QUANTUM ALGORITHMS FOR: QUANTUM PHASE ESTIMATION, APPROXIMATION OF THE TUTTE POLYNOMIAL AND BLACK-BOX STRUCTURES

by

HAMED AHMADI
B.S. Mathematics, Shahid Beheshti University, 2004
M.S. Mathematics, University of Tehran, 2007

A dissertation submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy
in the Department of Mathematics
in the College of Sciences
at the University of Central Florida
Orlando, Florida

Spring Term
2012

Major Professor:
Joseph P. Brennan

ABSTRACT

In this dissertation, we investigate three different problems in the field of Quantum computation. First, we discuss the quantum complexity of evaluating the Tutte polynomial of a planar graph. Furthermore, we devise a new quantum algorithm for approximating the phase of a unitary matrix. Finally, we provide quantum tools that can be utilized to extract the structure of black-box modules and algebras.

While quantum phase estimation (QPE) is at the core of many quantum algorithms known to date, its physical implementation (algorithms based on quantum Fourier transform (QFT)) is highly constrained by the requirement of high-precision controlled phase shift operators, which remain difficult to realize. In the second part of this dissertation, we introduce an alternative approach to approximately implement QPE with arbitrary constant-precision controlled phase shift operators.

The new quantum algorithm bridges the gap between QPE algorithms based on QFT and Kitaev's original approach. For approximating the eigenphase precise to the nth bit, Kitaev's original approach does not require any controlled phase shift operator. In contrast, QPE algorithms based on QFT or approximate QFT require controlled phase shift operators with precision of at least Pi/2n. The new approach fills the gap and requires only arbitrary constant-precision controlled phase shift operators. From a physical implementation viewpoint, the new algorithm outperforms Kitaev's approach.

The other problem we investigate relates to approximating the Tutte polynomial. We show that the problem of approximately evaluating the Tutte polynomial of triangular graphs at the points $(q, 1/q)$ of the Tutte plane is BQP-complete for (most) roots of unity q. We also consider circular graphs and show that the problem of approximately evaluating the Tutte polynomial of these graphs at the point $(e^{2\pi i/5}, e^{-2\pi i/5})$ is DQC1-complete and at points $(q^k, 1 + \frac{1-q^{-k}}{(q^{1/2}-q^{-1/2})^2})$ for some integer k is in BQP.

To show that these problems can be solved by a quantum computer, we rely on the relation of the Tutte polynomial of a planar G graph with the Jones and HOMFLY polynomial of the alternating link $D(G)$ given by the medial graph of G. In the case of our graphs the corresponding links are equal to the plat and trace closures of braids. It is known how to evaluate the Jones and HOMFLY polynomial for closures of braids.

To establish the hardness results, we use the property that the images of the generators of the braid group under the irreducible Jones-Wenzl representations of the Hecke algebra have finite order. We show that for each braid b we can efficiently construct a braid $\tilde{b}$ such that the evaluation of the Jones and HOMFLY polynomials of their closures at a fixed root of unity leads to the same value and that the closures of $\tilde{b}$ are alternating links.

The final part of the dissertation focuses on finding the structure of a black-box module or algebra. Suppose we are given black-box access to a finite module M or algebra over a finite ring R, and a list of generators for M and R. We show how to find a linear basis and structure constants for M in quantum poly$(\log |M|)$ time. This generalizes a recent quantum algorithm of Arvind *et al.* which finds a basis representation for rings. We then show that

our algorithm is a useful primitive allowing quantum computers to determine the structure of a finite associative algebra as a direct sum of simple algebras. Moreover, it solves a wide variety of problems regarding finite modules and rings. Although our quantum algorithm is based on Abelian Fourier transforms, it solves problems regarding the multiplicative structure of modules and algebras, which need not be commutative. Examples include finding the intersection and quotient of two modules, finding the additive and multiplicative identities in a module, computing the order of an module, solving linear equations over modules, deciding whether an ideal is maximal, finding annihilators, and testing the injectivity and surjectivity of ring homomorphisms. These problems appear to be exponentially hard classically.

