Span Programs, Electrical Flows, and Beyond: New Approaches to Quantum Algorithms

Guoming Wang

Electrical Engineering and Computer Sciences
University of California at Berkeley

Technical Report No. UCB/EECS-2014-64
http://www.eecs.berkeley.edu/Pubs/TechRpts/2014/EECS-2014-64.html

May 13, 2014

Copyright © 2014, by the author(s).
All rights reserved.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission.

Span Programs, Electrical Flows, and Beyond: New Approaches to Quantum Algorithms

by

Guoming Wang

A dissertation submitted in partial satisfaction of the
requirements for the degree of
Doctor of Philosophy

in

Computer Science

in the

Graduate Division

of the

University of California, Berkeley

Committee in charge:

Professor Umesh Vazirani, Chair
Professor Satish Rao
Assistant Professor Hartmut Haeffner

Spring 2014

The dissertation of Guoming Wang, titled Span Programs, Electrical Flows, and Beyond: New Approaches to Quantum Algorithms, is approved:

Chair _____ Date _____

_____ Date _____

_____ Date _____

University of California, Berkeley

Span Programs, Electrical Flows, and Beyond: New Approaches to Quantum Algorithms

Copyright 2014
by
Guoming Wang

Abstract

Span Programs, Electrical Flows, and Beyond: New Approaches to Quantum Algorithms

by

Guoming Wang

Doctor of Philosophy in Computer Science

University of California, Berkeley

Professor Umesh Vazirani, Chair

Over the last decade, a large number of quantum algorithms have been discovered that outperform their classical counterparts. However, depending on the main techniques used, most of them fall into only three categories. The first category uses quantum Fourier transform to achieve super-polynomial speedup in group-theoretical problems. The second category uses amplitude amplification to achieve polynomial speedup in search problems. The third category simulates quantum many-body systems. Finding a new class of quantum algorithms has proven a challenging task.

This dissertation explores several new approaches to developing quantum algorithms. These approaches include span programs, electrical flows and nonsparse Hamiltonian simulation. We demonstrate their power by successfully applying them to some useful problems, including tree detection, effective resistance estimation and curve fitting. All of these algorithms are time-efficient, and some of them are proven to be (nearly) optimal.

Span program is a linear-algebraic computation model originally proposed to prove circuit lower bounds. Recently, it is found to be closely related to quantum query complexity. We develop a span-program-based quantum algorithm for the following variant of the tree containment problem. Let T be a fixed tree. Given the $n \times n$ adjacency matrix of a graph G, we need to decide whether G contains T as a subgraph, or G does not contain T as a minor, under the promise that one of these cases holds. We call this problem is the subgraph/not-a-minor problem for T. We show that this problem can be solved by a quantum algorithm with $O(n)$ query complexity and $\tilde{O}(n)$ time complexity. This query complexity is optimal, and this time complexity is tight up to poly-logarithmic factors.

Electrical network theory has many applications to the design and analysis of classical algorithms. Its connection to quantum computation, however, remains mostly unclear. We give two quantum algorithms for approximating the effective resistance between two given vertices in an electrical network. Both of them have time complexity polynomial in $\log n$, d, c, $1/\phi$ and $1/\varepsilon$,

where n is the number of vertices, d is the maximum degree of the vertices, c is the ratio of the largest to the smallest edge resistance, ϕ is the conductance of the network, and ε is the relative error. Our algorithms have exponentially better dependence on n than classical algorithms. Furthermore, we prove that the polynomial dependence on the inverse conductance is necessary. As a consequence, our algorithms cannot be greatly improved. Finally, as a by-product, our second algorithm also produces a quantum state encoding the electrical flow between two given vertices in a network, which might be of independent interest.

Our algorithms are developed by using quantum tools to analyze the algebraic properties of graph-related matrices. While the first one relies on inverting the Laplacian matrix, the second one relies on projecting onto the kernel of the weighted incidence matrix. It is hopeful that more quantum algorithms could be designed in similar way.

Curve fitting is the process of constructing a (simple continuous) curve that has the best fit to a series of data points. It is a common practice in many fields of science and engineering, because it can help us to understand the relationships among two or more variables, and to infer the values of a function where no data are available. We propose efficient quantum algorithms for estimating the best-fit parameters and the quality of least-square curve fitting. The running times of our algorithms are polynomial in $\log n$, d, κ, ν, χ, $1/\Phi$ and $1/\varepsilon$, where n is the number of data points to be fitted, d is the dimension of the feature vectors, κ is the condition number of the design matrix, ν and χ are some parameters reflecting the variances of the design matrix and response vector, Φ is the fit quality, and ε is the tolerable error. Our algorithms have exponentially better dependence on n than classical algorithms. They are designed by combining the techniques of phase estimation and density matrix exponentiation for nonsparse Hamiltonian simulation.

To my family

Contents

Contents **ii**

1 Introduction **1**
1.1 Background 1
1.2 Summary of Results 7

2 Preliminaries **10**
2.1 Notation 10
2.2 Quantum Information 11
2.3 Quantum Computation 16

3 Quantum Algorithm for Tree Detection **25**
3.1 Introduction 25
3.2 Span Program and Quantum Query Complexity 26
3.3 Span Program for Tree Detection 29
3.4 Time-Efficient Implementation 42
3.5 Open Problems 48

4 Electrical Flows and Quantum Algorithms **50**
4.1 Overview 50
4.2 Spectral Graph Theory 52
4.3 Electrical Flows and Effective Resistances 53
4.4 Our Model 54
4.5 A Simple Quantum Algorithm for Estimating Effective Resistances 55
4.6 A Faster Quantum Algorithm for Estimating Effective Resistances 59
4.7 Generating Electrical Flows as Quantum States 69
4.8 Lower Bound on the Complexity of Effective Resistance Estimation 72
4.9 Discussion 73

5 Quantum Algorithms for Curve Fitting **75**
5.1 Introduction 75
5.2 Least-Square Curve Fitting 77

5.3 Our Model 78
5.4 Density Matrix Exponentiation 79
5.5 Quantum Algorithms for Estimating the Best-Fit Parameters $\hat{\theta}$ 84
5.6 Quantum Algorithm for Estimating the Fit Quality Φ 96
5.7 Open Problems 97

Bibliography **98**

Acknowledgments

First and foremost, I am deeply indebted to my advisor, Umesh Vazirani, for his guidance, support, and encouragement throughout these years. From steering me in the right direction, to providing useful technical advice, to improving my writing and presenting, he has given me invaluable help without which this thesis cannot be finished. He has also taught me innumerable lessons on the workings of academic research in general. His taste of problems, his commitment to research, and his scientific curiosity and creativity were a great source of inspiration. All of these will benefit me a lot in my future. Besides research, Umesh has kindly helped me on various non-technical matters, and I am immensely grateful for that.

I would also like to thank my master's advisor, Mingsheng Ying, for his support and advice at the beginning of my scientific career, and for his constant encouragement all the time.

Berkeley has offered me an excellent environment to carry out my research. I would like to thank the theory faculty for the many courses and seminars that I was fortunate to attend. I am also grateful to Satish Rao, Hartmut Haeffner and John Kubiatowicz for serving on my quals and thesis committees and for providing helpful advice.

I am grateful to several people who have directly or indirectly assisted me in writing this thesis. Thanks to Ben Reichardt for inspiring discussions on span programs and quantum query complexity, and for giving useful feedback on my algorithm for tree detection. Thanks to Zeph Laundau for suggesting the geometric proof of Lemma 40. Thanks to Robin Kothari for suggesting the proof of Theorem 48.

During my thesis research, I enjoyed extended visits to NEC Labs in Princeton and Institute for Quantum Computing (IQC) in Waterloo. I would like to thank many individuals at those institutes for their hospitality: at NEC, Martin Röetteler, Dmitry Gavinsky, and Tsuyoshi Ito; and at IQC, Andrew Childs, Richard Cleve, David Gossett, Robin Kothari, Rajat Mittal, Ashwin Nayak, and John Watrous. Special thanks for Martin for his kind invitation and considerate arrangement. Also, special thanks to Dima and Tsuyoshi for the great collaboration on communication complexity. I enjoyed working with them, and I have learned a lot from them.

Many thanks to Umesh, Martin and Kubi for writing letters of recommendation for my application of postdoctoral position. Also, many thanks to Thomas Vidick for providing helpful career advice.

I would like to thank many members of quantum information community, in addition to those mentioned above, for having useful discussions. These include Scott Aaronson, Andris Ambainis, André Chailloux, Paul Christiano, Andrew Drucker, Runyao Duan, Yuan Feng, Sevag Gharibian, Zhengfeng Ji, Stephen Jordan, Yi-Kai Liu, Urmila Mahadev, Anupam Prakash, Seung Woo Shin,

Mario Szegedy, Falk Unger, and Shengyu Zhang.

Soda Hall has been a pleasant place to stay. For creating an environment of general camaraderie, I would like to thank my fellow theory students: Anand Bhaskar, Antonio Blanca, Siu Man Chan, Siu On Chan, James Cook, Anindya De, Rafael Frongillo, Henry Lin, Lorenzo Orecchia, Jonah Sherman, Isabelle Stanton, Piyush Srivastava, Madhur Tulsiani, Thomas Watson, and all the others.

It would not be possible for me to embark on this journey without the unwavering support and encouragement of my family. I especially want to thank my mother for working so hard to make my brother and I have a good life, and for being understanding and supportive all the time. I am also grateful to my wife Wenhua for her love and patience. To them, I dedicate this thesis.

Chapter 1

Introduction

1.1 Background

Quantum computation is the study of using quantum-mechanical phenomenon, such as superposition and entanglement, to perform data processing tasks. This area was first introduced by Manin [101] and Feynman [67] in the early 1980s, who noticed that simulating quantum many-body systems is inherently difficult for classical computers (due to the exponential number of parameters to describe these systems) and suggested that quantum computers would be better suited for this task. In 1985, Deutsch [59] took this idea further and described a *universal quantum computer* — an abstract machine that captures all of the power of quantum computation. He also gave the first example of a problem that could be solved faster by a quantum computer than by a classical computer. His work was followed by a steady sequence of advances [28, 60, 118], culminating in 1994 with Shor's discovery of efficient quantum algorithms for factoring integers and calculating discrete logarithms [117]. Shor's algorithm, like its predecessors, is based on the idea of using a *quantum Fourier transform* to find periodicity. It has been generalized to solve a variety of algebraic problems, such as hidden subgroup [14, 15, 63, 69, 72, 77, 91, 103], hidden shift [45, 54, 57, 76, 102], Pell's equation [74], unit group [75], and so on [42, 52, 58, 131]. All of these algorithms achieve super-polynomial speedup over the best known classical algorithms, and have close connections to *representation theory* [68].

Meanwhile, in 1996, Grover [71] made an equally striking discovery — a quantum algorithm that achieves quadratic speedup for the unstructured search problem. Namely, given oracle access to a database of N items, one of which being marked, Grover's algorithm can find the marked item using only $O\left(\sqrt{N}\right)$ queries. This algorithm was subsequently generalized to the framework of *amplitude amplification* [35]. Since the unstructured search problem is extremely basic, Grover's search has been applied to many problems, such as collision finding [36], graph connectivity [62], and so on [12].

Quantum walk [66, 130], the quantum analogue of of classical random walk, is a further generalization of amplitude amplification. Following previous work on spatial search [1, 11, 47, 48], Ambainis [8] gave an optimal quantum-walk-based algorithm for the element distinctness prob-

lem. His approach was subsequently generalized [46, 99, 125] and applied to other problems, such as triangle finding [100], matrix product verification [38] and group commutativity [98]. Like amplitude amplification, all of these algorithms achieve polynomial speedup for the search problems.

Returning to the original motivation for quantum computation, Lloyd [95] demonstrated that quantum computers can be programmed to simulate any *local* quantum system efficiently. His result was subsequently extended to much large classes of quantum systems [2, 4, 30, 39, 73, 84, 85, 107, 108, 133, 134]. In addition, several quantum algorithms have also been proposed to approximate the ground and thermal states for some classes of Hamiltonians [109, 126](although the problem of finding the ground state energies of local Hamiltonians is QMA-complete [3, 89] [1] and hence probably requires exponential time on a quantum computer in the worst case).

The above three categories of quantum algorithms will surely be useful if large-scale quantum computers can be built. But they also raise the question of how broadly useful quantum computers could be. Indeed, although we have a large number of quantum algorithms today, most of them are developed by only few techniques (such as quantum Fourier transform and amplitude amplification), and they solve problems with similar flavours. It has proven challenging to find a new class of quantum algorithms that (greatly) outperform their classical counterparts.

This dissertation investigates several new approaches to developing quantum algorithms. These approaches include *span programs*, *electrical flows* and *nonsparse Hamiltonian simulation*. We demonstrate their power by successfully applying them to some useful problems, including *tree detection*, *effective resistance estimation* and *curve fitting*. All of these algorithms are *time-efficient* [2], and some of them are proven to be (nearly) optimal.

Our work is inspired by many recent developments in quantum algorithms. So, before describing our results in more detail, it is helpful to briefly review these developments.

1.1.1 Recent Developments in Quantum Algorithms

Formula Evaluation, Span Programs, Learning Graphs and Electrical Flows

An AND-OR formula is a rooted tree in which the internal nodes correspond to AND and OR gates, and the leaves are numbered. To a formula ϕ of size n and a numbering of the leaves from 1 to n corresponds a function $f_\phi : \{0,1\}^n \to \{0,1\}$. This function is defined on input $x = x_1 x_2 \dots x_n \in \{0,1\}^n$ by placing bit x_j on the j-th leaf, for $j \in [n]$ [3], and evaluating the gates toward the root. Evaluating AND-OR formulas is an important problem with many applications. For example, it allows us to solve the decision version of a two-player game tree. Classically, a full binary AND-OR tree of size N can be evaluated with $\Theta\left(N^{0.754\dots}\right)$ queries and this is optimal [116, 119].

In 2007, Farhi, Goldstone and Gutman [64] showed that a full binary AND-OR tree of size N can be evaluated with $O\left(\sqrt{N}\right)$ quantum queries, but in an nonconventional continuous-query

[1] QMA is the set of decision problems satisfying the following conditions: (1) if the answer is YES, there is a polynomial-size quantum proof which convinces a polynomial-time quantum verifier of the fact with high probability; (2) if the answer is NO, every polynomial-size quantum state is rejected by the verifier with high probability. It can be viewed as the quantum analogue of NP.

[2] In contrast, many previous quantum algorithms are only *query-efficient*.

[3] Throughout this dissertation, we use $[n]$ to denote the set $\{1, 2, \dots, n\}$, for any positive integer n.

model. Several improvements followed soon. Ambainis et al. [10] translated this algorithm to the conventional discrete-query model and extended it to evaluating arbitrary Boolean formulas with $O(N^{1/2+o(1)})$ quantum queries.

Later, Reichardt and Špalek [115] discovered a far-reaching generalization of this result. Namely, the quantum algorithm was generalized to evaluating *span programs* [86]. Span program is an algebraic model of computation originally proposed to prove circuit lower bounds. Informally speaking, a span program consists of a target vector τ and a finite set of input vectors $v_1, v_2, \ldots, v_m$ from some inner-product space. Each v_j is associated with a condition $x_i = 0$ or $x_i = 1$ for some $i \in [n]$. On input $x = x_1 x_2 \ldots x_n \in \{0,1\}^n$, the span program evaluates to 1 if τ can be written as the linear combination of the v_j's whose the associated conditions are true on x. Otherwise, the span program evaluates to 0.

Here is a simple example of span program. Consider the following vectors from $\mathbb{C}^2$:

$$\tau = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \quad v_1 = \begin{pmatrix} 1 \\ 1 \end{pmatrix}, \quad v_2 = \begin{pmatrix} 1 \\ e^{i2\pi/3} \end{pmatrix}, \quad v_3 = \begin{pmatrix} 1 \\ e^{-i2\pi/3} \end{pmatrix}. \tag{1.1}$$

Vectors v_1, v_2 and v_3 are associated with conditions $x_1 = 1$, $x_2 = 1$ and $x_3 = 1$, respectively. Then, this span program evaluates to 1 on input $x = x_1 x_2 x_3 \in \{0,1\}^3$ if and only if at least two of x_1, x_2 and x_3 are 1. In other words, this span program computes the majority function $\mathrm{Maj}(x_1, x_2, x_3)$.

Reichardt and Špalek invented a complexity measure called *witness size* for span programs. This measure generalizes the formula size: a Boolean formula of size S can be transformed into a span program with witness size S. They showed that a span program with witness size S can be evaluated by a quantum algorithm with $O(S)$ query complexity. More remarkably, Reichardt [111, 113] also discovered that for any Boolean function f, the smallest witness size of any span program for f is actually within a constant factor of the (bounded-error) quantum query complexity of f! More precisely, the general adversary bound [82] is a semidefinite program (SDP) that lower-bounds the quantum query complexity of a function. Reichardt considered the dual of this SDP and found that the dual SDP gives the span program with smallest witness size [4]. Thus, the span program approach is optimal (in terms of query complexity) for any Boolean function.

Reichardt's discovery leads to a novel approach to designing quantum algorithms. Namely, in order to devise a query-efficient quantum algorithm for a problem, we only need to build a span program with small witness size for this problem. To date, span-program-based quantum algorithms have been developed for formula evaluation [112, 114, 115], matrix rank [19], *st*-connectivity [24], subgraph detection [24] and graph collision [70] [5]. These algorithms have optimal or improved query complexity over previous algorithms. Span programs have a rich mathematical structure, and their potential has not been fully explored.

For many problems, however, finding the optimal span program is very hard. In fact, it is equivalent to solving a SDP of exponential size [111]. To surmount this problem, Belovs [20] introduced the framework of *learning graph* to systematically construct span programs for Boolean

[4] This also implies that the general adversary bound is tight for any Boolean function.

[5] The span programs for these problems are mostly designed in an ad hoc fashion.

functions with small 1-certificates [6]. A learning graph is a directed acyclic graph with vertices being the subsets of $[n]$, as shown by Fig. 1.1. One may think of it as simulating the development of our knowledge on the input. Initially, we know nothing about the input, and it is represented by vertex $\varnothing$. When in vertex $S \subseteq [n]$ [7], the values of the variables in S have been learned. For any $j \in [n] \setminus S$, vertex S is connected to vertex $S \cup \{j\}$ by an arc. This can be interpreted as querying the value of variable x_j. In other words, this arc *loads* element j, and S is the set of *loaded elements*. For any $x \in f^{-1}(1)$, there exists a 1-certificate for x contained in some vertex of the learning

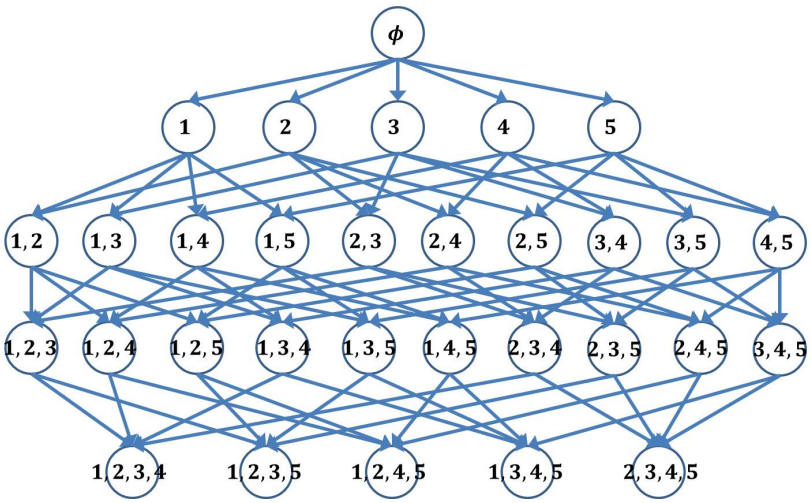

Figure 1.1: A learning graph.

graph. We call such vertices *accepting*. To check that $f(x) = 1$, we need a *loading procedure* that starts at vertex $\varnothing$ and ends at some accepting vertex S. This loading procedure can be randomized (i.e. at each step, it chooses the variable of load with some probability distribution). Thus, a loading procedure can be viewed as a *flow* from vertex $\varnothing$ (i.e. the only source) to the accepting vertices (i.e. the sinks) on the learning graph. There can be many such flows. Belovs's idea was to use the *minimum energy* of such flows to characterize the difficulty of computing $f(x)$. Here the energy of a flow $p : E \to \mathbb{R}$ is defined as $\sum_{e \in E} p^2(e)/w(e)$, where E is the set of arcs in the learning graph, and $w(e) > 0$ is the weight of arc $e \in E$. Intuitively, if there are many (qualitatively different) 1-certificates for x and they spread widely in the learning graph, then the optimal flow for x has small energy; otherwise, it has large energy. Alternatively speaking, the learning graph is associated with a span program, and this minimum energy is related to the witness size of that span program. Learning graphs have been used to develop query-efficient quantum algorithms for subgraph detection [92, 93, 136], associativity testing [93] and k-element distinctness [21, 23].

[6] Let $f : \{0,1\}^n \to \{0,1\}$ be a Boolean function. An assignment is a function $\sigma : S \to \{0,1\}$, where $S \subseteq [n]$. An assignment σ is called a b-certificate for f if any x consistent with σ (i.e. $x_j = \sigma(j)$ for any $j \in S$) is mapped to b by f, for $b \in \{0,1\}$.

[7] There can be multiple vertices corresponding to the same subset $S \subseteq [n]$.

One can regard the learning graph as an *electrical network* with edge resistance $1/w(e)$. Then, the optimal flow for $x \in f^{-1}(1)$ is exactly the unit *electrical flow* from vertex $\varnothing$ to the accepting vertices for x (pretending that they are glued together), and the optimal energy is just the *effective resistance* between these vertices. From this point of view, Belovs's work builds a connection between electrical network theory and quantum query complexity. Electrical network theory has found many applications in the design and analysis of classical algorithms (e.g. [55, 122]). But its connection to quantum computation remains mostly unclear (although there are a few results [20, 22, 25] in this direction). We will investigate this topic in Chapter 4.

We remark that most of the aforementioned span-program-based (or learning-graph-based) quantum algorithms are query-efficient, but not *time-efficient*. Although Reichardt [111, 113] has proposed a quantum-walk-based algorithm for evaluating any span program, the quantum walk in his algorithm might be difficult to implement using local gates. The time-efficient evaluation of span programs (or learning graphs) can be challenging.

Hamiltonian Simulation and Linear Equations

Solving large systems of linear equations is an important problem in virtually all fields of science and engineering. In this problem, we are given an $N \times N$ matrix A and an N-dimensional vector b, and need to find the vector x such that $Ax = b$. Many efficient classical algorithms have been developed for this problem, but all of them require $\mathrm{poly}(N)$ time. Anyway, it takes $\Omega(N)$ time to just write down the N-dimensional solution $x = A^{-1}b$.

In 2008, Harrow, Hassidim and Lloyd (HHL) [78] discovered a surprising quantum algorithm that allows to solve systems of linear equations in $\tilde{O}\left(\mathrm{polylog}(N) \cdot \kappa^2/\varepsilon\right)$ time [8], where κ is the condition number of A, and ε is the tolerable error — but in an unconventional sense. Specifically, suppose that A is sparse and efficiently row computable, i.e. A has $\mathrm{polylog}(N)$ nonzero entries per row and given a row index these entries can be computed in $\mathrm{polylog}(N)$ time. Also, suppose a quantum state proportional to $|b\rangle$ can be prepared in $\mathrm{polylog}(N)$ time. Then HHL's algorithm produces a quantum state $|\tilde{x}\rangle$ such that $\||\tilde{x}\rangle - |\bar{x}\rangle\| \leq \varepsilon$ in $\tilde{O}\left(\mathrm{polylog}(N) \cdot \kappa^2/\varepsilon\right)$ time, where $|\bar{x}\rangle$ is a normalized state proportional to $|x\rangle = A^{-1}|b\rangle$ [9]. This state is useful, because we can perform quantum measurements on it and learn certain properties of the solution $|x\rangle = A^{-1}|b\rangle$. This algorithm was subsequently applied to solve linear differential equations [29] and least-squares curve-fitting [132] (but only in the sparse case).

Recently, Ambainis [9] introduced a technique called *variable-time amplitude amplification* and used it to improve the complexity of HHL's algorithm to $\tilde{O}(\mathrm{polylog}(N) \cdot \kappa/\varepsilon^3)$. Further improvement nevertheless seems unlikely, as [78] showed that no quantum algorithm could solve matrix inversion [10] in $\kappa^{1-\delta} \cdot \mathrm{poly}(\log N, 1/\varepsilon)$ time for some constant $\delta > 0$, unless BQP=PSPACE

[8] Throughout this dissertation, we use the symbol $\tilde{O}$ to suppress poly-logarithmic factors. Namely, $\tilde{O}(f(n)) = O(f(n)(\log f(n))^b)$ for some constant b.

[9] In fact, HHL's algorithm can produce a quantum state approximately proportional to $f(A)|b\rangle$ for any easy-to-compute function f.

[10] Here we say that an algorithm solves matrix inversion if its input is a sparse matrix A specified by an oracle, and its output is the value of $\langle z|M|z\rangle$, where $M = |0\rangle\langle 0| \otimes I_{N/2}$ corresponds to measuring the first qubit and $|z\rangle$ is a

[11] Even so, matrix inversion appears much harder for classical computers, as [78] proved that no classical algorithm could solve matrix inversion in poly $(\kappa, \log N, 1/\varepsilon)$ time, unless BPP=BQP [12].

The basic idea of HHL's algorithm is as follows. Suppose A has the spectral decomposition $A = \sum_j \lambda_j |v_j\rangle\langle v_j|$. We can expand $|b\rangle$ in the eigenbasis of A. Namely, suppose $|b\rangle = \sum_j b_j |v_j\rangle$ for some coefficients b_j. Then the solution to $A|x\rangle = |b\rangle$ is given by

$$|x\rangle = \sum_j b_j \lambda_j^{-1} |v_j\rangle . \tag{1.2}$$

HHL's idea is to use the techniques for *sparse Hamiltonian simulation* to apply e^{iAt} to $|b\rangle$ for a superposition of different times t. This exponentiation of A, combined with phase estimation [87], allows us to decompose $|b_j\rangle$ in the eigenbasis of A and to find the corresponding eigenvalues λ_j. Informally, the state after this stage is close to $\sum_j b_j |v_j\rangle |\lambda_j\rangle$. Then, we only need to perform the linear map $|\lambda_j\rangle \to C\lambda_j^{-1} |\lambda_j\rangle$, where C is a normalizing constant. Although this is not a unitary operation, it can be approximately achieved by using a controlled-rotation and amplitude amplification. After it succeeds, we uncompute the $|\lambda_j\rangle$ register and are left with a state approximately proportional to $|x\rangle = \sum_j b_j \lambda_j^{-1} |v_j\rangle$.

As mentioned above, sparse Hamiltonian simulation plays a crucial role in HHL's algorithm. A Hamiltonian H is called *d-sparse* if each row of H contains at most d nonzero entries. Suppose such an H is given by an oracle for the positions and values of its nonzero entries. We want to simulate the unitary operation e^{iHt} by querying this oracle and using additional gates, for any given t. The first efficient algorithm for this problem was due to Aharonov and Ta-Shma [4]. Their idea is to use edge coloring to decompose H into a sum of Hamiltonians $\sum_{j=1}^{r} H_j$, where each H_j is easy to simulate. These terms are then recombined using the Lie-Trotter product formula [128], which states that

$$e^{iHt} \approx (e^{iH_1 t/n} e^{iH_2 t/n} \cdots e^{iH_r t/n})^n \tag{1.3}$$

for large n. This method was later improved using high-order product formulas and more efficient decompositions of the Hamiltonian [30, 50]. Recently, Berry et al. [32] gave a dramatically improved algorithm with $O(\tau \log(\tau/\varepsilon))$ query complexity and $O(\log N \cdot \tau \log(\tau/\varepsilon))$ time complexity, where $\tau = d^2 \|H\|_{\max} t$, and N is the dimension of H. Remarkably, this algorithm has only polylogarithmic dependence on $1/\varepsilon$. It is based on an efficient simulation of the continuous-query model by discrete quantum queries.

Nevertheless, the aforementioned methods only work for *sparse* Hamiltonian simulation. In contrast, efficient simulation of *nonsparse* Hamiltonians seems much harder, and there are fewer results [43, 49] in that direction. But in a recent paper, Lloyd, Mohseni and Rebentrost [96] introduced a novel technique called *density matrix exponentiation*, which allows to simulate the

normalized state proportional to $A^{-1}|0\rangle$.

[11] BQP is the class of decision problems that can be solved by a quantum computer in polynomial time with high probability. PSPACE is the class of decision problems that can be solved by a classical deterministic computer in polynomial space.

[12] BPP is the class of decision problems that can be solved by a classical probabilistic computer in polynomial time with high probability.

time evolution of any *positive semidefinite* (but not necessarily sparse) Hamiltonian. It is based on the observation that

$$\left\| \mathrm{tr}_1 \left(e^{iS\Delta t} \left(\rho \otimes \sigma \right) e^{-iS\Delta t} \right) - e^{i\rho\Delta t} \sigma e^{-i\rho\Delta t} \right\|_{\mathrm{tr}} = O\left((\Delta t)^2 \right), \tag{1.4}$$

where S is the swap operator, and ρ, σ are arbitrary density matrices. Namely, for any small Δt, we can approximate the unitary operation $e^{i\rho\Delta t}$ by the following procedure: (1) append a state ρ; (2) perform the unitary operation $e^{iS\Delta t}$ on the joint system; (3) trace out the appended system. This observation, combined with the Lie-Trotter product formula, allows us to simulate the unitary operation $e^{i\rho t}$ by consuming multiple copies of the state ρ. Furthermore, by running phase estimation on the operator $e^{i\rho t}$ starting with the state ρ, we can analyze the eigenvalues and eigenvectors of ρ. They call this phenomenon *quantum self-analysis* [13]. We will utilize their results to design efficient quantum algorithms in Chapter 5.

1.2 Summary of Results

This dissertation presents *time-efficient* quantum algorithms for several problems, including tree detection, effective resistance estimation and curve fitting. In developing them, we not only give fast solution to some practical problems, but also gain new insights to the power of quantum computation. Although our algorithms are not related to representation theory, most of them are significantly faster than their classical counterparts, and some of them are proven to be (nearly) optimal. Furthermore, some of them might be used as a subroutine to help solving other problems as well.

In the remainder of the introduction, we will give a brief overview of the three main chapters in this dissertation.

In **Chapter 3**, we describe a *span-program-based* quantum algorithm for tree detection. *Subgraph detection* is an important problem with numerous applications to biochemistry, circuit design and software engineering. Our algorithm solves the following variant of the tree containment problem. Let T be a fixed tree. Given the $n \times n$ adjacency matrix of a graph G, we need to decide whether G contains T as a subgraph, or G does not contain T as a *minor* [14], under the promise that one of these cases holds. We call this problem the *subgraph/not-a-minor* problem for T. We show that this problem can be solved by a quantum algorithm with $O(n)$ query complexity and $\tilde{O}(n)$ time complexity. This query complexity is optimal, and this time complexity is tight up to poly-logarithmic factors. To develop this algorithm, we first build an optimal span program for tree detection, and then give a time-efficient implementation of this span program using quantum walks.

[13] Of course, we can perform quantum state tomography to determine ρ completely. But quantum self-analysis turns out to be more efficient than this naive method.

[14] We say that a graph H is a minor of a graph G if H can be obtained from G by deleting and contracting edges of G, and removing isolated vertices. Here, contracting an edge (u,v) involves replacing u and v by a new vertex and connecting this vertex to the original neighbours of u and v.

In **Chapter 4**, we study quantum algorithms for *electrical flows* and *effective resistances*. *Electrical network theory* has many applications to the design and analysis of classical algorithms. Examples include the relation between effective resistances and commute times of random walks [40], the usage of effective resistances for graph sparsification [122], and the usage of electrical flows for approximating maximum flows [55, 94, 97]. Classically, to compute electrical flows and effective resistances, one need to solve a Laplacian linear system, and the currently best algorithms take $\tilde{O}(m)$ time, where m is the number of edges in the network.

We give two quantum algorithms for approximating the effective resistance between two given vertices in an electrical network. Both of them have time complexity polynomial in $\log n$, d, c, $1/\phi$ and $1/\varepsilon$, where n is the number of vertices, d is the maximum degree of the vertices, c is the ratio of the largest to the smallest edge resistance, ϕ is the conductance of the network, and ε is the relative error. Our algorithms run very fast when d and c are small and ϕ is large. In contrast, it is unknown whether classical algorithms can solve this case very fast. Furthermore, we prove that the polynomial dependence on the inverse conductance (i.e. $1/\phi$) is necessary. As a consequence, our algorithms cannot be greatly improved. Finally, as a by-product, our second algorithm also produces a quantum state encoding the electrical flow between two given vertices in a network, which might be of independent interest.

Our algorithms are based on using quantum tools to analyze the algebraic properties of graph-related matrices. While one of them relies on inverting the Laplacian matrix, the other relies on projecting onto the kernel of the weighted incidence matrix. It is hopeful that more quantum algorithms could be devised in similar way.

In **Chapter 5**, we study quantum algorithms for *curve fitting*. Curve fitting, also known as *regression analysis* in statistics, is the process of constructing a (simple continuous) curve that has the best fit to a series of data points. It is a common practice in many fields of science and engineering, because it can help us to understand the relationships among two or more variables, and to infer the values of a function where no data are available. Classically, in order to find the best-fit curve in the standard *least-square* approach, one needs to solve a linear system and it takes $\mathrm{poly}(n,d)$ time, where n is the number of points to be fitted, and d is the dimension of feature vectors (or equivalently, the number of parameters to be determined).

We propose efficient quantum algorithms for estimating the best-fit parameters and the quality of least-square curve fitting. The running times of our algorithms are polynomial in $\log n$, d, κ, ν, χ, $1/\Phi$ and $1/\varepsilon$, where n and d are defined as above, κ is the condition number of the design matrix, ν and χ are some parameters reflecting the variances of the design matrix and response vector, Φ is the fit quality [15], and ε is the tolerable error. Our algorithms run very fast when the given data are normal, in the sense that the design matrix is far from being singular, and the rows of design matrix and response vector do not vary too much in their norms. In contrast, it is unknown whether classical algorithms can solve this case very fast. Furthermore, different from previous quantum algorithms for this task, our algorithms work no matter the design matrix is sparse or not, and

[15] The time complexity of the algorithm for estimating Φ does not depend on $1/\Phi$.

they determine the best-fit curve completely [16]. They are designed by combining the techniques of phase estimation and *density matrix exponentiation* for nonsparse Hamiltonian simulation.

[16] Previous quantum algorithms [132] only work in the sparse case, and only produce a quantum state encoding this curve.

Chapter 2

Preliminaries

This chapter describes the notation that is used throughout this dissertation, and collects some important definitions and results in quantum information and computation. For more detailed background, the reader is encouraged to consult [89, 105, 110, 129].

2.1 Notation

Let $\mathbb{R}$ and $\mathbb{C}$ be the field of real and complex numbers, respectively. For any $z \in \mathbb{R}$, let $\operatorname{sgn}(z) := 1$ if $z \geq 0$, and -1 otherwise. In addition, let $\lfloor z \rfloor$ be the largest integer that is not greater than z, and let $\lceil z \rceil$ be the smallest integer that is not less than z. Let $\mathbb{N}$ be the set of positive integers. For any $n \in \mathbb{N}$, let $[n] := \{1, 2, \ldots, n\}$.

For any vector ψ, let $\|\psi\|$ be the Euclidean norm of ψ. In addition, let $\bar{\psi}$ be the normalized version of ψ, i.e. $\bar{\psi} := \psi / \|\psi\|$. Furthermore, let $|\psi\rangle := \psi$. Namely, $|\psi\rangle$ and ψ are essentially the same thing, but we use $|\psi\rangle$ to denote the (unnormalized) quantum state corresponding to ψ, not its classical description.

For any matrix A, we say that A is *s-sparse* if each row of A contains at most s nonzero entries. Furthermore, let $\mathcal{C}(A)$ be the column space of A, and let $\operatorname{Ker}(A)$ be the kernel of A. Then, let Π_A be the projection onto $\mathcal{C}(A)$, and let $\operatorname{Ref}_A := 2\Pi_A - I$ be the reflection about $\mathcal{C}(A)$.

The *spectral norm* of a matrix A, denoted by $\|A\|$, is the largest singular value of A. The *trace norm* of A, denoted by $\|A\|_{\mathrm{tr}}$, is the sum of the singular values of A. The *condition number* of a matrix A, denoted by $\kappa(A)$, is the ratio of A's largest singular value to its smallest singular value. For a matrix A with full column rank, the *Moore-Penrose pseudoinverse* of A, denoted by A^+, is defined as $(A^\dagger A)^{-1} A^\dagger$.

For an $N \times N$ Hermitian matrix A, we use $\lambda_i(A)$ to denote the i-th smallest eigenvalue of A (counted with multiplicity), for $i \in [N]$. For any two Hermitian matrices A and B, we use $A \succcurlyeq B$ to denote that $A - B$ is positive semidefinite.

A *Hilbert space* is a complex vector space with inner product. A linear operator A on a Hilbert space is *unitary* if $A^\dagger A = A A^\dagger = I$, where I is the identity operator. A linear operator A on a Hilbert space is a *projection* if $A^2 = A$.

Finally, as stated before, we use the symbol $\tilde{O}$ to suppress poly-logarithmic factors. Namely, $\tilde{O}(f(n)) = O(f(n)(\log f(n))^b)$ for some constant b.

2.2 Quantum Information

2.2.1 Basics of Quantum Mechanics

Quantum mechanics postulates that any *isolated* physical system is associated with a Hilbert space known as the *state space*, and the system is completely described by a *state vector*, which is a unit vector in its state space. The dimension of a system is defined as the dimension of its state space. The simplest quantum system is a *qubit*, which has a two-dimensional state space. Its state is described by

$$|\varphi\rangle = a|0\rangle + b|1\rangle, \tag{2.1}$$

where a and b are complex numbers satisfying $|a|^2 + |b|^2 = 1$, and $\{|0\rangle, |1\rangle\}$ is an orthonormal basis for the state space.

The evolution of a *closed* quantum system is described by a *unitary transformation*. Namely, the state $|\varphi_1\rangle$ of the system at time t_1 is related to the state $|\varphi_2\rangle$ of the system at time t_2 by a unitary operator U which depends only on the time t_1 and t_2,

$$|\varphi_2\rangle = U|\varphi_1\rangle. \tag{2.2}$$

A quantum measurement with k possible outcomes is described by a collection $\{E_1, E_2, \ldots, E_k\}$ of k *measurement operators*. These operators act on the state space of the system being measured, and each one of them corresponds to one possible outcome. If the state of the quantum system if $|\varphi\rangle$ before the measurement, then the probability that outcome i occurs is

$$p_i = \langle\varphi| E_i^\dagger E_i |\varphi\rangle, \tag{2.3}$$

and the corresponding post-measurement state is

$$|\varphi_i\rangle = \frac{E_i|\varphi\rangle}{\sqrt{\langle\varphi| E_i^\dagger E_i |\varphi\rangle}}. \tag{2.4}$$

The measurement operator satisfy the *completeness equation*:

$$\sum_{i=1}^{k} E_i^\dagger E_i = I. \tag{2.5}$$

This ensures that the probabilities of all possible outcomes sum to one:

$$1 = \sum_{i=1}^{k} p_i = \langle\varphi| E_i^\dagger E_i |\varphi\rangle. \tag{2.6}$$

A *projective* measurement with k possible outcomes is described by a collection $\{\Pi_1, \Pi_2, \ldots, \Pi_k\}$ of k *projections*. Namely, these operators satisfy $\Pi_i^2 = \Pi_i$ for $i \in [k]$, and $\sum_{i=1}^k \Pi_i = I$.

The state space of a composite system is the tensor product of the state spaces of the component systems. Namely, for a joint system AB, its state space is $\mathcal{H} = \mathcal{H}_A \otimes \mathcal{H}_B$, where $\mathcal{H}_A$, $\mathcal{H}_B$ are the state spaces of system A and B, respectively. The state of AB can be any unit vector in $\mathcal{H}$. In the simple case, the state of AB is just the tensor product of the state of A and the state of B. That is, the state of AB is $|\phi_A\rangle |\phi_B\rangle$, where $|\phi_A\rangle$ is the state of A and $|\phi_B\rangle$ is the state of B. But there are many other states of AB that cannot be written in this form. For example, consider a two-qubit system in the state

$$|\Phi\rangle = \frac{|00\rangle + |11\rangle}{\sqrt{2}}. \tag{2.7}$$

This state cannot be written as $|\phi_1\rangle |\phi_2\rangle$ for any two-dimensional states $|\phi_1\rangle$ and $|\phi_2\rangle$. So we say that $|\Phi\rangle$ is an *entangled* state. *Entanglement* is one of the most mysterious aspects of quantum mechanics, and plays a crucial role in quantum information processing. Hence, it has been extensively studied during the past decade (although it is not the focus of this dissertation). For more background on entanglement, we refer the read to [81].

The *density matrix* language provides a convenient means for describing quantum systems whose states are not completely known. Specifically, suppose a quantum system is in state $|\psi_i\rangle$ with probability p_i, for $i \in [k]$. We shall call $\{p_i, |\psi_i\rangle\}_{i\in[k]}$ an *ensemble of pure states*. The density matrix for the system is defined as

$$\rho := \sum_{i=1}^{k} p_i |\psi_i\rangle\langle\psi_i| . \tag{2.8}$$

Note that ρ is positive definite (i.e. $\rho \succcurlyeq 0$) and has trace 1 (i.e. $\mathrm{tr}(\rho) = 1$). We shall say that the system is in the *mixed state* ρ.

Suppose we perform a unitary operation U on a system in the state ρ. Then its state after this operation is $U\rho U^\dagger$. Meanwhile, suppose we perform a measurement $\{E_1, E_2, \ldots, E_m\}$ on a system in the state ρ. Then, the probability of obtaining outcome i is given by

$$p_i = \mathrm{tr}\left(E_i \rho E_i^\dagger\right) \tag{2.9}$$

and the corresponding post-measurement state is

$$\rho_i = \frac{E_i \rho E_i^\dagger}{\mathrm{tr}\left(E_i \rho E_i^\dagger\right)}. \tag{2.10}$$

Moreover, suppose a joint system AB is in the state ρ_{AB}. Then the *reduced density matrix* (or reduced state) of system A is

$$\rho_A := \mathrm{tr}_B(\rho_{AB}), \tag{2.11}$$

where $\mathrm{tr}_B()$ is the *partial trace* over system B defined as

$$\mathrm{tr}_B(|a_1\rangle\langle a_2| \otimes |b_1\rangle\langle b_2|) := |a_1\rangle\langle a_2| \,\mathrm{tr}(|b_1\rangle\langle b_2|). \tag{2.12}$$

We have mentioned that the evolution of a *closed* quantum system is described by a unitary operation. But in the real world, many physical systems suffer from unwanted interactions with the environment. To characterize the dynamics of such *open* quantum systems, we use the formalism of *quantum operations* [105], which has the form

$$\mathcal{E}(\rho) := \sum_i E_i \rho E_i^\dagger, \tag{2.13}$$

where the E_i's are called the *Kraus operators* of the operation $\mathcal{E}$ [1] and satisfy the completeness equation

$$\sum_i E_i^\dagger E_i = I. \tag{2.14}$$

In particular, unitary operations and quantum measurements are just special classes of quantum operations. Specifically, for a unitary operation U, it can be viewed as the operation

$$U(\rho) := U \rho U^\dagger. \tag{2.15}$$

For a quantum measurement $\{F_1, F_2, \ldots, F_k\}$, it can be viewed as the operation

$$\mathcal{F}(\rho) := \sum_{i=1}^{k} |i\rangle\langle i| \otimes F_i \rho F_i^\dagger \tag{2.16}$$

(followed by a projective measurement on the first register). Namely, the output is a classical-quantum (c-q) state. While the first register stores the measurement outcome, the second register stores the corresponding post-measurement state.

2.2.2 Distance Measures for Quantum States and Operations

Many quantum states are difficult to prepare exactly. Similarly, many quantum operations are too expensive to implement perfectly. Usually we are content with an approximation of these states or operations. To evaluate the quality of the approximations, we need some distance measures for quantum states and operations.

The *trace distance* between quantum states ρ_1 and ρ_2 is defined as

$$D(\rho, \sigma) := \frac{1}{2} \|\rho - \sigma\|_{\mathrm{tr}}. \tag{2.17}$$

This is a popular distance measure for quantum states. It has many nice properties, such as:

Lemma 1 (Triangle inequality)**.** *For any quantum states* ρ*,* σ *and* τ*, we have*

$$D(\rho, \sigma) \leq D(\rho, \tau) + D(\tau, \sigma). \tag{2.18}$$

[1] The E_i's can be rectangular matrices. In this case, the output has a different dimension from the input.

Lemma 2 (Contractivity under quantum operations [105])**.** *For any quantum operation $\mathcal{E}$ and states ρ, σ, we have*

$$D(\mathcal{E}(\rho_1), \mathcal{E}(\rho_2)) \leq D(\rho_1, \rho_2). \tag{2.19}$$

In particular, if $\mathcal{E} = U$ is unitary, then we have

$$D(U\rho U^\dagger, U\sigma U^\dagger) = D(\rho, \sigma). \tag{2.20}$$

Since the partial trace is a special kind of quantum operation, Lemma 2 implies:

Corollary 3. *For any quantum states ρ_{AB} and σ_{AB} on the joint system AB, we have*

$$D(\rho_A, \sigma_A) \leq D(\rho_{AB}, \sigma_{AB}), \tag{2.21}$$

where ρ_A, σ_A are the reduced states of ρ_{AB}, σ_{AB} on the system A, respectively.

