Efficient Reduction of Resources and Noise in Discrete Quantum Computing Circuits

A thesis submitted for the degree of

Doctor of Philosophy

by

Ritajit Majumdar

Advisor

Prof. Susmita Sur-Kolay

Advanced Computing and Microelectronics Unit
Indian Statistical Institute
203 B. T. Road, Kolkata - 700108

January, 2024

Ekam Sat Vipra Bahudha Vadanti - Rig Veda

The truth is One, the wise perceive it differently

Acknowledgements

The journey of PhD can never be completed alone. Although it is the student who strides through the path, he is supported directly or indirectly by many. I would like to take this opportunity to express my gratitude to all whose support has shaped my Ph.D. career.

First and foremost, I express my sincere gratitude to my supervisor, Prof Susmita Sur-Kolay, whose unwavering support has been instrumental in shaping my PhD career. She has always been supportive through my ups and downs, and her invaluable inputs have always improved the quality of my research. Her encouragement, motivation, and constructive criticism have propelled me towards achieving my goals. I am thankful to all the faculties of ACMU, and ISI in general, whose course-works have significantly increased my knowledge of the subject. I would especially like to thank Prof. Guruprasad Kar who gave me the first opportunity to learn about quantum computation and has always encouraged me since my bachelor days to become a good researcher.

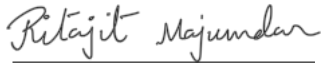
I am thankful to Amit da, Saikat da, and Nayana with whom I collaborated on a few studies. I am thankful to the Quantum Computing team at IBM India Research Lab who were my collaborators on multiple projects. The insights of Dhiraj have largely improved the quality of the publications coming out of this collaboration. The extensive knowledge of Dhinakaran and Shesha has also been crucial in shaping the project outcomes.

The Fulbright tenure has been a pivotal time in my PhD career. I thank Dr. Rajiv Joshi for helping me find a mentor at the IBM Watson Lab. I am thankful to Prof. Utpal Garain for helping me out with the PhD registration which was urgently required for the Fulbright application. My sincere gratitude to the then Dean of Studies, Prof. Debasis Sengupta, for smoothly processing the Joint Study Agreement between ISI and IBM Watson Lab. I am grateful to Pratibha Nair, USIEF, for encouraging me on the Fulbright tenure irrespective of certain issues with the fellowship amount.

I extend my gratitude to Chris Wood, my mentor at IBM Watson lab, for believing in me while I was struggling in the first few months. I thank him for helping me out in every step of the research and making time to continue the study even after my return. I am extremely grateful to Zlatko Minev, who used to take time off his busy schedule to have regular academic conversations with me and believed in me enough to later offer me a job.

Apart from my academic peers, I am thankful to my mother for her support during this period, and for being patient with me during those silent days when nothing worked out in research. This tenure would not have been a success without her being by my side, yet repeatedly reminding me that she has no idea what I really do. I am thankful to my lab mates – Kritanta, and Debasmita – for the wonderful time we had together. I am extremely grateful to Debasmita for often helping me maintain a work-life balance during this tenure, sometimes going out together to explore restaurants and cafes. Debasmita would also help me with my numerous extra-curricular ideas such as a being joint mentor at QIntern, mentoring undergraduate students with projects, and organizing the Qiskit Fall Fest 2022.

To all the individuals mentioned above, and to those whose names may have inadvertently been left out, I extend my sincere gratitude for the profound impact you have had on my academic and personal growth. It is through the collective support of these exceptional individuals that I stand before you today, presenting this thesis.



Ritajit Majumdar

Abstract

Quantum computers have shifted from a subject of theoretical interest to reality in recent years with multiple devices now available at research industry labs such as IBM, Google, and IonQ. However, quantum systems are highly susceptible to noise. Interaction with the environment corrupts the information content, leading to unreliable computation.

Near-term quantum computers do not have sufficient qubits to incorporate error correction. Therefore, other mechanisms are studied to lower the effect of noise. Quantum Approximate Optimization Algorithm (QAOA) is an algorithm family for finding approximate solutions to combinatorial optimization problems. Any such problem can be represented as a graph $G = (V, E)$, and the number of 2-qubit gates in the corresponding circuit scales linearly with the number of edges. 2-qubit gates are one of the noisiest components in current hardware. This thesis proposes three hardware-independent algorithms to lower the number of 2-qubit gates while ensuring functional equivalence. The first algorithm, based on Edge Coloring, eliminates up to $\lfloor \frac{|E|}{2} \rfloor$ gates while retaining the original depth of the circuit. The second algorithm, based on Depth First Search (DFS), eliminates $|V| - 1$ 2-qubit gates, which is shown to be optimal while increasing the depth of the circuit to some extent. The third heuristic algorithm retains the $|V| - 1$ 2-qubit gates elimination, yet restricts the increase in the circuit depth - thus yielding the best performance of the three. Finally, the heuristic is modified to respect the underlying hardware architecture and lower the number of SWAP gates.

Another method, called circuit cutting, where a circuit is partitioned into multiple smaller subcircuits, is shown to effectively lower noise. The subcircuits are computed individually, and the outcome is constructed on a classical computer. This thesis proposes two error mitigation techniques, targeted particularly for circuit cutting, which significantly improve the fidelity of the outcome.

These techniques for lowering the effect of noise are not sufficient for arbitrary long quantum computation; there quantum error correction (QECC) is mandated.

Quantum computers are inherently multi-valued, and accessing higher dimensions allows storage of more information using fewer qubits. The second part of this thesis shows the challenges of designing a ternary QECC from its binary counterpart. Naive efforts require two-step error correction, leading to a significant increase in the gate cost of the QECC circuit. Next, a necessary condition for stabilizer formulation is provided which allows easy carry-over of binary QECC to ternary, making error correction a single step.

For near-term devices, the decomposition of a 3-qubit Toffoli gate by temporary access to higher dimensions is shown to provide an exponential reduction in the depth of the decomposed circuit. The final chapter studies whether this decomposition method is still beneficial when augmented with error correction and concatenation. An analytical criterion is provided for which the resource requirement of error-corrected qutrit-assisted decomposition remains lower than the qubit-only decomposition.

Overall, this thesis contributes both to near-term and long-term quantum computing. The findings from Part I provide useful methods to improve the performance of algorithms in current quantum devices. Part II gives valuable insight into the challenges of incorporating near-term methods in conjunction with error correction and the design of higher dimensional QECC from binary codes.

List of publications related to the thesis

The thesis is based on the following publications of the author.

In Journals

1. Majumdar, R., Basu, S., Ghosh, S., & Sur-Kolay, S. (2018). Quantum error-correcting code for ternary logic. Physical Review A, 97(5), 052302.

2. Majumdar, R., & Sur-Kolay, S. (2023). Designing Ternary Quantum Error Correcting Codes from Binary Codes. Journal of Multiple-Valued Logic & Soft Computing, 40.

3. Majumdar, R., Saha, A., Chakrabarti, A., & Sur-Kolay, S. (2023). Intermediate Qutrit-assisted Multi-controlled Gate Decomposition with Quantum Error Correction. Quantum Information Processing 23, 42 (2024).

4. *(Submitted)* Majumdar, R., & Wood, C. J. (2022). Error mitigated quantum circuit cutting. arXiv preprint arXiv:2211.13431.

In Peer-reviewed International Conference Proceedings

1. Majumdar, R., & Sur-Kolay, S. (2020, November). Approximate ternary quantum error correcting code with low circuit cost. In 2020 IEEE 50th International Symposium on Multiple-Valued Logic (ISMVL) (pp. 34-39). IEEE.

2. Majumdar, R., & Sur-Kolay, S. (2020, October). Special session: Quantum error correction in near term systems. In 2020 IEEE 38th International Conference on Computer Design (ICCD) (pp. 9-12). IEEE.

3. *(Accepted)* Majumdar, R., Madan, D., Bhoumik, D., Vinayagamurthy, D., Raghunathan, S., & Sur-Kolay, S. (2023). Optimized QAOA ansatz design for two-body Hamiltonian problems. 37th International Conference on VLSI Design.

Preprints

1. Majumdar, R., Madan, D., Bhoumik, D., Vinayagamurthy, D., Raghunathan, S., & Sur-Kolay, S. (2021). Optimizing ansatz design in qaoa for max-cut. arXiv preprint arXiv:2106.02812.

2. Majumdar, R., Bhoumik, D., Madan, D., Vinayagamurthy, D., Raghunathan, S., & Sur-Kolay, S. (2021). Depth optimized ansatz circuit in QAOA for Max-Cut. arXiv preprint arXiv:2110.04637.

Table of Contents

List of Figures

List of Tables

CHAPTER 1

Introduction and the scope of the thesis

Contents

1.1 Introduction

The proposition to use a different kind of computation for the simulation of nature was first proposed by Richard Feynman [F+18]. He asserted that nature is

not classical, and therefore the computers that we have today, termed as *classical computers*, are not sufficient to faithfully simulate it. The number of parameters increases exponentially with the increment in the number of particles in the system. Therefore, simulation of nature largely falls short even with current supercomputers. This leads to approximate methods of simulation, which are used nowadays for most problems of interest.

Precise simulation of nature not only holds theoretical interest but also has applications in domains of practical importance such as chemistry, genetics, drug discovery, transportation, finance, etc. Approximate solutions used presently in problems from these domains put a limit on the quality of the result obtainable. Feynman envisioned that since nature is inherently quantum, computers that make explicit use of quantum mechanical principles called *quantum computers*, can aid in such scenarios.

A quantum computer is often defined as a device that follows the laws of quantum mechanics. However, this is a misnomer. Nature is inherently quantum, and every device in use eventually consists of quantum particles, thus obeying the laws of quantum mechanics. However, the computers used today are predominantly not quantum computers. So, what makes a computer *quantum*? A computer can be called a *quantum computer* if it makes explicit use of certain quantum mechanical phenomena such as superposition, interference, and entanglement. These phenomena are not observed in the macroscopic scale, and hence cannot be exploited by almost all the current computing devices. The following subsection provides a brief introduction to these three principles.

1.2 Quantum computer and its properties

A quantum state, in Dirac notation $|\psi\rangle$, satisfies the Schrödinger equation shown in Eq. (1.1). Here H is called the Hamiltonian which corresponds to the total energy of the system. Experimentally, such a quantum state can be realized using superconductors, trapped ions, photons, or other entities. This entire thesis will

not be concerned with the exact experimental implementation of a quantum state. Rather, it shall always be treated in an abstract mathematical way where $|\psi\rangle$ is represented as a column vector with its norm as 1 [NC02].

$$i\hbar\frac{d\,|\psi\rangle}{dt} = H\,|\psi\rangle \tag{1.1}$$

To adhere to the notation of bits generally used in computer science, the following representation is used in the literature

$$|0\rangle = \begin{pmatrix}1 & 0\end{pmatrix}^T \quad |1\rangle = \begin{pmatrix}0 & 1\end{pmatrix}^T$$

These two states correspond to the bits 0 and 1 in electronic systems and are called quantum bits or *qubit*. Qubits form the building blocks of quantum computers. Next, let us review the relevant properties of qubits that are not observed in the macroscopic world.

1. **Superposition**: As $|0\rangle$ and $|1\rangle$ are valid qubits, by linearity of the Schrödinger equation any state of the form $|\Psi\rangle = \alpha\,|0\rangle + e^{i\phi}\beta\,|1\rangle$, $\alpha, \beta \in \mathbb{C}$, $\phi \in \mathbb{R}$, is also a valid qubit. Such a state is said to be in a *superposition* of $|0\rangle$ and $|1\rangle$. A general qubit can be represented by a Bloch Sphere (refer to Fig. 1.1), where the two poles $|0\rangle$ and $|1\rangle$, correspond to the classical counterparts of 0 and 1. Any state within this sphere is a valid qubit.

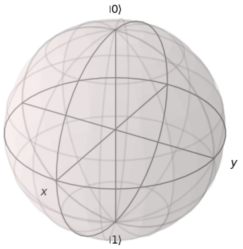


Figure 1.1: A Bloch sphere representation of a qubit. While valid classical bits correspond to the two poles only, any state within this sphere is a valid qubit

Measurement plays a big role in quantum systems. While it is possible to evolve the system in superposition, it collapses to $|0\rangle$ with probability $|\alpha|^2$ or $|1\rangle$ with probability $|\beta|^2$ when the state $|\psi\rangle$ is measured.

2. **Interference**: Quantum mechanical systems have wave-particle duality, i.e., they exhibit the properties of both waves and particles. Therefore, when two qubits interact, they can do so constructively or destructively by virtue of their wave nature. Since the measurement of a qubit yields a probabilistic outcome, this can cause a hindrance to computing with such a system. The attempt of any quantum algorithm is to interfere these superposition states such that the state encoding the desired solution interferes constructively, while the others interfere destructively, thus increasing the probability of obtaining the correct outcome. In other words, proper use of constructive and destructive interference is a necessity for quantum computing systems.

3. **Entanglement**: Entanglement is a property where two qubits are so correlated that the measurement of one qubit affects the state of the other, irrespective of the physical distance between them. This phenomenon, termed *spooky action at a distance* by Einstein, has applications in quantum communication, cryptography, and computing.

Note that quantum computing can broadly be classified into discrete and continuous. When the computation deals with variables and observables that take discrete values, it is said to be discrete variable quantum computation. On the other hand, certain systems deal with variables and observables that can take values from a continuous domain. Computation over such variables is termed continuous quantum computing. This thesis deals only with discrete-variable quantum computers For the rest of this thesis, any reference to *quantum computing* shall imply discrete-variable quantum computing.

1.3 Multi-valued quantum computing

As mentioned before, every quantum state is a unit vector in a Hilbert Space. In general, this Hilbert space is infinite-dimensional, going from $|0\rangle$, $|1\rangle$, $|2\rangle$, ... up to infinity. The difference $\Delta E_{i,i+1}$ in the energy of two consecutive states $|i\rangle$ and $|i+1\rangle$ diminishes with increasing i, and the spectrum becomes continuous in the asymptotic limit. In other words, a quantum state is inherently multi-valued. The usage of higher-dimensional quantum systems comes with an increase in search space. For example, the information content of an n-qubit quantum system can be stored in $\lceil \frac{n}{log_2 d} \rceil$ d-dimensional quantum states. This increase in search space, or equivalently the reduction in the number of quantum states, often leads to performance enhancement in applications such as quantum cryptography, search algorithms, decomposition of quantum gates, etc.

Nevertheless, it is an engineering challenge to deal with qudits in the laboratory. This is primarily because the energy difference $\Delta E_{i,i+1}$ reduces to zero as the dimension d increases. However, several studies have now shown experimental realization of 3-dimensional, or ternary quantum systems in superconductor [GCK+21] and ion-trap [Low23] devices. Furthermore, pulse simulation in IBM Quantum devices allows access to ternary quantum states. A ternary quantum system, called a *qutrit*, has the form, $|\psi\rangle = \alpha|0\rangle + e^{i\phi_1}\beta|1\rangle + e^{i\phi_2}\gamma|2\rangle$, $\phi_1, \phi_2 \in \mathbb{R}$, where $|\alpha|^2 + |\beta|^2 + |\gamma|^2 = 1$ [NC02].

1.4 Challenges of quantum computing

Asher Peres once famously stated: “Quantum phenomena do not occur in a Hilbert space. They occur in a laboratory." [Per97]. In other words, the theoretical studies on quantum computing assume the existence of one or more ideal qubits in isolation from the environment. However, in practice, this is impossible to achieve. Qubits interact with the environment, leading to errors in the computation. Imperfection

in the preparation of qubits, gate operations, measurement, and the natural tendency of a qubit to spontaneously release energy and settle in its ground state are some of the usual sources of noise or error in a quantum system. Fig. 1.2 shows an estimate of the error probability on a 5-qubit IBM Quantum device.

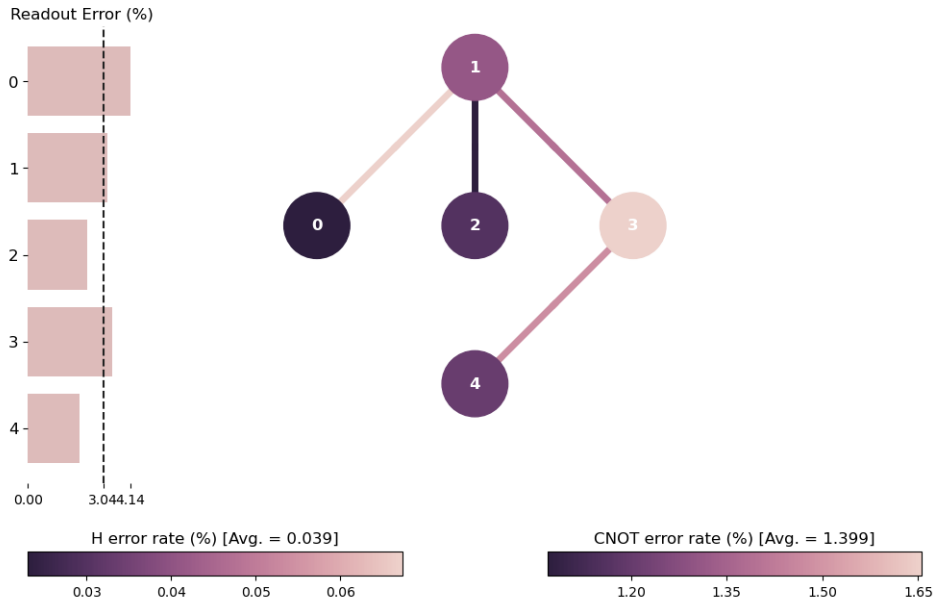


Figure 1.2: The noise profile of a 5-qubit IBM quantum device

Error, or noise, is thus the primary challenge in designing large-scale quantum computers capable of performing arbitrarily long and meaningful computations.

The goal of achieving reliable quantum computing naturally requires error correction and fault tolerance. However, the general working phenomenon of these methods is to encode the information of a single logical qubit into multiple physical qubits in order to protect it from noise. It has been estimated that tens of thousands of qubits may be necessary to achieve fault tolerance. Currently, the largest hardware from IBM has 433 qubits, with the plan to scale up to a thousand within a year. This is insufficient for error correction and fault tolerance. Algorithms for integer factorization, database search, phase estimation, etc. have deep circuits requiring multiple qubits. Therefore, such algorithms are not suitable for execution in current quantum computers.

This has led to rigorous studies on interim methods to lower the effect of noise and have useful quantum computation. Recently IBM has been able to show reliable computation of a 2D Ising model with more than 100 qubits using such error mitigation methods [KE+23].

1.5 Motivation and scope of this thesis

The primary motivation of this thesis is to study various techniques to improve the quality of computation under noise. While in the near term, this implies devising methods to reduce the effect of noise, the long-term goal is to implement error correction and fault tolerance. This thesis investigates both of these aspects in two parts.

1.5.1 Near-term quantum computing

Part I of this thesis comprises methods to improve the quality of computation in the absence of error correction. Hybrid quantum-classical algorithms have been developed for this era, in particular, to divide the workload between quantum hardware and classical hardware. This leads to lower qubit requirement, and lower depth of the quantum circuit, which suits the current absence of error correction. From a set of possible hybrid quantum-classical algorithms, this thesis focuses on *Quantum approximate optimization algorithms* (QAOA) which are used to find approximate solutions to combinatorial optimization problems. Efficient circuit synthesis methods for these algorithms are proposed to eliminate a significant number of noisy gates, leading to increased fidelity.

Another class of problems studied widely for near-term quantum computing is the trotterized circuit used for condensed matter physics such as the computation of ground state energy for Ising models. These circuits have a very symmetric structure, but can quickly become pretty dense. Circuit cutting is a method used to

partition a circuit into multiple smaller subcircuits such that each subcircuit can be executed independently. The final outcome is calculated by classical postprocessing over the outcomes of each subcircuit. This thesis explores the ability of circuit cutting to lower noise (since each subcircuit has a smaller number of qubits and/or gates), and proposes error mitigation methods specific to circuit cutting, to obtain significantly higher fidelity for trotterized circuits.

1.5.2 Error corrected quantum computing

Part II of this thesis focuses on error correction and fault tolerance for the design of quantum computers capable of performing arbitrary long computations. In particular, this part studies ternary quantum systems and discusses the challenges of circuit synthesis for ternary quantum error correcting codes (QECC). The salient questions are whether (i) it is non-trivial to carry over QECCs designed for binary quantum systems to the ternary systems without significantly increasing the cost of the QECC circuit, and (ii) there are any necessary conditions for the design of ternary QECCs such that they can be designed efficiently as an extension from known binary QECCs. This study concentrates on general stabilizer-based concatenation QECCs. Furthermore, certain methods for the decomposition of multi-qubit gates for near-term quantum computing, using intermediate ternary states, to obtain a reduction in circuit depth are also taken up. An in-depth analysis of the challenges of extending such methods to error correction, and an estimate of the resource required for the same would be useful.

1.6 Contributions and Organization of the thesis

The chapter-wise contributions of the thesis are listed below.

- Chapter 1 covers the basic principles and challenges of quantum computing and the scope of the thesis.

- Chapter 2 provides a review of the related works such as QAOA, circuit cutting, and the basic background of quantum error correction.

- **Part I**: Error suppression and mitigation for near-term quantum computation

  - Chapter 3 proposes two deterministic classical algorithms for the elimination of as many 2-qubit gates in a QAOA circuit corresponding to a graph $G = (V, E)$ as possible. The first algorithm based on edge coloring (EC) eliminates up to $\lfloor \frac{|V|}{2} \rfloor$ gates, without hampering the depth of the circuit. The second one based on depth first search (DFS) eliminates $|V| - 1$ 2-qubit gates, which is shown to be optimal, but with a moderate increase in the depth of the circuit.

  - Chapter 4 builds on the previous DFS-based algorithm by proposing a heuristic method to retain the optimal 2-qubit gate elimination while restraining the increase in the depth of the circuit. These three algorithms are hardware-independent. This chapter also proposes the modifications required to make the heuristic method more amenable to hardware constraints.

  - Chapter 5 shows the formulation of tomographic circuit cutting and proposes two error mitigation methods.

- **Part II**: Error correction for reliable quantum computation

  - Chapter 6 demonstrates the challenges of designing a 9-qutrit QECC for ternary quantum systems as a carry-over of the 9-qubit QECC. It enunciates that the resource requirement necessarily increases when designing the ternary QECC. Next, attempts to reduce the resource requirement are made by designing a 6-qutrit approximate QECC.

  - Chapter 7 presents the root cause for the increase in resources for the design of ternary QECC and provides a necessary condition for the carry-over of binary QECCs to ternary quantum systems without increasing the resources required.

  - Chapter 8 looks into the decomposition of multi-qubit gates using intermediate ternary states, proposed primarily for near-term quantum computers. In particular, analytical estimates and necessary conditions are derived for using this decomposition method together with quantum error correction.

- Chapter 9 summarizes the contributions of this thesis and discusses potential directions for future research.

CHAPTER 2

Background and Related works

Contents

2.1 Introduction

Quantum computing has come a long way since it was initially proposed by Feynman – from a theoretical notion to practical realization. The major hurdle for quantum computing has always been noise. Several studies have been performed to characterize, control, mitigate, and correct noise in quantum systems. These efforts have led to a new type of algorithm, called hybrid quantum-classical algorithms which are less susceptible to noise as these deal with low-depth quantum circuits. In the absence of error correction, which is an engineering challenge now, interim methods of error mitigation have been proposed to reduce the effect of noise. Circuit cutting has come up as a method to lower the effect of noise at the cost of some classical post-processing. In all these efforts, the final aim to be achieved is always error correction and fault tolerance – without which arbitrary long quantum computing is not possible.

This chapter first gives a broad overview of a few well-known types of noise which affect quantum systems. Next, it presents a background on QAOA, which is a hybrid quantum-classical algorithm and has been studied in this thesis from the circuit synthesis perspective. Then, a brief discussion involves general ideas of circuit cutting and error mitigation. Finally, an overview of error correction, in particular stabilizer QECC, and the necessary conditions for resource reduction are reviewed.

2.2 Noise in quantum systems

The imperfections of the laboratories and the interaction of qubits with the environment come into play for practical quantum computing. When the system interacts with the environment, the system-environment pair evolves unitarily.

However, this does not mandate that the evolution of the system alone is unitary as well. The most general evolution of an open quantum system can be described by the Kraus operators or the Lindblad Master Equation [NC02]. But a model, for which the evolution of the noisy system itself is unitary, has a much simpler representation since it can be denoted by a linear combination of the Pauli matrices (or operators) [NC02] I, X, Z and Y whose matrix forms are shown below:

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

Three main types of noise, where the evolution of the quantum state ρ is unitary, are as follows:

1. **Depolarization noise** [NC02]: The state ρ evolves to a maximally mixed state by some probability of error p and remains unchanged by probability $(1-p)$ (eq. (2.1)). This is equivalent to the scenario where each of the Pauli operators X, Y and Z occurs with equal probability.

$$\rho \to (1-p)\rho + p\frac{\mathbb{I}}{2}. \tag{2.1}$$

2. **Stochastic Pauli noise** [NC02]: This model is similar to the depolarization model, except that the probability of the Pauli operators X, Y, and Z are not necessarily the same. (eq. (2.2)).

$$\rho \to (1-p_x-p_y-p_z)\rho + p_x X\rho X + p_y Y\rho Y + p_z Z\rho Z. \tag{2.2}$$

   This simple change from the depolarization noise model can often lead to a significant effect on the circuit. A general quantum circuit consists of gates from both the Clifford group and the non-Clifford set of gates. Clifford gates map a Pauli operator to another Pauli operator, but this does not hold for non-Clifford gates. Therefore, if each gate in the circuit is suffering from stochastic Pauli noise, the overall noise of the circuit may not have a simple stochastic Pauli form.

3. **State preparation and measurement (SPAM) noise** [BSK+21]: SPAM is one of the primary sources of error in current quantum devices. This type of error captures the scenario where preparing the initial state and/or the final measurement are/is noisy. It is not possible to distinguish between these two types of errors, and hence the combined nomenclature. This model has a simple representation where the state is affected by a Pauli X operator only with probability p (eq. (2.3)).

$$\rho \to (1-p)\rho + pX\rho X \tag{2.3}$$

Note that this does not imply that other Pauli operators cannot act as noise operators on state preparation and measurement. However, the phase occurring from Pauli-Z or Y operators will not affect the measurement outcome.

However, as discussed before, not all noise models are unitary. Some of the models which do not have simple unitary representation are as follows:

4. **Amplitude and phase damping** [NC02]: A quantum state prepared in the excited state has a tendency to spontaneously emit the energy and return to the ground state. Such a noise is characterized by the parameter T_1 which indicates the half-life of such a spontaneous amplitude decay. Furthermore, the different energy states of a qubit in superposition tend to acquire different phases, thus causing a phase difference. The phase damping noise is characterized by the parameter T_2 which indicates the time required for the different energy states to return to their original configuration after acquiring different phases over time. The combined noise model, called amplitude and phase damping, is governed by the three Kraus operators

$$E_1 = \begin{pmatrix} 1 & 0 \\ 0 & \sqrt{(1-p_{AD})(1-p_{PD})} \end{pmatrix} \quad E_2 = \begin{pmatrix} 0 & \sqrt{p_{AD}} \\ 0 & 0 \end{pmatrix}$$

$$E_3 = \begin{pmatrix} 0 & 0 \\ 0 & \sqrt{(1-p_{AD})(1-p_{PD})} \end{pmatrix}$$

where $p_{AD} = exp(-t/T_1)$ and $p_{PD} = exp(-t/T_2)$, t being time.

5. **Crosstalk** [SPR+20]: A major issue in current quantum devices is crosstalk. In a device with H qubits $Q_1, Q_2, \ldots Q_H$, if a subset of the qubits is under operation, then ideally it should not affect the state of the remaining qubits. However, this is not always the case. It is often observed that computation over one qubit affects its neighboring, sometimes even far away, qubits. Such an effect is termed crosstalk. The crosstalk noise between two qubits q_i and q_j is, in general, modelled by an $R_{zz} = CNOT(q_i, q_j)(I \otimes R_z(\phi))CNOT(q_i, q_j)$ operator acting on the two qubits where ϕ is a scalar.

6. **Coherent rotation** [GD17]: Any unitary operator is a valid quantum gate. A quantum gate essentially rotates the qubit within the Hilbert Space. However, due to engineering defects, a rotation by an angle θ is often replaced by $\theta + \Delta\theta$ for a small value of $\Delta\theta$. If a gate G, having a coherent error of $\Delta\theta$ is repeated k times, then the noise accumulates to $k.\Delta\theta$. But since this is an over-rotation, coherent error results in a sinusoidal nature where the resultant state first diverges from the ideal state, and then again converges towards the ideal state as the accumulation of error due to over-rotation increases.

Note that there are several other sources of noise that may affect a quantum system, such as leakage, magnetic flux, etc. It is often difficult, if not impossible, to completely characterize the noise on a quantum system. Therefore, a method called Pauli twirling [WE16] is widely used now. In this method, an n-qubit quantum channel Λ is sandwiched between single-qubit Pauli gates sampled uniformly at random from the n-qubit Pauli group (see Fig. 2.1) such that the effective functionality remains unchanged. Irrespective of the original nature of Λ, the average of multiple occurrences of Pauli twirling results in a stochastic Pauli channel for any Λ.

Pauli twirling is thus a method to convert a complicated quantum channel into a simpler stochastic Pauli channel [VDBMT22a, vdBMKT22]. It also has an important role to play in QECC. Most of the QECCs are designed to correct

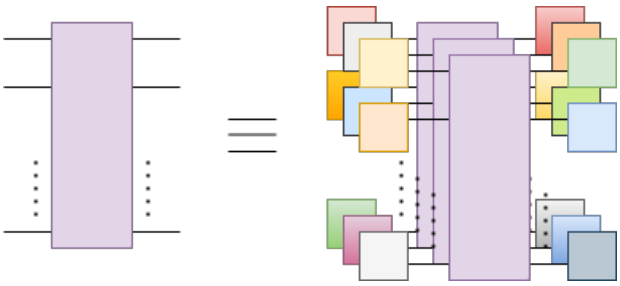


Figure 2.1: An example of Pauli twirling. Here the multi-qubit operator is a quantum channel, and the single qubit gates are Pauli gates sampled uniformly at random from the n-qubit Pauli group.

unitary error, which can always be expressed as a stochastic Pauli noise. Although QECCs specific to other types of noise models such as amplitude damping [GWYZ14, GKW+18] have been designed, these QECCs cannot be generalized to other noise models. However, Pauli twirling over any channel can help to produce an average channel that has a stochastic Pauli nature. Error correction over this average channel is a more generalized scenario to correct for any error on a quantum computer [KG15].

Current quantum devices do not have sufficient qubits for incorporating error correction. This era is often termed as the Noisy Intermediate-Scale Quantum (NISQ) era [Pre18a]. This has led to the design of a new type of algorithm called the hybrid quantum-classical algorithm which is predominantly aimed for NISQ quantum computers. The next section gives an overview of this algorithm, in particular the one relevant to combinatorial optimization.

2.3 Hybrid quantum-classical algorithms

Current quantum circuits need to be shallow with a small number of qubits. This helps the circuit to be less susceptible to noise. In [PMS+14] the authors proposed hybrid quantum-classical algorithms (also called variational algorithms), which was further improved in [MRBAG16]. These algorithms are useful primarily when

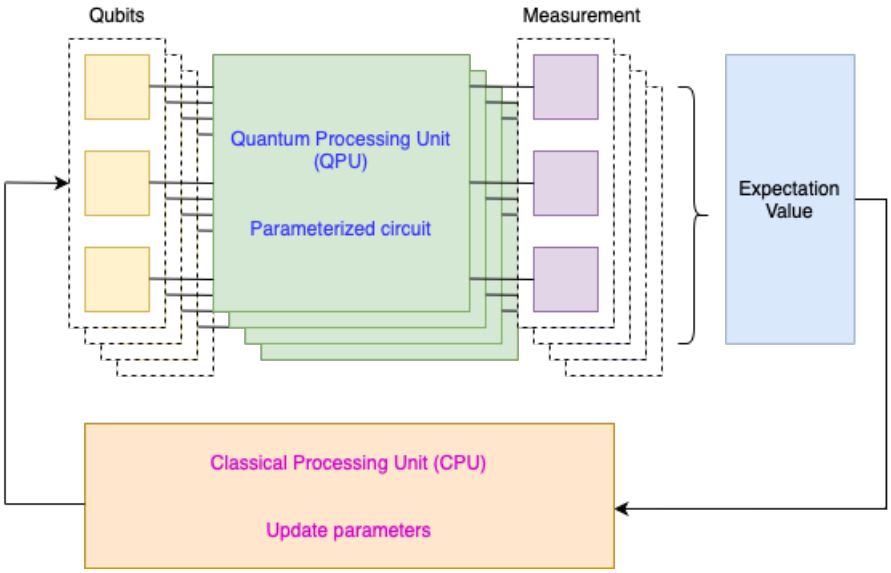


Figure 2.2: A schematic diagram of hybrid quantum-classical algorithms

the required outcome is the expectation value of an observable. The problem is an optimization problem, and hence the requirement is to optimize the expectation value. The entire workload is partitioned into a quantum part and a classical part. In general, there is a parameterized quantum circuit, called *ansatz*, which is executed on a quantum computer multiple times. From these outcomes, the expectation value of an observable is calculated classically. A classical optimizer is used to suggest a new set of parameters with the aim that the next iteration will produce an expectation value closer to the optimal solution. Fig. 2.2 shows a workflow of a hybrid quantum-classical algorithm.

Initial applications of these algorithms were in estimating the ground state energy of molecules [PMS+14, MRBAG16, KMT+17, TSB+21, YABAS20]. In [TCC+22], the authors provide a review of variational algorithms, primarily in the context of quantum chemistry. Later, this technique was shown to have application in quantum machine learning also [SSP15a, SWM+20, SSP15b, ASZ+21]. This thesis looks into the application of these variational algorithms for finding approximate solutions to combinatorial optimization problems, which is termed Quantum approximate optimization algorithm (QAOA). The next subsection provides the

general working principle and some background study on QAOA.

2.3.1 Quantum approximate optimization algorithm (QAOA)

QAOA is a hybrid quantum-classical algorithm, initially proposed by Farhi et al. [FGG14], to find approximate solutions to a combinatorial optimization problem. The idea of QAOA arises from Adiabatic Quantum Computing (AQC) [FGGS00]. Here, the goal is to find the ground state of an operator called the problem Hamiltonian H_P. For example, the problem Hamiltonian may encode some NP-Hard problems such as the Travelling Salesman [RMX+20], Vertex Coloring [CEB20], etc. Therefore, it is non-trivial to find its ground state, which provides the solution to the problem. The system is therefore prepared in the ground state of an initial Hamiltonian H_M, which is varied from H_M to H_P over a time duration T as $H = (1 - \frac{t}{T})H_M + \frac{t}{T}H_P$. Note that for $t = 0$, the Hamiltonian is H_M, and for $t = T$, it is H_P. The only requirement for H_M is that its ground state should be easy to prepare, and $[H_P, H_M] \neq 0$. The adiabatic theorem asserts that if this evolution is slow enough, then the system, which is initially in the ground state of H_M, settles to the ground state of H_P as $T \to \infty$. For finite T, one can obtain an approximate solution. Therefore, the goal is to have a good approximation of the optimal solution within a reasonable T.

QAOA is a trotterization of AQC, where the adiabatic evolution is traced by repeated application of two unitaries, namely the problem unitary $U(H_P, \gamma)$ and the mixer unitary $U(H_M, \beta)$, where γ and β are parameters. A depth-p QAOA ansatz can be represented mathematically as in Eq. (2.4).

$$|\psi(\vec{\gamma}, \vec{\beta})\rangle = U(H_P, \gamma_p)U(H_M, \beta_p)\dots U(H_P, \gamma_1)U(H_M, \beta_1)\,|\psi_0\rangle \tag{2.4}$$

Here $\vec{\gamma} = \{\gamma_p, \dots, \gamma_1\}$ and $\vec{\beta} = \{\beta_p, \dots, \beta_1\}$ are the parameters which are optimized in every iteration of the algorithm by a classical optimizer. The number of times the problem and mixer Hamiltonians are applied is referred to as the depth p of

the QAOA. The initial state $|\psi_0\rangle$ can be an equal superposition state, or a special state determined by the knowledge of the specific problem.

2.3.2 QAOA for Max-Cut

Farhi et al. first studied QAOA [FGG14] for the Max-Cut problem, which is defined as

Max-Cut Problem

Given a graph $G = (V, E)$ where $|V| = n$ and $|E| = m$, find a bipartition of the graph such that the number of edges crossing from one partition to the other is maximized.

For Max-Cut, the problem Hamiltonian is $H_P = -\frac{1}{2}\sum_{(i,j)\in E} Z_i Z_j$, where Z_i is the Pauli operator Z acting on the qubit corresponding to vertex i. The mixer Hamiltonian is not unique, but it is selected to be $H_M = \sum_{i\in V} X_i$ since this state can be prepared by a depth 1 circuit by applying X gates simultaneously on all the qubits. Farhi et al. also showed that their QAOA algorithm for $p = 1$ performs better than random guessing, providing an approximation ratio of 0.6924 for 3-regular graphs. This is still not as good as the best-known classical algorithm for Max-Cut [GW95] which achieves an approximation ratio of 0.878. However, the authors showed that the approximation ratio of QAOA is a non-decreasing function of p. Therefore, it can be expected that with increasing p, the approximation ratio of QAOA is likely to improve and that it can compete with, or even beat, the best-known classical algorithm for an acceptable value of p. In [B+19], the authors showed that it is not possible to outperform the algorithm of [GW95] using QAOA if p is a constant. Nevertheless, this does not rule out the possibility of having a p, which is polynomial in $|V|$, for which QAOA can outperform the algorithm of [GW95].

2.3.3 Variants of QAOA

Since the proposal of QAOA by Farhi et al. [FGG14], several studies have been done to improve the quality of the outcome obtained from this algorithm. This chapter reviews some of the major variations of QAOA proposed in the literature.

- **Warm-start QAOA**: The initial state of the vanilla QAOA is usually the equal superposition of all the qubits. Therefore, the circuit to prepare the initial state for QAOA has depth 1. In [EMW21], the authors showed that QAOA can converge faster to more accurate results if the initial state is a *good* solution of the optimization problem at hand, instead of an equal superposition. In other words, this method involves running a classical polynomial time heuristic algorithm for the optimization problem. This step produces a *good* solution, which is then provided as the initial state of QAOA. Note that the initial state of the QAOA can still be prepared with this method using a depth 1 circuit. Nevertheless, in [CFG+22] the authors presented certain scenarios where warm-start QAOA can get stuck in local optima, resulting in its inability to reach the global optima for the problem.

- **Quantum alternating operator ansatz**: The original vanilla QAOA proposed by Farhi et al. [FGG14] was aimed for unconstrained optimization problems. In [HWO+19], the authors extended it to constrained optimization problems. They called this version Quantum alternating operator ansatz so that the acronym is still QAOA. In order for QAOA to extend to constrained optimization problems, it is necessary to ensure that the Mixer Hamiltonian maps one valid solution to another (or a superposition of) valid solutions. Hence, the Mixer Hamiltonian for this version of QAOA is no longer a depth 1 circuit. Nevertheless, this extension allows QAOA to encompass significantly more optimization problems. Moreover, for this version of QAOA, it is necessary that the initial state is a valid solution, making it a version of the warm-start QAOA.

- **Adapt QAOA**: The ansatz of a depth p QAOA is shown in Eq. (2.4).

In [ZTB+22], the authors proposed a dynamical approach to the design of QAOA ansatz. Suppose a depth $p-1$ QAOA ansatz is prepared, and the task is to create a depth p ansatz. In their method, the authors proposed analytical methods involving gradient calculation to find the best circuit to append to the depth $p-1$ ansatz. This method makes the ansatz design more complicated but was shown to converge faster than the vanilla QAOA.

- **CVaR QAOA**: The general notion of any variation algorithm is to find the expectation value from the outputs of the quantum circuit, and feed it to the classical optimizer which proposes a better set of parameters. However, when the quantum circuit is executed multiple times, some of the times it may produce low-quality outcomes. Calculating the expectation value over all the outcomes lowers its value due to the presence of these low-quality outliers. This, in turn, misguides the classical optimizer. Therefore, in [BNR+20], the authors suggested the use of the top few best-quality outcomes from circuit execution, and calculating expectation value over these outcomes only. They called this method *Conditional Variance at Risk (CVaR)*, which is a term from finance. They studied QAOA, and more general variational algorithms, using this CVaR method and showed improvement in the quality of results primarily in domains such as finance and quantum chemistry. The performance of CVaR QAOA was comparable to the vanilla QAOA and did not offer a major improvement in general.

Apart from the ones reviewed here, there are several other minor variations of QAOA studied for faster convergence [LJJG22, APZB21]. A great deal of effort has been given to improve the overhead of classical optimization [AASG20, RSC+22, CRAB21]. However, the issue of circuit synthesis and elimination of 2-qubit gates for QAOA, which is studied in this thesis, deals only with the vanilla version of QAOA [FGG14].

The next section deals with another method, called circuit cutting, used for near-term quantum computation to lower the effective size of the circuit, and thus lower the noise in the system.

2.4 Circuit cutting

Circuit-knitting is an umbrella term that indicates various mechanisms to partition a large circuit into multiple smaller subcircuits. These methods involve (i) wire cutting where a circuit is partitioned into subcircuits by splitting between two gates (figuratively along a wire) [PHOW20, TTS+21]; (ii) gate cutting where a 2-qubit gate is replaced by multiple single qubit gates with feed-forward classical communication [MF21a, PS23]; and (iii) entanglement forging [EMG+22] where the problem Hamiltonian is partitioned into a product of smaller sub-Hamiltonians such that each sub-Hamiltonian leads to a smaller circuit.

These methods caught the eye of the community since they allowed users to execute larger circuits even if they had access to smaller hardware. However, the overhead that comes with these methods is that each subcircuit needs to be executed multiple times (usually to measure and/or prepare qubits in different basis states), and classical postprocessing is required to determine the probability distribution or expectation value of the uncut circuit from the outcome of the subcircuits. Efforts have been made to combine wire and gate cutting [BPK23] and reduce the overhead by incorporating classical communication [Ped23, BPS23].

This thesis focuses only on wire cutting mechanism. Although the term *circuit cutting* encompasses both wire and gate cutting, in this thesis this term will henceforth imply wire cutting only. The idea of such circuit cutting was first proposed in [PHOW20]. Given a circuit Φ, let us denote the expectation value of an observable A as $\Phi(A)$. Note that, for A, it is possible to express it as [TTS+21]

$$A = \frac{Tr\{A.I\}I + Tr\{A.X\}X + Tr\{A.Y\}Y + Tr\{A.Z\}Z}{2}$$

where I, X, Y, Z are the Pauli operators [NC10]. In other words,

$$\Phi(A) = \frac{1}{2} \sum_{P \in \{I,X,Y,Z\}} c_P \Phi_P(A),$$

where $\Phi_P(A) = Tr\{AP\}\rho_P$. Here, ρ_P denotes the eigenstates of the Pauli operator P, and c_P denotes the eigenvalue. Note that the mathematical expression $Tr\{AP\}\rho_P$ takes instances of both subcircuits into account where the former is measured in basis P and the latter is prepared in the state ρ_P. Since there are two eigenstates corresponding to each Pauli operator, this method results in four subcircuit instances for measurement basis and eight for preparation state. The uncut expectation value (or probability distribution) is obtained via classical post-processing.

In [TTS+21], the authors showed that the previous representation of the observable A is tomographically over-complete; It is possible to have a more succinct representation of $\Phi(A) = \sum_i Tr\{AO_i\}\rho_i$, where $O_i \in \{X, Y, Z\}$ and $\rho_i \in \{|0\rangle, |1\rangle, |+\rangle, |+i\rangle\}$. These two sets O_i and ρ_i are tomographically complete and hence denote the minimum number of subcircuits necessary. Here, there are three subcircuit instances for measurement basis and four for preparation state. The circuit-knitting-toolbox [BBB+23] is a useful tool for performing experiments for both wire and gate cutting.

A general drawback of cutting is that the classical postprocessing time scales exponentially in the number of cuts when the full probability distribution needs to be reconstructed. Therefore, this method is suitable only for circuits that can be split into disjoint subcircuits using a small (ideally constant) number of cuts only.

2.4.1 Circuit cutting as a method to improve fidelity

Initially, circuit cutting was presented as a method for the simulation of larger circuits on smaller devices. However, currently, there is quantum hardware with 400+ qubits, whereas the largest reliable experiment to date involves only $\sim$ 100 qubits [KE+23]. In this scenario, circuit cutting has emerged as a promising method to improve fidelity. Since each subcircuit has fewer qubits and/or gates, these are expected to be less susceptible to noise. In [ARS+21], the authors studied the effect of circuit cutting in the improvement of fidelity for different types of noise and used

circuit cutting on quantum circuits for combinatorial optimization in [STP+21]. In [BSCSK21], the authors proposed a machine learning-based method to find a good cut location so that the fidelity of each subcircuit is maximized. A follow-up of this was studied in [BDS+23] where the points of cut were decided based on the hardware noise profile. In [KMS+23], the authors used circuit cutting on variational algorithms and obtained a better estimate of the ground state of a Hamiltonian than the uncut circuit. This method was improved in [BMSSK23] where the authors formally expressed the concept of scheduling subcircuits to hardware as an optimization problem. Here, the authors obtained improved fidelity for different types of circuits using circuit cutting and optimal scheduling of the subcircuits.

2.4.2 Tomographic circuit cutting

Reconstruction of expectation values of the full circuit assumes accurate estimation of expectation values of each fragment, but statistical errors due to a finite number of samples of each fragment can lead to an invalid distribution which may be neither non-negative nor normalized. Scaling the distribution obtained and converting it into a valid probability may be attempted. However, maximum-likelihood tomography [PSSO21] to constrain each fragment to a valid physical state or channel results in a valid final distribution is shown to yield a higher fidelity with the ideal distribution as compared to scaling an invalid distribution.

Tomography is a procedure to characterize an unknown quantum state (called *state tomography*), or quantum channel (called *process tomography*). Circuit cutting can produce fragments that behave as an unknown quantum state or channel. Fig. 2.4 shows an example of cutting a 4-qubit GHZ circuit (shown in Fig. 2.3) into three fragments. The first fragment in Fig. 2.4 (a) is essentially a density matrix or quantum state characterized by the measurement M_j on the second qubit. The second fragment shown in Fig. 2.4 (b) is a channel characterized by both the preparation qubit P_i and the measurement basis M_j. Finally, the third fragment in Fig. 2.4 (c) is a Positive Operator Valued Measure (POVM) [NC02] since it corresponds to a measurement in some basis determined by the quantum gates before the measurement in a computational basis.

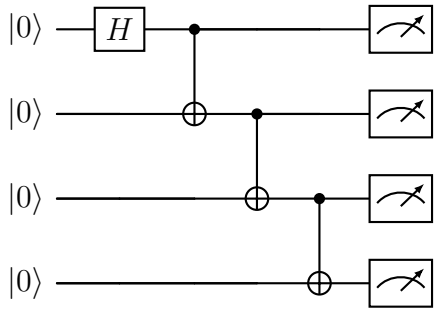


Figure 2.3: A 4-qubit GHZ circuit

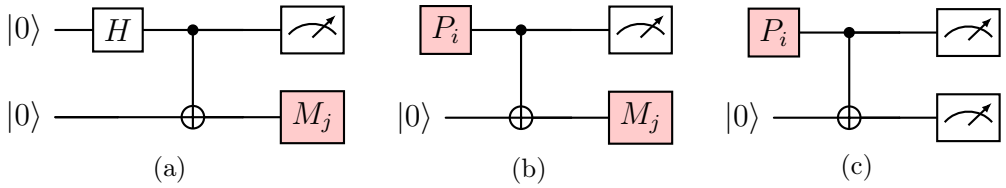


Figure 2.4: An example of cutting the 4 qubits GHZ state shown in Fig. 2.3 into 3 fragments: (a) the first one has only a single tomographic measurement M_j and behaves as a quantum state fragment, (b) the second one is a general fragment with both tomographic preparation and measurement is a quantum channel fragment, and (c) the third one has only a single tomographic preparation P_i and behaves as a POVM. Note that the number of effective qubits of the conditional tensor in the fragment corresponds to the number of tomographically prepared or measured qubits, which is 1-qubit in all three cases, rather than the total number of qubits in a fragment.

2.4.3 A brief introduction to quantum tomography

This study considers a general description of the tomography of a tensor T. This encompasses (i) state tomography when $T = \rho$ corresponds to a density matrix, (ii) process tomography when $T = \Lambda$ is a Choi-matrix [NC02], and (iii) measurement tomography when $T = M_j$ is a POVM element. In all three cases, quantum tomography of T consists of choosing a basis $\{B_j\}$ of tensors that spans T, where such a spanning set is called *tomographically complete* and can be used to experimentally measure the set of measurement probabilities $p_j = \langle\langle B_j | T \rangle\rangle$, where $|T\rangle\rangle$ denotes *vectorization* of the tensor T [GTW09]. For state, process, and measure-

ment tomography, the basis can be chosen as $B_i = M_i$, $B_{ij} = \rho_i^T \otimes M_j$, $B_i = \rho_i^T$ respectively, where $\{\rho_i\}$ is a tomographically complete preparation basis of input states, and $\{M_j\}$ is a tomographically complete basis of measurement POVMs.

Before proceeding further, a brief introduction to vectorization and Choi matrix are provided herein for the sake of completeness.

1. **Vectorization**: Vectorization of a density matrix ρ can be obtained by arranging the columns (row) of ρ one after another into a single column (row) to form a column (row) matrix $|\rho\rangle\rangle$.
   For example, if $\rho = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$, then its column vectorized form is given by $|\rho\rangle\rangle = \begin{pmatrix} a & c & b & d \end{pmatrix}^T$. The analysis [GTW09] over the vectorized form of a density matrix is equivalent to that over the density matrix itself. However, vectorization often makes the representation simpler.

2. **Choi matrix**: The action of a quantum channel is completely captured by the Choi matrix representation [NC02]. The Choi matrix $\Lambda_{\mathcal{E}}$ corresponding to a channel $\mathcal{E}$, is denoted by $\Lambda_{\mathcal{E}} = (I \otimes \mathcal{E}) |\Omega\rangle \langle\Omega|$, where $|\Omega\rangle$ is the unnormalized Bell state. The dimension of $\Lambda_{\mathcal{E}}$ is twice that of $\mathcal{E}$. Hence the completely positive trace-preserving (CPTP) conditional of $\mathcal{E}$ is transferred to the trace of the Choi matrix, as $\text{Trace}(\Lambda_{\mathcal{E}}) = 2$ for a single qubit channel. If $\Lambda_{\mathcal{E}}$ corresponding to $\mathcal{E}$ is known, then $\mathcal{E}(\rho) = Tr_1(\Lambda(\rho^T \otimes I))$ for any input state ρ, where Tr_1 denotes partial trace over subsystem 1. In other words, $\Lambda_{\mathcal{E}}$ captures the entire information of a process $\mathcal{E}$. A Choi matrix is also a valid density matrix. Therefore, process tomography of a channel $\mathcal{E}$ boils down to state tomography of $\Lambda_{\mathcal{E}}$ [WBC15].

If $\{B_j\}$ is tomographically complete, then the probabilities $\{p_j\}$ contain sufficient information to completely reconstruct T. This thesis considers two reconstruction methods which both implement a form of maximum-likelihood estimation. The first is *linear inversion* combined with re-scaling of the fitted tensor to enforce pos-

itivity as described in [SGS12], and the second is constrained linear-least squares estimation implemented as *semidefinite program* optimization problem.

