**The Thesis Committee for Amir Pascal Ebrahimi**
**certifies that this is the approved version of the following thesis:**

# CCZ Magic

APPROVED BY

SUPERVISING COMMITTEE:

Scott J. Aaronson, Supervisor

David Soloveichik

Andrew Glaudell

# CCZ Magic

**by**

**Amir Pascal Ebrahimi**

**THESIS**

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

**MASTER OF SCIENCE IN COMPUTER SCIENCE**

THE UNIVERSITY OF TEXAS AT AUSTIN

May 2023

# Acknowledgments

I was introduced, thankfully, to quantum computing in 2018 by a work colleague, Dylan Urquidi. Little did I know where it would lead, causing me to return to university in 2021; 18 years since obtaining a Bachelors of Science in Computer Science. In the time since then I've enjoyed entering a community of individuals who seem to be as enthused as I am about the prospect of building a functional quantum computer. My hope and aim is to see this realized within my own lifetime.

I want to thank Scott Aaronson for giving me the opportunity to take the thesis track in my program. The word "thanks" does not do any justice to the tireless effort, insightful feedback, encouragement, and (most importantly) patience that Andrew Glaudell, Harrison Eggers, and Nicholas Lee-Hone have provided me. I appreciate David Gosset answering my questions over email re: his and other's work. I must give thanks to Sylvio Drouin for supporting me while I was still at Unity to start learning about quantum computing. Along the way, I have made many friends who I would like to thank (in no particular order): Daniel Colomer and the rest of the unsys community, Diego Emilio Serrano, Michał Stęchły and the qosf community, Andrew Glassner, Bambordé Baldé and the Quantum Formalism (QF) community, Ronak Ramachandran, Shantanu Jha, the Qiskit community, Will Zeng and Nathan Shammah from the Unitary Fund community.

# Abstract

## CCZ Magic

Amir Pascal Ebrahimi, M.S.C.S.
The University of Texas at Austin, 2023

Supervisor: Scott J. Aaronson

Universal quantum computation can be achieved with a gate set that includes a generating set of Clifford gates and at least one non-Clifford gate [1]. When it comes to classically simulating quantum computation, Clifford gates generate an important class of circuits known as *stabilizer* circuits, which are efficiently simulable [2]. Stabilizer simulators can be extended to support universal quantum computation by using *magic state* injection gadgets and magic state decomposition. The computational scaling directly relates to the *exact stabilizer rank*, $\chi$, of the magic state, which is the minimum number of stabilizer states in superposition that are necessary to represent the state [3].

We show that decompositions of Clifford magic states of the form $|D\rangle = D\,|+\rangle^{\otimes n}$, where $D$ is a diagonal circuit composed of Z, CZ, and CCZ gates, have $\chi = 2$ for $n \leq 5$. We also establish that all CCZ circuits of $n \leq 5$ qubits are Clifford equivalent to $k \leq 2$

CCZ circuits. These results provide a new lower-bound for the complexity of simulating Clifford+CCZ circuits in general. We suggest the use of a *state-injection gadget* of $|D\rangle$ in contrast to the naïve approach of multiple state-injection gadgets of individual $|\mathtt{CCZ}\rangle$ states. Since extended stabilizer simulators have a gadget-based method of simulation, this approach can improve classical simulation of quantum computation when using a Clifford+CCZ gate set.

# Table of Contents

**Acknowledgments** **4**

**Abstract** **5**

**List of Tables** **10**

**List of Figures** **11**

**Chapter 1. Introduction** **13**
1.1 Outline . . . . . 14
1.2 Universal Quantum Computation . . . . . 15
1.3 Gate Sets . . . . . 16
1.3.1 Quantum gates . . . . . 16
1.3.2 Circuit notation . . . . . 18
1.3.3 Universality among gates . . . . . 20
1.3.4 Approximating arbitrary single qubit gates . . . . . 26
1.3.5 Universal finite gate sets . . . . . 28
1.3.5.1 $\texttt{CNOT} + \texttt{H} + \texttt{T}$ . . . . . 28
1.3.5.2 $\texttt{TOFFOLI} + \texttt{H} + \texttt{S}$ . . . . . 29
1.3.5.3 $\mathbb{S} + \texttt{CCZ}$ . . . . . 30

**Chapter 2. Classical Simulation of Quantum Computation** **32**
2.1 Stabilizer Simulation . . . . . 33
2.1.1 Pauli group . . . . . 34
2.1.2 Clifford group . . . . . 34
2.1.3 Stabilizer groups . . . . . 35
2.1.4 Stabilizer formalism . . . . . 36
2.1.5 Efficient simulation . . . . . 37
2.2 Practical Resource Limits . . . . . 38

**Chapter 3. Extended Stabilizer Simulation** **41**
3.1 Fault Tolerance . . . 41
3.1.1 Transversal $\bar{\mathtt{X}}$, $\bar{\mathtt{Z}}$, $\bar{\mathtt{H}}$, $\bar{\mathtt{S}}$, and $\overline{\mathtt{CNOT}}$ . . . 43
3.1.2 Transversal $\overline{\mathtt{T}}$ . . . 45
3.2 Magic States . . . 47
3.2.1 $|\mathtt{T}\rangle$ state injection gadget . . . 48
3.2.2 $|\mathtt{CCZ}\rangle$ state injection gadget . . . 50
3.2.3 Clifford corrections . . . 50
3.3 Stabilizer Rank . . . 51
3.3.1 Known bounds . . . 51
3.4 Extended stabilizer simulation methods . . . 52
3.4.1 Gadget-based method . . . 52
3.4.2 Sum-over-Cliffords method . . . 53
3.5 Practical Resource Limits . . . 53

**Chapter 4. Results** **55**
4.1 CCZ Magic . . . 56
4.1.1 CCZ constructed from T gates . . . 56
4.1.2 $|\mathtt{D}\rangle$ magic state injection . . . 57
4.2 Stabilizer Decompositions . . . 57
4.2.1 Notation . . . 58
4.2.2 CCZ magic state for three qubits . . . 58
4.2.3 CCZ magic states for four qubits . . . 61
4.2.4 CCZ magic states for five qubits . . . 63
4.2.4.1 Hypergraphs . . . 64
4.2.4.2 Transversal number . . . 66
4.2.5 CCZ magic states for six qubits . . . 67
4.3 Conclusion . . . 68

**Appendices** **70**

**Appendix A. Decompositions** **71**

**Appendix B. Algorithms** **81**

**Bibliography** **83**

# List of Tables

2.1 Floating-point resource scaling . . . . . 39
2.2 Floating-point resource scaling, state + unitary . . . . . 39

3.1 Pauli errors under conjugation by CNOT. . . . . 43

4.1 $|\mathtt{CCZ}\rangle$ decompositions . . . . . 59

A.1 Distinct CCZ magic state decompositions for five qubits . . . . . 79

# List of Figures


1.1 Identity operation on a wire . . . . . 18
1.2 Single gate U on a wire . . . . . 18
1.3 Multiple gates on a wire . . . . . 18
1.4 Multiple qubits initialized to $|0\rangle$ . . . . . 19
1.5 Ordering of qubits . . . . . 19
1.6 `CNOT` gate . . . . . 21
1.7 `SWAP` gate . . . . . 23
1.8 controlled-U gate . . . . . 23
1.9 Singly-controlled-U gate decomposition . . . . . 24
1.10 Doubly-controlled-U gate decomposition . . . . . 24
1.11 Multi-controlled-U gate decomposition . . . . . 25
1.12 `CCZ` gate . . . . . 30

3.1 `CNOT` error propagation . . . . . 42
3.2 $\bar{\mathtt{X}}$ and $\bar{\mathtt{Z}}$ transversal gates . . . . . 44
3.3 $\bar{\mathtt{H}}$ and $\bar{\mathtt{S}}$ transversal gates . . . . . 44
3.4 $\overline{\mathtt{CNOT}}$ transversal gate . . . . . 45
3.5 `T` gate via state teleportation . . . . . 45
3.6 $\overline{\mathtt{T}}$ transversal gate . . . . . 46
3.7 `T` gate via gate teleportation . . . . . 48
3.8 $\overline{\mathtt{T}}$ transversal gate with gate teleportation . . . . . 49
3.9 `CCZ` gate via gate teleportation . . . . . 50
3.10 Weeks vs. `T` gates . . . . . 54

4.1 `T` gate construction of `CCZ` . . . . . 56
4.2 $\mathtt{CCZ}\,|+\rangle^{\otimes 3} = |\mathtt{CCZ}\rangle$ . . . . . 59
4.3 Five-qubit, maximally filled `CCZ` circuit identity . . . . . 63
4.4 Hypergraph, $H_1$, for maximally filled `CCZ` circuit . . . . . 65

4.5 Hypergraph, $H_2$, for maximally filled CCZ circuit after reduction . . . 65

A.1 Distinct circuits for five qubit CCZ magic states . . . . . 71

# Chapter 1

# Introduction

Computation has been a pivotal utility to the advancement of humanity even before manifesting in the form of physical devices. The decades-long pursuit of realizing a useful, fault-tolerant quantum computer has been a fascinating story so far of human ambition and tenacity; rife with distant goals, no-go theorems, incredible resource requirements, fundamental physical challenges, pundits and naysayers, bold claims, breakthroughs and setbacks. Differentiating itself from *classical* computing, which bears its name from its connection to classical physics, quantum computing employs a computational model that relies on and draws its power from the theory of quantum mechanics. Specifically, the power of quantum computing lies in the potential of being able to solve some of the problems that would otherwise be considered intractable to classical computing, such as integer factorization, discrete logarithms, and quantum simulation. Born out of the fields of physics, computer science, and information theory, quantum information science offers awesome challenges to those who pursue its unforgiving and unapologetic conditions, such as dealing with environmental noise or designing quantum algorithms that offer exponential speedups over classical algorithms.

Many types of quantum computation exist, including adiabatic quantum

computing (AQC) [4], quantum annealing [5], measurement-based quantum computing (MBQC) [6], boson-sampling [7], continuous-variable quantum computing (CVQC) [8], gate model quantum computing [9], and topological quantum computing [10] to name a few. Not all approaches are equivalent, but some of these can be shown to be equivalent to a *universal* quantum computer or quantum Turing machine [11], which can express the full power of quantum computation. One universal and prevalent approach is the gate model – usually depicted in the form of *quantum circuits* – that will be the type of quantum computation that we will use throughout the remainder of this manuscript.

## 1.1 Outline

In what follows, we will explore what universal quantum computation is via the introduction of gate sets, the classical simulation thereof, their resource requirements, and the sometimes necessary augmentations in order to reach universality. Our results will show how to improve upon the already exponential-scaling resource requirements for classical simulation of quantum computation for a specific class of circuits. Some familiarity with quantum computing is assumed as we will not cover the basic foundations of quantum information science. Instead we refer the reader to other resources [9, 12, 13]. In the remainder of this chapter we will define what quantum gates are, introduce the important Solovay-Kitaev theorem, and look at a few important gate families. In the next chapter we will move on to the classical simulation of quantum computation and the inherent challenges that come; introduce yet another important theorem, the Gottesman-Knill theorem, for stabilizer

simulation; and explore in more detail the classical resource requirements needed for simulation. In the following chapter, we will come back to non-universal gate sets and their augmentation, and introduce the reader to the extended stabilizer simulation framework, which utilizes a concept known as *stabilizer rank*. Finally, we will move on to our research results and conclude with a brief summary and encouragement towards future work.

## 1.2 Universal Quantum Computation

Before we move into what gate sets are, let us linger for a moment on what universal computation is, both in the classical and quantum realm. *Universality* is a property that, in the context of computation, means that whatever type of computation can be performed on one physical device or machine can be performed by another that is universally equivalent. Over the course of the 1900s, the concept of universal classical computation has been revised into what is known as the Extended Church-Turing thesis, which emerged from the field of computational complexity created as a result of Alonzo Church and Alan Turing's original thesis. The thesis states:

**Proposition 1.2.1** (Extended Church-Turing Thesis [12])**.** *A probabilistic Turing machine can efficiently simulate any realistic model of computation.*

Efficiently is used here to mean that one model or machine can be reduced to another within a polynomial time factor. Realistic means that it can be physically realized.

David Deutsch in 1985, with some prior credit due to Paul Benioff, challenged

the Extended Church-Turing Thesis by showing that a quantum mechanical machine could solve problems, which no probabilistic Turing machine could solve [9]. So, we now have the Quantum Extended Church-Turing (QECT) thesis:

**Proposition 1.2.2** (Quantum Extended Church-Turing Thesis [12])**.** *A quantum Turing machine can efficiently simulate any realistic model of computation.*

It is unknown whether a more powerful, physically realizable computational model could displace even the QECT thesis.

## 1.3 Gate Sets

### 1.3.1 Quantum gates

The gate model of quantum computation has three stages – initialization, evolution, and measurement. First, we initialize a quantum state, which can be represented by a complex-valued vector. During the evolution stage, we perform computation by applying *unitary quantum gates* to states. Finally, we can observe the results of our computation by performing measurement of the state. For our work we focus on the evolution of quantum states via gates.

Unitary quantum gates can be represented naturally in the language of linear algebra. Using *Dirac* notation we can write an arbitrary quantum computation as:

$$|\psi'\rangle = G\,|\psi\rangle \tag{1.1}$$

where $|\psi\rangle \in \mathbb{C}^n$ is Dirac notation for an arbitrary quantum state in an $n$-dimensional complex vector space, $G$ is a quantum gate, and $|\psi'\rangle$ is the posterior state following

the action of the quantum gate $G$ on $|\psi\rangle$. For the remainder of this manuscript it can be assumed that any vector spaces we are referring to are complex-valued.

One property of *pure* quantum states from quantum mechanics is that the state has a normalization condition:

$$\| \, |\psi\rangle \, \|_2 = 1 = \sqrt{\langle\psi|\psi\rangle} \tag{1.2}$$

where $\| \cdot \|_2$ is the 2-norm and $\langle\cdot|\cdot\rangle$ is an inner product for the vector space. Moving forward we will use $\| \cdot \|$ to denote the 2-norm. An example of this normalization condition for a state vector in a 2-dimensional vector space, choosing the *computational basis* of $|0\rangle$ and $|1\rangle$, is:

$$\| \, |\psi\rangle \, \| = \| \, \alpha \, |0\rangle + \beta \, |1\rangle \, \| = |\alpha|^2 + |\beta|^2 = 1 \tag{1.3}$$

where the state vectors $|0\rangle$ and $|1\rangle$ are:

$$|0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \qquad |1\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \tag{1.4}$$

We will sometimes refer to this 2-dimensional state vector as representing a *qubit*, a fundamental unit of quantum information. For $n$ qubits we would have a state vector in a $2^n$-dimensional vector space. We consider quantum computation within a *closed* system where the normalization condition of quantum states is preserved. Therefore, we require our quantum gates, denoted $G$ here, to also preserve this normalization property:

$$\| \, G \, |\psi\rangle \, \| = 1 \tag{1.5}$$

In the representation of matrices, this property is fulfilled by *unitary* matrices, which can most succinctly be specified by the following constraint:

$$U^\dagger U = I \tag{1.6}$$

where the † operation is the Hermitian adjoint or conjugate transpose of a matrix.

### 1.3.2 Circuit notation

We now introduce a circuit notation for quantum gates acting on qubits via the abstraction of a *wire* [9]. The simplest gate is that of no gate, or the identity operation, on a wire:


Figure 1.1: Identity operation on a wire

For a single gate, $U$, we have:

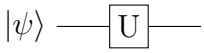



Figure 1.2: Single gate U on a wire

For multiple gates acting on a qubit, such as $|\psi'\rangle = AB\,|\psi\rangle$ we have:

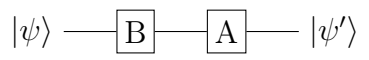



Figure 1.3: Multiple gates on a wire

Please note the order of the gates in Dirac notation vs. the reverse order in circuit notation.

