Classical Computation in the Quantum World

by

Cupjin Huang

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Computer Science and Engineering)
in the University of Michigan
2019

Cupjin Huang
cupjinh@umich.edu
ORCID ID: 0000-0002-7466-8033

To my family

Acknowledgments

This thesis would not be possible without the influence of many people.
I am very fortunate to be advised by Professor Yaoyun Shi. Yaoyun has contributed a lot to the industrial effort for developing quantum computers, and that has influenced me a lot. I would like to thank Yaoyun for the great opportunities he has offered me at this special stage of quantum computation. I would also like to thank my lab mates Kevin Sung, Fang Zhang and Mike Newman. Special thanks to Mike Newman, who has been a great collaborator, and an oracle for English writing ever since my graduate program started.
For most of the year of 2018, I have been a research intern at Alibaba Quantum Lab (AQL). Although only at its nascent stage, AQL has already gathered many brilliant people, and I feel very lucky to have the chance to work with some of them. I would like to thank Jianxin Chen and Xun Gao for their advisory in both research and programming, and interns Rui Chao and David Ding for inspiring discussions inside and outside of work. I would like to thank in particular my supervisor during the internship, Professor Márió Szegedy. He has been an insightful mentor, a passionate colleague, and a personable friend. His advice and guidance have proved invaluable for my research career.
I learned much from discussion and collaboration with many people. Valuable experience has gained from working with my advisor and group members, my internship colleagues, my collaborators Yuxiang Yang, Giulio Chiribella and András Gilyén, and various others for insightful discussions. Special thanks to Professors Ken Brown at Duke University for the invitation to visit. I have learned a lot from one week of the visit to Ken's and Jungsang's groups at Duke. I would also like to thank the organizers and participants of Summer Cluster: Challenges in Quantum Computation at Simons Institute, Summer 2018. I have benefited a lot from the inspiring talks and discussions with an excellent group of people.
I am very grateful for my committee members Yaoyun, Márió, Chris Peikert and Kim Winick for their service. Also, I would like to thank the administrative staff in the CSE department, for their kind and timely assistance in my moments of need.
Most of the work covered in this thesis is supported by NSF and Alibaba Group USA.
Most of all, I would like to thank my family, especially my parents and my fiancé, for their constant and unconditional love and support.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF APPENDICES

ABSTRACT

Quantum computation is by far the most powerful computational model allowed by the laws of physics. By carefully manipulating microscopic systems governed by quantum mechanics, one can efficiently solve computational problems that may be classically intractable; on the contrary, such speed-ups are rarely possible without the help of classical computation, since most quantum algorithms heavily rely on subroutines that are purely classical. A better understanding of the relationship between classical and quantum computation is indispensable, in particular in an era where the first quantum device exceeding classical computational power is within reach.

In the first part of the thesis, we study some differences between classical and quantum computation. We first show that quantum cryptographic hashing is maximally resilient against classical leakage, a property beyond reach for any classical hash function. Next, we consider the limitation of strong (amplitude-wise) simulation of quantum computation. We prove an unconditional and explicit complexity lower bound for a category of simulations called monotone strong simulation, and further prove conditional complexity lower bounds for general strong simulation techniques. Both results indicate that strong simulation is fundamentally unscalable.

In the second part of the thesis, we propose classical algorithms that facilitate quantum computing. We propose a new classical algorithm for the synthesis of a quantum algorithm paradigm called quantum signal processing. Empirically, our algorithm demonstrates numerical stability and acceleration of more than one magnitude compared to state-of-the-art algorithms. Finally, we propose a randomized algorithm for transversally switching between arbitrary stabilizer quantum error-correcting codes. It has the property of preserving the code distance and thus might prove useful for designing fault-tolerant code-switching schemes.

CHAPTER 1

Introduction

1.1 Quantum computation in the NISQ era

History of quantum computation. The dream of harnessing quantum mechanics to solve difficult computational problems has emerged from the age of Feynman [1], where a prototype of a quantum computing device was proposed to simulate the intrinsically quantum-mechanical behavior of microscopic particles. Quantum computing has since been studied extensively. In the mid-1990s, two of the most famous quantum algorithms were proposed respectively by Shor and Grover, showing the ability to accelerate classical computing super-polynomially for factorization [2], and quadratically for black-box search [3]. The theory for designing and analyzing quantum algorithms has been further developed in the following two decades, and prospective quantum algorithms and heuristics are believed to enhance classical computation in various fields including discrete optimization [4], machine learning [5, 6], chemistry [7, 8], material science [9], just to name a few.

Recent development in quantum computational devices. Although quantum computing is theoretically promising, constructing a quantum computer in real life is very difficult. Since quantum phenomena are only noticeable within the range of nanometers and typically nanoseconds, controlling quantum systems accurately has been one of the most challenging tasks in modern physics. Fortunately, with research and industry effort from various fields including physics, computer science, electrical engineering and so on, performance of quantum computating hardware has been improved a lot during the last decades. We are now entering the era of Noisy Intermediate-Scale Quantum computation (NISQ). The concept of NISQ era was first proposed in [10] to refer to the near-term goal of quantum computational devices. Based on current technology, it is expected that within a couple of years people can build a quantum device containing 50-100 physical qubits, where applying each (two-qubit) quantum gate would introduce an error above the magnitude of 0.1%. With such a high noise rate, any existing error-correcting scheme would introduce more noise into the device, thus it is not yet feasible to perform error-free quantum computation on such devices. However, unlike previously existing quantum devices, it is not clear if a quantum device in the NISQ era can be efficiently simulated on a classical computer (the naive algorithm would take $2^{50} \sim 2^{100}$ time). Therefore, it is possible that such a quantum device can outperform current classical computational power, even only on some very specific computational tasks. Before usign general quantum computation to solve classically hard computational problems, researchers in the field would like to first demonstrate the potential superiority of quantum computation with respect to their classical counterpart, with as near-term quantum devices as possible. Such a landmark is called *quantum supremacy*.

Classical computation in the NISQ era. With Moore's law coming to an end, people start to seek for alternatives for classical computational architecture. Quantum computation provides a novel computational model based on quantum mechanics, and is expected to further enhance the state-of-the-art computational capability. Theoretically proving that general-purpose quantum computers are more capable than classical computers (i.e. $BQP \neq BPP$) would be impossible without a major breakthrough in complexity theory. Instead, people have proved advantage of quantum computation for several restricted models, such as query complexity [11], shallow circuit complexity [12], oracle complexity [13], just to name a few. Related to this, researchers in quantum information theory have also discovered many interesting phenomena that are exclusive to quantum information [14].

On the other hand, quantum computation has its own shortcomings. Performing quantum computation is currently prohibitively expensive due to technical limitations. To make use of quantum mechanics, the system carrying out the computation must be put in a state where the quantum effects are both noticeable and controllable. Prominent quantum computers need to be put at a close-to-zero temperature ($\sim$ 20mK) to maximally reduce the uncontrollable thermal fluctuation, while a sophisticated controlling mechanism needs to be designed in order to drive the evolution of the quantum system at will. Therefore, it is desirable that computational tasks be delegated to significantly cheaper classical computational resources if possible. With NISQ-era quantum devices, it is an increasingly important task to enhance quantum computing with classical computational resources.

Overview of the thesis. This thesis summarizes the theoretical research projects the author have conducted during the graduate program, with a focus on the connection between classical and quantum computation. The thesis can be divided into the following two parts.

The first part of the thesis investigates the limitations of classical computation compared to quantum computation, in both the contexts of quantum hash and of classical simulation. We study a property called resilience against classical leakage, which holds true for all quantum cryptographic hash functions, but is impossible to be satisfied by any classical hash. We then show explicit complexity lower bounds for strong(amplitude wise) classical simulation of quantum computation. We propose a broad category of strong simulation methods called *monotone methods*, which contains almost all prominent strong simulators. We prove unconditional lower bounds for monotone methods, and conditional lower bounds for general strong simulation methods based on well-believed conjectures.

The second part of the thesis concerns enhancing quantum computation with classical algorithms. The author will propose two classical algorithms. The first algorithm synthesizes a wide class of quantum algorithms called quantum signal processing in a numerically stable way. The second algorithm focuses on designing a switching scheme between quantum error correcting codes, such that the code distance is preserved throughout.

More specifically, we present the following results.

1.2 Overview of results

1.2.1 Resillience of quantum hashing against classical leakage

In Chapter 3, we study the quantum cryptographic hash functions for their resilience against classical leakage. Cryptographic hash functions are fundamental primitives widely used in practice. For such a function $f : \{0,1\}^n \to \{0,1\}^m$, it is nearly impossible for an adversary to produce the hash $f(x)$ without knowing the secret message $x \in \{0,1\}^n$. Unfortunately, all hash functions are vulnerable under the side-channel attack, meaning that an adversary can easily forge a hash when exposed to a small amount of information, without knowing the secret itself. This is because typically $m \ll n$ and an adversary needs only m bits of information to be able to forge a hash. This is a grave concern for information security in practice.

In sharp contrast, we show that when quantum states are used, the leakage allowed can be almost the entire secret. More precisely, we call a function that maps n bits to m qubits a quantum cryptographic function if the maximum fidelity between two distinct hashes is negligible in n. We show that for any $k = n - \omega(\log n)$, all quantum cryptographic hash functions remain cryptographically secure when leaking k bits of information. By the quantum fingerprinting constructions of Buhrman et al. [15], for all $m = \omega(\log n)$, there exist such quantum cryptographic hash functions. We also show that one only needs

$\omega(\log^2 n)$ qubits to verify a quantum cryptographic hash, rather than the whole classical information needed to generate one.

Our result also shows a big difference between the set of classical-quantum states with a certain min-entropy, and the set of joint distributions with the same min-entropy. This represents a significant barrier for proving quantum security of classical-proof extractors.

1.2.2 Limitations on strong simulation of quantum computation

In Chapter 4 and 5, we consider the problem of classical simulation of n-qubit quantum circuits. A quantum circuit is a sequence of elementary quantum evolutions that outputs a quantum state at the end. The quantum state is then measured to give a random classical outcome. A *weak* simulator is an algorithm that given an input quantum circuit C, samples from its output distribution. A *strong* simulator, on the other hand, calculates the probability of an outcome x given inputs C and x. It is known that weak simulation of general quantum circuits is BQP complete and strong simulation is $\#P$ complete. What is not known is an explicit lower bound on the complexity of simulations, or, the fundamental limit to which quantum computation is "unsimulable" than classical computational power.

We focus on strong simulation. In Chapter [16], we identify a subclass of strong simulators we call monotone. This subclass encompasses almost all prominent simulation techniques. We prove an *unconditional* (i.e. without relying on any complexity-theoretic assumptions) and *explicit* lower bound on the running time of simulators within this subclass.

In Chapter 5, we relax the constraint of monotonicity and consider hardness of general strong simulations. Based on the Strong Exponential Time Hypothesis (SETH), we remark that a universal simulator computing *any* amplitude to precision $2^{-n}/2$ must take at least $2^{n-o(n)}$ time. We then compare strong simulators to existing SAT solvers, and identify the time-complexity below which a strong simulator would improve on state-of-the-art SAT solving. Finally, we investigate Clifford+T quantum circuits with a small number of T-gates and identify time complexity lower bound in terms of T-gate count below which a strong simulator would improve on state-of-the-art 3-SAT solving.

1.2.3 Finding angle sequences in quantum signal processing

In Chapter 6, we investigate a paradigm for quantum algorithm design called *quantum signal processing* recently proposed in [17]. Quantum signal processing is a novel way of implementing transformations of eigenvalues of a given unitary operation, using only one auxiliary ancilla wire. An entire mathematical machinery has been developed to address this problem in a sequence of works [8, 18, 19, 20, 21, 22, 23]. Most of the quantum algorithms can be formulated as quantum signal processing (or more generally, quantum singular value transformation [23]) and it is known that QSP-based algorithms achieve the optimal scaling for various problems including Hamiltonian simulation, matrix inversion, just to name a few. However, within quantum signal processing, there is a classical preprocessing step of *finding angle sequences* that is yet to be made numerically stable, substantially undermining the efficiency of the QSP algorithm.

We describe an algorithm for finding angle sequences in quantum signal processing, with a novel component we call *halving* based on a new algebraic uniqueness theorem, and another we call *capitalization*. Together, these two algorithmic ideas allow us to find angle sequences for important applications such as Hamiltonian simulation in standard double precision arithmetic, native to almost all hardware. The current best method, proposed in [24], could find the same only in arbitrary precision arithmetic, which needed to be emulated by software, thus incurring a substantial time overhead. We present experimental results that demonstrate the performance of the new algorithm.

1.2.4 Transversal code switching for general stabilizer codes

In Chapter 7, we propose a randomized algorithm that switched between arbitrary stabilizer error correcting codes, while preserving the code distance with high probability.

Quantum error correcting codes encodes logical quantum information into physical quantum systems in a way that local physical error happening on a few physical qubits would not corrupt the logical state. Such quantum codes deals with faulty qubits; however, for sake of general fault-tolerant quantum computation, faulty physical gates must also be taken into account. It would be desirable that general quantum computation be implemented on a quantum error correcting code, in a way that each logical gate decomposes as a tensor product of local operations on each individual physical qubit. If one component in the physical realization of the logical gate is faulty, at most one physical qubits gets corrupted, and such error can still be remedied by the error-correcting procedure. However, a famous no-go theorem states that general quantum computation is not possible with only transversal gates on an error-correcting code [25].

Various methods have been proposed in order to get around the no-go result, among which code swiching is a popular candidate [26, 27, 28, 29, 30]. The idea of code switching is to choose two (or more) error-correcting codes such that the individual logical transversal gate sets form a universal gate set when combined together. Logical computation is then performed by switching back and forth between the codes if necessary, and apply transversal logical gates on the appropriate code. Such code switching procedures cannot be made using only transversal gates according to the Eastin-Knill theorem; however, it is still possible to switch between some pairs of codes fault-tolerantly, using a slight generalization of transversal gates called transversal measurment.

A recently proposed scheme, called stabilizer rewiring algorithm (SRA)[31], achieves switching between arbitrary pair of stabilizer codes, with only transversal gates and measurements. The central idea of the SRA scheme is to slowly deform the starting code to the target through a series of intermediate codes. However, the SRA scheme fails to guarantee that logical quantum information is protected by error correcting code throughout.

By introducing randomness and modifying the SRA scheme, we propose a randomized variant we call rSRA, that preserves the distance of intermediate codes with high probability. Though not necessarily resulting in an ideal deformation path, the rSRA algorithm derives its usefulness from its generality. For specific code switching examples, it may be profitable to modify the circuit using the rSRA as a template, augmented with a larger class of fault-tolerant manipulations such as local Clifford gates, in order to search for a fault-tolerant mapping.

1.3 Dissertation Outline

This dissertation is divided into eight chapters. Chapter 2 introduces the basics of quantum information. This is meant to be an overview of the background knowledge to better understand this thesis. For a more complete guide on the basics of quantum information, we recommend [32].

Chapter 3 introduces the quantum cryptographic hash functions and studies the property of resilience against classical leakage. Chapter 4 and Chapter 5 investigates the limitation of classical simulation of quantum circuits. Chapter 6 introduces a numerically stable algorithm for finding angles for quantum signal processing. Chapter 7 introduces a randomized algorithm for switching between different stabilizer codes while preserving the distance. Each of these chapters is prefaced by an introduction summarizing the background and results in the context of other work.

Finally, in Chapter 8 we summarize these results. We then give a broad overview of potential avenues for future work.

1.3.1 Works appearing

The work in Chapter 3 is contained almost entirely in [33]. This work was presented as a poster at the 20th Annual Conference on Quantum Information Processing and at the 7th International Conference on Quantum Cryptography by the author, and has been submitted for publication. It was awarded the best poster at the 7th International Conference on Quantum Cryptography.

The work in Chapter 4 and part of the work in Chapter 5 are drawn from [16], while the rest of Chapter 5 is drawn from [34]. Part of [16] was presented as a lightning talk at Simons Institute during Summer Cluster: Challenges in Quantum Computation. The two works have been combined together and submitted for publication. A result similar to one of the main results in [34] was shown concurrently in [35]. The two papers [16] and [34] have been combined and submitted for publication.

The work in Chapter 6 is ongoing at the time of this thesis, and will be found in [36]. It has been presented in part as a poster at the 22nd annual Conference on Quantum Information Processing by the author.

Finally, the work in Chapter 7 is contained entirely in [37]. It has been presented as a poster at the 21st annual Conference on Quantum Information Processing by the author.

In all of the above works, the author of this thesis proposal has appeared as either the first author or co-first author, but has benefited tremendously from discussions with his co-authors, supervisors, colleagues, and friends. Other works to which the author has contributed as a graduate student, but which do not fit into the theme of this thesis, can be found in [38, 39, 40, 37].

CHAPTER 2

Preliminaries

In this chapter, we give a cursory review of quantum information and computation. We will recall important definitions and theorems that are required for this thesis, without trying to give a complete overview. We assume that the readers are familiar with linear algebra. We refer to [32] for more detailed knowledge on quantum computation and quantum information. Furthermore, for the sake of clarity, chapter-specific preliminary knowledge will be placed at the beginning of each individual chapter.

2.1 Asymptotic notations

For two functions $f, g : \mathbb{Z}^+ \to \mathbb{R}^+$, we define the asymptotic notations as follows:

- $f(n) = O(g(n))$ if there exists $n_0 \in \mathbb{Z}^+$ and c, such that $f(n) \leq c \cdot g(n)$ for all $n \geq n_0$;
- $f(n) = \Omega(g(n))$ if there exists $n_0 \in \mathbb{Z}^+$ and c, such that $f(n) \geq c \cdot g(n)$ for all $n \geq n_0$;
- $f(n) = \Theta(g(n))$ if $f = O(g(n))$ and $f = \Omega(g(n))$;
- $f(n) = o(g(n))$ if for all n' and c there exists $n \geq n'$ such that $f(n) < c \cdot g(n)$;
- $f(n) = \omega(g(n))$ if for all n' and c there exists $n \geq n'$ such that $f(n) > c \cdot g(n)$.

We further denote

- $f(n) = poly(n)$ if $f(n) = O(n^c)$ for some c;
- $f(n) = poly\log(n)$ if $f(n) = poly(\log n)$;
- $f(n) = negl(n)$ if $f(n) = o(n^{-c})$ for all $c > 0$.

We finally define variants of the big-O notation that we will use throughout the thesis:

- $f(n) = \tilde{O}(g(n))$ if there exists $h(n) = poly\log(n)$ such that $f(n) = O(g(n) \cdot h(n))$;
- $f(n) = O^*(g(n))$ if there exists $h(n) = poly(n)$ such that $f(n) = O(g(n) \cdot h(n))$.

The $\tilde{O}$-notation and the O^*-notation simplify asymptotic expressions respectively when polylogarithmic factors and polynomial factors are not of major concern.

2.2 Matrix norms and functions

For a matrix $M \in \mathbb{C}^{m\times n}$, denote $M^\dagger$ to be the transposed conjugate of M. A matrix M always have a singular value decomposition (SVD):

$$M = U\Sigma V^\dagger, \tag{2.1}$$

where $U, V^\dagger$ are isometries ($U^\dagger U = I_r = V^\dagger V = I_r$, r being the rank of M), and Σ is a diagonal matrix with positive diagonal entries, called the singular values, which turn out to be independent from the choice of U and V. A norm defined on $\mathbb{R}^r$ can then be applied to the singular values of a matrix, thus defining a norm on the space of complex matrices:

1. The ℓ_1 norm of the singular values is called the *trace norm* of the matrix;
2. The ℓ_2 norm of the singular values is called the *Frobenius norm* of the matrix, and
3. The ℓ_∞ norm of the singular values is called the *operator norm* of the matrix, and
4. The ℓ_p norm of the singular values is called the *Schatten-p norm* of the matrix for $p \geq 1$.

A square matrix $H \in \mathbb{C}^{d\times d}$ is called hermitian if $H^\dagger = H$. For a hermitian matrix H, it is known that the spectrum of it consists of real numbers, i.e. H can be spectrally decomposed as

$$H = \sum_i \eta_i \alpha_i \alpha_i^\dagger, \eta_i \in \mathbb{R}, \tag{2.2}$$

where $\alpha_i^\dagger \alpha_j = \delta_{ij}$, i.e. the set of vectors $\{\alpha_i\}$ forms an orthonormal basis of the underlying Hilbert space $\mathbb{C}^n$. Each η_i is called an eigenvalue of H, with corresponding eigenvector α_i. The singular values of H is the absolute values of the eigenvalues.

When all the eigenvalues are non-negative, the matrix H is said to be positive semidefinite (PSD), denoted by $\mathcal{H} \succcurlyeq 0$. We write $\rho \succcurlyeq \sigma$ if $\rho - \sigma \succcurlyeq 0$; note that $\succcurlyeq$ is a partial order on the set of hermitian matrices.

Finally, functions defined $\mathbb{R}$ or $\mathbb{R}_{\geq 0}$ can be applied to hermitian or PSD matrices respectively, by simply transforming the eigenvalues according to the function.

2.3 Quantum states, measurements and channels

2.3.1 Pure quantum systems

Pure states. Let us begin with the most basic object in quantum information, i.e. a single qubit. While a classical bit takes the value from the finite set $\{0, 1\}$, the state of a pure qubit is formalized mathematically as a unit vector in the Hilbert space $\mathbb{C}^2$, i.e.

$$\psi = \alpha \cdot \begin{pmatrix} 1 \\ 0 \end{pmatrix} + \beta \cdot \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \alpha, \beta \in \mathbb{C}, |\alpha|^2 + |\beta|^2 = 1. \tag{2.3}$$

We adopt the Dirac notation, where a column vector representing a state ψ is written as $|\psi\rangle$. Eq.(2.3) can then be more concisely written as

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle, \tag{2.4}$$

where $\{|0\rangle, |1\rangle\}$ represents a fixed orthonormal basis of the Hilbert space $\mathbb{C}^2$. A row vector is represented as a *ket* in the Dirac notation. One convention of the Dirac notation is that the Hermitian conjugate of $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ is denoted as a *bra* $\langle\psi| := \alpha^*\langle 0| + \beta^*\langle 1|$, where α^*, β^* are conjugates of α, β respectively. The inner product of a bra $\langle\phi|$ and a ket $|\psi\rangle$ is then denoted $\langle\phi|\psi\rangle$. With this notation, the unit-length constraint $|\alpha|^2 + |\beta|^2 = 1$ can be concisely written as $\langle\psi|\psi\rangle = 1$.

A quantum state can of course be over a finite set $\{0, \cdots, d-1\}$; in this case the quantum system is called a d-qudit, and a pure d-qudit is represented as a unit vector in the Hilbert space $\mathbb{C}^d$ similar to a pure qubit. More generally we denote the quantum system to be a (finite) dimensional Hilbert space $\mathcal{H}$.

Unitary evolution. Quantum evolutions are linear automorphisms on the space of quantum states. Mathematically, a linear map $U : \mathcal{H} \to \mathcal{H}$ mapping unit vectors to unit vectors is called a *unitary*. The set of all d-qudit unitaries forms the unitary group $U(d)$, or $SU(d)$ by ignoring the global phase.

In some cases, we will also study mappings from a quantum system into another quantum system, where the two quantum systems do not necessarily have the same dimension. In this case the mapping is described by an *isometry* rather than a *unitary*.

Quantum measurement. A quantum measurement is represented as an orthonormal basis $M := \{|\phi_i\rangle\}$ of the quantum system $\mathcal{H}$. Upon a measurement M being applied to a quantum state $|\psi\rangle$, the state collapses to one of the states $|\phi_i\rangle$ randomly and generates the corresponding classical outcome i, where

$$\Pr[X = i] = |\langle\phi_i|\psi\rangle|^2. \tag{2.5}$$

One can verify that the values $\{|\langle\phi_i|\psi\rangle|^2\}_{0\le i<d}$ always represent a probability distribution, and is oblivious of the global phase of $|\psi\rangle$. Since the outcomes of the quantum measurements are the only information regarding the quantum process perceivable by classical beings, it is safe to discard the global phase factor.

2.3.2 Mixed quantum systems

Composite system quantum states. The joint pure state of two quantum systems $\mathcal{H}_A$ and $\mathcal{H}_B$ is described by a unit vector in the product system $\mathcal{H}_A \otimes \mathcal{H}_B$. However, it is not always true that such a vector can be described as tensor product of pure quantum states on individual quantum systems, a phenomenon called *quantum entanglement*.

Mixed quantum states. Given an entangled state $|\psi\rangle_{AB}$ on the composite system AB, there is still a way of defining the local quantum state of $|\psi\rangle$ on the subsystem A. To see this, consider a quantum measurement $\{|\phi_i\rangle \otimes |\sigma_j\rangle\}_{0\le i<\dim\mathcal{H}_A, 0\le j<\dim\mathcal{H}_B}$. The measurement results in a joint distribution $(X, Y) \in [\dim\mathcal{H}_A] \times [\dim\mathcal{H}_B]$. Consider the marginal distribution of X:

$$\begin{aligned}
\Pr[X = i] &= \sum_j \Pr[X = i, Y = j] && (2.6)\\
&= \sum_j |(\langle\phi_i| \otimes \langle\sigma_j|)|\psi\rangle|^2 && (2.7)\\
&= \sum_j \mathrm{Tr}[|\psi\rangle\langle\psi|(|\phi_i\rangle\langle\phi_i| \otimes |\sigma_j\rangle\langle\sigma_j|)] && (2.8)\\
&= \mathrm{Tr}[|\psi\rangle\langle\psi|(|\phi_i\rangle\langle\phi_i| \otimes \sum_j |\sigma_j\rangle\langle\sigma_j|)] && (2.9)\\
&= \mathrm{Tr}[|\psi\rangle\langle\psi|(|\phi_i\rangle\langle\phi_i| \otimes I)] && (2.10)\\
&= \mathrm{Tr}[\mathrm{Tr}_B[|\psi\rangle\langle\psi|]|\phi_i\rangle\langle\phi_i|]. && (2.11)
\end{aligned}$$

Here the partial trace Tr_B is a linear mapping defined by

$$\mathrm{Tr}_B[|i\rangle\langle j|_A \otimes |k\rangle\langle l|_B] = \langle l|k\rangle \cdot |i\rangle\langle j|. \tag{2.12}$$

It can be seen that the marginal distribution of the outcome does not depend on the measurement on subsystem B at all, and the outcome distribution of any local measurement is completely characterized by the unit-trace, positive semidefinite operator $\mathrm{Tr}_B[|\psi\rangle\langle\psi|]$ mapping from $\mathcal{H}_A$ to $\mathcal{H}_A$.

Denote the set of linear mappings from $\mathcal{H}_A$ to $\mathcal{H}_B$ as $L(\mathcal{H}_A, \mathcal{H}_\mathcal{B})$, with $L(\mathcal{H}) := L(\mathcal{H}, \mathcal{H})$. The state of a quantum system is in the most general case described by a *density operator*, i.e. a positive semidefinite matrix in $L(\mathcal{H})$ with unit trace. The set of all such operators is denoted as $\mathcal{S}(\mathcal{H})$. The density operator associated to a pure state $|\psi\rangle$ is simply the rank-1 matrix $|\psi\rangle\langle\psi|$. A density operator with rank greater than 1 cannot be described by a state vector, and is called a *mixed state*. For a quantum state ρ_{AB} of the joint system AB, we sometimes write $\rho_A := \mathrm{Tr}_B[\rho_{AB}]$ and $\rho_B := \mathrm{Tr}_A[\rho_{AB}]$ to denote the states on the subsystems A and B.

Purification. Any density operator ρ can always be written as $\rho = \sum_i p_i |\psi_i\rangle\langle\psi_i|$, where $p_i \geq 0, \forall i$ and $\sum_i p_i = 1$. This indicates that a density operator can be regarded as probabilistic mixtures of pure states. Another useful way to look at density operators is through *purification*. By spectral decomposition, there is a unique decomposition $\rho = \sum_{i=0}^{\dim\mathcal{H}-1} p_i|\psi_i\rangle\langle\psi_i|$ where $\{|\psi_i\rangle\}_{0\leq i<\dim\mathcal{H}}$ forms an orthonormal basis. Then it is easy to verify that

$$\rho = \mathrm{Tr}_B[|\phi\rangle\langle\phi|], |\phi\rangle = \sum_{i=0}^{\dim\mathcal{H}-1} \sqrt{p_i}|\psi_i\rangle \otimes |i\rangle_B, \tag{2.13}$$

where $\{|i\rangle_B\}_{0\leq i<\dim\mathcal{H}}$ is an arbitrary choice of orthonormal vectors in a reference system B. This indicates that a density operator can always be regarded as the marginal state of a pure state on a composite system, by introducing a (possibly imaginary) reference system B. The pure state $|\phi\rangle$ is called a *purification* of ρ. All purifications of ρ are equivalent under local unitaries on the reference system.

Classical-quantum states. A special type of quantum states are the classical quantum states, i.e. states of the form

$$\rho_{XE} = \sum_x |x\rangle\langle x| \otimes \rho_x. \tag{2.14}$$

It is called classical-quantum since the X part of the state is purely classical. In the case that the E subsystem is discarded, we recover a classical probability distribution

$$\rho_X = \mathrm{Tr}_E[\rho_{XE}] = \sum_x \mathrm{Tr}[\rho_x] \cdot |x\rangle\langle x|. \tag{2.15}$$

Quantum channels. Quantum channels are mixed state extensions of unitary evolutions in the pure state picture. A quantum channel is a linear mapping from a quantum system to possibly another quantum system, $\mathcal{C} : \mathcal{S}(\mathcal{H}_A) \to \mathcal{S}(\mathcal{H}_B)$. Mathematically, a quantum channel is a completely positive, trace-preserving (CPTP) map: for any reference system $\mathcal{H}_C$ and any joint state ρ on the composite system $\mathcal{H}_A \otimes \mathcal{H}_C$, the result state one gets by applying the channel $\mathcal{C}$ locally on the system $\mathcal{H}_A$ is still a quantum state, in the composite system $\mathcal{H}_B \otimes \mathcal{H}_C$.

Two most basic examples of quantum channels are isometries (unitaries) $\mathcal{V}(\cdot) = V \cdot V^\dagger$ and partial traces $\mathcal{T}(\cdot) = \mathrm{Tr}_B[\cdot]$. According to the famous Stinespring's dilation theorem, any quantum channel $\mathcal{C}$ can be written as an isometry followed by a partial trace, i.e. $\mathcal{C} = \mathcal{T} \circ \mathcal{V}$. The isometry $\mathcal{V}$ is called the *purification* of $\mathcal{C}$, and similar to the case with mixed states, all purifications of a quantum channel are equivalent under local unitaries on the reference system.

There are two common ways of representing a quantum channel, namely the Choi-Jamiołkowski representation and the Kraus operator decomposition.

Choi-Jamiołkowski representation maps a channel $\mathcal{C}_{A\to B}$ to the state

$$\rho_{\mathcal{C}} := (\mathcal{C}_{A\to B} \otimes \mathcal{I}_C)(|\Phi\rangle\langle\Phi|_{AC}), \tag{2.16}$$

where the system C is chosen with the same dimension as that of A, and

$$|\Phi\rangle_{AC} := \frac{1}{\sqrt{\dim \mathcal{H}_A}} \sum_i |i\rangle_A \otimes |i\rangle_B \tag{2.17}$$

is the so-called Bell state. The mapping $\mathcal{C}$ can then be conveniently written as

$$\mathcal{C}(\sigma) = \mathrm{Tr}_C[\rho_{\mathcal{C}}(I \otimes \sigma^T)]. \tag{2.18}$$

Kraus operator decomposition is to write a channel $\mathcal{C}$ in the form

$$\mathcal{C}(\cdot) := \sum_i A_i \cdot A_i^\dagger, \tag{2.19}$$

where the Kraus operators $\{A_i\}$ satisfy the equality $\sum_i A_i^\dagger A_i = I$. A mapping is CPTP if and only if it has a Kraus operator decomposition.

POVM measurements. The most general quantum measurements of quantum states are called positive-operator-valued measurements (POVMs). A POVM is a set of positive semidefinite operators $\{M_i\}_{0\le i<s}$ summing up to I. Applying a POVM on a quantum state ρ yields a random classical outcome X with probability specified By

$$\Pr[X=i] = \mathrm{Tr}[M_i\rho]. \tag{2.20}$$

2.3.3 Norms and metrics

Trace distance. Trace distance is a measure of distance between quantum states by the trace norm of the difference, i.e.

$$T(\rho,\sigma) := \frac{1}{2}\|\rho-\sigma\|_{tr} = \frac{1}{2}\mathrm{Tr}[\sqrt{(\rho-\sigma)^2}] = \max_{0\preccurlyeq M\preccurlyeq I}\mathrm{Tr}[M(\rho-\sigma)]. \tag{2.21}$$

Operationally, the trace distance is the maximum statistical distance between distributions generated by a single POVM applied to each individual state.

Fidelity. Fidelity is a measure of closeness between two quantum states. For two pure states $|\phi\rangle$ and $|\psi\rangle$, the fidelity is defined as

$$F(|\phi\rangle,|\psi\rangle) := |\langle\phi|\psi\rangle|. \tag{2.22}$$

More generally, the fidelity of two mixed state is defined as

$$F(\rho,\sigma) := \max_{|\rho\rangle,|\sigma\rangle} F(|\rho\rangle,|\sigma\rangle), \tag{2.23}$$

where $|\rho\rangle, |\sigma\rangle$ are taken over purifications of ρ and σ respectively. It can be proven that

$$F(\rho,\sigma) = \mathrm{Tr}[\sqrt{\sqrt{\rho}\sigma\sqrt{\rho}}]. \tag{2.24}$$

Fidelity is not a metric on the space of quantum states; in fact, $F(\rho,\sigma)=1$ if and only if $\rho=\sigma$. However it relates very nicely to the trace distance:

$$1-F(\rho,\sigma) \le T(\rho,\sigma) \le \sqrt{1-F^2(\rho,\sigma)}. \tag{2.25}$$

2.4 Universal gate sets

Throughout this section, we work in the pure quantum system picture, i.e. we only consider pure quantum states, unitary evolutions and projective measurements. Furthermore, we assume that the quantum system we work on is the system of n qubits for a certain integer n. The underlying Hilbert space is then $(\mathbb{C}^2)^{\otimes k} \cong \mathbb{C}^{2^k}$.

Although all unitaries are physically realizable in principle, for computational purposes there are unitaries that are easy to implement and ones that are difficult to implement.

Gates. We consider the implemetation of a unitary in the circuit model, i.e. we try to implement a unitary by composing a finite sequence of basic unitaries we call *gates*. The sequence of gates is called a *quantum circuit* and the length of the sequence is called the size of the quantum circuit. The set of gates we consider as the basic building blocks are called a gate set.

Examples of quantum gates. Usually, we restrict quantum gates to be unitary operations acting only on a constant number of qubits. This is a natural restriction to the current quantum architecture in that one can only hope to accurately manipulate a constant number of qubits in unit time. Here we give some common example of quantum gates:

- Pauli gates: the single qubit gates

$$\mathrm{X} := \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \mathrm{Y} := \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}, \mathrm{Z} := \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \tag{2.26}$$

  form the single qubit Pauli group $\mathcal{P} = \{\pm 1, \pm i\} \times \{I, \mathrm{X}, \mathrm{Y}, \mathrm{Z}\}$. More generally, the Pauli group on n qubits $\mathcal{P}^n$ is defined by $\mathcal{P}^n = \mathcal{P}^{\otimes n}$, and its generators can be chosen to be X and Z gates on each individual qubit.

- Clifford gates: the Hadamard gate H, phase gate S and controlled-not gate CNOT are defined as follows:

$$\mathrm{H} := \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}, \mathrm{S} := \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix}, \mathrm{CNOT} := \begin{pmatrix} 1 & & & \\ & 1 & & \\ & & & 1 \\ & & 1 & \end{pmatrix}. \tag{2.27}$$

  These gates acting on all qubits (CNOT on all pairs of qubits) generate the *Clifford* *group* $\mathcal{C}_n$. It is the normalizer of the Pauli group $\mathcal{P}^n$ in $U(2^n)$, i.e.

$$\mathcal{C}_n := \{U \in U(2^n) | U\mathcal{P}_n U^\dagger = \mathcal{P}_n\}. \tag{2.28}$$

- Toffoli gate: The Toffoli gate is a gate on three qubits:

$$\mathrm{TOFFOLI} := \begin{pmatrix} 1 &&&&&&& \\ & 1 &&&&&& \\ && 1 &&&&& \\ &&& 1 &&&& \\ &&&& 1 &&& \\ &&&&& 1 && \\ &&&&&&& 1 \\ &&&&&& 1 & \end{pmatrix} \tag{2.29}$$

- T gate: The T gate is a single qubit gate

$$\mathrm{T} := \begin{pmatrix} 1 & 0 \\ 0 & \exp\{i\pi/4\} \end{pmatrix}. \tag{2.30}$$

Universal gate set. Since the set of all unitaries on $\mathbb{C}^{2^k}$ forms a continuous Lie group, it is not possible that all unitaries be generated as finite sequences drawn from a finite gate set. Instead, we define a gate set to be *universal* if the group of unitaries generated by a gate set is a dense subgroup of $SU(2^k)$. Solovay-Kitaev theorem shows that the length of the sequence for approximating an arbitrary unitary to ϵ precision scales polylogrithmically with respect to $1/\epsilon$. It also follows that all universal gate sets are equivalent in the sense that any circuit in one gate set can be approximated using another gate set with at most polynomial overhead.

The following are two examples of universal gate sets :

- $\{\mathrm{H}, \mathrm{S}, \mathrm{CNOT}, \mathrm{T}\}$. The gates H, S and CNOT generate the Clifford groupand the famous Gottesman-Knill theorem [41] states that such a Clifford circuit can be simulated in polynomial time classically. The T gate introduces non-Cliffordness into the gate set and they together generate a dense subgroup of $SU(2^n)$.

- $\{\mathrm{X}, \mathrm{CNOT}, \mathrm{TOFFOLI}, \mathrm{H}\}$. The gates $\mathrm{X}, \mathrm{CNOT}$ and $\mathrm{TOFFOLI}$ form a universal gate set for classical reversible computation, in that it always maps one computational basis state onto another, and all permutations of computational basis states can be implemented using $\mathrm{X}, \mathrm{CNOT}$ and $\mathrm{TOFFOLI}$, potentially with help of ancilla qubits. Together with the Hadamard gate H, this gate set generates a dense subgroup of $SO(2^n)$. Although not universal in the most general sense, this gate set is universal in that there is an efficient way of converting a polynomial-sized quantum circuit into one consisting of only $\mathrm{X}, \mathrm{CNOT}, \mathrm{TOFFOLI}$ and H, such that the outcome distribution is arbitrarily well approximated.