Linear Inversion

For a tomographically complete basis $\{B_j\}$ and outcome probabilities $\{p_j\}$ linear inversion amounts to constructing a *dual basis* [DMP00] $\{D_j\}$ defined by the orthogonality relation $\langle\langle D_i|B_j\rangle\rangle = \delta_{ij}$ as

$$|D_j\rangle\rangle = \left(\sum_i |B_i\rangle\rangle\langle\langle B_i|\right)^{-1} |B_j\rangle\rangle \tag{2.5}$$

The linear inversion estimate of the tensor T is then given by

$$T_{LIN} = \sum_i p_i D_i. \tag{2.6}$$

The linear inversion estimate of a state or channel is generally not positive or completely-positive respectively, however, performing a specific re-scaling of eigenvalues will result in a physical state that is consistent with the maximum likelihood estimated value under the assumption of Gaussian measurement noise [SGS12].

Constrained Least-Squares

For a tomographically complete basis $\{B_j\}$ and outcome probabilities $\{p_j\}$ constrained least-squares tomography is the optimization problem given as

$$T_{LS} = \arg\min_{T\geq 0} \frac{1}{2} \left\| \Sigma^{-1/2} \left(S_B|T\rangle\rangle - |p\rangle \right) \right\|_2^2 \tag{2.7}$$

where $|p\rangle = \sum_i p_i |i\rangle$ is a vector of measured outcome probabilities, $\Sigma^{-1/2}$ is a covariance matrix for the measurement outcome probabilities $\{p_j\}$ and $S_B|T\rangle\rangle = \sum_i |i\rangle\langle\langle B_i|T\rangle\rangle$ is a vector of expected probabilities for the model T.

Typically, additional constraints are also included in Eq. (2.7), such as T is trace 1 for state tomography, trace-preserving for process tomography, or the sum of POVM elements is the identity for measurement tomography. In all these cases, these constraints are positive-semidefinite and the resulting optimization problem is a semidefinite program.

A general issue with the tomographic approach is that tomography scales exponentially, $\mathcal{O}(4^n)$ for state tomography and $\mathcal{O}(12^n)$ for process tomography, with the number of qubits n. In [PSSO21] the authors proposed a mechanism called *conditional fragment tomography* in which the tomography can be performed only on a smaller number of qubits, and the reconstruction is *conditioned* on the qubits not involved in tomography, thus making it more scalable, and applied it in the context of circuit cutting. This method was shown to perform better than normalization of the obtained distribution from the subcircuits. In other words, under the noiseless scenario, the fidelity of the constructed distribution with the ideal uncut distribution was higher using conditional fragment tomography. Chapter 5 of this thesis builds on this method for real noisy scenarios. In particular, error mitigation methods particular to conditional fragment tomography are proposed to improve the fidelity in the presence of noise.

The methods discussed so far are applicable to lower the effect of noise in quantum computers in the absence of error correction. However, for arbitrary long computations, error correction is necessary. The following section provides a brief review of quantum error correction.

2.5 Quantum error correction

Errors in quantum systems are unwanted unitary operators. The general difficulty in developing error-correcting codes for quantum systems arises from the following constraints: (i) there are infinitely many possible errors, (ii) it is not possible to copy arbitrary quantum states [WZ82] to create redundancy, and (iii) measuring a

quantum state to check for errors collapses the system, thus losing the information.

Any unitary error on a qubit can be represented as a linear combination of the Pauli matrices [NC02], i.e.,

$$\mathbb{E} = a\mathbb{I} + b\sigma_x + c\sigma_y + d\sigma_z.$$

This thesis uses both the terminologies $\sigma_x, \sigma_y, \sigma_z$ and X, Y, Z interchangeably for the Pauli matrices (or operators). Peter Shor first provided a scheme for error correction [Sho95] where the system of interest is entangled with a few ancilla qubits. When the ancilla qubits are measured, the system collapses to a state with one of the four possible Pauli errors. This approach solves all the apparent hindrances of quantum error correction.

While the approach of Shor was from the viewpoint of a circuit, where he created the encoded states, Gottesman designed a mathematical formulation for quantum error correction, called stabilizers [Got97]. The mathematical formulation of Gottesman encompasses any stabilizer code, and the Shor code can also be derived from this formulation. More concretely, the approach of Shor code was to prepare the encoded state, while the approach of Gottesman was to find the group of operators under which the noiseless encoded state is invariant.

For a n-qubit system, a set of operators $S_1, S_2, \ldots S_m \subset \{I, \pm\sigma_x, \pm i\sigma_x, \pm\sigma_y, \pm i\sigma_y, \pm\sigma_z, \pm i\sigma_z\}^{\otimes n}$, where each σ_i is a Pauli operator [NC02], is said to stabilize a quantum state $|\psi\rangle$ if the following criteria are satisfied:

1. $\forall i,\ S_i |\psi\rangle = |\psi\rangle,\ 1 \leq i \leq m$;
2. $\forall e \in \mathcal{E},\ \exists\ j$, such that $S_j(e|\psi\rangle) = -(e|\psi\rangle)\ 1 \leq j \leq m$;
3. for $e, e' \in \mathcal{E},\ e \neq e',\ \exists\ j, k$ such that $S_j(e|\psi\rangle) \neq S_k(e'|\psi\rangle),\ 1 \leq j, k \leq m$;
4. $\forall i, j,\ [S_i, S_j] = 0,\ 1 \leq i, j \leq m$.

An n-qubit state with m stabilizers can encode $k = n-m$ logical qubits (i.e., $n-m$

qubits of information). A distance d quantum error correcting code (QECC) can correct up to $t = \lfloor \frac{d-1}{2} \rfloor$ errors. Such a QECC is denoted as an $[[n, k, d]]$ code. Shor code is a $[[9, 1, 3]]$ QECC, whose stabilizers are shown in Table 2.1. In this QECC, the information of a single *physical* qubit is distributed into 9 *physical* qubits, resulting in a single *logical* qubit. A logical qubit is more robust to noise than a physical qubit. In the table, an empty location in each row (or column) indicates the identity operator. Explicitly, the stabilizer S_1 has the form $ZZIIIIIII$, where the i^{th} operator acts on qubit q_i.

Table 2.1: Stabilizer for 9-qubit QECC (Shor code) [Sho95]

Stabilizers ↓ \ Qutrits →	q_1	q_2	q_3	q_4	q_5	q_6	q_7	q_8	q_9
S_1	Z	Z							
S_2		Z	Z						
S_3				Z	Z				
S_4					Z	Z			
S_5							Z	Z	
S_6								Z	Z
S_7	X	X	X	X	X	X			
S_8				X	X	X	X	X	X

The circuit of this QECC is shown in Fig. 2.5, where the initial circuit corresponds to encoding the information of a single qubit $|\psi\rangle$ into nine qubits, and the error block E is followed by decoding.

Shor's seminal paper shows that it is possible to correct unitary errors in a quantum system. Other QECCs such as the 7-qubit code by Steane [Ste96a] and the 5-qubit code by Laflamme [LMPZ96] are other schemes of encoding using fewer qubits. The 5-qubit code was shown to be optimal in the number of qubits in order to correct a single error.

All the QECCs mentioned above correct a single error only. However, bigger quantum circuits may incur more errors. Furthermore, the quantum gates used for encoding and decoding are also not perfect themselves and thus may incorporate further errors. In fact, executing the Shor code on a noisy quantum simulator of Qiskit [H+19], namely *FakeCairo*, which mimics the error map of the 27 qubit

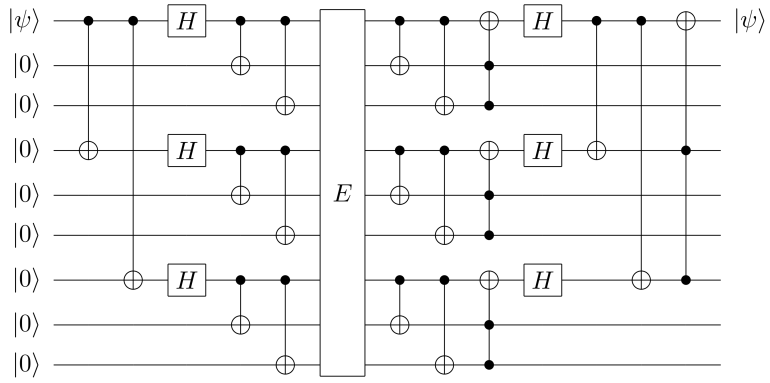


Figure 2.5: Circuit for Shor code
(Courtesy: https://quantumcomputinguk.org/tutorials/quantum-error-correction-shor-code-in-qiskit)

quantum hardware IBMQ Cairo, provided a fidelity of 0.79, i.e., the QECC has been unable to keep the state error-free. Correcting multiple errors can be achieved by a process called *concatenation* [NC02], where the information of a logical qubit itself is distributed into multiple logical qubits. This allows for correcting multiple errors at the cost of increased qubit and gate costs. However, the rate of increase in the number of gates due to concatenation is significantly less than the rate of lowering of error. Thus, the resultant quantum circuit becomes robust to multiple errors and can keep the system error-free even when each of its components is faulty. This is termed as *fault-tolerance* [Sho96].

The encoding circuit of Shor Code (Fig. 2.5) requires CNOT gates between different pairs of qubits which may not be adjacent in current hardware with restricted connectivity. Therefore, topological codes [FMMC12, CKYZ20, CZY+20], which restrict 2-qubit operations only between neighboring qubits, are widely studied. Recently, a quantum LDPC code was proposed that can accommodate more logical qubits than the other topological codes using the same number of physical qubits [Jon13]. However, topological codes usually require a large number of qubits for encoding. While LDPC code triumphs over topological codes with fewer qubit requirements, it requires long-range qubit interaction – which is not available in current quantum devices. Since resource reduction is a primary goal of this thesis, these QECCs are out of scope.

Part I

Error suppression and mitigation for NISQ devices

CHAPTER 3

Graph algorithmic approach to QAOA circuit size optimization

Contents

3.1 Introduction

Near-term quantum systems contain only a few hundred qubits, which are noisy. Such a quantum computer cannot reliably execute the much-celebrated quantum algorithms such as integer factorization [Sho97], search over unstructured databases [Gro96a], estimation of eigenvalues [NC02], solving linear systems of equations [HHL09] etc. for sufficiently large data. Quantum error correction is also not feasible with such a small number of qubits, and the error rates of current quantum hardware are not sufficiently small to allow fault tolerance. Therefore, the execution of these celebrated algorithms must wait a few more years.

However, in the meantime, quantum-classical hybrid algorithms have been developed in which some portion of the computing effort is offloaded to classical processors. This ensures that the quantum circuit is shallow, with a low gate count, and is thus less susceptible to error. These algorithms have applications in finding approximate solutions to NP-Hard combinatorial optimization problems [FGG14, FH16, GSL18], quantum chemistry such as determining the ground state of an electronic system [C$^+$19, GEBM19, S$^+$20, MRBAG16], and machine learning [BWP$^+$17, TM19].

The Quantum Approximate Optimization Algorithm (QAOA) [FGG14] was proposed primarily for finding approximate solutions to combinatorial optimization problems. A general notion, which Farhi expressed in a talk, was that we are missing out *good enough* use cases of a quantum computer in search for the *best*. In other words, apart from algorithms for integer factorization, database search, etc., even hybrid quantum-classical NISQ algorithms for quantum chemistry, and quantum machine learning thrive to obtain the best outcome. However, finding the best solution is often NP-Hard for combinatorial optimization problems, and therefore the users are happy to settle for a *good enough* solution. QAOA works in this *good enough* regime and aims to improve the quality of approximation and/or show quantum speed-up.

Many such instances of combinatorial optimization can be mathematically expressed as a quadratic unconstrained binary optimization (QUBO) problem. The following sections first briefly discuss QUBO, followed by QAOA and its formulation for QUBO problems.

3.2 Quadratic Unconstrained Binary Optimization (QUBO) and its Hamiltonian formulation

Many problems of interest can be expressed as a Quadratic Unconstrained Binary Optimization (QUBO). In this formulation, a problem with n variables is expressed as some function of x_j, $\forall j$, $x_j \in \{0, 1\}$, called the cost function. The cost function $C(x_1, x_2, \ldots, x_n)$, corresponding to the problem, can contain only single (x_j) or two variable interaction terms ($x_j x_k$) - thus the name *quadratic*. The target is to assign values to each x_j such that the cost function is minimized (or maximized).

An example of QUBO formulation is shown henceforth for the Max-Cut problem, which is one of the most widely studied problems for QAOA. The formal definition of the Max-Cut problem is given in Definition 3.1.

3.1: Max-Cut Problem

Given a graph $G = (V, E)$ where $|V| = n$ and $|E| = m$, find a bipartition of the graph such that the number of edges crossing from one partition to the other is maximized.

To obtain a QUBO formulation of this problem, consider a binary variable x_j assigned to each vertex j. If the graph is bipartitioned into V_1 and V_2 such that $V_1 \cup V_2 = V$ and $V_1 \cap V_2 = \phi$, then it can be decided that

$$x_j = \begin{cases} 1, & \text{if } j \in V_1 \text{ (or } V_2) \\ 0, & \text{otherwise.} \end{cases}$$

An edge (j, k) is said to belong to the cut if $i \in V_1$ and $j \in V_2$, or vice versa. The cost function for the Max-Cut problem [Wes01] can, then, be represented as

$$Maximize\ C(x_1, x_2, \ldots, x_n) = \sum_{(j,k)\in E} (x_j + x_k - 2x_j x_k)$$

For every edge (j, k), the cost function gets a value of 2 if (j, k) belongs to the cut, 0 otherwise. In general, QUBO formulation is meant for unconstrained optimization problems. However, constrained optimization problems can also be formulated as a QUBO by appending the constraints as penalties in the cost function [GKHD22].

A general QUBO has the form

$$\sum_{j,k} Q_{jk} x_j x_k + \sum_j b_j x_j + c \tag{3.1}$$

where Q_{jk}, b_j and c are coefficients. The QUBO formulation and its graph structure follow hand-to-hand. The previous example showed the QUBO formulation for a graph problem, namely Max-Cut. The QUBO of Eq. (3.1) can be interpreted as a graph with each x_j representing a vertex with weight b_j and each (x_j, x_k), for which $Q_{jk} \neq 0$ as an edge with weight Q_{jk}. Henceforth, a QUBO and its graphical representation will be used interchangeably.

To formulate a QUBO problem in the quantum domain, it is necessary to shift from $\{0, 1\}$ to $\{+1, -1\}$ variables. Any information regarding a quantum state corresponds to an observable, and the observable can be shown to have eigenvalues $\{+1, -1\}$ [GS18, NC02]. Therefore, the quantum representation of a QUBO problem belongs to the $\{+1, -1\}$ variable domain, where a qubit is associated with each vertex, and the outcome corresponds to the eigenvalue of some observable. A $\{0, 1\}$ variable x_j can be represented as a $\{+1, -1\}$ variable z_j where

$$x_j = \frac{1 - z_j}{2}.$$

For the rest of the thesis, z_j shall denote a $\{+1, -1\}$ variable, and Z_j will be the observable corresponding to z_j. Under this change of variable, the Max-Cut

problem has the representation

$$\begin{aligned}\sum_{(j,k)\in E}(x_j + x_k - 2x_jx_k) &= \sum_{(j,k)\in E}(\frac{1-z_j}{2} + \frac{1-z_k}{2} - 2\frac{1-z_j}{2}\frac{1-z_k}{2}) \\ &= \sum_{(j,k)\in E}(\frac{2-z_j-z_k-1+z_j+z_k-z_jz_k}{2}) \\ &= \sum_{(j,k)\in E}\frac{1-z_jz_k}{2}\end{aligned}$$

In the operator representation, each z_j is replaced by its corresponding observable Z_j, and each 1 is replaced by I. Associated with a quantum system is a hermitian operator H called the Hamiltonian [GS18], whose eigenvalue is the total energy of the system. The Hamiltonian representation of the Max-Cut problem is, thus, given in Eq. (3.2).

$$\text{Maximize } C(Z_1, Z_2, \ldots, Z_n) = \sum_{(j,k)\in E}\frac{I - Z_jZ_k}{2} \tag{3.2}$$

The Hamiltonian corresponding to the general QUBO in Eq. (3.1) is shown in Eq. (3.3).

$$\begin{aligned}H_P &= \sum_{j,k}Q_{jk}\frac{I-Z_j}{2}\cdot\frac{I-Z_k}{2} + \sum_j b_j\frac{I-Z_j}{2} + c \\ &= \sum_{j,k}\frac{Q_{jk}}{4}Z_jZ_k - \sum_j(\sum_{j,k}\frac{Q_{jk}}{4})Z_j - \sum_k(\sum_{j,k}\frac{Q_{jk}}{4})Z_k - \sum_j\frac{b_j}{2}Z_j \\ &\quad +(\sum_{j,k}\frac{Q_{jk}}{4} + \sum_j\frac{b_j}{2} + c)I \\ &= \sum_{j,k}\frac{Q_{jk}}{4}Z_jZ_k - \sum_j\frac{1}{2}(b_j + \sum_k Q_{jk})Z_j + (\sum_{j,k}\frac{Q_{jk}}{4} + \sum_j\frac{b_j}{2} + c)I \end{aligned}\tag{3.3}$$

Such a Hamiltonian H_P has only one or two variable terms. Henceforth such a Hamiltonian will be referred to as *two-body interaction Hamiltonian*.

A Quantum Approximate Optimization Algorithm (QAOA) is a quantum algorithm for finding good approximate solutions to combinatorial optimization problems. The initial studies on QAOA focused on the QUBO formulation of unconstrained optimization problems only. However, it is possible to design QUBO for certain constrained optimization problems as well by adding the constraints as penalty terms in the objective function [GKD18]. The following section provides a brief introduction to QAOA.

3.3 Quantum Approximate Optimization Algorithm (QAOA)

Quantum Approximate Optimization Algorithm (QAOA) is the trotterized version of adiabatic quantum computing (AQC). Therefore, a brief introduction to AQC is provided in the next subsection, followed by its trotterization, which is QAOA.

3.3.1 Adiabatic Quantum Computing (AQC)

Computation with quantum devices can be broadly classified into three types - (i) gate-based, (ii) measurement-based, and (iii) continuous Hamiltonian evolution. In the first one, a quantum state is evolved by application of multiple unitary operators, termed as *quantum gates*, and is finally measured to obtain the desired outcome with high probability. Most of the well-known quantum algorithms, such as Shor algorithm [Sho97], Grover's algorithm [Gro96a] fall under this category. Another approach to quantum computing is measurement-based where the desired state is prepared by repeated measurement in different bases [BBD+09]. It has been shown that these two approaches are equivalent to one another. However, there is another approach of quantum computation, used primarily for optimization problems, called *adiabatic quantum computation* [AL18]. In this method, a problem P is encoded as a Hamiltonian H_P such that the ground state of this

Hamiltonian is the optimal solution of the problem. The problems of interest, from domains of graph theory, chemistry, etc., are, in general, NP-complete. Hence, preparing the ground state of H_P is not trivial.

Adiabatic quantum computation prepares a time-dependent Hamiltonian $H(t) = (1-\frac{t}{T})H_M + \frac{t}{T}H_P$, where H_M is some other Hamiltonian, such that $[H_M, H_P] \neq 0$, whose ground state is easy to prepare. The Hamiltonian $H(t)$ is evolved *slowly* from time $t = 0$ to T. The value of T is governed by *Quantum Adiabatic Theorem.*

Theorem 3.1: Quantum Adiabatic Theorem

Given a time-dependent Hamiltonian $H(t) = (1 - \frac{t}{T})H_M + \frac{t}{T}H_P$, a system $|\psi(0)\rangle$, prepared in the ground state of H_M, evolves to the ground state $|\psi\rangle$ (T) of H_P in time $t = T$ if $T \propto \frac{1}{g_{min}^2}$, where $g_{min} = \min_{0\leq s\leq 1} E_1(s) - E_0(s)$, $E_i(s)$ being the i-th energy state of $H(t)$ at $s = \frac{t}{T}$.

A primary requirement for the success of adiabatic quantum computation is that $[H_P, H_M] \neq 0$, which ensures that $g_{min} > 0$ [FGGS00, FGG+01], leading to a transition from the ground of H_M to the ground state of H_P. If $[H_P, H_M] = 0$, then it can be shown that such a transition doesn't occur.

In general, it is often not trivial to determine g_{min}. Furthermore, for NP-Complete problems, $g_{min} \to 0$, thus making $T \to \infty$. Therefore, the primary interest is often to find good approximate solutions to the optimization problem in some finite time.

Currently, D-Wave has an adiabatic quantum computer with ~ 2000 qubits. However, this computation technique is suitable mainly for optimization problems and does not conform with the gate-based quantum computation approach. A trotterized version of adiabatic quantum computation is named *Quantum Approximate Optimization Algorithm* (QAOA) and is suited for gate-based quantum computation.

3.3.2 QAOA: Trotterization of AQC

Evolution of a quantum system, described by a Hamiltonian H, from the initial state $|\psi(0)\rangle$ to $|\psi(t)\rangle$ in time t is expressed as

$$|\psi(t)\rangle = exp(-\frac{i}{\hbar}H \cdot t)\,|\psi(0)\rangle$$

where $\hbar$ is the reduced Plank constant [GS18]. Now, $H(t) = (1 - \frac{t}{T})H_M + \frac{t}{T}H_P$. Therefore,

$$\begin{aligned} exp(-\frac{i}{\hbar}H \cdot t) &= exp(-\frac{i}{\hbar}[(1 - \frac{t}{T})H_M + \frac{t}{T}H_P] \cdot t) \\ &= exp(-i(\beta' H_M + \gamma' H_P)) \\ &= \lim_{p\to\infty}(exp(-i\frac{\beta'}{p}H_M) \cdot exp(-i\frac{\gamma'}{p}H_P))^p \\ &= \lim_{p\to\infty}(exp(-i\beta_p H_M) \cdot exp(-i\gamma_p H_P))^p \end{aligned}$$

The above equation follows from the Suzuki-Trotter decomposition, also termed as *trotterization*. This decomposition asserts that adiabatic quantum computing can be approximated by two sets of quantum gates $U(H_M, \beta_p) = exp(-i\beta_p H_M)$ and $U(H_P, \gamma_p) = exp(-i\gamma_p H_P)$, applied alternately for p *steps* [FGG14], resulting in a parameterized quantum circuit.

The Quantum Approximate Optimization Algorithm (QAOA) [FGG14] is a hybrid quantum-classical algorithm studied primarily for finding an approximate solution to combinatorial optimization problems. A QAOA is characterized by a Hamiltonian H_P that encodes the combinatorial optimization problem and a Mixer Hamiltonian H_M whose ground state is easy to prepare and $[H_P, H_M] \neq 0$. Two parameterized unitaries $U(H_P, \gamma) = exp(-i\gamma H_P)$ and $U(H_M, \beta) = exp(-i\beta H_M)$ are applied sequentially for $p \geq 1$ times on the initial state $|\psi_0\rangle$. Here $\gamma = \{\gamma_1, \gamma_2, \ldots, \gamma_p\}$ and $\beta = \{\beta_1, \beta_2, \ldots, \beta_p\}$, $\gamma_i, \beta_i \in \mathbb{R}\ \forall\ i$, are the parameters. The algorithm starts with some initial state $|\psi_0\rangle$, applies the two unitaries consecutively for p steps, and is finally measured. A single epoch of a level-p QAOA is represented as in

Eq. (3.4).

$$|\psi(\gamma,\beta)\rangle = (\Pi_{l=1}^{p} e^{(-i\beta_l H_M)} e^{(-i\gamma_l H_P)})\,|\psi_0\rangle \tag{3.4}$$

Such a parameterized circuit $|\psi(\gamma,\beta)\rangle$ is termed as *ansatz*. The parameters are usually initialized randomly, and after an epoch of iterations, that provides the expectation value $\langle\psi(\gamma,\beta)|H_P|\psi(\gamma,\beta)\rangle$, the parameters are updated by a classical optimizer. The next epoch uses this new set of parameters and is expected to provide an expectation value that is closer to the optimum solution to the problem. In other words, the objective function of a depth-p QAOA can be expressed as

$$\max_{\psi(\gamma,\beta)} \langle\psi(\gamma,\beta)|\, H_P\, |\psi(\gamma,\beta)\rangle \tag{3.5}$$

Farhi et al. first proposed QAOA [FGG14], and studied it in the context of finding a maximum cut in a graph, known as the Max-Cut problem. For 3-regular graphs, they showed that a $p = 1$ QAOA achieves an approximation ratio better than random guessing, and the approximation ratio is a non-decreasing function of p [GW95]. Therefore, QAOA is expected to be a potential candidate for quantum advantage using near-term devices. Henceforth, QAOA has also been studied to find approximate solutions to other combinatorial optimization problems as well [HWO$^+$19, Sal20, FB$^+$18, CEB20, FGG20]. The study in this and the next chapter will focus only on problems involving one- or two-body interaction Hamiltonians.

Eq. (3.3) shows the Hamiltonian formulation corresponding to generalized QUBO. To solve such a problem via QAOA, the operator $U(H_P,\gamma)$ takes the form as in Eq. (3.6). The terms involving only scalar coefficients have been discarded for brevity since they act as global phases and do not find their way into the corresponding quantum circuit.

$$U(H_P,\gamma) = \Pi_{j,k} exp(-i\gamma\frac{Q_{jk}}{4}Z_jZ_k)\cdot\Pi_j exp(-i\gamma\frac{1}{2}(b_j+\sum_k Q_{jk})Z_j \tag{3.6}$$

The circuit representation of Eq. (3.6) involves a bunch of $exp(-i\theta_1 Z_jZ_k) = R_{Z_jZ_k}$

and $exp(-i\theta_2 Z_j) = R_{Z_j}$ terms. While the latter is realized as $Rz(\theta_2)$ gates, with $\theta_2 = \frac{1}{2}(b_j + \sum_k Q_{jk})$ for each such gate, the former is realized as in Fig. 3.1 [Had18], where $\theta_1 = \frac{Q_{jk}}{4}$ for each such gate.

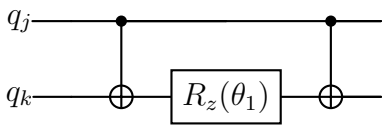


Figure 3.1: Circuit realization of the operator $R_{Z_i Z_j}$

For the rest of the thesis, a *step* would imply the part of a layer executing the circuit of Fig. 3.1. Multiple operators of the form $exp(-i\gamma Z_j Z_k)$ can be executed in a single step for disjoint pairs (j, k).

For the sake of completeness, we show the circuit diagram of the operator $U(H_P, \gamma)$ for an example two body Hamiltonian as in Eq. (3.7).

$$H_P = \frac{Q_{01}}{4} Z_0 Z_1 + \frac{Q_{12}}{4} Z_1 Z_2 - \frac{1}{2}(b_0 + Q_{01}) Z_0 - \frac{1}{2}(b_1 + Q_{12}) Z_1 \tag{3.7}$$

Therefore the circuit for $U(H_P)$, corresponding to this H_P, has two R_{zz} operators, and two R_z operators. For the sake of brevity, let us assume that the parameter corresponding to each of R_{zz} and R_z operators is θ_1 and θ_2 respectively. Under this assumption, the circuit of $U(H_P)$ is shown in Fig. 3.2.

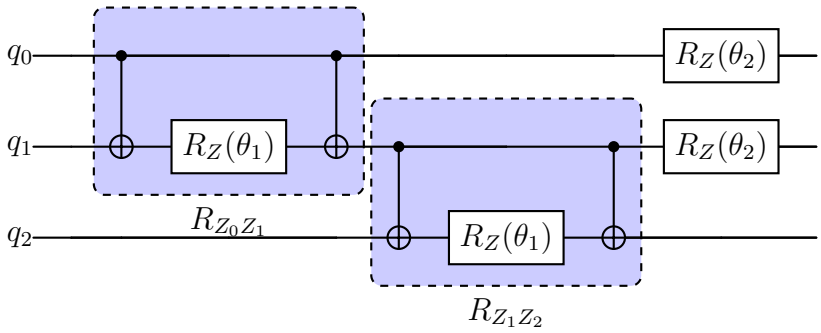


Figure 3.2: An example QAOA circuit of $U(H_P)$ where H_P is as in Eq. (3.7)

3.4 CNOT elimination in QAOA circuit

Ideally, the expectation value $\langle\psi(\gamma,\beta)|\, H_P\, |\psi(\gamma,\beta)\rangle$ of a QAOA is a non-decreasing function of p [FGG14]. However, experiments in real noisy hardware show that the expectation value starts decreasing beyond a certain p as an effect of noise [HSN+21]. In the absence of error correction, several error mitigation methods have been proposed in the literature [TBG17, BW20a, EBL18, ECBY21, BMKT22] that aims to lower the effect of noise on the circuit. Apart from general error mitigation techniques variation in the Mixer Hamiltonian [ZTB+22], or the Cost Function [LJJG22, BNR+20] have been proposed that either lowers the noise in the circuit or achieves faster convergence, thus reducing the depth of the circuit, and hence the effect of decoherence.

One of the primary sources of noise in current quantum devices [ibm22] is two-qubit operations such as CNOT. These gates are ~ 100 times more likely to be erroneous than single qubit gates. However, the circuit realization of a level-p QAOA corresponding to the generalized Hamiltonian of Eq. (3.3) requires $2mp$ CNOT gates, where $m = |E|$. Theorem 3.4 prescribes the condition where the first CNOT gate of an $R_{Z_jZ_k}$ (Fig 3.1) is irrelevant to the effect of the said operator and hence may be removed while retaining functional equivalence.

Theorem 3.4 depicts that the proposed elimination of CNOT gates is applicable only when the initial state $|\psi_0\rangle$ is an equal superposition of n qubits, where n is the number of vertices in the graph. Since the $R_{Z_jZ_k}$ terms appear as products $\Pi_{j,k} exp(-i\gamma\frac{Q_{jk}}{4}Z_jZ_k)$, these operators can be arranged in any order. Thus, we have the choice of randomly ordering the edges (j,k) in order to apply the operator $R_{Z_jZ_k}$. For example, in Fig. 3.2, $R_{Z_0Z_1}$ precedes $R_{Z_1Z_2}$. But it would have been functionally equivalent if $R_{Z_1Z_2}$ preceded $R_{Z_0Z_1}$. However, from the above discussion, it follows that if we arbitrarily choose edges for applying the operator $R_{Z_jZ_k}$, then it cannot be guaranteed that a large number of edges will conform to Corollary 3.1. The requirement, in fact, imposes a precedence ordering among the edges.

Theorem 3.2: Criteria for CNOT elimination

Let $|\psi\rangle$ be an n-qubit state prepared in a uniform superposition (up to relative phase) overall basis states $|x_1, \ldots, x_n\rangle$ such that the relative phase on each basis state is a function of a subset $S \subset \{1, 2, ..., n\}$ of the n qubits (and independent of remaining qubits) *i.e.*

$$|\psi\rangle = \frac{1}{\sqrt{2^n}} \sum_{x_1,\ldots,x_n} e^{(i\phi(x_S))} |x_1, ..., x_n\rangle$$

where $x_S = \{x_i : i \in S\}$ and $\phi(x_S)$ depicts the relative phase of each superposition state. For any two qubits $|j\rangle$ and $|k\rangle$, where $|k\rangle \notin S$, and for the two operators $U_1 = CNOT_{jk}(I_j \otimes R_z(\theta_1)_k)CNOT_{jk}$ and $U_2 = (I_j \otimes R_z(\theta_1)_k)CNOT_{jk}$, we have

$$U_1 |\psi\rangle = U_2 |\psi\rangle.$$

Proof. See Appendix A.1. □

Corollary 3.1

Given a graph G, a $R_{z_j z_k}$ operator corresponding to an edge (j, k) can be optimized by replacing U_1 with U_2, provided that the target vertex does not occur in any of the edge operators applied earlier. In other words, the following conditions are sufficient to optimize an edge:-

1. if the vertex j is being operated on for the first time, then it acts as either a control or a target for the CNOT gate corresponding to the operator;
2. The vertex j does not act as a target of the CNOT gate if it occurs as a part of any other edge operator applied earlier in the circuit.

Proof. See Appendix A.2 □

For the rest of the thesis, an edge is said to be *optimized* if the operator U_2 can be applied for that edge instead of U_1 while retaining functional equivalence. This process of eliminating CNOT gates is termed as *ansatz optimization*. Sections 3.5 and 3.6 provide two methods based on Edge Coloring (EC) and Depth First Search (DFS) respectively to algorithmically remove CNOT gates while retaining functional equivalence.

3.5 Edge Coloring based Ansatz Optimization

The R_{zz} operators are highly prone to noise since they require 2 CNOT gates each for their implementation. Moreover, they can potentially contribute a lot to the depth of the circuit since at a given step of the circuit, only the edge operators corresponding to a vertex disjoint set of vertices can be applied parallelly. Thus the minimum depth of the circuit corresponds to the minimum value k such that the set of edges E can be partitioned as a disjoint union $\cup_i E_i$ where each subset E_i consists of a vertex disjoint collection of edges. This, in turn, corresponds to the edge coloring problem in a graph [Wes01].

3.2: Edge Coloring (EC) Problem

Given a graph $G = (V, E)$, color all the edges of the graph using the minimum number of colors such that no two adjacent edges have the same color.

Given a graph $G = (V, E)$ and a set of colors $\chi' = \{\chi'_1, \chi'_2, \ldots, \chi'_k\}$, an edge coloring [Wes01] assigns a color to each edge $e \in E$, such that any two adjacent edges (*i.e.*, edges incident on a common vertex) must be assigned distinct colors. The edge coloring problem comprises coloring the edges using the minimum number of colors k. The operators corresponding to edges having the same color can, therefore, be executed in parallel. Moreover,

1. the number of colors in optimal coloring, called the chromatic index, corresponds to the minimum depth of the circuit;

2. edges having the same color corresponds to the operators $R_{z_j z_k}$ that can be executed simultaneously.

Lemma 3.1

Applying the R_{zz} operators of the QAOA ansatz circuit according to the disjoint edge sets provided by the Edge Coloring Algorithm on the input graph leads to a circuit with minimum depth.

Proof. See Appendix A.3. □

Obtaining the optimal edge coloring for a given graph is an NP-complete problem [Wes01]. It is not practical to allocate exponential time to find the optimal edge-coloring as a pre-processing step for QAOA. Therefore, approximate polynomial time solutions to the problem are investigated. Vizing's Theorem states that every simple undirected graph can be edge-colored using at most $\Delta + 1$ colors, where Δ is the maximum degree of the graph [Viz64]. This is within an additive factor of 1 since any edge coloring must use at least Δ colors. Misra and Gries algorithm [MG92] achieves the above bound constructively in $\mathcal{O}(n \cdot m)$ time. Algorithm 1 below computes the sets of edges having the same color using the Misra and Gries algorithm as a subroutine. It returns the largest set S_{max} of edges having the same color in the coloring computed by the Misra and Gries algorithm.

Algorithm 1 Edge Coloring based Ansatz Optimization

Input: A graph $G = (V, E)$.
Output: Largest set S_{max} of edges having the same color.
1: Use the Misra and Gries algorithm to color the edges of the graph G.
2: $S_i \leftarrow$ set of edges having the same color i, $1 \leq i \leq \chi'$.
3: $S_{max} \leftarrow max\{S_1, S_2, \ldots, S_{\chi'}\}$.
4: Return S_{max}.

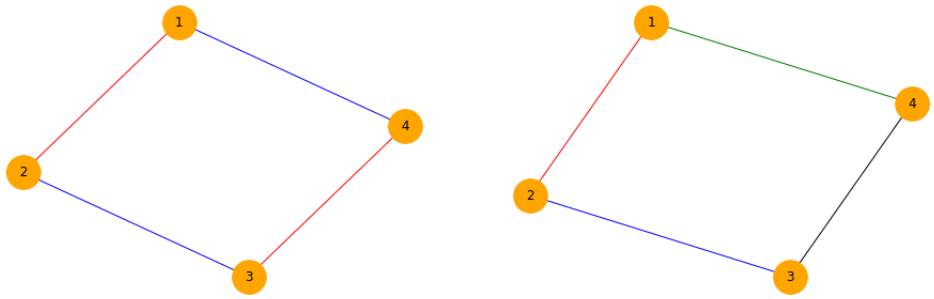
This EC approach provides the minimum depth achievable for QAOA ansatz using a polynomial time pre-processing. The operators corresponding to edges with the same color can be executed in parallel. Therefore, the operators corresponding to

the edges of S_{max} can be used as the first step of operators for maximum CNOT elimination. The other steps can be executed in any order.

Theorem 3.3

Every edge in the first step can be optimized according to Corollary 3.4.

Proof. See Appendix A.4. □

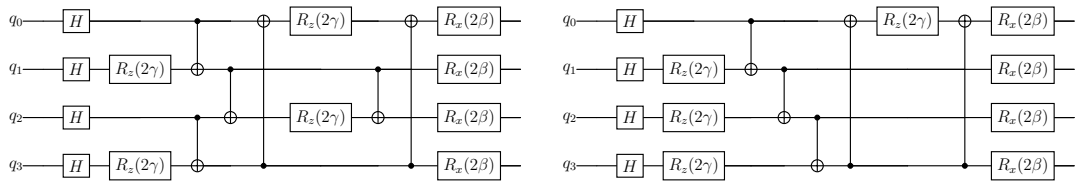


(a) Edge Coloring Based Optimization (b) Depth First Search Based Optimization

Figure 3.3: Depth of the QAOA ansatz circuit for Max-Cut when using (a) EC and (b) DFS-based method; edges having the same color can be executed simultaneously. The depth of the spanning tree in the DFS-based method is 4, compared to depth 2 for the EC-based method. However, the number of optimized edges in the EC-based method is 2, while that in the DFS-based method is 3.

A question then arises whether it is possible to optimize a few more edges in subsequent steps. It is, however, trivial to come up with examples where it cannot be so. Recall that an edge (i, j) in a graph corresponds to a term $\propto x_i x_j$ in the QUBO. For example, in the graph of Fig. 3.3 (a), if the red colored edges form the first step, and the blue colored edges form the second, then both the red edges can be optimized, but none of the blue edges can be done so. Therefore, there may be some scenarios where edges in subsequent steps can be optimized. But, in general, it is only the edges in the first step that can certainly be optimized. Since this edge coloring-based ansatz optimization method does not increase the depth of the circuit, it always leads to a more efficient circuit design than the traditional QAOA circuit with a depth reduced by 1 as the first level of CNOT is eliminated.

The figures and results are generated using the $p = 1$ QAOA circuit for the Max-Cut problem whose Hamiltonian is shown in Eq. (3.2). It is a special case of the generalized two-body interaction Hamiltonian with the one-body interaction coefficient $b_j = 0 \ \forall \ j$. Although the figures and results are generated for the Max-Cut problem only, the general idea, and hence the results, extend to all Hamiltonians of the form of Eq. (3.3).



(a) Optimized circuit by EC method (b) Optimized circuit by DFS method

Figure 3.4: Max-Cut QAOA ansatz with $p = 1$ corresponding to (a) EC and (b) DFS-based optimization. In (a), the first CNOT gates of the operators have been deleted. The operators corresponding to (q_1, q_2) and (q_3, q_0) act in parallel. In (b), the first CNOT gates of three operators have been deleted, but the depth has increased.

Fig. 3.3 shows a 2-regular graph with four vertices. In Fig. 3.3 (a), the depth of the circuit corresponding to the operator $exp(-i\gamma H_P)$ is 2; the edges of the same color can be operated on simultaneously. If the red (or blue) edges form the first layer, then those two edges are optimized. The circuit corresponding to EC based optimization method is shown in in Fig 3.4 (a).

3.5.1 Lower and upper bound on the number of optimized edges

If χ' be the chromatic index of a graph $G = (V, E)$, Misra and Gries Algorithm [MG92] can find a polynomial time coloring using at most $\Delta + 1$ colors, where Δ is the maximum degree of the graph. Therefore, on an average, $\lceil \frac{m}{\Delta+1} \rceil$ edges have the same color.

More precisely, two extreme cases arise: (i) the colors may be uniformly dis-

tributed, and the maximum number of edges having the same color is $\lceil \frac{m}{\Delta+1} \rceil$, or (ii) one of the colors is used dominantly for most of the edges. For all the edges adjacent to the same vertex, a particular color can be assigned to one of the edges only. Therefore, the dominant color can be used at most on $\lfloor \frac{n}{2} \rfloor$ edges, where $n = |V|$. Thus the possible number of optimized edges that can be obtained via the EC method is as shown in Eq. (3.8).

$$\lceil \frac{m}{\Delta+1} \rceil \leq \ \#\ Optimized\ Edges \leq \lfloor \frac{n}{2} \rfloor. \tag{3.8}$$

3.6 Depth First Search based Ansatz Optimization

The EC-based optimization method can eliminate $\leq \lfloor \frac{n}{2} \rfloor$ CNOT gates. The natural questions that follow are - (i) Do there exist other methods that can eliminate more CNOT gates while retaining functional equivalence, and (ii) what is the maximum possible number of CNOT gates that can be eliminated?

Theorem 3.4

Optimization of an ansatz for a QAOA for a Hamiltonian of the form of Eq. (3.3) with p=1, by deletion of the CNOT gate in the first level corresponding to an edge of the graph, can be done for no more than $n-1$ edges.

Proof. See A.6. □

Theorem 3.4 provides an answer to the second question. Corollary 3.1 provided two criteria for CNOT elimination. The EC method uses the first criteria where the edges in the first step are vertex disjoint, and therefore none of those vertices have been associated with any other CNOT gates previously. The second criterion, where a vertex is allowed to be associated with other CNOT gates as long as it does not act as a target for those CNOTs, leads to an ordering of the edges. If two CNOT gates acting on edges say (i, j) and (j, k), with the control on the first

qubit, and target on the second, then the edge (j, k) satisfies the criteria for CNOT elimination. This, in turn, refers to a spanning tree structure within the graph. A procedure for generating a spanning tree is the Depth First Search (DFS).

Algorithm 2, for obtaining the optimized QAOA ansatz, uses the standard DFS algorithm [CLRS09], by returning the tree edges or discovery edges forming the DFS tree. The corresponding QAOA circuit starts from the first vertex of the DFS tree; for every edge $e = (u, v)$ in the DFS tree, the vertex u is made the control and v the target for the CNOT gate and the edges are operated sequentially in the set E_{dfs} (the tree edges). Once every edge in the DFS tree has been operated on, the remaining edges can be executed in any order. In fact, the EC method can be used on the remaining edges to obtain the minimum depth of the circuit corresponding to these edges; CNOT gates however cannot be reduced any further.

Algorithm 2 DFS-based Ansatz Optimization

Input: A graph $G = (V, E)$.
Output: A list E_{dfs} of $n - 1$ edges.

1: $E_{dfs} = \{\}$
2: $u \leftarrow$ randomly selected vertex from V.
3: Start DFS from the vertex u. For every vertex v discovered from its predecessor v', $E_{dfs} = E_{dfs} \cup (v', v)$.
4: Return E_{dfs}.

Theorem 3.5

Each edge in the DFS tree can be optimized according to Corollary 3.4.

Proof. See A.5. □

For a graph with n vertices, the rooted spanning tree generated using DFS has $n - 1$ edges. From Theorem 3.5, the DFS-based optimization method provides $n - 1$ optimized edges, *i.e.*, a reduction in the number of CNOT gates by $n - 1$. Theorem 3.4 asserted that it is not possible to eliminate more than $n - 1$ edges. Thus, the DFS method is optimal in the number of optimized edges. Fig 3.3 (b) shows the edges corresponding to the DFS tree. If the root of the tree is vertex

3, then the edges in the DFS tree, which can be optimized, are $(3,2),(2,1)$ and $(1,4)$. The last edge $(4,1)$ cannot be optimized. Fig 3.4 (b) shows the $p = 1$ QAOA circuit for Max-Cut corresponding to the graph in Fig 3.3 (b), optimized using the DFS method.

Since the DFS method can eliminate almost twice as many CNOT gates as the EC method, it is expected that the resultant circuit, optimized using the DFS method, will be less noisy. If $|\psi\rangle$ is the state obtained from the noise-free (ideal) computation of the QAOA ansatz circuit, and the state obtained from noisy computation is $|\psi_e\rangle$, then the probability of success of the noisy computation, then, is defined as

$$P_{success} = |\langle\psi|\psi_e\rangle|^2 \tag{3.9}$$

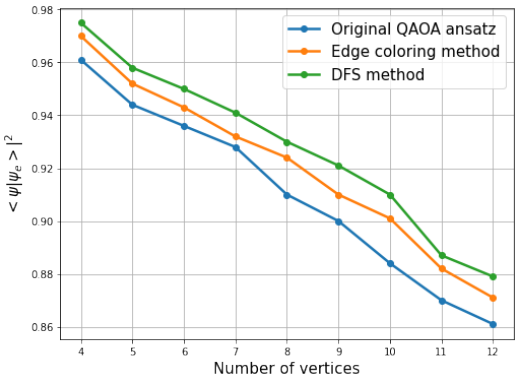
For a noise-free circuit, $P_{success} = 1$. The value lowers from 1 and approaches 0 as the noise in the system increases. Furthermore, executing a quantum circuit in real hardware has its facets. The circuit is usually not executed as it is in the hardware. It undergoes a process called transpilation in which

(i) the gates of the circuit are replaced with one, or a sequence of, basis gates which are actually executed in the quantum hardware. The basis gates of the IBM Quantum devices are $\{CNOT/ECR, SX, X, R_z$ and Identity$\}$ [ibm22],

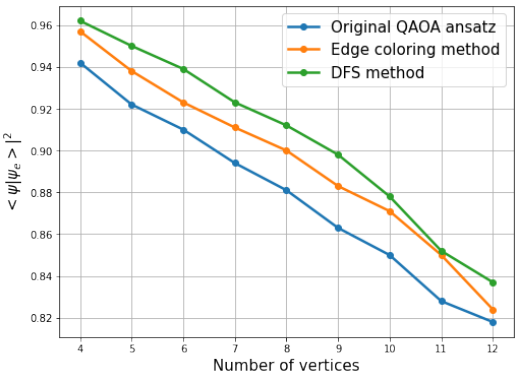
(ii) the circuit is mapped to the underlying connectivity (called the coupling map) of the hardware [BBMR20],

(iii) the number of gates in the circuit is reduced using logical equivalence [BW20b].

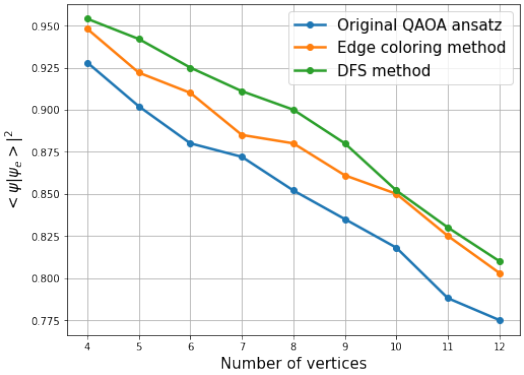
Transpilation can introduce SWAP gates in order to map the circuit to the hardware connectivity. Each SWAP gate is decomposed into three CNOT gates [NC02]. However, both of these methods are hardware-independent and any facet corresponding to the hardware does not affect the CNOT elimination. Fig. 3.5(a) - (d) shows the $P_{success}$ of the traditional QAOA ansatz [FGG14], EC based and the DFS based optimization methods for Erdos-Renyi graphs, where p_{edge} varies from



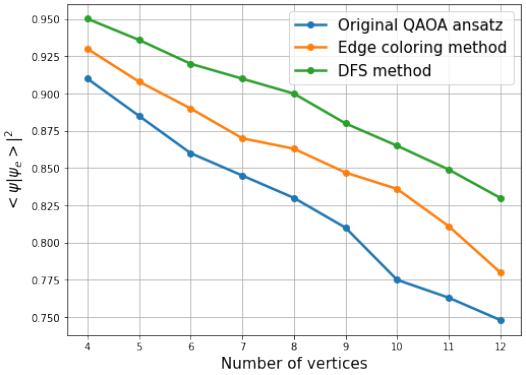
(a) Erdos-Renyi graphs with $p_{edge} = 0.4$



(b) Erdos-Renyi graphs with $p_{edge} = 0.6$



(c) Erdos-Renyi graphs with $p_{edge} = 0.8$



(d) Complete graphs

Figure 3.5: $|\langle\psi|\psi_e\rangle|^2$ for graphs of various sparsity: Erdos Renyi graphs ($p_{edge} = 0.4,\ 0.6,\ 0.8$) and complete graphs

0.4 to 1 (complete graph). It is evident that the circuits generated using the DFS method have a higher probability of success than the circuits generated using the EC method, which, in turn, have a higher success probability than the traditional QAOA circuits. Erdos-Renyi graphs with varying p_{edge} ensure that the methods are independent of the sparsity of the graph as well. Each of the values is averaged over 80 input graph instances, and each instance is an average of 2048 shots of the noisy circuits by the simulator model for *ibmq_manhattan*, which is a 65 qubit device from IBM Quantum [ibm22].

Theorem 3.6 asserts that the DFS method eliminates $n-1$ CNOT gates. However, the exact number of gates eliminated using the EC method is not predefined; rather

Table 3.1: # CNOT gates in Max-Cut QAOA ansatz post transpilation on *ibmq_manhattan* using (i) Traditional, (ii) Edge Coloring (EC), and (iii) DFS based optimization

Graph Family	# qubits (vertices)	# CNOT gates in Max-Cut QAOA ansatz circuit				
		Traditional	EC		DFS	
			# gates	% reduction	# gates	% reduction
Complete graph	10	90	85	5.5	81	10
	20	380	370	2.6	361	5
	30	870	855	1.7	841	3.3
	40	1560	1540	1.3	1521	2.5
	50	2450	2425	1	2401	2
	60	3540	3510	0.8	3481	1.6
Erdos-Renyi ($p_{edge} = 0.8$)	10	70	66	5.7	61	12.8
	20	302	292	3.3	283	6.3
	30	698	683	2.1	669	4.2
	40	1216	1197	1.6	1177	3.2
	50	1956	1931	1.3	1907	2.5
	60	2822	2792	1	2763	2.1
Erdos-Renyi ($p_{edge} = 0.6$)	10	50	46	8	41	18
	20	234	225	3.8	215	8.1
	30	504	491	2.6	475	5.8
	40	960	940	2.1	921	4.1
	50	1504	1479	1.7	1455	3.3
	60	2114	2085	1.4	2055	2.8
Erdos-Renyi ($p_{edge} = 0.4$)	10	36	31	13.9	27	25
	20	164	154	6.1	145	11.6
	30	362	348	3.9	333	8
	40	586	566	3.4	547	6.7
	50	950	925	2.6	901	5.2
	60	1468	1440	2	1409	4

it falls within the range of Eq. (3.8). Table 3.1 shows the CNOT count in the post transpilation (optimization_level=3) circuit [qis22] for the *ibmq_manhattan* device as the number of vertices varies from 10 to 60 for each of the graph families considered. It can be noted that the percentage reduction in the number of CNOT gates obtained by the DFS method is $\sim 2\times$ that of the EC method for all graph sizes and families.

3.6.1 Increase in depth vs CNOT elimination

The plots in Fig 3.5 assert that the DFS method outperforms the EC method in the probability of success. However, it is evident from Fig 3.3 and 3.4 that the DFS-based optimization can lead to an increase in the depth of the circuit.