Corollary 3 makes sense, since objects should become *less* distinguishable when only partial information is available.

Finally, if two pure states are close with respect to the Euclidean distance, then they are also close with respect to the trace distance:

Lemma 4. *For any pure states $|\varphi_1\rangle$ and $|\varphi_2\rangle$, we have*

$$D(|\varphi_1\rangle\langle\varphi_1|, |\varphi_2\rangle\langle\varphi_2|) \leq \||\varphi_1\rangle - |\varphi_2\rangle\|. \tag{2.22}$$

Proof. Suppose $\langle\varphi_1|\varphi_2\rangle = e^{i\gamma} \cdot \cos\theta$ for some $\gamma \in [0, 2\pi]$ and $\theta \in [0, \pi/2]$. Let $|\varphi_2'\rangle = e^{-i\gamma}|\varphi_2\rangle$. Then we have $\langle\varphi_1|\varphi_2'\rangle = \cos\theta \geq 0$ and

$$\left\||\varphi_1\rangle - |\varphi_2'\rangle\right\| \leq \||\varphi_1\rangle - |\varphi_2\rangle\|. \tag{2.23}$$

So it is sufficient to prove that

$$D(|\varphi_1\rangle\langle\varphi_1|, |\varphi_2\rangle\langle\varphi_2|) = D(|\varphi_1\rangle\langle\varphi_1|, |\varphi_2'\rangle\langle\varphi_2'|) \leq \left\||\varphi_1\rangle - |\varphi_2'\rangle\right\|. \tag{2.24}$$

Now let $|\varphi_\pm\rangle$ be the normalized state proportional to $|\varphi_1\rangle \pm |\varphi_2'\rangle$ respectively. Then, we can write $|\varphi_1\rangle$ and $|\varphi_2'\rangle$ as

$$\begin{aligned} |\varphi_1\rangle &= \cos(\theta/2)\,|\varphi_+\rangle + \sin(\theta/2)\,|\varphi_-\rangle, \\ |\varphi_2'\rangle &= \cos(\theta/2)\,|\varphi_+\rangle - \sin(\theta/2)\,|\varphi_-\rangle, \end{aligned} \tag{2.25}$$

where $|\varphi_+\rangle$ and $|\varphi_-\rangle$ are orthonormal. Then, by a direct calculation, one obtains

$$D(|\varphi_1\rangle\langle\varphi_1|, |\varphi_2'\rangle\langle\varphi_2'|) = \sin\theta \leq 2\sin(\theta/2) = \left\||\varphi_1\rangle - |\varphi_2'\rangle\right\|, \tag{2.26}$$

as desired. □

Now let us turn to the distance measure for quantum operations. For any quantum operations $\mathcal{E}$ and $\mathcal{F}$, let

$$D(\mathcal{E},\mathcal{F}) := \max_{\rho} D((\mathcal{E}\otimes I)(\rho),(\mathcal{F}\otimes I)(\rho)), \tag{2.27}$$

where I is the identity operation on an ancilla system (of arbitrary dimension), and ρ can be any state on the joint system. This is a well-defined function (if the original system has dimension d, then the maximum can be achieved with an ancilla system of dimension d^4 and some state ρ on the joint system). It satisfies many nice properties, such as:

Lemma 5 (Robustness under system extension)**.** *For any quantum operations $\mathcal{E}$ and $\mathcal{F}$, we have*

$$D(\mathcal{E}\otimes I,\mathcal{F}\otimes I) = D(\mathcal{E},\mathcal{F}). \tag{2.28}$$

Lemma 6 (Triangle inequality)**.** *For any quantum operations $\mathcal{E}_1$, $\mathcal{E}_2$ and $\mathcal{E}_3$, we have*

$$D(\mathcal{E}_1,\mathcal{E}_3) \le D(\mathcal{E}_1,\mathcal{E}_2) + D(\mathcal{E}_2,\mathcal{E}_3). \tag{2.29}$$

Lemma 7 (Composability)**.** *For any quantum operations $\mathcal{E}$, $\mathcal{E}'$, $\mathcal{F}$ and $\mathcal{F}'$, we have*

$$D(\mathcal{E}\circ\mathcal{F},\mathcal{E}'\circ\mathcal{F}') \le D(\mathcal{E},\mathcal{E}') + D(\mathcal{F},\mathcal{F}') \tag{2.30}$$

Finally, if two unitary operations are close with respect to the metric induced by the spectral norm, then they are also close with respect to the distance measure D:

Lemma 8. *For any unitary operations U_1 and U_2, we have*

$$D(U_1,U_2) \le \|U_1 - U_2\|. \tag{2.31}$$

Proof. By the convexity of trace norm (i.e. $\|pA+(1-p)B\|_{\mathrm{tr}} \le p\|A\|_{\mathrm{tr}} + (1-p)\|B\|_{\mathrm{tr}}$), we can see that the optimal ρ in Eq.(2.27) can be a pure state. Namely, there exists a pure state $|\psi\rangle$ on some extended system such that

$$D(U_1,U_2) = D((U_1\otimes I)(|\psi\rangle\langle\psi|),(U_2\otimes I)(|\psi\rangle\langle\psi|)). \tag{2.32}$$

Then, by Lemma 4, we have

$$\begin{array}{rcl} D((U_1\otimes I)(|\psi\rangle\langle\psi|),(U_2\otimes I)(|\psi\rangle\langle\psi|)) & \le & \|(U_1\otimes I)|\psi\rangle - (U_2\otimes I)|\psi\rangle\| \\ & = & \|((U_1-U_2)\otimes I)|\psi\rangle\|. \end{array} \tag{2.33}$$

Let $W = U_1 - U_2$. Then we have

$$\|(W\otimes I)|\psi\rangle\|^2 = \mathrm{tr}\left(W\sigma W^\dagger\right) \le \|W\|^2, \tag{2.34}$$

where σ is the reduced state of $|\psi\rangle$ on the first system. Combining Eq.(2.33) and Eq.(2.34) yields the desired result. □

For any quantum operation $\mathcal{E}$ and integer k, let $\mathcal{E}^k$ be the k-repetition of $\mathcal{E}$, i.e. $\mathcal{E}^k := \mathcal{E} \circ \mathcal{E} \circ \cdots \circ \mathcal{E}$, where the number of $\mathcal{E}$'s on the right-hand side is k. Then Lemma 7 implies that

$$D(\mathcal{E}^k, \mathcal{F}^k) \leq k \cdot D(\mathcal{E}, \mathcal{F}). \tag{2.35}$$

From now on, when we say that "a state ρ is prepared to accuracy ε", it means that we actually prepare a state ρ' satisfying $D(\rho, \rho') \leq \varepsilon$. Similarly, when we say that "an operation $\mathcal{E}$ is implemented to accuracy ε", it means that we actually implement an operation $\mathcal{E}'$ satisfying $D(\mathcal{E}, \mathcal{E}') \leq \varepsilon$. Lemma 7 implies that in order to implement the operation $\mathcal{E} = \mathcal{E}_1 \circ \mathcal{E}_2 \circ \cdots \circ \mathcal{E}_k$ to accuracy ε (where $\mathcal{E}_1$, $\mathcal{E}_2$, ..., $\mathcal{E}_k$ are arbitrary quantum operations), it is sufficient to implement each $\mathcal{E}_i$ to accuracy ε/k.

2.3 Quantum Computation

2.3.1 Quantum Circuits

Quantum algorithms are usually described by a *quantum circuit* that acts on some input qubits and terminates with a measurement. A quantum circuit is a sequence of *quantum gates* (usually chosen from a finite set) composed together. Each gate is a unitary operation that acts on a constant number of qubits. For example, the Hadamard gate H, defined as

$$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}. \tag{2.36}$$

is an extremely useful one-qubit gate in quantum computation. Other important one-qubit gates include the Pauli gates X, Y, Z and the $\pi/8$ gate T, which are given by

$$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \quad Y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}, \quad Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \tag{2.37}$$

and

$$T = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\frac{\pi}{4}} \end{pmatrix}. \tag{2.38}$$

Furthermore, the CNOT gate, defined as

$$\mathrm{CNOT} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}, \tag{2.39}$$

is arguably the most important two-qubit gate in quantum computation. The size of a circuit is defined as the number of gates in this circuit, and the time complexity of a quantum algorithm is defined as the size of the circuit describing it.

We say that a set of gates is *(exactly) universal* if any unitary operation (on any number of qubits) can be implemented perfectly by a sequence of gates from this set. It is known that the set of all one- and two-qubit gates is universal in this sense [61]. However, as far as the efficiency is concerned, most unitary operations on n qubits can only be realized by an exponentially large circuit [90]. In general, we are content with circuits that give good approximations of our desired unitary operations. Therefore, we say that a set of gates is *universal* if any unitary operation on a fixed number of qubits can be approximated to accuracy $\varepsilon > 0$ using a sequence of $\mathrm{polylog}(1/\varepsilon)$ gates from this set. It turns out that there are finite sets of gates that are universal. For instance, the gate set $\{H, T, \mathrm{CNOT}\}$ is universal in this sense.

One may wonder whether some universal gate sets are better than others. It turns out that the answer is essentially no: for any two universal gate sets S_1 and S_2, any circuit of T gates from S_1 can be implemented to accuracy $\varepsilon > 0$ by a circuit of $T \cdot \mathrm{polylog}(T/\varepsilon)$ gates from S_2, and there is an efficient classical algorithm for finding this circuit. This is a consequence of the Solovay-Kitaev theorem [79, 88, 120].

Many quantum algorithms, such as Grover's search [71] and Simon's algorithm [118], are described in the *quantum query model*, as shown in Figure 2.1. In this model, the input x is given by an *oracle* O_x which is a unitary operation. Beginning in a fixed state (usually $|00\ldots0\rangle$), a quantum algorithm alternates input-independent unitary operations and oracle queries. Finally, it measures the first few qubits to obtain the output. The query complexity of this algorithm is defined as the number of oracle queries it has made, and the time complexity of this algorithm is defined as its query complexity plus the number of elementary gates needed to implement the input-independent unitary operations [2].

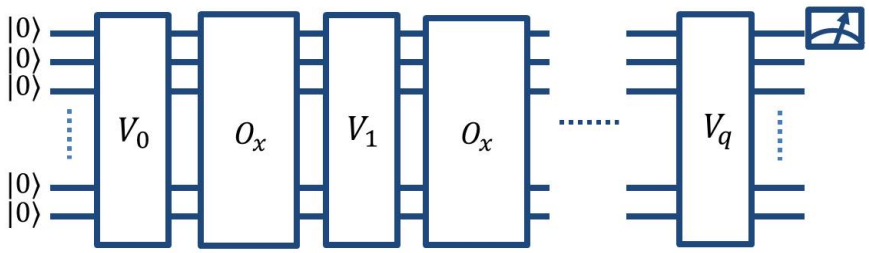

Figure 2.1: The quantum query model. Beginning in a fixed state, a quantum algorithm alternates input-independent unitary operations and oracle queries. For computing a Boolean function $f : \{0,1\}^n \to \{0,1\}$ on input $x = x_1 x_2 \ldots x_n$, the oracle O_x is the unitary operation that maps $|j,a,z\rangle$ to $|j,a \oplus x_j, z\rangle$, where $j \in [n]$, $a \in \{0,1\}$ and $|z\rangle$ describes the state of the working space.

Given any function f, there could be many quantum algorithms that compute it. The bounded-error quantum query complexity of f, denoted by $Q(f)$, is the minimum query complexity of any quantum algorithm that calculates f with error probability at most $1/3$. Quantum query complexity

[2] In other words, we assume that each oracle query takes a unit time. This is a common convention used in many literatures.

has been extensively studied during the past years, not only because it is easier to study than time complexity, but also because it provides some insights to the power of quantum computation. For example, the quantum component of Shor's algorithm, period finding, is a quantum query algorithm. Many tools, such as the polynomial method [18], adversary methods [6, 7, 17, 82, 121, 135], quantum walks [99, 125] and learning graphs [20], have been developed to prove lower and upper bounds on quantum query complexity.

This dissertation presents several quantum algorithms that are both query-efficient and time-efficient. Namely, these algorithms only make a few calls to the oracle, and they can efficiently process the information obtained from the oracle (i.e. the unitary operations V_j's in Fig. 2.1 can be implemented using a polynomial number of local gates).

2.3.2 Useful Techniques

In the remainder of this chapter, we present some important algorithmic tools for quantum computation, including phase estimation, amplitude amplification, amplitude estimation and quantum walk. They will become the basic building blocks of our algorithms.

Phase Estimation

Phase estimation was introduced by Kitaev [87], who used it to give an alternative derivation of Shor's factoring algorithm. To date, it has found numerous applications in quantum computation and information. This algorithm solves the following problem. Suppose a unitary operator U has an eigenvector $|v\rangle$ [3] with eigenvalue $e^{i\theta}$, where the value of θ is unknown. Phase estimation allows us to estimate θ, given access to a copy of the state $|v\rangle$ and a procedure to implement the controlled-U^{2^j} for suitable non-negative integers j.

Phase estimation uses two registers. The first register contains n qubits initially in the state $|00\ldots0\rangle$. Here t is a parameter depending on the accuracy we want to achieve. The second register begins with the state $|v\rangle$. This algorithm consists of two stages. The first stage begins by applying the Hadamard transform on the first register, followed by application of controlled-U on the second register, with U raised to successive power of two. The second stage is to apply the *inverse quantum Fourier transform* on the first register. Fig. 2.2 demonstrates the quantum circuit for phase estimation.

Let $T = 2^n$. By a direct calculation, one can see that the final state of this procedure is given by

$$\left|\Psi_{pe}\right\rangle := \frac{1}{T}\sum_{j=0}^{T-1} \frac{e^{iT\theta}-1}{e^{i(\theta-2\pi j/T)}-1} \left|j\right\rangle \left|v\right\rangle . \tag{2.40}$$

If $\theta = 2\pi k/T$ for some $k \in \{0,1,\ldots,T-1\}$, then the quantum Fourier transform will single out that phase in binary expansion. Namely, the final state is $|k\rangle|v\rangle$. Otherwise, there will be a probability distribution clustered around the correct phase. For any $\varepsilon > 0$, by setting $T = \Theta(1/\varepsilon)$ (i.e. $n =$

[3] Without loss of generality, we assume thta $|v\rangle$ is normalized.

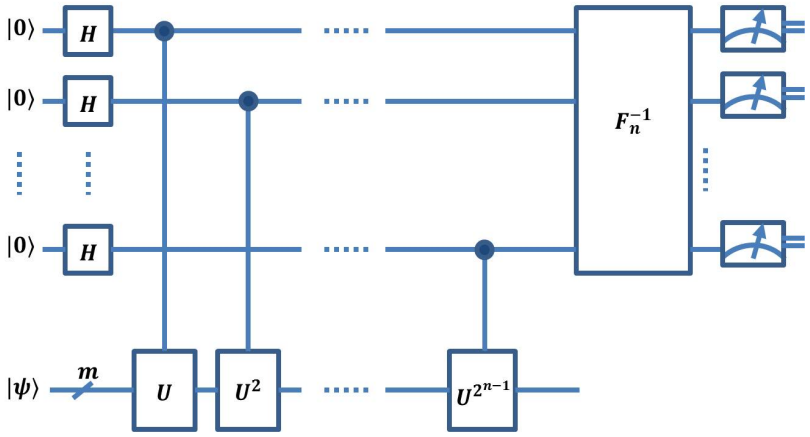

Figure 2.2: Quantum circuit for phase estimation.

$\Theta(\log(1/\varepsilon))$), we can make sure that

$$\left\langle \Psi_{pe} \right| \Pi(\theta-\varepsilon, \theta+\varepsilon) \left| \Psi_{pe} \right\rangle \geq 2/3, \tag{2.41}$$

where

$$\Pi(\alpha,\beta) := \Big(\sum_{j=\lceil \alpha T/(2\pi) \rceil}^{\lfloor \beta T/(2\pi) \rfloor} |j\rangle\langle j| \Big) \otimes I. \tag{2.42}$$

In other words, if we measure the first register of the final state in the standard basis, obtain the outcome k', and set $\theta' = 2\pi k'/T$, then we have

$$|\theta - \theta'| \leq \varepsilon \tag{2.43}$$

with probability at least $2/3$.

The above procedure has constant success probability. As pointed out by [104], we can concatenate r phase estimation circuits, and take the median of the r results. Then the failure probability will drop exponentially in r. Thus, to raise the success probability to $1-\delta$ for some small $\delta > 0$, we need to set $r = O(\log(1/\delta))$. To summarize, we obtain:

Lemma 9 (Phase Estimation [87, 104])**.** *Suppose U is a unitary operation and $|\phi\rangle$ is an eigenstate of U with eigenvalue $e^{i\theta}$ for some $\theta \in [0, 2\pi)$. Let ε, $\delta > 0$. Then there exists a quantum algorithm $\mathcal{A}$ that uses a copy of $|\phi\rangle$, $O(\log(1/\delta)/\varepsilon)$ controlled applications of U and $\mathrm{polylog}(1/(\varepsilon\delta))$ additional elementary gates, and produces an estimate θ' of θ such that $|\theta - \theta'| \leq \varepsilon$ with probability at least $1-\delta$.*

To avoid confusion between ε and δ, we will call ε the *precision (or accuracy)* of phase estimation, and call δ the *error rate* of phase estimation. Since the complexity of phase estimation is only logarithmic in $1/\delta$, we will assume that phase estimation never fails throughout this dissertation.

That is, we assume that phase estimation always outputs a θ' satisfying $|\theta - \theta'| \leq \varepsilon$. Although this is not really true, taking the error rate δ into account only increases the complexities of our algorithms by some poly-logarithmic factors (which will be ignored).

Amplitude Amplification

Amplitude amplification [35] was a generalization of Grover's search. It solves the following problem. Consider a Boolean function $f : X \to \{0,1\}$ that partitions the set X between its *good* and *bad* elements, where $x \in X$ is good if $f(x) = 1$ and bad otherwise. Suppose we have a quantum procedure $\mathcal{A}$ such that

$$\mathcal{A}|0\rangle = \sqrt{p}\,|\psi_1\rangle + \sqrt{1-p}\,|\psi_2\rangle, \tag{2.44}$$

where $|\psi_1\rangle = \sum_{x \in f^{-1}(1)} \alpha_x |x\rangle$ and $|\psi_2\rangle = \sum_{x \in f^{-1}(0)} \beta_x |x\rangle$ are normalized, and $p > 0$. Then, if we measure $\mathcal{A}|0\rangle$ in the basis $\{|x\rangle : x \in X\}$, then with probability p we find a good element $x \in f^{-1}(1)$. Classically, in order to raise this probability to $\Omega(1)$, we need $O(1/p)$ repetitions of this procedure. Surprisingly, amplitude amplification allows us to achieve this goal by repeating $\mathcal{A}$ and $\mathcal{A}^{-1}$ only $O\left(1/\sqrt{p}\right)$ times. In fact, it even preserves the two vectors $|\psi_1\rangle$ and $|\psi_2\rangle$ and only changes their amplitudes. Thus, it enables us to obtain the state $|\psi_1\rangle$ with constant probability.

The amplification process is realized by repeatedly applying the following unitary operation on the state $\mathcal{A}|0\rangle$:

$$Q = -\mathcal{A}U_0\mathcal{A}^{-1}U_f, \tag{2.45}$$

where $U_f : |x\rangle \to (-1)^{f(x)}|x\rangle$, and $U_0 = I - 2|0\rangle\langle 0|$. Geometrically, Q can be viewed as the composition of two reflections, one about span $\left(|x\rangle : x \in f^{-1}(0)\right)$, and another about $\mathcal{A}|0\rangle$. Let $\theta \in (0, \pi/2]$ be such that

$$\sin\theta = \sqrt{p}. \tag{2.46}$$

Then Q can be viewed as a rotation of angle 2θ in the two-dimensional space spanned by $\{|\psi_1\rangle, |\psi_0\rangle\}$, as shown in Fig. 2.3.

Therefore, we have

$$Q^k\mathcal{A}|0\rangle = \sin((2k+1)\theta)\,|\psi_1\rangle + \cos((2k+1)\theta)\,|\psi_0\rangle. \tag{2.47}$$

By setting $k \approx \frac{\pi}{4\sqrt{p}}$, we can make the amplitude of $|\psi_1\rangle$ close to 1. Then, measuring this state in the basis $\{|x\rangle : x \in X\}$ would yield a good element $x \in f^{-1}(1)$ with high probability.

Note that the above procedure assumes that we know on the value of p ahead of time. In fact, even if the value of p is unknown, we can use the technique of *exponential searching* [35] to amplify the amplitude of $|\psi_1\rangle$ to close to 1 by repeating $\mathcal{A}$ and $\mathcal{A}^{-1}$ only $O\left(1/\sqrt{p}\right)$ times. The cost is that we need to append some "junk" state (which stores the random numbers and other auxiliary information). Namely, the final state would be

$$\sqrt{q}\,|\psi_1\rangle\,|\phi_1\rangle + \sqrt{1-q}\,|\psi_0\rangle\,|\phi_0\rangle, \tag{2.48}$$

where $q \geq 2/3$, and $|\phi_1\rangle$, $|\phi_0\rangle$ are normalized states. To summarize, we have:

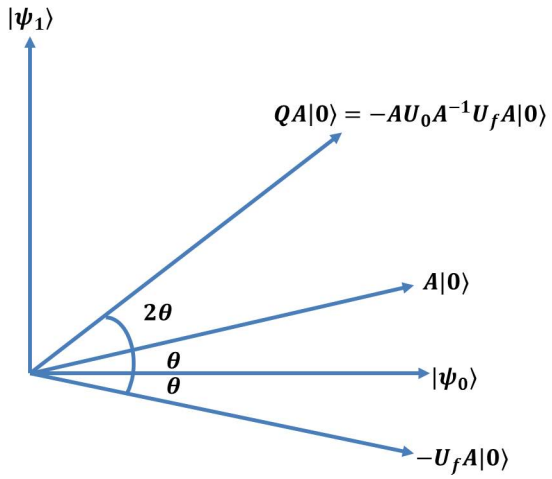

Figure 2.3: Geometrical illustration of amplitude amplification. The effect of Q is to rotate the original vector by an angle 2θ.

Lemma 10 (Amplitude Amplification [35])**.** *Suppose $\mathcal{A}$ is a quantum algorithm such that $\mathcal{A}|0\rangle = \sqrt{p}|1\rangle|\psi_1\rangle + \sqrt{1-p}|0\rangle|\psi_0\rangle$ where $p > 0$ and $|\psi_1\rangle$, $|\psi_0\rangle$ are some normalized states. Then there exists a quantum algorithm $\mathcal{A}'$ such that $\mathcal{A}'$ uses $O\left(1/\sqrt{p}\right)$ applications of $\mathcal{A}$ and $\mathcal{A}^{-1}$, and $\mathcal{A}'|0\rangle = \sqrt{q}|1\rangle|\psi_1\rangle|\phi_1\rangle + \sqrt{1-q}|0\rangle|\psi_0\rangle|\phi_0\rangle$ where $q \geq 2/3$ and $|\phi_1\rangle$, $|\phi_0\rangle$ are some normalized states.*

Lemma 10 implies that:

Corollary 11. *Suppose $\mathcal{A}$ is a quantum algorithm such that $\mathcal{A}|0\rangle = \sqrt{p}|1\rangle|\psi_1\rangle + \sqrt{1-p}|0\rangle|\psi_0\rangle$ where $p > 0$ and $|\psi_1\rangle$, $|\psi_0\rangle$ are some normalized states. Let $\delta > 0$. Then there exists a quantum algorithm $\mathcal{A}'$ that uses $O\left(\log(1/\delta)/\sqrt{p}\right)$ applications of $\mathcal{A}$ and $\mathcal{A}^{-1}$, and $\mathcal{A}'|0\rangle = \sqrt{q}|1\rangle|\psi_1\rangle|\varphi_1\rangle + \sqrt{1-q}|0\rangle|\psi_0\rangle|\varphi_0\rangle$ where $q \geq 1-\delta$ and $|\varphi_1\rangle$, $|\varphi_0\rangle$ are some normalized states.*

Proof. We run $k = O(\log(1/\delta))$ instances of the algorithm in Lemma 10 in parallel. Then by Lemma 10, we obtain the state

$$\left(\sqrt{r}|1\rangle|\psi_1\rangle|\phi_1\rangle + \sqrt{1-r}|0\rangle|\psi_0\rangle|\phi_0\rangle\right)^{\otimes k} = \sum_{\mathbf{i}\in\{0,1\}^k} \sqrt{r^{|\mathbf{i}|}(1-r)^{k-|\mathbf{i}|}}\,|i_1\rangle|\psi_{i_1}\rangle|\phi_{i_1}\rangle\ldots|i_k\rangle|\psi_{i_k}\rangle|\phi_{i_k}\rangle, \tag{2.49}$$

where $r \geq 2/3$, $\mathbf{i} = (i_1,\ldots,i_k)$, and $|\mathbf{i}| = \sum_{j=1}^k i_j$ is the Hamming weight of $\mathbf{i}$. Note that on the right-hand side of this equation, there exists only one term that does not contain $|\psi_1\rangle$ (in any position), which is the one corresponding to $\mathbf{i} = (0,\ldots,0)$, and its amplitude is $\sqrt{(1-r)^k} \leq \sqrt{\delta}$ by our choice of k. Now we perform on the state the following unitary operation: On the state $|i_1\rangle|\psi_{i_1}\rangle|\phi_{i_1}\rangle\ldots|i_k\rangle|\psi_{i_k}\rangle|\phi_{i_k}\rangle$, it finds the smallest j such that $i_j = 1$, and then, unless such j does not exist, it swaps $|i_1\rangle$ with $|i_j\rangle$, and swaps $|\psi_{i_1}\rangle$ with $|\psi_{i_j}\rangle$. Then, for each term except the one corresponding to $\mathbf{i} = (0,\ldots,0)$, the first two registers after this operation will be in the state $|1\rangle|\psi_1\rangle$.

Thus, the whole state after this operation can be written as $\sqrt{q}\,|1\rangle\,|\psi_1\rangle\,|\varphi_1\rangle + \sqrt{1-q}\,|0\rangle\,|\psi_0\rangle\,|\varphi_0\rangle$ for some $q \geq 1-\delta$ and normalized states $|\varphi_1\rangle$, $|\varphi_0\rangle$ (which are the states of the third to the last registers, conditioned on the first register being 1, 0 respectively). □

Amplitude Estimation

Amplitude estimation [35] is closely related to amplitude amplification. It solves the following problem. Recall the quantum procedure $\mathcal{A}$ satisfying Eq.(2.44). Amplitude estimation allows us to approximate p quadratically faster than classical methods. It is based on the observation that

$$\mathcal{A}\,|0\rangle = \frac{-i}{\sqrt{2}}\left(e^{i\theta}\,|\Psi_+\rangle - e^{-i\theta}\,|\Psi_-\rangle\right), \tag{2.50}$$

where $\theta = \arcsin\sqrt{p}$, and $|\Psi_\pm\rangle$ are the (normalized) eigenvectors of $Q = -\mathcal{A}U_0\mathcal{A}^{-1}U_f$ with eigenvalues $e^{\pm i2\theta}$. This suggests that we can estimate θ by running phase estimation on the unitary operator Q starting with the state $\mathcal{A}\,|0\rangle$, and then infer p from θ. Indeed, this is exactly how amplitude estimation works. By a direct calculation, one can obtain:

Lemma 12 (Amplitude Estimation [35], original version)**.** *Suppose $\mathcal{A}$ is a quantum algorithm such that $\mathcal{A}\,|0\rangle = \sqrt{p}\,|1\rangle\,|\psi_1\rangle + \sqrt{1-p}\,|0\rangle\,|\psi_0\rangle$ where $p \in (0,1)$ is unknown, and $|\psi_1\rangle$, $|\psi_0\rangle$ are some normalized states. Let M be any power of 2. Then there exists a quantum algorithm $\mathcal{A}'$ that uses $O(M)$ applications of $\mathcal{A}$ and $\mathcal{A}^{-1}$, and outputs p' ($0 \leq p' \leq 1$) such that*

$$|p'-p| \leq 2\pi\frac{\sqrt{p(1-p)}}{M} + \frac{\pi^2}{M^2} \tag{2.51}$$

with probability at least $8/\pi^2$.

It follows immediately from Lemma 12 that:

Corollary 13 (Amplitude Estimation, multiplicative version)**.** *Suppose $\mathcal{A}$ is a quantum algorithm such that $\mathcal{A}\,|0\rangle = \sqrt{p}\,|1\rangle\,|\psi_1\rangle + \sqrt{1-p}\,|0\rangle\,|\psi_0\rangle$ where $p \in (0,1)$ is unknown, and $|\psi_1\rangle$, $|\psi_0\rangle$ are some normalized states. Let $\varepsilon > 0$. Then there exists a quantum algorithm $\mathcal{A}'$ that uses $O\left(1/(\varepsilon\sqrt{p})\right)$ applications of $\mathcal{A}$ and $\mathcal{A}^{-1}$, and $\mathcal{A}'$ produces $p' \in (0,1)$ such that $|p-p'| \leq \varepsilon p$ with probability at least $2/3$.*

Corollary 14 (Amplitude Estimation, additive version)**.** *Suppose $\mathcal{A}$ is a quantum algorithm such that $\mathcal{A}\,|0\rangle = \sqrt{p}\,|1\rangle\,|\psi_1\rangle + \sqrt{1-p}\,|0\rangle\,|\psi_0\rangle$ where $p \in (0,1)$ is unknown, and $|\psi_1\rangle$, $|\psi_0\rangle$ are some normalized states. Let $\varepsilon > 0$. Then there exists a quantum algorithm $\mathcal{A}'$ that uses $O(1/\varepsilon)$ applications of $\mathcal{A}$ and $\mathcal{A}^{-1}$, and $\mathcal{A}'$ produces $p' \in (0,1)$ such that $|p-p'| \leq \varepsilon$ with probability at least $2/3$.*

Quantum Walks

Quantum walk [66, 130] is the quantum analogue of classical random walk. Based on the work of Ambainis [8] and Szegedy [125], Magniez, Nayak, Roland and Santha [99] proposed a general approach to quantize classical Markov chains and developed a formalism of quantum-walk-based search algorithms. Here we briefly review their results.

Let $P = (p_{x,y})$ be the transition matrix of any irreducible Markov chain on a finite space X. Let $P^* = (p^*_{x,y})$ be the time-reversed Markov chain of P. That is, we have $\pi_x p_{x,y} = \pi_y p^*_{y,x}$, where $\pi = (\pi_x)$ is the (unique) stationary distribution of P.

Let $\mathcal{A} = \mathrm{span}\,(|x\rangle|p_x\rangle : x \in X)$ and $\mathcal{B} = \mathrm{span}\left(\left|p^*_y\right\rangle|y\rangle : y \in X\right)$ be subspaces of $\mathcal{H} = \mathbb{C}^{X\times X}$, where

$$\begin{aligned} |p_x\rangle &= \textstyle\sum_{y\in X} \sqrt{p_{x,y}}\,|y\rangle\,, \\ \left|p^*_y\right\rangle &= \textstyle\sum_{x\in X} \sqrt{p^*_{y,x}}\,|x\rangle\,. \end{aligned} \tag{2.52}$$

Definition 15 (Quantum walk)**.** *Let* $\mathrm{Ref}(\mathcal{A})$ *and* $\mathrm{Ref}(\mathcal{B})$ *be the reflections about* $\mathcal{A}$ *and* $\mathcal{B}$*, respectively. The unitary operation* $W(P)$ *defined on* $\mathcal{H}$ *by*

$$W(P) = \mathrm{Ref}(\mathcal{B}) \cdot \mathrm{Ref}(\mathcal{A}) \tag{2.53}$$

is called the quantum walk based on the classical chain P*.*

Magniez et al.'s search algorithm is similar to amplitude amplification, in the sense that it also works by alternating two reflections. But it approximately implements one of the reflections by running phase estimation on the quantum walk operator $W(P)$. Formally, they proved that:

Lemma 16 ([99])**.** *Let* $\delta > 0$ *be the eigenvalue gap of a reversible, ergodic Markov chain* P *on a finite space* X*. Let* $M \subseteq X$ *be the set of marked elements, and let* $\varepsilon > 0$ *be a lower bound on the probability that an element chosen from the stationary distribution of* P *is marked whenever* M *is non-empty. Then, there is a quantum algorithm that with high probability, determines if* M *is empty or finds an element of* M*, with cost of order* $S + \frac{1}{\sqrt{\varepsilon}}\left(\frac{1}{\sqrt{\delta}}U + C\right)$*, where*

- S *is the cost of constructing* $\sum_{x\in X}\sqrt{\pi_x}\,|x\rangle|0\rangle$ *from* $|0\rangle|0\rangle$*,*
- U *is the cost of realizing any of the unitary operations*

$$\begin{aligned} |x\rangle|0\rangle &\to |x\rangle|p_x\rangle\,, \\ |0\rangle|y\rangle &\to \left|p^*_y\right\rangle|y\rangle \end{aligned} \tag{2.54}$$

  and their inverses, where $|p_x\rangle$ *and* $\left|p^*_y\right\rangle$ *are defined as in Eq.(2.52).*

- C *is the cost of realizing the conditional phase flip*

$$|x\rangle|y\rangle \to \begin{cases} -|x\rangle|y\rangle\,, & \text{if } x \in M; \\ |x\rangle|y\rangle\,, & \text{otherwise.} \end{cases} \tag{2.55}$$

The cost in Lemma 16 can be queries, time or space. Note that if we use classical random walks to search an element of M, the cost would be $S' + \frac{1}{\varepsilon}(\frac{1}{\delta}U' + C')$, where S', U' and C' are the classical counterpart of S, U and C. Therefore, quantum walks could provide a nearly quadratic speedup over Markov-chain-based algorithms (provided that the update cost dominates the total cost).

In this dissertation, however, we will *not* use quantum walks to do searching. Instead, we are mainly interested in the eigenvalues and eigenvectors of the quantum walk operator $W(P)$, and will run phase estimation on this operator to achieve certain goals (e.g. estimating effective resistances). For our work, Szegedy's spectral lemma is very important:

Lemma 17 (Spectral lemma [125])**.** *Let A and B be complex matrices such that they have the same number of rows and each of them has orthonormal columns. Let $D(A,B) = A^\dagger B$, and let $U(A,B) = \mathrm{Ref}_B \cdot \mathrm{Ref}_A$. Then all the singular values of $D(A,B)$ are at most 1. Let $\cos\theta_1$, $\cos\theta_2$, ..., $\cos\theta_l$ be the singular values of $D(A,B)$ that lie in the open interval $(0,1)$ counted with multiplicity. Then the following is a complete list of the eigenvalues of $U(A,B)$:*

1. *The $+1$ eigenspace of $U(A,B)$ is $(\mathcal{C}(A) \cap \mathcal{C}(B)) \oplus \left(\mathcal{C}(A)^\perp \cap \mathcal{C}(B)^\perp\right)$;*
2. *The -1 eigenspace of $U(A,B)$ is $\left(\mathcal{C}(A) \cap \mathcal{C}(B)^\perp\right) \oplus \left(\mathcal{C}(A)^\perp \cap \mathcal{C}(B)\right)$;*
3. *The other eigenvalues of $U(A,B)$ are $e^{2i\theta_1}, e^{-2i\theta_1}, e^{2i\theta_2}, e^{-2i\theta_2}, \ldots, e^{2i\theta_l}, e^{-2i\theta_l}$ counted with multiplicity.*

This lemma builds a connection between the singular values of $D(A,B)$ and the eigenvalues of $U(A,B)$ (which is actually more general than $W(P)$). It is significant, because it allows us to map a (possibly rectangular) non-unitary matrix $D(A,B)$ to a unitary matrix $U(A,B)$, and hence we can study the properties of $D(A,B)$ by studying the properties of $U(A,B)$ (which can be achieved by using many quantum tools such as phase estimation)! This fact is crucial for the algorithm in Chapter 3 for detecting trees, and the algorithm in Section 4.6 for estimating effective resistances.

Chapter 3

Quantum Algorithm for Tree Detection

3.1 Introduction

Given two graphs G and H, where G has more vertices than H, it is natural to ask whether G contains H as a subgraph. This problem, known as the *subgraph isomorphism* problem, has numerous applications in cheminformatics [16], circuit design [106] and software engineering [80]. If G and H are both given as input, then this problem is NP-complete. So it is unlikely to be solvable in polynomial time. However, if H is fixed and only G is given as input, then this problem, usually called the *H-containment* problem, can be solved efficiently. Specifically, if H contains k vertices, then the H-containment problem can be solved in $O\left(n^k\right)$ classical time, where n is the number of vertices in G. In fact, by exploiting H's structure cleverly, we can usually do much better. For example, if H is a tree, then the H-containment problem can be solved in $O(n^2)$ classical time [5] (assuming G is given by the $n \times n$ adjacency matrix).

Recently, there has been rising interest in developing fast quantum algorithms for the subgraph containment problem. In particular, the problem of triangle finding has received the most attention, perhaps due to its simplicity and its application to boolean matrix multiplication. Magniez, Santha and Szegedy [100] first gave two quantum algorithms for this problem, one based on Grover's search and the other based on quantum walk. They achieved $\tilde{O}\left(n^{13/10}\right)$ quantum query complexity for this problem. Later, Belovs [20] used learning graphs to improve the quantum query complexity of triangle finding to $O\left(n^{35/27}\right)$. His result was subsequently improved to $O\left(n^{9/7}\right)$ by Lee, Magniez and Santha [93]. This result was later recovered by Jeffery, Kothari and Magniez using nested quantum walks [83].

There has been also study on the quantum complexity of detecting other subgraphs. Childs and Kothari [51] studied the quantum query complexity of detecting paths, claws, cycles and bipartite subgraphs, etc. Later, Belovs and Reichardt [24] showed that detecting paths and subdivided stars can be done in $O(n)$ quantum queries and $\tilde{O}(n)$ quantum time. Meanwhile, Lee, Magniez, Santha [92] and Zhu [136] gave some upper bounds on the quantum query complexity of detecting arbitrary subgraphs. Their results were subsequently recovered by Jeffery, Kothari and Magniez using nested quantum walks [83]. It is worth mentioning that most of the aforementioned quantum

algorithms for subgraph detection are only query-efficient but not time-efficient.

In this chapter, we present a time-efficient span-program-based quantum algorithm for the following variant of tree containment problem:

Definition 18 (Subgraph/not-a-minor Problem)**.** *Let $T = (V_T, E_T)$ be a fixed tree. Given the $n \times n$ adjacency matrix of a graph $G = (V_G, E_G)$, we need to decide whether G contains T as a subgraph, or G does not contain T as a* minor*, under the promise that one of these cases holds. This problem is called the subgraph/not-a-minor problem for T.*

We show that this problem can be solved by a bounded-error quantum algorithm with $O(n)$ query complexity and $\tilde{O}(n)$ time complexity. Meanwhile, by a reduction from the unstructured search, one can show that this problem has $\Omega(n)$ quantum query complexity (see Proposition 4 of [24]). Therefore, our algorithm has optimal query complexity and nearly-optimal time complexity (tight up to poly-logarithmic factors).

Our work is closely related to the span program for undirected st-connectivity [24]. That span program works as follows. In order to test whether s and t are connected in an undirected graph $G = (V, E)$, we build a span program with target vector $|s\rangle - |t\rangle$ and input vectors $|u\rangle - |v\rangle$ for any $u, v \in V$. The input vector $|u\rangle - |v\rangle$ is available if and only if $(u, v) \in E$. Then the target vector lies in the span of the available input vectors if and only if there is a path connecting s and t in G. In other words, the basic idea of this span program is to *run a flow* from s to t in G.

We utilize the span program for undirected st-connectivity as follows. Suppose a tree T has root r and leaves f_1, f_2, ..., f_k. Then, in order to test whether G contains T as a subgraph, we check whether G contains k paths such that: (1) the j-th path resembles the path from r to f_j in T; (2) these k paths overlap in certain way so that their union looks like T. To accomplish this goal, we introduce a technique named *"parallel flows"*. Namely, we run k flows in parallel such that the j-th flow corresponds to the path from r to f_j, and let these flows *interfere* somehow to ensure that they meet the overlapping constraints. From an algebraic point of view, our span program is the "direct sum" of k span programs for undirected st-connectivity, but these k span-programs are also correlated somehow so that their solutions (i.e. the st-paths) overlap in desired way. This parallel-flow technique might be useful somewhere else (see Section 3.5).

For analyzing the witness size of our span program, we also prove a theorem (i.e. Claim 26) about the structure of graphs that do not contain certain tree as a minor, which might be of independent interest.

3.2 Span Program and Quantum Query Complexity

Span program is a linear-algebraic model of computation defined as follows:

Definition 19 (Span program [86])**.** *A span program $\mathcal{P}$ is a* 6*-tuple $(n, d, |\tau\rangle, \{|v_j\rangle : j \in [m]\}, I_{free}, \{I_{i,b} : i \in [n], b \in \{0,1\}\})$, where $|\tau\rangle \in \mathbb{R}^d$, $|v_j\rangle \in \mathbb{R}^d$ for any $j \in [m]$, and $I_{free} \cup \left(\cup_{i=1}^{n} I_{i,x_i}\right) = I \triangleq [m]$. $|\tau\rangle$ is called is the target vector, and each $|v_j\rangle$ is called an input vector. For any $j \in I_{free}$, we*

say that $|v_j\rangle$ is a free input vector; for any $j \in I_{i,b}$ for some $i \in [n]$ and $b \in \{0,1\}$, we say that $|v_j\rangle$ is labelled by (i,b).

To $\mathcal{P}$ corresponds a boolean function $f_{\mathcal{P}} : \{0,1\}^n \to \{0,1\}$ defined as follows: for $x = x_1, x_2, \ldots x_n \in \{0,1\}^n$,

$$f_{\mathcal{P}}(x) = \begin{cases} 1, & \text{if } \tau \in \text{span}\left(|v_j\rangle : j \in I_{free} \cup \left(\cup_{i=1}^n I_{i,x_i}\right)\right), \\ 0, & \text{otherwise.} \end{cases} \tag{3.1}$$

Namely, on input x, only the $|v_j\rangle$'s with $j \in I_{free} \cup \left(\cup_{i=1}^n I_{i,x_i}\right)$ are available, and $f_P(x) = 1$ if and only if the target vector lies in the span of the available input vectors.

For convenience, we say that $\mathcal{P}$ *accepts* x if $f_{\mathcal{P}}(x) = 1$, or *rejects* x if $f_{\mathcal{P}}(x) = 0$.

The complexity of a span program is measured by its *witness size* defined as follows:

Definition 20 (Witness size [111])**.** *Let $\mathcal{P} = (n, d, |\tau\rangle, \{|v_j\rangle : j \in [m]\}, I_{free}, \{I_{i,b} : i \in [n], b \in \{0,1\}\})$ be a span program. Let $I = I_{free} \cup \left(\cup_{i=1}^n \cup_{b\in\{0,1\}} I_{i,b}\right)$ and let $A = \sum_{j\in I} |v_j\rangle\langle j|$. Then, for any $x \in \{0,1\}^n$, let $I(x) = I_{free} \cup \left(\cup_{j=1}^n I_{j,x_j}\right)$, $\bar{I}(x) = \cup_{j=1}^n I_{j,x_j}$. Then, let $\Pi(x) = \sum_{j\in I(x)} |j\rangle\langle j|$, $\bar{\Pi}(x) = \sum_{j\in\bar{I}(x)} |j\rangle\langle j|$. The witness size of $\mathcal{P}$ on x, denoted by* $\text{wsize}(\mathcal{P}, x)$*, is defined as follows:*

- *If $f_P(x) = 1$, then $|\tau\rangle \in \mathcal{C}(A(\Pi(x)))$, so there exists $|w\rangle \in \mathbb{R}^m$ satisfying $A\Pi(x)|w\rangle = |\tau\rangle$. Any such $|w\rangle$ is a (positive) witness for x, and its size is defined as $\|\bar{\Pi}(x)|w\rangle\|^2$. Then* $\text{wsize}(P, x)$ *is defined as the minimal size among all such witnesses.*
- *If $f_P(x) = 0$, then $|\tau\rangle \notin \mathcal{C}(A(\Pi(x)))$, so there exists $|w'\rangle \in \mathbb{R}^d$ satisfying $\langle\tau|w'\rangle = 1$ and $\Pi(x)A^\dagger|w'\rangle = 0$. Any such $|w'\rangle$ is a (negative) witness for x, and its size is defined as $\|A^\dagger|w'\rangle\|^2$. Then* $\text{wsize}(P, x)$ *is defined as the minimal size among all such witnesses.*

For any $\mathcal{D} \subseteq \{0,1\}^n$ and $b \in \{0,1\}$, let

$$\text{wsize}_b(\mathcal{P}, \mathcal{D}) = \max_{x\in\mathcal{D}: f_P(x)=b} \text{wsize}(P, x). \tag{3.2}$$

Then the witness size of $\mathcal{P}$ over domain $\mathcal{D}$ is defined as

$$\text{wsize}(\mathcal{P}, \mathcal{D}) = \sqrt{\text{wsize}_0(P, \mathcal{D})\text{wsize}_1(P, \mathcal{D})}. \tag{3.3}$$

Surprisingly, for any (partial) boolean function, its least span program witness size is within a constant factor of its bounded-error quantum query complexity:

Theorem 21 ([111, 113])**.** *For any function $f : \mathcal{D} \to \{0,1\}$ where $\mathcal{D} \subseteq \{0,1\}^n$, let $Q(f)$ be the bounded-error quantum query complexity of f. Then*

$$Q(f) = \Theta\left(\inf_{\mathcal{P}: f_{\mathcal{P}|\mathcal{D}} = f} \text{wsize}(\mathcal{P}, \mathcal{D})\right), \tag{3.4}$$

where the infimum is over span programs $\mathcal{P}$ that compute a function agreeing with f on $\mathcal{D}$. Moreover, this infimum is achieved.