We can also represent a state of many qubits using many wires. Often, we will start with these qubits in a basis state, which is usually the computational basis state $|0\rangle$:

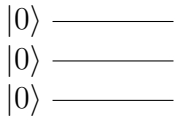



Figure 1.4: Multiple qubits initialized to $|0\rangle$

There are several ways to write multiple qubit states in Dirac notation. For example, the state above can be written as $|000\rangle$ or $|0\rangle^{\otimes 3}$. Generally, multiple qubit states are denoted in the following ways:

$$|xy\ldots z\rangle\,, \quad |x, y, \ldots, z\rangle\,, \quad |x\rangle \otimes |y\rangle \otimes \ldots \otimes |z\rangle$$

which would have the following order when mapped to a circuit diagram:

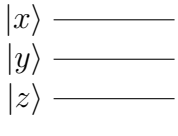



Figure 1.5: Ordering of qubits

If they are all the same state, we can also write the combined state as a tensor power: $|x\rangle^{\otimes n}$.

We will expand upon this circuit notation throughout the remainder of the manuscript as we introduce new types of gates.

### 1.3.3 Universality among gates

The concept of universality also exists within the context of gates. Within classical computation it is known that the `NAND` gate is a universal gate, which means that any other gate can be produced (or generated) by multiple copies of this gate. In quantum computation universality also exists, but requires a bit more work to reach. It aids us to separate gates into two categories: ones that operate on 2-dimensional state vectors, or single qubits, and ones that operate on $2^n$-dimensional state vectors, or multiple qubits, where $n \geq 2$. An arbitrary gate operating on a 2-dimensional state vector can be represented in the form of a $2 \times 2$ unitary matrix, which can be decomposed [9] as:

$$U = e^{ia} \begin{bmatrix} e^{-ib/2} & 0 \\ 0 & e^{ib/2} \end{bmatrix} \begin{bmatrix} \cos\frac{c}{2} & -\sin\frac{c}{2} \\ \sin\frac{c}{2} & \cos\frac{c}{2} \end{bmatrix} \begin{bmatrix} e^{-id/2} & 0 \\ 0 & e^{id/2} \end{bmatrix} \tag{1.7}$$

where $a, b, c, d \in \mathbb{R}$. There are an infinite number of gates of this type, but we will show later that each can be approximated to arbitrary accuracy using a finite set of gates.

For the gate type that operates on $2^n$-dimensional state vectors it was shown that any unitary matrix that operates on a $d$-dimensional vector space can be decomposed into a product of *two-level* unitary matrices [9]. A two-level unitary matrix of dimension $d \times d$ acts non-trivially only on $\leq 2$ elements of a vector. An example of a two-level unitary matrix that operates on $2^2$-dimensional state vectors is the controlled-`NOT` or `CNOT` gate. First, let's establish what the `NOT` gate is by looking at its unitary matrix:

$$U_{\texttt{NOT}} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \tag{1.8}$$

The NOT gate is sometimes referred to as X and also exists as a classical gate. The action of X on the computational basis states is:

$$\begin{aligned} \texttt{X}\,|0\rangle &\to |1\rangle \\ \texttt{X}\,|1\rangle &\to |0\rangle \end{aligned} \tag{1.9}$$

Now we can present the unitary matrix for CNOT where we see the NOT gate embedded in the lower $2 \times 2$ submatrix:

$$U_{\texttt{CNOT}} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{1.10}$$

As a quantum circuit, the CNOT gate is depicted as:

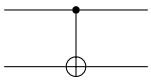


Figure 1.6: CNOT gate

We will refer to this gate interchangeably as CNOT or CX. The action of CNOT on a state $|x, y\rangle$ is:

$$\texttt{CNOT}\,|x, y\rangle \to |x, x \oplus y\rangle \tag{1.11}$$

where $\oplus$ is modulo-2 addition and $x, y \in \{0, 1\}$. As can be seen, this gate only effects a change on $|y\rangle$, which is in $\mathbb{C}^2$. We refer to $|x\rangle$ as the control and $|y\rangle$ as the target. It is also possible to swap the control and targets using SWAP gates, which are covered below. There are infinitely many other two-level unitary matrices of varying dimension, but this example is sufficient for our purposes.

With two-level unitary gates we can present our first level of universality:

**Theorem 1.3.1.** *Two-level unitary gates are universal.*

*Proof.* The proof for this would require multiple pages, which we will skip for brevity, but the main result is that we can express an arbitrary gate that operates on a $d$-dimensional vector space as:

$$U = V_1 \dots V_k \tag{1.12}$$

where each $V_i$ is a two-level unitary matrix and $k \leq (d-1) + (d-2) + \cdots + 1 = d(d-1)/2$. At each stage we are finding two-level unitary matrices that fix the first row and column with a one and zeros elsewhere and then proceed with each submatrix of reducing dimension.

For a full proof, see §4.5.1 in [9]. □

We need a few more pieces before we can make a stronger statement about the universality of quantum gates. One piece is that we need to be able swap qubits in our system. We call this the `SWAP` gate and represent it with the following unitary matrix:

$$U_{\texttt{SWAP}} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{1.13}$$

Fortunately, `SWAP` can be created with three `CNOT` gates:

$$U_{\texttt{SWAP}} = U_{\texttt{CX}_{01}} U_{\texttt{CX}_{10}} U_{\texttt{CX}_{01}} \tag{1.14}$$

where $U_{\texttt{CX}_{xy}}$ is the `CNOT` gate with control as $x$ and target as $y$.

As a quantum circuit this can be depicted as:

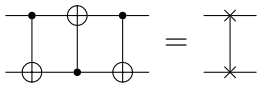


Figure 1.7: `SWAP` gate

Another operation that we need to be able to perform is a controlled bit flip conditioned on other qubits in our system. This will be possible with a multi-controlled `NOT` gate. So, let's define what an arbitrary controlled-U gate is first:

**Definition 1.3.1** (controlled-U gate for $n + k$ qubits)**.**

$$C^n(U)\,|x_1x_2\ldots x_n\rangle\,|\psi\rangle = |x_1x_2\ldots x_n\rangle\,U^{x_1x_2\ldots x_n}\,|\psi\rangle \tag{1.15}$$

where $x = x_1x_2\ldots x_n \in \{0,1\}^n$ is a bitstring and U is only performed on the $k$ qubits of $|\psi\rangle$ if the product of the bits $x_1, x_2, \ldots, x_n$ is 1.

As a quantum circuit, this can be depicted as:

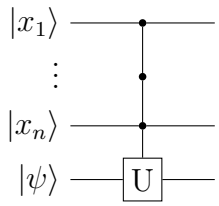



Figure 1.8: controlled-U gate

From Nielsen and Chuang §4.3 [9], we have that a singly-controlled gate can be performed by:

$$C^1(U) = (U_{e^{ia}} \otimes I)(I \otimes A)U_{\texttt{CX}_{01}}(I \otimes B)U_{\texttt{CX}_{01}}(I \otimes C) \tag{1.16}$$

where

$$U = e^{ia} A(\mathtt{X}) B(\mathtt{X}) C \tag{1.17}$$

$$U_{e^{ia}} = \begin{bmatrix} 1 & 0 \\ 0 & e^{ia} \end{bmatrix} \tag{1.18}$$

and $a \in \mathbb{R}$, $ABC = I$, and $\mathtt{X}$ is the $\mathtt{NOT}$ gate previously defined.

Depicted as a quantum circuit, this is:

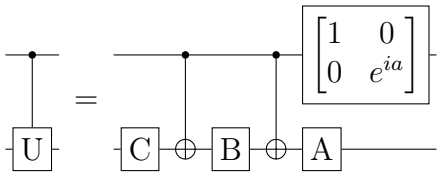



Figure 1.9: Singly-controlled-U gate decomposition

Next, for a doubly-controlled gate we can construct it by:

$$\begin{aligned} C^2(U) = (I \otimes U_{\mathtt{SWAP}})(C^1(V) \otimes I)(I \otimes U_{\mathtt{SWAP}}) \\ (U_{\mathtt{CX}_{01}} \otimes I)(I \otimes C^1(V^\dagger)) \\ (U_{\mathtt{CX}_{01}} \otimes I)(I \otimes C^1(V)) \end{aligned} \tag{1.19}$$

where $V$ is any unitary satisfying $V^2 = U$.

Depicted as a quantum circuit, this is:

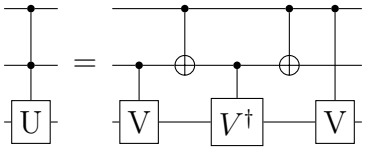



Figure 1.10: Doubly-controlled-U gate decomposition

Now, for a multi-controlled gate, $C^n(U)$, we can construct this by utilizing multiple $C^2(X)$ gates, also known as doubly-controlled `NOT` gates known as `TOFFOLI` or `CCX` gates, each targeting one of $n-1$ auxiliary qubits and a final $C^1(U)$ gate from the $(n-1)$th auxiliary qubit to the target qubit. Constructions also exist that make use of no auxiliary qubits, but this is sufficient for our purposes.

Depicted as a quantum circuit, a multi-controlled gate would be:

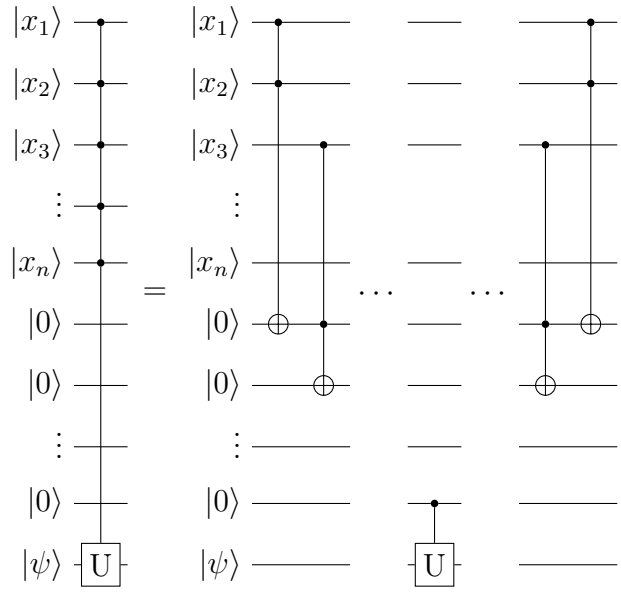



Figure 1.11: Multi-controlled-U gate decomposition

Now, with all of these pieces in place we can state the following theorem:

**Theorem 1.3.2.** *Single qubit and `CNOT` gates are universal.*

*Proof.* While a full proof would take many pages, we can consider some intuition behind why this is true. Above, we have Theorem 1.3.1 as well as a multi-controlled

single qubit gate. Since each two-level unitary gate operates only on $\leq 2$ vector components we can consider this a single qubit gate. For each two-level unitary gate, it operates on two computational basis states, which are separated by a number of bits known as their Hamming distance. We can use a *Gray code* to perform a series of controlled bit flips until we have the two states in a form that differ by only one qubit, perform a multi-controlled single qubit gate targeting the qubit that differs, and then perform the reverse series of controlled bit flips until we are back to the original terms in our state. When we do this for each of the $k$ two-level unitary gates, then the final result will be an implementation of an arbitrary $d$-dimensional gate.

For a full proof, see §4.5.2 in [9]. □

### 1.3.4 Approximating arbitrary single qubit gates

With Theorem 1.3.2 we are still in the realm of an infinite number of single qubit gates. What we desire is to be able to approximate any single qubit gate to arbitrary accuracy using a *finite* set of single qubit gates, which is necessary when considering fault-tolerant quantum computation and physically realizable gates. From Nielsen and Chuang, it is possible to approximate any single qubit gate to within any error $\epsilon > 0$ with only two specific gates, `H` and `T` that we will define later. The pedagogical approach presented by Nielsen and Chuang scales in the number of gates as $\Theta(1/\epsilon)$ [9]. Remarkably, the Solovay-Kitaev theorem [14] brings this down to being polylogarithmic in $m$[1].

[1]There exists a slightly better bound, which was discovered in 2016 by Ross and Selinger [15]

**Definition 1.3.2** ($\mathcal{SU}(2)$)**.** The special unitary group of degree 2 where $\mathcal{SU}(2) \subset \mathcal{U}(2)$. This group is the set of $2 \times 2$ unitary matrices with determinant 1.

**Definition 1.3.3** ($\epsilon$-net)**.** A subset $A \subset X$ is an $\epsilon$-net if $d_A(x) < \epsilon \ \forall \ x \in X$ where $d_A(x)$ is some distance measure.

**Theorem 1.3.3** (Solovay-Kitaev theorem)**.** *Let $\mathcal{G}$ be a finite set of elements in $SU(2)$ containing its own inverses, such that $\langle \mathcal{G} \rangle$ is dense in $SU(2)$. Let $\epsilon > 0$ be given. Then $\mathcal{G}_l$ is an $\epsilon$-net in $\mathcal{SU}(2)$ for $l = O(\log^c(1/\epsilon))$, where $c \approx 4$.*

*Proof.* For a full proof, see Appendix 3 of [9]. □

The crux of this theorem is to establish how many gates, $l$, from $\mathcal{G}$ must be combined to get $\epsilon$-close to an arbitrary single qubit unitary matrix. A necessary condition is that $\langle \mathcal{G} \rangle$ is dense in $SU(2)$, which means that it already has an arbitrarily good approximation for any single qubit gate. This approximation could be extremely long in $l$, so the question is whether the number of gates required for a given precision is practical. The Solovay-Kitaev theorem shows that we get $\epsilon$-close rather quickly as $l$ increases.

**Corollary 1.3.4.** *A finite set of single qubit gates that can approximate any arbitrary single qubit gate and the `CNOT` gate is universal.*

*Proof.* This follows naturally as a result of Theorem 1.3.2 and Theorem 1.3.3. □

### 1.3.5 Universal finite gate sets

We can now state a few families of finite gate sets that are universal. Note that each gate set listed here is closed under inverses.

#### 1.3.5.1 CNOT + H + T

The first family is the one that was mentioned above:

$$\mathbb{U}_1 = \{\texttt{CNOT}, \texttt{H}, \texttt{T}\} \tag{1.20}$$

Since we have not defined the H and T gates, let's define those now. First, the HADAMARD or H gate's unitary matrix is:

$$U_{\texttt{H}} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \tag{1.21}$$

where the action of H on the computational basis states is:

$$\begin{aligned} \texttt{H}\,|0\rangle &\to |+\rangle \\ \texttt{H}\,|1\rangle &\to |-\rangle \end{aligned} \tag{1.22}$$

where $\{|+\rangle, |-\rangle\}$ is considered the Hadamard basis. The state vectors for $|+\rangle$ and $|-\rangle$ are:

$$|+\rangle = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 1 \end{bmatrix} \qquad |-\rangle = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ -1 \end{bmatrix} \tag{1.23}$$

Next, let's look at the T gate's unitary matrix:

$$U_{\texttt{T}} = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\pi/4} \end{bmatrix} \tag{1.24}$$

where the action of T on the computational basis states is:

$$\begin{aligned} \texttt{T}\,|0\rangle &\to |0\rangle \\ \texttt{T}\,|1\rangle &\to e^{i\pi/4}\,|1\rangle \end{aligned} \tag{1.25}$$

Although not included in this gate set, one gate that is generated by the set is the S or PHASE gate. The unitary matrix for S is:

$$U_{\mathtt{S}} = \begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix} = U_{\mathtt{T}}^2 \tag{1.26}$$

where the action of S on the computational basis states is:

$$\begin{aligned} \mathtt{S}\,|0\rangle &\rightarrow |0\rangle \\ \mathtt{S}\,|1\rangle &\rightarrow |i\rangle \end{aligned} \tag{1.27}$$

Combined with CNOT and H we have the following set:

$$\mathbb{S} = \{\mathtt{CNOT}, \mathtt{H}, \mathtt{S}\} \tag{1.28}$$

which generates an important group known as the *Clifford group* (covered in Section 2.1.2). The Clifford group is important to the fault-tolerant construction of approximated single qubit gates, which will be covered in Chapter 3.

#### 1.3.5.2 TOFFOLI + H + S

The next gate set is useful as a stepping stone towards our final gate set.

$$\mathbb{U}_2 = \{\mathtt{TOFFOLI}, \mathtt{H}, \mathtt{S}\} \tag{1.29}$$

Here we can see that we no longer require the two qubit CNOT gate or the single qubit T gate since we are including the three qubit TOFFOLI gate. This gate set was proven to be universal in 2002 by Yaoyun Shi [16].