CHAPTER 3

Resilience of quantum hash against classical leakage

In this chapter, we investigate the differences between classical and quantum hash functions in the setting of the side-channel attack.

Cryptographic hash functions are fundamental primitives widely used in practice. For such a function $f : \{0,1\}^n \to \{0,1\}^m$, it is nearly impossible for an adversary to produce the hash $f(x)$ without knowing the secret message $x \in \{0,1\}^n$. Unfortunately, all hash functions are vulnerable under the side-channel attack, which is a grave concern for information security in practice. This is because typically $m \ll n$ and an adversary needs only m bits of information to be able to forge the hash.

In sharp contrast, we show that when quantum states are used, the leakage allowed can be almost the entire secret. More precisely, we call a function that maps n bits to m qubits a quantum cryptographic hash function if the maximum fidelity between two distinct hashes is negligible in n. We show that for any $k = n - \omega(\log n)$, all quantum cryptographic hash functions remain cryptographically secure when leaking k bits of information. By the quantum fingerprinting constructions of Buhrman et al. [15], for all $m = \omega(\log n)$, there exist such quantum cryptographic hash functions. We also show that one only needs $\omega(\log^2 n)$ qubits to verify a quantum cryptographic hash, rather than the entire classical information needed to generate one.

Our result also shows that to approximately produce a small amount of quantum side information on a classical secret, it may require almost the full information of the secret. This large gap represents a significant barrier for proving quantum security of classical-proof extractors.

3.1 Introduction

3.1.1 The problem and the motivation

Cryptographic hash functions are a fundamental primitive used widely in today's cryptographic systems. They are considered "workhorses of modern cryptography".[1] For simplicity, we focus our discussions on *keyless* (cryptographic) hash functions, each of which is an efficiently computable function h from some message space $\mathcal{M}$ to some *digest space* $\mathcal{T}$ [42, 43, 44]. Ideally, we want the hash function to have the following properties. First, the digest should be much shorter than the message. Depending on applications, the following security properties are desirable [44]. (1) *Collision resistant*: It is computationally infeasible to find any "collision", i.e., two distinct messages x and x', such that $h(x) = h(x')$. (2) *Preimage Resistance*: It is computationally infeasible to invert h. (3) *Second Preimage Resistance*: Given any message x, it should be computationally infeasible to find $x' \neq x$ with $h(x') = h(x)$.

Prominent examples of widely used cryptographic hash functions include SHA-256 and SHA-512, part of the SHA-2 algorithms that were designed by NSA and are US Federal Standards. These algorithms are used in UNIX and LINUX for secure password hashing, in Bitcoin for proof-of-work. As a motivating example, we consider how proof-of-work can be carried out through a hash. Suppose that Alice receives a trove of valuable documents $x \in \{0,1\}^n$, and Bob claims that he was the person producing and sending it. To prove his claim, he sends Alice a tag $t \in \{0,1\}^m$, which supposedly is the result of applying a cryptographic hash function $h : \{0,1\}^n \to \{0,1\}^m$ on x. Alice simply checks if $t = h(x)$. Accept if yes, reject otherwise. By the collision resistance property, it is nearly impossible that Bob can produce $h(x)$ without knowing x.

In practice, there may be information leakage of the message over time due to information transmission, adversarial attacks, etc. Therefore, it is rather desirable if the hash function is resilient against information leakage. We ask: how many bits ℓ about the message x can be leaked before the adversary is able to forge the tag $h(x)$ easily?

Cleary, $\ell \leq m$, since if the tag $h(x)$ itself is known to the adversary, he does not need to know more about x to pass the verification. This is rather disappointing, since m is typically much smaller than n. We then ask: what if a quantum tag is used instead? If the leakage is quantum, by the same reasoning, m remains a trivial and rather low upper-bound on ℓ. This leads us to our central question: *Can a quantum hash function be much more resilient to* classical *leakage?*

[1] Bob Schneier, `https://www.schneier.com/essays/archives/2004/08/cryptanalysis_of_md5.html`.

3.1.2 Quantum cryptographic hash functions

By a "quantum hash function," we simply mean a classical-to-quantum encoding $\phi : \{0,1\}^n \to \mathbb{C}^{2^m}$ that maps $x \in \{0,1\}^n$ to a pure m-qubit state $|\phi_x\rangle$. In a seminal paper, Buhrman et al. [15] introduced the notion of *quantum fingerprinting*. In their most general form, a quantum fingerprinting is the following.

Definition 3.1 (Generalized Quantum Fingerprinting (Buhrman et al. [15])). *A function $\phi : \{0,1\}^n \to \mathbb{C}^{2^m}$ is a (n, m, δ) (generalized) quantum fingerprinting where*

$$\delta := \max_{x,x':x\neq x'} |\langle\phi_x|\phi_{x'}\rangle| \,.$$

We use the convention that $\phi := |\phi\rangle\langle\phi|$ represent the projector for the pure state $|\phi\rangle$. If one replaces the predicate $h(x) = h(x')$ by the fidelity $F(\phi_x, \phi_{x'}) = |\langle\phi_x|\phi_{x'}\rangle|$, one sees that δ precisely quantifies the extent of collision resistance. For concreteness, we define what we mean by quantum cryptographic hash function as follows. For a function $\delta_n \in (0,1)$, we say δ_n is negligible in n if $\delta_n \leq 1/n^c$ for all $c > 0$ and all sufficiently large n.

Definition 3.2 (Quantum cryptographic hash function). *A (n, m, δ)-quantum fingerprinting ϕ is called a* quantum cryptographic hash function *if $\delta = \mathrm{negl}(n)$.*

We note that while classical cryptographic hash functions necessarily rely on computational assumptions for security, their quantum counterparts can achieve the three security properties (1-3) information-theoretically. We now proceed to formulate our leakage problem precisely. We consider average-case security and model classical side-channel information using a joint distribution of random variables XY called the *side information state*, $\eta_{XY} := \sum_{x,y} p_{x,y}|x\rangle\langle x| \otimes |y\rangle\langle y|$ on $\{0,1\}^n \times \{0,1\}^{n'}$. Here X represents the input message to be hashed and is uniformly distributed, and Y represents the side information. The largest probability of correctly guessing x conditioned on y is $p_g := p_g(X|Y)_\eta := \sum_y \max_x p_{x,y}$. The *conditional min-entropy* is $H_{\min}(X|Y)_\eta := -\log p_g(X|Y)_\eta$. We quantify the amount of leakage by $k := n - H_{\min}(X|Y)_\eta$.

The adversary is given the Y sub-system and creates a classical-quantum- (cq-) state ρ_{XE}, called the *forgery state*, through local quantum operations mapping the classical side information Y to the quantum side information E. The *verification scheme* for ϕ is the following measurement on the joint state XE: $V := \sum_x |x\rangle\langle x| \otimes \phi_x$. The probability of the forgery state to pass the verification scheme is then $e_s := \mathrm{Tr}(V\rho)$. Given the leakage ℓ, the optimal passing probability of a forgery state is denoted by $e_s^*(\ell)$ as a function of the leakage ℓ. In this setting, it is clear that $e_s^*(n) = 1$, since a party knowing the entire secret message can pass the test with probability 1. We can now define security precisely.

Definition 3.3 (Resilience against classical leakage)**.** *A (n, m, δ)-quantum cryptographic hash function ϕ is said to be σ-*resilient against ℓ bits of classical leakage *if for all forgery state ρ obtained from ℓ bits of side information, the probability of passing the verification scheme $e_s^*(\ell) \leq \sigma$. If no σ is specified, it is assumed that $\sigma = \mathrm{negl}(n)$.*

3.1.3 Main result

We show that quantum cryptographic hash functions can be extremely resilient to classical leakage. Our main theorem is informally stated below.

Theorem 3.1 (Main Theorem; Theorem 3.4)**.** *For all n and $k = n - \omega(\log n)$, all quantum cryptographic hash functions acting on n bits are resilient against k bits of classical leakage.*

Buhrman et al. [15] showed that for all n and $\delta \in (0, 1)$, there exists a (n, m, δ) quantum fingerprinting for $m = \log n + O(\log 1/\delta)$ for which explicit constructions can be derived from [45]. We thus have the following corollary.

Corollary 3.1. *For all n, $k = n - \omega(\log n)$, and $m = \omega(\log n)$, there exist efficient quantum cryptographic hash functions resilient to leaking k bits of information.*

One drawback of the verification scheme is that the verifier has to get access to full information about the original message X in order to perform the verification. In some cases, this would be a heavy burden on the verifier. One natural question to ask is that if it is possible to develop a lightweight verification scheme where the verifier does not need to read the whole message. More formally, let the verifier now receive k qubits of *advice state* and m bits of the *forgery state* provided by the adversary. An (n, k, m) verification scheme V would then be a joint measurement on the advice state together with the forgery state. This generalizes the original verification where $k = n$ and $V = \sum_x |x\rangle\langle x| \otimes \phi_x$.

Out next result shows that by increasing the hash a little bit we can dramatically reduce the size of the system needed by the verifier:

Theorem 3.2 (Theorem 3.5, informally stated)**.** *For all n, fix $k = m = \omega(\log^2 n), \ell \leq n - \omega(\log n)$. There exists a verification scheme V acting on $k + m$ qubits, together with an ensemble of k-qubit states $\{\rho_x\}$ such that*

Soundness: *No adversary with less than ℓ bits of classical side information can pass the test with non-negligible probability, i.e.,*

$$\sup_{Y: n-H_{\min}(X|Y)_\sigma \le \ell} \sup_{\{\sigma_Y\}} \mathbb{E}_{XY} \mathrm{Tr}[V(\rho_X \otimes \sigma_Y)] \le \mathrm{negl}(n). \tag{3.1}$$

Completeness: *Anyone knowing full information of the secret message x can pass the test with probability* 1*, i.e.,*

$$\forall x, \mathrm{Tr}[V(\rho_x \otimes \rho_x)] = 1. \tag{3.2}$$

Arunachalam et al. [46] showed that $\Omega(n)$ copies of a quantum cryptographic hash based on linear codes is necessary to recover the original n-bit classical message, regardless of the length of the hash itself. Our result shows the complementary aspect that only $\omega(\log n)$ copies are sufficient to ensure that the prover holds the classical message.

Our central technical result is the following. Recall that $p_g := 2^{\ell-n}$ is the optimal guessing probability of the message conditioned on the ℓ-bit side information.

Lemma 3.1 (Lemma 3.2, informally stated)**.** *For any (n, m, δ) quantum fingerprinting ϕ and any leakage of ℓ classical bits, the probability of the forgery state passing the verification scheme satisfies*

$$e_s \le p_g + \delta.$$

This implies that

$$O(p_g + \delta^2) \le e_s^* \le p_g + \delta, \tag{3.3}$$

by considering the cheating strategy of guessing x first and then applying ϕ. Consequently, when $\delta = \mathrm{negl}(n)$, the above inequality means that e_s^* is negligible if and only if p_g is negligible.

Since $\ell = n - O(\log n)$ is the threshold for $p_g(\ell)$ to be non-negligible, the bound Eq. (3.3) show that for quantum cryptographic hash functions, the leakage resilience can approach the *maximum* of $n - O(\log n)$ bits.

One counterpart of this result is shown in [46], saying that $\Omega(n)$ copies of quantum fingerprints would be necessary for an adversary to recover the original message with non-negligible probability. Thus, the quantum cryptographic hash functions based on fingerprinting have the following property: The hash itself is efficiently computable, but it is information-theoretically resilient to recovery of the message from the hash and to recovery of the hash from partial information of the message.

3.1.4 Implications on quantum-proof randomness extraction

Our result reveals some stark contrast between quantum and classical side information. This difference shows the difficulty of establishing the quantum security of classical-proof extractors. Roughly speaking, a randomness extractor is a deterministic algorithm which turns weakly random sources into near uniform [47, 48, 49]. These are fundamental objects with a wide range of applications in computational complexity, cryptography, and other areas [50, 51, 52, 53, 54]. In particular, they accomplished the important tasks of privacy amplification [55, 50, 56], by decoupling the correlation between the output and the side information.

A major open problem in randomness extraction is whether every classical-proof randomness extractor for k min-entropy sources is secure against quantum adversaries with a comparable amount but quantum side information. Loss of parameter is already shown to be inevitable in [57], but possibilities still remain in the case where the ranges of parameters are relevant to most typical applications.

If all quantum side information can be constructed from a comparable amount of classical side information, we would have resolved this major problem positively. Our result shows that this approach would necessarily fail. For details, see Section 3.5.

Theorem 3.3 (Theorem 3.7, informally stated)**.** *There exists a family of classical-quantum states with almost maximal conditional min-entropy, yet arbitrarily far in distance from any quantum state generated from local operations on some classical distribution with constant conditional min-entropy.*

3.1.5 Sketch of proofs

We approach the Seperation Lemma (Lemma 3.2) by first considering the following problem. Suppose that one has access to the Y part of a joint distribution η_{XY} and is able to apply a quantum channel on Y. Their goal is to generate a certain desired cq-state ρ_{XE}. Fix the state ρ_{XE} and consider the set of joint distributions that can be used to generate ρ_{XE}. The infimum of guessing probabilities among the set, called the *conversion parameter*, has a nice representation that can be lower bounded. Put into the context of hashing verification, an adversary trying to pass the verification test is essentially trying to forge the message-hash cq-state. This cannot be done if the adversary starts from a piece of side information, from which the probability of guessing the message correctly is below the conversion parameter. This indicates that a certain amount of classical information must have already been leaked to the adversary.

In the end, the proof reduces to estimating the operator norm of $\sum_i T_i$, where T_i's are projections to the fingerprint states, thus pairwise almost orthogonal. Cotlar-Stein Lemma gives us the desired bound.

3.2 Preliminaries

Let us start from a brief recap of classical information theory. For a distribution p over a finite set $\mathcal{X}$, we define the *support* of it to be

$$\mathrm{Supp}(p) := \{x \in \mathcal{X} | p(x) > 0\}. \tag{3.4}$$

The Shannon entropy of p, denoted $H(p)$, quantifies the amount of information contained in one sample of p:

$$H(p) := \sum_{x \in \mathrm{Supp}(p)} -p(x) \log(p(x)). \tag{3.5}$$

The Rényi entropy is a parametrized family of Shannon entropy:

$$H_\alpha(p) := \frac{1}{1-\alpha} \log \left(\sum_{x \in \mathrm{Supp}(p)} p(x)^\alpha \right), \quad \alpha \in \mathbb{R}_+ \setminus \{1\}. \tag{3.6}$$

The Shannon entropy serves as a special case of the Rényi entropy as the limit for $\alpha \to 1$. For a fixed distribution, the Rényi entropies are monotonically decreasing with respect to α. We have the *max-entropy* at $\alpha = 0$: $H_{\max}(p) := \log |\mathrm{Supp}(p)|$ and the *min-entropy* when α approaches infinity: $H_{\min}(p) := -\log \max_x p(x)$. For a random variable $X \sim p$, we write $H(X) := H(p)$.

For joint distributions $p \sim \mathcal{X} \times \mathcal{Y}$, joint entropies can be defined similarly. The conditional Rényi entropies quantify the amount of information $X \in \mathcal{X}$ could provide given that $Y \in \mathcal{Y}$ has already been known, where XY is a joint sample from the distribution p. Denote p_Y the marginal distribution of p on $\mathcal{Y}$. The conditional Rényi entropy is defined as

$$H_\alpha(X|Y)_p := \sum_{y \in \mathrm{Supp}(p_Y)} p_Y(y) H_\alpha(X|Y = y) = \mathbb{E}_{y \sim p_Y}[H_\alpha(X|Y = y)]. \tag{3.7}$$

3.2.1 Quantum conditional min-entropy

The min-entropy of a random variable X is defined as the negative logarithm of the probability of the most probable outcome. Operationally, this probability denotes the maximum probability that a party can guess the value of X without any prior knowledge despite its distribution.

Generalized to joint distributions, the conditional min-entropy of a random variable conditioned on another random variable Y denotes the negative logarithm of the maximum expected probability of successfully guessing the value of X, upon a (possibly) correlated message Y is revealed.

For a classical-quantum state $\rho_{XE} = \sum_{x\in\mathcal{X}} |x\rangle\langle x| \otimes \rho_x$, there are no longer properly defined conditional distributions conditioned on a "value" of E. Instead, we adopt the operational meaning of guessing probabiltiy. Given a quantum message E, one is asked to make the best guess of the message X. The most general thing such a party can perform is a POVM $\{M_x\}_{x\in\mathcal{X}}$. The probability of successfully guessing X is then

$$p = \sum_{x\in\mathcal{X}} \mathrm{Tr}[\rho_x M_x]. \tag{3.8}$$

Taking the maximum over all possible POVMs, we define the guessing probability of X conditioned on a quantum system E to be

$$p_g(X|E)_\rho := \max_{M_x \succcurlyeq 0; \sum_x M_x = I} \sum_x \mathrm{Tr}[\rho_x M_x]. \tag{3.9}$$

The conditional min-entropy for classical-quantum states are then defined as

$$H_{\min}(X|E)_\rho := -\log p_g(X|E)_\rho. \tag{3.10}$$

Applying duality results in semidefinite programming, there is a nice formulation of the guessing probability:

$$p_g(X|E)_\rho = \min_{\sigma:\sigma \succcurlyeq \rho_x, \forall x} \mathrm{Tr}[\sigma], \tag{3.11}$$

i.e. the guessing probability is the minimum trace of any PSD operator which is a *common roof* of all subnormalized states $\{\rho_x\}_{x\in\mathcal{X}}$.

3.3 The Separation Lemma

3.3.1 Conversion parameters and the Separation Lemma

Given a cq-state $\rho_{XE} := \sum_x |x\rangle\langle x| \otimes \rho_x$, we are now interested in determining the least amount of classical correlation, measured by the *conversion parameter*, that one needs in order to generate ρ by applying a quantum channel on the side information.

Definition 3.4 (Conversion Parameter). *Let $\rho_{XE} \in \mathcal{S}_{\leq}(\mathcal{X} \otimes \mathcal{E})$ be a cq-state. The* conversion parameter *of X conditioned on E is defined as*

$$p_{\downarrow}(X|E)_\rho := \inf_{\substack{\eta_{XY},\mathcal{C} \\ \mathcal{I}\otimes\mathcal{C}(\eta)=\rho}} p_g(X|Y)_\eta. \tag{3.12}$$

By monotonicity of guessing probability under quantum channels acting on the side information, we have $p_{\downarrow}(X|E)_\rho \geq p_g(X|E)_\rho$.

The definitions above would be less interesting if we do not allow the existence of an extra error term ϵ for the following reason. For a state ρ with the form

$$\rho_{XE} = \sum_x q_x |x\rangle\langle x| \otimes |\psi_x\rangle\langle\psi_x|, \tag{3.13}$$

where $|\psi_x\rangle$ for each x are distinct, the only classical side information to generate ρ losslessly would be the classical message itself; but if we allow the generated state to be ϵ-close to the desired state ρ, we may be able to approximate ρ using classical information with significantly lower guessing probability.

Definition 3.5 (Smooth Conversion Parameter). *Let $\epsilon \geq 0$ and $\rho_{XE} \in \mathcal{S}_{=}(\mathcal{X} \otimes \mathcal{E})$. Then the* ϵ-smoothed conversion parameter *of X conditioned on E is defined as*

$$p^{\epsilon}_{\downarrow}(X|E)_\rho := \inf_{\substack{\eta_{XY},\mathcal{C} \\ \|\mathcal{I}\otimes\mathcal{C}(\eta)-\rho\|_{tr}\leq\epsilon}} p_g(X|Y)_\eta. \tag{3.14}$$

One might suspect that, with a reasonable error tolerance, an arbitrary cq-state could be approximately generated from a classical joint distribution with similar guessing probability. The Separation Lemma, however, proves the opposite. When the quantum side information has a particular form, even approximating the state with a constant error requires almost the entire classical message in the beginning.

Lemma 3.2 (Separation Lemma)**.** *For all $\delta > 0, \epsilon \geq 0$ and state*

$$\rho_{XE} = \sum q_x |x\rangle\langle x| \otimes |\psi_x\rangle\langle\psi_x|$$

with maximum overlap

$$\delta = \max_{x,y:x\neq y} |\langle\psi_x|\psi_y\rangle|,$$

we have

$$1 - \epsilon/2 \leq (1-\delta)p_\downarrow^\epsilon(X|E)_\rho + \delta, \tag{3.15}$$

or equivalently,

$$p_\downarrow^\epsilon(X|E)_\rho \geq 1 - \frac{\epsilon}{2(1-\delta)}. \tag{3.16}$$

3.3.2 Characterizing the conversion parameter

The starting point of proving Lemma 3.2 is a better understanding of the smoothed conversion parameter $p_\downarrow^\epsilon(X|E)_\rho$ for given parameter ϵ. Recall that for guessing probability $p_g(X|E)_\rho$, we have

$$p_g(X|E)_\rho = \min_{\sigma:\forall x, \sigma \succcurlyeq \rho_x} \mathrm{Tr}[\sigma] \tag{3.17}$$

by SDP duality of quantum guessing probability. Similarly, we can obtain a rather nice formulation of the (smoothed) conversion parameter using the following lemma:

Lemma 3.3. *Let $\epsilon \geq 0$ and $\rho_{XE} = \sum_x |x\rangle\langle x| \otimes \rho_x \in \mathcal{S}_=(\mathcal{X}\otimes\mathcal{E})$, then*

$$p_\downarrow^\epsilon(X|E)_\rho = \min_{\substack{p_S,\hat{\rho}_S \\ \sum_x \|\sum_{x\in S} p_S\hat{\rho}_S - \rho_x\|_{tr} \leq \epsilon}} \sum_S p_S, \tag{3.18}$$

where S ranges over all nonempty subsets of $\mathcal{X}$.

Before proceeding to the proof of the lemma, we want to first narrow down the set of classical side information we need to consider in this context. The next lemma shows that it suffices to consider a special subclass of joint distributions we call *conditionally uniform*.

Lemma 3.4. *Let $\eta_{XY} = \sum_{xy} p_{xy}|x\rangle\langle x| \otimes |y\rangle\langle y|$ be a classical-classical- (cc-) state. Then there exists a cc-state $\eta'_{XS} \in \mathcal{S}_=(\mathcal{X}\otimes(2^{\mathcal{X}} \setminus \{\emptyset\}))$ with the following three properties:*

1. *$p_g(X|Y)_\eta = p_g(X|S)_{\eta'}$;*
2. *There exists a classical channel $C_{S\to Y}$ such that $(I \otimes C)(\eta') = \eta$;*

3. *η' is* conditionally uniform*: that is, the conditional distribution of X conditioned on S is the uniform distribution U_S, i.e.*

$$\Pr[X|S = s]_{\eta'} = \frac{1}{|s|}\delta_{X\in s}. \tag{3.19}$$

Lemma 3.4 indicates that conditionally uniform distributions are universal in the sense that all joint distributions can be generated from a conditionally uniform distribution with the same guessing probability, by only applying a channel on the side informaiton. For the purpose of determining the (smoothed) conversion parameter of a state ρ_{XE}, we can then without loss of generality assume that the state is being generated from a conditionally uniform distribution. This greatly simplifies the problem of determining conversion parameters, and leads to a nice representation of conversion parameters as stated in Lemma 3.3.

Proof. We provide a constructive proof. For sake of simplicity we will use classical probability notations, i.e. η_{XY} will be identified as the classical joint distribution p_{XY}.

All subset of $\mathcal{X}$ form a directed acyclic graph under containment relations, therefore we can recursively define

$$p_S(y) = \min_{x\in S} p(x, y) - \sum_{S\subsetneq S'} p_{S'}(y) \tag{3.20}$$

for all nonempty $S \subseteq \mathcal{X}$. Let $\hat{p}_S = \sum_y p_S(y)$ and $C_S = \mathrm{Supp}(p_S(y))$. We claim that the distribution

$$p'(x, S) = \hat{p}_S\delta_{x\in S}, \tag{3.21}$$

together with the channel

$$C(y|S) = p_S(y)/\hat{p}_S \tag{3.22}$$

satisfies our requirements for η'.

First we need to show that p'_{XS} is indeed a distribution, and C is indeed a classical channel. It suffices to show that $p_S(y) \geq 0$ for all S and y.

To prove this, we apply induction on size of the set to prove two things:

- $p_S(y) \geq 0$ for all S and y;
- For all S_1 and S_2 where no one contains the other, we have $C_{S_1} \cap C_{S_2} = \emptyset$.

The base case is when $|S|, |S_1|, |S_2| \geq |\mathcal{X}|$. Apparently we have $S = S_1 = S_2 = \mathcal{X}$, so the second statement is true. For the first one, we have by construction

$$p_{\mathcal{X}}(y) = \min_x p(x, y) \geq 0. \tag{3.23}$$

Let's move on to the induction step. Assume now that for all S_1, S_2 where

- $|S_1|, |S_2| > k$
- $S_1, S_2 \subsetneq S_1 \cup S_2$

we have $p_S(y) \geq 0$ for all y and $C_{S_1} \cap C_{S_2} = \emptyset$. Then for a given y and a subset S with $|S| = k$, the collection of subsets $\{S' | y \in \text{Supp}(S'), S \subsetneq S'\}$ must form a chain $S'_1 \subsetneq S'_2 \subsetneq \cdots \subsetneq S'_m$ under containment relations. Then

$$\begin{aligned} p_S(y) &= \min_{x \in S} p(x,y) - \sum_{S \subsetneq S'} p_{S'}(y) && (3.24) \\ &= \min_{x \in S} p(x,y) - \sum_{i=1}^{m} p_{S'_i}(y) && (3.25) \\ &= \min_{x \in S} p(x,y) - \left(\min_{x \in S'_1} p(x,y) - \sum_{i=2}^{m} p_{S'_i}(y) \right) - \sum_{i=2}^{m} p_{S'_i}(y) && (3.26) \\ &= \min_{x \in S} p(x,y) - \min_{x \in S'_1} p(x,y) && (3.27) \\ &\geq 0. && (3.28) \end{aligned}$$

Also, for S_1, S_2 not contained in each other and with size $\geq k$, we have

$$\begin{aligned} p_{S_1}(y) &= \min_{x \in S_1} p(x,y) - \sum_{S_1 \subsetneq S'} p_{S'}(x,y) && (3.29) \\ &= \left(\min_{x \in S_1} p(x,y) - \min_{x \in S_1 \cup S_2} p(x,y) \right) - \sum_{S_1 \subseteq S', S_2 \not\subseteq S'} p_{S'}(y) && (3.30) \\ &\leq \min_{x \in S_1} p(x,y) - \min_{x \in S_1 \cup S_2} p(x,y) && (3.31) \end{aligned}$$

and similarly

$$p_{S_2}(y) \leq \min_{x \in S_2} p(x,y) - \min_{x \in S_1 \cup S_2} p(x,y) \qquad (3.32)$$

using the induction hypothesis $p_S(y) \geq 0$ for all $|S| \geq k$. As

$$\min_{x \in S_1 \cup S_2} p(x,y) = \min\{\min_{x \in S_1} p(x,y), \min_{x \in S_2} p(x,y)\}, \qquad (3.33)$$

we know that either $p_{S_1}(y) = 0$ or $p_{S_2}(y) = 0$, which results in that $y \notin C_{S_1} \cap C_{S_2}$. As this holds for an arbitrary y, we must have $C_{S_1} \cap C_{S_2} = \emptyset$. This completes the induction step.

We then proceed to prove that p' and C satisfies the three properties listed in Lemma 3.4. Note that p' is conditionally uniform by construction, so it suffices to show just the first two properties.

1. To see property 1 of Lemma 3.4, note that the guessing probability of p and p' are respectively $p_g(X|Y)_p = \sum_y \max_x p(x,y)$ and $p_g(X|S)_{p'} = \sum_S \hat{p}_S = \sum_y \sum_S p_S(y)$. It then suffices to prove that $\sum_S p_S(y) = \max_x p(x,y)$ holds for all y.

   One direction is easy to prove: for all x we have

$$\sum_S p_S(y) \geq \sum_{\{x\}\subseteq S} p_S(y) \tag{3.34}$$

$$= p(x,y) - \sum_{\{x\}\subsetneq S} p_S(y) + \sum_{\{x\}\subsetneq S} p_S(y) \tag{3.35}$$

$$= p(x,y), \tag{3.36}$$

   thus $\sum_S p_S(y) \geq \max_x p(x,y)$. To prove the other direction, let's look more in detail into what we have proved so far. For every subset S, we have shown that the sum $\sum_{S\subsetneq S'} p_{S'}(y)$ can be reduced to summation on a chain. Without loss of generality, this chain can be completed to maximal length such that the difference of cardinalities of adjacent terms on this chain is 1, as additional terms would not change the final result. As

$$\sum_S p_S(y) = \sum_{\emptyset\subsetneq S} p_S(y), \tag{3.37}$$

   there exists a chain $\emptyset \subsetneq S_1 \subsetneq \cdots \subsetneq S_{|\mathcal{X}|} = \mathcal{X}$ such that $|S_i| = i$ and

$$\sum_S p_S(y) = \sum_{i=1}^{|\mathcal{X}|} p_{S_i}(y). \tag{3.38}$$

   As $|S_1| = 1$, there must exists x^* such that $S_1 = \{x^*\}$. Then

$$\sum_S p_S(y) = \sum_{i=1}^{|\mathcal{X}|} p_{S_i}(y) = p(x^*,y) \leq \max_x p(x,y), \tag{3.39}$$

   which proves the opposite direction of property 1.

2. For property 2, denote p''_{XY} the joint distribution we get from applying C onto p'. We

have

$$\begin{aligned}
p''(x,y) &= \sum_S p'(x,S)C(y|S) && (3.40)\\
&= \sum_S \delta_{x\in S}\hat{p}_S \cdot p_S(y)/\hat{p}_S && (3.41)\\
&= \sum_{x\in S} p_S(y) && (3.42)\\
&= p_{\{x\}}(y) + \sum_{\{x\}\subsetneq S} p_S(y) && (3.43)\\
&= p(x,y) - \sum_{\{x\}\subsetneq S} p_S(y) + \sum_{\{x\}\subsetneq S} p_S(y) && (3.44)\\
&= p(x,y). && (3.45)
\end{aligned}$$

This proves property 2, which finishes the entire proof of Lemma 3.4.

□

With Lemma 3.4, we are now ready to prove Lemma 3.3.

Proof of Lemma 3.3. Recall the definition of ϵ-smoothed conversion parameter:

$$p_{\downarrow}^{\epsilon}(X|E)_\rho = \inf_{\substack{\eta_{XY},\mathcal{C}_{\mathcal{Y}\to\mathcal{E}}\\ \|(\mathcal{I}\otimes\mathcal{C})(\eta)-\rho\|\le\epsilon}} p_g(X|Y)_\eta. \tag{3.46}$$

For every η, by Lemma 3.4, there exists a conditionally uniform joint distribution $\eta'_{XS} = \sum_S p_S(\sum_{x\in S}|x\rangle\langle x|)\otimes|S\rangle\langle S|$ with the same guessing probability as η, and an ϵ-approximation of ρ can also be generated from η' via local channel on the side information. Note that the guessing probability of η' is $\sum_S p_S$. A general quantum channel acting on the side information can be characterized as follows without loss of generality:

$$\mathcal{C}'_{2^{\mathcal{X}}\to\mathcal{E}}(\cdot) = \sum_S \langle S|\cdot|S\rangle\hat{\rho}_S. \tag{3.47}$$

Then we have

$$(\mathcal{I}\otimes\mathcal{C}')(\eta') = \sum_S(\sum_{x\in S}|x\rangle\langle x|)\otimes p_S\hat{\rho}_S, \tag{3.48}$$

which leads to

$$\|(\mathcal{I}\otimes\mathcal{C}')(\eta')-\rho\|_{tr} = \sum_x \|\sum_{x\in S} p_S\hat{\rho}_S - \rho_x\|_{tr}. \tag{3.49}$$

Lemma 3.3 is then proved by observing that a minimum can always be achieved since the set of conditionally uniform distributions over $\mathcal{X} \times (2^{\mathcal{X}} \setminus \{\emptyset\})$ is compact. $\square$

3.3.3 Proof of the Separation Lemma

With a nice formulation of the conversion parameter, we are now ready to proceed to the proof of the Separation Lemma (Lemma 3.2). In the proof of Lemma 3.2, we will make use of Cotlar-Stein Lemma, which gives a good estimate of the operator norm of the sum of near-orthogonal operators. We will attach the proof of the Cotlar-Stein Lemma in Appendix A for completeness.

Lemma 3.5 (Cotlar-Stein Lemma). *For a set of unit vectors* $\{|\psi_1\rangle, |\psi_2\rangle, \cdots, |\psi_n\rangle\}$ *with maximum fidelity* $\max_{i,j:i\neq j} |\langle\psi_i|\psi_j\rangle| \leq \delta$, *we have*

$$\lambda_{\max}\left(\sum_{i=1}^{n} |\psi_i\rangle\langle\psi_i|\right) \leq 1 + (n-1)\delta. \tag{3.50}$$

Proof of Lemma 3.2. By Lemma 3.3, we take a conditionally uniform distribution

$$\eta_{XZ} = \sum_S p_S (\sum_{x\in S} |x\rangle\langle x|) \otimes |S\rangle\langle S|, \tag{3.51}$$

together with an ensemble of quantum states $\{\hat{\rho}_S\}_{S\neq\emptyset}$, such that

$$p_g(X|Z)_\eta = \sum_S p_S = p_\downarrow^\epsilon(X|E)_\rho, \tag{3.52}$$

$$\Delta := \|\rho - \sigma\|_{tr} \leq \epsilon, \tag{3.53}$$

where $\sigma := \sum_S p_S(\sum_{x\in S} |x\rangle\langle x|) \otimes \hat{\rho}_S$.

We bound Δ from the left by noticing that

$$\Delta = \|\rho - \sigma\|_{tr} \tag{3.54}$$

$$= \max_{\|V\|_\infty \leq 1} \mathrm{Tr}[V(\rho - \sigma) \otimes p_S\hat{\rho}_S)] \tag{3.55}$$

$$= 2 \max_{\|V\|_\infty \leq 1, V \succcurlyeq 0} \mathrm{Tr}[V(\rho - \sigma)]. \tag{3.56}$$

$$\tag{3.57}$$

Take $V^* := \sum_x |x\rangle\langle x| \otimes |\psi_x\rangle\langle\psi_x|$. One can easily check that $\mathrm{Tr}[V^*\rho] = 1$ and thus

$\mathrm{Tr}[V^*\sigma] \geq 1 - \epsilon/2$. It then suffices to upper bound $\mathrm{Tr}[V^*\sigma]$.

$$\mathrm{Tr}[V^*\sigma] = \sum_x \langle\psi_x| \sum_{x\in S} p_S \hat{\rho}_S |\psi_x\rangle \tag{3.58}$$

$$= \sum_S p_S \mathrm{Tr}[\hat{\rho}_S \sum_{x\in S} |\psi_x\rangle\langle\psi_x|] \tag{3.59}$$

$$\leq \sum_S p_S \lambda_{\max}(\sum_{x\in S} |\psi_x\rangle\langle\psi_x|). \tag{3.60}$$

$$\tag{3.61}$$

By the Cotlar-Stein Lemma, we have

$$\lambda_{\max}(\sum_{x\in S} |\psi_x\rangle\langle\psi_x|) \leq 1 + (|S|-1)\delta. \tag{3.62}$$

Combining this with $p^\epsilon_\downarrow(X|E)_\rho = \sum_S p_S$ and $1 = \sum_S p_S |S|$,

$$\mathrm{Tr}[V^*\sigma] \leq \sum_S p_S \lambda_{\max}\left(\sum_{x\in S} |\psi_x\rangle\langle\psi_x|\right) \tag{3.63}$$

$$\leq \sum_S p_S \left(1 + (|S|-1)\delta\right) \tag{3.64}$$

$$= p^\epsilon_\downarrow(X|E)_\rho(1-\delta) + \delta. \tag{3.65}$$

Lemma 3.2 is proven by combining $\mathrm{Tr}[V^*\sigma] \geq 1 - \epsilon/2$ with $\mathrm{Tr}[V^*\sigma] \leq p^\epsilon_\downarrow(X|E)_\rho(1-\delta) + \delta$. □

3.4 Resilience of quantum hashing against classical leakage

Now let us consider the setting of quantum hashing. For the sake of simplicity, we assume that the classical message X is uniformly distributed over $\{0,1\}^n$, and all parties are computationally unbounded.

Suppose now that a prover A, either adversarial or honest, upon obtaining some side information Y of X, wants to show that he has full access to the classical message X. To show this, he needs to pass a test held by a verifier V who has full access to X. One way to do this is to send V an m-bit hash M, and V will accept or reject based on the classical messages X and M.

If the adversary wants to cheat as best as he can, the optimal strategy would be applying a deterministic mapping on the side information Y; similarly, if the verifier wants to distinguish adversarial parties from honest ones as best as he can, the optimal strategy would also be a deterministic algorithm. Therefore we can safely assume that both A and V are deterministic mappings.

Definition 3.6. *A mapping $h : \{0,1\}^n \to \{0,1\}^m$ is called σ-resilient against k bits of information leakage if for all classical side information Y of X such that $H_{\min}(X|Y) \geq n-k$ and for all mappings $f : \mathcal{Y} \to \{0,1\}^m$, we have*

$$\Pr_{XY}[f(Y) = h(X)] \leq \sigma. \tag{3.66}$$

Here $\mathcal{Y}$ is defined as the support of Y. If σ is not specified then it is assumed that $\sigma = \mathrm{negl}(n)$.

Ideally, one would want a hash function to be both short and resilient against much information leakage. Unfortunately these two requirements cannot be achieved at the same time. If an m-bit verification scheme V is resilient against k bits of classical leakage, one must have $m = k + \omega(\log n)$. This can be seen as follows: suppose otherwise that $m = k + O(\log n)$. Then an adversarial prover, upon getting the first k bits of the message an honest party would send to the verifier, guesses the remaining $O(\log n)$ bits uniformly at random. Such a prover would then have an inverse polynomial probability of passing the test.

