during decoding and no extra quantum resources are required.

Each bond touches two qubits: one on the primal and one on the dual lattice. Without loss of generality, we will consider the process of forming superchecks on the primal lattice using the non-adaptive method (an analogous process can be performed for the dual lattice). When a bond fails, the associated primal lattice qubit is removed from the error correction procedure by combining the two incident check operators to form a supercheck, and this process is repeated until all qubits involving failed bonds are removed. If a qubit that is adjacent to a lost bond is part of the correlation surface, the correlation surface is modified by multiplication by an

appropriate check or supercheck operator to remove the qubit from the correlation surface. If the removed qubits form a continuous string percolating the primal lattice such that a correlation surface cannot be formed, a percolation error has occurred and the code is uncorrectable.

### 5.2.2 Adaptive method

In the adaptive method, bond failures are mapped onto the qubits by measuring a qubit at one end of the bond in the  $Z$  basis. The qubit to measure in  $Z$  is chosen at random, such that qubits on the primal and dual lattices are equally likely to be chosen, and a qubit is only measured in  $Z$  if the adjacent qubit has not already been measured in  $Z$ .

In this case, the qubits that are measured in  $Z$  are treated identically to lost qubits in [Barrett and Stace, 2010], and formation of superchecks and correlation surfaces proceeds in the same manner as the non-adaptive scheme except that each failed bond affects only one of the primal or dual lattice at random, not both. A string of removed qubits that percolates the primal or dual lattices results in a percolation error, as in the non-adaptive method, although the rate of percolation errors is lower due to fewer qubits being removed. This adaptive approach leads to an improved threshold at the cost of requiring more quantum processing (i.e. the ability to change measurement basis during the computation).

It should be noted that the adaptive approach remains a measurement-based quantum computing scheme without any additional entangling gates, and measuring qubits in the  $Z$  basis is already required to perform quantum computation (although the locations of such  $Z$  measurements are generally pre-determined).

## 5.3 Threshold simulations

### 5.3.1 Error model

Simulations of both methods have been performed using the error model summarised in Table 5.1 to obtain error correction thresholds. In the simulations, each bond has a probability  $p_{\text{bond}}$  of failing; a failed bond is considered to have never existed. All bond failures occur independently and are heralded. Additionally, each measurement

outcome has an independent probability  $p_{\text{comp}}$  of being incorrect. This model is chosen to give an indication of the effect of failed bonds without considering a specific implementation. For example, one could assign a Pauli error probability to each CPHASE gate when constructing the lattice, but such a model is not appropriate for the scheme in [Gimeno-Segovia et al., 2015], where CPHASE gates are not used. The chosen error model is qualitatively similar to the *random-plaquette gauge model* used in [Wang et al., 2003] for the toric code, and gives a similar threshold in the absence of failed bonds.

Operation	Error model
Bond creation	Bond creation fails with probability $p_{\text{bond}}$ for each bond except those involving only qubits in the first two or final two layers. The locations of failed bonds are known to the decoder.
Qubit measurement	Incorrect measurement outcome with probability $p_{\text{comp}}$ , except red qubits in the first and final layers and black qubits in the second and penultimate layers

**Table 5.1:** Error model for TCS simulations with bond failures.

### 5.3.2 Simulation methods

An overview of the simulation is shown in Fig. 5.4. The simulations use lattices with a range of code distances  $d$ , with the first ‘input’ layer acting as a surface code state. This layer is followed by  $4d - 2$  layers of qubits, finishing with an ‘output’ surface code layer ( $4d - 1$  layers in total), giving the lattice a depth of  $2d$  cubes. For simplicity, we assume that bonds involving only qubits in the first two or final two layers always succeed, and measurements of the red qubits in the first and final layers and black qubits in the second and penultimate layers are always perfect; this is to ensure that the code is projected into a valid surface code state at each end following all measurements, as is standard with many quantum error correction simulations (in practice far more than  $4d$  layers would be involved in a computation).

Except for the  $Z$  basis measurements in the adaptive scheme, all qubits in the bulk of the lattice are measured in the  $X$  basis. The black qubits of the first and final layers are not measured as these form the data qubits of the input and output surface