We briefly mentioned the TOFFOLI or CCX gate in an earlier section, but for completeness we will show its action:

$$\mathtt{CCX}\,|x, y, z\rangle \rightarrow |x, y, (x \wedge y) \oplus z\rangle \tag{1.30}$$

The unitary matrix for CCX is:

$$U_{\texttt{CCX}} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \tag{1.31}$$

#### 1.3.5.3 $\mathbb{S} + \texttt{CCZ}$

We present a final family since it is important to our research:

$$\mathbb{U}_3 = \mathbb{S} + \{\texttt{CCZ}\} = \{\texttt{CNOT}, \texttt{H}, \texttt{S}, \texttt{CCZ}\} \tag{1.32}$$

In the literature, this family is more commonly known as Clifford+CCZ. Note that this gate set is equivalent to $\mathbb{U}_2$. The X gate can be generated by HSSH. The CNOT gate can be generated with X and TOFFOLI. Finally, the CCZ gate is equivalent to a TOFFOLI conjugated by H gates, which are in $\mathbb{U}_2$:

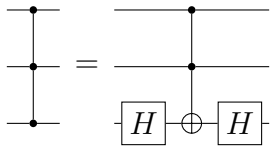



Figure 1.12: CCZ gate

The CCZ gate has the following action on a state:

$$\texttt{CCZ}\,|x, y, z\rangle \rightarrow (-1)^{x \cdot y \cdot z}\,|x, y, z\rangle \tag{1.33}$$

and is represented by the unitary matrix:

$$U_{\mathsf{ccz}} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \end{bmatrix} \tag{1.34}$$

# Chapter 2

# Classical Simulation of Quantum Computation

In the previous chapter we established that universal quantum computation can be performed via a finite set of gates; primarily due to the need for physical realization and fault tolerance. *Fault-tolerant quantum computation* is a paradigm where arbitrarily good quantum computation can be achieved even in the presence of noise and errors caused by faulty operations. By suppressing said errors below a certain threshold, reliable quantum computing is possible [17–20]. Fault tolerant constructions exist for all of the gates in the universal families that were previously defined. As we will see, finite gate sets are also important to the classical regime for efficiently simulating a subset of quantum circuits.

Classical simulation of quantum computation is important for many reasons including education, research, validation of bounds, and verification of results. Naïvely, a quantum computation can be performed via classical simulation by multiplying a vector of size $2^n$, for a pure state, by a unitary matrix of size $2^n \times 2^n$ for a system of $n$ qubits. Due to exponential scaling in $n$ this is *inefficient* for classical simulation and limits what size systems we are able to simulate. Generally, it becomes classically intractable to simulate systems on the order of $n \sim 60$ qubits [21], because the memory requirements effectively double with every qubit. Alternatively,

if we wish to simulate a mixed state, which is a classical ensemble of pure states, then a state is represented by a matrix of $2^n \times 2^n$, which limits us even further to roughly $n \sim 30$ qubits. Naturally, one might think to increase these bounds by partitioning a quantum system into smaller subsystems, but we would still require $2^n$ (or $4^n$ respectively) complex numbers to write the full state vector (or density matrix respectively) down in the result.

The following quote from Nielsen and Chuang [9] sums up the advantage of quantum computation and the challenge of classical simulation:

> Quantum computers can efficiently simulate quantum systems for which there is no known efficient classical simulation. Intuitively, this is possible for much the same reason any quantum circuit can be constructed from a universal set of gates.

## 2.1 Stabilizer Simulation

Many approaches exist beyond the naïve approach for classically simulating quantum computation including sum-over-paths [22], tensor networks [22], and stabilizer simulation [23]. Our focus in this work will be on stabilizer simulation, which restricts our computation to a subset of quantum circuits, but as a trade-off allows for efficient simulation (e.g. polynomial time in $n$ for $n$ qubits). First, we need to establish a few prerequisites.

### 2.1.1 Pauli group

In the language of group theory we define a group $\mathcal{P}$ known as the Pauli group, a subgroup of the unitary group $\mathcal{U}$, whose elements reflect a set of quantum gates:

**Definition 2.1.1** (Pauli group)**.**

$$\mathcal{P} = \langle \mathtt{X}, \mathtt{Y}, \mathtt{Z} \rangle = \{\pm I, \pm iI, \pm X, \pm iX, \pm Y, \pm iY, \pm Z, \pm iZ\} \subset \mathcal{U} \tag{2.1}$$

where $\mathtt{X}$, $\mathtt{Y}$, $\mathtt{Z}$ have the following matrix representations:

$$U_{\mathtt{X}} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \quad U_{\mathtt{Y}} = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix} \quad U_{\mathtt{Z}} = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \tag{2.2}$$

These groups extend to the $n$-qubit Pauli group, $\mathcal{P}(n)$, by taking an $n$-fold tensor product of all permutations of Pauli gates:

$$\mathcal{P}(n) = \{\otimes_{i=1}^{n} P_i \mid P_i \in \mathcal{P}\} \subset \mathcal{U}(n) \tag{2.3}$$

### 2.1.2 Clifford group

Next, we define a group $\mathcal{C}(n)$ known as the $n$-qubit Clifford group, a subgroup of $\mathcal{U}(n)$, which is generated by the set $\mathbb{S} = \{\mathtt{CNOT}, \mathtt{H}, \mathtt{S}\}$ previously defined in Equation 1.28.

**Definition 2.1.2** (n-qubit Clifford group)**.**

$$\begin{aligned} \mathcal{C}(n) &= \langle \mathbb{S} \rangle \\ &= \langle \mathtt{CNOT}, \mathtt{H}, \mathtt{S} \rangle \\ &= \{U \in \mathcal{U}(n) \mid UPU^\dagger \in \mathcal{P}(n) \ \forall \ P \in \mathcal{P}(n)\} \subset \mathcal{U}(n) \end{aligned} \tag{2.4}$$

So, the Clifford group is the normalizer of the Pauli group by mapping a Pauli element to another Pauli element within the same group. Elements of the Clifford group are known as Clifford gates and the subsequent circuits composed solely of Clifford gates are known as Clifford circuits. Alternatively, these can also be referred to as *stabilizer gates* and *stabilizer circuits*, respectively.

### 2.1.3 Stabilizer groups

Now, we consider a group $Stab_\psi(n)$ known as the $n$-qubit stabilizer group of $|\psi\rangle$, which is the set of unitary matrices that fix a state $|\psi\rangle$.

**Definition 2.1.3** (n-qubit Stabilizer group of $|\psi\rangle$)**.**

$$Stab_\psi(n) = \{U \in \mathcal{U}(n) \mid U|\psi\rangle = |\psi\rangle\} \subset \mathcal{U}(n) \tag{2.5}$$

Note that if $U \in Stab_\psi(n)$, then $U^{-1}$ must be as well since:

$$\begin{aligned} U|\psi\rangle &= |\psi\rangle \\ U^{-1}U|\psi\rangle &= U^{-1}|\psi\rangle \\ |\psi\rangle &= U^{-1}|\psi\rangle \end{aligned} \tag{2.6}$$

Also, if $U, V \in Stab_\psi(n)$, then so is $UV$:

$$UV|\psi\rangle = U(V|\psi\rangle) = U|\psi\rangle = |\psi\rangle \tag{2.7}$$

which establishes $Stab_\psi(n)$ as a group.

Unfortunately, $Stab_\psi(n)$ is not that remarkable because it takes exponentially many bits to specify the generators of the group [23]. However, if we restrict $Stab_\psi(n)$

to stabilizers within the Pauli group we only require a linear number of bits to specify the generators [23]. This interesting subgroup $\mathcal{S}_\psi(n)$ is known as the $n$-qubit stabilizing Pauli group of $|\psi\rangle$:

**Definition 2.1.4** (n-qubit stabilizing Pauli group of $|\psi\rangle$)**.**

$$\mathcal{S}_\psi(n) = Stab_\psi(n) \cap \mathcal{P}(n) \subset Stab_\psi(n) \tag{2.8}$$

Any stabilizing Pauli group is Abelian, since the Pauli group either commutes or anti-commutes. If any elements anti-commute, then a multiplicative factor of $-1$ would be introduced and the state would no longer be stabilized. Therefore, all elements must commute.

The states that can be stabilized by a Pauli subgroup are known as *stabilizer states*, which in addition to being efficient to compute are of primary import to quantum error correction as well. Furthermore, every stabilizer state is reachable by applying some Clifford circuit to the all zero state, $|0\rangle^{\otimes n}$.

### 2.1.4 Stabilizer formalism

We now introduce the stabilizer formalism, which simply put is the representation of a quantum state by specifying a generating set of the stabilizing Pauli group for that state rather than specifying its $2^n$ complex components. For stabilizer states, which are our primary interest, a state can be specified by $n$ stabilizing Pauli generators [2], which is an exponential improvement over $2^n$ complex components.

For example, a set of stabilizing generators for the state $|{+}{+}\rangle$ are:

$$\begin{aligned} \mathtt{IX} \rightarrow (\mathtt{I} \otimes \mathtt{X})\,|{+}{+}\rangle = \mathtt{I}\,|{+}\rangle \otimes \mathtt{X}\,|{+}\rangle = |{+}{+}\rangle \\ \mathtt{XI} \rightarrow (\mathtt{X} \otimes \mathtt{I})\,|{+}{+}\rangle = \mathtt{X}\,|{+}\rangle \otimes \mathtt{I}\,|{+}\rangle = |{+}{+}\rangle \end{aligned} \tag{2.9}$$

As was mentioned above, there is also a relation between the stabilizer states and the stabilizer circuits that generate them, so it is possible to specify Clifford gates in the stabilizer formalism by tracking where Clifford gates map elements of the Pauli group under conjugation.

### 2.1.5 Efficient simulation

With all the aforementioned we can now state the Gottesman-Knill theorem [2]:

**Theorem 2.1.1** (Gottesman-Knill theorem)**.** *Quantum computation of Clifford circuits using state preparation and measurement in the computational basis can be simulated efficiently on a classical computer.*

*Proof.* By keeping track of the $n$ stabilizer generators we can update them as Clifford gates are applied in $O(n)$ time for some $k$th qubit(s). Measurements in the computational basis require $O(poly(n))$ for checking commutation/anti-commutation with each of the stabilizer generators for a $\mathtt{Z}_k$ observable on the $k$th qubit. Along with state preparation, this means that we have an efficient simulation that only requires a polynomial number of steps classically per operation, which means that $m$ operations can be simulated in $O(poly(n) \cdot m)$ time.

For a full proof, see §10.5 in [9]. □

We call this form of simulation *stabilizer simulation* and note that since its inception there have been two major improvements:

- $O(n^2)$ time per deterministic measurement at a factor of two increase in the size of specifying an additional $n$ "destabilizer" generators that correspond to the stabilizer generators [23], stored in a binary matrix form known as a *tableau*.
- $O(n)$ time per deterministic measurement by tracking the *inverse* of the tableau [24], which brings overall simulation to $O(n \cdot m)$ time for $m$ operations.

## 2.2 Practical Resource Limits

We turn back now to looking more in-depth at practical resource limits for simulating quantum computation on current classical computing architectures. While 512-bit extensions exist for some instruction set architectures, practically we are still operating in a classical regime of 64-bit processing and memory addressing. As such, the maximum amount of addressable memory is $2^{64}$ bits or 16 exabytes[1]. For a single qubit we require four real numbers, two for each complex number in the vector representation, that can be approximated by either quad-precision (128-bit or $2^7$), double-precision (64-bit or $2^6$), single-precision (32-bit or $2^5$), or on some architectures half-precision (16-bit or $2^4$) floating-point format. If we were using all of the available memory address space to store an arbitrary multiple qubit state in a classical computer this would allow for a maximum size defined in Table 2.1 below.

[1]Using base 2 convention; Using *exbibyte* is a prescribed convention that is unpopular.

| format | pure (qubits) | mixed (qubits) |
|---|---|---|
| quad | 56 | 27 |
| double | 57 | 28 |
| single | 58 | 29 |
| half | 59 | 29 |

Table 2.1: Floating-point resource scaling[2] where we use the formula $2^n \cdot 2 \cdot [\text{format (bits)}] = 2^{64}$ for pure states and $2^{2n} \cdot 2 \cdot [\text{format (bits)}] = 2^{64}$ for mixed states, and solve for $n$.

Generally, for practical performance reasons we will want to be able to store a state and a unitary gate in memory at a minimum, so this further reduces our scaling:

| format | pure (qubits) | mixed (qubits) |
|---|---|---|
| quad | 28 | 27 |
| double | 29 | 28 |
| single | 29 | 28 |
| half | 30 | 29 |

Table 2.2: Floating-point resource scaling, state + unitary where we use the formula $2^n \cdot 2 \cdot [\text{format (bits)}] + 2^{2n} \cdot 2 \cdot [\text{format (bits)}] = 2^{64}$ for pure states + unitary and $2(2^{2n} \cdot 2 \cdot [\text{format (bits)}]) = 2^{64}$ for mixed states + unitary, and solve for $n$.

For comparison, the tableau representation in stabilizer simulation requires $2n(2n+1)$ bits: $n$ stabilizers and $n$ destabilizers with $2n$ bits for specifying each and 1 bit for the sign. Solving for $n$, we could hypothetically store a pure stabilizer state of $n \approx 2.147 \times 10^9$ qubits; a significant improvement. For stabilizer mixed states

[2]The above assumes a naïve matrix/vector format

there is linear overhead [23], which would reduce our number of qubits negligibly. For stabilizer simulation, what this means is that we are more practically limited by other constraints of classical computing architecture, such as actual physical memory size, cache size and type, instructions per cycle, multi-processing capabilities, and vectorized instructions. Many of these constraints have been addressed by Gidney in the currently best-performing stabilizer simulator – Stim [24].

# Chapter 3

# Extended Stabilizer Simulation

In the previous chapter we covered a subset of circuits, known as Clifford or stabilizer circuits, that can be classically simulated efficiently. Now we return to universality with the added requirement of fault tolerance. The Clifford gates lend themselves to fault-tolerant constructions in the realm of quantum error correction [25]. All we need then is to include a fault-tolerant construction of either `T` or `TOFFOLI`, both of which have been shown to exist [20, 26–29]. Most of these constructions are based on a teleportation protocol, which utilizes measurements in the computational basis along with existing fault-tolerant gates. We will take a look at these constructions and how they evolved to use a resource known as a *magic state*. Finally, we will return to classical simulation by relating the use of these magic states to an extended form of stabilizer simulation.

## 3.1 Fault Tolerance

Our purpose for mentioning fault tolerance in this manuscript is not with the intent to cover the topic extensively by any means, but rather to provide context for the relevance and importance of magic states. For a comprehensive introduction to fault tolerance we refer the reader to other resources [9, 25]. First, it is good to

establish what we mean by an operation being fault-tolerant within the context of quantum error correction:

**Definition 3.1.1** (Fault-tolerant quantum operation)**.** A quantum operation that introduces at most one Pauli error in each encoded block[1] of qubits is considered fault-tolerant [9].

As an example, we can consider how an error in a `CNOT` circuit propagates two errors [9], which illustrates how a `CNOT` applied to an unencoded state is not a fault-tolerant operation:

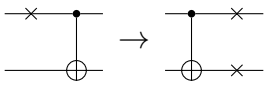


Figure 3.1: `CNOT` error propagation

We can look at this through the lens of commuting a gate through an `X` error on the first qubit:

$$U\mathtt{X}_1 = U\mathtt{X}_1U^\dagger U = \mathtt{X}_1\mathtt{X}_2U \tag{3.1}$$

where U is the `CNOT` gate here and $\mathtt{X}_i$ is the `X` gate on the $i$th qubit. From our discussion of the Clifford group previously we can see that this is same as the action of the `CNOT` gate on a Pauli error under conjugation. The full table of `CNOT`-conjugated Pauli errors is:

[1] An encoded block is the result of a quantum error correction encoding operation on an unencoded state.

