Cumulative Effects in Quantum Algorithms and Quantum Process Tomography

by

Shelby Kimmel

B.A. Astrophysics
Williams College (2008)

Submitted to the Department of Physics
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2014

© Massachusetts Institute of Technology 2014. All rights reserved.

Signature redacted

Author ..........

Signature redacted

Certified by.....

Signature redacted

Department of Physics

May 21, 2014

Edward Farhi

Cecil and Ida Green Professor of Physics

Thesis Supervisor

Signature redacted

Accepted by...

Signature redacted

Krishna Rajagopal

Physics Professor

Associate Department Head for Education
Abstract

This thesis comprises three results on quantum algorithms and quantum process tomography.

In the first section, I create a tool that uses properties of the quantum general adversary bound to upper bound the query complexity of Boolean functions. Using this tool I prove the existence of $O(1)$-query quantum algorithms for a set of functions called FAULT TREES. To obtain these results, I combine previously known properties of the adversary bound in a new way, as well as extend an existing proof of a composition property of the adversary bound.

The second result is a method for characterizing errors in a quantum computer. Many current tomography procedures give inaccurate estimates because they do not have adequate methods for handling noise associated with auxiliary operations. The procedure described here provides two ways of dealing with this noise: estimating the noise independently so its effect can be completely understood, and analyzing the worst case effect of this noise, which gives better bounds on standard estimates.

The final section describes a quantum analogue of a classical local search algorithm for Classical $k$-SAT. I show that for a restricted version of Quantum 2-SAT, this quantum algorithm succeeds in polynomial time. While the quantum algorithm ultimately performs similarly to the classical algorithm, quantum effects, like the observer effect, make the analysis more challenging.
Acknowledgments

I am indebted to many people who helped make this thesis and my research possible. To my advisor, Edward Farhi for his constant support, and for introducing me to so many exciting projects and people. To my thesis committee Peter Shor and Isaac Chuang, for feedback on issues ranging far beyond the contents of this thesis. To my many wonderful collaborators, Avinatan Hassidim, Bohua Zhan, Sam Gutmann, Andrew Childs, Robin Kothari, Marcus da Silva, Easwar Magesan, Isaac Chuang, Kristan Temme, Aram Harrow, Robin Blume-Kohout and Edward Farhi, for helping me to grow as a researcher by teaching me, challenging me, and encouraging me. To Scott Aaronson and Aram Harrow, for always making time to talk, whether about science or life.

I also want to thank all those people who helped me to maintain a vibrant work-life balance. For that I owe a debt to all of my friends, but especially to John Lee and Youbeau Oak for drumming with me even when they probably didn’t want to, and to my fellow CTP students, especially Ethan Dyer, Mark Mezei, and Jaehoon Lee, for always bringing their lunches back to Building 6 to eat with me. I'm grateful to my family, who has cheered me on in all my endeavors, and sat through more quantum computing talks that any non-scientist ever should. Finally I could not have done this without my supportive and loving husband, Paul Hess, with me every step of the way.

My work was supported by NSF Grant No. DGE-0801525, IGERT: Interdisciplinary Quantum Information Science and Engineering, by the U.S. Department of Energy under cooperative research agreement Contract Number DE-FG02-05ER41360, as well as Raytheon BBN Technologies Internal Research and Development.
Attributions and Contributions

Some of this thesis is taken from previously published material. When this is the case, a footnote at the beginning of the chapter will note which sections are taken from published papers. Some of this work has been done in collaboration with others. At the end of each chapter I have included a section in which I describe my contributions to that chapter. I have only included work in which I am a sole or significant contributor.
Contents

1 Introduction 13
  1.1 The Quantum Computer and the Beach .......................... 13
  1.2 Three-For-One Thesis ......................................... 16
    1.2.1 Composed Algorithms and the Adversary Upper Bound ... 16
    1.2.2 Robust Process Tomography ................................. 18
    1.2.3 Algorithm for Quantum k-SAT ............................... 20
  1.3 A Very Brief Introduction to the Notation and Mathematics of Quantum Computing ......................................................... 22
    1.3.1 Quantum Algorithms ....................................... 23

2 Quantum Adversary Upper Bound 25
  2.1 Introduction .................................................. 25
  2.2 A Nonconstructive Upper Bound on Query Complexity ........... 27
  2.3 Example where the Adversary Upper Bound is Useful ............ 29
  2.4 Non-Optimal Algorithm for Composed FAULT TREES ............ 33
    2.4.1 Set-Up .................................................. 33
    2.4.2 Span Program Algorithm .................................. 38
    2.4.3 Direct Functions ........................................ 42
  2.5 Optimal Span Program Algorithm for CONSTANT-FAULT TREES .. 43
  2.6 Conclusions .................................................. 47
  2.7 Acknowledgements ............................................. 48
  2.8 Contributions ................................................ 48
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>Robust Randomized Benchmarking</td>
<td>49</td>
</tr>
<tr>
<td>3.1</td>
<td>Introduction</td>
<td>49</td>
</tr>
<tr>
<td>3.2</td>
<td>Completely Positive Trace Preserving Maps: Notation and Properties</td>
<td>51</td>
</tr>
<tr>
<td>3.3</td>
<td>Randomized Benchmarking of Clifford Group Maps</td>
<td>53</td>
</tr>
<tr>
<td>3.3.1</td>
<td>RB Sequence Design</td>
<td>56</td>
</tr>
<tr>
<td>3.4</td>
<td>Tomographic Reconstruction from RB</td>
<td>57</td>
</tr>
<tr>
<td>3.4.1</td>
<td>Example: Single Qubit Maps</td>
<td>60</td>
</tr>
<tr>
<td>3.4.2</td>
<td>Beyond Unital Maps</td>
<td>63</td>
</tr>
<tr>
<td>3.5</td>
<td>Fidelity Estimation Beyond the Clifford Group</td>
<td>64</td>
</tr>
<tr>
<td>3.5.1</td>
<td>Average Fidelity to $T$</td>
<td>65</td>
</tr>
<tr>
<td>3.5.2</td>
<td>Average Fidelity to More General Unitaries</td>
<td>67</td>
</tr>
<tr>
<td>3.6</td>
<td>Bounding Error in Average Fidelity Estimates</td>
<td>69</td>
</tr>
<tr>
<td>3.6.1</td>
<td>Bounds on Average Fidelity of Composed Maps</td>
<td>70</td>
</tr>
<tr>
<td>3.6.2</td>
<td>Confidence Bounds on Fidelity Estimates</td>
<td>71</td>
</tr>
<tr>
<td>3.7</td>
<td>Experimental Implementation and Analysis</td>
<td>74</td>
</tr>
<tr>
<td>3.7.1</td>
<td>RBT Experiments</td>
<td>75</td>
</tr>
<tr>
<td>3.7.2</td>
<td>QPT Experiments</td>
<td>78</td>
</tr>
<tr>
<td>3.7.3</td>
<td>DFE Experiments</td>
<td>80</td>
</tr>
<tr>
<td>3.7.4</td>
<td>Bootstrapping</td>
<td>81</td>
</tr>
<tr>
<td>3.7.5</td>
<td>Experimental Results</td>
<td>83</td>
</tr>
<tr>
<td>3.8</td>
<td>Summary and Outlook</td>
<td>88</td>
</tr>
<tr>
<td>3.9</td>
<td>Acknowledgments</td>
<td>89</td>
</tr>
<tr>
<td>3.10</td>
<td>Contributions</td>
<td>90</td>
</tr>
<tr>
<td>4</td>
<td>Quantum Local Search Algorithm for Quantum 2-SAT</td>
<td>91</td>
</tr>
<tr>
<td>4.1</td>
<td>Introduction</td>
<td>91</td>
</tr>
<tr>
<td>4.2</td>
<td>Analysis of the Quantum Algorithm for Restricted Clauses</td>
<td>94</td>
</tr>
<tr>
<td>4.2.1</td>
<td>Runtime Bound for the Algorithm</td>
<td>95</td>
</tr>
<tr>
<td>4.2.2</td>
<td>Solving the Quantum 2-SAT Decision Problem</td>
<td>99</td>
</tr>
<tr>
<td>4.2.3</td>
<td>Difficulties with General Clauses</td>
<td>102</td>
</tr>
</tbody>
</table>
4.3 Conclusions .............................................. 103
4.4 Contributions ........................................... 104

A Fault Trees and the Adversary Upper Bound 105
   A.1 Direct Functions and Monotonic Functions ............... 105
   A.2 Trivial Inputs to Direct Functions ......................... 106
   A.3 Composition Proof .................................. 107

B Robust Process Tomography 113
   B.1 Unital Maps and the Linear Span Of Unitary Maps ....... 113
   B.2 Reconstruction of the Unital Part with Imperfect Operations .... 115
   B.3 Complete-Positivity of the Projection of Single Qubit Operations onto the Unital Subspace ................................. 116
   B.4 Bounds on Fidelity ................................... 120
   B.5 Ordering Of Error Maps ................................ 121

C Quantum 2-SAT with Less Restricted Clauses 125
   C.1 Analysis with an Extended Clause Set .................... 125
Chapter 1

Introduction

1.1 The Quantum Computer and the Beach

Quantum computers are a tantalizing technology that, if built, will change the way we solve certain difficult problems. Quantum computers replace the bits of a classical computer with quantum 2-dimensional systems (qubits), replace the standard logical operators AND and NOT with a set of unitary operations (such as HADAMARD, PHASE, and C-NOT), and replace read-out with quantum measurement. With these substitutions, one finds that certain computational problems can be solved in less time with a quantum computer than with a classical computer. The two most well known examples of quantum speed-ups are Shor’s factoring algorithm [68] and Grover’s search algorithm [34], but the number of quantum algorithms is growing\(^1\), and several new algorithms are included in this thesis.

Even though a working quantum computer may be many years away, already the ideas of quantum computation have revolutionized the way we think about computing and have broadened the toolbox of computer scientists. Quantum computers have forced a rewrite of the topography of computational complexity theory. Moreover, tools from quantum computing are proving useful in the classical world. For example, given a Boolean function \( f \), there are not many classical tools available to determine

how many fan-in AND and OR gates are needed to represent \( f \) as a formula. However, given an upper bound on the quantum query complexity of \( f \), one can immediately obtain a lower bound on the number of gates needed to express \( f \) as a formula [19]. For a survey of applications of quantum techniques used in classical proofs, see Ref. [27].

Two great challenges for quantum computer scientists are to determine the computational power of quantum computers, and to engineer physical systems whose error rates are low enough that quantum computing is possible. This thesis will address both these topics; we study and analyze several quantum algorithms, giving insight into the computational power of quantum computers, and we describe new methods for accurate quantum tomography, which has implications for the design of improved quantum systems.

Understanding the power and limits of quantum computation is closely related to algorithms. Each quantum algorithm gives us a new lens through which to view the interaction between quantum mechanics and computation. For example, different algorithms take advantage of various features of quantum mechanics, such as teleportation, entanglement, superposition, or the existence of a quantum adiabatic theorem. As we create algorithms that use these features as resources, we learn which elements are necessary for a quantum speed-up. Additionally, through the creation of algorithms, we gain a sense of which types of problems quantum computers are good at, and for which problems there is no quantum advantage. Tools for bounding algorithms make these intuitions more precise; for example, there is a theorem that there is no quantum algorithm that uses exponentially fewer queries than a classical algorithm to evaluate a total Boolean function [9]. In this thesis, we use several new tools to analyze and bound the performance of quantum algorithms.

Quantum process tomography, the characterization of a quantum operation acting on a quantum system, is a critical part of building quantum devices with lower error rates. Current small-scale quantum systems can not be scaled up to create larger quantum computers because their error rates are above the thresholds needed to apply quantum error-correcting codes. (This situation may soon be changing - recently a
superconducting qubit system was observed to have error rates at the threshold needed to apply a surface code [7]. In order to reduce errors in these quantum systems, it is helpful to be able to accurately characterize the errors that occur. Accurate characterization of errors using quantum process tomography can help to pinpoint sources of noise and suggest methods for reduction or correction. In this thesis, we will describe a process characterization procedure that is more accurate than many standard tomography procedures.

The three stories in this thesis (two on algorithms, one on tomography) are on the surface unrelated, but all take advantage of cumulative effects to analyze and understand the behavior of quantum systems. To understand these cumulative effects, we'll start with an analogy to the beach.

The beaches of Wellfleet, MA are eroding at a rate of about three feet per year [57] as the dune cliff is dragged out to the ocean. With an order of magnitude estimate of 10 waves a minute, each wave accounts, on average, for about 0.1 micrometers of shoreline erosion. The impact of a single wave on erosion is both small and extremely noisy. However, by taking the cumulative effect of millions of waves into account, we can easily estimate the average action of an individual wave, simply by taking a ruler to the beach once a year. Consider if instead we tried to calculate the average erosional power of a single wave by modeling the effect of a wave on the dunes, then averaged over the distribution of possible wave energies and wind speeds, and kept track of the changing dune shapes and sandbar configurations, which themselves depend on interactions with earlier waves. Compared to the ruler on the sand, this method will be computationally difficult and likely inaccurate.

While this example is essentially glorified averaging, the principles prove to be powerful in the quantum regime. We give two examples, one from quantum algorithms, and one from quantum process tomography, where observing cumulative effects allows for more accurate characterization of an individual algorithm or quantum process than would be possible through direct observation of that single algorithm or process. We call this the top-down approach.

The third example we consider also deals with cumulative effects in a quantum
algorithm, but in fact approaches the problem from a bottom-up approach; to go back
to our beach analogy, our approach is akin to estimating yearly erosion by accurately
c characterizing the effects of a single wave. We apply this bottom-up approach to
analyze a quantum algorithm where each step of the algorithm is well understood,
yet determining the cumulative effect of many steps of the algorithm is challenging.

In the next sections of this chapter, we introduce the three stories that will be
covered in this thesis, focussing on the importance of cumulative effects in each. The
final section in this chapter is a brief introduction to some of the basics of quantum
computing and quantum algorithms. If you are unfamiliar with those topics, we
suggest reading Section 1.3 first.

1.2 Three-For-One Thesis

All three topics in this thesis are related to quantum computing, but the techniques,
tools, and goals vary widely between them. However, all three topics deal with
cumulative effects in quantum systems.

1.2.1 Composed Algorithms and the Adversary Upper Bound

In Chapter 2, we study the relationship between quantum algorithms for a function
and quantum algorithms for that function composed with itself. A composed function
\( f^k \) is the function \( f^k = f \circ (f^{k-1}, \ldots, f^{k-1}) \), where \( f^1 = f \), and we call \( f \) the non-
composed function. To learn something about a composed function, a natural place
to start is to study the non-composed function.

Let's consider a specific example. Suppose we have a Boolean function \( f \) that
takes \( n \) bits as input \( f : \{0, 1\}^n \rightarrow \{0, 1\} \). Thus \( f^k \) is a function of \( n^k \) input bits. We
would like to know the number of input bit values to \( f^k \) we need to learn in order
to correctly predict the output of \( f^k \), assuming that we initially do not know the
values of any of the input bits. We call this quantity the exact query complexity of
\( f^k \). This seems like a difficult problem, but it becomes simple if we know something
about \( f \). Suppose we have an algorithm that allows us to predict the output of \( f \) with
certainty, after learning the values of $m$ input bits, where $m \leq n$. Then an obvious strategy to learn the output of $f^k$ is to apply the algorithm for $f$ at each level of composition. A simple inductive argument on $k$ shows that we can evaluate $f^k$ after learning $m^k$ bits. This algorithm is not necessarily optimal, but is at least an upper bound.

As this example shows, a bottom up approach is natural for learning about quantum algorithms for composed functions, since an algorithm for $f$ can produce an algorithm for $f^k$. In Chapter 2, we will show an example of an opposite top-down effect, where an algorithm for $f^k$ can provide information about an algorithm for $f$. In particular, we prove a theorem called the Adversary Upper Bound, which allows us to precisely relate the query complexity of an algorithm for a composed function to the query complexity of an algorithm for the non-composed function.

In Chapter 2, we apply the Adversary Upper Bound to a function $f$ with a quantum algorithm whose query complexity depends on two different contributions, the size of the function and the structure (which depends on specific properties of the input to $f$). In this algorithm for $f$, size is the dominant contribution. However, when we look at a similar algorithm for $f^k$, structure is the dominant contribution to query complexity, and size is less important. We show that because size is not a significant factor in the query complexity of the composed function, size must not have been a necessary component in the original algorithm for $f$, and in fact, the query complexity of $f$ must not depend on size at all. Studying the composed function algorithm allows us to learn something about the algorithm for the non-composed function.

Just as measuring the cumulative effect of many waves can wash out noise or error that might exist in an observation of a single wave, noting properties of a composed algorithm can clarify which parts of the algorithm for the non-composed function are necessary and which parts are noise. In our example, the dependence on depth turned out to be noise, but it was not obvious that this was the case until we analyzed the algorithm for the composed function.
1.2.2 Robust Process Tomography

As we mentioned in Section 1.1, quantum process tomography is an important tool for building better quantum computers. However, many standard techniques for characterizing quantum processes are subject to systematic errors. These systematic errors mean that even if arbitrarily large amounts of data are taken, the results will not converge to an accurate estimate of the process in question.

To see why this is the case, let’s consider one of the most straightforward methods for quantum process tomography [22], QPT. In this protocol, we prepare a state, apply the quantum process to be characterized, and then measure the resulting state using a projective measurement. If sufficiently many initial states are used, and if sufficiently many different measurements are made for each initial state, then in principle we can estimate all of the parameters of the unknown quantum process. However, if there is uncertainty in the initial state preparations or measurements, systematic errors and biases will likely be introduced into the estimate.

Key aspects of a QPT procedure are illustrated in Figure 1-1. For example, a common experiment that is performed in QPT is to prepare the state \( |0\rangle \), apply \( \mathcal{E} \) (where \( \mathcal{E} \) is the quantum process to be characterized), and then measure in the basis \( \{ |0\rangle, |1\rangle \} \). This idealized experiment is diagrammed in Figure 1-1a. However, in a real system, we can not prepare the \( |0\rangle \) state perfectly. In fact, we generally don’t even have an accurate characterization of the true state that is prepared. Likewise a perfect \( \{ |0\rangle, |1\rangle \} \) measurement is not realistic for a physical system, and we generally don’t have an accurate characterization of the true measurement that is performed.

While Figure 1-1a is the idealized experiment, Figure 1-1b shows the experiment that is actually performed. We represent our faulty state preparation as preparing the state \( |0\rangle \) perfectly followed by some (at least partially) unknown error process \( \Lambda_0 \), and we represent our faulty measurement as a perfect measurement in the \( \{ |0\rangle, |1\rangle \} \) basis, preceded by some (at least partially) unknown error process \( \Lambda_{0/1} \). The true state preparation and measurement can always be written in this form. Now we see that if we assume we have a perfect state preparation and perfect measurement, the
process that we characterize is not $\mathcal{E}$ as desired, but $\Lambda_{0/1} \circ \mathcal{E} \circ \Lambda_0$ (where $\circ$ denotes composition of quantum processes, and the order of application proceeds right to left).

![Diagram of QPT experiments](image)

Figure 1-1: We demonstrate a limitation of standard quantum process tomography (QPT) when there are uncertainties in state preparation and measurement. In Fig. (a), we diagram an ideal QPT experiment characterizing a process $\mathcal{E}$ by preparing a state $|0\rangle$, applying the process $\mathcal{E}$, and then measuring in the $\{|0\rangle, |1\rangle\}$ basis. In Fig. (b), we diagram the true QPT experiment, in which there are errors on state preparation and measurement. In Fig. (c) we show that when state preparation and measurement change, the errors on state preparation and measurement also change. These state preparation and measurement errors produce inaccuracies in the final estimate of $\mathcal{E}$, as well as estimates of $\mathcal{E}$ that are often not valid quantum processes.

While errors on state preparation and measurement result in QPT characterizing a different process from the desired one (we estimate $\Lambda_{0/1} \circ \mathcal{E} \circ \Lambda_0$ rather than $\mathcal{E}$), what is worse is what happens when we prepare a different state or measure in a different measurement basis. As before, these states and measurements have errors associated with them, but importantly, these are different errors than the errors associated with the state preparation $|0\rangle$ and the measurement $\{|0\rangle, |1\rangle\}$. We see this in Figure 1-1c where the state preparation $|+\rangle$ is associated with the error $\Lambda_+$ and the measurement $\{|+, |\rangle\}$ is associated with the error $\Lambda_{+/-}$. Now not only are we characterizing $\Lambda_i \circ \mathcal{E} \circ \Lambda_j$ rather than $\mathcal{E}$, but $\Lambda_i$ and $\Lambda_j$ change as state preparation and measurement...
change. If we assume that all state preparations and measurements are perfect, then not only will we get an inaccurate estimate of $E$, but the estimate will often not be a valid quantum process.

We see that QPT is not robust to uncertainties in state preparation and measurement. We would like a protocol that is robust - that produces an accurate estimate of $E$ even when there are uncertainties in state preparation and measurement. In Chapter 3 we create a robust protocol using cumulative effects. Instead of acting with $E$ a single time between state preparation and measurement, we apply $E$ multiple times. By doing this in a clever way, we can separate the signal due to $E$ from the noise of state preparation and measurement, resulting in more accurate estimates of $E$.

The top-down approach uses cumulative effects to separate out which contributions to a process are important and which are not. In Section 1.2.1 we described how we will use a top-down approach to determine which contributions to an algorithm are important and which are unnecessary. Here, we use a top-down approach to separate out which contributions to an experiment are due to the quantum operation we care about, and which are due to noise from state preparation, measurement, or other extraneous operations.

### 1.2.3 Algorithm for Quantum $k$-SAT

In Chapter 4, we again return our attention to algorithms. In particular we consider an algorithm for Quantum $k$-SAT. Quantum $k$-SAT is an important problem in quantum complexity because if we can efficiently solve Quantum $k$-SAT, then we can efficiently solve a host of related and difficult problems [16, 32]. Quantum $k$-SAT is a quantum version of a classical problem called Classical $k$-SAT. Both Quantum $k$-SAT and Classical $k$-SAT can be solved using a brute force search. A brute force search involves searching through all possible solutions to find the correct one. Since there are an exponential number of possible solutions for both Classical $k$-SAT and Quantum $k$-SAT, this takes an exponential amount of time (see the first row of Table 1.1).

For Classical $k$-SAT, using a simple local search algorithm significantly improves the speed of the classical algorithm. A local search algorithm takes the current state
Table 1.1: Comparing the performance of classical and quantum algorithms for Classical 3-SAT and Quantum 3-SAT. Local search is faster than brute search for Classical 3-SAT with a quantum or classical computer, but we don’t know how helpful a local search will be for solving Quantum 3-SAT on a quantum computer.

<table>
<thead>
<tr>
<th></th>
<th>Classical Algorithm for $n$-bit Classical 3-SAT</th>
<th>Quantum Algorithm for $n$-bit Classical 3-SAT</th>
<th>Quantum Algorithm for $n$-qubit Quantum 3-SAT</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Brute Search</strong></td>
<td>$O(2^n)$</td>
<td>$O(\text{poly}(n)2^{n/2})$ (Grover [3])</td>
<td>$O(\text{poly}(n)2^{n/2})$ [59]</td>
</tr>
<tr>
<td><strong>Local Search</strong></td>
<td>$O(1.334^n)$ [66]</td>
<td>$O(\text{poly}(n)1.334^{n/2})$ (Amplitude Amplification [3])</td>
<td>unknown</td>
</tr>
</tbody>
</table>

of the system and does a search among states that are similar to the current state. The local search technique also speeds up solving Classical $k$-SAT using a quantum computer. (See the second column of Table 1.1). However, it is unknown whether a local search can improve the performance of a quantum algorithm for Quantum $k$-SAT, and hence the “unknown” in the bottom right of Table 1.1. In Chapter 4 we investigate whether such a speed-up is possible.

While performing a quantum local search may be faster than brute search, the analysis of the performance of the algorithm is more complex. This complexity seems counterintuitive, because (as we will see in Chapter 4) at each time step, the quantum local search algorithm performs the same simple operation, which we understand completely. However, the algorithm is difficult to analyze because the progress it makes at a given time step depends on the state of the system at that step, producing a non-linear effect. It makes sense that the progress of the algorithm depends on the state of the system: in a local search we look for options that are close to the current state of the system, and if the current state is in a bad position, it might not be possible to find good options to move to.

In Chapter 4, to analyze the quantum local search algorithm, we use a bottom-up approach. As before, we study a cumulative process, but now we have precise
knowledge of a single step and need to extrapolate to determine the aggregate effects. To do this, we study the action of the algorithm in detail, and apply a tracking device that roughly keeps track of the state of the system over time. Using this strategy, we analyze the performance of the algorithm for a restricted version of Quantum 2-SAT, which is the simplest version of Quantum k-SAT.

### 1.3 A Very Brief Introduction to the Notation and Mathematics of Quantum Computing

Quantum computers manipulate quantum states. A pure quantum state of n qubits is an element of the $d = 2^n$ dimensional complex vector space with $l^2$-norm equal to 1, and is denoted by $|\psi\rangle$. Ideal operations that act on quantum states are unitary matrices of dimension $d$ (unitary operations ensure that the norm is preserved). More generally, a probabilistic quantum state $\rho$ of n qubits is a $2^n$ dimensional positive semidefinite matrix with trace 1. A pure state $|\psi\rangle$ corresponds to the rank-1 matrix $|\psi\rangle\langle\psi|$ where $\langle\psi|$ is the conjugate transpose of $|\psi\rangle$. General quantum processes acting on such n-qubit states are linear operators that take any valid quantum state $\rho$ to another valid quantum state $\rho'$; in other words, they must be completely positive and trace preserving (CPTP). Measurements are collections of positive operators $\{M_i\}$ whose sum is the identity, and the probability of outcome $i$ is $\text{tr}(M_i\rho)$.

One commonly used set of unitary operators are the Pauli operators. We denote the Pauli operators using the symbols $\sigma^x$, $\sigma^y$, and $\sigma^z$, where

$$
\sigma^x = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \quad \sigma^y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}, \quad \sigma^z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}.
$$  \tag{1.1}

If we have $n$ qubits, we use $\sigma^x_i$ to mean $\sigma^x$ acting on the $i^{th}$ qubit and the identity acting on all other qubits. That is

$$
\sigma^x_i = \mathbb{I}^{2^{n-1}} \otimes \sigma^x \otimes \mathbb{I}^{2^{n-1}},
$$  \tag{1.2}
where \( \mathbb{I}^d \) is the \( d \)-dimensional identity operator. (If the dimension is clear from context, we will write simply \( \mathbb{I} \).) Likewise for \( \sigma_y^a \) and \( \sigma_z^a \). Generalized Pauli operators on \( n \) qubits are operators of the form \( \{ \mathbb{I}, \sigma^x, \sigma^y, \sigma^z \}^\otimes n \). We label a generic operator of this form by \( \hat{P}_1 \), while \( \hat{P}_0 = \mathbb{I}^n \).

We often refer to the eigenstates of the Pauli operators:

- \(|0\rangle\) and \(|1\rangle\) denote the +1 and −1 eigenstates of \( \sigma^z \).
- \(|+\rangle\) and \(|-\rangle\) denote the +1 and −1 eigenstates of \( \sigma^x \).
- \(|\leftrightarrow\rangle\) and \(|\rightarrow\rangle\) denote the +1 and −1 eigenstates of \( \sigma^y \).

### 1.3.1 Quantum Algorithms

A major goal of quantum computing is to prove that quantum algorithms perform better than classical algorithms at a given task. However, there are several ways in which a quantum algorithm can perform better than a classical algorithm. For example, a quantum algorithm can use less work space than a classical algorithm, like Le Gall’s quantum streaming algorithm, which requires exponentially less space than a classical algorithm [45]. Alternatively, a quantum algorithm can use less time than the best known classical algorithm, like Shor’s factoring algorithm [68]. Another common way to gauge the performance of quantum algorithms is query complexity, which we explain below. In Chapter 2 we will use query complexity as a metric for the performance of our quantum algorithm, while in Chapter 4 we will use time as the metric.

Query complexity can only be used as a metric for algorithms that have oracles. An oracle is a black box unitary that encodes some information \( x \). For simplicity, in this example, we will assume that \( x \) is a string of \( 2^n \) Boolean bits, \( x = \{ x_1, \ldots, x_{2^n} \} \), with \( x_i \in \{0, 1\} \). \( x \) is initially unknown to the algorithm, and the goal of the algorithm is to determine \( f(x) \) for some function \( f \). Let \( \{|i\rangle\} \) be the standard \( 2^n \) orthonormal basis states on \( n \) qubits. Then we are given an oracle \( O_x \) that acts on those \( n \) qubits.
as

\[ O_x|i\rangle = (-1)^{x_i}|i\rangle. \] (1.3)

Thus we can apply \( O_x \) to the \( n \) qubits in order to learn bits of \( x \).

The action of any query algorithm can be viewed as interleaving applications of \( O_x \) with unitaries \( U_t \) that are independent of \( O_x \). If the starting state of the algorithm is \( |\psi_0\rangle \), then we call \( |\psi_t^x\rangle \) the state of the system after \( t \) steps, where

\[ |\psi_t^x\rangle = U_tO_x \ldots U_2O_xU_1O_x|\psi_0\rangle. \] (1.4)

Then the bounded error quantum query complexity of \( f \), which we denote \( Q(f) \), is defined as the minimum number of applications of \( O_x \) needed to evaluate \( f(x) \) with high probability, e.g. with probability \( 2/3 \), as long as \( x \) is taken from some allowed set \( X \). In other words, the bounded error quantum query complexity of \( f \) is the smallest integer \( T \) such that there is some sequence of unitaries \( \{U_1, \ldots, U_T\} \) such that for any \( x \in X \), there is a measurement of the state \( |\psi_T^x\rangle \) that gives the outcome \( f(x) \) with probability \( 2/3 \).