Now suppose that both the verifier and the prover have access to quantum power, while the information leakage still remains classical. Now the prover can send an m-qubit quantum system ρ to the verifier, and the verifier would perform a joint measurement on both X and ρ to determine whether to accept or not. Denote the state space of X, Y and M respectively by $\mathcal{X}, \mathcal{Y}$ and $\mathcal{M}$.

Definition 3.7 (Resilience of quantum fingerprinting against classical information leakage)**.** *An (n, m, δ) quantum fingerprinting ϕ is called σ-resilient against k bits of classical information leakage if for all classical side information Y of X such that $H_{\min}(X|Y) \geq n-k$ and all quantum channel $\mathcal{C}_{\mathcal{Y}\to\mathcal{M}}$, we have*

$$\mathrm{Tr}[V \cdot (\mathcal{I} \otimes \mathcal{C})(\eta_{XY})] \leq \sigma, \tag{3.67}$$

where

$$\eta_{XY} = \sum_{x,y} \mathrm{Tr}[X = x, Y = y] |x\rangle\langle x| \otimes |y\rangle\langle y|, \tag{3.68}$$

and

$$V = \sum_x |x\rangle\langle x| \otimes \phi_x. \tag{3.69}$$

When σ is not specified, it is assumed that $\sigma = \mathrm{negl}(n)$.

In the case where the inputs of can be assumed classical, it is safe to replace a general channel by mapping each classical information to a state. Therefore, $\mathcal{C}$ can be specified by

$$\mathcal{C}(\cdot) = \sum_y \langle y| \cdot |y\rangle \rho_y. \tag{3.70}$$

Then the passing probability $\mathrm{Tr}[V \cdot (\mathcal{I} \otimes \mathcal{C})(\rho_{XY})]$ can be written as $\mathbb{E}_{XY}[\mathrm{Tr}[\phi_X \rho_Y]]$.

In sharp contrast to the classical case, the Separation Lemma implies that there exists a (n, m, δ) quantum fingerprinting resilient against k bits of classical information leakage, where k is much larger than m. In fact we have the following theorem.

Theorem 3.4. *For all n, m, and $k = n - \omega(\log n)$, all quantum cryptographic hash functions mapping n bits to m qubits are resilient against k bits of classical information leakage. Furthermore, such functions always exist, and can be constructed efficiently when $m = \omega(\log n)$.*

Before proving this theorem, note that this theorem is tight on both sides. If $k = n - O(\log n)$, then an adversary knowing the side information can guess correctly the actual value of X with probability inverse polynomial, thus the success probability would also be non-negligible; on the other hand, if $m = O(\log n)$, there is certainly no (n, m)-cryptographic hash functions. We claim that an adversary with zero side information can still pass the test with non-negligible probability in this case.

To see this, let's start from the completeness condition. This is saying that there exist ρ_x's such that $\mathbb{E}_X[\mathrm{Tr}[M_X \rho_X]] = 1$. This can only happen when each term is 1, which in turn implies that $\mathrm{Tr}[M_x] \geq 1$ for all x. Now suppose the adversary has no side information about the classical message, so the best he can do is to prepare a state ρ. The success probability will then be

$$\mathbb{E}_X \mathrm{Tr}[M_X \rho] \leq \lambda_{\max}(\mathbb{E}_X[M_X]), \tag{3.71}$$

which can be approached when $\rho_0 = |\psi_0\rangle\langle\psi_0|$, $|\psi_0\rangle$ being the eigenvector corresponding to the largest eigenvalue of $\mathbb{E}_X[M_X]$. The operator norm can be lower bounded from the trace by

$$\lambda_{\max}(\mathbb{E}_X[M_X]) \geq \frac{\mathrm{Tr}[\mathbb{E}_X[M_X]}{\dim \mathcal{M}} \geq \frac{1}{poly(n)}, \tag{3.72}$$

resulting in a non-negligible passing probability without any side information.

Now let us proceed to the proof of Theorem 3.4.

Proof of Theorem 3.4. Take an (n, m, δ) quantum fingerprinting ϕ, where δ will be specified later. We know that such a finderprinting exists for $m = O(\log n + 2\log\frac{1}{\delta})$, therefore there exists $\delta = \mathrm{negl}(n)$ such that ϕ is a quantum cryptographic hash. The verification scheme associated to this quantum fingerprinting is then

$$V = \sum_x |x\rangle\langle x| \otimes \phi_x. \tag{3.73}$$

Upon getting $k = n - l$ bits of classical information, the guessing probability of the adversary to the classical message is upper bounded by 2^{-l}. Following the proof of the Separation Lemma, the probability that the adversary pass the test e_s is upper bounded by $2^{-l} + \delta$, which is still a negligible function of n given $l = \omega(\log n)$. □

3.4.1 A lightweighted verification scheme resilient against classical leakage

Our verification scheme is maximally resilient to classical leakage of information. However, it is still not good enough because the verifier may need to get full access to the whole message. One may ask if it is possible that the verifier only use a small amount of information from the message to perform the verification scheme, yet the verification is still resilient to classical leakage.

To formulate this idea, we generalize the definition of a verification scheme $V = (\mathcal{C}, M)$ with three parameters (n, k, m) played by two players A and B as follows:

1. A joint distribution XY, where the marginal distribution of $X \in \{0,1\}^n$ is uniformly random, is generated by nature.

2. A gets the k-qubit quantum state $\rho_X = \mathcal{C}(|X\rangle\langle X|)$ and B gets the side-information string Y.

3. B generates an m-qubit quantum state μ_Y and sends it to A.

4. A performs a joint measurement M on the state $\rho_X \otimes \mu_Y$. The game is successful if and only if the measurement accepts. The overall success probability is thus

$$e_V = \mathbb{E}_{XY}[M(\rho_X \otimes \mu_Y)]. \tag{3.74}$$

For a given ℓ, define the optimal success probability over all classical leakage Y where $H(X|Y) \geq \ell$ to be $e_V^*(\ell)$. We can then define the resilience formally:

Definition 3.8. *A (n, k, m)-verification scheme $V = (\mathcal{C}, M)$ is called σ-resilient to ℓ bits of classical leakage if we have*

- *Completeness: $e_V^*(n) = 1$, and*
- *Soundness: $e_V^*(\ell) \leq \sigma$.*

In the case where σ is not specified, it is assumed that σ is negligible.

Interestingly, one can use the power of quantum side information to reduce the size of the advice state.

Theorem 3.5. *For all n, there exists a (n, k, m)-verification scheme V which is resilient to ℓ bits of classical leakage whenever*

$$k = m = \omega(\log^2 n), n - \ell = \omega(\log n). \tag{3.75}$$

Proof. Proof by construction. Take $t, m' = \sqrt{k} = \omega(n)$. We first fix a (n, m')-quantum cryptographic hash function ϕ. for a given message x, the advice state would just be t-fold tensor product of ϕ_x's, namely

$$\rho_x := \phi_x^{\otimes t}. \tag{3.76}$$

Upon receiving the hash μ_Y consisting of t parts of m'-qubit states, the verifier performs SWAP test between all t pairs of ϕ_X and each of the qubit states, accepts if all SWAP tests pass and rejects otherwise. Note that an honest party having full access to X would be able to produce $\phi_X^{\otimes t}$ perfectly, thus successes with probability 1.

To see that this scheme is resilient against classical leakage, assume now that the state received by the verifier is a forgery state μ_Y. Regarding both ρ_X and μ_Y as t-partite states, we use $\rho_X^T, \mu_Y^T, SWAP^T$ to denote the marginal state on subsystems $T \subseteq [t]$ and the swap between the two subsystems respectively. The measurement corresponding to one copy of

SWAP test is $\frac{I+SWAP}{2}$, thus the success probability will be

$$e_V = \mathbb{E}_{XY}\left[\frac{\mathrm{Tr}[(I+SWAP)^{\otimes t}(\rho_X \otimes \mu_Y)]}{2^t}\right] \tag{3.77}$$

$$= \frac{1}{2^t}\sum_{T\in[t]} \mathbb{E}_{XY}\left[\mathrm{Tr}[\bigotimes_{i\in T} SWAP^i(\rho_X \otimes \mu_Y)]\right] \tag{3.78}$$

$$= \frac{1}{2^t}\sum_{T\in[t]} \mathbb{E}_{XY}\left[\mathrm{Tr}[SWAP^T(\rho_X^T \otimes \mu_Y^T)]\right] \tag{3.79}$$

$$= \frac{1}{2^t}\sum_{T\in[t]} \mathbb{E}_{XY}[\mathrm{Tr}[\rho_X^T \cdot \mu_Y^T]]. \tag{3.80}$$

$$\tag{3.81}$$

Here the third line comes by tracing out irrelevant states, and the fourth line comes from the identity $\mathrm{Tr}[SWAP(\rho \otimes \sigma)] = \mathrm{Tr}[\rho\sigma]$. For each term $\mathrm{Tr}[\rho_X^T \cdot \mu_Y^T]$ with nonempty T and any $i \in T$, we have

$$\mathrm{Tr}[(\rho_X^T \cdot \mu_Y^T)] \leq \mathrm{Tr}[(\phi^i \otimes I^{T\setminus\{i\}})\mu_Y^T] = \mathrm{Tr}[\phi_x \rho_Y^i]. \tag{3.82}$$

This gives us

$$e_V \leq \frac{1}{2^t} + \max_i \mathbb{E}_{XY}\mathrm{Tr}[\rho_Y^i \phi_X]. \tag{3.83}$$

By Theorem 3.4, forall Y such that $H_{\min}(X|Y) \geq \ell$, $i \in [t]$, $\mathbb{E}_{XY}\mathrm{Tr}[\rho_Y^i \phi_X] = \mathrm{negl}(n)$ given that ϕ is a quantum cryptographic hash function. $e_V^*(\ell) = \mathrm{negl}(n)$ then comes from that $t = \omega(\log n)$. □

3.5 Implications of the Separation Lemma on quantum-proof extractors

Let us now recall the setting of seeded extractor. A seeded extractor $Ext : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ takes a weak random source X as well as a much shorter, uniform and independent seed Y and outputs a nearly uniform distribution. Rigorously we have the following definition.

Definition 3.9 (Extractor). *A function $Ext : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ is called an* (k, ϵ)-extractor *if for all random source X such that $-\log p_g(X) \geq k$, we have*

$$\|U_m - Ext(X \otimes U_d)\| \leq \epsilon. \tag{3.84}$$

When used in the setting of privacy amplification, one would consider the case where there is a leakage of the random source. The output of the extractor then need to not only be close to uniform, but also be almost independent of the side information. Depending on whether the leakage is classical or quantum, we have the following definitions for *classical-proof* and *quanutm-proof* extractors respectively.

Definition 3.10 (Classical-proof Extractor)**.** *A function $Ext : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ is called an (k, ϵ)-*classical-proof extractor *if for all random source X with classical side information Y such that $-\log p_g(X|Y) \geq k$, we have*

$$\|U_m \otimes Y - Ext(X \otimes U_d)Y\| \leq \epsilon. \tag{3.85}$$

Definition 3.11 (Quantum-proof Extractor)**.** *A function $Ext : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ is called an (k, ϵ)-*quantum-proof extractor *if for every classical-quantum state ρ_{XE} such that $-\log p_g(X|E) \geq k$, we have*

$$\|U_m \otimes \rho_E - (Ext \otimes \mathcal{I})(\rho_{XE})\|_{tr} \leq \epsilon. \tag{3.86}$$

Intensive research on all three kinds of extractors has been done over the years. Fortunately, the following theorem says that a good extractor is essnetially a good classical-proof extractor, up to very little parameter loss:

Theorem 3.6 ([58])**.** *Any (k, ϵ)-extractor is a $(k + \log 1/\epsilon, 2\epsilon)$-classical-proof extractor.*

One long-standing open problem, then, is whether a classical-proof extractor is essentially a quantum-proof extractor with roughly the same parameter. Currently the best result is that a (k, ϵ)-classical proof extractor is a $(k + \log 2/\epsilon, O(2^{m/2}\sqrt{\epsilon}))$-extractor. For nonexponential blow-up of parameters, Gavinsky et al. [57] showed that there exists a $(k - \Theta(n), \epsilon)$-classical extractor which is not secure against $O(\log n)$ qubits of quantum side information. Proving a result without exponential blowup on the parameter in the practical range, however, is a very challenging problem.

The Separation Lemma here suggests that such a result without a drastic blowing up on the error parameter may not exist. To see this, we need to define some sets first.

Definition 3.12. *For every k, define the following sets:*

- *$C(k)$ is the set of all cq-states ρ_{XE} that can be generated from a classical random source XY with*

$$-\log \max_y p_g(X|Y = y) \geq k \tag{3.87}$$

  via a channel acting only on the Y part;

- *$CC(k)$ is the set of all cq-states ρ_{XE} that can be generated from a classical random source XY with*

$$-\log p_g(X|Y) \geq k \tag{3.88}$$

  via a channel acting only on the Y part;

- *$CQ(k)$ is the set of all cq-states ρ_{XE} such that $-\log p_g(X|E)_\rho \geq k$.*

Clearly we have $C(k) \subseteq CC(k) \subseteq CQ(k)$. To see the importance of these sets, we introduce alternative, though equivalent definitions of extractors, classical-proof and quantum-proof extractors:

Definition 3.13 (Alternative definition for extractors)**.** *A function $Ext : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ is called a (k,ϵ)-extractor (classical-proof extractor, quantum-proof extractor), if for all cq-states $\rho_{XE} \in C(k)$ $(CC(k), CQ(k))$ with $X \in \{0,1\}^n$ we have*

$$\|U_m \otimes \rho_E - (Ext \otimes \mathcal{I})(\rho_{XE})\|_{tr} \leq \epsilon. \tag{3.89}$$

The proof of Theorem 3.6 essentially makes use of the fact that every state ρ_{XE} in $CC(k + \log 1/\epsilon)$ is ϵ-close to the set $C(k)$, and thus a (k,ϵ)-extractor, when applied to ρ, would introduce at most 2ϵ error from the state $U_m \otimes \rho_E$ due to monoticity of trace distance. If one could obtain similar results between $CQ(k)$ and $CC(k)$, then we could easily prove that a classical-proof extractor is secret quantum-proof.

This turns out to not be the case according to Separation Lemma. Recall our definition of ϵ-smooth conversion parameter $p^\epsilon_\downarrow(X|E)_\rho$, which measures the least guessing probability of a classical distribution we need in order to generate the desired state ρ, i.e.

$$d(\rho, CC(k)) \leq \epsilon \Leftrightarrow -\log p^\epsilon_\downarrow(X|E)_\rho \geq k. \tag{3.90}$$

Theorem 3.7. *For any k, there exists a cq-state ρ_{XE} where $X \in \{0,1\}^{k(1+o(1))}$ such that $\rho \in CQ(k)$ while $d(\rho, CC(2)) \geq 0.98$.*

Proof. Fix $\delta = 0.02$ and $\epsilon = 0.98$. By the Johnson-Lindenstrauss Lemma, we have a cq-state

$$\rho_{XE} = 2^{-n} \sum_{x \in \{0,1\}^n} |x\rangle\langle x| \otimes |\psi_x\rangle\langle\psi_x| \in \mathcal{S}_=(\mathcal{X} \otimes \mathcal{E}) \tag{3.91}$$

such that $\log \dim \mathcal{E} = O(\log n)$ and $\max_{x \neq x'} |\langle\psi_x|\psi_{x'}\rangle|$. For given k, there exists $n = k(1 + o(1))$ such that $n - O(\log n) \geq k$. With that n, we have

$$-\log p_g(X|E)_\rho \geq k \Rightarrow \rho \in CQ(k) \tag{3.92}$$

as well as

$$-\log p_{\downarrow}^{\epsilon}(X|E)_{\rho} \leq 2. \tag{3.93}$$

By the Separation Lemma, we have

$$p_{\downarrow}^{\epsilon}(X|E)_{\rho} \geq 1 - \delta - \epsilon/2 \Rightarrow d(\rho, CC(2)) \geq 0.98. \tag{3.94}$$

□

The Separation Lemma suggests that an arbitrary classical extractor may not be quantum-proof, i.e. the sets $CQ(k)$ and $CC(k)$ are spatially separated. Nevertheless, one may still prove that a classical extractor is quantum-proof, but in order to do that, one might need to use additional properties of the extractor, other than that it can extract randomness from all states in $CC(k)$.

3.6 Related Works

Buhrman et al. [15] introduced the notion and provided the constructions of quantum fingerprinting. The application they focused on was for message identification. For our cryptographic applications, we are primarily interested in instances of a negligible fidelity. They did not discuss properties of quantum fingerprinting in an adversarial context like ours. It was observed and explored by [59] that quantum fingerprinting satisfies the security properties of cryptographic hash functions.

Side-channel attack is a major paradigm studied in the classical information security and cryptography community due to its high level of threat in practice [60, 61, 62, 63, 64, 65]. Side-channel key recovery attack has in particular drawn much attention [60]. However, these classical works address problems that necessarily require computational assumptions and many other works focus on the hardware aspects. To the best of our knowledge, this work is the first studying information theoretical security of quantum cryptography against classical side-channel attack.

CHAPTER 4

Limitations of monotone quantum simulation

In this chapter and Chapter 5, we turn our focus to classical simulation of quantum computation. In particular, we focus on the problem of strong (amplitude-wise) simulation of quantum circuits, and identify a subclass of simulators we call *monotone*. This subclass encompasses almost all prominent simulation techniques. We prove an *unconditional* (i.e. without relying on any complexity-theoretic assumptions) and *explicit* $(n-2)(2^{n-3}-1)$ lower bound on the running time of simulators within this subclass. Assuming the Strong Exponential Time Hypothesis (SETH), we further remark that a universal simulator computing *any* amplitude to precision $2^{-n}/2$ must take at least $2^{n-o(n)}$ time. We then compare strong simulators to existing SAT solvers, and identify the time-complexity below which a strong simulator would improve on state-of-the-art general SAT solving. Finally, we investigate Clifford+T quantum circuits with a small number of T-gates. Using the sparsification lemma, we identify time complexity lower bounds in terms of T-gate count below which a strong simulator would improve on state-of-the-art 3-SAT solving.

4.1 Introduction

4.1.1 Overview of classical simulation

Simulating quantum mechanics with classical computational power has been a long-standing problem for nearly two decades [1]. However, only very recently have people been able to accurately control quatum systems in a regime where classical simulation starts to appear challenging. Recently announced quantum processors with intermediate size and reasonable fidelity parameters [66, 67] are pushing the boundary of what classical simulations can handle. With this push, a series of works [68, 69, 70, 71, 72, 73, 74, 75, 38] have been dedicated to keep up with the recent improvement of quantum processors. However, were quantum computation to be genuinely more powerful than classical computation (i.e.

$BPP \subsetneq BQP$), we classical beings cannot hope to win this race in the long term. However, there are still many good reasons to try:

Verification of NISQ devices. As quantum processors are entering into the NISQ era [10], it is important that noise in the quantum devices be fully characterized. Classical simulation of such devices may prove invaluable in verifying that the quantum devices is behaving as expected, and identifying the type and possible physical origin of noise. Only then can physical devices resulting in the noise be improved, corresponding error correcting schemes be designed, and the overall fidelity of quantum processors be further increased.

Testing quantum algorithms and heuristics. With fully fault-tolerant quantum computing still out of reach, several quantum heuristics have been proposed in order to make use of errorneus quantum devices to solve real-life problems [4, 76, 77]. With the presence of insuppressable error, it is often very hard to obtain provable performance guarantees for these heuristics. One way to practically test the performance of these heuristics is via classical simulation [78]. Classical simulation can help determining promising candidates from the heuristics, and help guide the design of near-term quantum devices to better incorporate with such algorithms.

Identifying quantum supremacy. One fundamental question for the quantum computation community is to identify the boundary beyond which quantum computers could achieve something genuinely unattainable classically. This landmark, first proposed in [79] as *quantum supremacy*, represents the point beyond which behaviors of quantum mechanics would be practically infeasible to verify, and we would have to trust quantum mechanics instead of our limited ability to simulate it classically. In order to obtain more faith in this leap, it is vital that we classical beings push the boundary as far as possible. Only then will quantum supremacy be meaningful.

4.1.2 Strong and weak simulations

One major division within the landscape of quantum simulators is between strong and weak simulation. Strong quantum simulators compute the amplitude of a particular outcome, whereas weak simulators only sample from the output distribution of a quantum circuit.

Despite the implication of hardness from their names, we emphasize that no reduction from one type of simulation to the other exist either way where only polynomial overhead incurs. On the one hand, given an instance of strong simulation, the amplitude to be calculated could be exponentially small, thus it takes exponentially large number of trials of weak simulation to be able to estimate such small quantity with a reasonable precision. On the other hand, weak simulation of an input circuit can be done by evaluating a sequence of probability values via strong simulation over a series of augmented circuits. However, since the augmented circuits have different sizes from the size of the input circuit for weak simulation, such reduction does not immediately establish a relationship between comlexities of strong and weak simulations with the same input size. Had the complexity of strong simulation be exponential, such reduction would result in an exponential algorithm for weak simulation with a bigger exponent.

If one only wishes to design a classical computational device that tries to immitate the behavior of a quantum device, then weak simulation would suffice the purpose. However, strong simulation plays an important, and in some cases indispendible role for the following reasons:

- As a testbed of quantum computational devices, the purpose of simulation is often to extract or verify certain quantative properties of the quantum system being simulated. In many cases, such properties can be calculated with great accuracy if strong simulation is available, while estimating them from random outcomes of a weak simulation might require an prohibitive number of trials.

  Let us take quantum supremacy experiments as an example. Since the proposal of *quantum supremacy*, theorists and experimentalists have been designing and implementing experiments, in order to demonstrate that currently existing or near-term quantum devices have the ability to achieve some task that is practically intractable for classical computing power to achieve.

  With no error correction scheme for current quantum architectures at hand, predominant quantum supremacy proposals focus on supposedly hard sampling problems. It is widely believed that certain quantum circuits can produce classical distributions that are hard to sample from a classical computer. However, this does not directly lead to a verification scheme for quantum devices claiming to have sampled from such a distribution. One proposal by Aaronson and Chen [75] relates the hardness of quantum circuit sampling to a verification scheme they call Heavy Output Generation:

  Problem 4.1 (Problem 1, [80]). *Given as input a random quantum circuit C (drawn from some suitable ensemble), generate output strings $x_1, ..., x_k$, at least a $2/3$ fraction of which have greater than the median probability in the output distribution of C.*

To test whether a quantum device acheives quantum supremacy, one just run it repeatedly to get (supposedly) random classical outcomes $x_1, \cdots, x_k$, and apply strong simulation to test whether $\geq 2/3$ of the outcomes have probability greater than the median. It is concievable that classical devices cannot pass this test without prohibitive overhead. However, such scheme requires accurate compotation of the probabilities $\{p_i\}_{1 \leq i \leq k}$, and would not be possible without a strong simulator at hand.

- Even when weak simulation is the ultimate goal, there are not yet many weak simulation techniques that is intrinsically different from a strong simulation routine. Although there are stabilizer-based weak simulators that works well with dominantly Clifford circuits [41, 81, 82, 83], for most general quantum circuits, it is not known how weak simulation can be done significantly faster than using strong simulation as a subrountine.

  Strong simulation, on the other hand, has a natural representation in the context of matrix multiplication, or tensor network contraction in general [84]. Although exponentially-sized intermediate matrices or tensors might be inevitable, the problem of pushing the boundary for large matrix multiplication has long been a central topic in classical architecture design. Such accelaration for matrix multiplication has been native to most classical computational devices, especially GPUs and supercomputers. Those accelarations can be conviniently harnessed in order for a larger scale strong simulation.

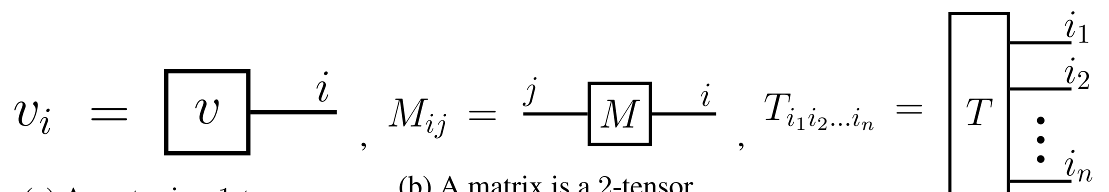
4.1.3 Limitation of monotone simulation methods

Although strong simulation is of great importance, here we give compelling evidence that it is fundamentally unscalable. This is hardly surprising: it is well-known that perfectly accurate strong simulation is $\#P$-hard. However, we give *explicit*, and in some contexts *unconditional* evidence for the hardness of strong simulation.

We identify a large class of simulation techniques we call *monotone*. This class includes most of the known techniques for general quantum circuit simulation. We place unconditional lower bounds on simulators within this class. In particular, we show that there exists a simple quantum circuit which will take any such simulator at least $(n-2)(2^{n-3}-1)$ time to simulate. Following the proof, we also find a $O(n^2 \cdot 2^n)$ algorithm for computing the permanent function with a monotone arithmetic circuit, which is not known before.

4.2 Preliminaries

Tensor Networks. A *tensor* is simply a multidimensional array. The number of dimensions of the array is called the *rank* of the tensor. The simplest examples of tensors are vectors and matrices, which are rank-1 and rank-2 tensors, respectively. Tensors are usually expressed graphically, see Figure 4.1. For simplicity, we assume that each index runs over $\{0,1\}$.



(a) A vector is a 1-tensor. (b) A matrix is a 2-tensor. (c) An n-tensor.

Figure 4.1: Examples of tensors.

A *tensor network* is a collection of tensors together with identifications among the open indices of the tensors. Throughout the paper, we restrict ourselves to *closed* tensor networks. These are tensor networks representing a scalar obtained by summing over all identified open indices. See Figure 4.2 for an example. The *shape* of a tensor network is the information given by the ranks of each tensor, together with the identifications of the open indices. In particular, it is the information which tells you how to contract the tensor network, but not what the values of the tensors are.

$$\sum_{\beta,\gamma,\delta\in\{0,1\}} A_\beta B_{\beta\gamma} C_\beta D_{\beta\gamma} E_{\gamma\delta\delta} \quad =$$

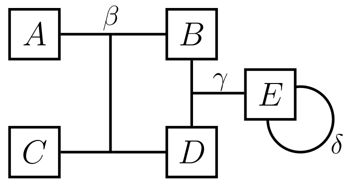


Figure 4.2: Example of a (closed) tensor network.

Monotone Arithmetic Circuits. In a monotone arithmetic circuit, each gate has fan-in degree 2 and is either a $+$-gate or a $\times$-gate. Furthermore, we assume that there is a single output node, see Figure 4.3 for an example.

Quantum Circuits. In a *quantum circuit* C, each gate has the same in- and out-degree, and represents a unitary matrix. The width of a quantum circuit $w(C)$, is the number of input nodes to the circuit. A special case of a quantum circuit is a *classical reversible* circuit, which will be introduced in Subsection 5.1.1.

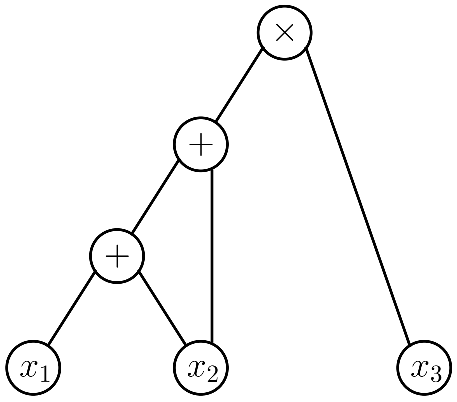


Figure 4.3: Example of a monotone arithmetic circuit computing the polynomial $(x_1 + 2x_2)x_3$.

4.3 Monotone method

When proving a lower bound, the model is at least as important as the bound itself. If the model is too restrictive, then the lower bound loses its worth. In this section, we introduce the *monotone method*, which describes a strong quantum simulation methodology. Although the model is restrictive enough to permit *unconditional* lower bounds, it also encompasses the majority of existing strong simulation techniques.

Before we define a monotone method rigorously, we describe it informally as a game. The game is played between a referee and you, and the aim of the game is to simulate a quantum circuit. First, the referee hands you a picture of a quantum circuit, but with some information missing. He has erased all of the nonzero coefficients in the gates appearing, replacing each with a different variable. You are allowed to spend as long as you like preparing your strategy; you may even use infinite time to do so.

When you are finally ready, you must commit to a *monotone* arithmetic circuit, with inputs given by the variables. Your monotone arithmetic circuit must simulate the original circuit with perfect accuracy *no matter what values the variables take*, and the time-complexity of your task is measured only by the length of your arithmetic circuit. With this game in mind, we set out to define a monotone method.

4.3.1 The skeleton of a tensor network

In the game we describe, the partial information about the circuit is very specific. Informally, we call the picture that the referee gives us the *skeleton* of the quantum circuit. More generally, we can consider the quantum circuit as a closed tensor network, and define an associated skeleton.

Skeleton of a Tensor Network. The *skeleton* of a tensor network is the hypergraph associated to the tensor network along with the locations of the nonzero entries in each tensor. Furthermore, we call the skeleton *closed* if the hypergraph is closed.

We further define the (closed) skeleton of a quantum circuit as the skeleton of its associated tensor network. To any *closed* skeleton, we can define an associated polynomial.

Associated Polynomial. Given a closed skeleton S, we can associate to it a polynomial $p(S)$ in the following way. First, replace each nonzero entry in each tensor appearing with a different variable. We can then regard S as a closed tensor network. Define $p(S)$ to be the polynomial obtained from contracting S.

See Figure 4.4 for an example of a quantum circuit, its skeleton, and its associated polynomial. Note that $p(S)$ is independent of the sequence of contractions and has nonnegative coefficients. To any quantum circuit C, we can also define the polynomial $p(C)$ to be the polynomial associated to its skeleton.

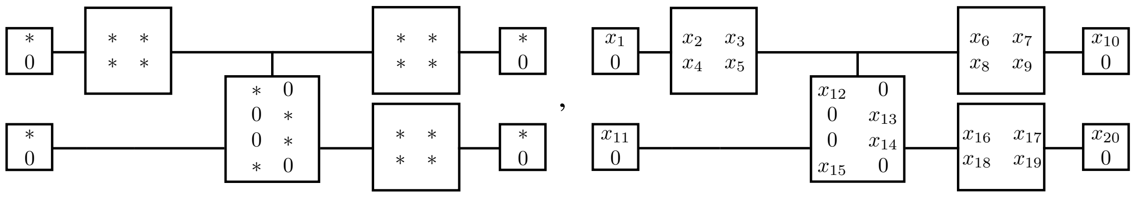


Figure 4.4: The left-hand diagram represents the skeleton associated to the quantum circuit $\langle 0|H_1H_2 \otimes CX_{1\to 2} \otimes H_1 \otimes I_2|0\rangle$, where the subscripts indicate the wires on which the gates act. According to the variables introduced in the right-hand diagram, its associated polynomial is $p(x_1,\ldots,x_{20}) = x_1x_{10}x_{11}x_{20}(x_2x_6x_{12}x_{16} + x_3x_8x_{14}x_{18})$.

4.3.2 Monotone methods

We will now define a monotone method. For any arithmetic circuit A, let $p(A)$ be its associated polynomial.

Monotone Method. A *monotone method* M is a mapping from closed skeletons to *monotone* arithmetic circuits that preserves the associated polynomials: for all closed skeletons S, $p(M(S)) = p(S)$. We define the *monotone complexity* of the skeleton S under the mapping M as $|M(S)|$. We can further extend a monotone method to a mapping on quantum circuits through their associated skeletons.

In particular, M itself can take exponential time to compute or even be non-uniform, allowing for uncomputable strategies. The complexity is measured only in terms of the complexity of the resulting arithmetic circuit. See Figure 4.5 for a high-level description relating strong quantum simulators to monotone methods.

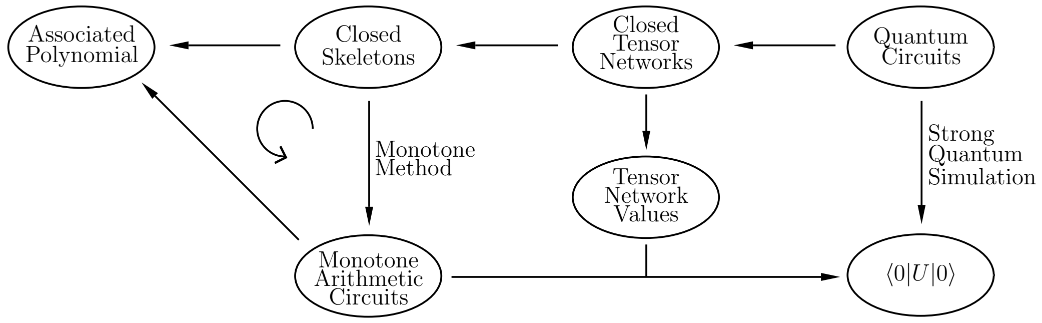


Figure 4.5: The relation between strong quantum simulation and monotone methods. A monotone method is any map from closed skeletons to monotone arithmetic circuits that makes the diagram commute.

To demonstrate the applicability of this model, we give a list of prominent strong simulation techniques, and show that they are all monotone methods.

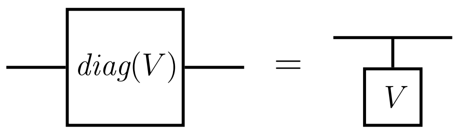
1. Feynman's path integral [85] is the simplest example of a monotone method applied to general tensor network contraction, involving only additions and multiplications of *all* coefficients appearing in the circuit.

2. Markov and Shi [84] proposed a tensor network contraction algorithm consisting of two phases. In the first phase, a nearly optimal ordering of contractions is decided. In the second phase, the tensor network is contracted accordingly. Given a general tensor network, finding the optimal ordering of contractions is an NP-hard problem; although there may be a contraction ordering with low complexity, finding it may take exponential time. One could carefully choose the tradeoff between these two phases so that the total complexity is split evenly.

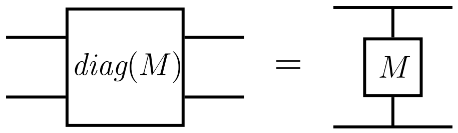
   In the second phase, contracting the tensors one by one is realized by a monotone arithmetic circuit. In particular, the algorithm itself is a monotone method. The complexity of *the second phase alone* is counted in the complexity of our model.

3. Several works [72, 71, 38, 73] preprocess the quantum circuit in a way that simplifies the tensor network, and ultimately results in a lower complexity during contraction. Specifically, they reduce the complexity of contracting diagonal gates. This preprocessing is oblivious to the nonzero coefficients of the gates. After deciding an order of contractions, the resulting procedure is realized by a monotone arithmetic

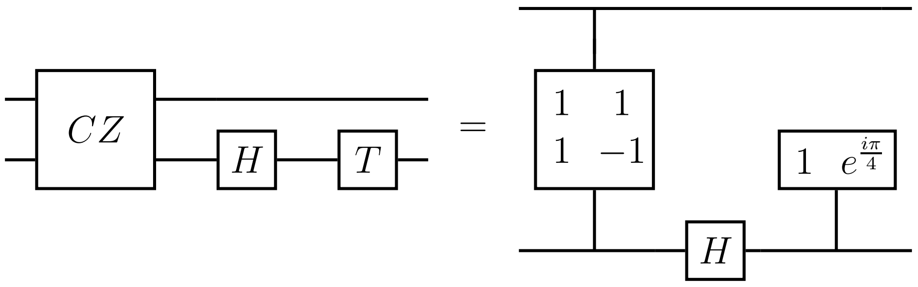
circuit with respect to the non-zero entries. The preprocessing step is illustrated in Figure 4.6.



(a) Simplification of a one-qubit diagonal gate.



(b) Simplification of a two-qubit diagonal gate.



(c) Naively contracting the left tensor network takes $32 + 8 = 40$ multiplication steps. Since the CZ and T gates are both diagonal, we can simplify the tensor network. The network on the right reduces the number of multiplications to $8 + 4 = 12$.

Figure 4.6: Preprocessing a tensor network to reduce the cost of contraction.

4. In [75], Aaronson and Chen propose a family of algorithms suited to different time-space trade-offs. These algorithms interpolate between naively evaluating the circuit, and implementing a variation of Savitch's theorem. Given a quantum circuit, one can decide the optimal algorithm according to the space and time constraints. Whichever algorithm you choose, the resulting process can be described by a monotone arithmetic circuit.

There are many more techniques that qualify as monotone methods, such as clever sparse matrix multiplication. However, there are also many tricks which do *not* fit into the monotone method. There exist tensor contractions which are non-monotone, first augmenting the network to introduce time-saving cancellations. Most famous among these is Strassen's fast matrix multiplication algorithm [86]. Recognizing that a circuit belongs to a restricted family such as Clifford or matchgate circuits can give exponential time savings [41, 87, 81, 88, 89]. Even recognizing small circuit identities may save on complexity [71], although this problem may be hard in general [90]. Finally, monotone methods are oblivious to the unitarity of the gates, which might be taken advantage of. However, for general quantum circuit simulation, we emphasize that the majority of savings are performed by tricks that fit within the monotone method framework.

4.4 An unconditional lower bound for monotone methods

We will now prove an *unconditional* lower bound on the time-complexity of a strong quantum simulator that uses a monotone method. The proof concept is simple: we will re-purpose the simulator to evaluate the permanent polynomial with constant overhead. Using existing unconditional lower bounds on monotone circuit evaluation of the permanent polynomial [91], we can impose unconditional lower bounds on the time-complexity of the simulator.

Permanent. The permanent of an $n \times n$ matrix M is given by $\sum_{\sigma \in S_n} \prod_{i=1}^{n} M_{i,\sigma(i)}$. The permanent polynomial for $n \times n$ matrices, given by replacing each entry M_{ij} of the matrix M by a distinct variable x_{ij}, is then

$$p(x_{11}, \ldots, x_{nn}) := \sum_{\sigma \in S_n} \prod_{i=1}^{n} x_{i,\sigma(i)}. \tag{4.1}$$

We note that there have been several works relating hard polynomial problems to the hardness of quantum circuits [92, 93], and the permanent polynomial is a particularly prominent example [94, 95].