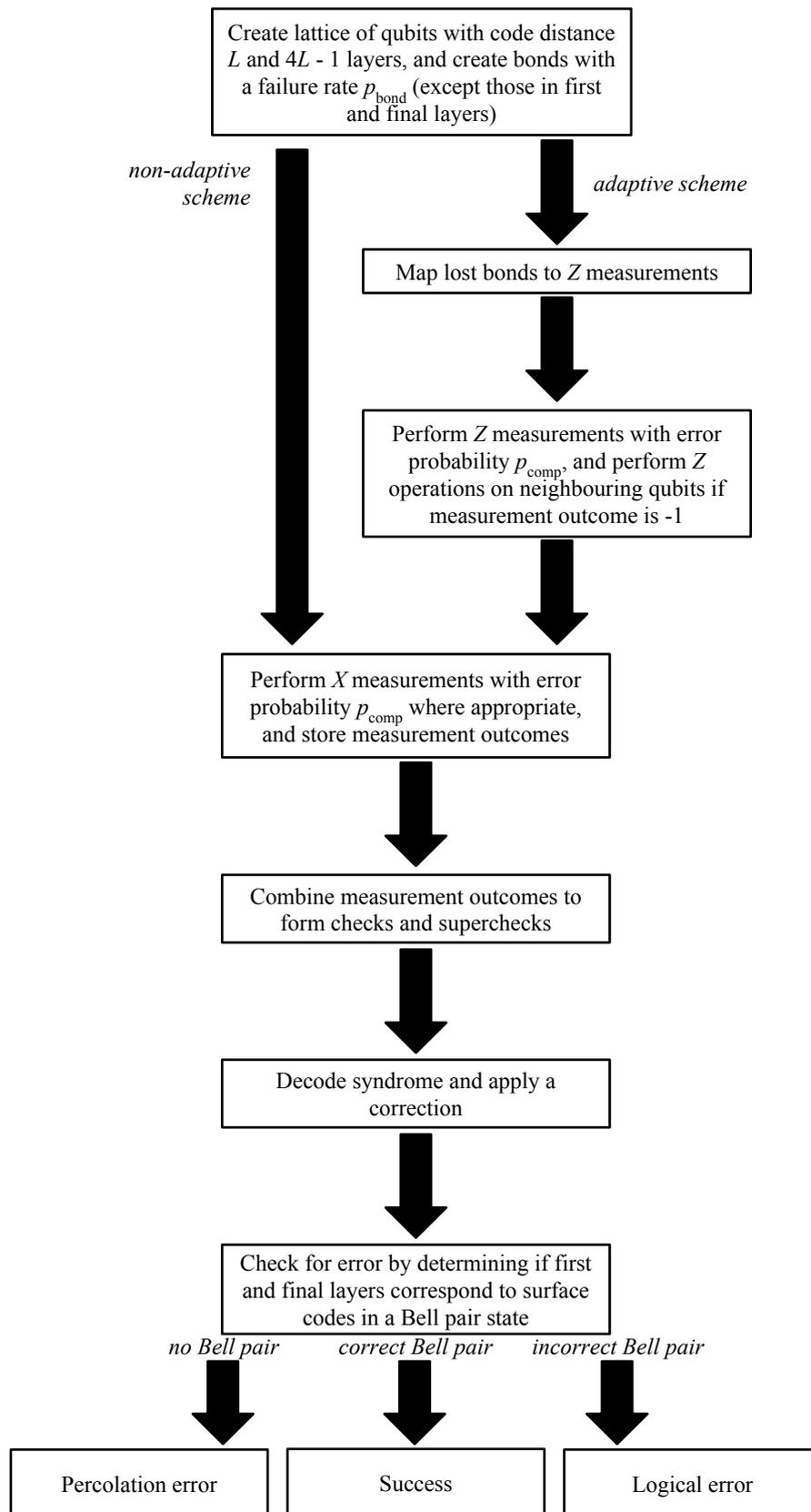


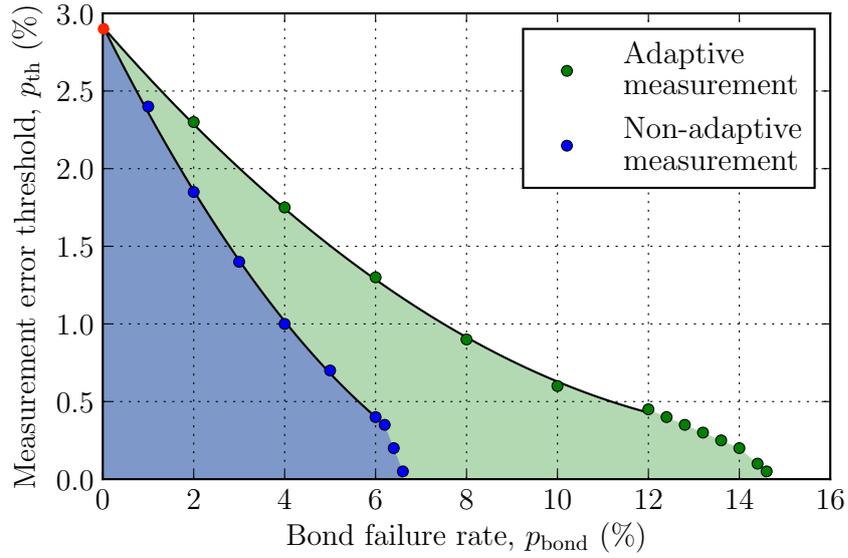
Figure 5.4: Steps for simulating TCS with bond failures.

codes. This leaves the first and final layers in surface code states, with the encoded state forming a Bell pair if a percolation error does not occur [Raussendorf et al., 2005]. The order in which gates are performed and the order in which measurements are performed are unimportant for the chosen error model provided that gates precede measurements on any particular qubit; the simulation creates bonds in an arbitrary order and then performs all measurements in an arbitrary order, but it would be equally valid to alternate rounds of bond creation and measurement.

*GraphSim* [Anders and Briegel, 2006] is used to track the quantum state of the TCS. The decoder uses the knowledge of the locations of failed bonds and mappings outlined above to form superchecks, and a minimum-weight perfect matching algorithm [Raussendorf et al., 2006, Raussendorf et al., 2007] based on *Blossom V* [Kolmogorov, 2009] is used to calculate the required correction, with the edge weights set using the methods in [Barrett and Stace, 2010]. If the resulting logical Bell pair between the surface codes of the first and final layers is  $|\Phi^+\rangle_L = \frac{1}{\sqrt{2}}(|00\rangle_L + |11\rangle_L)$ , the error correction is deemed successful. If the resulting Bell pair is not  $|\Phi^+\rangle_L$  or a percolation error occurs, then the error correction is deemed unsuccessful and a logical error has occurred. For each value of  $p_{\text{bond}}$ , a computational threshold,  $p_{\text{th}}$ , is determined from the intersection point of logical error rates for code distances of 7, 9, 11 and 13 [Wang et al., 2003], much like for the surface code simulations in the previous chapters. Each combination of  $d$ ,  $p_{\text{comp}}$  and  $p_{\text{bond}}$  is simulated for a minimum of  $2 \times 10^4$  runs to obtain clean thresholds.

## 5.4 Results

Fig. 5.5 shows the threshold results of the simulations for the non-adaptive and adaptive methods. In the absence of failed bonds, the error model results in a threshold of  $p_{\text{th}} \approx 2.9\%$ , in agreement with [Wang et al., 2003]. The threshold decreases with increasing bond failure rates for both schemes; the non-adaptive scheme has a fit of  $p_{\text{th}} = 0.029 - 0.587p_{\text{bond}} + 2.786p_{\text{bond}}^2$  applied between  $p_{\text{bond}} = 0\%$  and  $p_{\text{bond}} = 6\%$ , and the adaptive scheme has a fit of  $p_{\text{th}} = 0.029 - 0.336p_{\text{bond}} + 1.071p_{\text{bond}}^2$  between  $p_{\text{bond}} = 0\%$  and  $p_{\text{bond}} = 12\%$ . The threshold for the non-



**Figure 5.5:** Thresholds in the presence of failed bonds. The shaded regions indicate correctable error rate combinations. In the absence of bond failures ( $p_{\text{bond}} = 0$ ), the threshold (red) agrees with [Wang et al., 2003].

adaptive and adaptive schemes disappears at  $p_{\text{bond}} \approx 6.5\%$  and  $p_{\text{bond}} \approx 14.5\%$ , respectively; these limits are due to the percolation threshold for each method.

## 5.5 Discussion

To link these results to those in [Barrett and Stace, 2010], we consider an approximate mapping from the 24.9% percolation limit found for qubit loss (our error model results in an effective loss rate per qubit rather than per operation, so the threshold in [Barrett and Stace, 2010] is more relevant to our analysis than that in [Whiteside and Fowler, 2014]). In the non-adaptive approach, each bulk qubit has four bonds, and qubits with failed bonds are treated equivalently to lost qubits in [Barrett and Stace, 2010]. The probability of a bulk qubit having one or more failed bonds is  $1 - (1 - p_{\text{bond}})^4$ , resulting in an expected percolation threshold when  $1 - (1 - p_{\text{bond}})^4 = 0.249$ , or  $p_{\text{bond}} = 6.9\%$ , which is close to the value obtained.