In that figure, the depth of the circuit using EC and DFS methods are 2 and 4 respectively, while the number of CNOT gates eliminated is respectively 2 and 3. This mandates an investigation of the criteria for which the increase in depth is overshadowed by the number of CNOT gates eliminated, and the resulting circuit has a higher success probability.

In IBM Quantum Hardware, Rz gate is executed virtually [MWS+17a]. Therefore, the error in the R_{zz} operator arises from the CNOT gates only. Moreover, an increase in depth makes the circuit more prone to Amplitude Damping noise [NC02]. Amplitude Damping noise is characterized by the time duration t and the relaxation time T_1 of the hardware. The probability of obtaining a state $|q\rangle$, $q \in \{0, 1\}$, after time t is $exp(-\frac{t}{T_1})$.

Let the time duration and the error probability of each CNOT gate be t_{cx} and p_{cx} respectively. Let there be N levels of CNOT operations.[1] The time duration for multiple CNOT gates operating in parallel at each level is still t_{cx}. The probability of no error after N levels of operations, considering only relaxation error, is $exp\left(-\frac{Nt_{cx}}{T_1}\right)$. Let k be the number of CNOT gates in the original ansatz circuit. Therefore, the probability of no error after the operation of the CNOT gates, considering only CNOT gate error, is $(1-p_{cx})^k$. Under the assumption that relaxation and noisy CNOT gates are the only sources of error, Eq. (3.10) gives the probability of success after a single iteration of the QAOA ansatz.

$$P_{success} = (1 - p_{cx})^k \cdot exp\left(-\frac{Nt_{cx}}{T_1}\right) \tag{3.10}$$

Let after the optimization using the DFS-based method, k_1 CNOT gates have been reduced leading to an increase in N_1 levels of operations. The probability that this optimized circuit remains error-free is given in Eq. (3.11).

$$P^{opt}_{success} = (1 - p_{cx})^{(k-k_1)} \cdot exp\left(-\frac{(N + N_1)t_{cx}}{T_1}\right) \tag{3.11}$$

[1] A *level* is not the same as a *step*. A single *step* consists of two CNOT gates (or one if the edge is optimized) and one R_z gate. So each step consists of three (or two for optimized edge) levels.

The optimization is fruitful only when $P^{opt}_{success} \geq P_{success}$. Since $P^{opt}_{success} = P_{success} \cdot exp\left(-\frac{N_1 t_{cx}}{T_1}\right)/(1-p_{cx})^{k_1}$ and both $P^{opt}_{success}$ and $P_{success} \leq 1$, the required inequality holds only if $exp\left(-\frac{N_1 t_{cx}}{T_1}\right)/(1-p_{cx})^{k_1} \geq 1$.

$$\begin{aligned} exp\left(-\frac{N_1 t_{cx}}{T_1}\right) &\geq (1-p_{cx})^{k_1} \\ \Rightarrow N_1 &\leq \lambda \times k_1 \end{aligned} \tag{3.12}$$

where $\lambda = \left(\frac{-ln(1-p_{cx})\cdot T_1}{t_{cx}}\right)$ is defined in terms of parameters specific to the quantum device.

If the DFS-based method is not applied, then the number of steps is equal to the number of color classes (as in the EC method) +1 for a layer of R_z rotation gates corresponding to Z_i operators in H_P (Eq. (3.3)). The maximum number of color classes is $\Delta + 1$, and hence the number of levels of the circuit is $2\Delta + 2$ (the first step has one level of CNOT gates less than the others). When the DFS method is applied, the circuit can be divided into two disjoint sets of edges:

(i) Edges belonging to the DFS tree which can be optimized. The number of steps of this portion of the circuit is at most $n-1$ (*i.e.*, the depth of the DFS tree). Each of the operators corresponding to these edges contains a single CNOT gate only, leading to $n-1$ CNOT gates.

(ii) Edges that do not belong to the DFS tree and hence are not optimized. The operators corresponding to these edges can be applied in any order but after all the optimized edges. When removing the edges of the DFS tree, the degree of each vertex is reduced by at least 1. Therefore, the maximum degree of the remaining subgraph is at most $\Delta - 1$. Thus the number of steps for this portion of the circuit will be at most Δ (From Misra and Gries Algorithm). Each of the steps in this portion contains 2 CNOT gates, and hence the number levels of CNOT gates is 2Δ.

Therefore, the maximum number of steps of the circuit after applying the DFS-

based optimization is $n-1+2\Delta+1$, with the $+1$ accounting for a layer of R_z gates corresponding to Z_i operators in H_P. The maximum increase in levels due to this method is given by Eq. (3.13).

$$n-1+2\Delta+1-(2\Delta+2)=n-2 \tag{3.13}$$

The number of CNOT gates eliminated due to the DFS method is always $n-1$. Therefore, from Eq. (3.12) and (3.13), we get

$$\begin{aligned} n-2 &\leq \lambda\cdot(n-1) \\ \Rightarrow \lambda &\geq \frac{n-2}{n-1} \end{aligned} \tag{3.14}$$

Table 3.2 shows the average value of λ for some IBM Quantum [ibm22] devices, ranging from the comparatively more noisy *ibmq_melbourne* (although this device is no longer in action) to the latest *ibm_washington* having the 127 qubits *eagle* processor. The lower bound on λ by Eq. (3.14) is $\frac{n-2}{n-1}$, which, in the asymptotic limit, $\frac{n-2}{n-1} \to 1$. Thus, the proposed DFS-based optimization method leads to a lower error probability on any quantum device for which $\lambda \geq 1$. Table 3.2 readily shows that the IBM Quantum devices satisfy this requirement. Moreover, $\lambda = \left(\frac{-ln(1-p_{cx})\cdot T_1}{t_{cx}}\right)$. As the quantum systems evolve, it can be expected to have higher values of T_1, and lower values for p_{cx} and t_{cx}. This expectation ensures that the value of λ will increase with improved hardware. Therefore, it can be expected that the proposed optimization will hold in future hardware also.

Table 3.2: Average value of λ for four IBM Quantum machines [ibm22]

IBM Quantum devices	Avg value of λ
ibm_washington	13.78
ibmq_manhattan	3.6
ibmq_montreal	2.47
ibmq_sydney	3.35
ibmq_melbourne	2.03

3.7 Summary

This chapter provides two methods to lower the number of CNOT gates for QAOA circuit design for two-body Hamiltonian problems while maintaining functional equivalence. The edge coloring method can eliminate at most $\lfloor \frac{n}{2} \rfloor$ CNOT gates, whereas the DFS method can eliminate $n-1$ CNOT gates, with the latter being proved to be optimal. The DFS method, however, can end up increasing the depth of the circuit since it imposes an ordering on the execution of the CNOT gates. An analytical inequality was derived such that the removal of noise due to CNOT elimination overshadows the increase in noise due to an increase in depth. For some IBM Quantum devices, it was shown that this inequality is satisfied, i.e., the optimized QAOA leads to lower noise. An argument was made on why it is expected that this inequality will hold true in future devices too.

Although the DFS method is sufficient to lower the noise in the QAOA circuit, a general hope is to restrict the increase in depth of the circuit while still retaining the optimal CNOT elimination. A graph can have multiple DFS trees – not all of them have the same height. If one can choose spanning trees of lower height, then it might be possible to achieve the hope. The next chapter deals with a heuristic approach that can achieve this by finding optimal spanning trees from the problem graph.

CHAPTER 4

Greedy approach to QAOA circuit optimization

Contents

4.1 Introduction

Chapter 3 discussed Edge Coloring (EC) and Depth First Search (DFS) methods for lowering the number of CNOT gates in the QAOA circuit corresponding to a Hamiltonian of the form of Eq. (3.3), where n is the number of variables (or the number of vertices in the graph corresponding to the Hamiltonian). The former method can eliminate up to $\lfloor \frac{n}{2} \rfloor$ CNOT gates, while the latter can eliminate $n-1$ CNOT gates, which was shown to be optimal. CNOT gates being one of the primary sources of error in modern quantum devices [ibm22], these methods significantly reduce the noise in the circuit. The DFS method, although optimal, imposes an ordering of the edges, leading to an increase in the depth of the circuit. However, an analytical criterion was discussed for which the elimination of the CNOT gate overshadows the increase in depth, and the resulting circuit has a lower probability of error. It was shown that all the current quantum hardware conforms to the criteria, and hence the DFS method always leads to lower error probability.

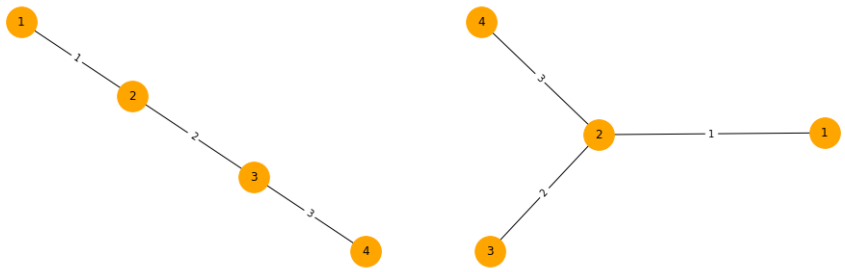
This chapter explores the fact that a graph can have multiple DFS trees that vary in height. The maximum height of a DFS tree for an n vertex graph is $n-1$. But there exists other DFS trees (or other forms of rooted spanning tree) having lower height which may result in circuits of lower depth. A circuit with lower depth is naturally preferable since it lowers the effect of decoherence while still retaining the optimal CNOT elimination. It is, however, not a trivial task to find a DFS tree for a given graph that is guaranteed to provide a low-depth ansatz circuit.

This chapter formally formulates the problem of finding a depth-optimized rooted spanning tree for a given graph and provides a heuristic method to find such a spanning tree. The edge coloring, DFS (Chapter 3), and the heuristic method are all hardware-independent. This chapter also proposes a method to modify the

heuristic function to make it hardware-friendly and reduce some swap gates if the coupling map and the initial layout are known.

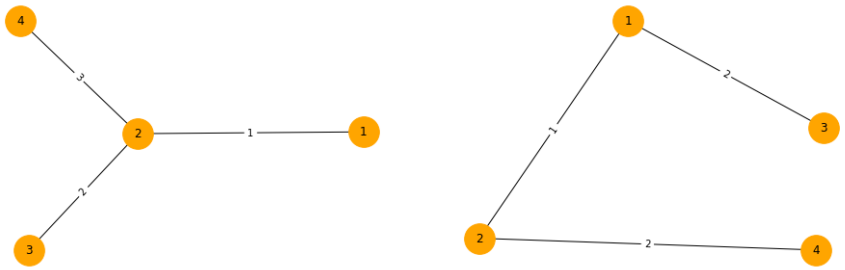
4.2 Motivation for a heuristic algorithm

The maximum height of a DFS tree with n vertices is $n-1$, and so is the depth of the corresponding circuit. However, Corollary 3.4 required some ordered set of edges forming a rooted spanning tree, and not necessarily a DFS tree. Therefore, the general notion would be to look into other rooted spanning trees for a graph that can arrest the increase in the depth of the QAOA circuit. This creates an apparent illusion that a rooted spanning tree with a lower height will always lead to a circuit with lower depth. If that were indeed always the case, then a Breadth First Search (BFS) tree should provide a spanning tree with minimum height. However, that is not always the case as depicted in Fig. 4.1. The two trees in Fig 4.1 (a) and (b) have different heights but result in circuits with the same depth (Fig 4.3). In both figures, the values associated with the edges depict the level at which the operator corresponding to that particular edge can be operated so that the optimization (i.e. elimination of CNOT gates) holds.



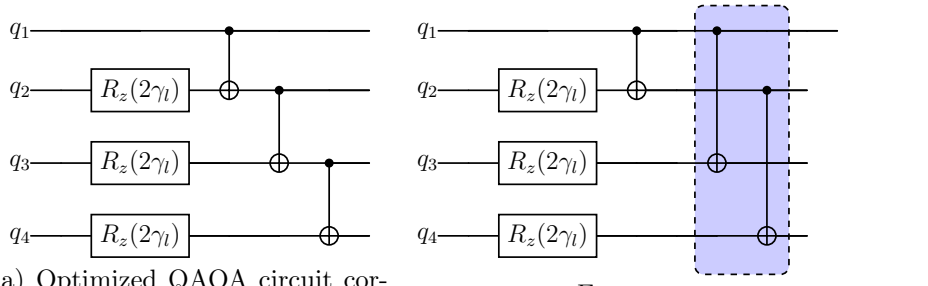
(a) A spanning tree of height 3 (b) A spanning tree of height 2

Figure 4.1: Two trees with different heights – the integer label on an edge is the step at which the operator $R_{z_j z_k}$ for edge (j,k) can be operated on. The maximum value of these labels is the depth of the circuit. The heights of the trees in subfigures (a) and (b) are 3 and 2 respectively. However, both of them lead to the same circuit shown in Fig 2.2 (a)



(a) A spanning tree of height 2 (b) Another spanning tree of height 2

Figure 4.2: Two trees with the same height but the number of steps of the circuit corresponding to the tree in subfigure (a) is 3, while that corresponding to the tree in subfigure (b) is 2.



(a) Optimized QAOA circuit corresponding to both the trees in Fig 4.1

(b) Optimized QAOA circuit corresponding to the tree in Fig 4.2 (b)

Figure 4.3: The quantum circuit of $U(H_P, \gamma)$ for Max-Cut QAOA ansatz corresponding to (a) both the trees in Fig 4.1 and (b) the tree in Fig 4.2 (b)

From the labels on the edges of the tree in Fig 4.1 (a), the operators $R_{z_1z_2}$, $R_{z_2z_3}$ and $R_{z_3z_4}$ are operated in this order. Although the tree in Fig 4.1 (b) has a lower height, it cannot result in a circuit with lower depth; here $R_{z_2z_3}, R_{z_2z_4}$ must be executed sequentially since they require access to the common qubit corresponding to vertex 2. On the other hand, both the trees in Fig 4.2 have the same height, but they result in two circuits of different depths. This is because in Fig 4.2 (b), if $R_{z_1z_2}$ is executed first, then $R_{z_1z_3}$ and $R_{z_2z_4}$ can be executed simultaneously. The circuits corresponding to the $U(H_P)$ operators for the two trees in Fig 4.2 are shown in Fig 4.3 (a) and (b) respectively.

4.2.1 Conjecture: Finding the rooted spanning tree that results in a circuit with minimum depth is NP-Complete

The examples in Fig. 4.1 and 4.2 show that simply finding a rooted spanning tree with minimum height is not sufficient to reduce the depth of the corresponding circuit. Rather, it is necessary to find a rooted spanning tree where multiple edges can be executed in parallel. This is similar to the Edge Coloring problem [Wes01], and was explored in Section 3.5. However, the problem here is more constrained than edge coloring. In a tree, the same color can be used for edges in alternate levels, i.e., it is possible to have edges of the same color in odd indexed levels 1, 3, ..., and in even indexed levels 2, 4, ... For example, in Fig. 4.1 (a), optimal edge coloring assigns the same color to the edges in levels 1 and 3 since they are disjoint. However, it was already discussed that operators corresponding to these two edges cannot execute in parallel. Therefore, we have an added constraint that an edge cannot be given the color of any of its ancestors. Therefore, a formal definition of the problem at hand is as follows:

Problem 1: Given a graph $G = (V, E)$, starting from a root vertex r, find a spanning tree T of G whose edges can be colored with the minimum number of colors satisfying the conditions that

i) no two edges incident on a common vertex have the same color; and

ii) no edge has the same color as that of any of its ancestors.

Optimal edge coloring itself is an NP-complete problem (although not for a tree), and the problem here at hand has additional constraints to it. In [RK05], the authors showed that finding a degree-constrained spanning tree, i.e., a spanning tree where the degree of any vertex is upper bounded by a predefined value, is NP-Complete. Problem 1 can be informally rephrased as finding a rooted spanning tree where the degree of the vertices is not too large to avoid a large number of edges on the same level, and yet not too small so that the number of levels is large.

Therefore, it is *conjectured* that finding the rooted spanning tree resulting in a minimum depth circuit is NP-complete. In this section, a greedy polynomial time algorithm is proposed to find a *better* solution instead of the vanilla flavor depth first search based method from Chapter 3.

4.2.2 Proposed cost function for the heuristic

First, a few terms are defined to clarify the requirements for a rooted spanning tree that would lead to a circuit with lower depth.

1. **Branching factor**: The branching factor of a vertex v is defined as the number of vertices that have been discovered in the rooted spanning tree from v.

   In other words, the branching factor of a vertex v is one less than the degree of that vertex in the spanning tree except for the root vertex, whose branching factor is equal to its degree. For example, in the tree of Fig. 4.1 (b), starting from the root labeled 1, the branching factor of the root is 1, that of vertex 2 is 2, and that of the leaf vertices is 0.

2. **Level**: If a vertex v is discovered in the rooted spanning tree from a vertex u, then the level of vertex v = level of vertex u + 1. The level of the root vertex is 0.

   The definition of level is essentially the same as that for BFS.

3. **Delayed start**: Delayed start is defined as the phenomenon where the vertices $v_1, \ldots v_k$ are discovered in the spanning tree from the same vertex v, and belong to the same level, but the edges $(v, v_1), \ldots (v, v_k)$ have to be operated on sequentially. This is because the adjacent edges share a common vertex, and simultaneous CNOT operations are not possible with a common control or target qubit. Therefore, the operation corresponding to the edge (v, v_i) is delayed as long as all the operations corresponding to the edges

$(v, v_j), 1 \leq j < i$ are not completed. The operator $R_{z_v z_{v_i}}$ can be operated earliest at the level $level(v) + i$.

An example of delayed start is observed in the tree of Fig. 4.1 (b). Although both the leaf vertices in that tree are at the same level, they cannot be operated on simultaneously. Therefore, they must be operated on two disjoint levels. Indeed delayed start is the reason that the depth of the circuit does not reduce directly with the height of the tree. A tree with a higher branching factor can experience more delayed start than a tree with a lower branching factor. On the other hand, the height of the tree increases with decreasing branching factor. Therefore, the objective is to find a rooted spanning tree that minimizes the two contrasting requirements - (i) the height of the tree, and (ii) the number of delayed starts.

Since the problem is expected to be NP-complete, it is natural to look for heuristic algorithms that optimize some cost functions. The criteria for designing a cost function can be summarized as follows:

- If the branching factors of the vertices are very high, i.e., close to the degree of the vertex, then the corresponding circuit will suffer from *delayed start*, leading to an increase in the depth. On the other hand, if the branching factor of the vertices is very low such as 1 or 2, then the height of the tree, and hence the depth of the circuit, will increase.
- Between two vertices u and v, it is better to branch the one at a lower level of the tree so that the edges in that branch may still have some opportunity to be executed in parallel with other edges at a higher level even after *delayed start*. An example of this is shown in Fig. 4.2 where both the trees have the same height, but the tree of Fig. 4.2 (b) will lead to a circuit with lower depth since the branching is closer to the root.

Respecting all the three criteria stated above, a cost function C_v, to be associated with every vertex v, is proposed here. Let n be the number of vertices in the

graph, l_v and v_{bf} be the level and the current branching factor of the vertex v respectively, and B be the maximum branching factor decided for any vertex in the spanning tree, then

$$C_v = (n - l_v) \cdot (B - v_{bf}) \tag{4.1}$$

When growing the spanning tree from a root vertex, the edge (v, w) for which the cost function C_v is maximum, is added to the tree. Note here that for a new edge (v, w), the cost function does not depend on the vertex w, but rather on the vertex v from which this edge is discovered (the algorithm is provided later on).

The term $(n - l_v)$ is always positive, since l_v starts from 0 and can go up to $n - 1$ at most (for a tree). On the other hand,

$$B - v_{bf} \begin{cases} > 0 & \text{if } v_{bf} < B \\ = 0 & \text{if } v_{bf} = B \\ < 0 & \text{if } v_{bf} > B. \end{cases} \tag{4.2}$$

Hence, an algorithm that maximizes this cost function should, in principle, (i) avoid branching lower down the tree, and (ii) avoid exceeding the maximum branching factor B for any vertex.

Algorithm 3 avoids branching at a vertex v for which $v_{bf} \geq B$. When $v_{bf} = B$, the cost function has a contribution of 0. Therefore, when generating the results, if the required maximum branching factor for the spanning tree is f, then B is taken to be $f + 1$. Furthermore, the term $(n - l_v)$ is higher for the vertices with lower l_v. Thus, if for two vertices $u \neq v$, $v_{bf} = u_{bf} < B$, the algorithm chooses to branch that vertex that has a lower level. This ensures that *delayed start* is closer to the root so that those branches still have some opportunity for parallel execution with some higher-level branches. Furthermore, if $v_{bf} > B$, the product of $(B - v_{bf})$ with $(n - l_v)$ leads to significantly low values for low l_v. This discourages branching more than B in lower levels of the tree strongly to prevent excessive *delayed start* (like in a BFS tree). In other words, the spanning tree generated by this heuristic

cost function is neither a BFS nor a DFS one, but rather an intermediate one.

Figure 4.4: Two spanning trees of the graph in Fig. 4.5 – (a) generated using the DFS method (Algorithm 2), (b) generated using the greedy heuristic method (Algorithm 3) with $B = 3$.

Theorem 4.1: Runtime of heuristic algorithm

Algorithm 3 finds a rooted spanning tree for a graph with n vertices and maximum degree Δ which satisfies the conditions in Problem 1 in $\mathcal{O}(\Delta \cdot n^2)$ time.

Proof. See A.7 □

For sparse graphs, $\Delta = \mathcal{O}(1)$ and for dense graphs $\Delta = \mathcal{O}(n)$. Therefore, the time complexity of the proposed Algorithm 3 varies between $\mathcal{O}(n^2)$ to $\mathcal{O}(n^3)$ depending on the sparsity of the given graph.

4.2.3 An illustration of Algorithm 3

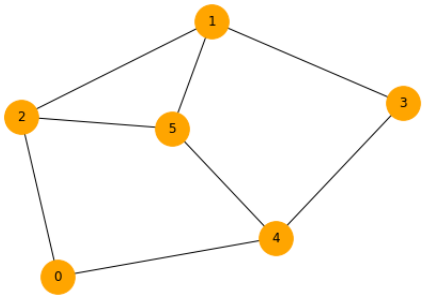
This subsection illustrates the DFS method in Algorithm 2 and the greedy heuristic method in Algorithm 3 in action on an example graph given in Fig. 4.5 (a). First, Fig. 4.5 (b) shows the traditional $p = 1$ QAOA circuit for Max-Cut for this graph. Then, Fig. 4.4 gives two spanning trees of the graph. The spanning tree in Fig. 4.4 (a) is generated using Algorithm 2 whereas the one in Fig. 4.4 (b) is generated

Algorithm 3 Cost function based rooted spanning tree generation

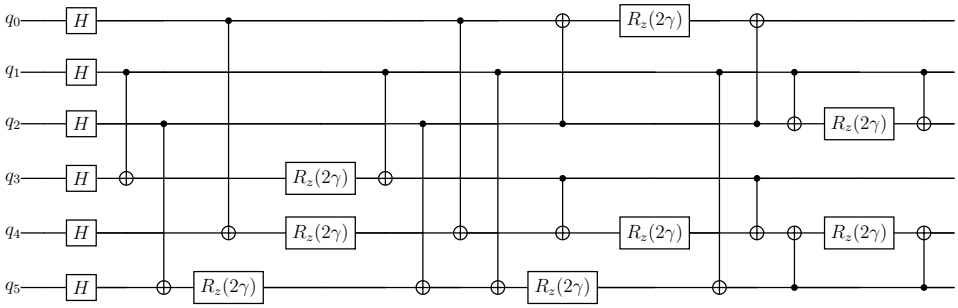
Input: A Graph $G = (V, E)$, $|V| = n$, $|E| = m$; maximum branching factor B.
Output: A Rooted Spanning Tree T of the Graph G.

$T = \{\}$.
$u_{bf} \leftarrow 0$ for all vertex u
$r \leftarrow$ randomly selected start vertex.
Visited = $\{r\}$; $r_{bf} = r_{bf} + 1$.
edges_to_add = neigh(r).
while $|Visited| < n$ **do**
  e = edges_to_add[0]; c = 0.
  for all $edge = (u, v) \in edges\_to\_add$ **do**
    cost = $(n - l_u) \cdot (B - u_{bf})$.
    if $cost > c$ **then**
      c = cost; e = edge.
    end if
  end for
  $T = T \cup \{e\}$.
  Visited = Visited $\cup \{y\}$, where $e = (x, y)$.
  $x_{bf} = x_{bf} + 1$.
  Remove all edges of the form $(*, q)$ from edges_to_add.
  for all $edge = (p, q) \in neigh(y)$ **do**
    if $q \notin Visited$ **then**
      edges_to_add = edges_to_add $\cup \{edge\}$.
    end if
  end for
end while

using Algorithm 3 with $B = 3$. Fig. 4.6 (a) and (b) show the optimized circuits for the $p = 1$ QAOA for Max-Cut for the graph in Fig. 4.5(a), where the optimized circuits are generated using the Algorithms 2 and 3 respectively.



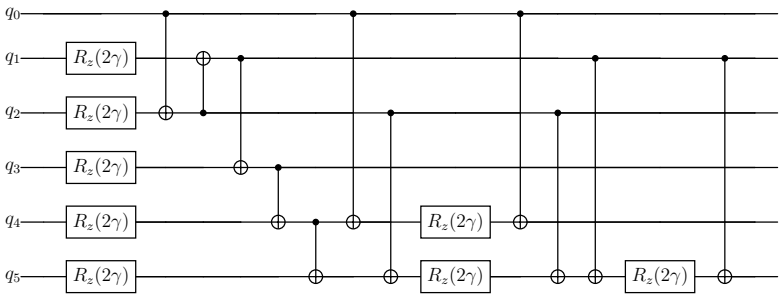
(a) An example graph with 6 vertices



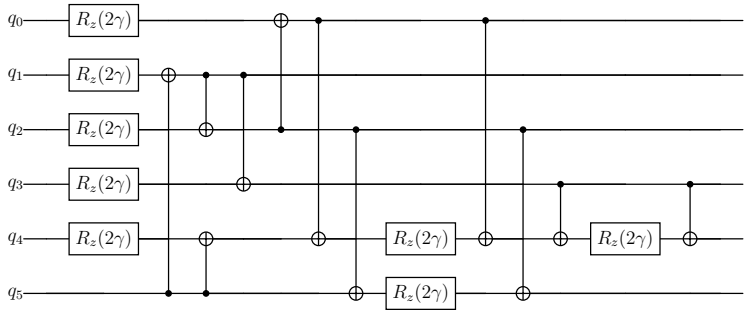
(b) Traditional $p = 1$ QAOA circuit for Max-Cut

Figure 4.5: The traditional $p = 1$ QAOA circuit for Max-Cut corresponding to $U(H_P, \gamma)$ for an example graph with 6 vertices. The values of the parameters are chosen randomly.

The depth of the *entire circuits* in Figs. 4.5(b), 4.6 (a) and 4.6 (b) are 11, 14 and 12 respectively. The number of CNOT gates in both the optimized circuits in Fig. 4.6 is 5 less than that in Fig. 4.5 (b). Note that the depth of both the optimized circuits is greater than that of the traditional QAOA for Max-Cut. However, the optimized circuit in Fig. 4.6 (b) can be considered to be superior since it requires 5 CNOT gates fewer than that in Fig. 4.5 (b), with an increase of depth by 1 only. Algorithm 3 can significantly arrest the increase in depth, and in certain cases can lead to a QAOA circuit with a depth lower than its traditional circuit.



(a) Optimized circuit using DFS method



(b) Optimized circuit using greedy heuristic method

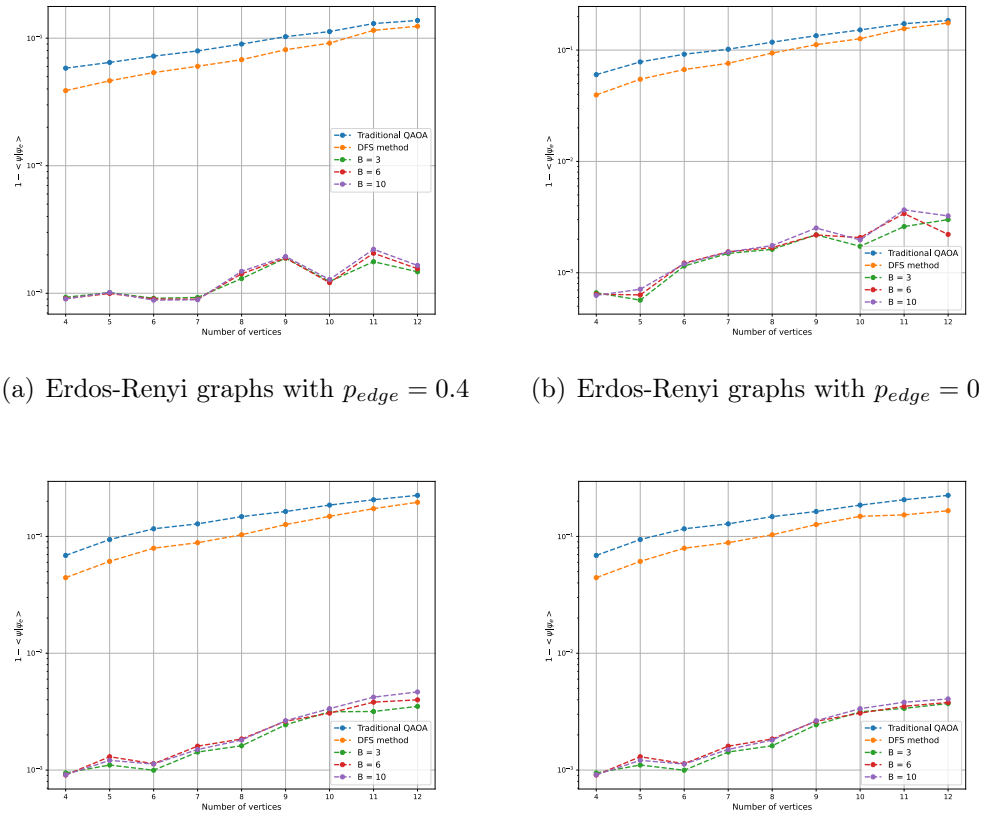
Figure 4.6: Optimized circuit for $U(H_P, \gamma)$ of $p = 1$ QAOA for Max-Cut corresponding to the two spanning trees in Fig. 4.4 respectively.

4.3 Simulation results

4.3.1 Increase in probability of success

QAOA consists of executing the same circuit with the same parameters multiple times to obtain an expectation value of the cut. The performance of the algorithm is determined by this expectation value of the obtained cut. Since our optimized QAOA circuit is functionally equivalent to the traditional QAOA circuit, the performance remains unchanged in the ideal noiseless scenario. For each iteration of the algorithm, let $|\psi\rangle$ denote the ideal state vector obtained via noiseless simulation. As real-world quantum devices are noisy, let $|\psi_e\rangle$ denote the noisy outcome obtained via noisy simulation. The probability of success is defined as $P_{success} = |\langle\psi|\psi_e\rangle|^2$. For a graph with n vertices, the optimization obtained by

Algorithm 2 reduced the number of CNOT gates by $n-1$. Algorithm 3 retains the $n-1$ reduction in the number of CNOT gates needed for the operator $U(H_P)$ in the ansatz and also arrests the increase in depth to a bare minimum (shown in the next subsection). This leads to a further improvement in $P_{success}$.



(a) Erdos-Renyi graphs with $p_{edge} = 0.4$

(b) Erdos-Renyi graphs with $p_{edge} = 0.6$

(c) Erdos-Renyi graphs with $p_{edge} = 0.8$

(d) Complete graphs

Figure 4.7: $1-P_{success}$ for Erdos Renyi Graphs ($p_{edge} = 0.4,\ 0.6,\ 0.8$) and complete graphs

The entire circuit of $U(H_P, \gamma)$ can be divided into two disjoint parts - one corresponding to the edges in the spanning tree, followed by the other edges in the input graph. Algorithm 3 can reduce the depth of the circuit corresponding to the spanning tree only. The circuit for the unoptimized edges remains the same as in Algorithms 1 and 2. Furthermore, the initialization and the Mixer Hamiltonian remain unchanged from the original QAOA design [FGG14]. Therefore, here only

the depth of the circuit corresponding to the spanning tree is compared.

When executing a circuit on hardware, the graph has to be mapped to the underlying hardware connectivity graph. This process is called transpilation. All the results in this section are generated after transpiling the original circuit in the *ibmq_manhattan* connectivity graph using the *transpilation* procedure of qiskit [H+19] with *optimization_level = 3*.

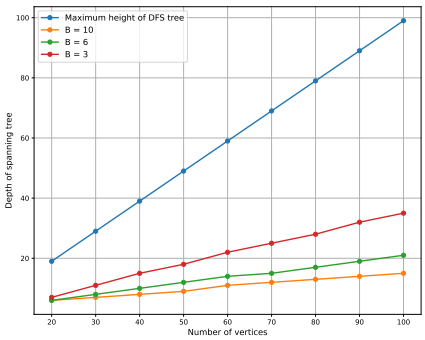
4.3.2 Reduction in the depth of the circuit

Table 4.1: Variation in the slope of the increase in depth with n for different values of B

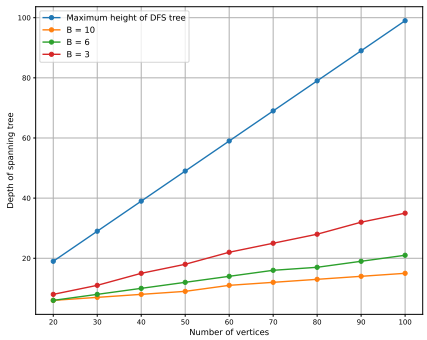
Graph Family	$B = 3$	$B = 6$	$B = 10$
Erdos Renyi ($p_{edge} = 0.4$)	0.35	0.1875	0.1125
Erdos Renyi ($p_{edge} = 0.6$)	0.35	0.1875	0.1125
Erdos Renyi ($p_{edge} = 0.8$)	0.3375	0.1875	0.125
Complete graph	0.3375	0.1875	0.125

In the worst case, the height of the DFS tree, and hence the depth of the corresponding circuit, can be as large as $n - 1$, n being the number of vertices in the graph. Fig. 4.8 (a)-(d) shows the reduction in the depth of the circuit of the spanning tree by the heuristic algorithm compared to the worst-case depth of the circuit corresponding to the maximum height of the DFS tree. Fig. 4.8 (a)-(d) show the reduction in depth for Erdos-Renyi graphs with p_{edge}, the probability of an edge, varying from 0.4 to 0.8, and complete graphs. For each type of graph the number of vertices n is varied from 20 to 100, and the value of the depth corresponding to each n is an average over 80 graph instances. The graph instances are the same for all the values of B. The graphs in Fig. 4.8 and Fig. 4.7 are averaged over all the possible n spanning trees generated by selecting each of the n vertices once as the root.

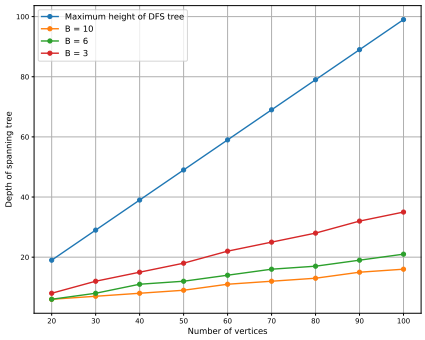
The graphs in Fig 4.8 indicate that the increase in the depth with n for various values of B is still linear. This is acceptable since the depth of a tree scales at



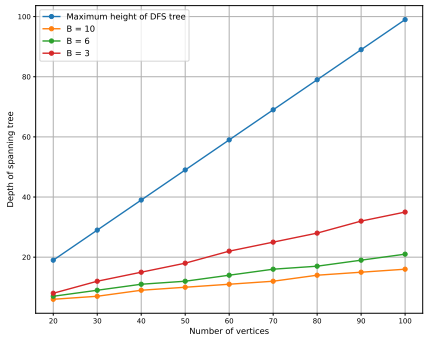
(a) Erdos-Renyi graphs with $p_{edge} = 0.4$



(b) Erdos-Renyi graphs with $p_{edge} = 0.6$



(c) Erdos-Renyi graphs with $p_{edge} = 0.8$



(d) Complete graphs

Figure 4.8: Depth of the circuit for different values of B: Erdos Renyi Graphs ($p_{edge} = 0.4,\ 0.6,\ 0.8$) and complete graphs

least logarithmically with the number of vertices. Moreover, a balanced tree is not necessarily the best in this scenario since it can suffer severely from delayed start.

In the worst case, where the depth is $n - 1$ for a graph with n vertices, the slope is $\simeq 1$. Table 4.1 shows the slopes of the curves for $B = 3$, 6, and 10 for each of the graph families considered. From the values, it is evident that the slope corresponding to the increase in depth is lowered by $\simeq \frac{1}{10}$ as the value of B increases.

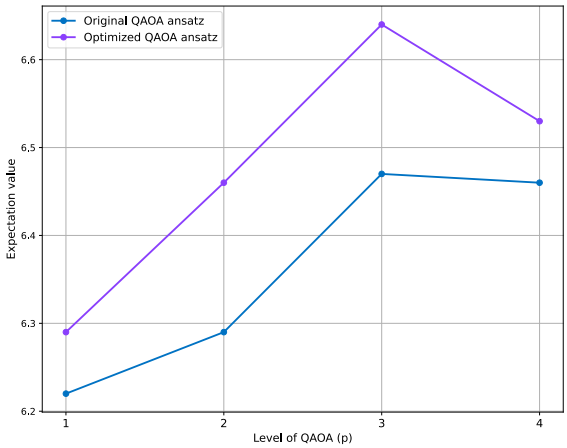


Figure 4.9: The expectation value of the optimized and unoptimized/original QAOA for Erdos-Renyi graph with $n = 12$ vertices and $p_{edge} = 0.4$. Optimized QAOA outperforms the unoptimized one for all values of p in the figure.

4.4 Usefulness of the method for $p > 1$ QAOA

A general issue with the optimization processes is that they are applicable only for the $p = 1$ level of QAOA. In general, it is not expected that $p = 1$ QAOA for any problem is sufficient to match up to the best classical algorithm known for it. However, note that if r is the number of CNOT gates in each level of QAOA, then for a level-p QAOA the total number of CNOT gates is rp. On the other hand, if the proposed optimization method is applied, then the total number of CNOT gates for a level-p QAOA becomes $rp - n + 1$, where n is the number of qubits. This is still a considerable improvement in the number of CNOT gates for large n for small values of p. Ideally, the approximation ratio produced by QAOA is a non-decreasing function of p, with the value becoming 1 as $p \to \infty$. However, in [HSN+21] the authors showed that in current hardware, the approximation ratio of QAOA starts to decrease for $p > 3$. In this regime, the noise of the circuit overwhelms the performance of the algorithm. Fig. 4.9 shows a similar trend for Erdos-Renyi graphs with $n = 12$ vertices, and the probability of edge $p_{edge} = 0.4$.

In this regime, the figure clearly shows that the optimized QAOA outperforms the unoptimized one. Although beyond $p = 3$, the approximation ratio goes down for both cases, the optimized QAOA outperforms the unoptimized one for $1 \leq p \leq 4$. This clearly shows that the effect of elimination of CNOT gates at $p = 1$ propagates to higher values of p as well.

The three algorithms proposed for optimized QAOA design were all hardware-independent. The following subsection looks into the modification of the heuristic in Algorithm 3 to make it more hardware-agnostic.

4.5 Hardware coupling map aware optimization

4.5.1 Motivation for hardware coupling map-based modification

Chapters 3 and 4 till now proposed three methods to lower the number of CNOTs in the QAOA ansatz for two-body interaction Hamiltonian problems, and the greedy heuristic algorithm (Algorithm 3) was shown to be superior among the three. All these three methods are oblivious of the hardware, and holds for any underlying hardware coupling map. However, while the number of CNOT gates can be eliminated irrespective of the underlying coupling map, the placement of the optimized QAOA circuit may lead to increased number of SWAP gates. This section focuses on the possibility of finding a rooted spanning tree that conforms with the underlying hardware connectivity if the hardware coupling map and the initial placement of the qubits are known *a priori*. In other words, during the construction of the rooted spanning tree, those edges can be preferably chosen for optimization for which there is a direct connection between the corresponding qubits in the hardware.

Quantum hardware can be represented as a graph with the vertices denoting physical qubits, and the edges denoting pairs of qubits between which two-qubit oper-

ations are possible – generally called *coupling map*. Fig. 4.10 shows the coupling map of a 5-qubit IBM Quantum device, named IBMQ Lima. From the figure, it is evident that 2-qubit gates between, say, qubits 0 and 2 are not possible. To perform such an operation, either qubit 0 or qubit 2 must be *swapped* with qubit 1. A *swap* gate is implemented via 3 CNOT gates. Therefore, inserting many swap gates in a circuit makes the computation slower and incorporates more error due to the increased number of CNOT gates. A large volume of research has been devoted to the placement of quantum circuits on the physical hardware [LDX19, YI22]. This study takes a different direction. Instead of designing a placement algorithm for the circuit generated by the greedy heuristic algorithm, it is possible to tweak the cost function itself to respect the placement constraints on a given hardware.

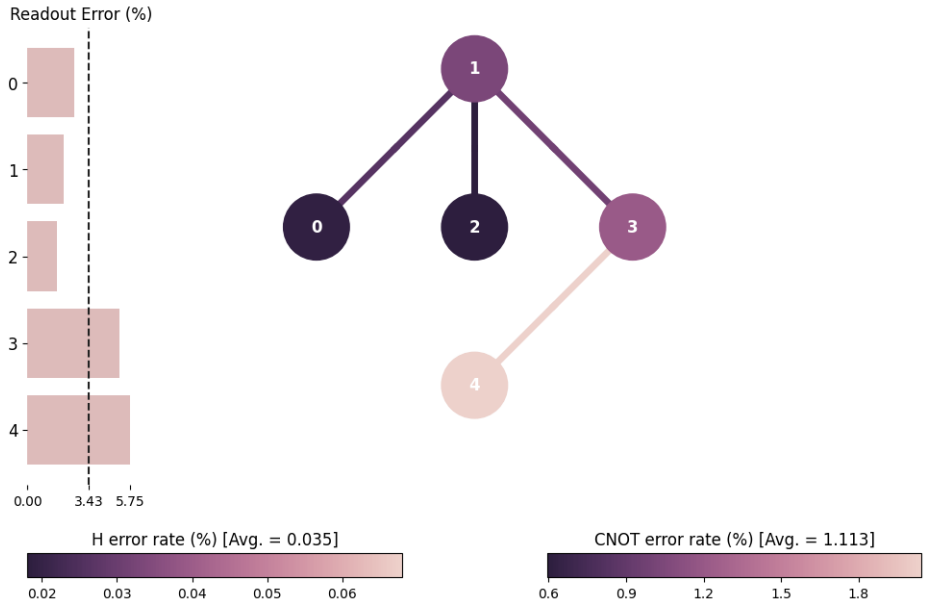


Figure 4.10: Coupling map and error probabilities of IBMQ Lima

This can be motivated with a simple example. Consider a graph with 5 vertices (Fig. 4.11) on which the QAOA algorithm is to be applied. Note that Algorithm 3 does not lead to a unique rooted spanning tree construction. The algorithm can randomly choose one of multiple edges for the construction of the rooted spanning tree if they all lead to the same cost function. Fig. 4.12 shows two QAOA circuits for Max-Cut obtained by applying the heuristic cost function method on the graph

of Fig. 4.11.

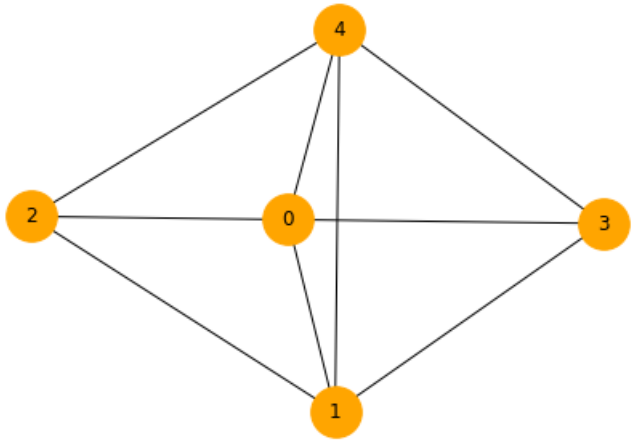


Figure 4.11: An example graph with 5 vertices

Qiskit [H+19] uses a method called *transpilation* that maps a circuit to the underlying hardware coupling map. Transpilation often follows randomized steps, and therefore different rounds of transpilation of the same circuit on the same hardware do not always lead to the same initial placement of the qubits. Therefore, as stated before, the hardware-aware heuristic algorithm will assume that the initial placement of the qubits on the underlying hardware is known beforehand. Experimentally, this can be achieved in Qiskit by fixing the seed of transpilation. For the motivational example, let the initial placement of the qubits be $\{q_0 : 2, q_1 : 1, q_2 : 0, q_3 : 3, q_4 : 4\}$, where the notation $q_i : j$ implies that the logical qubit q_i, corresponding to vertex i of the input graph, is placed on the physical qubit j of the underlying hardware. With this information, one immediately notices that the implementation of some of the CNOT gates of Fig. 4.12 (a), such as $CNOT(q_0, q_3)$ would require one or more SWAP gates. On the other hand, this SWAP gate could have been avoided if one would have chosen the edge (q_1, q_3) instead. A different QAOA ansatz, also obtained by applying the heuristic cost function, is shown in Fig. 4.12 (b) which, now, contains the optimized edge (q_1, q_3) instead of (q_0, q_3). However, Algorithm 3 does not distinguish between these two cases as long as they do not lead to different values of the cost function.

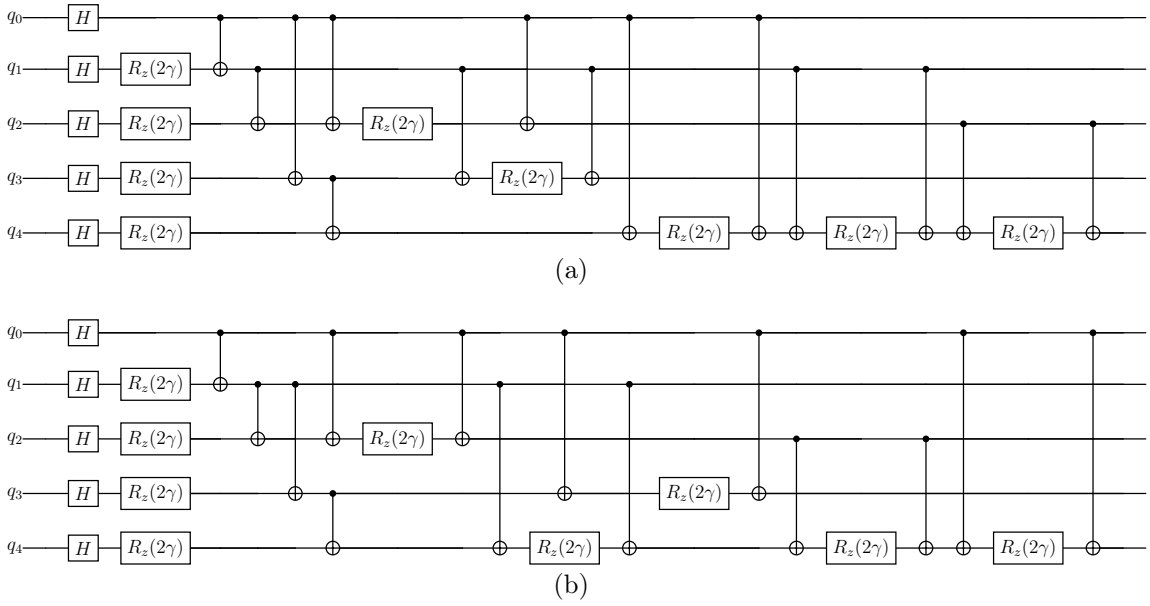
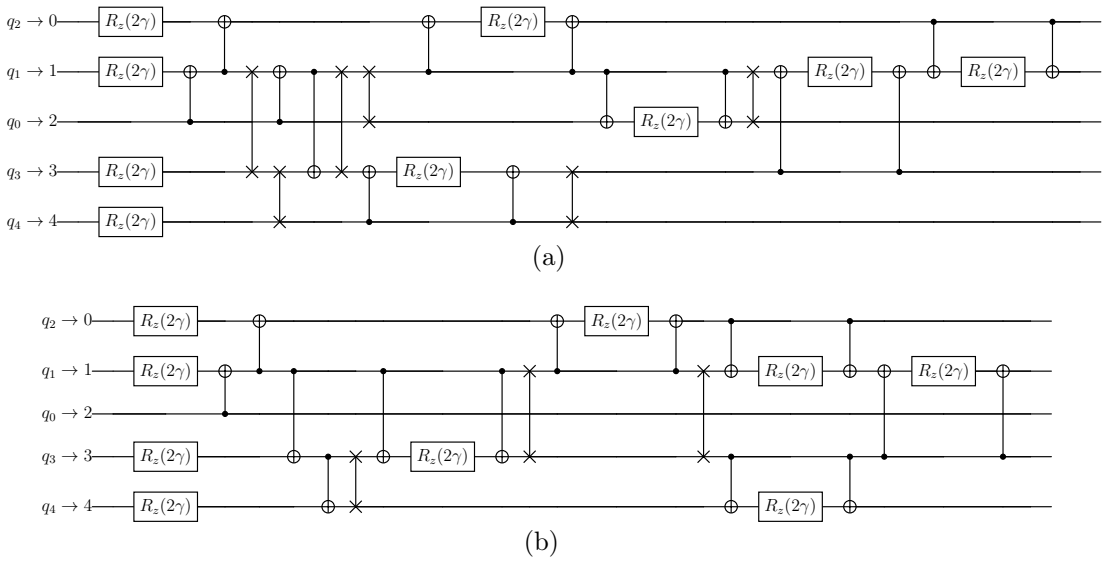


Figure 4.12: Two different QAOA circuits corresponding to the graph of Fig. 4.11 obtained using the greedy heuristic Algorithm 3

Fig. 4.13 (a) and (b) show the transpiled circuits corresponding to the two QAOA circuits of Fig. 4.12 (a) and (b) respectively. The initial placement of the qubits for both cases is the same. It is easy to see that the circuit of Fig. 4.13 (b) is preferable since it contains fewer SWAP gates.

The following subsection modifies the heuristic cost function of Algorithm 3 based on hardware information. Note that some transpilation methods have been proposed specifically for QAOA ansatz on the heavy hexagonal architecture of IBM Quantum machines [WVG+22]. This study adheres to the general transpilation procedure used in Qiskit and focuses on modifying the heuristic cost function to lower the number of SWAP gates. It may be possible to design more optimized transpilation for the methods introduced in this thesis, which is postponed for future endeavors.



(a)

(b)

Figure 4.13: Transpilation of the two different circuits (a) corresponding to Fig. 4.12 (a), and (b) corresponding to Fig. 4.12 (b), obtained by applying the greedy heuristic method on the QAOA ansatz of the graph in Fig. 4.11.