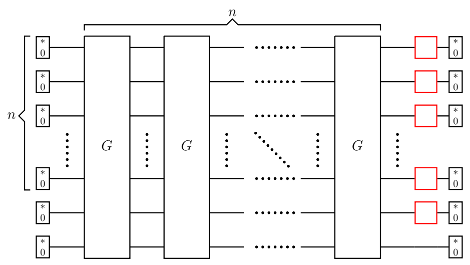
Theorem 4.1 (Jerrum & Snir)**.** *A monotone arithmetic circuit computing the permanent polynomial for $n \times n$ matrices must have size at least $n(2^{n-1} - 1)$.*

First, we construct a hard family of skeletons, parametrized by n. This family is hard in the sense that a monotone restriction of its associated polynomial computes the permanent of an $n \times n$ matrix. We then use Theorem 4.1 to lower bound the time-complexity of any monotone method applied to this skeleton. Finally, we find one (among many) quantum circuits with this skeleton. We conclude the following.

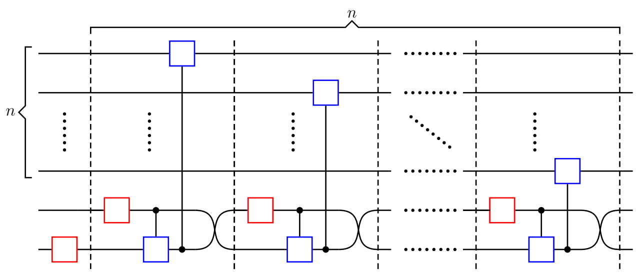
Theorem 4.2. *There exists a quantum circuit C of width $n + 2$ and depth $3n^2 + 1$ such that for any monotone method M, $|M(C)| \geq n(2^{n-1} - 1)$.*

We want to re-emphasize that this lower bound is *explicit* and *unconditional*.

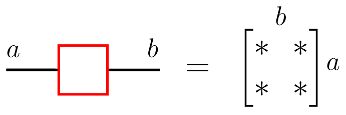
Proof of Theorem 4.2. Consider the skeleton S defined in the figure below.



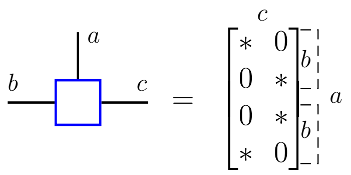
(a) Skeleton defined in terms of G.



(b) Gadget G used in Figure 4.7a.



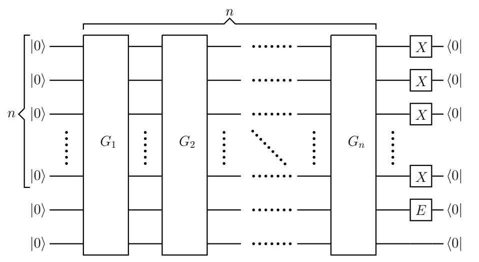
(c)



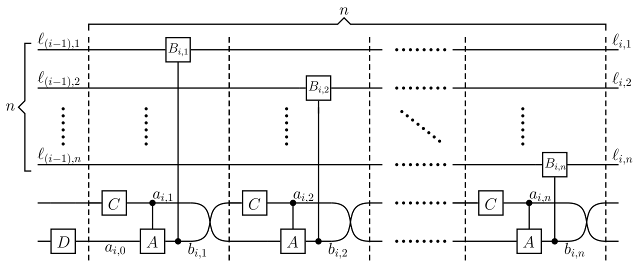
(d)

Figure 4.7: The skeleton we use to prove Theorem 4.2. The whole circuit is depicted in (a), and the gadget G is depicted in (b). The nonzero locations of the red component are shown in (c) and the nonzero locations of the blue component are shown in (d). To obtain a quantum circuit, we can replace the red components with Hadamard gates and the blue components with CNOT gates.

The polynomial $p(S)$ defined by the skeleton in Figure 4.7 is not the permanent polynomial. However, we can replace the variables of $p(S)$ with a combination of x_{ij}, 0, and 1 so that it becomes the permanent of the matrix (x_{ij}). Thus, if there were a monotone arithmetic circuit computing $p(S)$, then a potentially smaller circuit would compute the permanent polynomial. The replacements are shown in Figure 4.8. The pictorial proof that the resulting circuit computes the permanent polynomial can be found in Figure 4.9. □



(a) Tensor network T_{perm}.



(b) Gadget G_i in Figure 4.7d.

$$b - \overset{a}{\boxed{A}} - c = \begin{array}{c} c \\ \begin{bmatrix}1 & 0\\ 0 & 1\\ 0 & 0\\ 1 & 0\end{bmatrix} \end{array} \begin{array}{l} b \\ b \end{array} a \;,\quad b - \underset{a}{\boxed{B_{i,j}}} - c = \begin{array}{c} c \\ \begin{bmatrix}1 & 0\\ 0 & 1\\ 0 & x_{ij}\\ 0 & 0\end{bmatrix} \end{array} \begin{array}{l} b \\ b \end{array} a \;,\quad a - \boxed{X} - b = \begin{array}{c} b \\ \begin{bmatrix}0 & 1\\ 1 & 0\end{bmatrix} \end{array} a \;,$$

$$a - \boxed{C} - b = \begin{array}{c} b \\ \begin{bmatrix}1 & 1\\ 1 & 1\end{bmatrix} \end{array} a \;,\quad a - \boxed{D} - b = \begin{array}{c} b \\ \begin{bmatrix}0 & 0\\ 1 & 0\end{bmatrix} \end{array} a \;,\quad a - \boxed{E} - b = \begin{array}{c} b \\ \begin{bmatrix}1 & 0\\ 1 & 0\end{bmatrix} \end{array} a$$

Figure 4.8: The tensor network T_{perm} that realizes the permanent.

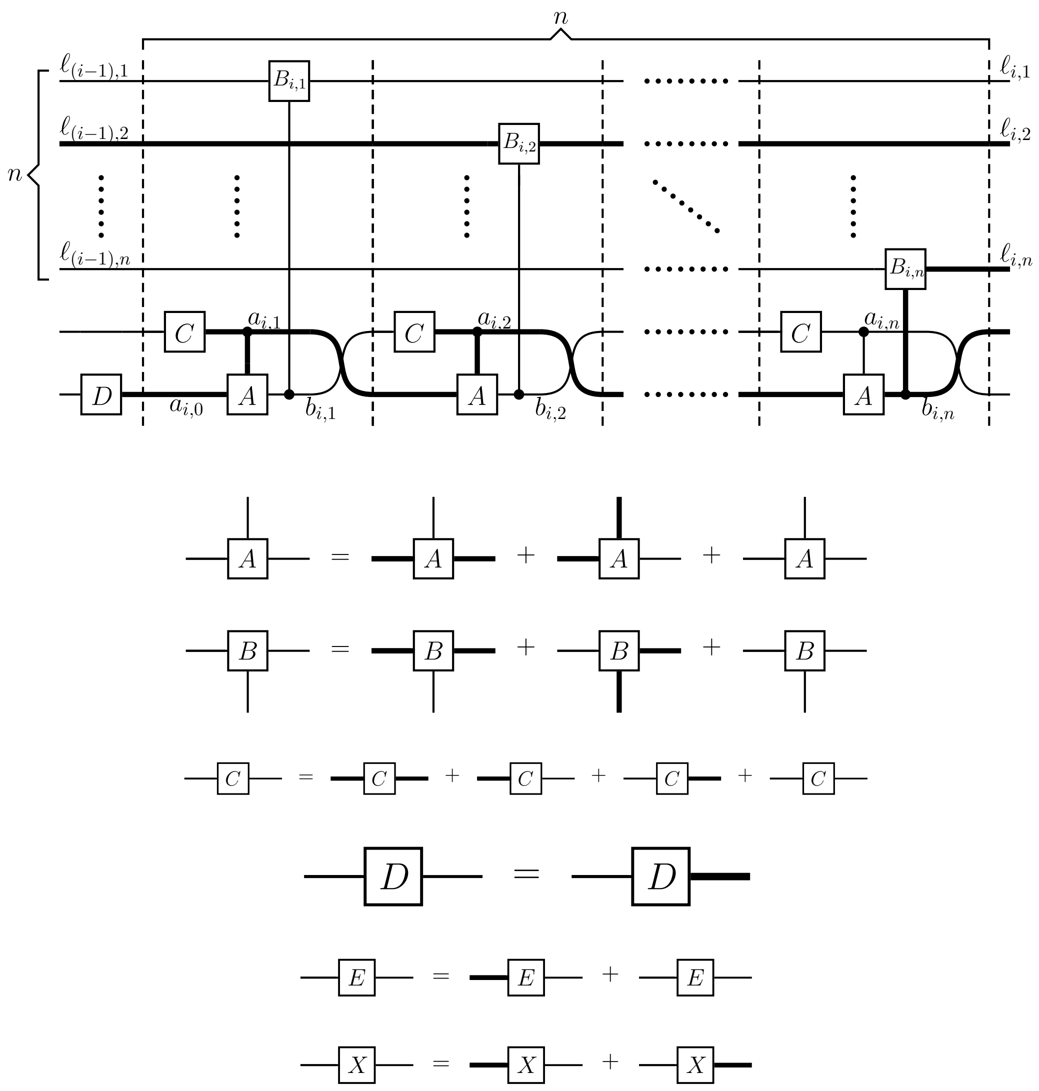


Figure 4.9: A pictorial proof of Theorem 4.2. To contract the network in Figure 4.8, first fix a labeling of the edges and then multiply together the corresponding tensor elements from each tensor. Then, sum over all such labelings. Note that if any of these tensor elements is zero, then the corresponding term in the sum is also zero. We now illustrate that there is a one-to-one correspondence between the nonzero terms in the sum and the terms in the n by n permanent. Namely for every permutation $\pi \in S_n$, the product $x_{1,\pi(1)} \dots x_{n,\pi(n)}$ appears as a nonzero term, and every other term is zero. In this figure, the wires with thicker lines are labeled 1, while the thinner lines are labeled 0. One can check that no other labeling contributes to the sum. To visualize this, we have listed all the nonzero labeling combinations for the A, B, C, D, E, and X tensors.

4.5 Remarks and open questions

In this chapter, we proved explicit lower bounds for existing strong simulation methods. Almost all prominent simulation methods fall into the class of monotone methods, which only focus on the locations of non-zero entries (i.e. the *skeleton* of the tensor network) and are oblivious and monotone with respect to the nonzero coefficients. We then reduce the problem of computing the permanent using monotone arithmetic circuits to evaluating a quantum circuit using monotone methods. Therefore, the hardness of the former implies the hardness of the latter.

It is worth noting that probably *most* skeletons have large monotone complexity using *any* monotone method, and our reduction to a skeleton for the permanent is just a choice. We choose the permanent as one of the few candidates for which an unconditional monotone lower bound is known. Also, the result focuses on the worst-case complexity of strong simulation by artificially constructing hard instances that cannot be simulated efficiently. For example, we reduce to the permanent to apply a known unconditional lower-bound to a particular closed skeleton. While we can probably show that *many* closed skeletons are hard by manipulating the proof of [91], proving average-case hardness for general strong simulation might require a worst- to average-case reduction, similar to [93].

Also worth noting is that many hard skeletons may be realized by easy circuits. In the proof of Theorem 4.2, the quantum circuit we construct from Figure 4.7 is Clifford, and so it can be simulated in *polynomial* time classically. Probably, there are quantum circuits which realize the same skeleton and are fundamentally hard to simulate. Monotone methods cannot distinguish between the two: they are oblivious to certain circuit structures altogether.

Interestingly, in the proof of Theorem 4.2, contracting the tensor T_{perm} from left to right yields a monotone circuit computing the permanent in time $O(n^2 2^n)$. To our knowledge, this is the fastest such monotone algorithm. If restricted to monotone circuits, naive enumeration over all permutations takes time $\Omega(n!)$, whereas a Savitch-type trick (as in [75]) reduces the time complexity to $O(4^n)$. The fastest general algorithms computing the permanent are non-monotone, and run in roughly $O(n2^n)$ time [96, 97]. It would be interesting if our method could be modified to yield a monotone circuit matching the lower bound of $n(2^{n-1} - 1)$.

We have considered only the total time-complexity of a simulator. More generally, one may be interested in restricting to space-efficient simulators. Along this line, there is the Feynman path-integral which has time complexity $O(2^{nd})$. More recently, Aaronson and Chen [75] use Savitch's Theorem to show that one can achieve an $O(d^n)$ time-complexity.

Does there then exist an even faster space-efficient strong simulator with time-complexity $O(d \cdot 2^n)$? Can we fine-tune such a simulator to achieve a general time-space tradeoff, as in [75]?

CHAPTER 5

Limitations of general strong quantum simulations

In this chapter, we continue the topic of explicit lower bounds for strong simulation proposed in Chapter 4. We investigate the time complexity of general strong simulation methods which are not necessarily monotone, by relating the problem of strong simulation to widely believed computational assumptions. Assuming the Strong Exponential Time Hypothesis, we remark that there exist simple quantum circuits for which any strong simulator with accuracy $2^n/2$ must take at least $2^{n-o(n)}$ time to simulate. We further compare strong simulators to state-of-the-art SAT solvers. We identify parameter thresholds on strong simulators above which such a simulator would imply immediate gains on existing solvers.

We further address the hardness of simulation for Clifford+T circuits in terms of the number of T gates, where we employ the sparsification lemma [98] to identify the efficiency threshold below which a strong simulator would improve on state-of-the-art 3-SAT solving. In particular, assuming the regular Exponential Time Hypothesis, we conclude that strong simulation also takes time exponential in the T-count. Regarding strong simulation of Clifford+T circuits with low T-count, a non-monotone strong simulator has been constructed by Bravyi and Gosset [81] (and extended in [82]) with a relevant exponential factor of $2^{0.47N}$, N being the number of T-gates (also called T-count) of a given circuit.

While these bounds are concerned with algorithmic complexity rather than clever memory allocation and parallelization, they indicate that strong simulation of hundreds of qubits is fundamentally intractable. In [81, 82], a weak simulator is proposed with a relevant exponential factor of $2^{0.23N}$. Although such a simulator does not offer an advantage in simulating general quantum circuits, it illustrates the advantage of weak simulation. Our evidence for the fundamental hardness of strong simulation indicates that intrinsically different *weak* simulators *must* be the focus in order to scale up.

5.1 Conditional lower bounds for strong simulators with respect to number of qubits

Theorem 4.2 only holds for monotone simulation methods. To prove a super-polynomial lower bound on general strong simulation, we turn to a conditional hardness argument[1]. In this section we prove that $2^{n-o(n)}$ is a lower bound for *any* strong simulator with accuracy 2^{-n} under the Strong Exponential Time Hypothesis (SETH).

Recall that conditional hardness always takes the form: if problem A is hard then problem B is hard; although a conditional hardness proof does not deliver absolute evidence for the hardness of B, it is protected by the hardness of A. The best choice for A is then a problem for which we have overwhelming evidence of hardness.

One of the most studied problems in computer science is the SAT problem together with its special cases, k-SAT for different k.

The SAT Problem. The input for SAT with size parameters n and m is a Boolean formula

$$\phi(x_1, \ldots, x_n) = \bigwedge_{i=1}^{m} \left(\bigvee_{j=1}^{k_i} l_{i,j} \right) \qquad l_{i,j} \in \{x_1, \ldots, x_n, \neg x_1, \ldots, \neg x_n\} \tag{5.1}$$

where $l_{i,j}$ are called *literals* and the sub-expressions $C_i = \bigvee_{j=1}^{k_i} l_{ij}$ are called *clauses*. Without loss of generality we require that every variable has at most one occurrence in every clause, implying $k_i \leq n$ for $1 \leq i \leq m$. We are interested in formulas ϕ with polynomial length: $m = n^{O(1)}$.

Given ϕ, the task is to determine whether there exists an assignment $x_i \to \{0, 1\}$ $(1 \leq i \leq n)$ such that $\phi(x_1, \ldots, x_n) = 1$, i.e. if ϕ is satisfiable. For this specific task, one can assume that each clause of the input problem is of length at least 2 since all singleton clauses can be easily eliminated.

The k-SAT Problem. The k-SAT problem is the SAT problem with the restriction that every instance ϕ has every k_i at most k.

The following two questions are unresolved.

Problem 5.1. *Can the 3-SAT problem be solved in time $(1+\epsilon)^n \mathrm{poly}(m)$ for every $\epsilon > 0$? Algorithms have been repeatedly improved upon [99, 100, 101, 102] to reach the current state of the art at around 1.3^n time.*

[1]Note that an unconditional proof would yield an advance in one of the hardest problems in theoretical computer science, showing $P \neq \#P$; no algorithm can count the number of solutions of Boolean expressions of size n in time $n^{O(1)}$.

Problem 5.2. *Can the SAT problem be solved in time* $2^{\alpha n}\text{poly}(m)$ *for some* $\alpha < 1$*, when* $m = poly(n)$*? The best current algorithm [103] runs in time* $\frac{2^n}{2^{n/O(\log m/n)}}$.

An improved algorithm for SAT would make a tremendous impact on many areas of computer science. This has lead to the somewhat widespread belief that reducing the time-complexity of the SAT problem will hit a hard limit. This belief has been formalized in the following two commonly held hypotheses.

Exponential Time Hypothesis (ETH). The answer to Problem 5.1 is no. There is an $\epsilon > 0$ such that the time complexity of 3-SAT is at least $(1+\epsilon)^n\text{poly}(m)$.

Strong Exponential Time Hypothesis (SETH). The answer to Problem 5.2 is no. Any algorithm deciding SAT must take at least $2^{n-o(n)}\text{poly}(m)$ time.

We show that if a strong quantum simulator can reach a certain efficiency, then the SETH would be refuted. This efficiency is quantified in terms of both the time-complexity and accuracy of the simulator.

Theorem 5.1. *Assume the SETH holds. Then a universal quantum simulator which can approximate* any *output amplitude to precision* $2^{-n}/2$ *on a quantum circuit with poly*(n) *operations must take at least* $2^{n-o(n)}$ *time.*

The proof of Theorem 5.1 uses a reduction from SAT to argue that if the simulator can determine an amplitude up to a certain precision, then it could solve a corresponding SAT problem. The reduction can be summarized as follows.

(i) For a SAT instance ϕ construct a reversible circuit $\mathcal{C}' = \mathcal{C}'_\phi$ with sub-linear space overhead and polynomial time overhead which can compute $\phi(x)$ for an assignment x.

(ii) Choose a basis state (e.g. $|0\dots0\rangle$) which counts the number of assignments satisfying ϕ in its amplitude when running C_ϕ, a quantum circuit constructed from $\mathcal{C}'$:

$$\langle 0\dots0|\mathcal{C}_\phi|0\dots0\rangle = \frac{|\{x \in \{0,1\}^n \mid \phi(x) = 1\}|}{2^n} \tag{5.2}$$

Step (i) is purely classical while step (ii) utilizes the quantum power of the simulator. Any lower-bound on the time-complexity of SAT then implies a lower bound on the time complexity of the simulator. Theorem 5.1 utilizes the strongest conjectured lower bounds available for SAT to imply the sharpest conjectured lower bounds for the parameters of a strong simulator. To push our bounds even further, we determine the threshold parameters beyond which a simulator would improve upon the best known algorithms solving SAT.

Theorem 5.2. *Assume there is a strong simulator that runs in time* $\frac{2^n}{2^{n/o(\log m/n)}}$. *Then this would improve on the running time of the best SAT solver.*

In section 5.1.1 we address point (i). Next, in section 5.1.2, we address point (ii) and prove Theorems 5.1 and 5.2.

5.1.1 Reversible evaluation of a SAT formula

In this section we construct a reversible circuit $\mathcal{C}'$ evaluating a given SAT formula using sub-linear space overhead and polynomial time overhead. This problem was first efficiently solved by Charles H. Bennett in 1989 [104]. For the sake of completeness and explicit constants, we reproduce the argument here, but emphasize that our construction follows [104]. Bennett expresses the problem in the language of Turing Machines rather than circuits,[2] but the proof ideas are otherwise identical.

Reversible Circuits. A reversible classical circuit consists of reversible gates. A reversible classical gate F is simply an invertible function $F : \{0,1\}^d \to \{0,1\}^d$, for some d. Typically d is one, two, or three. An important example of a reversible classical gate is the Toffoli gate $\mathrm{TOFFOLI}(x,y,z) = (x,y,z \oplus (x \wedge y))$ acting on 3 bits.

Universal Gate Set. Throughout, our choice of universal gate set for reversible computation is

$$\mathcal{G} = \{\mathrm{TOFFOLI}, \mathrm{CNOT}, \mathrm{NOT}\}. \tag{5.3}$$

The above choice influences the circuits for which our lower bound argument holds up to constants. The particular gate set we have chosen has the nice property that all its elements are self-inverse.

Tidy Computation. We say a reversible circuit $C : \{0,1\}^{n+a(n)+1} \to \{0,1\}^{n+a(n)+1}$ *tidily computes* a function $f : \{0,1\}^n \to \{0,1\}$ if

$$\forall\, x \in \{0,1\}^n, y \in \{0,1\}, \quad C(x, 0^{a(n)}, y) = (x, 0^{a(n)}, y \oplus f(x)). \tag{5.4}$$

We call the $a(n)$ extra bits the *ancilla* bits. The tidiness comes from the fact that the input and the ancilla bits are restored by the end of the computation. From the perspective of quantum circuits, every reversible gate can be seen as a unitary transformation. By linearity, the action of a reversible circuit C computing f in Equation (5.4) can be extended to an action on the Hilbert space $C^{\{0,1\}^n}$:

[2]A uniform rather than non-uniform computational device.

$$C \quad \text{applied to} \quad \textstyle\sum_x \alpha_x |x, 0^{a(n)}, 0\rangle \quad \text{is} \quad \sum_x \alpha_x |x, 0^{a(n)}, f(x)\rangle.$$

Note that the ancilla bits are decoupled from the input and output registers.

Lemma 5.1 (**MAIN**, [104]). *Suppose ϕ is a SAT formula with n variables and m clauses. Then there is a circuit $\mathcal{C}'$ that tidily computes ϕ with*

$$s(\mathcal{C}') \leq 8 \times 3^{\lceil \log n \rceil + \lceil \log m \rceil} - 1, \qquad w(\mathcal{C}') \leq n + 2(\lceil \log n \rceil + \lceil \log m \rceil). \tag{5.5}$$

Our proof scheme follows that in [104]. Notice that ϕ can be written as a binary tree with ANDs and ORs as in Figure 5.1. Thus, we will first implement ANDs and ORs of circuits in a reversible way. We will then recursively compose $\mathcal{C}' = \mathcal{C}'_\phi$ by exploiting the binary tree structure. In the rest of the section, we will prove Lemma 5.1 using these steps. To get the best constants we first define the following.

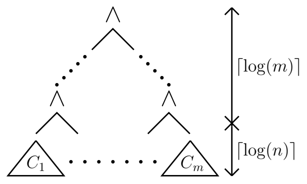


Figure 5.1: SAT instance ϕ expressed as a binary tree.

Untidy Computation. We say that a reversible circuit $C : \{0,1\}^{n+a(n)+1} \to \{0,1\}^{n+a(n)+1}$ *untidily computes* a function $f : \{0,1\}^n \to \{0,1\}$ if $C(x,0,0) = (*,*,f(x))$ for all $x \in \{0,1\}^n$.

Lemma 5.2. *We can convert a circuit U that untidily computes f into a circuit that tidily computes f. The conversion doubles the size of the circuit along with adding one extra CNOT gate and one extra ancilla wire, as shown in Figure 5.2.*

Proof. The circuit in Figure 5.2 executes UGU^{-1}, where G is a CNOT gate acting on the new output wire (b in the figure), and controlled by the old output (a in the figure). More precisely,

$$|x, 0, \underbrace{0}_{a}, \underbrace{0}_{b}\rangle \xrightarrow{U} |\alpha, \beta, f(x), 0\rangle \xrightarrow{G} |\alpha, \beta, f(x), f(x)\rangle \xrightarrow{U^{-1}} |x, 0, 0, f(x)\rangle$$

$$|x, 0, \underbrace{0}_{a}, \underbrace{1}_{b}\rangle \xrightarrow{U} |\alpha, \beta, f(x), 1\rangle \xrightarrow{G} |\alpha, \beta, f(x), \neg f(x)\rangle \xrightarrow{U^{-1}} |x, 0, 0, \neg f(x)\rangle$$

Observe that the added wire carries the output, while the original output wire serves as an ancilla. □

The main building blocks of $\mathcal{C}'$ that implement binary AND and OR as reversible circuits come from the following lemma.

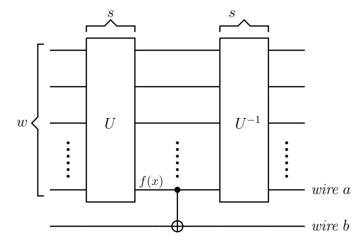
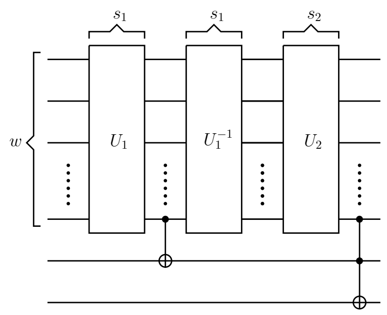


Figure 5.2: The construction that restores the input and ancilla wires after an untidy computation of $f(x)$ and produces the output on the added wire b. Initially, the output of U was sent to wire a.

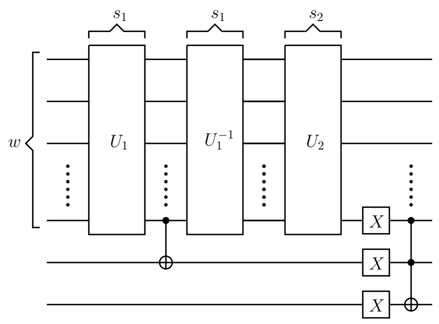
Lemma 5.3. *Suppose U_1 and U_2 untidily compute f_1 and f_2 with width at most w and sizes s_1 and s_2, respectively. Then there is a circuit of width at most $w + 2$ and size $2s_1 + s_2 + 2$ that untidily computes $f_1 \wedge f_2$, and a circuit of width at most $w + 2$ and size $2s_1 + s_2 + 5$ that untidily computes $f_1 \vee f_2$.*

Note that in Lemma 5.3, we could have performed tidy computations rather than untidy computations with meager overhead. We avoid this for two reasons. First, the untidy computation has a simpler circuit. Second, the constant factor loss will culminate in a polynomial loss due to the iterative construction in the proof of Theorem 5.1. Thus, it is more economical to untidily compute everything until the end, and then make the circuit tidy by employing Lemma 5.2. The bounds are specific to the gate set $\mathcal{G}$, which results in the slight asymmetry between the AND and OR circuits.

Proof of Lemma 5.3. The reversible AND and OR of the circuits, with the appropriate sizes and widths, are constructed explicitly below.



Circuit for $f_1 \wedge f_2$



Circuit for $f_1 \vee f_2$

□

Corollary 5.1. *Untidily computing $f_1 \wedge \ldots \wedge f_n$ takes width at most $w + 2\lceil \log n \rceil$ and size at most $3^{\lceil \log_2 n \rceil}(s + 1) - 1$, where s and w are the simultaneous size and width upper bounds for circuits that untidily-compute the set of f_i.*

Corollary 5.2. *Untidily computing* $f_1 \vee \ldots \vee f_n$ *takes width at most* $w + 2\lceil \log n \rceil$ *and size at most* $3^{\lceil \log_2 n \rceil}(s + \frac{5}{2}) - \frac{5}{2}$.

The Reversible Circuit for $\phi = C_1 \wedge \ldots \wedge C_m$**.** We first create circuits $\mathcal{C}_1, \ldots, \mathcal{C}_m$ that untidily compute clauses $C_1, \ldots, C_m$, using Corollary 5.2. From these circuits, we then construct a reversible circuit that untidily computes their conjunction, using Corollary 5.1. The resulting circuit will untidily compute ϕ with the following parameters.

Corollary 5.3. *Suppose* ϕ *has* n *variables and* m *clauses. Then we can untidily compute* ϕ *using a reversible circuit of width at most* $w + 2(\lceil \log n \rceil + \lceil \log m \rceil)$ *and size at most* $4 \times 3^{\lceil \log_2 n \rceil + \lceil \log_2 m \rceil} - 1$.

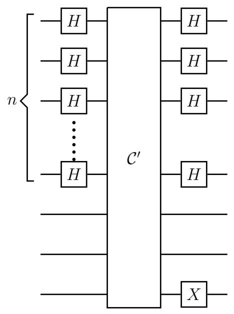
Finally, the centerpiece Lemma 5.1 then follows from applying Lemma 5.2 to the untidy circuit in Corollary 5.3.

5.1.2 Reducing SAT to strong simulation

Given a SAT instance ϕ with n variables and m clauses, we would like to construct a quantum circuit $\mathcal{C}_\phi$ so that

$$\vartheta := \langle 0 \dots 0|\mathcal{C}_\phi|0 \dots 0\rangle = \frac{|\{x \in \{0,1\}^n \mid \phi(x) = 1\}|}{2^n}. \tag{5.6}$$

Let $\mathcal{C}'_\phi$ be a (classical) reversible circuit coming from Lemma 5.1 that tidily computes ϕ. Then the quantum circuit $\mathcal{C}_\phi$ on the right satisfies Equation 5.6.



The construction of $\mathcal{C}_\phi$, where $\mathcal{C}'$ comes from Section 5.4.

5.1.3 Relating the parameters

By comparing the parameters of ϕ and $\mathcal{C}_\phi$, we can tie the complexity of the approximate strong simulation problem to the complexity of the SAT problem.

Proof of Theorem 5.1. Suppose we had a strong simulator that could approximate ϑ to within an additive error of $2^{-n}/2$. Then running the simulator on $\mathcal{C}_\phi$, we would be able to tell if ϕ is satisfiable or not: if $\vartheta < 2^{-n}/2$ then ϕ is not satisfiable, otherwise it is.

Let the number of qubits of $\mathcal{C}_\phi$ be $n' = w(\mathcal{C}_\phi)$ and its size be $s = s(\mathcal{C}_\phi)$. Suppose we could run this simulator in time $2^{(1-c)n'}s^{O(1)}$ for some $c > 0$. Then, substituting the bounds for $w(\mathcal{C}_\phi)$ and $s(\mathcal{C}_\phi)$ in Lemma 5.1, we obtain a running time of

$$2^{(1-c)(n+2\log n+2\log m)}(nm)^{\log_2 3} = 2^{(1-c)n}(mn)^{O(1)}. \tag{5.7}$$

This would contradict the SETH. □

Proof of Theorem 5.2. Suppose that our simulator can approximate amplitude ϑ to within accuracy $2^{-n}/2$ and in time $\frac{2^n}{2^{n/o(\log m/n)}}$. Then this simulator would solve the SAT problem in time $\frac{2^{n+2\log n+2\log m}}{2^{n/o(\log m/n)}}$, which is better than $\frac{2^n}{2^{n/O(\log m/n)}}$, beating the currently known best SAT solver [103]. □

Note that a simple padding argument further extends Theorem 5.1 to the following.

Corollary 5.4. *Assume the SETH and let $0 < \alpha \leq 1$. Then any strong simulator with approximation precision $2^{-\alpha n}/2$ must take $2^{\alpha n - o(n)}$ time.*

5.2 Conditional lower bounds in terms of T-gate count

Recent results [81, 82] have shown that a quantum circuit can be strongly simulated in time $O^*(2^{0.47N})$, where N is the number of $T = Z^{1/4}$-gates in an otherwise Clifford circuit[3]. As Clifford+T gates form a universal gate set, this simulation method yields a substantial speed-up on circuits that are predominantly Clifford. However, we show that such strong simulation method would necessarily scale exponentially with respect to the number of T-gates.

Theorem 5.3. *Assuming the Exponential Time Hypothesis (ETH), there is an $\epsilon > 0$ such that any strong simulation that can determine if $\langle 0|C|0|\rangle \neq 0$ of a polynomial-sized quantum circuit C formed from the Clifford+T gate set with N T-gates takes time at least $2^{\epsilon N}$.*

To express an explicit lower bound, we have the following theorem.

Theorem 5.4. *Assume that there exists a strong simulator that, for any Clifford+T circuit with N T-gates, can determine if $\langle 0|C|0\rangle \neq 0$ in time $O(2^{2.2451\times 10^{-8}N})$. Then this would improve on the current state-of-the-art* 3-SAT *solver by achieving an $O(1.3^n)$ runtime for $m = \text{poly}(n)$, where n denotes the number of variables of the* 3-SAT *instance and m denotes the number of clauses.*

Theorem 5.3 relies on the following.

Lemma 5.4 (Corollary 2, [98])**.** *Assuming the ETH, there exists constant $a > 0$ such that any classical algorithm solving* 3-SAT *instances with length L takes 2^{aL} time, where again L is the* length *of the formula.*

Finally, to compute explicit constants, Theorem 5.4 requires the following.

Lemma 5.5. *Assume that a classical algorithm solves* 3-SAT *in time $O(2^{3.1432\times 10^{-7}L})$, where L is the length of the formula, m_2 is the number of 2-clauses, m_3 is the number of 3-clauses, so that $m = m_2 + m_3$ and $L = 2m_2 + 3m_3 - 1$.*

Then one can create a 3-SAT *solver that achieves an $O(1.3^n)$ running-time for $m = \text{poly}(n)$, where n denotes the number of variables of the* 3-SAT *instance and m denotes the number of clauses.*

The rest of the section will be arranged as follows. We first describe the sparsification lemma, which was developed in [98] to address the type of reduction found in Lemmas 5.4

[3]With slight abuse of notation, we also allow the inverse of the T-gate ($T^\dagger = P^\dagger T$) in the gate set and define the T-count of the circuit to be the number of T and $T^\dagger$ gates altogether.

and 5.5. The necessity of the sparsification lemma comes from reducing the 3-SAT problem to quantum circuits, as the number of T-gates in the reduced instance will depend on L, the *length* of the 3-SAT instance, rather than n, the *number of variables*. The reduction from 3-SAT to strong simulation of quantum circuits is described at the end of this section, while a full proof of the sparsification lemma with explicit constants is provided in the Appendix B.

5.2.1 The sparsification lemma

Recall that 3-SAT problems are SAT problems with the promise that each clause contains at most 3 literals. The *length* L of an instance is the number of AND/OR gates in the formula, namely $L = \sum_{i=1}^{m} k_i - 1$. We are interested in formulae ϕ with polynomial length. For 3-SAT, we have $L = 2m_2 + 3m_3 - 1$ where m_2 is the number of 2-clauses and m_3 is the number of 3-clauses.

Due to subsequent improvements [99, 100, 101, 102] the current state-of-the-art 3-SAT solver takes time $\approx 1.3^n$ (and in fact, the precise exponent is slightly worse). A classical algorithm breaking this bound would have a huge impact on theoretical computer science.

When trying to lower bound the running time of a 3-SAT solver in terms of L and not of n, the ETH initially seems to be of little help, as L can be as large cubic in n. The following sparsification Lemma, however, gives the desired n to L conversion.

Lemma 5.6 (Sparsification lemma, [98])**.** *Given any $\epsilon > 0$, there is an algorithm A_ϵ that, on any* 3-SAT *instance ϕ with n variables, outputs a list $\ell = \phi_1, \cdots, \phi_k$ of* 3-SAT *instances in time $O_\epsilon(2^{\epsilon n})$, satisfying:*

- *$k \leq O_\epsilon(2^{\epsilon n})$;*
- *each formula ϕ_i has length at most $c(\epsilon)n$, where $c(\epsilon)$ does not depend on n;*
- *ϕ is satisfiable if and only if one of the generated sub-instances are satisfiable: $\phi = \bigvee_{i=1}^{l} \phi_i$.*

Among the consequences of the sparsification lemma are Lemmas 5.4 and 5.5. To prove Lemma 5.5, we must also compute the constants implicit in the sparsification lemma. These can be found in the Appendix.

5.2.2 From 3-SAT to Clifford+T

Here we give a reduction from the 3-SAT problem to the problem of strongly simulating quantum circuits, with the following properties.

(i) For a 3-SAT instance ϕ with length L, we construct a quantum circuit $\mathcal{C}'_\phi$ with $2L$ Toffoli gates and $\mathrm{poly}(L)$ NOT and CNOT gates, which can compute $\phi(x)$ for any assignment x.

(ii) We choose a basis state (e.g. $|0\ldots0\rangle$) which counts the number of assignments satisfying ϕ in its amplitude when running C_ϕ, a quantum circuit constructed from $\mathcal{C}'_\phi$, satisfying

$$\langle 0\ldots0|\mathcal{C}_\phi|0\ldots0\rangle = \frac{|\{x \in \{0,1\}^n \mid \phi(x) = 1\}|}{2^n}. \tag{5.8}$$

Lemma 5.7. *Suppose ϕ is a SAT formula with length L. Then, in time polynomial in L, we can construct a reversible circuit $\mathcal{C}'$ that computes ϕ with at most $2L$ Toffoli gates such that all ancilla bits are restored to their original $|0\rangle$ state.*

Before proving Lemma 5.7, we first define a specific form of reversible computation called diagonal computation.

Diagonal Computation. We say that a reversible circuit $C : \{0,1\}^n \to \{0,1\}^{n+a(n)+1}$ *diagonally computes* a function $f : \{0,1\}^n \to \{0,1\}$ if for all x, $C(x,0,0) = (x,*,f(x))$.

Diagonal computation is a special type of untidy computation, where the inputs on the input wires are preserved after the computation. This helps us compose diagonal computation circuits in a gate-efficient manner. The name diagonal comes from the fact that diagonal computation can be regarded as a controlled unitary of a quantum circuit, thus being block-diagonal in its matrix form.