In particular, a span program with small witness size can be converted into a quantum algorithm with small query complexity:

Corollary 22. *For any function $f : \mathcal{D} \to \{0,1\}$ where $\mathcal{D} \subseteq \{0,1\}^n$, if $\mathcal{P}$ is a span program computing a function agreeing with f on $\mathcal{D}$, then there exists a bounded-error quantum algorithm that evaluates f with $O(\mathrm{wsize}(\mathcal{P},\mathcal{D}))$ queries.*

An Example. Consider the undirected *st*-connectivity problem. Given a graph $G = (V,E)$ and two vertices $s,t \in V$, we need to decide whether there is a path connecting s and t. We build a span program $\mathcal{P}$ for this problem as follows. The vectors of $\mathcal{P}$ are from the space $\mathcal{H} := \mathrm{span}(\{|v\rangle : v \in V\})$. The target vector is $|(s,t)\rangle := |s\rangle - |t\rangle$. The input vectors are of the form $|(u,v)\rangle := |u\rangle - |v\rangle$, for any $u,v \in V$. The input vector $|(u,v)\rangle$ is available on G if and only if $(u,v) \in E$. One can easily see that the target vector lies in the span of available input vectors if and only if s and t are connected in G.

Now let us analyze the witness size of this span program. We need to consider its positive and negative cases separately:

- If s and t are connected by a path of length d, say, $v_0 := s \to v_1 \to v_2 \to \cdots \to v_{d-1} \to v_d := t$, then we can write the target vector $|(s,t)\rangle$ as

$$|(s,t)\rangle = \sum_{j=0}^{d-1} \left(\left| (v_j, v_{j+1}) \right\rangle \right), \tag{3.5}$$

  where each $\left|(v_j,v_{j+1})\right\rangle$ is an available input vector. This implies that the positive witness size of $\mathcal{P}$ is at most $d = O(n)$.

- Otherwise, s and t are in different connected components of G. Let $S \subset V$ be the vertex set of the connected component containing s. Consider the vector

$$\left|w'\right\rangle := \sum_{v \in S} |v\rangle . \tag{3.6}$$

  We claim that $|w'\rangle$ is a negative witness. To see this, note that $\langle (s,t)|w'\rangle = 1$, since $s \in S$ and $t \notin S$. Moreover, for any available input vector $|(u,v)\rangle$, u and v must be in the same connected component of G. This means that either $u,v \in S$ or $u,v \notin S$. In both cases, we have $\langle (u,v)|w'\rangle = 0$.

  To analyze the negative witness size, we note that if an input vector $|(u,v)\rangle$ is not orthogonal to $|w'\rangle$, then we must have $u \in S$, $v \notin S$, or vice versa. This implies that $|\langle (u,v)|w'\rangle| = 1$. Since there are at most $O\left(n^2\right)$ such input vectors, the negative witness size of $\mathcal{P}$ is at most $O\left(n^2\right)$.

Combining the positive and negative cases, we know that the witness size of $\mathcal{P}$ is $O\left(n^{3/2}\right)$. So by Corollary 22, the undirected *st*-connectivity problem has quantum query complexity $O\left(n^{3/2}\right)$ (in fact, this problem has quantum time complexity $\tilde{O}\left(n^{3/2}\right)$, as shown in [24]).

3.3 Span Program for Tree Detection

In this section, we build a span program for the subgraph/not-a-minor problem for any tree, and prove that this span program has $O(n)$ witness size. Then it follows from Corollary 22 that this problem has $O(n)$ quantum query complexity.

The following notation will be useful. For any tree $T = (V_T, E_T)$, let $V_{T,i}$ and $V_{T,l}$ be the set of internal nodes and leaves of T respectively. Also, let r be denote the root of T. For any $x \in V_T$, let $C(x)$ be the set of x's children, and let T^x be the subtree of T rooted at x, and let V_T^x be the set of nodes in T^x, and let $V_{T,i}^x$, $V_{T,l}^x$ be the set of internal and leaf nodes in T^x respectively. Furthermore, let s and t be two special nodes not present in T. Let $V_{T,l}^s = V_{T,l}^t = V_{T,l}$. Then, for any $x, y \in V_T \cup \{s,t\}$, let $S_{x,y} = V_{T,l}^x \cap V_{T,l}^y$. We say that $x, y \in V_T \cup \{s,t\}$ are *adjacent* (denoted by $x \sim y$) if $(x,y) \in E_T$ or $\{x,y\} = \{s,t\}$ or $\{x,y\} = \{s,r\}$ or $\{x,y\} = \{t,f\}$ for some $f \in V_{T,l}$. Then for any $x \in V_T \cup \{s,t\}$ and $f \in V_{T,l}^x$, let $M(x,f) = \{y \in V_T : \ x \sim y, \ f \in S_{x,y}\}$. It is easy to check that $|M(x,f)| = 2$.

For any graph $G = (V_G, E_G)$ and $U \subseteq V_G$, let $G|_U \triangleq (U, E_{G|_U})$ be the *induced subgraph* of G on the vertex set U. Namely, for any $u, v \in U$, $(u,v) \in E_{G|_U}$ if and only if $(u,v) \in E_G$. Furthermore, for any subset $\mathcal{E}$ of E_G, we use $\mathcal{E}(G)$ to denote the graph obtained by contracting the edges in $\mathcal{E}$. Note that $\mathcal{E}(G)$ is well-defined, because the final graph is independent of the order of the edge contractions. In addition, if the vertices $v_1, v_2, \ldots, v_k \in V_G$ are combined together in $\mathcal{E}(G)$ (and no other vertex is combined with them), then we denote this new vertex as $w \triangleq \{v_1, \ldots, v_k\}$ and we say that w *contains* $v_1, \ldots, v_k$. Finally, for any $u \in V_G$ we say that u is *involved* in $\mathcal{E}$ if there exists $v \in V_G$ such that $(u,v) \in \mathcal{E}$.

The main result in this section is:

Theorem 23. *Let $T = (V_T, E_T)$ be an arbitrary tree. Then there exists a bounded-error quantum algorithm for the subgraph/not-a-minor problem for T with $O(n)$ query complexity.*

Proof. Let $G = (V_G, E_G)$ be a graph with n vertices. We need to decide whether G contains T as a subgraph or G does not contain T as a minor, under the promise that one of the cases holds.

Color coding. We use the color coding technique from [5]. Namely, we map each vertex $u \in V_G$ to a uniformly random node $c(u) \in V_T$, and the vertices of G are colored independently. Then, we discard all the "badly" colored edges, i.e. we remove any edge $(u,v) \in E_G$ such that $(c(u), c(v)) \notin E_T$. Let $c : V_G \to V_T$ be a random coloring, and let $G_c = (V_G, E_{G_c})$ be the colored graph corresponding to c [1].

We say that G_c contains a correctly colored T-subgraph if if there is an injection $\iota : V_T \to V_G$ such that: (1) $c \circ \iota$ is the identity, i.e. $c(\iota(a)) = a$ for any $a \in V_T$; (2) for any $x, y \in V_T$, if $(x,y) \in E_T$, then $(\iota(x), \iota(y)) \in E_{G_c}$.

We will construct a span program that accepts if G_c contains a correctly colored T-subgraph, and rejects if G_c does not contain T as a minor. Note that if G contains T as a subgraph, then this

[1] We can determine whether an edge is present in G_c or not by querying the presence of this edge in G and using the information about c.

subgraph is colored correctly with probability at least $|V_T|^{-|V_T|} = \Omega(1)$. So G_c contains T as a subgraph with constant probability. On the other hand, if G does not contain T as a minor, then G_c does not contain T as a minor either. Thus, evaluating our span program for a constant number of independent colorings would suffice to detect T with probability at least $2/3$.

Span program. Our span program $\mathcal{P}$ is defined over the $|V_{T,l}|\,(n+2)$-dimensional space spanned by the vectors

$$\{|u\rangle \otimes |f\rangle : u \in \{s,t\} \cup V_G, f \in V_{T,l}\}, \tag{3.7}$$

where $\langle u|v\rangle = \delta_{u,v}$, for any $u,v \in \{s,t\} \cup V_G$, and $\langle f|g\rangle = \delta_{f,g}$, for any $f,g \in V_{T,l}$.

The target vector of $\mathcal{P}$ is

$$|\tau\rangle \triangleq (|s\rangle - |t\rangle) \otimes \left(\sum_{f \in V_{T,l}} |f\rangle \right). \tag{3.8}$$

The input vectors of $\mathcal{P}$ include the following ones:

- For any $u \in c^{-1}(r)$, there is a free input vector

$$|(s,u)\rangle \triangleq (|s\rangle - |u\rangle) \otimes \left(\sum_{f \in V_{T,l}} |f\rangle \right). \tag{3.9}$$

- For any $f \in V_{T,l}$ and $u \in c^{-1}(f)$, there is a free input vector

$$|(u,t)\rangle \triangleq (|u\rangle - |t\rangle) \otimes |f\rangle). \tag{3.10}$$

- For any $x \in V_{T,i}$, $y \in C(x)$, $u \in c^{-1}(x)$ and $v \in c^{-1}(y)$, there is an input vector

$$|(u,v)\rangle \triangleq (|u\rangle - |v\rangle) \otimes \left(\sum_{f \in V_{T,l}^y} |f\rangle \right), \tag{3.11}$$

  and this input vector is available if and only if $(u,v) \in E_{G_c}$.

Here is a more compact way to describe these input vectors. We add two special vertices s and t to G_c, and color s, t as themselves, i.e. $c(s) = s$ and $c(t) = t$. Furthermore, we connect s to the vertices in $c^{-1}(r)$, and connect t to the vertices in $c^{-1}(f)$ for any $f \in V_{T,l}$. Let G'_c be this modified graph. For any $x \in V_T \cup \{s,t\}$, we call $c^{-1}(x)$ a *block*. Then G'_c contains only edges between *adjacent* blocks. Namely, $c^{-1}(x)$ and $c^{-1}(y)$ are adjacent if and only if x and y are adjacent, i.e. $x \sim y$. Then, for any u,v in adjacent blocks, we have an input vector

$$|(u,v)\rangle = (|u\rangle - |v\rangle) \otimes \left(\sum_{f \in S(c(u),c(v))} |f\rangle \right), \tag{3.12}$$

and this input vector is available if and only if the edge (u,v) is present in G'_c.

An example. Let T be the complete 2-level binary tree, and let G be a 12-vertex graph shown in Fig.3.3. Let $c : V_G \to V_T$ be a coloring defined as $c(u_1) = c(u_2) = r$, $c(u_3) = c(u_4) = d_1$, $c(u_5) = c(u_6) = d_2$, $c(u_7) = f_1$, $c(u_8) = c(u_9) = f_2$, $c(u_{10}) = c(u_{11}) = f_3$, $c(u_{12}) = f_4$. Then, after removing the badly colored edges (such as (u_2,u_8)), the colored graph G_c is shown in Fig.3.3.

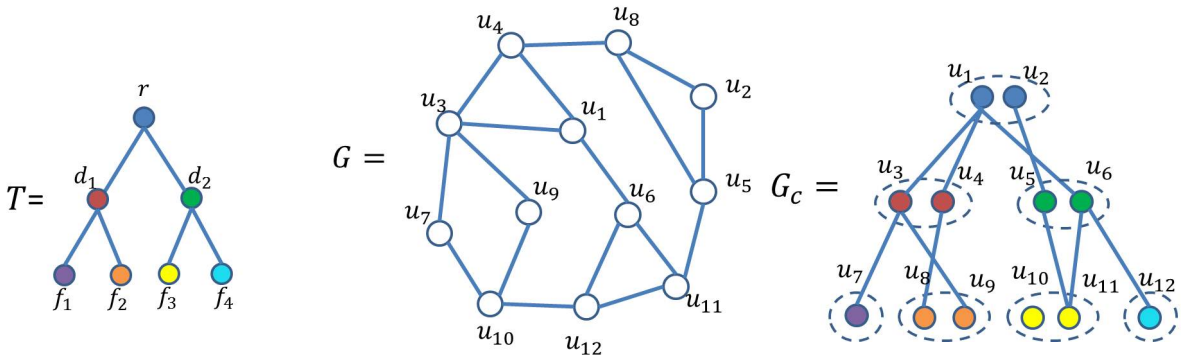

Figure 3.1: An example of coloring coding.

Then the span program $\mathcal{P}$ is defined as follows:

- Target vector: $(|s\rangle - |t\rangle) \otimes (\sum_{j\in[4]} |f_j\rangle)$;
- Free input vectors:
  - $(|s\rangle - |u_i\rangle) \otimes (\sum_{j\in[4]} |f_j\rangle)$, for $i \in [2]$;
  - $(|u_j\rangle - |t\rangle) \otimes |f_{\chi(j)}\rangle$, for $j \in \{7,8,\ldots,12\}$, where $\chi(7) = 1$, $\chi(8) = \chi(9) = 2$, $\chi(10) = \chi(11) = 3$, $\chi(12) = 4$;
- Other input vectors:
  - $(|u_i\rangle - |u_j\rangle) \otimes (|f_1\rangle + |f_2\rangle)$ for $i \in \{1,2\}$ and $j \in \{3,4\}$;
  - $(|u_i\rangle - |u_j\rangle) \otimes (|f_3\rangle + |f_4\rangle)$ for $i \in \{1,2\}$ and $j \in \{5,6\}$;
  - $(|u_i\rangle - |u_7\rangle) \otimes |f_1\rangle$ for $i \in \{3,4\}$;
  - $(|u_i\rangle - |u_j\rangle) \otimes |f_2\rangle$ for $i \in \{3,4\}$ and $j \in \{8,9\}$;
  - $(|u_i\rangle - |u_j\rangle) \otimes |f_3\rangle$ for $i \in \{5,6\}$ and $j \in \{10,11\}$;
  - $(|u_i\rangle - |u_{12}\rangle) \otimes |f_4\rangle$ for $i \in \{5,6\}$.

Witness size. Next, we will show that our span program $\mathcal{P}$ indeed solves the subgraph/not-a-minor problem for T, and along the way we also obtain upper bounds on the positive and negative witness

sizes of $\mathcal{P}$. We will consider the positive and negative cases separately.

Positive case. Let us first consider the positive case, i.e. G_c contains T as a subgraph.

Lemma 24. *Suppose G_c contains T as a subgraph. Then $\mathcal{P}$ accepts G_c. Moreover, the witness size of $\mathcal{P}$ on G_c is $O(1)$.*

Proof. Suppose G_c contains T as a subgraph. Then there exists an injection $\iota : V_T \to V_G$ such that, for any $a \in V_{T,i}$ and $b \in C(a)$, $(\iota(a),\iota(b)) \in E_{G_c}$ and hence the input vector $|(\iota(a),\iota(b))\rangle$ is available. Thus, the target vector $|\tau\rangle$ can be written as

$$|\tau\rangle = |(s,\iota(r))\rangle + \sum_{a\in V_{T,i}} \sum_{b\in C(a)} |(\iota(a),\iota(b))\rangle + \sum_{f\in V_{T,l}} |(\iota(f),t)\rangle. \tag{3.13}$$

So $\mathcal{P}$ accepts G_c. Furthermore, the witness size of $\mathcal{P}$ on G_c is $|E_T| = O(1)$, since T has constant size. □

For example, consider the T and G_c in Fig.3.3. G_c contains T as a subgraph, and $\mathcal{P}$ accepts G_c. The solution is

$$\begin{aligned} |\tau\rangle \;=\; & |(s,u_1)\rangle + |(u_1,u_3)\rangle + |(u_1,u_6)\rangle + |(u_3,u_7)\rangle + |(u_3,u_9)\rangle + |(u_6,u_{11})\rangle + |(u_6,u_{12})\rangle \\ & + |(u_7,t)\rangle + |(u_9,t)\rangle + |(u_{11},t)\rangle + |(u_{12},t)\rangle\,. \end{aligned} \tag{3.14}$$

This solution can be graphically represented by four "parallel" flows from s to t shown in Fig.3.2.

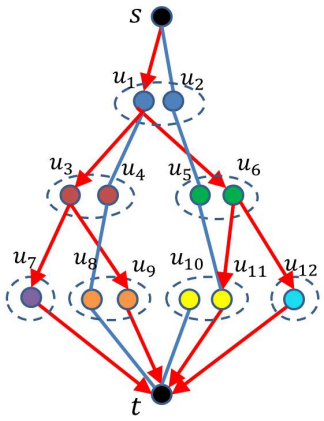

Figure 3.2: A graphical representation of the solution to the span program $\mathcal{P}$ in the positive case. For each edge used by the flows, the coefficient for the corresponding input vector is $+1$. The coefficient for any other input vectors is 0.

Negative case. Now let us consider the negative case, i.e. G_c does not contain T as a *minor*.

But first, let us explain why we do not just consider the case G_c does not contain T as a *subgraph*. The problem is that the span program $\mathcal{P}$ fails to solve this problem in general. Namely,

Claim 25. *There exists some G_c which does not contain T as a subgraph but $\mathcal{P}$ accepts it.*

Proof. For example, consider the T and G_c in Fig.3.3. Here T is the complete 2-level binary tree, and G_c is a 12-vertex graph shown in Fig.3.3. G_c does not contain T as a subgraph, but $\mathcal{P}$ accepts G_c. The solution is

$$\begin{aligned}|\tau\rangle \;=\; & |(s,u_1)\rangle - |(s,u_2)\rangle + |(s,u_3)\rangle + |(u_1,u_5)\rangle + |(u_1,u_6)\rangle - |(u_2,u_6)\rangle - |(u_2,u_6)\rangle + |(u_3,u_7)\rangle \\ & + |(u_3,u_8)\rangle + |(u_5,u_9)\rangle + |(u_5,u_{10})\rangle + |(u_8,u_{11})\rangle + |(u_8,u_{12})\rangle + |(u_9,t)\rangle + |(u_{10},t)\rangle \\ & + |(u_{11},t)\rangle + |(u_{12},t)\rangle\,.\end{aligned} \tag{3.15}$$

This solution can be graphically represented as four flows from s to t shown in Fig.3.3. Notice that these flows move back and forth between adjacent "layers", where each layer is colored by the nodes at the same depth of T. Also, note that if we contract the edges (u_1,u_6), (u_2,u_6), (u_2,u_7) and (u_3,u_7) of G_c, we would get a new graph isomorphic to T. In other words, G_c contains T as a *minor*.

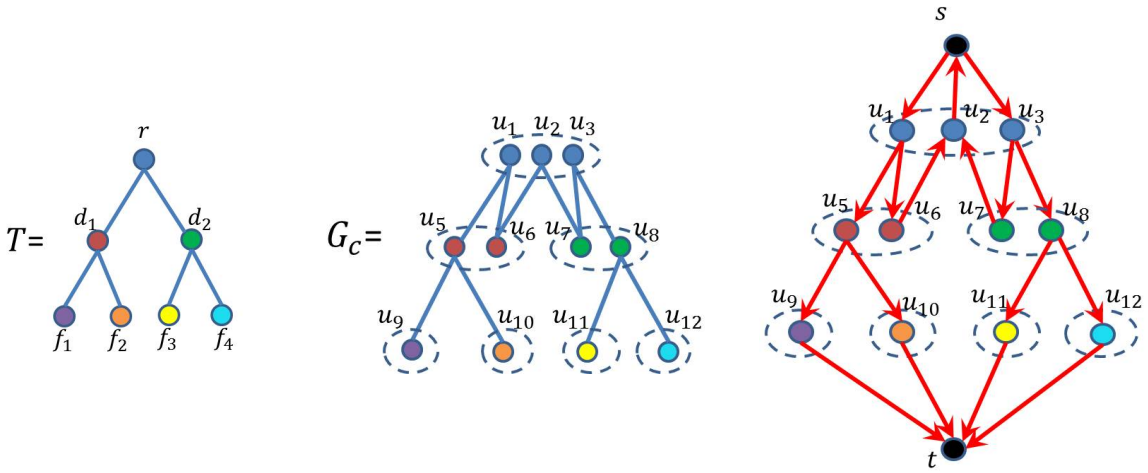

Figure 3.3: An example showing that the span program $\mathcal{P}$ does not solve the tree containment problem in general. Here G_c does not contain T as a subgraph, but $\mathcal{P}$ accepts it. The solution is graphically illustrated by Fig.3.3. For each edge used by the flows, the coefficient for the corresponding input vector is $+1$ or -1, depending on whether the flow moves towards t or s, respectively. The coefficient for any other input vector is 0.

□

We can directly prove that if $\mathcal{P}$ accepts G_c, then G_c must contain T as a minor. But here we prefer to use a different approach. We will assume that G_c does not contain T as a minor. Then, we will show that $\mathcal{P}$ must reject G_c by explicitly giving a negative witness for G_c. The advantage of this approach is that we not only get to know that $\mathcal{P}$ accepts only certain graphs that contain T as a minor, but also obtain an upper bound on the negative witness size of $\mathcal{P}$.

Now let us return to the negative case, i.e. G_c does not contain T as a minor. For convenience, we introduce the following notation. For any $(a,b) \in E_T$ and $u \in c^{-1}(a)$, let $N_{a\to b}(u) = \{v \in c^{-1}(b) : (u,v) \in E_{G_c}\}$. For any $U \subseteq c^{-1}(a)$, let $N_{a\to b}(U) = \cup_{u\in U} N_{a\to b}(u)$. For any $a \in V_T$, let $Y_c^a = c^{-1}(V_T^a) = \cup_{b\in V_T^a} c^{-1}(b)$ and let $G_c^a = G_c|_{Y_c^a}$.

We claim that G_c must satisfy the following property:

Claim 26. *For any graph $G = (V_G, E_G)$ and coloring $c : V_G \to V_T$, if $G_c = (V_G, E_{G_c})$ does not contain T as a minor, then there exist $\{V_a \subseteq c^{-1}(a) : a \in V_{T,i}\}$ and $\{V_{a,b} \subseteq c^{-1}(a) : a \in V_{T,i}, b \in C(a)\}$ such that*

1. *$\forall a \in V_{T,i}$, V_a is the disjoint union of $V_{a,b}$'s for $b \in C(a)$;*
2. *$V_r = c^{-1}(r)$;*
3. *$\forall a \in V_{T,i}$, $\forall b \in C(a) \cap V_{T,i}$, $N_{a\to b}(V_{a,b}) \subseteq V_b$ and $N_{b\to a}(V_b) \subseteq V_{a,b}$;*
4. *$\forall a \in V_{T,i}$, $\forall b \in C(a) \cap V_{T,l}$, $N_{a\to b}(V_{a,b}) = \varnothing$.*

Intuitively, the V_a's contain the "bad" vertices that are responsible for the fact that G_c does not contain T as a minor. For any vertex $u \in V_a$, there is no subgraph of G_c^a that can be contracted into a tree rooted at u and isomorphic to T^a. In particular, since G_c does not contain T as a minor, for any vertex $u \in c^{-1}(r)$, there is no subgraph of G_c that can be contracted into a tree rooted at u and isomorphic to T, and hence $V_r = c^{-1}(r)$. Furthermore, if $u \in V_{a,b} \subseteq V_a$ for some $b \in C(a)$, then u is "bad" because its "children" in $c^{-1}(b)$ are "bad". Namely, if we attempt to use u as the root of T^a, then we will fail because we will not be able to get a *complete* T^b (which is a subtree of T^a). In the degenerate case, b is a leaf, and $u \in V_{a,b}$ if and only if it has no neighbor in $c^{-1}(b)$. Condition 3 tells us that these "bad" vertices form a "connected component" in some sense, and it is crucial for bounding the witness size. Fig.3.4 demonstrates an example of such V_a's and $V_{a,b}$'s.

We defer the proof of Claim 26 to Section 3.3.1. Now let us consider its consequence:

Lemma 27. *Suppose G_c does not contain T as a minor. Then $\mathcal{P}$ rejects it. Moreover, the witness size of $\mathcal{P}$ on G_c is $O\left(n^2\right)$.*

Proof. Using Claim 26, we build a negative witness for G_c as follows. Let

$$|w\rangle = |w_s\rangle + \sum_{a\in V_{T,i}} |w_a\rangle, \tag{3.16}$$

where

$$\begin{aligned} |w_s\rangle &= |s\rangle \otimes \left(\frac{1}{|V_{T,l}|} \sum_{f\in V_{T,l}} |f\rangle\right), \\ |w_a\rangle &= \sum_{b\in C(a)} \left|w_{a,b}\right\rangle \end{aligned} \tag{3.17}$$

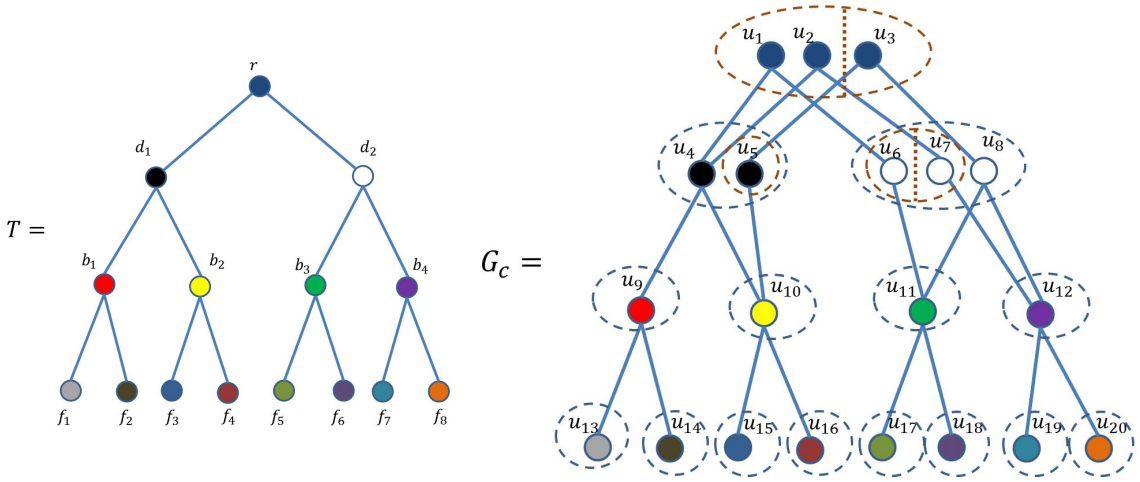

Figure 3.4: An illustration of the partition in Claim 26. In this example, T is the full 3-level binary tree, and G_c does not contain T as a minor. We have $V_r = \{u_1, u_2, u_3\}$, $V_{r,d_1} = \{u_3\}$, $V_{r,d_2} = \{u_1, u_2\}$, $V_{d_1} = V_{d_1,b_1} = \{u_5\}$, $V_{d_2} = \{u_6, u_7\}$, $V_{d_2,b_3} = \{u_7\}$, $V_{d_2,b_4} = \{u_6\}$, and any other V_a or $V_{a,b}$ is $\varnothing$. Note that, $N_{r\to d_2}(V_{r,d_2}) = \{u_6, u_7\} = V_{d_2}$, $N_{d_2\to b_3}(V_{d_2,b_3}) = \varnothing = V_{b_3}$ and $N_{d_2\to b_4}(V_{d_2,b_4}) = \varnothing = V_{b_4}$, etc.

where

$$\left|w_{a,b}\right\rangle = \left(\sum_{u\in V_{a,b}} |u\rangle\right) \otimes \left(\frac{1}{|V^b_{T,l}|} \sum_{f\in V^b_{T,l}} |f\rangle\right). \tag{3.18}$$

Now we prove that $|w\rangle$ is a valid negative witness. First,

$$\langle w|\tau\rangle = \langle w_s|\tau\rangle + \sum_{a\in V_{T,i}} \sum_{b\in C(a)} \left\langle w_{a,b}|\tau\right\rangle = 1 + 0 = 1. \tag{3.19}$$

Next, we show that $|w\rangle$ is orthogonal to all available input vectors on G_c. We deal with four kinds of available input vectors separately:

1. For any free input vector $|(s,u)\rangle = (|s\rangle - |u\rangle) \otimes (\sum_{f\in V_{T,l}} |f\rangle)$ where $u \in c^{-1}(r)$: By conditions 1 and 2 of Claim 26, $c^{-1}(r) = V_r = \cup_{b\in C(r)} V_{r,b}$, so we can find $b \in C(r)$ such that $u \in V_{r,b}$. Then we have $\langle w|(s,u)\rangle = 0$, since $\langle w_s|(s,u)\rangle = 1$, $\left\langle w_{r,b}|(s,u)\right\rangle = -1$, and $\left\langle w_{a',b'}|(s,u)\right\rangle = 0$ for any other (a',b').

2. For any free input vector $|(v,t)\rangle = (|v\rangle - |t\rangle) \otimes |f\rangle$ where $v \in c^{-1}(f)$ for some $f \in V_{T,l}$: We have $\langle w|(v,t)\rangle = 0$, since $\langle w_s|(v,t)\rangle = \left\langle w_{a,b}|(v,t)\right\rangle = 0$ for any (a,b).

3. For any available input vector $|(u,v)\rangle = (|u\rangle - |v\rangle) \otimes (\sum_{f\in V^b_{T,l}} |f\rangle)$ where $u \in c^{-1}(a)$, $v \in c^{-1}(b)$ for some $a \in V_{T,i}$ and $b \in C(a) \cap V_{T,i}$: The availability of this input vector implies $(u,v) \in E_{G_c}$. Then, by condition 3 of Claim 26, there are two possible cases:

- $u \in V_{a,b}$, $v \in V_b$: In this case, $\langle w_{a,b}|(u,v)\rangle = 1$. Also, by condition 1 of Claim 26, there exists $d \in C(b)$ such that $v \in V_{b,d}$. Then $\langle w_{b,d}|(u,v)\rangle = -1$, and $\langle w_s|(u,v)\rangle = \langle w_{a',b'}|(u,v)\rangle = 0$ for any other (a',b'). Thus, $\langle w|(u,v)\rangle = 0$.
- $u \notin V_{a,b}$, $v \notin V_b$: In this case, we simply have $\langle w_s|(u,v)\rangle = \langle w_{a',b'}|(u,v)\rangle = 0$ for any (a',b'). Hence, $\langle w|(u,v)\rangle = 0$.

4. For any available input vector $|(u,v)\rangle = (|u\rangle - |v\rangle) \otimes |f\rangle$ where $u \in c^{-1}(a)$, $v \in c^{-1}(f)$ for some $a \in V_{T,i}$ and $f \in C(a) \cap V_{T,l}$: The availability of this input vector implies $(u,v) \in E_{G_c}$. Then, by condition 4 of Claim 1, we must have $u \notin V_{a,f}$. It follows that $\langle w_s|(u,v)\rangle = \langle w_{a',b'}|(u,v)\rangle = 0$ for any (a',b'), and hence $\langle w|(u,v)\rangle = 0$.

Now, note that $|w\rangle$ can be written as

$$|w\rangle = \sum_{u \in \{s\} \cup V_G} \sum_{f \in V_{T,l}} \mu_{u,f} |u\rangle \otimes |f\rangle \tag{3.20}$$

where $|\mu_{u,f}| \leq 1$ for any (u,f). Meanwhile, every input vector of $\mathcal{P}$ is the sum of a constant number of some $\pm|u\rangle \otimes |f\rangle$'s. Thus, the inner product between $|w\rangle$ and any input vector has norm $O(1)$. Since there are $O(n^2)$ input vectors in $\mathcal{P}$, the negative witness size of $\mathcal{P}$ on G_c is at most $O(n^2)$. □

Combining Lemma 24 and Lemma 27 together, we know that $\mathcal{P}$ solves the subgraph/not-a-minor problem for T, and it has witness size $O(n)$. Then, by Corollary 22, this problem can be solved by a bounded-error quantum algorithm with $O(n)$ query complexity.

□

3.3.1 Proof of Claim 26

Proof of Claim 26. For convenience, we introduce the following notation. Let $H = (V_H, E_H)$ be an arbitrary graph. For any $u \in V_H$, we say that (H,u) is *good* with respect to T if there exists a subgraph $H' = (V_{H'}, E_{H'})$ of H and $\mathcal{E} \subseteq E_{H'}$ such that $\mathcal{E}(H')$ is isomorphic to T and the root of $\mathcal{E}(H')$ contains u (and thus u must be involved in $\mathcal{E}$).

We will prove the following claim (which is stronger than Claim 26):

Claim 28. *For any graph $G = (V_G, E_G)$ and coloring $c : V_G \to V_T$, there exist $L \in \mathbb{N}$, $W \subseteq c^{-1}(r)$, $\{V_{a,l} \subseteq c^{-1}(a) : a \in V_{T,i}, l \in [L]\}$ and $\{V_{a,b,l} \subseteq c^{-1}(a) : a \in V_{T,i}, b \in C(a), l \in [L]\}$, such that:*

1. *$\forall a \in V_{T,i}$, $\forall l \in [L]$, $V_{a,l}$ is the disjoint union of $V_{a,b,l}$'s for $b \in C(a)$;*
2. *$c^{-1}(r)$ is the disjoint union of W and $V_{r,l}$'s for $l \in [L]$;*
3. *$\forall a \in V_{T,i}$, $\forall b \in C(a) \cap V_{T,i}$, $\forall l \in [L]$, $N_{a\to b}(V_{a,b,l}) \subseteq V_{b,l}$ and $N_{b\to a}(V_{b,l}) \subseteq V_{a,b,l}$;*
4. *$\forall a \in V_{T,i}$, $\forall b \in C(a) \cap V_{T,l}$, $\forall l \in [L]$, $N_{a\to b}(V_{a,b,l}) = \varnothing$;*
5. *Let $U_l = \cup_{a \in V_{T,i}} V_{a,l}$, $\forall l \in [L]$, and let $U = \cup_{l \in [L]} U_l$. Then $U_1, U_2, \ldots, U_L$ are the vertex sets of the connected components of $G_c|_U$;*

6. *Let $Z = V_G \setminus U$. Then $\forall w \in W$, $(G_c|_Z, w)$ is good with respect to T.*

Let us first show that Claim 28 indeed implies Claim 26. Suppose G_c does not contain T as a minor. By applying Claim 28, we obtain the W, $V_{a,l}$'s and $V_{a,b,l}$'s satisfying the above conditions. Now define $V_a = \cup_{l\in[L]} V_{a,l}$ and $V_{a,b} = \cup_{l\in[L]} V_{a,b,l}$, for any $a \in V_{T,i}$ and $b \in C(a)$. We claim that these V_a's and $V_{a,b}$'s satisfy all the conditions of Claim 26. This is because:

- Since the $V_{a,l}$'s and $V_{a,b,l}$'s satisfy conditions 1, 3 and 4 of Claim 28, and the V_a's or $V_{a,b}$'s are simply the union of the $V_{a,l}$'s or $V_{a,b,l}$'s respectively, it is obvious that the V_a's and $V_{a,b}$'s satisfy conditions 1, 3 and 4 of Claim 26;
- Since G_c does not contain T as a minor, by condition 4 of Claim 28, we must have $W = \varnothing$. Then, by condition 2 of Claim 28 and $V_r = \cup_{l\in[L]} V_{r,l}$, we have $V_r = c^{-1}(r)$. So condition 2 of Claim 26 is also fulfilled.

Now it remains to prove Claim 28.

Proof of Claim 28. Proof by induction on the depth of T.

1. **Basis**: Suppose T has depth 0. Namely, T contains only a root node r. Then we simply let $W = c^{-1}(r) = V_G$ and $L = 0$ (or let $V_{r,l} = \varnothing$ for any l). Then all the conditions of Claim 28 are trivially satisfied.

   Suppose T has depth 1. Namely, T contains a root node r and its children $f_1, f_2, \ldots, f_k$ (which are the leaves of T). We build the desired W, $V_{a,l}$'s and $V_{a,b,l}$'s as follows:

   - Initially, set $W \leftarrow \varnothing$ and $L \leftarrow 0$.
   - For each vertex $u \in c^{-1}(r)$, do:
     a) If $N_{r\to f_j}(u) \neq \varnothing$ for every $j \in [k]$, then set $W \leftarrow W \cup \{u\}$;
     b) Otherwise, there exists some $j \in [k]$ such that $N_{r\to f_j}(u) = \varnothing$. Then:
        - Set $L \leftarrow L + 1$;
        - Set $V_{r,L} \leftarrow \{u\}$ and $V_{r,f_j,L} \leftarrow \{u\}$;
        - Set $V_{r,f_{j'},L} \leftarrow \varnothing$ for any $j' \neq j$.

   Now we show that the W, $V_{a,l}$'s and $V_{a,b,l}$'s obtained by this algorithm satisfy all the conditions of Claim 28:

   - By the construction, it is obvious that $V_{r,l}$ is the disjoint union of $V_{r,f_j,l}$'s for $j \in [k]$, for any l. So condition 1 is fulfilled;
   - Since each vertex in $c^{-1}(r)$ is put into either W or some $V_{r,l}$, we know that $c^{-1}(r)$ is the disjoint union of W and the $V_{r,l}$'s. So condition 2 is also fulfilled;
   - Since there is only one internal node r and all of its children are leaves, the situation described by condition 3 does not exist and hence condition 3 is automatically satisfied;

- By the definition of $V_{r,l}$'s and $V_{r,f_j,l}$'s, we have $N_{r\to f_j}(V_{r,f_j,l}) = \varnothing$. So condition 4 is also satisfied;
- Since each U_l contains only one vertex, $G_c|_U$ is simply a collection of isolated vertices, and hence $U_1, U_2, \ldots, U_L$ are the vertex sets of the connected components of $G_c|_U$. Therefore, condition 5 is fulfilled;
- For any $w \in W$, it has a neighbor in each of $c^{-1}(f_1), c^{-1}(f_2), \ldots, c^{-1}(f_k)$, and hence $G_c|_Z$ contains a tree that is isomorphic to T and its root is w. So $(G_c|_Z, w)$ is good with respect to T. Thus, condition 6 is fulfilled.

2. **Inductive step**: Suppose that Claim 28 holds for any graph G' and coloring c' with respect to any tree T' of depth at most d.

   Let T be a tree of depth $d+1$. Suppose its root r has k children $d_1, d_2, \ldots, d_k$. For each $j \in [k]$, since T^{d_j} has depth at most d, we can apply Claim 28 to $G_c^{d_j}$ with respect to T^{d_j}, and obtain the W^j, $V^j_{a,\beta}$'s and $V^j_{a,b,\beta}$'s (where $a \in V^{d_j}_{T,i}$, $b \in C(a)$ and $\beta \in [L_j]$ for some L_j) satisfying the conditions of Claim 28. In particular, let $U^j_\beta = \cup_{a \in V^{d_j}_{T,i}} V^j_{a,\beta}$ for $\beta \in [L_j]$, and let $U^j = \cup_{\beta \in [L_j]} U^j_\beta$, for $j \in [k]$. Then $U^j_1, U^j_2, \ldots, U^j_{L_j}$ are the vertex sets of the connected components of $G_c|_{U^j}$ (which is a subgraph of $G_c^{d_j}$), for any $j \in [k]$. Let $Q^j = c^{-1}(r) \cup U^j$ for $j \in [k]$, and let $Q = \cup_{j\in[k]} Q^j$. Then let $H^j = G_c|_{Q^j}$, and let $H = G_c|_Q$. Note that each H^j is a subgraph of H.

   Now consider the connected components of H and H^j's. Suppose the vertex sets of the connected components of H are $A_1, A_2, \ldots, A_m$, and the vertex sets of the connected components of H^j are $B^j_1, B^j_2, \ldots, B^j_{m_j}$ for some m_j, for $j \in [k]$. Note that each A_i is the union of some B^j_t's (for different (j,t)'s), while each B^j_t is the union of several U^j_β's (for different β's) and some subset of $c^{-1}(r)$. Let $E_{i,j} = \{t \in [m_j] : B^j_t \subseteq A_i\}$, and let $F_{j,t} = \{\beta \in [L_j] : U^j_\beta \subseteq B^j_t\}$, for $i \in [m]$, $j \in [k]$ and $t \in [m_j]$. Note that $A_i \cap c^{-1}(r) = \bigcup_{t\in E_{i,j}} \left(B^j_t \cap c^{-1}(r)\right)$ for any i, j.

   We build the desired W, $V_{a,l}$'s and $V_{a,b,l}$'s as follows:

   - Initially, set $W \leftarrow \varnothing$ and $L \leftarrow 0$.
   - For $i := 1$ to m do:
     a) If $N_{r\to d_j}(A_i \cap c^{-1}(r)) \cap W^j \neq \varnothing$ for every $j \in [k]$, then set $W \leftarrow W \cup (A_i \cap c^{-1}(r))$;
     b) Otherwise, there exists some $j \in [k]$ such that $N_{r\to d_j}(A_i \cap c^{-1}(r)) \cap W^j = \varnothing$. Then:
        For each $t \in E_{i,j}$ do:
        - Set $L \leftarrow L+1$;
        - Set $V_{r,L} \leftarrow B^j_t \cap c^{-1}(r)$ and $V_{r,d_j,L} \leftarrow B^j_t \cap c^{-1}(r)$;
        - Set $V_{a,L} \leftarrow \cup_{\beta\in F_{j,t}} V^j_{a,\beta}$ and $V_{a,b,L} \leftarrow \cup_{\beta\in F_{j,t}} V^j_{a,b,\beta}$, for any $a \in V^{d_j}_{T,i}$ and $b \in C(a)$;

– Set any other $V_{a,L}$ or $V_{a,b,L}$ to be $\varnothing$.

Now we show that the W, $V_{a,l}$'s and $V_{a,b,l}$'s obtained by the above algorithm satisfy all the conditions of Claim 28.

- To prove that the $V_{a,l}$'s and $V_{a,b,l}$'s satisfy conditions 1, 3 and 4, we consider two possible cases separately:
  - $a \neq r$: Since the $V^j_{a,\beta}$'s and $V^j_{a,b,\beta}$'s satisfy the conditions 1, 3 and 4 for each $j \in [k]$ (by the inductive hypothesis), and the $V_{a,l}$'s or $V_{a,b,l}$'s are simply the union of several $V^j_{a,\beta}$'s or $V^j_{a,b,\beta}$'s (for consistent choice of j's and β's) respectively, it is easy to see that the $V_{a,l}$'s and $V_{a,b,l}$'s also satisfy conditions 1, 3 and 4.
  - $a = r$:
    a) By construction, for any l, exactly one of the $V_{r,d_j,l}$'s (for $j \in [k]$) equals $V_{r,l}$, and the other $V_{r,d_j,l}$'s are all empty. So $V_{r,l}$ is indeed the disjoint union of the $V_{r,d_j,l}$'s (for $j \in [k]$). Hence, condition 1 is satisfied.
    b) For any internal node d_j, for any l, we have three possible cases:
    * $V_{r,d_j,l} = V_{d_j,l} = \varnothing$;
    * $V_{r,d_j,l} = B^j_t \cap c^{-1}(r)$ and $V_{d_j,l} = \cup_{\beta \in F_{j,t}} V^j_{d_j,\beta}$ for some $t \in [m_j]$;

    For the first case, we have $N_{r\to d_j}(V_{r,d_j,l}) = V_{d_j,l} = \varnothing$ and $N_{d_j\to r}(V_{d_j,l}) = V_{r,d_j,l} = \varnothing$. For the second case, since B^j_t is a connected components of H^j, we also have $N_{r\to d_j}(V_{r,d_j,l}) \subseteq V_{d_j,l}$ and $N_{d_j\to r}(V_{d_j,l}) \subseteq V_{r,d_j,l}$. Therefore, condition 3 is fulfilled.
    c) For any leaf d_j, we have $W^j = c^{-1}(d_j)$. So if $N_{r\to d_j}(A_i \cap c^{-1}(r)) \cap W^j = \varnothing$, then $N_{r\to d_j}(A_i \cap c^{-1}(r)) = \varnothing$. This implies that $N_{r\to d_j}(V_{r,d_j,l}) = \varnothing$ for any l. Hence, condition 4 is also satisfied.
- Since $c^{-1}(r) = \bigcup_{i\in[m]}(A_i \cap c^{-1}(r))$ and $A_i \cap c^{-1}(r) = \bigcup_{t\in E_{i,j}} \left(B^j_t \cap c^{-1}(r)\right)$ for any i, j, by the construction of W and the $V_{r,l}$'s, we know that they form a partition of $c^{-1}(r)$. Thus, condition 2 is also fulfilled.
- Let $U_l = \cup_{a\in V_{T,i}} V_{a,l}$ for $l \in [L]$, and let $U = \cup_{l\in[L]} U_l$. Recall that $U^j_1, U^j_2 \ldots, U^j_{L_j}$ are the vertex sets of the connected components of $G_c|_{U^j}$ for any $j \in [k]$ (by the inductive hypothesis). But viewing from the bigger graph H^j, different U^j_β and $U^j_{\beta'}$ might become connected via some vertex in $c^{-1}(r)$. Now each B^j_t is the vertex set of a connected component of H^j, so it is the union of some subset of $c^{-1}(r)$ and some U^j_β's (for different β's). Then, by construction, we have:
  - For any l, $U_l = B^j_t$ for some j and t. So the vertices in U_l are connected in $G_c|_U$;
  - For any $l \neq l'$, U_l and $U_{l'}$ are disjoint;
  - For any $l \neq l'$, U_l and $U_{l'}$ do not share any vertex in $c^{-1}(r)$. This is because:

  - For any $i \neq i'$, A_i and $A_{i'}$ do not share any vertex in $c^{-1}(r)$, because A_i and A_j are the vertex sets of different connected components of H. It follows that, for any $i \neq i'$, $t \in E_{i,j}$ and $t' \in E_{i',j'}$, B_t^j and $B_{t'}^{j'}$ do not share any vertex in $c^{-1}(r)$, since $B_t^j \subseteq A_i$ and $B_{t'}^{j'} \subseteq A_{i'}$;
  - Also, for any $t \neq t'$, B_t^j and $B_{t'}^j$ do not share any vertex in $c^{-1}(r)$, since B_t^j and $B_{t'}^j$ are the vertex sets of different connected components of H^j;
  - Now, for $l \neq l'$, we have $U_l \cap c^{-1}(r) = B_t^j \cap c^{-1}(r)$ for some j and t, and $U_{l'} \cap c^{-1}(r) = B_{t'}^{j'} \cap c^{-1}(r)$ for some j' and t'. There are two possible cases: (1) either $t \in E_{i,j}$ and $t' \in E_{i',j'}$ where $i \neq i'$; (2) or $t, t' \in E_{i,j}$ and $t \neq t'$. Either way, the above facts imply that U_l and $U_{l'}$ do not share any vertex in $c^{-1}(r)$.