There are several powerful techniques for putting lower bounds on query complexity, such as the adversary method [2, 4, 39, 46] and the polynomial method [10]. These lower bounding techniques allow us to accurately characterize the query complexity of quantum algorithms, and from there, compare quantum query complexity to classical query complexity. However, it is important to keep in mind that while query complexity is an important theoretic tool, for practical purposes, measures like time and space complexity are more significant.
Chapter 2

Quantum Adversary Upper Bound

2.1 Introduction

The general adversary bound has proven to be a powerful concept in quantum computing. Originally formulated as a lower bound on the quantum query complexity of Boolean functions [39], it was proven to be a tight bound both for the query complexity of evaluating discrete finite functions and for the query complexity of the more general problem of state conversion [46]. The general adversary bound is the culmination of a series of adversary methods [2, 4] used to put lower bounds on quantum query complexity. While versions of the adversary method have been useful for improving lower bounds [5, 40, 63], the general adversary bound itself can be difficult to apply, as the quantity for even simple, few-bit functions must usually be calculated numerically [39, 63].

One of the nicest properties of the general adversary bound is that it behaves well under composition [46]. This fact has been used to lower bound the query complexity

---

of evaluating composed total functions, and to create optimal algorithms for composed total functions [63]. Here, we extend one of the composition results to partial Boolean functions, and use it to upper bound the query complexity of Boolean functions, using a tool called the Adversary Upper Bound (AUB).

Usually, finding an upper bound on query complexity involves creating an algorithm. However, using the composition property of the general adversary bound, the AUB produces an upper bound on the query complexity of a Boolean function $f$, given only an algorithm for $f$ composed $k$ times. Because the AUB is nonconstructive, it doesn't tell us what the algorithm for $f$ that actually achieves this promised query complexity might look like. The procedure is a bit counter-intuitive: we obtain information about an algorithm for a simpler function by creating an algorithm for a more complex function. This is similar in spirit to the tensor power trick, where an inequality between two terms is proven by considering tensor powers of those terms.

We describe a class of oracle problems called CONSTANT-FAULT TREES, for which the AUB proves the existence of an $O(1)$-query algorithm. While the AUB does not give an explicit $O(1)$-query algorithm, we show that a span program algorithm achieves this bound. The previous best algorithm for CONSTANT-FAULT TREES has query complexity that is polylogarithmic in the size of the problem.

In Section 2.2 we describe and prove the AUB. In Section 2.3 we look at a simple function called the 1-FAULT NAND TREE, and apply the AUB to prove that this function can be evaluated using $O(1)$ quantum queries. For the remainder of the chapter, we consider more general CONSTANT-FAULT TREES, and go into the application of the AUB in more detail. As described above, to apply the AUB to a function, we need to have an algorithm for that composed function, so in 2.4, we create a quantum algorithm for composed CONSTANT-FAULT TREES. Once we have the algorithm for composed CONSTANT-FAULT TREES, we use the AUB to show that these functions can also be evaluated using $O(1)$ quantum queries. Because the AUB is nonconstructive, we need to separately find an $O(1)$-query algorithm, so in

---

Section 2.5 we describe an optimal algorithm for \textsc{Constant-Fault Trees} that uses span programs.

### 2.2 A Nonconstructive Upper Bound on Query Complexity

The AUB relies on the fact that the general adversary bound behaves well under composition and is a tight lower bound on quantum query complexity. The standard definition of the general adversary bound is not necessary for our purposes, but can be found in [40], and an alternate definition appears in Appendix A.3.

Our procedure applies to Boolean functions. A function \( f \) is Boolean if \( f : S \rightarrow \{0, 1\} \) with \( S \subseteq \{0, 1\}^n \). We say \( f \) is a total function if \( S = \{0, 1\}^n \) and \( f \) is partial otherwise. Given a Boolean function \( f \) and a natural number \( k \), we define \( f^k \), "\( f \) composed \( k \) times," recursively as \( f^k = f \circ (f^{k-1}, \ldots, f^1) \), where \( f^1 = f \).

Now we state the Adversary Upper Bound (AUB):

**Theorem 2.2.1.** Suppose we have a (possibly partial) Boolean function \( f \) that is composed \( k \) times, \( f^k \), and a quantum algorithm for \( f^k \) that requires \( O(J^k) \) queries. Then \( Q(f) = O(J) \), where \( Q(f) \) is the bounded-error quantum query complexity of \( f \).

(For background on bounded-error quantum query complexity and quantum algorithms, see [2].) There are seemingly similar results in the literature; for example, Reichardt proves in [61] that the query complexity of a function composed \( k \) times, when raised to the \( 1/k^{th} \) power, is equal to the adversary bound of the function, in the limit that \( k \) goes to infinity. This result gives insight into the exact query complexity of a function, and its relation to the general adversary bound. In contrast, our result is a tool for upper bounding query complexity, possibly without gaining any knowledge of the exact query complexity of the function.

One might think that the AUB is useless because an algorithm for \( f^k \) usually comes from composing an algorithm for \( f \). If \( J \) is the query complexity of the algorithm for \( f \), one expects the query complexity of the resulting algorithm for \( f^k \) to be at
least $J^k$, as we described in Section 1.2.1. In this case, the AUB gives no new insight. Luckily for us, composed quantum algorithms do not always follow this scaling. If there is a quantum algorithm for $f$ that uses $J$ queries, where $J$ is not optimal (i.e. is larger than the true bounded error quantum query complexity of $f$), then the number of queries used when the algorithm is composed $k$ times can be much less than $J^k$. If this is the case, and if the non-optimal algorithm for $f$ is the best known, the AUB promises the existence of an algorithm for $f$ that uses fewer queries than the best known algorithm, but, as the AUB is nonconstructive, it gives no information as to the form of the algorithm that achieves that query complexity.

We need two lemmas to prove the AUB, both of which are related to the general adversary bound of $f$, which we denote $\text{ADV}^\pm(f)$. $\text{ADV}^\pm(f)$ is a semi-definite program that depends on $f$. We will not define it explicitly here as the definition is notation-heavy and not crucial for understanding the ideas in this chapter. (The definition of the dual of $\text{ADV}^\pm$ is given in Appendix A.3 and the standard definition can be found in [39].) Rather, we will focus on the following two properties of $\text{ADV}^\pm(f)$:

**Lemma 2.2.2.** For any Boolean function $f : S \to \{0, 1\}$ with $S \subseteq \{0, 1\}^n$ and natural number $k$,

$$\text{ADV}^\pm(f^k) \geq (\text{ADV}^\pm(f))^k.$$  

Høyer et al. [39] prove Lemma 2.2.2 for total Boolean functions\(^3\), and the result is extended to more general total functions in [46]. Our contribution is to extend the result in [46] to partial Boolean functions. While the AUB holds for total functions, the nontrivial applications we discuss involve partial functions. The proof of Lemma 2.2.2 closely follows the proof in [46] and can be found in Appendix A.3.

**Lemma 2.2.3.** (Lee, et al. [46]) For any function $f : S \to E$, with $S \subseteq D^n$, and $E, D$ finite sets, the bounded-error quantum query complexity of $f$, $Q(f)$, satisfies

$$Q(f) = \Theta(\text{ADV}^\pm(f)).$$

\(^3\)While the statement of Theorem 11 in [39] seems to apply to partial functions, it is mis-stated; their proof actually assumes total functions.
We now prove Theorem 2.2.1:

**Proof.** Given an algorithm for $f^k$ that requires $O(J^k)$ queries, by Lemma 2.2.3,

$$\text{ADV}^+(f^k) = O(J^k).$$  \hfill (2.3)

Combining Eq. 2.3 and Lemma 2.2.2,

$$(\text{ADV}^+(f))^k = O(J^k).$$  \hfill (2.4)

Raising both sides to the $1/k^{th}$ power,

$$\text{ADV}^+(f) = O(J).$$  \hfill (2.5)

We now have an upper bound on the general adversary bound of $f$. Finally, using Lemma 2.2.3 again, we obtain

$$Q(f) = O(J).$$  \hfill (2.6)

\[ \square \]

### 2.3 Example where the Adversary Upper Bound is Useful

In this section we describe a function, called the 1-FAULT NAND TREE [75], for which the AUB gives a better upper bound on query complexity than any previously known quantum algorithm. To apply the AUB, we need an algorithm for the composed 1-FAULT NAND TREE. In the next section, we describe an algorithm that solves not only the composed 1-FAULT NAND TREE but a broader class of functions called CONSTANT-FAULT TREES. In the current section, we focus on the 1-FAULT NAND TREE as a specific, easy to understand example with potential applications.
The NAND Tree is a complete, binary tree of depth $d$, where each node is assigned a bit value. (For a brief introduction to tree terminology, see Figure 2-1.) The leaves are assigned arbitrary values, and any internal node $v$ is given the value \( \text{nand}(\text{val}(v_1), \text{val}(v_2)) \), where $v_1$ and $v_2$ are $v$'s children, and $\text{val}(v_i)$ is the value of the node $v_i$, and NAND is the 2-bit Boolean function whose output is 0 if and only if both inputs have value 1 (Not-AND).

![Figure 2-1: This is a 2-ary (or “binary”) complete rooted tree of depth 4. The circles are called nodes, and one node is labeled as the root. Nodes that are connected to only one other node are called “leaves.” Each node is connected to other nodes, some of which are further from the root and some of which are closer to the root. For some node $v$, we call the nodes that are connected to $v$ but farther from the root the “children” of $v$. $v_1$ and $v_2$ are children of $v$. This tree is binary because every node except the leaves has two children. (In an $n$-ary tree, each node would have $n$ children.) The depth of a node is its distance from the root. The height of a node is depth of that node subtracted from the maximum depth of the tree. This is a complete tree because all of the leaves have the same depth.](image-url)

To evaluate the NAND Tree, one must find the value of the root given an oracle for the values of the leaves. (The NAND Tree is equivalent to solving $\text{nand}^d$, although the composition we use for the AUB is not the composition of the NAND function, but of the NAND Tree as a whole.) For arbitrary inputs, Farhi et al. showed that there exists an optimal quantum algorithm in the Hamiltonian model to solve the NAND Tree in $O(2^{0.5d})$ time [30], and this was extended to a standard discrete algorithm
with quantum query complexity $O(2^{0.5d})$ [18, 62]. Classically, the best algorithm requires $\Omega(2^{0.75d})$ queries [65]. Here, we consider the 1-Fault NAND Tree, which is a NAND Tree with a promise that the input satisfies certain conditions.

**Definition 2.3.1. (1-Fault NAND Tree [75])** Consider a NAND Tree of depth $d$ (as described above), where the values of all nodes are known. Then to each node $v$, we assign an integer $\kappa(v)$ such that:

- $\kappa(v) = 0$ for leaf nodes.
- Otherwise $v$ has children $v_1$ and $v_2$:
  - If $\text{val}(v_1) = \text{val}(v_2)$, we call $v$ a trivial node, and $\kappa(v) = \max_{i \in \{1,2\}} \kappa(v_i)$,
  - If $\text{val}(v_1) \neq \text{val}(v_2)$, we call $v$ a fault node. Let $v_i$ be the node such that $\text{val}(v_i) = 0$. Then $\kappa(v) = 1 + \kappa(v_i)$.

A tree satisfies the 1-fault condition if $\kappa(v) \leq 1$ for any node $v$ in the tree.

The 1-fault condition is a limit on the amount and location of faults within the tree. Whether a tree is a 1-fault tree or not depends on the values of the leaves, as the values of the leaves determine the values of all other nodes in the tree. In a 1-Fault NAND Tree, if a path moving from a root to a leaf encounters any fault node and then passes through the 0-valued child of the fault node, there can be no further fault nodes on the path. An example of a 1-Fault NAND Tree is given in Figure 2-2.

The condition of the 1-Fault NAND Tree may seem strange, but it has a nice interpretation when considering the correspondence between NAND Trees and game trees⁴. Every NAND Tree corresponds to a two-player game, where the paths from the root to the leaves correspond to the the possible paths of play, and the values of the leaves determine which of the two players win in that particular end-game. If we evaluate such a NAND Tree, where the values of the leaves determine the possible end-games, then the value of the root of the tree tells which player wins if both players

---

play optimally. The NAND TREE is thus a simple example of a problem that can be
generalized to understanding more complex two-player games such as chess or go.

With the interpretation of the NAND TREE as a game tree, the more specific
case of the 1-FAULT NAND TREE corresponds to a game in which, if both players
play optimally, there is at most one point in the sequence of play where a player’s
choice affects the outcome of the game. Furthermore, if a player makes the wrong
choice at the decision point, the game again becomes a single-decision game, where
if both players play optimally for the rest of play, there is at most one point where
a player’s choice affects the outcome of the game. It does not seem farfetched that
such a game might exist with practical applications, for which it would be important
to know which player will win.

Figure 2-2: An example of a 1-FAULT NAND TREE of depth 4. Fault nodes are
highlighted by a double circle. The node \( v \) is a fault since one of its children \( (v_1) \) has
value 0, and one \( (v_2) \) has value 1. Among \( v_1 \) and its children, there are no further
faults, as required by the 1-fault condition. There can be faults among \( v_2 \) and its
children, and indeed, \( v_2 \) is a fault. There can be faults among the 1-valued child of
\( v_2 \) and its children, but there can be no faults below the 0-valued child.

In Section 2.4, we show that there exists an algorithm for the \( d \)-depth 1-FAULT
NAND TREE that requires \( O(d^3) \) queries to an oracle for the leaves. However, when
the \( d \)-depth 1-FAULT NAND TREE is composed \( \log d \) times, the same algorithm re-
quires \( O(d^5) \) queries. Applying the AUB to the 1-FAULT NAND TREE composed
\( \log d \) times, we find:

**Corollary 2.3.1.** The \( d \)-depth 1-FAULT NAND TREE can be evaluated using \( O(1) \)
Proof. Using our algorithm for composed 1-FAULT NAND TREES, we have that

\[ Q(\text{1-FAULT NAND TREE}^{\log d}) = O(d^5). \]  (2.7)

Using Theorem 2.2.1, we have

\[
Q(\text{1-FAULT NAND TREE}) = O\left(d^{\frac{5 \log d}{\log d}}\right) \\
= O\left(c^{\frac{5 \log d}{\log d}}\right) \\
= O(c^5) \\
= O(1) \tag{2.8}
\]

where \(c\) is some constant.

Corollary 2.3.1 shows the power of the AUB. Given a non-optimal algorithm that scales polynomially with the depth of the 1-FAULT NAND TREE, the AUB tells us that in fact the 1-FAULT NAND TREE can be solved using \(O(1)\) queries for any depth.

2.4 Non-Optimal Algorithm for Composed Fault Trees

In this section, we describe a span program-based algorithm for \(k\)-FAULT TREES, of which the 1-FAULT NAND TREE is a specific example. This algorithm is not an optimal algorithm, but will allow us to apply the AUB as in Corollary 2.3.1 to prove the existence of an algorithm with improved performance.

2.4.1 Set-Up

The algorithm for \(k\)-FAULT TREES is based on the span program formulation of [63]. In this subsection, we give several definitions. We define \(k\)-FAULT TREES as well as
span programs and witness size. After laying the groundwork in the current section, in Section 2.4.2 we prove bounds on the performance of the algorithm.

Span programs are linear algebraic ways of representing a Boolean function. Every Boolean function can be represented using a span program, and in fact there are an infinite number of span programs that represent a given function. We consider span programs that have a particularly simple form, and we call them direct span programs. The simplified form means that these direct span programs are not general - i.e. not every Boolean function can be represented using direct span programs. Functions that can be represented by these simplified span program are called direct functions. (For a more general definition of span programs, see Definition 2.1 in [63]).

**Definition 2.4.1.** (Direct Span Program, adapted from Definition 2.1 in [63]) Let a span program $P_f$, for a function $f : S \rightarrow \{0, 1\}$, $S \subset \{0, 1\}^n$, consist of a "target" vector $\tau = (1, 0, \ldots, 0)$ and "input" vectors $\mu_j$ for $j \in \{1, \ldots, n\}$, with $\tau, \mu_j \in \mathbb{C}^N$ for $N \in \mathbb{N}$. We call $x$ the input to $f$, with $x$ composed $n$ input bits $\{x_1, \ldots, x_n\}$. Each $\mu_j$ is associated with a single-bit Boolean function $\chi_j$ that acts on the input bit $x_j$, where $\chi_j(x_j) = x_j$ or $\chi_j(x_j) = \bar{x}_j$ depending on the specific function $f$. The vectors $\mu_j$ satisfy the condition that $f(x) = 1$ if and only if there exists a linear combination of the $\mu_j$'s such that $\sum_j a_j \chi_j(x_j) \mu_j = \tau$, with $a_j \in \mathbb{C}$ for $j \in \{1, \ldots, n\}$. Note that if $\chi_j(x_j) = 0$, $\mu_j$ is effectively not included in the sum. We call $A$ the matrix whose columns are the $\mu_j$'s: $A = (\mu_1, \ldots, \mu_n)$. Any function that can be represented by such a span program is called a direct function.

We use *input bit* to refer to an individual bit input to a function, e.g. $x_j$, while we use *input* to refer to the full set of input bits that are given to a function, e.g. $x$.

Compared to general span programs, we have the condition that each input $x_j$ corresponds to exactly one input vector - this "direct" correspondence gives the span programs their name. As a result, for each direct function there exists two special inputs, $x^*$ and $\bar{x}^*$: for all of the input bits of $x^*$, $\chi_j(x_j) = 1$, and for all the input bits of $\bar{x}^*$, $\chi_j(x_j) = 0$. This means $x^*$ and $\bar{x}^*$ differ at every bit, $f(x^*) = 1$, $f(\bar{x}^*) = 0$, and $f$ is monotonic on every shortest path between $x^*$ and $\bar{x}^*$ (that is, all paths of length 34
While all direct functions are monotonic between two inputs $x^*$ and $\tilde{x}^*$, not all such monotonic functions are direct functions. We prove this by counterexample in Appendix A.1.

One example of direct functions are threshold functions, which determine whether at least $h$ of the $n$ input bits have value 1. Such a threshold function has a span program where $\chi_j(x_j) = x_j$ for $j \in \{1, \ldots, n\}$, and where for any set of $h$ input vectors, but no set of $h - 1$ input vectors, there exists a linear combination that equals $\tau$. It is not hard to show that such vectors exist. For threshold functions, $\tilde{x}^* = (0, \ldots, 0)$ and $x^* = (1, \ldots, 1)$.

Given a span program for a direct function, one can create an algorithm with query complexity that depends on the witness size of the span program [63]:

**Definition 2.4.2.** (Witness Size, based on Definition 3.6 in [63])

Given a direct function $f$, corresponding span program $P_f$, input $x = (x_1, \ldots, x_n)$, $x_j \in \{0, 1\}$, and a vector $s$ of costs, $s \in (\mathbb{R})^n$, $s = (s_1, \ldots, s_n)$, let $r_i$ be the rows of $A$ (where $A$ is the matrix whose columns are the input vectors $\mu_i$ of the span program $P_f$). $r_0$ is the first row of $A$, $r_1$ the second, and so on. Note $r_i \in \mathbb{C}^n$. The functions $\chi_i$ correspond to the input vectors of $P_f$, as in Definition 2.4.1. Then the witness size is defined as follows:

- If $f(x) = 1$, let $w$ be a vector in $\mathbb{C}^n$ with elements $w_j$ such that the inner product $r_0 w = 1$, while $r_i w = 0$ for $i \geq 1$, and $w_j = 0$ if $\chi_j(x_j) = 0$. Then

\[
\text{WSIZE}_a(P_f, x) = \min_w \sum_j s_j |w_j|^2. \tag{2.9}
\]

- If $f(x) = 0$, let $w$ be a vector that is a linear combination of the vectors $r_i$, with the coefficient of $r_0$ equal to 1, and with $w_j = 0$ if $\chi_j(x_j) = 1$. Then

\[
\text{WSIZE}_a(P_f, x) = \min_w \sum_j s_j |w_j|^2. \tag{2.10}
\]

---

5 This definition is equivalent to the definition of witness size given in Definition 3.6 in [63]. The reader can verify that we've replaced the dependence of the witness size on $A$ with a more explicit dependence on the rows and columns of $A$. For the case of $f(x) = 0$ we use what they call $A^\dagger w$ as the witness instead of $w$. 

35
**Notation:** If $s = (1, \ldots, 1)$, we write \( \text{wsize}_1(P_f, x) \).

We discuss how the witness size is related to query complexity below, but first, we introduce the functions we hope to evaluate.

Consider a tree composed of a Boolean function $f$. The $f$-Tree is a complete, $n$-ary tree of depth $d$, where each node is assigned a bit value. The leaves are assigned arbitrary values, and any internal node $v$ is given the value $f(val(v_1), \ldots, val(v_n))$, where $\{v_i\}$ are $v$'s children, and $val(v_i)$ is the value of the node $v_i$. To evaluate the $f$-Tree, one must find the value of the root given an oracle for the values of the leaves.

In particular, we consider $k$-Fault $f$-Trees:

**Definition 2.4.3 ($k$-Fault $f$-Trees).** Consider an $n$-ary $f$-Tree of depth $d$, where throughout the tree nodes have been designated as trivial or fault, and the child nodes of each node have been designated as strong or weak in relation to their parent. All nodes (except leaves) have at least 1 strong child node. For each node $v$ in the tree, let $G_v$ be the set of strong child nodes of $v$. Then to each node $v$, we assign an integer $\kappa(v)$ such that:

- $\kappa(v) = 0$ for leaf nodes.
- $\kappa(v) = \max_{b \in G_v} \kappa(b)$ if $v$ is trivial.
- Otherwise $\kappa(v) = 1 + \max_{b \in G_v} \kappa(b)$.

A tree satisfies the $k$-faults condition if $\kappa(v) \leq k$ for all nodes $v$ in the tree.

In particular, any tree such that any path from the root to a leaf encounters only $k$ fault nodes is a $k$-fault tree. Comparing to Definition 2.3.1, it is clear that Definition 2.4.3 is a generalization of the 1-Fault NAND Tree. From here forward, for simplicity we call $k$-Fault $f$-Trees simply $k$-Fault Trees, but it is assumed that we mean a tree composed of a single function $f$, and where furthermore $f$ is a direct function.

Before our final definitions and lemmas, we need some additional notation. Consider an $f$-Tree of depth $d$ with input $x$ to the leaves of the tree, and consider a
node \( v \). If \( v \) is at height \( \delta \), then the value of \( v \) is \( f^{\delta}(x(v)) \) where \( x(v) \) is the subset of input \( x \) assigned to leaves that are descendants of \( v \). We call \( f_v \) the function that determines value of \( v \), so \( f_v = f^{\delta} \). We also need notation for the values of \( v \)'s children. If \( \{v_1, \ldots, v_n\} \) are \( v \)'s children, then \( x(v_i) \) is the input to the node \( v_i \) - that is, the values of the leaves that are descendants of \( v_i \). Then

\[
f^{\delta}(x(v)) = f(f^{\delta-1}(x(v_1)), \ldots, f^{\delta-1}(x(v_n))),
\]

so \( \text{val}(v_i) = f^{\delta-1}(x(v_i)) \). We denote

\[
\bar{x}(v) = (f^{\delta-1}(x(v_1)), \ldots, f^{\delta-1}(x(v_n)))
\]

\[
= (\text{val}(v_1), \ldots, \text{val}(v_n)).
\]

We can now relate the query complexity of evaluating a tree composed of composed Boolean functions to the witness size of subtrees of the tree:

**Lemma 2.4.1.** ([Section 4.4 from [63]]) Consider an \( f \)-TREE of depth \( d \). Consider a node \( v \) and its corresponding function \( f_v \) and input \( x(v) \). Let \( P_{f_v} \) be a span program corresponding to \( f_v \). Then the subformula complexity \( z_{f_v} \) of the function \( f_v \) with input \( x(v) \), is defined by

\[
z_{f_v}(x(v)) \leq c_1 + \text{WSIZE}_1(P_{f_v}, x(v))(1 + c_2E),
\]

where \( c_1 \) and \( c_2 \) are constants, and \( E > 0 \) is a value to be chosen. For a node \( v \) that is a leaf, \( z_{f_v} = 1 \). Suppose \( E \) has the property that \( E \ll z_{f_v}(x(v)) \) for all nodes \( v \) in the tree and for all possible inputs \( x \) to \( f \). Then

\[
Q(f) = O(1/E).
\]

Finally, we can define trivial, fault, strong, and weak nodes in terms of the witness size and span programs:

37
Definition 2.4.4. (Trivial and Fault) Given an \( f\)-TREE with input \( x \), let \( v \) be a node at height \( \delta \). Then \( v \) is trivial if \( \text{WSIZE}_1(P_f, \tilde{x}(v)) = 1 \). \( v \) is a fault if \( \text{WSIZE}(P_f, \tilde{x}(v)) > 1 \).

Definition 2.4.5. (Strong and Weak) For a direct function \( f \) with input \( x = (x_1, \ldots, x_n) \), and input functions \( \{\chi_j\} \), the \( j \)-th input bit is strong on input \( x \) if \( f(x) = \chi_j(x_j) \), and weak otherwise. When we consider an \( f\)-TREE, if a node \( v \) has children with values \( \tilde{x}(v) \), then we say the child node \( v_j \) is strong if \( f(\tilde{x}(v)) = \chi_j(x_j) \) and weak otherwise.

2.4.2 Span Program Algorithm

In this section, we create a span program-based quantum algorithm for a \( k \)-FAULT TREE of depth \( d \) that requires \( O(d^3 \omega^k) \) queries, where \( \omega = \max_x \text{WSIZE}_1(P_f, \tilde{x}) \), for \( f \) a direct function.

We need two lemmas. First, a result about weak inputs:

**Lemma 2.4.2.** The costs \( s_j \) corresponding to weak inputs do not affect the witness size.

**Proof.** From the definition of witness size (Definition 2.4.2), for \( f(x) \) equals 0 or 1, we require \( w_j = 0 \) for all \( j \) where \( \chi_j(x_j) \neq f(x) \). This means \( s_j \) is multiplied by 0 for all such \( j \), and therefore does not affect the witness size. \( \square \)

Second, a result about subformula complexity for composed functions

**Lemma 2.4.3.** Let \( f : S \to \{0, 1\} \), \( S \subseteq \{0, 1\}^n \) and \( g : C \to \{0, 1\} \), \( C \subseteq \{0, 1\}^m \), and consider the composed function \((f \circ g)(x)\) with \( x = (x^1, \ldots, x^n), x^i \in C \) and \( g(x^i) \in S \forall i \). Let \( \tilde{x} = (g(x^1), \ldots, g(x^n)) \in S \), so \((f \circ g)(x) = f(\tilde{x})\). Let \( P_g \) be a span program for \( g \), and as usual, \( P_f \) be a span program for \( f \). Let \( z_j \) be the subformula complexity for the span program \( P_g \) on input \( x^j \), and \( Z = (z_1, \ldots, z_c) \). Then the subformula complexity of \( f \circ g \) on input \( x \) is

\[
z_{f \circ g}(x) \leq c_1 + \text{WSIZE}_Z(P_f, \tilde{x})(1 + c_2 E \max_j z_j)
\]

\[
\leq c_1 + \text{WSIZE}_1(P_f, \tilde{x}) \max_j z_j (1 + c_2 E \max_j z_j).
\]

(2.15)
Applying Lemma 2.4.2, we can rewrite Eq. 2.15 as

\[ z_f(x) \leq c_1 + \text{WSIZE}_1(P_f, \tilde{x}) \max_{j \text{ strong}} j z_j(1 + c_2 E \max z_j). \] (2.16)

Now we can prove the non-optimal algorithm for \( k \)-FAULT TREES:

**Theorem 2.4.4.** Given a \( k \)-FAULT TREE of depth \( d \) with nodes evaluating the direct function \( f \), we can create an algorithm based on a span program \( P_f \) for \( f \) that evaluates the root of the tree with \( O(d^3 \omega^k) \) queries, where \( \omega = \max_x \text{WSIZE}_1(P_f, x) \).

**Proof.** To first approximation, the subformula complexity of a composed function is the product of the witness sizes of the constituent functions. This can be seen from the second line of Eq. 2.15, if ignoring the constants \( c_1 \) and \( c_2 \). So to get a rough estimate of the subformula complexity, (and hence via Lemma 2.4.1 of the query complexity), we can look at a path from the root to a leaf, and take the product \( \Pi_{v \text{ path}} \text{WSIZE}(P_f, \tilde{x}(v \text{ path})) \) where \( v \text{ path} \) are all of the nodes encountered on the path. Taking the maximum over all paths gives a good estimate of subformula complexity. The condition on the maximum number of faults along each path then gives a bound on the complexity throughout the tree, corresponding to the factor of \( \omega^k \) in the query complexity. It remains to take into consideration corrections coming from \( c_1 \) and \( c_2 \) in Eq. 2.15. It turns out that these two factors contribute the factor of \( d^3 \) to the query complexity.

We prove that when \( E = cd^{-3} \omega^{-k} \),

\[ \max_{x \in S_v} z_{f_v}(x(v)) < c' \delta \omega^{\kappa_v} \left(1 + \frac{c_2 c c'}{d^2}\right)^\delta \] (2.17)

where \( v \) is node at height \( \delta \), and \( S_v \) is the set of possible inputs to \( f_v \) such that \( \kappa(v) = \kappa_v \) (see Definition 2.3.1 for an explanation of \( \kappa \)). \( c' \) is a constant larger than 1, dependent on \( c_1 \), and \( c \) is chosen such that \( c_2 c c' \ll 1 \) and \( c' c \ll 1 \). Since \( E = cd^{-3} \omega^{-k} \), and for all nodes in the tree \( \delta \leq d \) and \( \kappa_v \leq k \), we have from Eq. 2.17 that

\[ z_{f_v}(x(v)) < c' \delta \omega^{\kappa_v} \left(1 + \frac{c_2 c c'}{d^2}\right)^\delta \ll cd^{-3} \omega^{-k} = E. \] (2.18)
By Lemma 2.4.1, this implies the query complexity of the algorithm is \( Q(k\text{-FAULT TREE}) = O(cd^{-3}w^{-k}) \) as desired.