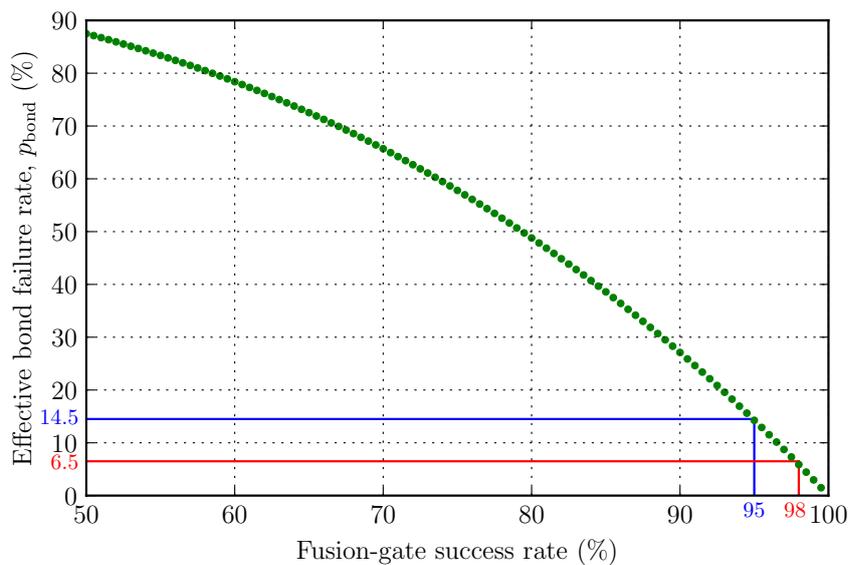
For the adaptive scheme, each failed bond is mapped to just one of the two adjacent qubits. Therefore, the probability of a qubit being measured in the  $Z$  basis receives a contribution of  $\frac{1}{2}p_{\text{bond}}$  from each incident bond. The probability that a particular qubit with four attempted bonds is measured in  $Z$  is therefore

$1 - (1 - \frac{1}{2}p_{\text{bond}})^4$ , resulting in an expected percolation threshold when  $1 - (1 - \frac{1}{2}p_{\text{bond}})^4 = 0.249$ , or  $p_{\text{bond}} = 13.8\%$ . This is slightly lower than the value obtained in the simulations because it does not account for qubits that have multiple lost bonds (i.e. there is not a one-to-one mapping between the number of missing bonds and the number of qubits measured in  $Z$  — the number of qubits measured in  $Z$  will sometimes be less than the number of lost bonds).

### 5.5.1 Fault-tolerant linear optical quantum computation

The phenomenological bond loss model assumed in this chapter can be connected to the microcluster scheme in [Gimeno-Segovia et al., 2015], which generates large-scale cluster states from elementary three-qubit GHZ resource states by performing a sequence of probabilistic fusion gates. To do so, we have performed additional numerical simulations on a modified version of this microcluster scheme to find the relationship between fusion-gate success rates and bond failure rates. The simulation involves using fusion gates to repeatedly create TCS lattices (unlike the brickwork lattice used in the original linear optical proposal) with code distance  $d = 6$  and fusion-gate success rates ranging from 50% to 99.5% in steps of 0.5%. The proportion of missing bonds is measured for each lattice, and this is then averaged over all runs for each fusion-gate success rate to give an effective bond failure rate.

The results of the simulation, shown in Fig. 5.6, suggest that the adaptive scheme would require a fusion-gate success rate in excess of 95% to perform fault-tolerant quantum computation, and the non-adaptive scheme would require a fusion-gate success rate in excess of 98%. The fusion-gate success rate can be increased to this level by using larger resources to perform the gate, and although this increases the resources *per fusion gate*, it is not clear that this will increase the overall resources required, as the scheme we present would not require additional procedures to perform fault-tolerant quantum computation, unlike that in [Gimeno-Segovia et al., 2015].



**Figure 5.6:** Simulation results showing the relationship between fusion-gate success rates and bond failure rates when building a linear optics device like that in [Gimeno-Segovia et al., 2015]. The blue and red lines illustrate the minimum fusion-gate success rates required for the adaptive and non-adaptive methods respectively.

## 5.6 Conclusion

In this chapter, we have shown that deterministic entangling gates are not required to perform fault-tolerant universal quantum computation. By adapting TCS schemes, probabilistic heralded entangling gate failure rates as high as 6.5% can be tolerated with no additional quantum overhead at the cost of a lower measurement error threshold, and rates as high as 14.5% can be tolerated if this restriction is relaxed to allow adaptive measurements.

The error model used in the simulations has been chosen primarily as a toy model for the linear optics scheme in [Gimeno-Segovia et al., 2015], so our findings are most relevant to linear optical systems, although the model does not include qubit loss, which is likely to occur in realistic scenarios. Despite being motivated by linear optics, our approach for dealing with bond failures is sufficiently general that it can be applied to any TCS scheme with heralded non-deterministic entangling gates, albeit with different thresholds, as every scheme will undergo a different noise model and experience different error propagation.

Additionally, the shared features of topological codes mean that our results give a qualitative insight for other topological codes, such as the surface code, with non-deterministic two-qubit gates; the error model used in this chapter is similar to a surface code experiencing entanglement failure and data qubit leakage.

Avenues for future work in this area include considering unheralded entanglement failure, where the locations of missing bonds are unknown, and combining this approach with a scheme such as that of [Nickerson et al., 2014] to attempt to reduce qubit and time overheads for networks of trapped ions with non-deterministic entanglement between groups of qubits.

## Chapter 6

# Summary and outlook

In this thesis, we have considered the challenges involved in performing fault-tolerant universal quantum computation with realistic error models that take account of more than just decoherence during computation, and we have suggested methods for handling such errors with topological error correction schemes.

In Chapter 3, we presented a blueprint for fault-tolerant quantum computation with Rydberg atoms and outlined some of the benefits and challenges of such a scheme compared to other physical realisations. We then performed a gate-based threshold simulation of the scheme to obtain an error threshold of 1.25% for depolarising noise, a result that suggests gate fidelities need to be improved substantially before it is viable to build a surface code quantum computer using Rydberg atoms. Nonetheless, Rydberg atoms remain a promising candidate for realising fault-tolerant universal quantum computation in the mid- to long-term as they have good potential for scalability once quantum operations can be performed with sufficiently low error rates.