Table 3.1: Pauli errors under conjugation by CNOT.

| Operation | Input | Output |
|---|---|---|
| $\texttt{CNOT}_{12}$ | $\texttt{X}_1$ | $\texttt{X}_1\texttt{X}_2$ |
| | $\texttt{X}_2$ | $\texttt{X}_2$ |
| | $\texttt{Z}_1$ | $\texttt{Z}_1$ |
| | $\texttt{Z}_2$ | $\texttt{Z}_1\texttt{Z}_2$ |
| | $\texttt{Y}_1$ | $\texttt{Y}_1\texttt{X}_2$ |
| | $\texttt{Y}_2$ | $\texttt{Z}_1\texttt{Y}_2$ |

From the outcome of this table it may be difficult to see how a CNOT could ever have a fault-tolerant construction, but we will soon see an explicit construction of such a gate. *Transversality*, which permits logical operations by applying physical gates to only single qubits in each block of a code, can be used to achieve fault-tolerant encoded gates. By limiting gates to only a single qubit in a code block, it means that errors cannot spread to other qubits in the code block [9].

### 3.1.1 Transversal $\bar{\texttt{X}}$, $\bar{\texttt{Z}}$, $\bar{\texttt{H}}$, $\bar{\texttt{S}}$, and $\overline{\texttt{CNOT}}$

As was shown in Nielsen and Chuang [9] we present constructions of the logical X and Z gates – denoted as $\bar{\texttt{X}}$ and $\bar{\texttt{Z}}$ – of the [[7, 1, 3]] Steane code, which uses 7 physical qubits to encode 1 logical qubit and can correct at most 1 physical error.

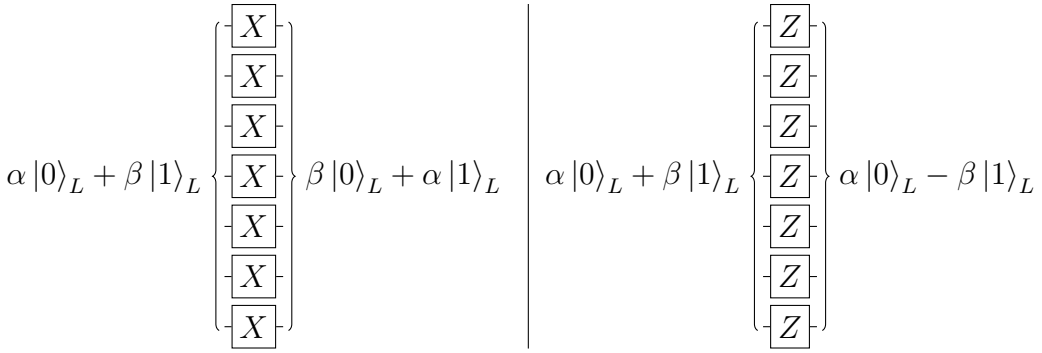



Figure 3.2: $\bar{\mathtt{X}}$ and $\bar{\mathtt{Z}}$ transversal gates

The Steane code also permits transversal implementations of logical $\bar{\mathtt{H}}$ and $\bar{\mathtt{S}}$:

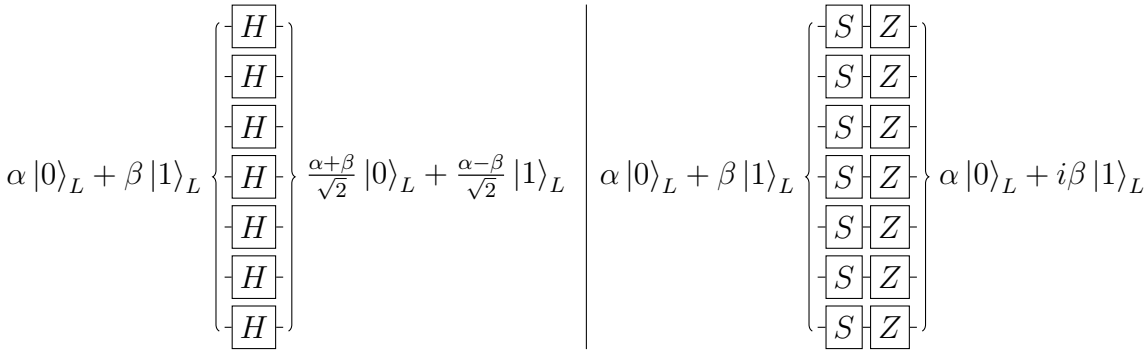



Figure 3.3: $\bar{\mathtt{H}}$ and $\bar{\mathtt{S}}$ transversal gates

Returning to our discussion about the CNOT gate we can present from Nielsen and Chuang [9] a transversal implementation of a logical $\overline{\mathtt{CNOT}}$ for the Steane code:

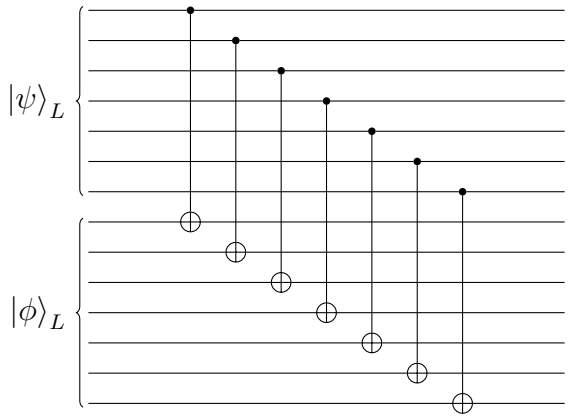



Figure 3.4: $\overline{\texttt{CNOT}}$ transversal gate

What's remarkable about this construction is that if we consider any of the errors in Table 3.1 we see that any physical `CNOT` propagates an error to at most a single qubit in either code block, which satisfies our definition of fault-tolerance [9].

### 3.1.2 Transversal $\overline{\texttt{T}}$

We conclude this section with a fault-tolerant construction of the logical $\overline{\texttt{T}}$ gate as provided in Nielsen and Chuang [9]. We will make use of *state teleportation*, which effectively replaces the state of an auxiliary qubit where the `T` operation is occurring with the `T` gate applied on $|\psi\rangle$:

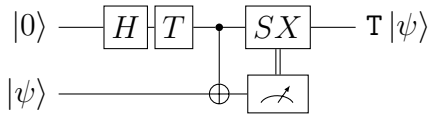



Figure 3.5: `T` gate via state teleportation

where `SX` here is the `S` gate followed by the `X` gate, both classically conditioned on

the measurement. These classically conditioned gates are sometimes referred to as *Clifford corrections*, which we've already established are fault-tolerant.

Utilizing the fault-tolerant approaches above and additionally fault-tolerant measurements, which we have not covered, we reach a fault tolerant construction of the $\overline{\mathtt{T}}$ gate for the Steane code:

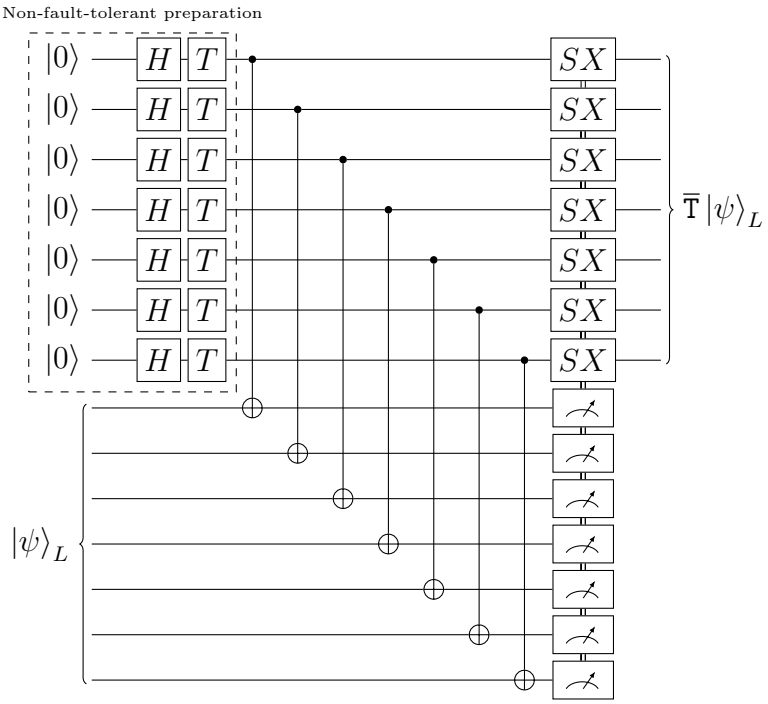



Figure 3.6: $\overline{\mathtt{T}}$ transversal gate

where the dashed box includes the only non-fault-tolerant procedure of the overall teleportation. Each of these auxiliary qubits must be prepared in the following state:

$$\mathtt{TH}\,|0\rangle = \frac{|0\rangle + e^{i\pi/4}\,|1\rangle}{\sqrt{2}} \tag{3.2}$$

The procedure of preparing these states reliably has been described by Zhou et al. [26].

## 3.2 Magic States

What if we were to use *gate teleportation* rather than state teleportation? Referring back to Figure 3.5 we can see that $|\psi\rangle$ ends up being swapped with the auxiliary qubit. Instead, we can have a desired gate teleported to $|\psi\rangle$ by consuming a resource known as a magic state.

**Definition 3.2.1** (Magic state)**.** A magic state is a non-Clifford state that can augment fault-tolerant Clifford operations to provide universal quantum computation, assuming that a ready supply of high-fidelity states exists [30].

We have actually seen a magic state already, which is known as the $|\mathtt{T}\rangle$ state:

$$|\mathtt{T}\rangle = \mathtt{T}\,|+\rangle = \frac{|0\rangle + e^{i\pi/4}\,|1\rangle}{\sqrt{2}} \tag{3.3}$$

This state exists in what is referred to as the 3rd level of the Clifford hierarchy [27] and is called a *Clifford magic state* [31]. Note that this term is not to be confused with Clifford gates, which are part of the 2nd level of the Clifford hierarchy. One important property (by definition) [31] of Clifford magic states $|V\rangle \equiv V\,|+\rangle^{\otimes m}$ is:

$$VP_jV^\dagger \in \mathcal{C}(m)\ \forall\ P_j \in \mathcal{P}(m) \tag{3.4}$$

where $V$ is a diagonal $m$-qubit unitary gate and $P_j$ is a Pauli gate on the $j$th qubit. The significance of Equation 3.4 is that any Clifford magic state permits Clifford

corrections, which were briefly mentioned in Section 3.1.2 and are covered in more detail in Section 3.2.3.

Another Clifford magic state that is important to our research is the $|\mathtt{CCZ}\rangle$ state:

$$|\mathtt{CCZ}\rangle = \mathtt{CCZ}\,|+\rangle^{\otimes 3} = \frac{1}{2\sqrt{2}} \sum_{x,y,z\in\{0,1\}} (-1)^{x\cdot y\cdot z}\,|x,y,z\rangle \tag{3.5}$$

We will consider this state in more depth in Chapter 4.

### 3.2.1 $|\mathtt{T}\rangle$ state injection gadget

In order to teleport a gate we use a *state injection gadget* [26] as an alternative to Figure 3.5:

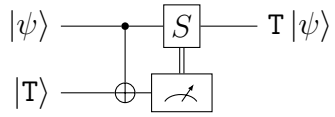



Figure 3.7: T gate via gate teleportation

Updating Figure 3.6 to utilize state injection we have the following $\overline{\mathtt{T}}$ gate implementation for the Steane code:

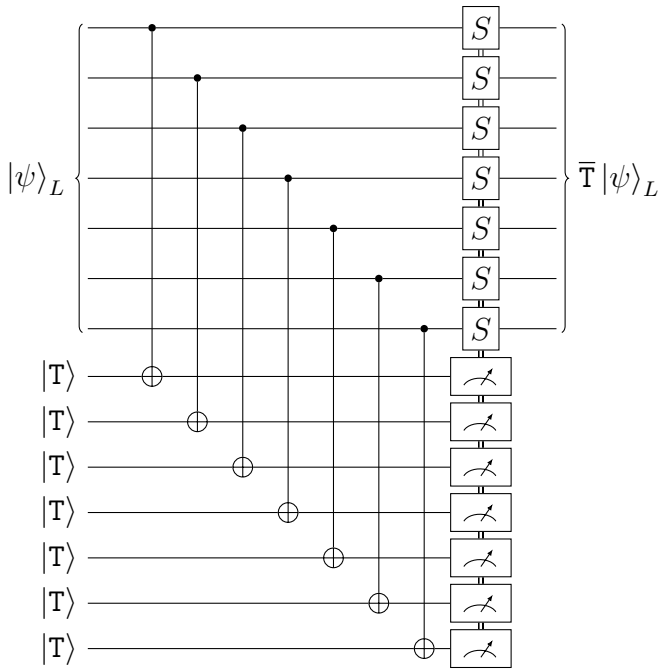



Figure 3.8: $\overline{\mathtt{T}}$ transversal gate with gate teleportation

It is worth noting that the full procedure here is considered fault-tolerant. The $|\mathtt{T}\rangle$ magic states provided are assumed to be fault-tolerantly prepared using a process known as *magic state distillation* [30], which purifies a state from many copies of mixed states.

Another aspect of Figure 3.8 to track are the resource requirements for implementing a logical $\overline{\mathtt{T}}$ gate. For every logical $\overline{\mathtt{T}}$ gate of the Steane code in our circuit we would need seven $|\mathtt{T}\rangle$ states; written $|\mathtt{T}\rangle^{\otimes 7}$. These resource requirements, while being important to quantum computation, are applicable to simulation in a classical setting as well, which we cover in Section 3.3.

### 3.2.2 $|\mathtt{CCZ}\rangle$ state injection gadget

We also have a state injection gadget for $|\mathtt{CCZ}\rangle$ from Beverland et al. [32]:

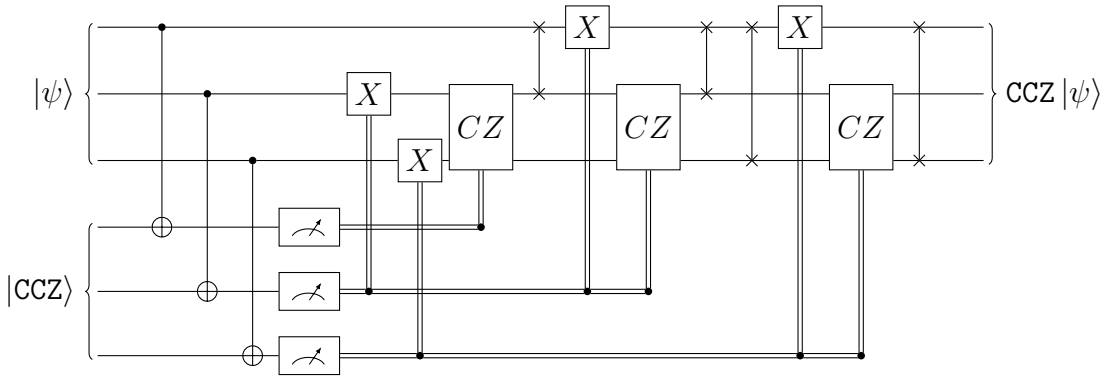



Figure 3.9: CCZ gate via gate teleportation

Note that the Steane code does not have a transversal implementation of logical $\overline{\mathtt{CCZ}}$ using state injection, but there are other codes that admit transversal $\overline{\mathtt{CCZ}}$s [33]. While this state injection gadget has more Clifford corrections than the $|\mathtt{T}\rangle$ state injection gadget, some circuits may be more efficiently implemented with $|\mathtt{CCZ}\rangle$ gadgets.

### 3.2.3 Clifford corrections

We briefly mentioned Clifford corrections above and will now expand further. We have seen constructions of two explicit state injection gadgets for $|\mathtt{T}\rangle$ and $|\mathtt{CCZ}\rangle$ in Figure 3.7 and Figure 3.9, respectively. The operations that are classically conditioned on measurement of the magic state are the Clifford corrections needed to realize the intended gate operation. For any Clifford magic state, $|V\rangle \equiv V\,|+\rangle^{\otimes t}$, if we have a measurement outcome $m = 0^t$, a bitstring of $t$ zeros, then we will have suc-

cessfully teleported the gate in the state gadget. For any other measurement $m \neq 0^t$ we will have implemented a gate $V_m$, which can be corrected by first determining the following:

$$C_x = \prod_{j:m_j=1} V X_j V^\dagger \tag{3.6}$$

and then applying this correction, $C_x$, to $V_m$. The result will be the Clifford gate(s) that should be conditioned on measurement $m$ and applied to $|\psi\rangle$ in the magic state gadget [31]. It is necessary to consider all such non-zero measurements in the state injection gadget in order to correct for all possible errors.