We prove Eq 2.17 by induction on the height of the tree. For leaves, \( z = 1 < c' \), so Eq. 2.17 holds in the base case.

For the inductive case, consider a node \( v \) with \( \kappa = \eta \) and height \( \delta \). \( v \) has \( n \) child nodes, \((v_1, \ldots, v_n)\). All of \( v \)'s children have height \( \delta - 1 \), so we assume that the inductive assumption Eq. 2.17 holds for the subformula complexities of those nodes.

We consider the trivial and fault cases separately. We first consider the case that \( v \) is trivial. This means

\[
\text{wsizer}_1(P_f, x(v)) = 1, \quad (2.19)
\]

Because of the \( k \)-faults condition, for each strong input node to \( v_i \) we have \( \kappa(v_i) \leq \eta \), and so applying our inductive assumption, we get

\[
z_{f_{v_i}}(x(v_i)) < c'(\delta - 1)\omega^\eta \left( 1 + \frac{c_2 c'}{d^2} \right)^{\delta - 1}. \quad (2.20)
\]

On the other hand weak input nodes \( v_j \) only have the condition that \( \kappa(v_j) \leq k \). Applying our inductive assumption again, we obtain for weak input nodes

\[
z_{f_{v_j}}(x(v_j)) < c'(\delta - 1)\omega^k \left( 1 + \frac{c_2 c'}{d^2} \right)^{\delta - 1}. \quad (2.21)
\]

Plugging Eqs. 2.19-2.21 into Eq. 2.16, we have

\[
z_{f_v}(x) \leq c_1 + c'(\delta - 1)\omega^\eta \left( 1 + \frac{c_2 c'}{d^2} \right)^{\delta - 1} \left( 1 + c_2 E c'(\delta - 1)\omega^k \left( 1 + \frac{c_2 c'}{d^2} \right)^{\delta - 1} \right) \\
= c_1 + c'(\delta - 1)\omega^\eta \left( 1 + \frac{c_2 c'}{d^2} \right)^{\delta - 1} \left( 1 + \frac{c_2 c'(\delta - 1)}{d^3} \left( 1 + \frac{c_2 c'}{d^2} \right)^{\delta - 1} \right) \quad (2.22)
\]

where the second line is obtained using \( E = cd^{-3}w^{-k} \). Keeping only terms of order
\[ c_2 cc'/d^2 \] in the second term of the second line, we have

\[ z_{f_v}(x) \leq c_1 + c'(\delta - 1)\omega^\eta \left( 1 + \frac{c_2 cc'}{d^2} \right)^{\delta - 1} \left( 1 + \frac{c_2 cc'(\delta - 1)}{d^3} \right) \]

\[ \leq c_1 + c'(\delta - 1)\omega^\eta \left( 1 + \frac{c_2 cc'}{d^2} \right)^\delta \]

\[ \leq c'\delta\omega^\eta \left( 1 + \frac{c_2 cc'}{d^2} \right)^\delta \quad (2.23) \]

where we subsumed the \( c_1 \) term into the \( c' \) term by choosing \( c' \gg c_1 \) and using that \( \omega^\eta \left( 1 + \frac{c_2 cc'}{d^2} \right)^\delta > 1 \). Thus we see that in the trivial case, the inductive assumption holds.

For the case when \( v \) is a fault, we now have the condition that for each strong input node to \( v_i \), \( \kappa(v_i) \leq \eta - 1 \), and so applying our inductive assumption, we get

\[ z_{f_v}(x(v_i)) < c'(\delta - 1)\omega^{\eta - 1} \left( 1 + \frac{c_2 cc'}{d^2} \right)^{\delta - 1}. \quad (2.24) \]

However, we still have that for hand weak input nodes \( v_j \) the condition is that \( \kappa(v_j) \leq k \), so these nodes satisfy Eq. 2.21. From here, the analysis follows Eqs. 2.22-2.23 very closely, with the same result, proving Eq. 2.17.

We call \( k \)-FAULT TREES with \( k \) a constant CONSTANT-FAULT TREES. Then we have the following:

**Corollary 2.4.5.** The \( d \)-depth CONSTANT-FAULT TREE composed of a direct function \( f \) can be evaluated using \( O(1) \) queries.

**Proof.** Consider composing a \( b \)-FAULT TREE of depth \( d \), where \( b \) is a constant, with itself \( k \) times. This produces a \((b \times k)\)-FAULT TREE of depth \( d \times k \). Choosing \( k = \log d \) we create a \((b \log d)\)-FAULT TREE of depth \( d \log d \). By Theorem 2.4.4 the
query complexity of evaluating such a tree is

\[
Q((b\text{-FAULT TREE})^{\log d}) \leq Q((b \log d)\text{-FAULT TREE}) = O(w^{b \log d}(d \log d)^3) \\
= O(d(d \log d)^3) \\
= O(d^5). \quad (2.25)
\]

Now we can apply Theorem 2.2.1 to get

\[
Q(b\text{-FAULT TREE}) = O\left((d^5)^{\frac{1}{\log 2}}\right) = O(1). \quad (2.26)
\]

\[\square\]

2.4.3 Direct Functions

The reader might have noticed that our quantum algorithm does not seem to depend explicitly on the function being a direct function. In fact, the algorithm may hold for non-direct functions. However, the analysis becomes more complex in this case, and potentially breaks down. For example, for general Boolean functions, whether an input bit is strong or weak depends not only on the value of that input bit, but on the values of the other input bits as well. The subformula complexity also depends on multiple input bits, so as a result there is no easy way to factorize out the contributions of strong and weak inputs as in Eq. 2.16.

Direct functions have another benefit: one can create \(k\)-FAULT TREES for any number \(1 \leq k \leq d\) where \(d\) is the depth of the tree. This is because direct functions are always trivial on at least two inputs, \(x\) and \(y\), such that \(f(x) = 0\) and \(f(y) = 1\). To see why having two trivial inputs is important, consider a NAND TREE where only the input \((0, 0)\) to NAND is trivial. NAND\((0, 0) = 1\) so in a NAND TREE all trivial nodes must have value 1. Suppose you have a node \(v\) that is trivial. The children of \(v\) can not be trivial because they must have value 0, and 0-valued nodes can not be trivial. Therefore any path from the root to a leaf must contain at least \(\lfloor d/2 \rfloor\) faults, since only every other node on the path can be trivial. As a result, one can only have
In Appendix A.2 we prove that direct functions can be made trivial on inputs $x^*$ and $\tilde{x}^*$. By definition $f(x^*) = 1$ and $f(\tilde{x}^*) = 0$, so if these inputs are trivial it allows for the creation of trees with arbitrary numbers of faults. It is easy to show that NAND is a direct function, with $\tilde{x}^* = (1, 1)$ and $x^* = (0, 0)$. Consequently, these inputs are trivial, as was claimed in Definition 2.3.1.

### 2.5 Optimal Span Program Algorithm for Constant-Fault Trees

While we use the AUB to prove the existence of $O(1)$-query quantum algorithms for CONSTANT-FAULT TREES, we don't yet have an algorithm that requires $O(1)$ queries. The algorithm of Section 2.4, which we used to prove Corollary 2.4.5, is far from optimal; for CONSTANT-FAULT TREES of depth $d$ it uses $O(d^3)$ queries. In this section, we describe an asymptotically optimal algorithm that uses $O(1)$ queries for CONSTANT-FAULT TREES. The algorithm combines properties of the witness size of direct functions with a different version of span program algorithms.

There have been many iterations of span program-based quantum algorithms, due to Reichardt and others [46, 61, 63]. In the previous section, we described an algorithm for FAULT TREES that uses the span program formulation described in Definition 2.1 in [61], one of the earliest versions. Using more recent advancements in span program technology, we show in this section that:

**Theorem 2.5.1.** Given a $k$-FAULT TREE composed of the direct function $f$, there is a quantum algorithm that evaluates the tree using $O(w^k)$ queries, where $\omega = \max_x \text{FSIZE}_1(P_f, x)$. In particular, for a CONSTANT-FAULT TREE, the algorithm requires $O(1)$ queries.

It makes sense that the optimal algorithm uses span programs, not just because span programs can always be used to create optimal algorithms [46], but because the AUB
is based on properties of the general adversary bound, and there is strong duality between the general adversary bound and span programs.

The algorithm we describe in this section solves a slightly more general $k$-fault tree than that described in Definition 2.3.1. There, we had the condition that $\kappa(v) \leq k$ for all nodes $v$ in the whole tree. For the algorithm we describe in this section, we only need $\kappa(v) \leq k$ for $v$ the root node.

There are several new properties of span program algorithms that we take advantage of in order to prove the result of this section. First for a function $f : S \rightarrow \{0, 1\}$, with $S \subseteq \{0, 1\}^n$, input $x \in S$, and span program $P_f$, we have

$$Q(f) = O \left( \max_{x \in S} \text{WSIZE}_1(P_f, x) \right).$$

(2.27)

This is an simplification over the complicated relationship between query complexity and witness size given in Lemma 2.4.1.

The second property of span programs we use is a result regarding the witness size of composed functions. This composition lemma is simpler and tighter than Lemma 2.4.3:

**Lemma 2.5.2.** (based on Theorem 4.3 in [61]) Let $f : S \rightarrow \{0, 1\}$, $S \subseteq \{0, 1\}^n$ and $g : C \rightarrow \{0, 1\}$, $C \subseteq \{0, 1\}^m$, and consider the composed function $(f \circ g)(x)$ with $x = (x^1, \ldots, x^n)$, $x^i \in C$ and $g(x^i) \in S \forall i$. Let $\tilde{x} = (g(x^1), \ldots, g(x^n)) \in S$. Let $P_g$ be a span program for $g$, $P_f$ be a span program for $f$, and $s \in (\mathbb{R}^+)^{n \times m}$. Then there exists a span program $P_{f \circ g}$ for $f \circ g$ such that

$$\text{WSIZE}_s(P_{f \circ g}, x) \leq \text{WSIZE}_t(P_f, \tilde{x}) \leq \text{WSIZE}_1(P_f, \tilde{x}) \max_{i \in [n]} \text{WSIZE}_{s_i}(P_g, x^i)$$

(2.28)

where $t = (\text{WSIZE}_{s_1}(P_g, x^1), \ldots, \text{WSIZE}_{s_n}(P_g, x^n))$ and $s_i$ is a vector of the $i^{th}$ set of $m$ elements of $s$.

The main difference between this lemma and that in [61] is that the witness size here is input dependent, as is needed for partial functions. We also use a single inner function $g$ instead of $n$ different functions $g_i$, but we allow each inner function to have
a different input. The proof of this result follows exactly using the proof of Theorem 4.3 in [61], so we will not repeat it here.

We previously showed that for any direct function \( f \), one can create a span program \( P_f \) with the following properties\(^6\):

- \( \text{WSIZE}_1(P_f, x) = 1 \) if input \( x \) makes the function trivial.
- \( \text{WSIZE}_1(P_f, x) \leq w \), if input \( x \) makes the function a fault, where \( w \) is a constant depending only on \( f \).
- For \( \text{WSIZE}_s(P_f, x) \), \( s_i \) do not affect the witness size, where the \( i^{th} \) input bit is weak.

Using the properties of strong and weak nodes in direct functions, we see that \( \text{WSIZE}_s(P_f, \tilde{x}) \) in Eq. 2.28 doesn't depend on \( t_i = \text{WSIZE}_s(P_g, x^i) \) for weak inputs \( i \). Thus, we can rewrite Eq. 2.28 as

\[
\text{WSIZE}_s(P_f, x) \leq \text{WSIZE}_1(P_f, \tilde{x}) \max_{i \in [n]} \text{WSIZE}_s(P_g, x^i). \quad (2.29)
\]

We now prove Theorem 2.5.1:

**Proof.** For a direct function \( f \), we know there is a span program \( P \) such that \( \text{WSIZE}_1(P_f, x) \leq w \) for all fault inputs and \( \text{WSIZE}_1(P_f, x) = 1 \) for trivial inputs. We show that this implies the existence of a span program for the \( k \)-FAULT TREE with witness size \( \leq w^k \).

We use an inductive proof on \( d \), the depth of the tree. For the base case, consider a depth-1 tree. This is just a single direct function. If it is a 0-FAULT TREE, then the input is promised to make the function trivial, and so for any allowed input \( x \), \( \text{WSIZE}_1(P_f, x) = 1 \), and so \( Q(f) = 1 \). If it is a 1-FAULT TREE, then the input might

---

\(^6\)We note that in 2.4, we use a different version of span programs than those used to prove Eq. 2.27. However Reichardt shows in [61] how to transform from one span program formulation to another, and proves that there is a transformation from the span programs of the previous section to the ones needed for Lemma 2.5.2 that does not increase the witness size and that uses the weighting vector in the same way.
make the function a fault, but by the definition of $\omega$, for any input $x$, $\text{WSIZE}_1(P_f, x) \leq \omega$, and so $Q(f) = \omega$. Thus for the base case, depth-1 tree, we have $Q(f) = \omega^k$.

Consider a depth $d$, $k$-FAULT TREE with input $x$. Consider the root node $v_0$ with children $\{v_1, \ldots, v_n\}$. (In the following we use the notation introduced before Lemma 2.4.1.) If $\bar{x}(v_0)$ makes $f$ a fault, then by Eq. 2.29 we know there exists a span program $P_{j_{v_0}}$ for $f_{v_0}$ such that

$$\text{WSIZE}_1(P_{j_{v_0}}, x) \leq \text{WSIZE}_1(P_f, \bar{x}) \times \max_{i \in [n]} \text{WSIZE}_1(P_{v_i}, x(v_i))$$

$$\leq \omega \times \max_{i \in [n]} \text{WSIZE}_1(P_{v_i}, x(v_i)). \quad (2.30)$$

Let $j$ be the index of the child that maximizes the second line. By the definition of $k$-FAULT TREES, $f_{v_j}$ is a $(\kappa-1)$-FAULT TREE of depth $d-1$ for some $\kappa \leq k$. By inductive assumption, there is a span program for $f_{v_j}$ satisfying $\text{WSIZE}_1(P_{v_j}, x(v_j)) \leq \omega^{k-1}$, so $f_{v_0}$ satisfies

$$\text{WSIZE}_1(P_{j_{v_0}}, x) \leq \omega^k, \quad (2.31)$$

and so by Eq. 2.27 there is a quantum algorithm for the tree that uses $O(\omega^k)$ queries.

Given the same setup, but now assuming the input $\bar{x}$ makes $f$ trivial, then by Eq. 2.29 we have:

$$\text{WSIZE}_1(P_{j_{v_0}}, x) \leq \text{WSIZE}_1(P_f, \bar{x}) \times \max_{i \in [n]} \text{WSIZE}_1(P_{v_i}, x(v_i))$$

$$= 1 \times \max_{i \in [n]} \text{WSIZE}_1(P_{v_i}, x(v_i)). \quad (2.32)$$

Let $j$ be the index of the child that maximizes the second line. By the definition of FAULT TREES, $f_{v_j}$ is a $\kappa$-FAULT TREE with $\kappa \leq k$ of depth $d-1$. From our inductive assumption $\text{WSIZE}_1(P_{v_j}, x(v_j)) \leq \omega^{k-1}$, so again, we have

$$\text{WSIZE}_1(P_{j_{v_0}}, x) \leq \omega^k, \quad (2.33)$$

46
and again using Eq. 2.27 there is a quantum algorithm for the tree that uses $O(\omega^k)$ queries.

Zhan et al. show that a classical algorithm needs $\Omega(\log d)$ queries to evaluate a depth-$d$ CONSTANT-FAULT TREE [75]. This gives us our quantum speed-up, since by Theorem 2.5.1, there exists a quantum algorithm that requires constant queries for any depth. However, the key point of this result is not the speed-up, but the application of the AUB.

\section*{2.6 Conclusions}

We describe a method for upper bounding the quantum query complexity of Boolean functions using the Adversary Upper Bound (AUB). Using this method, we show that CONSTANT-FAULT TREES can always be evaluated using $O(1)$ queries. Furthermore, we create an algorithm with a matching upper bound using span programs.

We would like to find other examples where the AUB is useful, although we suspect that CONSTANT-FAULT TREES are a somewhat unique case. However, there could be other types of quantum walk algorithms, for example, to which the AUB might be applied. In any case, this work suggests that new ways of upper bounding the general adversary bound could give us a second window into quantum query complexity beyond algorithms.

It would also be interesting to determine whether there are algorithms for CONSTANT-FAULT TREES that are time-efficient as well as query-efficient. If there is a time-efficient algorithm, and if one could efficiently determine whether an input satisfies the $k$-faults condition, then these algorithms would potentially give a speed-up for real-world problems.

The existence of the AUB tells us something general about the structure of quantum algorithms. There is a natural way that quantum algorithms should compose, and if an algorithm does not compose in this natural way, then something is non-optimal.
2.7 Acknowledgements

Many thanks to Rajat Mittal for generously explaining the details of the composition theorem for the general adversary bound. Thanks to the anonymous FOCS reviewer for pointing out problems with a previous version, and also for encouraging me to find a constant query span program algorithm. Thanks to Bohua Zhan, Avinatan Hassidim, Eddie Farhi, Andy Lutomirski, Paul Hess, and Scott Aaronson for helpful discussions.

2.8 Contributions

All work in this chapter and in Appendix A is entirely my own except Section 2.4 and Appendix A.2. Those sections were done in collaboration with Bohua Zhan and Avinatan Hassidim. I contributed the idea for the direct functions extension and worked out earlier versions of some of the proofs. The final versions of most of the proof ideas are due to Bohua Zhan, although I have done extensive reworkings for clarity in this thesis.
Chapter 3

Robust Randomized Benchmarking

3.1 Introduction

While quantum process tomography [22] is a conceptually simple approach to the characterization of quantum operations on states, its implementation suffers from a number of fundamental drawbacks. These obstacles range from its exponential scaling with the size of the system, to its dependence on precise knowledge of state preparation and measurement. Precise knowledge about state preparation requires precise knowledge about operations and measurements, leading to a difficult non-linear estimation problem [71, 72, 70, 51]. Lack of precise knowledge about state preparation and measurement can also lead to significant systematic errors in the reconstructed operations [74]. Recently, randomized benchmarking (RB) protocols have been shown to lead to estimates of the average fidelity to Clifford group operations in a manner that is robust against imprecise knowledge about state preparation and measurement, and therefore largely free of some of the systematic errors that can affect standard tomographic reconstructions [28, 43, 48, 49, 31, 50].

We describe a procedure that provides an almost complete description of any quantum map in a way that is robust against many errors that plague standard tomographic procedures. Specifically, we can estimate the unital part [36, 11] of any trace-preserving map, which includes all parameters necessary to describe deterministic as well as random unitary evolution. Furthermore, we show that a related protocol can be used to efficiently estimate the average fidelity to unitary operations outside the Clifford group, again in a way that is accurate even in the presence of state preparation, measurement, and unitary control errors.

Both procedures use RB protocols as a tool, combined with several new results: we show that Clifford group maps span the unital subspace of quantum maps, and that important unitaries outside the Clifford group can be expressed as linear combinations of few Clifford group maps. These insights, combined with new error strategies and analysis, allow us to robustly characterize maps that were previously inaccessible.

Our error analysis rigorously proves that randomized benchmarking decays can be fit efficiently. We also prove new results on the average fidelity of composed maps, which is important for RB, but is also of significance to any procedure where direct access to a quantum map is limited.

This paper is organized as follows. In Section 3.2 we give background on general properties of quantum operations. In Section 3.3 we sketch the RB protocol and describe the information that can be extracted from such experiments. In Section 3.4 we describe how the information from RB experiments can be used to tomographically reconstruct the unital part of any experimental map even in the presence of imperfect randomizing operations. In Section 3.5, we show that it is possible to efficiently bound the fidelity of any such experimental map to a class of unitaries capable of universal quantum computation. In Section 3.6, we analyze error propagation in these protocols. This section includes new bounds on the effect of imperfect randomizing operations, and rigorous bounds on the number of samples needed to achieve some desired error and confidence. Finally, in Section 3.7, we discuss an experimental implementation of these robust protocols and compare the results to standard tomographic procedures.
3.2 Completely Positive Trace Preserving Maps: Notation and Properties

Throughout this paper, we will restrict the discussion to Hermiticity-preserving linear operations on quantum states—more specifically, linear operations on multiqubit states, so that the Hilbert space dimension will always be \( d = 2^n \) for \( n \) qubits. The physical operations within this class that are commonly considered are completely-positive (CP) trace-preserving (TP) operations [41, 20, 44]. We refer to these operations on quantum systems as maps, and will denote them by the calligraphic fonts \( \mathcal{A}, \mathcal{B}, \text{etc.} \) The composition of two maps will be denoted \( \mathcal{A} \circ \mathcal{B} \), meaning \( \mathcal{B} \) acts on a state first, and then \( \mathcal{A} \) acts. Even when discussing unitary evolution, we will refer to the corresponding maps. The notable exceptions are the identity unitary \( \mathds{1} \), and the unitaries in the multi-qubit Pauli group \( \mathcal{P} \), which will be denoted \( \hat{P}_i \)—although the corresponding maps \( \mathcal{I} \) and \( \mathcal{P}_i \) will also be used in some contexts. We will use the standard convention where \( \mathcal{A} \circ \mathcal{B} = \mathcal{I} \). We use \( \mathcal{T} \) to mean the map corresponding to the unitary \( e^{-i\mathbf{\sigma}^*} \).

A map \( \mathcal{E} \) is TP iff \( \text{tr} \mathcal{E}(\rho) = \text{tr} \mathcal{E}(\hat{\rho}) \) for all \( \hat{\rho} \), which in turn leads to the requirement that \( \mathcal{E}^\dagger(\mathds{1}) = \mathds{1} \), where \( \mathcal{E}^\dagger \) is the Heisenberg picture representation of \( \mathcal{E} \). Any linear map \( \mathcal{E} \) can be written as

\[
\mathcal{E}(\rho) = \sum_{i,j=0}^{d^2-1} \chi_{ij}^\mathcal{E} \hat{P}_i \rho \hat{P}_j, \tag{3.1}
\]

which is known as the \( \chi \) matrix representation of \( \mathcal{E} \). The map \( \mathcal{E} \) is CP iff \( \chi^\mathcal{E} \) is positive semidefinite, and the TP condition \( \mathcal{E}^\dagger(\mathds{1}) = \mathds{1} \) translates to \( \sum_{i,j} \chi_{ij}^\mathcal{E} \hat{P}_i \hat{P}_j = \mathds{1} \), which implies \( \text{tr} \chi^\mathcal{E} = 1 \) [22]. A map \( \mathcal{E} \) is unital if \( \mathcal{E}(\mathds{1}) = \mathds{1} \).

It is often necessary to compute the representation of the composition of two maps. While such a calculation can be cumbersome in the \( \chi \) representation, Liouville representations are more convenient for describing the action of composed maps on quantum states [13]. In the Liouville representation, an operator \( \hat{\rho} \) is represented by a column vector \( |\hat{\rho}\rangle \), and maps are represented by matrices acting on these vec-
tors, such that the composition of maps corresponds to matrix multiplication. The most convenient choice of basis for these vectors and matrices depends on the application, but for our purposes we will use the basis of Pauli operators, and will call this the Pauli-Liouville representation (which appears to have no standard name in the literature, despite being widely used [47, 42, 35, 60, 69, 21]). For a map \( E \), the Pauli-Liouville representation is given by

\[
E^{(PL)} = \sum_{i,j=0}^{d^2-1} \frac{\text{tr}[\mathcal{E}(\hat{P}_i)\hat{P}_j]}{d}|i\rangle\langle j|,
\]

(3.2)

where \( \hat{P}_i \) and \( \hat{P}_j \) are \( n \)-qubit Pauli operators. Hermiticity preservation implies that all matrix elements of \( E^{(PL)} \) are real. The \( k \)th entry in the vector \( |\hat{\rho}\rangle \) representing a density matrix \( \hat{\rho} \) corresponds to \( \text{tr}\hat{\rho}\hat{P}_k \). This ensures that the Pauli-Liouville representation of any CPTP map can be written as [47, 42]

\[
E^{(PL)}(1_1^T, E_{(U^t)}(\psi)) = \frac{1}{d} \begin{pmatrix} \tau_\mathcal{E}^T & 0 \end{pmatrix},
\]

(3.3)

where \( \tau_\mathcal{E} \) is a \( d^2 - 1 \) dimensional column vector, \( \mathcal{E} \) is a \((d^2 - 1) \times (d^2 - 1)\) matrix.

We will quantify how distinct a map \( \mathcal{E} \) is from a particular unitary map \( U \) by the average fidelity \( \overline{F}(\mathcal{E}, U) \), which can be written as

\[
\overline{F}(\mathcal{E}, U) = \int d\mu(\psi) \langle \psi | (U^t \circ \mathcal{E}(|\psi\rangle\langle\psi|)) | \psi \rangle,
\]

(3.4)

with integration taken over the unitarily invariant Fubini-Study measure [11]. This definition also implies \( \overline{F}(\mathcal{E}, U) = \overline{F}(\mathcal{E} \circ U^t, I) = \overline{F}(U^t \circ \mathcal{E}, I) \). The average fidelity is closely related to the trace overlap between \( \mathcal{E}^{(PL)} \) and \( U^{(PL)} \), as well as to \( \chi_0^{E \circ U^t} \), by
For simplicity and clarity, here, and throughout the paper, we omit the superscripts from the Pauli-Liouville representation of superoperators whenever they occur within trace expressions, as these expressions already include superscripts indicating Hermitian conjugates.

3.3 Randomized Benchmarking of Clifford Group Maps

Randomized benchmarking (RB) [28, 43, 48, 49, 31, 50] consists of a family of protocols to robustly estimate the average fidelity $\overline{F}(E, U)$ between an experimental quantum map $E$ and an ideal unitary map $U$. In this context, robustness refers to the ability to estimate $\overline{F}(E, U)$ in a manner that is insensitive to imprecise or even biased knowledge about state preparation, measurement, and controlled unitary evolution. Such imperfections can lead to systematic errors, e.g., in fidelity estimates based on standard tomographic reconstruction protocols [51].

We now describe a framework that can be used to understand existing RB protocols, but which allows us to highlight how our protocol differs from previous procedures. RB protocols consist of $k$ repeated applications of $E$, each time preceded by independently chosen randomizing unitary maps $D_i$ where $1 \leq i \leq k$, and, after the last application of $E$, followed by a recovery map $D_{k+1}$. The randomizing unitaries are chosen such that, if (i) the sequence is applied to a fixed initial state $|\psi\rangle$, (ii) $E$ is identical to a certain unitary map $U$, and (iii) the randomizing maps $D_i$ are perfect, then the final state would be identical to the initial state. If the first $k$ randomizing operations are chosen from the Haar measure over unitary maps [28, 11] or from a set
with the same first- and second-order moments as the Haar measure \[23\], the fidelity between the initial and final states can be shown to decay exponentially with \( k \) at a rate that depends only on \( F(\mathcal{E},\mathcal{U}) \) \[28, 48, 49\]. The RB literature typically assumes either (1) \( \mathcal{U} = \mathcal{I} \) and \( \mathcal{E} \) represents the errors from the randomizing operations, or (2) \( \mathcal{U} \) is some other unitary map, and \( \mathcal{E} \) is its potentially faulty implementation. However, we emphasize our description is more general, and as we will demonstrate later, allows us to reconstruct a major portion of arbitrary \( \mathcal{E} \), not just implementations of the randomizing operations.

In a realistic setting one cannot assume that the initial state is pure and exactly known, that one knows what observable is measured exactly, or that the randomizing operations are applied noiselessly. However, these assumptions are not necessary for the RB protocol to work: the initial state can be any mixed state \( \hat{\rho}_0 \neq \frac{1}{d} \mathbb{I} \), the measured observable \( \hat{M} \) can be any observable where \( \text{tr} \hat{\rho}_0 \hat{M} \neq \frac{1}{d} \text{tr} \hat{M} \), and the rate of decay \( p \) of the measured expectation value is still related to \( F(\mathcal{E},\mathcal{U}) \) in the same way. The randomizing operations need not be noiseless either \[48, 49\], as long as the imperfect randomizing operations correspond to \( \mathcal{N} \circ \mathcal{D}_i \), with \( \mathcal{N} \) representing some arbitrary CPTP error map (some of these restrictions may be relaxed, leading to more complex decays \[48, 49\], and although our protocols generalize straightforwardly to such scenarios we do not discuss them here for the sake of brevity). Under these more realistic assumptions, \( F_k(\mathcal{E} \circ \mathcal{N},\mathcal{U}) \), the average of \( \langle \hat{M} \rangle \) over the choice of randomizing operations, for sequences of length \( k \), is given by

\[
F_k(\mathcal{E} \circ \mathcal{N},\mathcal{U}) = A_0 p^k + B_0,
\]

where \( A_0 \) and \( B_0 \) are constants that contain partial information about the preparation and measurement (including imperfections), and

\[
p = \frac{d \ F(\mathcal{E} \circ \mathcal{N},\mathcal{U}) - 1}{d - 1},
\]

\[
= \frac{\text{tr} \mathcal{U}^\dagger \mathcal{E} \mathcal{N} - 1}{d^2 - 1}.
\]

54
By estimating $F_k(\mathcal{E} \circ \mathcal{N}, \mathcal{U})$ for different values of $k$, it is possible to isolate $p$ (which contains the desired information about $\mathcal{E}$) from $A_0$ and $B_0$ (which contain the undesired information about preparation and measurement), creating a protocol that is largely free of systematic errors caused by imprecise knowledge of state preparation and measurement\(^2\).