In Chapter 4, we introduced a novel method for handling permanent fabrication errors on the surface code that result from imperfect manufacturing processes. Our approach is sufficiently general that it can be applied to a surface code quantum computer based on almost any physical system with no additional quantum processing. Qubit fabrication error rates as high as 8% and entangling gate (link) fabrication error rates as high as 10% can be tolerated, although both of these come at the cost of reduced Pauli error thresholds. Combinations of both link and qubit fabrication

errors can also be handled using this approach, but the trade-off between the two was not investigated in this thesis and is left for future work.

Finally, in Chapter 5, we showed that it is possible to perform universal fault-tolerant quantum computation without deterministic entangling gates. We obtained error thresholds that indicate probabilistic entanglement failure rates as high as 6.5% can be tolerated without any additional quantum processing, and rates as high as 14.5% can be tolerated by relaxing this requirement and allowing measurement in an alternative basis. These findings are particularly important for linear optical schemes and may also be of relevance to implementations based on networks of trapped ions.

There are several areas for future work to follow up on our findings. One key area to investigate is unheralded fabrication errors and bond failures, where the locations of failed components are not known. This is important because it may not always be possible to determine with certainty whether a component works reliably, and it may be difficult or impossible to identify components that become faulty after manufacture or during a computation. Topological cluster state schemes are likely to be robust enough to handle a level of unheralded bond loss, as this will be much like using the adaptive method from Chapter 5 without some of the decoder optimisations and would therefore simply result in a lower computational error threshold. However, the case is less clear for surface code devices with permanent unheralded fabrication errors as such faults will create undetectable logical qubits that may interact with and corrupt the encoded information — these errors would therefore be extremely damaging, so it is important either to prevent them from occurring or to find methods to mitigate their effects.

Another area of future work is to find approaches to overcome the problem of supercheck operator weights increasing with code distance, as identified in Section 4.4.1. This problem, which is not unique to our results, means that the supercheck approach might not be indefinitely scalable and it is possible that it will fail for large codes once the supercheck operators become sufficiently large that they cannot be measured reliably.

The findings in this thesis demonstrate the versatility of topological error correction schemes such as the surface code: these schemes were originally proposed to mitigate the effects of quantum decoherence, but we have shown that, with minimal modification, they can also handle errors that at first glance appear to undermine their fundamental topology. As such, these codes remain among the forerunners for realising fault-tolerant universal quantum computation.

Although parts of this thesis have been motivated by schemes based around Rydberg atoms and linear optics, the author considers the front-runner for building a scalable universal fault-tolerant computer to be superconducting qubits, which have already been used to experimentally demonstrate error suppression with a linear array of nine qubits [Kelly et al., 2015], albeit only bit-flip errors and not phase-flip errors.

# Bibliography

- [Aaronson and Arkhipov, 2011] Aaronson, S. and Arkhipov, A. (2011). The computational complexity of linear optics. In *Proceedings of the Forty-third Annual ACM Symposium on Theory of Computing*, STOC '11, pages 333–342, New York, NY, USA. ACM.
- [Aaronson and Gottesman, 2004] Aaronson, S. and Gottesman, D. (2004). Improved simulation of stabilizer circuits. *Phys. Rev. A*, 70:052328, arXiv:quant-ph/0406196.
- [Aharonov and Ben-Or, 2008] Aharonov, D. and Ben-Or, M. (2008). Fault-Tolerant quantum computation with constant error rate. *SIAM J. Comput.*, 38(4):1207–1282, arXiv:quant-ph/9906129.
- [Aharonov et al., 1993] Aharonov, Y., Davidovich, L., and Zagury, N. (1993). Quantum random walks. *Phys. Rev. A*, 48(2):1687–1690.
- [Alberti et al., 2016] Alberti, A., Robens, C., Alt, W., Brakhane, S., Karski, M., Reimann, R., Widera, A., and Meschede, D. (2016). Super-resolution microscopy of single atoms in optical lattices. *New J. Phys.*, 18(5):053010, arXiv:1512.07329 [quant-ph].
- [Anders and Briegel, 2006] Anders, S. and Briegel, H. J. (2006). Fast simulation of stabilizer circuits using a graph-state representation. *Phys. Rev. A*, 73(2):022334, arXiv:quant-ph/0504117.

- [Bacon, 2006] Bacon, D. (2006). Operator quantum error-correcting subsystems for self-correcting quantum memories. *Phys. Rev. A*, 73(1):012340, arXiv:quant-ph/0506023.
- [Ballance et al., 2016] Ballance, C. J., Harty, T. P., Linke, N. M., Sepiol, M. A., and Lucas, D. M. (2016). High-Fidelity quantum logic gates using Trapped-Ion hyperfine qubits. *Phys. Rev. Lett.*, 117(6):060504, arXiv:1512.04600 [quant-ph].
- [Barenco et al., 1995] Barenco, A., Bennett, C. H., Cleve, R., DiVincenzo, D. P., Margolus, N., Shor, P., Sleator, T., Smolin, J. A., and Weinfurter, H. (1995). Elementary gates for quantum computation. *Phys. Rev. A*, 52(5):3457–3467, arXiv:quant-ph/9503016.
- [Barends et al., 2014] Barends, R., Kelly, J., Megrant, A., Veitia, A., Sank, D., Jeffrey, E., White, T. C., Mutus, J., Fowler, A. G., Campbell, B., Chen, Y., Chen, Z., Chiaro, B., Dunsworth, A., Neill, C., O’Malley, P., Roushan, P., Vainsencher, A., Wenner, J., Korotkov, A. N., Cleland, A. N., and Martinis, J. M. (2014). Superconducting quantum circuits at the surface code threshold for fault tolerance. *Nature*, 508(7497):500–503, arXiv:1402.4848 [quant-ph].
- [Barredo et al., 2016] Barredo, D., de Léséleuc, S., Lienhard, V., Lahaye, T., and Browaeys, A. (2016). An atom-by-atom assembler of defect-free arbitrary two-dimensional atomic arrays. *Science*, 354(6315):1021–1023, arXiv:1607.03042 [quant-ph].
- [Barrett and Stace, 2010] Barrett, S. D. and Stace, T. M. (2010). Fault tolerant quantum computation with very high threshold for loss errors. *Phys. Rev. Lett.*, 105:200502, arXiv:1005.2456 [quant-ph].
- [Bazant, 2000] Bazant, M. Z. (2000). Largest cluster in subcritical percolation. *Phys. Rev. E*, 62:1660, arXiv:cond-mat/9905191.
- [Bernien et al., 2017] Bernien, H., Schwartz, S., Keesling, A., Levine, H., Omran, A., Pichler, H., Choi, S., Zibrov, A. S., Endres, M., Greiner, M., Vuletić, V.,