Lemma 5.8. *Suppose U_1 and U_2 diagonally compute f_1 and f_2 with at most a_1 and a_2 ancilla wires and t_1 and t_2 Toffoli gates, respectively, over the same set of input wires. Then there exists:*

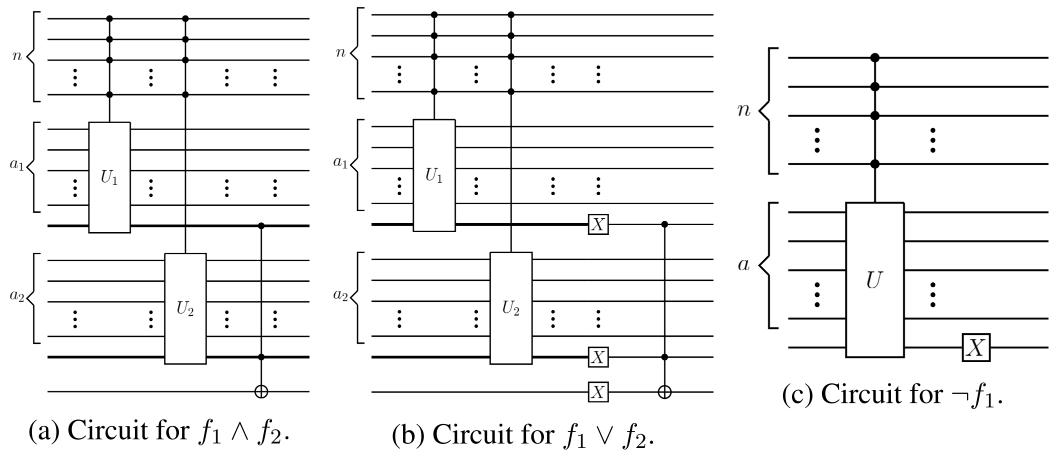
(a) a circuit with at most $a_1 + a_2 + 2$ ancilla wires and $t_1 + t_2$ Toffoli gates that diagonally computes $f_1 \wedge f_2$,

(b) a circuit with at most $a_1 + a_2 + 2$ ancilla wires and $t_1 + t_2$ Toffoli gates that diagonally computes $f_1 \vee f_2$,

(c) a circuit with at most a_1 ancilla wires and t_1 Toffoli gates that diagonally computes $\neg f_1$, and

(d) a circuit with at most $a_1 + 1$ ancilla wires and $2t_1$ Toffoli that tidily computes f_1.

Proof of Lemma 5.8. The reversible diagonal AND, OR and NOT of the circuits, with the appropriate sizes and widths, are constructed explicitly below. Moreover, a circuit diagonally computing f_1 is also untidily computing f_1, and so (d) can be proven by converting untidy circuits to tidy circuits.



(a) Circuit for $f_1 \wedge f_2$. (b) Circuit for $f_1 \vee f_2$. (c) Circuit for $\neg f_1$.

□

Property (i) is a direct application of Lemma 5.7. For property (ii), we use the following standard Toffoli decomposition [32].

Lemma 5.9. *A Toffoli gate can be written as composition of* 7 *T-gates and* 8 *Clifford gates.*

Proof of Lemma 5.9. The circuit computing a Toffoli gate with 7 T-gates and 8 Clifford gates is explicitly constructed below in Fig. 5.4.

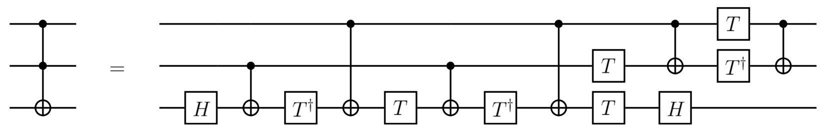


Figure 5.4: Explicit decomposition of the Toffoli gate into Clifford+T gates with minimum T-count.

□

Combining Lemmas 5.7 and 5.9, we obtain the following.

Corollary 5.5. *Given a* 3-SAT *formula* ϕ *with* n *variables and length* L*, one can efficiently construct a quantum circuit* C_ϕ *consisting of only Clifford gates and at most* $14L$ *T-gates, so that*

$$\langle 0|C_\phi|0\rangle = \frac{|\{x \in \{0,1\}^n | \phi(x) = 1\}|}{2^n}. \tag{5.9}$$

Consequently, strong simulation of such a circuit up to accuracy $2^{-n}/2$ is as hard as determining whether ϕ is satisfiable.

Proof of Theorem 5.3. Assume that Theorem 5.3 is false. Then for any $\epsilon > 0$ we have a quantum circuit simulator that runs in time $O(2^{\frac{\epsilon}{14}N})$, where N is the number of T-gates and the size of the circuit is $O(N)$. If we apply the reduction in Corollary 5.5, then we can solve the 3-SAT problem in time $O(2^{\epsilon L})$, contradicting Lemma 5.4. $\square$

Proof of Theorem 5.4. According to Corollary 5.5 a 3-SAT instance of length L reduces to the strong simulation problem of a linear size quantum circuit with Clifford gates and at most $N(= 14L)$ T-gates. If there were a solution to the simulation problem with running time $O(2^{2.2451\times10^{-8}N})$, then any 3-SAT instance of length L could be solved in time $O(2^{2.2451\times10^{-8}\times14L}) = O(2^{3.1432\times10^{-7}L})$, contradicting Lemma 5.5. $\square$

5.3 Conclusion

5.3.1 Summary

For general strong simulators, we showed that they can be harnessed to solve the #SAT problem if the simulation method meets a certain accuracy. The #SAT problem is not only #P-hard; it is widely believed that solving #SAT takes about 2^n time under the Strong Exponential Time Hypothesis. This allows us to prove an *explicit* conditional lower bound on all strong simulation methods with high accuracy.

Recently, there has been focus on Clifford+T quantum circuit simulation, and more generally on other magic-state inspired simulation methods [82]. It is then natural to ask whether the running time could scale sub-exponentially in terms of T-gate count. We demonstrated that such a simulator would violate the Exponential Time Hypothesis. Moreover, we showed an explicit exponential lower bound which, if violated, would result in an improvement on state-of-the-art 3-SAT solving. Unfortunately, the overhead required to adapt strong simulation to 3-SAT solving greatly diminishes this constant. It would be of great practical interest to find a larger lower-bound for this exponent.

Furthermore, we would expect that similar arguments could be used to prove lower bounds on *stabilizer rank*, which is the quantity relevant to the running time of simulators in [81, 82]. However, we cannot immediately conclude exponential lower bounds on stabilizer ranks as computing the decomposition itself (rather than its exponential scaling) may be the bottleneck for strong simulation. Furthermore, as stabilizer rank lower bounds do not have obvious far-reaching complexity theoretic implications, it is reasonable to hope for unconditional restrictions [82].

Our bounds focus on general quantum circuits, showing that there exist hard instances with reasonable size that cannot be evaluated efficiently using current simulation methods. However, such reductions might not be applicable if the circuit is drawn from a restricted class (for example, circuits consisting of only Clifford gates). Proving that a certain restricted class of quantum circuits admits a hard instance requires additional effort [105]. We also suspect that most quantum circuits are hard to simulate in a stronger sense, but we leave this consideration to future work.

5.3.2 Open problems

Larger Additive Gaps. Theorem 5.1 addresses the hardness of strong simulation up to accuracy $O(2^{-n})$, but it could be that strong simulation up to accuracy $O(2^{-n/2})$ would also take $2^{n-o(n)}$ time. Approximation up to accuracy $2^{-n/2}$ is particularly interesting because $2^{-n/2}$ is the typical value of an amplitude over a randomly chosen basis vector. A more general question is determining, for some $a < b$, the complexity of deciding whether an amplitude is at most a or at least b. An avenue for proving such results may come from the hardness of approximating the #SAT problem.

Lower Bounds on Weak Simulators. Can we prove explicit lower bounds for weak simulators? It is well known that there can be no efficient weak simulator unless PH collapses to the third level. However, it is plausible that some weak simulators may run in subexponential (e.g. $2^{\sqrt{n}}$) time, which may be affordable in the near future.

Superior Weak Simulators. Given the compelling evidence that strong simulation is hard, can we design a weak simulator that runs general quantum circuits in time $o(d \cdot 2^n)$? The simulator constructed by Bravyi and Gosset [81] is an excellent example of the potential gains afforded by weak (versus strong) simulation.

CHAPTER 6

Finding angle sequences in quantum singal processing

Among the recent breakthroughs in the area of quantum algorithms, quantum signal processing receives much attention for its simplicity and extendability. First proposed in a seminal paper by Low and Chuang [106], quantum signal processing has been used to design various algorithms with optimal query complexity and/or gate complexity [106, 18, 23].

The QSP algorithm uses an ancilla qubit as a control qubit and a controlled version of any given unitary U. The goal of the algorithm is to perform a spectral transformation of the unitary U in a black-box manner, by only querying the controlled U rather than classically look into the matrix U. The central idea of the algorithm can be best described as alternating between applying a single qubit rotation on the ancilla wire, and applying a controlled U from the ancilla wire to the data qubits. Thanks to a rich algebraic structure theorem, all appropriately normalized polynomials of the unitary U can be implemented this way, given that the single qubit rotation angles are chosen carefully. Though theoretically appealing, one potential drawback of the QSP algorithm is the lack of a numerically stable algorithm of finding the desired single qubit rotations (called the *angle sequence*) given the polynomial to be implemented. When the degree of the polynomial we want to implement is high, one often needs to perform very high accuracy arithmetics, which could be infeasible in some scenarios.

In this chapter, we describe an algorithm for finding angle sequences in quantum signal processing, with a novel component we call *halving* based on a new algebraic uniqueness theorem, and another we call *capitalization*. Together, these two algorithmic ideas allow us to find angle sequences for important applications such as Hamiltonian simulation in standard double precision arithmetic, native to almost all hardware. The current best method [24] could find the same only in arbitrary precision arithmetic, which needed to be emulated by software, thus incurring a substantial time overhead. We present experimental results that demonstrate the performance of the new algorithm.

6.1 Introduction

Many recent works in quantum computation consider the problem of transforming the spectrum of an unknown unitary in a black box manner. In a sequence of works [8, 18, 19, 20, 21, 22, 23] an entire mathematical machinery was developed to address this problem. In particular, a novel paradigm called *qubitization* was introduced by [20, 21], which is an elegant technique based on *quantum signal processing* [17]. Quantum signal processing achieves polynomial transformations in a very efficient way using only a few ancilla qubits with the help of a sequence of single qubit rotation gates. In [23] these techniques were further developed to produce a Quantum Singular Value Transformation (QSVT) algorithm, with a wide range of applications from Hamiltonian simulation to Gibbs sampling. However, finding a sequence of angles for the rotation gates to achieve a desired transformation can be a challenging (classical) task [107]. In order to demonstrate the practicality of quantum algorithms based on quantum signal processing, it is important to show that there is an efficient classical algorithm to find the angles. This question was very recently addressed by Haah [24], who showed that this is indeed the case and that the problem can be treated more elegantly by considering Laurent polynomials, that is, polynomials with both positive and negative powers.

6.1.1 Main result

The main result of our work can be best summarized as a new algorithm for finding these angles, which shows surprising numerical stability. Specifically, our work presents novel contributions to this field in two ways:

1. We further develop the mathematics of quantum signal processing by defining and analyzing the algebras that naturally arise from the problem, which we identify as Cayley-Dickson algebras [108]. In particular, we prove a uniqueness of decomposition theorem in the algebra, which is the basis for a new algorithm we call *halving*.

2. We conduct numerical experiments implementing this new algorithm that also make use of a new method we call *capitalization*. Together, these two algorithmic ideas allow us to perform quantum signal processing for high degree polynomials in, for the first time, standard double precision arithmetic. The previous method of Haah [24] could only do this using high precision arithmetic. As this needs to be emulated by software, a substantial time overhead is incurred.

6.1.2 Application

One of the main applications of quantum signal processing is Hamiltonian simulation. The complexity of simulating Hamiltonians has been a highly studied topic [109, 110, 8, 106, 111], with many results achieving optimal complexity with respect to certain parameters. As quantum computing enters the noisy intermediate-scale quantum (NISQ) [10] era, it becomes increasingly important to optimize the size of the quantum circuits. The algorithmic techniques previously developed include Linear Combination of Unitaries [8] and Qubitization [20]. These techniques proved to be very useful for constructing efficient quantum algorithms for other problems as well. In our work we apply our algorithm to the Hamiltonian simulation problem, conducting numerical experiments simulating Hamiltonian evolution to time scales two or more orders of magnitude longer than what was previously possible [107].

6.2 Quantum signal processing

Quantum signal processing [17] is like a no-look pass in basketball: our goal is to build a quantum circuit that transforms a black box operator — we may call it the *signal* — without peeking into the box itself. The black box operator is in most cases a quantum circuit itself, so it is unitary. This will be our assumption throughout.

An example of signal processing is when we run W twice ($C = WW$). The new operator becomes W^2 regardless of what W is, and in the process we did not look at W. In this very simple case the Hilbert space, H, of our circuit C was the same as the Hilbert space of W — no ancilla wires were added. In more interesting cases we assume that we have access to a particular controlled version of W, which we denote by $\widetilde{W}$ and which acts on the Hilbert space $\widetilde{H} = \mathbb{C}^2 \otimes H$ as [24]:

$$\widetilde{W} = \begin{pmatrix} W & 0 \\ 0 & W^{-1} \end{pmatrix}. \tag{6.1}$$

By making good use of the extra *control wire* we can now design circuits that alternate between one-qubit actions on the control wire and $\widetilde{W}$ operations. It will be convenient to write our circuit as

$$C = M \cdot R'_d \widetilde{W} R'_{d-1} \cdots R'_1 \widetilde{W} \cdot R'_0. \tag{6.2}$$

where every R'_j is $R_j \otimes I_H$, and M is a post-selection operation:

Post-selection Operator. Consider a quantum circuit $C = M \cdot C'$ with one ancilla qubit and N input wires, which, setting the ancilla qubit to $|0\rangle$, runs

$$C' = \begin{pmatrix} C'_{00} & C'_{01} \\ C'_{10} & C'_{11} \end{pmatrix} \qquad C'_{00}, C'_{01}, C'_{10}, C'_{11} \in \mathrm{M}(2^N, \mathbb{C}) \tag{6.3}$$

and finally performs a post-selection operation M on the ancilla wire. This latter operation declares success if the ancilla is measured 0 in the computational basis and outputs the state in the input wires; otherwise it runs C' again and again, until success is attained, and only then outputs. A small calculation shows that the thus-obtained action on the N qubits is not unitary (although C' itself unitary) but is defined by the linear operator C'_{00} in the fashion: $|\psi\rangle \to \frac{C'_{00}|\psi\rangle}{|C'_{00}|\psi\rangle|}$. The success probability matters: the post-selection process works well if the probability of success, which is $|C'_{00}|\psi\rangle|$, is not very small.

In equation (6.2) we assume that R_j $(0 \leq j \leq d)$ is an arbitrary element of $SU(2)$, the group of two dimensional unitary matrices with unit determinant. Later we will restrict R_j to be X-rotations, which will still allow us to build most, if not all, of the useful signal processing circuits. Figure 6.1 illustrates the structure of a typical quantum signal processing circuit.

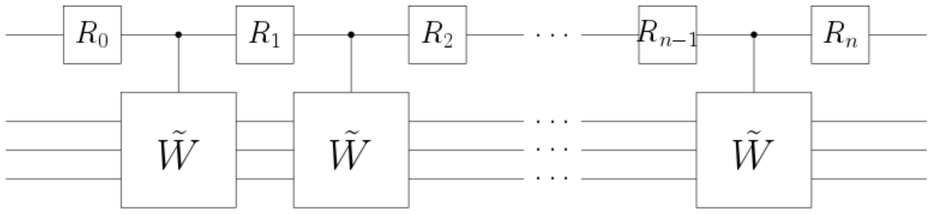


Figure 6.1: An illustration of a quantum signal processing circuit. State preparation and post selection operators are omitted for simplicity.

Laurent Polynomials. Because the blocks of $\widetilde{W}$ are W, W^{-1}, or zero, and because each rotation R'_j only linearly combines the four blocks of the operator it receives, the sequence of operations in equation (6.2) without the final M represents an operator whose all four blocks are Laurent polynomials [24] of W, i.e. polynomials with both positive and negative powers. The final M just picks the top left block of this operator which is itself an operator on the non-ancilla wires and has the form $F(W) = \sum_{-n}^{n} c_i W^i$ for some n and c_i. It is easy to see that $n \leq d$. From what we have said about post-selection, C acts on H according to $|\psi\rangle \to \frac{F(W)|\psi\rangle}{|F(W)|\psi\rangle|}$. The effect of C on an arbitrary operator W is said to be described by $F(w) = \sum_{-n}^{n} c_i w^i$ if $F(W)$ is the operator to which C transforms W.

In practice (Hamiltonian simulation, etc.), the Laurent polynomial F is given to us, and our goal is to design the sequence $R_0, R_1, \dots, R_d$, which produces it. A small but very useful lemma will help us achieve this goal:

Lemma 6.1. *If C has the effect $\sum_{-n}^{n} c_i W^i$ on all* one dimensional *unitary operators $W \in U(1)$, then it has the same Laurent polynomial as effect on* all *unitary operators.*

For the proof of the lemma let us investigate how C' acts on a state $\underbrace{|\phi\rangle}_{\text{ancilla}} \otimes \underbrace{|\psi\rangle}_{H}$, where $|\psi\rangle$ is an eigenvector of W with eigenvalue $e^{i\theta}$. The R'_j operations change only the first register and not the second. The operator $\widetilde{W}$ gives a controlled phase shift of the second register, which by the "kick-back" effect can be represented as a $\begin{pmatrix} e^{i\theta} & 0 \\ 0 & e^{-i\theta} \end{pmatrix}$ operation on the first register and no change on the second register. Therefore the subspace $\mathbb{C}^2 \otimes |\psi\rangle$ remains invariant under C' and furthermore, restricted to this subspace, C' acts as if we plugged in for $\widetilde{W}$ the controlled (i.e. "tilde") version of the one dimensional unitary $(e^{i\theta})$. This is already sufficient to argue that W is transformed in the same way as the one dimensional unitaries, since all its eigenvectors uniformly do so.

We can in fact explicitly give $F(w) = \sum_{-n}^{n} c_i w^i$. Let w denote $e^{i\theta}$ and $\widetilde{w}$ denote $e^{i\theta\sigma_z} = \mathrm{diag}(w, w^{-1})$. Then, when we plug in the one dimensional w into C', we get:

$$C'(w) = \begin{pmatrix} C'_{00}(w) & C'_{01}(w) \\ C'_{10}(w) & C'_{11}(w) \end{pmatrix} = R_d \cdot \widetilde{w} \cdot R_{d-1} \cdots R_1 \cdot \widetilde{w} \cdot R_0. \tag{6.4}$$

Here $C'_{00}(w), \dots, C'_{11}(w)$ are Laurent polynomials of w. Finally, with M we get $F(w) = C'_{00}(w)$.

An algebra emerges. The design goal is now clear: Find 1-qubit unitaries $R_0, \dots, R_n$ and Laurent polynomials $C'_{01}(w)$, $C'_{10}(w)$, $C'_{11}(w)$ such that equation (6.4) holds with $F(w) = C'_{00}(w)$. Later we will see that it is sufficient to restrict ourselves to $d \leq n$. Further, we also have the restriction that C' is unitary for all unit complex numbers w.

Let M(2, $\mathbb{C}[w, w^{-1}]$) denote the ring of 2 by 2 matrices over the Laurent polynomials with complex coefficients. We may view this ring as an algebra over the real or over the complex numbers. It turns out that the most natural view of the algebras that arise in our investigation is to treat them as algebras *over the real numbers*. This algebra certainly contains all operators that, *when we treat w as a single unknown*, we may ever encounter in the expression (6.4) as a partial product.

A way to attain our goal is then to do the following two steps:

1. **Completion**: Find $C'_{01}(w)$, $C'_{10}(w)$, $C'_{11}(w)$ such that these together with $C'_{00}(w) = F(w)$ form the four components of $C' \in \mathrm{M}(2, \mathbb{C}[w, w^{-1}])$ that is unitary for every value of w.

2. **Decomposition**: Find rotations $R_0, R_1, \cdots, R_d$ such that the product

$$R_d \cdot \widetilde{w} \cdot R_{d-1} \cdots R_1 \cdot \widetilde{w} \cdot R_0 \tag{6.5}$$

gives C'. That indeed $d = n$ can be taken we will prove later.

These two steps are the way to go in Haah's paper [24]. We will propose a new algorithm for the second step based on a new theorem as well as a new method for preprocessing F, and obtain excellent experimental results. The fact that a rich, well-structured subset of elements of M(2, $\mathbb{C}[w, w^{-1}]$) can be split as a product of degree-one elements was first observed in [17]. We strengthen this result with a uniqueness theorem that allows us to do the decomposition in a binary tree manner rather than sequentially. Our experiments show a substantial increase in numerical stability when we do the decomposition this way.

6.3 Algebras associated with quantum signal processing

In the previous section we have seen how quantum signal processing translates to a task of 1. **Completion** and then 2. **Decomposition** in M(2, $\mathbb{C}[w, w^{-1}]$). In this section we define two useful sub-algebras of M(2, $\mathbb{C}[w, w^{-1}]$), where both 1. and 2. are already possible and the number of free parameters is reduced. We would like to remind the reader that we view M(2, $\mathbb{C}[w, w^{-1}]$) as an infinite-dimensional algebra *over the real numbers*.

The Low algebra. To define the smaller of the two sub-algebras of M(2, $\mathbb{C}[w, w^{-1}]$) (and for practical purposes the more interesting one), we write down a general element of the algebra in terms of the variable

$$\widetilde{w} = \begin{pmatrix} w & 0 \\ 0 & w^{-1} \end{pmatrix}. \tag{6.6}$$

The motivation comes from the paper of Low et al. [17] that implicitly worked in this algebra. Let I, X, Y, Z be the Pauli matrices as usual. Then already expressions of the form

$$A(\widetilde{w}) + B(\widetilde{w}) \cdot iX \qquad A, B \in \mathbb{R}[x, x^{-1}] \tag{6.7}$$

contain all operators that *when we treat w as a single unknown* we may ever encounter as a partial product when $R_0, \dots, R_d$ are all X-rotations:

$$R_X(\alpha) = \cos\alpha \cdot I + \sin\alpha \cdot iX. \tag{6.8}$$

Low et al. has shown that this is all we need when our target F has real coefficients and satisfies a parity constraint. Furthermore, if we change the post-selection by introducing a more general final measurement, this framework encapsulates a very rich set of F's, so we need not go further.

The Haah algebra. If we nevertheless want to obtain all possible Laurent polynomials in the upper left corner including those with complex coefficients, there is a bigger sub-algebra [24] of M(2, $\mathbb{C}[w, w^{-1}]$) that accomplishes that. The elements of this sub-algebra are written as

$$A(\widetilde{w}) + B(\widetilde{w}) \cdot iX + C(\widetilde{w}) \cdot iY + D(\widetilde{w}) \cdot iZ \qquad A, B, C, D \in \mathbb{R}[x, x^{-1}]. \tag{6.9}$$

6.4 The syntactic versus semantic view

Let us make here some clarifying remarks about our expressions. When writing down polynomials, one way of viewing them (Laurent or otherwise) is *semantic*. In this interpretation polynomials are functions from a set (often a field $\mathbb{F}$, vector space, algebraic variety) to a ring, module, etc. Another way of viewing polynomials is *syntactic*. In this case two polynomials are different if the sequence of their coefficients differ (formal polynomials).

The two views can lead to different definitions. Consider for instance x^3 in GF(2). By Fermat's little theorem (or simply by checking both replacements in GF(2)) we conclude that semantically x^3 is the same as x. Syntactically however they are obviously different.

In our case the domain over which we should view expressions (6.7) and (6.9) is the unit circle $U(1) = \{w \mid |w| = 1\}$. This is because when we instantiate w, it is a one dimensional unitary operator, that is a phase. Now, $\widetilde{w}$ is an element of M(2, $\mathbb{C}[w, w^{-1}]$), giving matrix values to expressions (6.7) and (6.9) with Laurent polynomials as elements. We may then ask whether it can occur that two expressions in (6.9) do not equal syntactically, but they equal semantically. It is easy to show however, using the fundamental theorem of algebra, that this may not happen. This permits us to switch back and forth between the two interpretations as suits us.

6.5 Star operation, unitary and Hermitian elements, degree

To define the star operation in M(2, $\mathbb{C}[w, w^{-1}]$) we take the semantic view: the star, M^* of an element $M \in$ M(2, $\mathbb{C}[w, w^{-1}]$) will have the property that if we instantiate M and M^* over any given $w \in U(1)$, then we get two matrices that are conjugate transposes of each other. Thus e.g. $\widetilde{w}^* = \widetilde{w}^{-1}$, because w and w^{-1} are conjugates of each other for any $w \in U(1)$. On constant matrices (i.e. without the variable w) the star operator works as usual:

$$(iX)^* = -iX \quad , \quad (iY)^* = -iY \quad , \quad (iZ)^* = -iZ. \tag{6.10}$$

In general, if $a, b, c, d \in \mathbb{C}[w, w^{-1}]$ and we obtain a^* from a by conjugating the coefficients and swapping w and w^{-1} (similarly for b, c, d), then we can define the star operation and unitarity for an element in $\mathrm{M}(2, \mathbb{C}[w, w^{-1}])$ as

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix}^* = \begin{pmatrix} a^* & c^* \\ b^* & d^* \end{pmatrix} \; ; \qquad M \text{ is unitary } \leftrightarrow \; MM^* = I. \tag{6.11}$$

This definition leads to:

Lemma 6.2. *The Haah and Low algebras are closed under the star operation of* M(2, $\mathbb{C}[w, w^{-1}]$).

Cayley-Dickson Algebras. The Haah and Low algebras are Cayley-Dickson algebras, although we do not directly use this fact.

Hermitian elements. Let us now determine the Hermitian and anti Hermitian elements of the Haah algebra. Notice that the following elements of the Haah algebra are Hermitian:

$$\widetilde{w}^j + \widetilde{w}^{-j}, (\widetilde{w}^j - \widetilde{w}^{-j}) \cdot iZ, j \in \mathbb{N} \tag{6.12}$$

and therefore all their linear combinations.

Lemma 6.3. *The set of the Hermitian elements of the Haah algebra consists exactly of linear combinations of* $\widetilde{w}^j + \widetilde{w}^{-j}$, $(\widetilde{w}^j - \widetilde{w}^{-j}) \cdot iZ$ $(n \in \mathbb{N})$.

Proof. First notice that the following elements and therefore all their linear combinations are anti-Hermitian, i.e. $M^* = -M$:

$$\widetilde{w}^j - \widetilde{w}^{-j}, \widetilde{w}^{\pm j} \cdot iX, \widetilde{w}^{\pm j} \cdot iY, (\widetilde{w}^j + \widetilde{w}^{-j}) \cdot iZ, j \in \mathbb{N}. \tag{6.13}$$

Next notice that the elements in expressions (6.12) and (6.13) together span the Haah algebra. This finishes the proof, since if a Hermitian element M is written as $M_1 + M_2$, where M_1 is Hermitian and M_2 is anti-Hermitian, then $M_2 = M - M_1$ must be both Hermitian and anti-Hermitian, so M_2 must be zero. $\square$

We remark that the set of all Hermitian elements of the Haah algebra form an algebra, which is also the center of the Haah algebra (since both iX and iZ are in the Haah algebra, the center can only contain elements that are proportional to I).

Degree. Each element $M(w)$ of $\mathrm{M}(2, \mathbb{C}[w, w^{-1}])$ has a *degree*, which is the maximum absolute value of any exponent of w that ever occurs in M.

Lemma 6.4. *In equation (6.9) the maximum degree among A, B, C and D as formal Laurent polynomials coincides with the above notion of degree.*

When we multiply regular polynomials the degrees add up. The same does not hold for Laurent polynomials, especially ones with matrix coefficients. The next lemma, crucial for our main lemma, gives a case in the Haah algebra when we can be certain that the degrees do add up.

Lemma 6.5. *Let M be a Hermitian element of the Haah algebra and M' be an arbitrary element of* $\mathrm{M}(2, \mathbb{C}[w, w^{-1}])$. *Then* $\deg MM' = \deg M + \deg M'$.

Proof. Let M be of degree n. By Lemma 6.3 we can assume that $M = \sum_{j=0}^{n} \lambda_j(\widetilde{w}^j + \widetilde{w}^{-j}) + \mu_j(\widetilde{w}^j - \widetilde{w}^{-j}) \cdot iZ$ where $\lambda_j, \mu_j \in \mathbb{R}$ and $\lambda_n \pm \mu_n i \neq 0$. Let

$$M' = \begin{pmatrix} A(w) & B(w) \\ C(w) & D(w) \end{pmatrix} \quad \in \quad \mathrm{M}(2,\mathbb{C}[w, w^{-1}]) \tag{6.14}$$

be of degree n'. We show that the degree of MM' is $n + n'$. Clearly, it is sufficient to show that the degree of $M'' = \lambda_n(\widetilde{w}^n + \widetilde{w}^{-n})M' + \mu_n(\widetilde{w}^n - \widetilde{w}^{-n}) \cdot iZM'$ is $n + n'$. We can write M'' as

$$\begin{pmatrix} ((\lambda_n + \mu_n i)w^n + (\lambda_n - \mu_n i)w^{-n})A(w) & ((\lambda_n + \mu_n i)w^n + (\lambda_n - \mu_n i)w^{-n})B(w) \\ ((\lambda_n - \mu_n i)w^n + (\lambda_n + \mu_n i)w^{-n})C(w) & ((\lambda_n - \mu_n i)w^n + (\lambda_n + \mu_n i)w^{-n})D(w) \end{pmatrix} \tag{6.15}$$

and it is straightforward to see that the highest degree term cannot cancel whether the exponent is positive or negative. $\square$

Parity Subgroups. We will in particular be interested in elements of $\mathrm{M}(2, \mathbb{C}[w, w^{-1}])$ with parity, that is, they are in the parity subgroup:

Definition 6.1 (Parity subgroup of real Laurent polynomials). *The* parity subgroup $P = P_0 \cup P_1 \subset \mathbb{R}[w, w^{-1}]$ *is defined to be the subgroup of Laurent polynomials with parity constraint, that is, an element of this group either has even parity:*

$$P_0 = \left\{ p(w) = \sum_{k=-n}^{n} p_k w^k \mid p_k = 0 \, \textit{for all odd } k \right\} \tag{6.16}$$

or odd:

$$P_1 = \left\{ p(w) = \sum_{k=-n}^{n} p_k w^k \mid p_k = 0 \, \textit{for all even } k \right\}. \tag{6.17}$$

Note that P is closed under multiplication but not addition. Starting from P, we can define the parity subgroup of the Haah algebra as the set of elements of the form

$$A(\widetilde{w}) + B(\widetilde{w}) \cdot iX + C(\widetilde{w}) \cdot iY + D(\widetilde{w}) \cdot iZ, \qquad A, B, C, D \in P_\epsilon, \quad \epsilon \in \{0, 1\} \tag{6.18}$$

and similarly for the Low algebra. Note that both the element $\widetilde{w}$ and the rotations $R_i \in SU(2)$ satisfy parity constraints, thus all elements we can hope to get form the composition of $\widetilde{w}$ and one-qubit rotations must be parity elements.

Remark Note that not all unitary elements satisfy the parity constraint. The element $(2I + (w + w^{-1})iX + (w - w^{-1})Y)/\sqrt{8}$ would be a counterexample.

Low et al. [17] proved an interesting unique decomposition theorem about its unitary elements, which is then generalized by Haah [24] as follows:

Theorem 6.1 ([17, 23, 24]). *For every unitary parity element U in the Haah albegra with degree d, there exists a unique decomposition of U into degree-0 and degree-1 terms:*

$$U(w) = R_d \cdot \widetilde{w} \cdot R_{d-1} \cdots R_1 \cdot \widetilde{w} \cdot R_0 \tag{6.19}$$

Moreover, when U lies in the Low algebra, $R_0, R_1, \ldots, R_d$ are all X-rotations.

6.6 The main lemma

Assume we are given a unitary element, U, of the Haah algebra to be decomposed into a product of linear terms. Instead of solving for one linear term at a time as in [24], we want to decompose U as U_1U_2 where U_1 and U_2 have degrees roughly half of that of U. For any element of the Haah algebra it is proven that such decomposition exists, but is it unique? Conceivably, it could occur that $U = V_1V_2$, where V_1 and V_2 are not unitary. And if we find V_1 and V_2 instead of U_1 and U_2, we cannot continue with the decomposition. A consequence of our Main Lemma is that this is impossible.

Lemma 6.6 (Main Lemma)**.** *Let M be in the Haah algebra, M' in $\mathrm{M}(2, \mathbb{C}[w, w^{-1}])$. Then*

$$\deg MM' \geq \deg M' - \deg M + \deg M^*M. \tag{6.20}$$

Proof. Since $\deg M = \deg M^*$ we have that $\deg M^*MM' \leq \deg M + \deg MM'$. On the other hand M^*M is a Hermitian element of the Haah algebra, so by Lemma 6.5 we have that $\deg M^*MM' = \deg M^*M + \deg M'$, which completes the proof. □

We can now prove:

Corollary 6.1. *Let U be a degree-d unitary parity element of the Haah algebra. Then the set of equations*

$$\begin{aligned} \deg(V^*U) &\leq d - l, \\ \deg(V) &\leq l, \\ V(w = 1) &= I, \\ V &\in P_{l \bmod 2} \end{aligned} \tag{6.21}$$

has a unique solution V which is a unitary element of the Haah algebra. Moreover, V lies in the Low algebra if U does.

Proof. We only prove the first part of the theorem. The second is an easy consequence. By Theorem 6.1, U can be decomposed as

$$U = R_d \cdot \widetilde{w} \cdot R_{d-1} \cdot \cdots \cdot R_1 \cdot \widetilde{w} \cdot R_0, \tag{6.22}$$

where $R_0, \ldots, R_d$ are all unique. Take

$$V = R_d \cdot \widetilde{w} \cdot R_{d-1} \cdot \cdots \cdot R_{d-l+1} \cdot \widetilde{w} \cdot \prod_{i=d-l+1}^{d} R_i^{-1}. \tag{6.23}$$

It is easy to see that V satisfies the set of equations.

To prove uniqueness, suppose that there exists another element V' which satisfies the equations. By Lemma 6.6,

$$d - l \geq \deg V'^* U \geq \deg U - \deg V' + \deg V'V'^* \geq d - l + \deg V'V'^*. \tag{6.24}$$

Therefore, $\deg V'V'^* \leq 0$. We leave this as an inequality since the degree of the zero Laurent polynomial is negative infinite. However, since $V'(1) = I$, $V'^*(1) = I$ and so $V'V'^*$ is not zero. Thus, the inequality is actually an equality and so V' must be proportional to a unitary. As $V'(1) = I$, V' itself is a parity unitary and therefore must be equal to V by Theorem 6.1. □

Note that the conditions on the degrees in Corollary 6.1 are inequalities, but the solution we obtain in the end will satisfy them with equality. This is because the degree of a product can be at most the sum of the degrees.

6.7 Algorithm

We now provide an outline of our angle finding algorithm for quantum signal processing. Let $F(w) \in \mathbb{R}[w, w^{-1}]$ be the function that we wish our circuit to have the effect of. It will be natural for us to define the norm of a Laurent polynomial as

$$\|F\| \equiv \max_{w \in U(1)} |F(w)|. \tag{6.25}$$

Now, it is clear that a necessary condition for a circuit to have the effect of F is that $\|F\| < 1$ so that the completion can be done. This is also actually sufficient, as shown in [17, 23, 24] and as we will see in the following.

Capitalization In many applications of quantum signal processing, $F(w)$ contains terms with very small coefficients, especially for the higher order terms. This can arise for instance when we are trying to achieve the effect of some analytic function and we use a Taylor series approximation. In [24], this is cited as the primary source of numerical instability. We propose an ad hoc solution by a procedure we call *capitalization*. Namely, given that we wish to perform quantum signal processing to some error tolerance ε, we can add leading order terms to $F(w)$ with coefficients on the order of ε and then run our algorithm. Combined with our new algorithm, we show in our experiments that this does extremely well empirically, allowing us to solve instances orders of magnitude larger than what was previously possible, for example in [107].

A similar preprocessing technique was used in [24], where all the coefficients of the given Laurent polynomial are rounded to multiples of ϵ/n, where n is the degree. This would be an alternative way to resolve the numerical instability.

Completion via root finding The first step is completion, namely finding a unitary U in the Low algebra such that the upper left corner of U is $F(w)$. To do this, we simply solve for the real Laurent polynomial G such that $F(\widetilde{w}) + G(\widetilde{w}) \cdot iX$ is a unitary element of the Low algebra. It is not difficult to see that this translates to the equation

$$F(w)F(w^{-1}) + G(w)G(w^{-1}) = 1. \tag{6.26}$$

As done in [17, 23, 24], this equation can be solved using a root finding method. Namely, we solve for the roots of the expression

$$1 - F(w)F(w^{-1}) \tag{6.27}$$

over *all* complex numbers. Then, all real roots will come in pairs $\{r_i, 1/r_i\}_i$ while all non-real complex roots will come in quadruples $\{z_j, 1/z_j, \bar{z}_j, 1/\bar{z}_j\}_j$. Hence,

$$1 - F(w)F(w^{-1}) \propto \prod_i (w - r_i)(1/w - r_i) \prod_j (w - z_j)(1/w - z_j)(w - \bar{z}_j)(1/w - \bar{z}_j). \tag{6.28}$$

Evaluating this at $w = 1$, the constant of proportionality has to be real and positive by the constraint on F. Denoting this by α, we conclude

$$G(w) = \sqrt{\alpha} \prod_i (w - r_i) \prod_j (w - z_j)(w - \bar{z}_j) \tag{6.29}$$

is a possible solution.[1] Note that by construction $G(w)$ is a real Laurent polynomial.

Decomposition via halving We now give an algorithm for the decomposition step in quantum signal processing motivated by Corollary 6.1. Let U be a degree d parity unitary in the Low algebra. We wish to find unitaries V_1, V_2 of degree $l, d - l$ respectively such that $U = V_1 V_2$. This could be done by expressing V_1^* as a collection of real variables

[1]This is not unique since, for instance, we could have chosen the same with $w \mapsto 1/w$.

$\{x_n, x'_n\}_{n=-l}^{l}$ using the structure of Low algebra elements:

$$V_1^* = \sum_{n=-l}^{l} X_n w^n \tag{6.30}$$

where X_n is of the form

$$X_n = \begin{pmatrix} x_n & ix'_n \\ ix'_{-n} & x_{-n} \end{pmatrix}. \tag{6.31}$$

Note that the parity condition will eliminate half of the variables. The unitary U can be put into a similar form:

$$U = \sum_{n=-d}^{d} A_n w^n \tag{6.32}$$

so that

$$V_1^* U = \sum_{n=-d-l}^{d+l} \left(\sum_{\substack{|n_1| \le l, |n_2| \le d \\ n_1+n_2=n}} X_{n_1} A_{n_2} \right) w^n. \tag{6.33}$$

Then, we can enforce the condition that $V_1^* U$ is of degree at most $d-l$ by setting for $n < -d+l$ and $n > d-l$:

$$\sum_{\substack{|n_1| \le l, |n_2| \le d \\ n_1+n_2=n}} X_{n_1} A_{n_2} = 0, \tag{6.34}$$

where 0 here is the zero matrix. This gives a system of linear equations for the real variables. We also add the additional linear equations from $V(1) = I$:

$$\mathrm{Tr}\left[\sigma_\mu \sum_{n=-l}^{d_2} X_n \right] = (1, 0, 0, 0), \tag{6.35}$$

where σ_μ is the vector of Pauli matrices preceded by I. By Corollary 6.1 this system of linear equations has a unique solution and returns a degree l parity unitary V_1^* such that $V_1^* U$ is of degree $d-l$.