Case (1) discussed above is the original scenario considered in the RB literature [28, 43, 48, 49], where $\mathcal{U} = \mathcal{I}$ and $\mathcal{E} = \mathcal{I}$, so the observed decay leads to a direct estimate of $\overline{F}(\mathcal{N}, \mathcal{I})$, i.e., how well the randomization operations are implemented. Case (2) discussed above is the extension of RB to the extraction of information about $\overline{F}(\mathcal{E}, \mathcal{U})$, where $\mathcal{E}$ is one of the randomizing operations in the experiment and $\mathcal{U}$ is its unitary idealization. This is a recent development sometimes referred to as *interleaved RB* [31, 50], but we do not make such a distinction in this paper. The previously known result in this case is that $\overline{F}(\mathcal{E}, \mathcal{U})$ can be bounded by experimentally estimating $\overline{F}(\mathcal{E} \circ \mathcal{N}, \mathcal{U})$ and $\overline{F}(\mathcal{N}, \mathcal{I})$, and in Section 3.6.1 we provide more general bounds (with fewer assumptions) for the same purpose.

While the RB protocol is valid for any choice of randomizing operations discussed above, we emphasize that, in order to ensure the protocols remain scalable in the number of qubits, $\mathcal{U}$ and $\mathcal{D}_i$ are restricted to be unitary maps in the Clifford group, since this allows for scalable design of the randomizing sequences via the Gottesman-Knill theorem [33]. Moreover, although previous works have applied the RB protocols only to $\mathcal{E}$ very close to Clifford group maps, we emphasize that no restriction beyond $\mathcal{E}$ being CPTP needs to be imposed. The restricted applications of the RB protocols in previous work was partially due to the bounds used to isolate $\overline{F}(\mathcal{E}, \mathcal{U})$ being only useful when $\mathcal{E}$ is close to a Clifford group map. Since we are interested in extracting information about arbitrary $\mathcal{E}$, we consider here tomographic reconstruction techniques that *do not* rely on these bounds. We also design efficient techniques for

\(^2\)In full generality, $p$ corresponds to an eigenvalue of the map resulting from randomizing $\mathcal{E}$ by conjugation with elements of either the full unitary group or the Clifford group [42, 60, 29, 69]. The eigenvalue interpretation can be used to more clearly see how independence of the estimate from the initial and final states comes about, and it can also be more naturally generalized to cases where the randomizing operations are elements of the Pauli group [69]. Randomization over more general operations can also be considered [8].
average-fidelity estimates that rely on new and improved general bounds on $\bar{F}(E, U)$.

In summary, RB allows for efficient estimation of $\bar{F}(E \circ N, U)$ and efficient bounding of $\bar{F}(E, U)$ for $U$ in the Clifford group. These estimates can be obtained without relying on perfect information about preparation and measurement errors, thereby avoiding some of the systematic errors that may be present in standard tomographic protocols due to these imperfections.

### 3.3.1 RB Sequence Design

A compact way to describe how RB sequences are constructed uses twirling [12, 23, 29, 8, 55]. Although this is not how this construction is typically described, we found it to be convenient, and include it for completeness.

If $E$ is an arbitrary quantum map, and $S$ is a set of maps $\{S_0, \cdots \}$, the average map

$$E_{S_i \in S}[S_i^\dagger \circ U^\dagger \circ E \circ S_i] = \frac{1}{|S|} \sum_{S_i \in S} S_i^\dagger \circ U^\dagger \circ E \circ S_i,$$

is called the twirl of $U^\dagger \circ E$ over $S$, where $E_i$ denotes the expectation value over uniformly random choices for $S_i \in S$. If $S$ is the Clifford group or any other unitary 2-design [23], then

$$E_i[S_i^\dagger \circ U^\dagger \circ E \circ S_i(\hat{\rho})] = p\hat{\rho} + \frac{(1-p)}{d} I$$

where $p = \frac{\text{tr} U^\dagger E - 1}{d^2 - 1} = \frac{d F(EU) - 1}{d - 1}$ as before.

A length $k$ RB sequence consists of applying the twirled channel repeatedly to the same state $k$ times, i.e.,

$$E_{i}[S_k^\dagger \circ U_k^\dagger \circ E \circ S_i \circ \cdots \circ S_i \circ U^\dagger \circ E \circ S_i(\hat{\rho})],$$

$$=E_{i}[D_{i+1} \circ E \circ D_{i+1} \circ \cdots \circ D_{i+1} \circ E \circ D_{i+1}(\hat{\rho})],$$

$$=p^k \hat{\rho} + \frac{(1 - p^k)}{d} I.$$
where

\[ D_\ell = \begin{cases} S_{i_\ell}, & \ell = 1, \\ S_{i_\ell} \circ S_{i_{\ell-1}} \circ U^\dagger, & 1 < \ell \leq k \\ S_{i_k} \circ U^\dagger, & \ell = k + 1, \end{cases} \]  

(3.13)

and \( E_\ell \) denotes the expectation value over uniformly random choices for \( S_\ell \in S \) for all \( \ell \).

The RB protocol to estimate \( F(E, U) \) then consists of (i) choosing sequence of \( S_\ell \) for \( 1 < \ell \leq k \), (ii) applying the alternating sequence of \( D_\ell \) and \( E \), as prescribed in (3.12), to a fixed initial state, (iii) measuring the resulting state, and (iv) averaging over random choices for \( S_\ell \) to obtain \( F_k \). The \( F_k \) can be fit against (3.7), yielding an estimate for \( p \), even in the presence of imperfections. As we prove in Section 3.6, this estimate can be obtained efficiently in the number of qubits, desired accuracy, and confidence. Note that neither \( E \) nor \( U \) need to be elements of the Clifford group. However, we will generally consider the case where \( E \) is not a Clifford group map, while \( U \) will be chosen to be a Clifford group map. Choosing \( U \) to be a Clifford group element \( C_i \) makes the design of the experiments for \( n \)-qubits efficient [33, 49], while leaving \( E \) unconstrained affords us greater flexibility and has no impact on experiment design.

We will normally use \( C_i \) to mean a specific Clifford map to which we are measuring an overlap, and \( S_i \) to mean a Clifford map that is used in a twirling operation.

### 3.4 Tomographic Reconstruction from RB

As discussed above, RB can efficiently provide bounds on the fidelities of an arbitrary CPTP map \( E \) with any element of the Clifford group—in a manner that is robust against preparation and measurement errors, as well as imperfections in the twirling operations. Here we demonstrate that the collection of such fidelities of a fixed \( E \) to a set of linearly independent Clifford group maps can be used to reconstruct a large portion of \( E \). The advantage of this approach is that the robustness properties of the
estimates obtained via RB carry over to this tomographic reconstruction.

Using the Liouville representation of quantum maps, it is clear that an estimate of the average fidelity $F(\mathcal{E}, \mathcal{U})$ leads to an estimate of $\text{tr}\mathcal{U}^\dagger\mathcal{E}$, and thus all information that can be extracted from these fidelities for a fixed $\mathcal{E}$ is contained in the projection of $\mathcal{E}$ onto the linear span of unitary maps. It turns out to be unnecessary to consider the span of arbitrary unitary maps, as the following result demonstrates (see Appendix B.1 for the proof).

**Lemma 3.4.1.** The linear span of unitary maps coincides with the linear span of Clifford group unitary maps. Moreover, the projection of a TP map to this linear span is a unital map.

Given a set of linearly independent vectors that span a subspace, and the inner product of an unknown vector with all elements of that set, it is a simple linear algebra exercise to determine the projection of the unknown vector onto the subspace. Similarly, measuring the average fidelity of some TP map $\mathcal{E}$ to a Clifford group map $\mathcal{C}$ is equivalent to measuring such an inner product—the matrix inner product $\text{tr}(\mathcal{E}\mathcal{C}^\dagger)$. Since Clifford maps span the unital subspace of quantum CPTP maps, measuring the inner product of $\mathcal{E}$ with a set of maximal linearly independent elements of the Clifford group is sufficient to reconstruct the projection of $\mathcal{E}$ onto the unital subspace. We call this projection the *unital part* of $\mathcal{E}$, and denote it by $\mathcal{E}'$.

Since the unitality condition constrains only how the map acts on the identity component of a state, $\mathcal{E}'$ can be obtained by changing how $\mathcal{E}$ acts on that component. Defining $Q$ to be the projector into the identity component of any operator, and $Q^\perp$ to be the projection into the orthogonal complement (i.e. $Q + Q^\perp = I$), one finds that

$$\mathcal{E} = \mathcal{E} \circ (Q^\perp + Q) = \mathcal{E} \circ Q^\perp + \mathcal{E} \circ Q,$$

$$\mathcal{E}' = \mathcal{E} \circ Q^\perp + Q,$$

which indicates that $\mathcal{E}$ and $\mathcal{E}'$ map traceless operators in the same way. The maps $\mathcal{E}$
and $\mathcal{E}'$ have Pauli-Liouville representations

$$
\mathcal{E}^{\text{(PL)}} = \begin{pmatrix} 1 & \bar{O}' \\ \bar{\tau}_{\mathcal{E}} & \mathbb{E} \end{pmatrix}, \\
\mathcal{E}'^{\text{(PL)}} = \begin{pmatrix} 1 & \bar{O}' \\ \bar{\tau}_{\mathcal{E}} & \mathbb{E} \end{pmatrix},
$$

so we refer to $\bar{\tau}_{\mathcal{E}}$ as the non-unital part of $\mathcal{E}$. It is then clear that $\mathcal{E}'$ is described by $(d^2 - 1)^2$ real parameters if $\mathcal{E}$ is TP, while $\mathcal{E}$ itself is described by $(d^2 - 1)d^2$ real parameters. The unital part of $\mathcal{E}$ contains the vast majority of the parameters needed to describe $\mathcal{E}$—in fact, over 93% of the parameters for two qubits, and over 99% of the parameters for four qubits.

As discussed, one limitation of RB is that in a realistic setting it can only provide bounds for $\overline{\mathcal{F}}(\mathcal{E}, \mathcal{C}_i)$ (and therefore $\text{tr} \mathcal{E} C_i^1$) due to the imperfections in the randomizing operations. Clearly these bounds can only lead to a description of parameter-space regions compatible with $\mathcal{E}'$ as opposed to any point estimator, even in the absence of statistical fluctuations. Our approach to reconstruct $\mathcal{E}'$ is to avoid these bounds altogether and instead use the following result, which we prove in Appendix B.2.

**Lemma 3.4.2.** If $(\mathcal{E} \circ \mathcal{N})'$ is the unital part of $\mathcal{E} \circ \mathcal{N}$ and $\mathcal{N}'$ is the unital part of $\mathcal{N}$, and all these operations are trace preserving, then $\mathcal{E}' = (\mathcal{E} \circ \mathcal{N})' \circ (\mathcal{N}')^{-1}$ whenever $(\mathcal{N}')^{-1}$ exists.

This allows us to reconstruct $\mathcal{E}'$ from the reconstructions of $(\mathcal{E} \circ \mathcal{N})'$ and $\mathcal{N}'$. As both $(\mathcal{E} \circ \mathcal{N})'$ and $\mathcal{N}'$ are related directly to decay rates, we can create a point estimate of $\mathcal{E}'$, without recourse to the bounds needed in standard RB to characterize $\mathcal{E}$.

It should be noted that the only cases where $(\mathcal{N}')^{-1}$ does not exist are when $\mathcal{N}$ completely dephases some set of observables (i.e., maps them to something proportional to the identity). However, the experimental setting where tomographic reconstructions are interesting is precisely in the regime where $\mathcal{N}$ is far from depolarizing any observable, so that $(\mathcal{N}')^{-1}$ is typically well defined\(^3\). The penalty, of

\(^3\)For $\mathcal{N}$ chosen at random to have unitary dilations that are Haar distributed, $(\mathcal{N}')^{-1}$ appears to exist with probability 1, so it appears the requirement that $\mathcal{N}$ be close to $\mathcal{I}$ can be significantly weakened.
course, is that the application of $(N')^{-1}$ leads to greater statistical uncertainty in the estimate of $E'$ thanks to the uncertainties in the reconstructions of $N'$ and $(E \circ N)'$ as well as uncertainty propagation due to multiplication by $(N')^{-1}$, but larger experimental ensembles can be used to compensate for this, as is discussed in the section that follows.

Moreover, writing the imperfect randomizing operations as $N \circ S_i$ instead of $S_i \circ N^*$ for some different map $N^*$ is merely a convention, and Lemma 3.4.2 can be trivially adjusted to such a different convention. In the physical regimes where RB estimates are expected to be valid, the choice of conventions is largely immaterial (see Appendix B.5 for more details).

This result shows that the average fidelities with a spanning set of Clifford group unitary maps can lead, not only to a point estimator of the unital part of any TP map, but also to a point estimator of the average fidelity of $E$ to any unitary map—i.e., information from multiple RB experiments can eliminate the need for the loose bounds on the average fidelity considered in [50]. This comes at the cost of efficiency, as the unital part of a map—like the complete map—contains an exponential number of parameters. However, for a small number of qubits the overhead of reconstructing the unital part is small, and therefore it is still advantageous to perform this cancelation to get better estimates of the error.

3.4.1 Example: Single Qubit Maps

In order to reconstruct the unital part of a single-qubit map, one must first consider a set of linearly-independent maps corresponding to unitaries in the Clifford group. As this group contains 24 elements, there are many different choices for a linearly independent set spanning the 10-dimensional unital subspace. One particular choice
of unitaries leading to linearly independent maps is

\[
\begin{align*}
\hat{C}_0 &= I, & \hat{C}_1 &= e^{-\frac{i}{3} \sigma^x}, \\
\hat{C}_2 &= e^{-\frac{i}{3} \sigma^y}, & \hat{C}_3 &= e^{-\frac{i}{3} \sigma^z}, \\
\hat{C}_4 &= e^{-\frac{i}{3} \frac{\sigma^x+\sigma^y+\sigma^z}{\sqrt{3}}}, & \hat{C}_5 &= e^{-\frac{i}{3} \frac{\sigma^x+\sigma^y+\sigma^z}{\sqrt{3}}}, \\
\hat{C}_6 &= e^{-\frac{i}{3} \frac{\sigma^x-\sigma^y+\sigma^z}{\sqrt{3}}}, & \hat{C}_7 &= e^{-\frac{i}{3} \frac{\sigma^x-\sigma^y+\sigma^z}{\sqrt{3}}}, \\
\hat{C}_8 &= e^{-\frac{i}{3} \frac{\sigma^x+\sigma^y-\sigma^z}{\sqrt{3}}}, & \hat{C}_9 &= e^{-\frac{i}{3} \frac{\sigma^x+\sigma^y-\sigma^z}{\sqrt{3}}}.
\end{align*}
\]

In a noiseless setting, estimating the average fidelities between these Clifford maps and the map

\[
\mathcal{H}^{\text{(PL)}} = \begin{pmatrix}
1 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 \\
0 & 0 & -1 & 0 \\
0 & 1 & 0 & 0
\end{pmatrix},
\]

corresponding to the single-qubit Hadamard gate, leads to the decays illustrated in Figure 3-1. The corresponding \(p\) values are

\[
\begin{align*}
p_0 = p_2 = p_8 = p_9 &= \frac{1}{3}, \\
p_1 = p_3 = p_4 = p_5 = p_6 = p_7 &= \frac{1}{3}.
\end{align*}
\]

Note, in particular, that some \(p\) values are negative, which simply indicates an oscillatory exponential-decay behaviour. While these decay rates are much larger (i.e., the \(p\) values are much smaller) than those typically seen in previous RB protocols, we show in Section 3.6.2 that it is possible to efficiently estimate any decay rate to fixed accuracy, no matter the size.

If one considers a noisy setting, where \(\mathcal{N}\) is not the identity, the decay rates are modified by \(\mathcal{N}\), but after reconstructing \(\mathcal{N}'\) and \((\mathcal{E} \circ \mathcal{N})'\) separately, one is able to reconstruct \(\mathcal{E}'\). To see that errors in the estimate of \(\mathcal{N}'\) will not create unmanageable
Figure 3-1: RB decays used to estimate the fidelity between an ideal Hadamard gate and $\hat{C}_0$ (green circles, with $p = \frac{1}{3}$), and $\hat{C}_1$ (orange squares, with $p = -\frac{1}{3}$). The decays corresponding to each of the remaining average fidelities coincide with one of these two representative decays. Note that these decays are much faster than decays previously estimated in RB, as they correspond to the average fidelities between very different maps. The data points are offset along the x-axis for clarity.

errors in the estimate of $\mathcal{E}$, consider how errors in the estimate of $\mathcal{N}'$ affect the estimate of $(\mathcal{N}')^{-1}$. The relative error in the estimate of $(\mathcal{N}')^{-1}$ is given by [37]

$$\frac{\| (\mathcal{N}')^{-1} - (\mathcal{N}' + \mathcal{G}')^{-1} \|}{\| (\mathcal{N}')^{-1} \|} \leq \frac{\kappa(\mathcal{N}')}{{(1 - \kappa(\mathcal{N}'))} \| \mathcal{G}' \| \| \mathcal{N}' \|,}$$

(3.25)

as long as

$$\| \mathcal{G}' \| \| (\mathcal{N}')^{-1} \| < 1,$$

(3.26)

where $\mathcal{G}'$ is the error in the estimate of $\mathcal{N}'$, and $\kappa(\mathcal{N}')$ is the condition number for the matrix inversion of $\mathcal{N}'$ with respect to the matrix norm $\| \cdot \|$. The condition number of $\mathcal{A}$ is given by $\kappa(\mathcal{A}) = \| \mathcal{A}^{-1} \| \| \mathcal{A} \|$ if $\mathcal{A}$ is non-singular, and $\infty$ otherwise.

If we choose $\| \cdot \|$ to be the spectral norm, even when $\mathcal{N}'$ is the depolarizing map...
\[ \mathcal{D}(\hat{\rho}) = \delta \hat{\rho} + (1 - \delta) \frac{\lambda}{2}, \] the condition number of \( \mathcal{N}' \) is given by \( \kappa(\mathcal{N}') = \frac{1}{|\lambda|} \). Similarly, if \( \mathcal{N}' \) is the dephasing map \( \mathcal{Z}(\hat{\rho}) = \frac{1+\gamma}{2} \hat{\rho} + \frac{1-\gamma}{2} \sigma^z \hat{\rho} \sigma^z \), one finds \( \kappa(\mathcal{N}') = \frac{1}{|\gamma|} \). Thus, even for \( \delta \) and \( \gamma \) polynomially close to 0, a polynomial increase in the number of statistical samples can be used to ensure an estimate of the inverse of \( \mathcal{N} \) to any polynomial accuracy with high probability.

### 3.4.2 Beyond Unital Maps

What does the reconstruction of \( \mathcal{E}' \) tell us about \( \mathcal{E} \)? We prove in Appendix B.3 that

**Lemma 3.4.3.** The unital part of a CPTP single-qubit map is always a CPTP map.

This means that the unital part of a single-qubit map imposes no lower bound on the magnitude of the non-unital part of that map—the non-unital part can always be set to 0.

For a single qubit, the unital part does impose stringent conditions on the maximum size of the non-unital part. Up to unitary rotations, any map can be written in the Pauli-Liouville representation as [42]

\[
\begin{pmatrix}
1 & 0 & 0 & 0 \\
t_1 & \lambda_1 & 0 & 0 \\
t_2 & 0 & \lambda_2 & 0 \\
t_3 & 0 & 0 & \lambda_3
\end{pmatrix},
\]

(3.27)

where \( \lambda_i \) and \( t_i \) are real valued parameters. The \( \lambda_i \), corresponding to the unital part, can be estimated using the techniques already described, but as Lemma 3.4.3 demonstrates, no useful lower bound on \( |t_i| \) can be obtained. However, for the map to be positive, it is necessary that \( |t_i| \leq 1 - |\lambda_i| \) [42], which gives upper bounds on the magnitudes of the non-unital parameters.

The fact that, for single-qubit maps, \( \mathcal{E}' \) is always CP can be turned around to say that statistically significant non-CP estimates of \( \mathcal{E}' \) imply statistically significant non-CP estimates of \( \mathcal{E} \), and may be used as witnesses of systematic errors in the experiments [74, 54].
Lemma 3.4.3 fails in the case of multiple qubits, and it is not difficult to construct counter-examples. Numerical experiments indicate that CPTP maps chosen at random by drawing unitary dilations from the Haar distribution lead to non-CP unital parts with probability \( \sim 1 \). This implies that, while it may not be possible to test complete-positivity of a general map by testing only its unital part, the reconstruction of the unital part of a multi-qubit map yields lower-bounds on the magnitudes of the non-unital parameters. Thus, while this result precludes the use of the unital part of a multi-qubit map to test for systematic errors in experiments, it does provide more information about the non-unital parameters.

### 3.5 Fidelity Estimation Beyond the Clifford Group

Previous RB results showed how to bound the average fidelity of Clifford operations [49, 50]. While the maps in the Clifford group form an integral part of current approaches to scalable fault-tolerance in quantum computers, universal quantum computation is only possible if operations outside the Clifford group are also considered. We would like to be able to not only efficiently verify the performance of Clifford gates, but also would like to be able to verify the performance of universal circuits. However, there are strong indications that quantum computers are strictly more powerful than classical computers; for example, if classical computers could efficiently simulate certain classes of non-universal quantum circuits, it would imply a collapse of the polynomial hierarchy [17], and so is considered highly unlikely. It is therefore extremely unlikely that classical computers can efficiently predict the behaviour of a general poly\((n)\)-depth quantum circuit\(^4\), and without these predictions, it is not possible to check if a quantum computer is behaving as desired. For these fundamental reasons, we do not expect that it is possible to efficiently estimate the average fidelity to a general quantum map.

It is important to note, however, that it is possible to efficiently simulate some

---

\(^4\)Here we take a circuit to mean a composition of quantum maps on \(n\) qubits, and the depth to correspond to the number of maps composed.
circuits that contain maps outside the Clifford group. In particular, Aaronson and Gottesman [1] have proven that circuits consisting of Clifford group maps and a logarithmic number of maps outside the Clifford group can be simulated efficiently. Despite being efficiently simulatable, these circuits can be thought of as discrete components that enable universality under composition, and thus the ability to verify their implementation is of great practical importance. We now show how our methods can be extended to allow for efficient estimation of the average fidelity of any experiment to such circuits.

3.5.1 Average Fidelity to $\mathcal{T}$

The canonical example of a map outside the Clifford group is the operation $\mathcal{T}$ corresponding to the unitary $\mathcal{T} = e^{-i\pi \sigma^z}$. This gate is commonly used in combination with Clifford group operations to form a gate set that is universal for quantum computation [15]. In this section we show how to efficiently bound the average fidelity of a map $\mathcal{E}$ to $\mathcal{U} = \mathcal{T}$.

In Section 3.4 we prove that Clifford maps span the space of unital maps. This implies that, in the Pauli-Liouville representation, any unitary map $\mathcal{U}^{(PL)}$ can be written as a linear combination of Clifford maps

$$\mathcal{U}^{(PL)} = \sum \beta^i \mathcal{C}^{(PL)}_i,$$

(3.28)

with $\beta^i \in \mathbb{R}$. By linearity,

$$\text{tr} \mathcal{E} \mathcal{N} \mathcal{U} = \sum \beta^i \text{tr} \mathcal{E} \mathcal{N} (\mathcal{C}^i_1)$$

so

$$\overline{F}(\mathcal{E} \circ \mathcal{N}, \mathcal{U}) = \sum \beta^i \overline{F}(\mathcal{E} \circ \mathcal{N}, \mathcal{C}^i_1) + \frac{1}{d+1} \left(1 - \sum \beta^i\right).$$

(3.30)

For an arbitrary unitary $\mathcal{U}$, the number of non-zero $\beta^i$, which we denote by $N_{\mathcal{U}}$,
can be as large as \( O(d^2) \). However \( T^{(PL)} \) can be written as a linear combination of three Clifford maps. The support of \( T^{(PL)} \) is given by the maps corresponding to the Clifford group unitaries, \( I, \sigma^z, \) and \( e^{-i\frac{\pi}{2} \sigma^z} \), with the corresponding coefficients \( \frac{1}{2}, \frac{1-\sqrt{2}}{2}, \) and \( \frac{1}{\sqrt{2}} \). Thus, to estimate \( \overline{F}(E \circ \mathcal{N}, \mathcal{T}) \), one only needs to estimate 3 average fidelities to Clifford group maps (instead of the 10 necessary for reconstruction of the unital part).

Suppose one estimates each fidelity \( \overline{F}(E \circ \mathcal{N}, \mathcal{C}_i) \) for all of the \( \mathcal{C}_i \) in the linear combination to within \( \epsilon' \) with confidence \( 1 - \delta' \). In Section 3.6.2 we will show this requires \( O\left( \frac{N_{\mathcal{N}} \log \frac{1}{\delta'}}{\epsilon'} \right) \) samples. From Eq. 3.30 it is clear that one can obtain an estimate \( \overline{F} \) such that

\[
\text{Pr}\left( \left| \overline{F} - \overline{F}(E \circ \mathcal{N}, \mathcal{U}) \right| \geq \epsilon' \sum_i |\beta_i^{\mathcal{U}}| \right) \leq N_{\mathcal{N}} \delta'.
\]

Choosing \( \delta' = \delta / N_{\mathcal{U}} \) and \( \epsilon' = \epsilon / \sum_i |\beta_i^{\mathcal{U}}| \) gives

\[
\text{Pr}(\left| \overline{F} - \overline{F}(E \circ \mathcal{N}, \mathcal{U}) \right| \geq \epsilon) \leq \delta,
\]

and requires \( O\left( N_{\mathcal{U}} \left( \sum_i |\beta_i^{\mathcal{U}}| \right)^4 \log \frac{N_{\mathcal{N}}}{\delta} \right) \) samples.

For the particular case of the \( \mathcal{T} \) map, one finds \( \sum_i |\beta_i^{\mathcal{T}}| = \sqrt{2} \), so an estimate for the average fidelity to \( \mathcal{T} \) can be obtained by the following procedure:

1. Perform RB with \( O\left( \frac{1}{\epsilon^4} \log \frac{1}{\delta} \right) \) samples for each relevant fidelity \( \overline{F}(E \circ \mathcal{N}, \mathcal{C}_i) \).

   This requires \( O\left( \frac{1}{\epsilon^4} \log \frac{1}{\delta} \right) \) total samples and results in an estimate \( \overline{F} \) such that

   \[
P(|\overline{F} - \overline{F}(E \circ \mathcal{N}, \mathcal{T})| \geq \epsilon) \leq \delta.
\]

2. Perform RB with \( O\left( \frac{1}{\epsilon^4} \log \frac{1}{\delta} \right) \) samples to obtain an estimate \( \overline{F}_{\mathcal{N}} \) of \( \overline{F}(\mathcal{N}, \mathcal{T}) \) such that

   \[
P(|\overline{F}_{\mathcal{N}} - \overline{F}(\mathcal{N}, \mathcal{T})| \geq \epsilon) \leq \delta,
\]

66
3. In Section 3.6.1, we show how to bound the fidelity of $\overline{F}(\mathcal{E}, \mathcal{U})$, given estimates of $\overline{F}(\mathcal{E} \circ \mathcal{N}, \mathcal{U})$ and $\overline{F}(\mathcal{N}, \mathcal{I})$. Apply the bounds of Section 3.6.1 for $\overline{F}(\mathcal{E} \circ \mathcal{N}, \mathcal{T}) = \overline{F} \pm \epsilon$, and for $\overline{F}(\mathcal{N}, \mathcal{I}) = \overline{F}_N \pm \epsilon$, to obtain bounds on $\overline{F}(\mathcal{E}, \mathcal{T})$ that are valid with probability at least $1 - 2\delta$.

This procedure trivially extends to bounding the fidelity of $\mathcal{E}$ to the case where $\mathcal{T}$ acts on a single qubit and the identity acts on $n - 1$ qubits. The sampling complexity remains the same, but the time complexity changes, as the classical preprocessing time needed to make a single average fidelity estimate scales as $O(n^4)$ [50]. Similar arguments can be used to show that the sampling complexity of determining the average fidelity of $\mathcal{E}$ to any $1$- or $2$-qubit unitary acting on $n$ qubits is constant, with the same classical preprocessing time complexity. In the next section, we will discuss more general operations acting on $n$ qubits.

3.5.2 Average Fidelity to More General Unitaries

It is possible to efficiently bound the average fidelity of a map $\mathcal{E}$ to a unitary $\mathcal{U}$ when $\mathcal{U}$ is a composition of $O(poly(n))$ Clifford maps and $O(log(n))$ $T$ maps on $n$ qubits (i.e., maps that acts as $\mathcal{T}$ on one qubit and as the identity on the remaining $n - 1$ qubits). Under these constraints,

(i) $\mathcal{U}^{(PL)}$ can be efficiently decomposed into a linear combination of $O(poly(n))$ Clifford maps. [i.e. $N_\mathcal{U} = O(poly(n))]$

(ii) The coefficients $\beta_i^{\mathcal{U}}$ in the linear combination satisfy $\sum_i |\beta_i^{\mathcal{U}}| = O(poly(n))$

Following the argument of Section 3.5.1, the sampling complexity scales like

$$O \left( N_\mathcal{U} \left( \frac{\sum_i |\beta_i^{\mathcal{U}}|}{\epsilon} \right)^4 \log \frac{N_\mathcal{U}}{\delta'} \right) , \quad (3.35)$$

so together (i) and (ii) guarantee that the sampling complexity of bounding $F(\mathcal{E}, \mathcal{U})$ is $O(poly(n))$. Since (i) guarantees that the decomposition is efficient, and the classical preprocessing time needed to make a single sample scales as $O(n^4)$, the time complexity is also $O(poly(n))$. 