4.5.2 Hardware oriented modification of cost function

Let $Q_H = \{H_1, H_2, \ldots, H_m\}$ be the physical qubits in a quantum hardware, and $Q_G = \{q_1, q_2, \ldots, q_n\}$ be the logical qubits associated with the n-vertex input graph G, $m \geq n$. A hardware coupling map is a graph H with Q_H as the vertices. If two qubits H_i and H_j can perform a 2-qubit operation, then there is an edge (H_i, H_j) in H, and these two qubits are called neighbors. If two qubits H_k and H_l are not neighbors, then one or more SWAP gates are used on one or both of them to bring them adjacent to each other before applying the 2-qubit operation involving them. For example, in Fig. 4.10, qubits 0 and 1 are neighbors, while 0 and 2 are not. Each SWAP gate is realized using 3 $CNOT$ gates. Since the primary motivation of the circuit optimization is to lower the number of $CNOT$ gates, this section shows the usage of hardware information for designing a better heuristic cost function that, when selecting the edges that can be optimized, giving preference to those vertices in the transpiled circuit which are neighbors.

Let D_H be a hardware distance matrix such that the value of $D_H[i][j]$ is the

distance between the qubits i and j on the hardware. Two adjacent qubits are considered to have a distance of 1. In other words, the number of SWAP gates required to make the two qubits i and j adjacent is $D_H[i][j] - 1$. One can find this hardware distance matrix by deploying the Floyd-Warshall All-Pair-Shortest-Path Algorithm [CLRS09]. Note that the hardware graph is static. Therefore, although the required runtime to find the distance matrix D_H is $\mathcal{O}(n^2)$, n being the number of qubits in the hardware, this needs to be performed only once. The information can be stored for subsequent uses on different input graphs.

Ideally, choosing an edge between two far-away qubits for optimization should be discouraged. The higher the distance between the two qubits, the more should such an edge be avoided for optimization. This information is stored in the initial placement I such that $I(q) = i$ implies that the vertex (or circuit qubit) q has been placed in the hardware qubit i. Therefore, the distance between two vertices, say u and v, in the hardware, is given by $D_H[I(u)][I(v)]$. This information is inserted into the cost function algorithm by subtracting a penalty term $\eta \cdot (D_H[I(u)][I(v)] - 1)$ from the cost function. Furthermore, a pre-defined maximum branching factor B is no longer suitable for this modified cost function. Rather, once a vertex v has been placed to a qubit q, the maximum branching factor v_{bf} of that vertex is assigned to be

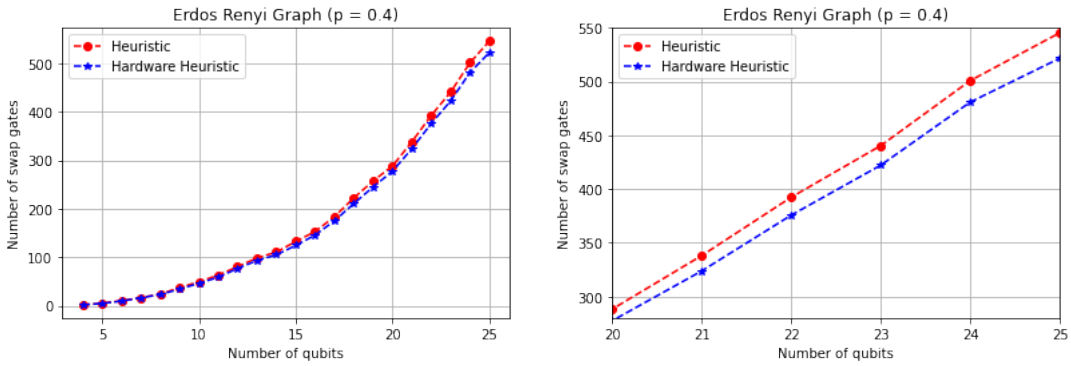
$$v_{bf}^{max} \begin{cases} = degree(q) & \text{if v is the root} \\ = degree(q) - 1 & \text{otherwise.} \end{cases} \tag{4.3}$$

The degree of the qubits in the hardware can be easily obtained by using Breadth-First-Search in a time linear in the number of vertices and edges. Once more, since the hardware graph is fixed, the time complexity for finding the degree of each qubit is $\mathcal{O}(1)$. One can consider finding the degree of the qubits and the distance matrix as a constant time pre-processing step to the algorithm.

Algorithm 4 depicts the modified greedy heuristic algorithm tailored to the underlying hardware information.

4.5.3 Reduction in the number of SWAP gates

Previous results using Algorithm 3 were already shown to outperform the edge coloring, DFS, and the traditional QAOA ansatz in terms of error probability. For those simulations, as discussed before, the circuits were transpiled to the underlying heavy hexagonal architecture of *ibmq_manhattan*. Therefore, the error probability obtained in those results included any SWAP gates that had to be inserted during the transpilation procedure. The results were shown to be better even with the errors due to inserted SWAP gates. Naturally Algorithm 4 cannot perform any worse in terms of error probability than Algorithm 3 since the former tries to lower the number of SWAP gates inserted. In the worst-case scenario, where the number of SWAP gates remains the same for both algorithms, the errors will be the same. In other scenarios, Algorithm 4 will provide lower error due to a lower number of SWAP gates. The results in this section, therefore, focus only on the number of SWAP gates that can be lowered using Algorithm 4 over Algorithm 3. The experiments are performed for QAOA for the Max-Cut problem, and the results are averaged over 50 random instances of Erdős-Renyi graphs with probability of edge $p_{edge} \in \{0.4, 0.6, 0.8, 1\}$.



(a) Number of swap gates in the transpiled circuit with and without hardware information

(b) Zoomed in plot of the same graph

Figure 4.14: Number of SWAP gates in the transpiled circuit by using Algorithm 4 (termed Hardware Heuristic) and Algorithm 3 (termed Heuristic)

The results in Fig. 4.14 show that for a small number of qubits, both approaches

lead to the same number of SWAP gates. This is expected because it is easier to place a small circuit on a large hardware due to the available redundancy. However, as the number of qubits in the circuit increases, an average reduction in the number of SWAP gates by $\simeq 30$ is observed when Algorithm 4 is used. The obtained percentage reduction for different Erdős-Renyi graphs is reported in Table 4.2. A very similar result is obtained for all four types of Erdős-Renyi graphs used. Therefore, only the results for Erdős-Renyi graphs with $p_{edge} = 0.4$ are shown in Fig. 4.14 (a). Fig. 4.14 (b) shows a zoomed-in portion of the plot of Fig. 4.14 (a) when the number of qubits is high, for better visualization.

Algorithm 4 Hardware aware cost function based rooted spanning tree generation

Input: A graph $G = (V, E)$, $|V| = n$, $|E| = m$; hardware distance matrix D_H; initial placement I; v_{bf}^{max} for all v; penalty η.
Output: A rooted spanning tree T of the graph G.

$T = \{\}$.
$u_{bf} \leftarrow 0$ for all vertex u.
$r \leftarrow$ randomly selected start vertex.
Visited $= \{r\}$; $r_{bf} = r_{bf} + 1$.
edges_to_add = neigh(r).
while $|Visited| < n$ **do**
  $e = edges\_to\_add[0]$; $c = 0$.
  for all $edge = (u, v) \in edges\_to\_add$ **do**
    $cost = (n - l_u) \cdot (u_{bf}^{max} - u_{bf}) - \eta \cdot (D_H[I(u)][I(v)] - 1)$.
    if $cost > c$ **then**
      $c = cost$; $e = edge$.
    end if
  end for
  $T = T \cup \{e\}$.
  Visited = Visited $\cup \{y\}$, where $e = (x, y)$.
  $x_{bf} = x_{bf} + 1$.
  Remove all edges of the form $(*, q)$ from edges_to_add.
  for all $edge = (p, q) \in neigh(y)$ **do**
    if $q \notin Visited$ **then**
      $edges\_to\_add = edges\_to\_add \cup \{edge\}$.
    end if
  end for
end while

Finally, Table 4.2 provides the maximum and the average percentage reduction in

SWAP gates obtained over all the graphs used for simulation. Note that, as obvious from Fig. 4.14, the maximum reductions are obtained for graphs with a higher number of vertices. The average is lower than the maximum since graphs with a smaller number of vertices show little to no improvement when using Algorithm 4 over Algorithm 3.

Table 4.2: Maximum and average percentage reduction in the number of SWAP gates when using Algorithm 4 instead of Algorithm 3 for various classes of Erdős-Renyi graphs with probability of edge p_{edge}

	$p_{edge} = 0.4$	$p_{edge} = 0.6$	$p_{edge} = 0.8$	$p_{edge} = 1$
Maximum	8.37	8.57	11.24	11.54
Average	4.91	4.94	5.7	5.73

The results show that the maximum percentage reduction in the number of SWAP gates remains more or less the same for $p_{edge} \in \{0.4, 0.6\}$ and $p_{edge} \in \{0.8, 1\}$ and increases from the former to the later set. This is legitimate since with the increasing density of the graph more SWAP gates become essential. Therefore, a better choice of edges is expected to be more critical in such cases than for more sparse graphs. A similar trend is observed for the average percentage reduction as well.

Note that the results reported are for $\eta = 1$. Varying η did not have an impact on the result. A reason for this seems to be that the penalty $(D_H[I(u)][I(v)] - 1)$, for some pair of vertices u and v, is dependent on the distance between them in the hardware. Increasing the value of η simply scales the penalty term, and therefore does not add any extra information to the algorithm. For the same reason, any other function of the distance (polynomial, exponential, logarithmic, etc.) as the penalty term is not studied. The expectation is that any monotonically increasing function of $(D_H[I(u)][I(v)] - 1)$ should suffice as the penalty term in the algorithm, and should provide the same outcome.

These results, therefore, establish that if the knowledge of hardware connectivity and the initial placement of qubits is known beforehand, then the heuristic cost function can be tweaked appropriately to reduce a few SWAP gates. It may be possible to design a better transpilation method targeted to this type of optimized

ansatz circuit only which is postponed for future studies.

4.6 Summary

This chapter proposed a hardware-independent polynomial time method to eliminate CNOT gates in the ansatz design of QAOA for 2-body Hamiltonian problems for discrete combinatorial optimization. This method outperforms the previously proposed methods based on DFS and Edge Coloring (Chapter 3) by retaining the elimination of $n-1$ CNOT gates but restricting the increase in depth of the circuit. Finally, if the hardware coupling map and the initial placement of the qubits are known *a priori*, then this heuristic method can be further improved to select those edges for the spanning tree that conform to the underlying coupling map – thus reducing the number of SWAP gates required. Both of these methods work only on $p = 1$ QAOA. However, although there is no such optimization known for $p > 1$, the effect of CNOT elimination at p=1 propagates to the higher values of p and better expectation values of the optimization results for $p > 1$ QAOA even.

Chapter 3 and 4 provided some algorithmic methods to lower the gate count in the circuit, effectively lowering the noise. Another method for noise reduction is to partition a circuit into multiple smaller subcircuits, such that each subcircuit is less susceptible to noise. This method, called *circuit cutting*, and some improvements on it, are discussed in the following chapter.

CHAPTER 5

Error mitigation by quantum circuit cutting

Contents

5.1 Introduction

The limited number of qubits on near-term quantum devices is a significant constraint on the size and type of quantum computation problems that can be evaluated on them. Circuit knitting, an umbrella term for combining results of two or more smaller quantum processors to logically form a larger device, has been suggested as a top-down approach to scalability [BDG+22], in addition to the approaches addressed in the last two chapters. For such a logical device, it is useful to *cut* a quantum circuit into smaller pieces so that each of them can be executed on the hardware at hand. Circuit knitting can be broadly classified into (i) cutting the wire (termed as circuit cutting henceforth) [PHOW20, PSSO21, TTS+21, LMH+22, UADM22], (ii) replacing two-qubit gates by mid circuit measurements and classical feed-forward conditional options (often called gate cutting) [MF21a, PS22, MF21b], and (iii) partitioning the problem into multiple weakly interacting sub-problems (e.g. entanglement forging [EMG+22], frozen-qubit QAOA [AADQ22]). This chapter focuses on cutting the wire to create smaller circuit fragments such that each fragment is small enough to be computed on the quantum hardware individually. Henceforth, the term *circuit cutting* will imply cutting the wire only. Since the cutting is along the wire, all the gates from the original circuit are retained in the ensemble of the fragments (see Figs. 5.1 and 5.2).

The expectation value of the original circuit can be retrieved by classically combining the expectation values of the individual fragments obtained in different preparation and measurement bases [PHOW20]. An example to reconstruct the probability distribution of Fig. 5.1 from the two fragments in Fig. 5.2 is given below. The calculation of the probability of obtaining the outcome 0000, denoted as $P(0000)$, from the circuit of Fig. 5.1 is presented and the probabilities of the

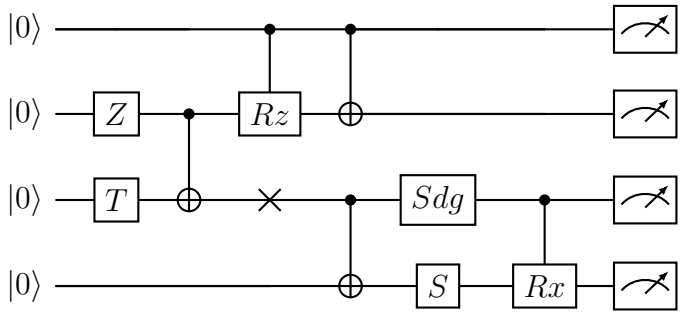
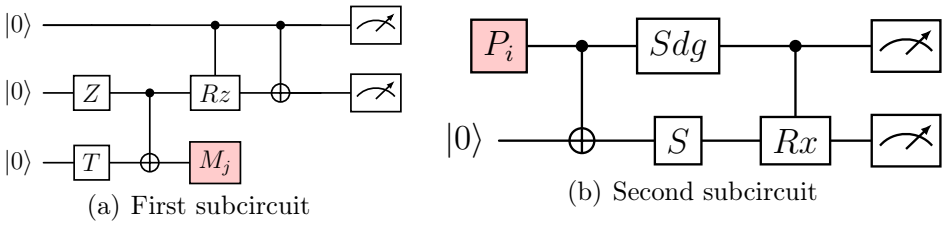


Figure 5.1: An example circuit with the red cross signifying the location of cut



(a) First subcircuit

(b) Second subcircuit

Figure 5.2: An example of cutting the 4 qubits quantum circuit of Fig. 5.1 into 2 fragments. P_i and M_j denote tomographically complete preparation and measurement basis respectively.

other states can be computed similarly.

The variable to be calculated from the first subcircuit (Fig. 5.2 (a)), where $P(abc)M_j$ denoting the probability of obtaining the outcome abc when the cut qubit undergoes measurement $M_j \in \{I, X, Y, Z\}$, are considered in Eq. (5.1). Note that measurements in I and Z bases are essentially the same. Therefore, a single measurement in the Z basis is sufficient to calculate the terms involving measurement in I and Z bases.

$$\begin{aligned} p_{1,1} &= P(000)I + P(001)I + P(000)Z - P(001)Z \\ p_{1,2} &= P(000)I + P(001)I - P(000)Z + P(001)Z \\ p_{1,3} &= P(000)X - P(001)X \\ p_{1,4} &= P(000)Y - P(001)Y \end{aligned} \tag{5.1}$$

Similarly, the variables to be calculated from the second subcircuit (Fig. 5.2 (b)), where $P(ab)\,|P_i\rangle$ denoting the probability of obtaining the outcome ab when the cut qubit was prepared in state $P_i \in \{|0\rangle, |1\rangle, |+\rangle, |+i\rangle\}$, are given in Eq. (5.2).

$$\begin{aligned} p_{2,1} &= P(00)\,|0\rangle \\ p_{2,2} &= P(00)\,|1\rangle \\ P_{2,3} &= 2P(00)\,|+\rangle - P(00)\,|0\rangle - P(00)\,|1\rangle \\ P_{2,4} &= 2P(00)\,|+i\rangle - P(00)\,|0\rangle - P(00)\,|1\rangle\,. \end{aligned} \tag{5.2}$$

The final probability $P(0000)$ for the original circuit in Fig. 5.1 is then retrieved as

$$P(0000) = \frac{1}{2}\sum_i p_{1,i} \otimes P_{2,i}. \tag{5.3}$$

Note that recombination of the probability for a single cut requires four multiplications. Therefore, for k cuts, the classical recombination time scales as $\mathcal{O}(4^k)$, thus making a large number of cuts impractical. There is no method to find the best cut location in a given circuit, and indeed, there is no clear definition as to which cut qualifies as the *best* cut. However, studies have been performed to find the cut locations which minimize (i) the empirical postprocessing time [TTS+21], or (ii) the error on each fragment [BSCSK21].

As mentioned in Chapter 2, circuit cutting can produce fragments that behave as an unknown quantum state or channel. Sec. 2.4.2 of Chapter 2 discussed about the role of tomography in this context, and its different forms such as Linear Inversion and Constrained Least Square. The description of tomography from Chapter 2 has also been extended to conditional fragment tomography in the noiseless scenario for the sake of efficiency. This chapter proposes two error mitigation methods specific to conditional fragment tomography.

5.2 Error mitigation for Conditional Fragment Tomography

First, we provide a brief introduction on *Conditional Fragment Tomography*, as introduced by Perlin et al. in [PSSO21].

5.2.1 A brief introduction to Conditional Fragment Tomography

Consider a cut fragment $\mathcal{F}$ with m-qubit measurements corresponding to the original circuits outputs, and k qubits corresponding to the cut qubits as either a state, channel, or POVM fragment. For example, in Fig. 5.2 (a), $m = 2$ and $k = 1$.

Rather than reconstructing a full description of the fragment tensor which would be inefficient, one can take advantage of the block-diagonal structure and instead reconstruct the set of conditional fragment components $\{T(s)\}$ for $s \in \{0,1\}^m$. This is done by choosing a tomographically complete basis $\{B_j\}$ on the k cut qubit subsystem, and also defining an orthonormal (but tomographically incomplete) basis $\{\Pi_s\}$ on the m conditional measurement outcome qubits. Typically, this second basis is chosen as computational basis $\Pi_s = |s\rangle\langle s|$. With these two bases, tomography of the fragment tensor $T_{\mathcal{F}}$ can be considered in terms of the tensor product basis $B_i \otimes \Pi_s$, where the probability of observing an outcome (i, s) is given by

$$p_{i,s} = \langle\langle B_i \otimes \Pi_s | T_{\mathcal{F}} \rangle\rangle = \sum_{s'} \langle\langle B_i | T(s') \rangle\rangle \langle s' | \Pi_s | s' \rangle \tag{5.4}$$

For $\Pi_s = |s\rangle\langle s|$, we have

$$p_{i,s} = \langle\langle B_i | T(s) \rangle\rangle. \tag{5.5}$$

This tomographic reconstruction of $T(s)$ can be performed via linear inversion, least-squares optimization, or any other tomography fitter by fixing the index s and performing tomography fitting using the set of probabilities $\{p_{i,s}\}$. The fragment $\mathcal{F}$ can thus be represented as a block-diagonal $n = k + m$ qubit tensor

$$T_{\mathcal{F}} = \sum_{s \in \{0,1\}^m} T(s) \otimes |s\rangle\langle s| \tag{5.6}$$

This implies that in general, for a fragment $\mathcal{F}$ with m conditioning measurements and k cut qubits, there are 2^m conditional k-qubit tensor tomography fitting procedures to run on the measurement data.

5.3 Error mitigation on quantum circuit cutting

In general, tomography is used to accurately characterize the system at hand. As the system is often noisy, tomography is usually used to accurately characterize the noise in the system. Therefore the use of error correction or mitigation with tomography is fallacious. However, for circuit cutting, tomography is used as a subroutine to characterize the *ideal* system. Therefore, mitigation of the errors in the system to accurately characterize the ideal system is mandated. Circuit cutting presents several opportunities for error mitigation in addition to those that can be applied to standard circuits (e.g., Zero Noise Extrapolation, Probabilistic Error Cancellation) [TBG17]. Since individual fragments contain fewer gates than the original circuit (Refer to Fig. 5.2), they may contain less overall noise [ARS+21, BSCSK21], which may make them more amenable to error mitigation techniques such as probabilistic error cancellation which exhibit exponential scaling with the total noise strength of the circuit [TBG17, EBL18, BMKT22]. Furthermore, tomography is used as a subroutine for reconstruction [PSSO21] mitigation techniques, such as eigenvalue truncation or re-scaling, and hence can be applied during the fitting which would not have been possible otherwise.

This chapter proposes and investigates the performance of two new forms of tomography-specific error mitigation which are not possible with standard cir-

cuit execution. These are (i) *measurement error mitigated constrained least square (MEMCLS) tomography*, which aims to remove the effect of measurement errors during the tomographic reconstruction by performing a constrained fit of all conditional fragments simultaneously using the knowledge of the readout error model, and (ii) *dominant eigenvalue truncation* (DEVT) which involves truncation of the reconstructed state or channel to its largest eigenstate.

5.3.1 Measurement Error Mitigated Constrained Least Square (MEMCLS)

A significant source of error in tomographic experiments is so-called state preparation and measurement (SPAM) errors. If an ideal preparation and measurement basis is used in a tomography fitter, any errors in these processes are attributed to those in the reconstructed state or channel itself. In current quantum devices, measurement errors are the dominant source of SPAM error and are in the range of 0.5%-10% for a single-qubit measurement depending on the architecture [NKSG21]. Thus a variety of error mitigation schemes for classical readout errors have been proposed which involve some characterization process of the measurement error model, and processing of measurement outcomes to attempt to undo these effects [CBB+22, NKSG21, VDBMT22a]. When performing tomography, it is inadvisable to apply such mitigation techniques to process counts before using them during tomographic fitting. The reason is that these techniques modify the circuit outcome without properly modifying the variance of the distribution which are then used in the tomographic fitting. Instead, one can perform general measurement error mitigation as part of the fitting procedure if they have a well-characterized measurement error model by using the noisy POVM elements directly in the tomography fitter basis, which in turn can be used to construct a noisy error mitigated dual basis for linear inversion [SGS12], or used directly in the least-squares objective function to find the maximum likely fit.

A simple example can be considered for this. Let us assume that a density matrix ρ can be represented as $\rho = \sum_j p_j \Pi_j$, where Π_j are the projectors and p_j are the

corresponding probabilities. In other words, if N copies of ρ are measured in the basis $\{\Pi_j\}$, then the probability of observing the outcome $\Pi_j = \frac{n_j}{N} = p_j$. Now, let us assume that due to measurement error, the projectors that are actually getting measured are Π'_j, and the corresponding probabilities are p'_j. Since Π'_j are unknown, the reconstructed state will be $\rho' = \sum_j p'_j \Pi_j$, which will not agree with the actual state ρ. On the other hand, if some characterization experiment is performed to learn the noisy projector (or POVMs) Π'_j, then the reconstruction $\rho \approx \sum_j p'_j \Pi'_j$ is more accurate. The accuracy of this reconstruction depends on the characterization of the noisy POVMs Π'_j – the better the characterization, the more accurate the reconstruction. Measurement error mitigated tomography uses this ideology by determining the noise POVMs and fitting the tomographic data using this noisy basis.

Conditional tomography presents a challenge since this kind of error mitigation can only be applied to the basis elements of the tomography fitter, while the non-tomographic measurements used to condition the data for each fragment component will also be noisy. This means that instead of using Eq. (5.5) to define our conditional probability distribution, we should use Eq. (5.4) where Π_s is no longer diagonal, but represents our measurement error model on the conditioning qubit measurements. If a classical readout error model is considered, then these conditional qubit bases can be written as

$$\Pi_s^{noise} = \sum_{s'} P(s|s') \, |s\rangle\langle s'| \tag{5.7}$$

where $P(s|s')$ is the probability of recording a true outcome s as the noisy outcome s'. The matrix of readout error probabilities $A = \sum_{s,s'} P(s|s') \, |s\rangle\langle s'|$ is typically called an *assignment matrix* [NKSG21]. Using the assignment matrix readout error model our noisy conditional probabilities for fragment tomography are given by

$$p_{i,s}^{noise} = \sum_{s'} P(s|s') \langle\langle B_i | T(s') \rangle\rangle. \tag{5.8}$$

where B_i can also be chosen to be a noisy basis element corresponding to the

measurement error on the cut-qubit measurements. To apply mitigation during reconstruction a modification of the conditional least-squares fitter in Eq. (2.7) can be made to simultaneously fit all fragments $T(s)$ using the readout error probabilities $P(s|s')$ using the optimization objective in Eq. 5.9.

$$\{T(s)^{mit}_{LS}\} = \arg\min_{\{T(s)\geq 0\}} \frac{1}{2} \left\| \Sigma^{-1/2} \sum_{s} \left(\sum_{s'} P(s|s') \langle\langle B_i | T(s') \rangle\rangle - p_{i,s} \right) |s\rangle \right\|_2^2 \tag{5.9}$$

5.3.2 Dominant Eigenvalue Truncation (DEVT)

Ideally, a noiseless quantum circuit maps a pure state ρ_{in} to another pure state ρ_{out}. However, in reality, the output state ρ_{noisy} is a mixed state due to the effect of noise. Dominant eigenvalue truncation (DEVT) asserts that when the strength of the noise is low, the largest eigenvector $|\psi_1\rangle$ of ρ_{noisy} has a significant overlap with ρ_{out}, and hence can be considered to be a very close approximation of ρ_{out}. The inaccuracy, when the noiseless state is approximated by the largest eigenvalue of the noisy state, is captured by a quantity termed *coherent mismatch*. If $\rho_{out} = |\psi\rangle \langle\psi|$, then the coherent mismatch c is defined as [Koc21]

$$c = 1 - \langle\psi_1|\psi\rangle \tag{5.10}$$

Applicability of DEVT assumes a noise model for which the noisy output state ρ_{noisy} can be represented as a convex mixture of the ideal outcome ρ_{out} and some error density matrix ρ_{err} as in Eq. (5.11), p is the probability of error [Koc21].

$$\rho_{noisy} = (1-p)\rho_{out} + p\rho_{err} \tag{5.11}$$

In such a scenario, the coherent mismatch c is upper bounded as in Eq. (5.12)

[Koc21], where $\delta = (\frac{1}{1-p} - 1)\mu_1$, μ_1 being the largest eigenvalue of ρ_{err}.

$$c \leq \frac{\delta^2}{4} + \mathcal{O}(\frac{\delta^4}{16}) \tag{5.12}$$

This study will focus on applying the DEVT to quantum channels as the example circuits in Fig. 5.3, and 5.4 contain only channel fragments. The Choi-matrix for an n-qubit CPTP quantum channel $\mathcal{E}$ can be written in its eigenbasis as

$$\Lambda_{\mathcal{E}} = \sum_i |K_i\rangle\rangle\langle\langle K_i| \tag{5.13}$$

where the matrices $\{K_i\}$ correspond to the canonical Kraus decomposition [WBC15], and we assume they are ordered such that $\langle\langle K_i|K_i\rangle\rangle \geq \langle\langle K_{i+1}|K_{i+1}\rangle\rangle$. The DEVT approximation to $\mathcal{E}$ is then the pure-state Choi-matrix

$$\Lambda_{DEVT(\mathcal{E})} = \frac{2^n}{\langle\langle K_0|K_0\rangle\rangle}|K_0\rangle\rangle\langle\langle K_0|. \tag{5.14}$$

To include this as a mitigation strategy in circuit cutting, DEVT is applied to each individual channel tensor fragment $\Lambda_{\mathcal{E}} = T_i(s_i)$ in Eq. (5.6), and these truncated fragments are used when computing the outcome probabilities.

Note that when applied to quantum channels the DEVT may result in a truncated channel which is not trace-preserving, even if the non-truncated channel is. This is because a channel truncated to a single eigenvector is trace-preserving if and only if $K_0^\dagger K_0 = p\mathbb{I}$, which requires that the largest Kraus matrix be a scaled unitary $K_0 = \sqrt{p}U$.

5.4 Simulation and numerical results

This section shows the simulation result of conditional fragment tomography applied to circuit cutting in the presence of various noise models. For performing

the circuit cutting reconstruction, three different tomography fitters are considered: linear inversion (LIN), constrained least-squares (CLS) (both discussed in Chapter 2), and measurement error mitigated constrained least squares (MEM-CLS) (discussed in Sec. 5.2). Each fitter is compared with and without Dominant Eigenvalue Truncation (DEVT) error mitigation. All tomography reconstruction experiments consisted of $10,000$ trials, or *shots*, and were implemented using a modified version of the process-tomography experiment from the *Qiskit Experiments* [qex22] Python package. For comparison with direct simulation, the effect of standard A-matrix inversion readout error mitigation was also included on the estimated probabilities using the *M3* mitigation package in Qiskit [NKSG21, mth22].

To study the effects of error mitigation, a variety of noise models were considered. All noisy simulations were done using the *Qiskit Aer* simulator [aer22] with a local noise model by decomposing the simulated circuits into Controlled-NOT, and 1-qubit gates (SX, X, RZ), which is currently the basis gate set of IBM quantum devices), and Z-basis measurements. The noise was then added to either the 2-qubit gates, 1-qubit gates, or single-qubit measurements using the same noise parameters for all qubits to simplify analysis. Note that in IBM Quantum devices, the RZ gate is not physically executed, rather its effect is accounted for in the software by a rotation of axis [MWS+17b]. Hence, RZ is essentially a virtual gate causing no error. Therefore, a single qubit gate error was added to the X and SX gates only.

Measurement errors were simulated using the classical readout error model as described in Sec. 5.4.1. For gate errors, a local Markovian gate model was considered where each noisy gate is simulated as $\mathcal{U}_{noise} = \mathcal{E} \cdot U$ where U is the ideal gate unitary, and $\mathcal{E}$ is a completely-positive trace preserving (CPTP) quantum noise channel, which can be written in the Kraus representation as $\mathcal{E}(\rho) = \sum_i K_i \rho K_i^\dagger$, where $\sum_i K_i^\dagger K_i = \mathbb{I}$ [WBC15]. Four different gate noise models were simulated for the channel $\mathcal{E}$ – namely, (i) depolarizing, (ii) stochastic Pauli, (iii) amplitude damping, and (iv) coherent noise, where the noise was applied to both 1 and 2 qubit gates. The results from the gate error are shown in Sec. 5.4.2. Finally, Sec. 5.4.3 shows the simulation results for non-mixed unitary errors such as amplitude damping

and coherent rotation.

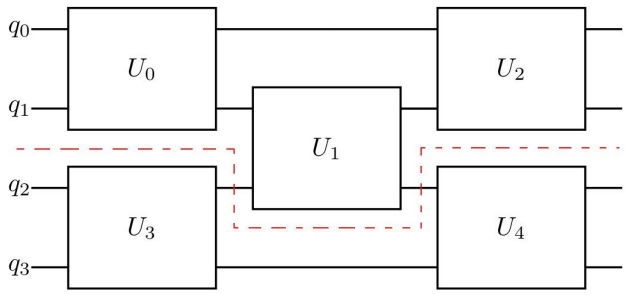


Figure 5.3: Cutting a 4-qubit cluster unitary circuit into 2 fragments. The dotted red line denotes the cut.

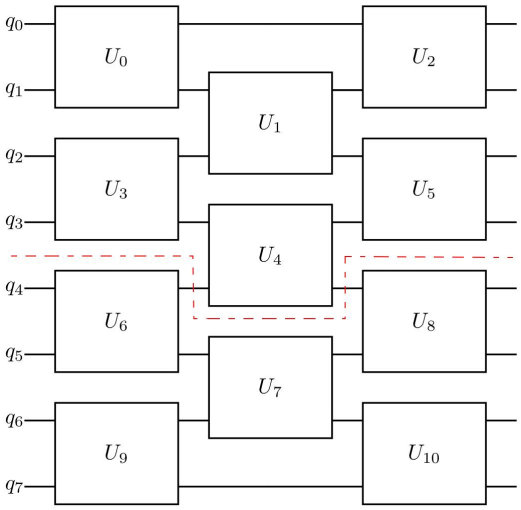


Figure 5.4: Cutting an 8-qubit cluster unitary circuit into 2 fragments. The dotted red line denotes the cut.

The numerical studies consider a cluster unitary circuit consisting of alternating layers of random 2-qubit unitary gates. Such circuits are representative of Trotterized simulation used in applications for near-term quantum devices [BMKT22, CA+21]. The experiments consider clusters with a fixed depth of 3 layers of random unitaries between adjacent qubits and the problem of estimating the full outcome probability distribution of Z-basis measurements on all qubits. The direct simulation of the full uncut circuit is compared to the simplest circuit cutting configuration with 2 or 3 fragments under a variety of noise models. For 2-fragments, comparison simulations were performed for original uncut circuits containing 4

qubits (Fig. 5.3), 8 qubits (Fig. 5.4) and 12 qubits, resulting in 1-qubit conditional process tomography fragment tensors with 2, 4, and 6 conditioning qubit measurements from the uncut circuit respectively.

To evaluate the performance of circuit cutting compared to direct simulation, the estimated probability distribution from both methods is compared to the expected ideal distribution using the 1-norm distance measure

$$D(P) = \frac{1}{2}\|P - Q\|_1 = \frac{1}{2}\sum_i |p_i - q_i| \tag{5.15}$$

where p_i and q_i are the outcome probabilities for the estimated distribution P, and ideal target distribution Q respectively. Note that this measure is equivalent to the *trace distance* $T(\rho, \sigma) = \frac{1}{2}\mathrm{Tr}|\rho - \sigma|$ between two quantum states if the probability distributions are considered to be diagonal density matrices [NC02]. Therefore, when $P = Q$ (or $\rho = \sigma$), it is expected that $D(P) = 0$.

5.4.1 Measurement Noise

First measurement noise was simulated with a symmetric single-qubit assignment matrix

$$A = \begin{pmatrix} 1 - p_{meas} & p_{meas} \\ p_{meas} & 1 - p_{meas} \end{pmatrix}. \tag{5.16}$$

In this noise model, the gates are considered to be ideal. Fig. 5.5 shows the reconstructed distributions trace distance for CLS, MEMCLS both with and without DEVT, and the uncut circuit both with and without M3 readout error mitigation when using $p_{meas} \in \{0.01, 0.05\}$. For $p_{meas} = 0.01$, circuit cutting alone improves the performance for all the scenarios. When $p_{meas} = 0.05$, for 8-qubit and 12-qubit 2-fragment circuits, again circuit cutting alone improves performance, as was reported in prior studies [ARS+21, BSCSK21]. This improvement increases with the number of measurement qubits in the original circuit, while for the 4-qubit case, the opposite is seen with the uncut circuit having lower error than the cut circuit reconstruction. This is because, for the 4-qubit case, the total number of outcome

probabilities used in each fragment for the tomographic reconstruction is larger than the number of probabilities in the original circuit (24 vs 16), while for the 8 and 12-qubit cases, each fragment has significantly fewer probabilities than the original circuit (96 vs 256 and 384 vs 4096 respectively).

Including readout error mitigation using MEMCLS or CLS with DEVT improves the circuit cutting performance, with CLS+DEVT outperforming MEMCLS alone, and being comparable to MEMCLS+DEVT. This suggests that DEVT is more effective than MEMCLS for mitigating pure readout errors in tomographic reconstructions. The same trends are also observed for reconstruction using 3 fragments shown in Fig. 5.5 (c) and (d). Henceforth, for other noise models, only the results for circuit cutting with 2 fragments are presented, since for equal-sized fragments the reconstruction error per fragment is constant and the 2-fragment case is representative of the relative performance of the different methods.

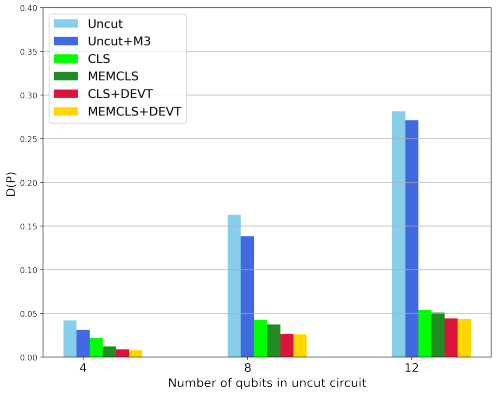
5.4.2 Gate Noise

Next, the effects of gate error are included as well as measurement readout error. In current devices 2-qubit gate error is one of the dominant sources of circuit error, so we consider several different 2-qubit noise models applied to all CNOT gates in our circuit. In all cases, we fix the measurement error to be the symmetric classical readout error described in Sec. 5.4.1 with $p_{meas} = 0.01$ or 0.05, and including a 1-qubit gate error model (depolarizing or Pauli) with $p_1 = 10^{-4}$ on single-qubit gates.

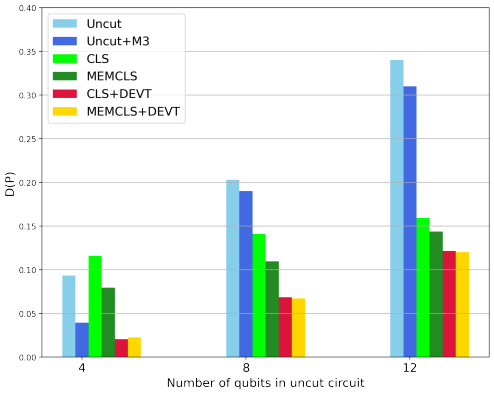
The first case we consider is a 2-qubit depolarizing noise channel given by the map

$$\mathcal{E}_{depol}(\rho) = (1 - p_{depol})\rho + p_{depol}\frac{\mathbb{I}}{4}. \tag{5.17}$$

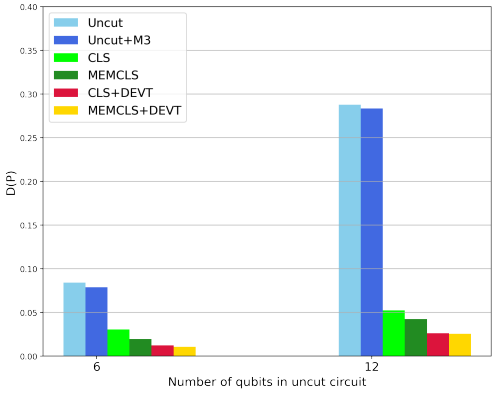
First, 2-qubit depolarizing probabilities of $p_{depol} = 0.01$ and 0.02, shown respectively in Fig. 5.6 (a) and (b) was simulated. The results show that both CLS and



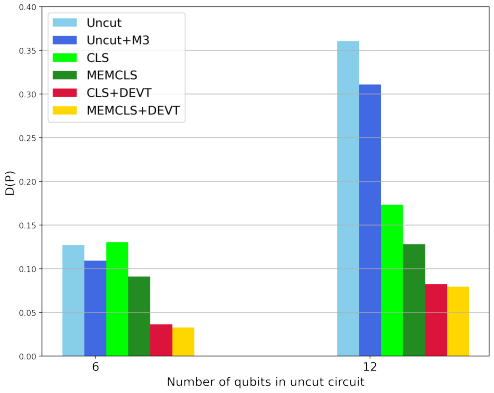
(a) 2 fragment circuit cutting with $p_{meas} = 0.01$



(b) 2 fragment circuit cutting with $p_{meas} = 0.05$



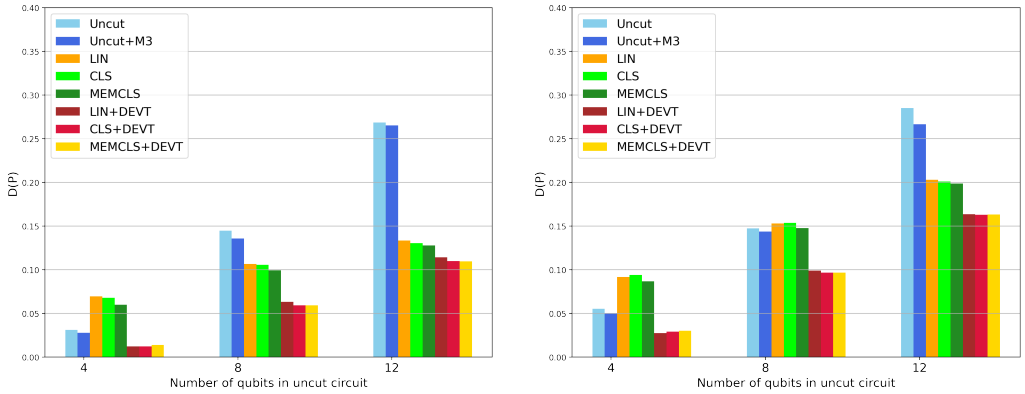
(c) 3 fragment circuit cutting with $p_{meas} = 0.01$



(d) 3 fragment circuit cutting with $p_{meas} = 0.05$

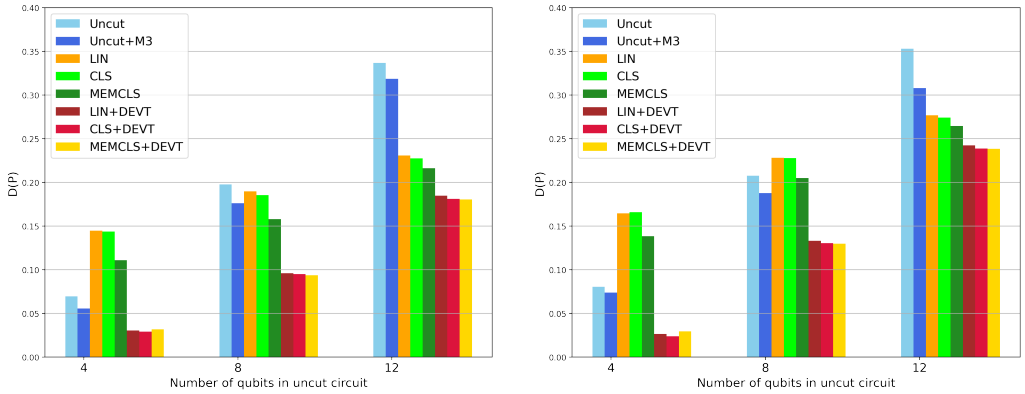
Figure 5.5: Performance of tomographic circuit cutting reconstruction using 2 and 3 fragments under the effect of local symmetric readout error with a readout error probability of $p_{meas} \in \{0.01, 0.05\}$. The vertical axis is the trace distance (Eq. (5.15)) of the reconstructed probability distribution from the noiseless probability distribution. The cut circuit reconstruction was performed using both constrained least-squares conditional tomography fitter (CLS) and a readout error mitigated fitting (MEMCLS) fitter using the noisy basis corresponding to the classical readout error noise parameter both with and without DEVT mitigation. The original circuit (uncut) was measured with and without M3 readout error mitigation for comparison.

LIN tomography fitters with DEVT perform better than MEMCLS alone and that when applying DEVT mitigation all tomography fitters have comparable performance. This suggests that if one is employing DEVT, then full MEMCLS is not



(a) $p_{depol} = 0.01$, $p_{meas} = 0.01$

(b) $p_{depol} = 0.02$, $p_{meas} = 0.01$



(c) $p_{depol} = 0.01$, $p_{meas} = 0.05$

(d) $p_{depol} = 0.02$, $p_{meas} = 0.05$

Figure 5.6: Performance of error mitigated 2-fragment tomographic circuit cutting reconstruction under 2-qubit depolarizing gate noise (Eq. (5.17)) with $p_{depol} = 0.01$ (left) or $p_{depol} = 0.02$ (right), local symmetric readout error with $p_{meas} = 0.01$ (top) or $p_{meas} = 0.05$ (bottom), and single qubit gate depolarizing error of $p_1 = 10^{-4}$. The two-qubit depolarizing noise parameter was Errors on single qubit gates are fixed to 10^{-4}. Cut circuit reconstruction was compared using linear inversion (LIN), constrained least-squares (CLS), and readout error mitigated CLS (MEMCLS) tomography fitters both with and without dominant eigenvalue truncation (DEVT) mitigation. The original circuit (uncut) was measured with and without M3 readout error mitigation for comparison.

required over standard CLS or basic LIN tomography fitting, since DEVT alone is sufficient to mitigate the effect of both depolarizing gate error and measurement readout error on quantum circuits. However, as the size of the fragments increases, so does the number of conditional qubits and the effect of measurement

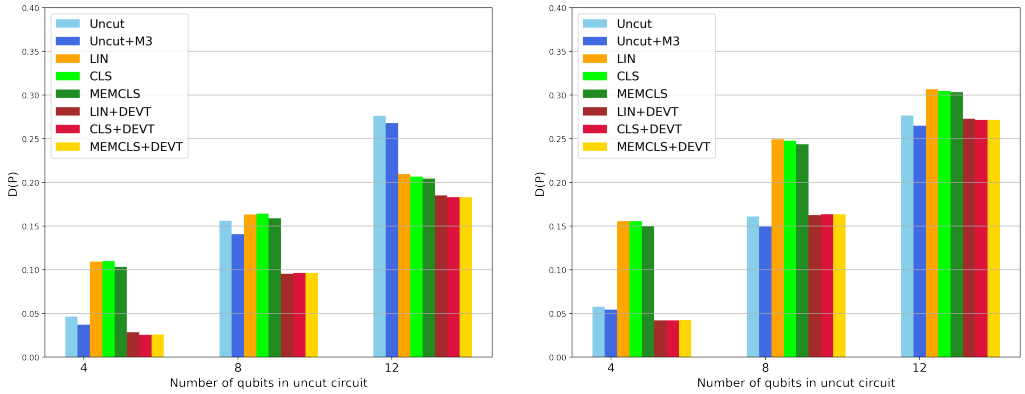
error on them, making measurement error more and more dominant. Thus the performance gap between MEMCLS and CLS with DEVT lowers with increasing size of fragments. Moreover, numerical data suggests that the rate of increase of trace distance increases with the increase in p_{depol} and the number of qubits in the fragment, which is expected. If one considers that circuit error in a layer can be approximated as a depolarizing error, then $1 - D(P)$ obtained using DEVT would scale as $\mathcal{O}(nmp)^2$, where n is the number of qubits, m is the number of gate layers, and p is the layer depolarizing error probability (See Appendix A.9 for details).

Depolarizing noise was expected to be the most favorable case for DEVT. However, it is not a realistic model for most quantum devices. A more general mixed-unitary model is a general Pauli channel with different error rates. Here, a biased Pauli channel was considered where the Z error term is more likely than the X or Y terms. This is given by the map

$$\mathcal{E}_{\text{pauli}}(\rho) = (1 - (3 + b)p)\rho + pX\rho X + pY\rho Y + p(1 + b)Z\rho Z \tag{5.18}$$

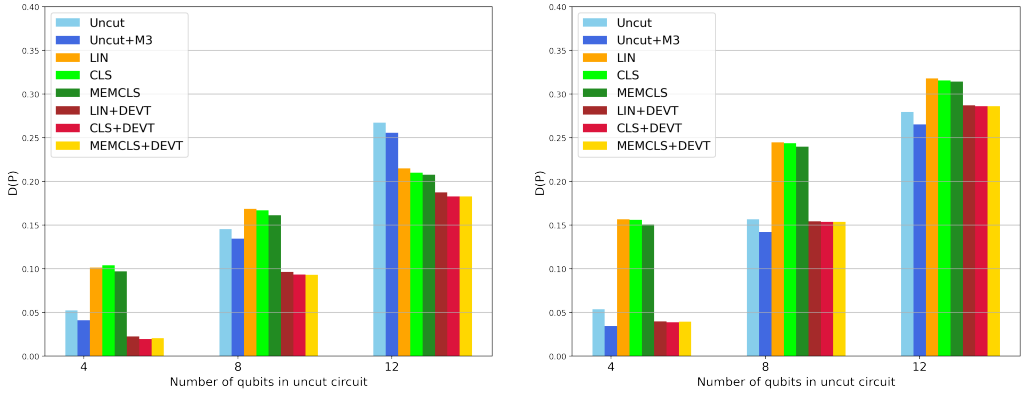
where p is the X and Y error probability, and b is a bias term added to the Z error. The simulations were with values of $p = 0.01$ and 0.02 in Fig. 5.7 (a) and (b) respectively with a bias term of $b = 0.1$, and in Fig. 5.7 (c) and (d) respectively with a bias term of $b = 0.5$. An n-qubit Pauli noise channel is defined as the tensor product of individual qubit noise channel $(\mathcal{E}_{\text{pauli}})^{\otimes n}$.

While DEVT does not improve the results as much as for depolarizing noise for 2-qubit gates with an error probability of 0.01 and for both tomography fitters and both values of bias, it still provides an advantage over both the uncut circuit and the cut circuit without DEVT. However, when the probability of gate error is increased to 0.02, for a bias of 0.1, it is noticed that DEVT hardly provides any improvement over the uncut circuit. In fact, for the 12 qubit circuit, when the bias is 0.5, even DEVT provides a result that is slightly worse than that of the uncut circuit. Appendix A.10 discusses that the form of the output density matrix for the biased Pauli noise model deviates from Eq. (5.11), leading to a poorer result as compared to the uniform depolarization noise model. However, these numerical



(a) Two qubit gate error $p = 0.01$ and bias $b = 0.1$

(b) Two qubit gate error $p = 0.02$ and bias $b = 0.1$



(c) Two qubit gate error $p = 0.01$ and bias $b = 0.5$

(d) Two qubit gate error $p = 0.02$ and bias $b = 0.5$

Figure 5.7: Performance of error mitigated 2-fragment tomographic circuit cutting reconstruction under a 2-qubit tensor product biased Pauli error channel (Eq. (5.18)) with $p_X = p_Y = p$, and $p_z = p(1 + b)$ with probabilities $p = 0.01$ (left) and $p = 0.02$ (right), and biases $b = 0.1$ (top) and $b = 0.5$ (bottom), local symmetric readout error with $p_{meas} = 0.05$, and single qubit gate depolarizing error of $p_1 = 10^{-4}$. Cut circuit reconstruction was compared using linear inversion (LIN), constrained least-squares (CLS) tomography fitters, and readout error mitigated CLS (MEMCLS) both with and without dominant eigenvalue truncation (DEVT) mitigation. The original circuit (uncut) was measured with and without M3 readout error mitigation for comparison.

results clearly suggest that DEVT provides an improvement over circuit cutting without DEVT in every scenario, thus consolidating its necessity for circuit cutting with noise.

In both Fig. 5.6 and 5.7 it can be observed that the reconstruction error increases by a larger amount when increasing the gate noise parameter for all tomographic circuit cutting methods than is observed when measuring the uncut circuit. This effect was consistent across all noise models considered henceforth and requires further investigation.

5.4.3 Non-Mixed Unitary Gate Errors

Next two other representative cases of non-mixed-unitary error were considered, amplitude damping and coherent noise, both of which result in an error map, not of the form in Eq. (5.11), and hence should be unfavorable for DEVT. For these simulations, measurement readout error was not included so as to asses DEVT for these gate errors without including improvement from its effectiveness for mitigating measurement readout errors.