- and Lukin, M. D. (2017). Probing many-body dynamics on a 51-atom quantum simulator. [arXiv:1707.04344](#) [quant-ph].
- [Beterov et al., 2009] Beterov, I. I., Ryabtsev, I. I., Tretyakov, D. B., and Entin, V. M. (2009). Quasiclassical calculations of blackbody-radiation-induced depopulation rates and effective lifetimes of Rydberg  $nS$ ,  $nP$ , and  $nD$  alkali-metal atoms with  $n \leq 80$ . *Phys. Rev. A*, 79(5):052504, [arXiv:0810.0339](#) [physics.atom-ph].
- [Beterov and Saffman, 2015] Beterov, I. I. and Saffman, M. (2015). Rydberg blockade, Forster resonances, and quantum state measurements with different atomic species. *Phys. Rev. A*, 92(4):042710, [arXiv:1508.07111](#) [quant-ph].
- [Bombin, 2010] Bombin, H. (2010). Topological order with a twist: Ising anyons from an abelian model. *Phys. Rev. Lett.*, 105(3):030403, [arXiv:1004.1838](#) [cond-mat.str-el].
- [Bombin and Martin-Delgado, 2006] Bombin, H. and Martin-Delgado, M. A. (2006). Topological quantum distillation. *Phys. Rev. Lett.*, 97(18):180501, [arXiv:quant-ph/0605138](#).
- [Boykin et al., 1999] Boykin, P. O., Mor, T., Pulver, M., Roychowdhury, V., and Vatan, F. (1999). On universal and fault-tolerant quantum computing: a novel basis and a new constructive proof of universality for Shor's basis. In *40th Annual Symposium on Foundations of Computer Science (Cat. No.99CB37039)*, pages 486–494.
- [Bravyi and Kitaev, 2005] Bravyi, S. and Kitaev, A. (2005). Universal quantum computation with ideal Clifford gates and noisy ancillas. *Phys. Rev. A*, 71(2):022316, [arXiv:quant-ph/0403025](#).
- [Bravyi et al., 2014] Bravyi, S., Suchara, M., and Vargo, A. (2014). Efficient algorithms for maximum likelihood decoding in the surface code. *Phys. Rev. A*, 90(3):032326, [arXiv:1405.4883](#) [quant-ph].

- [Bravyi and Vargo, 2013] Bravyi, S. and Vargo, A. (2013). Simulation of rare events in quantum error correction. *Phys. Rev. A*, 88:062308, arXiv:1308.6270 [quant-ph].
- [Bravyi and Yu. Kitaev, 1998] Bravyi, S. B. and Yu. Kitaev, A. (1998). Quantum codes on a lattice with boundary. arXiv:quant-ph/9811052.
- [Briegel and Raussendorf, 2001] Briegel, H. J. and Raussendorf, R. (2001). Persistent entanglement in arrays of interacting particles. *Phys. Rev. Lett.*, 86(5):910–913, arXiv:quant-ph/0004051.
- [Brion et al., 2007] Brion, E., Pedersen, L. H., Mølmer, K., Chutia, S., and Saffman, M. (2007). Universal quantum computation in a neutral-atom decoherence-free subspace. *Phys. Rev. A*, 75(3):032328, arXiv:quant-ph/0611246.
- [Brion et al., 2008] Brion, E., Pedersen, L. H., Saffman, M., and Mølmer, K. (2008). Error correction in ensemble registers for quantum repeaters and quantum computers. *Phys. Rev. Lett.*, 100(11):110506, arXiv:0710.1717 [quant-ph].
- [Browne and Rudolph, 2005] Browne, D. E. and Rudolph, T. (2005). Resource-efficient linear optical quantum computation. *Phys. Rev. Lett.*, 95(1):010501, arXiv:quant-ph/0405157.
- [Chiaverini et al., 2004] Chiaverini, J., Leibfried, D., Schaetz, T., Barrett, M. D., Blakestad, R. B., Britton, J., Itano, W. M., Jost, J. D., Knill, E., Langer, C., et al. (2004). Realization of quantum error correction. *Nature*, 432(7017):602–605.
- [Cirac and Zoller, 1995] Cirac, J. I. and Zoller, P. (1995). Quantum computations with cold trapped ions. *Phys. Rev. Lett.*, 74(20):4091–4094.
- [Clarke and Wilhelm, 2008] Clarke, J. and Wilhelm, F. K. (2008). Superconducting quantum bits. *Nature*, 453(7198):1031–1042.
- [Córcoles et al., 2015] Córcoles, A. D., Magesan, E., Srinivasan, S. J., Cross, A. W., Steffen, M., Gambetta, J. M., and Chow, J. M. (2015). Demonstration of a

- quantum error detection code using a square lattice of four superconducting qubits. *Nat. Commun.*, 6:6979, arXiv:1410.6419 [quant-ph].
- [Cory et al., 1998] Cory, D. G., Price, M. D., Maas, W., Knill, E., Laflamme, R., Zurek, W. H., Havel, T. F., and Somaroo, S. S. (1998). Experimental quantum error correction. *Physical Review Letters*, 81(10):2152, arXiv:quant-ph/9802018.
- [Criger and Terhal, 2016] Criger, B. and Terhal, B. (2016). Noise thresholds for the  $[[4, 2, 2]]$ -concatenated toric code. *QIC*, 16(15&16):1261–1281, arXiv:1604.04062 [quant-ph].
- [Crow et al., 2016] Crow, D., Joynt, R., and Saffman, M. (2016). Improved error thresholds for Measurement-Free error correction. *Phys. Rev. Lett.*, 117(13):130503, arXiv:1510.08359 [quant-ph].
- [Dennis et al., 2002] Dennis, E., Kitaev, A., Landahl, A., and Preskill, J. (2002). Topological quantum memory. *J. Math. Phys.*, 43(9):4452–4505, arXiv:quant-ph/0110143.
- [Deutsch, 1985] Deutsch, D. (1985). Quantum theory, the Church-Turing principle and the universal quantum computer. *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, 400(1818):97–117.
- [Devoret et al., 2004] Devoret, M. H., Wallraff, A., and Martinis, J. M. (2004). Superconducting qubits: A short review. arXiv:cond-mat/0411174.
- [Duclos-Cianci and Poulin, 2010] Duclos-Cianci, G. and Poulin, D. (2010). Fast decoders for topological quantum codes. *Phys. Rev. Lett.*, 104(5):050504, arXiv:0911.0581 [quant-ph].
- [Farhi et al., 2000] Farhi, E., Goldstone, J., Gutmann, S., and Sipser, M. (2000). Quantum computation by adiabatic evolution. arXiv:quant-ph/0001106.
- [Feynman, 1982] Feynman, R. P. (1982). Simulating physics with computers. *Int. J. Theor. Phys.*, 21(6-7):467–488.