67
We prove (i) by induction on \( t \), the number of \( T \) maps in the circuit, and \( c \), the number of Clifford maps in the circuit. We show one can decompose \( \mathcal{U}^{\text{(PL)}} \) into a linear combination of at most \( 3^t \) terms, where each Clifford map in the linear combination is written as a composition of at most \( t + c \) Clifford maps. The base case is given by

- \( t = 1, c = 0: \mathcal{U} \) is a single \( T \), and \( \mathcal{U}^{\text{(PL)}} \) can be written as a linear combination of 3 Clifford maps.
- \( t = 0, c = 1: \mathcal{U} \) is a Clifford and so \( \mathcal{U}^{\text{(PL)}} \) can be written as a linear combination of 1 Clifford map.

For the inductive case, assume one has a unitary \( \mathcal{U} \) which is a composition of \( t \) \( T \) maps and \( c \) Clifford maps. By inductive assumption, \( \mathcal{U}^{\text{(PL)}} \) can be written as

\[
\mathcal{U}^{\text{(PL)}} = \sum_{i=1}^{M} \beta_i^{\mathcal{U}} \prod_{j=1}^{N_i} \mathcal{C}^{\text{(PL)}}_{i,j},
\]

where \( M \leq 3^t \) and \( N_i \leq t + c \). Now consider composing \( \mathcal{U} \) with a Clifford \( \mathcal{C} \). Then

\[
\mathcal{C}^{\text{(PL)}} \mathcal{U}^{\text{(PL)}} = \left( \sum_{i=1}^{M} \beta_i^{\mathcal{U}} \mathcal{C}^{\text{(PL)}} \prod_{j=1}^{N_i} \mathcal{C}^{\text{(PL)}}_{i,j} \right),
\]

and one obtains a linear combination of \( \leq 3^t \) terms, each a composition of \( c + t + 1 \) Clifford maps. Likewise, if \( \mathcal{U} \) is composed with \( T \), then

\[
\mathcal{T}^{\text{(PL)}} \mathcal{U}^{\text{(PL)}} = \sum_{i=1}^{M} \sum_{k=1}^{3} \beta_i^{\mathcal{U}} \beta_k^{\mathcal{T}} \mathcal{C}^{\text{(PL)}}_{k} \prod_{j=1}^{N_i} \mathcal{C}^{\text{(PL)}}_{i,j},
\]

where the \( \mathcal{C}^{\text{(PL)}}_k \) are the three Clifford maps involved in the linear combination of \( \mathcal{T}^{\text{(PL)}} \), so one obtains a linear combination of \( 3^{t+1} \) elements, each a composition of \( t + c + 1 \) Clifford maps, as desired.

Therefore, if one has a unitary map \( \mathcal{U} \) composed of \( O(\text{poly}(n)) \) Clifford maps and \( O(\log(n)) \) \( T \) maps, one can write \( \mathcal{U}^{\text{(PL)}} \) as a linear combination of \( O(\text{poly}(n)) \) Clifford maps, where each term in the linear combination is a composition of at most \( O(\text{poly}(n)) \) Clifford maps. A sequence of \( O(\text{poly}(n)) \) Clifford maps can be efficiently
simplified into a single Clifford map using the Gottesman-Knill Theorem [33]. The average fidelity estimate to \( \mathcal{U} \) is obtained by estimating the average fidelities to these simplified Clifford maps.

To see that (ii) also holds, suppose one calculates a linear combination for \( \mathcal{U}^{(PL)} \) based on the above construction. It is possible that different terms in the linear combination result in the same Clifford map, but for simplicity we treat each term separately, so that our estimate of the complexity is an upper bound. Then if the circuit decomposition of \( \mathcal{U} \) contains \( t \) \( T \) maps,

\[
\sum_i |\beta_i|^2 \leq \left( \sum_i |\beta_i^T|^2 \right)^t = \sqrt{2}^t,
\]

so \( \sum_i |\beta_i|^2 \) scales, at most, as \( O(\text{poly}(n)) \) for \( t = O(\log n) \).

These results demonstrate that robust estimates of the average fidelities to unitary maps outside the Clifford group can be obtained efficiently, scaling polynomially in the number of qubits.

3.6 Bounding Error in Average Fidelity Estimates

In this section, we bound sources of error that occur in RB procedures. There are two sources of uncertainty we consider. When trying to efficiently estimate the average fidelity \( \mathcal{E} \) without inverting \( \mathcal{N} \), as we do in Section 3.5, we lack direct access to \( \overline{F}(\mathcal{E}, \mathcal{U}) \) and instead can only estimate \( \overline{F}(\mathcal{E} \circ \mathcal{N}, \mathcal{U}) \) and \( \overline{F}(\mathcal{N}, \mathcal{T}) \). This leads to error on our estimate of \( \overline{F}(\mathcal{E}, \mathcal{U}) \). We also consider statistical error from the sampling of random variables, and show that we can efficiently fit RB decays to any constant error. As a consequence, this allows us to efficiently bound the average fidelity to maps outside the Clifford group, as described in Section 3.5. We address these two effects separately. These types of uncertainties can be found in many contexts, so we expect the analysis in Section 3.6.1 and Section 3.6.2 has broader applications.
3.6.1 Bounds on Average Fidelity of Composed Maps

In this section, we show how to bound $F(E, U)$, when you have estimates of $F(e_0A, U)$ and $F(A, I)$. In Appendix B.4 we prove

$$\chi_{0,0}^{A \circ B} = \chi_{0,0}^A \chi_{0,0}^B \pm \left( 2\sqrt{(1 - \chi_{0,0}^A) \chi_{0,0}^A (1 - \chi_{0,0}^B) \chi_{0,0}^B} + (1 - \chi_{0,0}^A)(1 - \chi_{0,0}^B) \right). \quad (3.40)$$

Setting $B = N$, $A = U^l \circ E$, and using Eq. 3.6 gives bounds on $F(E, U)$ as a function of $F(E \circ N, U)$ and $F(N, I)$.

This bound is valid for any maps $A$ and $B$. There exist $A$ and $B$ that saturate the upper bound, but the lower bound is not tight, for reasons we discuss in Appendix B.4. Generally, this method gives better bounds when the operation $E$ is close to $U$ and when $N$ is close to $I$ (i.e. the imperfections in the randomizing operations are small). Because these lower and upper bounds—just as the bounds in Ref. [50]—are not close to each other except in the regime where $E$ is close to $U$, they are not useful for the type of tomographic reconstruction performed in section Section 3.4, where an arbitrary map might be far from a Clifford map or from a map that is composed of Clifford maps and $O(\text{poly}(n))$ T maps.

Previous work on average-fidelity estimates based on RB have derived the bound [50]

$$\chi_{0,0}^A = \frac{(d^2 - 1)\chi_{0,0}^{A \circ B}}{d^2 \chi_{0,0}^B} \pm E \quad (3.41)$$

$$E = \left| \chi_{0,0}^B - \left( \frac{(d^2 - 1)\chi_{0,0}^{A \circ B}}{d^2 \chi_{0,0}^B} \right) + \left( \frac{d^2 - 1}{d^2} - \chi_{0,0}^B \right) \right|, \quad (3.42)$$

which is only valid when $F(A, I) \geq 2F(B, I) - 1$, or, in the fidelity estimation context, when $F(E, U)$ is close to $1^5$. There is no way to directly verify from the experimental data that this requirement holds, but in order to compare the bounds in Ref. [50] with the bounds derived here, we use Eq. 3.40 to bound region of validity of Eq. 3.41.

---

5 In [50], $E$ is given as the minimum of two functions. However, in almost all realistic cases, this second function never minimizes, and in fact our bounds are always better than the second function, so we have left it out.
As illustrated in Figure 3-2, the bounds derived here are better when \( \bar{F}(A \circ B, I) \) is close to 1, but are applicable to the entire range of parameters without additional assumptions about the maps involved.

![Figure 3-2: Bounds on \( \chi_0^A \) versus \( \chi_{0,0}^{ABC} \), when \( \chi_{0,0}^B \) is fixed at 0.995. Our bounds are solid blue, while the bounds of Ref. [50] are red dashed. The bounds of Ref. [50] are valid in the green shaded region, while our bounds are valid for all values of \( \chi_{0,0}^{ABC} \).](image)

### 3.6.2 Confidence Bounds on Fidelity Estimates

Here we show how to extract \( F(\mathcal{E}, \mathcal{U}) \) from the estimated points \( F_k(\mathcal{E}, \mathcal{U}) \) (the average fidelity of a length-\( k \) RB sequence - see Section 3.3). We rigorously bound the error and sampling complexity of this non-linear fit.

One can easily show that, using the Hoeffding bound, an estimate \( \tilde{F}_k \) for \( F_k(\mathcal{E}, \mathcal{U}) \) can be obtained such that [48]

\[
\Pr(|\tilde{F}_k - F_k(\mathcal{E}, \mathcal{U})| \geq \epsilon') \leq \delta'
\]

(3.43)

with a number of samples \( O \left( \frac{1}{\epsilon'^2} \log \frac{1}{\delta'} \right) \) that is independent of the number of qubits in the system. What we show here is that this allows for \( p \) (and thus \( \bar{F}(\mathcal{E}, \mathcal{U}) \)) to be estimated with a number of samples that also scales well with some desired accuracy.
and confidence. In standard RB experiments, $p$ is estimated by numerical fits to the $\tilde{F}_k$ with many different sequence lengths, but the dependence of the error on the number of samples per sequence length is difficult to analyze. Here we take a different approach that leads to simple bounds on the accuracy and confidence.

Since $F_k(E, U) = A_0 p^k + B_0$, it is easy to see that

$$p = \frac{F_2 - F_\infty}{F_1 - F_\infty}, \quad (3.44)$$

and therefore, at least in principle, $p$ can be estimated by using only sequences of length 1 and 2, along with a sequence long enough to ensure $|A_0 p^k| \ll |B_0|^6$, with corresponding expectation denoted by $F_\infty$. Assuming each $\tilde{F}_i$ is estimated with accuracy $\epsilon'$ and confidence $1 - \delta'$, and that 0 is not in the confidence interval for $\tilde{F}_1 - \tilde{F}_\infty$, it follows that the estimate $\tilde{p}$ for $p > 0$ and $A_0 > 0$ satisfies

$$\frac{p - \frac{2\epsilon'}{A_0 p}}{1 + \frac{2\epsilon'}{A_0 p}} \leq \tilde{p} \leq \frac{p + \frac{2\epsilon'}{A_0 p}}{1 - \frac{2\epsilon'}{A_0 p}} \quad (3.45)$$

with probability at least $1 - 3\delta'$ (similar expressions hold for the cases negative $p$ or $A_0$, but, for simplicity, we focus on the expressions for the positive case). If $|A_0|$ or $|p|$ are small, these bounds diverge, so we will show how to handle these cases.

In most experimental situations where this procedure might be used, $|A_0|$ will be large, close to 1, and so we would like to bound $|A_0|$ away from 0. However, if $|p|$ is small, it is difficult to put a lower bound on $|A_0|$. Since $A_0$ is independent of the sequences being used, we can instead choose to estimate $A_0$ from a different set of sequences for a decay with $p \approx 1$. Once we have a lower bound on $|A_0|$, we can use that to deal with the case that $|p|$ is close to 0.

First we show how to put a lower bound on $|A_0|$. We use a new set of sequences for a decay with large $p$. We denote the $F_i$ estimates for these sequences as $\tilde{F}'_i$, and assuming the confidence interval for $\tilde{F}'_2 - \tilde{F}'_\infty$ does not include 0, $A_0$ is bounded below.

---

*In practice this sequence length can be estimated roughly from the rough rate of decay in experiments without an accurate estimate for any of the model parameters.*
via

\[ A_0 \geq \frac{(\tilde{F}_1 - \tilde{F}_\infty - 2\epsilon')^2}{\tilde{F}_2 - \tilde{F}_\infty + 2\epsilon'} \equiv a, \quad (3.46) \]

with probability at least \( 1 - 3\delta' \).

Now that we have a lower bound on \( |A_0| \), we can use this to determine in which cases \( p \) is large enough that we can use Eq. 3.45, and in which cases \( p \) is small enough that we can set our estimate of \( p \) to be 0. From Eq. 3.43,

\[ ap - 2\epsilon' \leq A_0 p - 2\epsilon' \leq \tilde{F}_1 - \tilde{F}_\infty, \quad (3.47) \]

and thus

\[ p \leq \frac{\tilde{F}_1 - \tilde{F}_\infty + 2\epsilon'}{a} \quad (3.48) \]

so that if one desires an accuracy \( \epsilon \) for \( \tilde{p} \), whenever

\[ \frac{\tilde{F}_1 - \tilde{F}_\infty + 2\epsilon'}{a} \leq \epsilon \quad (3.49) \]

one can set \( \tilde{p} = 0 \) thereby avoiding the divergent confidence intervals while still providing estimates with the desired accuracy.

Similarly, from Eq. 3.43,

\[ \tilde{F}_1 - \tilde{F}_\infty \leq A_0 p + 2\epsilon', \quad (3.50) \]

so whenever

\[ \frac{\tilde{F}_1 - \tilde{F}_\infty + 2\epsilon'}{a} \geq \epsilon, \quad (3.51) \]

it follows that

\[ a\epsilon - 4\epsilon' \leq A_0 p \leq A_0, \quad (3.52) \]
so choosing \( c' = 4e^2a \) one can safely Taylor expand Eq. 3.45 to first order in \( \epsilon \) to obtain

\[
p - \epsilon - O(\epsilon^2) \leq \overline{p} \leq p + \epsilon + O(\epsilon^2),
\]

(3.53)

with probability at least \( 1 - \delta = 1 - 6\delta' \), as desired, using \( O\left(\frac{1}{\epsilon^4 \log \frac{3}{\delta}}\right) \) samples.

This immediately gives that an estimate \( \tilde{F} \) for \( F(\mathcal{E}, \mathcal{U}) \) can be obtained such that

\[
\Pr(|\tilde{F}_k - F(\mathcal{E}, \mathcal{U})| \geq \epsilon) \leq \delta
\]

(3.54)

with \( O\left(\frac{1}{\epsilon^4 \log \frac{1}{\delta}}\right) \) samples.

### 3.7 Experimental Implementation and Analysis

In the previous sections, we described new methods for robust quantum process characterization. In this section, we describe the experimental implementation of these new procedures, and compare the results to standard tomographic methods. We call the method described in Section 3.4, which produces a robust estimate of the unital part of any map, Randomized Benchmarking Tomography (RBT). We call the method described in Section 3.5, which efficiently bounds the fidelity of any map to unitaries composed of a logarithmic number of \( T \) gates and a polynomial number of Clifford gates, Direct Fidelity Estimation (DFE). We compare these two new techniques to the tomography method of Chuang and Nielsen [22], which we call Quantum Process Tomography (QPT). QPT was introduced in the Introduction in Section 1.2.2, and we will describe QPT in more detail in this chapter in Section 3.7.2.

Using a superconducting transmon qubit at Raytheon BBN Technologies, we applied the RBT, DFE, and QPT procedures to the experimental implementations of the single qubit gates

\[
\text{Hadamard} = e^{-i\frac{\pi}{2}(\sigma^x + \sigma^z)}/\sqrt{2} \quad \text{and} \quad \text{XYZ} = e^{-i\frac{\pi}{4}(\sigma^x + \sigma^y + \sigma^z)}/\sqrt{3}.
\]

(3.55)
Note that XYZ is not a Clifford group unitary, and so is inaccessible using standard RB procedures. We also characterized the average operation error, $\mathcal{N}$. For RBT and DFE, $\mathcal{N}$ is the average error of the twirling operations. For QPT, $\mathcal{N}$ is the average error associated with state preparation and measurement operations.

### 3.7.1 RBT Experiments

In Section 3.3, we saw that to perform an RB experiment, we must choose randomized operations from a set $\mathcal{S}$ where $\mathcal{S}$ is a unitary 2-design. Unless otherwise noted, we choose $\mathcal{S}$ to be the 12-element unitary 2-design that consists of a subset of single-qubit Cliffords. While the Clifford group is also a unitary 2-design, it has 24 elements. For error reduction, for some experiments, we exhaustively run all possible sequences, and so using a group with only 12 elements instead of 24 elements significantly reduces the number of experiments we run.

RBT characterizes an operation $\mathcal{E}$ by measuring the trace overlaps between $\mathcal{E}$ and elements $C_i \in \mathcal{S}$. To estimate the unital part of $\mathcal{E}$, we require overlaps between $\mathcal{E}$ and a set of elements of $\mathcal{S}$ that form a minimal spanning set for the unital subspace. In the case of single qubit operations, a minimal spanning set is 10 unitary operations.

We use the following 10 unitaries in $\mathcal{S}$:

\[
\begin{align*}
\hat{C}_1 &= \mathbb{I} \\
\hat{C}_2 &= e^{-i\frac{\pi}{6}\sigma_z} \\
\hat{C}_3 &= e^{-i\frac{\pi}{6}\sigma_y} \\
\hat{C}_4 &= e^{-i\frac{\pi}{6}\sigma_z} \\
\hat{C}_5 &= e^{-i\frac{\pi}{3}\sigma_x + \sigma_y + \sigma_z} \\
\hat{C}_6 &= e^{-i\frac{\pi}{3}\sigma_x + \sigma_y + \sigma_z} \\
\hat{C}_7 &= e^{-i\frac{\pi}{3}\sigma_x - \sigma_y + \sigma_z} \\
\hat{C}_8 &= e^{-i\frac{\pi}{3}\sigma_x - \sigma_y + \sigma_z} \\
\hat{C}_9 &= e^{-i\frac{\pi}{3}\sigma_x + \sigma_y - \sigma_z} \\
\hat{C}_{10} &= e^{-i\frac{\pi}{3}\sigma_x + \sigma_y - \sigma_z}.
\end{align*}
\]

When we perform an RB experiment, we estimate

\[
F_k(\mathcal{E} \circ \mathcal{N}, C_i) = A_0 p_i^k + B_0,
\]

for several different sequence lengths $k$, where $\mathcal{N}$ is the average error channel on the
elements of $S$. From Eq. 3.8,

$$p_i = \frac{\text{tr} C_i^k E N - 1}{d^2 - 1}. \tag{3.58}$$

For most of these decay experiments, we get decays with very small $p_i$ values, as shown in Figure 3-1. We call these “fast” decays, as opposed to the “slow” decays typically seen in standard RB experiments where $p$ is close to 1. As a result, for each overlap unitary $C_i$, we need only measure $F_k(E \circ N, C_i)$ for sequence lengths $k = \{1, 2, 3\}$, because for larger $k$, $F_k(E \circ N, C_i) \approx B_0$, and we obtain no additional information from those sequences.

We also estimate $F_\infty(E \circ N, C_i) = B_0$. If the final measurement operation is $\hat{M}$, then (see [49], Eq. 3.27),

$$F_\infty(E \circ N, C_i) \equiv F_\infty = \text{tr} \left( \hat{M} N \left( \frac{I}{d} \right) \right), \tag{3.59}$$

where $F_\infty$ doesn’t depend on $E$ or $C_i$, but depends only on the measurement operation $\hat{M}$. To estimate $F_\infty$, we use the depolarizing action of $S$ on states. That is

$$E_{S_i \in S} S_i(\rho) = \frac{I}{d} \tag{3.60}$$

for any state $\rho$. Therefore

$$F_\infty = \text{tr} \left( \hat{M} N \left( \frac{I}{d} \right) \right) = E_{S_i \in S} \text{tr}(\hat{M} N \circ S_i(\rho_0)). \tag{3.61}$$

Since $N \circ S_i$ is the experimental implementation of $S_i$, we can estimate $F_\infty$ by averaging over sequences where we prepare an initial state, apply an element of $S$ at random, and then measure.

In standard RB, estimates of $F_k(E \circ N, C_i)$ suffer from two sources of error. There is statistical error, due to the probabilistic nature of the experimental outcomes. There is also sampling error; $F_k(E \circ N, C_i)$ is ideally an average over all sequences of length
but it is normally impractical to test all possible sequences. Instead, one randomly samples from sequences of length $k$, and the average of these random samples will approach the true average, with an error that decreases with the number of samples [49]. For our purposes, since we have so few data points (only $k = \{1, 2, 3, \infty\}$) for each decay, we would like to minimize the error as much as possible. Consequently, rather than sampling sequences at random, we exhaustively run all possible sequences to avoid sampling error. Since we have 12 twirling operations in $\mathcal{S}$, to run all sequences for $F_k$ requires $12^k$ experiments, while to estimate $F_\infty$ requires 12 experiments. For $F_1$ and $F_\infty$ sequences, we repeat each experiment 12 times so that the total number of experiments for these sequences is $12^2$. We do this so that the statistical error on these estimates is closer to that of $F_2$ and $F_3$, which have $12^2$ and $12^3$ experiments respectively.

Rather than estimate the decay parameter using the methods of Section 3.6.2, we find we get smaller errors if we fit the data to an exponential decay using a non-linear least squares fitting method (in MATLAB lsqnonlin). As we mentioned, the decay parameter $p_i$ is small for most $C_i$. When we use lsqnonlin to fit such a decay, the fitting routine will typically find an optimal fit where $p_i$ is large, and instead $A_0$ is small. To account for this, we fit all 10 decays at the same time, and also include an additional slow decay obtained from a standard randomized benchmarking experiment. For all of these decays, $A_0$ and $B_0$, which depend on state preparation and measurement, should be the same. Our fitting routine finds the value of $A_0$ and $B_0$ that give the best fits for the whole set of 11 decays. In particular, the slow decay has a decay parameter $p \approx 1$, and so the fitting routine can not cheat and set $A_0$ to be small. Given a set of $p_i$'s from this fitting routine, we can convert them into a set of estimates of $\text{tr}(E N C_i)$ using Eq. 3.58.

Let $\vec{O} = (O_1, \ldots, O_{10})$, where $O_i$ is the estimate of $\text{tr}(E N C_i)$. To turn $\vec{O}$ into an estimate of $E \circ N$, we perform linear inversion. To describe linear inversion, we need some new notation. Let $|T\rangle$ be the column majorized vector of the matrix $T$ formed
by stacking the columns of $T$. That is,

$$\text{if } T = \begin{pmatrix} 1 & 2i \\ 3 & 4i \end{pmatrix} \text{ then } |T\rangle = \begin{pmatrix} 1 \\ 3 \\ 2i \\ 4i \end{pmatrix}. \quad (3.62)$$

We denote the conjugate transpose of $|T\rangle$ by $\langle T|$.

Then note that

$$\text{tr}((\mathcal{E}\mathcal{N}c_i^\dagger) = \langle \mathcal{C}_i^{(PL)}|\mathcal{E}^{(PL)}\mathcal{N}^{(PL)}\rangle, \quad (3.63)$$

where recall all matrices inside the tr are in the Pauli-Liouville representation. This implies

$$\tilde{\mathcal{O}} = \mathcal{P}|\tilde{\mathcal{E}} \circ \tilde{\mathcal{N}}^{(PL)}\rangle \quad (3.64)$$

where $\mathcal{P}$ is the matrix whose $i^{th}$ row is $\langle c_i^{(PL)}|$ and $\tilde{\mathcal{E}} \circ \tilde{\mathcal{N}}$ is our estimate of $\mathcal{E} \circ \mathcal{N}$. Then to obtain $\tilde{\mathcal{E}} \circ \tilde{\mathcal{N}}$ from $\tilde{\mathcal{O}}$, calculate

$$\mathcal{P}^+\tilde{\mathcal{O}} = |\tilde{\mathcal{E}} \circ \tilde{\mathcal{N}}^{(PL)}\rangle, \quad (3.65)$$

where $\mathcal{P}^+$ is the Moore-Penrose pseudo-inverse of $\mathcal{P}$. (We can’t use the true inverse because $\mathcal{P}$ has dimension $10 \times 16$.) This procedure is known as linear inversion, and gives us our estimate of the unital part of $\mathcal{E} \circ \mathcal{N}$.

We repeat the procedure to estimate the unital part of $\mathcal{N}$ alone, and then use the inversion procedure of Section 3.4 to obtain an estimate of $\mathcal{E}$.

### 3.7.2 QPT Experiments

For each QPT experiment, we prepare an initial state, apply the unknown map $\mathcal{E}$, and then perform tomography on the resulting state [22]. We prepare multiple initial
states to obtain a complete characterization of $\mathcal{E}$.

In an ideal world, we would prepare the initial states

$$
\rho_1 = |0\rangle\langle 0| \\
\rho_2 = |1\rangle\langle 1| \\
\rho_3 = |+\rangle\langle +| \\
\rho_4 = |-\rangle\langle -| \\
\rho_5 = |\leftrightarrow\rangle\langle \leftrightarrow| \\
\rho_6 = |\rightarrow\rangle\langle \rightarrow|
$$

and measure the expectation values of the following operators:

$$
M_1 = \sigma^x \\
M_2 = \sigma^x \\
M_3 = \sigma^y
$$

(3.67)

However, because we don’t have perfect state preparation and measurement, in reality we prepare the states $\hat{\rho}_i \approx \rho_i$ and perform the measurements $\hat{M}_i \approx M_i$. Thus each of our experimental outcomes is an estimate of $\text{tr}(\hat{M}_j\mathcal{E}(\hat{\rho}_i))$.

In standard QPT procedures, there is no way to get an accurate characterization of $\hat{M}_i$ or $\hat{\rho}_i$, so we must instead make assumptions about their form. We make the standard assumption

$$
\hat{M}_i = \alpha M_i + \beta \mathbb{1} \\
\hat{\rho}_i = \rho_i
$$

(3.68)

i.e. that state preparation is perfect, and that each measurement suffers from a depolarizing error with strength $\alpha$, and a relaxation error (amplitude damping channel) with strength $\beta$. For superconducting qubits, this is a fairly accurate first-order approximation of the true errors on state preparation and measurement.

Under these assumptions, our experimental observables $O'_{ij}$ are estimates of

$$
\text{tr}(\hat{M}_j\mathcal{E}(\hat{\rho}_i)) = \alpha \text{tr}(M_j\mathcal{E}(\rho_i)) + \beta.
$$

(3.69)
To estimate $\alpha$ and $\beta$, we need only observe

$$
\text{tr}(\hat{M}_1\hat{\rho}_1) = \alpha + \beta, \quad \text{and} \quad \text{tr}(\hat{M}_1\hat{\rho}_2) = -\alpha + \beta.
$$

(3.70)

We call $\hat{\alpha}$ and $\hat{\beta}$ our estimates of $\alpha$ and $\beta$. Our estimate $O_{ij}$ of $\text{tr}(M_j\mathcal{E}(\rho_j))$ is thus

$$
O_{ij} = \frac{O_{ij} - \hat{\beta}}{\hat{\alpha}}.
$$

(3.71)

As in RBT, we perform linear inversion to estimate $\mathcal{E}$. Using the Pauli-Liouville representation, $\rho_i^{(\text{PL})}$ and $\hat{M}_j^{(\text{PL})}$ are vectors with components given by the trace overlap with the Pauli operators (see Section 3.2), so

$$
O_{ij} = \hat{M}_j^{(\text{PL})\dagger}\hat{\mathcal{E}}^{(\text{PL})}\rho_i^{(\text{PL})} = \text{tr}\left(\hat{\rho}_i^{(\text{PL})}\hat{M}_j^{(\text{PL})\dagger}\hat{\mathcal{E}}^{(\text{PL})}\right) = \text{tr}(P_{ij}\hat{\mathcal{E}}^{(\text{PL})}),
$$

(3.72)

where $\hat{\mathcal{E}}$ is our estimate of $\mathcal{E}$ and $P_{ij} = \rho_i^{(\text{PL})}\hat{M}_j^{(\text{PL})\dagger}$ is a $d^2 \times d^2$ matrix. Using the column majorized vector notation, we can write

$$
\vec{O} = \mathcal{P}|\hat{\mathcal{E}}^{(\text{PL})}\rangle
$$

(3.73)

where the $(i,j)^{\text{th}}$ row of $\mathcal{P}$ is $\langle P_{ij}|$ and the $(i,j)^{\text{th}}$ element of the vector $\vec{O}$ is $O_{ij}$. Finally, we obtain our estimate of $\mathcal{E}$ by computing

$$
\mathcal{P}^\dagger \vec{O} = |\hat{\mathcal{E}}^{(\text{PL})}\rangle.
$$

(3.74)

### 3.7.3 DFE Experiments

In Section 3.5, we describe how to bound the fidelity of experimental maps to unitaries that can be written as a linear combination of twirling maps. We apply this technique to Hadamard and XYZ. Hadamard is an element of the full Clifford group, but not of the reduced 12-element 2-design. As a result, to characterize Hadamard, we choose
the twirling set to be the full single qubit Clifford group. In that case, Hadamard can be written as a linear combination of a single twirling operation, so we need to only observe a single decay (along with the decay for the average Clifford error).

For XYZ, we twirled over the 12 element 2-design, as XYZ can be written as a linear combination of three maps from this set:

\[ XYZ^{(PL)} = 0.9107C_1 + 0.3333C_5 - 0.244C_6 \] 

(3.75)

where \( C_i \) are given in Eq. 3.56. We need only to observe the three decays corresponding to the unitaries \( \{C_1, C_5, C_6\} \), (along with the decay for the average twirling error) to estimate the average fidelity of XYZ.

### 3.7.4 Bootstrapping

Implementing the protocols of Sections 3.7.1-3.7.3 gives a point estimate of \( E \). Ideally, we would obtain error bars for this point estimate by running the experiment many times, and the range of point estimates from these repetitions would provide an error estimate. The repeated experiments will differ from the original experiment due to statistical fluctuations,\(^7\) and so these error bars will represent the level of statistical uncertainty. These error bars do not capture any inherent bias in the experimental set-up that could lead to errors.

While we might like to repeat the experiment many times, due to time constraints, this is not feasible. In addition, even if we had been able to rerun the experiment many times, it would be good to use the extra data from these additional experiments towards our point estimate, rather than just using them to create error bars.