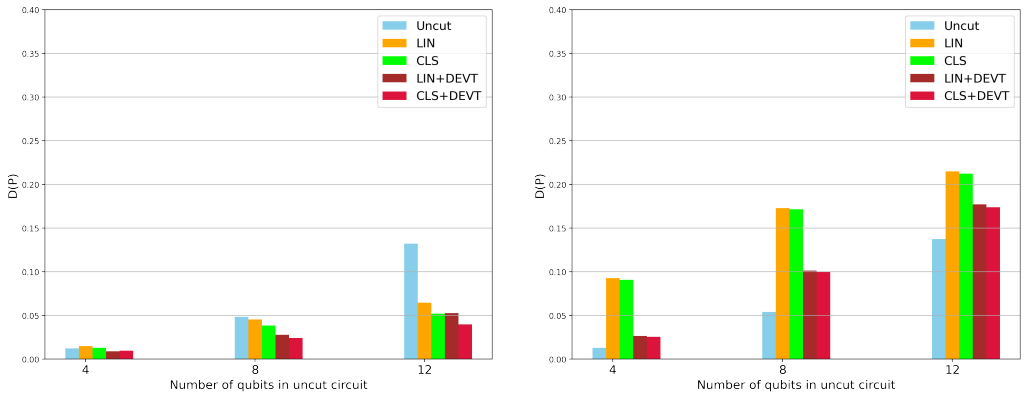
First, a 2-qubit gate error consisting of a tensor product of 1-qubit amplitude damping channels $\mathcal{E}_{amp}(\rho) = K_0\rho K_0^\dagger + K_1\rho K_1^\dagger$ with

$$K_0 = \begin{pmatrix} 1 & 0 \\ 0 & \sqrt{1-\gamma} \end{pmatrix} \quad K_1 = \begin{pmatrix} 0 & \sqrt{\gamma} \\ 0 & 0 \end{pmatrix} \tag{5.19}$$

was considered where the simulation was for damping parameter values of $\gamma = 0.001$, shown in Fig. 5.8 (a), and $\gamma = 0.01$, shown in Fig. 5.8 (b). It is observed that for $\gamma = 0.001$, when the channel can be considered to be very close to identity, DEVT, and circuit cutting in general, are able to attain improved performance. For $\gamma = 0.01$, both LIN and CLS with DEVT provide an improvement over either fitter without DEVT, however, it is not able to match the performance of the full circuit, and in fact, the cut circuit reconstruction appears more sensitive to the gate errors than the uncut circuit.

Next, a coherent error of the form

$$U_{err} = exp(-i\Delta\theta H_{CNOT}), \tag{5.20}$$

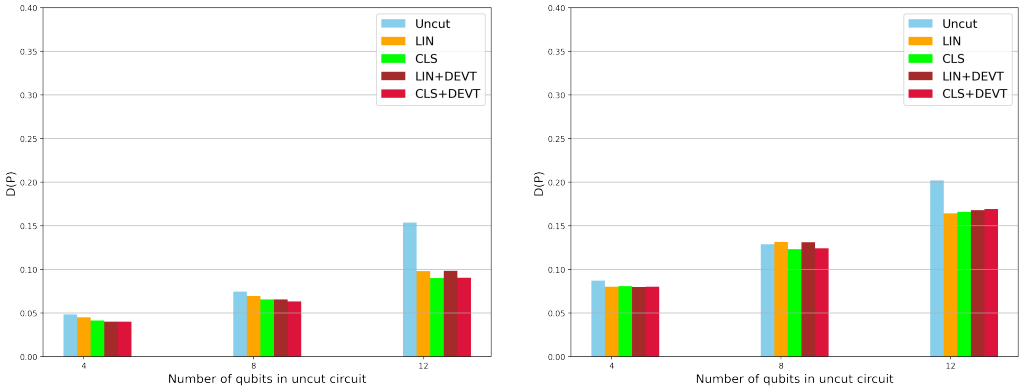


(a) Amplitude damping error with $\gamma = 0.001$ (b) Amplitude damping error with $\gamma = 0.01$

Figure 5.8: Performance of error mitigated 2-fragment tomographic circuit cutting reconstruction under a 2-qubit tensor product amplitude damping error channel (Eq. (5.19)) with damping parameter $\gamma = 0.001$ (a), and $\gamma = 0.01$ (b). Cut circuit reconstruction was compared using linear inversion (LIN) and constrained least-squares (CLS) tomography fitters, both with and without dominant eigenvalue truncation (DEVT) mitigation. Direct measurement of the original circuit (uncut) is shown for comparison.

was considered where $H_{CNOT} = log(U_{CNOT})/(-i)$ is the generator of a CNOT as a rotation gate, which is an approximate model of coherent errors due to imperfect gate calibration. The results for values of $\Delta\theta = \frac{\pi}{64}$ and $\Delta\theta = \frac{\pi}{32}$ are shown in Fig. 5.9 (a) and Fig. 5.9 (b) respectively. In this case, it was observed that DEVT provides essentially no improvement to regular tomography fitting, though importantly it can be seen that it also does not make the circuit cutting reconstruction significantly worse. One additional observation is that for the largest fragment size, the cut circuit performs better than the uncut circuit, likely due to there being fewer total gates for the coherent error to accumulate in a single fragment. Moreover, for both of these noise models, measurement error was not considered, and therefore, MEMCLS essentially becomes equivalent to CLS.

For the case of amplitude damping, the gate errors will also affect the conditioning qubit measurement outcomes, which could amplify errors in the tomographic reconstruction. This is because, for such a noise model, not only the individual circuit fragments are expected to be erroneous, but also the conditional qubits are



(a) Coherent error with $\Delta\theta = \frac{\pi}{64}$ (b) Coherent error with $\Delta\theta = \frac{\pi}{32}$

Figure 5.9: Performance of error mitigated 2-fragment tomographic circuit cutting reconstruction under a 2-qubit tensor product coherent rotation error channel (Eq. (5.20)) with rotation error $\Delta\theta = \pi/64$ (a), and $\Delta\theta = \pi/32$ (b). Cut circuit reconstruction was compared using linear inversion (LIN) and constrained least-squares (CLS) tomography fitters, both with and without dominant eigenvalue truncation (DEVT) mitigation. Direct measurement of the original circuit (uncut) is shown for comparison.

accumulated towards some value, thus making the entire reconstruction severely faulty. Similarly, for coherent noise, the noisy density matrix is expected to deviate significantly from the ideal one, so that the overlap of the largest eigenvalue with the ideal state lowers. DEVT is not expected to generate fruitful results in such scenarios [Koc21]. However, the numerical results show that DEVT still achieves a better performance than simple circuit cutting even under such scenarios. Therefore, it seems safe to deduce that if circuit cutting is used in such noisy scenarios, it is still not detrimental to apply DEVT.

5.4.4 DEVT with twirled noise

The numerical results verify that DEVT is not very useful for noise models that do not conform to the form of Eq. (5.11). However, a general method to convert any quantum channel into a mixed unitary channel is Twirling. In this method, the gates in a quantum circuit are padded with single qubit gates such that the

overall functionality remains constant. However, these extra padding gates rotate, or twirl the noise in the channel. This twirling process is repeated multiple times, with the twirling gates selected uniformly at random. The average over multiple such twirled instances is shown to generate a mixed unitary channel. The most widely used form of twirling is Pauli twirling, where the twirling gates are sampled uniformly at random from the single qubit Pauli gates [WE16]. For example, Fig. 5.10 shows an instance where a CNOT gate is padded with arbitrary Pauli gates. Note that the gates in the left padding can be chosen arbitrarily, whereas the right padding is selected to ensure that the overall functionality remains the same.

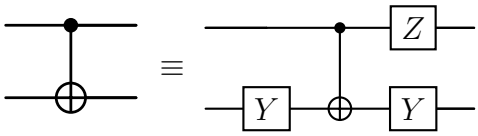
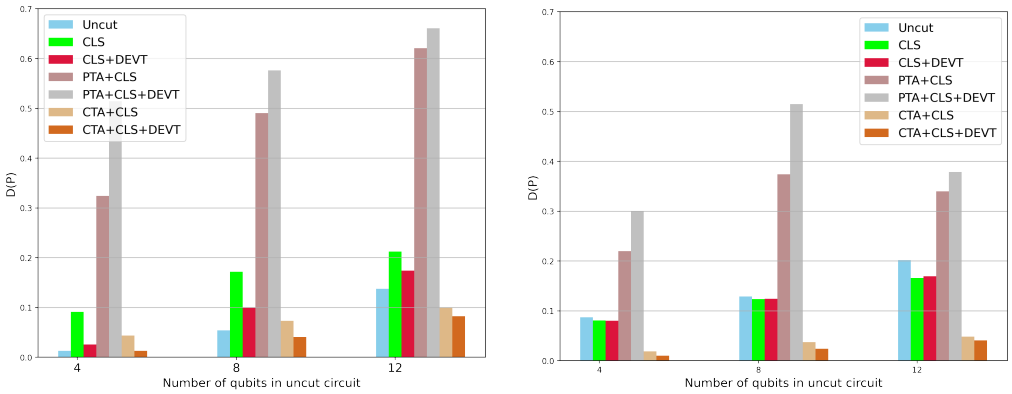


Figure 5.10: An example of Pauli twirling on CNOT gate

In the case of an amplitude damping channel Pauli twirling will result in a biased Pauli channel with $p_x = p_y = \frac{\gamma}{4}$ and $p_z = \frac{(1-\sqrt{1-\gamma})^2}{4}$, which in terms of the Pauli noise model considered in Sec. 5.4.2 corresponds to a negative bias parameter, i.e., the channel has a higher probability of Pauli X or Y error than that of Pauli Z error. For $\gamma = 0.01$, $p_x = p_y = 0.0025$, and $p_z = 6 \times 10^{-6}$. As shown in Fig. 5.7, DEVT will be expected to perform best when the resulting Pauli channel is closer to a combination of 1 or more depolarizing channels on any of the collections of subsystems; in other words, when the bias is close to 0. On the other hand, Clifford twirling [MGE12] is shown to result in an average channel that is depolarizing in nature. Therefore, naturally, DEVT is expected to perform well on such a channel. However, the set of Clifford gates contains CNOTs, which itself is a prominent source of noise in current quantum devices. Therefore, using Clifford gates to twirl a channel, in reality, may end up increasing the noise even more.

Fig. 5.11 shows the results for simulating both the Pauli-twirled approximation (PTA) and Clifford-twirled approximation (CTA) to the amplitude damping and coherent error noise models from Sec. 5.4.3. The CTA is performed via numerical methods, without running the twirling under noise; in other words, it assumes ideal



(a) Amplitude damping error with $\gamma = 0.01$ (b) Coherent error with $\Delta\theta = \frac{\pi}{32}$

Figure 5.11: Performance of error mitigated 2-fragment tomographic circuit cutting reconstruction using the Pauli-twirled approximation (PTA) and Clifford twirled approximation (CTA) of 2-qubit tensor product amplitude damping noise with $\gamma = 0.01$ (a) and coherent noise with $\Delta\theta = \frac{\pi}{32}$ (b). Cut circuit reconstruction was compared using linear inversion (LIN), constrained least-squares (CLS) tomography fitters, both with and without dominant eigenvalue truncation (DEVT) mitigation, and PTA or CTA. Direct measurement of the original circuit (uncut) is shown for comparison.

twirl gates. The results from Fig. 5.11 indicate that, under this assumption, CTA should produce a channel that is amenable to DEVT mitigation. However, such an assumption is not practical in reality. In Fig. 5.11 it is observed that applying DEVT to the biased Pauli channel resulting from PTA of these noise models leads to significantly worse results compared to the un-twirled noise models. These results indicate that in the context of circuit cutting Pauli twirling should only employed if the resulting noise is not highly biased and close to a depolarizing channel.

5.5 Scalability of tomographic circuit cutting

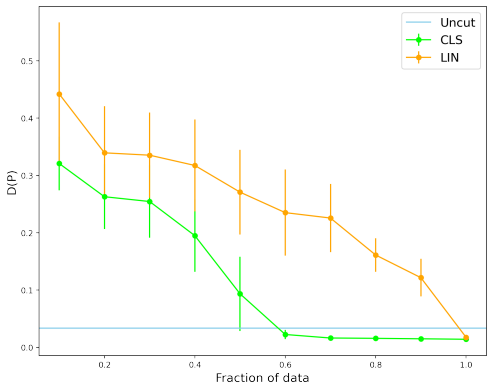
Tomography itself is not a very scalable process; process tomography scales as 12^k with the number of qubits k. The method of conditional tomography [PSSO21]

arrests the value of k to a small number per fragment under the assumption that the number of cut qubits per fragment is small. However, even then conditional process tomography is unscalable beyond a few cut qubits. Moreover, the number of conditional tomography increases exponentially with the number of conditional qubits as well. This section addresses the possibility of reducing the resources required for tomographic circuit cutting in terms of (i) using partial tomographic data, and (ii) the number of conditional tomography experiments.

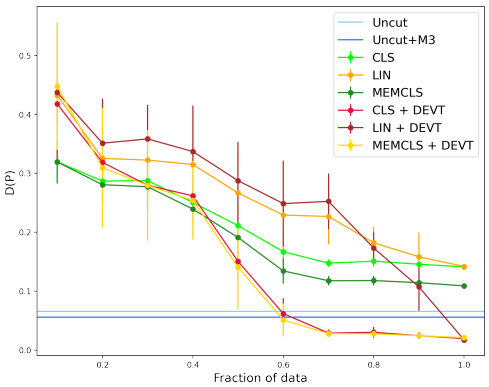
5.5.1 Circuit cutting with partial data

A k-qubit conditional tomography experiment using the standard Pauli basis and all measurement outcomes require the execution of 12^k quantum circuits from the 4^k preparation states and 3^k measurement bases respectively. This means that full tomography is typically only practical for 2-3 qubit fragments in the process tomography case, or up to 5-6 qubits for state tomography fragments. For a larger number of qubits, the classical post-processing required for linear inversion tomography can be significantly faster than for constrained least-squared tomography, which in the basic implementation of linear-least squares requires storing the full basis matrix of all vectorized basis elements. A natural question is whether partial tomography techniques are suitable for circuit cutting to reduce the number of experiments that need to be run. While there are many proposals for more scalable tomography fitters, this subsection investigates the two standard fitters, linear inversion and constrained least squares, with partial data.

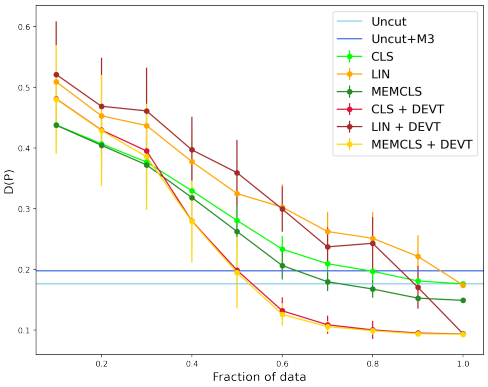
Using the same experiment data as in Sec. 5.4.2 with a 2-qubit depolarizing gate error of $p_2 = 0.01$, 1-qubit depolarizing gate error of $p_1 = 10^{-4}$ and symmetric readout error of $p_{meas} = 0.05$, tomographic reconstruction was performed using a randomly sampled fraction f of the full tomographic data ranging from $f = 0.1$ to $f = 1$, where the 100% case corresponds to the previously presented results. The results corresponding to partial tomography are shown in Fig. 5.12. For each data point the average of 10 trials over random basis selection were considered for both the LIN and CLS. Fig. 5.12 shows that when using partial data the CLS



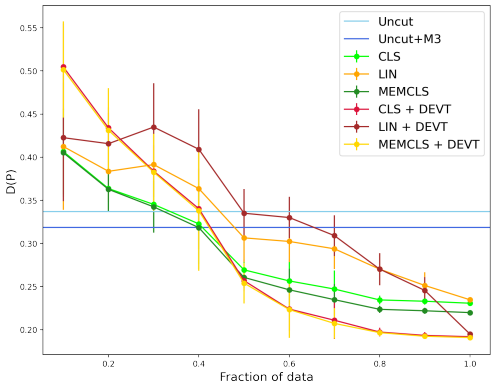
(a) Noiseless data where the uncut circuit contains 4 qubits



(b) Noisy data where the uncut circuit contains 4 qubits



(c) Noisy data where the uncut circuit contains 8 qubits



(d) Noisy data where the uncut circuit contains 12 qubits

Figure 5.12: Performance of error mitigated 2-fragment tomographic circuit cutting reconstruction using partial tomography data. Data is sampled as a subset of full data from Sec. 5.4.2 and averaged over 10 samples per data point. The noise model is a 2-qubit depolarizing gate noise with $p_{depol} = 0.01$, local symmetric readout error with $p_{meas} = 0.05$, and single qubit gate depolarizing error of $p_1 = 10^{-4}$. Cut circuit reconstruction was compared using linear inversion (LIN), constrained least-squares (CLS) tomography fitters, and readout error mitigated CLS (MEMCLS) both with and without dominant eigenvalue truncation (DEVT) mitigation. The original circuit (uncut) was measured with and without M3 readout error mitigation for comparison.

fitter greatly outperforms linear inversion. This is to be expected as it has been shown that the positive constraints added to linear least squares are equivalent to compressed sensing tomography [KKD15].

In the absence of noise, Fig. 5.12 (a) shows that CLS tomography performs at its maximum value when $f \simeq 0.6$. The standard deviation also drops to ~ 0 as the value of trace distance saturates. However, for LIN, the trace distance lowers almost linearly with an increasing fraction of data and attains its optimal trace distance only when the entire data set (i.e., $f = 1$) is used. This holds true in the presence of depolarizing gate and symmetric readout noise in Fig. 5.12 (b), (c), and (d) respectively, though when noise is included the fraction of data required to match the full data case also increases. This is to be expected as more noisy states and channels are higher ranks and will be less suitable with compressed sensing. Furthermore, as the number of qubits in the circuit increases, so does the effect of noise and the saturation point shifts towards higher values of f. In all cases with measurement error, the results show that MEMCLS has slightly superior performance to CLS for all fractions of data, with the improvement increasing slightly the fraction of data used.

When including DEVT mitigation it is observed that applying DEVT performs markedly better with partial data when using CLS and MEMCLS than with LIN. With both CLS and MEMCLS, DEVT provides a noticeable improvement when using $f > 0.5$, i.e., 50% of the data, with the exact value changing slightly with the size of the fragments, while for smaller fractions of data, it increases the reconstruction error. For LIN, however, DEVT is only beneficial for $f > 0.8$.

In conclusion, if partial tomography measurement is used to reduce the overall number of experiments required for circuit cutting reconstruction, then there is a noticeable difference between CLS and LIN fitters, both with and without DEVT, and CLS should be strongly preferred over LIN. Furthermore, DEVT is still beneficial to reduce the overall error in the circuit cutting reconstruction with partial data using $f > 0.5$ of the data.

5.5.2 Reducing the number of conditional tomography experiments

Poor scalability of tomography limits the number of cut qubits per fragment that is feasible in applications. The previous subsection addressed the issue of using tomography with partial data to improve the scalability. Furthermore, even after reconstructing all conditional fragment components for each fragment, there are 2^n tensor contractions that must be performed to reconstruct a full probability distribution of an n-qubit uncut circuit. However, there are myriads of problems in quantum chemistry, combinatorial optimization, quantum machine learning, etc. that only require computing the expectation value of low-weight Pauli observables, not the full distribution, and circuit cutting can be implemented more efficiently in these cases to reduce the exponential number of tensor contractions to a polynomial number.

In order to evaluate the expectation value of a weight $d < m$ Pauli operator for a fragment $\mathcal{F}$ with k cut qubits and m qubit measurements from the original circuit, it suffices to evaluate the conditional components T_s only over the d non-identity qubits corresponding to the observable, and marginalize over the rest. This reduces the complexity of tensor contractions required to compute $\langle P \rangle$ to $\mathcal{O}(2^d \cdot \binom{m}{d})$. For a fixed d, 2^d is $\mathcal{O}(1)$, and $\binom{m}{d}$ is $\mathcal{O}(m^d)$, which becomes the effective complexity of tensor contraction.

To take a more concrete example, consider the estimation of a Hamiltonian

$$H = \sum_i c_i Z_i + \sum_{i \neq j} \chi_{ij} Z_i Z_j$$

which requires only one and two-body interactions over the qubits. Such Hamiltonians are extremely common in Quantum Chemistry and other hardware-efficient near-term applications. For such a Hamiltonian, there will be $\binom{m}{2}$ weight-2 ob-

servables, and m weight-1 observables per fragment. Therefore, the overall complexity of tensor contraction boils down to $\{2^2 \cdot \mathcal{O}(m^2) + 2 \cdot \mathcal{O}(m)\}$ which is $\mathcal{O}(m^2)$. For most practical purposes, m scales as $\mathcal{O}(n)$ where n is the total number of qubits in the fragment. For example, when there are k cut qubits in a fragment, $m = n - k$. Therefore, the complexity of tensor contraction can be lowered to $\mathcal{O}(n^2)$ if finding expectation values of weight-2 Pauli observables is sufficient. It remains an interesting open problem to determine problems of interest where these proposed methods can be implemented to make circuit cutting more scalable.

5.6 Summary

This study explored how an error-mitigated tomography approach to circuit cutting can improve the overall performance of evaluating quantum circuit outputs in the presence of gate and measurement noise. This builds on the previous work in [PSSO21] which showed the advantages of using tomography over the original circuit cutting method [PHOW20] in noiseless ideal simulations where only errors due to measurement sampling statistics were included. Across all simulations, it is observed that in the presence of gate noise, the circuit cutting reconstructed probabilities exhibit a greater sensitivity to the gate noise strength than the uncut circuit. This was true across all noise models considered and emphasizes the importance of error mitigation techniques that can be applied to circuit cutting. The results from simulations demonstrated that in the presence of symmetric readout error measurement noise and certain forms of gate noise, applying DEVT can greatly improve tomographic circuit cutting estimates during tomography reconstruction. For non-symmetric readout noise, this can be made to look symmetric by Pauli twirling of the measurements to randomly flip the expected bit outcomes and then correcting in post-processing [vdBMT22b]. The form of the gate noise is important for DEVT to be effective, and in particular uniform noise close to a depolarizing channel is most effective with DEVT. While DEVT was found to provide little advantage for amplitude damping and coherent noise, it did not significantly increase the error in the reconstruction. One important result is that

DEVT was shown to perform very poorly for highly biased Pauli noise, and could dramatically increase the error in the reconstruction. This is an important consideration to keep in mind if techniques such as Pauli-twirling are used to convert gate noises to Pauli channels. One possible way that this might be circumvented is to use probabilistic error amplification techniques, such as used in [LB17], to make a highly biased gate noise more uniform, and hence more amenable to DEVT mitigation.

Another important consideration when performing tomographic circuit cutting is the choice of tomography fitting procedure. The required time for post-processing can be a significant factor in applying circuit cutting to problems that have more than a handful of circuit fragments to be evaluated. However, since the tomographic reconstruction of each fragment is independent from other fragments this can be easily parallelized on classical computing resources. In this study, two of the most commonly used full tomography fitters, namely linear inversion and constrained least-squares optimization, were compared. In typical tomography applications, the main trade-off between these two fitters is that linear inversion fitting is significantly faster, while constrained least squares is more accurate, especially when including readout error mitigation via noisy basis elements in the fitting or only using partial tomography data. In the context of circuit cutting with DEVT mitigation, it was observed that linear inversion was comparable to constrained least squares when full tomographic data was available, and in particular, DEVT was effective at mitigating the effect of measurement errors in tomography without requiring the specialized measurement-error mitigated conditional tomographic fitter proposed in this study. This can allow for significantly faster tomographic post-processing.

If circuit cutting is to be considered for applications requiring a number of cut qubits that are not realistic for obtaining full tomographic data then partial tomographic techniques will be required for the reconstruction. The simulation results indicate that in this case constrained least squares method performed significantly better than linear inversion, both with and without DEVT, since this method exhibits properties of compressed sensing. This also indicates that estimation tech-

niques such as classical shadow tomography are most likely not an important consideration for circuit cutting since techniques such as shadow estimation [HKP20], which are equivalent to linear inversion partial tomography in the Pauli basis as considered here, should not be expected to provide a benefit for circuit cutting problems over constrained tomography fitting methods, and furthermore are not suitable for use with DEVT as a mitigation method to improve their performance with partial data.

Part II

Error correction for reliable quantum computation

CHAPTER 6

Quantum error correcting code for ternary logic

Contents

6.1 Introduction

The first part of this thesis dealt with near-term quantum systems, where the number of qubits is not enough to incorporate error correction. Therefore, methods such as the efficient design of circuits to lower the number of noisy gates, circuit cutting, etc. were employed to lower the effect of noise. However, error correction is necessary for arbitrary long computation. This part of the thesis looks into the efficient design of error correction circuits, and the challenges of extending NISQ methods to the error correction era.

Quantum systems are inherently multi-valued. A general d-dimensional quantum state called a qudit, consists of 0, 1, ..., and $d-1$ levels. An arbitrary qudit has the form $|\psi\rangle_d = \sum_{k=0}^{d-1} \alpha_k |k\rangle$, where $\alpha_k \in \mathbb{C}\ \forall\ k$, and $\sum_{k=0}^{d-1} |\alpha_k|^2 = 1$. Higher dimensional quantum systems can express a larger computational space using less number of qudits. For example, an n qubit system has a computational space of 2^n. This computational space can be achieved using $k < n$ qudits, where $k = \lceil \frac{n}{logd} \rceil$. In terms of practical implementation, for example in superconductors, a qudit is an anharmonic oscillator, whose energy levels are the levels of the qudit. However, the gap $\Delta E_{i,i+1}$ between two adjacent energy levels E_i and E_{i+1} decreases with increasing i; the system essentially becoming continuous as $i \to \infty$ [GS18]. Therefore, it is often a difficult engineering problem to effectively handle computations involving higher dimensions, thus increasing the error in the system.

Nevertheless ternary quantum system (or qutrit), i.e., a system involving levels $|0\rangle, |1\rangle$ and $|2\rangle$ have been realized experimentally [CLKAGG22, GJAE+20, GCK+21]. Qutrit systems have been shown to outperform qubit systems in cryptographic protocols [BPP00], quantum random walk [AAKV01, Won15, SMS+18,

SMS+21], and decomposition of unitary matrices [GBD+19, SMS+20]. The decomposition of unitary matrices will be explored in more detail in Chapter 8.

The realization of arbitrary large ternary quantum systems mandates error correction. In [Cha97] the author suggested that a higher dimensional quantum error correcting code (QECC) can be carried over from its binary counterpart. In other words, a binary QECC can be extended to a d-dimensional QECC by replacing the binary error channel with its d-dimensional formulation. This chapter shows the extension of such a binary error channel into a ternary one and presents a ternary version of the 9-qubit QECC [Sho95].

6.2 Errors in ternary quantum system

An arbitrary ternary quantum system, or a qutrit, can be represented as $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle + \gamma|2\rangle$, where $\alpha, \beta, \gamma \in \mathbb{C}$ and $|\alpha|^2 + |\beta|^2 + |\gamma|^2 = 1$. Any arbitrary unwanted unitary operator can be considered as an error on the qutrit. In general, for error correction, the notion is to find a spanning set of unitaries such that correcting the errors from the spanning set is sufficient to correct any error on the system. For qubit systems, Pauli matrices form this spanning set. A well-known trace 0 spanning set for 3×3 unitaries is the Gell-Mann matrices [GM62]. This is a set of eight matrices as shown below.

$$\lambda_1 = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \quad \lambda_2 = \begin{pmatrix} 0 & -i & 0 \\ i & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \quad \lambda_3 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

$$\lambda_4 = \begin{pmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \end{pmatrix} \quad \lambda_5 = \begin{pmatrix} 0 & 0 & -i \\ 0 & 0 & 0 \\ i & 0 & 0 \end{pmatrix} \quad \lambda_6 = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix}$$

$$\lambda_7 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & -i \\ 0 & i & 0 \end{pmatrix} \quad \lambda_8 = \frac{1}{\sqrt{3}} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -2 \end{pmatrix}$$

However, the first seven of the Gell-Mann matrices map a qutrit to a qubit. For example, the action of λ_1 on $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle + \gamma|2\rangle$ is given by

$$\begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} \alpha \\ \beta \\ \gamma \end{pmatrix} = \begin{pmatrix} \beta \\ \alpha \\ 0 \end{pmatrix} = \beta|0\rangle + \alpha|1\rangle.$$

In other words, λ_1 mapped the qutrit to a qubit. Therefore, the Gell-Mann matrices are not suitable as a spanning basis for unitary errors on ternary quantum systems. So, in the spirit of errors in binary systems, different types of bit and phase errors for ternary quantum systems are studied first. Both the bit and phase errors can affect the entire three-dimensional space or any two-dimensional subspace of the vector. The former is termed as *ternary errors* and the latter as *binary errors*.

6.2.1 Bit errors on qutrits

As discussed above, it is possible to have errors in a three-dimensional system whose support is one or two dimensions only. In other words, $\exists\ r_i$ and c_j, which are respectively the $i-th$ row and the $j-th$ column of the error matrix, to be identical to that of the identity operator. In a ternary system, there can be three such pairwise swaps, namely X_{01}, X_{12}, X_{20}. A single qutrit pairwise swap error operates only on any two of the three basis states, i.e., the amplitudes of two out of the three basis states get swapped in the presence of such an error. The matrices corresponding to these errors are as follows:

$$X_{01} = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}; \qquad X_{01}|\psi\rangle = \alpha|1\rangle + \beta|0\rangle + \gamma|2\rangle$$

$$X_{12} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix}; \qquad X_{12}|\psi\rangle = \alpha|0\rangle + \beta|2\rangle + \gamma|1\rangle$$

$$X_{20} = \begin{pmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{pmatrix}; \qquad X_{20}\left|\psi\right\rangle = \alpha\left|2\right\rangle + \beta\left|1\right\rangle + \gamma\left|0\right\rangle.$$

These matrices are *self-adjoint*. Hence if any of these errors occur, applying the same error matrix again can correct it. In addition to bit-flip errors, purely ternary errors, affecting all three dimensions non-trivially, are possible. These errors cause cyclic shifts of the basis vectors. Two types of shifts are possible - clockwise shift ($0 \to 1 \to 2$) and anticlockwise shift ($0 \leftarrow 1 \leftarrow 2$). The mathematical formulation of *clockwise shift* (X_1) is $\left|j\right\rangle \xrightarrow{X_1} \left|j+1\right\rangle$ mod 3 and that of *anticlockwise shift* (X_2) is $\left|j\right\rangle \xrightarrow{X_2} \left|j-1\right\rangle$ mod 3. It is interesting to note that the stabilizer proposed by Gottesman in [Got99] for higher dimensional errors corresponds to the *clockwise shift* (X_1).

It can be easily verified that the respective matrices corresponding to errors X_1 and X_2 are -

$$X_1 = \begin{pmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix} \qquad X_2 = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix}$$

The action of these cyclic shift errors on the error-free state $\left|\psi\right\rangle$ can be mathematically represented as

$$\begin{aligned} X_1\left|\psi\right\rangle &= \alpha\left|1\right\rangle + \beta\left|2\right\rangle + \gamma\left|0\right\rangle \\ X_2\left|\psi\right\rangle &= \alpha\left|2\right\rangle + \beta\left|0\right\rangle + \gamma\left|1\right\rangle. \end{aligned}$$

The matrices for *shift* errors are not self-adjoint. However, each type of shift error occurring twice in succession produces the other type of shift error, i.e., $X_2 = X_1^2$ and $X_1 = X_2^2$; further $X_1^{-1} = X_2$ and $X_2^{-1} = X_1$. Thus to correct an X_1 error, one can apply X_2 and vice-versa. It can be checked that any combination of these five types of errors (pairwise swap and shift) results in one of these five errors or identities. Thus these five errors exhaust the list of possible bit errors on a qutrit.

6.3 Phase errors on qutrits

Phase errors on qutrits are different from those on qubits. Dephasing error on a qubit adds a phase of $exp(i\phi)$ to the ideal state. It is possible to express $exp(i\phi)$ as

$$exp(i\phi) = cos(\phi)I + i.sin(\phi)Z$$

where $Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$. Therefore, it suffices to correct the phase error Z on a system. In a ternary system, it is more convenient to deal with phase errors being the cube root of unity ω. However, this initial argument keeps the form of Z error unchanged from binary systems and later shows that the phase of ω is easier to deal with in ternary systems.

On interaction with the environment, a qutrit can undergo rotation by some arbitrary angles, where the basis states $|1\rangle$ and $|2\rangle$ may incur different phase errors given by $e^{i\theta}$ and $e^{i\phi}$ respectively. Note that a phase can occur on $|0\rangle$ as well, which can be ignored as a global phase by suitably modifying the phases on $|1\rangle$ and $|2\rangle$. Such an error changes the error-free state $|\psi\rangle$ as

$$\begin{aligned} \alpha|0\rangle + \beta|1\rangle + \gamma|2\rangle &\rightarrow \alpha e^{ia_0}|0\rangle + \beta e^{ia_1}|1\rangle + \gamma e^{ia_2}|2\rangle \\ &= e^{ia_0}(\alpha|0\rangle + \beta e^{i(a_1-a_0)}|1\rangle + \gamma e^{i(a_2-a_0)}|2\rangle) \\ &\simeq \alpha|0\rangle + \beta e^{i\theta}|1\rangle + \gamma e^{i\phi}|2\rangle . \end{aligned}$$

The corresponding error operator is denoted as $R_{\theta\phi} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & e^{i\theta} & 0 \\ 0 & 0 & e^{i\phi} \end{pmatrix}$. Using the formula $e^{\pm i\theta} = cos\theta \pm isin\theta$, the error matrix can be represented up to a global phase of $e^{i\frac{\theta+\phi}{2}}$, as in Eq. (6.1).

$$[cos\tfrac{\theta}{2}\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} - isin\tfrac{\theta}{2}\begin{pmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{pmatrix}] \cdot [cos\tfrac{\phi}{2}\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} - isin\tfrac{\phi}{2}\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -1 \end{pmatrix}]$$

$$= cos\frac{\theta}{2}cos\frac{\phi}{2}\mathbb{I} - isin\frac{\theta}{2}cos\frac{\phi}{2}Z_1 - icos\frac{\theta}{2}sin\frac{\phi}{2}Z_2 \\ -sin\frac{\theta}{2}sin\frac{\phi}{2}Z_{12}. \tag{6.1}$$

where

$$Z_1 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad Z_2 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -1 \end{pmatrix} \quad Z_{12} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -1 \end{pmatrix} = Z_1Z_2 = Z_2Z_1$$

The action of these operators on an arbitrary qutrit is given by

$$\begin{aligned} \alpha|0\rangle + \beta|1\rangle + \gamma|2\rangle &\xrightarrow{Z_1} \alpha|0\rangle - \beta|1\rangle + \gamma|2\rangle \\ \alpha|0\rangle + \beta|1\rangle + \gamma|2\rangle &\xrightarrow{Z_2} \alpha|0\rangle + \beta|1\rangle - \gamma|2\rangle \\ \alpha|0\rangle + \beta|1\rangle + \gamma|2\rangle &\xrightarrow{Z_{12}} \alpha|0\rangle - \beta|1\rangle - \gamma|2\rangle \end{aligned}$$

Eq. (6.2) depicts the overall error model considered henceforth, which consists of bit errors, phase errors, and their combination, called Y errors. A different phase error, using the cube root of unity ω, is considered later in this section, and the spanning of any arbitrary unitary errors is shown later in Chapter 7.

$$E = a\mathbb{I}_3 + \sum_{i=1}^{2} b_i Z_i + \sum_{\substack{m,n=0 \\ m\neq n}}^{2} (c_{mn}X_{mn} + \sum_{j=1}^{2} d_{mnj}Y_{mnj}) \tag{6.2}$$

where $a, b, c_{mn}, d_{mnj} \in \mathbb{C}$ are constants. $\mathbb{I}_3$ is the 3×3 identity operator and $Y_{mnj} = iZ_jX_{mn}$ takes into account when both bit error (X) and phase error (Z) occur simultaneously. This error model tackles both binary as well as ternary bit and phase errors.

6.4 Shor code for qutrits

This section deals with the possibility of extending the 9-qubit QECC by Shor (discussed in Chapter 2) to the ternary regime. In accordance with the encoding scheme of Shor code, the information of a single qutrit $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle + \gamma|2\rangle$ is encoded into nine qutrits to form a logical qutrit $|\psi\rangle_L = \alpha|0\rangle_L + \beta|1\rangle_L + \gamma|2\rangle_L$, where

$$\begin{aligned}
|0\rangle_L &= \frac{1}{3\sqrt{3}}(|000\rangle + |111\rangle + |222\rangle)(|000\rangle + |111\rangle + |222\rangle) \\
&\quad (|000\rangle + |111\rangle + |222\rangle) \\
|1\rangle_L &= \frac{1}{3\sqrt{3}}(|000\rangle + \omega|111\rangle + \omega^2|222\rangle)(|000\rangle + \omega|111\rangle + \omega^2|222\rangle) \\
&\quad (|000\rangle + \omega|111\rangle + \omega^2|222\rangle) \\
|2\rangle_L &= \frac{1}{3\sqrt{3}}(|000\rangle + \omega^2|111\rangle + \omega|222\rangle)(|000\rangle + \omega^2|111\rangle + \omega|222\rangle) \\
&\quad (|000\rangle + \omega^2|111\rangle + \omega|222\rangle).
\end{aligned}$$

In short, the logical basis states can be represented as

$$|i\rangle_L = \frac{1}{3\sqrt{3}}(|000\rangle + \omega^i|111\rangle + \omega^{2i}|222\rangle)^{\otimes 3}$$

It is easy to check that $|0\rangle_L$, $|1\rangle_L$ and $|2\rangle_L$ are orthogonal to each other. In order to correct the error in Eq.(6.2), ancilla state(s) $|\zeta\rangle$ is entangled with the system. Finally, the ancilla state(s) is measured which gives a classical outcome called *error syndrome*. The error syndrome denotes the type of error that has occurred. The resultant state after entanglement of the ancilla state(s) $|\zeta\rangle$ is a superposition of the form

$$\begin{aligned}
cos\frac{\theta}{2}cos\frac{\phi}{2}\mathbb{I}|\psi\rangle|\zeta_I\rangle - isin\frac{\theta}{2}cos\frac{\phi}{2}Z_1|\psi\rangle|\zeta_{Z_1}\rangle - icos\frac{\theta}{2}sin\frac{\phi}{2}Z_2|\psi\rangle|\zeta_{Z_2}\rangle \\
-sin\frac{\theta}{2}sin\frac{\phi}{2}Z_{12}|\psi\rangle|\zeta_{Z_{12}}\rangle
\end{aligned}$$

where $|\zeta_i\rangle$ indicates the ancilla qubit with i-th error syndrome. Upon measurement of the ancilla qubits, the superposition collapses. If the ancilla state collapses with i-th syndrome, then the encoded state also collapses with the i-th error on the system.

$$\begin{aligned}\begin{pmatrix}1&0&0\\0&0&1\\0&1&0\end{pmatrix} &= \begin{pmatrix}1&0&0\\0&1&0\\0&0&1\end{pmatrix} + \begin{pmatrix}1&0&0\\0&\omega&0\\0&0&\omega^2\end{pmatrix} + \begin{pmatrix}1&0&0\\0&\omega^2&0\\0&0&\omega\end{pmatrix} \\ &+ \begin{pmatrix}0&1&0\\0&0&1\\1&0&0\end{pmatrix} + \begin{pmatrix}0&\omega&0\\0&0&1\\\omega^2&0&0\end{pmatrix} + \begin{pmatrix}0&\omega^2&0\\0&0&1\\\omega&0&0\end{pmatrix} \\ &+ \begin{pmatrix}0&0&1\\1&0&0\\0&1&0\end{pmatrix} + \begin{pmatrix}0&0&\omega\\1&0&0\\0&\omega^2&0\end{pmatrix} + \begin{pmatrix}0&0&\omega^2\\1&0&0\\0&\omega&0\end{pmatrix}\end{aligned} \tag{6.3}$$

6.4.1 Stabilizer formulation for ternary Shor code

A ternary quantum system can be perturbed with both *binary* and *ternary* errors. However, it is not trivial to deal with *binary* errors in this setting. However, any *binary* error can be represented as a linear combination of *ternary error*. For example, the error operator X_{12} can be written (up to a normalization factor) as a linear combination of shift and phase operators as shown in Eq. (6.3). Note that the error operators with angles ω and ω^2 can be considered as $R_{\theta\phi}$ errors for particular values of θ and ϕ. Hence, a code that can correct shift and phase errors can also correct pairwise swap errors occurring on qutrits.

6.4.2 Stabilizer structure for error detection

Gottesman defined stabilizers for higher dimensional spin systems as

$$X_d |j\rangle = |j+1\rangle \bmod \mathrm{d} \qquad Z_d |j\rangle = \omega^j |j\rangle$$

where d is the dimension of the quantum state. For qutrit systems, $d = 3$. Chapter 2 discussed the stabilizers for binary Shor code. Consider the first two stabilizers $S_1 = ZZIIIIIII$ and $S_2 = IZZIIIIII$. It is easy to see that if each Z is replaced by Z_d in the two stabilizers, they no longer commute. Therefore, S_1 and S_2 cannot be valid stabilizers for ternary Shor code. This mandates a different stabilizer structure altogether for the ternary Shor code. For this study, the stabilizers are selected to be

$$S_1 = Z_dZ_dZ_dIIIIII,\ S_2 = IIIZ_dZ_dZ_dIII,\ S_3 = IIIIIIZ_dZ_dZ_d$$
$$S_4 = X_dX_dX_dIIIIII,\ S_5 = IIIX_dX_dX_dIII,\ S_6 = IIIIIIX_dX_dX_d.$$

It can be verified that $[S_i, S_j] = 0\ \forall\ 1 \leq i, j \leq 6$. Stabilizers S_1, S_2, S_3 only checks for shift errors while stabilizers S_4, S_5, S_6 only checks for errors with ω, ω^2 phase. Table 6.1 notes the eigenvalues corresponding to stabilizers $Z_dZ_dZ_dIIIIII$ and $X_dX_dX_dIIIIII$ for different error states. The actions of the other stabilizers are similar. However, the stabilizers for shift errors can only detect the presence and the type of error, but not their location. Hence, a second step is required to find the location of the errors.

Table 6.1: Stabilizers for ternary errors

Error state	$Z_dZ_dZ_dIIIIII$	$X_dX_dX_dIIIIII$
$\|000\rangle + \|111\rangle + \|222\rangle$	1	1
$\|200\rangle + \|011\rangle + \|122\rangle$	ω^2	
$\|020\rangle + \|101\rangle + \|212\rangle$	ω^2	
$\|002\rangle + \|110\rangle + \|221\rangle$	ω^2	
$\|100\rangle + \|211\rangle + \|022\rangle$	ω	
$\|010\rangle + \|121\rangle + \|202\rangle$	ω	
$\|001\rangle + \|112\rangle + \|220\rangle$	ω	
$\|000\rangle + \omega\|111\rangle + \omega^2\|222\rangle$		ω^2
$\|000\rangle + \omega^2\|111\rangle + \omega\|222\rangle$		ω

6.4.3 Circuit for error correction

After applying the stabilizer $Z_dZ_dZ_dIIIIII$, if the eigenvalue is ω^2, it implies that X_2 error has occurred in any one of the first three qutrits. However, it cannot identify the qutrit which has incurred the error. Therefore, a second step is required which can compare the first three qutrits and identify the erroneous one. Note here that in the absence of error, the first three qutrits are always in the same state. This property can be exploited to find the mismatch in parity between different groups of two qutrits from the first three, thus identifying the erroneous one.

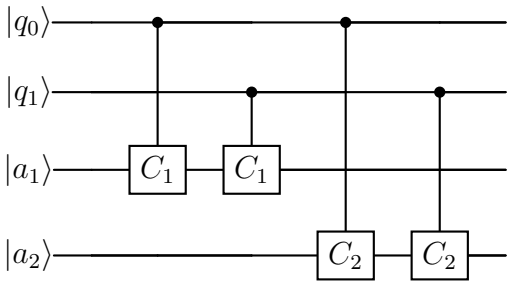


Figure 6.1: Circuit to compare the parity of two qutrits

Fig 6.1 shows the circuit that checks whether two qutrits are in the same state. In this circuit, $|q_0\rangle$ and $|q_1\rangle$ are the qutrits of interest, and $|a_0\rangle$ and $|a_1\rangle$ are the ancilla which stores the outcome of the parity measurement. The ancilla can be *qubits* only since the possible values to store are 0 and 1 corresponding to parity match or mismatch respectively. Using qutrits as an ancilla does not hamper the error correction procedure, but qutrits are not necessary. The action of the gates C_1 and C_2 are defined as

$$C_i : \texttt{if (control = i) then target = target + 1 (mod 3)}, \quad i \in \{1, 2\}.$$

The truth table of this circuit is shown in Table 6.2. The states of the ancilla qubits comprise the error syndrome. When the syndrome is 00, it implies that both the qutrits are in the same state. Any other error syndrome indicates that

the qutrits are in different states. Note that, the states of the ancilla qubits do not specify the exact states of the qutrits. Therefore, this does not collapse the qutrit superposition but provides only the parity information between the qutrits. However, the values in the ancilla qubit reveal sufficient information. For example, if the ancilla qubits are 10, it implies one of the two qutrits is in state 1, but does not reveal which one.

Table 6.2: Truth table for the circuit in Fig 6.1

$\|q_0\rangle$	$\|q_1\rangle$	$\|a_0\rangle$	$\|a_1\rangle$
0	0	0	0
0	1	1	0
0	2	0	1
1	0	1	0
1	1	0	0
1	2	1	1
2	0	0	1
2	1	1	1
2	2	0	0

Muthukrishnan and Stroud proposed a set of single and two-qubit ternary gates in [MS00] and showed their implementation in Ion-trap devices. These gates, often termed MS gates, are universal in that any ternary gate can be implemented by cascading one or more MS gates. $C2$ is one of the MS gates. Two other types of MS gates, namely MS+1 and MS+2, are defined as

$$MS + i\,|j\rangle = |j + i\rangle \bmod 3.$$

While C_1 is not an MS gate, it can be implemented using three MS gates as shown in Fig 6.2.

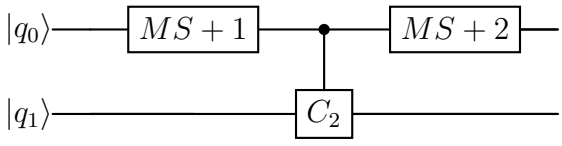


Figure 6.2: Implementation of C_1 gate using 3 MS gates

Fig 6.3 shows the circuit to determine whether the three qutrits are in the same state or not. In the figure, $|q_0\rangle - |q_2\rangle$ are the qutrits on interest which form the logical qutrit, and $|a_0\rangle - |a_3\rangle$ are ancilla *qubits* initialized in $|0\rangle$. Following the truth table from Table 6.2, if all the qutrits are in the same state, then the syndrome is 000. Otherwise one of the three bits will differ, which identifies the qutrit on which the error has occurred.

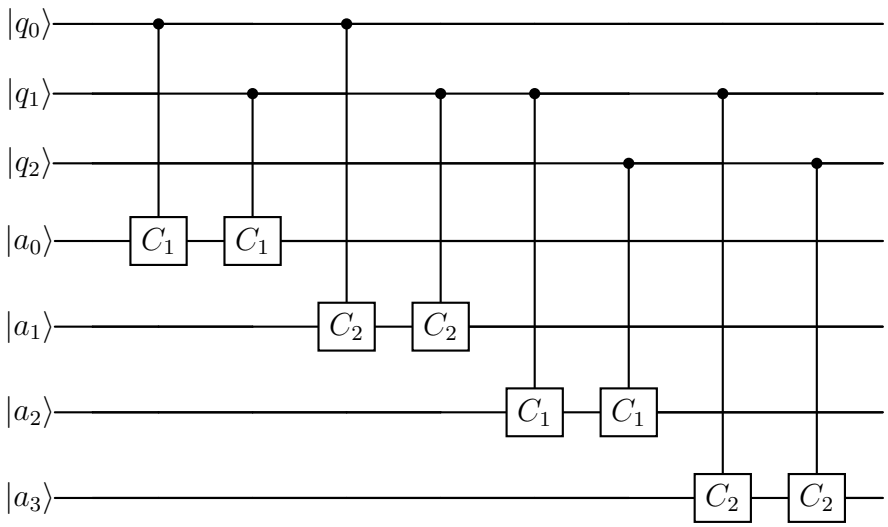


Figure 6.3: Circuit for qutrit error correction

Next, the problem of correcting phase errors (Z_1, Z_2 and Z_{12}) is addressed. In qubit systems, phase errors can be corrected by changing to the Hadamard basis, where phase errors behave like bit errors. The $\{|+\rangle, |-\rangle, ||\rangle\}$ basis for qutrits is equivalent to the Hadamard basis for qubits.

$$
\begin{aligned}
|+\rangle &= \frac{1}{\sqrt{3}}(|0\rangle + |1\rangle + |2\rangle). \\
|-\rangle &= \frac{1}{\sqrt{3}}(|0\rangle + \omega^2 |1\rangle + \omega |2\rangle). \\
||\rangle &= \frac{1}{\sqrt{3}}(|0\rangle + \omega |1\rangle + \omega^2 |2\rangle).
\end{aligned}
$$

However, it is clear that the bit-flip nature of the Hadamard basis does not hold true in a three-dimensional system. In order to correct a phase error in qutrit, the

three unitary matrices as proposed in [DW11] are considered:

$$H^{01} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 & 0 \\ 1 & -1 & 0 \\ 0 & 0 & \sqrt{2} \end{pmatrix} \quad H^{12} = \frac{1}{\sqrt{2}} \begin{pmatrix} \sqrt{2} & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 1 & -1 \end{pmatrix}$$
$$H^{20} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 0 & 1 \\ 0 & \sqrt{2} & 0 \\ 1 & 0 & -1 \end{pmatrix}.$$

These matrices are similar to Hadamard operation on two states while the third state is kept unchanged. Applying H^{01} on these states changes $|0\rangle$ and $|1\rangle$ to $|+\rangle$ and $|-\rangle$ respectively, while the state $|2\rangle$ remains unchanged. Hence, if there is a phase error on $|1\rangle$ with respect to $|0\rangle$, then it can be easily detected as it will flip the states $|+\rangle$ and $|-\rangle$. Similarly, by applying H^{20} any phase error between $|0\rangle$ and $|2\rangle$ can be detected. Since $Z_{12} = Z_1 Z_2$, correcting Z_1 and Z_2 one after the another corrects Z_{12}.

6.4.4 Performance analysis of ternary Shor code

The code proposed here is a repetition code, where each of the logical qutrits ($|0\rangle_L$, $|1\rangle_L$, $|2\rangle_L$) are arranged in three blocks of three qutrits each. This approach is similar to Shor code [Sho95] for qubits. If p is the probability that a single qutrit is affected by decoherence, then the probability that none of the nine qutrits decohere is $(1-p)^9$. This code fails if more than one qutrit incurs an error. The probability that at least two qutrits have error is $1-(1-p)^9-9p(1-p)^8 = 1-(1+8p)(1-p)^8 \approx 36p^2$. Hence, when the error probability is less than $\frac{1}{36}$, this technique provides an improved method to preserve the coherence of the qutrits. The performance of our proposed code is in accordance with the Shor code. However, this apparent similarity to Shor Code vanishes in the error correction process. Unlike Shor Code, the error correction is twofold - in the first step error is detected, and then its location is identified.

In the error model chosen for qutrits in this thesis, there are two independent bit errors (X_1, X_2) and two independent phase errors (Z_1, Z_2) and their product (Z_{12}). So there are three phase errors. Hence, according to Eq.(7.2), there can be six $Y_{mnj} = iZ_jX_{mn}$ errors. Reliable detection of these eleven errors and the error-free state demands each error state and the error-free state to be in different orthogonal subspaces. An n-qutrit quantum system resides in a 3^n-dimensional Hilbert Space. So, in an n-qutrit code, the number of orthogonal subspaces required cannot be more than 3^n. To accommodate all these eleven errors and the error-free state in separate orthogonal subspaces for each of the three logical qutrits in an n-qutrit code, Eq.(6.4) should be satisfied.

$$3(11n + 1) \leq 3^n. \tag{6.4}$$

The minimum value of n for which this inequality is satisfied is five. So five qutrits are necessary for correcting a single error in a qutrit. A similar bound was achieved by Laflamme et al [LMPZ96] for qubits. Therefore, the ternary Shor code is not optimal in the number of qutrits. Moreover, since the correction steps for this proposed QECC are two-fold, the overhead of error correction is significant. The next portion of this chapter thrives to design a QECC such that (i) it requires a fewer number of qutrits, and (ii) can correct errors in a single step.