## 3.3 Stabilizer Rank

We return to classical simulation by turning our attention to a useful tool for state analysis known as stabilizer rank.

**Definition 3.3.1** (Exact Stablizer Rank, $\chi$ [3])**.** Suppose $|\psi\rangle$ is a pure $n$-qubit state. The exact stabilizer rank $\chi(|\psi\rangle)$ is the smallest integer $k$ such that $|\psi\rangle$ can be written as:

$$|\psi\rangle = \sum_{\alpha=1}^{k} c_\alpha |\phi_\alpha\rangle, \tag{3.7}$$

for some $n$-qubit stabilizer states $|\phi_\alpha\rangle$ and some complex-valued coefficients $c_\alpha$.

Note that $\chi(|\psi\rangle) = 1$ for all $|\psi\rangle$ that are stabilizer states; Otherwise $\chi(|\psi\rangle) > 1$.

### 3.3.1 Known bounds

Calculating the stabilizer decomposition of an arbitrary state $|\psi\rangle$ and subsequently stabilizer rank is non-trivial. The stabilizer rank of single-qubit magic

states to the $m$-th tensor power is currently lower-bounded by $\Omega(m)$ [34] and $T^{\otimes m}$ is specifically upper-bounded by $O(2^{\alpha m})$ where $\alpha \leq \log_2(3)/4 \approx 0.3962$ [35].

## 3.4 Extended stabilizer simulation methods

Utilizing exact stabilizer decompositions of states or, correspondingly, gates along with existing stabilizer simulation methods we can create an extended stabilizer simulation. There are two primary approaches – a gadget-based method or a sum-over-Cliffords method, both of which are described below. First, from Bravyi et al. [31] we represent an arbitrary circuit U as:

$$U = C_m V_m C_{m-1} V_{m-1} \dots C_1 V_1 C_0 \tag{3.8}$$

which acts on input state $|0\rangle^{\otimes n}$ and where $\{C_j\}$ are Clifford circuits and $\{V_j\}$ are non-Clifford gates. The main premise of extended stabilizer simulation is that with each $V_j$ we can utilize an exact stabilizer decomposition and then proceed with a stabilizer simulation of $C_{j+1}$ for each of the stabilizer states. The branching factor is equivalent to the multiplicative factor $\chi_j$ at each $V_j$. The stabilizer simulator that is used must track phase properly for each term as it is no longer irrelevant within the context of extended stabilizer simulation.

### 3.4.1 Gadget-based method

The gadget-based method simulates a quantum computer that would use state-injection gadgets of non-Clifford states in order to effect gate teleportation. We would transform an existing circuit $U$ as described above into a gadget-based or *gad-*

*getized* circuit where each $V_j$ would be similar to Figure 3.7 or Figure 3.9. The resulting circuit would have $n+\eta$ qubits where the $\eta$ additional qubits correspond to Clifford magic states $|V_j\rangle$. It was shown by Bravyi et al. that $\chi(U|0\rangle^{\otimes n}) = \chi(\bigotimes_j |V_j\rangle)$ [31].

### 3.4.2 Sum-over-Cliffords method

The sum-over-Cliffords method described by Bravyi et al. [31] constructs a decomposition of U above in the form of a superposition over Clifford unitaries:

$$U = \Sigma_j c_j K_j \tag{3.9}$$

where each $K_j$ is a unitary Clifford and each $c_j$ is a complex-valued coefficient. We can see that applying the right-hand side of Equation 3.9 to a state $|0\rangle^{\otimes n}$ will produce the desired result of $U|0\rangle^{\otimes n}$. Since the right-hand side of Equation 3.9 are all Clifford unitaries, which produce stabilizer states, then it follows that $\chi(U|0\rangle^{\otimes n}) = j$. This simulation method utilizes a stabilizer simulation for each $K_j$.

## 3.5 Practical Resource Limits

Previously, we looked at the practical resource limits for stabilizer simulation, so now we turn our attention to extended stabilizer simulation. We already know that execution time scales exponentially in the number of non-Clifford gates. Getting a sense of what this means practically can be helpful. We use the estimate of 15 seconds for simulating 20 thousand qubits, 8 million Clifford gates, and 1 million measurements based on the results[2] from the currently best-performing stabilizer

[2]Gidney specified using an Intel Core i7-8650U CPU @ 1.90GHz and 16GB of RAM

simulator – Stim [24].

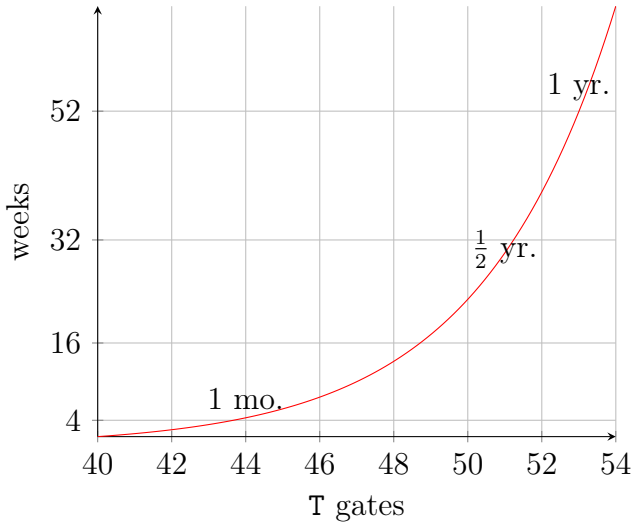



Figure 3.10: Weeks vs. `T` gates; Estimated time to execute on a single machine using Stim with 20 thousand qubits, 8 million Clifford gates, 1 million measurements, and the specified number of `T` gates.

# Chapter 4

# Results

When considering extended stabilizer simulation via magic state injection we can choose which magic state we want to use for universality. We turn our focus now to the decomposition of CCZ magic states over multiple qubits. Prior to this work it was known that CCZ magic states could be efficiently used for new distillation protocols involving parity checks [36] or *synthillation* [37], which combines distillation and injection into one step. Our work demonstrates that CCZ magic states can also be efficiently used in extended stabilizer simulation. We found that all CCZ magic states of $n \leq 5$ qubits have stabilizer decompositions that are rank-2. We also establish that CCZ circuits of $n \leq 5$ qubits all reduce to $k \leq 2$ CCZ gates and additional Clifford gates, which is known as *Clifford equivalency*. There are 15 unique, real-valued rank-2 decompositions (Table 4.1) of the three qubit $|\mathtt{CCZ}\rangle$ magic state. We show that all CCZ circuits composed of four qubits reduce to a single CCZ with additional Clifford gates and therefore are also rank-2 (Lemma 4.2.2). For CCZ circuits composed of five qubits, we show that each distinct circuit has a single, real-valued rank-2 decomposition and is reducible to either one or two CCZs (Appendix A). These circuit reductions connect stabilizer rank to the *transversal number* (defined in Section 4.2.4.2) when considering the arbitrary stabilizer state basis form. We propose that extended stabilizer simulations use a state injection gadget of a multiple CCZ circuit rather

than individual $|\texttt{CCZ}\rangle$ magic states to improve performance.

## 4.1 CCZ Magic

Much has been said about $|\texttt{T}\rangle$ Clifford magic states in the literature [31,35,38]. Here we decide to focus on CCZ magic states of the form:

$$|D\rangle = D\,|+\rangle^{\otimes m} \tag{4.1}$$

where $D$ is a diagonal circuit composed of Z, CZ, and CCZ gates. It is well-known that diagonal matrices commute, so it is possible to move all of the Z and CZ gates to the beginning or end of the circuit to simplify analysis. Additionally, $\texttt{Z} \subset \mathbb{P}$ and $\texttt{CZ} \subset \mathcal{C}(2)$, which means that neither of these gates will affect the stabilizer rank of the resulting states. So, for the remainder of this chapter we will focus only on combinations of CCZ gates.

### 4.1.1 CCZ constructed from T gates

Before we proceed to our results, let us first consider the known construction of a CCZ gate from T gates based on a similar construction for the TOFFOLI gate [39]:

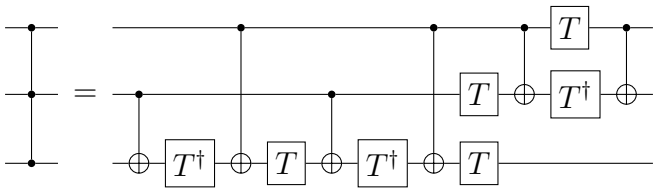



Figure 4.1: T gate construction of CCZ

If we were to use the gadget-based extended stabilizer simulation from the pre-

vious chapter this would require seven $|V_j\rangle \equiv |\mathtt{T}\rangle$-like states for a single $|\mathtt{CCZ}\rangle$. Qassim et al. showed that $\chi(|\mathtt{T}\rangle^{\otimes 6}) \leq 6$, so we would have $\chi(|\mathtt{T}\rangle^{\otimes 7}) \leq \chi(|\mathtt{T}\rangle^{\otimes 6})\chi(|\mathtt{T}\rangle) = 12$ since stabilizer rank is submultiplicative (i.e. $f(xy) \leq f(x)f(y)$) [35]. The best yet stabilizer rank of $\chi(|\mathtt{T}\rangle^{\otimes 7}) \leq 12$ was also numerically found by Bravyi et al. [31]. Selinger found that if we allow the use of auxiliary qubits and measurements, then we only require 4 T gates [38], which was proven to be optimal via resource theory [40]. We would then, at a minimum, have a stabilizer rank of $\chi(|\mathtt{T}\rangle^{\otimes 4}) \leq 4$ per $|\mathtt{CCZ}\rangle$ [38] by using this approach.

### 4.1.2 $|\mathtt{D}\rangle$ magic state injection

We established in Section 3.2 that non-Clifford gates, such as T, can generate what are known as Clifford magic states. We established in the previous Section 4.1.1 that CCZ can be constructed from T and Clifford gates. So, it follows that our $|D\rangle$ states are in fact Clifford magic states. Furthermore, we established in Section 3.2.3 that we can provide Clifford corrections to any Clifford magic state. Therefore, we reinforce here that magic state injection is possible with sub-circuits composed of Z, CZ, and CCZ gates.

## 4.2 Stabilizer Decompositions

Now, we demonstrate our results by looking at CCZ circuits over three, four, and five qubits.

### 4.2.1 Notation

In order to be succinct in our results we will utilize a property described by García et al. – that all non-zero amplitudes of stabilizer states have the same magnitudes [41]. Since our results only deal with real stabilizer states, we will use a *ternary* notation where a stabilizer state will be described by:

**Definition 4.2.1** (Ternary form)**.**

$$|\psi\rangle = \gamma \sum_{x \in \{0,1\}^n} a_x |x\rangle \tag{4.2}$$

where $\gamma \in \mathbb{C}$ is a normalization factor that will be omitted, and $a_x \in \{-1, 0, 1\}$ is a phase factor for a specific computational basis term in the state.

As an example, the state vector for $|\texttt{CCZ}\rangle$ (written as a row for compactness):

$$|\psi\rangle = \begin{bmatrix} 0.354 & 0.354 & 0.354 & 0.354 & 0.354 & 0.354 & 0.354 & -0.354 \end{bmatrix}^T \tag{4.3}$$

has the ternary form:

$$|\psi\rangle = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & -1 \end{bmatrix}^T \tag{4.4}$$

### 4.2.2 CCZ magic state for three qubits

Let's start with a decomposition of a three qubit CCZ magic state, which is usually denoted $|\texttt{CCZ}\rangle$ (see Figure 4.2). Trivially, one can see that having a non-Clifford gate, such as a CCZ, would automatically require a stabilizer rank $\chi > 1$.

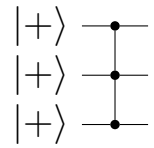



Figure 4.2: $\mathtt{CCZ}\,|+\rangle^{\otimes 3} = |\mathtt{CCZ}\rangle$

It is easily verified that multiple rank-2 decompositions exist and, in fact, there are 15 decompositions of this magic state:

Table 4.1: $|\mathtt{CCZ}\rangle$ decompositions

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 2 | 1 | 1 | 1 | −1 | 1 | 1 | 1 | −1 |
| | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| 3 | 1 | 1 | 1 | 1 | 1 | −1 | 1 | −1 |
| | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| 4 | 1 | −1 | 1 | 1 | 1 | 1 | 1 | −1 |
| | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 5 | 1 | 1 | 1 | 1 | 1 | 1 | −1 | −1 |
| | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| 6 | 1 | 1 | −1 | 1 | 1 | 1 | 1 | −1 |
| | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| 7 | 1 | 1 | 1 | 1 | −1 | 1 | 1 | −1 |
| | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| 8 | 1 | −1 | −1 | −1 | −1 | −1 | −1 | 1 |
| | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 9 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 0 |
| | 1 | 0 | 1 | 0 | 0 | 1 | 0 | −1 |
| 10 | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 0 |
| | 0 | −1 | −1 | 0 | −1 | 0 | 0 | 1 |
| 11 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 |
| | 1 | 0 | 0 | 1 | 1 | 0 | 0 | −1 |
| 12 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 |
| | 0 | −1 | 0 | −1 | 0 | −1 | 0 | 1 |

Table 4.1 – continued from previous page

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 13 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 |
| | 1 | 1 | 0 | 0 | 0 | 0 | 1 | $-1$ |
| 14 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 |
| | 0 | 0 | $-1$ | $-1$ | 0 | 0 | $-1$ | 1 |
| 15 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| | 0 | 0 | 0 | 0 | $-1$ | $-1$ | $-1$ | 1 |

*Note*: Does not include coefficients $c_\alpha$ of the stabilizer decomposition

Given a `CCZ` magic state $|\psi\rangle$, the following algorithm was used for finding all real-valued, rank-2 stabilizer decompositions in our results. Since all amplitudes are equal and we are searching for linear combinations of stabilizer states where the signs will match $|\psi\rangle$, we only need to check the following:

$$\begin{aligned} |\psi\rangle &\stackrel{?}{=} \pm|a\rangle \pm |b\rangle \\ |\psi\rangle &\stackrel{?}{=} \pm|a\rangle \pm 2|b\rangle \\ |\psi\rangle &\stackrel{?}{=} \pm 2|a\rangle \pm |b\rangle \end{aligned} \tag{4.5}$$

where $|a\rangle$ and $|b\rangle$ are stabilizer states. All states are in ternary form.

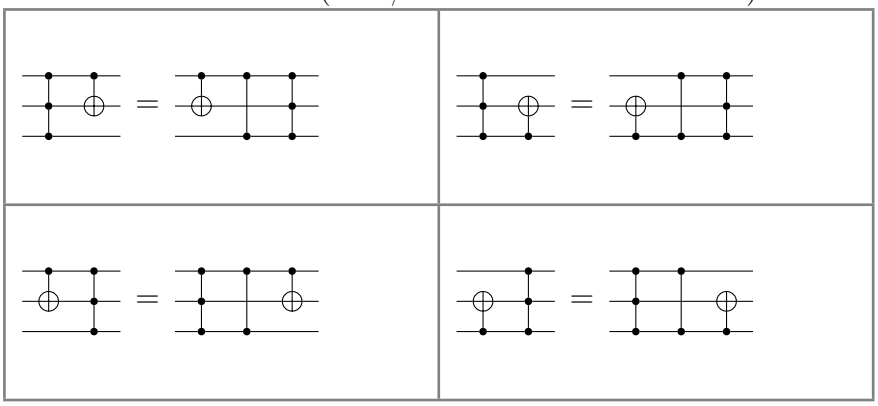
**Algorithm 1** Real Stabilizer Rank-2 Decomposition Search

**Require:** $n > 0$, $|\psi\rangle$
  $target \leftarrow |\psi\rangle$ in ternary form
  $decompositions \leftarrow []$
  $states \leftarrow$ all real, n-qubit stabilizer states in ternary form
  **for** $i$, $a$ in enumerate($states$) **do**
    **for** $j$, $b$ in enumerate($states$[i+1:]) **do**

```
        if ±a ± b = target
            ∨ ± a ± 2 * b = target
            ∨ ± 2 * a ± b = target then
            decompositions ← decompositions + (a, b)
        end if
    end for
end for
return decompositions
```

Before we move to CCZ circuits over 4 qubits, let's consider the commutation rules for CCZ and CNOT. These will be useful for reducing four-qubit multiple CCZ circuits to single CCZ circuits.