- [Feynman, 1986] Feynman, R. P. (1986). Quantum mechanical computers. *Found. Phys.*, 16(6):507–531.
- [Finnila et al., 1994] Finnila, A. B., Gomez, M. A., Sebenik, C., Stenson, C., and Doll, J. D. (1994). Quantum annealing: A new method for minimizing multidimensional functions. *Chem. Phys. Lett.*, 219(5):343–348, arXiv:chem-ph/9404003.
- [Fowler, 2013a] Fowler, A. G. (2013a). Coping with qubit leakage in topological codes. *Phys. Rev. A*, 88(4):042308, arXiv:1308.6642 [quant-ph].
- [Fowler, 2013b] Fowler, A. G. (2013b). Optimal complexity correction of correlated errors in the surface code. arXiv:1310.0863 [quant-ph].
- [Fowler et al., 2012] Fowler, A. G., Mariantoni, M., Martinis, J. M., and Cleland, A. N. (2012). Surface codes: Towards practical large-scale quantum computation. *Phys. Rev. A*, 86(3):032324, arXiv:1208.0928 [quant-ph].
- [Fowler and Martinis, 2014] Fowler, A. G. and Martinis, J. M. (2014). Quantifying the effects of local many-qubit errors and nonlocal two-qubit errors on the surface code. *Phys. Rev. A*, 89(3):032316, arXiv:1401.2466 [quant-ph].
- [Fowler et al., 2009] Fowler, A. G., Stephens, A. M., and Groszkowski, P. (2009). High-threshold universal quantum computation on the surface code. *Phys. Rev. A*, 80(5):052312, arXiv:0803.0272 [quant-ph].
- [Fowler et al., 2010] Fowler, A. G., Wang, D. S., Hill, C. D., Ladd, T. D., Van Meter, R., and Hollenberg, L. C. L. (2010). Surface code quantum communication. *Phys. Rev. Lett.*, 104(18):180503, arXiv:0910.4074 [quant-ph].
- [Fujii and Tokunaga, 2012] Fujii, K. and Tokunaga, Y. (2012). Error and loss tolerances of surface codes with general lattice structures. *Phys. Rev. A*, 86:020303, arXiv:1202.2743 [quant-ph].

- [Ghosh and Fowler, 2015] Ghosh, J. and Fowler, A. G. (2015). Leakage-resilient approach to fault-tolerant quantum computing with superconducting elements. *Phys. Rev. A*, 91:020302, arXiv:1406.2404 [quant-ph].
- [Gimeno-Segovia et al., 2015] Gimeno-Segovia, M., Shadbolt, P., Browne, D. E., and Rudolph, T. (2015). From Three-Photon Greenberger-Horne-Zeilinger states to ballistic universal quantum computation. *Phys. Rev. Lett.*, 115(2):020502, arXiv:1410.3720 [quant-ph].
- [Gottesman, 1997] Gottesman, D. (1997). Stabilizer codes and quantum error correction. arXiv:quant-ph/9705052.
- [Gottesman, 1998] Gottesman, D. (1998). The heisenberg representation of quantum computers. arXiv:quant-ph/9807006.
- [Grover, 1996] Grover, L. K. (1996). A fast quantum mechanical algorithm for database search. In *Proceedings of the Twenty-eighth Annual ACM Symposium on Theory of Computing, STOC '96*, pages 212–219, New York, NY, USA. ACM.
- [Häffner et al., 2008] Häffner, H., Roos, C. F., and Blatt, R. (2008). Quantum computing with trapped ions. *Phys. Rep.*, 469(4):155–203, arXiv:0809.4368 [quant-ph].
- [Harrow et al., 2009] Harrow, A. W., Hassidim, A., and Lloyd, S. (2009). Quantum algorithm for linear systems of equations. *Phys. Rev. Lett.*, 103(15):150502, arXiv:0811.3171 [quant-ph].
- [Hastings and Geller, 2015] Hastings, M. B. and Geller, A. (2015). Reduced space-time and time costs using dislocation codes and arbitrary ancillas. *Quantum Inf. Comput.*, 15(11-12):962–986, arXiv:1408.3379 [quant-ph].
- [Herold et al., 2015] Herold, M., Campbell, E. T., Eisert, J., and Kastoryano, M. J. (2015). Cellular-automaton decoders for topological quantum memories. *npj Quantum Information*, 1:15010, arXiv:1406.2338 [quant-ph].

- [Horsman et al., 2012] Horsman, C., Fowler, A. G., Devitt, S., and Van Meter, R. (2012). Surface code quantum computing by lattice surgery. *New J. Phys.*, 14(12):123011, arXiv:1111.4022 [quant-ph].
- [Isenhower et al., 2011] Isenhower, L., Saffman, M., and Mølmer, K. (2011). Multi-bit CkNOT quantum gates via Rydberg blockade. *Quantum Inf. Process.*, 10(6):755, arXiv:1104.3916 [quant-ph].
- [Jau et al., 2015] Jau, Y.-Y., Hankin, A. M., Keating, T., Deutsch, I. H., and Biedermann, G. W. (2015). Entangling atomic spins with a Rydberg-dressed spin-flip blockade. *Nat. Phys.*, 12(1):71–74, arXiv:1501.03862 [quant-ph].
- [Jin et al., 2012] Jin, X.-M., Yi, Z.-H., Yang, B., Zhou, F., Yang, T., and Peng, C.-Z. (2012). Experimental quantum error detection. *Scientific reports*, 2:626.
- [Kadowaki and Nishimori, 1998] Kadowaki, T. and Nishimori, H. (1998). Quantum annealing in the transverse Ising model. *Phys. Rev. E*, 58(5):5355–5363, arXiv:cond-mat/9804280.
- [Kelly et al., 2015] Kelly, J., Barends, R., Fowler, A. G., Megrant, A., Jeffrey, E., White, T. C., Sank, D., Mutus, J. Y., Campbell, B., Chen, Y., Chen, Z., Chiaro, B., Dunsworth, A., Hoi, I.-C., Neill, C., O’Malley, P. J. J., Quintana, C., Roushan, P., Vainsencher, A., Wenner, J., Cleland, A. N., and Martinis, J. M. (2015). State preservation by repetitive error detection in a superconducting quantum circuit. *Nature*, 519(7541):66–69, arXiv:1411.7403 [quant-ph].
- [King et al., 2015] King, J., Yarkoni, S., Nevisi, M. M., Hilton, J. P., and McGeoch, C. C. (2015). Benchmarking a quantum annealing processor with the time-to-target metric. arXiv:1508.05087 [quant-ph].
- [Kitaev, 1997] Kitaev, A. Y. (1997). Quantum computations: algorithms and error correction. *Russian Math. Surveys*, 52(6):1191.
- [Kitaev, 2003] Kitaev, A. Y. (2003). Fault-tolerant quantum computation by anyons. *Ann. Phys.*, 303(1):2–30, arXiv:quant-ph/9707021.