6.5 Six qutrit degenerate approximate QECC

The aim of designing efficient QECC is to reduce: (i) the number of qutrits for encoding, and (ii) the cost of encoding and decoding circuits. These two requirements are often difficult to achieve simultaneously. Encoding and decoding circuits with higher gate count tend to reduce the computational speedup, and can also incorporate further errors [MBMSK16]. The previous attempt at extending a binary QECC to ternary resulted in a two-step error correction. Therefore, the focus of this section is more on designing QECCs focused on ternary systems.

QECC can be designed by combining two classical error-correcting codes. If $C_1 = [n, k_1, d_1]$ and $C_2 = [n, k_2, d_2]$ are two classical linear codes such that $C_2^{\perp} \subseteq C_1$ and $k_2 < k_1$, then the parity check matrices of the two codes can be combined to form the stabilizers of an $[[n, k_2 - k_1, min\{d_1, d_2\}]]$ QECC [CS96]. Such QECC, called CSS code, readily implies that the set of stabilizers S can be partitioned into two disjoint subsets S_x and S_z, where the operators in $S_x \in \{I, \sigma_x\}^{\otimes n}$ and the operators in $S_z \in \{I, \sigma_z\}^{\otimes n}$. $S_x(S_z)$ is obtained by converting the 0 and 1 of the parity check matrix of one of the classical codes into I and $\sigma_x(\sigma_z)$ respectively. The quantum circuit for CSS QECCs usually has a lower gate count [DMN13, MBSK17], and unlike non-CSS codes, these codes can correct a single bit flip and a single phase flip error simultaneously if they occur on different qubits. Shor [Sho95] and Steane code [Ste96a] are examples of CSS code, but Laflamme's code [LMPZ96] is not. Sarvepalli et al. [SK10] showed that encoding the information of a single qubit into at least 6 qubits is necessary for the existence of a CSS code. Nevertheless, Shaw et al. [SWO+08] proved that a 6-qubit CSS code can exist only when external entanglement is shared between the encoder and the decoder. Their argument readily carries over to ternary systems as well. This raises the question of whether it is possible to achieve approximate error correction with a CSS structure for 6 qutrits in the absence of shared entanglement, and whether such an approximate QECC (AQECC) can have lower gate counts as well.

Consider a QECC where k qutrits of information are encoded into a codeword of $n > k$ qutrits, and the encoded state is $|\psi\rangle$. If $\mathcal{E}$ is the set of all single qutrit errors on $|\psi\rangle$, then the QECC is said to be degenerate if there exists $e, e' \in \mathcal{E}$, $e \neq e'$ such that $e|\psi\rangle = e'|\psi\rangle = |\phi\rangle$. For such scenarios, it is not necessary to distinguish between those errors. Rather, if the error state $|\phi\rangle$ can be identified, then the recovery map can operate either $e^{\dagger}$ or $e'^{\dagger}$ on $|\phi\rangle$ in order to correct the error.

For qubit systems, Shor's 9-qubit code [Sho95] is a degenerate code, but Steane's 7-qubit [Ste96a] and Laflamme's 5-qubit codes [LMPZ96] are not. The 9-qutrit QECC discussed before is also a degenerate code. In the error model of Eq. (6.2), the number of possible bit (phase) errors on a qutrit is 2. Assuming the presence

of at most a single error at any instant, there are at most $2n$ possible bit (phase) error patterns for an n-qutrit QECC. Each stabilizer has three possible outcomes (1, ω, ω^2) for qutrit systems. In order to uniquely identify $2n$ bit (phase) error patterns and the error-free state, the required number of stabilizers is at least $log_3(2n+1)$. Since for an n-qutrit code, the total number of stabilizers is $n-1$ [Got97], the necessary condition for a non-degenerate CSS code is

$$2\lceil log_3(2n+1)\rceil \leq n-1 \tag{6.5}$$

Eq. (6.5) is satisfied for $n \geq 7$. Therefore, a 6-qutrit code, which maintains the CSS structure, must be degenerate.

6.5.1 Proposed encoding scheme for the AQECC

The information of a single qutrit $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle + \gamma|2\rangle$ is encoded into six qutrits as $|\psi\rangle_L = \alpha|0\rangle_L + \beta|1\rangle_L + \gamma|2\rangle_L$, where

$$\begin{aligned}
|0\rangle_L &= |000000\rangle + |010201\rangle + |020102\rangle + |102010\rangle + |112211\rangle + |122112\rangle \\
&+ |201020\rangle + |211221\rangle + |221122\rangle \\
|1\rangle_L &= |111111\rangle + |121012\rangle + |101210\rangle + |210121\rangle + |220022\rangle + |200220\rangle \\
&+ |012101\rangle + |022002\rangle + |002200\rangle \\
|2\rangle_L &= |222222\rangle + |202120\rangle + |212021\rangle + |021202\rangle + |001100\rangle + |011001\rangle \\
&+ |120212\rangle + |100110\rangle + |110011\rangle
\end{aligned}$$

This encoding is degenerate since there are multiple errors that take $|\psi\rangle_L$ to the same error state. For example, Z_1 error on either the second or the sixth qutrit maps $|\psi\rangle_L$ to the same error state.

This encoding scheme satisfies the necessary condition for error correction [KLV00] which states that for any error $\sigma \in \mathcal{E}$

$$\langle 0_L| \sigma |0_L\rangle = \langle 1_L| \sigma |1_L\rangle = \langle 2_L| \sigma |2_L\rangle$$

The sufficient condition for error correction [KLV00] states that for any errors $\sigma_m, \sigma_n \in \mathcal{E}$,

$$\langle i_L| \sigma_m^\dagger \sigma_n |j_L\rangle = \delta_{ij}\alpha_{mn}$$

where $i, j \in \{0, 1, 2\}$, $\alpha_{mn} \in \mathbb{C}$ and δ_{ij} is the Dirac-delta function. Let σ_k^j denote the error σ_k on the j^{th} qubit. It can be checked that the sufficient condition is satisfied for all $\sigma_m, \sigma_n \in \mathcal{E}$ except when $\{\sigma_m, \sigma_n\} = \{X_k^3, X_k^4\}$, $k \in \{1, 2\}$. Thus although the encoding scheme can detect a bit error (and hence Y error) on the 3^{rd} or the 4^{th} qutrit of the encoded state, it fails to distinguish (and hence correct) the error states. It can, however, correct every other error in $\mathcal{E}$ exactly. Due to this single instance where the code behaves like an error-detecting code instead of an error-correcting code, the term *approximate QECC (AQECC)* has been assigned to it.

6.5.2 Proposed stabilizer structure for the AQECC

The stabilizer structure for the 9-qutrit QECC seems insufficient for single-step error correction. Therefore, here the stabilizer structure of [Got98] is extended to

$$X_1 |j\rangle = |j+1\rangle \bmod 3 \quad X_2 |j\rangle = |j+2\rangle \bmod 3$$
$$Z_1 |j\rangle = \omega^j |j\rangle \quad Z_2 |j\rangle = \omega^{2j} |j\rangle$$

where $j \in \{0, 1, 2\}$, and $\omega^3 = 1$. Note that

$$X_2 = X_1 X_1 \quad Z_2 = Z_1 Z_1.$$

The ternary stabilizers, presented below, are n-fold tensor products of $\{I, X_1, X_2, Z_1, Z_2\}$.

Correcting phase errors

The stabilizers for phase error correction are

$$\begin{aligned} S_1 &= I \otimes X_1 \otimes I \otimes X_2 \otimes I \otimes X_1 \\ S_2 &= X_1 \otimes I \otimes X_2 \otimes I \otimes X_1 \otimes I \end{aligned}$$

These two stabilizers partition the qutrits into *probable* phase error subsets due to degeneracy, as shown in Table 6.3.

Table 6.3: Partition of the qutrits into probable phase error subsets

	Error type	S_1	S_2	Probable error qutrits
1	Z_1	1	ω	q_3
2		1	ω^2	q_1, q_5
3		ω	1	q_4
4		ω^2	1	q_2, q_6
5	Z_2	1	ω	q_1, q_5
6		1	ω^2	q_3
7		ω	1	q_2, q_6
8		ω^2	1	q_4

For example, Z_1 error on q_1, q_5, and Z_2 error on q_3 map the codeword to the same error state. Because of this degeneracy, it is neither necessary to uniquely identify the phase error, nor the affected qutrit. Rather, if $S_2 = \omega^2$, one can either correct Z_1 error on q_1 or q_5, or Z_2 error on q_3 to restore the error-free state. A similar argument holds for the other probable error subsets as well. From the stabilizer structure, the qutrits can be partitioned into two subsets $g_1 = \{q_1, q_3, q_5\}$ and $g_2 = \{q_2, q_4, q_6\}$ such that for a single phase error in g_1, only S_2 has eigenvalue $\neq 1$ and for a single phase error in g_2, only S_1 has eigenvalue $\neq 1$. A single phase error can be corrected, as stated before, even without uniquely identifying the error type or the erroneous qutrit.

Lemma 6.1

The proposed 6-qutrit AQECC can correct two phase errors simultaneously if these occur on two qutrits such that one of them belongs to g_1 and the other belongs to g_2.

Proof. Since the stabilizers S_1 and S_2 operate on a disjoint set of qutrits, a single phase error cannot result in both the stabilizers having eigenvalues $\neq 1$. Therefore, if both the stabilizers show outcome $\neq 1$, then obviously there is a single phase error in a qutrit of g_1 as well as g_2. Since, for this code, it is not necessary to distinguish among qutrits of the same subset for reliable correction, if both the eigenvalues are $\neq 1$, the phase of a single qutrit from each subset can be corrected according to Table 6.3 in order to obtain the error-free state. Therefore, two-phase errors, when occurring on the two qutrits from two disjoint subsets g_1 and g_2, can be reliably corrected by the proposed AQECC. □

Correcting bit errors

The stabilizers for bit error correction are

$$\begin{aligned} S_3 &= Z_1 \otimes Z_2 \otimes Z_1 \otimes Z_2 \otimes I \otimes I \\ S_4 &= I \otimes I \otimes Z_1 \otimes Z_2 \otimes Z_1 \otimes Z_2 \\ S_5 &= Z_1 \otimes Z_1 \otimes I \otimes I \otimes Z_2 \otimes Z_2 \end{aligned}$$

Let X_i^j denote the error X_i occurring on the j^{th} qutrit. In Table 6.4 we explicitly show the action of the stabilizers on the error states.

As stated earlier, the two pairs of rows (3, 4) and (9, 10) in Table 6.4 indicate that a bit error on the third or fourth qutrit can be detected, but neither the affected qutrit nor the type of error can be uniquely identified. All the other bit errors can be uniquely identified and hence exactly corrected.

Table 6.4: Correcting a single bit error with the proposed AQECC

	Error Type	S_3	S_4	S_5
1	X_1^1	ω	1	ω
2	X_1^2	ω^2	1	ω
3	X_1^3	ω	ω	1
4	X_1^4	ω^2	ω^2	1
5	X_1^5	1	ω	ω^2
6	X_1^6	1	ω^2	ω^2
7	X_2^1	ω^2	1	ω^2
8	X_2^2	ω	1	ω^2
9	X_2^3	ω^2	ω^2	1
10	X_2^4	ω	ω	1
11	X_2^5	1	ω^2	ω
12	X_2^6	1	ω	ω

6.5.3 Performance Analysis

The proposed AQECC fails if the error is of type X or Y and it occurs on the third or fourth qutrit. In fact for the $Y = ZX$ type of error, the proposed code can correct the Z part, and the state is left with uncorrected X error. Let p be the probability of a single error, then the probability that the proposed AQECC fails is

$$\begin{aligned} & Prob\ (uncorrected\ single\ error) \\ = \ & Prob\ (error\ on\ q_3\ or\ q_4 \mid error\ type\ X\ or\ Y) \cdot \\ & Prob\ (error\ type\ X\ or\ Y) \end{aligned} \tag{6.6}$$

For a symmetric error model, as in Eq. 6.2,

$$Prob\ (error\ is\ of\ type\ X\ or\ Y) \ = \ 6p/8$$

Since each qutrit is equally likely to be erroneous,

$$Prob\ (error\ on\ q_3\ or\ q_4 \mid error\ type\ X\ or\ Y) = 1/3$$

Therefore, for a symmetric error model, where each type of error occurs with probability $p/8$, the probability that the proposed AQECC fails to correct a single error is $1/3 \cdot 6p/8 = p/4$. In other words, our proposed AQECC is able to correct a single error exactly with probability 0.75 for a symmetric error model.

In general in quantum systems error is asymmetric. Let p_x, p_z, and p_y be the probability of a single bit error, phase error, and Y error respectively, then typically [IM07], $p_y = p_x = 0.01p_z$. Therefore,

$$Prob\ (error\ is\ of\ type\ X\ or\ Y) \ = \ 6p_x/8$$

Using the relation $p_y = p_x = 0.01p_z$ in Eq. (6.2), we note that

$$\begin{aligned} 2p_z + 2p_x + 4p_y &= p \\ \Rightarrow p_x &= p/208 \end{aligned}$$

Therefore,

$$Prob\ (error\ is\ of\ type\ X\ or\ Y) \ = \ 6p/1648$$

Each qutrit is equally likely to be erroneous in the asymmetric error model as well. Therefore, the probability that the proposed AQECC fails to correct a single error is $1/3 \cdot 6p/1648 = p/824$. In other words, for an asymmetric error model, the proposed AQECC, on average, is able to correct a single error exactly with probability $823/824$ ($= 0.9988$).

6.5.4 Error correction circuit for the proposed AQECC

Instead of the C_1 and C_2 gates used for the 9-qutrit QECC, a more generalized form is considered here called the $C+T$ gate, whose action is defined as

$$C+T: \sum_{x,y\in\{0,1,2\}} |x,(x+y)\%3\rangle \langle x,y| .$$

Since $Z_2 = Z_1 Z_1$, each Z_2 operator is realized using a cascade of two $C+T$ gates. For the realization of each $C+T$ gate, it is required to cascade two gates for the two cases: (i) the first one changes the target only if the control is 2, and (ii) the second one changes the target only if the control is 1. Case (i) itself is one of the MS gates [MS00], while the gate of the case (ii) requires 3 MS gates (Fig. 6.2). Therefore, 4 MS gates are necessary to implement a single $C+T$ gate.

The circuit to correct a single bit error, shown in Fig. 6.4, follows from the stabilizers S_3, S_4 and S_5. In the circuit, $|q_0\rangle$ to $|q_5\rangle$ are the six data qutrits, whereas $|a_0\rangle$ to $|a_2\rangle$ are ancilla qutrits necessary for syndrome detection without measuring the data qutrits directly.

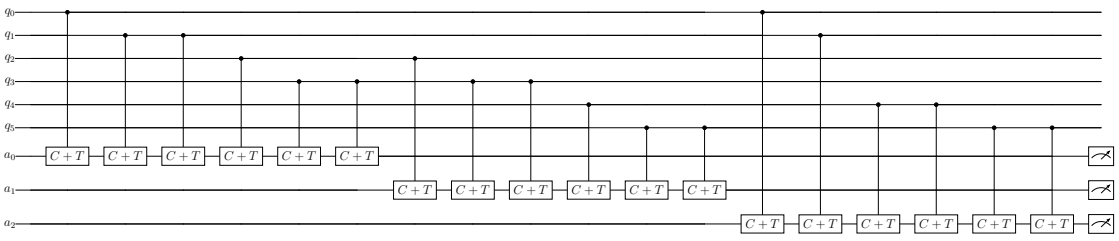


Figure 6.4: Circuit to correct a single bit error with the 6-qutrit AQECC

For qubit systems, phase errors behave like bit errors on the Hadamard basis. The natural extension of the Hadamard basis in ternary quantum systems is the Chrestenson basis [HMM85]. Two conjugate Chrestenson bases b_1 and b_2 are defined below.

Chrestenson basis b_i, $i \in \{1, 2\}$:

$$\begin{aligned} |+_i\rangle &= \frac{1}{\sqrt{3}}(|0\rangle + |1\rangle + |2\rangle) \\ |-_i\rangle &= \frac{1}{\sqrt{3}}(|0\rangle + \omega^i |1\rangle + \omega^{2i} |2\rangle) \\ ||_i\rangle &= \frac{1}{\sqrt{3}}(|0\rangle + \omega^{2i} |1\rangle + \omega^i |2\rangle) \end{aligned}$$

Conversion from the usual computational basis to the bases b_1 and b_2 can be done using the following Chrestenson gates Ch_1 and Ch_2 respectively [HMM85]:

$$Ch_1 = \frac{1}{\sqrt{3}} \begin{pmatrix} 1 & 1 & 1 \\ 1 & \omega & \omega^2 \\ 1 & \omega^2 & \omega \end{pmatrix} \quad Ch_2 = \frac{1}{\sqrt{3}} \begin{pmatrix} 1 & 1 & 1 \\ 1 & \omega^2 & \omega \\ 1 & \omega & \omega^2 \end{pmatrix}$$

The circuit to correct (multiple) phase errors is shown in Fig. 6.5. The circuits for correcting bit error and phase error require three and two ancilla qutrits respectively. Therefore, a total of 5 ancilla qutrits are required.

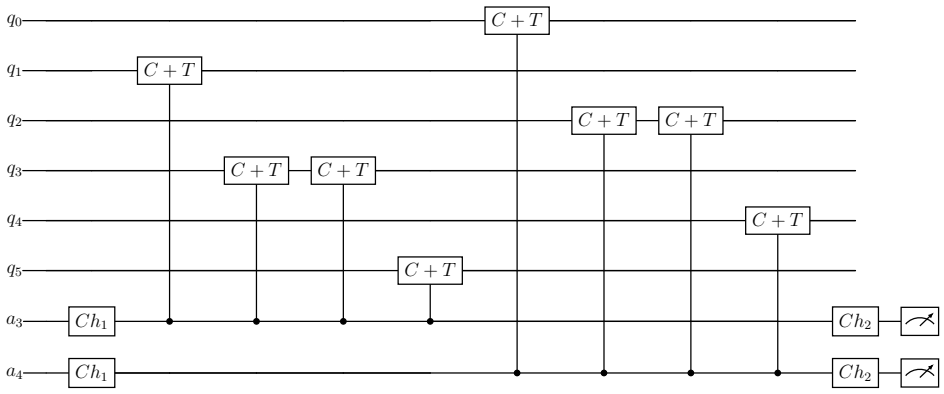


Figure 6.5: Circuit to correct phase error with the 6-qutrit AQECC

6.5.5 Comparison of quantum cost

This subsection shows a comparative analysis of the quantum cost for the error correction circuits of the 9-qutrit QECC and the 6-qutrit AQECC. The quantum cost is analyzed in terms of MS gates and Chrestenson gates.

As discussed earlier, each $C+T$ gate can be implemented using 4 MS gates. From Fig. 6.4, the circuit to correct a single bit error requires 18 $C+T$ gates and thus has a quantum cost of $18 \times 4 = 72$. The circuit for phase error correction requires 8 $C+T$ gates (i.e. 32 MS gates), along with 4 Chrestenson gates. Therefore, the total quantum cost of the circuit of the 6-qutrit AQECC is $72 + 32 + 4 = 108$.

In the circuit of the 9-qutrit QECC, each stabilizer to correct a bit error comprises three Z_1 operators. Therefore the quantum cost for each stabilizer is $3 \times 4 = 12$. The total quantum cost for three such stabilizers is, therefore, 36. However, in addition to that, it requires a second step for detecting the location of the error. The quantum cost of the gates for that step is 16, which makes the total quantum cost of their circuit for correcting bit error to be $36 + 16 = 52$.

However, in order to correct phase errors, the individual subspaces ({0,1},{1,2} and {2,0}) are corrected. Phase errors behave like bit errors when each subspace is converted to the Hadamard basis, and hence must be corrected similarly to bit error correction. Therefore, the quantum cost to correct a single subspace is the same as that of the bit error correction circuit. Furthermore, in order to correct a single subspace, the following procedure is necessary: apply Hadamard gates to each qutrit, apply the bit error correction circuit, and restore the qutrits to the computational basis by applying the Hadamard gates again. Therefore the quantum cost for correcting a single subspace is $52 + 18 = 70$. This procedure is repeated thrice in order to correct all three subspaces. Therefore, the total quantum cost of the circuit in [MBGSK18] for correcting a single phase error is 210. Table 6.5 reports the comparison of the quantum cost of the circuit for 9-qutrit exact QECC in [MBGSK18] and our proposed 6-qutrit AQECC.

Table 6.5: Comparison of the Quantum Cost of the circuits of the 9-qutrit QECC and the 6-qutrit AQECC

	Circuit for bit error	Circuit for phase error	Total
9-qutrit QECC	52	210	262
Proposed AQECC	72	36	106
% reduction			59.5

6.6 Summary

This chapter provides the first effort to design a ternary QECC as a carry-over of its binary counterpart. It shows that the cost of the QECC circuit increases necessarily since the error correction needs to be performed in two steps. A follow-up effort to lower the qubits and gate counts was shown by the formulation of a 6-qutrit AQECC. However, this AQECC fails to detect the location of errors in certain cases. The next chapter dives into the fundamental question of the necessary condition required to design a ternary QECC as a carry-over of its binary counterpart and shows the design of a 9-qutrit QECC which has significantly less gate count and can correct errors in a single step.

CHAPTER 7

Designing Ternary Quantum Error Correcting Codes from Binary Codes

Contents

7.1 Introduction

Quantum systems are inherently multi-valued. It was expected that QECC for binary quantum systems could readily be carried over to higher dimensional systems [Cha97]. A generalized higher dimensional Pauli group was proposed by Gottesman [Got98] for designing stabilizers for higher dimensional QECCs consisting of the operators X_1 and Z_1, where

$$X_1 |j\rangle = |j+1\rangle \bmod d; \qquad Z_1 |j\rangle = \omega^j |j\rangle$$

where $j \in \{0, 1, \ldots, d-1\}$, X and Z commonly denote bit and phase errors respectively. For a ternary system, $d = 3$, and ω is the cube root of unity. However, Chapter 6 showed otherwise. The ternary Shor code, whose stabilizer formulation was based on the generalized higher dimensional Pauli group, was necessarily different than the binary Shor code. This resulted in error correction with multiple steps, leading to higher gate costs. This made error correction largely impractical in ternary QECC, since such a high gate count and depth of the QECC circuit would make the computation slower, and may incorporate significant errors in the system. This implied that binary QECCs extended to higher dimensions would necessarily have higher implementation complexity (i.e., increased gate count and depth of the QECC).

Chapter 6 also presented a 6-qutrit approximate QECC (AQECC). Although this formulation failed to correct errors in every possible instance, it provided an extension of the higher dimensional Pauli group proposed in [Got98]. This chapter revisits the ternary Shor code with the set of stabilizer components used in the 6-qutrit AQECC. This chapter also shows that the extended higher dimensional Pauli group spans the 3×3 operator space. Therefore, any 3×3 unitary error can be corrected by a QECC which can correct the errors in this extended higher-dimensional Pauli group. Apart from X_1 and Z_1, this group also comprises

$X_2 = X_1.X_1$ and $Z_2 = Z_1.Z_1$. Although these are not independent stabilizer terms, these are shown to be necessary if a ternary QECC is to be designed as an extension of a binary QECC. Furthermore, it is shown that only three of the four operators $\{X_1, X_2, Z_1, Z_2\}$ are sufficient to derive a ternary stabilizer structure similar to Shor code for the encoding of the 9-qutrit QECC proposed in Chapter 6. The circuit design of the 9-qutrit QECC, with stabilizer derived from the extended Pauli group, using the gate set proposed by Muthukrishnan and Stroud [MS00] achieves 51.9% and 23.07% reductions respectively in the quantum cost and depth of the error correction circuit from the ternary Shor code in Chapter 6.

It is shown that for ternary Steane and Laflamme codes, the extended higher dimensional Pauli group is necessary for stabilizer formulation in order to retain the structure of their binary counterpart. This study paves a path for easy extension of existing binary QECCs to ternary.

7.2 A Spanning Basis for Ternary Quantum Operators

This section proposes a set of operators that span the space of all 3×3 unitary operators. A QECC which can correct these errors, can also correct any unitary error on the system. In continuation of the terminology used in Chapter 6, errors on ternary quantum systems are classified as *binary* or *ternary* where

1. A binary error acts non-trivially on a subspace of the 3-dimensional Hilbert Space;

2. A ternary error acts non-trivially on the entire 3-dimensional Hilbert Space.

Consider the set of matrices $\sigma_i, i = 1, 2, ..., 9$, where ω is the cube-root of unity. Note that each σ_i is a binary error corresponding to either a bit error or a product

of bit and phase errors. For example, σ_1 is a binary bit error occurring on the subspace of $\{|1\rangle, |2\rangle\}$; and σ_2 has a phase error in addition to σ_1. It adds a phase of ω and ω^2 on $|1\rangle$ and $|2\rangle$ respectively. The other matrices can be interpreted similarly.

$$\sigma_1 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix} \quad \sigma_2 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & \omega^2 \\ 0 & \omega & 0 \end{pmatrix} \quad \sigma_3 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & \omega \\ 0 & \omega^2 & 0 \end{pmatrix}$$
$$\sigma_4 = \begin{pmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{pmatrix} \quad \sigma_5 = \begin{pmatrix} 0 & 0 & \omega^2 \\ 0 & 1 & 0 \\ \omega & 0 & 0 \end{pmatrix} \quad \sigma_6 = \begin{pmatrix} 0 & 0 & \omega \\ 0 & 1 & 0 \\ \omega^2 & 0 & 0 \end{pmatrix}$$
$$\sigma_7 = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad \sigma_8 = \begin{pmatrix} 0 & \omega^2 & 0 \\ \omega & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad \sigma_9 = \begin{pmatrix} 0 & \omega & 0 \\ \omega^2 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Before presenting the new single step 9-qutrit QECC, it is required to show that $\sigma_i, 1 \leq i \leq 9$ form a basis for 3×3 unitary operators. In particular, Lemma 7.2 shows that the σ_is are linearly independent, and Lemma 7.2 shows that they span the 3×3 operator space.

Lemma 7.1

The σ_is, $1 \leq i \leq 9$ are linearly independent.

Proof. Let us assume that there exists $\Lambda_i, 1 \leq i \leq 9$ such that $\sum_{i=1}^{9} \Lambda_i \sigma_i = 0$, and $\Lambda_i \neq 0$ for all i. Then:

$$\begin{aligned} \Lambda_1 + \Lambda_2 + \Lambda_3 &= 0 & \Lambda_1 + \omega^2\Lambda_2 + \omega\Lambda_3 &= 0 \\ \Lambda_1 + \omega\Lambda_2 + \omega^2\Lambda_3 &= 0 & \Lambda_4 + \Lambda_5 + \Lambda_6 &= 0 \\ \Lambda_4 + \omega^2\Lambda_5 + \omega\Lambda_6 &= 0 & \Lambda_4 + \omega\Lambda_5 + \omega^2\Lambda_6 &= 0 \\ \Lambda_7 + \Lambda_8 + \Lambda_9 &= 0 & \Lambda_7 + \omega^2\Lambda_8 + \omega\Lambda_9 &= 0 \\ \Lambda_7 + \omega\Lambda_8 + \omega^2\Lambda_9 &= 0 \end{aligned}$$

Note that these nine equations can be grouped into three sets, each set containing three equations. No two sets of equations involve the same coefficients. The first three, the second three, and the last three equations form such sets. The proof is shown for the set of first three equations involving coefficients $\Lambda_1, \Lambda_2, \Lambda_3$. The proof for the other two sets is similar.

If the set of matrices σ_i, $1 \leq i \leq 9$ are not linearly independent, at least two of the three coefficients $\Lambda_1, \Lambda_2, \Lambda_3$ must be non-zero. If only one of them is non-zero, then the first equation is not satisfied. Without loss of generality, let us assume that $\Lambda_1 = -(\Lambda_2 + \Lambda_3) \neq 0$. Substituting for Λ_1 in the second and third equations yields

$$\begin{aligned}(\omega^2 - 1)\Lambda_2 &= (1-\omega)\Lambda_3 \Rightarrow \frac{\Lambda_2}{\Lambda_3} = \frac{1-\omega}{\omega^2-1} \\ (\omega - 1)\Lambda_2 &= (1-\omega^2)\Lambda_3 \Rightarrow \frac{\Lambda_2}{\Lambda_3} = \frac{1-\omega^2}{\omega-1}\end{aligned}$$

Equating the ratios of Λ_2 and Λ_3 gives $\omega = \omega^2$, which is also not possible. Therefore, in order to satisfy the first set of three equations, each of the coefficients must be zero.

Extending a similar argument to the other two sets yields the contradiction that if $\sum_{i=1}^{9} \Lambda_i \sigma_i = 0$, then $\Lambda_i = 0$ for all i. □

Lemma 7.2

For any 3×3 matrix M, there exists $\lambda_i, 1 \leq i \leq 9$ such that $\sum_{i=1}^{9} \lambda_i \sigma_i = M$.

Proof. For ternary systems, any 3×3 unitary operator is a probable error. Consider a 3×3 matrix

$$M = \begin{pmatrix} M_{11} & M_{12} & M_{13} \\ M_{21} & M_{22} & M_{23} \\ M_{31} & M_{32} & M_{33} \end{pmatrix} \tag{7.1}$$

where $M_{ij} \in \mathbb{C}$, $1 \leq i, j \leq 3$. It is to be noted that a matrix M may not necessarily be unitary for all values of M_{ij} and hence may not represent a quantum error. Nevertheless, this generalized 3×3 matrix is considered for now.

Let us consider that $M = \sum_{i=1}^{9} \lambda_i \sigma_i$ and check for the existence of suitable λ_is. This yields the following nine equations:

$$\begin{aligned}
\lambda_1 + \lambda_2 + \lambda_3 &= M_{11} & \lambda_1 + \omega^2\lambda_2 + \omega\lambda_3 &= M_{23} \\
\lambda_1 + \omega\lambda_2 + \omega^2\lambda_3 &= M_{32} & \lambda_4 + \lambda_5 + \lambda_6 &= M_{22} \\
\lambda_4 + \omega^2\lambda_5 + \omega\lambda_6 &= M_{13} & \lambda_4 + \omega\lambda_5 + \omega^2\lambda_6 &= M_{31} \\
\lambda_7 + \lambda_8 + \lambda_9 &= M_{33} & \lambda_7 + \omega^2\lambda_8 + \omega\lambda_9 &= M_{12} \\
\lambda_7 + \omega\lambda_8 + \omega^2\lambda_9 &= M_{21}
\end{aligned}$$

Note that each set of three equations has a similar structure. Therefore, proof with the first three equations is presented, and that for the other two sets of three equations are similar.

The first three equations can be represented as

$$\begin{pmatrix} 1 & 1 & 1 \\ 1 & \omega^2 & \omega \\ 1 & \omega & \omega^2 \end{pmatrix} \begin{pmatrix} \lambda_1 \\ \lambda_2 \\ \lambda_3 \end{pmatrix} = \begin{pmatrix} M_{11} \\ M_{23} \\ M_{32} \end{pmatrix}$$

Therefore,

$$\begin{aligned}
\begin{pmatrix} \lambda_1 \\ \lambda_2 \\ \lambda_3 \end{pmatrix} &= \begin{pmatrix} 1 & 1 & 1 \\ 1 & \omega^2 & \omega \\ 1 & \omega & \omega^2 \end{pmatrix}^{-1} \begin{pmatrix} M_{11} \\ M_{23} \\ M_{32} \end{pmatrix} \\
&= \Omega^{-1} \begin{pmatrix} M_{11} \\ M_{23} \\ M_{32} \end{pmatrix}
\end{aligned}$$

Since the determinant of the matrix Ω is non-zero, it is invertible. Therefore it

is always possible to find $\lambda_1, \lambda_2, \lambda_3$ which satisfy the set of three equations. Since each set of three equations has a disjoint set of coefficients, similar arguments hold for the other two sets also. Therefore, for any such matrix M, it is always possible to find linearly independent parameters $\lambda_i, 1 \leq i \leq 9$ such that $\sum_{i=1}^{9} \lambda_i \sigma_i = M$. □

Now the theorem states

Theorem 7.1

A QECC that can correct the matrices $\sigma_i, 1 \leq i \leq 9$ can correct any error on a qutrit.

Proof. If $\{M\}$ is the set of all 3×3 matrices, and $\{E\}$ is the set of all possible quantum errors such that every $\mathcal{E} \in \{E\}$ is a unitary matrix, then $\{E\} \subset \{M\}$. Therefore, any $\mathcal{E} \in \{E\}$ can also be written as a linear combination of $\sigma_i, 1 \leq i \leq 9$ by Lemma 7.2. If a QECC can correct each of the σ_is, it can also correct any error $\mathcal{E}$ on the quantum system. □

Lemma 7.2 considered a spanning set of *binary errors*. However, *ternary errors* are more suitable to deal with in a ternary QECC. Four ternary errors, corresponding to binary bit and phase errors, are presented whose actions on a general qutrit $|\psi\rangle = \alpha |0\rangle + \beta |1\rangle + \gamma |2\rangle$ are shown below:

$$\begin{aligned} Z_i |\psi\rangle &= \alpha |0\rangle + \omega^i \beta |1\rangle + \omega^{2i} \gamma |2\rangle ; \\ X_i |\psi\rangle &= \alpha |0 \oplus i\rangle + \beta |1 \oplus i\rangle + \gamma |2 \oplus i\rangle \end{aligned}$$

where $i \in \{1, 2\}$ and $\oplus$ represents addition modulo 3. In accordance with [MBGSK18], the errors X_1 and X_2 are named as *bit shift errors*, and the errors Z_1 and Z_2 as *phase errors*. The matrix representation of these four ternary errors is shown below:

$$X_1 = \begin{pmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix} \quad X_2 = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix}$$

$$Z_1 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \omega & 0 \\ 0 & 0 & \omega^2 \end{pmatrix} \quad Z_2 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \omega^2 & 0 \\ 0 & 0 & \omega \end{pmatrix}$$

All the $\sigma_i, 1 \leq i \leq 9$ matrices can be written as a linear combination of these matrices and their products. Therefore, any 3×3 quantum error $\mathcal{E}$ can be written as a linear combination of X_1, X_2, Z_1 and Z_2 and their products. The formulation (upto a scalar coefficient) of σ_1 and σ_2 using X_1, X_2, Z_1 and Z_2 is shown explicitly. The other matrices can also be formulated similarly.

$$\begin{aligned} \sigma_1 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix} &= \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} + \begin{pmatrix} 1 & 0 & 0 \\ 0 & \omega & 0 \\ 0 & 0 & \omega^2 \end{pmatrix} + \begin{pmatrix} 1 & 0 & 0 \\ 0 & \omega^2 & 0 \\ 0 & 0 & \omega \end{pmatrix} \\ &+ \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix} + \begin{pmatrix} 0 & \omega & 0 \\ 0 & 0 & 1 \\ \omega^2 & 0 & 0 \end{pmatrix} + \begin{pmatrix} 0 & \omega^2 & 0 \\ 0 & 0 & 1 \\ \omega & 0 & 0 \end{pmatrix} \\ &+ \begin{pmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix} + \begin{pmatrix} 0 & 0 & \omega^2 \\ \omega & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix} + \begin{pmatrix} 0 & 0 & \omega \\ \omega^2 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix} \\ &= I + Z_1 + Z_2 + X_2 + \omega Z_2 X_2 + \omega^2 Z_1 X_2 \\ &\quad + X_1 + \omega^2 Z_2 X_1 + \omega Z_1 X_1 \end{aligned}$$

$$\begin{aligned}\sigma_2 = \begin{pmatrix}1 & 0 & 0\\ 0 & 0 & \omega^2\\ 0 & \omega & 0\end{pmatrix} &= \begin{pmatrix}1 & 0 & 0\\ 0 & \omega^2 & 0\\ 0 & 0 & \omega\end{pmatrix}\begin{pmatrix}1 & 0 & 0\\ 0 & 0 & 1\\ 0 & 1 & 0\end{pmatrix}\\ &= I + Z_1 + Z_2 + Z_2X_2 + \omega Z_1X_2 + \omega^2 X_2\\ &\quad + Z_2X_1 + \omega^2 Z_1X_1 + \omega X_1\end{aligned}$$

Hence, any 3×3 unitary error on a qutrit can be expressed as a linear combination of X_1, X_2, Z_1, and Z_2 or the product of two or more of them. Further, a general 1-qutrit unitary error can be represented as in Eq. (7.2).

$$\mathcal{E} = \delta\mathbb{I}_3 + \sum_{i=1}^{2}\eta_i Z_i + \sum_{j=1}^{2}\mu_j X_j + \sum_{i,j}\xi_{ij}Y_{ij} \tag{7.2}$$

where $\mathbb{I}_3$ is the 3×3 identity matrix, X_j and Z_i are the bit shift and phase errors respectively, $Y_{ij} \propto Z_iX_j$; and $\delta, \eta, \mu, \xi \in \mathbb{C}$.

7.3 Stabilizers for 9-qutrit QECC

It was shown in Sec. 7.2 that any 3×3 unitary operator, which is a potential error, can be represented as a linear combination of the products of X_1, X_2 and Z_1, Z_2. Therefore, a QECC that corrects for each of these four errors sequentially can correct any 3×3 unitary error.

The operators X and Z do not commute. One can verify that

$$\begin{aligned}Z_iX_i &= \omega X_iZ_i \text{ for } i\in\{1,2\};\\ Z_iX_j &= \omega^2 X_jZ_i \text{ for } i,j\in\{1,2\}\ i\neq j\end{aligned} \tag{7.3}$$

This implies that if two n-qutrit stabilizers, $S_k \in \{X_p, I\}^{\otimes n}$ and $S_l \in \{Z_q, I\}^{\otimes n}$, where $p, q \in \{1, 2\}$, then

1. S_k and S_l commute if and only if the number of locations where both the stabilizers have non-identity operators is $3h$, $h \in \mathbb{Z}^+ \cup \{0\}$;
2. If S_k (or S_l) has non-identity operators in more than one location, then it alone cannot distinguish between errors in those locations.

In other words, if two such stabilizers S_k and S_l are to commute, then they must have non-identity operators at three mutual locations. For example, consider the stabilizers $S_k = Z_1 \otimes Z_1 \otimes I$ and $S_l = X_1 \otimes X_1 \otimes X_1$. These two are the stabilizers operating on the first three qubits of the binary Shor Code. In binary system, $[X \otimes X, Z \otimes Z]$, and therefore these two commute. But now, in ternary, these two do not commute, and can no longer be considered valid stabilizers.

Since in a binary setting $[X \otimes X, Z \otimes Z] = 0$, the stabilizers of the binary QECC are designed so that the locations of X and Z in different stabilizers overlap in an even number of positions. Therefore, it may be possible to directly carry over a binary QECC to the ternary regime using only X_1 and Z_1 as stabilizer components if the locations overlap in $6h$ locations, for $h \geq 0$. However, for binary QECCs such as the Shor [Sho95], Steane [Ste96a] and Laflamme codes [LMPZ96], the X and Z operators in the stabilizers do not overlap in $6h$ locations.

Consider another simple example for illustration: it was already shown that the stabilizers, extended directly from binary Shor code, no longer commute. In order for these stabilizers to commute, the X and Z operators must overlap in three locations.

Suppose an example three qutrit codeword (not a complete QECC) $|\psi\rangle$ is stabilized by two operators $S_k = X_1 X_1 X_1$ and $S_l = Z_1 Z_1 Z_1$. First note that it is necessary to have three non-identity operators in both S_k and S_l to ensure that they commute. Therefore, it has already deviated from the stabilizer structure of binary Shor

Code [Sho95]. Furthermore, if, say, an X_1 error occurs on any one of the three qutrits, resulting in an erroneous state $|\psi\rangle_e$, the eigenvalue of $S_l |\psi_e\rangle$ remains the same irrespective of the location of the error. Therefore, this stabilizer can only detect the presence of error. Some follow-up steps will be necessary to determine the location of the error.

7.3.1 Retrieving the binary 9-qubit QECC stabilizer structure

The logical states from the encoding scheme used for the 9-qutrit code in [MBGSK18], similar to that of the Shor code, is

$$|i\rangle_L = \frac{1}{3\sqrt{3}}(|000\rangle + \omega^i |111\rangle + \omega^{2i} |222\rangle)^{\otimes 3}$$

for $i \in \{0, 1, 2\}$. The set of stabilizers that was used is

$$Z_1Z_1Z_1IIIIII,\ IIIZ_1Z_1Z_1III,\ IIIIIIZ_1Z_1Z_1$$
$$X_1X_1X_1IIIIII,\ IIIX_1X_1X_1III,\ IIIIIIX_1X_1X_1.$$

It is to be noted that this immediately deviates from the stabilizer structure of the Shor code. On the other hand, this structure is also necessary to ensure that the stabilizers commute. Further, as illustrated before, using $Z_1Z_1Z_1IIIIII$ as a stabilizer restricts its ability to distinguish between bit errors on the first three positions, which led to a second step of correction in [MBGSK18]. Therefore if the stabilizer components are restricted to X_1 and Z_1 (or X_2 and Z_2) only, then the structure of the Shor code cannot be retained in the ternary settings.

In order to extend QECCs from binary to ternary regime, it is useful to retain the stabilizer structure that completely characterizes a QECC. For fault-tolerance, a qudit is encoded once, while detection and correction of errors may be performed

multiple times along the computation. The same stabilizer structure implies that the error correction circuit remains similar. In order to retain the stabilizer structure, more than one X and/or Z type component can be used in the stabilizer formulation. Usage of all four of them is sufficient (as used in [MSK20]), but not necessary, as we show below.

Table 7.1 shows the set of ternary stabilizers, constructed using $\{Z_1, Z_2\}$ and X_1, that has the exact stabilizer structure of the Shor code. In Table 7.1, the empty locations indicate the identity operator and an operator in column q_i indicates that the operator is applied on qutrit q_i. For example, the entire stabilizer S_1 is $Z_1 \otimes Z_2 \otimes I \otimes I \otimes I \otimes I \otimes I \otimes I \otimes I$, which implies the operators Z_1 and Z_2 operate on qutrits $q1$ and $q2$ respectively, and the rest of the qutrits have identity acting on them. The stabilizer structure of Table 7.1 shows a technique for deriving a ternary QECC from its binary counterpart.

Table 7.1: Ternary stabilizer for 9-qutrit QECC retaining the structure of Shor code [Sho95]

Stabilizers ↓ \ Qutrits →	q1	q2	q3	q4	q5	q6	q7	q8	q9
S_1	Z_1	Z_2							
S_2		Z_2	Z_1						
S_3				Z_1	Z_2				
S_4					Z_2	Z_1			
S_5							Z_1	Z_2	
S_6								Z_2	Z_1
S_7	X_1	X_1	X_1	X_1	X_1	X_1			
S_8				X_1	X_1	X_1	X_1	X_1	X_1

Table 7.2 shows the actions of the stabilizers S_1, S_2 for bit error correction when an error occurs on one of the first three qutrits. The block structure of this encoding implies that the same follows trivially for the other two blocks of three qutrits as well. The codeword is denoted as $|\psi\rangle$, and X_i^j (Z_i^j) implies the error X_i (Z_i) on qutrit j, $i \in \{1, 2\}$, $j \in \{1, 2, 3\}$. Similarly, Table 7.3 shows the action of the stabilizers when a phase error occurs on qutrits $q1, q4$, and $q7$. The first qutrit from each block of three qutrits is selected deliberately. Since this is a degenerate code, the action of phase error on any qutrit in the same block is equivalent.

Table 7.2: Correction of bit errors

Errors ↓ \ Stabilizers →	S_1	S_2
$\|\psi\rangle$	$+1$	$+1$
$X_1^1 \|\psi\rangle$	ω	$+1$
$X_1^2 \|\psi\rangle$	ω^2	ω
$X_1^3 \|\psi\rangle$	$+1$	ω^2
$X_2^1 \|\psi\rangle$	ω^2	$+1$
$X_2^2 \|\psi\rangle$	ω	ω^2
$X_2^3 \|\psi\rangle$	$+1$	ω

Table 7.3: Correction of phase errors

Errors ↓ \ Stabilizers →	S_7	S_8
$\|\psi\rangle$	$+1$	$+1$
$Z_1^1 \|\psi\rangle$	ω^2	$+1$
$Z_1^4 \|\psi\rangle$	ω^2	ω^2
$Z_1^7 \|\psi\rangle$	$+1$	ω^2
$Z_2^1 \|\psi\rangle$	ω	$+1$
$Z_2^4 \|\psi\rangle$	ω	ω
$Z_2^7 \|\psi\rangle$	$+1$	ω

Note that changing the positions of Z_1 and Z_2 in the stabilizers keeps the code, and its quantum cost (discussed in next section) unchanged. For the error correction in Table 7.2, this change simply swaps between the ω and ω^2 eigenvalues. Similarly, one can use X_2 as the stabilizer component as well, without effectively changing the code. But since $X_2 = X_1X_1$, the quantum cost of the circuit will increase.

There may be a different 9-qutrit QECC similar to Shor code that uses $\{X_1, X_2\}$ and Z_1/Z_2 as the stabilizer components. However, using X_1 and X_2 together can largely change the structure of the encoded qutrit — that structure is not pursued here.

The ability to retain the stabilizer structure of binary QECC comes with some restrictions on the definition of a logical operator on the encoded state, which is presented in the next subsection.

7.3.2 Restrictions on logical Pauli Operators

An operator on the encoded quantum state is said to be a logical operator if it changes the encoded state. For example, an operator X_L is said to be a *logical* X operator if $|0\rangle_L \xrightarrow{X_L} |1\rangle_L \xrightarrow{X_L} |2\rangle_L$. Two criteria that a logical operator must satisfy are as follows:

1. the logical operator should commute with all the stabilizers;
2. the weight of a logical operator, i.e., the number of non-identity operators, must be greater than or equal to the distance of the code.

The first requirement is trivial, since if the logical operator does not commute with all the stabilizers, then the operator is treated as an error. On the other hand, the distance of the code is determined by the minimum weight of an operator that commutes with all the stabilizers [Got97]. If a weight w operator commutes with all the stabilizers, then it is not treated as an error, and therefore the distance of the QECC must be less than w.

It is *posited* that, as proposed by Gottesman [Got98], the logical Pauli X and Z operators contain only $X_1 |j\rangle = |j+1\rangle \mod d; \quad Z_1 |j\rangle = \omega^j |j\rangle$. It may be tempting to consider any combination of X_1, X_2 or Z_1, Z_2 as logical operators. However,

(i) Applying $X_2 = X_1.X_1$ and $Z_2 = Z_1.Z_1$ as components of the logical operator would imply that two operations are performed sequentially on a qutrit (see Sec. 7.4) as a part of a logical operator.

(ii) This cannot retain the stabilizer structure of the Shor code in a ternary system.

While the first claim is trivial, a deeper dive is required into the second one. Since Shor code has the tensor product structure of three qutrits, the argument can be

restricted to the first three qutrits only; it follows trivially to the other two sets as well. Suppose any combination of X_1, X_2 or Z_1, Z_2 is accepted as a logical operator as long as they commute with all the stabilizers, and are themselves not stabilizers. Since this code can correct a single error on the qutrit, the distance of the code must be at least 3. In other words, there should not exist any operator e, which is not a stabilizer, that commutes with all the stabilizers but has a weight less than 3. Note that the operator $Z_1IZ_2IIIIII$ with a weight of 2, is not a stabilizer but commutes with all the stabilizers. This would imply that it is a logical operator. On the other hand, one can verify that there is no such operator of weight less than 3 consisting only of $\{I, X_1\}$ or $\{I, Z_1\}$ that has this property.

Next, it may appear that using all four of $\{X_1, X_2, Z_1, Z_2\}$ to construct the stabilizers can serve the purpose. Table 7.4 shows an example to certify that it is indeed not the case. Even in the stabilizer structure of Table 7.4, $Z_1Z_1IIIIII$ is an operator with the weight of 2 that satisfies the requirements of a logical operator. It can be verified that any permutation of the components of these stabilizers carries the same drawback.

Table 7.4: An attempt to a different stabilizer structure for ternary Shor code

Qutrits → / Stabilizers ↓	q1	q2	q3
S_1	Z_1	Z_2	
S_2		Z_1	Z_2
S_7	X_1	X_2	X_1

Therefore, this implies that in order to design a QECC for a ternary quantum system similar to the binary Shor code, one of the following conditions must be satisfied:

(i) the stabilizer structure needs to be changed if only X_1 and Z_1 are used as components for stabilizers;

(ii) the components of the logical operator must be restricted to X_1 and Z_1 only.

The primary intent of this study is to retain the stabilizer structure, since that

allows easy extension of binary QECCs to the ternary regime, and conforms to this restriction on logical operators.

7.4 Circuit Realization of the 9 qutrit QECC

While any 3×3 unitary matrix is a valid quantum gate, it may not be implementable in quantum hardware. Therefore, usually a set of *basis gates* is defined which can be implemented in practice and used to realize any unitary matrix. In [MS00], Muthukrishnan and Stroud proposed a set of single and two-qutrit gates, called the MS gates, that can be implemented on an Ion-Trap device. The ternary counterpart of the Hadamard gate, called the Chrestenson gate [HMM85], is also known to be implementable. The MS gates and the Chrestenson gate form a universal basis set for ternary systems and can be used to realize any ternary quantum gate [MS00].

To the best of our knowledge, there is no notion of the quantum cost of gates for ternary quantum systems. For this study, the quantum cost of each MS gate and Chrestenson gate is assumed to be 1. The quantum cost of all the circuits henceforth is provided in terms of the number of MS and Chrestenson gates required to implement them.

The multi-step correction procedure required in [MBGSK18] led to a very high gate cost of the QECC circuit. Next, the circuit design of the 9-qutrit code is revisited using the updated stabilizer structure proposed above. A lone Z_1 operator is equivalent to a $C+T$ gate, having outer product notation as

$$C+T : \sum_{x,y\in\{0,1,2\}} |x,(x+y)\%3\rangle \langle x,y| .$$

Since $Z_2 = Z_1 Z_1$, it is equivalent to a cascade of two $C+T$ gates. The bit error correction circuit corresponding to each stabilizer is shown in Fig. 7.1.

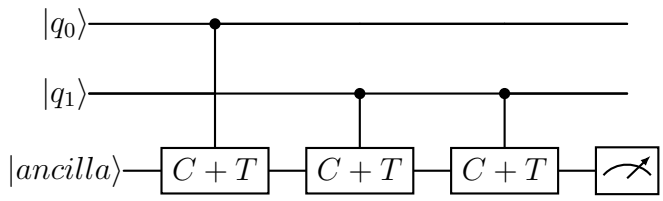


Figure 7.1: Circuit corresponding to each stabilizer of the form $Z_1 \otimes Z_2$ for bit error correction

For the six stabilizers $S_1, \ldots, S_6$, that correct bit errors on the codeword, the total gate count in terms of $C+T$ gates is thus 18. It was shown by Majumdar et al. in [MSK20] that a single $C+T$ gate can be decomposed into 3 MS gates. Therefore, the quantum cost of the circuit corresponding to bit error correction is $18 \times 3 = 54$.