However, in real implementations the element we are to decompose might not be exactly unitary. This may be due to numerical imprecision or errors introduced in the completion step. We therefore propose to solve the linear system by least squares. This gives a recursive algorithm:

```
procedure DECOMPOSE($M$)
    if $\deg M = 1$ then
        return $\{M\}$
    else
        solve least squares problem:
            $\deg M_1^* M \leq \deg M - \lceil \frac{\deg M}{2} \rceil$,
            $\deg M_1 \leq \lceil \frac{\deg M}{2} \rceil$, $M_1(1) = I$,
            $M_1 \in P_{\lceil \frac{\deg M}{2} \rceil \mod 2}$
        return DECOMPOSE($M_1$)$\cup$ DECOMPOSE($M_1^* M$)
    end if
end procedure
```

Figure 6.2: The algorithm DECOMPOSE.

6.8 Experimental results

We perform numerical experiments implementing our algorithms and compare them to existing methods. The algorithm given in [24] also used root finding for completion but implemented decomposition differently. Namely, given a degree d unitary U in the Low algebra, they sequentially solve for the single qubit rotations $L_i = R_i$ in equation (6.19) from $i = d$ to 1. More specifically, they start with U, and solve for L_d such that

$$\deg(\widetilde{w}^{-1} \cdot L_d^* \cdot U) \leq n - 1. \tag{6.36}$$

The same is then repeated for $L_d^* U$ to obtain L_{d-1} and so on. We shall refer to this method as *carving*. Due to the iterative nature of this method, the error introduced at the beginning would blow up during the carving process. Therefore, it requires very high precision arithmetic in the completion step. Our halving algorithm, on the other hand, can be performed with standard 64-bit machine precision throughout.

In the following, we report the results of experiments in which we tested our algorithm using the Python `numpy` package on a MacBook Pro with 2.9 GHz Intel Core i5 processor and 8 GB memory.

Hamiltonian simulation. Hamiltonian simulation is one of the most important applications of quantum signal processing [107, 24, 23]. Formally, the problem is the following. Given a Hermitian H, with $\|H\| \leq 1$, and time-interval for the evolution τ, we wish to implement the unitary $e^{-iH\tau}$. When quantum walk is used [112, 113], the unitary W can be implemented with a quantum circuit, whose eigenvalues $\mp e^{\pm i\theta_\lambda}$ are associated with those λ from the Hamiltonian as

$$\sin \theta_\lambda = \lambda/2. \tag{6.37}$$

The task is then to achieve the effect of the function $F : e^{i\theta} \to e^{2i\tau \sin\theta}$. More specifically,

$$F(w) = \exp\left(\tau \frac{w^2 - w^{-2}}{2}\right) = \sum_{k\in\mathbb{Z}} J_k(\tau) w^{2k}, \tag{6.38}$$

where J_k are the Bessel functions of the first kind. In Equation (6.38), the function $F(w)$ is already in the form of Laurent series whose coefficients decrease exponentially. One can approximate $F(w)$ up to an additive error ϵ for $w \in U(1)$ by truncating it into $n = \Omega(\tau + \log(1/\epsilon))$ terms [24],

$$\hat{F}_\epsilon(w) = \sum_{k=-n/2}^{n/2} J_k(\tau) w^{2k} \tag{6.39}$$

where, in experiments, n can be chosen as $2\lceil \frac{e}{2}\tau + \ln(1/\epsilon)\rceil$ [24]. Empirical results also suggest that one needs to scale down $\hat{F}_\epsilon(w)$ by a factor $\eta \in (0, 1)$, in order to have more numerical stability during the root-finding completion. However, the downscaling would decrease the success probability in the post-selection by a factor of η. This can be regarded as a tradeoff between the classical preprocessing phase versus the actual time complexity of the algorithm.

Specifically, the inputs to the algorithm are the time-interval τ, error tolerance ϵ and scaling factor η. We then further truncate the list of coefficients of $\eta \cdot \hat{F}_\epsilon(w)$ to standard double precision. The resulting Laurent polynomial would be the input of the angle-finding algorithm.

Running time scaling with respect to τ. We performed angle decomposition for different τ, with fixed error tolerance $\epsilon = 0.001$ and $\eta = 0.7$. For the experiments, the Laurent polynomial is truncated to guarantee an error upper bound of $\epsilon/10$, with capitalizing parameter, i.e. the coefficients of the appended highest- and lowest-degree terms set to $\epsilon/3$. The detailed numerical results are illustrated in Fig. 6.3.

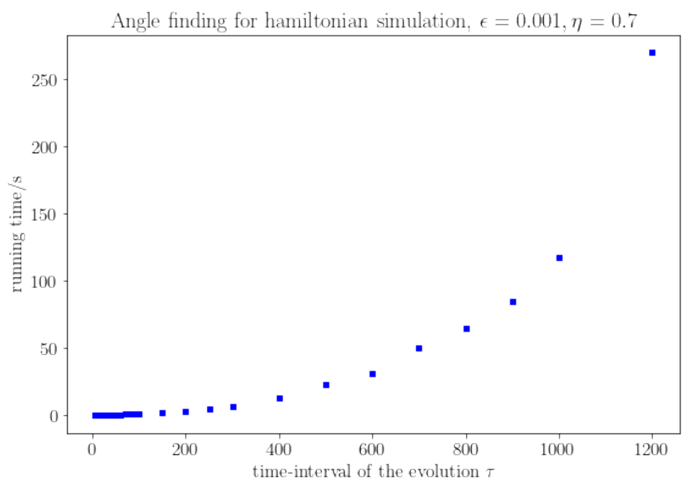


Figure 6.3: The running time for angle finding using the halving method. Here the error tolerance and the scaling factor is fixed to 10^{-3} and 0.7 respectively. The running time scales as a cubic function with respect to the degree of the Laurent polynomial, hence also cubic with respect to the evolution time parameter τ. Note that an instance with $\tau = 1200$ can be efficiently solved within 5 minutes.

Comparison to the carving method. To compare the performance of the halving and the carving method, we compare the capability of carving and halving with respect to the time-interval of the evolution and the error tolerance. The problem of angle finding becomes more difficult with larger τ and smaller ϵ. We call a parameter pair (τ, ϵ) *achievable* with respect to an angle-finding method, if that angle finding method can output a Laurent polynomial ϵ-close to the Hamiltonian simulation function $e^{\tau \frac{w^2 - w^{-2}}{2}}$ within machine precision. Again, the Laurent polynomial is truncated to guarantee an error upper bound $\epsilon/10$, with capitalizing parameter, set to $\epsilon/3$.

The numerical results for the achievable region for the parameters are illustrated in Fig. 6.4. It can be seen that the halving method has a far bigger achievable region than that of the carving method.

Random-coefficient experiment. We further demonstrate the performance of the halving algorithm via a family of random distributions over the set of Laurent polynomials with a given degree n, which we call the random-coefficient distribution. First, $n+1$ real numbers $F_{-n}, F_{-n+2}, \ldots, F_n$ are sampled i.i.d. uniformly at random from $[-1, 1]$. Then, the polynomial $F(w) = \sum_{k=-n, n-k \text{ even}}^{n} F_k w^k$ is rescaled to infinity norm η for a fixed scaling factor $\eta = 0.5$.

Note that the random-coefficient instances do not experience exponential decay of the leading coefficient, thus there is no need for initial capitalization. Without a small leading

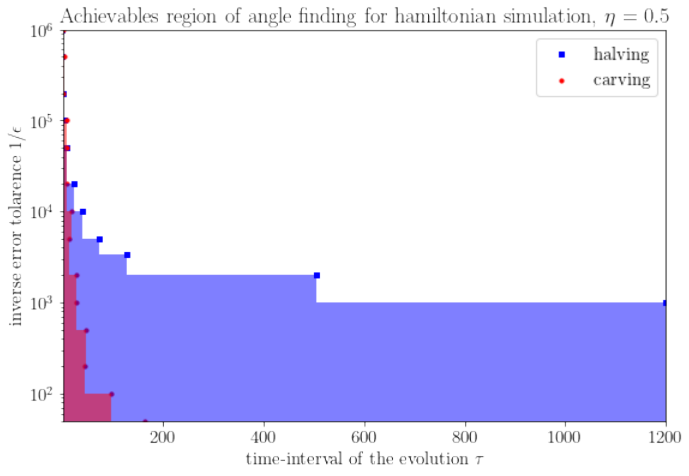


Figure 6.4: The achievable parameter regions for the Hamiltonian simulation problem with machine precision, with the carving and the halving method. Note that the y-axis is log scaled.

coefficient, Fig. 6.5 effectively shows the dependency of the final error with respect to the degree of the random instance. The carving method performed well up to degree 100, and then the averaged error quickly blows up, whereas the averaged error for the halving method stayed under 10^{-6} throughout the experiments, up to the highest degree 5000 for which an experiment was carried out.

Empirical error analysis. In [24], an error analysis is carried out to show that the precision p_c needed for carving scales as $poly(n, \log(1/\epsilon))$ in order to solve for angle sequences for a Laurent polynomial with degree n and error tolerance ϵ.

Also, when the leading coefficients are decaying exponentially, the carving algorithm would suffer from numerical instability, although this issue can be easily resolved by setting each coefficient to be a multiple of ϵ/n. The same issue occurred to the halving algorithm as well but this was resolved by capitalization.

Denoting the magnitude of the largest leading coefficients δ, it is plausible that

$$p_c = poly(n, \log(1/\epsilon), \log(1/\delta)).$$

Our experimental results gives evidence that the precision p_h needed for halving scales as

$$p_h = poly(\log n, \log(1/\epsilon), \log(1/\delta)),$$

since it does not suffer from the error blow-up from the iterative carving. If so, then the halving method would be a truly numerically stable algorithm. Further theoretical work

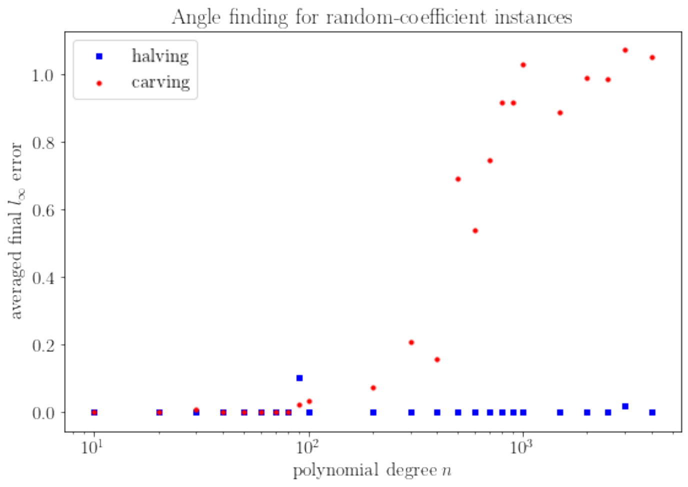


Figure 6.5: Comparison of the final l_∞ error of the angle finding algorithm, over random-coefficient instances of different degrees. For each degree, 10 random-coefficient instances are given as input to the angle-finding algorithm, and the final l_∞ errors are averaged to generate the plot. Note that the x-axis is log-scaled for clarity. Regarding error rate above 0.05 as failure, the carving method would fail when the degree approaches 100, whereas the halving method still behaves numerically stable up to degree 5000. The anomalous blue point at degree $n = 90$ is possibly due to a hard instance from the distribution.

needs to be done to demonstrate that it is indeed the case.

6.9 Discussion

An alternative algorithm for completion. Recall that in completion we are given a Laurent polynomial $A(w) \in \mathbb{R}[w, w^{-1}]$ of degree d and we need to find $B(w) \in \mathbb{R}[w, w^{-1}]$ such that

$$A(\widetilde{w}) + B(\widetilde{w}) \cdot iX \tag{6.40}$$

is unitary. This can be solved using an iterative method with some initial guess of a Laurent polynomial B' which we choose to be also of degree d. Then, we solve for a degree d perturbation δB such that $A(\widetilde{w}) + (B'(\widetilde{w}) + \delta B(\widetilde{w})) \cdot iX$ is unitary. Taking a first order approximation, we get the following Laurent polynomial equation:

$$A(w^{-1})A(w) + B'(w^{-1})\delta B(w) + \delta B(w^{-1})B'(w) + B'(w^{-1})B'(w) = 1. \tag{6.41}$$

Again, we can solve equation (6.41) by converting it into a linear system where the variables are the coefficients of δB. Since we took a first order approximation, this system might not have a solution. Therefore we again take a least squares approach to obtain the following algorithm:

```
procedure COMPLETE($A, \epsilon$)
    generate random real Laurent polynomial $B'$
    set $M = A(\widetilde{w}) + B'(\widetilde{w}) \cdot iX$
    while $\|M^*M - I\| > \varepsilon$ do
        solve least squares problem: $M + \delta B(\widetilde{w}) \cdot iX$ is unitary
        set $M = M + \delta B(\widetilde{w}) \cdot iX$
    end while
end procedure
```

Figure 6.6: Algorithm COMPLETE (real Laurent polynomial A, error tolerance parameter ε)

We compare this to the root-finding approach used by [17] and [24] and our heuristic of iterative least squares. Starting from a random perturbation B', experiments showed that this procedure converges well on most input instances. By setting a unitarity threshold ϵ, it is observed that the iterative least square method terminates earlier than the root finding approach. See Fig. 6.7 for details.

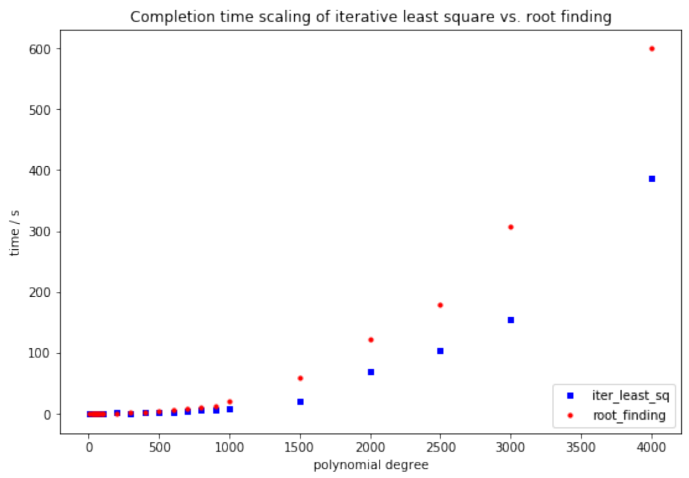


Figure 6.7: Time scaling of the completion phase, using the root-finding and the iterative least square method respectively. For the iterative least square method, the termination condition is set such that the unitarity, namely $\|UU^* - I\|$ goes below 10^{-8}. The blue dots represent experiment results for the iterative least square method, whereas the red dots are for experiments with root finding. It can be observed that iterative least square method has a constant factor advantage over root finding in terms of the running time.

More efficient angle finding algorithm. Both the algorithm in [24] and the algorithm in this chapter run in time $\tilde{O}(n^3)$, n being the degree of the instance of consideration. It would be desirable if a classical algorithm with substantial acceleration compared to these two algorithms can be invented.

The bottleneck of a faster algorithm lies in the root-finding algorithm during the completion phase. Given a Laurent polynomial $F \in \mathbb{R}[w, w^{-1}]$ of degree d, the goal is to find another Laurent polynomial of the same degree $G \in \mathbb{R}[w, w^{-1}]$ such that $F(\tilde{w}) + G(\tilde{w}) \cdot iX$ is unitary throughout. The current algorithm solves G by computing the roots for $G(w)G(w^{-1}) = 1 - F(w)F^{\dagger}(w^{-1})$ and assigning them appropriately to G. It would be desirable if either the roots can be implied from specialized inputs for the problem, or solving for the roots can be circumvented at all with the design of a novel algorithm. We leave this to future work.

Theoretical guarantee for numerical stability. The halving algorithm proves numerical stability in practice, in particular in the problem of finding angle sequences for Hamiltonian simulation. However, it is yet to be shown that the halving algorithm is guaranteed to be stable numerically.

The bottleneck of the problem lies in an upper bound on the condition number of the system of linear equations Eqn. (6.21). With an upper bound polynomial with respect to the degree n and the largest leading coefficient δ, we would be able to show that an initial error would propagate only with a factor polynomial to n and δ in one step of halving, and consequently in logarithmic levels of the recursive halving algorithm. The algorithm would then be guaranteed to be numerically stable. We leave this to future work.

Derandomization. In the halving algorithm, randomness is introduced in the completion phase, where one pair of the complex roots out of a quadruple, or one real root out of a pair needs to be picked. Our experiment shows that different roots selected would result in different numerical stability, and potentially influence the performance of the angle finding algorithm. Further study needs to be done for the completion part in order to resolve this issue.

CHAPTER 7

Transversal switching between stabilizer codes

In this chapter, we study the problem of switching between stabilizer codes while preserving the error-correcting property throughout. More specifically, we propose a randomized variant of the stabilizer rewiring algorithm (SRA), a method for constructing a transversal circuit mapping between any pair of stabilizer codes. As gates along this circuit are applied, the initial code is deformed through a series of intermediate codes before reaching the final code. With this randomized variant, we show that there always exists a path of deformations which preserves the code distance throughout the circuit, while using at most linear overhead in the distance. Furthermore, we show that a random path will almost always suffice, and discuss prospects for implementing general fault-tolerant code switching circuits.

7.1 Introduction

It is an oft-cited fact that no quantum error-correcting code can implement a universal transversal logical gate set [25, 114, 115]. As a result, there have been several attempts to circumvent this no-go theorem to achieve universal fault-tolerant quantum computation. These candidates include magic state distillation [116, 117], gauge fixing [118, 119], and more recently pieceable fault-tolerance [120, 121]. These last two candidates can be seen as a special case of the more general approach of code switching [27, 26, 28, 29, 30].

Code switching is a natural idea: given two codes, map information encoded in one code to information encoded in the other. For this mapping to be fault-tolerant, we must often perform several intermediate error-correction steps to ensure that faults do not grow out of hand. Thus, it is essential that during a circuit switching between codes, the extremal error-correcting codes are deformed through a series of intermediate error-correcting codes from one to another. This notion of intermediate error-correction was used in [26] to implement universal transversal computation by switching between the Steane and Reed-Muller codes, whose complementary transversal gate sets are universal when taken together. However, universal fault-tolerant computation is not the only consideration in choosing error-correcting codes, and different codes can be tailored to different tasks. For this reason, it would be nice to have a way of converting between different quantum codes fault-tolerantly.

Simply decoding and re-encoding information is undesirable, since the bare information becomes completely unprotected during this transformation. Past work has succeeded in constructing fault-tolerant circuits for switching between particular quantum error-correcting codes fault-tolerantly, while providing guarantees that these circuits are optimal within some framework [27].

Recently, [31] considered switching between generic stabilizer codes, and proposed the stabilizer rewiring algorithm (SRA) for constructing a transversal circuit mapping between *any* pair of stabilizer codes. The circuit complexity scales quadratically with the code length, and depends on a choice of presentation for the code generators. Different presentations will result in different circuits mapping between different sets of at most n intermediate codes. This circuit necessarily fails to be fault-tolerant when these intermediate codes have low distance. This leads to the central question: *is there an efficient way of fault-tolerantly converting between generic stabilizer codes?*

7.1.1 Results

Towards this goal, we propose a randomized variant of the SRA, the randomized SRA (rSRA). We show that for any pair of stabilizer codes, with at most linear overhead with respect to the distance of the codes, there always exists a transversal circuit that maps between intermediate codes of high distance. Furthermore, using slightly more overhead, such a path can be found with high probability. In particular, we show the following.

Theorem 7.1 (Theorem 7.2, Informal)**.** *For any two $[[n, k, d]]$ stabilizer codes S_1 and S_2, the rSRA scheme gives a transversal circuit mapping from S_1 to S_2 where each intermediate code has distance at least d with probability $1 - \varepsilon$, using*

$$m = O\left(d \log \frac{n}{d} + \log \frac{1}{\varepsilon}\right) \tag{7.1}$$

ancilla qubits.

This *distance-preserving* property is a necessary, but not sufficient condition to ensure a fault-tolerant mapping. So while the algorithm does *not* necessarily yield a fault-tolerant conversion, it gives a universal upper bound on the number of ancilla qubits required for distance-preserving transversal code transformation. As was noted in [31], the usefulness of this scheme is in its generality. While the upper bound may be of independent conceptual interest, we hope that with modification, the rSRA can be applied as a useful schema for searching for fault-tolerant paths between small codes.

7.2 Preliminaries

7.2.1 Classical codes

We start the preliminaries with classical error correcting codes. Throughout this section, all additions and multiplications are done in the binary field $\mathbb{Z}_2$.

An $[n, k]$ *code* C is a subset of $T := \mathbb{Z}_2^n$, where $k := \log |C|$. Such a code is called an $[n, k, d]$ code, where

$$d := \min_{u,v \in C, u \neq v} \|u - v\|_1. \tag{7.2}$$

Here, we use $\| \cdot \|_1$ to denote the Hamming distance. An $[n, k, d]$ code is known to correct $\lfloor \frac{d-1}{2} \rfloor$ bits of error; that is to say, given a perturbed code u' and the promise that $u' = u + \delta u$ for some $u \in C$ and $\|\delta u\|_1 \leq \lfloor \frac{d-1}{2} \rfloor$, such a u must be unique. In the presence of only bit flip errors with weight at most $\lfloor \frac{d-1}{2} \rfloor$, a codeword can be transmitted losslessly since such error can always be corrected.

A code is most generally represented as a subset of $\mathbb{Z}_2^n$ of size 2^k, thus might not be manageable due to the exponential size. A restricted subset of error correcting codes, called *linear codes*, are defined to be k-dimensional linear subspaces of $\mathbb{Z}_2^n$ rather than arbitrary subsets of size 2^k.

One of the concise representation for linear codes is the *check matrix*. For any $[n, k, d]$ linear code C, there exists $S \in \mathbb{Z}_2^{(n-k) \times n}$ such that $C = \ker(S)$, and

$$d = \min_{v \in \ker(S)} \|v\|_1. \tag{7.3}$$

Moreover, any element $c' \in T$ not in the code C would generate a nonzero *syndrome* $Sc' \in \mathbb{Z}_2^{n-k}$ that only depends on the perturbation of c' from a codeword. Once a low-weight error e is found such that $Sc' = Se$, it is guaranteed that $c' \oplus e$ is a codeword; for decoding a $[n, k, d]$ linear code, one can just compute the syndrome and finding the lowest weight error e that generates the given syndrome.

7.2.2 Quantum codes

Extending the idea of classical codes, an $[[n, k]]$ quantum code C is simply a 2^k dimensional subspace in the Hilbert space $(\mathbb{C}^2)^{\otimes n}$. Such a code is said to have distance d, or equivalently being an $[[n, k, d]]$-code, if for every set B of less than d qubits, there exists a recovery channel fully restoring the encoded states by only acting on the rest $n - d + 1$ qubits:

$$\forall B : |B| < d, \exists \mathcal{R}_B, \forall |\phi\rangle \in C, \mathcal{R}_B(\mathrm{Tr}_B[|\phi\rangle\langle\phi|]) = |\phi\rangle\langle\phi|. \tag{7.4}$$

One of the biggest differences between quantum and classical error correcting codes is the presence of superposition, and thus the presence of the phase flip error Z in addition to the bit flip error X. Classical codes would necessarily fail to be quantum error-correcting codes for the following reason. Consider two distinct classical codewords x and y and assume that they differ on the first bit without loss of generality. The state $\frac{1}{\sqrt{2}}(|x\rangle - |y\rangle)$ can either be regarded as an undisturbed encoded state, or another codeword state $\frac{1}{\sqrt{2}}(|x\rangle - |y\rangle)$ with a Z error on the first qubit. With the two scenarios identical to any recovery channel, it is not possible that the single qubit Z error can be perfectly corrected.

The quantum counterpart of linear codes is called *stabilizer codes*. Recall from Section 2.4 the n-qubit Pauli group $\mathcal{P}^n$. Then a *stabilizer group* $S \subseteq \mathcal{P}^n$ is an abelian subgroup of the Pauli group not containing $-I$. To any such stabilizer group S, we can associate a subspace $C_S \subseteq (\mathbb{C}^2)^{\otimes n}$ defined as the simultaneous $+1$-eigenspace of all the operators in S. We call such a subspace C_S a *stabilizer code*.

For a stabilizer code C_S with parameters $[[n, k, d]]$, there exists a more concise expression of the code distance d in terms of the stabilizer group S. The normalizer $\mathcal{N}_{\mathcal{P}^n}(S)$ represents the set of logical Pauli operators for C_S, and so

$$d := \min_{L \in \mathcal{N}(S) \setminus S} (|L|) \tag{7.5}$$

where $|\cdot|$ denotes the weight of the Pauli operator. Note that the smallest number of stabilizers generating the corresponding stabilizer subgroup is $n - k$.

Given any stabilizer group S, if we choose a generating set G_S for S, we can define a syndrome map

$$Syn_G : \mathcal{P}^n \longrightarrow \{0, 1\}^{n-k} \tag{7.6}$$

$$Syn_G(e)_i = \begin{cases} 0 \text{ if } [e, g_i] = 0 \\ 1 \text{ if } \{e, g_i\} = 0 \end{cases} \tag{7.7}$$

for $G = (g_1, \dots g_{n-k})$. Then equivalently,

$$d = \min_{L \in \ker(Syn_G) \setminus S} (|L|) \tag{7.8}$$

and is independent of the choice of G.

Another convenient formalism for describing stabilizer groups is as subspaces of symplectic vector spaces over the binary field $\mathbb{F}_2$, and we will use the two formulations interchangeably. For any $P \in \mathcal{P}^n$, if

$$P \propto X^{a_1} Z^{b_1} \otimes X^{a_2} Z^{b_2} \dots \otimes X^{a_n} Z^{b_n} \tag{7.9}$$

then we can associate to P the vector $\vec{P} := (\vec{a}|\vec{b})^T \in \mathbb{F}_2^{2n}$. We can equip $\mathbb{F}_2^{2n}$ with a symplectic bilinear form

$$\langle \vec{v}, \vec{w} \rangle := \vec{v}^T B \vec{w} \tag{7.10}$$

where B is the $2n \times 2n$ block matrix defined by

$$B = \begin{pmatrix} 0 & I \\ I & 0 \end{pmatrix}. \tag{7.11}$$

Then Paulis P, Q commute if any only if their associated vectors $\vec{P}, \vec{Q}$ are orthogonal in this vector space. Thus, we can equivalently define a stabilizer group as a self-orthogonal subspace of this vector space. A *generator matrix* G is then a choice of basis for this subspace, so that for C an $[[n, k]]$ code, G will be a rank $(n-k)$ matrix of shape $2n \times (n-k)$. The syndrome map can then be similarly defined as

$$Syn_G(\vec{P}) = G^T B \vec{P}. \tag{7.12}$$

Further note that for any $A \in GL(\mathbb{F}_2, n-k)$, for any generator matrix G for S, GA^T is also a generator matrix for S. The syndrome map satisfies

$$Syn_{GA^T}(\vec{P}) = (GA^T)^T B \vec{P} = AG^T B \vec{P} = A \cdot Syn_G(\vec{P}). \tag{7.13}$$

So any action on the generator matrix induces a corresponding action on the syndrome vectors themselves.

7.2.3 Transversality

A quantum error-correcting code alone can be used to store quantum information in a way that is resistent to external noise. However, the error-correcting property does not suffice in order to perform quantum computation on the encoded logical information.

Fault tolerance. We call a circuit C on a class of encoded inputs *t-fault-tolerant* if it is t-fault-tolerant in the exRec formalism [122]. Formally, given error correction procedure EC, C is t-fault-tolerant if for any choice of t faulty components in the combined circuit $EC \cdot C \cdot EC$, a faultless version of EC applied to the output of the combined circuit can successfully recover the data. If $t \geq 1$ we may simply call the circuit fault-tolerant.

Transversal gates. Fault-tolerance takes into account both errors on the physical qubits and errors on the quantum operations that act on the physical qubits. One easiest way to achieve fault-tolerance is through transversal gates. Suppose that a logical gate U_L acts on l logical systems $A_1, \dots, A_l$, corresponding to l physical systems $B_1, \dots, B_l$. We label each qubit system B_{ij} as the j-th physical qubit in the i-th physical state. The gate U_L is called transversal, if it is decomposable as follows:

$$U_L = U_1 \otimes U_2 \otimes \cdots \otimes U_n, \tag{7.14}$$

where each U_j acts on the physical qubits $B_{ij}, 1 \leq i \leq l$.

One can observe that a transversal logical gate is fault-tolerant. If one initial qubit or one single component in the logical gate is faulty, the error would only corrupt at most one physical qubit in each logical qubit. The data can be fully restored by performing error correction procedure after appying the transversal logical operation.

It would be desirable to find an error-correcting code which admits a universal transversal gate set; however, this has been proven to be impossible, a result known as the Eastin-Knill theorem [25].

Code switching. Among the attempts people have been trying to circumvent this no-go result, a general approach called code-switching has been proposed. The central idea of code switching is the following. Although no single code admits a universal transversal logical gate set, there are pairs of codes whose transversal logical gate sets are universal when combined together. One example is the [[7,1,3]] Steane code with transversal Clifford gates, and the [[15, 1, 3]] Reed-Muller code with transversal T-gates. To perform universal fault-tolerant quantum computation, one just needs to switch back and forth between such pairs of codes. Of course, switching between the two codes cannot be done via only transversal gates according to the Eastin-Knill Theorem.

Transversal measurement. One way of fault-tolerantly switching between a pair of codes is through transversal measurement. For measurements with a binary outcome, we sometimes represent the POVM $\{M_0, M_1\}$ by the difference $P = M_0 - M_1$. Recall that $M_0 + M_1 = I$ so M_0 and M_1 can be fully recovered given P. A transversal measurement P is one that can be decomposed into tensor products on each physical qubit:

$$P = P_1 \otimes P_2 \otimes \cdots \otimes P_n. \tag{7.15}$$

A circuit for performing transversal measurements fault-tolerantly is first proposed by [123], and an illustration is given in Appendix D. Note that although such measurement is performed individually on each physical qubit, the classical outcomes of the measurements are then gathered together to produce further instructions for the quantum circuit. This introduces non-local information propogation that a transversal gate cannot achieve.

Finally, we call a circuit transversal if every operation in the circuit can be implemented through transversal gates and transversal measurements. A code switching scheme is called transversal if it only involves transversal gates and transversal measurements.

7.3 The rSRA schematic

The rSRA modifies the SRA presented in [31], whose central insight is the following. Consider two stabilizer groups S, S' with generating sets G, G' satisfying the following nice property:

$$G = \{g, g_1, \ldots, g_l\} \tag{7.16}$$

$$G' = \{g', g_1, \ldots, g_l\} \tag{7.17}$$

where $\{g, g'\} = 0$. We call two such codes for which one can choose such generating sets *adjacent*. Then one can readily check that the Clifford gate $\frac{1}{\sqrt{2}}(1 + g'g)$ maps information encoded in the stabilizer code defined by G to the same information encoded in the stabilizer code defined by G'. Letting $|\psi\rangle_G$ denote a logical state in the code associated to G, we see that

$$\forall i, g_i \cdot \frac{1}{\sqrt{2}}(1+g'g)|\psi\rangle_G = \frac{1}{\sqrt{2}}(1+g'g)g_i|\psi\rangle_G \tag{7.18}$$

$$= \frac{1}{\sqrt{2}}(1+g'g)|\psi\rangle_G; \tag{7.19}$$

$$g' \cdot \frac{1}{\sqrt{2}}(1+g'g)|\psi\rangle_G = \frac{1}{\sqrt{2}}(g'+g'g'g)|\psi\rangle_G \tag{7.20}$$

$$= \frac{1}{\sqrt{2}}(g'+g)|\psi\rangle_G \tag{7.21}$$

$$= \frac{1}{\sqrt{2}}(g'+g)g|\psi\rangle_G \tag{7.22}$$

$$= \frac{1}{\sqrt{2}}(1+g'g)|\psi\rangle_G. \tag{7.23}$$

Eq. 7.18 holds since $[g_i, g] = 0$ and $[g_i, g'] = 0$ for all i. Eq. 7.19 holds since all the g_i's stabilize $|\psi\rangle_G$. Eq. 7.21 and Eq. 7.23 hold since g' and g both square to the identity as Pauli operators. Finally, Eq. 7.22 holds since g stabilizes $|\phi\rangle_G$.

The insight is that this mapping can be done transversally. While the Clifford transformation described need not be transversal, it can be simulated by a transversal Pauli measurement supplemented by a transversal Pauli gate controlled on classical information. This is similar to gauge-fixing, in which one measures a logical operator of the gauge and then applies a corresponding logical gauge operator conditioned on the outcome. To see this, consider the circuit described by:

1. Measure g'.
2. Apply g conditioned on measurement outcome -1.

Let $P^{\pm}$ denote the projector onto the $+1/-1$ eigenspace of g'. Then, if the measurement outcome is $+1$,

$$\frac{1}{\sqrt{2}}(1+g'g)|\psi\rangle_G = \frac{1}{\sqrt{2}}(1+g')|\psi\rangle_G = \sqrt{2}P^{+}|\psi\rangle_G. \tag{7.24}$$

Furthermore,
If the measurement outcome is -1,

$$\frac{1}{\sqrt{2}}(1+g'g)|\psi\rangle_G = \frac{1}{\sqrt{2}}(g-gg')|\psi\rangle_G \tag{7.25}$$

$$= \frac{1}{\sqrt{2}}g(1-g')|\psi\rangle_G \tag{7.26}$$

$$= \sqrt{2}gP^-|\psi\rangle_G. \tag{7.27}$$

Thus, we see that we can *transversally* perform the mapping $|\psi\rangle_G \to |\psi\rangle_{G'}$.

Now consider the more general case in which we have (non-adjacent) S, S' describing $[[n,k]]$ and $[[n',k]]$ codes respectively. We now describe a general randomized algorithm for outputting a circuit switching between these two codes, similar to [31], and will later show that this circuit is distance-preserving with high probability. The inputs are arbitrary generator matrices G, G' for stabilizer groups S, S', along with a choice of ancilla size $m \in \mathbb{N}$.

7.3.1 Preparing the generator matrices

1. Append $|0\rangle$ ancilla to the smaller code so that the codes are of equal size. We now assume that both codes are $[[n,k]]$ codes.

2. Append $|0\rangle^{\otimes m}$ to the first code, and $|+\rangle^{\otimes m}$ to the second. Note that this is equivalent to defining a pair of new stabilizer codes

$$\hat{S} = \langle S \otimes I^{\otimes m}, I^{\otimes n} \otimes Z \otimes I^{\otimes m-1}, \ldots, I^{\otimes n+m-1} \otimes Z\rangle, \tag{7.28}$$

$$\hat{S}' = \langle S' \otimes I^{\otimes m}, I^{\otimes n} \otimes X \otimes I^{\otimes m-1}, \ldots, I^{\otimes n+m-1} \otimes X\rangle. \tag{7.29}$$

3. Choose $G_A = G'_A$ to be a basis for the subspace defined by $\hat{S} \cap \hat{S}'$.

4. Choose G_B to extend the basis of G_A to a basis for $\mathcal{N}(\hat{S}') \cap \hat{S}$ and choose G'_B to extend the basis of G_A to a basis for $\mathcal{N}(\hat{S}) \cap \hat{S}'$.

5. Choose G_C to extend the basis $G_A \cup G_B$ to a basis for $\hat{S}$ and G'_C to extend the basis $G'_A \cup G'_B$ to a basis for $\hat{S}'$.

6. Let H be the *commutativity matrix* for G_C, G'_C defined by $H := {G'_C}^T B G_C$. By Lemma C.2, H is invertible with dimension $|G_C| \times |G_C|$, where $|G_C| \geq m$. So we

can choose $M, N \in GL(\mathbb{F}_2, |G_C|) : M^T H N = I_{|G_C|}$ and redefine

$$G_C \leftarrow G_C \cdot M \tag{7.30}$$
$$G'_C \leftarrow G'_C \cdot N. \tag{7.31}$$

7. Choose uniformly at random $V, V' \in_r \mathbb{F}_2^{|G_C| \times |G_B|}$ and a $U \in_r GL(\mathbb{F}_2, |G_C|)$.

8. Redefine

$$G_C^T \leftarrow U(V G_B^T + G_C^T) \tag{7.32}$$
$${G'_C}^T \leftarrow (U^{-1})^T (V' {G'_B}^T + {G'_C}^T) \tag{7.33}$$
$$\tag{7.34}$$

Note that this does not change the commutativity matrix since

$$U(V G_B^T + G_C^T) B (G'_C + G'_B {V'}^T) U^{-1} = I_{|G_C|}. \tag{7.35}$$

9. Let $G_B = \{g_1, \ldots, g_{|G_B|}\}$ and $G'_B = \{g'_1, \ldots, g_{|G'_B|}\}$. For each $g_i \in G_B$, choose $\overline{g_i}$ satisfying

$$[\overline{g_i}, G_A] = 0 \tag{7.36}$$
$$[\overline{g_i}, G_C] = 0 \tag{7.37}$$
$$[\overline{g_i}, G'_C] = 0 \tag{7.38}$$
$$[\overline{g_i}, \{g_{i+1}, \ldots, g_{|G_B|}\}] = 0 \tag{7.39}$$
$$[\overline{g_i}, \{g'_{i+1}, \ldots, g'_{|G_B|}\}] = 0 \tag{7.40}$$
$$[\overline{g_i}, \{\overline{g_1}, \ldots, \overline{g_{i-1}}\}] = 0 \tag{7.41}$$
$$\{\overline{g_i}, g_i\} = 0 \tag{7.42}$$
$$\{\overline{g_i}, g'_i\} = 0. \tag{7.43}$$

To see that such a choice of $\overline{g_i}$ always exists, note that it must satisfy at most $2n$ affine linear equations, all of which are linearly independent, in a space of dimension $2n$.