  – For any $(j,\beta) \neq (j',\beta')$, there is no edge between U_β^j and $U_{\beta'}^{j'}$ in $G_c|_U$;

  – The above facts imply that U_l and $U_{l'}$ are disconnected in $G_c|_U$ for any $l \neq l'$.

  Combining these facts, we know that $U_1, U_2, \ldots, U_L$ are the vertex sets of the connected components of $G_c|_U$. So condition 5 is fulfilled.

- Let $Z^j = Y_c^{d_j} \setminus U^j$, for $j \in [k]$. Note that $Z^j \cap U = \varnothing$. Then, by the inductive hypothesis, for any $w \in W^j$, $(G_c|_{Z^j}, w)$ is good with respect to T^{d_j}.

  Now consider any $w \in W$. Since $c^{-1}(r) = \bigcup_{i\in[m]}(A_i \cap c^{-1}(r))$, there exists $i \in [m]$ such that $w \in A_i$. Then by construction, we must have $(A_i \cap c^{-1}(r)) \subseteq W$, and hence $A_i \cap V_{r,l} = \varnothing$ for any l, and hence $A_i \cap U = \varnothing$.

  Pick any $w^j \in N_{r\to d_j}(A_i \cap c^{-1}(r)) \cap W^j$, for $j \in [k]$. Since $(G_c|_{Z^j}, w^j)$ is good with respect to T^{d_j}, $G_c|_{Z^j}$ contains a subgraph $\bar{G}_j$ such that $\bar{G}_j$ can be contracted into a tree $\bar{T}^j$ which is isomorphic to T^{d_j} and the root of $\bar{T}^j$ contains w^j. Importantly, the vertices in $c^{-1}(r)$ and U^j are not involved in such contractions (because they are not in Z^j). Do such contractions for each $j \in [k]$.

  Meanwhile, since A_i is a connected component of H, we can contract it into a single vertex. This contraction can be performed *simultaneously* with the above contractions. The reason is that A_i contains only some vertices in $c^{-1}(r)$ and U^j's, which are not involved in any of the above contractions. Thus, even after the above contractions, we can still contract A_i into a single vertex $v_i \triangleq A_i$ which contains w. Also, v_i will be connected to $w^1, w^2, \ldots, w^k$, which are the roots of $\bar{T}^1, \ldots, \bar{T}^k$ which are isomorphic to $T^{d_1}, T^{d_2}, \ldots, T^{d_k}$ respectively. Thus, the resulting graph contains a tree isomorphic to T. Fig.3.5 illustrates an example of such transformations.

  Now since $A_i \cap U = \varnothing$ and $Z^j \cap U = \varnothing$ for any $j \in [k]$, the above transformation involves only some vertices in $Z = V_G \setminus U$. Hence, $(G_c|_Z, w)$ is good with respect to T. So condition 6 is also fulfilled.

□

□

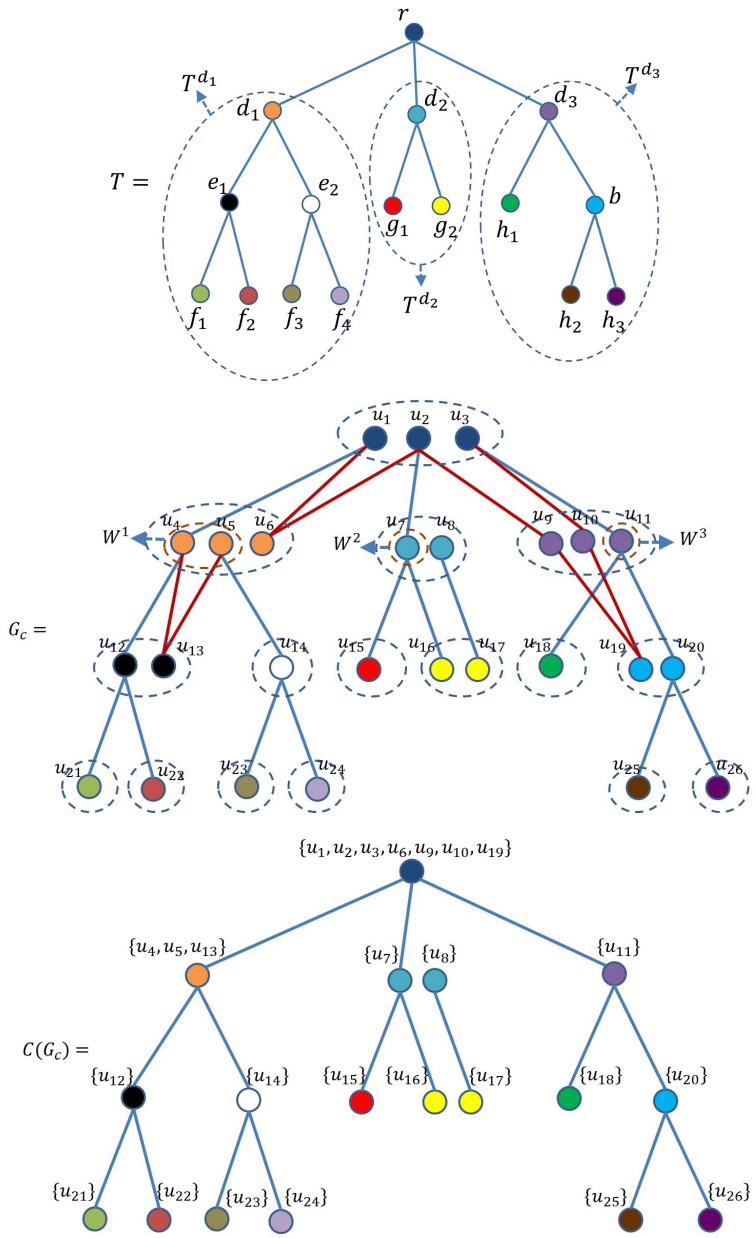

Figure 3.5: An illustration of the transformation described in the proof of Claim 28. In this example, T's root r has 3 children d_1, d_2, d_3. For the given G_c, we have $W^1 = \{u_4, u_5\}$, $U^1 = \{u_6\}$, $W^2 = \{u_7\}$, $U^2 = \{u^8\}$, $W^3 = \{u_{11}\}$, $U^3 = \{u_9, u_{10}, u_{19}\}$, $A_1 = \{u_1, u_2, u_3, u_6, u_9, u_{10}, u_{19}\}$ and $A_2 = \{u_8\}$. Note that $N_{r\to d_1}(A_1 \cap c^{-1}(r)) \cap W^1 = \{u_4\}$, $N_{r\to d_2}(A_1 \cap c^{-1}(r)) \cap W^2 = \{u_7\}$ and $N_{r\to d_3}(A_1 \cap c^{-1}(r)) \cap W^3 = \{u_{11}\}$. By contracting the edges (u_1, u_6), (u_2, u_6), (u_2, u_9), (u_9, u_{19}), (u_{10}, u_{19}), (u_3, u_{10}), all the vertices in A_1 are combined together. We also contract the edges (u_4, u_{13}) and (u_5, u_{13}) to obtain a tree isomorphic to T^{d_1} in the subgraph $G_c|_{Z^1}$. Let $\mathcal{E}$ be the set of the aforementioned edges. Then, the resulting graph $\mathcal{E}(G_c)$ contains a tree isomorphic to T. Hence, $W = \{u_1, u_2, u_3\}$.

3.4 Time-Efficient Implementation

Theorem 23 implies the existence of a *query-efficient* quantum algorithm for tree detection. However, this algorithm is not necessarily *time-efficient*. Although there is a quantum-walk-based algorithm for evaluating any span program [111, 113], the quantum walk in that algorithm might be difficult to implement using local gates. Nevertheless, by using some ideas from [24], we manage to overcome this problem (for our span program $\mathcal{P}$) and hence give a time-efficient implementation of the algorithm in Theorem 23.

Theorem 29. *The algorithm in Theorem 23 can be implemented in $\tilde{O}(n)$ quantum time.*

Proof. We use the general approach of [113] for evaluating span programs. This approach relies on an "effective" spectral gap of a quantum walk operator associated with the span program. Specifically, suppose Q is an arbitrary span program with input vectors $|v_1\rangle, \ldots, |v_k\rangle \in \mathbb{R}^d$ and target vector $|\tau\rangle \in \mathbb{R}^d$. Let $\mathcal{D} \subseteq \{0,1\}^n$, and let $W_1 = \text{wsize}_1(Q, \mathcal{D})$, $W_0 = \text{wsize}_0(Q, \mathcal{D})$, $W = \text{wsize}(Q, \mathcal{D})$ be the positive, negative and overall witness size of Q over domain $\mathcal{D}$, respectively. We assume that $W_1, W_0, W, k = \text{poly}(n)$. Pick a constant $C > \max\{10, 1/W\}$. Let $\alpha = C\sqrt{W_1}$ and $|\tilde{\tau}\rangle = |\tau\rangle / \alpha$. Then define

$$V \triangleq |\tilde{\tau}\rangle\langle 0| + \sum_{j \in [k]} |v_j\rangle\langle j|. \tag{3.21}$$

Let $R_\Lambda = 2\Lambda - I$, where Λ is the projection onto $\text{Ker}(V)$. Moreover, for any $x \in \mathcal{D}$, let $R_x = 2\Pi(x) - I$, where $\Pi(x) \triangleq \sum_{j \in \{0\} \cup I(x)} |j\rangle\langle j|$, where $I(x)$ is the index set of available input vectors on input x. The algorithm for evaluating Q work in the Hilbert space $\mathcal{H} \triangleq \text{span}(|0\rangle, |1\rangle, \ldots, |k\rangle)$. On input x, it starts in $|0\rangle$ and runs phase estimation on $U \triangleq R_\Lambda R_x$ with precision $1/(10CW)$ and error rate $1/10$, and it accepts if and only if the measured phase is 0. This algorithm uses $O(W)$ controlled applications of U, and it correctly evaluates Q on x with probability at least $2/3$. The key component of this algorithm is the implementation of R_Λ and R_x. Usually R_x can be implemented with few input queries and $\text{polylog}(n)$ local gates. But R_Λ can be much harder to implement. Let T_0 and T_1 be the time required to implement R_x and R_Λ respectively. Then the time complexity of this algorithm is $\tilde{O}(W(T_0 + T_1))$.

Now we apply this general approach to our span program $\mathcal{P}$ for tree detection. To efficiently implement the reflection R_Λ, we invoke the spectral lemma. Specifically, we will find two matrices A and B such that: (1) they have the same number of rows; (2) each of them has orthonormal columns; (3) $V' \triangleq A^\dagger B = \frac{1}{\sqrt{4n}} V$. Then, Lemma 17 implies that the -1 eigenspace of $U(A,B) = \text{Ref}_B \cdot \text{Ref}_A$ is $\Big(\mathcal{C}(A) \cap \mathcal{C}(B)^\perp\Big) \oplus \Big(\mathcal{C}(A)^\perp \cap \mathcal{C}(B)\Big)$. Note that $\mathcal{C}(A)^\perp \cap \mathcal{C}(B) = B(\text{Ker}(V')) = B(\text{Ker}(V))$, and $\mathcal{C}(A) \cap \mathcal{C}(B)^\perp$ is orthogonal to $\mathcal{C}(B)$. Thus, to "effectively" implement R_Λ, we embed $\mathcal{H}$ into $B(\mathcal{H})$ and treat the -1 eigenspace of $U(A,B)$ as $\text{Ker}(V)$. That is, we simulate the behavior of R_Λ on any $|\phi\rangle \in \mathcal{H}$ by the behavior of R_{-1} on $B(|\phi\rangle) \in B(\mathcal{H})$, where R_{-1} is the reflection about the -1 eigenspace of $U(A,B)$. This simulation is valid because B is an isometry. Now, R_{-1} can be (approximately) implemented by running phase estimation on $U(A,B)$ and multiplying the phase by -1 if the measured eigenvalue is very close to -1. The precision of this phase estimation depends

on the eigenvalue gap around -1 of $U(A,B)$. Let T_A and T_B be the time required to implement R_A and R_B respectively, and let $\delta_{A,B}$ is the eigenvalue gap around -1 of $U(A,B)$. Then R_Λ can be implemented in time $\tilde{O}((T_A+T_B)/\delta_{A,B})$.

Before describing A and B, let us first modify the span program $\mathcal{P}$ to make it possess a more uniform structure. Specifically, we modify it as follows:

1. We add dummy vertices to each block of G'_c, so that each block contains exactly n vertices. Then we fill in the graph with never-available edges between adjacent blocks, so that there is a complete bipartite graph between any two adjacent blocks.

2. We normalize each input vector to unit length. Specifically, for an input vector $|(u,v)\rangle$, we scale it by a factor of $1/\sqrt{2|S(c(u),c(v))|}$.

3. We scale the target vector $|\tau\rangle$ by a factor of $1/\sqrt{|V_{T,l}|}$ so that it has length $\sqrt{2}$.

4. We pick a constant $C > \max\left\{10, 1/W, 1/\sqrt{W_1}\right\}$. Let $\alpha = C\sqrt{W_1} > 1$ and

$$|\tilde{\tau}\rangle = \frac{1}{\alpha}(|s\rangle - |t\rangle) \otimes \left(\frac{1}{\sqrt{|V_{T,l}|}} \sum_{f \in V_{T,l}} |f\rangle\right). \tag{3.22}$$

   We add a never-available input vector

$$|\gamma\rangle \triangleq \sqrt{1-\frac{1}{\alpha^2}}(|t\rangle - |s\rangle) \otimes \left(\frac{1}{\sqrt{|V_{T,l}|}} \sum_{f \in V_{T,l}} |f\rangle\right). \tag{3.23}$$

It is easy to check that this modified span program still computes the same function, and its positive and negative witness size remain $O(1)$ and $O\left(n^2\right)$ respectively.

Now, since each block of G'_c contains n vertices, we represent the vertices and edges of G'_c as follows. We denote the k-th vertex in the block $c^{-1}(x)$ as (x,k). In particular, we denote the original s as $(s,1)$ and denote the original t as $(t,1)$. Then, for the edge between the vertices (x_1,k_1) and (x_2,k_2), we denote it as (x_1,k_1,x_2,k_2). But this leads to a problem. Namely, this edge also has another representation – (x_2,k_2,x_1,k_1). This could make the implementation of Ref_A and Ref_B a little harder. We solve this problem by making two copies of each input vector (except $|\gamma\rangle$), so that one copy corresponds to the representation (x_1,k_1,x_2,k_2) and the other corresponds to the respresentation (x_2,k_2,x_1,k_1). Furthermore, we make $|\tilde{\tau}\rangle$ and $|\gamma\rangle$ correspond to $(s,1,t,1)$ and $(t,1,s,1)$ respectively.

Now define

$$I \triangleq \{(x,k,f) : x \in V_T \cup \{s,t\}, k \in [n], f \in V_{T,l}^x\} \tag{3.24}$$

and

$$J \triangleq \{(x_1,k_1,x_2,k_2) : x_1,x_2 \in V_T \cup \{s,t\}, x_1 \sim x_2,\ \ k_1,k_2 \in [n]\}. \tag{3.25}$$

Then for our span program $\mathcal{P}$, we have

$$V = \sum_{j \in J} |v_j\rangle\langle j| \tag{3.26}$$

where

$$|v_j\rangle = \begin{cases} \frac{1}{\sqrt{2}}(|x,k\rangle - |y,k'\rangle) \otimes \left(\frac{1}{\sqrt{|S_{x,y}|}} \sum_{f \in S_{x,y}} |f\rangle\right), & \text{if } j = (x,k,y,k') \notin \{(s,1,t,1), \\ & (t,1,s,1)\}, \\ \frac{1}{\alpha}(|s,1\rangle - |t,1\rangle) \otimes \left(\frac{1}{\sqrt{|V_{T,l}|}} \sum_{f \in V_{T,l}} |f\rangle\right), & \text{if } j = (s,1,t,1), \\ \sqrt{1-\frac{1}{\alpha^2}}(|t,1\rangle - |s,1\rangle) \otimes \left(\frac{1}{\sqrt{|V_{T,l}|}} \sum_{f \in V_{T,l}} |f\rangle\right), & \text{if } j = (t,1,s,1). \end{cases} \tag{3.27}$$

Now, we will find unit vectors $\{|a_i\rangle : i \in I\}$ and $\{|b_j\rangle : j \in J\}$ such that

$$\langle a_i|j\rangle \langle i|b_j\rangle = \frac{1}{\sqrt{4n}} \langle i|V|j\rangle, \quad \forall i,j. \tag{3.28}$$

Then we define

$$\begin{aligned} A &= \sum_{i \in I} (|i\rangle \otimes |a_i\rangle) \langle i|, \\ B &= \sum_{j \in J} (|b_j\rangle \otimes |j\rangle) \langle j|. \end{aligned} \tag{3.29}$$

It immediately follows that

$$V' \triangleq A^\dagger B = \frac{V}{\sqrt{4n}}, \tag{3.30}$$

as desired.

The $|a_i\rangle$'s are defined as follows:

- For $i = (x,k,f)$, where $x \notin \{s,t\}$ or $k \neq 1$, let

$$|a_i\rangle = \frac{1}{\sqrt{4n}} \sum_{y \in M(x,f)} \sum_{k' \in [n]} (|x,k,y,k'\rangle + |y,k',x,k\rangle). \tag{3.31}$$

  Note that since $|M(x,f)| = 2$, $|a_i\rangle$ is indeed a unit vector.

- For $i = (s,1,f)$, let

$$\begin{aligned} |a_i\rangle = &\frac{1}{\sqrt{4n}} \left(\sum_{k \in [n]} (|s,1,r,k\rangle + |r,k,s,1\rangle) + \sum_{k=2}^{n} (|s,1,t,k\rangle + |t,k,s,1\rangle) \right) \\ &+ \frac{1}{\sqrt{2n}} \left(\frac{1}{\alpha} |s,1,t,1\rangle + \sqrt{1-\frac{1}{\alpha^2}} |t,1,s,1\rangle \right). \end{aligned} \tag{3.32}$$

- For $i = (t,1,f)$, let

$$\begin{aligned} |a_i\rangle &= \frac{1}{\sqrt{4n}}\left(\sum_{k\in[n]}(|t,1,f,k\rangle + |f,k,t,1\rangle) + \sum_{k=2}^{n}(|t,1,s,k\rangle + |s,k,t,1\rangle)\right) \\ &\quad + \frac{1}{\sqrt{2n}}\left(\frac{1}{\alpha}|s,1,t,1\rangle + \sqrt{1-\frac{1}{\alpha^2}}\,|t,1,s,1\rangle\right). \end{aligned} \tag{3.33}$$

The $|b_j\rangle$'s are defined as follows:

- For $j = (x_1,k_1,x_2,k_2)$, let

$$|b_j\rangle = \frac{1}{\sqrt{2|S_{x_1,x_2}|}}\sum_{f\in S_{x_1,x_2}}(|x_1,k_1,f\rangle - |x_2,k_2,f\rangle). \tag{3.34}$$

It is easy to check that these $|a_i\rangle$'s and $|b_j\rangle$'s are indeed unit vectors and they satisfy Eq.(3.28), as promised.

Now we describe the implementation of R_x and R_Λ. We embed the space $\mathcal{H} = \text{span}\,(|j\rangle : \ j \in J)$ into $B(\mathcal{H}) = \text{span}\,(|b_j\rangle|j\rangle : \ j \in J)$. To implement R_x, we detect j in the second register and multiply the phase by -1 if $|v_j\rangle$ is an unavailable input vector on x (this can be tested by querying the edge labelled by j). To implement R_Λ, we run phase estimation on the quantum walk operator $U(A,B) = \text{Ref}_B \cdot \text{Ref}_A$ with precision $\delta_{A,B}/3$ and multiply the phase by -1 if the measured eigenvalue is $(\delta_{A,B}/3)$-close to -1. It remains to analyze the spectral gap $\delta_{A,B}$ around -1 of $U(A,B)$ and give explicit implementation of Ref_A and Ref_B.

Lemma 30. *The eigenvalue gap $\delta_{A,B}$ around -1 of $U(A,B) = \text{Ref}_B \cdot \text{Ref}_A$ is $\Omega(1)$.*

Proof. By Lemma 17, it is sufficient to show that the singular value gap around 0 of $V' = A^\dagger B$ is $\Omega(1)$. To prove this, it is sufficient to show that the smallest non-zero eigenvalue of

$$\Delta \triangleq V'V'^\dagger = \frac{1}{4n}\sum_{j\in J}|v_j\rangle\langle v_j| \tag{3.35}$$

is $\Omega(1)$. Let $F = \{(x,y) : \ x,y \in V_T \cup \{s,t\},\ x \sim y\}$, and let $F(x,y) = \{(x,k,y,k') : \ k,k \in [n]\}$ for any $(x,y) \in F$. Then we can write Δ as

$$\Delta = \sum_{(x,y)\in F}\Delta_{x,y} \tag{3.36}$$

where

$$\Delta_{x,y} \triangleq \frac{1}{4n}\sum_{j\in F(x,y)}|v_j\rangle\langle v_j| \tag{3.37}$$

Now, for any $j=(x,k,y,k')$, where $(x,y)\notin\{(s,t),(t,s)\}$, we have

$$\begin{aligned}\big|v_j\big\rangle\big\langle v_j\big| &= \frac{1}{2|S_{x,y}|}\sum_{f,f'\in S_{x,y}}\left(|x,k,f\rangle\langle x,k,f'|+|y,k,f\rangle\langle y,k,f'|\right)\\ &\quad -\frac{1}{2|S_{x,y}|}\sum_{f,f'\in S_{x,y}}\left(|x,k,f\rangle\langle y,k',f'|+|y,k',f'\rangle\langle x,k,f|\right).\end{aligned}\tag{3.38}$$

Taking the sum of Eq.(3.38) over $k,k'\in[n]$ yields

$$\Delta_{x,y}=A_{x,y}\otimes I_n+\frac{1}{n}B_{x,y}\otimes E_n,\tag{3.39}$$

where $I_n=\sum_{k\in[n]}|k\rangle\langle k|$ and $E_n=\sum_{k,k'\in[n]}|k\rangle\langle k'|$, and $A_{x,y}$ and $B_{x,y}$ are some matrices independent of n (here we have switched the order of k-register and f-register). This holds for any $(x,y)\notin\{(s,t),(t,s)\}$.

On the other hand, we also have

$$\Delta_{s,t}+\Delta_{t,s}=\bar{A}\otimes I_n+\frac{1}{n}\bar{B}\otimes E_n,\tag{3.40}$$

where $\bar{A}$ and $\bar{B}$ are some matrices independent of n. This can be derived from the fact that for any $j=(s,k,t,k')$ or (t,k',s,k), where $(k,k')\neq(1,1)$,

$$\begin{aligned}\big|v_j\big\rangle\big\langle v_j\big| &= \frac{1}{2|V_{T,l}|}\sum_{f,f'\in V_{T,l}}\left(|s,k,f\rangle\langle s,k,f'|+|t,k,f\rangle\langle t,k,f'|\right)\\ &\quad -\frac{1}{2|V_{T,l}|}\sum_{f,f'\in V_{T,l}}\left(|s,k,f\rangle\langle t,k',f'|+|t,k',f'\rangle\langle s,k,f|\right)\end{aligned}\tag{3.41}$$

and the fact that

$$\begin{aligned}\big|v_{(s,1,t,1)}\big\rangle\big\langle v_{(s,1,t,1)}\big|+\big|v_{(t,1,s,1)}\big\rangle\big\langle v_{(t,1,s,1)}\big| &= \frac{1}{|V_{T,l}|}\sum_{f,f'\in V_{T,l}}\left(|s,1,f\rangle\langle s,1,f'|+|t,1,f\rangle\langle t,1,f'|\right)\\ &\quad -\frac{1}{|V_{T,l}|}\sum_{f,f'\in V_{T,l}}\left(|s,1,f\rangle\langle t,1,f'|+|t,1,f'\rangle\langle s,1,f|\right).\end{aligned}\tag{3.42}$$

Now by Eqs.(3.39) and (3.40) we obtain

$$\Delta=A\otimes I_n+\frac{1}{n}B\otimes E_n,\tag{3.43}$$

where A and B are some matrices independent of n. Then, some simple linear algebra shows that the spectrum of Δ (i.e. the set of eigenvalues sans multiplicity) does not depend on n. In particular, the smallest non-zero eigenvalue of Δ is $\Omega(1)$, as desired. □

Lemma 31. Ref_A *can be implemented in* $\mathrm{polylog}(n)$ *time.*

Proof. Since Ref_A is the reflection about the span of $|i\rangle \otimes |a_i\rangle$'s, we can implement it as $U_A O_A U_A^\dagger$, where U_A is any unitary operation that transforms $|i\rangle \otimes |\bar{0}\rangle$ to $|i\rangle \otimes |a_i\rangle$, and O_A is the reflection about $|\bar{0}\rangle$ on the second subsystem. Obviously, O_A can be implemented in $\mathrm{polylog}(n)$ time. So it remains to show that U_A can be implemented in $\mathrm{polylog}(n)$ time.

Observe that $|a_i\rangle$ can be written as:

$$|a_{(x,k,f)}\rangle = \frac{1}{\sqrt{4n}} \sum_{y \in M(x,f)} \sum_{k' \in [n]} Q\left(|x,k,y,k'\rangle + |y,k',x,k\rangle\right), \tag{3.44}$$

where Q is a unitary operation acting on the span of $|x,k,y,k'\rangle$'s such that

$$\begin{aligned} &Q|x,k,y,k'\rangle = |x,k,y,k'\rangle, \qquad \forall\, (x,k,y,k) \notin \{(s,1,t,1),(t,1,s,1)\}; \\ &Q\left(\frac{1}{\sqrt{2}}|s,1,t,1\rangle + \frac{1}{\sqrt{2}}|t,1,s,1\rangle\right) = \frac{1}{\alpha}|s,1,t,1\rangle + \sqrt{1-\frac{1}{\alpha^2}}\,|t,1,s,1\rangle. \end{aligned} \tag{3.45}$$

So we can generate $|i\rangle \otimes |a_i\rangle$ from $|i\rangle \otimes |\bar{0}\rangle$ as follows:

$$\begin{aligned} |x,k,f\rangle \otimes |0,0,0,0\rangle &\to |x,k,f\rangle \otimes |x,k,0,0\rangle \\ &\to |x,k,f\rangle \otimes \frac{1}{\sqrt{n}} \sum_{k' \in [n]} |x,k,0,k'\rangle \\ &\to |x,k,f\rangle \otimes \frac{1}{\sqrt{2n}} \sum_{y \in M(x,f)} \sum_{k' \in [n]} |x,k,y,k'\rangle \\ &\to |x,k,f\rangle \otimes \frac{1}{\sqrt{4n}} \sum_{y \in M(x,f)} \sum_{k' \in [n]} \left(|x,k,y,k'\rangle + |y,k',x,k\rangle\right) \\ &\to |x,k,f\rangle \otimes \frac{1}{\sqrt{4n}} \sum_{y \in M(x,f)} \sum_{k' \in [n]} Q\left(|x,k,y,k'\rangle + |y,k',x,k\rangle\right) \\ &= |x,k,f\rangle \otimes |a_{(x,k,f)}\rangle, \end{aligned} \tag{3.46}$$

where

- The first step is accomplished by performing a unitary operation $U_{A,1}$ that transforms $|x,k,0,0\rangle$ to $|x,k,x,k\rangle$ on the first, second, fourth and fifth registers;
- The second step is accomplished by performing a unitary operation $U_{A,2}$ that transforms $|0\rangle$ to $\frac{1}{\sqrt{n}} \sum_{k' \in [n]} |k'\rangle$ on the last register;
- The third step is accomplished by performing a unitary operation $U_{A,3}$ that transforms $|x,f\rangle \otimes |0\rangle$ to $|x,f\rangle \otimes \frac{1}{\sqrt{2}} \sum_{y \in M(x,f)} |y\rangle$ on the first, third and sixth registers;
- The fourth step is accomplished by performing a unitary operation $U_{A,4}$ that transforms $|x,k\rangle \otimes |x,k,y,k'\rangle$ to $|x,k\rangle \otimes \frac{1}{\sqrt{2}}(|x,k,y,k'\rangle + |y,k',x,k\rangle)$ on all but the third registers,
- The last step is accomplished by performing Q on the last four registers.

Formally, let $U_A = QU_{A,4}U_{A,3}U_{A,2}U_{A,1}$. Clearly, all of $U_{A,1}$, $U_{A,2}$, $U_{A,3}$, $U_{A,4}$ and Q can be implemented in polylog(n) time, and hence so is U_A. □

Lemma 32. Ref$_B$ *can be implemented in* polylog(n) *time.*

Proof. Since Ref$_B$ is the reflection about the span of $|b_j\rangle \otimes |a_j\rangle$'s, we can implement it as $U_B O_B U_B^\dagger$, where U_B is any unitary operation that transforms $|\bar{0}\rangle \otimes |j\rangle$ to $|b_j\rangle \otimes |j\rangle$, and O_B is the reflection about $|\bar{0}\rangle$ on the first subsystem. Obviously, O_B can be implemented in polylog(n) time. So it remains to show that U_B can be implemented in polylog(n) time.

We generate $|b_j\rangle \otimes |j\rangle$ from $|\bar{0}\rangle \otimes |j\rangle$ as follows:

$$\begin{aligned} |0,0,0\rangle \otimes |x_1,k_1,x_2,k_2\rangle &\rightarrow |0,0\rangle \otimes \left(\frac{1}{\sqrt{|S_{x_1,x_2}|}} \sum_{f \in S_{x_1,x_2}} |f\rangle\right) \otimes |x_1,k_1,x_2,k_2\rangle \\ &\rightarrow \frac{1}{\sqrt{2}}(|x_1,k_1\rangle - |x_2,k_2\rangle) \otimes \left(\frac{1}{\sqrt{|S_{x_1,x_2}|}} \sum_{f \in S_{x_1,x_2}} |f\rangle\right) \otimes |x_1,k_1,x_2,k_2\rangle \\ &= |b_{(x_1,k_1,x_2,k_2)}\rangle \otimes |x_1,k_1,x_2,k_2\rangle . \end{aligned} \tag{3.47}$$

where

- The first step is accomplished by performing a unitary operation $U_{B,1}$ that transforms $|0\rangle \otimes |x_1,x_2\rangle$ to $\left(\frac{1}{\sqrt{|S_{x_1,x_2}|}} \sum_{f \in S_{x_1,x_2}} |f\rangle\right) \otimes |x_1,x_2\rangle$ on the third, fourth and sixth registers.
- The second step is accomplished by performing a unitary operation $U_{B,2}$ that transforms $|0,0\rangle \otimes |x_1,k_1,x_2,k_2\rangle$ to $\frac{1}{\sqrt{2}}(|x_1,k_1\rangle - |x_2,k_2\rangle) \otimes |x_1,k_1,x_2,k_2\rangle$ on the first two and last four registers.

Formally, let $U_B = U_{B,2}U_{B,1}$. Clearly, both $U_{B,1}$ and $U_{B,2}$ can be implemented in polylog(n) time, and hence so is U_B. □

Combining Lemma 30, Lemma 43 and Lemma 44, we know that R_Λ can be implemented in polylog(n) time. Since R_x can be also implemented in polylog(n) time, $U = R_\Lambda R_x$ can be implemented in polylog(n) time. Then, since $\mathcal{P}$ has witness size $O(n)$, the time complexity of our algorithm is $\tilde{O}(n)$, as claimed.

□

3.5 Open Problems

Our work raises many interesting questions:

As mentioned in Section 3.3, our span program $\mathcal{P}$ accepts *some* graphs that contain T as a minor but not as a subgraph. So it fails to solve the T-containment problem. However, it is not true that $\mathcal{P}$ accepts *every* graph containing T as a minor. One can see that a graph must possess

certain structure to be accepted by $\mathcal{P}$. Nevertheless, this structure is not easy to characterize. It would be interesting to know exactly what kind of graphs are accepted by $\mathcal{P}$. If we have a better understanding of this matter, we might be able to modify $\mathcal{P}$ to make it solve the T-containment problem.

In this chapter, we have focused on the detection of trees. It is also worth studying the detection of other subgraphs, such as cycles and cliques. Can we design time-efficient span-program-based quantum algorithms for these problems as well?

Perhaps the most interesting direction is to investigate the potential of our "parallel-flow" technique for designing span programs. In particular, can we use this technique to improve learning graphs [20]? The basic idea of learning graph is to run a single flow from a vertex (i.e. the empty set) to some vertices (i.e. those contain a 1-certificate) on an exponentially large graph, and its efficiency is determined by the energy of this flow. We observe that many decision problems can be decomposed into several *correlated* subproblems, so that an input is a positive instance of the original problem if and only if it is a positive instance of all the subproblems. Perhaps we can improve learning graphs by dividing the original flow into several parallel flows, so that: (1) each flow corresponds to one subproblem; (2) these flows are correlated in a way similar to the subproblems. It is possible that the total energy of these flows is smaller than that of the original flow. Can we formalize this idea and use it to improve previous learning-graph-based quantum algorithms?

Chapter 4

Electrical Flows and Quantum Algorithms

4.1 Overview

Electrical network theory has many implications on classical computation. The beautiful idea of viewing a (weighted) graph as an electrical network has yielded fruitful results in algorithm design and analysis. Examples include the relation between *effective resistances* and commute times of random walks [40], the usage of effective resistances for graph sparsification [122], and the usage of *electrical flows* for approximating maximum flows [55, 94, 97]. How to quickly compute the basic quantities about electrical networks, such as electrical flows and effective resistances, has become not only important for electronic design, but also crucial for the fast solution of many computational problems.

To compute the electrical flow between two given vertices in an electrical network, one need to solve a Laplacian linear system. This can be accomplished in $\tilde{O}(m)$ time by using Spielman and Teng's linear-system solver [123], where m is the number of edges. Since every flow in such a network needs an m-dimensional vector to represent, this method is essentially optimal. On the other hand, it is unclear whether $\tilde{O}(m)$ time is necessary for the computation of effective resistances. To our knowledge, the best known classical algorithm [122] for this task also relies on inverting the Laplacian and takes $\tilde{O}(m)$ time (Specifically, the algorithm of [122] builds in $\tilde{O}(m)$ time a $O(\log n) \times n$ matrix Z from which the effective resistance between any two vertices can be computed in $O(\log n)$ time, where n is the number of vertices).

In this chapter, we develop two quantum algorithms for approximating the effective resistance between two given vertices in an electrical network. Both of them have time complexity polynomial in $\log n$, d, c, $1/\phi$ and $1/\varepsilon$, where n is the number of vertices, d is the maximum degree of the vertices, c is the ratio of the largest to the smallest edge resistance, ϕ is the conductance of the network, and ε is the relative error. In particular, when d and c are small and ϕ is large, our algorithms run very fast. In contrast, it is unknown whether classical algorithms can solve this case fast. Furthermore, we prove that the polynomial dependence on the inverse conductance (i.e. $1/\phi$) is necessary. As a consequence, our algorithms cannot be significantly improved.

Our algorithms are based on using quantum tools to analyze the algebraic properties of graph-

related matrices. While one of them relies on inverting the Laplacian matrix, the other relies on projecting onto the kernel of the weighted incidence matrix.

Specifically, let $G = (V, E, w)$ be a connected weighted graph with n vertices and edge weights $w_e > 0$. Let $s, t \in V$ be different, and let $R_{s,t}$ be the effective resistance between s and t (with respect to edge resistance $r_e = 1/w_e$). Our first algorithm is based on the equation

$$R_{s,t} = \chi_{s,t}^T L^+ \chi_{s,t}, \tag{4.1}$$

where L is the Laplacian of G [1], and $\chi_{s,t} = |s\rangle - |t\rangle$ is an n-dimensional vector. Our strategy is to first produce $L^+\chi_{s,t}$ as a quantum state, and then to estimate its inner product with $\chi_{s,t}$ by quantum measurements. The first step is achieved by invoking (an improved version of) Harrow, Hassidim and Lloyd's quantum algorithm for solving linear systems of equations [78]. Since their algorithm depends polynomially on the (pseudo) condition number of the coefficient matrix in the linear system, our algorithm depends polynomially on the (pseudo) condition number of L, which in turn is related to the conductance ϕ of G. Thus, our algorithm has a polynomial dependence on $1/\phi$.

Our second algorithm is faster than the first one. It is based on the following observation. Let us modify G by adding an edge (labelled by 0) between s and t with unit resistance. Let G' be this modified graph, and let M' be the "weighted" incidence matrix of G'. Then we show that the vectors in the kernel of M' correspond to the s-t flows (of arbitrary values) in G. Let Π be the projection onto the kernel of M'. Then we prove that $\Pi|0\rangle$ corresponds exactly to the *electrical s-t flow* (of some value) in G. Moreover, we can infer $R_{s,t}$ from the value of $\|\Pi|0\rangle\|$. Thus, to estimate $R_{s,t}$, it is sufficient to perform the projective measurement $\{\Pi, I - \Pi\}$ on the state $|0\rangle$, and to estimate the probability of seeing the outcome corresponding to Π. Furthermore, as a by-product, this algorithm also obtains the state $\Pi|0\rangle$, which can be easily transformed into a state (approximately) proportional to the electrical s-t flow in G. This fact might be useful somewhere.

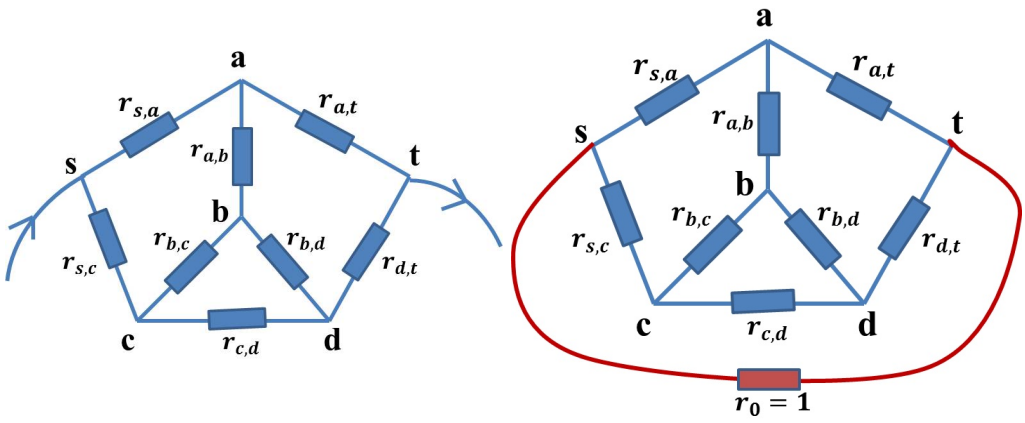

Figure 4.1: An electrical network G and the corresponding G'.

[1] Here L^+ is the *Moore-Penrose Pseudoinverse* of L

The key component of our second algorithm is the implementation of the measurement $\{\Pi, I-\Pi\}$. To achieve this, we generalize the previous method for evaluating the span program for *st*-connectivity [24]. Specifically, we use the spectral lemma to map the kernel of M' to the -1 eigenspace of a quantum walk operator U. Then, we detect this subspace by running phase estimation [87, 104] on U and checking whether the measured eigenvalue is close to -1 or not. The efficiency of this method depends on the eigenvalue gap around -1 or U, which turns out to be related to the conductance of G. Therefore, our second algorithm also has polynomial dependence on $1/\phi$.

We prove the necessity of the polynomial dependence on $1/\Phi$ by building a reduction from PARITY to effective resistance estimation. PARITY is the problem in which we are given an n-bit string $x = x_1 x_2 \ldots x_n$ and are required to answer the value of $\mathrm{PARITY}(x) = \oplus_{i=1}^{n} x_i$. We construct a graph $G(x)$ from x such that if $\mathrm{PARITY}(x) = 0$, then the effective resistance between two special vertices is low; otherwise, the effective resistance between them is high. Thus, by approximating this quantity (to a reasonable accuracy), we can distinguish these two cases and solve PARITY. Since PARITY has quantum query complexity $\Omega(n)$ [18, 65], this implies that estimating effective resistances in $G(x)$ also requires $\Omega(n)$ queries, which is polynomial (but not poly-logarithmic) in the inverse conductance of $G(x)$.

As stated before, by identifying a (weighted) graph as an electrical network and studying the behaviors of electrical flows or effective resistances in this network, one can learn many useful properties of the original graph. We hope that our algorithms for estimating effective resistances and generating electrical flows (as quantum states) could be helpful for solving other problems.

4.2 Spectral Graph Theory

In this section, we briefly introduce *spectral graph theory*, which is the study of the relation between the combinatorial properties (e.g. connectivity, bipartiteness) of a graph and the algebraic properties (e.g. eigenvalues, eigenvectors) of matrices associated to the graph. For more details on this subject, we refer the reader to [56].

Let $G = (V, E, w)$ be a weighted undirected graph with n vertices and m edges and edge weights $w_e > 0$. Let $E(v)$ be the set of edges incident to v, for any $v \in V$. Then let $\deg(v) := |E(v)|$ be the *unweighted degree* of v, and let $\widetilde{\deg}(v) := \sum_{e \in E(v)} w_e$ be the *weighted degree* of v. Furthermore, let $\deg(G) := \max_{v \in V} \deg(v)$ be the *unweighted degree* of G.

Let us orient the edges of G arbitrarily. Then, for any $e \in E$, let e^+, e^- be the head, tail of e respectively. Next, for any $v \in V$, let $E^+(v) := \{e \in E(v) : e^+ = v\}$ and $E^-(v) := \{e \in E(v) : e^- = v\}$.

Now let us define a few matrices about G. Let

$$D_{n \times n} := \mathrm{diag}\left(\widetilde{\deg}(v)\right)_{v \in V} \tag{4.2}$$

be the *weighted degree matrix* of G. Let

$$W_{m \times m} := \mathrm{diag}\,(w_e)_{e \in E} \tag{4.3}$$

be the *edge weight matrix* of G. Let $B_{n\times m}$ be defined as

$$B_{v,e} := \begin{cases} 1, & \text{if } v = e^+, \\ -1, & \text{if } v = e^-, \\ 0, & \text{otherwise .} \end{cases} \tag{4.4}$$

be the *(oriented) vertex-edge incidence matrix* of G. Then, let

$$L := BWB^T \tag{4.5}$$

be the *Laplacian* of G, and let

$$\mathcal{L} := D^{-1/2}LD^{-1/2} \tag{4.6}$$

be the *normalized Laplacian* of G.

Meanwhile, for any $A, B \subset V$, let $w(A) := \sum_{v\in A} \widetilde{\deg}(v)$, and let $w(A,B) := \sum_{u\in A, v\in B, (u,v)\in E} w_{u,v}$. Then, for any $S \subset V$, the *conductance* of S is defined as

$$\phi(S) := \frac{w(S, V\setminus S)}{\min(w(S), w(V\setminus S))} \tag{4.7}$$

Then, the *conductance* of G, denoted by $\phi(G)$, is defined as

$$\phi(G) := \min_{S\subset V} \phi(S) \tag{4.8}$$

Remarkably, Chegeer [41] established a connection between the algebraic quantity $\lambda_2(\mathcal{L})$ and the combinatorial quantity $\phi(G)$:

Theorem 33 (Chegeer's Inequality [41])**.**

$$\frac{\lambda_2(\mathcal{L})}{2} \le \phi(G) \le 2\sqrt{\lambda_2(\mathcal{L})}. \tag{4.9}$$

4.3 Electrical Flows and Effective Resistances

Let $G = (V, E, w)$ be a connected weighted undirected graph with n vertices and m edges and edge weight $w_e > 0$. We identify G as an electrical network of resistors, where edge e has resistance $r_e := 1/w_e$ (or equivalently, conductance w_e). Let $s, t \in V$ be distinct. An *s-t flow* is a function $\mathbf{f} : E \to \mathbb{R}$ that obeys the flow-conservation constraints:

$$\sum_{e\in E^+(v)} \mathbf{f}(e) - \sum_{e\in E^-(v)} \mathbf{f}(e) = 0, \quad \forall\, v \in V\setminus\{s,t\}. \tag{4.10}$$

The *value* $|\mathbf{f}|$ of the flow $\mathbf{f}$ is defined to be the net flow out of the source vertex, i.e.

$$|\mathbf{f}| := \sum_{e\in E^-(s)} \mathbf{f}(e) - \sum_{e\in E^+(s)} \mathbf{f}(e). \tag{4.11}$$

A *unit s-t* flow is a *s-t* flow of value 1. Moreover, the *energy* (with respect to $r_e = 1/w_e$) of the flow $\mathbf{f}$ is defined as

$$\mathcal{E}(\mathbf{f}) := \sum_{e \in E} r_e \mathbf{f}^2(e) = \sum_{e \in E} \frac{\mathbf{f}^2(e)}{w_e}. \tag{4.12}$$

The *electrical s-t flow of value F* is the flow that minimizes $\mathcal{E}(\mathbf{f})$ among all *s-t* flows $\mathbf{f}$ of value F. This flow can be shown to be unique. Furthermore, the *effective s-t resistance*, denoted by $R_{s,t}$, is the energy $\mathcal{E}(\mathbf{i})$ of the *unit* electrical *s-t* flow $\mathbf{i}$.