Phase error correction is usually performed in a basis that is a 45° rotation from the computational basis. Conversion from the usual computational basis to this rotated basis can be done using the following Chrestenson gates Ch_1 and Ch_2 respectively [HMM85]:

$$Ch_1 = \frac{1}{\sqrt{3}} \begin{pmatrix} 1 & 1 & 1 \\ 1 & \omega & \omega^2 \\ 1 & \omega^2 & \omega \end{pmatrix} \quad Ch_2 = \frac{1}{\sqrt{3}} \begin{pmatrix} 1 & 1 & 1 \\ 1 & \omega^2 & \omega \\ 1 & \omega & \omega^2 \end{pmatrix}$$

It can be verified that $Ch_1 Ch_2 = I$, and

$$Ch_1 X_1 Ch_2 = Z_1 \qquad Ch_1 X_2 Ch_2 = Z_2.$$

Therefore, the circuit for phase error correction can be implemented using $C+T$ gates as necessary, with each of the 9 qutrits padded with a Ch_1 and Ch_2 on the two ends (see Fig. 7.2). Each phase error correction stabilizer thus requires six $C+T$ gates and two Chrestenson gates. Therefore, the quantum cost for phase error correction is $2 \times (6 \times 3 + 2 \times 9) = 72$. The total quantum cost of the error correction circuit, thus, becomes 126. The quantum cost of the previous 9-qutrit QECC circuit was 262 (see Table 7.5), which gives this implementation a 51.9%

reduction in gate cost.

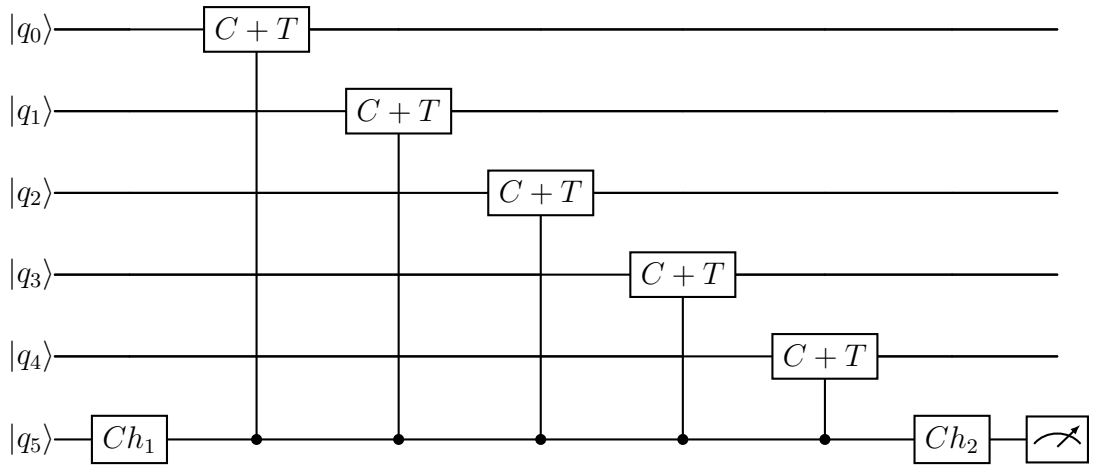


Figure 7.2: Circuit corresponding to each stabilizer of the form $\otimes_{i=0}^{5} X_i$ for phase error correction

Another parameter necessary to assess a circuit design is the depth of the circuit. The depth of a quantum circuit is defined as the maximum number of gates on any input-to-output path. In the 9-qutrit QECC structure proposed here, the maximum number of stabilizer operators is 6 for each of S_6 and S_7. Since each X_1 operator can be realized using 3 MS gates, and considering the two Chrestenson gates padded for the basis change, the depth of the error correction circuit is $6 \times 3 + 2 = 20$. This obtains a percentage savings of 23.07 in terms of depth.

Table 7.5 compares the quantum cost and the depth of the circuits of the previous 9-qutrit QECC [MBGSK18], the 6-qutrit AQECC [MSK20], and the 9-qutrit QECC proposed in this article. Although the cost of the proposed circuit is higher than that for the 6-qutrit AQECC, this QECC can correct all single errors on the system, unlike the 6-qutrit AQECC.

Table 7.5: Comparison of the Quantum Cost and depth of circuit of the 6-qutrit AQECC and the 9-qutrit QECC

	Circuit for bit error	Circuit for phase error	Total	Depth
9-qutrit QECC [MBGSK18]	52	210	262	26
6-qutrit AQECC [MSK20]	72	44	116	8
Proposed 9-qutrit QECC	54	72	126	20
% Savings w.r.t [MBGSK18]			51.9	23.07

7.5 Ternary Steane and Laflamme codes

The previous sections showed the limitations of the earlier approach to carry over binary QECCs to ternary and proposed a method to overcome that limitation. The stabilizer structure of binary Shor Code for its ternary counterpart was derived as well. It can be easily verified that the limitations mentioned are retained in a direct carryover from the 7-qubit Steane Code [Ste96a] and the 5-qubit Laflamme Code [LMPZ96] to their corresponding ones in the ternary regimen as well, if only X_1 and Z_1 are used as stabilizer operators.

7.5.1 Binary to ternary Steane code

The stabilizer structure of binary Steane code is

$$\begin{aligned} S_1 &= I \otimes I \otimes I \otimes X \otimes X \otimes X \otimes X \\ S_2 &= I \otimes X \otimes X \otimes I \otimes I \otimes X \otimes X \\ S_3 &= X_1 \otimes I \otimes X \otimes I \otimes X \otimes I \otimes X \\ S_4 &= I \otimes I \otimes I \otimes Z \otimes Z \otimes Z \otimes Z \\ S_5 &= I \otimes Z \otimes Z \otimes I \otimes I \otimes Z \otimes Z \\ S_6 &= Z_1 \otimes I \otimes Z \otimes I \otimes Z \otimes I \otimes Z \end{aligned} \tag{7.4}$$

Recall from Sec. 7.3 that for ternary systems, $[X^{\otimes 3}, Z^{\otimes 3}] = 0$. Therefore, it is sufficient to use only $X_1(X_2)$ and $Z_1(Z_2)$ operators for ternary stabilizers if the X and Z operators overlap in $6h$ positions ($h \geq 1$). In the stabilizers of binary Steane Code, the X and Z operators overlap in 4 locations. Therefore, direct extension of this QECC to the ternary regime using only $X_1(X_2)$ and $Z_1(Z_2)$ operators will violate the commutation requirement of the stabilizers.

However, by using X_1, Z_1, and Z_2, the stabilizer structure of binary Steane QECC can be retained in the ternary regime, as shown below:

$$\begin{aligned} S_1 &= I \otimes I \otimes I \otimes X_1 \otimes X_1 \otimes X_1 \otimes X_1 \\ S_2 &= I \otimes X_1 \otimes X_1 \otimes I \otimes I \otimes X_1 \otimes X_1 \\ S_3 &= X_1 \otimes I \otimes X_1 \otimes I \otimes X_1 \otimes I \otimes X_1 \\ S_4 &= I \otimes I \otimes I \otimes Z_1 \otimes Z_2 \otimes Z_2 \otimes Z_1 \\ S_5 &= I \otimes Z_1 \otimes Z_2 \otimes I \otimes I \otimes Z_2 \otimes Z_1 \\ S_6 &= Z_1 \otimes I \otimes Z_2 \otimes I \otimes Z_2 \otimes I \otimes Z_1 \end{aligned} \tag{7.5}$$

7.5.2 Binary to ternary Laflamme code

Similarly, the stabilizer structure of binary Laflamme code is

$$\begin{aligned} S_1 &= I \otimes X \otimes Z \otimes Z \otimes X \\ S_2 &= X \otimes I \otimes X \otimes Z \otimes Z \\ S_3 &= Z_2 \otimes X \otimes I \otimes X \otimes Z \\ S_4 &= Z \otimes Z \otimes X \otimes I \otimes X \end{aligned} \tag{7.6}$$

Unlike Shor and Steane code, Laflamme code is a non-CSS code, i.e., X and Z operators are present in the same stabilizer. The X and Z operators overlap in 2 positions, and hence cannot be extended to a ternary regime by using only $X_1(X_2)$ and $Z_1(Z_2)$ operators. On the other hand, using X_1, Z_1 and Z_2, the stabilizer

structure of binary Laflamme QECC can be retained in the ternary regime, as shown below:

$$\begin{aligned} S_1 &= I \otimes X_1 \otimes Z_1 \otimes Z_2 \otimes X_1 \\ S_2 &= X_1 \otimes I \otimes X_1 \otimes Z_1 \otimes Z_2 \\ S_3 &= Z_2 \otimes X_1 \otimes I \otimes X_1 \otimes Z_1 \\ S_4 &= Z_1 \otimes Z_2 \otimes X_1 \otimes I \otimes X_1 \end{aligned} \tag{7.7}$$

The examples of Steane and Laflamme's code further solidify that using $\{Z_1, Z_2\}$ and X_1 is sufficient to derive ternary stabilizer QECCs similar to its corresponding binary one. Otherwise, the structure of ternary QECCs cannot be derived directly from its binary counterpart.

7.6 Summary

This chapter explored a fundamental question of what is required to extend a binary QECC to the ternary regime without changing its stabilizer structure. The chapter illustrates that using X_1 and Z_1 alone as components of the stabilizers is not sufficient to design a ternary QECC which is a direct extension of the corresponding binary QECC. Instead, it is necessary to use Z_1, Z_2 and X_1 for this purpose. The reasons for the shortcoming of the 9-qutrit QECC in Chapter 6 is discussed in detail, and the 9-qutrit QECC, maintaining the stabilizer structure of binary Shor code, was then derived using the operators Z_1, Z_2 and X_1. This proposed QECC attains a percentage reduction of 51.9 and 23.07 in quantum cost and depth respectively over the previous one. It is also shown that this requirement is not special for Shor code only. Rather the same requirement remains for any ternary QECC designed as a carry-over of its binary counterpart — design of ternary Steane and Laflamme codes were shown as explicit examples. This result

opens up a myriad of research prospects. It may be worth designing a ternary QECC using X_1, X_2 and Z_1/Z_2, and comparing its quantum cost and depth with the design pursued in this article. For any d-dimensional quantum system, there are $Z_1, \ldots, Z_{d-1}$ possible phase errors. The stabilizer design of any d-dimensional QECC, as a low-cost implementable extension of binary QECC, remains to be explored.

CHAPTER 8

Intermediate Qutrit-assisted Toffoli Decomposition with Quantum Error Correction

Contents

8.1 Introduction

Current quantum devices are engineered to execute a finite number of one or two-qubit gates, termed as the basis gate set [ibm22]. Any arbitrary quantum operator $\mathcal{O}$ is equivalent to a cascade of gates taken from this basis gate set [NC02]. This method is often termed as the decomposition of the operator $\mathcal{O}$. A Toffoli gate is a gate with 3 qubits and finds applications in many important aspects of quantum computing such as quantum error correction [NC02], Grover's algorithm [Gro96b], etc. Several works have focused on the efficient decomposition of the Toffoli gate due to its significance in quantum computing [Sel13, AMMR13, Jon13]. A trade-off between depth and qubit-count of Toffoli decomposition has been observed in the literature [AMMR13]: the depth of the decomposed circuit can be reduced by allowing ancilla qubits whereas the depth increases when the qubit-count is kept fixed to the original three qubits.

In [GBD+20], the authors proposed temporary usage of $|2\rangle$, a higher dimension state within a qubit system, and showed an exponential reduction in the depth of the decomposition circuit for a Toffoli gate without any ancilla qubits. This has triggered studies on potential applications of temporary usage of $|2\rangle$ within a qubit system. It has been shown to improve the implementation of arithmetic circuits [SCCC22, SCC23], and even eliminate the need for SWAP gates in a limited connectivity quantum hardware [SSC22]. This method was experimentally verified on a superconducting quantum device in [GCK+21], and generalized to $d \geq 2$ dimensional quantum circuit for multi-controlled Toffoli gates in [SMS+22].

Accessing higher dimensions and applying higher dimensional gates, nevertheless, invoke more errors in the system than qubits. On the other hand, the exponential reduction in the depth of the circuit lowers the effect of damping error. Numerical studies [GBD+20, SMS+22, SCCC22, SCC23] show that the overall error on the system is reduced by this method. In other words, the exponential reduction in depth overshadows the temporary usage of higher dimensions. This method of lowering the depth of the circuit via intermediate qutrits is proposed primarily for

the Noisy Intermediate Scale Quantum (NISQ) era [Pre18b], where the number of qubits is up to a few hundred, and the qubits are noisy due to the absence of error correction. In general, a reduction in the depth of the circuit is expected to be useful both with and without error correction. However, in this method, reduction in depth comes at the cost of higher noise arising from higher dimensional qudits and gates which may incur more cost in terms of error correction. In this paper, we focus on the study of the resources required, especially the number of gates when this decomposition is applied together with error correction and concatenation. The primary question that we address is whether this method, involving more error due to usage of the higher dimension, can lead to lower resources than the earlier decomposition methods, which have a higher number of gates in the decomposition when error correction and concatenation are involved.

Adopting the qutrit-assisted decomposition technique for error correction is not without challenges. For concatenated quantum circuits, the qubits must be encoded at the beginning of the computation, and every operation (e.g. gate operation, syndrome measurement, error correction) must be performed on the encoded qubits. We show here that an encoded qubit cannot be made to access dimension $|2\rangle$ temporarily without entertaining the possibility of errors that remain undetected. In other words, a qubit accessing the higher dimension at any point in time must be treated as a qutrit throughout the computation. This immediately deviates from the fact that in the NISQ era, the qubit is raised to the higher dimension only temporarily. In other words, the error-corrected circuit now becomes a hybrid qubit-qutrit circuit. As qutrits are in general noisier than qubits [FMT+22], the number of levels of concatenation is expected to be higher to reach the desired accuracy, which in turn increases the resource requirement of the circuit.

This chapter looks into the resource requirement, in terms of gate count, when error correction and concatenation are associated for both the qubit-only decomposition and qutrit-assisted decomposition.

8.2 Decomposition of gates using higher dimension

In [GBD+19, BDG+20], the authors proposed temporary occupation of the state $|2\rangle$ for decomposition of Toffoli gates. Maintaining binary input and output allows this circuit construction to be inserted into any pre-existing qubit-only circuit. A Toffoli decomposition via qutrits has been portrayed in Fig. 8.1 [GBD+19, BDG+20]. More specifically, the goal is to carry out a NOT operation on the target qubit (the third qubit $|q_2\rangle$) as long as the two control qubits are both $|1\rangle$. First, a $|1\rangle$-controlled X_{+1}, where $+1$ denotes that the target qubit is incremented by 1(mod 3), is performed on $|q_0\rangle$ and $|q_1\rangle$, the first and the second qubits. This upgrades $|q_1\rangle$ to $|2\rangle$ if and only if both $|q_0\rangle$ and $|q_1\rangle$ are $|1\rangle$. Then, a $|2\rangle$-controlled X gate is applied to the target qubit $|q_2\rangle$. Therefore, X is executed only when both $|q_0\rangle$ and $|q_1\rangle$ are initially $|1\rangle$. These control qubits are reinstated to their original states by a $|1\rangle$-controlled X_{-1} gate, which reverses the effect of the first gate. That the $|2\rangle$ state from ternary quantum systems can be used instead of an ancilla to store temporary information, is the most important aspect of this decomposition. Thus, to decompose the Toffoli gate, 3 generalized ternary $CNOT$ gates are sufficient with circuit depth 3. In fact, no T gate is required.

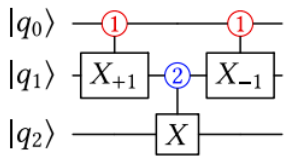


Figure 8.1: An example of Toffoli decomposition with an intermediate qutrit, where input and output are qubits. The red controls activate on $|1\rangle$ and the blue controls activate on $|2\rangle$. The first gate temporarily elevates q_1 to $|2\rangle$ if both q_0 and q_1 were $|1\rangle$. X operation is then only performed if q_1 is $|2\rangle$. The final gate acts as a mirror of the first gate and restores q_0 and q_1 to their original states. [GBD+19].

8.3 Criterion for qutrit-assisted Toffoli decomposition along with error correction

Consider a quantum circuit qc that has been encoded using a QECC $\mathcal{C}$. This circuit can be decomposed into basis gates using either the qubit or qubit-qutrit decomposition method as discussed above. Let p_2 and $p_{2,3}$ be the respective probability of error in these two cases. A few of the qubits behave as qutrits at certain cycles of execution in the qubit-qutrit decomposition [GBD+19]. Literature suggests that these qubits are treated as qutrits only for a limited period of time when it requires access to the state $|2\rangle$. For example, in Fig. 8.1, $|q_1\rangle$ behaves as a qutrit only during the execution of the two qutrit gates on $|q_1\rangle$ and $|q_2\rangle$, and as a qubit otherwise. However, Theorem 8.1 shows that when error-corrected qubit-qutrit decomposition is considered, a qubit that requires access to the state $|2\rangle$ at any point in the circuit, must be treated as a qutrit all along.

Theorem 8.1

Given an error-corrected qubit-qutrit decomposition, a quantum state which requires access to the state $|2\rangle$ at any point in the circuit, must be encoded using a qutrit QECC.

Proof. Let $\mathcal{C}$ be the codespace for a QECC, where each basis state $|i\rangle$ is mapped to the logical state $|i\rangle_L = \sum_{c_i} |c_i\rangle$, where each $|c_i\rangle \in \mathcal{C}_i \subseteq \mathcal{C}$. Now, for a general qubit QECC, $c_i \in \{0,1\}^{\otimes n}\ \forall\ i$. For some time period $[t, t+\Delta t]$, this state is raised to access the basis state $|2\rangle$ for the purpose of computation. This can be modelled as an operator X_1 acting on the logical state, where $X_1 = \begin{pmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}$ [MSK20, MSK22]. For an n-qubit QECC, $|c_i\rangle = |c_{i_1} c_{i_2} \dots c_{i_n}\rangle$, since initially $c_i \in \{0,1\}^{\otimes n}\ \forall\ i$, applying X_1 takes c_i to $c_i \in \{1,2\}^{\otimes n}$. Now, during the time period $[t, t+\Delta t]$, a bit error occurring on the physical qubit (or qutrit) c_{i_k} which was in the state $|2\rangle$ makes it $|0\rangle$. As a result, now the erroneous logical state $c_i \in \{0,1,2\}^{\otimes n}$.

In other words, the system, which should have been spanned by the $\{1, 2\}$ subspace only, is now spanned by $\{0, 1, 2\}$ subspace. Therefore, when the system is lowered back to the original computational space by applying an operator $X_2 = X_1^{-1}$, it no longer returns to the $\{0, 1\}$ subspace but remains in the $\{0, 1, 2\}$ subspace. Such an error behaves like a leakage, and cannot be corrected by general qubit QECCs. □

It is obviously possible that some errors, even acting during the $[t, t + \Delta t]$ time period, do not leak the system out of its computational space. However, Theorem 8.1 argues that there exists errors which can do so, and hence cannot be corrected using a qubit QECC. An example can be given using the simplest QECC – a 3-qubit repetition code. Consider a qubit $|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle$, which requires access to the computational state $|2\rangle$ during time period $[t, t + \Delta t]$, has been encoded as $|\psi\rangle_L = \alpha\,|000\rangle + \beta\,|111\rangle$ for error correction. When this qubit requires access to state $|2\rangle$, the logical state $|\psi\rangle_L$ is raised to $\alpha\,|111\rangle + \beta\,|222\rangle$ using the X_1 operator. If a bit error occurs on, say, the first qubit while it is accessing the state $|2\rangle$, the erroneous state becomes $\alpha\,|211\rangle + \beta\,|022\rangle$. Restoring this system back to a qubit configuration by applying X_2 on this erroneous state takes the state to $\alpha\,|100\rangle + \beta\,|211\rangle$, which has undergone leakage from the computational space. Hence, error correction using binary QECC is no longer possible.

Therefore, if a qubit is allowed to access the state $|2\rangle$ at any point in the circuit, it must be treated as a qutrit from the input stage and encoded accordingly for error correction. Theorem 8.1 does not pose any restrictions on the qubits that do not require access to the state $|2\rangle$ to be encoded using binary QECC for error correction. For example, in Fig. 8.1, $|q_0\rangle$ and $|q_2\rangle$ may be encoded using binary QECC, but $|q_1\rangle$ must be encoded using a ternary QECC.

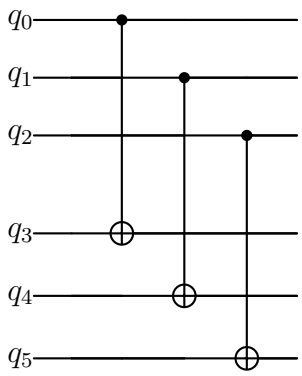
A general notion is that qutrits and ternary quantum gates are noisier than qubits and binary quantum gates respectively [FMT+22]. Therefore, it may be possible to attain an accuracy of ϵ using k_2 and k_3 levels of concatenation for qubits and qutrits respectively, where $k_2 \leq k_3$. A natural inference, therefore, is to use fewer levels of concatenation for qubits than for qutrits in the concatenated circuit to reduce the qubit cost. In other words, the circuit complexity can be reduced if

one can use $k_2 < k_3$ levels of concatenation for qubits and qutrits respectively, to acquire the same accuracy for both qubits and qutrits. However, Theorem 8.2 below asserts otherwise.

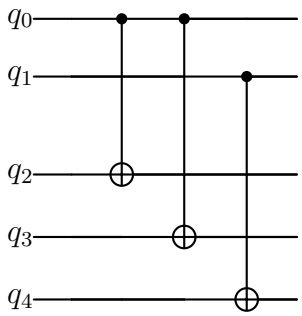
Theorem 8.2

For the implementation of two-qubit logical gates via interaction between the participating qubits (or qutrits), both the qubits and qutrits in the hybrid qubit-qutrit circuit must be encoded with the same number of levels of concatenation for fault-tolerant implementation irrespective of their respective probability of error.

Before proceeding into the proof of this theorem, it is necessary to revisit a first principle requirement of fault-tolerance [Sho96] – error from one component should flow to at most a single component. For example, in Fig. 8.2 (a), an error on one qubit can flow to at most another qubit via the CNOT gate. However, in Fig. 8.2 (b), error on qubit q_0 will flow to both qubits q_2 and q_3. Such a scenario violates the requirement of fault tolerance.



(a) Fault-tolerant representation of encoded CNOT gate



(b) Non fault-tolerant representation of encoded CNOT gate

Figure 8.2: Two realizations of an encoded CNOT gate – (a) can be made fault-tolerant via concatenation since error from a qubit can flow to one qubit only, while (b) cannot be made fault-tolerant via concatenation since error from q_0 will flow to both q_2 and q_3

Proof. On the contrary, assume that the qubits and qutrits can be concatenated

using k_2 and k_3 levels of concatenation, where $k_2 \neq k_3$. WLOG, $k_2 < k_3$. Let G be a transversal two-qubit gate for the QECC used, operating over q_2 and q_3, where q_2 (q_3) is a qubit (qutrit). However, since $k_2 < k_3$, the number of qubits encoding q_2 is less than the number of qutrits encoding q_3. Therefore, by pigeonhole principle, there exists at least one qubit on which two encoded gates, involving two distinct qutrits, operate. This violates the requirement of fault tolerance. □

There exist other methods to implement logical two-body gates such as topological effects in a surface code [FMMC12]. This method assumes the existence of multiple logical qubits in a single surface code lattice, and the CNOT operation is performed by applying a bunch of operators extending from one logical qubit around the other. Nevertheless, having both qubits and qutrits in a single surface code lattice would require a significant change in the structure of the lattice (where some stabilizers will be binary and others ternary). It is an engineering challenge to implement it if it is at all possible. It is left out as a future scope to study such qubit-qutrit surface code design and whether this theorem still holds good under such a scenario. For now, it can be considered that a single surface code lattice encodes a single logical qubit (or qutrit), and two-body gates are performed by the interaction between these two lattices [MBMSK16], where this theorem still holds good.

Theorem 8.2 asserts that the level of concatenation, and hence the size of the circuit, is governed by the probability of error on qutrits. Thus the error-corrected representation of the qubit-qutrit decomposition (i) has a lower depth of the circuit due to its more efficient decomposition, and (ii) leads to circuits with higher resource requirements since it is governed by the probability of errors on qutrits. The pertinent issue is to determine the criteria for which the increase in the size of the circuit is overshadowed by the number of gates reduced due to more efficient qubit-qutrit decomposition. Theorems 8.3 and 8.4 address the following two parameters: (i) the accuracy of the two types of decomposition if both use the same number of levels of concatenation and (ii) the increase in the number of levels of concatenation required for the qubit-qutrit decomposition to obtain the same accuracy as that of the qubit only based decomposition.

Theorem 8.3

Given a quantum circuit C, let C_2 and C_3 be the two decompositions for C involving qubit gates only and qubit-qutrit gates, with error probabilities p_2 and $p_{2,3}$ respectively. After k levels of concatenation in both, the accuracy $\epsilon_3 = \delta \cdot \epsilon_2$ obtained by $C_{2,3}$, where ϵ_2 is the accuracy obtained by C_2 and $\delta > 0$, is given by

$$log(\delta) = 2^k log(\frac{c_3.p_{2,3}}{c_2.p_2}) + log(\frac{c_2}{c_3})$$

where $\frac{1}{c_2}$ and $\frac{1}{c_3}$ are the thresholds of the binary and the ternary QECCs respectively.

For example, Fig. 8.3 and 8.1 show examples of decomposition of a Toffoli gate using only qubit gates, and both qubit and qutrit gates respectively.

In all the proofs in this chapter henceforth, the logarithm is with respect to base 2. Note that any other base is equally acceptable for the logarithm for all the calculations. However, base 2 makes some calculations easier, and the final form of the equations simpler.

Proof. Follows from the Threshold Theorem [NC02]; see Appendix A.11. □

Theorem 8.4

Given a quantum circuit C, let C_2 and $C_{2,3}$ be the two decompositions for C involving qubit gates only and qubit-qutrit gates, with error probabilities p_2 and $p_{2,3}$ respectively. Then, C_2 as well as $C_{2,3}$ achieves an accuracy of ϵ after k_2 and k_3 levels of concatenation respectively, where

$$k_3 = \lceil k_2 + log(\frac{log(c_2.p_2) - \frac{1}{2^{k_2}} log(\frac{c_2}{c_3})}{log(\delta) + log(c_3.p_2)}) \rceil$$

and $\frac{1}{c_2}$, $\frac{1}{c_3}$ are the thresholds of the binary and ternary QECCs respectively.

Before presenting the proof of this theorem, it should be emphasized here that fault-tolerance is attainable only if $c.p < 1$, where p is the probability of error and $\frac{1}{c}$ is the threshold of the QECC used. Here, both $c.p_2$ and $c.p_{2,3} = \delta.c.p_2$ are required to be less than 1 for effective fault tolerance via concatenation. Moreover, the value of $log(\frac{log(c_2.p_2)-\frac{1}{2^{k_2}}log(\frac{c_2}{c_1})}{log(\delta)+log(c_1.p_2)})$ can be a fraction. Therefore, ceiling is used to ensure that k_3 is an integer.

Proof. Follows from the Threshold Theorem [NC02]; see Appendix A.12. □

Currently, surface code is known to have a threshold of $1\% = 0.01$ [FSG09]. For IBM Quantum devices, $CNOT$ is one of the most prominent sources of error in the quantum systems today, having an error probability > 0.01. Therefore, it is not possible to lower the error probability by increasing the levels of concatenation in current quantum systems. This prohibits an experimental comparison of concatenation on the qubit-qutrit decomposition and the qubit-only decomposition. Therefore, for the rest of the paper, any numerical values will assume a futuristic scenario, where p is sufficiently low enough so that $c.p < 1$.

For the sake of better visualization, Table 8.1 assumes $c.p_{2,3} = 0.9$ and 0.5, and vary δ to determine the difference in the level of concatenations k_3 and k_2.

Table 8.1: Difference in levels of concatenation $\lceil k_3 - k_2 \rceil$ with varying δ

$c.p_{2,3}$	δ	$c.p_2$	$\lceil k_3 - k_2 \rceil$
0.9	1.5	0.6	3
	2	0.45	3
	3	0.3	4
	4	0.225	4
	5	0.18	5
0.5	1.5	0.33	1
	2	0.25	1
	3	0.167	2
	4	0.125	2
	5	0.1	2

Since the value of $p_{2,3} \geq p_2$, it is expected that $\lceil k_3 - k_2 \rceil \geq 1$. Increasing levels of

concatenation increases the size of the resulting circuit exponentially. Therefore, the question we ask is whether the resource requirement can be lowered using the qutrit-assisted decomposition in contrast to the qubit-only decomposition in a concatenated error correction scenario.

8.4 Resource estimation of fault-tolerant circuits

Assume that the error-correcting code used requires at most G gates to encode a single gate at each level of concatenation. Note that the exact value of G depends on whether the implementation of a logical gate is transversal [EK09]. This issue is addressed later on. Therefore, the size of a circuit consisting of R gates after k levels of concatenation are upper bounded by $G^k \cdot R$. Let the number of gates in qubit-qutrit and qubit-only decomposition of a circuit be $R_{2,3}$ and R_2 respectively. Then, the resource of qubit-qutrit decomposition is lower if

$$\begin{aligned} G^{k_3} \cdot R_{2,3} &\leq G^{k_2} \cdot R_2 \\ \Rightarrow R_{2,3} &\leq \frac{R_2}{G^{\lceil k_3 - k_2 \rceil}} \end{aligned} \tag{8.1}$$

Eq. (8.1) provides the criteria for which the qubit-qutrit decomposition can result in a smaller circuit even though it requires more levels of concatenation.

Consider a circuit where $\mathcal{G}$ denotes the set of individual gates used in the circuit. Let n_g denote the number of gate type $g \in \mathcal{G}$ in the circuit. Then, after k levels of concatenation, the total number of gates N_k in the concatenated circuit is given in Eq. (8.2), where κ_g denotes the number of gates required for the implementation of the gate logical gate g_L.

$$N_k = \Big(\sum_{g \in \mathcal{G}} \kappa_g n_g\Big)^k \tag{8.2}$$

The exact value of κ_g depends on whether the gate g can be implemented transversally in the QECC used.

Theorem 8.5

Let $\mathcal{G}_2$ and $\mathcal{G}_{2,3}$ be the set of types of gates in a circuit, realized respectively by using the qubit-only and the qubit-qutrit decomposition and n_g be the number of gates of type g. Then the qubit-qutrit decomposition leads to a smaller number of gates if

$$log(\frac{log(c_2.p_2) - \frac{1}{2^{k_2}} log(\frac{c_2}{c_3})}{log(\delta) + log(c_3.p_2)}) \leq k_2 \cdot \frac{log \frac{\sum_{g \in \mathcal{G}_2} (\kappa_g n_g)}{\sum_{g \in \mathcal{G}_{2,3}} (\kappa_g n_g)}}{log \sum_{g \in \mathcal{G}_{2,3}} (\kappa_g n_g)} \tag{8.3}$$

where $\frac{1}{c_2}$ and $\frac{1}{c_3}$ are the thresholds of the binary and ternary QECCs respectively.

Proof. See Appendix A.13. □

Theorem 8.5 provides a relation involving p_2 the probability of error, $\frac{1}{c}$ the threshold of the QECC used, δ the ratio of error probabilities for qutrits and qubits, k_2 the levels of the concatenation of qubit-only system, and the gate counts of both types of decompositions. The value of κ_g for each gate g depends on whether it can be implemented transversally or not. For gates that can be implemented transversally, κ_g is equal to the distance of the QECC. For other cases, the value of κ_g can increase significantly [MBSK17] because the implementation may even be probabilistic [GBL+23].

8.5 Challenges for achieving fault-tolerance

This chapter so far has derived the necessary conditions for using the qutrit-assisted decomposition of Toffoli gate in conjunction with quantum error correction. It also discussed the resource requirement for concatenation code and provided the criterion for which this method uses fewer resources compared to the qubit-only decomposition. However, in order to achieve *fault-tolerance*, it is necessary to implement

the gates in encoded form. This implementation dictates the error flow between encoded components. An encoded gate that cannot be implemented transversally, may lead to a significant increase in error among the encoded components and hence may pose a challenge in achieving fault-tolerance.

The qutrit-assisted Toffoli decomposition primarily consists of two types of CNOT gates (refer to Fig. 8.3), namely 1-controlled and 2-controlled CNOT gate – while the former is transversal in the Steane Code, the latter is not.

8.5.1 Implementing encoded gates for Steane Code

Let us assume that Steane Code [Ste96b] has been used for encoding the qubits. As discussed earlier, qubits that require access to higher dimensions at any point in the computation must be treated as a qutrit throughout the computation. Therefore, 7-qutrit ternary Steane Code is required for encoding some of the quantum states, and the stabilizers are [MSK23]:

$$\begin{aligned} S_1 &= I \otimes I \otimes I \otimes X_1 \otimes X_1 \otimes X_1 \otimes X_1 \\ S_2 &= I \otimes X_1 \otimes X_1 \otimes I \otimes I \otimes X_1 \otimes X_1 \\ S_3 &= X_1 \otimes I \otimes X_1 \otimes I \otimes X_1 \otimes I \otimes X_1 \\ S_4 &= I \otimes I \otimes I \otimes Z_1 \otimes Z_2 \otimes Z_2 \otimes Z_1 \\ S_5 &= I \otimes Z_1 \otimes Z_2 \otimes I \otimes I \otimes Z_2 \otimes Z_1 \\ S_6 &= Z_1 \otimes I \otimes Z_2 \otimes I \otimes Z_2 \otimes I \otimes Z_1 \end{aligned} \tag{8.4}$$

where $X_1 |j\rangle = |j+1\rangle \mod 3$, $Z_1 |j\rangle = \omega^j |j\rangle$ $Z_2 = Z_1 Z_1$. The stabilizers of binary Steane code are similar, with only X and Z operators, where the addition is modulo 2.

For a qubit-qutrit setting with binary and ternary Steane code, it was verified via an exhaustive search that the 1-controlled ternary CNOT gate can be implemented transversally, i.e., $\text{CNOT}_L(|i\rangle_L |j\rangle_L)$, $i \in \{0,1\}$, $j \in \{0,1,2\}$ can be implemented

by executing CNOT gate individually on the 7 qubits used to encode $|i\rangle_L$ and $|j\rangle_L$. Note that in this case, the control is always a qubit, and the target is a qutrit. However, the 2-controlled CNOT gate does not seem to adhere to any transversal implementation for the Steane Code.

With this additional observation, the resource estimation for an adder circuit, in terms of gate count, is studied in the next subsection for both qubit-only and qubit-qutrit decompositions.

8.6 Comparison of resource requirements for decomposition of an adder circuit

This section studies the resource estimation for qubit-only and qubit-qutrit decomposition of an adder circuit. The gain in resource for qubit-qutrit decomposition is estimated using the inequality of Theorem 8.5 for different levels of concatenation when the cost of non-transversal implementation of the 2-controlled CNOT gate is taken to be a multiplicative constant of the transversal implementation of the 1-controlled CNOT gate. Note that, since the exact non-transversal implementation is not available, this study does not reflect on the flow of error in the concatenated circuit.

8.6.1 Overview of circuit decomposition for the adder

For an n-qubit adder circuit [DKRS06], the qubit-only decomposition of the Toffoli gate is shown in Fig. 8.3.

Let $Toffoli\_count$, $CNOT\_count$, T_count, and H_count denote the total count of Toffoli, $CNOT$, T, and Hadamard gates required respectively. Then

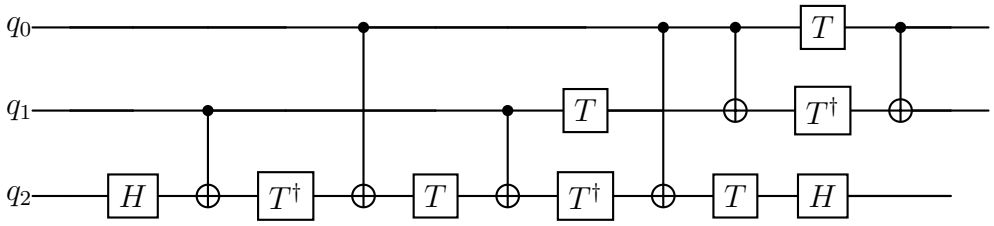


Figure 8.3: Toffoli decomposition with Clifford+T gates

$$Toffoli\_count_{add} = 10n - 3w(n) - 3w(n-1) - 3\log_2 n - 3\log_2(n-1) - 7 \quad (8.5)$$

where $w(n)$ denotes the number of ones in the binary representation of n. For the sake of simplicity, $w(n)$ is taken to be equal to n and $w(n-1)$ equal to $n-1$. With these values, the gate counts for the qubit-only decomposition of the Toffoli gates are given in Table 8.2.

Table 8.2: Gate counts for qubit-only decomposition of Toffoli gate for adder circuit

Gate type	Gate Count (as a function of n)
$CNOT$	$24n - 18\log_2 n - 18\log_2(n-1) - 24$
H	$8n - 6\log_2 n - 6\log_2(n-1) - 8$
T	$14n - 28\log_2 n - 28\log_2(n-1) - 21$

For qubit-qutrit decomposition, a Toffoli gate is decomposed using only 1-controlled and 2-controlled ternary CNOT gates. From Fig. 8.1 the number of 1-controlled ternary CNOT gates required for a Toffoli decomposition is noted to be twice that of 2-controlled ternary CNOT gates. The total number of 1-controlled and 2-controlled ternary CNOT gates required for qubit-qutrit decomposition of the adder circuit, as obtained from [SCCC22], are:

$$\begin{aligned} \#\text{1-controlled ternary CNOT} &= 8n - 6\log_2 n - 6\log_2(n-1) - 8 \\ \#\text{2-controlled ternary CNOT} &= 4n - 3\log_2 n - 3\log_2(n-1) - 4. \end{aligned}$$

8.6.2 Comparison of resource requirements

While the qubit-qutrit decomposition is capable of removing the requirement of T gates from the circuit, the 2-controlled ternary CNOT gate still remains non-transversal like the T gate. A T gate is a non-transversal gate for many QECCs and has a much costlier implementation. Several techniques have been proposed in the literature for efficient fault-tolerant implementation of T gates [P+22, Lit19], which are primarily for surface codes, and often involve complicated processes such as teleportation. For the sake of simplicity, in this chapter the fault-tolerant decomposition of T gates is considered with respect to Steane code [NC02] as shown in Fig. 8.4. The H, T, and SX gates are implemented transversally to attain the overall effect of an encoded T gate. Therefore, if Steane code is used, then κ_g for transversal gates is 7, whereas that for T gates is $4 \times 7 = 28$. Denote the gate count of the non-transversal implementation of a gate using Steane Code by $\tilde{\kappa_g} = \kappa_g/7$. Therefore, for T gate, $\tilde{\kappa_g} = 4$. Note that this implementation requires ancilla qubits. Therefore, if an n-qubit circuit involves m T gates, it leads to an overall circuit with $n + m$ qubits.

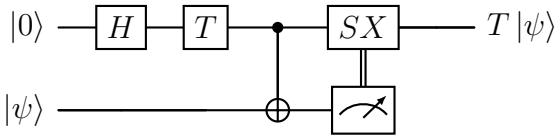


Figure 8.4: Fault tolerant implementation of T gate with Steane code

Let N_2 and $N_{2,3}$ denote the total number of gates required for the error-corrected implementation of the adder circuit using qubit-only and qubit-qutrit decomposition for a single level of concatenation. The number of gates increases exponentially with the number of levels of concatenation. $N_2(g)$ denotes the number of gates g in the concatenated implementation for qubit-only decomposition and $N_{2,3}(g)$ for qubit-qutrit decomposition.

Then for the qubit-only decomposition of an adder

$$\begin{aligned} N_2 &= 7 \times (N_2(CNOT) + N_2(H) + 4 \times N_2(T)) \\ &= 616n - 952log_2 n - 952log_2(n-1) - 798. \end{aligned} \tag{8.6}$$

As discussed earlier, for the Steane Code the 1-controlled ternary CNOT gate is transversal but the 2-controlled ternary CNOT is not. Then for the qubit-qutrit decomposition of an adder,

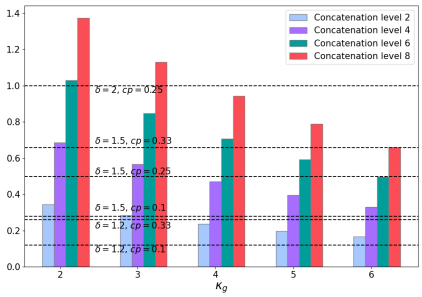
$$\begin{aligned} N_{2,3} &= 7 \times N_{2,3}(\text{1-controlled ternary CNOT}) + N_{2,3}(\text{2-controlled ternary CNOT}) \\ &= (56n - 42log_2 n - 42log_2(n-1) - 56) \\ &+ \tilde{\kappa_g} \times (28n - 21log_2 n - 21log_2(n-1) - 28). \end{aligned} \tag{8.7}$$

Note that if a transversal implementation were possible for the 2-controlled ternary CNOT, then $\tilde{\kappa_g} = 1$. In the following subsection, the value of $\tilde{\kappa_g}$ is varied to determine the scenarios (i.e. the gate cost of non-transversal implementation of the 2-controlled ternary CNOT gate) for which qubit-qutrit decomposition leads to a lower gate cost. For simplicity, in the numerical analysis, $c_2 = c_3 = c$.

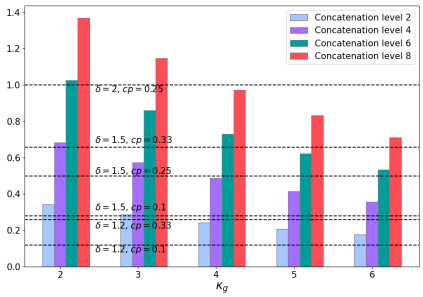
8.6.3 Numerical analysis

Fig. 8.5 presents the numerical values for the adder.

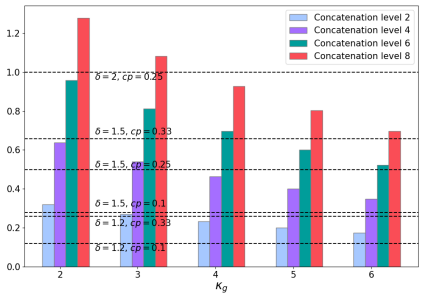
The resource requirement of the qubit-qutrit decomposition is lower than that for the qubit-only decomposition if the LHS of the inequality of Eq. (8.3) is less than that of the RHS. In Fig. 8.5, the RHS is shown as bars for different values of κ_g and levels of concatenation. The LHS for different values of δ and $c.p$ are shown as horizontal dashed lines. Therefore, the requirement of the inequality translates to the fact that qubit-qutrit decomposition requires a lower number of gates than that for the qubit-only decomposition if the bar goes above the corresponding



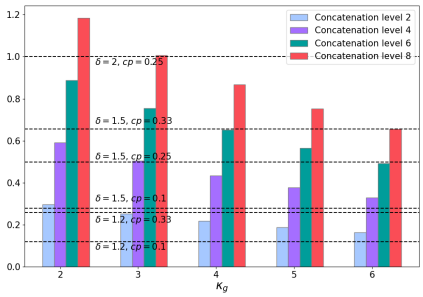
(a) 50 qubit adder



(b) 100 qubit adder



(c) 300 qubit adder



(d) 800 qubit adder

Figure 8.5: For $2 \leq \tilde{\kappa_g} \leq 6$ (refer to Eq. (8.7), the values of the RHS of the inequality of Eq. (8.3) are the heights of the bar-plots. The LHS of the inequality is indicated by the horizontal dashed lines for different values of δ and $c.p$. The qubit-qutrit decomposition leads to lower resource requirements for certain δ and $c.p$ when the LHS of Eq. (8.3) is less than RHS, i.e., the bar plots are higher than the corresponding horizontal dashed line.

horizontal dashed line. The primary observations from the plots are summarized below:

(i) For a fixed value of $\tilde{\kappa_g}$, increasing the level of concatenation increases the height of the bars. This implies that if a higher level of concatenation is used, then the qubit-qutrit decomposition eventually triumphs because of the exponential reduction in the number of gates required for the same. On the other hand, as the value of $\tilde{\kappa_g}$ increases, the heights of the bars are lower. This is also expected because as the cost of the non-transversal

implementation of the 2-controlled ternary CNOT gate increases, the benefit of the exponential reduction in gate count by the qubit-qutrit decomposition also diminishes.

(ii) The horizontal plots appear higher as the values of δ and $c.p$ increase. In other words, the noisier the qutrit hardware, the more difficult it is to obtain lower resources using qubit-qutrit decomposition. In the future, if the noise profile of qubit and qutrit devices become similar, then we expect that this qubit-qutrit decomposition will lead to lower resource requirements even for a high value of $\tilde{\kappa_g}$.

(iii) Finally, as the number of qubits is increased, the height of the bars increases initially and then decreases again. Therefore, apart from the value of $\tilde{\kappa_g}$ and the noise profile of the hardware, the number of qubits also plays a role in determining whether qubit-qutrit decomposition can provide benefit in terms of resource.

Fig. 8.5 illustrates the key observations with an adder circuit. Similar observations may be performed for other circuits of interest that require the decomposition of the Toffoli gates.

8.7 Summary

Many methods have been devised for the near-term quantum circuits which aim to lower the gate count and/or depth of the circuit. Although these methods may not be necessary in the error-corrected era of quantum computation, it may be beneficial to make use of these methods in that era as well. The question that remained largely unanswered is whether it is trivial to carry over those methods from near-term to error-corrected quantum computation. One such method is the use of higher dimensions in some intermediate cycles of computation to lower the depth of the circuit. This chapter analytically studied the challenges of extending this method to the error-corrected regime.

This study opens up a myriad of research directions. Primarily the study of resources for different binary and ternary QECCs by finding the proper transversal and non-transversal implementation of the gates is very salient. It can provide deeper insight into the settings where this type of decomposition is useful in the fault-tolerant era. It is also of interest whether using higher dimensions allows transversal (or non-transversal implementation with a low value of κ_g) implementation of the 2-controlled ternary CNOT gates. Future studies along this direction can conclusively dictate whether it is beneficial to use qutrit-assisted Toffoli decomposition in the fault-tolerant era.

CHAPTER 9

Conclusions and future directions

Contents

This thesis contributed to advancing the performance of both near-term and long-term quantum computing. For near-term quantum computation, in particular QAOA, the thesis provided algorithms to eliminate multiple 2-qubit gates, thus lowering the effect of noise, and improving the fidelity of the outcome. For general quantum circuits, the thesis provides two novel methods of error mitigation, targeted particularly for circuit cutting, to improve the performance of computation. For long-term quantum computing, this thesis provides insights into the challenges of designing ternary QECCs and provides the necessary conditions to ensure that errors can be corrected in a single step. Finally, this thesis also looks into the challenges of using methods from near-term quantum computing in conjunction with error correction, particularly in the context of Toffoli decomposition.

9.1 Summary

The contributions of this thesis are summarized on a chapter-by-chapter basis as follows -

- Chapter 3 provides two hardware-independent algorithms to eliminate multiple CNOT gates in a QAOA circuit while maintaining functional equivalence. The first method, based on Edge Coloring can eliminate $\lfloor \frac{|E|}{2} \rfloor$ CNOT gates for a graph $G = (V, E)$. The second method, based on Depth First Search, eliminates $|V| - 1$ CNOT gates, which is shown to be optimal, but increases the depth of the circuit to some extent which can be, at most, linear in the number of vertices. However, it was shown analytically that the noise lowered due to the elimination of CNOT gates overshadows the excess noise due to increased circuit depth, and the final optimized circuit exhibits lower noise probability.

- Chapter 4 proposes a heuristic algorithm that retains the optimal CNOT elimination for QAOA but restricts the increase in the depth of the circuit. It was numerically shown that, while the increase in depth using the heuristic algorithm is still linear in the number of vertices, the slope is lowered by $\simeq \frac{1}{10}$ as compared to the DFS approach. Finally, this algorithm is modified to respect the connectivity constraints of current quantum computers in order to lower the number of SWAP gates. The simulation results show that this modified heuristic, on average, exhibits a 5% reduction in the number of SWAP gates when compared to the original heuristic algorithm.

- Chapter 5 studies the performance of tomographic circuit cutting for different noise models. While circuit cutting itself can suppress the effect of noise, it still requires computing multiple instances for each subcircuit which may result in accumulation of noise. This chapter proposes two circuit-cutting-specific error mitigation methods, namely Measurement Error Mitigated Constrained Least Square (MEMCLS) and Dominant Eigenvalue Truncation (DEVT), which are shown to improve the fidelity of the outcome. DEVT,

together with MEMCLS, showed the best performance when the noise was close to depolarization. However, for stochastic Pauli noise, the performance degrades as the asymmetry in the noise increases; although even for this case they are shown to perform better than circuit cutting alone. Finally, some discussions are done on the scalability of this method. The results show that tomographic circuit cutting, together with DEVT, retains optimal performance even with partial ($\sim 60\%$) tomographic data.

- Chapter 6 studies the design of ternary QECC circuits as a carry-over of known binary codes. It shows that the stabilizer structure, and hence the gate cost, necessarily increases for the ternary counterpart. In particular, for the 9-qubit QECC, this chapter shows that the ternary counterpart requires more than one step to correct errors. Finally, this chapter proposed a 6-qutrit approximate code (AQECC) that can correct multiple phase errors in a single step but fails to identify the location of bit-flip errors in a few scenarios. The probability of success of the AQECC is $\sim 75\%$ for depolarization noise and increases to $\sim 99\%$ when the probability of Pauli-Z error is $100x$ that of Pauli-X error. The quantum cost of the circuit of this AQECC is 59.5% lower than that of the 9-qutrit QECC.

- Chapter 7 provides the necessary condition for the design of a ternary QECC as a carry-over of a binary one. It provides reasons why the previous attempts to design ternary QECC from binary one could not correct errors in a single step. Using the necessary condition, it shows the design of 9-qutrit, 7-qutrit, and 5-qutrit codes from their respective binary QECC, which can correct errors in a single step. In particular, designing the circuit of the 9-qutrit QECC maintaining this criterion achieves a reduction in the quantum cost by 51.9% as compared to the design of Chapter 6.

- Chapter 8 deals with the challenges of extending methods from near-term quantum computation to the error-corrected era. In particular, this chapter looks at the decomposition of Toffoli gates using intermediate qutrits. This method, without error correction, is shown to provide an exponential reduction in the depth of the decomposed circuit. This chapter first points out the

changes that need to be made to equip this method with error correction. For example, any qubit, that requires access to intermediate qutrits at any computation cycle, must be treated as a qutrit throughout for successful error correction. Finally, this chapter provides the analytical criterion for which the qutrit-assisted decomposition, in conjunction with error correction and concatenation, requires a lower gate count than qubit-only decomposition.

9.2 Future directions

There are several promising directions for future research based on the key contributions of this thesis.

The method for elimination of CNOT gates has been studied only for the context of QAOA in this thesis. However, such elimination methods are useful for any quantum circuits since they can immediately lower the noise. One future direction is to study similar methods for the circuits of other domains of importance such as VQE, Quantum machine learning, etc.

Chapter 7 provides a necessary condition for the design of ternary QECCs from binary ones. Recently, efficient preparation of qudits up to 5 dimensions, which have the same noise probability as qubits, has been proposed. Using higher dimensions can show significant improvement in the qubit count for certain problems. It may be worthwhile to study similar necessary conditions for the design of higher dimensional QECCs as carry-over of binary ones. It is expected that the necessary criteria will be very similar, with some room for freedom in circuit design. It is of interest to verify which of those design mechanisms leads to the lowest gate count, or if they are all equivalent in that respect. Moreover, this study is specific to concatenation stabilizer codes. Extension of other codes, such as topological codes, and LDPC codes, to ternary systems remains an open area of study.