We could use the error analysis of Section 3.6.2 to produce error bars. However, errors bars derived using that analysis will be too large for several reasons. First, the Hoeffding bound used in that error analysis is far from tight, and second, since we use a non-linear least squares fitting routine rather than the protocol of Section 3.6.2,

\(^7\)Ideally the repeated experiments will only differ from the original experiment due to statistical fluctuations; in the real world, experimental elements may drift over time, likewise producing estimates that drift over time. This produces a bias in the error bars.
we should have even better error bars.

To circumvent these difficulties, we use bootstrapping. Bootstrapping produces error bars that reflect statistical variations, but don't require additional experimental runs, and yet provide more accurate error estimates than the analysis of Section 3.6.2. Additionally we don’t need to make any assumptions about the underlying probability distribution of the data in order to use bootstrapping. For an excellent introduction to bootstrapping, see Ref. [53].

To perform bootstrapping, we create many simulated datasets based on the original dataset. These simulated datasets will capture statistical fluctuations in the original experiments. We then run each simulated dataset through the same fitting and linear inversion routines we used on the original dataset. The spread in the estimates that result from these simulated datasets then give error bars on the original estimate.

In order to facilitate the creation of simulated datasets, we take our original dataset in a specific way. For each observable, we do $100^2$ measurements. For QPT an observable is defined by a specific state preparation and measurement. For RBT and DFE, an observable is defined by a specific sequence of twirling operations. We divide the $100^2$ measurements for each observable into 100 bins, where each bin contains the mean of 100 measurement outcomes. After the binning step, we have 100 estimates of the expectation value of each observable, which we call samples. These 100 samples approximate the statistical distribution of possible outcomes for that observable.

Our simulated datasets have the same form as our original dataset: for each experimental observable, we have 100 simulated samples. To create these simulated samples, we sample uniformly at random, with replacement, from the original 100 samples of that observable. This is known as non-parametric bootstrapping. Using this method, we create 1000 bootstrapped datasets for each original dataset. We run each of these 1000 bootstrapped datasets through our fitting routines (see Sections 3.7.1-3.7.3). Each bootstrapped dataset produces its own point estimate for any quantity of interest. We then look at the spread of point estimates from all of the bootstrapped data, and pick out the 5th and 95th percentile estimates to form the 90
percent confidence interval error bars for the original estimate. Using 100 samples and 1000 bootstrapped datasets will give good statistical error bars with high probability [53].

3.7.5 Experimental Results

In Figure 3-3, we plot the average fidelity of the implemented Hadamard, XYZ, and operation error maps to their corresponding ideal unitary maps. We use QPT, RBT, and DFE to determine the average fidelity. Using QPT and RBT, we first create a complete estimate of the implemented map, or an estimate of the unital part of the map, respectively, and then estimate the average fidelity using Eq. 3.8. For DFE, we do not learn all of the parameters of the maps, and instead estimate the average fidelity directly. Using RBT, for each gate, we create estimates both with inversion (I) of the average twirling operation error $N$, as described in Section 3.4, and with no inversion (NI) of the twirling error. Additionally, as described in Section B.5, we have a gauge choice as to whether the average twirling error occurs before or after the perfect Clifford operation. The reconstruction obtained when we assume the error occurs before (after) the perfect Clifford is labeled Pre (Post).

There are several things to note in Figure 3-3:

1. All error bars except DFE error bars on Hadamard and XYZ are 90 percent confidence intervals derived using the bootstrapping method described in Section 3.7.4.

2. The error bars for Hadamard and XYZ using DFE are exceptionally large because they come from the bounds in Section 3.6.1 (in addition to bootstrapping). These error bars are large because we don’t have enough information about the average twirling operation error $N$ to accurately estimate the average fidelity of $\mathcal{E}$ given the average fidelity of $N \circ \mathcal{E}$.

3. Figure 3-3(a) includes DFE estimates, and so the $y$–axis has a larger range to fit the full error bars on these estimates. Figure 3-3(b) is a zoomed in version that does not include the DFE estimates, and so has a smaller $y$–axis range.
Figure 3-3: Average fidelity of the implemented Hadamard and XYZ gates to their ideal implementation, and of the operation error to the identity. RBT estimates with inversion (I) and without inversion (NI), and with Clifford errors assumed to occur before (Pre) or after (Post) the perfect implemented Clifford (see Section B.5) are plotted. Estimates obtained using DFE and QPT protocols are shown for comparison. Figure (a) includes DFE estimates, while Figure (b) does not - otherwise the plots differ only in scale. All error bars are obtained using bootstrapping, and the DFE estimates for Hadamard and XYZ include the uncertainty resulting from the bounds derived in Section 3.6.1.
4. The larger error bars on QPT estimates vs. RBT estimates is a result of taking different amounts of data for the two experiments, and is not indicative of a more efficient experiment. In fact QPT requires fewer experiments to achieve the same error bars.

5. We see that QPT estimates are close to RBT estimates without inversion (NI). This makes sense, because the map that is actually characterized by QPT is a composition of the implemented gate $E$ with the average error from state preparation and measurement. The state preparation and measurement for QPT involves applying Cliffords, so the map that QPT characterizes is approximately the composition of $E$ with the average Clifford error. Thus RBT without inversion and QPT both characterize maps that are close to $E \circ \mathcal{N}$, where $\mathcal{N}$ is the average Clifford error.

6. Choosing a gauge where the Clifford error occurs before or after the Clifford (Pre or Post), does not have a large effect on the fidelity estimate.

7. Fidelity estimates using RBT with inversion (I) are larger than QPT. We believe these larger fidelity estimates are more accurate than those obtained using QPT. QPT can’t isolate $E$, and so errors due to state preparation and measurement affect the estimate of $E$, making it look like the gate is performing more poorly than it actually is.

**Systematic Errors**

For our RBT and QPT estimates, we use linear inversion, as described in Sections 3.7.1-3.7.2. Linear inversion does not constrain the estimate to be CPTP. Since the experimental map must be CPTP (by the laws of quantum mechanics), if our estimate is significantly non-CPTP, it is a sign of systematic errors in our tomography procedure. As a result, we can use the degree to which the estimate is non-CPTP to test for systematic errors in our tomography procedure. In particular, we use a test called the negative witness test [54]. While originally formulated to test for systematic errors in state tomography procedures, it can easily be adapted to test
for systematic errors in process tomography procedures using the Choi-Jamilkowski isometry between quantum maps and quantum states [20, 41]. We can only use this test for the RBT and QPT protocols, since DFE does not give enough information to determine whether the map is CPTP or not.

To perform the negative witness test, we use the first 50 samples (out of the 100 total samples, see Section 3.7.4) of each observable to create an estimate of the map in the Choi-Jamilkowski representation, which is a matrix representation. From this matrix, we extract the eigenvector with the smallest corresponding eigenvalue. This is called the negativity witness. We then create a separate estimate of the process using the second 50 samples from each observable and calculate the trace overlap overlap of this second reconstructed process with the negativity witness. For a valid CPTP map, all eigenvalues in the Choi-Jamilkowski representation are positive, so if the overlap is negative, this might be a sign of systematic errors in the tomography protocol.

Statistical fluctuations due to the probabilistic nature of the experiments can also cause the reconstructed map to have negative eigenvalues, so we need to create statistical error bars to determine if the negativity is due to systematic errors or statistical fluctuations. We do this using bootstrapping (see Section 3.7.4). We create simulated datasets by bootstrapping the second set of 50 samples for each observable, but use the same negative witness (the one formed using the first set of 50 samples) to test for negativity in all of these simulated estimates [54]. The results of this test are shown in Figure 3-4, with error bars from 1000 sets of bootstrapped data showing 90 percent confidence intervals.

There are several things to note in Figure 3-4:

1. The results of the negative witness test show that QPT and RBT are with high probability suffering from systematic errors. If the negativity were due to statistical fluctuations instead of systematic errors, we would see 0 in the range of the error bars.

2. The RBT estimate for the operation error is significantly negative, implying
Figure 3-4: Results of the negative witness test for RBT and QPT protocols. RBT estimates with inversion (I) and without inversion (NI), and with Clifford errors assumed to occur before (Pre) or after (Post) the perfect implemented Clifford (see Section B.5) are plotted. Figure (a) shows the negativity test results for all possible RBT and QPT estimates, while Figure (b) shows the negativity test results for QPT and for RBT without inversion, to highlight the fact RBT results are less negative than QPT results. All error bars are obtained using bootstrapping.
that this estimate is far from a valid quantum process. As a result, when we use this estimate to invert the operation errors on our Hadamard and XYZ maps, we get both more negative and more varied results, as seen in the red (I) data points of Figure 3-4(a).

3. If we ignore the estimates with inversion, and focus on RBT without inversion (NI) and QPT, as in Figure 3-4(b), we see RBT estimates are less negative than QPT estimates. This may be a sign that systematic errors are less severe in RBT than in QPT. The negative witness test is a one way test: if the estimate is negative, there must be systematic errors, but if the estimate is not negative, systematic errors could still exist. Thus while a less negative result probably points to fewer systematic errors, we can not prove this.

We have some preliminary evidence that systematic errors in RBT are due to the fitting procedure. In particular, fitting multiple decays at once using the non-linear least squares method seems to introduce some bias into the results. We are currently investigating new fitting procedures that will hopefully produce results that are not biased.

### 3.8 Summary and Outlook

We have proved that, using information from multiple RB experiments, it is possible to reconstruct the unital part of any completely-positive trace-preserving map in a way that is robust against preparation and measurement errors, thereby avoiding some forms of systematic errors that plague more traditional tomographic reconstruction protocols. The unital part of a map consists of the vast majority of the parameters of that map, including all parameters necessary to describe any deterministic unitary map, as well as any random unitary map, such as dephasing with respect to any eigenbasis.

We also presented a robust procedure for bounding the average fidelity to an arbitrary unitary, and show that this protocol is efficient for a large class of unitaries
outside of the Clifford group. The overhead of the procedure depends on how the unitary is decomposed as a linear combination of Clifford group unitary maps, and we give rigorous bounds on the number of samples needed to achieve some desired accuracy and confidence in the fidelity estimate.

Finally, we demonstrated these tomography protocols experimentally, and found some evidence that our new protocols are more accurate than standard tomography methods. We still found evidence of systematic errors in our new procedures, and are working to tease out the source of these errors.

The extension of these results to non-qubit systems remains an open problem. In addition, the characterization of the non-unital part of a map in a robust manner seems to present a larger challenge than the characterization of the unital part, although recent advancements like Gate Set Tomography [14] provide a way to access this non-unital information. For many experiments, noise due to time correlations is a significant issue, and our methods assume that errors are Markovian. It would be useful if these methods could be adapted to non-Markovian noise.

3.9 Acknowledgments

We thank Easwar Magesan for many illuminating discussions. This research was funded by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), through the Army Research Office grant W911NF-10-1-0324. All statements of fact, opinion or conclusions contained herein are those of the authors and should not be construed as representing the official views or policies of IARPA, the ODNI, or the U.S. Government. S.K. is grateful to Raytheon BBN Technologies for supporting her through several internships where this work was completed.
3.10 Contributions

This research was a true collaborative effort. Most of the theory was done with Marcus da Silva at Raytheon BBN Technologies. The experimental data was taken by Colm Ryan, Blake Johnson, and Tom Ohki, of Raytheon BBN technologies, and some of their older code was used in the analysis of Section 3.7. My largest contributions were the ideas and proofs for Sections 3.4.2, 3.5, 3.6, B.3, B.4, and B.5. I wrote the code to perform the bootstrapping and major parts of the analysis in Section 3.7 and created all of the plots in that section.
Chapter 4

Quantum Local Search Algorithm for Quantum 2-SAT

4.1 Introduction

For the $n$-bit classical constraint satisfaction problem $k$-SAT, several algorithms have been developed that beat the exhaustive search runtime bound of $2^n$. They provide a runtime with a mildly exponential scaling, $O(r^n)$ with $r < 2$. One such algorithm is Schöning’s probabilistic algorithm that finds a solution of 3-SAT in a time that is a polynomial multiple of $O(1.334^n)$ [66]. Although variants of the algorithm had been known for some time [52, 58], Schöning was the first to prove the runtime bound for $k \geq 3$. For 2-SAT, Papadimitriou earlier introduced a variant of this algorithm that finds a satisfying assignment (if there is one) in polynomial time with high probability [58]. The algorithm for 2-SAT is as follows:

*Classical Algorithm*

- Pick a random string.
- Repeat $O(n^2)$ times.
  - If unsatisfied clauses exist, randomly choose one of the unsatisfied clauses and flip one of the bits, chosen at random, that the clause acts on.
As mentioned in Section 1.2.3, this is a local search algorithm, so called because the state of the system at each step differs from the previous state by a single bit flip.

The quantum algorithm that we consider is the natural generalization of this procedure to the quantum domain for the problem Quantum $k$-SAT, which is the natural generalization of Classical $k$-SAT to the quantum domain. We now give the definition of Quantum $k$-SAT as it was introduced by Bravyi [16]:

**Definition 4.1.1 (Quantum $k$-SAT).** Let $c = \Omega(n^{-g})$ with $g$ a positive constant. Given a set of $L$ rank one projectors (called "clauses") $\Phi_\alpha = |\phi_\alpha\rangle \langle \phi_\alpha|$ each supported on $k$ out of $n$ qubits, one must decide between the following two cases:

1. The **YES instance**: There exists a state, $|G\rangle \in \mathbb{C}^{2^n}$ that satisfies $\Phi_\alpha |G\rangle = 0$ for all $\alpha = \{1 \ldots L\}$.

2. The **NO instance**: We have for all $|\psi\rangle \in \mathbb{C}^{2^n}$ that $\sum_{\alpha=1}^{L} \langle \psi | \Phi_\alpha | \psi \rangle \geq c$.

Bravyi [16] showed that there is a polynomial time classical algorithm for Quantum 2-SAT, and he also showed that Quantum $k$-SAT for $k \geq 4$ is complete for QMA$_1$. Recently, it was shown that quantum 3-SAT is QMA$_1$ complete [32].

We now give our quantum algorithm for $k$-SAT, but in this paper we focus on $k = 2$.

**Quantum Algorithm**

- Choose an initial state at random, i.e. initialize in the maximally mixed state $\frac{1}{2^n}$.

- Repeat $T$ times.
  
  - Choose $\alpha$ uniformly at random from $\{1, \ldots, L\}$, and measure $\Phi_\alpha$. If outcome 1 is measured, choose one of the qubits in the support of $\Phi_\alpha$ at random and apply a Haar random unitary to that qubit. If outcome 0 is measured do nothing.

- Use a verification procedure to test whether the resulting state is close to a state that satisfies all of the clauses. This can be done by performing phase estimation of the Hamiltonian $H = \sum_\alpha \Phi_\alpha$, or by repeatedly measuring the projectors $\{\Phi_\alpha\}$ on the final state to see if there are violations.
Variants of this algorithm have been analyzed previously in different contexts. A similar algorithm was proposed to prepare graph states and Matrix Product States dissipatively [73] and a variant was used as a tool for the constructive proof of a symmetric quantum local Lovász lemma for commuting projectors [6, 67].

Given a YES instance of Quantum 2-SAT, one might expect that the Quantum Algorithm will converge to a satisfying state in polynomial time, since Quantum 2-SAT is in $P$. We show that this is indeed the case, at least for a restricted set of clauses. Bravyi [16] showed that a YES instance of Quantum 2-SAT always has a product state solution. Without loss of generality, we can take this solution to be $|0\rangle^\otimes n$. In this basis all of the clauses are of the form

**General Clauses:**

\[ \Phi_\alpha = |\phi_\alpha\rangle \langle \phi_\alpha|, \quad \text{with} \quad |\phi_\alpha\rangle = a_\alpha |01\rangle_{i,j} + b_\alpha |10\rangle_{i,j} + c_\alpha |11\rangle_{i,j}, \quad (4.1) \]

where $i, j$ label the two qubits in the clause $\Phi_\alpha$. For reasons that we will discuss later, we can only prove that the Quantum Algorithm succeeds in polynomial time if the clauses are restricted to have $c_\alpha = 0$. This restriction can be somewhat relaxed, and in Appendix C, we show that the algorithm succeeds in polynomial time if for every clause either $c_\alpha = 0$ or $a_\alpha = b_\alpha = 0$. So we work with

**Restricted Clauses:**

\[ \Phi_\alpha = |\phi_\alpha\rangle \langle \phi_\alpha|, \quad \text{with} \quad |\phi_\alpha\rangle = a_\alpha |01\rangle_{i,j} + b_\alpha |10\rangle_{i,j}. \quad (4.2) \]

Note that $|0\rangle^\otimes n$ and $|1\rangle^\otimes n$ are now both satisfying states.

In addition to only allowing Restricted Clauses, for the Quantum Algorithm to succeed in polynomial time we need a restriction on the eigenvalues of the positive
semi-definite operator

\[ H = \sum_{\alpha=1}^{L} \Phi_{\alpha}. \] (4.3)

\( H \) has eigenvalue 0, and let \( \epsilon \) be the size of the next largest eigenvalue. We require that \( \epsilon = \Omega(n^{-m}) \) for some constant \( g_1 \).

### 4.2 Analysis of the Quantum Algorithm for Restricted Clauses

In this section, we show that the quantum algorithm described above converges in polynomial time for Quantum 2-SAT when the clauses are all of the form of Eq. 4.2. The algorithm can be viewed as a quantum Markov process that converges to a steady state that is annihilated by all the clauses. A quantum Markov process is described by a completely positive trace preserving (CPTP) map [24]. Call \( \rho_t \) the state of the system at time \( t \), so \( \rho_0 = \mathbb{I}/2^n \). The CPTP map \( T \) describes the update of \( \rho_t \) at each step of the chain, so \( \rho_{t+1} = T(\rho_t) \).

Call \( T_{\alpha} \) the map that describes the procedure of checking whether clause \( \Phi_{\alpha} \) is satisfied, and if it is not satisfied, applying a random unitary to one of the qubits in the support of \( \Phi_{\alpha} \). Let \( i \) and \( j \) be the two qubits associated with clause \( \Phi_{\alpha} \). Then

\[ T_{\alpha}(\rho) = (\mathbb{I} - \Phi_{\alpha}) \rho (\mathbb{I} - \Phi_{\alpha}) + \frac{1}{2} \Lambda_i(\Phi_{\alpha} \rho \Phi_{\alpha}) + \frac{1}{2} \Lambda_j(\Phi_{\alpha} \rho \Phi_{\alpha}) \] (4.4)

where \( \Lambda_i \) is the unitary twirl map acting on qubit \( i \):

\[ \Lambda_i(\rho) = \int d[U_i] U_i \rho U_i^\dagger = \frac{\mathbb{I}_i}{2} \otimes \text{tr}_i[\rho]. \] (4.5)

where \( d[U_i] \) is the Haar measure. At each time step, we choose \( \alpha \) from \( \{1, \ldots, L\} \) uniformly and random and apply the map \( T_{\alpha} \). This corresponds to the CPTP update.
One useful property of $\mathcal{T}$ is that the probability of being in the satisfying subspace can not decrease under the action of $\mathcal{T}$. Call $\Pi_0$ the projector onto the subspace spanned by satisfying states, so $\Pi_0\Phi_\alpha = 0$ for all $\alpha$. Then

$$\text{tr}[\Pi_0\rho_{t+1}] - \text{tr}[\Pi_0\rho_t] = \frac{1}{L} \sum_\alpha \text{tr} \left[ \Pi_0 (I - \Phi_\alpha) \rho_t (I - \Phi_\alpha) \right]$$

$$+ \frac{1}{2} \Pi_0 \Lambda_i (\Phi_\alpha \rho_t \Phi_\alpha) + \frac{1}{2} \Pi_0 \Lambda_j (\Phi_\alpha \rho_t \Phi_\alpha) - \text{tr}[\Pi_0\rho_t]$$

$$= \frac{1}{2L} \sum_\alpha \text{tr}[\Pi_0 (\Lambda_i (\Phi_\alpha \rho_t \Phi_\alpha) + \Lambda_j (\Phi_\alpha \rho_t \Phi_\alpha))]$$

$$\geq 0,$$  (4.7)

since $\text{tr}[\Pi \rho] \geq 0$ for any projector $\Pi$ and any state $\rho$.

### 4.2.1 Runtime Bound for the Algorithm

In analyzing the classical algorithm, Papadimitriou and Schöning kept track of the Hamming distance between the current string and the satisfying assignment. We use a quantum version of this trick. The quantities we track are the expectation values of $\hat{S}$ and $\hat{S}^2$, where $\hat{S}$ is twice the total spin:

$$\hat{S} = \sum_{i=1}^n \sigma_i^z \quad \text{and} \quad \hat{S}^2 = \sum_{i,j=1}^n \sigma_i^z \sigma_j^z.$$  (4.8)

Note that $\hat{S}$ is closely related to the quantum Hamming weight operator $\sum_{i=1}^n \frac{1}{2} (1 - \sigma_i^z)$. 

95
Let $T^\dagger$ be the dual of $T$, so that
\[
\text{tr}[\hat{S}T(\rho)] = \text{tr}[T^\dagger(\hat{S})\rho] \quad \text{and} \quad \text{tr}[\hat{S}^2T(\rho)] = \text{tr}[T^\dagger(\hat{S}^2)\rho]. \tag{4.9}
\]

First consider
\[
T^\dagger_\alpha(\hat{S}) = (1-\Phi_\alpha)\hat{S}(1-\Phi_\alpha) + \frac{1}{2}\Phi_\alpha\Lambda_i(\hat{S})\Phi_\alpha + \frac{1}{2}\Phi_\alpha\Lambda_j(\hat{S})\Phi_\alpha, \tag{4.10}
\]
where $i, j$ are the two qubits where $\Phi_\alpha$ acts. Note that $\hat{S} - \sigma_i^x - \sigma_j^z$ is left invariant under the action of $T^\dagger_\alpha$, so
\[
T^\dagger_\alpha(\hat{S}) = \hat{S} - \sigma_i^x - \sigma_j^z + (1-\Phi_\alpha)(\sigma_i^x + \sigma_j^z)(1-\Phi_\alpha)
+ \frac{1}{2}\Phi_\alpha\Lambda_i(\sigma_i^x + \sigma_j^z)\Phi_\alpha + \frac{1}{2}\Phi_\alpha\Lambda_j(\sigma_i^x + \sigma_j^z)\Phi_\alpha. \tag{4.11}
\]

Due to the special properties of the restricted clauses, c.f. Eq. 4.2, we have
\[
\Phi_\alpha(\sigma_i^x + \sigma_j^z) = (\sigma_i^x + \sigma_j^z)\Phi_\alpha = 0, \tag{4.12}
\]
for all $\alpha$, which together with $\Lambda_i(\sigma_i^z) = 0$ and $\Lambda_i(\sigma_j^z) = \sigma_j^z$ for $i \neq j$ gives
\[
T^\dagger_\alpha(\hat{S}) = \hat{S}. \tag{4.13}
\]

This implies
\[
T^\dagger(\hat{S}) = \hat{S}, \tag{4.14}
\]
so we see that the expectation value of $\hat{S}$ is unchanged by the action of $T$ on a state:
\[
\text{tr}[\hat{S}T(\rho)] = \text{tr}[\hat{S}\rho]. \tag{4.15}
\]

Whereas the expectation value of $\hat{S}$ can not change under the action of $T$, the expectation value of $\hat{S}^2$ does change. Now $\Phi_\alpha$ acts on qubits $i$ and $j$, so we break up
\( \hat{S}^2 \) accordingly as

\[
\hat{S}^2 = \left[ \hat{S}^2 - 2\sigma_i^z\sigma_j^z - 2 \sum_{k \neq i,j} \sigma_k^z (\sigma_i^z + \sigma_j^z) \right] \\
+ \left[ 2\sigma_i^z\sigma_j^z + 2 \sum_{k \neq i,j} \sigma_k^z (\sigma_i^z + \sigma_j^z) \right].
\] (4.16)

\( T_\alpha^I \) leaves the first term unchanged since qubits \( i \) and \( j \) are not present. Now

\[
T_\alpha^I (\sigma_i^z\sigma_j^z) = (1-\Phi_\alpha)\sigma_i^z\sigma_j^z(1-\Phi_\alpha) + \frac{1}{2} \Phi_\alpha \Lambda_i (\sigma_i^z\sigma_j^z) \Phi_\alpha + \frac{1}{2} \Phi_\alpha \Lambda_j (\sigma_i^z\sigma_j^z) \Phi_\alpha.
\] (4.17)

Because of the special properties of the clauses, c.f. Eq. 4.2, in addition to the properties in Eq. 4.12, we have

\[
\Phi_\alpha \sigma_i^z\sigma_j^z = \sigma_i^z\sigma_j^z\Phi_\alpha = -\Phi_\alpha.
\] (4.18)

Since \( \Lambda_i (\sigma_i^z) = 0 \), we have

\[
T_\alpha^I (\sigma_i^z\sigma_j^z) = \sigma_i^z\sigma_j^z + \Phi_\alpha.
\] (4.19)

Now consider

\[
T_\alpha^I (\sigma_k^z (\sigma_i^z + \sigma_j^z)) = (1-\Phi_\alpha)\sigma_k^z (\sigma_i^z + \sigma_j^z)(1-\Phi_\alpha) \\
+ \frac{1}{2} \Phi_\alpha \Lambda_i (\sigma_k^z (\sigma_i^z + \sigma_j^z)) \Phi_\alpha \\
+ \frac{1}{2} \Phi_\alpha \Lambda_j (\sigma_k^z (\sigma_i^z + \sigma_j^z)) \Phi_\alpha \\
= \sigma_k^z (\sigma_i^z + \sigma_j^z).
\] (4.20)

where we have again used Eq. 4.12. Putting the pieces together gives

\[
T_\alpha^I (\hat{S}^2) = \hat{S}^2 + 2\Phi_\alpha.
\] (4.21)
The change in the expectation value of $\hat{S}^2$ after the action of $T$ is thus

$$\text{tr}[\hat{S}^2 T(\rho)] - \text{tr}[\hat{S}^2 \rho] = \frac{1}{L} \sum_{\alpha} 2 \text{tr}[\Phi_\alpha \rho] \geq 0.$$  (4.22)

We see that the expectation value of $\hat{S}^2$ always increases as long as $\rho_t$ is not annihilated by all of the $\{\Phi_\alpha\}$. The satisfying states $|0\rangle^{\otimes n}$ and $|1\rangle^{\otimes n}$ are the states that have the maximum expectation value of $\hat{S}^2$. As long as the system is not in the satisfying subspace, the expectation value of $\hat{S}^2$ increases, driving the system towards a mixture of the satisfying states $|0\rangle^{\otimes n}$ and $|1\rangle^{\otimes n}$. If there are other satisfying states besides $|0\rangle^{\otimes n}$ and $|1\rangle^{\otimes n}$, that can only help us, since we might end up in one of these additional satisfying states before reaching $|0\rangle^{\otimes n}$ or $|1\rangle^{\otimes n}$. We make this intuition regarding the algorithm precise in the following.

**Result 4.2.1.** Given clauses $\{\Phi_\alpha\}$ where $\Phi_\alpha = |\phi_\alpha\rangle\langle\phi_\alpha|$ are restricted as in Eq. 4.2, and $\epsilon$ is the size of the first non-zero eigenvalue of $H = \sum_\alpha \Phi_\alpha$ then for $T > \frac{n^2 L}{2(1-p)\epsilon}$, $\rho_T = T^T(\rho_0)$ has a fidelity with the ground state subspace that is bounded by $\text{tr}[\Pi_0 \rho_T] \geq p$.

**Proof by Contradiction:**

Suppose $\text{tr}[\Pi_0 \rho_T] < p$. From Eq. 4.7, $\text{tr}[\Pi_0 \rho_t]$ can not decrease with increasing $t$. So for all $t \leq T$,

$$\text{tr}[\Pi_0 \rho_t] < p$$  (4.23)

or equivalently,

$$\text{tr}[\Pi_0 \rho_t] > 1 - p.$$  (4.24)

Given that the spectral gap of $H$ is $\epsilon$, we have

$$\text{tr}[H \rho_t] > \epsilon \text{tr}[\Pi_0 \rho_t].$$  (4.25)
Combining Eqs. 4.24-4.25 gives

\[ \text{tr}[H \rho_t] > \epsilon(1 - p) \]  

(4.26)

for all \( t \leq T \).

Now \( \text{tr}[\hat{S}^2 \rho_0] \geq 0 \), for all \( \rho \) so

\[
\text{tr}[\hat{S}^2 \rho_T] = \sum_{t=0}^{T-1} \left( \text{tr}[\hat{S}^2 \rho_{t+1}] - \text{tr}[\hat{S}^2 \rho_t] \right) \\
= \sum_{t=0}^{T-1} \frac{1}{L} \sum_{\alpha} 2 \text{tr}[\Phi_\alpha \rho_t] \\
= \sum_{t=0}^{T-1} \frac{2}{L} \text{tr}[H \rho_t] \\
> \frac{(1 - p)2\epsilon T}{L} 
\]

(4.27)

Setting \( T \geq \frac{n^2 L}{2(1 - p)\epsilon} \) gives \( \text{tr}[\hat{S}^2 \rho_T] > n^2 \), a contradiction, since the maximum eigenvalue of \( \hat{S}^2 \) is \( n^2 \). Therefore, our assumption was incorrect, and for \( T \geq \frac{n^2 L}{2(1 - p)\epsilon} \), we must have \( \text{tr}[\Pi_0 \rho_T] \geq p \). \( \square \)

### 4.2.2 Solving the Quantum 2-SAT Decision Problem

We have shown with the restricted clause set, the Quantum Algorithm produces a state in the satisfying subspace with high probability in polynomial time, when the operator \( H \) is polynomially gapped (i.e. when \( \epsilon \) is not exponentially small in \( n \)). However, to solve Quantum 2-SAT we need to decide if there is a satisfying state or not. We will show that the quantum algorithm can also be used to solve this decision problem with high probability, by first running it for a fixed time \( T \) and then using a verification procedure. Several verification procedures are possible, including phase estimation on \( H \). However, the technique presented here is simple, and requires only local operations:

**Verification Procedure**

- Given some state \( \rho \), repeat \( K \) times, for \( K \) to be chosen later.
Choose $\alpha$ uniformly at random and measure $\Phi_\alpha$.