An alternative definition of effective resistance is as follows. Imagine that we inject some currents at some vertices, and extract some currents at some other vertices, such that the total amount of injected currents equals the total amount of extracted currents. This will induce some electrical potentials at the vertices and some electrical currents through the edges. Let $\mathbf{i}_{ext} : V \to \mathbb{R}$ be such that $\mathbf{i}_{ext}(v)$ is the current injected at vertex v [2]. Let $\mathbf{v} : V \to \mathbb{R}$ be such that $\mathbf{v}(v)$ is the induced potential at vertex v. Let $\mathbf{i} : E \to \mathbb{R}$ be such that $\mathbf{i}(e)$ is the induced current through edge e. By Ohm's law, the current through an edge is equal to the potential difference across its ends times its conductance:

$$\mathbf{i} = WB^T\mathbf{v}. \tag{4.13}$$

In addition, by Kirchoff's current law, the sum of the currents entering a vertex is equal to the amount injected at the vertex:

$$B\mathbf{i} = \mathbf{i}_{ext}. \tag{4.14}$$

Combining these two equations, we get

$$\mathbf{i}_{ext} = B\mathbf{i} = B(WB^T\mathbf{v}) = L\mathbf{v}. \tag{4.15}$$

Since $\mathbf{i}_{ext} \perp \mathrm{Ker}(L) = \mathrm{span}(\mathbf{1})$, we have

$$\mathbf{v} = L^+\mathbf{i}_{ext}. \tag{4.16}$$

Now suppose we inject a unit current at s and extract the same amount of current at t. In this case, $\mathbf{i}_{ext} = \chi_{s,t} := |s\rangle - |t\rangle$. Then the effective *s-t* resistance is defined as the potential difference between s and t. Namely,

$$R_{s,t} = \mathbf{v}(s) - \mathbf{v}(t) = \chi_{s,t}^T\mathbf{v} = \chi_{s,t}^T L^+ \chi_{s,t}. \tag{4.17}$$

4.4 Our Model

Throughout this chapter, we identify weighted graphs as electrical networks. Specifically, let $G = (V, E, w)$ be a weighted graph with n vertices and edge weights $w_e > 0$. We identify it as an electrical network with edge resistance $r_e = 1/w_e$. We always assume that this graph is given in the *incidence list* model. Namely, there are three quantum oracles O_1, O_2 and O_3 for G:

[2] $\mathbf{i}_{ext}(v)$ can be positive or negative depending on whether we inject or extract some current at vertex v.

- O_1 takes a vertex v and an integer j as input, and returns the j-th incident edge of v. Formally, for $v \in V$ and $j \in [n]$, $O_1 |v\rangle |j\rangle = |v\rangle |e_j\rangle$ where e_j is the j-th incident edge of v. If such edge does not exist, i.e. $\deg(v) < j$, then let $O_1 |v\rangle |j\rangle = |v\rangle |-j\rangle$ [3].
- O_2 takes an edge e as input, and returns the two endpoints of e. Formally, for $e = (u,v) \in E$, $O_2 |e\rangle |0\rangle |0\rangle = |e\rangle |u\rangle |v\rangle$.
- O_3 takes an edge e as input, and returns the weight of e. Formally, for $e \in E$, $O_3 |e\rangle |0\rangle = |e\rangle |w_e\rangle$.

Our algorithms have access to O_1, O_2, O_3 and their inverses. Given two vertices $s,t \in V$ and an accuracy parameter $\varepsilon > 0$, they output a number $\tilde{R}_{s,t}$ such that $|\tilde{R}_{s,t} - R_{s,t}| \leq \varepsilon R_{s,t}$ with probability at least $2/3$. Namely, they estimate $R_{s,t}$ to a relative error ε with high probability.

The query complexity of an algorithm in this model is defined as the number of calls to O_1, O_2, O_3 and their inverses. The time complexity of an algorithm in this model is defined as its query complexity plus the number of additional one- and two-qubit gates used. The complexities of our algorithms will be characterized in terms of n, $d := \deg(G)$, $c := \max_{e \in E} w_e / \min_{e \in E} w_e$, $\phi := \phi(G)$ and ε.

4.5 A Simple Quantum Algorithm for Estimating Effective Resistances

In this section, we describe a simple quantum algorithm for approximating the effective resistance between two given vertices in a connected weighted graph. Similar to classical algorithms, this algorithm also depends on solving the Laplacian linear system $L\mathbf{v} = \chi_{s,t}$. However, it does not write down the solution $\mathbf{v} = L^{+}\chi_{s,t}$ explicitly. Instead, it just produces a quantum state approximately proportional to $\mathbf{v}$. Then it estimates $R_{s,t}$ by making appropriate measurement on this state. Although this algorithm is slower than the (main) algorithm presented in the next section, we still put it here not only because it is simple, but also because it is a first attempt by a quantum algorithm person to estimate effective resistances.

Before describing this algorithm, it is useful to prove the following lemmas. Let $a := \min_{e \in E} w_e$ and $b := \max_{e \in E} w_e$. Then $c = b/a$. Furthermore, we have

Lemma 34.

$$\lambda_2(L) = \Omega\left(a\phi^2\right), \quad \lambda_n(L) = O\left(bd\right). \tag{4.18}$$

Proof. Consider the normalized Laplacian $\mathcal{L} = D^{-1/2} L D^{-1/2}$, where $D = \operatorname{diag}\left(\widetilde{\deg}(v)\right)_{v \in V}$. By Chegeer's inequality, we have $\lambda_2(\mathcal{L}) = \Omega\left(\phi^2\right)$. Moreover, it always holds that $\lambda_n(\mathcal{L}) = O(1)$.

[3] We assume that each edge $e \in E$ is labelled by a positive number.

Furthermore, since $a \leq \widetilde{\deg}(v) = \sum_{e \in E(v)} w_e \leq bd$ for any $v \in V$, we have

$$aI \preccurlyeq D \preccurlyeq bdI, \tag{4.19}$$

Then, by $L = D^{1/2} \mathcal{L} D^{1/2}$, we get

$$\lambda_2(L) \geq a\lambda_2(\mathcal{L}) = \Omega\left(a\phi^2\right), \tag{4.20}$$

and

$$\lambda_n(L) \leq bd\lambda_n(\mathcal{L}) = O\left(bd\right). \tag{4.21}$$

□

Corollary 35.

$$\Omega\left(\frac{1}{bd}\right) \leq R_{s,t} \leq O\left(\frac{1}{a\phi^2}\right). \tag{4.22}$$

Proof. Since $R_{s,t} = \chi_{s,t}^T L^+ \chi_{s,t}$, where $\chi_{s,t} \in \mathcal{C}(L)$ and $\|\chi_{s,t}\|^2 = 2$, we have

$$\Omega\left(\frac{1}{\lambda_n(L)}\right) \leq R_{s,t} \leq O\left(\frac{1}{\lambda_2(L)}\right). \tag{4.23}$$

Then the desired result follows from Lemma 34. □

Now we describe an improved version of HHL's algorithm:

Lemma 36 (Implied by [78] and [32])**.** *Let A be a d-sparse $N \times N$ Hermitian matrix such that all the nonzero eigenvalues of A are between $1/\kappa$ and 1. Let z be an N-dimensional unit vector such that $z \in \mathcal{C}(A)$. Let O_A be an oracle that maps $|i\rangle|j\rangle|0\rangle$ to $|i\rangle|h(i,j)\rangle\left|A_{i,h(i,j)}\right\rangle$, for any $i, j \in [N]$, where $h(i,j)$ is the column index of the j-th nonzero entry in the i-th row of A [4]. Let O_z be an oracle that maps $|0\rangle$ to $|z\rangle$. Let $x = A^+ z$. Then $|\bar{x}\rangle$ can be prepared to accuracy $\varepsilon > 0$ with $\tilde{O}\left(d^2\kappa^2/\varepsilon\right)$ queries to O_A, O_z and their inverses, and $\tilde{O}\left(\mathrm{polylog}\,(N) \cdot d^2\kappa^2/\varepsilon\right)$ additional one- and two-qubit gates. Furthermore, $\|x\|$ can be estimated to a relative error ε with $\tilde{O}\left(d^2\kappa^2/\varepsilon^2\right)$ queries to O_A, O_z and their inverses, and $\tilde{O}\left(\mathrm{polylog}\,(N) \cdot d^2\kappa^2/\varepsilon^2\right)$ additional one- and two-qubit gates.*

Proof. Suppose A has the spectral decomposition

$$A = \sum_{j=1}^{m} \lambda_j \left|v_j\right\rangle\left\langle v_j\right|, \tag{4.24}$$

where λ_j's are the nonzero eigenvalues of A and $1/\kappa \leq \lambda_j \leq 1$, and $\left|v_j\right\rangle$'s are the corresponding eigenvectors of A. Since $|z\rangle \in \mathcal{C}(A) = \mathrm{span}\left(|v_1\rangle, |v_2\rangle, \ldots, |v_m\rangle\right)$, we can write it as

$$|z\rangle = \sum_{j=1}^{m} c_j \left|v_j\right\rangle \tag{4.25}$$

[4] If j is greater than the number of nonzero entries in the i-th row of A, then let $h(i,j) := -j$ and $A_{i,-j} := 0$.

for some coefficients c_j's. Then we have

$$|x\rangle = A^+ |b\rangle = \sum_{j=1}^{m} c_j \lambda_j^{-1} |v_j\rangle . \tag{4.26}$$

Now we use an improved version of HHL's algorithm to produce the state $|\bar{x}\rangle$ approximately. This algorithm works as follows (for convenience, we assume that phase estimation is perfect in the following description, and we will take the error of phase estimation into account later):

1. Starting with the state $|0\rangle$, we prepare the state $|z\rangle$ by calling O_z once.

2. We run phase estimation on the unitary operation e^{iA} starting with the state $|z\rangle$, and get the state
$$\sum_{j=1}^{m} c_j |v_j\rangle |\lambda_j\rangle . \tag{4.27}$$

3. We append a qubit in the state $|0\rangle$ and perform the controlled-rotation
$$|\lambda\rangle |0\rangle \to |\lambda\rangle \left(\frac{C}{\lambda} |1\rangle + \sqrt{1 - \frac{C^2}{\lambda^2}} |0\rangle \right) \tag{4.28}$$
on the last two registers, where $C = \Theta(1/\kappa)$. Then we obtain the state
$$\sum_{j=1}^{m} c_j |v_j\rangle |\lambda_j\rangle \left(\frac{C}{\lambda_j} |1\rangle + \sqrt{1 - \frac{C^2}{\lambda_j^2}} |0\rangle \right) . \tag{4.29}$$

4. We measure the last qubit in the standard basis. Then, conditioned on seeing outcome 1, the rest of the state is proportional to
$$\sum_{j=1}^{m} c_j \lambda_j^{-1} |v_j\rangle |\lambda_j\rangle . \tag{4.30}$$
Moreover, since $1/\kappa \leq \lambda_j \leq 1$, this outcome occurs with probability
$$q := C^2 \sum_{j=1}^{m} c_j^2 \lambda_j^{-2} = \Omega \left(\frac{1}{\kappa^2} \right) . \tag{4.31}$$

5. We uncompute the last register by undoing phase estimation. Then we get a state proportional to
$$\sum_{j=1}^{m} c_j \lambda_j^{-1} |v_j\rangle = |x\rangle . \tag{4.32}$$

6. The above procedure only succeeds with probability $\Omega\left(1/\kappa^2\right)$. By Corollary 11, we can raise this probability to $1-O(\varepsilon)$ by repeating this procedure and its inverse $O(\kappa\log(1/\varepsilon))$ times. This would ensure that we get a state ε-close to $|\bar{x}\rangle$.

Now we take the error of phase estimation into consideration, and analyze the resource cost of this algorithm. In step 2, we do not get λ_j exactly, but instead we get some (random) $\tilde{\lambda}_j \approx \lambda_j$. Thus, we only obtain the states in steps 3-5 approximately. In order to get a state ε-close to $|\bar{x}\rangle$, we need to make sure that $|\tilde{\lambda}_j - \lambda_j| = O\left(\varepsilon \cdot \lambda_j\right)$. Then, since $1/\kappa \leq \lambda_j \leq 1$, we need to set the precision of phase estimation to be $\Theta(\varepsilon/\kappa)$. This implies that we need to simulate e^{iAt} for some $t = O(\kappa/\varepsilon)$ in phase estimation. To achieve this, the original HHL's algorithm uses the method of [30] for sparse Hamiltonian simulation. Here we use the most recent method of [32]. Compared to the previous methods [4, 30, 50], this one not only scales better in the sparsity of the Hamiltonian, but also scales only poly-logarithmically in the inverse error. So we can safely ignore the precision issue of Hamiltonian simulation [5]. Now, since A is d-sparse and $\|A\| \leq 1$, by Theorem 1.1 of [32], we can simulate e^{iAt} for $t = O(\kappa/\varepsilon)$ with $\tilde{O}\left(d^2\kappa/\varepsilon\right)$ queries to O_A and its inverse, and $\tilde{O}\left(\log N \cdot d^2\kappa/\varepsilon\right)$ additional one- and two-qubit gates. It follows that one iteration of steps 1-5 requires $\tilde{O}\left(d^2\kappa/\varepsilon\right)$ queries to O_A, O_z and their inverses, and $\tilde{O}\left(\text{polylog}(N) \cdot d^2\kappa/\varepsilon\right)$ additional one- and two-qubit gates. Since amplitude amplification requires $O(\kappa\log(1/\varepsilon))$ repetitions of steps 1-5 and their inverses, the total resource cost of this algorithm is $\tilde{O}\left(d^2\kappa^2/\varepsilon\right)$ queries to O_A, O_z and their inverses, and $\tilde{O}\left(\text{polylog}(N) \cdot d^2\kappa^2/\varepsilon\right)$ additional one- and two-qubit gates, as claimed.

The algorithm for estimating $\|x\|$ is very similar to the one above. The only difference is that now we run *amplitude estimation* (instead of amplitude amplification) to estimate $q = C^2\|x\|^2$. In order to estimate $\|x\|$ to a relative error ε, we need to esimate q to a relative error $O(\varepsilon)$. This means that amplitude estimation needs $O(\kappa/\varepsilon)$ repetitions of steps 1-5 and their inverses, since $q = \Omega(1/\kappa)$. By a similar analysis, we get that this algorithm requires $\tilde{O}\left(d^2\kappa^2/\varepsilon^2\right)$ queries to O_A, O_z and their inverses, and $\tilde{O}\left(\text{polylog}(N) \cdot d^2\kappa^2/\varepsilon^2\right)$ additional one- and two-qubit gates, as claimed.

□

Now we are ready to state the main result of this section:

Proposition 37. *Let $G = (V, E, w)$ be a connected weighted graph given in the incidence list model. Let $s, t \in V$ be different, and let $\varepsilon > 0$. Then $R_{s,t}$ can be estimated to a relative error ε with probability at least $2/3$ in $\tilde{O}\left(\text{polylog}(n) \cdot d^8c^5/(\phi^{10}\varepsilon^2)\right)$ quantum time, where $n = |V|$, $d = \deg(G)$, $c = \max_{e\in E} w_e / \min_{e\in E} w_e$ and $\phi = \phi(G)$.*

Proof. Before describing our algorithm, let us make a few observations. First, by Lemma 34, we can rescale the edge weights w_e's to make $\lambda_2(L) \geq 1/\kappa$ and $\lambda_n(L) \leq 1$, where $\kappa = O\left(cd/\phi^2\right)$. Obviously, this rescaling does not change the difficulty of estimating $R_{s,t}$. Also, after this rescaling, we have $\Omega(1) \leq R_{s,t} \leq O(\kappa)$. Second, let O_L be an oracle for L that maps $|i\rangle|j\rangle|0\rangle$ to

[5] Taking the precision issue of Hamiltonian simulation into account only increases the complexity of our algorithm by a poly-logarithmic factor

$|i\rangle|f(i,j)\rangle\left|L_{i,f(i,j)}\right\rangle$, for any $i,j\in[n]$, where $f(i,j)$ is the column index of the j-th nonzero entry in the i-th row of L [6]. Then we can implement O_L by calling O_1, O_2, O_3 and their inverses $O(\log n+d)$ times.

Now recall the equation

$$R_{s,t}=\langle\chi_{s,t}|L^+|\chi_{s,t}\rangle=\langle\chi_{s,t}|\mathbf{v}\rangle=\sqrt{2}\|\,|\mathbf{v}\rangle\,\|\cdot\langle\overline{\chi_{s,t}}|\bar{\mathbf{v}}\rangle\,, \tag{4.33}$$

where $|\mathbf{v}\rangle=L^+|\chi_{s,t}\rangle$ is the electrical potential induced by the unit electrical s-t flow. So, in order to estimate $R_{s,t}$ to a relative error ε, it is sufficient to estimate both $\|\mathbf{v}\|$ and $\langle\overline{\chi_{s,t}}|\bar{\mathbf{v}}\rangle$ to a relative error $O(\varepsilon)$. Note that $\||\mathbf{v}\rangle\|=\|L^+|\chi_{s,t}\rangle\|=O(\kappa)$, since $\lambda_2(L)\geq 1/\kappa$ and $\||\chi_{s,t}\rangle\|=\sqrt{2}$. Then, by $R_{s,t}=\Omega(1)$, we have $\langle\overline{\chi_{s,t}}|\bar{\mathbf{v}}\rangle=\Omega(1/\kappa)$.

To estimate $\||\mathbf{v}\rangle\|=\|L^+|\chi_{s,t}\rangle\|=\sqrt{2}\,\|L^+|\overline{\chi_{s,t}}\rangle\|$, we directly invoke the second algorithm in Lemma 36. Since O_L can be implemented in $O(\log n+d)$ time, all the nonzero eigenvalues of L are between $1/\kappa$ and 1, and $|\overline{\chi_{s,t}}\rangle=(|s\rangle-|t\rangle)/\sqrt{2}$ can be prepared in $O(\log n)$ time, Lemma 36 implies that we can estimate $\||\mathbf{v}\rangle\|$ to a relative error $O(\varepsilon)$ in $\tilde{O}\left(\mathrm{polylog}\,(n)\cdot d^3\kappa^2/\varepsilon^2\right)=\tilde{O}\left(\mathrm{polylog}\,(n)\cdot d^5c^2/(\phi^4\varepsilon^2)\right)$ time.

To estimate $\langle\overline{\chi_{s,t}}|\bar{\mathbf{v}}\rangle$, we invoke the first algorithm in Lemma 36 to prepare the state $|\bar{\mathbf{v}}\rangle$ approximately. Then, we perform the measurement $\{|\overline{\chi_{s,t}}\rangle\langle\overline{\chi_{s,t}}|\,,I-|\overline{\chi_{s,t}}\rangle\langle\overline{\chi_{s,t}}|\}$ on this state, and estimate the probability of getting the outcome corresponding to $|\overline{\chi_{s,t}}\rangle\langle\overline{\chi_{s,t}}|$ by amplitude estimation. Since $\langle\overline{\chi_{s,t}}|\bar{\mathbf{v}}\rangle=\Omega(1/\kappa)$ and we want to estimate it to a relative error $O(\varepsilon)$, we need to prepare the state $|\bar{\mathbf{v}}\rangle$ to accuracy $O\left(\varepsilon/\kappa^2\right)$, and amplitude estimation requires $O(\kappa/\varepsilon)$ repetitions of the "prepare-and-measure" procedure and its inverse. Then, since O_L can be implemented in $O(\log n+d)$ time and all the nonzero eigenvalues of L are between $1/\kappa$ and 1, Lemma 36 implies that this takes $\tilde{O}\left(\mathrm{polylog}\,(n)\cdot d^3\kappa^5/\varepsilon^2\right)=\tilde{O}\left(\mathrm{polylog}\,(n)\cdot d^8c^5/(\phi^{10}\varepsilon^2)\right)$ time.

Overall, the time complexity of this algorithm is $\tilde{O}\left(\mathrm{polylog}\,(n)\cdot d^8c^5/(\phi^{10}\varepsilon^2)\right)$, as claimed.

□

Remark 38. *Ambainis [9] has introduced a technique called* variable-time amplitude amplification *and used it to improve the dependence of HHL's algorithm on the (pseudo) condition number* κ*. Using his technique, we can also slightly improve the dependence of the algorithm in Proposition 37 on the parameters d, c and* ϕ*. But even this improved version is still worse than than the (main) algorithm described in the next section.*

4.6 A Faster Quantum Algorithm for Estimating Effective Resistances

In this section, we present a faster quantum algorithm for approximating the effective resistance between two given vertices in a connected weighted graph. This algorithm does not need to solve linear systems. Instead, it is based on an observation about the kernel of the weighted incidence

[6] If j is larger than the number of nonzero entries in the i-th row of L, then let $f(i,j):=-j$ and $L_{i,-j}:=0$.

matrix of a slightly modified graph, and it crucially relies on implementing the projection onto this subspace. This is achieved by running phase estimation on a quantum walk operator. As a by-product, (a slight variant of) this algorithm also produces a quantum state approximately proportional to the electrical *s*-*t* flow, which might be of independent interest.

Theorem 39. *Let $G = (V,E,w)$ be a connected weighted graph given in the incidence list model. Let $s,t \in V$ be different, and let $\varepsilon > 0$. Then $R_{s,t}$ can be estimated to a relative error ε with probability at least $2/3$ in $\tilde{O}\left(\log n \cdot \min\left(d^{3/2}c^{3/2}, d^2c\right)/(\phi^2\varepsilon)\right)$ quantum time, where $n = |V|$, $d = \deg(G)$, $c = \max_{e\in E} w_e / \min_{e\in E} w_e$ and $\phi = \phi(G)$.*

Proof. Before describing this algorithm, let us make a few modifications to the graph $G = (V,E,w)$. First, by Lemma 34, we can rescale the edge weights w_e's so that $\lambda_2(L) \geq 1/\kappa$ and $\lambda_n(L) \leq 1$, where $\kappa = \Theta\left(dc/\phi^2\right)$. Specifically, we can achieve this by making $a := 1/(dc) \leq w_e \leq b := 1/d$. Obviously, this rescaling does not change the difficulty of estimating $R_{s,t}$. Also, in this case, we get $\Omega(1) \leq R_{s,t} \leq O(\kappa)$.

Next, we add an edge between s and t, and label it by 0, and set its weight to be 1. Let $G' = (V,E',w)$ be this modified graph, where $E' = E \cup \{0\}$, w_e remains the same for $e \in E$ and $w_0 = 1$ [7]. Now we orient each edge $e \in E$ arbitrarily, and orient the edge 0 as $t \to s$. For any object F (such as B, W, L, $E^+(v)$, $E^-(v)$, O_i) defined for G, let F' be the analog of F defined for G'. That is, for example, L' is the Laplacian of G', and O'_1, O'_2, O'_3 are the analog of O_1, O_2, O_3 for G', respectively. Note that O'_i can be implemented by calling O_i only $O(1)$ times, for $i \in [3]$

Let M' be the weighted incidence matrix of G' defined as follows:

$$M' := \sum_{e\in E'} |\psi_e\rangle\langle e|, \tag{4.34}$$

where

$$|\psi_e\rangle := \sqrt{w_e}\left(\left|e^+\right\rangle - \left|e^-\right\rangle\right), \quad \forall e \in E'. \tag{4.35}$$

In particular, we have

$$|\psi_0\rangle := \chi_{s,t} = |s\rangle - |t\rangle. \tag{4.36}$$

Note that

$$M' = B'W'^{1/2}. \tag{4.37}$$

Now consider $\mathrm{Ker}(M')$, which is a subspace of $\mathcal{H} := \mathrm{span}(|e\rangle : e \in E')$. One can easily see that

$$\mathrm{Ker}\left(M'\right) = \mathrm{span}\left(g(\mathbf{f}) : \ \mathbf{f} \text{ is an } s-t \text{ flow in } G\right) \tag{4.38}$$

where

$$g(\mathbf{f}) := |\mathbf{f}|\,|0\rangle + \sum_{e\in E} \frac{\mathbf{f}(e)}{\sqrt{w_e}}|e\rangle. \tag{4.39}$$

[7] If G already contains an edge between s and t, then G' will contain two edges between s and t. Although G' is in fact a multigraph in this case, we can still define its Laplacian, incidence matrix, etc. in the natural way.

Namely, the vectors in $\mathrm{Ker}(M')$ correspond to the *s-t* flows (of arbitrary values) in G [8]

Now let Π be the projection onto $\mathrm{Ker}(M')$. We claim that

Lemma 40.

$$\Pi|0\rangle \;\propto\; |\Phi\rangle := g(\mathbf{i}) = |0\rangle + \sum_{e\in E} \frac{\mathbf{i}(e)}{\sqrt{w_e}}|e\rangle\,, \tag{4.40}$$

where $\mathbf{i}$ *is the unit electrical s-t flow in G.*

Thus, if we prepare the state $|0\rangle$ and project it onto the subspace $\mathrm{Ker}(M')$, we would obtain a state corresponding exactly to the *electrical s-t* flow (of some value) in G!

We will give two proofs of Lemma 40. The first proof is analytical and more formal. The second one is geometric and more intuitive.

Proof 1 of Lemma 40. It is sufficient to show that for any $|\Psi\rangle \in \mathrm{Ker}(M')$, if $|\Psi\rangle \perp |\Phi\rangle$, then $|\Psi\rangle \perp |0\rangle$.

Let $|\Psi\rangle \in \mathrm{Ker}(M')$ be such that $|\Psi\rangle \not\perp |0\rangle$. After appropriate rescaling of $|\Psi\rangle$, we can assume

$$|\Psi\rangle = g(\mathbf{f}) = |0\rangle + \sum_{e\in E} \frac{\mathbf{f}(e)}{\sqrt{w_e}}|e\rangle \tag{4.41}$$

for some unit *s-t* flow $\mathbf{f}$ in G. Now, suppose

$$|\Psi\rangle \perp |\Phi\rangle = g(\mathbf{i}) = |0\rangle + \sum_{e\in E} \frac{\mathbf{i}(e)}{\sqrt{w_e}}|e\rangle\,. \tag{4.42}$$

It follows that

$$-1 = \sum_{e\in E} \frac{\mathbf{f}(e)\mathbf{i}(e)}{w_e}. \tag{4.43}$$

We will show that this implies the existence of some unit *s-t* flow $\mathbf{f}' \neq \mathbf{i}$ in G such that $\mathcal{E}(\mathbf{f}') < \mathcal{E}(\mathbf{i})$. But this is contradictory to the fact that $\mathbf{i}$ has the minimum energy among all unit *s-t* flows! So $|\Psi\rangle \not\perp |\Phi\rangle$.

Specifically, let $\mathbf{f}' = \beta\mathbf{f} + (1-\beta)\mathbf{i}$, where $\beta \in (0,1)$ is a parameter to be chosen later. Obviously, $\mathbf{f}'$ is a valid unit *s-t* flow for any β. Now consider the energy of $\mathbf{f}'$.

$$\begin{aligned}
\mathcal{E}(\mathbf{f}') &= \sum_{e\in E} \frac{(\beta\mathbf{f}(e) + (1-\beta)\mathbf{i}(e))^2}{w_e} && (4.44)\\
&= \beta^2 \sum_{e\in E} \frac{\mathbf{f}(e)^2}{w_e} + (1-\beta)^2 \sum_{e\in E} \frac{\mathbf{i}(e)^2}{w_e} + 2\beta(1-\beta)\sum_{e\in E} \frac{\mathbf{f}(e)\mathbf{i}(e)}{w_e} && (4.45)\\
&< \beta^2\mathcal{E}(\mathbf{f}) + (1-\beta)^2\mathcal{E}(\mathbf{i}) && (4.46)
\end{aligned}$$

[8] In particular, we allow *s-t* flows of value 0 in G. These flows are linear combinations of directed cycles in G.

Let $\gamma = \dfrac{\mathcal{E}(\mathbf{f})}{\mathcal{E}(\mathbf{i})}$. Then let $\beta = \dfrac{1}{1+\gamma}$. Then we get

$$\mathcal{E}(\mathbf{f}') < (\beta^2\gamma + (1-\beta)^2)\mathcal{E}(\mathbf{i}) = \frac{\gamma}{1+\gamma}\mathcal{E}(\mathbf{i}) < \mathcal{E}(\mathbf{i}), \tag{4.47}$$

as desired. □

Proof 2 of Lemma 40. Consider the geometric picture shown in Fig.4.2.

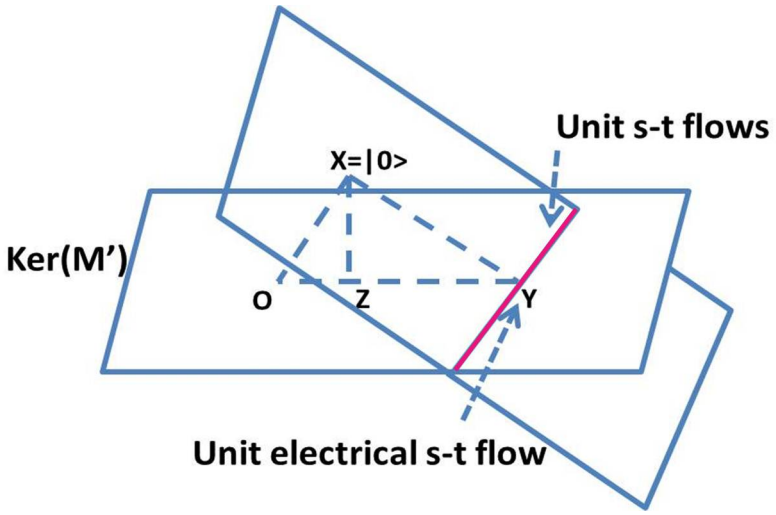

Figure 4.2: The geometrical picture for Proof 2 of Lemma 40.

Let O be the origin of $\mathcal{H} = \text{span}\,(|e\rangle : e \in E')$, and let X be the point such that $\overrightarrow{OX} = |0\rangle$. Then $L_1 := \text{Ker}\,(M')$ can be viewed as a hyperplane in the space $\mathcal{H}$. Moreover, let $L_2 = \{|\Psi\rangle : \ \langle 0|\Psi\rangle = 1\}$. Namely, for any $|\psi\rangle \in L_2$, we have $|\Psi\rangle = |0\rangle + |\Psi'\rangle$ for some vector $|\Psi'\rangle \perp |0\rangle$. So L_2 is the hyperplane which is orthogonal to the vector $\overrightarrow{OX}$ and touches the point X.

Consider $L_3 := L_1 \cap L_2$. One can easily see that

$$L_3 = \left\{ g(\mathbf{f}) = |0\rangle + \sum_{e\in E} \frac{\mathbf{f}(e)}{\sqrt{w_e}} |e\rangle : \ \mathbf{f} \text{ is a unit } s-t \text{ flow in } G \right\}. \tag{4.48}$$

Namely, the points in L_3 corresponds to the unit s-t flows in G. Furthermore, for any $Y' \in L_3$, if Y' corresponds to the s-t flow $\mathbf{f}$, then we have $\|\overrightarrow{OY'}\|^2 = 1 + \mathcal{E}(\mathbf{f})$ and $\|\overrightarrow{XY'}\|^2 = \mathcal{E}(\mathbf{f})$, because $\overrightarrow{OX} = |0\rangle \perp \overrightarrow{XY'}$.

Now let Z be the point in L_1 such that $\overrightarrow{OZ} = \Pi|0\rangle$. Let us extend the vector $\overrightarrow{OZ}$ until it meets L_3, and let Y be the intersection point. We claim that $\overrightarrow{OY} = g(\mathbf{i}) = |\Phi\rangle$. Namely, Y corresponds exactly to the unit electrical s-t flow $\mathbf{i}$.

Recall that Z is the unique point in L_1 such that $\|\overrightarrow{XZ}\|$ is minimized. For any $Y' \in L_3$, let Z' be the unique point on the line $\overline{OY'}$ that is closest to X. Then Z is the optimal Z' among all choices of Y'. Now, suppose Y' corresponds to unit s-t flow $\mathbf{f}$. Then we have $\|\overrightarrow{OX}\| = 1$, $\|\overrightarrow{OY'}\|^2 = 1 + \mathcal{E}(\mathbf{f})$, $\|\overrightarrow{XY'}\|^2 = \mathcal{E}(\mathbf{f})$. Since $\overrightarrow{XZ'} \perp \overrightarrow{OY'}$, we can get $\|\overrightarrow{XZ'}\| = \sqrt{\mathcal{E}(\mathbf{f})/(1+\mathcal{E}(\mathbf{f}))}$. In order to minimize $\|\overrightarrow{XZ'}\|$, we need to minimize $\mathcal{E}(\mathbf{f})$. So Y must correspond to $\mathbf{i}$, since it is the unit s-t flow with the minimum energy.

□

Now we describe our strategy for estimating $R_{s,t}$. Let $\left|\overline{\Phi}\right\rangle = |\Phi\rangle / \||\Phi\rangle\|$. Lemma 40 implies that there exists an orthonormal basis $\{|\Phi\rangle, |\Psi_1\rangle, \ldots, |\Psi_l\rangle\}$ for $\mathrm{Ker}(M')$ such that $|\Psi_i\rangle \perp |0\rangle$ for any i. Then, by

$$\Pi = \left|\overline{\Phi}\right\rangle\left\langle\overline{\Phi}\right| + \sum_i |\Psi_i\rangle\langle\Psi_i|, \tag{4.49}$$

we obtain

$$p := \langle 0|\Pi|0\rangle = |\left\langle 0|\overline{\Phi}\right\rangle|^2. \tag{4.50}$$

Meanwhile, note that

$$\||\Phi\rangle\|^2 = 1 + \sum_{e\in E} \frac{\mathbf{i}(e)^2}{w_e} = 1 + R_{s,t}. \tag{4.51}$$

So we have

$$|\left\langle 0|\overline{\Phi}\right\rangle|^2 = \frac{1}{1+R_{s,t}}. \tag{4.52}$$

Combining Eq.(4.50) and Eq.(4.52) yields

$$R_{s,t} = \frac{1}{p} - 1. \tag{4.53}$$

Thus, $R_{s,t}$ can be inferred from p. To get to know p, we perform the projective measurement $\{\Pi, I-\Pi\}$ on the state $|0\rangle$, and estimate the probability of seeing the outcome corresponding to Π. Note that by $\Omega(1) \le R_{s,t} \le O(\kappa)$, we have $\Omega(1/\kappa) \le p \le O(1)$. So, in order to estimate $R_{s,t}$ to a relative error ε, we need to estimate p to a relative error $O(\varepsilon)$.

Lemma 41. *The measurement $\{\Pi, I-\Pi\}$ can be implemented to accuracy $\delta > 0$ in $\tilde{O}(\mathrm{polylog}(1/\delta) \cdot \log n \cdot \min(dc, d^{3/2}c^{1/2})/\phi)$ time* [9].

Since this algorithm depends only poly-logarithmically on the inverse accuracy $1/\delta$, from now on we will assume that $\{\Pi, I-\Pi\}$ is implemented perfectly (taking the accuracy issue into account only increases the complexity of our algorithm by a poly-logarithmic factor).

Our algorithm works as follows:

1. We perform the measurement $\{\Pi, I-\Pi\}$ on the state $|0\rangle$ by using the algorithm in Lemma 41.

[9] Recall that we identify the measurement $\{\Pi, I-\Pi\}$ as the operation $\mathcal{E}(\rho) := |0\rangle\langle 0| \otimes \Pi\rho\Pi + |1\rangle\langle 1| \otimes (I-\Pi)\rho(I-\Pi)$.

2. We run amplitude estimation on step 1 to estimate $p = \langle 0|\Pi|0\rangle$. Since $p = \Omega(1/\kappa)$, and we want to estimate p to a relative error $O(\varepsilon)$, amplitude estimation requires $O\left(\sqrt{\kappa}/\varepsilon\right)$ repetitions of step 1 and its inverse.

3. Let $\tilde{p}$ be the estimate of p. Then we return $\tilde{R}_{s,t} := 1/\tilde{p} - 1$ as the estimate of $R_{s,t}$.

By Lemma 41, this algorithm has time complexity

$$\tilde{O}\left((\sqrt{\kappa}/\varepsilon)\cdot \log n \cdot \min\left(dc, d^{3/2}c^{1/2}\right)/\phi\right) = \tilde{O}\left(\log n \cdot \min\left(d^{3/2}c^{3/2}, d^2 c\right)/(\phi^2\varepsilon)\right), \tag{4.54}$$

(recall that $\kappa = \Theta\left(dc/\phi^2\right)$), as claimed.

□

4.6.1 Proof of Lemma 41

Proof of Lemma 41. To implement the measurement $\{\Pi, I-\Pi\}$, we generalize the previous method for evaluating the span program for *st*-connectivity [24]. Namely, we map $\mathrm{Ker}(M')$ to the -1 eigenspace of a quantum walk operator by using the spectral lemma.

Specifically, we will find two matrices A, B such that they have the same number of rows and each of them has orthonormal columns, and $M'' := A^\dagger B = YM'$ for some invertible matrix Y. It is easy to see that $\mathrm{Ker}(M'') = \mathrm{Ker}(M')$. Then, by Lemma 17, we know that the -1 eigenspace of $U(A,B) = \mathrm{Ref}_B \cdot \mathrm{Ref}_A$ is $\left(\mathcal{C}(A)\cap\mathcal{C}(B)^\perp\right)\oplus\left(\mathcal{C}(A)^\perp\cap\mathcal{C}(B)\right)$. Note that $\mathcal{C}(A)^\perp\cap\mathcal{C}(B) = B(\mathrm{Ker}(M'')) = B(\mathrm{Ker}(M'))$, and $\mathcal{C}(A)\cap\mathcal{C}(B)^\perp$ is orthogonal to $\mathcal{C}(B)$. This implies that, for any state $|\psi\rangle \in \mathcal{H} = \mathrm{span}(|e\rangle : e \in E')$, we can simulate the projection of $|\psi\rangle$ onto $\mathrm{Ker}(M')$ by using the projection of $B(|\psi\rangle)$ onto the -1 eigenspace of $U(A,B)$ (this simulation is valid since B is an isometry). Then, to project a state (in $B(\mathcal{H})$) onto the -1 eigenspace of $U(A,B)$, we run phase estimation on $U(A,B)$ and check whether the measured eigenvalue is close to -1 or not.

Now we describe A and B. Let $m = |E|$. We will find two set of unit vectors $\{|a_v\rangle : v \in V\} \subset \mathbb{R}^{m+1}$ and $\{|b_e\rangle : e \in E'\} \subset \mathbb{R}^n$ such that for any $v \in V$ and $e \in E'$,

$$\langle a_v|e\rangle\langle v|b_e\rangle = \begin{cases} \sqrt{\dfrac{w_e}{2\widetilde{\deg}'(v)}}, & \text{if } v = e^+, \\ -\sqrt{\dfrac{w_e}{2\widetilde{\deg}'(v)}}, & \text{if } v = e^-, \\ 0, & \text{otherwise.} \end{cases} \tag{4.55}$$

Then we let

$$\begin{aligned} A &:= \sum_{v\in V} |a_v\rangle|v\rangle\langle v|, \\ B &:= \sum_{e\in E'} |e\rangle|b_e\rangle\langle e|. \end{aligned} \tag{4.56}$$

This implies that

$$M'' = A^\dagger B = \frac{1}{\sqrt{2}}(D')^{-1/2}M', \tag{4.57}$$

where $D' = \mathrm{diag}\left(\widetilde{\deg}'(v)\right)_{v\in V}$ is invertible. So $\mathrm{Ker}\,(M'') = \mathrm{Ker}\,(M')$.

The $|a_v\rangle$'s and $|b_e\rangle$'s are defined as follows. For any $v \in V$, let

$$|a_v\rangle := \frac{1}{\sqrt{\widetilde{\deg}'(v)}} \sum_{e\in E'(v)} \sqrt{w_e}\,|e\rangle\,, \tag{4.58}$$

and for any $e \in E'$, let

$$|b_e\rangle := \frac{1}{\sqrt{2}}\left(\left|e^+\right\rangle - \left|e^-\right\rangle\right). \tag{4.59}$$

Obviously, the $|a_v\rangle$'s and $|b_e\rangle$'s are unit vectors. Also, one can check that they satisfy Eq.(4.55), as desired.

Now, to (effectively) implement the measurement $\{\Pi, I-\Pi\}$, we run phase estimation on $U(A,B) = \mathrm{Ref}_B \cdot \mathrm{Ref}_A$ starting with the state $B|0\rangle = |0\rangle|b_0\rangle$, and check the measured eigenvalue is $(\Delta/3)$-close to -1 or not, where Δ is the eigenvalue gap around -1 of $U(A,B)$. Hence, we need to run phase estimation with precision $O(\Delta)$. Let T_A and T_B be the time for implementing Ref_A and Ref_B respectively. Then the running time of this algorithm is $\tilde{O}\left((T_A+T_B)/\Delta\right)$.

Lemma 42. *The eigenvalue gap Δ around -1 of $U(A,B)$ is $\Omega\left(\frac{\phi}{\sqrt{dc}}\right)$.*

Lemma 43. *Ref_A can be implemented to accuracy $\delta > 0$ in $\tilde{O}\left(\log n \cdot \min\left(\sqrt{dc}, d\right) \cdot \mathrm{polylog}\left((1/\delta)\right)\right)$ time.*

Lemma 44. *Ref_B can be implemented in $\tilde{O}(\log n)$ time.*

It follows immediately from Lemma 42, Lemma 43 and Lemma 44 that our algorithm for implementing $\{\Pi, I-\Pi\}$ to accuracy $\delta > 0$ takes $\tilde{O}\left(\mathrm{polylog}\,(1/\delta)\cdot \log n \cdot \min\left(dc, d^{3/2}c^{1/2}\right)/\phi\right)$ time, as claimed. □

Proof of Lemma 42. Let Δ_1 be the singular value gap around 0 of $D(A,B) = A^\dagger B$, and let Δ_2 be the eigenvalue gap around 0 of $D(A,B)D(A,B)^\dagger$. Obviously $\Delta_1 = \sqrt{\Delta_2}$.

We claim that $\Delta = \Theta(\Delta_1)$ and $\Delta_2 = \Omega(1/\kappa)$. Then it follows that $\Delta = \Omega\left(1/\sqrt{\kappa}\right) = \Omega\left(\phi/\sqrt{dc}\right)$.

First, we prove $\Delta = \Theta(\Delta_1)$. Note that the eigenvalue closest to -1 of $U(A,B)$ is $e^{i(\pi\pm\gamma)}$ for some $\gamma = \Theta(\Delta)$ [10]. Then, by Lemma 17, the singular value closest to 0 of $D(A,B)$ is $\cos(\pi/2-\gamma/2) = \sin(\gamma/2) = \Theta(\Delta)$.

Next, we prove $\Delta_2 = \Omega(1/\kappa)$. Note that by Eq.(4.57), we have

$$D(A,B)D(A,B)^\dagger = \frac{1}{2}(D')^{-1/2}M'(M')^\dagger(D')^{-1/2} = \frac{1}{2}(D')^{-1/2}L'(D')^{-1/2} = \frac{1}{2}\mathcal{L}', \tag{4.60}$$

[10] Lemma 17 implies that the spectrum of $U(A,B)$ is symmetric about the real line.

where

$$L' = M'(M')^\dagger = |\psi_0\rangle\langle\psi_0| + L \succcurlyeq L. \tag{4.61}$$

Meanwhile, for any $v \in V$, we have $a = 1/(dc) \leq \widetilde{\deg}(v) \leq bd = 1$. Since $\widetilde{\deg}'(v) = \widetilde{\deg}(v) + 1$ for $v \in \{s,t\}$ and $\widetilde{\deg}'(v) = \widetilde{\deg}(v)$ for other v's, we obtain $\widetilde{\deg}(v) \leq \widetilde{\deg}'(v) \leq O(dc) \cdot \widetilde{\deg}(v)$. This implies

$$D \preccurlyeq D' \preccurlyeq O(dc) \cdot D. \tag{4.62}$$

Then, by Eqs.(4.61),(4.62) and Chegeer's inequality, we get

$$\lambda_2(\mathcal{L}') = \Omega\left(\frac{\lambda_2(\mathcal{L})}{dc}\right) = \Omega\left(\frac{\phi^2}{dc}\right) = \Omega\left(\frac{1}{\kappa}\right). \tag{4.63}$$

Hence, $\Delta_2 = \Omega(1/\kappa)$, as desired. □

Proof of Lemma 43. Let Q_1 be the unitary operation that maps $|0\rangle|v\rangle$ to $|a_v\rangle|v\rangle$ for $v \in V$, and let R_1 be the reflection about span$(|0\rangle|v\rangle : v \in V)$. Then

$$\mathrm{Ref}_A = Q_1 R_1 Q_1^\dagger. \tag{4.64}$$

Obviously, R_1 can be implemented in $\tilde{O}(\log n)$ time. So it remains to show that Q_1 can be implemented to accuracy $O(\delta)$ in the claimed time.