- [Knill, 2005] Knill, E. (2005). Quantum computing with realistically noisy devices. *Nature*, 434(7029):39, arXiv:quant-ph/0410199.
- [Knill and Laflamme, 1996] Knill, E. and Laflamme, R. (1996). Concatenated quantum codes. arXiv:quant-ph/9608012.
- [Knill and Laflamme, 1998] Knill, E. and Laflamme, R. (1998). Power of one bit of quantum information. *Phys. Rev. Lett.*, 81(25):5672–5675, arXiv:quant-ph/9802037.
- [Knill et al., 2001] Knill, E., Laflamme, R., and Milburn, G. J. (2001). A scheme for efficient quantum computation with linear optics. *Nature*, 409(6816):46–52.
- [Kok et al., 2007] Kok, P., Munro, W. J., Nemoto, K., Ralph, T. C., Dowling, J. P., and Milburn, G. J. (2007). Linear optical quantum computing with photonic qubits. *Rev. Mod. Phys.*, 79(1):135–174, quant-ph/0512071.
- [Kolmogorov, 2009] Kolmogorov, V. (2009). Blossom V: a new implementation of a minimum cost perfect matching algorithm. *Math. Prog. Comput.*, 1(1):43.
- [Li et al., 2010] Li, Y., Barrett, S. D., Stace, T. M., and Benjamin, S. C. (2010). Fault tolerant quantum computation with nondeterministic gates. *Phys. Rev. Lett.*, 105(25):250502, arXiv:1008.1369 [quant-ph].
- [Lim et al., 2006] Lim, Y. L., Barrett, S. D., Beige, A., Kok, P., and Kwok, L. C. (2006). Repeat-until-success quantum computing using stationary and flying qubits. *Phys. Rev. A*, 73(1):012304, arXiv:quant-ph/0508218.
- [Loss and DiVincenzo, 1998] Loss, D. and DiVincenzo, D. P. (1998). Quantum computation with quantum dots. *Phys. Rev. A*, 57(1):120–126, arXiv:cond-mat/9701055.
- [MacCormick et al., 2016] MacCormick, C., Bergamini, S., Mansell, C., Cable, H., and Modi, K. (2016). Supraclassical measurement using single-atom control of an atomic ensemble. *Phys. Rev. A*, 93(2):023805, arXiv:1511.02741 [quant-ph].

- [Maller et al., 2015] Maller, K. M., Lichtman, M. T., Xia, T., Sun, Y., Piotrowicz, M. J., Carr, A. W., Isenhower, L., and Saffman, M. (2015). Rydberg-blockade controlled-not gate and entanglement in a two-dimensional array of neutral-atom qubits. *Phys. Rev. A*, 92(2):022336, arXiv:1506.06416 [quant-ph].
- [Mansell and Bergamini, 2014] Mansell, C. W. and Bergamini, S. (2014). A cold-atoms based processor for deterministic quantum computation with one qubit in intractably large hilbert spaces. *New J. Phys.*, 16(5):053045, arXiv:1309.7920 [quant-ph].
- [Martinez Dorantes, 2016] Martinez Dorantes, M. (2016). *Fast non-destructive internal state detection of neutral atoms in optical potentials*. PhD thesis, Universitäts-und Landesbibliothek Bonn.
- [Montanaro, 2015] Montanaro, A. (2015). Quantum speedup of Monte Carlo methods. *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, 471(2181), arXiv:1504.06987 [quant-ph].
- [Müller et al., 2009] Müller, M., Lesanovsky, I., Weimer, H., Büchler, H. P., and Zoller, P. (2009). Mesoscopic Rydberg gate based on electromagnetically induced transparency. *Phys. Rev. Lett.*, 102(17):170502, arXiv:0811.1155 [quant-ph].
- [Nagayama et al., 2017] Nagayama, S., Fowler, A. G., Horsman, D., Devitt, S. J., and Van Meter, R. (2017). Surface code error correction on a defective lattice. *New J. Phys.*, 19(2):023050, arXiv:1607.00627 [quant-ph].
- [Napp and Preskill, 2013] Napp, J. and Preskill, J. (2013). Optimal Bacon-Shor codes. *Quantum Inf. Comput.*, 13(5-6):490–510, arXiv:1209.0794 [quant-ph].
- [Nickerson et al., 2014] Nickerson, N. H., Fitzsimons, J. F., and Benjamin, S. C. (2014). Freely scalable quantum technologies using cells of 5-to-50 qubits with very lossy and noisy photonic links. *Phys. Rev. X*, 4(4):041041, arXiv:1406.0880 [quant-ph].

- [Nielsen and Chuang, 2000] Nielsen, M. A. and Chuang, I. L. (2000). *Quantum Computation and Quantum Information*. Cambridge University Press.
- [O’Gorman et al., 2016] O’Gorman, J., Nickerson, N. H., Ross, P., Morton, J., and Benjamin, S. C. (2016). A silicon-based surface code quantum computer. *NJP Quantum Inf.*, 2:15019, arXiv:1406.5149 [quant-ph].
- [Poulin, 2005] Poulin, D. (2005). Stabilizer formalism for operator quantum error correction. *Phys. Rev. Lett.*, 95(23):230504, arXiv:quant-ph/0508131.
- [Preskill, 1997] Preskill, J. (1997). Fault-tolerant quantum computation. arXiv:quant-ph/9712048.
- [Raussendorf et al., 2005] Raussendorf, R., Bravyi, S., and Harrington, J. (2005). Long-range quantum entanglement in noisy cluster states. *Phys. Rev. A*, 71(6):062313, arXiv:quant-ph/0407255.
- [Raussendorf and Briegel, 2001] Raussendorf, R. and Briegel, H. J. (2001). A one-way quantum computer. *Phys. Rev. Lett.*, 86(22):5188–5191.
- [Raussendorf and Harrington, 2007] Raussendorf, R. and Harrington, J. (2007). Fault-tolerant quantum computation with high threshold in two dimensions. *Phys. Rev. Lett.*, 98(19):190504, arXiv:quant-ph/0610082.
- [Raussendorf et al., 2006] Raussendorf, R., Harrington, J., and Goyal, K. (2006). A fault-tolerant one-way quantum computer. *Ann. Phys.*, 321(9):2242–2270, arXiv:quant-ph/0510135.
- [Raussendorf et al., 2007] Raussendorf, R., Harrington, J., and Goyal, K. (2007). Topological fault-tolerance in cluster state quantum computation. *New J. Phys.*, 9(6):199, arXiv:quant-ph/0703143.
- [Reed et al., 2012] Reed, M. D., DiCarlo, L., Nigg, S. E., Sun, L., Frunzio, L., Girvin, S. M., and Schoelkopf, R. J. (2012). Realization of three-qubit quantum error correction with superconducting circuits. *Nature*, 482(7385):382–385, arXiv:1109.4948 [quant-ph].