**Lemma 4.2.1** (CCZ / CNOT commutation rules)**.**



*Proof.* Matrices of the L.H.S. and R.H.S. are equal. □

### 4.2.3 CCZ magic states for four qubits

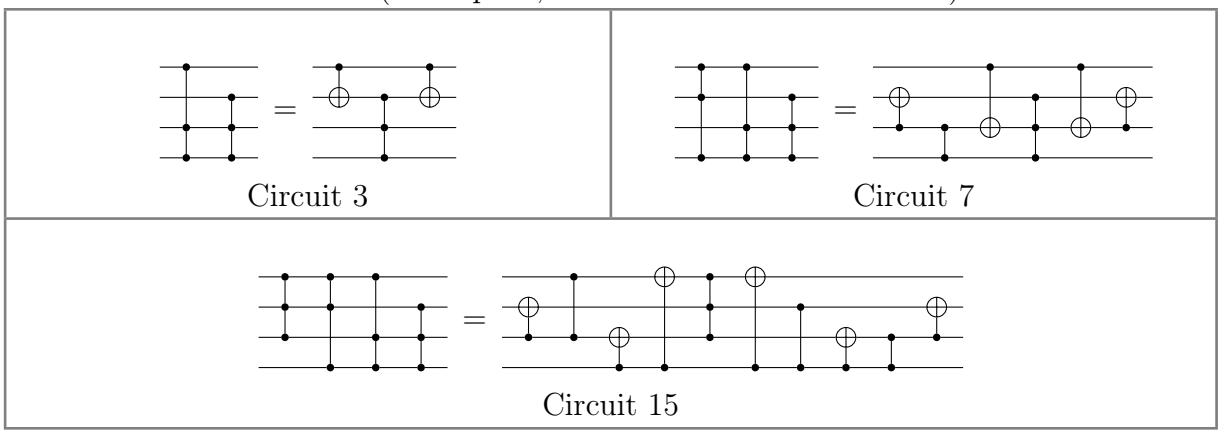
For four qubits there are $\binom{4}{3} = 4$ ways to place a single CCZ and therefore $2^4 = 16$ circuits (including the identity circuit). We wish to include only the distinct circuits from this set with respect to conjugation by SWAPs. First, we start with a

circuit representation over $\mathbb{Z}_2$ in matrix form to represent all the permutations of `CCZ` placement:

$$\begin{bmatrix} 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 \end{bmatrix} \tag{4.6}$$

To find the distinct circuits, we start by enumerating all bitstrings in $\mathbb{Z}_2^4$, converting those to diagonal matrices with each bitstring as the diagonal, and multiplying each with our matrix above. Next, we perform all $4! = 24$ permutations of row swaps, compare against our set of distinct circuits, and only admit membership if no duplicate is found. So, excluding circuits over the subspace of three qubits and the circuits that are equivalent up to swap conjugation we are left with 3 distinct circuits. Among these distinct circuits, each is reducible to a single `CCZ` circuit as demonstrated in the following identities[1]:

**Lemma 4.2.2** (Four-qubit, distinct `CCZ` circuit identities)**.**




[1]Circuit 3 has an identity that was already presented by Campbell et al. [36], while the remaining identities have not been presented in literature known to us.

*Proof.* Matrices of the L.H.S. and R.H.S. are equal. □

Since all four-qubit CCZ circuits can be reduced to circuits with a single CCZ we can conclude that every four-qubit CCZ circuit has stabilizer decompositions that are rank-2.

### 4.2.4 CCZ magic states for five qubits

For five qubits, there are $\binom{5}{3} = 10$ ways to place a single CCZ and thus $2^{10} = 1024$ circuits (including the identity circuit). Excluding circuits over the subspace of three and four qubits and circuits that are equivalent up to swap conjugation, we have 29 distinct[2] circuits. For each of these circuits, there exists only a single, real-valued rank-2 decomposition. See Appendix A.

What property allows all five qubit CCZ circuits to have rank-2 stabilizer decompositions? As it turns out, all five qubit CCZ circuits reduce to at most two adjacent CCZ gates along with some Clifford gates. As an example, consider how the maximally-filled circuit of 10 CCZs reduces:

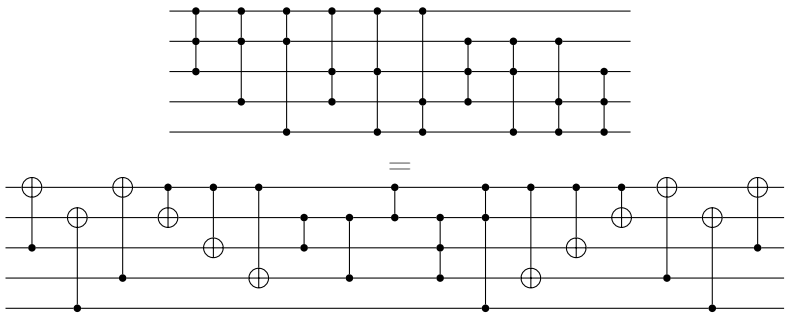


Figure 4.3: Five-qubit, maximally filled CCZ circuit identity

[2]According to the same approach used in Section 4.2.3

To demonstrate that all 29 distinct circuits reduce to one or two CCZs and additional Clifford gates we used an approach of conjugating each circuit by various CNOT circuits until a minimal decomposition was found. The CNOT circuits were generated by enumerating all $5 \times 5$ invertible matrices of $GL(5, 2)$ and using an affine construction [42] to instantiate the circuit. The resulting identities are shown in Appendix A.

Note here that when we construct the magic state for this circuit we use an input state $|+\rangle^{\otimes 5}$, which is stabilized by the first CNOT portion of the circuit identity at the bottom of Figure 4.3. Effectively, we can remove these gates when considering the stabilizer decomposition. So, now we have remaining diagonal Clifford gates, CCZs, and additional Clifford gates following. We can commute the diagonal Clifford gates past the CCZ gates and combine them with the other Clifford gates. Finally, we are left with two CCZs, which share a common qubit. If we split the $|+\rangle$ state of this common qubit into $|0\rangle$ and $|1\rangle$ states, then we can consider the action of the CCZ gates on these two states. For $|0\rangle$, a CCZ gate acts like the identity gate. For $|1\rangle$, a CCZ gate will behave like a CZ gate on the other two qubits. Both of the states are stabilizer states, so overall it provides a stabilizer decomposition of rank-2.

#### 4.2.4.1 Hypergraphs

Prior to the reduction we have the following hypergraph:

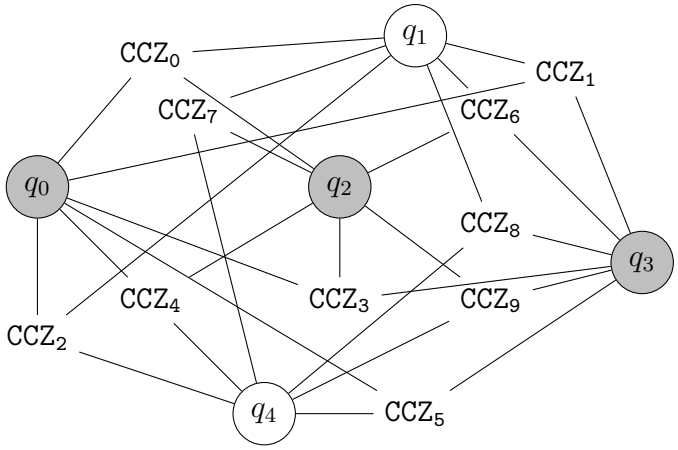



Figure 4.4: Hypergraph, $H_1$, for maximally filled CCZ circuit
Enclosed nodes are vertices of the hypergraph (qubits) and open nodes are hyperedges (CCZs). Gray nodes denote a minimal set of qubits that connects all CCZs.

Following the reduction from Figure 4.3 we have the following hypergraph:

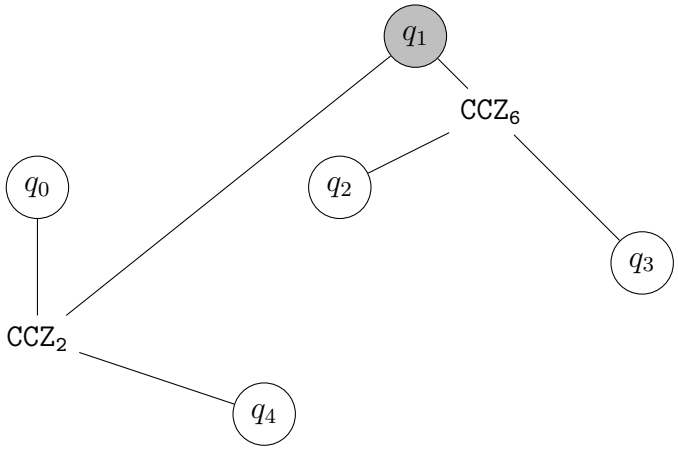



Figure 4.5: Hypergraph, $H_2$, for maximally filled CCZ circuit after reduction
Enclosed nodes are vertices of the hypergraph (qubits) and open nodes are hyperedges (CCZs). Gray nodes denote a minimal set of qubits that connects all CCZs.

#### 4.2.4.2 Transversal number

We notice that the number of qubits required to span all CCZs reduces from three to one. The property we are observing is known as the transversal number [43]:

**Definition 4.2.2** (Transversal Number of a Hypergraph, $\tau$)**.** Let $H = (V, E)$ be a hypergraph with $n$ vertices and $m$ edges. The transversal number is the minimum cardinality of a set of vertices that intersects all edges of H.

The upper-bound of a transversal number is:

**Lemma 4.2.3** (Upper bound of $\tau(H)$ [43])**.** *For some positive constant $c_k$ the transversal number of a k-uniform hypergraph is upper-bounded by:*

$$\tau(H) \leq c_k(n + m) \tag{4.7}$$

Relating this to stabilizer rank, we have:

**Lemma 4.2.4** (Upper bound of $\chi(|D\rangle)$ [44])**.** *For some magic state $|D\rangle$, it is upper bounded by:*

$$\chi(|D\rangle) \leq 2^{\tau(H)} \tag{4.8}$$

*Proof.* Since $|D\rangle = D\,|+\rangle^{\otimes m}$ we can split one of the $|+\rangle$ qubits to $|0\rangle$ and $|1\rangle$. We can then evaluate the circuit on each of the two states and wherever a CCZ interacts with the state it either becomes identity or a CZ gate. If we split on a set $S$ of qubits that span all CCZs, then we have an upper bound of stabilizer rank as $2^{|S|}$. The cardinality of the minimal set of qubits is the transversal number. □

Since CCZs span three qubits, we care about 3-uniform hypergraphs where $n$ is the number of qubits and $m$ is the number of CCZs. It is known that $c_3 = 1/4$ [45], so we have $\tau(H) \leq \frac{n+m}{4}$. Applying this to Figure 4.4, we would have $\tau(H_1) \leq \frac{5+10}{4} = 3.25$. Note that $\tau$ must be an integer, so $\tau(H_1) \leq 3$, which is supported by the above graph. Therefore, $\chi(|D\rangle) \leq 2^3 = 8$.

Now, if we apply the same treatment to Figure 4.5, then we have $\tau(H_2) \leq \frac{5+2}{4} = 1.75$, which is also supported by the above graph where $\tau(H_2) = 1$. Therefore, $\chi(|D\rangle) \leq 2^1 = 2$ and matches the rank of our stabilizer decompositions.

One way to interpret these results is to consider the *basis* in which the the transversal number bound is being applied. In the computational basis, the transversal number bounds the actual stabilizer rank loosely. However, in the CNOT-transformed circuit of Figure 4.3, we can see that in the arbitrary stabilizer state basis form we have the true stabilizer rank of the magic state.

### 4.2.5 CCZ magic states for six qubits

At six qubits we have our first case of a disjoint set of CCZs: $\texttt{CCZ} \otimes \texttt{CCZ}$. Upon exhaustive search[3] using the rank-2 search algorithm (Algorithm 1) over all $9,694,080$ real-valued stabilizer states, no rank-2 decomposition was found. We know there exists a stabilizer rank of $\chi(|\texttt{CCZ}\rangle \otimes |\texttt{CCZ}\rangle) \leq 4$ by taking the tensor product of $|\texttt{CCZ}\rangle$ decompositions. We could neither prove nor disprove the existence of a rank-3 decomposition. See Appendix B for an additional algorithm that provides

[3]Using 6 cores of an Intel i7 w/ 16GB of memory this took 324.25 hours.

approximate upper-bounds on stabilizer rank.

## 4.3 Conclusion

It is known that utilizing CCZ magic states for extended stabilizer simulation would be more performant than T magic states for synthesizing a CCZ when using gadget-based simulation. The CCZ gate is equivalent to TOFFOLI when conjugated by a single set of H gates. A TOFFOLI gate requires 4 T gates and 3 $\mathtt{T}^\dagger$ gates, or at best 4 T gates when using auxiliary qubits and measurement. When it comes to resource estimation, a quantum algorithm that makes use of CCZ gates would require considerably fewer $|\mathtt{CCZ}\rangle$ magic states than an equivalent one composed of $|\mathtt{T}\rangle$ magic states by a factor of 7 (or 4, respectively if using auxiliary qubits and measurement).

When considering five-qubit partitions of a circuit that use CCZs, we have demonstrated that the number of CCZ gates is reducible by up to a factor of 5 (see Figure 4.3). It is known that using five-qubit CCZ magic state injection can be more efficient than repeated single-$|\mathtt{CCZ}\rangle$ magic state injection [36]. However, it isn't trivial to see how to apply this to classical simulation. For extended stabilizer simulations, we have shown that a sub-circuit of up to 10 CCZs can be reduced to a single rank-2 decomposition (see Appendix A) instead of utilizing two rank-2 $|\mathtt{CCZ}\rangle$ decompositions when the circuit is first reduced. Effectively, this results in tracking only 2 stabilizer states instead of 4 states per five-qubit partition of CCZ circuits. These results combined with the rank-2 decompositions for three and four qubit CCZ circuits provide a new lower-bound for the complexity of simulating Clifford+CCZ circuits in general.

Bravyi et al. suggest that the stabilizer rank for CCZ magic states might be upper-bounded by the transversal number, $\tau$, of a 3-uniform hypergraph [31] (i.e. the minimum number of qubits/wires that span all CCZ gates) where $\chi(|\psi\rangle) \leq 2^{\tau}$. However, we found via experimentation that the transversal number approach overestimates the actual stabilizer rank significantly if the circuit is not first reduced. For example, the transversal number for a maximally-filled five qubit circuit with 10 CCZs is 3, but we found that it is reducible to 1 and supports the stabilizer rank of 2 that we also found. What we gather from these results is that the transversal number is loose in the computational basis, but might still be valid in the arbitrary stabilizer state basis form. We conjecture that the stabilizer rank for CCZ magic states may be as simple as $2^{\lfloor n/3 \rfloor}$ for $n$ qubits, so this remains an open question for further exploration.

Supporting code for this research is available on GitHub at:
https://github.com/amirebrahimi/stabilizer-toolkit/.