Now that we have prepared the generator matrices, we will step-by-step map between adjacent codes transversally.

7.3.2 Applying the transformation

10. For $1 \leq i \leq |G_B|$ indexing the elements of G_B, perform the transformation $g_i \mapsto \overline{g_i}$. Note that the resulting stabilizer codes are adjacent, and so the preceding discussion gives a transversal circuit for each mapping.

11. For $1 \leq i \leq |G_C|$ indexing the elements of G_C, perform the transformation $g_i \mapsto g'_i$. Again, since the codes are adjacent, the mapping can be done transversally.

12. For $1 \leq i \leq |G_B|$ indexing the elements of G'_B, perform the transformation $\overline{g_i} \mapsto g'_i$ starting from $i = |G_B|$ and working backwards towards $i = 1$. Again, we have a transversal circuit for each mapping.

13. Discard the ancilla.

This randomized variant differs from the original SRA in several ways. First, there is the introduction of ancilla, which we will see are vital for preserving the distance. Next, the SRA fixes the generating sets G, G' subject to the same G_A and G_C conditions, but with different G_B conditions. Namely, the SRA fixes the $\overline{g}$ to be the product of the complementary logical operators to those operators in G_B and G'_B, which can be seen as nontrivial logical operators on the opposite code. This allows for a certain degree of freedom in choosing the order in which one converts between the two codes, but restricts the G_C,G'_C that are available to use. Also in the SRA, only the set of valid permutations among G_B and G_C are considered, which restricts the search for a distance-preserving mapping. In the rSRA, we consider the full set of invertible transformations on G_C for a better chance of success. Finally, the transformation described above is *symmetric* in the sense that switching from G to G' or G' to G after step 9 results in the same set of intermediate codes. We will see that this simplifies the set of errors we must consider.

7.4 Distance bounds

We now show that, with low overhead and high probability, the described rSRA will yield a distance-preserving circuit. More specifically, we show that the intermediate codes preserve the distance of the extremal codes.

Theorem 7.2. *Let S,S' be any two stabilizer codes with parameters $[[n_1, k, d_1]]$ and $[[n_2, k, d_2]]$, respectively. Let*

$$d := \min\{d_1, d_2\}, n := \max\{n_1, n_2\} \tag{7.44}$$

. Then, the rSRA will output a distance-preserving circuit mapping information encoded in S to information encoded in S' with probability $1-\epsilon$ using

$$m = O(d \log \frac{n}{d} + \log \frac{1}{\varepsilon}) \tag{7.45}$$

ancilla qubits.

Proof. Consider a particular error $e : |e| < d$. There are four different types of errors to consider.

(1) $\underline{e \in S \cap S'}$: In this case, $e \in Span(G_A)$, and so remains passively corrected throughout the transformation.

(2) $\underline{e \in S \setminus \mathcal{N}(S')}$: In this case, we can decompose $e = g_A + g_B + g_C$ where $g_A \in Span(G_A), g_B \in Span(G_B)$, and $g_C \in Span(G_C)$. Furthermore, $g_C \neq 0$, or else e would be a logical operator of weight $< d$ for S'. Thus, e must be detected by G'_C, and so it remains detectable after step 11. In particular, before the end of step 11, e must fall out of the intermediate stabilizer group. Suppose this occurs for the first time when transforming between two adjacent codes whose stabilizer groups differ by g, g'. Then we can write $e = g + \sum_i a_i g_i$, and as g' commutes with all other g_i, it must be that $\{e, g'\} = 0$. Since g' remains in each intermediate code up through step 11, e must be detectable throughout.

(3) $\underline{e \in S' \setminus \mathcal{N}(S)}$: This error is just an error of type (2) when performing the opposite transformation from S' to S. By symmetry of the scheme, the set of intermediate codes during this opposite transformation is the same, and so these errors remain detectable by the preceding argument.

(4) $\underline{e \notin \mathcal{N}(S) \cup \mathcal{N}(S')}$: Let $G_C^{(0)}, G_C'^{(0)}$ be the bases G_C and G'_C we choose after step 6 in the rSRA scheme, and let $G_C^{(1)}, G_C'^{(1)}$ be the bases we choose after step 8. Note that the syndrome map for $G_C^{(1)}$ can then be expressed as

$$Syn_{G_C^{(1)}}(e) = U(V \cdot Syn_{G_B}(e) + Syn_{G_C^{(0)}}(e)). \tag{7.46}$$

In this case it must be that

$$(Syn_{G_A}(e)|Syn_{G_B}(e)|Syn_{G_C^{(0)}}(e))^T \neq 0, \tag{7.47}$$

$$(Syn_{G_A}(e)|Syn_{G'_B}(e)|Syn_{G'^{(0)}_C}(e))^T \neq 0. \tag{7.48}$$

Note that if $Syn_{G_A}(e) \neq 0$, then e is always detectable since each intermediate code includes the check operators from G_A. Thus, we only need to consider the case where $Syn_{G_A}(e) = 0$, and so we can assume that

$$(Syn_{G_B}(e)|Syn_{G_C^{(0)}}(e))^T \neq 0 \tag{7.49}$$

and

$$(Syn_{G'_B}(e)|Syn_{G'^{(0)}_C}(e))^T \neq 0. \tag{7.50}$$

Let P_e denote the probability that the error e is undetectable in some intermediate code over the random choices of U, V, and V'. We divide P_e into three parts. Let A_e denote the event that $Syn_{G_C^{(1)}}(e) = 0$, B_e the event that $Syn_{G'^{(1)}_C}(e) = 0$, and let C_e denote the event that both $Syn_{G_C^{(1)}}(e)$ and $Syn_{G_C{'}^{(1)}}(e)$ are nonzero, yet e becomes undetectable on some intermediate code during the transformation. Then $P_e \leq \Pr[A_e] + \Pr[B_e] + \Pr[C_e]$. We bound $\Pr[A_e]$, $\Pr[B_e]$, and $\Pr[C_e]$ separately. To bound $\Pr[A_e]$, note that

$$Syn_{G_C^{(1)}}(e) = U(V \cdot Syn_{G_B}(e) + Syn_{G_C^{(0)}}(e)). \tag{7.51}$$

Since $U \in GL(\mathbb{F}_2, n-k)$, A_e occurs if and only if $V \cdot Syn_{G_B}(e) + Syn_{G_C^{(0)}}(e) = 0$. If $Syn_{G_B}(e) = 0$, it must be the case that $Syn_{G_C^{(0)}}(e) \neq 0$, and so $Syn_{G_C^{(1)}}(e) \neq 0$; otherwise $Syn_{G_B}(e) \neq 0$ and $V \cdot Syn_{G_B}(e) + Syn_{G_C^{(0)}}(e)$ is uniformly random over $\{0,1\}^{|G_C|}$. In either case, we have

$$\Pr[A_e] \leq 2^{-|G_C|}. \tag{7.52}$$

Repeating the same argument shows that $\Pr[B_e] \leq 2^{-|G_C|}$ as well. To bound $\Pr[C_e]$, define

$$v = V \cdot Syn_{G_B}(e) + Syn_{G_C^{(0)}}(e), \tag{7.53}$$

$$w = V' \cdot Syn_{G'_B}(e) + Syn_{G'^{(0)}_C}(e). \tag{7.54}$$

Since $Uv, (U^{-1})^T w \neq 0$, e will be detectable during steps 10 and 12, and so C_e occurs only if e becomes undetectable during step 11. Specifically, it must be that $Syn_{G_A}(e) = 0, Syn_{\overline{G}_B}(e) = 0$, and the last 1 in the vector Uv occurs before the first 1 in the vector $(U^{-1})^T w$. This is because we are sequentially replacing the check operators of G with the check operators of G', and so an error becomes undetectable for some intermediate code only if we produce some zero syndrome during this sequence of substitutions. By Lemma C.1, for two nonzero vectors $v, w \in \{0,1\}^{|G_C|}$, the probability that the last 1 in Uv comes before the first 1 in $(U^{-1})^T w$ is bounded by $(|G_C| - 1) \cdot 2^{-|G_C|}$.

Summing these three terms, we have $P_e \leq (|G_C| + 1) \cdot 2^{-|G_C|}$. Taking a union bound, the probability P that any of the intermediate codes fail to detect any error of weight less than d is upper bounded by

$$P \leq \sum_{e:|e|<d} P_e \leq |\{e : |e| < d\}| \cdot (|G_C| + 1) \cdot 2^{-|G_C|}. \tag{7.55}$$

Taking a Chernoff bound, we get that this is in turn upper bounded as

$$P \leq 4^{n+m} \cdot e^{-D(\frac{d-1}{n+m}||\frac{3}{4})(n+m)} \cdot (|G_C| + 1) \cdot 2^{-|G_C|} \tag{7.56}$$

where $D(\cdot||\cdot)$ is the KL-divergence. By the quantum singleton bound, we can assume $\frac{d-1}{n+m} < \frac{d-1}{n} < \frac{3}{4}$. Furthermore, by Lemma C.2, $|G_C|$ is given by rank$(G^T B G')$, which is at least m. So the probability of failure can be further upper bounded by

$$P \leq 4^{n+m} \cdot e^{-D(\frac{d-1}{n+m}||\frac{3}{4})(n+m)} \cdot (m + 1) \cdot 2^{-m}. \tag{7.57}$$

It suffices to choose m such that the above quantity is upper bounded by ϵ in order to achieve a high probability of success. In particular, the case $\varepsilon = 1$ upper bounds the minimum number of ancilla qubits required for a fault-tolerant transformation. By Lemma C.3 we observe that taking

$$m = O(d \log \frac{n}{d} + \log \frac{1}{\varepsilon}) \tag{7.58}$$

is sufficient for the rSRA scheme to succeed with probability $1 - \epsilon$.

□

7.5 Discussion

Theorem 7.2 shows that with high probability, the rSRA will produce a transversal circuit with intermediate codes that have distances *at least* the minimum of the distances of the extremal codes. It is important to note that this does *not* necessarily imply fault-tolerance. The reason is because, when measuring g', the randomness in the outcome prevents us from using that syndrome bit during error-correction. More specifically, consider the following two scenarios.

1. We project onto the $(+1)$-eigenspace of g'.
2. We project onto the (-1)-eigenspace of g' and simultaneously experience an error that anticommutes with only g'.

Then we cannot distinguish these two scenarios using only our syndrome bits, and so cannot correct the resulting error. More generally, we can cast the property required for fault-tolerance in terms of subsystem codes. For every conversion between adjacent codes, we consider the subsystem code with a single gauge degree of freedom corresponding to gauge operators g' and g. Then the resulting conversion will be t-fault-tolerant precisely when the resulting subsystem code has distance $2t + 1$. This is because the redundant syndrome information can diagnose errors without the syndrome bit associated to g', and so ensure that we project onto the correct eigenspace. For this reason, additional techniques may be required to achieve fault-tolerance using the rSRA, such as error-detection on the ancilla. We leave this to future work.

These techniques contrast with recent results from [120], where it was shown that pieceable fault-tolerance offers generic *fault-tolerant* code switching between stabilizer codes subject to certain constraints. However, their techniques require that the codes are nondegenerate and have some set of native fault-tolerant Clifford gates, allowing a fault-tolerant SWAP gate *between* different codes. One could also consider preparing a second code state and using logical teleportation to achieve a fault-tolerant mapping [28].

Practically, on small examples, one finds that often *no* ancilla qubits are required to find a distance-preserving circuit, which is desirable as the resulting circuit may then be fault-tolerant. In general, this can be attributed to a coarse accounting of $|G_C|$ in terms of the number m of ancilla qubits. In most cases, $N(S) \cap N(S')$ will be small, and so the ancilla will be superfluous.

Moreover, the multi-qubit gate complexity of the algorithm is $\sum_{P \in \{\overline{g_i}\} \cup G_B \cup G_C} |P|$, so that choosing a low weight generating set is ideal for reducing the complexity of the code switching circuit. For this reason, LDPC codes might provide more efficient code switching circuits, although preserving the distance may depend on choosing a high weight set of generators.

This algorithm derives its usefulness from its generality. For specific code switching examples, it may be profitable to modify the circuit using the rSRA as a template, augmented with a larger class of fault-tolerant manipulations such as local Clifford gates, in order to search for a *fault-tolerant* mapping. For large code sizes, the use of high-weight Shor-style measurements is limiting as it requires large verified CAT states. Thus, this technique may be most useful as a step in a concatenated scheme, or simply as a search ansatz.

One subtlety about the rSRA is that, while it outputs a distance-preserving circuit switching between two codes with high probability, this is difficult to check. This follows from the difficulty of computing the minimum distance of a generic error-correcting code, which is a co-NP-hard problem in general [124]. Indeed, even when restricting to a particular distance, this check remains extremely costly. This poses a barrier to derandomizing the algorithm, which would be one desirable avenue for future improvement.

Another such improvement would be to minimize overhead. One could imagine taking a random local clifford transformation in order to increase the size of G_C, rather than introducing ancilla. Such a strategy would be interesting since locally equivalent codes have nearly identical properties. Of course, modifying the algorithm to ensure fault-tolerance is the most important improvement.

If it is true that one can always choose locally equivalent representatives for which the rSRA provides a distance-preserving conversion without ancilla, this would suggest that all error-protected information in stabilizer codes is, in some sense, "transversally equivalent". This contrasts with the diverse set of equivalence classes of locally unitarily equivalent codes, which can be identified as distinct submanifolds of Grassmanians. Indeed, it may be of conceptual interest to interpret these upper-bounds in a broader framework of fault-tolerance, such as the one investigated in [125].

Similarly, the generality of the rSRA provides an aesthetically nice interpretation of error-protected information. It suggests that, with the addition of some minimal overhead, any stabilizer error-protected encoding of information is indeed "transversally equivalent" to any other.

CHAPTER 8

Summary and conclusions

In this thesis, we have studied several problems in the interdisciplinary of classical and quantum information. We investigated situations where quantum information proves intrinsically different from their classical counterpart, and designed classical algorithms that facilitate quantum computing by providing more robust and accurate implementations. In particular, we have contributed the following.

8.1 Resillience of quantum hashing against classical leakage

8.1.1 Summary

In Chapter 3, we studied the comparison between two properties of a classical-quantum correlation, namely the amount of initial classical-classical correlation required to generate it (i.e. the conversion parameter), and the maximum amount of classical-classical correlation it can possibly yield post-measurement (i.e. the guessing probability). Although the two quantities are obviously the same for classical-classical correlations, we proved an arbitrary separation between the two quantities for classical-quantum correlations.

As an application, we looked at quantum cryptographic hash functions. Such functions are examples where the hash itself hardly contains any information of the classical secret message, yet generating it correctly requires full information of the message. It was then proved that quantum cryptographic hashing is maximally resilient to classical leakage. The result also indicated that inherently new approaches must be found in order to prove the existence of quantum-proof extractors with comparable parameters as their classical counterparts.

8.1.2 Future work

Existence of quantum-proof extractors. The result in [33] originated from the author's investigation into non-constructive proof of the existence of quantum-proof extractors. One long-standing open problem is in what range of parameters do (non-constructive) quantum-proof extractors exist. Probabilistic methods yield existence of randomness extractors with optimal parameters, and the existence of classical-proof extractors follow naturally; however, the argument fails for quantum-proof extractors since it is no longer possible to define conditional distributions conditioned on a pre-measurement quantum state.

One reason that the existence of quantum-proof extractors is hard to prove is the lack of results to classify classical-quantum states. For classical joint distributions, Lemma 3.4 indicates that in order to prove that a certain function is a classical-proof extractor, it suffices to evaluate its performance over the set of conditionally uniform distributions. Such a set is a polyhedron of finite dimension, thus in principle easy to deal with. Classical-quantum states, on the other hand, does not yet have such a reduction into the finite-dimensional case. Future work is needed to further investigate the structure of the set of classical-quantum states with a certain conditional min-entropy (defined as $CQ(k)$ in 3.5).

8.2 Limitations of classical strong simulations

8.2.1 Summary

In Chapter 4 and 5, we studied the limitation of classical strong simulations of quantum computation.

One natural representation of a general quantum circuit is a tensor network, and almost all predominant strong simulators achieve strong simulation by tensor network contraction, with various techniques for resource saving. In particular, all those techniques fall into a two-step paradigm: first, a monotone arithmetic circuit with respect to certain entries of the tensor network is constructed, and second, the values of the entries are put into the monotone arithmetic circuit to yield the result amplitude value. We call strong simulations in this paradigm *monotone*. While techniques can be applied to simplify the monotone arithmetic circuits in the first step, the monotone arithmetic circuit is inherently large on size if the function to be computed is sufficiently complicated. Applying hardness results of computing the permanent using monotone arithmetic circuit, we proved an unconditional and explicit lower bound of the complexity of monotone strong simulation.

For more general strong classical simulations, unconditional lower bounds are no longer available without a breakthrough in complexity theory (i.e. answering the question whether

$\#P = P$), and we go to conditional lower bounds, with a goal of proving lower bounds as explicit as possible using the most widely-believed computational assumptions. We prove that strong simulation of general n-qubit quantum circuit with polynomial size takes at least exponential time in terms of n based on the Exponential Time Hypothesis (ETH); the exponent can be made explicit based on the Strong Exponential Time Hypothesis (SETH). We further explore the hardness of strong simulation in terms of the number of T gates in a Clifford+T circuit, and proved a strong exponential lower bound based on ETH. Lower bounds with explicit exponents were also computed based on the state-of-the art SAT and 3-SAT solvers.

8.2.2 Future work

Superior weak simulator. Since strong quantum simulation is fundamentally unscalable, one must go to weak simulation methods that are intrinsically different from strong simulation methods. It has been long known that efficient exact (or multiplicatively approximate) weak simulation would result in the polynomial hierarchy collapsing to the third level [126], and efficient additively approximate weak simulation would defeat the purpose of quantum computing (i.e. $BQP = BPP$). However, how exactly difficult weak simulation is, and to what size of quantum devices can we hope to perform weak simulation, are important questions yet to be answered. Understanding the difference between weak and strong simulations, identifying fundamental limitations of weak simulations and designing efficient weak simulation methods will be increasingly important with the rapid improvements of quantum devices.

Space efficient strong simulation. Our work only focused on the total time of strong simulation. Although the results set a boundary beyond which strong simulation would be almost absolutely intractable, current simulation methods can still be greatly improved with clever memory allocation and parallelization. In times where space becomes a limiting factor, we are also interested in space-efficient strong simulators, or the tradeoff between time and space for strong simulators. Along this line, there is the Feynman path-integral which has time complexity $O(2^{nd})$, where n is the number of qubits and d is the number of indices in the tensor network (linear to the number of gates). More recently, Aaronson and Chen [75] used Savitchs Theorem to show that one can achieve a $O(d^n)$time-complexity. Does there then exist an even faster space-efficient strong simulator with time-complexity $O(d \cdot 2^n)$? Can we fine-tune such a simulator to achieve a general time-space tradeoff, as in [75]?

8.3 Numerically stable algorithm for angle finding

8.3.1 Summary

In Chapter 6, we investigated the problem of finding angle sequences in quantum signal processing. Quantum signal processing is a simple and elegant quantum algorithm paradigm which achieves optimal complexity for various quantum algorithmic tasks. However, one drawback of quantum signal processing is that there is yet to be a numerically stable algorithm for finding the angles for the single qubit rotations. Although running the quantum circuit is efficient, designing the quantum circuit itself might suffer from numerical instability of angle-finding algorithms.

We investigated the problem from an algebraic perspective, by looking at Laurant polynomials over $M(2, \mathbb{C})$, and some subalgebras of it we called Low and Haah algebra respectively. We analyzed the unitary elements with parity lying in those two algebras, and showed that any element in the unitary parity group is uniquely decomposable. This allows us to decompose a unitary element in a binary manner by solving over-determined linear systems, such that initial error would only propagate polynomially with respect to the degree of the element. We further provided experimental evidence that our method is indeed numerically stable, by showing that angle finding for Hamiltonian simulation can be done more than one magnitude faster than previous algorithms.

8.3.2 Future work

Theoretical proof for numerical stability. We are yet to be able to show that our method is theoretically guaranteed to be numerically stable. Such a proof requires either a better understanding of the linear system we are trying to solve, or a modification of the algorithm (e.g. using weighted least-square instead of the least-square method). We leave this to future work.

More efficient angle-finding algorithms. Both our algorithm and the current state-of-the-art [24] run in time $\tilde{O}(n^3)$, and it is not clear that such time complexity is the best one can achieve. To come up with a faster algorithm, one would need to find an alternative to the root finding step in the completion phase, either by finding roots more quickly for specialized inputs or by designing a novel algorithm that does not need the root information at all. We leave this to future work.

8.4 Transversal switching between stabilizer codes

8.4.1 Summary

In Chapter 7, we considered the problem if finding a path of deformation from one stabilizer code to another, while the intermediate codes all have a large distance. Based on the proposal in [31], we proved theoretical upper bounds that any intermediate code fails to have a large distance using the probabilistic method. With ancilla qubits introduced, this quantity can be made arbitrarily small, yielding a randomized algorithm for code switching while preserving the distance with high probability.

8.4.2 Future work

Fault-tolerant switching between stabilizer codes. The result in Chapter 7 derives its usefulness from its generality. However, there are major drawbacks of the rSRA scheme, which might be improved in future work to generate a fully fault-tolerant code-switching scheme:

- The distance-preserving property is unclear how to verify and how to make use of. Given a stabilizer code, verifying that it is indeed of a certain distance d is a co-NP-complete problem. Although the probabilistic method almost certainly yields a distance-preserving deformation path, it is difficult to verify for a single trial that a distance-preserving path has been found. Moreover, for active error correction, distance is not the only limiting factor for error correction. Being able to *decode* a syndrome, i.e. to find a low-weight error configuration generating a certain syndrome is necessary for correcting that error. It is not known how one can design efficient decoders for each of the randomly generated intermediate codes. It would be more favorable to derandomize the rSRA scheme, such that each individual code has a provable distance and is equipped with an efficient decoder.
- In some cases, especially when we are switching between non-degenerate codes, the distance-preserving property automatically yields fault-tolerance. This is not the case, however, for more general stabilizer codes with degeneracy. It might be the case that different methods other than sequential deformation are needed to overcome this issue.

APPENDIX A

Proof of the Cotlar-Stein Lemma

We present here the proof of the Cotlar-Stein Lemma used in Chapter 3, Section 3.3 for completeness.

Lemma A.1 (Cotlar-Stein Lemma)**.** *For a set of unit vectors* $\{|\psi_1\rangle, |\psi_2\rangle, \cdots, |\psi_n\rangle\}$ *with maximum fidelity* $\max_{i,j:i\neq j} |\langle\psi_i|\psi_j\rangle| \leq \delta$*, we have*

$$\lambda_{\max}\left(\sum_{i=1}^{n} |\psi_i\rangle\langle\psi_i|\right) \leq 1 + (n-1)\delta. \tag{A.1}$$

Proof. We use the fact that the operator norm is upper bounded by all Schatten p norms, i.e.

$$\lambda_{\max}(\rho) = \|\rho\|_\infty = \lim_{p\to\infty} \left(\mathrm{Tr}[\rho^p]\right)^{1/p}. \tag{A.2}$$

For an arbitrary positive integer m, let's now bound

$$\|\sum_{i=1}^{n} |\psi_i\rangle\langle\psi_i|\|_m^m = \mathrm{Tr}\left[(\sum_{i=1}^{n} |\psi_i\rangle\langle\psi_i|)^m\right]. \tag{A.3}$$

We have

$$\begin{aligned}
\mathrm{Tr}\left[(\sum_{i=1}^{n} |\psi_i\rangle\langle\psi_i|)^m\right] &= \sum_{i_1,\cdots,i_m\in[n]} \mathrm{Tr}\left[\prod_{j=1}^{m} |\psi_{i_j}\rangle\langle\psi_{i_j}|\right] && \text{(A.4)}\\
&= \sum_{i_1,\cdots,i_m\in[n]} \prod_{j=1}^{m-1} \langle\psi_{i_j}|\psi_{i_{j+1}}\rangle \cdot \langle\psi_{i_m}|\psi_{i_1}\rangle && \text{(A.5)}\\
&\leq \sum_{i_1,\cdots,i_m\in[n]} \prod_{j=1}^{m-1} |\langle\psi_{i_j}|\psi_{i_{j+1}}\rangle| \cdot |\langle\psi_{i_m}|\psi_{i_1}\rangle|. && \text{(A.6)}
\end{aligned}$$

(A.7)

Using the fact that both $|\psi_{i_m}\rangle$ and $|\psi_{i_1}\rangle$ are unit vectors, we have $|\langle\psi_{i_m}|\psi_1\rangle| \leq 1$. Then

$$\begin{aligned}
\mathrm{Tr}\left[(\sum_{i=1}^{n}|\psi_i\rangle\langle\psi_i|)^m\right] &\leq \sum_{i_1,\cdots,i_m\in[n]}\prod_{j=1}^{m-1}|\langle\psi_{i_j}|\psi_{i_{j+1}}\rangle| && \text{(A.8)}\\
&\leq \sum_{i_1,\cdots,i_{m-1}\in[n]}\prod_{j=1}^{m-2}|\langle\psi_{i_j}|\psi_{i_{j+1}}\rangle|\cdot\sum_{i_m}|\langle\psi_{i_{m-1}}|\psi_{i_m}\rangle| && \text{(A.9)}\\
& && \text{(A.10)}
\end{aligned}$$

Note that for every i_{m-1}, the term $\sum_{i_m}|\langle\psi_{i_{m-1}}|\psi_{i_m}\rangle|$ can be upper bounded by $1+(n-1)\delta$. Repeatedly applying this argument, we have

$$\begin{aligned}
\mathrm{Tr}[(\sum_{i=1}^{n}|\psi_i\rangle\langle\psi_i|)^m] &\leq \sum_{i_1,\cdots,i_{m-1}\in[n]}\prod_{j=1}^{m-2}|\langle\psi_{i_j}|\psi_{i_{j+1}}\rangle|\cdot(1+(n-1)\delta) && \text{(A.11)}\\
&\leq \sum_{i_1,\cdots,i_{m-2}\in[n]}\prod_{j=1}^{m-3}|\langle\psi_{i_j}|\psi_{i_{j+1}}\rangle|\cdot(1+(n-1)\delta)^2 && \text{(A.12)}\\
&\leq \cdots && \text{(A.13)}\\
&\leq \sum_{i_1}(1+(n-1)\delta)^{m-1} && \text{(A.14)}\\
&= n\cdot(1+(n-1)\delta)^{m-1}. && \text{(A.15)}
\end{aligned}$$

Therefore, for every m we have

$$\lambda_{\max}(\sum_{i=1}^{n}|\psi_i\rangle\langle\psi_i|) \leq (1+(n-1)\delta)^{1-\frac{1}{m}}\cdot n^{1/m}. \tag{A.16}$$

The result follows by letting $m\to\infty$. □

APPENDIX B

Proof of the Sparsification Lemma

Given a 3-SAT instance $\phi = C_1 \wedge C_2 \wedge \dots C_m$, we identify each clause $C_1, C_2, \dots C_m$ as a subset of all literals $\{x_1, \neg x_1, x_2, \neg x_2, \dots, x_n, \neg x_n\}$. We start from a simple Boolean identity:

$$(a \vee b) \wedge (a \vee c) = a \vee (b \wedge c). \tag{B.1}$$

This identity implies the following Lemma.

Lemma B.1. *For an arbitrary subset $\{C_1, \dots, C_{m'}\}$ of clauses of ϕ and for $C := \bigcap_{i=1}^{m'} C_i$, we have $\phi = \phi_1 \vee \phi_2$, where*

$$\phi_1 = C \wedge C_{m'+1} \wedge C_{m'+2} \wedge \cdots \wedge C_m, \tag{B.2}$$

$$\phi_2 = (C_1 \setminus C) \wedge (C_2 \setminus C) \wedge \cdots \wedge (C_{m'} \setminus C) \wedge C_{m'+1} \wedge C_{m'+2} \wedge \cdots \wedge C_m. \tag{B.3}$$

Given that ϕ is a 3-SAT instance, both ϕ_1 and ϕ_2 are also 3-SAT instances. Moreover, we also have the following.

Lemma B.2. *Let ϕ, ϕ_1, ϕ_2 be defined as in Lemma B.1. Then neither of the new instances has length greater than the original: $L(\phi) \geq L(\phi_1),\ L(\phi_2)$.*

Sunflowers. We call a collection $C_1, \dots, C_{m'}$ of clauses a (k, h)-*sunflower* (with $h > 0$) if

- Each C_i contains exactly k literals, and
- $C := \bigcap_{i=1}^{m'} C_i$ contains h literals.

C is then called the *heart* of the sunflower and the collection $\{C_1 \setminus C, \cdots, C_{m'} \setminus C\}$ of clauses are called *petals*. The algorithm for sparsification then keeps a set of current 3-SAT formulas whose disjunction is ϕ. Moreover, it repeatedly replaces a formula in this set with two formulas as long as it finds a collection of its clauses that is a large sunflower. One of these new formulas is obtained from the original by replacing the sunflower with its petals, while the other is obtained by replacing the sunflower with its heart.

SPARSIFICATION ALGORITHM. For 3-SAT instances, there are three kinds of sunflowers: $(2,1)$-sunflowers, $(3,2)$-sunflowers, and $(3,1)$-sunflowers. Consider the following algorithm parametrized by θ_1, θ_2: $1 \leq \theta_1 \leq \theta_2$ to be determined. Call a sunflower *good* if it is a $(2,1)$- or $(3,2)$-sunflower of size at least θ_1, or a $(3,1)$-sunflower of size at least θ_2. Among good sunflowers, $(2,1)$-sunflowers have higher priority than $(3,2)$-sunflowers, and $(3,2)$-sunflowers have higher priority than $(3,1)$-sunflowers. Our sparsification algorithm first creates an empty list ℓ, which is a global variable, and then calls (once) the recursive SPARSIFY algorithm below.

```
procedure SPARSIFY($\phi$)
    if $\phi$ does not contain a good sunflower then
        append $\phi$ to $\ell$.
    else
        let $C_1, C_2, \ldots, C_{m'}$ be a good sunflower in $\phi$ with the highest priority and let
$C$ be the heart
        $\phi_h$ =REDUCE($C \wedge C_{m'+1} \wedge C_{m'+2} \wedge \cdots \wedge C_m$)
        $\phi_p$ =REDUCE(($C_1 \backslash C) \wedge (C_2 \backslash C) \wedge \cdots \wedge (C_{m'} \backslash C) \wedge C_{m'+1} \wedge C_{m'+2} \wedge \cdots \wedge C_m$)
        SPARSIFY($\phi_h$); SPARSIFY($\phi_p$)
    end if
end procedure
```

Figure B.1: The Algorithm SPARSIFY.

```
procedure REDUCE($\phi$)
    while $\phi$ contains two clauses $C_i$ and $C_j$, $C_i \subseteq C_j$ do
        remove $C_j$ from $\phi$
    end while
    return $\phi$
end procedure
```

Figure B.2: The Algorithm REDUCE.

Note that the algorithm traverses through a binary recursion tree rooted at ϕ, where each node corresponds to a 3-SAT formula. The set of 3-SAT formulae corresponding to leaf nodes is the collection of instances in ℓ, which is the list we needed to construct. A recursive application of Lemma B.1 gives that $\bigvee_{\phi_i \in \ell} \phi_i = \phi$. We further prove the following.

- Each leaf node corresponds to a formula of length at most $\eta(\theta_1, \theta_2)n$, where

$$\eta(\theta_1, \theta_2) := 2(\theta_1 + \theta_2); \tag{B.4}$$

- There are at most $2^{\gamma(\theta_1,\theta_2)n}$ nodes in the tree, where

$$\gamma(\theta_1, \theta_2) := 4\theta_1 \times H(\frac{1}{4\theta_1^2} + \frac{1}{\theta_2}); \qquad H(p) := -p\log_2 p - (1-p)\log(1-p). \tag{B.5}$$

- Our algorithm runs in time $O(2^{\gamma(\theta_1,\theta_2)}poly(n))$. This follows immediately from the above.

Maximum Length of Each Leaf Node. Given a 3-SAT instance ϕ^*, denote the number of 2-clauses by m_2 and the number of 3-clauses by m_3. Clearly, $L(\phi^*) = 2m_2(\phi^*) + 3m_3(\phi^*) - 1$. Let $d_j(\phi^*)$ be the maximum number of clauses of size j with an nonempty intersection. We have the following observation (by counting the total number of literals in 2-, respectively 3-, clauses):

$$2m_2(\phi^*) \leq 2n \cdot d_2(\phi^*)$$
$$3m_3(\phi^*) \leq 2n \cdot d_3(\phi^*).$$

For a formula ϕ^* on a leaf node, since there are no $(2,1)$-sunflowers of size at least θ_1, we have $d_2(\phi^*) < \theta_1$. Similarly, $d_3(\phi^*) < \max(\theta_1, \theta_2) = \theta_2$. Together with Lemma B.2, these give:

$$L(\phi^*) \;=\; 2m_2(\phi^*) + 3m_3(\phi^*) - 1 \;<\; 2(\theta_1 + \theta_2)m(\phi^*) \;\leq\; 2(\theta_1 + \theta_2)m(\phi). \tag{B.6}$$

Number of Leaf Nodes. To upper bound the number of leaf nodes, we need the notion of *immigrant clauses*. For any formula on some node of the recursion tree, call a clause *immigrant* if that clause is not present in the root. For any path from the root to a leaf, all immigrant clauses that newly appear are distinct (i.e. it cannot happen that an immigrant clause disappears and then reappears later). This leads to the following observation.

Observation B.1. REDUCE *only happens when a newly introduced immigrant clause is contained in previously present clauses.*

The high-level idea of the proof is as follows: we show that there are at most a linear number of immigrant clauses ever introduced. Since in each round at least one immigrant clause is introduced, the recursion tree has linear depth. Moreover, many immigrant clauses are created whenever the petals of a sunflower are taken, and so there must be few such steps, further restricting the number of leaves.

Let $r_2(\phi^*)$ be the maximum number of immigrant 2-clauses with nonempty intersection. Clearly $r_2(\phi^*) \leq d_2(\phi^*)$. The following holds for every node in the recursion tree.

Lemma B.3. *For every formula ϕ^* in the recursion tree, $r_2(\phi^*) \leq 2\theta_1 - 1$.*

Proof. The proof follows by induction from top to bottom. For the root, ϕ, we have $r_2(\phi) = 0$. Next consider a non-top node v on which a new immigrant 2-clause is created, and the corresponding formula ϕ^*. There are two cases to consider.

- ϕ^* takes the heart of a $(3, 2)$-sunflower from its parent ϕ'. Since ϕ' does not have a $(2, 1)$-sunflower of size θ_1, $d_2(\phi') \leq \theta_1 - 1$. By only adding one new 2-clause, we have that
$$r_2(\phi^*) \leq d_2(\phi^*) \leq d_2(\phi') + 1 \leq \theta_1 \leq 2\theta_1 - 1. \tag{B.7}$$

- ϕ^* takes the petals of a $(3, 1)$-sunflower from its parent ϕ'. Similar to the former case, $d_2(\phi') \leq \theta_1 - 1$. Assume that $r_2(\phi^*) \geq 2\theta_1$. Then there exists a literal y which appeared in at least $\theta_1 + 1$ of the newly-formed petals. However, this is not possible as there would be a $(3, 2)$-sunflower of size at least $\theta_1 + 1$ in ϕ', and the algorithm would choose that sunflower instead of a $(3, 1)$-sunflower.

□

This leads to the following observation.

Observation B.2. *An immigrant clause of size one can only reduce at most $2\theta_1 - 1$ immigrant clauses of size two.*

There are at most n immigrant 1-clauses introduced (literals corresponding to a single variable can be immigrant at most once), and so there are at most $(2\theta_1 - 1)n$ immigrant 2-clauses reduced by them (because in each reduction at most $2\theta_1 - 1$ immigrant clauses are eliminated). When the algorithm arrives at a leaf, ϕ^*, because $r_2(\phi^*) \leq 2\theta_1 - 1$, we have that the total number of immigrant 2-clauses that remains is at most $\frac{(2\theta_1 - 1)2n}{2} < (2\theta_1 - 1)n$. Thus the number of immigrant 2-clauses ever introduced is at most $(4\theta_1 - 2)n$. In each step going down in the recursion tree at least one new immigrant one- or two- clause was created, and so depth of the recursion tree is at most $(4\theta_1 - 2)n + n < 4\theta_1 n$. This alone would not be sufficient to get a good estimate on the number number of leaves, but we further observe the following.

Each time the petals of a sunflower are taken, either at least θ_1 1-clauses are introduced, or at least θ_2 2-clauses are introduced. Therefore, the number of petals taken along a path from the root to a leaf is at most $\frac{n}{\theta_1} + \frac{4\theta_1 n}{\theta_2}$. This gives the bound

$$\sum_{i=0}^{\frac{n}{\theta_1}+\frac{4\theta_1 n}{\theta_2}} \binom{4\theta_1 n}{i} \leq 2^{\gamma(\theta_1,\theta_2)n}, \tag{B.8}$$

on the number of leafs, where $\gamma(\theta_1, \theta_2) \leq 4\theta_1 H(\frac{1}{4\theta_1^2} + \frac{1}{\theta_2})$.

Optimization. First, note that Equation B.8 and $H(p)/p(1 + \log_2 \frac{1}{p}) \longrightarrow 1$ at $p = 0$ given that $\gamma(\theta_1, \theta_2)$ can be arbitrarily small at $\theta_2 = 4\theta_1^2$ and for θ_1 sufficiently large. This, together with Equation B.6, gives the Sparsification Lemma.

Next, we compute the values for θ_1 and θ_2 that optimize the hardness reductions from instances whose size parameters are expressed in terms of n to instances whose size parameters are expressed in terms of L.

B.1 Proof of Lemma 5.4

In this section, we prove Lemma 5.4 for completeness. Recall Lemma 5.4:

Lemma B.4 (Lemma 5.4, restates)**.** *Assuming the ETH, there exists constant $a > 0$ such that any classical algorithm solving* 3-SAT *instances with length L takes 2^{aL} time, where again L is the* length *of the formula.*

Proof. Suppose that for each $a > 0$, contradictory to Lemma 5.4, there is an algorithm $\textsc{Solve}_a$ that solves 3-SAT in time $O(2^{aL})$. We show that the existence of such a family of algorithms implies the existence of a family of algorithms that solve 3-SAT in time $O(2^{\epsilon n})$ for every $\epsilon > 0$, contradicting the ETH. Given $\epsilon > 0$, set $\epsilon' = \epsilon/2$. Consider the following algorithm.