We give two methods for implementing Q_1:

- Method 1: Fix any $v \in V$. Consider the following procedure.

  1. Starting with $|v\rangle|0\rangle$, we prepare the state

  $$|v\rangle\left(\frac{1}{\sqrt{\deg'(v)}}\sum_{e\in E'(v)}|e\rangle\right) \tag{4.65}$$

  by calling O_1' and its inverse $O(\log n)$ times (we can determine $\deg'(v)$ exactly by calling O_1' and its inverse $\tilde{O}(\log n)$ times).

  2. We append a qubit in the state $|0\rangle$ and call O_3', obtaining the state

  $$|v\rangle\left(\frac{1}{\sqrt{\deg'(v)}}\sum_{e\in E'(v)}|e\rangle|w_e\rangle\right). \tag{4.66}$$

  3. We append another qubit in the state $|0\rangle$ and perform the controlled-rotation:

  $$|w_e\rangle|0\rangle \to |w_e\rangle\left(\sqrt{w_e}|1\rangle + \sqrt{1-w_e}|0\rangle\right). \tag{4.67}$$

  Note that since $w_e \leq 1$ for any $e \in E'$, this is a valid unitary operation.

4. Now, if we measure the last qubit, then conditioned on the outcome being 1, the rest of the state would be
$$|v\rangle \left(\frac{1}{\sqrt{\widetilde{\deg}'(v)}} \sum_{e \in E'(v)} \sqrt{w_e} |e\rangle |w_e\rangle \right). \tag{4.68}$$
Furthermore, since $w_e \geq 1/(dc)$ for any $e \in E'$, the probability of seeing outcome 1 is at least $1/(dc)$.
5. We uncompute $|w_e\rangle$ by calling the inverse of O'_3, obtaining the state
$$|v\rangle \left(\frac{1}{\sqrt{\widetilde{\deg}'(v)}} \sum_{e \in E'(v)} \sqrt{w_e} |e\rangle \right) |0\rangle = |v\rangle |a_v\rangle |0\rangle . \tag{4.69}$$
6. This procedure succeeds with probability $\Omega(1/(dc))$. By Corollary 11, we can raise this probability to $1 - O(\delta)$ by repeating this procedure and its inverse $\tilde{O}\left(\sqrt{dc}\log(1/\delta)\right)$ times. This would ensure that we have implemented O_1 to accuracy $O(\delta)$.

Clearly, this implementation of Q_1 takes $\tilde{O}\left(\sqrt{dc} \cdot \log n \cdot \log(1/\delta)\right)$ time.

- Method 2: We will give a circuit that maps $|v\rangle |0\rangle$ to $|v\rangle |\varphi_v\rangle$ such that $\| |a_v\rangle - |\varphi_v\rangle \| = O(\delta)$, for any $v \in V$. This would ensure that we have implemented Q_1 to accuracy $O(\delta)$.

  Pick an integer $M = \Theta\left(d/\delta^2\right)$. Fix any $v \in V$. Let $r = \deg'(v)$ and let $e_1, e_2, \ldots, e_r$ be the incident edges of v. Then, for any $i \in [r]$, let $S_{v,i} = \sum_{j=1}^{i} w_{e_j}$. Also, let $S_{v,0} = 0$. Note that $S_{v,r} = \widetilde{\deg}'(v)$.

  Next, for $0 \leq i \leq r$, let $M_{v,i} = \lceil MS_{v,i}/S_{v,r} \rceil$. Then, let $Z_{v,i} = M_{v,i} - M_{v,i-1}$, for $i \in [r]$. Note that $M_{v,r} = M = \sum_{i=1}^{r} Z_{v,i}$.

  Finally, we define
$$|\varphi_v\rangle = \sum_{i=1}^{r} \sqrt{\frac{Z_{v,i}}{M}} |e_i\rangle . \tag{4.70}$$
  Then, by construction, we have
$$\left| \frac{w_{e_i}}{\widetilde{\deg}'(v)} - \frac{Z_{v,i}}{M} \right| = O\left(\frac{1}{M}\right), \quad \forall i \in [r]. \tag{4.71}$$
  This implies that
$$\| |a_v\rangle - |\varphi_v\rangle \| = O\left(\sqrt{\frac{d}{M}}\right) = O(\delta), \tag{4.72}$$
  as desired.

Now we describe how to map $|v\rangle|0\rangle$ to $|v\rangle|\varphi_v\rangle$. For any $k \in [M]$, let $h_v(k) = (j,t)$ if $M_{v,j-1} < k \leq M_{v,j}$ and $k = M_{v,j-1} + t$. Note that for any $k \in [M]$, a unique (j,t) satisfies this condition. So the function h_i is well-defined. Consider the following procedure:

1. We map $|v\rangle|0\rangle$ to $|v\rangle|r\rangle$ by calling O'_1 and its inverse $\tilde{O}(\log n)$ times.
2. We create the state
$$|v\rangle|r\rangle\left(\bigotimes_{j=1}^{r}\left|w_{e_j}\right\rangle\right) \tag{4.73}$$
by calling O'_1, O'_3 and their inverses $\tilde{O}(d)$ times.
3. We compute the $M_{v,j}$'s for all $j \in [r]$, obtaining the state
$$|v\rangle|r\rangle\left(\bigotimes_{j=1}^{r}\left|w_{e_j}\right\rangle\right)\left(\bigotimes_{j=1}^{r}\left|M_{v,j}\right\rangle\right) \tag{4.74}$$
4. We append an ancilla register in the state $\frac{1}{\sqrt{M}}\sum_{k=1}^{M}|k\rangle$, obtaining the state
$$|v\rangle|r\rangle\left(\bigotimes_{j=1}^{r}\left|w_{e_j}\right\rangle\right)\left(\bigotimes_{j=1}^{r}\left|M_{v,j}\right\rangle\right)\left(\frac{1}{\sqrt{M}}\sum_{k=1}^{M}|k\rangle\right). \tag{4.75}$$
5. We compute $h_v(k)$ for each $k \in [M]$, obtaining the state
$$\begin{aligned} & |v\rangle|r\rangle\left(\bigotimes_{j=1}^{r}\left|w_{e_j}\right\rangle\right)\left(\bigotimes_{j=1}^{r}\left|M_{v,j}\right\rangle\right)\left(\frac{1}{\sqrt{M}}\sum_{k=1}^{M}|h_v(k)\rangle\right) \\ = \; & |v\rangle|r\rangle\left(\bigotimes_{j=1}^{r}\left|w_{e_j}\right\rangle\right)\left(\bigotimes_{j=1}^{r}\left|M_{v,j}\right\rangle\right)\left(\frac{1}{\sqrt{M}}\sum_{j=1}^{r}\sum_{t=1}^{Z_{v,j}}|j,t\rangle\right). \end{aligned} \tag{4.76}$$
6. We perform the unitary operation that maps $|j\rangle\left(\frac{1}{\sqrt{Z_{v,j}}}\sum_{t=1}^{Z_{v,j}}|t\rangle\right)$ to $|j\rangle|0\rangle$ on the last register, obtaining the state
$$|v\rangle|r\rangle\left(\bigotimes_{j=1}^{r}\left|w_{e_j}\right\rangle\right)\left(\bigotimes_{j=1}^{r}\left|M_{v,j}\right\rangle\right)\left(\sum_{j=1}^{r}\sqrt{\frac{Z_{v,j}}{M}}|j,0\rangle\right). \tag{4.77}$$
7. We convert this state into
$$|v\rangle|r\rangle\left(\bigotimes_{j=1}^{r}\left|w_{e_j}\right\rangle\right)\left(\bigotimes_{j=1}^{r}\left|M_{v,j}\right\rangle\right)\left(\sum_{j=1}^{r}\sqrt{\frac{Z_{v,j}}{M}}|e_j,0\rangle\right). \tag{4.78}$$
by calling O'_1 once.

8. We uncompute the $M_{v,j}$'s by undoing step 3, then uncompute the w_{e_j}'s by undoing step 2, and finally uncompute r by undoing step 1. Eventually, we obtain the state

$$|\varphi_v\rangle = |v\rangle \sqrt{\frac{Z_{v,j}}{M}} |e_j\rangle . \tag{4.79}$$

Overall, this circuit takes $\tilde{O}(d \cdot \log n \cdot \text{polylog}(M)) = \tilde{O}(d \cdot \log n \cdot \text{polylog}(1/\delta))$ time.

We may choose to use method 1 or 2 to implement Q_1 (depending on which of d and c is larger). So Q_1 can be implemented in the claimed time.

□

Proof of Lemma 44. Let Q_2 be the unitary operation that maps $|e\rangle|0\rangle$ to $|e\rangle|b_e\rangle$ for $e \in E'$, and let R_2 be the reflection about span$(|e\rangle|0\rangle : e \in E')$. Then

$$\text{Ref}_B = Q_2 R_2 Q_2^\dagger. \tag{4.80}$$

Obviously, R_2 can be implemented in $\tilde{O}(\log n)$ time. For Q_2, it can be implemented as follows: for any $e \in E'$, we do

$$\begin{aligned}
|e\rangle|0\rangle|0\rangle|0\rangle &\to |e\rangle|e^+\rangle|e^-\rangle|0\rangle && (4.81)\\
&\to |e\rangle|e^+\rangle|e^-\rangle \frac{1}{\sqrt{2}}(|e^+\rangle - |e^-\rangle) && (4.82)\\
&\to |e\rangle|0\rangle|0\rangle \frac{1}{\sqrt{2}}(|e^+\rangle - |e^-\rangle) && (4.83)\\
&= |e\rangle|0\rangle|0\rangle|b_e\rangle . && (4.84)
\end{aligned}$$

The first and last step can be done by calling O_2' and $(O_2')^{-1}$ respectively. The second step can be implemented in $\tilde{O}(\log n)$ time. So Q_2 can be implemented in $\tilde{O}(\log n)$ time. □

4.7 Generating Electrical Flows as Quantum States

The algorithm in Theorem 39 can be slightly modified to produce a quantum state approximately proportional to

$$|\mathbf{i}\rangle = \sum_{e \in E} \mathbf{i}(e)|e\rangle , \tag{4.85}$$

i.e. the unit electrical s-t flow in G. Recall that we have shown

$$\Pi|0\rangle \propto |\Phi\rangle = |0\rangle + \sum_{e \in E} \frac{\mathbf{i}(e)}{\sqrt{w_e}} |e\rangle . \tag{4.86}$$

The algorithm in Theorem 39 uses amplitude estimation to estimate $\|\Pi|0\rangle\|$ (from which $R_{s,t}$ can be inferred). Now, we use *amplitude amplification* to approximately obtain the state $|\overline{\Phi}\rangle$. Then, we can transform this state into a state close to $|\tilde{\mathbf{i}}\rangle$. Formally, we have the following result:

Theorem 45. *Let $G=(V,E,w)$ be a connected weighted graph given in the incidence list model. Let $s,t\in V$ be different, let $\mathbf{i}$ be the unit electrical s-t flow in G, and let $\varepsilon>0$. Then $\left|\bar{\mathbf{i}}\right\rangle$ can be prepared to accuracy $\varepsilon>0$ in $\tilde{O}\left(\log n\cdot\mathrm{polylog}\left(1/\varepsilon\right)\cdot\min\left(d^{3/2}c^2,d^2c^{3/2}\right)/\phi^2\right)$ time, where $n=|V|$, $d=\deg(G)$, $c=\max_{e\in E}w_e/\min_{e\in E}w_e$ and $\phi=\phi(G)$.*

Proof. We continue to use the notation introduced in the proof of Theorem 39. Also, we still rescale the edge weights to make $1/(dc)\leq w_e\leq 1/d$ and $\Omega(1)\leq R_{s,t}\leq O(\kappa)$, where $\kappa=\Theta\left(dc/\phi^2\right)$. This rescaling does not change the difficulty of preparing $\left|\bar{\mathbf{i}}\right\rangle$.

Lemma 46. *$\left|\overline{\Phi}\right\rangle$ can be prepared to accuracy $\delta>0$ in $\tilde{O}(\log n\cdot\mathrm{polylog}\left(1/\delta\right)\cdot\min\left(d^{3/2}c^{3/2},d^2c\right)/\phi^2)$ time.*

Proof. Consider the following algorithm:

1. We perform the measurement $\{\Pi,I-\Pi\}$ on the state $|0\rangle$ by using the algorithm in Lemma 41.

2. The probability of seeing the outcome corresponding to Π in step 1 is $\Omega(1/\kappa)$. By Corollary 11 we can raise this probability to $1-O(\delta)$ by repeating step 1 and its inverse $O\left(\sqrt{\kappa}\log(1/\delta)\right)$ times. This ensures that we obtain a state $O(\delta)$-close to $\left|\overline{\Phi}\right\rangle$.

By Lemma 41, this algorithm has running time

$$\tilde{O}\left(\sqrt{\kappa}\cdot\mathrm{polylog}\left(1/\delta\right)\cdot\log n\cdot\min\left(dc,d^2c\right)/\phi\right)=\tilde{O}\left(\log n\cdot\mathrm{polylog}\left(1/\delta\right)\cdot\min\left(d^{3/2}c^{3/2},d^2c\right)/\phi^2\right), \tag{4.87}$$

as claimed [11].

□

Since the algorithm in Lemma 46 depends only poly-logarithmically on the inverse error, from now on we assume that $\left|\overline{\Phi}\right\rangle$ can be prepared perfectly (taking the error issue into account only increases the complexity of our algorithm for preparing $\left|\bar{\mathbf{i}}\right\rangle$ by a poly-logarithmic factor).

Consider the following algorithm for preparing $\left|\bar{\mathbf{i}}\right\rangle$:

1. We use the algorithm in Lemma 46 to prepare the state

$$\left|\overline{\Phi}\right\rangle=\frac{1}{\sqrt{1+R_{s,t}}}\left(|0\rangle+\sum_{e\in E}\frac{\mathbf{i}(e)}{\sqrt{w_e}}|e\rangle\right). \tag{4.88}$$

[11] A subtle issue is that, this algorithm actually produces an approximation of $B\left|\overline{\Phi}\right\rangle$ instead of $\left|\overline{\Phi}\right\rangle$, since we are working in the space $B(\mathcal{H})$. But we can easily convert this state into an approximation of $\left|\overline{\Phi}\right\rangle$ by running Q_2^{-1} (in the proof of Lemma 44) which maps $|e\rangle|b_e\rangle$ to $|e\rangle|0\rangle$. This extra step takes only $\tilde{O}(\log n)$ time and hence is negligible.

2. We perform the measurement $\{|0\rangle\langle 0|, I - |0\rangle\langle 0|\}$ on this state. Then, conditioned on the outcome corresponding to $I - |0\rangle\langle 0|$, we would get the state

$$\left|\overline{\Phi}'\right\rangle := \frac{1}{\sqrt{R_{s,t}}} \left(\sum_{e \in E} \frac{\mathbf{i}(e)}{\sqrt{w_e}} |e\rangle \right) \tag{4.89}$$

Moreover, this outcome occurs with probability

$$p_1 := \frac{R_{s,t}}{1 + R_{s,t}} = \Omega(1), \tag{4.90}$$

since $R_{s,t} = \Omega(1)$.

3. We append a qubit in the state $|0\rangle$ and calling O_3 once, obtaining the state

$$\frac{1}{\sqrt{R_{s,t}}} \left(\sum_{e \in E} \frac{\mathbf{i}(e)}{\sqrt{w_e}} |e\rangle |w_e\rangle \right). \tag{4.91}$$

4. We append another qubit in the state $|0\rangle$ and perform the controlled-rotation:

$$|w_e\rangle |0\rangle \to |w_e\rangle \left(\sqrt{dw_e} |1\rangle + \sqrt{1 - dw_e} |0\rangle \right). \tag{4.92}$$

Note that this is a valid unitary operation since $1/c \leq dw_e \leq 1$, for any e. Then we measure the last qubit. Conditioned on seeing outcome 1, the rest of the state becomes proportional to

$$\sum_{e \in E} \mathbf{i}(e) |e\rangle |w_e\rangle. \tag{4.93}$$

Moreover, this outcome occurs with probability

$$p_2 = \frac{d}{R_{s,t}} \cdot \|\mathbf{i}\|^2 = \Omega(1/c), \tag{4.94}$$

since $dw_e \geq 1/c$ for any e.

5. We uncompute $|w_e\rangle$ by calling the inverse of O_3, and obtain the state $\left|\tilde{\mathbf{i}}\right\rangle$.

6. The above procedure succeeds with probability $p_1 p_2 = \Omega(1/c)$. By Corollary 11, we can enhance this probability to $1 - O(\varepsilon)$ by repeating steps 1-5 and their inverses $O(\sqrt{c}\log(1/\varepsilon))$ times. This ensures that we obtain a state $O(\varepsilon)$-close to $\left|\tilde{\mathbf{i}}\right\rangle$.

By Lemma 46, this algorithm takes $\tilde{O}\left(\log n \cdot \text{polylog}(1/\varepsilon) \cdot \min(d^{3/2}c^2, d^2c^{3/2})/\phi^2\right)$ time, as claimed.

□

Remark 47. *We can estimate $\|\mathbf{i}\|$ by combining the algorithm in Theorem 39 and a variant of the algorithm in Theorem 45. Specifically, we can estimate p_2 in the proof of Theorem 45 using amplitude estimation (instead of amplitude amplification). We can also estimate $R_{s,t}$ using the algorithm in Theorem 39. Then $\|\mathbf{i}\|$ can be inferred from them using Eq.(4.94). The running time of this algorithm is polynomial in $\log n$, d, c, $1/\phi$ and the inverse error.*

4.8 Lower Bound on the Complexity of Effective Resistance Estimation

So far, we have presented two quantum algorithms for approximating $R_{s,t}$ for any connected weighted graph $G = (V,E,w)$ and any $s,t \in V$. Both of them have time complexity polynomial in $\log n$, d, c, $1/\phi$ and $1/\varepsilon$, where $n = |V|$, $d = \deg(G)$, $c = \max_{e\in E} w_e / \min_{e\in E} w_e$, $\phi = \phi(G)$ and ε is the relative error. In particular, the polynomial dependence on the inverse conductance (i.e. $1/\phi$) is interesting. One may wonder whether this dependence is necessary. In other words, is there a quantum algorithm with $\mathrm{poly}(\log n, d, c, \log(1/\phi), 1/\varepsilon)$ time complexity for this task? If so, then we could estimate effective resistances super-fast in general and it might have some immediate implications to other problems. It turns out that no such algorithm exist, as implied by the following theorem:

Theorem 48. *For any $n \in \mathbb{N}$, there exists a connected weighted graph $G = (V,E,w)$ and two vertices $s,t \in V$ such that $|V| = 10n+2$, $\deg(G) = 3$, $w_e = 1$ for any $e \in E$, $\phi(G) = \Theta(1/n)$, and, assuming G is given in the incidence list model, any quantum algorithm that estimates $R_{s,t}$ to a relative error $1/10$ with probability at least $2/3$ must make $\Omega(n)$ queries.*

Proof. We will build a reduction from PARITY to effective resistance estimation. PARITY is the problem that, given an n-bit string $x = x_1 x_2 \ldots x_n$, determine the value of $\mathrm{PARITY}(x) := x_0 \oplus x_1 \oplus \ldots \oplus x_n$. We will map the string x to a graph $G(x)$ with two distinguished vertices s,t such that

- If $\mathrm{PARITY}(x) = 0$, then $R_{s,t} \le n$;
- If $\mathrm{PARITY}(x) = 1$, then $R_{s,t} \ge 4n$.

Hence, by estimating $R_{s,t}$ to a relative error $1/10$, we can distinguish these two cases and solve PARITY. Since PARITY has $\Omega(n)$ bounded-error quantum query complexity [18, 65], any quantum algorithm that estimates $R_{s,t}$ to a relative error $1/10$ with probability at least $2/3$ must make $\Omega(n)$ queries. We remark that similar constructions have been used to prove the lower bounds on the query complexity of sparse Hamiltonian simulation [30, 32].

Now we describe the construction of the graph $G(x)$. To build it, we start with $10n+2$ vertices, which are labelled by (i,a) for $i \in [n+1]$ and $a \in \{0,1\}$, and $[j,b]$ for $j \in [4n]$ and $b \in \{0,1\}$. Then we add the following edges, and set the weight of each edge to be 1:

- For any $i \in [n]$ and $a \in \{0,1\}$, we add an edge $((i,a),(i+1,a \oplus x_i))$. Namely, if $x_i = 0$, then we add the edges $((i,0),(i+1,0))$ and $((i,1),(i+1,1))$; otherwise, we add the edges $((i,0),(i+1,1))$ and $((i,1),(i+1,0))$.
- For any $j \in [4n-1]$ and $b \in \{0,1\}$, we add an edge $([j,b],[j+1,b])$. Moreover, we add the edges $((1,0),[1,0])$, $((1,0),[1,1])$, $([4n,0],(n+1,0))$ and $([4n,1],(n+1,1))$.

Namely, to create the graph $G(x)$, we concatenate n gadgets, each of which is of parallel-type or crossing-type depending on the value of x_i. Then, we add two "long" paths between $(1,0)$ and

$(n+1,0)$ and between $(1,0)$ and $(n+1,1)$, respectively. For example, Fig.4.3 shows the graph $G(x)$ for the string $x = 11010$. Note that by construction, $G(x)$ is always connected.

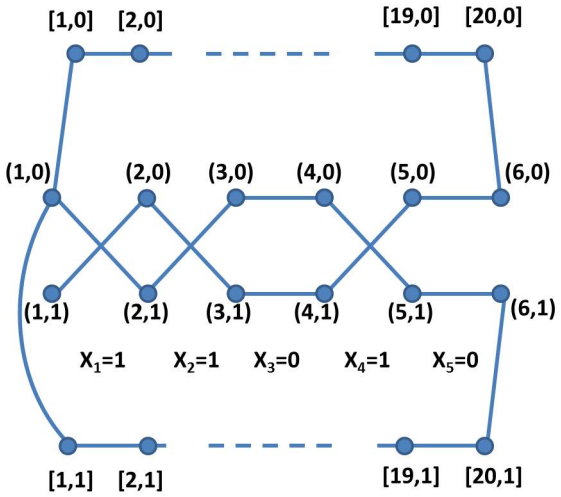

Figure 4.3: The graph $G(x)$ for the string $x = 11010$.

Now, let $s = (1,0)$ and $t = (n+1,0)$. Observe that if $\text{PARITY}(x) = 0$, then there is a path of length n from s to t that involves only the first kind of edges. That is, $s = (1,0) \to (2,x_1) \to (3,x_1 \oplus x_2) \to \cdots \to (n+1, \oplus_{i=1}^{n} x_i) = (n+1,0) = t$. On the other hand, if $\text{PARITY}(x) = 1$, this path would lead to $(n+1,1)$ instead. So if $\text{PARITY}(x) = 0$, there exist two disjoint paths between s and t, one with length n and another with length $4n+1$. This implies that $R_{s,t} \leq n$. On the other hand, if $\text{PARITY}(x) = 1$, then there is a unique path between s and t with length $4n+1$. Thus, we have $R_{s,t} \geq 4n$ in this case.

Clearly, we have $\deg(G(x)) = 3$. Moreover, we have $\phi(G(x)) = \Theta(1/n)$. To see this, consider the cut that separates the vertices $\{(i,a) : i \leq n/2, a \in \{0,1\}\} \cup \{[i,b] : i \leq 2n, b \in \{0,1\}\}$ from the rest. Obviously, this cut has conductance $O(1/n)$. Meanwhile, since $G(x)$ is connected and it has $\Theta(n)$ edges, we have $\phi(G(x)) = \Omega(1/n)$. Therefore, $G(x)$ satisfies all the desired conditions. This concludes the proof.

□

4.9 Discussion

As mentioned before, the theory of electrical networks has provided many valuable insights into classical computation. Its implication to quantum computation, however, remains mostly unclear. We do know that quantum query complexity can be upper bounded by some function of the effective resistance between certain vertices in the *learning graph* [20, 22]. Apart from this, not much is known. It is an exciting direction to explore the potential of electrical flows for designing

fast quantum algorithms. In particular, it would be interesting to know whether our algorithms for estimating effective resistances and generating electrical flows (as quantum states) can be utilized to achieve significant quantum speedup for other problems.

Many previous quantum algorithms for graph problems are based on searching (via quantum walk or amplitude amplification), and hence are *combinatorial* in spirit. Our algorithms, on the other hand, rely on analyzing the algebraic properties of the matrices associated with graphs, and hence are *algebraic* in nature. It would be interesting to devise more quantum algorithms in similar way. In particular, *spectral graph theory* [56] has been very useful for designing classical approximation algorithms for NP-complete problems (e.g. [33, 127]). It is worth studying whether they can help the development of quantum algorithms as well.

Chapter 5

Quantum Algorithms for Curve Fitting

5.1 Introduction

Curve fitting [13], also known as *regression analysis* in statistics, is the process of constructing a (simple continuous) curve that has the best fit to a series of data points (e.g. see Fig.5.1). It can be used to understand the relationships among two or more variables, or to infer the values of a function where no data are available. It also provides an efficient means of data compression. Therefore, it has found numerous applications in science, engineering and economics. How to quickly fit a large amount of data into a simple model has become an important problem for many tasks.

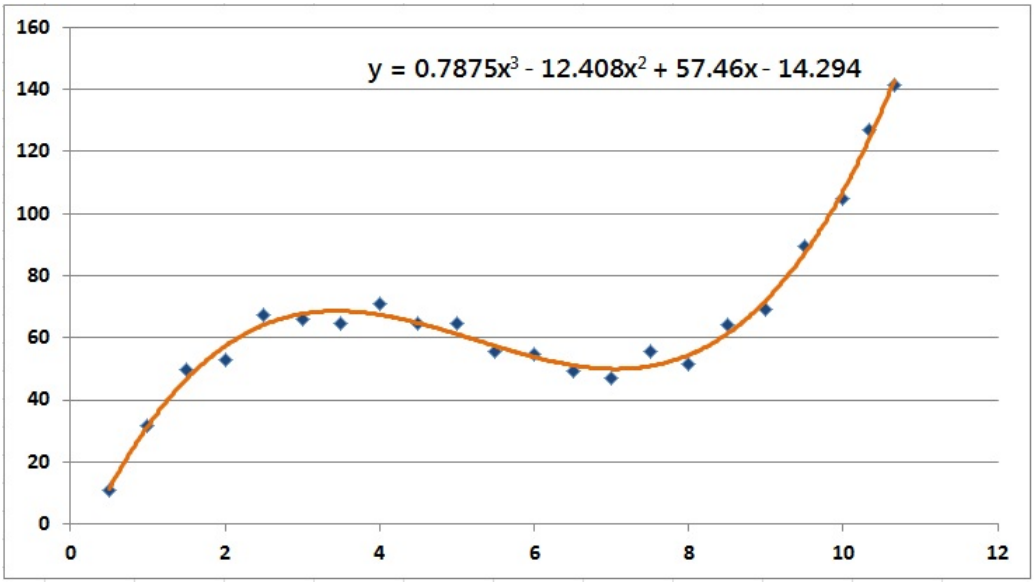

Figure 5.1: An example of curve fitting.

Least squares [37] is one of the most popular methods of curve fitting. This method minimizes the sum of squared residuals, where a residual is the difference between an observed value and the value predicted by a model. Formally, we need to find the parameter vector θ that minimizes the quantity $\|\mathbf{F}\theta-\mathbf{y}\|^2$, where $\mathbf{F}$ is the *design matrix*, and $\mathbf{y}$ is the *response vector*. The solution is known to be

$$\hat{\theta}=(\mathbf{F}^{\mathrm{T}}\mathbf{F})^{-1}\mathbf{F}^{\mathrm{T}}\mathbf{y}. \tag{5.1}$$

To compute this solution, the best classical algorithm needs to take $\mathrm{poly}(n,d)$ time, where n is the number of data points to be fitted, and d is the dimension of *feature vectors* (namely, $\mathbf{F}$ is an $n\times d$ matrix).

Recently, Harrow, Hassidim and Lloyd (HHL) [78] proposed a fast quantum algorithm for solving systems of linear equations (but in an unconventional sense). Later, building upon this work, Wiebe, Braun and Lloyd (WBL) [132] gave quantum algorithms for curve fitting in the case that $\mathbf{F}$ is sparse. Specifically, they gave a quantum algorithm for estimating the fit quality, and also a quantum algorithm for computing the best-fit parameters $\hat{\theta}$, but in an unconventional sense. Namely, under the assumption that there exists a high-quality fit [1] and the state proportional to $\mathbf{y}$ can be prepared in $\mathrm{polylog}(n)$ time, their algorithm produces a quantum state approximately proportional to $\hat{\theta}$ in $\mathrm{poly}(\log n,\log d,s,\kappa,1/\varepsilon)$ time, where s is the sparsity of $\mathbf{F}$, κ is the condition number of $\mathbf{F}$, and ε is the tolerable error. Then, by performing some measurements on this state and collecting the outcome statistics, one can learn certain properties of $\hat{\theta}$. However, since $\hat{\theta}$ is a d-dimensional vector, one cannot fully determine $\hat{\theta}$ in $\mathrm{polylog}(d)$ time. Thus, one cannot completely construct the fitted curve in $\mathrm{polylog}(d)$ time. It is worth mentioning that WBL's algorithms, like HHL's, rely on a combination of the techniques of phase estimation [87, 104] and sparse Hamiltonian simulation [4, 30, 53, 32, 43, 50, 53].

In this chapter, we present efficient quantum algorithms for estimating the best-fit parameters and the quality of least-square curve fitting in the general case. Namely, our algorithms work no matter $\mathbf{F}$ is sparse or not. The running times of our algorithms are polynomial in $\log n$, d, κ, ν, χ, $1/\Phi$ and $1/\varepsilon$, where n is the number of data points to be fitted, d is the dimension of feature vectors, κ is the condition number of the design matrix, ν and χ are some parameters reflecting the variances of the design matrix and response vector, Φ is the fit quality [2], and ε is the tolerable error. Our algorithms run very fast when the given data are normal, in the sense that $\mathbf{F}$ is far from being singular, and the rows of $\mathbf{F}$ and $\mathbf{y}$ do not vary too much in their norms. Meanwhile, it is unknown whether classical algorithms can solve this case very fast.

Our algorithms differ from WBL's algorithms in several aspects. First, our algorithms do produce the full classical description of $\hat{\theta}$ (not just a quantum state proportional to $\hat{\theta}$), so they completely determine the fitted curve. But on the other hand, our algorithms have running times poly-logarithmic in n, but not poly-logarithmic in d (as stated before, it is impossible to have time complexity poly-logarithmic in d in this case). Second, our algorithms use recent technique for nonsparse Hamiltonian simulation, so they can solve data fitting in the general case. Finally,

[1] The authors of [132] did not specify this condition explicitly. But it was implied by their assumption $\left\|\mathbf{F}^{\mathrm{T}}y\right\|=1$ in the description of their algorithms.

[2] The time complexity of the algorithm for estimating Φ does not depend on $1/\Phi$.

for estimating the best-fit parameters $\hat{\theta} = (\mathbf{F}^{\mathrm{T}}\mathbf{F})^{-1}\mathbf{F}^{\mathrm{T}}\mathbf{y}$, WBL's algorithm consists of two stages: stage 1 generates a state proportional to $z := \mathbf{F}^{\mathrm{T}}y$, and stage 2 generates a state proportional to $\hat{\theta} = (\mathbf{F}^{\mathrm{T}}\mathbf{F})^{-1}z$. Each stage is based on utilizing (a variant of) HHL's algorithm, which relies on analyzing the singular value decomposition of $\mathbf{F}$ or the spectral decomposition of $\mathbf{F}^{\mathrm{T}}\mathbf{F}$, respectively. We notice that these two decompositions are essentially the same. So these two stages can be carried out simultaneously. Thus, our algorithm consists of only one stage, and it generates the state proportional to $\hat{\theta}$ in one shot. This leads to a saving of running time for estimating $\hat{\theta}$.

Our algorithms are developed by combining phase estimation and recent technique for non-sparse Hamiltonian simulation. Specifically, Lloyd, Mohseni and Rebentrost [96] introduced a *density matrix exponentiation* technique, which allows us to simulate the unitary operation $e^{i\rho t}$ by consuming multiple copies of the state ρ. Then, by running phase estimation on the unitary operation $e^{i\rho}$ staring with the state ρ, we can analyze the eigenvalues and eigenvectors of ρ. They call this phenomenon *quantum self-analysis*. We utilize their results as follows. First, we prepare a state σ proportional to $\mathbf{F}\mathbf{F}^{\mathrm{T}}$. Then σ's eigenvalues are related to the singular values of $\mathbf{F}$, and its eigenvectors are the (left) singular vectors of $\mathbf{F}$. Then, the density matrix exponentiation technique allows us to implement $e^{i\sigma t}$ for any t. Next, by running phase estimation on $e^{i\sigma}$ starting with state $|\mathbf{y}\rangle$, we "effectively" break $|\mathbf{y}\rangle$ into several pieces, where each piece is either parallel to one of $\mathbf{F}$'s (left) singular vectors, or orthogonal to the column space of $\mathbf{F}$. Then, we can perform different quantum operations on each piece of $|\mathbf{y}\rangle$. This, along with some extra work, enables us to estimate the quality and best-fit parameters of least-square curve fitting.

5.2 Least-Square Curve Fitting

Given a set of n points $\{(x_{i,1}, x_{i,2}, \ldots, x_{i,k}, y_i)\}_{i=1}^{n}$ in $\mathbb{R}^{k+1}$, the goal of curve fitting is to find a simple continuous function that has the best fit to these data points. Formally, let $\mathbf{x}_i := (x_{i,1}, x_{i,2}, \ldots, x_{i,k})^T$, for $i \in [n]$. Also, let $f_j : \mathbb{R}^k \to \mathbb{R}$ be some simple function, for $j \in [d]$. Then we want to approximate y_i with a function of $\mathbf{x}_i$ of the form

$$f(\mathbf{x}, \theta) := \sum_{j=1}^{d} f_j(\mathbf{x})\theta_j, \tag{5.2}$$

where $\theta := (\theta_1, \theta_2, \ldots, \theta_d)^T$ are some parameters [3]. In the least-square approach, we find the optimal parameters $\hat{\theta}$ by minimizing the sum of squared residuals, i.e.

$$E := \sum_{i=1}^{n} |f(\mathbf{x}_i, \theta) - y_i|^2. \tag{5.3}$$

Now, let $\mathbf{F}$ be the $n \times d$ matrix such that $\mathbf{F}_{i,j} = f_j(\mathbf{x}_i)$, for $i \in [n]$ and $j \in [d]$. $\mathbf{F}$ is called the *design matrix*, and $\mathbf{F}_i := (f_1(\mathbf{x}_i), f_2(\mathbf{x}_i), \ldots, f_d(\mathbf{x}_i))^T$ is called the i-th *feature vector*, for $i \in [n]$. In addition, let $\mathbf{y} = (y_1, y_2, \ldots, y_n)^T$. $\mathbf{y}$ is called the *response vector*. Then one can see that

$$E = \|\mathbf{F}\theta - \mathbf{y}\|^2. \tag{5.4}$$

[3] The most common case is that each f_j is a monomial of $\mathbf{x}$, and hence f is a polynomial of $\mathbf{x}$.

Hence, the best-fit parameters $\hat{\theta}$ are given by

$$\hat{\theta} = \mathbf{F}^{+}\mathbf{y} = (\mathbf{F}^{\mathrm{T}}\mathbf{F})^{-1}\mathbf{F}^{\mathrm{T}}\mathbf{y}. \tag{5.5}$$

Correspondingly, the fitted values of $\mathbf{y}$ are

$$\hat{\mathbf{y}} = \mathbf{F}\hat{\theta} = \mathbf{F}(\mathbf{F}^{\mathrm{T}}\mathbf{F})^{-1}\mathbf{F}^{\mathrm{T}}\mathbf{y} = \Pi_{\mathbf{F}}\mathbf{y}, \tag{5.6}$$

and the residuals are

$$\hat{\varepsilon} = \mathbf{y} - \hat{\mathbf{y}} = (I - \mathbf{F}(\mathbf{F}^{\mathrm{T}}\mathbf{F})^{-1}\mathbf{F}^{\mathrm{T}})\mathbf{y} = (I - \Pi_{\mathbf{F}})\mathbf{y}. \tag{5.7}$$

Geometrically, $\hat{\mathbf{y}}$ is exactly the projection of $\mathbf{y}$ onto $\mathcal{C}(\mathbf{F})$. To measure the quality of this fit, we introduce the quantity

$$\Phi := \frac{\|\hat{\mathbf{y}}\|^2}{\|\mathbf{y}\|^2}. \tag{5.8}$$

Namely, Φ is the squared cosine of the angle between $\mathbf{y}$ and $\hat{\mathbf{y}}$. The larger Φ is, the better the fit is. Note that $\hat{\mathbf{y}} \perp \hat{\varepsilon}$ and hence

$$\hat{E} := \|\hat{\varepsilon}\|^2 = \left\|\mathbf{F}\hat{\theta} - y\right\|^2 = (1-\Phi)\|y\|^2. \tag{5.9}$$

We have assumed $\mathrm{rank}(\mathbf{F}) = d$ (and hence $\mathbf{F}^{\mathrm{T}}\mathbf{F}$ is invertible) in the above statement. This is a reasonable assumption, because if otherwise, either the f_j's are linearly dependent (e.g. $f_2 = 2f_1$), or we simply do not have enough data to do the fitting. In each case, a revision of $\mathbf{F}$ is required.

5.3 Our Model

We will study quantum algorithms for estimating the best-fit parameters $\hat{\theta}$ and the fit quality Φ, in the following model. We assume that $\mathbf{F}$ is given as a quantum oracle $O_{\mathbf{F}}$ defined as

$$O_{\mathbf{F}}|i\rangle|j\rangle|0\rangle = |i\rangle|j\rangle\left|\mathbf{F}_{i,j}\right\rangle, \quad \forall i \in [n], \ \forall j \in [d]. \tag{5.10}$$

Namely, $O_{\mathbf{F}}$ takes a row index i and column index j as input, and returns the value of $\mathbf{F}_{i,j}$. In addition, we assume that $\mathbf{y}$ is given as a quantum oracle $O_{\mathbf{y}}$ defined as

$$O_{\mathbf{y}}|i\rangle|0\rangle = |i\rangle|y_i\rangle, \quad \forall i \in [n]. \tag{5.11}$$

Namely, $O_{\mathbf{y}}$ takes a row index i as input, and returns the value of y_i. An algorithm in this model has access to $O_{\mathbf{F}}$, $O_{\mathbf{y}}$ and their inverses. Its query complexity is defined as the number of calls to $O_{\mathbf{F}}$, $O_{\mathbf{y}}$ and their inverses. Its time complexity is defined as its query complexity plus the number of additional one- and two-qubit gates used.

Without loss of generality, we will assume that $\mathrm{tr}\left(\mathbf{F}^{\mathrm{T}}\mathbf{F}\right) = \sum_{i=1}^{n}\sum_{j=1}^{d}|\mathbf{F}_{i,j}|^2 = 1$ and $\|\mathbf{y}\| = 1$. This can be achieved by scaling the original $\mathbf{F}$ and $\mathbf{y}$ appropriately. Clearly, this rescaling does not change the difficulty of estimating $\hat{\theta}$ or Φ.

5.4 Density Matrix Exponentiation

In this section, we review the *density matrix exponentiation* technique of [96]. This technique allows us to simulate the unitary operation $e^{i\rho t}$ by consuming multiple copies of the state ρ. It will be crucial for our algorithms.

Lemma 49 (Implicit in [96])**.** *Let ρ be a D-dimensional quantum state. Then $e^{i\rho t}$ can be simulated to accuracy $O(\varepsilon)$ with $O\left(t^2/\varepsilon\right)$ copies of ρ and $\tilde{O}(\log D \cdot t^2/\varepsilon)$ additional one- and two-qubit gates.*

Proof. The simulation method is based on the following observation. Let σ be any D-dimensional state. Then we have

$$\mathcal{E}_x(\sigma) := \mathrm{tr}_1\left(e^{iSx}\left(\rho \otimes \sigma\right) e^{-iSx}\right) = \sigma + ix[\rho,\sigma] + O\left(x^2\right), \tag{5.12}$$

where S is the swap operator, i.e. $S|i\rangle|j\rangle = |j\rangle|i\rangle$ for any $i, j \in [D]$. Meanwhile, we have

$$\mathcal{F}_x(\sigma) := e^{i\rho x}\,\sigma\, e^{-i\rho x} = \sigma + ix[\rho,\sigma] + O\left(x^2\right). \tag{5.13}$$

Therefore,

$$D(\mathcal{E}_x(\sigma), \mathcal{F}_x(\sigma)) = O\left(x^2\right). \tag{5.14}$$

In fact, one can check that for any state τ (with a D-dimensional subsystem),

$$D((\mathcal{E}_x \otimes I)(\tau), (\mathcal{F}_x \otimes I)(\tau)) = O\left(x^2\right). \tag{5.15}$$

This implies that

$$D(\mathcal{E}_x, \mathcal{F}_x) = O\left(x^2\right). \tag{5.16}$$

Now we use n repeated applications of $\mathcal{E}_{t/n}$ to simulate $e^{i\rho t}$. Since

$$D(\mathcal{E}_{t/n}^n, \mathcal{F}_{t/n}^n) = D(\mathcal{E}_{t/n}^n,\ e^{i\rho t}) = O\left(t^2/n\right), \tag{5.17}$$

in order to make $D(\mathcal{E}_{t/n}^n,\ e^{i\rho t}) = O(\varepsilon)$, it is sufficient to set $n = \Theta\left(t^2/\varepsilon\right)$. This algorithm consumes $n = O\left(t^2/\varepsilon\right)$ copies of ρ. Furthermore, it needs to implement $e^{iSt/n}$ once in each application of $\mathcal{E}_{t/n}$. As shown in [44], $e^{iSt/n}$ can be implemented in $\tilde{O}(\log D)$ time. Thus, this algorithm uses $O\left(t^2/\varepsilon\right)$ copies of ρ and $\tilde{O}(n \cdot \log D) = \tilde{O}(\log D \cdot t^2/\varepsilon)$ additional one- and two-qubit gates. □

The following lemma says that we do not need to prepare ρ exactly in order to simulate $e^{i\rho t}$ well. A good approximation of ρ would be sufficient.

Lemma 50. *Let ρ and ρ' be D-dimensional quantum states such that $\|\rho - \rho'\| = O(\varepsilon/t)$. Then $e^{i\rho t}$ can be simulated to accuracy $O(\varepsilon)$ with $O\left(t^2/\varepsilon\right)$ copies of ρ' and $\tilde{O}(\log D \cdot t^2/\varepsilon)$ additional one- and two-qubit gates.*

Proof. By Lemma 49, there exists a quantum algorithm that simulates $e^{i\rho' t}$ to accuracy ε using $O\left(t^2/\varepsilon\right)$ copies of ρ' and $\tilde{O}(\log D \cdot t^2/\varepsilon)$ additional two-qubit gates. So it is sufficient to prove that

$$D(e^{i\rho t}, e^{i\rho' t}) = O(\varepsilon). \tag{5.18}$$

Claim 51. *Let A and B be any two Hermitian matrices. Then $\left\|e^{iA} - e^{iB}\right\| = O(\|A-B\|)$.*

Proof. Let $C(x) = e^{iAx}e^{iB(1-x)}$, for $x \in [0,1]$. Then

$$e^{iA} - e^{iB} = \int_0^1 \frac{dC(x)}{dx} dx = \int_0^1 ie^{iAx}(A-B)e^{iB(1-x)} dx. \tag{5.19}$$

Thus,

$$\left\|e^{iA} - e^{iB}\right\| \leq \int_0^1 \left\|e^{iAx}(A-B)e^{iB(1-x)}\right\| dx \leq \int_0^1 \|A-B\| \, dx = \|A-B\|. \tag{5.20}$$

□

Claim 51 implies that

$$\left\|e^{i\rho t} - e^{i\rho' t}\right\| = O\left(\left\|\rho - \rho'\right\| t\right) = O(\varepsilon). \tag{5.21}$$

It follows that $D(e^{i\rho t}, e^{i\rho' t}) = O(\varepsilon)$, as desired.

□

5.4.1 Simulating $e^{i\mathbf{F}\mathbf{F}^{\mathrm{T}}t}$

Now we utilize the technique of density matrix exponentiation to simulate $e^{i\mathbf{F}\mathbf{F}^{\mathrm{T}}t}$ for any given t. Let

$$\sigma = \mathbf{F}\mathbf{F}^{\mathrm{T}}. \tag{5.22}$$

Then, $\sigma \succcurlyeq 0$ and $\mathrm{tr}(\sigma) = \mathrm{tr}\left(\mathbf{F}^{\mathrm{T}}\mathbf{F}\right) = 1$. So we can view σ as a density matrix. With the help of Lemma 50, we only need to consider how to (approximately) prepare the state σ.