# Appendices

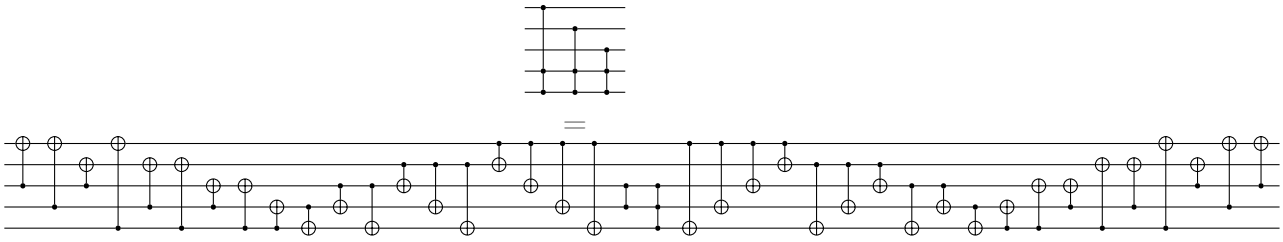
# Appendix A

# Decompositions

The following circuits are enumerated by placing all $\binom{5}{3} = 10$ `CCZ`s in series and using binary counting over $\mathbb{Z}_2^{10}$ to select whether a `CCZ` is included in the circuit. For example, Circuit 19 has a selection vector $\big[0000010011\big]$, which selects the 6th, 9th, and 10th columns from the matrix in Equation A.1 The decomposition mapping corresponds to the entry in Table A.1.

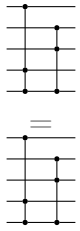
$$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 \end{bmatrix} \tag{A.1}$$