- [Rudolph, 2017] Rudolph, T. (2017). Why I am optimistic about the silicon-photonics route to quantum computing. *APL Photonics*, 2(3):030901, arXiv:1607.08535 [quant-ph].
- [Saffman, 2016] Saffman, M. (2016). Quantum computing with atomic qubits and Rydberg interactions: progress and challenges. *J. Phys. B At. Mol. Opt. Phys.*, 49(20):202001, arXiv:1605.05207 [quant-ph].
- [Saffman et al., 2010] Saffman, M., Walker, T. G., and Mølmer, K. (2010). Quantum information with Rydberg atoms. *Rev. Mod. Phys.*, 82(3):2313–2363, arXiv:0909.4777 [quant-ph].
- [Shor, 1997] Shor, P. W. (1997). Polynomial-Time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM J. Comput.*, 26(5):1484–1509, arXiv:quant-ph/9508027.
- [Singer et al., 2005] Singer, K., Stanojevic, J., Weidemüller, M., and Côté, R. (2005). Long-range interactions between alkali Rydberg atom pairs correlated to the ns–ns, np–np and nd–nd asymptotes. *J. Phys. B At. Mol. Opt. Phys.*, 38(2):S295.
- [Stace and Barrett, 2010] Stace, T. M. and Barrett, S. D. (2010). Error correction and degeneracy in surface codes suffering loss. *Phys. Rev. A*, 81(2):022317, arXiv:0912.1159 [quant-ph].
- [Stace et al., 2009] Stace, T. M., Barrett, S. D., and Doherty, A. C. (2009). Thresholds for topological codes in the presence of loss. *Phys. Rev. Lett.*, 102:200501, arXiv:0904.3556 [quant-ph].
- [Stephens, 2014] Stephens, A. M. (2014). Fault-tolerant thresholds for quantum error correction with the surface code. *Phys. Rev. A*, 89(2):022321, arXiv:1311.5003 [quant-ph].

- [Suchara et al., 2015] Suchara, M., Cross, A. W., and Gambetta, J. M. (2015). Leakage suppression in the toric code. *Quantum Info. Comput.*, 15(11-12):997, arXiv:1410.8562 [quant-ph].
- [Sykes and Essam, 1964] Sykes, M. F. and Essam, J. W. (1964). Exact critical percolation probabilities for site and bond problems in two dimensions. *J. Math. Phys.*, 5(8):1117–1127.
- [Tang and Miao, 2016] Tang, Y.-C. and Miao, G.-X. (2016). Robust surface code topology against sparse fabrication defects in a superconducting-qubit array. *Phys. Rev. A*, 93:032322.
- [Wallquist et al., 2009] Wallquist, M., Hammerer, K., Rabl, P., Lukin, M., and Zoller, P. (2009). Hybrid quantum devices and quantum engineering. *Phys. Scr.*, 2009(T137):014001, arXiv:0911.3835 [quant-ph].
- [Wang et al., 2003] Wang, C., Harrington, J., and Preskill, J. (2003). Confinement-Higgs transition in a disordered gauge theory and the accuracy threshold for quantum memory. *Ann. Phys.*, 303(1):31–58, arXiv:quant-ph/0207088.
- [Wang et al., 2011] Wang, D. S., Fowler, A. G., and Hollenberg, L. C. L. (2011). Surface code quantum computing with error rates over 1%. *Phys. Rev. A*, 83:020302, arXiv:1009.3686 [quant-ph].
- [Wang et al., 2016] Wang, Y., Kumar, A., Wu, T.-Y., and Weiss, D. S. (2016). Single-qubit gates based on targeted phase shifts in a 3D neutral atom array. *Science*, 352(6293):1562–1565, arXiv:1601.03639 [quant-ph].
- [Weimer et al., 2011] Weimer, H., Müller, M., Büchler, H. P., and Lesanovsky, I. (2011). Digital quantum simulation with Rydberg atoms. *Quantum Inf. Process.*, 10(6):885, arXiv:1104.3081 [quant-ph].
- [Weimer et al., 2010] Weimer, H., Müller, M., Lesanovsky, I., Zoller, P., and Büchler, H. P. (2010). A Rydberg quantum simulator. *Nat. Phys.*, 6(5):382–388, arXiv:0907.1657 [quant-ph].

- [Weiss et al., 2004] Weiss, D. S., Vala, J., Thapliyal, A. V., Myrgren, S., Vazirani, U., and Whaley, K. B. (2004). Another way to approach zero entropy for a finite system of atoms. *Phys. Rev. A*, 70(4):040302.
- [Whiteside and Fowler, 2014] Whiteside, A. C. and Fowler, A. G. (2014). Upper bound for loss in practical topological-cluster-state quantum computing. *Phys. Rev. A*, 90(5):052316, arXiv:1409.4880 [quant-ph].
- [Wood and Gambetta, 2017] Wood, C. J. and Gambetta, J. M. (2017). Quantification and characterization of leakage errors. arXiv:1704.03081 [quant-ph].
- [Wootters and Zurek, 1982] Wootters, W. K. and Zurek, W. H. (1982). A single quantum cannot be cloned. *Nature*, 299(5886):802–803.
- [Xia et al., 2015] Xia, T., Lichtman, M., Maller, K., Carr, A. W., Piotrowicz, M. J., Isenhower, L., and Saffman, M. (2015). Randomized benchmarking of single-qubit gates in a 2D array of neutral-atom qubits. *Phys. Rev. Lett.*, 114(10):100503, arXiv:1501.02041 [quant-ph].