Let

$$|\mathbf{F}\rangle = \sum_{i=1}^{n} |i\rangle |\mathbf{F}_i\rangle = \sum_{i=1}^{n}\sum_{j=1}^{d} \mathbf{F}_{i,j} |i\rangle |j\rangle, \tag{5.23}$$

where $|\mathbf{F}_i\rangle = \sum_{j=1}^{d} \mathbf{F}_{i,j} |j\rangle$. Then, since $\||\mathbf{F}\rangle\|^2 = \sum_{i=1}\sum_{j=1}^{d} |\mathbf{F}_{i,j}|^2 = 1$, $|\mathbf{F}\rangle$ is a normalized quantum state. Furthermore, the reduced state of $|\mathbf{F}\rangle$ on the first subsystem is $\sigma = \mathbf{F}\mathbf{F}^{\mathrm{T}}$. This implies that, to (approximately) produce the state σ, it is sufficient to (approximately) produce the state $|\mathbf{F}\rangle$.

Lemma 52. *Suppose $\alpha \leq \|\mathbf{F}_i\| \leq \beta$ for any $i \in [n]$. Let $\nu = \beta/\alpha$. Then $|\mathbf{F}\rangle$ can be prepared to accuracy $\delta > 0$ in $\tilde{O}(\mathrm{polylog}(n) \cdot \nu d \log(1/\delta))$ time.*

Proof. Let $U_{\mathbf{F}}$ be the unitary operation defined as

$$U_{\mathbf{F}} |i\rangle |0\rangle = |i\rangle |\|\mathbf{F}_i\|\rangle, \quad \forall i \in [n]. \tag{5.24}$$

Clearly, $U_{\mathbf{F}}$ can be implemented in $O(d)$ time (since we can query $\mathbf{F}_{i,1}, \mathbf{F}_{i,2}, \ldots, \mathbf{F}_{i,d}$ and compute $\|\mathbf{F}_i\|$ from them).

Next, let $V_{\mathbf{F}}$ be the unitary operation defined as

$$V_{\mathbf{F}} |i\rangle |0\rangle = |i\rangle |\bar{\mathbf{F}}_i\rangle, \quad \forall i \in [n]. \tag{5.25}$$

(Recall that by definition $\bar{\mathbf{F}}_i = \mathbf{F}_i / \|\mathbf{F}_i\|$.) We have that:

Claim 53. *$V_{\mathbf{F}}$ can be implemented to accuracy $\gamma > 0$ in $\tilde{O}(d \cdot \mathrm{polylog}(1/\gamma))$ time.*

Proof. We will describe a quantum circuit that maps $|i\rangle |0\rangle$ to $|i\rangle |\phi_i\rangle$ such that $\|\phi_i - \bar{\mathbf{F}}_i\| = O(\gamma)$, for any $i \in [n]$. This ensures that this circuit implements $V_{\mathbf{F}}$ to accuracy $O(\gamma)$, as desired

Pick an integer $M = \Theta(d/\gamma^2)$. Fix any $i \in [n]$. Let $S_{i,j} = \sum_{l=1}^{j} \mathbf{F}_{i,j}^2$, for $j \in [d]$. Note that $S_{i,d} = \|\mathbf{F}_i\|^2$. Also, let $S_{i,0} = 0$. Then, let $M_{i,j} = \lceil M S_{i,j} / S_{i,d} \rceil$, for $0 \le j \le d$. Then let $Z_{i,j} = M_{i,j} - M_{i,j-1}$, for $j \in [d]$. Finally, let

$$|\phi_i\rangle = \sum_{j=1}^{d} \phi_{i,j} |j\rangle = \sum_{j=1}^{d} \mathrm{sgn}(\mathbf{F}_{i,j}) \sqrt{\frac{Z_{i,j}}{M}} |j\rangle. \tag{5.26}$$

Then, by construction, we have

$$|\phi_{i,j} - \bar{\mathbf{F}}_{i,j}| = O\left(\frac{1}{\sqrt{M}}\right) = O\left(\frac{\gamma}{\sqrt{d}}\right), \quad \forall j \in [d], \tag{5.27}$$

where $\bar{\mathbf{F}}_{i,j} = \mathbf{F}_{i,j} / \|\mathbf{F}_i\|$. It follows that $\|\phi_i - \bar{\mathbf{F}}_i\| = O(\gamma)$, as claimed.

Now we describe how to map $|i\rangle |0\rangle$ to $|i\rangle |\phi_i\rangle$. For any $k \in [M]$, let $h_i(k) = (j,t)$ if $M_{i,j-1} < k \le M_{i,j}$ and $k = M_{i,j-1} + t$. Note that for any $k \in [M]$, a unique (j,t) satisfies this condition. So the function h_i is well-defined. Consider the following procedure:

1. We create the state $|i\rangle (\bigotimes_{j=1}^{d} |\mathbf{F}_{i,j}\rangle)$ by using $O(d)$ queries to $O_{\mathbf{F}}$.

2. We compute the $M_{i,j}$'s for $j \in [d]$, obtaining the state

$$|i\rangle (\bigotimes_{j=1}^{d} |\mathbf{F}_{i,j}\rangle)(\bigotimes_{j=1}^{d} |M_{i,j}\rangle). \tag{5.28}$$

3. We append a register in the state $\frac{1}{\sqrt{M}} \sum_{k=1}^{M} |k\rangle$, obtaining the state

$$|i\rangle (\bigotimes_{j=1}^{d} |\mathbf{F}_{i,j}\rangle)(\bigotimes_{j=1}^{d} |M_{i,j}\rangle) \left(\frac{1}{\sqrt{M}} \sum_{k=1}^{M} |k\rangle\right) \tag{5.29}$$

4. We compute $h_i(k)$ for each $k \in [M]$, obtaining the state

$$\begin{aligned} & |i\rangle \left(\bigotimes_{j=1}^{d} \left|\mathbf{F}_{i,j}\right\rangle\right)\left(\bigotimes_{j=1}^{d} \left|M_{i,j}\right\rangle\right) \left(\frac{1}{\sqrt{M}} \sum_{k=1}^{M} |h_i(k)\rangle\right) \\ = \quad & |i\rangle \left(\bigotimes_{j=1}^{d} \left|\mathbf{F}_{i,j}\right\rangle\right)\left(\bigotimes_{j=1}^{d} \left|M_{i,j}\right\rangle\right) \left(\frac{1}{\sqrt{M}} \sum_{j=1}^{d} \sum_{t=1}^{Z_{i,j}} |j,t\rangle\right) \end{aligned} \tag{5.30}$$

5. We perform the unitary operation that maps $|j\rangle \left(\frac{1}{\sqrt{Z_{i,j}}} \sum_{t=1}^{Z_{i,j}} |t\rangle\right)$ to $|j\rangle |0\rangle$ on the last register, obtaining the state

$$|i\rangle \left(\bigotimes_{j=1}^{d} \left|\mathbf{F}_{i,j}\right\rangle\right)\left(\bigotimes_{j=1}^{d} \left|M_{i,j}\right\rangle\right) \left(\sum_{j=1}^{d} \sqrt{\frac{Z_{i,j}}{M}} |j,0\rangle\right) \tag{5.31}$$

6. We multiply the phase of each term by the sign of $\mathbf{F}_{i,j}$, obtaining the state

$$|i\rangle \left(\bigotimes_{j=1}^{d} \left|\mathbf{F}_{i,j}\right\rangle\right)\left(\bigotimes_{j=1}^{d} \left|M_{i,j}\right\rangle\right) \left(\sum_{j=1}^{d} \operatorname{sgn}\left(\mathbf{F}_{i,j}\right) \sqrt{\frac{Z_{i,j}}{M}} |j,0\rangle\right) \tag{5.32}$$

7. We uncompute the $M_{i,j}$'s by undoing step 2, then uncompute the $\mathbf{F}_{i,j}$'s by undoing step 1. Eventually, we obtain the sttae

$$|i\rangle |\phi_i\rangle = |i\rangle \left(\sum_{j=1}^{d} \operatorname{sgn}\left(\mathbf{F}_{i,j}\right) \sqrt{\frac{Z_{i,j}}{M}} |j\rangle\right) \tag{5.33}$$

   as desired.

Clearly, this algorithm runs in $\tilde{O}(d \cdot \operatorname{polylog}(M)) = \tilde{O}(d \cdot \operatorname{polylog}(1/\gamma))$ time, as claimed.

□

Note that the time complexity of implementing $V_{\mathbf{F}}$ is only poly-logarithmic in the inverse accuracy. So from now on, we assume that $V_{\mathbf{F}}$ is implemented perfectly (taking the accuracy issue into account only increases the time complexities of our algorithms by a poly-logarithmic factor).

Now consider the following algorithm for preparing $|\mathbf{F}\rangle$:

1. We prepare the state $\frac{1}{\sqrt{n}} \sum_{i=1}^{n} |i\rangle |0\rangle |0\rangle$, and convert it into the state

$$\frac{1}{\sqrt{n}} \sum_{i=1}^{n} |i\rangle \left|\bar{\mathbf{F}}_i\right\rangle |\|\mathbf{F}_i\|\rangle \tag{5.34}$$

   by calling $V_{\mathbf{F}}$ and $U_{\mathbf{F}}$ once.

2. We append a qubit in the state $|0\rangle$, and perform the controlled-rotation

$$|z\rangle|0\rangle \to |z\rangle\left(z\beta^{-1}|1\rangle + \sqrt{1-z^2\beta^{-2}}|0\rangle\right) \tag{5.35}$$

on the last two registers (recall that $\|\mathbf{F}_i\| \leq \beta$), obtaining the state

$$\frac{1}{\sqrt{n}}\sum_{i=1}^{n}|i\rangle\left|\bar{\mathbf{F}}_i\right\rangle|\|\mathbf{F}_i\|\rangle\left(\|\mathbf{F}_i\|\beta^{-1}|1\rangle + \sqrt{1-\|\mathbf{F}_i\|^2\beta^{-2}}|0\rangle\right). \tag{5.36}$$

3. We measure the last qubit in the standard basis. Then conditioned on seeing outcome 1, the rest of the state becomes proportional to

$$\sum_{i=1}^{n}|i\rangle\|\mathbf{F}_i\|\left|\bar{\mathbf{F}}_i\right\rangle|\|\mathbf{F}_i\|\rangle = \sum_{i=1}^{n}|i\rangle|\mathbf{F}_i\rangle|\|\mathbf{F}_i\|\rangle. \tag{5.37}$$

Furthermore, since $\|\mathbf{F}_i\| \geq \alpha = \beta/\nu$, the probability of seeing outcome 1 is $\Omega\left(1/\nu^2\right)$.

4. We uncompute the $|\|\mathbf{F}_i\|\rangle$ by performing the inverse of $U_{\mathbf{F}}$ on the first and third registers, obtaining the state $|\mathbf{F}\rangle = \sum_{i=1}^{n}|i\rangle|\mathbf{F}_i\rangle$.

5. The above procedure succeeds only with probability $\Omega\left(1/\nu^2\right)$. We use amplitude amplification to raise this probability to $1 - O(\delta)$. This ensures that we have prepared $|\mathbf{F}\rangle$ to accuracy $O(\delta)$. By Corollary 11, this requires $\tilde{O}(\nu\log(1/\delta))$ repetitions of the above procedure and its inverse.

Clearly, this algorithm has time complexity $\tilde{O}(\text{polylog}(n)\cdot\nu d\log(1/\delta))$, as claimed.

□

Lemma 52 immediately implies:

Lemma 54. *Suppose* $\alpha \leq \|\mathbf{F}_i\| \leq \beta$ *for any* $i \in [n]$. *Let* $\nu = \beta/\alpha$. *Then* σ *can be prepared to accuracy* $\delta > 0$ *in* $\tilde{O}(\text{polylog}(n)\cdot\nu d\log(1/\delta))$ *time.*

Combining Lemma 54 and Lemma 50, we obtain:

Lemma 55. *Suppose* $\alpha \leq \|\mathbf{F}_i\| \leq \beta$ *for any* $i \in [n]$. *Let* $\nu = \beta/\alpha$. *Then* $e^{i\sigma t}$ *can be simulated to accuracy* $\delta > 0$ *in* $\tilde{O}(\text{polylog}(n)\cdot\nu dt^2/\delta)$ *time.*

Proof. We use the algorithm in Lemma 54 to prepare σ to accuracy $O(\delta/t)$. Then we use the algorithm in Lemma 50 to simulate $e^{i\sigma t}$ to accuracy $O(\delta)$. By Lemma 54 and Lemma 50, this algorithm has time complexity $\tilde{O}(\text{polylog}(n)\cdot\nu dt^2/\delta)$, as claimed. □

5.5 Quantum Algorithms for Estimating the Best-Fit Parameters $\hat{\theta}$

In this section, we present two quantum algorithms for the estimation of $\hat{\theta} = (\mathbf{F}^{\mathrm{T}}\mathbf{F})^{-1}\mathbf{F}^{\mathrm{T}}\mathbf{y}$. The first algorithm produces an estimate of $\left\|\hat{\theta}\right\|$ (i.e. the norm of $\hat{\theta}$), and the second one produces an estimate of $\bar{\theta} := \hat{\theta}/\left\|\hat{\theta}\right\|$ (i.e. the normalized version of $\hat{\theta}$). Then, by multiplying them together, we obtain an estimate of $\hat{\theta} = \left\|\hat{\theta}\right\| \cdot \bar{\theta}$.

Before describing these algorithms, it is beneficial to consider the singular value decomposition of $\mathbf{F}$ and write $\hat{\theta}$ as the linear combination of the (right) singular vectors of $\mathbf{F}$. Suppose $\mathbf{F}$ has the singular value decomposition

$$\mathbf{F} = \sum_{j=1}^{d} s_j \left|u_j\right\rangle\left\langle v_j\right|, \tag{5.38}$$

where $s_1 \leq s_2 \leq \cdots \leq s_d$ are the singular values of $\mathbf{F}$. Then we have

$$\mathrm{tr}\left(\mathbf{F}^{\mathrm{T}}\mathbf{F}\right) = \sum_{j=1}^{d} s_j^2 = 1, \tag{5.39}$$

Let $\kappa \geq \kappa(\mathbf{F}) = s_d/s_1$. Then

$$\frac{1}{\kappa\sqrt{d}} \leq s_1 \leq s_2 \leq \cdots \leq s_d \leq \frac{\kappa}{\sqrt{d}}. \tag{5.40}$$

Meanwhile, we have

$$\sigma = \mathbf{F}\mathbf{F}^{\mathrm{T}} = \sum_{j=1}^{d} s_j^2 \left|u_j\right\rangle\left\langle u_j\right|. \tag{5.41}$$

This implies that

$$\mathcal{C}(\sigma) = \mathcal{C}(\mathbf{F}) = \mathrm{span}\left(\left\{\left|u_1\right\rangle, \left|u_2\right\rangle, \ldots, \left|u_d\right\rangle\right\}\right). \tag{5.42}$$

Therefore, the 1-eigenspace of $e^{i\sigma}$ is exactly $\mathrm{Ker}(\mathbf{F})$.

Now suppose $\left|\hat{\mathbf{y}}\right\rangle = \sum_{j=1}^{d} \alpha_j \left|u_j\right\rangle$. Then, by $\|\mathbf{y}\| = 1$, we get

$$\Phi = \left\|\hat{\mathbf{y}}\right\|^2 = \sum_{j=1}^{d} \alpha_j^2. \tag{5.43}$$

Furthermore, we have

$$\left|\hat{\theta}\right\rangle = (\mathbf{F}^{\mathrm{T}}\mathbf{F})^{-1}\mathbf{F}^{\mathrm{T}}\left|\mathbf{y}\right\rangle = \sum_{j=1}^{d} \alpha_j s_j^{-1} \left|v_j\right\rangle, \tag{5.44}$$

which implies that

$$\left\|\hat{\theta}\right\|^2 = \sum_{j=1}^{d} \alpha_j^2 s_j^{-2}. \tag{5.45}$$

Then it follows from Eq.(5.40) and Eq.(5.43) that

$$\frac{d\Phi}{\kappa^2} \leq \left\|\hat{\theta}\right\|^2 \leq d\Phi\kappa^2. \tag{5.46}$$

The following lemma gives an upper bound on the time complexity of preparing the state $|\mathbf{y}\rangle$, and will be useful for our algorithms.

Lemma 56. *Suppose* $\eta \leq |y_i| \leq \zeta$, *for any* $i \in [n]$. *Let* $\chi = \zeta/\eta$. *Then* $|\mathbf{y}\rangle$ *can be prepared to accuracy* $\delta > 0$ *in* $\tilde{O}(\text{polylog}(n) \cdot \chi \log(1/\delta))$ *time.*

Proof. Consider the following algorithm:

1. We prepare the state $\frac{1}{\sqrt{n}} \sum_{i=1}^{n} |i\rangle |0\rangle$ and call $O_{\mathbf{y}}$ once, obtaining the state

$$\frac{1}{\sqrt{n}} \sum_{i=1}^{n} |i\rangle |y_i\rangle . \tag{5.47}$$

2. We append a qubit in the state $|0\rangle$, and perform the controlled-rotation

$$|z\rangle |0\rangle \to |z\rangle \left(z\zeta^{-1} |1\rangle + \sqrt{1 - z^2\zeta^{-2}} |0\rangle \right), \tag{5.48}$$

   on the last two registers (recall that $|y_i| \leq \zeta$), obtaining the state

$$\frac{1}{\sqrt{n}} \sum_{i=1}^{n} |i\rangle |y_i\rangle \left(y_i\zeta^{-1} |1\rangle + \sqrt{1 - y_i^2\zeta^{-2}} |0\rangle \right). \tag{5.49}$$

3. We measure the last qubit in the standard basis. Then conditioned on seeing outcome 1, the rest of the state is proportional to

$$\sum_{i=1}^{n} y_i |i\rangle |y_i\rangle \tag{5.50}$$

   Furthermore, since $|y_i| \geq \eta = \zeta/\chi$, the probability of seeing this outcome is $\Omega\left(1/\chi^2\right)$.

4. We uncompute the $|y_i\rangle$ by uncalling $O_{\mathbf{F}}$, obtaining the state $|\mathbf{y}\rangle = \sum_{i=1}^{n} y_i |i\rangle$.

5. The above procedure succeeds only with probability $\Omega\left(1/\chi^2\right)$. We use amplitude amplification to raise this probability to $1 - O(\delta)$. This ensures that we have prepared $|\mathbf{y}\rangle$ to accuracy $O(\delta)$. By Corollary 11, this requires $\tilde{O}(\chi \log(1/\delta))$ repetitions of steps 1-4 and their inverses.

Clearly, this algorithm has time complexity $\tilde{O}(\text{polylog}(n) \cdot \chi \log(1/\delta))$, as claimed.

□

Note that the time complexity of preparing $|\mathbf{y}\rangle$ is only poly-logarithmic in the inverse accuracy. So from now on, we assume that $|\mathbf{y}\rangle$ is prepared perfectly (taking the accuracy issue into account only increases the time complexities of our algorithms by some poly-logarithmic factors).

5.5.1 Quantum Algorithm for Estimating $\left\|\hat{\theta}\right\|$

Theorem 57. *Suppose $\alpha \leq \|\mathbf{F}_i\| \leq \beta$, for any $i \in [n]$, and $\|\mathbf{F}^+\| \leq 1/a$, $\|\mathbf{F}\| \leq b$. Moreover, suppose $\eta \leq |y_i| \leq \zeta$, for any $i \in [n]$. Let $\nu = \beta/\alpha$, $\chi = \zeta/\eta$ and $\kappa = b/a$. Then $\left\|\hat{\theta}\right\|$ can be estimated to a relative error $\varepsilon > 0$ with probability at least $2/3$ in $\tilde{O}(\mathrm{polylog}(n) \cdot \kappa(\chi + \nu d^3\kappa^6/(\varepsilon^3\Phi))/(\varepsilon\sqrt{\Phi}))$ quantum time.*

Proof. Consider the following algorithm (for convenience, we assume that phase estimation is perfect in the following description, and we will take the error of phase estimation into account later):

1. We use the algorithm in Lemma 56 to prepare the state $|\mathbf{y}\rangle$.
2. We run phase estimation on $e^{i\sigma}$ starting with $|\mathbf{y}\rangle$, obtaining the state
$$\sum_{j=1}^{d} \alpha_j \left|u_j\right\rangle \left|s_j^2\right\rangle + |\hat{\varepsilon}\rangle |0\rangle . \tag{5.51}$$
3. We append a qubit in the state $|0\rangle$ and perform the controlled-rotation
$$|z\rangle |0\rangle \to |z\rangle \left(\frac{a}{\sqrt{z}} |1\rangle + \sqrt{1 - \frac{a^2}{z}} |0\rangle \right), \tag{5.52}$$
on the last two registers (note that $s_j \geq a$), obtaining a state proportional to
$$\sum_{j=1}^{d} \alpha_j \left|u_j\right\rangle \left|s_j^2\right\rangle \left(\frac{a}{s_j} |1\rangle + \sqrt{1 - \frac{a^2}{s_j^2}} |0\rangle \right) + |\hat{\varepsilon}\rangle |0\rangle , \tag{5.53}$$
4. We measure the last qubit in the standard basis. Then, conditioned on seeing outcome 1, the rest of the state becomes proportional to
$$\sum_{j=1}^{d} \alpha_j s_j^{-1} \left|u_j\right\rangle \left|s_j^2\right\rangle . \tag{5.54}$$
Furthermore, the probability of getting outcome 1 is
$$q := a^2 \sum_{j=1}^{d} \alpha_j^2 s_j^{-2} = a^2 \left\|\hat{\theta}\right\|^2 . \tag{5.55}$$
Since $\sum_{j=1}^{d} \alpha_j^2 = \Phi$, and $s_j \leq b = a\kappa$, we have $q = \Omega\left(\Phi/\kappa^2\right)$.
5. We use amplitude estimation to estimate q to a relative error $O(\varepsilon)$ with probability at least $2/3$. Since $q = \Omega\left(\Phi/\kappa^2\right)$, this requires $\tilde{O}(\kappa/(\varepsilon\sqrt{\Phi}))$ repetitions of the above procedure and its inverse.

6. Let q' be the estimate of q. Then we return $\sqrt{q'}/a$ as the estimate of $\left\|\hat{\theta}\right\|$.

Now we take the error of phase estimation into account, and analyze the time complexity of this algorithm. In step 2, we do not get the eigenphase s_j^2 exactly, but instead get some (random) $\lambda_j \approx s_j^2$ (although phase estimation singles out the eigenphase 0 perfectly). This implies that we only obtain the states in steps 2-4 approximately. Since we want to estimate q to a relative error $O(\varepsilon)$, we need to make sure that $\left|\lambda_j - s_j^2\right| = O\left(\varepsilon s_j^2\right)$. Then, by $s_j^2 \geq 1/(d\kappa^2)$, we need to set the precision of phase estimation to be $O\left(\varepsilon/(d\kappa^2)\right)$. It follows that we need to simulate $e^{i\sigma t}$ to accuracy $O\left(\varepsilon\Phi/\kappa^2\right)$ [4] for $t = O\left(d\kappa^2/\varepsilon\right)$ during phase estimation. This can be done in $\tilde{O}(\text{polylog}(n)\cdot\nu d(d\kappa^2/\varepsilon)^2/(\varepsilon\Phi/\kappa^2)) = \tilde{O}(\text{polylog}(n)\cdot\nu d^3\kappa^6/(\varepsilon^3\Phi))$ time by Lemma 55. Meanwhile, $|\mathbf{y}\rangle$ can be prepared in $\tilde{O}(\text{polylog}(n)\cdot\chi)$ time by Lemma 56. Therefore, one iteration of steps 1-3 takes $\tilde{O}(\text{polylog}(n)\cdot(\chi+\nu d^3\kappa^6/(\varepsilon^3\Phi)))$ time. Since amplitude estimation requires $\tilde{O}(\kappa/(\varepsilon\sqrt{\Phi}))$ repetitions of steps 1-4 and their inverses, this algorithm takes $\tilde{O}(\text{polylog}(n)\cdot\kappa(\chi+\nu d^3\kappa^6/(\varepsilon^3\Phi))/(\varepsilon\sqrt{\Phi}))$ time, as claimed.

□

Remark 58. *We can reduce the failure probability of the algorithm in Theorem 57 to arbitrarily small $\delta > 0$ by repeating this algorithm $O(\log(1/\delta))$ times and taking the median of the estimates obtained.*

5.5.2 Quantum Algorithm for Estimating $\bar{\theta}$

Suppose $\bar{\theta} = (\bar{\theta}_1, \bar{\theta}_2, \ldots, \bar{\theta}_d)^T$. Our algorithm for estimating $\bar{\theta}$ consists of two parts. The first part estimates $|\bar{\theta}_1|$, $|\bar{\theta}_2|$, …, $|\bar{\theta}_d|$ (i.e. the norm of each entry). The second part determines $\text{sgn}\left(\bar{\theta}_1\right)$, $\text{sgn}\left(\bar{\theta}_2\right)$, …, $\text{sgn}\left(\bar{\theta}_d\right)$ (i.e. the sign of each entry). Both parts depend on the following algorithm for producing the state $\left|\bar{\theta}\right\rangle$.

Proposition 59. *Suppose $\alpha \leq \|\mathbf{F}_i\| \leq \beta$, for any $i \in [n]$, and $\|\mathbf{F}^+\| \leq 1/a$, $\|\mathbf{F}\| \leq b$. Moreover, suppose $\eta \leq |y_i| \leq \zeta$, for any $i \in [n]$. Let $\nu = \beta/\alpha$, $\chi = \zeta/\eta$ and $\kappa = b/a$. Then $\left|\bar{\theta}\right\rangle$ can be prepared to accuracy $\varepsilon > 0$ in $\tilde{O}(\text{polylog}(n)\cdot\kappa(\chi+\nu d^3\kappa^6/(\varepsilon^3\Phi))/\sqrt{\Phi})$ time.*

Before proving this proposition, let us recall the singular value decomposition of $\mathbf{F}$ as shown in Eq.(5.38). Although $|v_1\rangle$, $|v_2\rangle$, …, $|v_d\rangle$ are d-dimensional vectors, we from now on consider them as n-dimensional vectors (that is, we embed $\mathbb{R}^d$ into $\mathbb{R}^n$ in the natural way). Now let

$$\begin{aligned} \tau^+ &= \sum_{j=1}^{d} s_j^2 \left|w_j^+\right\rangle\left\langle w_j^+\right|, \\ \tau^- &= \sum_{j=1}^{d} s_j^2 \left|w_j^-\right\rangle\left\langle w_j^-\right|, \end{aligned} \tag{5.56}$$

[4] We want the error caused by the imperfection of simulating $e^{i\sigma t}$ to be $O(\varepsilon q)$.

where

$$\begin{aligned}\left|w_j^+\right\rangle &= \frac{1}{\sqrt{2}}\left(|0\rangle\left|v_j\right\rangle + |1\rangle\left|u_j\right\rangle\right),\\ \left|w_j^-\right\rangle &= \frac{1}{\sqrt{2}}\left(|0\rangle\left|v_j\right\rangle - |1\rangle\left|u_j\right\rangle\right).\end{aligned} \tag{5.57}$$

Both τ^+ and τ^- are $2n$-dimensional quantum states. The following lemma says that they can be prepared quickly:

Lemma 60. *Suppose $\alpha \leq \|\mathbf{F}_i\| \leq \beta$, for any $i \in [n]$. Let $\nu = \beta/\alpha$. Then $\tau^\pm$ can be prepared to accuracy $\delta > 0$ in $\tilde{O}(\text{polylog}(n) \cdot \nu d \log(1/\delta))$ time.*

Proof. Consider the following algorithm:

1. We use the algorithm in Lemma 52 to prepare the state

$$|\mathbf{F}\rangle = \sum_{i=1}^{n}\sum_{j=1}^{d}\mathbf{F}_{i,j}|i\rangle|j\rangle = \sum_{j=1}^{d} s_j\left|u_j\right\rangle\left|v_j\right\rangle, \tag{5.58}$$

   where in the second step we perform the Schmidt decomposition of $|\mathbf{F}\rangle$, which corresponds to the singular value decomposition of $\mathbf{F}$.

2. We append a qubit in the state $|\pm\rangle = \frac{1}{\sqrt{2}}(|0\rangle \pm |1\rangle)$ and an n-dimensional register in the state $|0\rangle$, and perform the "addressing" operation

$$|i_1\rangle|i_0\rangle|j\rangle|0\rangle \to |i_1\rangle|i_0\rangle|j\rangle\left|i_j\right\rangle, \quad \forall i_0, i_1 \in [n], \ \ \forall j \in \{0,1\}. \tag{5.59}$$

   Then we obtain the state

$$\frac{1}{\sqrt{2}}\sum_{j=1}^{d} s_j\left|u_j\right\rangle\left|v_j\right\rangle\left(|0\rangle\left|v_j\right\rangle \pm |1\rangle\left|u_j\right\rangle\right) = \sum_{j=1}^{d} s_j\left|u_j\right\rangle\left|v_j\right\rangle\left|w_j^\pm\right\rangle. \tag{5.60}$$

   Then the reduced state of this state on the last register is $\tau^\pm = \sum_{j=1}^{d} s_j^2\left|w_j^\pm\right\rangle\left\langle w_j^\pm\right|$, as desired.

By Lemma 52, this algorithm has time complexity $\tilde{O}(\text{polylog}(n) \cdot \nu d \log(1/\delta))$, as claimed. □

Combining Lemma 50 and Lemma 60, we obtain:

Lemma 61. *Suppose $\alpha \leq \|\mathbf{F}_i\| \leq \beta$, for any $i \in [n]$. Let $\nu = \beta/\alpha$. Then $e^{i\tau^\pm t}$ can be simulated to accuracy $\delta > 0$ in $\tilde{O}(\text{polylog}(n) \cdot \nu d t^2/\delta)$ time.*

Proof. We use the algorithm in Lemma 60 to prepare $\tau^\pm$ to accuracy $O(\delta/t)$. Then we use the algorithm in Lemma 50 to simulate $e^{i\tau^\pm t}$ to accuracy $O(\delta)$. It follows from Lemma 60 and Lemma 50 that this algorithm has time complexity $\tilde{O}(\text{polylog}(n) \cdot \nu d t^2/\delta)$, as claimed. □

Now, let

$$\tau = \tau^+ - \tau^- = \sum_{j=1}^{d} s_j^2 \left(\left| w_j^+ \right\rangle \left\langle w_j^+ \right| - \left| w_j^- \right\rangle \left\langle w_j^- \right| \right). \tag{5.61}$$

Namely, τ has eigenvalues $\pm s_j^2$'s and eigenvectors $\left| w_j^{\pm} \right\rangle$'s. We can simulate $e^{i\tau t}$ by composing the simulations of $e^{i\tau^+ t}$ and $e^{i\tau^- t}$:

Lemma 62. *Suppose $\alpha \leq \|\mathbf{F}_i\| \leq \beta$, for any $i \in [n]$. Let $\nu = \beta/\alpha$. Then $e^{i\tau t}$ can be simulated to accuracy $\delta > 0$ in $\tilde{O}(\mathrm{polylog}(n) \cdot \nu d t^2/\delta)$ time.*

Proof. We use Suzuki's method [124] for simulating $e^{i(A+B)t}$, where A, B are arbitrary Hermitian matrices satisfying $\|A\|, \|B\| \leq 1$. This method works as follows. Define a function $S_{2k}(x)$ recursively: let

$$S_2(x) = e^{iAx/2} e^{iBx} e^{iAx/2}, \tag{5.62}$$

and let

$$S_{2k}(x) = [S_{2k-2}(p_k x)]^2 S_{2k-2}((1-4p_k)x)[S_{2k-2}(p_k x)]^2 \tag{5.63}$$

where $p_k = (4 - 4^{1/(2k-1)})^{-1}$ for any $k \geq 2$. Then we have:

Claim 63 ([124])**.** *For any $k \in \mathbb{N}$,*

$$\left\| e^{i(A+B)x} - S_{2k}(x) \right\| = O\left(|x|^{2k+1} \right). \tag{5.64}$$

This implies that for any $k \in \mathbb{N}$,

$$\left\| e^{i(A+B)t} - \left(S_{2k}\left(\frac{t}{n} \right) \right)^n \right\| = O\left(\frac{t^{2k+1}}{n^{2k}} \right). \tag{5.65}$$

To make the right-hand side $O(\delta)$, we need to set $n = \Theta\left(t^{1+\frac{1}{2k}} \delta^{-\frac{1}{2k}} \right)$. Then, $(S_{2k}(t/n))^n$ is the product of $O(n) = O\left(t^{1+\frac{1}{2k}} \delta^{-\frac{1}{2k}} \right)$ terms, where each term is of the form e^{iAt_j} or e^{iBt_j} for some $t_j = O(t/n) = O\left(\delta^{\frac{1}{2k}} t^{-\frac{1}{2k}} \right)$.

Now we simulate $e^{i\tau t}$ by setting $A = \tau^+$ and $B = -\tau^-$. We need to implement each term, which is of the form $e^{i\tau^+ t_j}$ or $e^{-i\tau^- t_j}$ for $t_j = O\left(\delta^{\frac{1}{2k}} t^{-\frac{1}{2k}} \right)$, to accuracy $O(\delta/n) = O\left(\delta^{1+\frac{1}{2k}} t^{-1-\frac{1}{2k}} \right)$. By Lemma 62, this takes $\tilde{O}(\mathrm{polylog}(n) \cdot \nu d t^{1-\frac{1}{2k}} / \delta^{1-\frac{1}{2k}})$ time. Since there are totally $O\left(t^{1+\frac{1}{2k}} \delta^{-\frac{1}{2k}} \right)$ terms, this algorithm has time complexity $\tilde{O}(\mathrm{polylog}(n) \cdot \nu d t^2/\delta)$, as claimed. (It is interesting that this complexity is independent of k. But a better way to simulate $e^{i\rho T}$ using multiple copies of ρ might change this fact.)

□

Now we have all the ingredients to prove Proposition 59:

Proof of Proposition 59. Suppose $|\hat{\mathbf{y}}\rangle = \sum_{j=1}^{d} \alpha_j |u_j\rangle$, where $\sum_{j=1}^{d} \alpha_j^2 = \Phi$. Then we have

$$|1\rangle|\hat{\mathbf{y}}\rangle = \sum_{j=1}^{d} \alpha_j |1\rangle |u_j\rangle = \frac{1}{\sqrt{2}} \sum_{j=1}^{d} \alpha_j \left(\left|w_j^+\right\rangle - \left|w_j^-\right\rangle \right). \tag{5.66}$$

Consider the following algorithm for preparing $\left|\bar{\theta}\right\rangle$ (again, we assume that phase estimation is perfect in the following description, and we will take the error of phase estimation into account later):

1. We prepare the state $|1\rangle|\mathbf{y}\rangle$ [5] by using the algorithm in Lemma 56.

2. We run phase estimation on $e^{i\tau}$ starting with $|1\rangle|\mathbf{y}\rangle$, obtaining the state

$$\frac{1}{\sqrt{2}} \sum_{j=1}^{d} \alpha_j \left(\left|w_j^+\right\rangle |s_j^2\rangle - \left|w_j^-\right\rangle |-s_j^2\rangle \right) + |1\rangle|\hat{\varepsilon}\rangle|0\rangle. \tag{5.67}$$

3. We perform the measurement $\{|0\rangle\langle 0|, I - |0\rangle\langle 0|\}$ on the last register. Then, conditioned on seeing the outcome corresponding to $I - |0\rangle\langle 0|$, the state becomes proportional to

$$\frac{1}{\sqrt{2}} \sum_{j=1}^{d} \alpha_j \left(\left|w_j^+\right\rangle |s_j^2\rangle - \left|w_j^-\right\rangle |-s_j^2\rangle \right) \tag{5.68}$$

   Furthermore, the probability of seeing this outcome is $\Phi = \sum_{j=1}^{d} \alpha_j^2$.

4. We append a qubit in the state $|0\rangle$, and perform the controlled-rotation

$$|z\rangle|0\rangle \to |z\rangle \left(\frac{a|z|^{1/2}}{z} |1\rangle + \sqrt{1 - \frac{a^2|z|}{z^2}} |0\rangle \right), \tag{5.69}$$

   on the last two registers (note that $s_j \geq a$), obtaining a state proportional to

$$\begin{aligned} & \frac{1}{\sqrt{2}} \sum_{j=1}^{d} \alpha_j \Big[\left|w_j^+\right\rangle \left|s_j^2\right\rangle \left(a s_j^{-1} |1\rangle + \sqrt{1 - a^2 s_j^{-2}} |0\rangle \right) \\ - & \left|w_j^-\right\rangle \left|-s_j^2\right\rangle \left(-a s_j^{-1} |1\rangle + \sqrt{1 - a^2 s_j^{-2}} |0\rangle \right) \Big] \end{aligned} \tag{5.70}$$

5. We measure the last qubit in the standard basis. Then, conditioned on seeing outcome 1, the rest of the state is proportional to

$$\frac{1}{\sqrt{2}} \sum_{j=1}^{d} \alpha_j \left(s_j^{-1} \left|w_j^+\right\rangle |s_j^2\rangle + s_j^{-1} \left|w_j^-\right\rangle |-s_j^2\rangle \right) \tag{5.71}$$

   Furthermore, since $s_j \leq b = \kappa a$, the probability of seeing outcome 1 is $\Omega\left(1/\kappa^2\right)$.

[5] The dimension of the first register is 2.

6. We uncompute the $\left|s_j^2\right\rangle$'s and $\left|-s_j^2\right\rangle$'s by undoing phase estimation, obtaining a state proportional to

$$\frac{1}{\sqrt{2}}\sum_{j=1}^{d}\alpha_j s_j^{-1}\left(\left|w_j^+\right\rangle+\left|w_j^-\right\rangle\right)=\sum_{j=1}^{d}\alpha_j s_j^{-1}\left|0\right\rangle\left|v_j\right\rangle=\left|0\right\rangle\left|\hat{\theta}\right\rangle. \tag{5.72}$$

   The reduced state of this state on the second register is $\left|\bar{\theta}\right\rangle$, as desired.

7. The above procedure only succeeds with probability $q:=\Omega\left(\Phi/\kappa^2\right)$. We using amplitude amplification to raise this probability to $1-O(\varepsilon)$. This ensures that we have prepared $\left|\bar{\theta}\right\rangle$ to accuracy $O(\varepsilon)$. By Corollary 11, this requires $\tilde{O}(\kappa\log(1/\varepsilon)/\sqrt{\Phi})$ repetitions of steps 1-6 and their inverses.

Now we take the error of phase estimation into account, and analyze the time complexity of this algorithm. In step 2, we do not get the eigenphase $\pm s_j^2$ exactly, but instead get some (random) $\lambda_j\approx\pm s_j^2$ (although phase estimation singles out the eigenphase 0 perfectly). This implies that we only obtain the states in steps 2-6 approximately. Since we want to prepare $\left|\bar{\theta}\right\rangle$ to accuracy $O(\varepsilon)$, we need to make sure that $|\lambda_j-s_j^2|=O\left(\varepsilon s_j^2\right)$. Since $s_j^2\geq 1/(d\kappa^2)$, we need to set the precision of phase estimation to be $O\left(\varepsilon/(d\kappa^2)\right)$. This implies that we need to simulate $e^{i\tau t}$ to accuracy $O\left(\varepsilon\Phi/\kappa^2\right)$ [6] for $t=O\left(d\kappa^2/\varepsilon\right)$ during phase estimation. Then by Lemma 62, this takes $\tilde{O}(\text{polylog}(n)\cdot\nu d(d\kappa^2/\varepsilon)^2/(\varepsilon\Phi/\kappa^2))=\tilde{O}(\text{polylog}(n)\cdot\nu d^3\kappa^6/(\varepsilon^3\Phi))$ time. Meanwhile, by Lemma 56, it takes $\tilde{O}(\text{polylog}(n)\cdot\chi)$ time to prepare $|\mathbf{y}\rangle$. Thus, one iteration of steps 1-6 takes $\tilde{O}(\text{polylog}(n)\cdot(\chi+\nu d^3\kappa^6/(\varepsilon^3\Phi)))$ time. Since amplitude amplification requires $\tilde{O}(\kappa\log(1/\varepsilon)/\sqrt{\Phi})$ repetitions of steps 1-6 and their inverses, this algorithm has time complexity $\tilde{O}(\text{polylog}(n)\cdot\kappa(\chi+\nu d^3\kappa^6/(\varepsilon^3\Phi))/\sqrt{\Phi})$, as claimed.

□

Remark 64. *The final state of the algorithm in Proposition 59 is of the form*

$$|\Psi\rangle:=\sqrt{p}\,|1\rangle\,|0\rangle\left|\bar{\theta}'\right\rangle|\phi\rangle+\sqrt{1-p}(|1\rangle\,|1\rangle\,|\tilde{\varphi}\rangle+|0\rangle\,|0\rangle\,|\tilde{\psi}_0\rangle+|0\rangle\,|1\rangle\,|\tilde{\psi}_1\rangle) \tag{5.73}$$

where $p=1-O(\varepsilon)$, $\bar{\theta}'=(\bar{\theta}'_1,\bar{\theta}'_2,\ldots,\bar{\theta}'_d)^T$ *satisfies* $\left\|\bar{\theta}'\right\|=1$ *and* $\left|\bar{\theta}'_j-\bar{\theta}_j\right|=O(\varepsilon)$, *and* $|\phi\rangle$ *is some normalized state, and* $|\tilde{\varphi}\rangle$, $|\tilde{\varphi}_0\rangle$, $|\tilde{\varphi}_1\rangle$ *are some unnormalized states* [7]. *This fact will be useful for the following algorithms for estimating* $|\bar{\theta}_j|$ *and* $\text{sgn}\left(\bar{\theta}_j\right)$.

Proposition 65. *Suppose* $\alpha\leq\|\mathbf{F}_i\|\leq\beta$, *for any* $i\in[n]$, *and* $\|\mathbf{F}^+\|\leq 1/a$, $\|\mathbf{F}\|\leq b$. *Moreover, suppose* $\eta\leq|y_i|\leq\zeta$, *for any* $i\in[n]$. *Let* $\nu=\beta/\alpha$, $\chi=\zeta/\eta$ *and* $\kappa=b/a$. *Then for any* $j\in[d]$, $\left|\bar{\theta}_j\right|$ *can be estimated up to an additive error* $\varepsilon>0$ *with probability at least* $2/3$ *in* $\tilde{O}(\text{polylog}(n)\cdot\kappa(\chi+\nu d^3\kappa^6/(\varepsilon^3\Phi))/(\varepsilon^2\sqrt{\Phi}))$ *quantum time.*

[6] We want the error caused by the imperfection of simulating $e^{i\tau t}$ at most $O(\varepsilon q)$.

[7] The dimensions of $|\tilde{\varphi}\rangle$, $|\tilde{\varphi}_0\rangle$ and $|\tilde{\varphi}_1\rangle$ are d times of the dimension of $|\phi\rangle$.

Proof. Consider the following algorithm:

1. We run the algorithm in Proposition 59 to get the state $|\Psi\rangle$ in Eq.(5.73).
2. We measure the first three registers of $|\Psi\rangle$ in the standard basis. Then the probability of seeing outcome $(1,0,j)$ is $q_j := p\left|\bar{\theta}'_j\right|^2$.
3. We use amplitude estimation to estimate q_j up to an additive error $O\left(\varepsilon^2\right)$ with probability at least $2/3$. This requires $\tilde{O}(1/\varepsilon^2)$ repetitions of the above procedure and its inverse.
4. Let q'_j be the estimate of q_j. Then we return $\sqrt{q'_j}$ as the estimate of $\left|\bar{\theta}_j\right|$.

Now we prove the correctness of this algorithm. Since $p = 1 - O(\varepsilon)$ and $\left|\bar{\theta}'_j - \bar{\theta}_j\right| = O(\varepsilon)$, we have $\left|\sqrt{q_j} - \left|\bar{\theta}_j\right|\right| = O(\varepsilon)$. Meanwhile, with probability at least $2/3$, we have $\left|q'_j - q_j\right| = O\left(\varepsilon^2\right)$, which implies that $\left|\sqrt{q'_j} - \sqrt{q_j}\right| = O(\varepsilon)$. Then it follows that $\left|\sqrt{q'_j} - \left|\bar{\theta}_j\right|\right| \leq \left|\sqrt{q'_j} - \sqrt{q_j}\right| + \left|\sqrt{q_j} - |\bar{\theta}_j|\right| = O(\varepsilon)$, as desired.

Now we analyze the time complexity of this algorithm. By Proposition 59, one iteration of steps 1-2 takes $\tilde{O}(\text{polylog}(n) \cdot \kappa(\chi + \nu d^3\kappa^6/(\varepsilon^3\Phi))/\sqrt{\Phi})$ time. Since amplitude estimation requires $\tilde{O}(1/\varepsilon^2)$ repetitions of steps 1-2 and their inverses, this algorithm takes $\tilde{O}(\text{polylog}(n) \cdot \kappa(\chi + \nu d^3\kappa^6/(\varepsilon^3\Phi))/(\varepsilon^2\sqrt{\Phi}))$ time, as claimed.

□

Proposition 66. *Suppose* $\alpha \leq \|\mathbf{F}_i\| \leq \beta$*, for any* $i \in [n]$*, and* $\|\mathbf{F}^+\| \leq 1/a$*,* $\|\mathbf{F}\| \leq b$*. Moreover, suppose* $\eta \leq |y_i| \leq \zeta$*, for any* $i \in [n]$*. Let* $\nu = \beta/\alpha$*,* $\chi = \zeta/\eta$ *and* $\kappa = b/a$*. Then for any* $j \in [d]$*, if* $\left|\bar{\theta}_j\right| \geq \delta$*, then* $\text{sgn}\left(\bar{\theta}_j\right)$ *can be determined correctly with probability at least* $2/3$ *in* $\tilde{O}(\text{polylog}(n) \cdot \kappa^2(\chi + \nu d^3\kappa^5/(\delta^3\sqrt{\Phi}))/(\delta\Phi))$ *quantum time.*

Proof. Determining $\text{sgn}\left(\bar{\theta}_j\right)$ is more complicated than estimating $\left|\bar{\theta}_j\right|$. The problem is that the algorithm in Proposition 59 only produces a quantum state $\left|\bar{\theta}\right\rangle$ and one cannot measure the global phase of a quantum state [8]. To overcome this problem, we modify the design matrix $\mathbf{F}$ and the response vector $\mathbf{y}$, such that we know the sign of $\bar{\theta}_{j_0}$ for sure, for some j_0. Then we only need to determine whether $\text{sgn}\left(\bar{\theta}_j\right)$ agrees with $\text{sgn}\left(\bar{\theta}_{j_0}\right)$ or not, for each $j \neq j_0$.