Figure A.1: Distinct[1]circuits for five qubit `CCZ` magic states

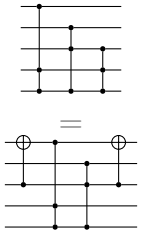


Circuit 19 → Decomposition 1

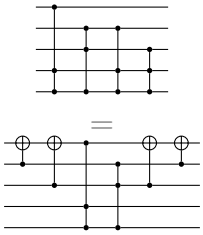
[1]As characterized in Section 4.2.3



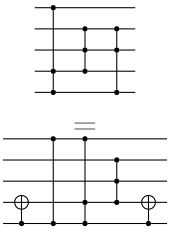
Circuit 20 → Decomposition 2



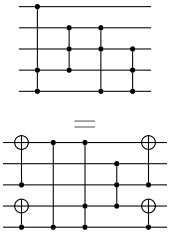
Circuit 21 → Decomposition 3



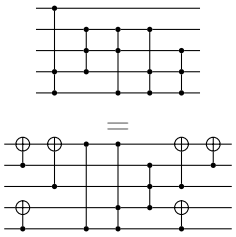
Circuit 23 → Decomposition 4



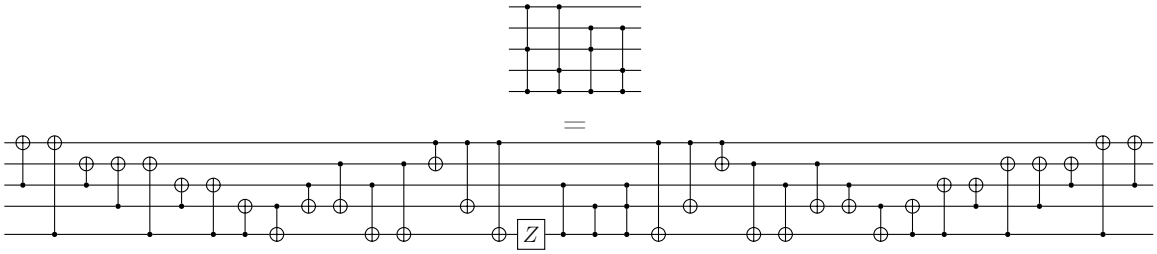
Circuit 28 → Decomposition 5



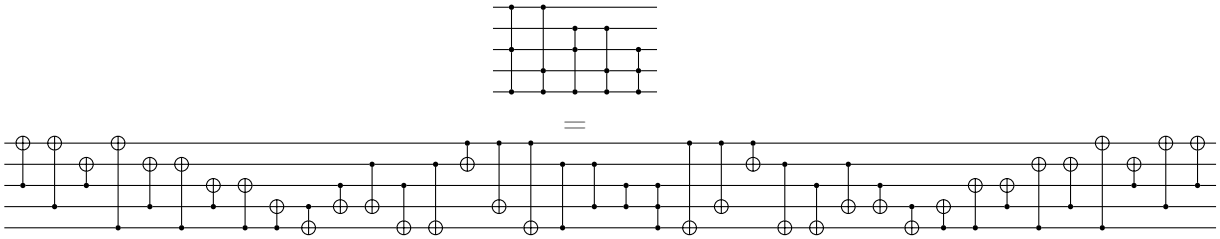
Circuit 29 → Decomposition 6



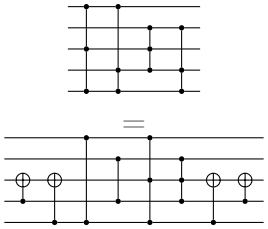
Circuit 31 → Decomposition 7



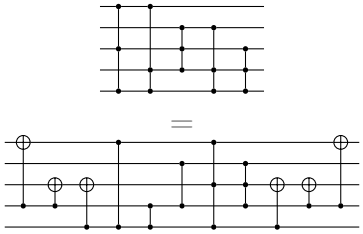

Circuit 54 → Decomposition 8



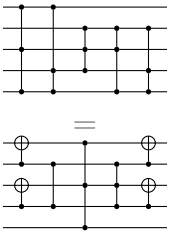
Circuit 55 → Decomposition 9



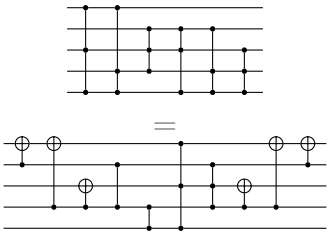
Circuit 58 → Decomposition 10



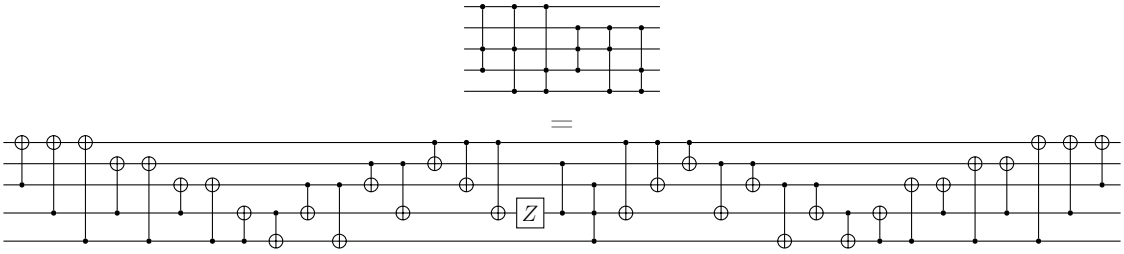
Circuit 59 → Decomposition 11



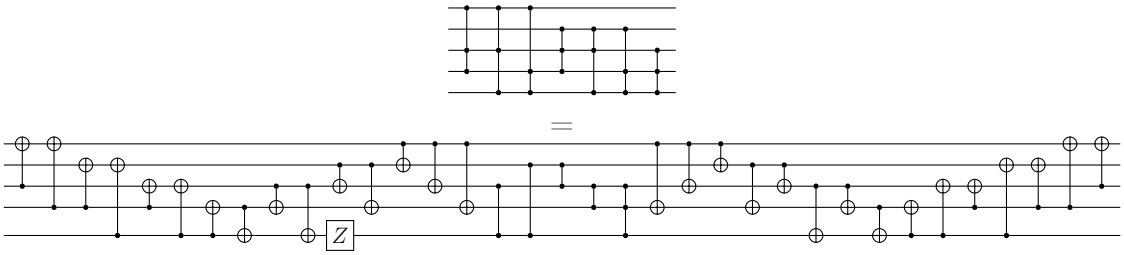
Circuit 62 → Decomposition 12



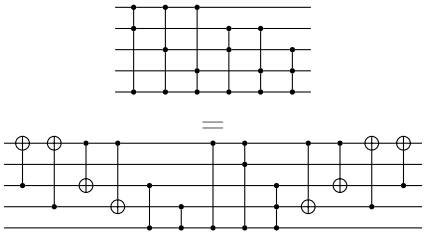
Circuit 63 → Decomposition 13



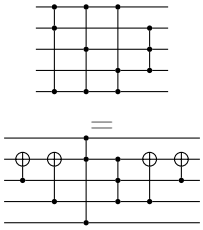

Circuit 126 → Decomposition 14



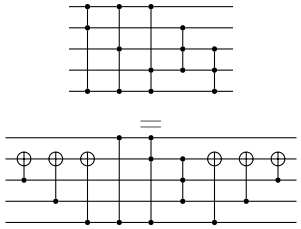

Circuit 127 → Decomposition 15



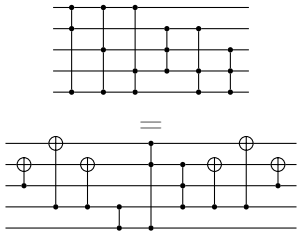

Circuit 183 → Decomposition 16



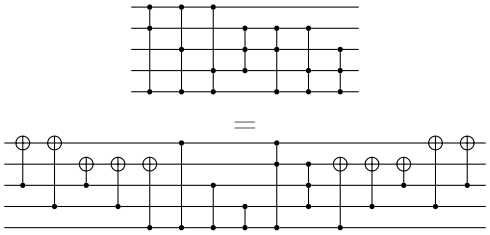

Circuit 184 → Decomposition 17



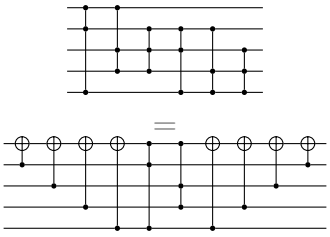
Circuit 185 → Decomposition 18



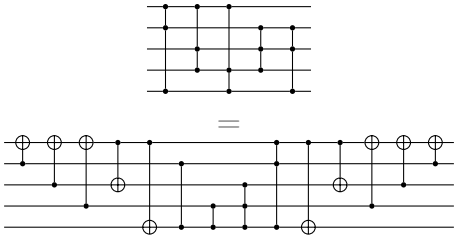
Circuit 187 → Decomposition 19



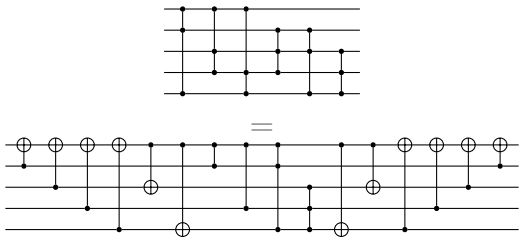
Circuit 191 → Decomposition 20



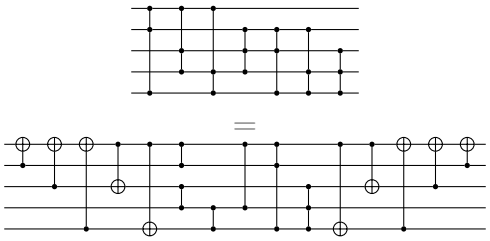
Circuit 207 → Decomposition 21



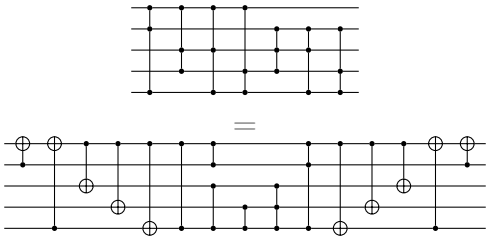
Circuit 220 → Decomposition 22



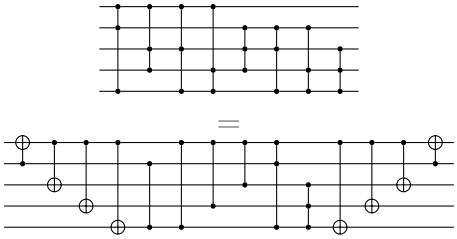
Circuit 221 → Decomposition 23



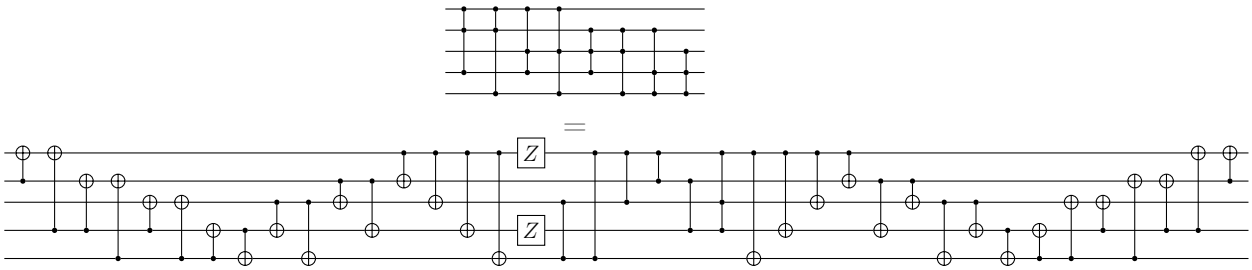
Circuit 223 → Decomposition 24



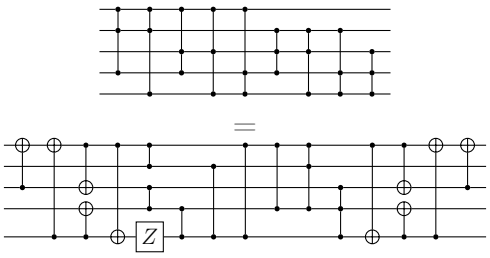
Circuit 254 → Decomposition 25



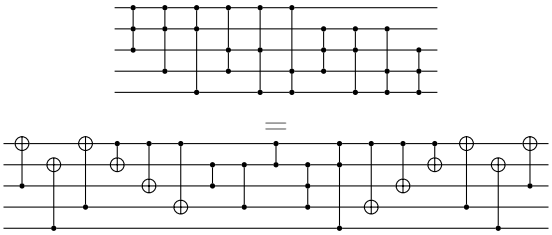
Circuit 255 → Decomposition 26




Circuit 495 → Decomposition 27




Circuit 511 → Decomposition 28



Circuit 1023 → Decomposition 29

Table A.1: Distinct CCZ magic state decompositions for five qubits

| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| | 0 | 0 | 0 | 0 | 1 | 1 | 1 | −1 | 1 | 1 | 1 | −1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | −1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | −1 |
| 2 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 |
| | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | −1 | 0 | −1 | 0 | 1 | 0 | −1 | 0 | 1 | 0 | −1 | 0 | 1 | 0 | −1 | 0 | −1 | 0 | 1 |
| 3 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | −1 | 0 | 1 | 0 | 1 | 0 | −1 | 0 | 1 | 0 | 1 | 0 | −1 | 0 | 1 | 0 | 1 | 0 | −1 | 0 | −1 | 0 | −1 | 0 | −1 |
| | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 |
| 4 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | −1 | 0 | 1 | 0 | −1 | 0 | −1 | 0 | −1 | 0 | 1 | 0 | −1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | −1 | 0 | 1 | | |
| | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | | |
| 5 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | −1 | −1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | −1 | −1 | 0 |
| | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | −1 | 1 | 0 | 0 | −1 | 1 | 0 | 0 | −1 | 1 | 0 | 0 | −1 |
| 6 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | −1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | −1 | 1 | 0 | 0 | −1 | 1 | 0 | 0 | −1 | 1 | 0 | 0 | 1 | | | | |
| | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | −1 | −1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | −1 | −1 | 0 | | | | |
| 7 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | −1 | −1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | −1 | −1 | 0 | | | | |
| | 1 | 0 | 0 | 1 | 1 | 0 | 0 | −1 | 1 | 0 | 0 | −1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | −1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | −1 | | | | |
| 8 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | −1 | 1 | −1 | 1 | 1 | 1 | 1 | 1 | −1 | 1 | −1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 9 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | −1 | 1 | −1 | 0 | 0 | 0 | 0 | 1 | −1 | 1 | −1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| | 0 | 0 | 0 | 0 | 1 | 1 | 1 | −1 | 1 | 1 | 1 | −1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | −1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | −1 |
| 10 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 1 |
| | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | −1 | 0 | 1 | −1 | 0 | 1 | 0 | 0 | −1 | 0 | 1 | 0 | −1 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 0 |
| 11 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | −1 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | −1 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | −1 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | −1 |
| | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | −1 | 0 | 1 | −1 | 0 | 1 | 0 | 0 | −1 | 0 | 1 | 0 | −1 | 1 | 0 | 0 | 1 | 0 | −1 | −1 | 0 |
| 12 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | −1 | 1 | −1 | 0 | 0 | 0 | 0 | 1 | −1 | 1 | −1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 |
| | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | −1 | −1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | −1 | −1 |
| 13 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | −1 | 1 | −1 | 0 | 0 | 0 | 0 | 1 | −1 | 1 | −1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 |
| | 1 | 1 | 0 | 0 | 0 | 0 | 1 | −1 | 1 | 1 | 0 | 0 | 0 | 0 | −1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | −1 | 1 | 1 | 0 | 0 | 0 | 0 | −1 | 1 |
| 14 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | −1 | 1 | −1 | −1 | −1 | 1 | 1 | 1 | −1 | 1 | −1 | −1 | −1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 15 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | −1 | −1 | 1 | 0 | 0 | 0 | 0 | 1 | −1 | −1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| | 0 | 0 | 0 | 0 | 1 | 1 | 1 | −1 | 1 | 1 | 1 | −1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | −1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | −1 |
| 16 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | −1 | 0 | 1 | 0 | −1 | 0 | −1 | 0 | −1 | 0 | 1 | 0 | −1 | 0 | −1 | 0 | −1 | 0 | −1 | 0 | −1 | 0 | −1 | 0 | 1 |
| | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 |
| 17 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | −1 | −1 | 0 | 0 | 1 | −1 | 1 | −1 | 0 | 0 | 1 | −1 | 0 | 0 | 0 | 0 | −1 | 1 |
| | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 |
| 18 | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | −1 | 0 | −1 | 1 | 0 | 0 | −1 | 1 | 0 | 1 | 0 | 0 | −1 |
| | 0 | 1 | 1 | 0 | 1 | 0 | 0 | −1 | 1 | 0 | 0 | 1 | 0 | 1 | −1 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | −1 | 1 | 0 | 0 | 1 | 0 | 1 | −1 | 0 |
| 19 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | −1 | 0 | 0 | 1 | −1 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | −1 | 0 | 0 | 1 | −1 | 1 | 1 | 0 | 0 |
| | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | −1 | −1 | 0 | 0 | 1 | −1 | 1 | −1 | 0 | 0 | 1 | −1 | 0 | 0 | 0 | 0 | −1 | 1 |
| 20 | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | −1 | 0 | −1 | 1 | 0 | 0 | −1 | 1 | 0 | 1 | 0 | 0 | −1 |
| | 0 | 1 | 1 | 0 | 1 | 0 | 0 | −1 | 1 | 0 | 0 | −1 | 0 | −1 | −1 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | −1 | 1 | 0 | 0 | −1 | 0 | −1 | −1 | 0 |
| 21 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | −1 | 1 | 0 | 0 | −1 | 0 | −1 | −1 | 0 | 1 | 0 | 0 | 1 | 0 | 1 | −1 | 0 | 0 | −1 | 1 | 0 | 1 | 0 | 0 | 1 |
| | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 0 |
| 22 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | −1 | −1 | 0 | 1 | 0 | 0 | −1 | 0 | 1 | −1 | 0 | 0 | −1 | 1 | 0 | 1 | 0 | 0 | −1 |
| | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 0 |
| 23 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | −1 | 0 | 0 | 1 | 1 | 1 | −1 | 0 | 0 | 0 | 0 | 1 | −1 | 1 | 1 | 0 | 0 | 1 | −1 | 0 | 0 | 0 | 0 | 1 | 1 |
| | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | −1 | −1 | 1 | 1 | 0 | 0 | 0 | 0 | −1 | −1 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 |
| 24 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | −1 | −1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | −1 | −1 | 1 | −1 | 1 | −1 | 0 | 0 | 0 | 0 |
| | 0 | 0 | 0 | 0 | 1 | 1 | 1 | −1 | 1 | 1 | 1 | −1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | −1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | −1 |
| 25 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | −1 | 1 | −1 | 1 | −1 | 1 | −1 |
| | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | −1 | 1 | −1 | −1 | −1 | 1 | 1 | 1 | −1 | 1 | −1 | 1 | −1 | −1 | −1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 26 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | −1 | 0 | −1 | 0 | 1 | 0 | 1 | 0 | −1 | 0 | −1 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | |
| | 0 | 1 | 0 | 1 | 0 | 1 | 0 | −1 | 1 | 0 | 1 | 0 | 1 | 0 | −1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | −1 | 0 | 0 | −1 | 0 | −1 | 0 | −1 | 0 | 1 | |

Table A.1 – continued from previous page

| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 27 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | −1 | −1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | −1 | −1 | 1 |
| | 0 | 0 | 0 | 0 | 1 | 1 | 1 | −1 | 1 | 1 | 1 | −1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | −1 | −1 | −1 | 1 | −1 | −1 | −1 | 0 | 0 | 0 | 0 |
| 28 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | −1 | −1 | 0 | 0 | 1 | 1 | 0 | 0 | −1 | −1 | 0 | 0 | −1 | −1 | 0 | 0 | −1 | −1 | 0 |
| | 1 | 0 | 0 | 1 | 1 | 0 | 0 | −1 | 1 | 0 | 0 | −1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | −1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | −1 |
| 29 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | −1 | 1 | 0 | 0 | −1 | 0 | −1 | −1 | 0 | 1 | 0 | 0 | −1 | 0 | −1 | −1 | 0 | 0 | −1 | −1 | 0 | −1 | 0 | 0 | 1 |
| | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 0 |

*Note*: Does not include coefficients $c_\alpha$

# Appendix B

# Algorithms

Next, we include the following algorithm because it provided a fast approximation to upper-bound the stabilizer rank for any state $|\psi\rangle$, regardless of having real or imaginary components. The premise is to start with the worst-case stabilizer decomposition of $|\psi\rangle$ in the computational basis, $k = 2^n$, and then work to reduce the rank by combining stabilizer states from the decomposition. We iterate through all $\binom{k}{2}$ stabilizer states from the decomposition until we find one that (a) is still a stabilizer state when summed and (b) still decomposes $|\psi\rangle$. We repeat this process until $k$ can no longer be reduced.

**Algorithm 2** Stabilizer Bounded Rank Decomposition Search

**Require:** $n > 0$, $|\psi\rangle$
  $A \leftarrow Identity(2^n)$
  $candidates \leftarrow [A]$
  $best \leftarrow A$
  **while** $len(candidates) > 0$ **do**
    $A \leftarrow candidates.pop()$

```
    for combo in (A.cols choose 2) do
        combined ← normalize(sum(combo))
        if !is_stabilizer_state(combined) then
            continue
        end if
        B ← remaining_columns(A, combo)
        x ← lstsq_solve(Bx = |ψ⟩)
        if Bx = |ψ⟩ then
            C ← B * x              (* is element-wise multiplication)
            if len(C.cols) < len(best.cols) then
                best ← C
                candidates ← candidates + C
                break
            end if
        end if
    end for
end while
normalize(best.cols)
return best
```

# Bibliography


[1] Gabriele Nebe, E. M. Rains, and N. J. A. Sloane. The invariants of the Clifford groups, 2000. arXiv math/0001038.

[2] Daniel Gottesman. The Heisenberg representation of quantum computers, 1998. arXiv quant-ph/9807006.

[3] Sergey Bravyi, Graeme Smith, and John A. Smolin. Trading classical and quantum computational resources. *Phys. Rev. X*, 6(2), 2016.

[4] D. Aharonov, W. van Dam, J. Kempe, Z. Landau, S. Lloyd, and O. Regev. Adiabatic Quantum Computation is Equivalent to Standard Quantum Computation. In *45th Annual IEEE Symposium on Foundations of Computer Science*. IEEE, 2004.

[5] A. B. Finnila, M. A. Gomez, C. Sebenik, C. Stenson, and J. D. Doll. Quantum Annealing: A New Method for Minimizing Multidimensional Functions. *Chem. Phys. Lett.*, 219(5-6):343–348, 1994.

[6] Robert Raußendorf. Measurement-based quantum computation with cluster states. *Int. J. Quantum Inf.*, 07(06):1053–1203, 2009.

[7] Scott Aaronson and Alex Arkhipov. The Computational Complexity of Linear Optics. In *Proceedings of the forty-third annual ACM symposium on Theory of computing*, New York, NY, USA, 2011. ACM.

[8] Seth Lloyd and Samuel L. Braunstein. Quantum Computation over Continuous Variables. *Phys. Rev. Lett.*, 82(8):1784–1787, 1999.

[9] Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information: 10th Anniversary Edition.* Cambridge University Press, Cambridge, England, 2012.

[10] Chetan Nayak, Steven H. Simon, Ady Stern, Michael Freedman, and Sankar Das Sarma. Non-Abelian Anyons and Topological Quantum Computation. *Rev. Mod. Phys.*, 80(3):1083–1159, 2008.

[11] Abel Molina and John Watrous. Revisiting the simulation of quantum turing machines by quantum circuits. *Proc. Math. Phys. Eng. Sci.*, 475(2226):20180767, 2019.

[12] Phillip Kaye, Raymond Laflamme, and Michele Mosca. *An Introduction to Quantum Computing.* Oxford University Press, London, England, 2006.

[13] N. David Mermin. *Quantum Computer Science: An Introduction.* Cambridge University Press, Cambridge, England, 2012.

[14] A. Yu Kitaev. Quantum computations: algorithms and error correction. *Russ. Math. Surv.*, 52(6):1191–1249, 1997.

[15] Neil J. Ross and Peter Selinger. Optimal ancilla-free Clifford+T approximation of z-rotations. *Quantum Inf. Comput.*, 16(11&12):901–953, 2016.

[16] Yaoyun Shi. Both Toffoli and Controlled-NOT need little help to do universal quantum computation, 2002. arXiv quant-ph/0205115.

[17] Dorit Aharonov and Michael Ben-Or. Fault-Tolerant Quantum Computation with Constant Error Rate. *SIAM J. Comput.*, 38(4):1207–1282, 2008.

[18] A. Yu Kitaev. Fault-tolerant quantum computation by anyons. *Ann. Phys. (N. Y.)*, 303(1):2–30, 2003.

[19] Emanuel Knill, Raymond Laflamme, and Wojciech H Zurek. Resilient quantum computation. *Science*, 279(5349):342–345, 1998.

[20] P. W. Shor. Fault-tolerant quantum computation. In *Proceedings of 37th Conference on Foundations of Computer Science*. IEEE Comput. Soc. Press, 1996.

[21] Frank Arute, Kunal Arya, Ryan Babbush, Dave Bacon, Joseph C Bardin, Rami Barends, Rupak Biswas, Sergio Boixo, Fernando G S L Brandao, David A Buell, Brian Burkett, Yu Chen, Zijun Chen, Ben Chiaro, Roberto Collins, William Courtney, Andrew Dunsworth, Edward Farhi, Brooks Foxen, Austin Fowler, Craig Gidney, Marissa Giustina, Rob Graff, Keith Guerin, Steve Habegger, Matthew P Harrigan, Michael J Hartmann, Alan Ho, Markus Hoffmann, Trent Huang, Travis S Humble, Sergei V Isakov, Evan Jeffrey, Zhang Jiang, Dvir Kafri, Kostyantyn Kechedzhi, Julian Kelly, Paul V Klimov, Sergey Knysh, Alexander Korotkov, Fedor Kostritsa, David Landhuis, Mike Lindmark, Erik Lucero, Dmitry Lyakh, Salvatore Mandrà, Jarrod R McClean, Matthew McEwen, Anthony Megrant, Xiao Mi, Kristel Michielsen, Masoud Mohseni, Josh Mutus,

Ofer Naaman, Matthew Neeley, Charles Neill, Murphy Yuezhen Niu, Eric Ostby, Andre Petukhov, John C Platt, Chris Quintana, Eleanor G Rieffel, Pedram Roushan, Nicholas C Rubin, Daniel Sank, Kevin J Satzinger, Vadim Smelyanskiy, Kevin J Sung, Matthew D Trevithick, Amit Vainsencher, Benjamin Villalonga, Theodore White, Z Jamie Yao, Ping Yeh, Adam Zalcman, Hartmut Neven, and John M Martinis. Quantum supremacy using a programmable superconducting processor. *Nature*, 574(7779):505–510, 2019.

[22] Yongshan Ding and Frederic T. Chong. *Quantum computer systems: Research for noisy intermediate-scale quantum computers*. Springer International Publishing, Cham, Switzerland, 2020.

[23] Scott Aaronson and Daniel Gottesman. Improved simulation of stabilizer circuits. *Phys. Rev. A*, 70(5), 2004.

[24] Craig Gidney. Stim: a fast stabilizer circuit simulator. *Quantum*, 5:497, 2021.

[25] Daniel Gottesman. Theory of fault-tolerant quantum computation. *Phys. Rev. A*, 57(1):127–137, 1998.

[26] Xinlan Zhou, Debbie W. Leung, and Isaac L. Chuang. Methodology for quantum logic gate constructions, 2000. arXiv quant-ph/0002039.

[27] Daniel Gottesman and Isaac L. Chuang. Demonstrating the viability of universal quantum computation using teleportation and single-qubit operations. *Nature*, 402(6760):390–393, 1999.

[28] Michael Vasmer and Aleksander Kubica. Morphing quantum codes. *PRX Quantum*, 3(3), 2022.

[29] Andrew M. Steane. Efficient fault-tolerant quantum computing. *Nature*, 399(6732):124–126, 1999.

[30] Sergey Bravyi and Jeongwan Haah. Magic state distillation with low overhead, 2012. arXiv 1209.2426.

[31] Sergey Bravyi, Dan Browne, Padraic Calpin, Earl Campbell, David Gosset, and Mark Howard. Simulation of quantum circuits by low-rank stabilizer decompositions. *Quantum*, 3:181, 2019.

[32] Michael Beverland, Earl Campbell, Mark Howard, and Vadym Kliuchnikov. Lower bounds on the non-Clifford resources for quantum computations. *Quantum Sci. Technol.*, 5(3):035009, 2020.

[33] Michael Vasmer and Dan E. Browne. Three-dimensional surface codes: Transversal gates and fault-tolerant architectures. *Phys. Rev. A (Coll. Park.)*, 100(1), 2019.

[34] Shir Peleg, Amir Shpilka, and Ben Lee Volk. Lower bounds on stabilizer rank. *Quantum*, 6:652, 2022.

[35] Hammam Qassim, Hakop Pashayan, and David Gosset. Improved upper bounds on the stabilizer rank of magic states. *Quantum*, 5(606):606, 2021.

[36] Earl T. Campbell and Mark Howard. Magic state parity-checker with pre-distilled components. *Quantum*, 2:56, 2018.

[37] Earl T. Campbell and Mark Howard. Unified framework for magic state distillation and multiqubit gate synthesis with reduced resource cost. *Phys. Rev. A (Coll. Park.)*, 95(2), 2017.

[38] Peter Selinger. Quantum circuits of $T$-depth one. *Phys. Rev. A*, 87:042302, Apr 2013.

[39] David Gosset, Vadym Kliuchnikov, Michele Mosca, and Vincent Russo. An algorithm for the T-count. *Quantum Information & Computation*, 14(15–16):1261–1276, Nov 2014.

[40] Mark Howard and Earl Campbell. Application of a resource theory for magic states to fault-tolerant quantum computing. *Phys. Rev. Lett.*, 118(9), 2017.

[41] Héctor J. García, Igor L. Markov, and Andrew W. Cross. On the Geometry of Stabilizer States. *Quantum Info. Comput.*, 14(7 & 8):683–720, May 2014.

[42] Matthew Amy, Jianxin Chen, and Neil J. Ross. A finite presentation of CNOT-dihedral operators. *Electron. Proc. Theor. Comput. Sci.*, 266:84–97, 2018.

[43] Noga Alon. Transversal numbers of uniform hypergraphs. *Graphs and Combinatorics*, 6:1–4, 1990.

[44] David Gosset, 2023. personal communication.

[45] Zsolt Tuza. Covering all cliques of a graph. *Discrete Math.*, 86(1–3):117–126, jan 1991.