1. Given a 3-SAT instance ϕ over n variables, run the SPARSIFICATION ALGORITHM and get k 3-SAT instances $\phi_1, \ldots, \phi_k$, where $k \leq 2^{\epsilon' n}$, each of length at most $c(\epsilon')n$.

2. Solve every instance $\phi_1, \ldots, \phi_k$ using the algorithm $\textsc{Solve}_{\epsilon'/c(\epsilon')n}$.

3. If any of the ϕ_i are satisfiable, output 1; otherwise output 0.

By the Sparsification Lemma, Step 1 takes time $2^{\epsilon' n} poly(n)$ time. Solving an instance in Step 2 takes time $2^{\frac{\epsilon'}{c(\epsilon')} c(\epsilon')n}$ since $L \leq c(\epsilon')n$. The total running time of Step 2 is then

$2^{(\epsilon'+\epsilon')n} = 2^{\epsilon n}$. Finally, Step 3 combines the results from Step 2, and so takes time $O(2^{\epsilon' n})$. The overall running time is then dominated by $2^{\epsilon n}$. $\square$

B.2 Proof of Lemma 5.5

In this section, we present the proof of Lemma 5.5 for completeness. The proof of Lemma 5.5 is very similar to the proof of Lemma 5.4, except that we now have to calculate the explicit constants. Recall Lemma 5.5:

Lemma B.5 (Lemma 5.5, restated). *Assume that a classical algorithm solves* 3-SAT *in time $O(2^{3.1432\times 10^{-7}L})$, where L is the length of the formula, m_2 is the number of* 2*-clauses, m_3 is the number of* 3*-clauses, so that $m = m_2 + m_3$ and $L = 2m_2 + 3m_3 - 1$.*

Then one can create a 3-SAT *solver that achieves an $O(1.3^n)$ running-time for $m = \text{poly}(n)$, where n denotes the number of variables of the* 3-SAT *instance and m denotes the number of clauses.*

Proof. Assume we had a 3-SAT solver SOLVE that runs in time $o(2^{3.1432\times 10^{-7}L})$ on instances of length L. From it, we construct a 3-SAT solver SOLVE' that runs in time $o(1.3^n)$, beating the current best 3-SAT solver. Let $\theta_1 = 109.395$ and $\theta_2 = 58367.2$. SOLVE' will then

- Run the SPARSIFICATION ALGORITHM on input ϕ to get a list ℓ of $2^{\gamma(\theta_1,\theta_2)n}$ sparse instances in time $O(2^{\gamma(\theta_1,\theta_2)n} poly(n))$, each with length $\eta(\theta_1,\theta_2)n$. We have $\bigvee_{\phi_i \in \ell} \phi_i = \phi$.

- Use SOLVE to solve each instance ϕ_i in time $o(2^{3.1432\times 10^{-7}\eta(\theta_1,\theta_2)n})$. The total running time is then less than
$$2^{3.1432\times 10^{-7}(\eta(\theta_1,\theta_2)+\gamma(\theta_1,\theta_2))n}. \tag{B.9}$$

- If any of the instances are satisfiable, it outputs 1, otherwise it outputs 0. This step takes time $O(2^{\gamma(\theta_1,\theta_2)n})$.

The dominating term in the running time is then $2^{3.1432\times 10^{-7}(\eta(\theta_1,\theta_2)+\gamma(\theta_1,\theta_2))n}$. An easy calculation shows that SOLVE' runs in $o(1.3^n)$, beating the current best bound. $\square$

APPENDIX C

Lemmas for code distance bounds

In this appendix, we provide proofs of some technical lemmas in Chapter 7 for completeness.

Lemma C.1. *Let* $v, w \in \{0,1\}^n \setminus \{0\}$ *and* $U \in_r GL(\mathbb{F}_2, n)$. *Let* $i_0 = \max\{i : (U \cdot v)_i = 1\}$ *and* $i_1 = \min\{i : ((U^{-1})^T \cdot w)_i = 1\}$. *Then,*

$$\Pr[i_0 < i_1] \leq (n-1) \cdot 2^{-n}. \tag{C.1}$$

Proof. Let $\langle \cdot, \cdot \rangle$ be the dot product over $\mathbb{F}_2$. Note that $\langle v, w \rangle = \langle U \cdot v, (U^{-1})^T \cdot w \rangle$. If $\langle v, w \rangle = 1$, then there must be at least one entry where both $U \cdot v$ and $(U^{-1})^T \cdot w$ are 1 for whichever U we choose, and so $\Pr[i_0 < i_1] = 0$. Therefore we only need to consider the case in which $\langle v, w \rangle = 0$.

Consider the action of $GL(\mathbb{F}_2, n)$ on A defined by $U(v, w) \to (U \cdot v, (U^{-1})^T \cdot w)$, where $A = \{(v, w) | v, w \in \{0,1\}^n \setminus \{0\}, \langle v, w \rangle = 0\}$. We show that the action is transitive by showing that for all such pairs (v, w), there always exists a U sending (e_1, e_n) to (v, w), where e_1, e_n are $(1, 0, \ldots, 0)$ and $(0, 0, 0, \ldots, 1)$, respectively. Given such a (v, w), first extend v to a basis for $w^\perp$, say $(u_1 = v, u_2, \ldots, u_{n-1})$, and then extend it to the whole space by adding in u_n. We claim that $U = (u_1, \cdots, u_n)$ is the desired matrix. It is sufficient to show that the last column w' of $(U^{-1})^T$ is exactly w. We have $U^T w' = e_n$ given that $U^T (U^{-1})^T = I$, and that $U^T w = e_n$ by construction of U. Then, since U is invertible, $w = w'$.

A uniformly random distribution over invertible U then induces a uniformly random distribution over A. Then $\Pr[i_0 < i_1]$ can then be bounded by counting the number of such pairs in A:

$$\Pr[i_0 < i_1] = \frac{\sum_{i_0=1}^{n} 2^{i_0-1}(2^{n-i_0}-1)}{(2^n-1)(2^{n-1}-1)} \tag{C.2}$$
$$= \frac{(n-2)2^{n-1}+1}{(2^n-1)(2^{n-1}-1)} \tag{C.3}$$
$$\leq (n-1)\cdot 2^{-n} \tag{C.4}$$

when $n \geq 2$. Note that $|A| = 0$ when $n = 1$, so $\Pr[i_0 < i_1] \leq (n-1)\cdot 2^{-n}$ holds for all $n \geq 0$. □

Lemma C.2. *Let G_A, G_B, G_C and G_A, G'_B, G'_C be the matrices defined up to step 5 in the rSRA scheme. The commutativity matrix $H = G_C^T B G'_C$ is invertible, and its dimension is $|G_C|$, with $|G_C| \geq m$.*

Proof. For the two codes $\hat{S}$ and $\hat{S}'$, take arbitrary generator matrices G,G' and define $H' = G^T B G'$. Note that any two choices of generator matrices for the same code differ by an invertible row transformation, so the rank of H' is invariant under different choices of the generator matrices. In particular, letting $G = (G_A|G_B|G_C), G' = (G'_A|G'_B|G'_C)$, we have

$$H' = \begin{bmatrix} 0 & & \\ & 0 & \\ & & H \end{bmatrix}. \tag{C.5}$$

Note that $\operatorname{rank}(H) = |G_C|$, or else there would exist a combination of the rows of ${G'_C}^T$ that are orthogonal to all the columns of G_C. Since all the vectors in G'_C are already orthogonal to G_A and G_B by definition, this cannot happen as no vector in G'_C lies in $\mathcal{N}(S)$. The same argument applies to G'_C as well. Therefore H is invertible, with $\operatorname{rank}(H') = \operatorname{rank}(H) = |G_C|$, and is independent of the choice of G_C.

To show that $|G_C| > m$, take $\bar{G}_C = (I^{\otimes n} \otimes Z \otimes I^{\otimes m-1}, \ldots, I^{\otimes n+m-1} \otimes Z)$ and $\bar{G}'_C = (I^{\otimes n} \otimes X \otimes I^{\otimes m-1}, \ldots, I^{\otimes n+m-1} \otimes X)$, each of size m. By extending them to generator matrices $\bar{G}$ and $\bar{G}'$ for $\hat{S}$ and $\hat{S}'$ respectively, we get a commutativity matrix $\bar{H}$ with an invertible submatrix of size $m \times m$, namely

$$\bar{G}_C^T B \bar{G}'_C = I_m, \tag{C.6}$$

and so $\operatorname{rank}(\bar{H}) = |G_C| \geq m$. □

Lemma C.3. *For $D(\cdot||\cdot)$ the KL-divergence, let*

$$P(n,m,d) = 4^{n+m} e^{-D(\frac{d}{n+m}\|\frac{3}{4})(n+m)} \cdot (m+1) \cdot 2^{-m}. \tag{C.7}$$

Then $P < \varepsilon$ for some $m = O(d \log \frac{n}{d} + \log \frac{1}{\varepsilon})$.

Proof. Let $\alpha = m/n$. Then $P(n,m,d) < \varepsilon$ can be rewritten as

$$f(n,m,d) := \log \frac{P(n,m,d)}{\varepsilon} \tag{C.8}$$

$$= \log \frac{m+1}{\varepsilon} + n\Bigg((2+\alpha)\log 2 \tag{C.9}$$

$$- (1+\alpha) D\bigg(\frac{d}{n(1+\alpha)} \| \frac{3}{4} \bigg) \Bigg) < 0. \tag{C.10}$$

We first compute the dominant term, i.e. the α such that

$$(2+\alpha)\ln 2 - (1+\alpha) D\left(\frac{d}{n(1+\alpha)} \| \frac{3}{4} \right) = 0. \tag{C.11}$$

Doing this we obtain

$$(2 - \frac{\alpha}{1+\alpha}) \ln 2 = D\left(\frac{d}{n(1+\alpha)} \| \frac{3}{4} \right) \tag{C.12}$$

$$(2 - \frac{\alpha}{1+\alpha}) \ln 2 \geq 2 \ln 2 + \frac{d}{n(1+\alpha)} \left(\ln \frac{d}{3n(1+\alpha)} - 1 \right) \tag{C.13}$$

$$\alpha n \leq \frac{d}{\ln 2} \left(\ln \frac{3n(1+\alpha)}{d} + 1 \right) \tag{C.14}$$

$$m \leq \frac{1}{\ln 2} d (\log \frac{n}{d} + (1 + \ln 3)), \tag{C.15}$$

where we have used convexity of $D(p\|q) - p \ln p$ with respect to p. Letting $\tilde{\alpha}$ denote the solution to $(2+\alpha)\ln 2 - (1+\alpha) D\left(\frac{d}{n(1+\alpha)} \| \frac{3}{4} \right) = 0$, we have that $\tilde{m} := \tilde{\alpha} n = O(d + d \log \frac{n}{d})$.

We now have that $f(n, \tilde{m}, d) = \log \frac{\tilde{m}+1}{\epsilon}$. Taking the derivative of f with respect to m, for all $\alpha > \tilde{\alpha}$ we have

$$\frac{\partial f(n,m,d)}{\partial m} = \tag{C.16}$$

$$= \frac{1}{m+1} - D(\frac{d}{(n+m)} \| \frac{3}{4}) - (m+n)\frac{\partial D(\frac{d}{n+m}\|\frac{3}{4})}{\partial m} \tag{C.17}$$

$$\leq \frac{1}{m+1} - \frac{2+\tilde{\alpha}}{1+\tilde{\alpha}}\log 2 + \frac{d}{m+n}\left(\log \frac{d}{3(m+n-d)}\right) \tag{C.18}$$

$$\leq \frac{1}{m+1} - \frac{1}{1+\tilde{\alpha}}\log 2 + \frac{d}{n+m}\left(\log \frac{d}{3(n+m-d)}\right) \tag{C.19}$$

$$\leq -\frac{1}{1+\tilde{\alpha}}\ln 2 + 0.1 \tag{C.20}$$

for $m \geq 10$. For fixed n, $\tilde{\alpha}$ is monotonically increasing as a function of d. By the quantum singleton bound, $\frac{d-1}{n} < \frac{1}{2}$, and $\tilde{\alpha} < 3$ even in this case. Therefore $\frac{\partial f(n,m,d)}{\partial m} \leq -0.05$ when $m \geq 10$, so taking

$$m = \tilde{m} + 20\log\frac{\tilde{m}+1}{\epsilon} + O(1) = O(d\log\frac{n}{d} + \log\frac{1}{\varepsilon}) \tag{C.21}$$

suffices to make $f(n,m,d) < 0$. □

APPENDIX D

Fault-tolerant measurement

For completeness, we include a code switching circuit between adjacent codes, using Shor-style measurement [123]. We assume access to a collection of verified CAT states. Let $g = P_1 \otimes \ldots \otimes P_n$ and $g' = P_1' \otimes \ldots \otimes P_n'$. The measurements are done on the supports of g and g'. To make the diagram simpler, we suppose that the supports include qubits $1, 2$, and n. Then the circuit obtained from the SRA to convert from the code with stabilizer g' to the adjacent code with stabilizer g is given by the following.

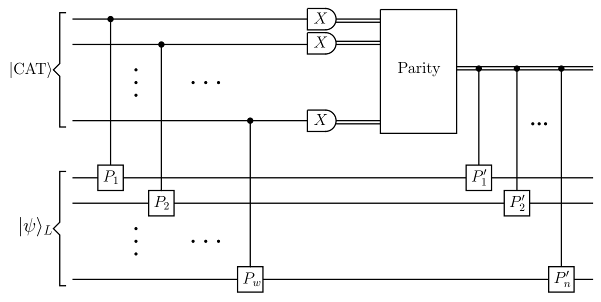


Figure D.1: A generic circuit switching between adjacent codes using Shor-style measurement.

BIBLIOGRAPHY

[1] Richard P. Feynman. Simulating physics with computers. *International Journal of Theoretical Physics*, 21(6-7):467–488, 1982.

[2] Peter W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM Journal on Computing*, 26(5):1484–1509, 1997. Earlier version in FOCS'94. arXiv: quant-ph/9508027

[3] Lov K. Grover. A fast quantum mechanical algorithm for database search. In *Proceedings of the 28th ACM Symposium on Theory of Computing (STOC)*, pages 212–219, 1996. arXiv: quant-ph/9605043

[4] Edward Farhi, Jeffrey Goldstone, and Sam Gutmann. A quantum approximate optimization algorithm. arXiv: 1411.4028, 2014.

[5] Aram W. Harrow, Avinatan Hassidim, and Seth Lloyd. Quantum algorithm for linear systems of equations. *Physical Review Letters*, 103(15):150502, 2009. arXiv: 0811.3171

[6] Seth Lloyd, Masoud Mohseni, and Patrick Rebentrost. Quantum principal component analysis. *Nature Physics*, 10:631–633, 2014. arXiv: 1307.0401

[7] Andrew M. Childs and Nathan Wiebe. Hamiltonian simulation using linear combinations of unitary operations. *Quantum Information and Computation*, 12(11&12):901–924, 2012. arXiv: 1202.5822

[8] Dominic W. Berry, Andrew M. Childs, and Robin Kothari. Hamiltonian simulation with nearly optimal dependence on all parameters. In *Proceedings of the 56th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 792–809, 2015. arXiv: 1501.01715

[9] R. Harris, Y. Sato, A. J. Berkley, M. Reis, F. Altomare, M. H. Amin, K. Boothby, P. Bunyk, C. Deng, C. Enderud, et al. Phase transitions in a programmable quantum spin glass simulator. *Science*, 361(6398):162–165, 2018.

[10] John Preskill. Quantum Computing in the NISQ era and beyond. *Quantum*, 2:79, 2018. arXiv: 1801.00862

[11] Andris Ambainis. Understanding quantum algorithms via query complexity. In *Proceedings of International Congress of Mathematicians'2018*, 2017. arXiv: 1712.06349

[12] Sergey Bravyi, David Gosset, and Robert König. Quantum advantage with shallow circuits. *Science*, 362(6412):308–311, 2018. arXiv: 1704.00690

[13] Avishay Tal. Oracle separation of bqp and ph. 2018.

[14] Mark M. Wilde. *Quantum information theory*. Cambridge University Press, 2013.

[15] Harry Buhrman, Richard Cleve, John Watrous, and Ronald de Wolf. Quantum fingerprinting. *Physical Review Letters*, 87(16):167902, 2001. arXiv: quant-ph/0102001

[16] Cupjin Huang, Michael Newman, and Márió Szegedy. Explicit lower bounds on strong quantum simulation. arXiv: 1804.10368, 2018.

[17] Guang Hao Low, Theodore J. Yoder, and Isaac L. Chuang. Methodology of resonant equiangular composite quantum gates. *Physical Review X*, 6(4):041067, 2016. arXiv: 1603.03996

[18] Andrew M. Childs, Robin Kothari, and Rolando D. Somma. Quantum algorithm for systems of linear equations with exponentially improved dependence on precision. *SIAM Journal on Computing*, 46(6):1920–1950, 2017. arXiv: 1511.02306

[19] Joran van Apeldoorn, András Gilyén, Sander Gribling, and Ronald de Wolf. Quantum SDP-solvers: Better upper and lower bounds. In *Proceedings of the 58th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 403–414, 2017. arXiv: 1705.01843

[20] Guang Hao Low and Isaac L. Chuang. Hamiltonian simulation by qubitization. arXiv: 1610.06546, 2016.

[21] Guang Hao Low and Isaac L. Chuang. Hamiltonian simulation by uniform spectral amplification. arXiv: 1707.05391, 2017.

[22] Shantanav Chakraborty, András Gilyén, and Stacey Jeffery. The power of block-encoded matrix powers: improved regression techniques via faster Hamiltonian simulation. arXiv: 1804.01973, 2018.

[23] András Gilyén, Yuan Su, Guang Hao Low, and Nathan Wiebe. Quantum singular value transformation and beyond: exponential improvements for quantum matrix arithmetics. arXiv: 1806.01838, 2018.

[24] Jeongwan Haah. Product decomposition of periodic functions in quantum signal processing. arXiv: 1806.10236, 2018.

[25] Bryan Eastin and Emanuel Knill. Restrictions on transversal encoded quantum gate sets. *Physical Review Letters*, 102:110502, July 2009. arXiv: 0811.4262

[26] Jonas T. Anderson, Guillaume Duclos-Cianci, and David Poulin. Fault-tolerant conversion between the Steane and Reed-Muller quantum codes. *Physical Review Letters*, 113:080501, 2014. arXiv: 1403.2734

[27] Charles D. Hill, Austin G. Fowler, David S. Wang, and Lloyd C. L. Hollenberg. Fault-tolerant quantum error correction code conversion. *Quantum Information and Computation*, 13(5-6):439–451, 2013. arXiv: 1112.2417

[28] Todd A. Brun, Yi-Cong Zheng, Kung-Chuan Hsu, Joshua Job, and Ching-Yi Lai. Teleportation-based fault-tolerant quantum computation in multi-qubit large block codes. 2015. arXiv: 1504.03913

[29] Hector Bombín and Miguel Angel Martin-Delgado. Quantum measurements and gates by code deformation. *Journal of Physics A: Mathematical and Theoretical*, 42(9):095302, 2009.

[30] Hendrik Poulsen Nautrup, Nicolai Friis, and Hans J. Briegel. Fault-tolerant interface between quantum memories and quantum processors. *Nature Communications*, 8(1):1321, 2017.

[31] Kristina R. Colladay and Erich J. Mueller. Rewiring stabilizer codes. *New Journal of Physics*, 20(8):083030, 2018. arXiv: 1707.09403

[32] Michael A. Nielsen and Isaac L. Chuang. *Quantum computation and quantum information*. Cambridge University Press, 2000.

[33] Cupjin Huang and Yaoyun Shi. Quantum hashing is maximally secure against classical leakage. arXiv: 1701.01091, 2017.

[34] Cupjin Huang, Michael Newman, and Márió Szegedy. Explicit lower bounds on strong simulation of quantum circuits in terms of t-gate count. arXiv: 1902.04764, 2019.

[35] Tomoyuki Morimae and Suguru Tamaki. Fine-grained quantum computational supremacy. arXiv: 1901:01637, 2019.

[36] Rui Chao, Dawei Ding, András Gilyén, Cupjin Huang, and Mario Szegedy. Finding angles for quantum signal processing with machine precision.

[37] Cupjin Huang and Michael Newman. Transversal switching between generic stabilizer codes. arXiv: 1709.09282, 2017.

[38] Jianxin Chen, Fang Zhang, Cupjin Huang, Michael Newman, and Yaoyun Shi. Classical simulation of intermediate-size quantum circuits. arXiv: 1805.01450, 2018.

[39] Jarrod R. McClean, Ian D. Kivlichan, Damian S. Steiger, Yudong Cao, E. Schuyler Fried, Craig Gidney, Thomas Häner, Vojtěch Havlíček, Zhang Jiang, Matthew Neeley, et al. OpenFermion: the electronic structure package for quantum computers. arXiv: 1710.07629, 2017.

[40] Fang Zhang, Cupjin Huang, Michael Newman, Kevin Sung, and Yaoyun Shi. Limitations on testing quantum theory. 2017.

[41] Daniel Gottesman. The heisenberg representation of quantum computers. In *Proceedings of the 22nd International Colloquium on Group Theoretical Methods in Physics*, pages 32–43, 1999. arXiv: quant-ph/9807006

[42] Dan Boneh and Victor Shoup. *A Graduate Course in Applied Cryptography*. 2015. Available at https://crypto.stanford.edu/~dabo/cryptobook/draft_0_2.pdf.

[43] Bart Preneel. Cryptographic hash functions. *European Transactions on Telecommunications*, 5(4):431–448, 1994.

[44] Phillip Rogaway and Thomas Shrimpton. Cryptographic hash-function basics: Definitions, implications, and separations for preimage resistance, second-preimage resistance, and collision resistance. In *International Workshop on Fast Software Encryption*, pages 371–388. Springer, 2004.

[45] Joseph Naor and Moni Naor. Small-bias probability spaces: Efficient constructions and applications. *SIAM Journal on Computing*, 22(4):838–856, 1993.

[46] Srinivasan Arunachalam and Ronald de Wolf. Optimal quantum sample complexity of learning algorithms. *Journal of Machine Learning Research*, 19(71):1–36, 2018. arXiv: 1607.00932

[47] Noam Nisan. Pseudorandom generators for space-bounded computation. *Combinatorica*, 12(4):449–461, 1992.

[48] Noam Nisan and David Zuckerman. Randomness is linear in space. *Journal of Computer and System Sciences*, 52(1):43–52, 1996.

[49] Luca Trevisan. Extractors and pseudorandom generators. *Journal of the ACM*, 48(4):860–879, 2001.

[50] Charles H Bennett, Gilles Brassard, and Jean-Marc Robert. How to reduce your enemys information. In *Conference on the Theory and Application of Cryptographic Techniques*, pages 468–476. Springer, 1985.

[51] Yevgeniy Dodis and Daniel Wichs. Non-malleable extractors and symmetric key cryptography from weak secrets. pages 601–610. ACM, 2009.

[52] Oded Goldreich, Rehovot Israel, and David Zuckerman. Another proof that $BPP \subseteq PH$ (and more). In *Electronic Colloquium on Computational Complexity*. Citeseer, 1997.

[53] Madhu Sudan, Luca Trevisan, and Salil Vadhan. Pseudorandom generators without the XOR lemma. In *Proceedings of the 31st ACM Symposium on Theory of Computing (STOC)*, pages 537–546. ACM, 1999.

[54] Dana Moshkovitz. Parallel repetition from fortification. In *Proceedings of the 55th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 414–423. IEEE, 2014.

[55] Charles H. Bennett, Gilles Brassard, Claude Crépeau, and Ueli M Maurer. Generalized privacy amplification. *IEEE Transactions on Information Theory*, 41(6):1915–1923, 1995.

[56] Ueli M. Maurer. Secret key agreement by public discussion from common information. *IEEE Transactions on Information Theory*, 39(3):733–742, 1993.

[57] Dmitry Gavinsky, Julia Kempe, Iordanis Kerenidis, Ran Raz, and Ronald de Wolf. Exponential separations for one-way quantum communication complexity, with applications to cryptography. In *Proceedings of the 39th ACM Symposium on Theory of Computing (STOC)*, pages 516–525. ACM, 2007.

[58] Robert T. König and Barbara M. Terhal. The bounded-storage model in the presence of a quantum adversary. *IEEE Transactions on Information Theory*, 54(2):749–762, 2008. arXiv: quant-ph/0608101

[59] F. M. Ablayev and A. V. Vasiliev. Cryptographic quantum hashing. *Laser Physics Letters*, 11(2):025202, 2013.

[60] François-Xavier Standaert, Tal Malkin, and Moti Yung. A unified framework for the analysis of side-channel key recovery attacks. In Antoine Joux, editor, *Advances in Cryptology - EUROCRYPT 2009, 28th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Cologne, Germany, April 26-30, 2009. Proceedings*, volume 5479 of *Lecture Notes in Computer Science*, pages 443–461. Springer, 2009.

[61] Moni Naor and Gil Segev. Public-key cryptosystems resilient to key leakage. In *Advances in Cryptology-CRYPTO 2009*, pages 18–35. Springer, 2009.

[62] Adi Akavia, Shafi Goldwasser, and Vinod Vaikuntanathan. Simultaneous hardcore bits and cryptography against memory attacks. In *Theory of Cryptography Conference*, pages 474–495. Springer, 2009.

[63] Yevgeniy Dodis, Kristiyan Haralambiev, Adriana López-Alt, and Daniel Wichs. Efficient public-key cryptography in the presence of key leakage. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 613–631. Springer, 2010.

[64] Eric Brier and Marc Joye. Weierstraß elliptic curves and side-channel attacks. In *International Workshop on Public Key Cryptography*, pages 335–345. Springer, 2002.

[65] Marc Joye and Jean-Jacques Quisquater. Hessian elliptic curves and side-channel attacks. In *International Workshop on Cryptographic Hardware and Embedded Systems*, pages 402–410. Springer, 2001.

[66] Julian Kelly, R. Barends, A. G. Fowler, . Megrant, E. Jeffrey, T. C. White, D. Sank, J. Y. Mutus, B. Campbell, Yu Chen, et al. State preservation by repetitive error detection in a superconducting quantum circuit. *Nature*, 519(7541):66, 2015. arXiv: 1411.7403

[67] Chao Song, Kai Xu, Wuxin Liu, Chui-Ping Yang, Shi-Biao Zheng, Hui Deng, Qiwei Xie, Keqiang Huang, Qiujiang Guo, Libo Zhang, et al. 10-qubit entanglement and parallel logic operations with a superconducting circuit. *Physical Review Letters*, 119(18):180511, 2017.

[68] K. De Raedt, K. Michielsen, H. De Raedt, B. Trieu, G. Arnold, M. Richter, T. Lippert, H. Watanabe, and N. Ito. Massively parallel quantum computer simulator. *Computer Physics Communications*, 176:121–136, January 2007. arXiv: quant-ph/0608239

[69] Thomas Häner and Damian S Steiger. 0.5 petabyte simulation of a 45-qubit quantum circuit. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, page 33. ACM, 2017.

[70] M. Smelyanskiy, N. P. D. Sawaya, and A. Aspuru-Guzik. qHiPSTER: The Quantum High Performance Software Testing Environment. arXiv: 1601.07195, January 2016.

[71] Edwin Pednault, John A Gunnels, Giacomo Nannicini, Lior Horesh, Thomas Magerlein, Edgar Solomonik, and Robert Wisnieff. Breaking the 49-qubit barrier in the simulation of quantum circuits. arXiv: 1710.05867, 2017.

[72] Sergio Boixo, Sergei V Isakov, Vadim N Smelyanskiy, and Hartmut Neven. Simulation of low-depth quantum circuits as complex undirected graphical models. 2017. arXiv: 1712.05384

[73] Zhaoyun Chen, Qi Zhou, Cheng Xue, Xia Yang, Guangcan Guo, and Guoping Guo. 64-Qubit Quantum Circuit Simulation. arXiv: 1802.06952, 2018.

[74] R. Li, B. Wu, M. Ying, X. Sun, and G. Yang. Quantum supremacy circuit simulation on Sunway TaihuLight. April 2018. arXiv: 1804.04797

[75] Scott Aaronson and Lijie Chen. Complexity-theoretic foundations of quantum supremacy experiments. In *Proceedings of the 32nd IEEE Conference on Computational Complexity (CCC)*, page 22. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2017. arXiv: 1612.05903

[76] Alberto Peruzzo, Jarrod McClean, Peter Shadbolt, Man-Hong Yung, Xiao-Qi Zhou, Peter J Love, Alán Aspuru-Guzik, and Jeremy L. Obrien. A variational eigenvalue solver on a photonic quantum processor. *Nature Communications*, 5, 2014. arXiv: 1304.3061

[77] Jarrod R McClean, Jonathan Romero, Ryan Babbush, and Alán Aspuru-Guzik. The theory of variational hybrid quantum-classical algorithms. *New Journal of Physics*, 18(2):023023, 2016. arXiv: 1509.04279

[78] Leo Zhou, Sheng-Tao Wang, Soonwon Choi, Hannes Pichler, and Mikhail D Lukin. Quantum approximate optimization algorithm: Performance, mechanism, and implementation on near-term devices. arXiv: 1812.01041, 2018.

[79] Sergio Boixo, Sergei V Isakov, Vadim N Smelyanskiy, Ryan Babbush, Nan Ding, Zhang Jiang, John M Martinis, and Hartmut Neven. Characterizing quantum supremacy in near-term devices. *Nature Physics*, 14:595–600, 2018. arXiv: 1608.00263

[80] Scott Aaronson. Shadow tomography of quantum states. In *Proceedings of the 50th ACM Symposium on Theory of Computing (STOC)*, pages 325–338, 2018. arXiv: 1711.01053

[81] Sergey Bravyi and David Gosset. Improved classical simulation of quantum circuits dominated by clifford gates. *Physical Review Letters*, 116(25):250501, 2016. arXiv: 1601.07601

[82] Sergey Bravyi, Dan Browne, Padraic Calpin, Earl Campbell, David Gosset, and Mark Howard. Simulation of quantum circuits by low-rank stabilizer decompositions. 2018. arXiv: 1808.00128

[83] Ryan S. Bennink, Erik M. Ferragut, Travis S. Humble, Jason A. Laska, James J. Nutaro, Mark G. Pleszkoch, and Raphael C. Pooser. Unbiased simulation of near-clifford quantum circuits. *Physical Review A*, 95(6):062337, 2017. arXiv: 1703.00111

[84] Igor L. Markov and Yaoyun Shi. Simulating quantum computation by contracting tensor networks. *SIAM Journal on Computing*, 38(3):963–981, 2008. arXiv: quant-ph/0511069

[85] Richard P. Feynman, Albert R. Hibbs, and Daniel F. Styer. *Quantum mechanics and path integrals*. Courier Corporation, 1965.

[86] Volker Strassen. Gaussian elimination is not optimal. *Numerische Mathematik*, 13(4):354–356, 1969.

[87] Leslie G Valiant. Quantum circuits that can be simulated classically in polynomial time. *SIAM Journal on Computing*, 31(4):1229–1254, 2002.

[88] Scott Aaronson and Daniel Gottesman. Improved simulation of stabilizer circuits. *Physical Review A*, 70(5):052328, 2004. arXiv: quant-ph/0406196

[89] Sergey Bravyi, Graeme Smith, and John A. Smolin. Trading classical and quantum computational resources. *Physical Review X*, 6(2):021043, 2016.

[90] Zhengfeng Ji and Xiaodi Wu. Non-identity check remains QMA-complete for short circuits. arXiv: 0906.5416, 2009.

[91] Mark Jerrum and Marc Snir. Some exact complexity results for straight-line computations over semirings. *Journal of the ACM*, 29(3):874–897, 1982.

[92] Ashley Montanaro. Quantum circuits and low-degree polynomials over. *Journal of Physics A: Mathematical and Theoretical*, 50(8):084002, 2017. arXiv: 1607.08473

[93] Adam Bouland, Bill Fefferman, Chinmay Nirkhe, and Umesh Vazirani. Quantum supremacy and the complexity of random circuit sampling. 2018. arXiv: 1803.04402

[94] Terry Rudolph. Simple encoding of a quantum circuit amplitude as a matrix permanent. *Physical Review A*, 80(5):054302, 2009. arXiv: 0909.3005

[95] Scott Aaronson and Alex Arkhipov. The computational complexity of linear optics. In *Proceedings of the 43rd ACM Symposium on Theory of Computing (STOC)*, pages 333–342, 2011. arXiv: 1011.3245

[96] Richard A. Brualdi and Herbert John Ryser. *Combinatorial matrix theory*, volume 39. Springer, 1991.

[97] David G. Glynn. The permanent of a square matrix. *European Journal of Combinatorics*, 31(7):1887–1891, 2010.

[98] Russell Impagliazzo, Ramamohan Paturi, and Francis Zane. Which problems have strongly exponential complexity? *Journal of Computer and System Sciences*, 63(4):512–530, 2001.

[99] Ramamohan Paturi, Pavel Pudlák, and Francis Zane. Satisfiability coding lemma. In *Proceedings of the 38th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 566–574. IEEE, 1997.

[100] Ramamohan Paturi, Pavel Pudlák, Michael E Saks, and Francis Zane. An improved exponential-time algorithm for k-SAT. *Journal of the ACM*, 52(3):337–364, 2005.

[101] T. Schöning. A probabilistic algorithm for k-SAT and constraint satisfaction problems. In *Proceedings of the 40th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 410–414. IEEE, 1999.

[102] Timon Hertli. 3-SAT faster and simpler—Unique-SAT bounds for PPSZ hold in general. *SIAM Journal on Computing*, 43(2):718–729, 2014. arXiv: 1103.2165

[103] Chris Calabro, Russell Impagliazzo, and Ramamohan Paturi. A duality between clause width and clause density for SAT. volume 1, pages 252–260. IEEE, 2006.

[104] Charles H. Bennett. Time/space trade-offs for reversible computation. *SIAM Journal on Computing*, 18(4):766–776, 1989.

[105] Alexander M. Dalzell, Aram W. Harrow, Dax Enshan Koh, and Rolando L. La Placa. How many qubits are needed for quantum computational supremacy? arXiv: 1805.05224, 2018.

[106] Guang Hao Low and Isaac L. Chuang. Optimal Hamiltonian simulation by quantum signal processing. *Physical Review Letters*, 118(1):010501, 2017. arXiv: 1606.02685

[107] Andrew M. Childs, Dmitri Maslov, Yunseong Nam, Neil J. Ross, and Yuan Su. Toward the first quantum simulation with quantum speedup. arXiv: 1711.10980, 2017.

[108] Abraham Adrian Albert. Quadratic forms permitting composition. *Annals of Mathematics*, 43(1):161–177, 1942.

[109] Seth Lloyd. Universal quantum simulators. *Science*, 273(5278):1073–1078, 1996.

[110] Dominic W. Berry, Graeme Ahokas, Richard Cleve, and Barry C. Sanders. Efficient quantum algorithms for simulating sparse Hamiltonians. *Communications in Mathematical Physics*, 270(2):359–371, 2007. arXiv: quant-ph/0508139

[111] Jeongwan Haah, Matthew B. Hastings, Robin Kothari, and Guang Hao Low. Quantum algorithm for simulating real time evolution of lattice hamiltonians. arXiv: 1801.03922, 2018.

[112] Andrew M. Childs. On the relationship between continuous- and discrete-time quantum walk. *Communications in Mathematical Physics*, 294(2):581–603, 2010. arXiv: 0810.0312

[113] Dominic W. Berry and Andrew M. Childs. Black-box Hamiltonian simulation and unitary implementation. *Quantum Information and Computation*, 12(1&2):29–62, 2012. arXiv: 0910.4157

[114] Bei Zeng, Andrew Cross, and Isaac L. Chuang. Transversality versus universality for additive quantum codes. *IEEE Transactions on Information Theory*, 57:6272–6284, September 2011. arXiv: 0706.1382

[115] Michael Newman and Yaoyun Shi. Limitations on transversal computation through quantum homomorphic encryption. *Quantum Information and Computation*, 18(11& 12):0927–0948, 2018. arXiv: 1704.07798

[116] Austin G. Fowler, Simon J. Devitt, and Cody Jones. Surface code implementation of block code state distillation, January 2013. arXiv: 1301.7107

[117] Sergey Bravyi and Jeongwan Haah. Magic state distillation with low overhead. *Physical Review A*, 86:052329, 2012. arXiv: 1209.2426

[118] Adam Paetznick and Ben W. Reichardt. Universal fault-tolerant quantum computation with only transversal gates and error correction. *Physical Review Letters*, 111:090505, April 2013. arXiv: 1304.3709

[119] Hector Bombín. Gauge color codes: Optimal transversal gates and gauge fixing in topological stabilizer codes, August 2015. New Journal of Physics.

[120] Theodore J. Yoder, Ryuji Takagi, and Isaac L. Chuang. Universal fault-tolerant gates on concatenated stabilizer codes. *Physical Review X*, 6:031039, March 2016. arXiv: 1603.03948

[121] Theodore J. Yoder. Universal fault-tolerant quantum computation with Bacon-Shor codes. arXiv: 1705.01686, May 2017.

[122] Panos Aliferis, Daniel Gottesman, and John Preskill. Quantum accuracy threshold for concatenated distance-3 codes. *Quantum Information and Computation*, 6:97–165, 3 2006. arXiv: quant-ph/0504218

[123] Peter W. Shor. Fault-tolerant quantum computation. In *Proceedings of the 37th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 56–65, 1996.

[124] Ilya Dumer, Daniele Micciancio, and Madhu Sudan. Hardness of approximating the minimum distance of a linear code. *IEEE Transactions on Information Theory*, 49(1):22–37, January 2003.

[125] Daniel Gottesman and Lucy Liuxuan Zhang. Fibre bundle framework for unitary quantum fault tolerance. arXiv: 1309.7062, 2013.

[126] Michael J. Bremner, Richard Josza, and Dan J. Shepherd. Classical simulation of commuting quantum computations implies collapse of the polynomial hierarchy. *Proceedings of the Royal Society A*, 467, August 2010. arXiv: 1005.1407