- If outcome 0 is obtained for all $K$ measurements, return PASS. Otherwise return FAIL.

**Result 4.2.2.** Given a YES instance of quantum $k$-SAT and a state $\rho$ with $\text{tr}[\Pi_0 \rho] \geq p$, or a NO instance with promise gap $c$ and a state $\rho$ with no restrictions, then

$$\mathbb{P}(\text{PASS}|\text{YES}) \geq p^K \quad \text{and} \quad \mathbb{P}(\text{PASS}|\text{NO}) \leq (1 - c/L)^K,$$

where $\mathbb{P}(\text{PASS}|\text{YES})$ is the probability that the Verification Procedure outputs PASS in the case of a YES instance, and $\mathbb{P}(\text{PASS}|\text{NO})$ is the probability that the procedure outputs PASS in the case of a NO instance.

**Proof.** We call $\rho_s$ the state of the system after $s$ measurements of the Verification Procedure have been made, so $\rho_0 = \rho$. We call outcome 0 a "good" outcome and outcome 1 a "bad" outcome.

Suppose we are in the YES case. We now show that a good outcome can only increase the probability that the system has overlap with the satisfying subspace. Let $\Phi_\alpha$ be the $s^{th}$ projector measured, and suppose that a good outcome is obtained. Then the state of the system after the measurement is

$$\rho_s = \frac{(I - \Phi_\alpha) \rho_{s-1} (I - \Phi_\alpha)}{\text{tr}[(I - \Phi_\alpha) \rho_{s-1}]}.$$  \hspace{1cm} (4.29)

We have

$$\text{tr}[\Pi_0 \rho_s] - \text{tr}[\Pi_0 \rho_{s-1}] = \frac{\text{tr}[(I - \Phi_\alpha) \rho_{s-1} (I - \Phi_\alpha) \Pi_0]}{\text{tr}[(I - \Phi_\alpha) \rho_{s-1}]} - \text{tr}[\Pi_0 \rho_{s-1}]$$

$$= \left( \frac{1}{\text{tr}[(I - \Phi_\alpha) \rho_{s-1}]} - 1 \right) \text{tr}[\Pi_0 \rho_{s-1}]$$

$$\geq 0.$$  \hspace{1cm} (4.30)

Since $\text{tr}[\Pi_0 \rho_s]$ is non-decreasing under the action of good outcomes, if only good outcomes occur for all times up to $K$, we have that $\text{tr}[\Pi_0 \rho_s] > p$ for all $0 \leq s \leq K$. 

100
The probability that the \( s \)th measurement has a bad outcome is
\[
\frac{1}{L} \sum_{\alpha} \text{tr}[\Phi_{\alpha} Q_{s-1}] = \frac{1}{L} \text{tr}[H Q_{s-1}].
\] (4.31)

Therefore, the probability that the \( s \)th outcome is good is
\[
1 - \frac{1}{L} \text{tr}[H Q_{s-1}].
\] (4.32)

The largest eigenvalue of \( H \) is upper bounded by \( L \), so
\[
1 - \frac{1}{L} \text{tr}[H Q_{s-1}] > 1 - \text{tr}[(I - \Pi_0) Q_{s-1}] = \text{tr}[\Pi_0 Q_{s-1}].
\] (4.33)

Since \( \text{tr}[\Pi_0 Q_s] > p \) for all \( 0 \leq s \leq K \), we have that the individual probability that any of the \( K \) measurements is good is at least \( p \). The probability that all \( K \) outcomes are good is \( \mathbb{P}(\text{PASS|YES}) > p^K \).

In the NO case, for any state \( Q_s \), the probability of a good outcome is
\[
1 - \frac{1}{L} \text{tr}[H Q_s] \leq 1 - c/L,
\] (4.34)
where \( c \) is the promise gap. Hence the probability of getting all good outcomes is \( \mathbb{P}(\text{PASS|NO}) \leq (1 - c/L)^K \). □

Since \( c = \Omega(\text{poly}(1/n)) \) and \( L = O(\text{poly}(n)) \), we have \( c/L = \Omega(\text{poly}(1/n)) \). Thus \( 1 - c/L \) can be close to 1, and we need to take \( K \) large if we want \( \mathbb{P}(\text{PASS|NO}) \) to be less than a fixed constant, say \( 1/3 \). Let
\[
K = \ln 3 \frac{L}{c}.
\] (4.35)
Then using a Taylor expansion

\[ (1 - c/L)^K = \left( (1 - c/L)^{L/c} \right)^{\ln 3} < e^{-\ln(3)} = 1/3. \]

(4.36)

We also want \( \mathbb{P}(PASS|YES) = p^K \) to be greater than some fixed constant. Let \( p = 1 - ac/L \) and choose \( a = 1 - \ln 2/\ln 3 \). One can numerically check that for \( 0 \leq c/L \leq 1 \)

\[ p^K = \left( (1 - ac/L)^{L/c} \right)^{\ln 3} \geq .6. \]

(4.37)

Thus for \( K = O(poly(n)) \) we have a constant separation between \( \mathbb{P}(PASS|NO) \) and \( \mathbb{P}(PASS|YES) \), which can then be amplified using standard techniques.

### 4.2.3 Difficulties with General Clauses

We have only been able to prove the Quantum Algorithm works in polynomial time when we restrict the form of the clauses and have a polynomial gap in the eigenspectrum of \( H \). In this section, we describe what breaks down when more general clauses are included in the instance.

The restricted clauses never cause the expectation of \( \hat{S}^2 \) to decrease. However, when we include clauses of the form

**General Clauses:**

\[ \Phi_\alpha = |\phi_\alpha \rangle \langle \phi_\alpha|, \quad \text{with} \quad |\phi_\alpha \rangle = a_\alpha |01\rangle_{i,j} + b_\alpha |10\rangle_{i,j} + c_\alpha |11\rangle_{i,j}, \]  

(4.38)

the expectation of \( \hat{S}^2 \) can either increase or decrease under the action of \( T_\alpha \), depending on the state of the system.

Consider a clause of the form \( \Phi_\alpha = |\phi_\alpha \rangle \langle \phi_\alpha| \) with \( |\phi_\alpha \rangle = |+1\rangle_{1,2} \) acting on the
state $\rho = |011\rangle\langle 011|_{1,2,3}$. One can easily check that

$$\text{tr}[^{2}\rho] = 5, \quad \text{tr}[^2\tau(\rho)] = 4.5,$$  \hspace{1cm} (4.39)

so the expectation value of $^2$ decreases.

When there are sufficiently many General Clauses, $|0\rangle^{\otimes n}$ is the only satisfying state, so one might guess that a good tracking measure would be the expectation value of $^\dagger$, which if it always increases, would bring the system closer and closer to $|0\rangle^{\otimes n}$. However, for General Clauses, $^\dagger$ can also increase or decrease, and in fact for $\rho$ and $\Phi_\alpha$ as above,

$$\text{tr}[^\dagger \rho] = 2, \quad \text{tr}[^\dagger \tau(\rho)] = 1.75.$$  \hspace{1cm} (4.40)

While in principle the expectation value $^\dagger$ and $^2$ under the action of $\tau$ can increase or decrease, in numerical experiments, we find that they always increase.

The analysis in Section 4.2 was simple because the changes in expectation value of $^\dagger$ and $^2$ did not depend on the details of the state of the system, but rather only on the overlap of the state with the satisfying subspace. With general clauses, the changes in expectation value of $^\dagger$ and $^2$ depend on the specifics of the state of the system, making these operators less useful as tracking devices.

### 4.3 Conclusions

We have shown that with a restricted clause set and a gapped problem Hamiltonian, the Quantum Algorithm will converge to a satisfying state with high probability in polynomial time. This is comparable to the performance of the Classical Algorithm for Classical 2-SAT. We also show how to verify whether the clauses permit a satisfying state.

The quantum algorithm described in this report can be used to solve Quantum $k$-SAT, but it is an open question what the runtime is for $k > 2$ or even for $k = 2$ with an unrestricted clause set. Schöning’s analysis showed that a version of the Classical
Algorithm solves Classical $k$-SAT faster than brute search, so we are curious whether the Quantum Algorithm may be able to solve Quantum $k$-SAT faster than Grover search. However, few tools exist to analyze the progress of such an algorithm, so in this report, we suggest tracking operators such as $\hat{S}$ and $\hat{S}^2$, which prove successful for analyzing the algorithm in the case of Quantum 2-SAT with restricted clauses.

4.4 Contributions

This work was done in collaboration with Edward Farhi and Kristan Temme. I am responsible for the idea of using a restricted clause set, and I devised the analysis for the quantum algorithm as well as the verification procedure.
Appendix A

Fault Trees and the Adversary
Upper Bound

A.1 Direct Functions and Monotonic Functions

From the definition of direct span programs, it is clear that direct functions are monotonic on any path from \( \bar{x}^* \) to \( x^* \). However, in this section we will prove that the converse is not true, and that there exist monotonic functions that are not direct functions.

**Theorem A.1.1.** Not all monotonic Boolean functions are direct functions.

**Proof.** We prove the result by creating a counterexample. Consider the following monotonic Boolean function of 4 variables:

\[
  f(x_1, x_2, x_3, x_4) = \begin{cases} 
0 & \text{if } (x_1, x_2, x_3, x_4) = \{(0, 0, 0, 0), (1, 0, 0, 0), (0, 1, 0, 0), (0, 0, 1, 0), (0, 0, 0, 1), (0, 1, 1, 0), (0, 0, 1, 1)\} \\
1 & \text{otherwise.} 
\end{cases}
\]  

(A.1)

One can easily verify that \( f \) is monotonic from input \((0, 0, 0, 0)\) to \((1, 1, 1, 1)\), but we will show that there is no direct span program that can represent \( f \), and therefore, \( f \)
is not a direct function.

Recall that to each input bit $x_j$ is assigned an input vector $\mu_j$. $\mu_j$ is associated with the single bit function $\chi_j$, as in Definition 2.4.1. We first show that for all input bits to $f$, $\chi_j(x_j) = x_j$. Suppose for contradiction that $\chi_j(x_j) = \bar{x}_j$. Consider the inputs $(0, 1, 0, 0)$ and $(1, 1, 0, 0)$. Since $f(0, 1, 0, 0) = 0$, using $\chi_j(x_j) = \bar{x}_j$, we have $\tau \notin \text{SPAN}\{\mu_1, \chi_2(1)\mu_1, \chi_3(0)\mu_3, \chi_4(0)\mu_4\}$. But since $f(1, 1, 0, 0) = 1$, we also have $\tau \in \text{SPAN}\{\chi_2(1)\mu_1, \chi_3(0)\mu_3, \chi_4(0)\mu_4\}$. It is impossible to take a vector away from a set and increase the span, so we have a contradiction. The same argument can be applied to each input bit in turn to show that for all input bits, $\chi_j(x_j) = x_j$.

Now consider $f(1, 1, 0, 0) = 1$ and $f(1, 0, 1, 0) = 1$. These two inputs require $\tau \in \text{SPAN}\{\mu_1, \mu_2\}$ and $\tau \in \text{SPAN}\{\mu_1, \mu_3\}$, or equivalently, there exist real numbers $a_2, b_2, a_3$ and $b_3$ such that

$$\mu_2 = a_2\tau + b_2\mu_1$$
$$\mu_3 = a_3\tau + b_3\mu_1.$$  \hspace{1cm} (A.2)

$f(0, 1, 1, 0) = 0$, so $\tau \notin \text{SPAN}\{\mu_2, \mu_3\}$. Since $\mu_2$ and $\mu_3$ are themselves linear combinations of $\tau$ and $\mu_1$, the only way $\tau$ will not be in their span is if $a_2 = a_3$, and $b_2 = b_3$. Thus $\mu_2 = \mu_3$.

Finally, consider $f(0, 0, 1, 1) = 0$, which implies $\tau \notin \text{SPAN}\{\mu_3, \mu_4\}$. Because $\mu_2 = \mu_3$, we have $\tau \notin \text{SPAN}\{\mu_2, \mu_4\}$. We also have $f(0, 1, 0, 1) = 1$, so this implies $\tau \in \text{SPAN}\{\mu_2, \mu_4\}$, giving a contradiction. \hfill $\square$

### A.2 Trivial Inputs to Direct Functions

**Theorem A.2.1.** Let $f$ be a direct boolean function, and let $P_f$ be a span program corresponding to $f$. Then we can create a new span program $P'_f$ that also represents $f$, but which is trivial on the inputs $x^*$ and $\bar{x}^*$.

**Proof.** If $P_f$ is our original span program with rows $r_i$ as in Definition 2.4.2, then we make a new span program $P'_f$ by changing $r_0$ to be orthogonal to all other $r_i$'s for
To do this, we let $R_1$ be the subspace spanned by the rows $r_i$, $i \geq 1$. Let $\Pi_{R_i}$ be the projector onto that subspace. Then we take $r_0 \to (I - \Pi_{R_1})r_0$. Now $r_0$ satisfies $r_i^\top r_0 = 0$ for $i > 0$. Looking at how we choose $w$ in Def. 2.4.2, this transformation does not affect our choice of $w$. (For $f(x) = 1$, $w$ has zero inner product with elements of the subspace spanned by the rows $r_i$, $i \geq 1$, so taking those parts out of $r_0$ preserves $w^\top r_0 = 1$, which is the main constraint on $w$. For $f(x) = 0$, $w$ is a sum of $r_i$ with $r_0$ having coefficient 1. But the part of $r_0$ that is not in the subspace $R_1$ will still have coefficient 1, and we are free to choose the coefficients of the $r_i$ ($i > 0$) terms to make up the rest of $w$ so that it is the same as before.) Hence there is no effect on the witness size or the function represented by the span program.

When $f(x) = 1$ and $\chi_j(x_j) = 1$ for all $j$, then $w = r_0/|r_0|$ is a valid $w$. (Although perhaps not the smallest possible $w$.) Likewise, when $f(x) = 0$ and $\chi_j(x_j) = 0$ for all $j$, $w = r_0$ is a valid $w$. (Although perhaps not the smallest possible $w$). Seting $|r_0| = 1$ by multiplying $r_0$ by a scalar, we can obtain both witness sizes equal to at most 1. This gives a span program for $f$ such that for $x^*$ and $\bar{x}^*$, $\text{WSIZE}(P_f, x) \leq 1$. \hfill $\Box$

We have shown that we can make a span program for $f$ with inputs $x^*$ and $\bar{x}^*$ trivial. However, the final scaling step that sets $|r_0| = 1$ may increase the witness size for other inputs (faults) compared to the original span program. Since we are limiting the number of faults, this doesn’t hurt our query complexity.

### A.3 Composition Proof

In this section, we will prove Lemma 2.2.2:

**Lemma 2.2.2.** For any Boolean function $f : S \to \{0, 1\}$ with $S \subseteq \{0, 1\}^n$ and natural number $k$, 

$$\text{ADV}^\pm(f^k) \geq (\text{ADV}^\pm(f))^k. \tag{2.1}$$

This proof follows Appendix C from Lee et al. [46] very closely, including most notation. The difference between this Lemma and that in [46] is that $f$ is allowed to be partial. We write out most of the proof again because it is subtle where the
partiality of \( f \) enters the proof, and to allow this appendix to be read without constant reference to [46].

First, we use an expression for the general adversary bound derived from the dual program of the general adversary bound:

\[
ADV^\pm(g) = \max_{W, \Omega \Omega J = \Omega} W \bullet J
\]
subject to \( W \circ G = 0 \)
\[
\Omega \pm W \circ \Delta_i \succeq 0
\]
\[
\text{Tr}(\Omega) = 1 \tag{A.3}
\]

where \( g : C \rightarrow \{0, 1\} \), with \( C \subseteq \{0, 1\}^m \) and all matrices are indexed by \( x, y \in C \), so e.g. \([W]_{xy}\) is the element of \( W \) in the row corresponding to input \( x \) and column corresponding to input \( y \). \( W \) can always be chosen to be symmetric. \( G \) satisfies \([G]_{xy} = \delta_{g(x),g(y)}\), and \( \Delta_i \) satisfies \([\Delta_i]_{xy} = 1 - \delta_{x_i, y_i}\), with \( x_i \) the value of the \( i \)th bit of the input \( x \). We call \( \Delta_i \) the filtering matrix. \( \succeq 0 \) means positive semidefinite, \( J \) is the all 1's matrix, and \( W \bullet J \) means take the sum of all elements of \( W \). When \( \circ \) is used between uppercase or Greek letters, it denotes Hadamard product, while between lowercase letters, it denotes composition.

We want to determine the adversary bound for a composed function \( f \circ g \) consisting of the functions \( g : C \rightarrow \{0, 1\} \) with \( C \subseteq \{0, 1\}^m \) and \( f : S \rightarrow \{0, 1\} \) with \( S \subseteq \{0, 1\}^n \). We consider the input to \( f \circ g \) to be a vector of inputs \( x = (x^1, \ldots, x^n) \) with \( x^i \in C \).

Given an input \( x \) to the composed function, we denote the input to the \( f \) part of the function as \( \tilde{x} : \tilde{x} = (g(x^1), \ldots, g(x^n)) \). Let \((W, \Omega)\) be an optimal solution for \( g \) with \( ADV^\pm(g) = d_g \) and \((V, \Lambda)\) be an optimal solution for \( f \) with \( ADV^\pm(f) = d_f \). To clarify the filtering matrices, we say \( \Delta^g_q \) is indexed by inputs to \( g \), \( \Delta^f_p \) is indexed by inputs to \( f \), and \( \Delta^{f \circ g}_{(p,q)} \) is indexed by inputs to the composed function \( f \circ g \). (So \( \Delta^{f \circ g}_{(p,q)} \) refers to the \((pm + q)\)th bit of the input string.)

We assume that the initial input \( x = (x^1, \ldots, x^n) \) is valid for the \( g \) part of the composition, i.e. \( x^i \in C \forall i \). A problem might arise if \( \tilde{x} \), the input to \( f \), is not an element of \( S \). This is an issue that Lee et al. do not have to deal with, but
which might affect the proof. Here we show that the proof goes through with small modifications.

The main new element we introduce is a set of primed matrices, which extend the matrices indexed by inputs to \( f \) to be indexed by all elements of \( \{0,1\}^n \), not just those in \( S \). For a primed matrix \( A' \), indexed by \( x, y \in \{0,1\}^n \), if \( x \notin S \) or \( y \notin S \), then \( [A']_{xy} = 0 \). We use similar notation for matrices indexed by \( x = (x^1, \ldots, x^n) \) where \( \bar{x} \in S \); we create primed matrices by extending the indeces to all inputs \( x \) by making those elements with \( \bar{x} \notin S \) have value 0. Notice if the extended matrices \( (W', \Omega') \) are a solution to the dual program, then the reduced matrices \( (W, \Omega) \) are also a solution.

For matrices \( A' \) indexed by \( \{0,1\}^n \), we define a new matrix \( \tilde{A}' \) indexed by \( C^n \), as \( [\tilde{A}]_{xy} = [A]_{\bar{x}y} \), where \( \bar{x} \) is the output of the \( g \) functions on the input \( x \), and likewise for \( \tilde{y} \). \( \tilde{A}' \) expands each element of \( A' \) into a block of elements.

Before we get to the main lemma, we will need a few other results:

**Lemma A.3.1.** [46] Let \( M' \) be a matrix labeled by \( x \in \{0,1\}^n \), and \( \tilde{M}' \) be defined as above. Then if \( M' \succeq 0 \), \( \tilde{M}' \succeq 0 \).

**Proof.** This claim is stated without proof in [46]. \( \tilde{M}' \) is created by turning all of the elements of \( M' \) into block matrices with repeated inputs. When an index \( x \in \{0,1\}^n \) is expanded to a block of \( k \) elements, there are \( k - 1 \) eigenstates of \( \tilde{M}' \) that only have nonzero elements on this block and that have eigenvalue 0. By considering all \( 2^n \) blocks (each element of \( \{0,1\}^n \) becomes a block) we obtain \( 2^n(k - 1) \) 0-valued eigenvectors. Next we use the eigenvectors \( \tilde{v}^i \) of \( M' \) to create new vectors \( \tilde{\tilde{v}}^i \) in the space of \( \tilde{M}' \). We give every element in the \( x^{th} \) block of \( \tilde{v}^i \) the value \( \tilde{v}^i(x)/k \), where \( \tilde{v}^i(x) \) is the \( x^{th} \) element of \( \tilde{v}^i \). The vectors \( \tilde{\tilde{v}}^i \) complete the basis with the 0-valued eigenvectors, and are orthogonal to the 0-valued vectors, but not to each other. However, the \( \tilde{\tilde{v}}^i \) have the property that \( \tilde{\tilde{v}}^T \tilde{M}' \tilde{\tilde{v}}^j = \delta_{ij} \lambda_i \) where \( \lambda_i \) is the eigenvalue of \( \tilde{\tilde{v}}^i \), so \( \lambda_i \geq 0 \). Thus using these vectors as a basis, we have that \( \tilde{u}^T \tilde{M}' \tilde{u} \geq 0 \) for all vectors \( \tilde{u} \).

The following is identical to Claim C.1 from [46] and follows because there is no restriction that \( g \) be a total function. Thus we state it without proof:
Lemma A.3.2. For a function \( g \), there is a solution to the dual program, \((W, \Omega)\), such that \( \text{ADV}^\pm(g) = d_g, d_g \Omega \pm W \geq 0 \), and \( \sum_{x:g(x) = 1} \Omega(x, x) = \sum_{x:g(x) = 0} \Omega(x, x) = 1/2 \).

In Lemma A.3.3, we will show that \( \text{ADV}^\pm(f \circ g) = \text{ADV}^\pm(f) \text{ADV}^\pm(g) \), which implies Lemma 2.2.2.

Lemma A.3.3. A solution to the dual program for \( f \circ g \) is \((U, T)\), where \((U', T') = (c \times \tilde{V}' \circ (d_g \Omega + W)^{\otimes n}, c \times d_g^{n-1} \tilde{A}' \circ \Omega^{\otimes n})\) and \( c = 2^n d_g^{(n-1)} \). \((U, T)\) give the adversary bound \( \text{ADV}^\pm(f \circ g) = d_g d_f \).

Proof. The first thing to check is that \( U' \) and \( T' \) are valid primed matrices, or otherwise we can not recover \( U \) and \( T \). Because each of \( U' \) and \( T' \) are formed by Hadamard products with primed matrices, they themselves are also primed matrices.

We next calculate the objective function, and afterwards check that \((U', T')\) satisfy the conditions of the dual program.

The objective function gives:

\[
J \cdot (c \tilde{V}' \circ (d_g \Omega + W)^{\otimes n}) = c \sum_{a,b \in S} [V]_{ab} \sum_{x,y} \prod_i (d_g[\Omega]_{x^i y^i} + [W]_{x^i y^i})
\]

\[
= c \sum_{a,b \in S} [V]_{ab} \prod_i \sum_{x^i y^i} (d_g[\Omega]_{x^i y^i} + [W]_{x^i y^i}) \tag{A.4}
\]

where in the first line we’ve replaced \( V' \) by \( V \) because adding extra 0’s does not affect the sum. In the second line, \( a_i \) and \( b_i \) are the \( i \)th bits of \( a \) and \( b \) respectively, and we’ve changed the order of multiplication and addition. This ordering change is not affected by the fact that \( f \) is partial, since the first summation already fixes an input to \( f \).

We now examine the sum

\[
\sum_{x^i y^i} (d_g[\Omega]_{x^i y^i} + [W]_{x^i y^i}). \tag{A.5}
\]
We consider the cases $a_i = b_i$, and $a_i \neq b_i$ separately. When $a_i = b_i$, because $W \circ G = 0$, we know that $[W]_{x^i y^i} = 0$, so in this case, only $[\Omega]_{x^i y^i}$ is non-zero. Since $\Omega$ is diagonal, it only has non-zero values when $x^i = y^i$, and using Lemma A.3.2, the sum is $d_g/2$. When $a_i \neq b_i$, then $x^i \neq y^i$, so $[\Omega]_{x^i y^i} = 0$. In this case, the sum will include exactly half of the elements of $W$: either those elements with $g(x^i) = 0$ and $g(y^i) = 1$, or with $g(x^i) = 1$ and $g(y^i) = 0$. Since $W$ is symmetric, this amounts to $\frac{1}{2} W \bullet J = d_g/2$. Multiplying $n$ times for the product over the $i$'s and using the definition of the objective function for $f$ gives the final result:

$$J \bullet (c \tilde{V}' \circ (d_g \Omega + W)^{\otimes n}) = c \times d_f \left( \frac{d_g}{2} \right)^n = d_f d_g$$

(A.6)

Now we show that $U'$ and $\Upsilon'$ satisfy the conditions of the dual program. We require that $[U']_{xy} = 0$ for $(f \circ g)(x) = (f \circ g)(y)$. Notice $U' = 0$ whenever $\tilde{V}' = 0$, and $[\tilde{V}]_{xy} = 0$ for $(f \circ g)(x) = (f \circ g)(y)$, so this requirement holds. Likewise $\Upsilon'$ is a diagonal matrix because it can only be non-zero where $\Omega^{\otimes n}$ is non-zero, and $\Omega^{\otimes n}$ is diagonal.

Next we will show that $\Upsilon' \pm U' \circ (\Delta'_{f, g})' \succeq 0$. From Lemma A.3.2 and from the conditions on the dual programs for $f$ and $g$, we have $d_g \Omega \pm W \succeq 0$, $\Omega \pm W \circ \Delta_q \succeq 0$, and $A' \pm V' \circ (\Delta'_p)' \succeq 0$. Then by Lemma A.3.1, $\tilde{A}' \pm \tilde{V}' \circ (\Delta'_p)' \succeq 0$. Since tensor and Hadamard products preserve semidefinite positivity, we get

$$0 \preceq (\tilde{A}' \pm \tilde{V}' \circ (\Delta'_p)') \circ \left( (d_g \Omega + W)^{\otimes (p-1)} \otimes (\Omega + W \circ \Delta_q) \otimes (d_g \Omega + W)^{\otimes (n-p)} \right),$$

(A.7)

where these matrices are indexed by all elements of $C^n$. $[W]_{x^i y^i} = 0$ for $x = y$ while $\tilde{A}'$ is only nonzero for elements $[\tilde{A}]_{xy}$ with $x = y$, so any terms involving a Hadamard of $W$ and $\tilde{A}'$ are 0. Similarly, the $\Omega$ in the $p^{th}$ tensor product is only nonzero for $x^p = y^p$, but for these inputs, the term $(\Delta'_p)'$ is always zero, so in fact the non-zero terms of this $\Omega$ do not contribute. Thus we are free to replace this $\Omega$ with $d_g \Omega \circ \Delta_q$. 

111
We obtain

\[ 0 \leq d_g^{n-1} \tilde{A}' \circ \Omega^\otimes n \pm (\tilde{V}' \circ (\tilde{A}'_p)'') \circ ((d_g \Omega + W)^{\otimes (p-1)} \otimes (d_g^{\ast} \circ \Delta_q^{\ast} + W \circ \Delta_q^{\ast}) \otimes (d_g \Omega + W)^{\otimes (n-p)}) \]

\[ 0 \leq d_g^{n-1} \tilde{A}' \circ \Omega^\otimes n \pm (\tilde{V}' \circ (\tilde{A}'_p)'') \circ ((d_g \Omega + W)^{\otimes n} \circ \{f^{\otimes (p-1)} \otimes \Delta_q^{\ast} \otimes f^{\otimes (n-p)} \}) \].

(A.8)

Finally, the term \((\tilde{A}'_p)''\) can be written as \(J - G\) acting on only the \(p\)th term in the tensor product \((d_g \Omega + W)^{\otimes n}\), so we need to evaluate \((J - G) \circ (d_g \Omega + W) \circ \Delta_q^{\ast}\). We have \((J - G) \circ \Omega = \Delta_q^{\ast} \circ \Omega = 0\), and \((J - G) \circ W = W\), so we can remove \((\tilde{A}'_p)''\) without altering the expression.

Now the term \(\{f^{\otimes (p-1)} \otimes \Delta_q^{\ast} \otimes f^{\otimes (n-p)}\}\) is almost \(\Delta^f_{(p,q)}\), except it is like a primed matrix; its indexes are all elements in \(C^n\), not just valid inputs to \(f\), yet it is not primed, in that some of its elements to non-valid inputs to \(f\) are non-zero. However it is involved in a Hadamard product with \(\tilde{V}'\), a primed matrix, so all of the terms corresponding to non-valid inputs are zeroed, and we can make it be a primed matrix without affecting the expression. We obtain

\[ 0 \leq d_g^{n-1} \tilde{A}' \circ \Omega^\otimes n \pm (\tilde{V}' \circ ((d_g \Omega + W)^{\otimes n} \circ (\Delta^f_{(p,q)})')) \],

(A.9)

which is precisely the positivity constraint of the dual program.

Finally, we need to check that \(\text{Tr}(c d_g^{n-1} \tilde{A}' \circ \Omega^\otimes n) = 1:\)

\[ \text{Tr}(c d_g^{n-1} \tilde{A}' \circ \Omega^\otimes n) = c d_g^{n-1} \sum_{a \in S} [\Lambda]_{aa} \sum_i [\Omega]_{x^i z^i} \]

\[ = c d_g^{n-1} \sum_{a \in S} [\Lambda]_{aa} \prod_i [\Omega]_{x^i z^i} \]

\[ = c d_g^{n-1} \left(\frac{1}{2}\right)^n = 1, \]

(A.10)

where all of the tricks here follow similarly from the discussion following Eq. A.4. □

Lemma 2.2.2 now follows from Lemma A.3.3 along with a simple inductive argument.
Appendix B

Robust Process Tomography

B.1 Unital Maps and the Linear Span Of Unitary Maps

The Pauli-Liouville representation is particularly convenient when discussing the Clifford group of n-qubit unitary maps, because, in this representation, such maps are monomial matrices [26, 25]. In the particular case of qubits, $E_{ij}^{(PL)} \in \{\pm 1, 0\}$ for a unitary in the Clifford group. Given these facts, we can now straightforwardly prove the result about the linear span of Clifford group maps on n qubits. First, we need to prove a small result about Clifford group unitaries.