Formally, let us pick an $n \times d$ matrix $\mathbf{G} = (\mathbf{G}_{i,j})$ such that:

- $\mathbf{G}$'s columns are orthogonal and have norm $1/\sqrt{d}$. Namely, $\sum_{i=1}^n \mathbf{G}_{i,j}\mathbf{G}_{i,j'} = \delta_{j,j'}/d$, for any $j, j' \in [d]$.
- $|\mathbf{G}_{i,1}| = 1/\sqrt{nd}$, for any $i \in [n]$;
- There exists constants c_1 and c_2 such that $c_1/n \leq \sum_{j=1}^d |\mathbf{G}_{i,j}|^2 \leq c_2/n$, for any $i \in [n]$.

[8] However, we can determine, say, $\text{sgn}\left(\bar{\theta}_1\bar{\theta}_2\right)$, provided that $|\bar{\theta}_1|$ and $|\bar{\theta}_2|$ are big enough.

Such matrices exist and can be easily constructed (e.g. from the Hadamard matrix). Furthermore, let z be an n-dimensional vector defined as

$$\mathbf{z} = (z_1, z_2, \ldots, z_n)^T = \left\|\hat{\theta}\right\| \cdot (\mathbf{G}_{1,1}, \mathbf{G}_{2,1}, \ldots, \mathbf{G}_{n,1})^T. \tag{5.74}$$

Namely, $\mathbf{z}$ equals the first column of $\mathbf{G}$ times $\left\|\hat{\theta}\right\|$.

Now let

$$\mathbf{F}' = \frac{1}{\sqrt{2}} \begin{pmatrix} \mathbf{F} & 0 \\ 0 & \mathbf{G} \end{pmatrix}, \qquad \mathbf{y}' = \frac{1}{\sqrt{1 + d^{-1} \left\|\hat{\theta}\right\|^2}} \begin{pmatrix} \mathbf{y} \\ \mathbf{z} \end{pmatrix}. \tag{5.75}$$

Then, we have

$$\hat{\theta}' := ((\mathbf{F}')^{\mathrm{T}} \mathbf{F}')^{-1} (\mathbf{F}')^{\mathrm{T}} \mathbf{y}' = \sqrt{\frac{2}{1 + d^{-1} \left\|\hat{\theta}\right\|^2}} \begin{pmatrix} \hat{\theta}_1 \\ \hat{\theta}_2 \\ \vdots \\ \hat{\theta}_d \\ \left\|\hat{\theta}\right\| \\ 0 \\ \vdots \\ 0 \end{pmatrix}. \tag{5.76}$$

Namely, $\hat{\theta}'$ is proportional to the concatenation of $\hat{\theta} = (\hat{\theta}_1, \hat{\theta}_2, \ldots, \hat{\theta}_d)^T$ and $\xi := (\left\|\hat{\theta}\right\|, 0, \ldots, 0)^T$. Note that we know that the $(d+1)$-th entry of $\hat{\theta}'$ is always positive. Furthermore, let $\bar{\theta}' = (\bar{\theta}'_1, \bar{\theta}'_2, \ldots, \bar{\theta}'_{2d})$ be the normalized version of $\hat{\theta}'$. Then one can see that $\bar{\theta}'_j = \bar{\theta}_j / \sqrt{2}$ for $j \in [d]$, $\bar{\theta}'_{d+1} = 1/\sqrt{2}$, and $\bar{\theta}'_j = 0$ for $j > d+1$. To determine $\operatorname{sgn}(\bar{\theta}_j) = \operatorname{sgn}\left(\bar{\theta}'_j\right)$ for any $j \in [d]$, our strategy is to decide whether $\bar{\theta}'_j + \bar{\theta}'_{d+1}$ is larger than $\bar{\theta}'_{d+1}$ or not.

By construction, we have $\operatorname{tr}\left((\mathbf{F}')^{\mathrm{T}} \mathbf{F}'\right) = \|\mathbf{y}'\| = 1$. Let $\mathbf{F}'_i = (\mathbf{F}'_{i,1}, \mathbf{F}'_{i,2}, \ldots, \mathbf{F}'_{i,d})$, for $i \in [n]$. Then let $\alpha' = \min_{i \in [n]} \|\mathbf{F}'_i\|$ and $\beta' = \max_{i \in [n]} \|\mathbf{F}'_i\|$. Then we have $\alpha' = \Omega(\alpha)$, $\beta' = O(\beta)$, and hence $\nu' := \beta'/\alpha' = O(\nu)$. Moreover, since all the singular values of $\mathbf{G}$ are $1/\sqrt{d}$, by Eq.(5.40), we know that all the singular values of $\mathbf{F}'$ are between $\Omega(1/a)$ and $O(b)$. It follows that $\kappa' := \kappa(\mathbf{F}') = O(\kappa)$. Finally, let $\eta' = \min_{i \in [n]} |\mathbf{y}'_i|$ and $\zeta' = \max_{i \in [n]} |\mathbf{y}'_i|$. Then we have

$$\begin{aligned} \eta' &\geq \frac{\min\left(\eta, \left\|\hat{\theta}\right\| / \sqrt{nd}\right)}{\sqrt{1 + \left\|\hat{\theta}\right\|^2 / d}}, \\ \zeta' &\leq \frac{\max\left(\zeta, \left\|\hat{\theta}\right\| / \sqrt{nd}\right)}{\sqrt{1 + \left\|\hat{\theta}\right\|^2 / d}}. \end{aligned} \tag{5.77}$$

Then, using the facts that $\eta \geq 1/(\chi\sqrt{n})$, $\zeta \leq \chi/\sqrt{n}$, and $\sqrt{d\Phi}/\kappa \leq \left\|\hat{\theta}\right\| \leq \sqrt{d\Phi}\kappa$, we get $\chi' := \zeta'/\eta' = O\left(\chi\kappa/\sqrt{\Phi}\right)$. Then it follows from Proposition 59 that we can prepare a state of the form

$$|\Psi\rangle := \sqrt{p}\,|1\rangle\,|0\rangle \left|\bar{\theta}''\right\rangle |\phi\rangle + \sqrt{1-p}(|1\rangle\,|1\rangle\,|\tilde{\varphi}\rangle + |0\rangle\,|0\rangle\,|\tilde{\psi}_0\rangle + |0\rangle\,|1\rangle\,|\tilde{\psi}_1\rangle) \tag{5.78}$$

where $p = \Omega(1)$, $\bar{\theta}'' = (\bar{\theta}''_1, \bar{\theta}''_2, \ldots, \bar{\theta}''_{2d})^T$ satisfies $\left\|\bar{\theta}''\right\| = 1$ and $\left|\bar{\theta}''_j - \bar{\theta}'_j\right| = O(\delta)$, and $|\phi\rangle$ is some normalized state, and $|\tilde{\varphi}\rangle$, $|\tilde{\varphi}_0\rangle$, $|\tilde{\varphi}_1\rangle$ are some unnormalized states, in $\tilde{O}(\text{polylog}(n) \cdot \kappa^2(\chi + \nu d^3\kappa^5/(\delta^3\sqrt{\Phi}))/\Phi)$ time.

Now suppose we measure the first two registers of the state in Eq.(5.78) in the standard basis. Then with probability $\Omega(1)$, we obtain the outcome $(1,0)$. Accordingly, the rest of the state becomes $\left|\bar{\theta}''\right\rangle|\Phi\rangle$. Then, let $\left|\pm_j\right\rangle = \frac{1}{\sqrt{2}}(|j\rangle \pm |d+1\rangle)$, for any $j \in [d]$. Then, we measure the state $\left|\bar{\theta}'\right\rangle$ in the basis $\{\left|+_j\right\rangle, \left|-_j\right\rangle\} \cup \{|i\rangle : i \in [2d], i \notin \{j, d+1\}\}$, and estimate the probability of seeing the outcome corresponding to $\left|+_j\right\rangle$. One can see that this probability is

$$q'' := \frac{(\bar{\theta}''_j + \bar{\theta}''_{d+1})^2}{2} \approx \frac{(\bar{\theta}'_j + \bar{\theta}'_{d+1})^2}{2} = q := \frac{(\bar{\theta}_j + 1)^2}{4}. \tag{5.79}$$

In fact, we have $|q'' - q| = O(\delta)$. Now, if $\bar{\theta}_j \leq -\delta$, we would have $q \leq (1-\delta)/4$; if $\bar{\theta}_j \geq \delta$, we would have $q \geq (1+\delta)/4$. Therefore, by estimating q'' up to an additive error $O(\delta)$, we can distinguish these two cases and hence determine $\text{sgn}(\bar{\theta}_j)$. We can estimate q'' using amplitude estimation, and this requires $O(1/\delta)$ repetitions of the above procedure and its inverse. Therefore, the time complexity of this algorithm is $\tilde{O}(\text{polylog}(n) \cdot \kappa^2(\chi + \nu d^3\kappa^5/(\delta^3\sqrt{\Phi}))/(\delta\Phi))$, as claimed.

In the above argument, we have assumed that we know $\left\|\hat{\theta}\right\|$ exactly. In fact, we only need to estimate it to a relative error $O(\delta)$. This would cause at most $O(\delta)$ error to the above argument. By Theorem 57, estimating $\left\|\hat{\theta}\right\|$ to a relative error $O(\delta)$ takes $\tilde{O}(\text{polylog}(n) \cdot \kappa(\chi + \nu d^3\kappa^6/(\delta^3\Phi))/(\delta\sqrt{\Phi}))$ time. This running time is smaller than that of the above procedure.

□

Remark 67. *We can reduce the failure probability of the algorithms in Proposition 65 and Proposition 66 to arbitrarily small $\gamma > 0$ by repeating them $O(\log(1/\gamma))$ times and taking the median or majority of the estimates obtained.*

Combining Proposition 65 and Proposition 66, we obtain:

Theorem 68. *Suppose $\alpha \leq \|\mathbf{F}_i\| \leq \beta$, for any $i \in [n]$, and $\|\mathbf{F}^+\| \leq 1/a$, $\|\mathbf{F}\| \leq b$. Moreover, suppose $\eta \leq |y_i| \leq \zeta$, for any $i \in [n]$. Let $\nu = \beta/\alpha$, $\chi = \zeta/\eta$ and $\kappa = b/a$. Then there exists a quantum algorithm that produces an estimate $\bar{\theta}'$ of $\bar{\theta}$ such that $\left\|\bar{\theta}' - \bar{\theta}\right\| \leq \varepsilon$ with probability at least $2/3$, in $\tilde{O}(\text{polylog}(n) \cdot (\kappa d^2\chi/(\varepsilon^2\sqrt{\Phi}) + \kappa^2 d^{1.5}\chi/(\varepsilon\Phi) + \kappa^7 d^{6.5}\nu/(\varepsilon^5\Phi^{1.5}))$ quantum time.*

Proof. Consider the following algorithm:

- For each $j \in [d]$:
  1. We use the algorithm in Proposition 65 to estimate $\left|\bar{\theta}_j\right|$ up to an additive error $\varepsilon/(8\sqrt{d})$ with probability at least $1 - 1/(6d)$.
  2. Let α_j be the estimate of $\left|\bar{\theta}_j\right|$. Then:

- If $\alpha_j \geq \varepsilon/(8\sqrt{d})$, then we use the algorithm in Proposition 66 (setting $\delta = \varepsilon/(8\sqrt{d})$) to determine $\operatorname{sgn}\left(\bar{\theta}_j\right)$ with probability at least $1 - 1/(6d)$. Let β_j be the output of this algorithm. Then we set $\bar{\theta}'_j = \alpha_j \beta_j$;
- Otherwise, we set $\bar{\theta}'_j$ to be $+\alpha_j$ or $-\alpha_j$ arbitrarily.

Let $S = \{j \in [d] \ : \ \alpha_j \geq \varepsilon/(8\sqrt{d})\}$. Then, with probability at least $2/3$, we have

$$\left|\alpha_j - \left|\bar{\theta}_j\right|\right| \leq \frac{\varepsilon}{8\sqrt{d}}, \quad \forall j \in [d], \tag{5.80}$$

and

$$\beta_j = \operatorname{sgn}\left(\bar{\theta}_j\right), \quad \forall j \in S. \tag{5.81}$$

This implies that

$$\left|\bar{\theta}'_j - \bar{\theta}_j\right| \leq \frac{\varepsilon}{8\sqrt{d}}, \quad \forall j \in S. \tag{5.82}$$

Meanwhile, we have

$$\left|\bar{\theta}_j\right| \leq \frac{\varepsilon}{4\sqrt{d}}, \quad \forall j \notin S. \tag{5.83}$$

Thus,

$$\left|\bar{\theta}'_j - \bar{\theta}_j\right| \leq \frac{\varepsilon}{2\sqrt{d}}, \quad \forall j \notin S. \tag{5.84}$$

It follows that

$$\left\|\bar{\theta}' - \bar{\theta}\right\|^2 = \sum_{j \in S} \left|\bar{\theta}'_j - \bar{\theta}_j\right|^2 + \sum_{j \notin S} \left|\bar{\theta}'_j - \bar{\theta}_j\right|^2 \leq \frac{\varepsilon^2}{64} + \frac{\varepsilon^2}{4} \leq \varepsilon^2. \tag{5.85}$$

This proves the correctness of this algorithm.

Now we analyze the time complexity of this algorithm. For each $j \in [d]$, step 1 takes

$$\tilde{O}(\operatorname{polylog}(n) \cdot \kappa d(\chi + \nu d^{4.5} \kappa^6/(\varepsilon^3 \Phi))/(\varepsilon^2 \sqrt{\Phi})) \tag{5.86}$$

time by Proposition 65, and step 2 takes

$$\tilde{O}(\operatorname{polylog}(n) \cdot \kappa^2 \sqrt{d}(\chi + \nu d^{4.5} \kappa^5/(\varepsilon^3 \sqrt{\Phi})/(\varepsilon \Phi)) \tag{5.87}$$

time by Proposition 66. Since we need to do step 1 and 2 for every $j \in [d]$, this algorithm has time complexity

$$\tilde{O}(\operatorname{polylog}(n) \cdot (\kappa d^2 \chi/(\varepsilon^2 \sqrt{\Phi}) + \kappa^2 d^{1.5} \chi/(\varepsilon \Phi) + \kappa^7 d^{6.5} \nu/(\varepsilon^5 \Phi^{1.5})), \tag{5.88}$$

as claimed.

□

Remark 69. *We can reduce the failure probability of the algorithm in Theorem 68 to arbitrarily small $\delta > 0$ by making the success probabilities in steps 1 and 2 to be $1 - O(\delta/d)$. The running time of this algorithm will be increased by a factor of $O(\log(d/\delta))$.*

5.6 Quantum Algorithm for Estimating the Fit Quality Φ

So far, we have presented two quantum algorithms for the estimation of the best-fit parameters $\hat{\theta}$. One may notice that they both rely on some knowledge of the fit quality Φ. Now we show that this is without loss of generality, because Φ can be estimated quickly, as indicated by the following theorem:

Theorem 70. *Suppose $\alpha \leq \|\mathbf{F}_i\| \leq \beta$, for any $i \in [n]$. Moreover, suppose $\eta \leq |y_i| \leq \zeta$, for any $i \in [n]$. Let $\nu = \beta/\alpha$, $\chi = \zeta/\eta$ and $\kappa = \kappa(\mathbf{F})$. Then Φ can be estimated up to an additive error $\varepsilon > 0$ with probability at least $2/3$ in $\tilde{O}(\text{polylog}(n) \cdot (\chi + \nu d^3\kappa^4/\varepsilon)/\varepsilon)$ quantum time.*

Proof of Theorem 70. Our strategy for estimating $\Phi = \|\hat{\mathbf{y}}\|^2 = \|\Pi_{\mathbf{F}} |\mathbf{y}\rangle\|^2$ is as follows. We prepare the state $|\mathbf{y}\rangle$, and perform the projective measurement $\{\Pi_{\mathbf{F}}, I - \Pi_{\mathbf{F}}\}$ on it, and estimate the probability of seeing the outcome corresponding to $\Pi_{\mathbf{F}}$. The measurement $\{\Pi_{\mathbf{F}}, I - \Pi_{\mathbf{F}}\}$ is implemented by running phase estimation on $e^{i\sigma}$ and checking whether the eigenphase is close to 0 or not.

Specifically, suppose $|\hat{\mathbf{y}}\rangle = \sum_{j=1}^{d} \alpha_j |u_j\rangle$. Consider the following algorithm for estimating $\Phi = \|\hat{\mathbf{y}}\|^2 = \sum_{j=1}^{d} \alpha_j^2$ (again, we assume that phase estimation is perfect in the following description, and we will take the error of phase estimation into account later):

1. We use the algorithm in Lemma 56 to prepare the state $|\mathbf{y}\rangle$.
2. We run phase estimation on $e^{i\sigma}$ starting with $|\mathbf{y}\rangle$, obtaining the state
$$\sum_{j=1}^{d} \alpha_j |u_j\rangle |s_j^2\rangle + |\hat{\varepsilon}\rangle |0\rangle, \tag{5.89}$$
3. We perform the measurement $\{|0\rangle\langle 0|, I - |0\rangle\langle 0|\}$ on the second register. Then, conditioned on seeing the outcome corresponding to $I - |0\rangle\langle 0|$, the state becomes proportional to
$$\sum_{j=1}^{d} \alpha_j |u_j\rangle |s_j^2\rangle \tag{5.90}$$
Furthermore, the probability of seeing this outcome is $q := \sum_{j=1}^{d} \alpha_j^2 = \Phi$.
4. We use amplitude estimation to estimate q up to an additive error $O(\varepsilon)$ with probability at least $2/3$. This requires $\tilde{O}(1/\varepsilon)$ repetitions of the above procedure and its inverse.

Now we take the error of phase estimation into account, and analyze the time complexity of this algorithm. In step 2, we do not get the eigenphase s_j^2 exactly, but get some (random) $\lambda_j \approx s_j^2$ (although phase estimation can single out the eigenphase 0 perfectly). This implies we only obtain the states in steps 2-3 approximately. Since we want to estimate q to a relative error $O(\varepsilon)$, we need to make sure that $\left|\lambda_j - s_j^2\right| \leq s_j^2/3$ (so that $\lambda_j \neq 0$). Since $s_j^2 \geq 1/(d\kappa^2)$, we need to set the precision of phase estimation to be $O\left(1/(d\kappa^2)\right)$. It follows that we need to simulate $e^{i\sigma t}$ to accuracy

$O(\varepsilon)$ [9] for $t = O\left(d\kappa^2\right)$ during phase estimation. This can be done in $\tilde{O}(\text{polylog}(n)\cdot\nu d(d\kappa^2)^2/\varepsilon) = \tilde{O}(\text{polylog}(n)\cdot\nu d^3\kappa^4/\varepsilon)$ time by Lemma 55. Meanwhile, it takes $\tilde{O}(\text{polylog}(n)\cdot\chi)$ time to prepare $|\mathbf{y}\rangle$ by Lemma 56. Thus, one iteration of steps 1-3 takes $\tilde{O}(\text{polylog}(n)\cdot(\chi+\nu d^3\kappa^4/\varepsilon))$ time. Since amplitude estimation requires $\tilde{O}(1/\varepsilon)$ repetitions of steps 1-3 and their inverses, this algorithm takes $\tilde{O}(\text{polylog}(n)\cdot(\chi+\nu d^3\kappa^4/\varepsilon)/\varepsilon)$ time, as claimed.

□

Remark 71. *We can reduce the failure probability of the algorithm in Theorem 70 to arbitrarily small $\delta > 0$ by repeating this algorithm $O(\log(1/\delta))$ times and taking the median of the estimates obtained.*

5.7 Open Problems

We conclude by pointing out several directions for future research:

First, our work suggests that quantum algorithms might be able to solve curve fitting exponentially faster than classical algorithms. But we do not actually prove it. We only show that quantum algorithms can solve curve fitting extremely fast when d, κ, ν and χ are sufficiently small. But it is unknown whether classical algorithms can solve this case fast. It would be interesting to know the biggest possible gap between our algorithms and classical algorithms.

Second, in this chapter, we have focused on proving *upper* bounds on the quantum complexity of curve fitting. It is also worth studying *lower* bounds on the quantum complexity of the same problem. In particular, we would like to know if it is possible to generate the state $\left|\bar{\theta}\right\rangle$ to accuracy $\varepsilon > 0$ in $\text{poly}(\log n, \log d, \kappa, \nu, \chi, 1/\Phi, 1/\varepsilon)$ time? Note that this complexity is poly-logorithmic in both n and d. If so, it would have immediate implications to the simulation of quantum many-body systems, as suggested by [132]. We suspect that such algorithm does not exist. Can this be proved under some complexity assumption, say, BQP $\neq$ PSPACE?

Third, Ambainis [9] has introduced a technique called *variable-time amplitude amplification* and used it to improve HHL's algorithm. Since our algorithms for estimating the best-fit parameters have a similar structure to HHL's algorithm, can his technique be used to improve our algorithms as well?

Fourth, as one can see, our algorithms crucially depend on the ability to simulate the evolutions of nonsparse Hamiltonians. Here we have used the density matrix exponentiation method of [96]. It would be interesting to know whether this method is optimal. Namely, is there a more efficient way of simulating $e^{i\rho t}$ by consuming multiple copies of ρ? If so, it would immediately lead to an improvement of our algorithms.

Finally, given the wide applications of curve fitting, it is promising that our algorithms could be utilized to solve practical problems in many fields.

[9] We want the disturbance caused by the imperfection of simulating $e^{i\sigma t}$ to be at most $O(\varepsilon)$.

Bibliography

[1] S. Aaronson and A. Ambainis. Quantum search of spatial regions. *Theory of Computing* 1, pages 47-79, 2005.

[2] D. S. Abrams and S. Lloyd. Simulation of many-body Fermi systems on a universal quantum computer. *Physical Review Letters* 79, pages 2586-2589, 1997.

[3] D. Aharonov, T. Naveh. Quantum NP - A Survey. arXiv:quant-ph/0210077, 2002.

[4] D. Aharonov and A. Ta-Shma. Adiabatic quantum state generation and statistical zero knowledge. In *Proceedings of the 35th ACM Symposium on Theory of Computing (STOC)*, pages 20-29, 2003.

[5] N. Alon, R. Yuster, and U. Zwick. Color-coding. *Journal of the ACM*, 42(4), pages 844-856, 1995.

[6] A. Ambainis. Quantum lower bounds by quantum arguments. *Journal of Computer and System Sciences* 64(4), pages 750-767, 2002.

[7] A. Ambainis. Polynomial degree vs. quantum query complexity. In *Proceedings of the 44th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 230-239, 2003.

[8] A. Ambainis. Quantum walk algorithm for element distinctness. In *Proceedings of the 45th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 22-31, 2004.

[9] A. Ambainis. Variable time amplitude amplification and a faster quantum algorithm for solving systems of linear equations. In *Proceedings of the 29th Symposium on Theoretical Aspects of Computer Science (STACS)*, page 636-647, 2012.

[10] A. Ambainis, A. M. Childs, B. W. Reichardt, R. Špalek and S. Zhang. Any AND-OR formula of size N can be evaluated in time $N^{1/2+o(1)}$ on a quantum computer. In *Proceedings of the 48th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 363-372, 2007.

[11] A. Ambainis, J. Kempe and A. Rivosh. Coins make quantum walks faster. In *Proceedings of the 16th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1099-1108, 2005.

[12] A. Ambainis and R. Špalek. Quantum algorithms for matching and network flows. In *Proceedings of the 23rd Annual Symposium on Theoretical Aspects of Computer Science (STACS)*, pages 172-183, 2006.

[13] S. L. Arlinghaus. *PHB Practical Handbook of Curve Fitting*. CRC Press, 1994.

[14] D. Bacon, A. M. Childs and W. van Dam. From optimal measurement to efficient quantum algorithms for the hidden subgroup problem over semidirect product groups. In *Proceedings of the 46th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 469-478, 2005.

[15] D. Bacon, A. M. Childs and W. van Dam. Optimal measurements for the dihedral hidden subgroup problem. *Chicago Journal of Theoretical Computer Science* 2006(2).

[16] A.T. Balaban. *Chemical applications of graph theory*. Academic Press, 1976.

[17] H. Barnum, M. Saks and M. Szegedy. Quantum decision trees and semidefinite programming. In *Proceedings of 18th IEEE Conference on Computational Complexity (CCC)*, pages 179-193, 2003.

[18] R. Beals, H. Buhrman, R. Cleve, M. Mosca and R. de Wolf. Quantum lower bounds by polynomials. *Journal of the ACM* 48(4), pages 778-797, 2001.

[19] A. Belovs. Span-program-based quantum algorithm for the rank problem. arXiv:1103.0842, 2011.

[20] A. Belovs. Span programs for functions with constant-sized 1-certificates. In *Proceedings of the 44th ACM Symposium on Theory of Computing (STOC)*, pages 77–84, 2012.

[21] A. Belovs. Learning-graph-based Quantum Algorithm for k-distinctness. In *Proceedings of the 53th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 207-216, 2012.

[22] A. Belovs, A. M. Childs, S. Jeffery, R. Kothari and F. Magniez. Time-Efficient Quantum Walks for 3-Distinctness. In *Proceedings of the 40th International Colloquium on Automata, Languages and Programming (ICALP)*, pages 105-122, 2013.

[23] A. Belovs and T. Lee. Quantum algorithm for k-distinctness with prior knowledge on the input. arXiv:1108.3022, 2011.

[24] A. Belovs and B. W. Reichardt. Span programs and quantum algorithms for st-connectivity and claw detection. In *Proceedings of the 20th European Symposia on Algorithm (ESA)*, pages 193-204, 2012.

[25] A. Belovs and A. Rosmanis. On the Power of Non-Adaptive Learning Graphs. In *Proceedings of the 28th IEEE Conference on Computational Complexity (CCC)*, pages 44-55, 2013.

[26] C. H. Bennett, G. Brassard, C. Crépeau, R. Jozsa, A. Peres and W. K. Wootters. Teleporting an Unknown Quantum State via Dual Classical and EinsteinPodolskyRosen Channels. *Physical Review Letters* 70, pages 1895-1899, 1993.

[27] C. Bennett and S. J. Wiesner. Communication via one- and two-particle operators on Einstein-Podolsky-Rosen states. *Physical Review Letters* 69, 2881, 1992.

[28] E. Bernstein and U. Vazirani. Quantum complexity theory. *SIAM Journal on Computing* 26(5), pages 1411-1473, 1997.

[29] D. W. Berry. Quantum algorithms for solving linear differential equations. arXiv:1010.2745, 2010.

[30] D. W. Berry, G. Ahokas, R. Cleve and B. C. Sanders. Efficient quantum algorithms for simulating sparse Hamiltonians. *Communications in Mathematical Physics* 270, 359, 2007.

[31] D. W. Berry and A. M. Childs. Black-box Hamiltonian simulation and unitary implementation. *Quantum Information and Computation* 12, 29, 2012.

[32] D. W. Berry, A. M. Childs, R. Cleve, R. Kothari and R. D. Somma. Exponential improvement in precision for simulating sparse Hamiltonians. arXiv:1312.1414, 2013.

[33] P. Biswal, J. R. Lee and S. Rao. Eigenvalue bounds, spectral partitioning, and metrical deformations via flows. *J*ournal of the ACM 57(3), article 13, 2010.

[34] P. O. Boykin, T. Mor, M. Pulver, V. Roychowdhury and F. Vatan. On universal and fault-tolerant quantum computing. *Information Processing Letters* 75, pages 101-107, 2000.

[35] G. Brassard, P. Høyer, M. Mosca and A. Tapp. Quantum Amplitude Amplification and Estimation. *Quantum Computation and Information*, edited by S. J. Lomonaco and H. E. Brandt, AMS, volume 305 of *AMS Contemporary Mathematics Series*, pages 5374, arXiv:quant-ph/0005055.

[36] G. Brassard, P. Høyer and A. Tapp. Quantum Algorithm for the Collision Problem. In *Proceedings of the 3rd Latin American Theoretical Informatics Symposium (LATIN)*, pages 163-169, 1998.

[37] O. Bretscher. *Linear Algebra With Applications, 3rd ed*. Upper Saddle River NJ: Prentice Hall.

[38] H. Buhrman and R. Špalek. Quantum verification of matrix products. In *Proceedings of the 17th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 880-889, 2006.

[39] T. Byrnes and Y. Yamamoto. Simulating lattice gauge theories on a quantum computer. *Physical Review A* 73, 022328, 2006.

[40] A. K. Chandra, P. Raghavan, W. L. Ruzzo, R. Smolensky and P. Tiwari. The electrical resistance of a graph captures its commute and cover times. *Computational Complexity* 6, 312 (1997).

[41] J. Cheeger. A lower bound for smallest eigenvalue of the Laplacian. In *Problems in Analysis, pages*, pages 195-199, Princeton University Press, 1970.

[42] D. Cheung, D. Maslov, J. Mathew and D. Pradhan. On the design and optimization of a quantum polynomial-time attack on elliptic curve cryptography. In *Proceedings of the 3rd Workshop on Theory of Quantum Computation, Communication, and Cryptography*, pages 96-104, 2008.

[43] A. M. Childs. On the relationship between continuous- and discrete-time quantum walk. *Communications in Mathematical Physics* 294, pages 581-603, 2010.

[44] A. M. Childs, R. Cleve, E. Deotto, E. Farhi, S. Gutmann and D. A. Spielman. Exponential algorithmic speedup by quantum walk. In *Proceedings of the 35th ACM Symposium on Theory of Computing (STOC)*, pages 59-68, 2003.

[45] A. M. Childs and W. van Dam. Quantum algorithm for a generalized hidden shift problem. In *Proceedings of the 18th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1225-1234, 2007.

[46] A. M. Childs and J. M. Eisenberg. Quantum algorithms for subset finding. *Quantum Information and Computation* 5, 593, 2005.

[47] A. M. Childs and J. Goldstone. Spatial search and the Dirac equation. *Physical Review A* 70, 042312, 2004.

[48] A. M. Childs and J. Goldstone. Spatial search by quantum walk. *Physical Review A* 70, 022314, 2004.

[49] A. M. Childs and R. Kothari. Limitations on the simulation of non-sparse Hamiltonians. *Quantum Information and Computation* 10, pages 669-684, 2010.

[50] A. M. Childs and R. Kothari. Simulating sparse Hamiltonians with star decompositions. *Lecture Notes in Computer Science* 6519, pages 94-103 (2011).

[51] A. M. Childs and R. Kothari. Quantum query complexity of minor-closed graph properties. In *Proceedings of the 28th Symposium on Theoretical Aspects of Computer Science (STACS)*, pages 661–672, 2011.

[52] A. M. Childs, L. J. Schulman and U. V. Vazirani. Quantum algorithms for hidden nonlinear structures. In *Proceedings of the 48th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 395-404, 2007.

[53] A. M. Childs and N. Wiebe. Hamiltonian simulation using linear combinations of unitary operations. *Quantum Information and Computation* 12, pages 901-924, 2012.

[54] A. M. Childs and P. Wocjan. On the quantum hardness of solving isomorphism problems as nonabelian hidden shift problems. *Quantum Information and Computation* 7(5-6), pages 504-521, 2007.

[55] P. Christiano, J. A. Kelner, A. Madry, D. A. Spielman and S.-H. Teng. Electrical Flows, Laplacian Systems, and Faster Approximation of Maximum Flow in Undirected Graphs. In *Proceedings of the 43th ACM Symposium on Theory of Computing (STOC)*, pages 273-281, 2011.

[56] F. R. K. Chung. *Spectral Graph Theory*. American Mathematical Society, 1997.

[57] W. van Dam, S. Hallgren and L. Ip. Quantum algorithms for some hidden shift problems. *SIAM Journal on Computing* 36(3), pages 763-778, 2006.

[58] W. van Dam and G. Seroussi. Efficient quantum algorithms for estimating Gauss sums. arXiv:quant-ph/0207131, 2002.

[59] D. Deutsch. Quantum theory, the Church-Turing principle, and the universal quantum computer. *Proceedings of the Royal Society of London*. Series A 400, pages 97-117, 1985.

[60] D. Deutsch and R. Jozsa. Rapid solution of problems by quantum computation. *Proceedings of the Royal Society: Mathematical and Physical Sciences* 439, pages 553-558, 1992.

[61] D. P. DiVincenzo. Two-bit gates are universal for quantum computation. *Physical Review A* 51, pages 1015-1022, 1995.

[62] C. Dürr, M. Heiligman, P. Høyer and M. Mhalla. Quantum query complexity of some graph problems. In *Proceedings of the 31st International Colloquium on Automata, Languages and Programming (ICALP)*, pages 48-493, 2004.

[63] M. Ettinger and P. Høyer. On quantum algorithms for noncommutative hidden subgroups. *Advances in Applied Mathematics* 25, pages 239-251, 2000.

[64] E. Farhi, J. Goldstone and S. Gutmann. A quantum algorithm for the Hamiltonian NAND tree. arXiv:quant-ph/0702144, 2007.

[65] E. Farhi, J. Goldstone, S. Gutmann and M. Sipser. Limit on the speed of quantum computation in determining parity. *Physical Review Letters* 81(24), pages 5442-5444, 1998.

[66] E. Farhi and S. Gutmann. Quantum computation and decision trees. *Physical Review A* 58, pages 915-928, 1998.

[67] R. P. Feynman. Simulating physics with computers. *International Journal of Theoretical Physics* 21, pages 467-488, 1982.

[68] W. Fulton and J. Harris. *Representation theory. A first course*. Graduate Texts in Mathematics, Readings in Mathematics 129, New York: Springer-Verlag, 1991.

[69] D. Gavinsky. Quantum solution to the hidden subgroup problem for poly-near-Hamiltonian groups. *Quantum Information and Computation* 4(3), pages 229-235, 2004.

[70] D. Gavinsky and T. Ito. A quantum query algorithm for the graph collision problem. arXiv:1204.1527, 2012.

[71] L. K. Grover. A fast quantum mechanical algorithm for database search. In *Proceedings of the 28th ACM Symposium on the Theory of Computing (STOC)*, pages 212-219, 1996.

[72] M. Grigni, L. J. Schulman, M. Vazirani and U. Vazirani. Quantum mechanical algorithms for the nonabelian hidden subgroup problem. *Combinatorica* 24(1), pages 137-154, 2004.

[73] A. Aspuru-Guzik, A. D. Dutoi, P. J. Love and M. Head-Gordon. Simulated quantum computation of molecular energies. *Science* 309, pages 1704-1707, 2005.

[74] S. Hallgren. Polynomial-time quantum algorithms for Pells equation and the principal ideal problem. *Journal of the ACM* 54(1), article 4, 2007.

[75] S. Hallgren. Fast quantum algorithms for computing the unit group and class group of a number field. In *Proceedings of the 37th ACM Symposium on Theory of Computing (STOC)*, pages 468-474, 2005.

[76] S. Hallgren, C. Moore, M. Rötteler, A. Russell and P. Sen. Limitations of quantum coset states for graph isomorphism. In *Proceedings of the 38th ACM Symposium on Theory of Computing (STOC)*, pages 604-617, 2006.

[77] S. Hallgren, A. Russell and A. Ta-Shma. The hidden subgroup problem and quantum computation using group representations. *SIAM Journal on Computing* 32(4), pages 916-934, 2003.

[78] A. W. Harrow, A. Hassidim and S. Lloyd. Quantum algorithm for linear systems of equations. *Physical Review Letters* 103, 150502 (2009).

[79] A. W. Harrow, B. Recht, and I. L. Chuang. Efficient discrete approximations of quantum gates. *Journal of Mathematical Physics* 43(9), pages 4445-4451, 2002.

[80] R. Heckel. Graph transformation in a nutshell. *Electronic Notes in Theoretical Computer Science* 148, pp. 187198, 2006.

[81] R. Horodecki, P. Horodecki, M. Horodecki and K. Horodecki. Quantum entanglement. *Review of Modern Physics* 81, 865, 2009.

[82] P. Høyer, T. Lee and R. Spalek. Negative weights make adversaries stronger. In *Proceedings of the 39th ACM Symposium on the Theory of Computing (STOC)*, pages 526-535 , 2007.

[83] S. Jeffery, R. Kothari, and F. Magniez. Nested Quantum Walks with Quantum Data Structures. In *Proceedings of ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1474-1485, 2013.

[84] S. Jordan, K. Lee and J. Preskill. Quantum algorithms for quantum field theories. *Science* 336, pages. 1130-1133, 2012.

[85] I. Kassal, S. P. Jordan, P. J. Love, M. Mohseni and A. Aspuru-Guzik. Quantum algorithms for the simulation of chemical dynamics. *Proceedings of the National Academy of Sciences* 105, pages 18681-18686, 2008.

[86] M. Karchmer, and A. Wigderson. On span programs. In *Proceedings of the 8th IEEE Symp. Structure in Complexity Theory*, pages 102-111, 1993.

[87] A. Y. Kitaev. Quantum measurements and the Abelian Stabilizer Problem. arXiv:quant-ph/9511026, 1995.

[88] A. Y. Kitaev. Quantum computations: Algorithms and error correction. *Russian Mathematical Surveys* 52(6), pages 1191-1249, 1997.

[89] A. Y. Kitaev, A. Shen and M. N. Vyalyi. *Classical and Quantum Computation*. American Mathematical Society, 2002.

[90] E. Knill. Approximation by quantum circuits. Technical Report LAUR-95-2225, Los Alamos National Laboratory, arXiv:quant-ph/9508006, 1995.

[91] G. Kuperberg. A subexponential-time quantum algorithm for the dihedral hidden subgroup problem. *SIAM Journal on Computing* 35(1), pages 170-188, 2005.

[92] T. Lee, F. Magniez, and M. Santha. A learning graph based quantum query algorithm for finding constant-size subgraphs. *Chicago Journal of Theoretical Computer Science*, 2012.

[93] T. Lee, F. Magniez and M. Santha. Improved Quantum Query Algorithms for Triangle Finding and Associativity Testing. In *Proceedings of ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1486-1502, 2013.

[94] Y. T. Lee, S. Rao and N. Srivastava. A new approach to computing maximum flows using electrical flows. In *Proceedings of the 45th ACM Symposium on Theory of Computing (STOC)*, pages 755-764, 2013.

[95] S. Lloyd. Universal quantum simulators. *Science* 273, pages 1073-1078, 1996.

[96] S. Lloyd, M. Mohseni and P. Rebentrost. Quantum principal component analysis. arXiv: 1307.0401, 2013.

[97] A. Madry. Navigating Central Path with Electrical Flows: from Flows to Matchings, and Back. In *Proceedings of the 54th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 253-262, 2013.

[98] F. Magniez and A. Nayak. Quantum complexity of testing group commutativity. *Algorithmica* 48(3), pages 221-232, 2007.

[99] F. Magniez, A. Nayak, J. Roland and M. Santha. Search via quantum walk. In *Proceedings of the 39th ACM Symposium on Theory of Computing (STOC)*, pages 575-584, 2007.

[100] F. Magniez, M. Santha, and M. Szegedy. Quantum algorithms for the triangle problem. In *Proceedings of ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1109-1117, 2005.

[101] Y. Manin. Computable and uncomputable. Sovetskoye Radio, 1980.

[102] C. Moore, D. N. Rockmore, A. Russell and L. J. Schulman. The power of strong Fourier sampling: Quantum algorithms for affine groups and hidden shifts. *SIAM Journal on Computing* 37(3), pages 938-958, 2007.

[103] M. Mosca and A. Ekert. The hidden subgroup problem and eigenvalue estimation on a quantum computer. In *Proceedings of the 1st NASA International Conference on Quantum Computing and Quantum Communication*, 1999.

[104] D. Nagaj, P. Wocjan and Y. Zhang. Fast amplification of QMA. *Quantum Information and Computation* 9, 1053 (2009).

[105] M. A. Nielsen and I. L. Chuang. *Quantum Computation and Quantum Information*. Cambridge University Press, 10th Anniversary Edition, 2011.

[106] M. Ohlrich, C. Ebeling, E. Ginting, and L. Sather. SubGemini: identifying subcircuits using a fast subgraph isomorphism algorithm. In *Proceedings of the 30th International Design Automation Conference*, pages 3137, 1993.

[107] G. Ortiz, J.E. Gubernatis, E. Knill, and R. Laflamme. Quantum algorithms for Fermionic simulations. *Physical Review A* 64, 022319, 2001.

[108] D. Poulin, A. Quarry, R. D. Somma and F. Verstraete. Quantum simulation of time-dependent Hamiltonians and the convenient illusion of Hilbert space. *Physical Review Letters* 106, 170501, 2011.

[109] D. Poulin and P. Wocjan. Sampling from the thermal quantum Gibbs state and evaluating partition functions with a quantum computer. *Physical Review Letters* 103, 220502, 2009.

[110] J. Preskill. Lecture Notes for Physics 229: Quantum Information and Computation, Caltech, Fall 1998.

[111] B. W. Reichardt. Span programs and quantum query complexity: The general adversary bound is nearly tight for every boolean function. In *Proceedings of the 50th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 544-551, 2009.

[112] B. W. Reichardt. Span-program-based quantum algorithm for evaluating unbalanced formulas. In *6th Conf. on Theory of Quantum Computation, Communication and Cryptography (TQC)*, 2011.

[113] B. W. Reichardt. Reflections for quantum query algorithms. In *Proceedings of ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 560-569, 2011.

[114] B. W. Reichardt. Faster quantum algorithm for evaluating game trees. In *Proceedings of ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 546-559, 2011.

[115] B. W. Reichardt and R. Špalek. Span-program-based quantum algorithm for evaluating formulas. In *Proceedings of the 40th ACM Symposium on Theory of Computing (STOC)*, pages 103-112, 2008.

[116] M. Saks and A. Wigderson. Probabilistic Boolean decision trees and the complexity of evaluating game trees. In *Proceedings of the 27th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 29-38, 1986.

[117] P. W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM Journal on Computing* 26(5), pages 1484-1509, 1997.

[118] D. R. Simon. On the power of quantum computation. *SIAM Journal on Computing* 26(5), pages 1474-1483, 1997.

[119] M. Snir. Lower bounds on probabilistic linear decision trees. *Theoretical Computer Science* 38, pages 69-82, 1985.

[120] R. Solovay. Lie groups and quantum circuits. Mathematical Sciences Research Institute, 2000.

[121] R. Špalek and M. Szegedy. All Quantum Adversary Methods are Equivalent. In *Proceedings of the 32nd international conference on Automata, Languages and Programming (ICALP)*, pages 1299-1311, 2005.

[122] D. A. Spielman and N. Srivastava. Graph Sparsification by Effective Resistances. In *Proceedings of the 40th ACM Symposium on Theory of Computing (STOC)*, pages 563-568, 2008.

[123] D. A. Spielman and S.-H. Teng. Nearly-linear time algorithms for graph partitioning, graph sparsification, and solving linear systems. In *Proceedings of the 36th ACM Symposium on Theory of Computing (STOC)*, pages 81-90, 2004.

[124] M. Suzuki. Fractal decomposition of exponential operators with applications to many-body theories and Monte Carlo simulations. *Physical Letters A* 146(6), pages 319-323, 1990.

[125] M. Szegedy. Quantum speed-up of Markov chain based algorithms. In *Proceedings of the 45th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 32-41, 2004.

[126] K. Temme, T. J. Osborne, K. G. Vollbrecht, D. Poulin, and F. Verstraete. Quantum Metropolis Sampling. *Nature* 471, pages 87-90, 2011.

[127] L. Trevisan. Max cut and the smallest eigenvalue. In *Proceedings of the 41st ACM Symposium on Theory of Computing (STOC)*, pages 263-272, 2009.

[128] H. F. Trotter. On the product of semi-groups of operators. *Proceedings of the American Mathematical Society* 10 (4), pages 545-551, 1959.

[129] U. Vazirani. Lecture Notes for CS 294: Quantum Computation, UC Berkeley, Spring 2009.

[130] J. Watrous. Quantum simulations of classical random walks and undirected graph connectivity. *Journal of Computer and System Sciences* 62, pages 376-391, 2001.

[131] J. Watrous. Quantum algorithms for solvable groups. In *Proceedings of the 33rd ACM Symposium on Theory of Computing (STOC)*, pages 60-67, 2001.

[132] N. Wiebe, D. Braun and S. Lloyd. Quantum data fitting. *Physical Review Letters* 109, 050505 (2012).

[133] L.-A. Wu, M. S. Byrd and D. A. Lidar. Polynomial-Time Simulation of Pairing Models on a Quantum Computer. *Physical Review Letters* 89, 057904, 2002.

[134] C. Zalka. Simulating quantum systems on a quantum computer. *Proceedings of the Royal Society A* 454, pages 313-322, 1998.

[135] S. Zhang. On the power of Ambainis's lower bounds. *Theoretical Computer Science* 339(2-3), pages 241-256, 2005.

[136] Y. Zhu. Quantum query complexity of subgraph containment with constant-sized certificates. arXiv:1109.4165, 2011.