The qubit-qutrit decomposition has a plethora of future prospects. It will be interesting to have a fault-tolerant design for some problem of interest, which

requires qubit-qutrit decomposition of Toffoli gates. Such a study will provide more concrete evidence of whether this method is useful in the fault-tolerant era of quantum computing or not. Eventually, it was shown in that chapter that some gates, required for qubit-qutrit decomposition, are not transversal in Steane Code. It is of utmost importance to study the non-transversal implementation of those gates for the Steane Code and to find some QECC where it is transversal.

These future studies are expected to provide further insights into improving the performance of quantum computation both in the near-term and long-term. Continued research efforts in these future directions will pave the way for more practical use-cases of quantum computation, and enable the design of fault-tolerant quantum computing for the future.

APPENDIX A

Proofs

A.1 Proof of Theorem 3.4

Let us consider the action of the operators U_1 and U_2 on any edge (j,k).

$$\begin{aligned}
U_1 |\psi\rangle &= CNOT_{jk}(I_j \otimes R_z(\theta_1)_k)(CNOT_{jk}) |\psi\rangle \\
&= \sum_{x_1,\ldots,x_n} CNOT_{jk}(I_j \otimes R_z(\theta_1)_k)(CNOT_{jk}) e^{i\phi(x_S)} |x_1, ..., x_n\rangle \\
&= \sum_{x_1,\ldots,x_n} CNOT_{jk}(I_j \otimes R_z(\theta_1)_k) e^{i\phi(x_S)} |x_1, .., x'_k = x_j \oplus x_k, ., x_n\rangle \\
&= \sum_{x_1,\ldots,x_n} e^{i(\phi(x_S) - \frac{\theta_1}{2}(x_j \oplus x_k))} CNOT_{jk} |x_1, .., x'_k = x_j \oplus x_k, ., x_n\rangle \\
&= \sum_{x_1,\ldots,x_n} e^{i(\phi(x_S) - \frac{\theta_1}{2}(x_j \oplus x_k))} |x_1, ..., x_n\rangle \qquad \text{(A.1)}
\end{aligned}$$

where $e^{i\phi(x_S)}$ is the cumulative effect of operators acting on previous edges; it is 0 if (j,k) is the first one in the circuit. We have dropped the normalization constant

for brevity.

Similarly,

$$\begin{aligned} U_2 |\psi\rangle &= CNOT_{jk}(I_j \otimes R_z(\theta_1)_{x_k}) |\psi\rangle \\ &= CNOT_{jk} \sum_{x_1,...,x_n} e^{i((\phi(x_S))-\frac{\theta_1}{2}x_k)} |x_1, ..., x_n\rangle \\ &= \sum_{x_1,...,x_n} e^{i((\phi(x_S))-\frac{\theta_1}{2}x_k)} |x_1, .., x_j \oplus x_k, .., x_n\rangle \end{aligned} \tag{A.2}$$

where the qubit in the k^{th} position changes to $x_j \oplus x_k$ due to the $CNOT_{jk}$ operation. By substituting $x'_k = x_j \oplus x_k$ in the above equation, we get

$$\begin{aligned} U_2 |\psi\rangle &= \sum_{x_1,...,x_n} e^{i((\phi(x_S))-\frac{\theta_1}{2}x_k)} |x_1, .., x_j \oplus x_k, .., x_n\rangle \\ &= \sum_{x_1,..,x'_k,..,x_n} e^{i((\phi(x_S))-\frac{\theta_1}{2}(x_j \oplus x'_k))} |x_1, .., x'_k, .., x_n\rangle \\ &= \sum_{x_1,..,x_k,..,x_n} e^{i((\phi(x_S))-\frac{\theta_1}{2}(x_j \oplus x_k))} |x_1, .., x_k, .., x_n\rangle \end{aligned} \tag{A.3}$$

Here since $k \notin S$, the substitution in the second last step, does not change the phase $e^{i\phi(x_S)}$. The last step is valid since x'_k is a running index and hence can be changed to x_k. Thus Eq. (A.1) and Eq. (A.3) are identical.

A.2 Proof of Corollary 3.4

The first time we consider an edge adjacent to a vertex j, where $j \notin x_S$, (see Theorem 3.4) the relative phase $\phi(x_S)$ does not depend on j. Thus it satisfies the condition of Theorem 3.4 and allows optimization of the operator.

On the other hand, if the vertex j occurs as part of an edge operator already

applied, the phase on the basis state ϕ can potentially depend on S, *i.e.* $j \in S$. By not allowing it to act as target, we satisfy the conditions of Theorem 3.4.

A.3 Proof of Lemma 3.5

Let us assume that there exists some other method to obtain a circuit with a lower depth. It is evident that all the edges, which can be operated on simultaneously, correspond to the same color in the graph. Therefore, if a circuit with a lower depth exists, then we can color the edges in the same layer with the same color, and obtain a better edge coloring of the input graph. However, we already assumed an optimal edge coloring, and this contradicts our assumption.

A.4 Proof of Theorem 3.5

For every edge (u, v) in the first layer, both the vertices are adjacent to an edge for the first time, *i.e.*, both $u, v \notin S$. Therefore, it satisfies the criteria of Corollary 3.4, and hence can be optimized. In fact, any one of the qubits corresponding to the two vertices can be selected as the control for the CNOT operation.

A.5 Proof of Theorem 3.6

We prove this by the method of induction. Let u be the vertex from which the DFS tree starts. Then u is being operated on for the first time, and, hence, can act both as a control/target for the CNOT operation corresponding to the first edge (Corollary 3.4). Choose u to be the control.

Base case: If v is the vertex that is discovered from u via the edge (u, v), then choosing u as the control and v as the target satisfies Corollary 3.4. Therefore,

the edge (u, v) can be optimized.

Induction hypothesis: Let the DFS tree has been constructed upto some vertex j, and every edge (e_1, e_2) in this DFS tree so far can be optimized, *i.e.* e_1 acts as the control and e_2 as the target.

Induction step: Let the next vertex in the DFS tree, that is discovered from some vertex i, is k. From DFS algorithm, the vertex i must have been discovered in some previous step. Since vertex k was not previously discovered, so $k \notin x_S$ and hence the edge (i, k) can be optimized if we select i to be the control and k as the target.

A.6 Proof of Theorem 3.6

Let us assume that there is some method by which at least n edges can be optimized. Now, the connected subgraph which contains all the n vertices and at least n optimized edges must contain a cycle. Let (u, v) be an edge of this cycle, *i.e.*, if (u, v) is removed then the residual graph is a tree (in case there are $> n$ edges, the removal of edges can be performed recursively till such an edge (u, v) is obtained whose removal makes the residual graph a tree). For this edge (u, v), both the vertices u and v are endpoints of some other optimized edges as well. Therefore, from Corollary 3.4 both u and v must act as the control for the CNOT gate corresponding to the edge (u, v) in order for this edge to be optimized, which is not possible. Therefore, it is not possible to optimize for more than $n-1$ edges.

A.7 Proof of Lemma 4.2.2

Let r be the randomly chosen root vertex of the spanning tree. Therefore, the choice of root does not require any computational time. Since Δ is the maximum degree of the graph, r can have at most Δ neighbours. Finding the maximum cost

function among these neighbours require $\mathcal{O}(\Delta)$ time. Subsequent vertices in the spanning tree can have at most $\Delta - 1$ neighbours since one of its neighbour must be its parent in the spanning tree. Therefore, the total time requirement in all the steps is

$$\begin{aligned} W &\leq \Delta \text{ (to create the spanning tree upto two vertices)} \\ &\leq \Delta + (\Delta - 1) \text{ (to create the spanning tree upto three vertices)} \\ &\leq 3\Delta - 2 \text{ (to create the spanning tree upto four vertices)} \\ &\vdots \end{aligned}$$

Therefore,

$$\begin{aligned} W &\leq \sum_{i=1}^{n-1} (i \cdot \Delta - (i-1)) \\ &= \Delta \cdot \sum_{i=1}^{n-1} i - \sum_{i=1}^{n-1} (i-1) \\ &= \mathcal{O}(\Delta \cdot n^2) \end{aligned}$$

A.8 DEVT with Measurement Errors

The results in fig. 5.5 establish numerically that DEVT mitigates measurement error better than MEMCLS, and padding MEMCLS with DEVT does not improve the result any further. In other words, DEVT is self sufficient for measurement error mitigation. Here it is shown analytically for linear inversion method of tomography that tomography with measurement error results in a noisy density matrix (or Choi matrix) of the form $\mathcal{E}(\rho) = (1-p)\rho + p\rho_{err}$. Therefore, DEVT alone is sufficient to mitigate the effect of measurement error.

Consider that $\{\Pi_j\}$ is a tomographically complete basis, with each Π_j being a projector. However, due to measurement error, the projectors are replaced by

POVMs of the form $\tilde{\Pi}_j = (1-p)\Pi_j + p\Pi'_j$, p being the probability of measurement error, and Π'_j is the linear combination of one or more unwanted projectors forming the POVM. If ρ be the state which is being measured, then the probability of success after measuring $\tilde{\Pi}_j$ is

$$\begin{aligned} \tilde{p_j} &= Tr[\tilde{\Pi}_j \rho] \\ &= (1-p)Tr[\Pi_j \rho] + pTr[\Pi'_j \rho] \\ &= (1-p)p_j + pp'_j \end{aligned}$$

The recreation of the state is carried out by creating the dual basis $|D_j\rangle\rangle = (\sum_j |\Pi_j\rangle\rangle\langle\langle\Pi_j|)^{-1}|\Pi_j\rangle\rangle$ [DMP00]. Since, it is not expected that the exact form of POVM due to noise is known, it can be assumed that the dual basis remains the same irrespective of the noise. Therefore, the recreated state $\tilde{\rho}$

$$\begin{aligned} \tilde{\rho} &= \sum_j \tilde{p_j} D_j \\ &= (1-p)\sum_j p_j D_j + p\sum_j p'_j D_j \\ &= (1-p)\rho + p\rho_{err} \end{aligned}$$

Therefore, as long as the largest eigenvalue of $\tilde{\rho}$ has a significant overlap with ρ, DEVT is sufficient to reestablish the error-free state ρ from the state $\tilde{\rho}$ created due to measurement error. In other words, DEVT alone is sufficient to mitigate measurement errors.

A.9 DEVT with depolarizing noise

Effect of depolarization error on a quantum state ρ is denoted as in eq. (5.17). Note that the effect of depolarization noise model is readily similar to the required effect of noise for applying DEVT, as shown in eq. (5.11) [Koc21]. A depolarization

channel remains depolarization even after $m \geq 1$ layers of gates. A single qubit state in a depolarization channel, after m layers of gates, has the form $\rho_{noisy} = (1-p)^m\rho + [1-(1-p)^m]\frac{\mathbb{I}}{2}$, where, for simplicity, p is assumed to be the probability of error for each gate. Therefore, for a n-qubit circuit with depth m, the effective erroneous state can be represented as

$$\begin{aligned} \rho^n_{noisy} &= \otimes^n_{i=1}\rho^{noisy}_i \\ &= \otimes^n_{i=1}(1-p)^m\rho_i + [1-(1-p)^m]\frac{\mathbb{I}}{2} \\ &= (1-p)^{n.m}\rho^n_{out} + \rho_{err}. \end{aligned} \tag{A.4}$$

Note that ρ_{err} is a summation of multiple density matrices. It is not possible to ascertain the largest eigenvalue of ρ_{err} without explicit information of ρ_i, $\forall\ i$. Hence, the only consideration possible is the worst case scenario that the largest eigenvalue of $\rho_{err} \leq 1$. Therefore, putting $\delta \leq (\frac{1}{(1-p)^{n.m}} - 1)$, an upper bound of the coherent mismatch c is obtained for depolarization noise model.

$$\begin{aligned} c &\leq \frac{\delta^2}{4} = \frac{1}{4}(\frac{1}{(1-p)^{n.m}} - 1)^2 \\ &= \frac{1}{4}[\frac{1-(1-n.m.p+\mathcal{O}(n^2m^2p^2))}{1-n.m.p+\mathcal{O}(n^2m^2p^2)}]^2 \\ &\approx \frac{1}{4}[\frac{n.m.p}{1-n.m.p}]^2 = \mathcal{O}((n.m.p)^2) \end{aligned} \tag{A.5}$$

A.10 DEVT with Pauli noise

This section briefly touches upon the seemingly worse performance of DEVT for pauli noise as opposed to that of depolarization noise. The evolution of a density matrix under pauli noise is shown in eq. (5.18), which conform to the form required for DEVT as in eq. (5.11). However, consider a circuit which applies k layers of gates on the input density matrix ρ_{in}. The ideal output density matrix ρ_{out} is,

thus, given by

$$\rho_{out} = G_k G_{k-1} \dots G_1 \rho_{in} G_1^\dagger \dots G_{k-1}^\dagger G_k^\dagger$$

Considering noisy implementation of each gate layer G_i to be $G_i' = P_i G_i$, where P_i consists of one or more pauli errors, the noisy output density matrix, under the action of such a noisy channel, becomes

$$\rho_{out}^{noisy} = \Pi_{i=1}^{k} P_i G_i \rho_{in} G_i^\dagger P_i$$

Since, in general, each G_i does not necessarily consist of Clifford gates only, the following scenario is obtained:

$$\rho_{out}^{noisy} = P_k P_{k-1} G_k G_{k-1} \Pi_{i=i}^{k-2} (P_i G_i \rho_{in} G_i^\dagger P_i) G_{k-1}^\dagger G_k^\dagger P_{k-1} P_k + [P_{k-1}, G_k]$$

In other words, the noisy output density matrix takes the eventual form

$$\rho_{out}^{noisy} = \Pi_{i=1}^{k} P_i (\Pi_{j=1}^{k} G_j \rho_{in} G_j^\dagger) P_i + comm \tag{A.6}$$

where *comm* denotes all the commutator terms, i.e., stochastic Pauli noise on each gate does not create a resultant stochastic Pauli circuit error. Therefore, the eventual form of the noisy density matrix deviates from eq. (5.11), resulting in a poorer performance of DEVT for this noise model.

A.11 Proof of Theorem 8.3

Proof. Let the accuracy obtained using only qubit and qubit-qutrit decomposition after k levels of concatenations be ϵ_2 and ϵ_3 respectively, where $\epsilon_3 = \delta \cdot \epsilon_2$. Let $\frac{1}{c_3}$ and $\frac{1}{c_2}$ be the thresholds of the ternary and binary QECCs used for encoding. Then,

$$\begin{aligned}
\frac{1}{c_3}(c_3.p_{2,3})^{2^k} &= \frac{\delta}{c_2}(c_2.p_2)^{2^k} \\
\Rightarrow \frac{c_2}{c_3}(c_3.p_{2,3})^{2^k} &= \delta(c_2.p_2)^{2^k} \\
\Rightarrow c(c_3.p_{2,3})^{2^k} &= \delta(c_2.p_2)^{2^k} \text{ where } c = \frac{c_2}{c_3} \\
\Rightarrow log(c) + 2^k log(c_3.p_{2,3}) &= log(\delta) + 2^k.log(c_2.p_2) \\
\Rightarrow log(\delta) - log(c) &= 2^k[log(c_3.p_{2,3}) - log(c_2.p_2)] \\
\Rightarrow log(\delta) - log(c) &= 2^k log\frac{p_{2,3}}{c.p_2} \\
\Rightarrow log(\delta) &= 2^k log\frac{p_{2,3}}{c.p_2} + log(c) \qquad \text{(A.7)}
\end{aligned}$$

□

A.12 Proof of Theorem 8.4

Proof. The accuracy obtained after k levels of concatenation with a QECC having threshold $\frac{1}{c}$ is $\frac{1}{c}(c.p)^{2^k}$, where p is the probability of error. In the current setting, both types of decomposition are attaining the same accuracy after k_2 and k_3 levels of concatenations. If $\frac{1}{c_3}$ and $\frac{1}{c_2}$ be the thresholds for ternary and binary QECCs used, then,

$$
\begin{aligned}
\frac{1}{c_3}(c_3.p_{2,3})^{2^{k_3}} &= \frac{1}{c_2}(c_2.p_2)^{2^{k_2}} \\
\Rightarrow \frac{c_2}{c_3}(c_3.p_{2,3})^{2^{k_3}} &= \delta(c_2.p_2)^{2^{k_2}} \\
\Rightarrow c(c_3.p_{2,3})^{2^{k_3}} &= \delta(c_2.p_2)^{2^{k_2}} \text{ where } c = \frac{c_2}{c_3} \\
\Rightarrow log(c) + 2^{k_3} log(c_3.p_{2,3}) &= 2^{k_2} log(c_2.p_2) \\
\Rightarrow 2^{k_3} log(c_3.p_{2,3}) &= 2^{k_2} log(c_2.p_2) - log(c) \\
\Rightarrow 2^{k_3-k_2} log(c_3.p_{2,3}) &= log(c_2.p_2) - \frac{1}{2^{k_2}} log(c) \\
\Rightarrow 2^{k_3-k_2} log(c_3.p_{2,3}) &= log(c_2.p_2) - log(c)^{\frac{1}{2^{k_2}}} \\
\Rightarrow 2^{k_3-k_2} &= \frac{log(c_2.p_2) - log(c)^{\frac{1}{2^{k_2}}}}{log(c_3.p_{2,3})} \\
\Rightarrow k_3 - k_2 &= log(\frac{log(c_2.p_2) - log(c)^{\frac{1}{2^{k_2}}}}{log(c_3.p_{2,3})}) \\
\Rightarrow k_3 &= k_2 + log(\frac{log(c_2.p_2) - \frac{1}{2^{k_2}} log(\frac{c_2}{c_3})}{log(\delta) + log(c_3.p_2)}) \text{ where } p_{2,3} = \delta.p_2
\end{aligned}
$$

□

A.13 Proof of Theorem 8.5

Proof. Qutrit-assisted decomposition is beneficial when the overall gate count of such decomposition is lower than that of the qubit decomposition. In other words,

$$(\sum_{g \in \mathcal{G}_{2,3}} \kappa_g n_g)^{k_3} \leq (\sum_{g \in \mathcal{G}_2} \kappa_g n_g)^{k_2}$$

Now,

$$\begin{aligned}
(\sum_{g\in\mathcal{G}_{2,3}} \kappa_g n_g)^{k_3} &\leq (\sum_{g\in\mathcal{G}_2} \kappa_g n_g)^{k_2} \\
\Rightarrow k_3 log(\sum_{g\in\mathcal{G}_{2,3}} \kappa_g n_g) &\leq k_2 log(\sum_{g\in\mathcal{G}_2} \kappa_g n_g) \\
\Rightarrow \frac{k_3}{k_2} &\leq \frac{log(\sum_{g\in\mathcal{G}_2} \kappa_g n_g)}{log(\sum_{g\in\mathcal{G}_{2,3}} \kappa_g n_g)} \\
\Rightarrow \frac{k_3}{k_2} - 1 &\leq \frac{log(\sum_{g\in\mathcal{G}_2} \kappa_g n_g)}{log(\sum_{g\in\mathcal{G}_{2,3}} \kappa_g n_g)} - 1
\end{aligned}$$

$$\Rightarrow k_3 - k_2 \leq k_2 \cdot \frac{log(\frac{\sum_{g\in\mathcal{G}_2} \kappa_g n_g}{\sum_{g\in\mathcal{G}_{2,3}} \kappa_g n_g})}{log(\sum_{g\in\mathcal{G}_{2,3}} \kappa_g n_g)} \tag{A.8}$$

By Theorem 8.3,
$k_3 = k_2 + log(\frac{log(c_2.p_2) - \frac{1}{2^{k_2}} log(\frac{c_2}{c_3})}{log(\delta) + log(c_3.p_2)})$. Substituting this in Eq. (A.8),

$$log(\frac{log(c_2.p_2) - \frac{1}{2^{k_2}} log(\frac{c_2}{c_3})}{log(\delta) + log(c_3.p_2)})) \leq k_2 \cdot \frac{log \frac{\sum_{g\in\mathcal{G}_2} (\kappa_g n_g)}{\sum_{g\in\mathcal{G}_{2,3}} (\kappa_g n_g)}}{log \sum_{g\in\mathcal{G}_{2,3}} (\kappa_g n_g)}$$

□

Bibliography

[AADQ22] R Ayanzadeh, N Alavisamani, P Das, and M Qureshi. Frozenqubits: Boosting fidelity of qaoa by skipping hotspot nodes. *arXiv preprint arXiv:2210.17037 [quant-ph]*, 2022.

[AAKV01] D Aharonov, A Ambainis, J Kempe, and U Vazirani. Quantum walks on graphs. In *Proceedings of the Thirty-third Annual ACM Symposium on Theory of Computing*, STOC '01, pages 50–59, New York, NY, USA, 2001. ACM.

[AASG20] M Alam, A Ash-Saki, and S Ghosh. Accelerating quantum approximate optimization algorithm using machine learning. In *2020 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pages 686–689. IEEE, 2020.

[aer22] Qiskit aer. https://github.com/Qiskit/qiskit-aer, 2022.

[AL18] T Albash and D A Lidar. Adiabatic quantum computation. *Reviews of Modern Physics*, 90(1):015002, 2018.

[AMMR13] M Amy, D Maslov, M Mosca, and M Roetteler. A meet-in-the-middle algorithm for fast synthesis of depth-optimal quantum circuits. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 32(6):818–830, 2013.

[APZB21] V Akshay, H Philathong, I Zacharov, and J Biamonte. Reachability deficits in quantum approximate optimization of graph problems. *Quantum*, 5:532, 2021.

[ARS+21] T Ayral, F Régent, Z Saleem, Y Alexeev, and M Suchara. Quantum divide and compute: exploring the effect of different noise sources. *SN Computer Science*, 2(3):1–14, 2021.

[ASZ+21] A Abbas, D Sutter, C Zoufal, A Lucchi, A Figalli, and S Woerner. The power of quantum neural networks. *Nature Computational Science*, 1(6):403–409, 2021.

[B+19] S Bravyi et al. Obstacles to state preparation and variational optimization from symmetry protection. *arXiv preprint arXiv:1910.08980*, 2019.

[BBB+23] Luciano B, Agata M B, Sergey B, et al. Circuit Knitting Toolbox. https://github.com/Qiskit-Extensions/circuit-knitting-toolbox, 2023.

[BBD+09] H J Briegel, D E Browne, W Dür, R Raussendorf, and M Van den Nest. Measurement-based quantum computation. *Nature Physics*, 5(1):19–26, 2009.

[BBMR20] A Bhattacharjee, C Bandyopadhyay, B Mondal, and H Rahaman. A survey report on recent progresses in nearest neighbor realization of quantum circuits. In *Soft Computing: Theories and Applications*, pages 57–68. Springer, 2020.

[BDG+20] J M Baker, C Duckering, P Gokhale, N C Brown, K R Brown, and F T Chong. Improved quantum circuits via intermediate qutrits. *ACM Transactions on Quantum Computing*, 1(1), October 2020.

[BDG+22] S Bravyi, O Dial, J M Gambetta, D Gil, and Z Nazario. The future of quantum computing with superconducting qubits. *Journal of Applied Physics*, 132(16):160902, 2022.

[BDS+23] S Basu, A Das, A Saha, A Chakrabarti, and S Sur-Kolay. Fragqc: An efficient quantum error reduction technique using quantum circuit fragmentation. *arXiv preprint arXiv:2310.00444*, 2023.

[BMKT22] E Berg, Z K Minev, A Kandala, and K Temme. Probabilistic error cancellation with sparse pauli-lindblad models on noisy quantum processors. *arXiv preprint arXiv:2201.09866*, 2022.

[BMSSK23] D Bhoumik, R Majumdar, A Saha, and S Sur-Kolay. Distributed scheduling of quantum circuits with noise and time optimization. *arXiv preprint arXiv:2309.06005*, 2023.

[BNR+20] P Barkoutsos, G Nannicini, A Robert, I Tavernelli, and S Woerner. Improving variational quantum optimization using cvar. *Quantum*, 4:256, 2020.

[BPK23] S Brandhofer, I Polian, and K Krsulich. Optimal partitioning of quantum circuits using gate cuts and wire cuts. *arXiv preprint arXiv:2308.09567*, 2023.

[BPP00] H Bechmann-Pasquinucci and A Peres. Quantum cryptography with 3-state systems. *Phys. Rev. Lett.*, 85:3313–3316, Oct 2000.

[BPS23] L Brenner, C Piveteau, and D Sutter. Optimal wire cutting with classical communication. *arXiv preprint arXiv:2302.03366*, 2023.

[BSCSK21] S Basu, A Saha, A Chakrabarti, and S Sur-Kolay. i-qer: An intelligent approach towards quantum error reduction. *ACM Transactions on Quantum Computing*, 2021.

[BSK+21] Sergey Bravyi, Sarah Sheldon, Abhinav Kandala, David C Mckay, and Jay M Gambetta. Mitigating measurement errors in multiqubit experiments. *Physical Review A*, 103(4):042605, 2021.

[BW20a] G Barron and C Wood. Measurement error mitigation for variational quantum algorithms. *arXiv preprint arXiv:2010.08520*, 2020.

[BW20b] L Burgholzer and R Wille. Advanced equivalence checking for quantum circuits. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2020.

[BWP+17] J Biamonte, P Wittek, N Pancotti, P Rebentrost, N Wiebe, and S Lloyd. Quantum machine learning. *Nature*, 549(7671):195–202, 2017.

[C+19] Y. Cao et al. Quantum chemistry in the age of quantum computing. *Chemical reviews*, 119(19):10856–10915, 2019.

[CA+21] M Cerezo, A Arrasmith, et al. Variational quantum algorithms. *Nature Reviews Physics*, 3(9):625–644, 2021.

[CBB+22] Z Cai, R Babbush, S C Benjamin, S Endo, W J. Huggins, Y Li, J R McClean, and T E O'Brien. Quantum error mitigation. *arXiv preprint arXiv:2210.00921*, 2022.

[CEB20] J Cook, S Eidenbenz, and A Bärtschi. The quantum alternating operator ansatz on maximum k-vertex cover. In *2020 IEEE International Conference on Quantum Computing and Engineering (QCE)*, pages 83–92. IEEE, 2020.

[CFG+22] M Cain, E Farhi, S Gutmann, D Ranard, and E Tang. The qaoa gets stuck starting from a good classical string. *arXiv preprint arXiv:2207.05089*, 2022.

[Cha97] H. F. Chau. Correcting quantum errors in higher spin systems. *Phys. Rev. A*, 55:R839–R841, Feb 1997.

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

[CLKAGG22] A Cervera-Lierta, M Krenn, A Aspuru-Guzik, and A Galda. Experimental high-dimensional greenberger-horne-zeilinger entanglement with superconducting transmon qutrits. *Phys. Rev. Applied*, 17:024062, Feb 2022.

[CLRS09] T Cormen, C Leiserson, R Rivest, and C Stein. *Introduction to algorithms*. MIT press, 2009.

[CRAB21] E Campos, D Rabinovich, V Akshay, and J Biamonte. Training saturation in layerwise quantum approximate optimization. *Physical Review A*, 104(3):L030401, 2021.

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

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

[DKRS06] T G Draper, S A Kutin, E M Rains, and K M Svore. A logarithmic-depth quantum carry-lookahead adder. *Quantum Info. Comput.*, 6(4):351–369, jul 2006.

[DMN13] S J Devitt, W J Munro, and K Nemoto. Quantum error correction for beginners. *Reports on Progress in Physics*, 76(7):076001, 2013.

[DMP00] G M D'Ariano, L Maccone, and M Paris. Orthogonality relations in quantum tomography. *Physics Letters A*, 276(1):25–30, 2000.

[DW11] Y Di and H Wei. Elementary gates for ternary quantum logic circuit. *arXiv preprint arXiv:1105.5485*, 2011.

[EBL18] S Endo, S Benjamin, and Y Li. Practical quantum error mitigation for near-future applications. *Physical Review X*, 8(3):031027, 2018.

[ECBY21] S Endo, Z Cai, S Benjamin, and X Yuan. Hybrid quantum-classical algorithms and quantum error mitigation. *Journal of the Physical Society of Japan*, 90(3):032001, 2021.

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

[EMG+22] A Eddins, M Motta, T P Gujarati, S Bravyi, A Mezzacapo, C Hadfield, and S Sheldon. Doubling the size of quantum simulators by entanglement forging. *PRX Quantum*, 3(1):010309, 2022.

[EMW21] D J Egger, J Mareček, and S Woerner. Warm-starting quantum optimization. *Quantum*, 5:479, 2021.

[F+18] R P Feynman et al. Simulating physics with computers. *Int. J. Theor. Phys*, 21(6/7), 2018.

[FB+18] M Fingerhuth, T Babej, et al. A quantum alternating operator ansatz with hard and soft constraints for lattice protein folding. *arXiv preprint arXiv:1810.13411*, 2018.

[FGG+01] E Farhi, J Goldstone, S Gutmann, J Lapan, A Lundgren, and D Preda. A quantum adiabatic evolution algorithm applied to random instances of an np-complete problem. *Science*, 292(5516):472–475, 2001.

[FGG14] E Farhi, J Goldstone, and S Gutmann. A quantum approximate optimization algorithm. *arXiv preprint arXiv:1411.4028*, 2014.

[FGG20] E Farhi, D Gamarnik, and S Gutmann. The quantum approximate optimization algorithm needs to see the whole graph: A typical case. *arXiv preprint arXiv:2004.09002*, 2020.

[FGGS00] E Farhi, J Goldstone, S Gutmann, and M Sipser. Quantum computation by adiabatic evolution. *arXiv preprint quant-ph/0001106*, 2000.

[FH16] E. Farhi and A. Harrow. Quantum supremacy through the quantum approximate optimization algorithm. *arXiv preprint arXiv:1602.07674*, 2016.

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

[FMT+22] L E Fischer, D Miller, F Tacchino, P K Barkoutsos, D J Egger, and I Tavernelli. Ancilla-free implementation of generalized measurements for qubits embedded in a qudit space. *arXiv preprint arXiv:2203.07369*, 2022.

[FSG09] A G Fowler, A M Stephens, and P Groszkowski. High-threshold universal quantum computation on the surface code. *Physical Review A*, 80(5):052312, 2009.

[GBD+19] P Gokhale, J M Baker, C Duckering, N C Brown, K R Brown, and F T Chong. Asymptotic improvements to quantum circuits via qutrits. In *Proceedings of the 46th International Symposium on Computer Architecture*, pages 554–566, 2019.

[GBD+20] P Gokhale, J Baker, C Duckering, F Chong, N Brown, and K Brown. Extending the frontier of quantum computers with qutrits. *IEEE Micro*, 40(3):64–72, 2020.

[GBL+23] A Gonzales, A Babu, J Liu, Z Saleem, and M Byrd. Fault tolerant quantum error mitigation. 2023.

[GCK+21] A Galda, M Cubeddu, N Kanazawa, P Narang, and N Earnest-Noble. Implementing a ternary decomposition of the toffoli gate on fixed-frequencytransmon qutrits. *arXiv preprint arXiv:2109.00558*, 2021.

[GD17] Daniel Greenbaum and Zachary Dutton. Modeling coherent errors in quantum error correction. *Quantum Science and Technology*, 3(1):015007, 2017.

[GEBM19] H Grimsley, S Economou, E Barnes, and N Mayhall. An adaptive variational algorithm for exact molecular simulations on a quantum computer. *Nature communications*, 10(1):1–9, 2019.

[GJAE+20] P Gokhale, A Javadi-Abhari, N Earnest, Y Shi, and F T Chong. Optimized quantum compilation for near-term algorithms with open-

pulse. In *2020 53rd Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*, pages 186–200. IEEE, 2020.

[GKD18] F Glover, G Kochenberger, and Y Du. A tutorial on formulating and using qubo models. *arXiv preprint arXiv:1811.11538*, 2018.

[GKHD22] F Glover, G Kochenberger, R Hennig, and Y Du. Quantum bridge analytics i: a tutorial on formulating and using qubo models. *Annals of Operations Research*, pages 1–43, 2022.

[GKW+18] M Grassl, L Kong, Z Wei, Z Q Yin, and B Zeng. Quantum error-correcting codes for qudit amplitude damping. *IEEE Transactions on Information Theory*, 64(6):4674–4685, 2018.

[GM62] M Gell-Mann. Symmetries of baryons and mesons. *Phys. Rev.*, 125:1067–1084, Feb 1962.

[Got97] D Gottesman. Stabilizer codes and quantum error correction. *arXiv preprint quant-ph/9705052*, 1997.

[Got98] D Gottesman. Fault-tolerant quantum computation with higher-dimensional systems. In *NASA International Conference on Quantum Computing and Quantum Communications*, pages 302–313. Springer, 1998.

[Got99] D Gottesman. *Fault-Tolerant Quantum Computation with Higher-Dimensional Systems*, pages 302–313. Springer Berlin Heidelberg, Berlin, Heidelberg, 1999.

[Gro96a] L K. Grover. A fast quantum mechanical algorithm for database search. In *Proceedings of the Twenty-eighth Annual ACM Symposium on Theory of Computing (STOC)*, STOC '96, pages 212–219, New York, NY, USA, 1996. ACM.

[Gro96b] L K Grover. A fast quantum mechanical algorithm for database search. STOC '96, pages 212–219, New York, NY, USA, 1996. ACM.

[GS18] D J Griffiths and D F Schroeter. *Introduction to quantum mechanics*. Cambridge university press, 2018.

[GSL18] A. Garcia-Saez and J. I. Latorre. Addressing hard classical problems with adiabatically assisted variational quantum eigensolvers. *arXiv preprint arXiv:1806.02287*, 2018.

[GTW09] A Gilchrist, D R Terno, and C J Wood. Vectorization of quantum operations and its use. *arXiv preprint arXiv:0911.2539*, 2009.

[GW95] M Goemans and D Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *Journal of the ACM (JACM)*, 42(6):1115–1145, 1995.

[GWYZ14] M Grassl, Z Wei, Z Q Yin, and B Zeng. Quantum error-correcting codes for amplitude damping. In *2014 IEEE International Symposium on Information Theory*, pages 906–910. IEEE, 2014.

[H+19] A Héctor et al. Qiskit: An open-source framework for quantum computing, 2019.

[Had18] S A Hadfield. *Quantum algorithms for scientific computing and approximate optimization*. Columbia University, 2018.

[HHL09] A W Harrow, A Hassidim, and S Lloyd. Quantum algorithm for linear systems of equations. *Physical review letters*, 103(15):150502, 2009.

[HKP20] H Huang, R Kueng, and J Preskill. Predicting many properties of a quantum system from very few measurements. *Nature Physics*, 16(10):1050–1057, 2020.

[HMM85] S. L. Hurst, D. M. Miller, and J. C. Muzio. *Spectral Techniques in Digital Logic*. London; Toronto: Academic Press, 1985.

[HSN+21] M Harrigan, K Sung, M Neeley, K Satzinger, F Arute, K Arya, J Atalaya, J Bardin, R Barends, S Boixo, et al. Quantum approximate optimization of non-planar graph problems on a planar superconducting processor. *Nature Physics*, 17(3):332–336, 2021.

[HWO+19] S Hadfield, Z Wang, B O'Gorman, E G Rieffel, D Venturelli, and R Biswas. From the quantum approximate optimization algorithm to a quantum alternating operator ansatz. *Algorithms*, 12(2):34, 2019.

[ibm22] IBM Quantum. https://quantum-computing.ibm.com/, 2022.

[IM07] L Ioffe and M Mézard. Asymmetric quantum error-correcting codes. *Phys. Rev. A*, 75:032345, Mar 2007.

[Jon13] C Jones. Low-overhead constructions for the fault-tolerant toffoli gate. *Phys. Rev. A*, 87:022328, Feb 2013.

[KE+23] Y Kim, A Eddins, et al. Evidence for the utility of quantum computing before fault tolerance. *Nature*, 618(7965):500–505, 2023.

[KG15] A Katabarwa and M R Geller. Logical error rate in the pauli twirling approximation. *Scientific reports*, 5(1):1–6, 2015.

[KKD15] A Kalev, R L Kosut, and I H Deutsch. Quantum tomography protocols with positivity are compressed sensing protocols. *npj Quantum Information*, 1(1):15018, 2015.

[KLV00] E Knill, R Laflamme, and L Viola. Theory of quantum error correction for general noise. *Phys. Rev. Lett.*, 84:2525–2528, Mar 2000.

[KMS+23] T Khare, R Majumdar, R Sangle, A Ray, P V Seshadri, and Y Simmhan. Parallelizing quantum-classical workloads: Profiling the impact of splitting techniques. *arXiv preprint arXiv:2305.06585*, 2023.

[KMT+17] A Kandala, A Mezzacapo, K Temme, M Takita, M Brink, J M Chow, and J M Gambetta. Hardware-efficient variational quantum eigensolver for small molecules and quantum magnets. *Nature*, 549(7671):242–246, 2017.

[Koc21] B Koczor. The dominant eigenvector of a noisy quantum state. *New Journal of Physics*, 23(12):123047, 2021.

[LB17] Y Li and S C Benjamin. Efficient variational quantum simulator incorporating active error minimization. *Phys. Rev. X*, 7:021050, Jun 2017.

[LDX19] Gushu Li, Yufei Ding, and Yuan Xie. Tackling the qubit mapping problem for nisq-era quantum devices. In *Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems*, pages 1001–1014, 2019.

[Lit19] D Litinski. A game of surface codes: Large-scale quantum computing with lattice surgery. *Quantum*, 3:128, 2019.

[LJJG22] J Larkin, M Jonsson, D Justice, and G G Guerreschi. Evaluation of qaoa based on the approximation ratio of individual samples. *Quantum Science and Technology*, 2022.

[LMH+22] A Lowe, M Medvidović, A Hayes, L J O'Riordan, T R Bromley, J M Arrazola, and N Killoran. Fast quantum circuit cutting with randomized measurements. *arXiv preprint arXiv:2207.14734 [quant-ph]*, 2022.

[LMPZ96] R Laflamme, C Miquel, J P Paz, and W H Zurek. Perfect quantum error correcting code. *Phys. Rev. Lett.*, 77:198–201, Jul 1996.

[Low23] P J Low. Control and readout of high-dimensional trapped ion qudits. 2023.

[MBGSK18] R Majumdar, S Basu, S Ghosh, and S Sur-Kolay. Quantum error-correcting code for ternary logic. *Phys. Rev. A*, 97:052302, May 2018.

[MBMSK16] R Majumdar, S Basu, P Mukhopadhyay, and S Sur-Kolay. Error tracing in linear and concatenated quantum circuits. *arXiv preprint arXiv:1612.08044*, 2016.

[MBSK17] R Majumdar, S Basu, and S Sur-Kolay. A method to reduce resources for quantum error correction. In *International Conference on Reversible Computation*, pages 151–161. Springer, 2017.

[MF21a] K Mitarai and K Fujii. Constructing a virtual two-qubit gate by sampling single-qubit operations. *New Journal of Physics*, 23(2):023021, 2021.

[MF21b] K Mitarai and K Fujii. Overhead for simulating a non-local channel with local channels by quasiprobability sampling. *Quantum*, 5:388, 2021.

[MG92] J Misra and D Gries. A constructive proof of vizing's theorem. In *Information Processing Letters*. Citeseer, 1992.

[MGE12] E Magesan, J M Gambetta, and J Emerson. Characterizing quantum gates via randomized benchmarking. *Phys. Rev. A*, 85:042311, Apr 2012.

[MRBAG16] J McClean, J Romero, R Babbush, and A Aspuru-Guzik. The theory of variational hybrid quantum-classical algorithms. *New Journal of Physics*, 18(2):023023, 2016.

[MS00] A Muthukrishnan and C. R. Stroud. Multivalued logic gates for quantum computation. *Phys. Rev. A*, 62:052309, Oct 2000.

[MSK20] R Majumdar and S Sur-Kolay. Approximate ternary quantum error correcting code with low circuit cost. In *2020 IEEE 50th International Symposium on Multiple-Valued Logic (ISMVL)*, pages 34–39. IEEE, 2020.

[MSK22] R Majumdar and S Sur-Kolay. Designing ternary quantum error correcting codes from binary codes. *Journal of Multiple-Valued Logic & Soft Computing*, To Appear, 2022.

[MSK23] R Majumdar and S Sur-Kolay. Designing ternary quantum error correcting codes from binary codes. *Journal of Multiple-Valued Logic & Soft Computing*, 40, 2023.

[mth22] Mthree. https://github.com/Qiskit-Partners/mthree, 2022.

[MWS+17a] D McKay, C Wood, S Sheldon, J Chow, and J Gambetta. Efficient z gates for quantum computing. *Physical Review A*, 96(2):022330, 2017.

[MWS+17b] D C. McKay, C J. Wood, S Sheldon, J M. Chow, and J M. Gambetta. Efficient z gates for quantum computing. *Phys. Rev. A*, 96:022330, Aug 2017.

[NC02] M A Nielsen and I Chuang. Quantum computation and quantum information, 2002.

[NC10] M A Nielsen and I L Chuang. *Quantum computation and quantum information*. Cambridge university press, 2010.

[NKSG21] P D. Nation, H Kang, N Sundaresan, and J M Gambetta. Scalable mitigation of measurement errors on quantum computers. *PRX Quantum*, 2(4):040326, 2021.

[P+22] L Postler et al. Demonstration of fault-tolerant universal quantum gate operations. *Nature*, 605(7911):675–680, 2022.

[Ped23] E Pednault. An alternative approach to optimal wire cutting without ancilla qubits. *arXiv preprint arXiv:2303.08287*, 2023.

[Per97] Asher Peres. *Quantum theory: concepts and methods*, volume 72. Springer, 1997.

[PHOW20] T Peng, A W Harrow, M Ozols, and X Wu. Simulating large quantum circuits on a small quantum computer. *Physical Review Letters*, 125(15):150504, 2020.

[PMS+14] A Peruzzo, J R McClean, P Shadbolt, et al. A variational eigenvalue solver on a photonic quantum processor. *Nature communications*, 5(1):4213, 2014.

[Pre18a] J Preskill. Quantum computing in the nisq era and beyond. *Quantum*, 2:79, 2018.

[Pre18b] J Preskill. Quantum computing in the nisq era and beyond. *Quantum*, 2:79, Aug 2018.

[PS22] C Piveteau and D Sutter. Circuit knitting with classical communication. *arXiv preprint arXiv:2205.00016 [quant-ph]*, 2022.

[PS23] C Piveteau and D Sutter. Circuit knitting with classical communication. *IEEE Transactions on Information Theory*, 2023.

[PSSO21] M A Perlin, Z H Saleem, M Suchara, and J C Osborn. Quantum circuit cutting with maximum-likelihood tomography. *npj Quantum Information*, 7(1):1–8, 2021.

[qex22] Qiskit experiments. https://github.com/Qiskit/qiskit-experiments, 2022.

[qis22] Qiskit Transpiler. https://qiskit.org/documentation/apidoc/transpiler.html, 2022.

[RK05] M Rahman and M Kaykobad. Complexities of some interesting problems on spanning trees. *Information Processing Letters*, 94(2):93–97, 2005.

[RMX+20] Yue Ruan, Samuel Marsh, Xilin Xue, Zhihao Liu, Jingbo Wang, et al. The quantum approximate algorithm for solving traveling salesman problem. *Computers, Materials and Continua*, 63(3):1237–1247, 2020.

[RSC+22] D Rabinovich, R Sengupta, E Campos, V Akshay, and J Biamonte. Progress towards analytically optimal angles in quantum approximate optimisation. *Mathematics*, 10(15):2601, 2022.

[S+20] N. Sawaya et al. Strategies for digital quantum simulation of bosons. *Bulletin of the American Physical Society*, 65, 2020.

[Sal20] Z H Saleem. Max-independent set and the quantum alternating operator ansatz. *International Journal of Quantum Information*, 18(04):2050011, 2020.

[SCC23] A Saha, A Chattopadhyay, and A Chakrabarti. Robust quantum arithmetic operations with intermediate qutrits in the NISQ-era. *International Journal of Theoretical Physics*, 62(4), apr 2023.

[SCCC22] A Saha, T Chatterjee, A Chattopadhyay, and A Chakrabarti. Intermediate qutrit-based improved quantum arithmetic operations with application on financial derivative pricing. *arXiv preprint arXiv:2205.15822*, 2022.

[Sel13] P Selinger. Quantum circuits of t-depth one. *Phys. Rev. A*, 87:042302, Apr 2013.

[SGS12] J A Smolin, J M Gambetta, and G Smith. Efficient method for computing the maximum-likelihood quantum state from measurements with additive gaussian noise. *Physical review letters*, 108(7):070502, 2012.

[Sho95] P W. Shor. Scheme for reducing decoherence in quantum computer memory. *Phys. Rev. A*, 52:R2493–R2496, Oct 1995.

[Sho96] P W Shor. Fault-tolerant quantum computation. In *Proceedings of 37th conference on foundations of computer science*, pages 56–65. IEEE, 1996.

[Sho97] P W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM J. Comput.*, 26(5):1484–1509, October 1997.

[SK10] P Sarvepalli and A Klappenecker. Degenerate quantum codes and the quantum hamming bound. *Phys. Rev. A*, 81:032318, Mar 2010.

[SMS+18] A Saha, R Majumdar, D Saha, A Chakrabarti, and S Sur-Kolay. Search of clustered marked states with lackadaisical quantum walks. *arXiv preprint arXiv:1804.01446*, 2018.

[SMS+20] A Saha, R Majumdar, D Saha, A Chakrabarti, and S Sur-Kolay. Asymptotically improved grover's algorithm in any dimensional quantum system with novel decomposed n-qudit toffoli gate. *arXiv preprint arXiv:2012.04447*, 2020.

[SMS+21] A Saha, R Majumdar, D Saha, A Chakrabarti, and S Sur-Kolay. Faster search of clustered marked states with lackadaisical quantum walks. *arXiv preprint arXiv:2107.02049*, 2021.

[SMS+22] A Saha, R Majumdar, D Saha, A Chakrabarti, and S Sur-Kolay. Asymptotically improved circuit for a d-ary grover's algorithm with advanced decomposition of the n-qudit toffoli gate. *Physical Review A*, 105(6):062453, 2022.

[SPR+20] Mohan Sarovar, Timothy Proctor, Kenneth Rudinger, Kevin Young, Erik Nielsen, and Robin Blume-Kohout. Detecting crosstalk errors in quantum information processors. *Quantum*, 4:321, 2020.

[SSC22] A Saha, D Saha, and A Chakrabarti. Moving quantum states without swap via intermediate higher-dimensional qudits. *Physical Review A*, 106(1):012429, 2022.

[SSP15a] M Schuld, I Sinayskiy, and F Petruccione. An introduction to quantum machine learning. *Contemporary Physics*, 56(2):172–185, 2015.

[SSP15b] M Schuld, I Sinayskiy, and F Petruccione. Simulating a perceptron on a quantum computer. *Physics Letters A*, 379(7):660–663, 2015.

[Ste96a] A. M. Steane. Error correcting codes in quantum theory. *Phys. Rev. Lett.*, 77:793–797, Jul 1996.

[Ste96b] A M Steane. Error correcting codes in quantum theory. *Physical Review Letters*, 77(5):793, 1996.

[STP+21] Z H Saleem, T Tomesh, M A Perlin, P Gokhale, and M Suchara. Divide and conquer for combinatorial optimization and distributed quantum computation. *arXiv preprint arXiv:2107.07532*, 2021.

[SWM+20] R Sweke, F Wilde, J Meyer, M Schuld, P K Fährmann, B Meynard-Piganeau, and J Eisert. Stochastic gradient descent for hybrid quantum-classical optimization. *Quantum*, 4:314, 2020.

[SWO+08] B Shaw, M M. Wilde, O Oreshkov, I Kremsky, and D A. Lidar. Encoding one logical qubit into six physical qubits. *Phys. Rev. A*, 78:012337, Jul 2008.

[TBG17] K Temme, S Bravyi, and J M Gambetta. Error mitigation for short-depth quantum circuits. *Physical review letters*, 119(18):180509, 2017.

[TCC+22] J Tilly, H Chen, S Cao, D Picozzi, K Setia, Y Li, E Grant, L Wossnig, I Rungger, et al. The variational quantum eigensolver: a review of methods and best practices. *Physics Reports*, 986:1–128, 2022.

[TM19] G. Torlai and R. Melko. Machine-learning quantum states in the nisq era. *Annual Review of Condensed Matter Physics*, 11, 2019.

[TSB+21] H L Tang, V Shkolnikov, G S Barron, H R Grimsley, N J Mayhall, E Barnes, and S E Economou. qubit-adapt-vqe: An adaptive algorithm for constructing hardware-efficient ansätze on a quantum processor. *PRX Quantum*, 2(2):020310, 2021.

[TTS+21] W Tang, T Tomesh, M Suchara, J Larson, and M Martonosi. Cutqc: using small quantum computers for large quantum circuit evaluations. In *Proceedings of the 26th ACM International Conference on Architectural Support for Programming Languages and Operating Systems*, pages 473–486, 2021.

[UADM22] G Uchehara, T M. Aamodt, and O Di Matteo. Rotation-inspired circuit cut optimization. *arXiv preprint arXiv:2211.07358*, 2022.

[vdBMKT22] E van den Berg, Z K Minev, A Kandala, and K Temme. Probabilistic error cancellation with sparse pauli-lindblad models on noisy quantum processors. *arXiv e-prints*, pages arXiv–2201, 2022.

[VDBMT22a] E Van Den Berg, Z K Minev, and K Temme. Model-free readout-error mitigation for quantum expectation values. *Physical Review A*, 105(3):032620, 2022.

[vdBMT22b] E van den Berg, Z K Minev, and K Temme. Model-free readout-error mitigation for quantum expectation values. *Phys. Rev. A*, 105:032620, Mar 2022.

[Viz64] V Vizing. On an estimate of the chromatic class of a p-graph. *Discret Analiz*, 3:25–30, 1964.

[WBC15] C J Wood, J D Biamonte, and D G Cory. Tensor networks and graphical calculus for open quantum systems. *Quant. Inf. Comp.*, 15:0579–0811, 2015.

[WE16] J J Wallman and J Emerson. Noise tailoring for scalable quantum computation via randomized compiling. *Phys. Rev. A*, 94:052325, Nov 2016.

[Wes01] D West. *Introduction to Graph Theory*, volume 2. Prentice hall Upper Saddle River, 2001.

[Won15] T G Wong. Grover search with lackadaisical quantum walks. *Journal of Physics A: Mathematical and Theoretical*, 48(43):435304, 2015.

[WVG+22] J Weidenfeller, L Valor, J Gacon, C Tornow, L Bello, S Woerner, and D Egger. Scaling of the quantum approximate optimization algorithm on superconducting qubit based hardware. *arXiv preprint arXiv:2202.03459*, 2022.

[WZ82] W K Wootters and W H Zurek. A single quantum cannot be cloned. *Nature*, 299(5886):802–803, 1982.

[YABAS20] Y S Yordanov, V Armaos, C Barnes, and D Arvidsson-Shukur. Iterative qubit-excitation based variational quantum eigensolver. *arXiv preprint arXiv:2011.10540*, 2020.

[YI22] Ed Younis and Costin Iancu. Quantum circuit optimization and transpilation via parameterized circuit instantiation. In *2022 IEEE International Conference on Quantum Computing and Engineering (QCE)*, pages 465–475. IEEE, 2022.

[ZTB+22] L Zhu, H L Tang, G S Barron, F A Calderon-Vargas, N J Mayhall, E Barnes, and S E Economou. Adaptive quantum approximate optimization algorithm for solving combinatorial problems on a quantum computer. *Physical Review Research*, 4(3):033029, 2022.