Claim 1. For any two n-qubit Pauli operators $\hat{P}_i \neq 0$ and $\hat{P}_j \neq 0$, there exists a Clifford group unitary $\hat{C}$ such that $\hat{C}\hat{P}_i\hat{C}^\dagger = \hat{P}_j$.

Proof. This claim shows that there are no subsets of non-identity multi-qubit Pauli operators that do not mix under the action of the Clifford group. $\hat{P}_i$ and $\hat{P}_j$ can both be written as tensor products of single-qubit Pauli operators and identity operators $I$, where in the tensor product of each there is at least one element that is not $I$. Using local Clifford group unitaries one can take each non-identity element in each tensor product to the single-qubit Pauli operator $\sigma^x$. We call these new Pauli operators $\hat{P}_i'$ and $\hat{P}_j'$. Now the problem is equivalent to finding Clifford group unitaries that take one tensor product of $\sigma^x$ and $I$ to another.
Let $\text{CNOT}_{k,l}$ denote the controlled-not unitary with qubit $k$ as a control and qubit $l$ as a target. The CNOT is a well known unitary in the Clifford group with the property that $\text{CNOT}_{k,l} \sigma_i^x \text{CNOT}_{k,l} = \sigma_i^x \otimes \sigma_l^x$, where we use $\sigma_i^x$ to denote $\sigma^x$ acting on the $i^{th}$ qubit. In this way one can increase or decrease the number of $\sigma^x$'s in the tensor product decomposition of $\hat{P}_i'$ using unitary maps, as long as there is at least one $\sigma^x$ in the tensor product. This means that any tensor product of $\mathbb{I}$'s and $\sigma^x$'s on $n$ qubits can be mapped to any other tensor product of $\mathbb{I}$ and $\sigma^x$'s on $n$ qubits through the use of CNOT unitaries—in particular, one can map $\hat{P}_i'$ to $\hat{P}_j'$.

Now we can prove the intended result.

**Lemma 3.4.1.** The linear span of unitary maps coincides with the linear span of Clifford group unitary maps. Moreover, the projection of a TP map to this linear span is a unital map.

*Proof.* It suffices to show that any matrix element in the unital part of a map (in the Pauli-Liouville representation) can be written as a linear combination of Clifford group unitary maps.

The Pauli-Liouville representation of unitaries in the $n$-qubit Clifford group are monomial matrices with non-zero entries equal to $\pm 1$. For any given such unitary $\hat{C}$, one can construct $4^n$ orthogonal unitaries of the form $\hat{P}_i \hat{C}$, with corresponding $4^n$ mutually orthogonal Pauli-Liouville representation matrices. Pauli operators are diagonal in the Pauli-Liouville representation, so that for a fixed $\hat{C}$, the Pauli-Liouville representations of all $\hat{P}_i \hat{C}$ have support in the same set of $4^n$ matrix elements as the Pauli-Liouville representation of $\hat{C}$, and thus the values of any of these matrix elements for any map $\mathcal{E}$ can be recovered by collecting the Hilbert-Schmidt inner products between $\mathcal{E}^{(\text{PL})}$ and the Pauli-Louville representation of the map for the $\hat{P}_i \hat{C}$, i.e., $\text{tr} \mathcal{E}(\mathcal{P}_i \hat{C})^\dagger$. From Claim 1, one can choose a Clifford group unitary that has support on any particular matrix element in the unital block, therefore any unital matrix can be written as a linear combination of Clifford group unitary maps. Since Clifford group maps are unital, this concludes the proof. □
B.2 Reconstruction of the Unital Part with Imperfect Operations

In the main body of this paper we describe how RB allows for the reconstruction of the unital parts of $E \circ N$ and $N$, where $E$ is some quantum operations one would like to characterize, and $N$ is the error operation associated with each of the randomizing operations. We now prove the result which allows for the estimation of the unital part of $E$ alone, given an estimate of the unital part of $N$.

Lemma 3.4.2. If $(E \circ N)'$ is the unital part of $E \circ N$ and $N'$ is the unital part of $N$, and all these operations are trace preserving, then $E' = (E \circ N)' \circ (N')^{-1}$ whenever $(N')^{-1}$ exists.

Proof. Any trace-preserving linear map $A$ can be written in the Pauli-Liouville representation as

$$A^{(PL)} = \begin{pmatrix} 1 & 0^T \\ i_A & T_A \end{pmatrix},$$

where, as discussed previously, the unital part is

$$A^{(PL)} = \begin{pmatrix} 1 & 0^T \\ 0 & T_A \end{pmatrix}.$$  \hfill (B.2)

The Pauli-Liouville representation of the composition of two trace-preserving linear maps $A$ and $B$ is given by the multiplication of the Pauli-Liouville representations, resulting in

$$(A \circ B)^{(PL)} = \begin{pmatrix} 1 & 0^T \\ i_A + T_A i_B & T_A T_B \end{pmatrix},$$

\hfill (B.3)
and thus

$$(A \circ B)^{(PL)} = \begin{pmatrix}
1 & 0^T \\
0 & T_A T_B
\end{pmatrix},$$

$$(A \circ B)^{(PL)} = (A)^{(PL)} (B)^{(PL)}.$$  \hspace{1cm} (B.4)

It follows immediately that

$$(A)^{(PL)} = (A \circ B)^{(PL)} [(B)^{(PL)}]^{-1},$$

if the inverse exists, and

$$A' = (A \circ B') \circ (B')^{-1},$$ \hspace{1cm} (B.6)

by the Pauli-Liouville isomorphism. The lemma follows by setting $A = \mathcal{E} \circ \mathcal{N}$ and $B = \mathcal{N}$. \hfill \square

B.3 Complete-Positivity of the Projection of Single Qubit Operations onto the Unital Subspace

In this appendix, we prove that for a CPTP map $\mathcal{E}$ acting on a single qubit, $\mathcal{E}'$, the unital part of $\mathcal{E}$ (see Eq. 3.16), is always a CPTP map.

Recall that the Pauli-Liouville representation of a single qubit map $\mathcal{E}$ may be written as

$$\mathcal{E}^{(PL)} = \begin{pmatrix}
1 & 0 \\
\hat{i} \hat{\epsilon} & T_{\epsilon}
\end{pmatrix}. \hspace{1cm} (B.8)$$
King and Ruskai [42] show that there exist unitary maps $U$ and $V$ such that

$$U\mathcal{E}V = \mathcal{E}$$  \hspace{1cm} (B.9)

where

$$\mathcal{E}^{(PL)} = \left( \begin{array}{cc}
1 & 0 \\
\vec{\tau} & D
\end{array} \right) = \left( \begin{array}{cccc}
1 & 0 & 0 & 0 \\
\tau_1 & \lambda_1 & 0 & 0 \\
\tau_2 & 0 & \lambda_2 & 0 \\
\tau_3 & 0 & 0 & \lambda_3
\end{array} \right).$$ \hspace{1cm} (B.10)

To prove $\mathcal{E}'$ is CPTP, we first show that $\mathcal{E}'$ (the projection of $\mathcal{E}$ onto the unital block) is always CPTP, and then we prove that if $\mathcal{E}'$ is CPTP, $\mathcal{E}'$ is CPTP.

**Lemma B.3.1. For single qubit operations, $\mathcal{E}'$ is always CPTP**

**Proof.** Ruskai et al. [64] prove that $\mathcal{E}$ is CP if and only if the following three conditions are met:

$$\left( \lambda_1 + \lambda_2 \right)^2 \leq \left( 1 + \lambda_3 \right)^2 - \tau_3^2 - \left( \tau_1^2 + \tau_2^2 \right) \left( \frac{1 + \lambda_3 \pm \tau_3}{1 - \lambda_3 \pm \tau_3} \right)^2 \hspace{1cm} (B.11)$$

$$\left( \lambda_1 - \lambda_2 \right)^2 \leq \left( 1 - \lambda_3 \right)^2 - \tau_3^2 - \left( \tau_1^2 + \tau_2^2 \right) \left( \frac{1 - \lambda_3 \pm \tau_3}{1 + \lambda_3 \pm \tau_3} \right)^2 \hspace{1cm} (B.12)$$

$$\left( 1 - \left( \lambda_1^2 + \lambda_2^2 + \lambda_3^2 \right) - \left( \tau_1^2 + \tau_2^2 + \tau_3^2 \right) \right)^2 \geq \left( \tau_1^2 + \tau_2^2 + \tau_3^2 \right)^2 \hspace{1cm} (B.13)$$

where in Eq. B.11 and Eq. B.12, if $|\lambda_3| + |\tau_3| = 1$, then $\tau_1$ and $\tau_2$ must be 0 for the map to be CP.

To prove the lemma, we need to show that for any values of $\{\tau_i\}$ and $\{\lambda_i\}$ that satisfy Eqs. B.11-B.13, these equations are still satisfied when $\tau_i = 0$ for $i = 1, 2, 3$.

Notice that in Eqs. B.11-B.12 when we set $\tau_i = 0$, we require

$$\left( \lambda_1 \pm \lambda_2 \right)^2 \leq \left( 1 \pm \lambda_3 \right)^2. \hspace{1cm} (B.14)$$
However, it is easy to show [64] that Eqs. B.11-B.12 combined with positivity constraints imply

\[(\lambda_1 \pm \lambda_2)^2 \leq (1 \pm \lambda_3)^2, \quad \text{(B.15)}\]

so we only need to consider Eq. B.13.

Looking at Eq. B.13, we see that we need to show

\[
\begin{align*}
(1 - (\lambda_1^2 + \lambda_2^2 + \lambda_3^2))^2 & \geq \\
4((\lambda_1 \lambda_2)^2 + (\lambda_2 \lambda_3)^2 + (\lambda_3 \lambda_1)^2 - 2\lambda_1 \lambda_2 \lambda_3). \quad \text{(B.16)}
\end{align*}
\]

Because \(\tau\)'s only increase the value of the right hand side of Eq. B.13, we have

\[
\begin{align*}
(1 - (\lambda_1^2 + \lambda_2^2 + \lambda_3^2) - (\tau_1^2 + \tau_2^2 + \tau_3^2))^2 & \geq \\
4(\lambda_1^2 \lambda_2^2 + \lambda_2^2 \lambda_3^2 + \lambda_3^2 \lambda_1^2 - 2\lambda_1 \lambda_2 \lambda_3). \quad \text{(B.17)}
\end{align*}
\]

Note that if \((\lambda_1^2 + \lambda_2^2 + \lambda_3^2) + (\tau_1^2 + \tau_2^2 + \tau_3^2) \leq 1\), then we obtain

\[
\begin{align*}
(1 - (\lambda_1^2 + \lambda_2^2 + \lambda_3^2))^2 & \geq \\
(1 - (\lambda_1^2 + \lambda_2^2 + \lambda_3^2) - (\tau_1^2 + \tau_2^2 + \tau_3^2))^2, \quad \text{(B.18)}
\end{align*}
\]

which combined with Eq. B.17 gives the desired result.

To show that \((\lambda_1^2 + \lambda_2^2 + \lambda_3^2) + (\tau_1^2 + \tau_2^2 + \tau_3^2) \leq 1\), note that when \(\hat{\mathcal{E}}\) acts on the completely mixed state, for the outcome to be positive, we require \((\tau_1^2 + \tau_2^2 + \tau_3^2) \leq 1\). Furthermore, when \(\hat{\mathcal{E}}\) acts on one of the \(\sigma^z\) eigenstates, we obtain \(((\lambda_1 \pm \tau_1)^2 + \tau_2^2 + \tau_3^2) \leq 1\), where the \(\pm\) depends on which eigenstate is chosen and whether \(\tau_1\) and \(\lambda_1\) have the same sign. This implies that \((\lambda_1^2 + \tau_1^2 + \tau_2^2 + \tau_3^2) \leq 1\), and similarly for \(\lambda_2\) and \(\lambda_3\) (with the \(\sigma^x\) and \(\sigma^y\) eigenstates). Combining these inequalities and the positivity
Lemma 3.4.3. The unital part of a CPTP single-qubit map is always a CPTP map.

Proof. Lemma B.3.1 shows that the projection of $\mathcal{E}$ onto its unital part results in a CP map. So here we show this implies the projection of the map $\mathcal{E}$ onto its unital part results in a CP map.

Because $\mathcal{U}$ and $\mathcal{V}$ are unitaries, their Pauli-Liouville representations have the form

$$
\mathcal{U}^{(\text{PL})} = \begin{pmatrix} 1 & \bar{\alpha} \\ \bar{\alpha} & U \end{pmatrix}, \quad \mathcal{V}^{(\text{PL})} = \begin{pmatrix} 1 & \bar{\alpha} \\ \bar{\alpha} & V \end{pmatrix}.
$$

So

$$
\mathcal{E}^{(\text{PL})} = \mathcal{U}^{(\text{PL})} \mathcal{E}^{(\text{PL})} \mathcal{V}^{(\text{PL})} = \begin{pmatrix} 1 & \bar{\alpha} \\ \bar{\alpha} U^T UD V \end{pmatrix}.
$$

Now suppose $\mathcal{E}'^{(\text{PL})} = \begin{pmatrix} 1 & \bar{\alpha} \\ 0 & D \end{pmatrix}$ is a valid CP map. Then

$$
\mathcal{W}^{(\text{PL})} = \mathcal{U}^{(\text{PL})} \mathcal{E}'^{(\text{PL})} \mathcal{V}^{(\text{PL})} = \begin{pmatrix} 1 & \bar{\alpha} \\ 0 & UD V \end{pmatrix}
$$

is also a valid CPTP map because the composition of valid quantum maps is always a valid quantum map. However, by Eq. 3.16 $\mathcal{W}$ is equal to $\mathcal{E}'$, so the unital part of a single qubit map is always CPTP. \qed
B.4 Bounds on Fidelity

Recall that for an operation $\mathcal{E}$, the $\chi$-matrix representation is

$$\mathcal{E}(\hat{\rho}) = \sum_{i,j} \chi_{i,j}^\mathcal{E} \hat{P}_i \hat{P}_j.$$  \hfill (B.23)

Due to complete positivity constraints $\chi$ matrix elements satisfy

$$\chi_{i,j}^\mathcal{E} \leq \sqrt{\chi_{i,i}^\mathcal{E} \chi_{j,j}^\mathcal{E}}.$$  \hfill (B.24)

Composing two maps, their $\chi$-matrix representations compose as

$$A \circ B(\hat{\rho}) = \sum_{m,n,k,j} \chi_{m,n}^A \chi_{k,j}^B \hat{P}_m \hat{P}_k \hat{P}_j \hat{P}_n.$$  \hfill (B.25)

Let $\pi_i(m)$ be the index such that $\hat{P}_{\pi_i(m)} \hat{P}_m = \hat{P}_i$. Then using the fact that the absolute value is greater than the real or imaginary parts of a complex number, we obtain

$$\chi_{i,i}^{A \circ B} = \chi_{i,i}^A \chi_{0,0}^B \pm \left( 2 \sum_{m \neq 0} |\chi_{\pi_i(m),i}^A| |\chi_{m,0}^B| \right) + \sum_{m,n \neq 0} \left| \chi_{\pi_i(m),\pi_i(n)}^A |\chi_{m,n}^B| \right|.$$  \hfill (B.26)

Looking at the term $\sum_{m \neq 0} |\chi_{\pi_i(m),i}^A| |\chi_{m,0}^B|$ and using Eq. B.24 and the Cauchy-Schwarz inequality, we have

$$\sum_{m \neq 0} |\chi_{\pi_i(m),i}^A| |\chi_{m,0}^B| \leq \sqrt{\sum_{m \neq 0} |\chi_{\pi_i(m),i}^A|^2 \sum_{m \neq 0} |\chi_{m,0}^B|^2}$$

$$\leq \sqrt{\sum_{m \neq 0} \chi_{\pi_i(m),\pi_i(m)}^A \chi_{i,i}^A \sum_{m \neq 0} \chi_{m,m}^B \chi_{0,0}^B}$$

$$= \sqrt{(1 - \chi_{i,i}^A) \chi_{i,i}^A (1 - \chi_{0,0}^B) \chi_{0,0}^B}.$$  \hfill (B.27)
Similarly, the term $\sum_{m,n \neq 0} |\chi^A_{\pi_i(m),\pi_i(n)}| |\chi^B_{m,n}|$ gives

$$\sum_{m,n \neq 0} |\chi^A_{\pi_i(m),\pi_i(n)}| |\chi^B_{m,n}| \leq \sqrt{\sum_{m,n \neq 0} |\chi^A_{\pi_i(m),\pi_i(n)}|^2 \sum_{m,n \neq 0} |\chi^B_{m,n}|^2}$$

$$\leq \sqrt{\sum_{m,n \neq 0} \chi^A_{\pi_i(m),\pi_i(n)} \chi^A_{\pi_i(n),\pi_i(n)} \sum_{m,n \neq 0} \chi^B_{m,m} \chi^B_{n,n}} = (1 - \chi^A_{i,i})(1 - \chi^B_{0,0})$$  \hspace{1cm} (B.28)

So we have

$$\chi_{i,i}^{A \circ B} = \chi_{i,i}^A \chi_{0,0}^B \pm 2 \sqrt{(1 - \chi_{i,i}^A) \chi_{i,i}^A (1 - \chi_{0,0}^B) \chi_{0,0}^B} \pm (1 - \chi_{i,i}^A)(1 - \chi_{0,0}^B).$$  \hspace{1cm} (B.29)

Setting $i = 0$ gives the desired result.

To see why the lower bound is in general not tight, consider Eq. B.26. For the lower bound on $\chi_{i,i}^{A \circ B}$ we take all of the terms of the form $\chi^A_{\pi_i(m),\pi_i(n)} \chi^B_{m,n}$ and replace them with $-|\chi^A_{\pi_i(m),\pi_i(n)}| |\chi^B_{m,n}|$ because many of these terms have unknown phases, which in the worst case can have value $-1$. However, when $m = n$, because $\chi$ is positive semidefinite, we get terms of the form $\chi^A_{\pi_i(m),\pi_i(n)} \chi^B_{m,m} = |\chi^A_{\pi_i(m),\pi_i(m)}| |\chi^B_{m,m}|$, so we are subtracting terms which should actually be added. However, there is no way to address this issue without obtaining more information about the $\chi$ matrix.

### B.5 Ordering Of Error Maps

Throughout this paper, we chose to describe the noisy maps as the composition of the ideal map and some error map (applied in that order). That is, the noisy implementation of the map $C_i$ is expressed as

$$\mathcal{N}_i \circ C_i$$  \hspace{1cm} (B.30)
where $\mathcal{N}_i$ is the error map and $\mathcal{C}_i$ is the ideal Clifford map. This choice can be made without loss of generality, and has no effect on experimental observations. That is, we could instead express the implementation of the Clifford $\mathcal{C}_i$ as

$$\mathcal{C}_i \circ \mathcal{N}_i^*$$

where $\mathcal{N}_i^*$ is, in general, a different error map. The average fidelity of these noisy maps to $\mathcal{C}_i$ is the same, or, equivalently, $\mathcal{N}_i$ and $\mathcal{N}_i^*$ have the same average fidelity to the identity. However, in general $\mathcal{N}_i \neq \mathcal{N}_i^*$. Other process metrics are immune to this problem because they take the error to be additive rather than multiplicative, and so there is no ordering choice to be made or imposed.

If all error maps are identical for either of the conventional choices ($\mathcal{N}_i = \mathcal{N}$ or $\mathcal{N}_i^* = \mathcal{N}^*$) then Eq. 3.7 holds, and small deviations from these cases lead to perturbative corrections that generalize the results in Ref. [48, 49]. If the error maps are close to the identity, both perturbative models are likely to be valid, so $\mathcal{N} \approx \mathcal{N}^*$—the question of which convention is used become immaterial. However, if either $\mathcal{N}$ or $\mathcal{N}^*$ is far from the identity, low order perturbative expansions may not be valid for one of the conventions. Fits to individual RB decays cannot differentiate between these two cases, and the bounds used to isolate the error in $\mathcal{E}$ from $\mathcal{N}$ or $\mathcal{N}^*$ do not depend on this conventional choice, so as long as Eq. 3.7 holds for some separation of the error and ideal channel.

A problem arises when one attempts to use Lemma 3.4.2, as, unless $\mathcal{N} \approx \mathcal{N}^*$, the choice of conventions becomes important. The physical regime where, e.g., $\mathcal{N}_i \approx \mathcal{N}$ is precisely the regime where $\mathcal{N}_i \approx \mathcal{N}_i^* \approx \mathcal{I}$, and so this not not likely to be a problem in practice—within the accuracy of the perturbative expansions to Eq. 3.7, the inversion in Lemma 3.4.2 will be valid, as would a similar inversion taking the error map to occur before the ideal map.

In the more general formal cases where, e.g., $\mathcal{N}_i \approx \mathcal{N}$ but the $\mathcal{N}_i^*$ are very different from each other, there appears to be no way to choose the appropriate convention from individual observations. It may simply be the case that the $\mathcal{E}'$ reconstruction via
Lemma 3.4.2 using one convention is highly unphysical, while the other is physical, indicating which convention should be used. In the absence of this indication of systematic errors, however, one should report both reconstructions or simply choose the worst of the two.
Appendix C

Quantum 2-SAT with Less Restricted Clauses

C.1 Analysis with an Extended Clause Set

In Section 4.2, we showed that the Quantum Algorithm produces a satisfying state with high probability in polynomial time for Quantum 2-SAT if the clauses are of a certain form, which we now call Type I Clauses:

Type I Clauses:

$$\Phi_\alpha = |\phi_\alpha\rangle\langle\phi_\alpha|,$$

with

$$|\phi_\alpha\rangle = a_\alpha|01\rangle_i,j + b_\alpha|10\rangle_i,j.$$  \hfill (C.1)

In this appendix, we will show that the Quantum Algorithm converges in polynomial time for Quantum 2-SAT when all clauses are of Type I or Type II:

Type II Clauses:

$$\Phi_\alpha = |\phi_\alpha\rangle\langle\phi_\alpha|,$$

with

$$|\phi_\alpha\rangle = |11\rangle_i,j.$$  \hfill (C.2)

With Type I and Type II clauses, $|0\rangle^\otimes n$ is a satisfying state. As before, we also
have the condition that \( H = \sum_\alpha \Phi_\alpha \) must have a polynomial sized gap \( \epsilon \) between its 0-eigenvalue and the next largest eigenvalue.

In Section 4.2 we showed that for \( \Phi_\alpha \) a Type I clause,

\[
\begin{align*}
\text{tr}[\hat{S}\mathcal{T}_\alpha(\rho)] - \text{tr}[\hat{S}\rho] &= 0, \\
\text{tr}[\hat{S}^2\mathcal{T}_\alpha(\rho)] - \text{tr}[\hat{S}^2\rho] &= 2 \text{tr}[\Phi_\alpha \rho],
\end{align*}
\]

Next we observe that Type II clauses exhibit the following properties:

\[
\Phi_\alpha(\sigma_i^x + \sigma_j^z) = (\sigma_i^x + \sigma_j^z)\Phi_\alpha = -2\Phi_\alpha, \quad \Phi_\alpha\sigma_i^x\sigma_j^z = \sigma_i^x\sigma_j^z\Phi_\alpha = \Phi_\alpha. \tag{C.5}
\]

Applying Eq. C.5 to Eq. 4.11 and to the analysis in Eqs. 4.16-4.20, we have that for Type II clauses

\[
\begin{align*}
\mathcal{T}_\alpha^I(\hat{S}) &= \hat{S} + \Phi_\alpha, \\
\mathcal{T}_\alpha^I(\hat{S}^2) &= \hat{S}^2 - 2\Phi_\alpha + 2 \sum_{k \neq i, j} \sigma_k^z\Phi_\alpha.
\end{align*}
\]

Combining the effects of Type I and Type II clauses, we have

\[
\begin{align*}
\mathcal{T}^I(\hat{S}) &= \hat{S} + \frac{1}{L} \sum_{\alpha \in \text{Type I}} \Phi_\alpha, \\
\mathcal{T}^I(\hat{S}^2) &= \hat{S}^2 + \frac{2}{L} \sum_{\alpha \in \text{Type I}} \Phi_\alpha + \frac{2}{L} \sum_{\alpha \in \text{Type II}} \left( -\Phi_\alpha + \sum_{k \neq i, j} \sigma_k^z\Phi_\alpha \right). \tag{C.8}
\end{align*}
\]

When only Type I clauses are present, the expectation of \( \hat{S}^2 \) could only increase, but now Type II clauses can cause \( \hat{S}^2 \) to decrease. However, whenever \( \rho_t \) is not annihilated by all of the clauses, either the expectation value of \( \hat{S} \) increases (if a Type II clause is measured), or the expectation value of \( \hat{S}^2 \) increases (if a Type I clause is measured). Since \( |0\rangle^{\otimes n} \) has the maximum possible expectation value for both \( \hat{S} \) and \( \hat{S}^2 \), both of these effects work to drive the system into the satisfying state \( |0\rangle^{\otimes n} \). Of course if there are additional satisfying states besides \( |0\rangle^{\otimes n} \) this can only help us
because we might end up in a satisfying subspace without being driven there by the above effects.

**Result C.1.1.** Given clauses \( \{ \Phi_\alpha \} \), where \( \Phi_\alpha \) are Type I or Type II, and \( \epsilon \) is the size of the first non-zero eigenvalue of \( H = \sum_\alpha \Phi_\alpha \), then for \( T \geq \frac{2\ln^2}{2(1-p)\epsilon} \), \( \rho_T = T^T(\rho_0) \) will have fidelity with the ground state subspace that is bounded by \( \text{tr}[\Pi_0 \rho_T] \geq p \).

**PROOF BY CONTRADICTION:** Suppose \( \text{tr}[\Pi_0 \rho_T] < p \). As in Result 4.2.1, this implies \( \text{tr}[H \rho_t] > (1 - p)\epsilon \) for all \( t \leq T \). Now \( \text{tr}[\hat{S}_t^2 \rho_0] \geq 0 \), so

\[
\text{tr}[\hat{S}_t^2 \rho_T] \geq \sum_{i=0}^{T-1} \left( \text{tr}[\hat{S}_i^2 \rho_{i+1}] - \text{tr}[\hat{S}_i^2 \rho_i] \right)
= \sum_{i=0}^{T-1} \frac{2}{L} \left( \sum_{\alpha \in \text{Type I}} \text{tr}[\Phi_\alpha \rho_i] + \sum_{\alpha \in \text{Type II}} \text{tr} \left[ \left( -1 + \sum_{k \neq i,j} \sigma_k \right) \Phi_\alpha \rho_i \right] \right). \tag{C.10}
\]

In the Type II sum, the term \( -1 + \sum_{k \neq i,j} \sigma_k \) has eigenvalues that are larger than \(- (n-1)\), so using that \( \Phi_\alpha \) and \( -1 + \sum_{k \neq i,j} \sigma_k \) commute (they act on different qubits), we obtain

\[
\text{tr}[\hat{S}_t^2 \rho_T] \geq \frac{2}{L} \sum_{i=0}^{T-1} \left( \sum_{\alpha \in \text{Type I}} \text{tr}[\Phi_\alpha \rho_i] - (n-1) \sum_{\alpha \in \text{Type II}} \text{tr}[\Phi_\alpha \rho_i] \right). \tag{C.11}
\]

By our assumption, we have

\[
\sum_{\alpha \in \text{Type II}} \text{tr}[\Phi_\alpha \rho_i] + \sum_{\alpha \in \text{Type I}} \text{tr}[\Phi_\alpha \rho_i] > (1 - p)\epsilon, \tag{C.12}
\]

which implies

\[
\sum_{\alpha \in \text{Type I}} \text{tr}[\Phi_\alpha \rho_i] > (1 - p)\epsilon - \sum_{\alpha \in \text{Type II}} \text{tr}[\Phi_\alpha \rho_i]. \tag{C.13}
\]

127
Substituting this expression for Type I clauses into Eq. C.11, we have

\[
\text{tr}[\hat{S}^2 \rho_T] > \frac{T^2}{L} \sum_{i=0}^{T-1} \left( (1 - p) \epsilon - n \sum_{\alpha \in \text{Type II}} \text{tr}[\Phi_{\alpha} \rho_t] \right)
\]

\[
= \frac{T^2(1 - p) \epsilon}{L} - \frac{2n}{L} \sum_{t=0}^{T-1} \sum_{\alpha \in \text{Type II}} \text{tr}[\Phi_{\alpha} \rho_t].
\]  
(C.14)

From Eq. C.8 we have

\[
\sum_{t=0}^{T-1} \frac{1}{L} \sum_{\alpha \in \text{Type II}} \text{tr}[\Phi_{\alpha} \rho_t] = \sum_{t=0}^{T-1} (\text{tr}[\hat{S} \rho_{t+1}] - \text{tr}[\hat{S} \rho_t])
\]

\[
= \text{tr}[\hat{S} \rho_T] - \text{tr}[\hat{S} \rho_0]
\]

\[
\leq n.
\]  
(C.15)

Plugging Eq. C.15 into Eq. C.14, we obtain

\[
\text{tr}[\hat{S}^2 \rho_T] > \frac{T^2(1 - p) \epsilon}{L} - 2n^2.
\]  
(C.16)

Setting

\[
T \geq \frac{3Ln^2}{2(1 - p) \epsilon}
\]  
(C.17)

gives \(\text{tr}[\hat{S}^2 \rho_T] > n^2\), a contradiction \(\Box\)
Bibliography


133