Thesis Supervisor: Joseph P. Brennan

Title: Professor of Mathematics

To my Mother, Father, Sara and Kaveh.

ACKNOWLEDGMENTS

I would like to thank all people who have helped and inspired me during my doctoral study.

I would like to express my gratitude to my advisor, Dr. Joseph Brennan, for his excellent guidance, caring and patience. Without his help, this work would not be possible. I would also like to thank the members of my committee, Eduardo Mucciolo and Dan Marinescu for their support, guidance and helpful suggestions. My special thanks goes to Xin Li for being on my committee and for all the help, advice, support and confidence he gave as the graduate coordinator.

I would like to take this opportunity to thank the Department of Mathematics, in particular the department chair Piotr Mikusinski, for making an encouraging and fun environment for graduate studies. I extend my gratitude to the department staff, especially Norma Robles, for their tireless efforts facilitating all official matters.

I would also like to thank my lab mates Chen-Fu Chang, Stephen Fulwider, Yiu Yu Ho, Bernd Losert, Abdullah Mahmud, Bo Sun, Greg Tener and Mahadevan Vasudevan, all my friends and the great people I had the chance to meet, especially my roommate Omid Madani, for all the fun times we have had together.

In particular, I want to express my deepest gratitude to my former advisor Pawel Wocjan for his assistance, help and inspiration in developing and writing this thesis. His ideas inspired my work in quantum computing and he provided both direction and advice for the

development of the ideas in this thesis. He also was of the greatest assistance in my graduate study by providing partial support as a Graduate Research Assistant through his National Science Foundation grants CCF-0726771 and CCF-0746600.

My family has been vital in helping me reach this accomplishment through their constant encouragement and support. There is really no way to fully express my gratitude to my parents for everything they have done for me. I dedicate this dissertation to them.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1
INTRODUCTION

In this dissertation, we find quantum algorithms for mathematical problems that are classically hard. First, we discuss the quantum complexity of evaluating the Tutte polynomial of a planar graph. Furthermore, we devise a new quantum phase estimation algorithm using arbitrary degree phase-shift operators. Finally, we provide quantum tools to find the structure of black-box modules and algebras.

1. Quantum Phase Estimation with Arbitrary Phase Shift Operators

In Chapter 3 we provide a new quantum algorithm for estimating the phase of a unitary matrix. Quantum phase estimation (QPE) plays a core role in many quantum algorithms [Hal07, Sho94, Sho97, Sze04, WCN09]. Some interesting algebraic and theoretic problems can be addressed by QPE, such as prime factorization [Sho94], discrete-log finding [Sho97], and order finding.

Problem. ***[Phase Estimation]*** *Let U be a unitary matrix with eigenvalue $e^{2\pi i\varphi}$ and corresponding eigenvector $|u\rangle$. Assume only a single copy of $|u\rangle$ is available, the goal is to find $\widetilde{\varphi}$ such that*

$$\Pr(|\widetilde{\varphi} - \varphi| < \frac{1}{2^n}) > 1 - c, \tag{1.1}$$

where c is a constant less than $\frac{1}{2}$.

In this dissertation we investigate a more general approach for the QPE algorithm. This approach completes the transition from Kitaev's original approach that requires no controlled

phase-shift operators, to QPE with approximate quantum Fourier transform (AQFT). The standard QPE algorithm utilizes the complete version of the inverse QFT. The disadvantage of the standard phase estimation algorithm is the high degree of phase-shift operators required. Since implementing exponentially small phase-shift operators is costly or physically not feasible, we need an alternative way to use lower precision operators. This was the motivation for AQFT being introduced — for lowering the cost of implementation while preserving high success probability.

In AQFT the number of required phase-shift operators drops significantly with the cost of lower success probability. Such compromise demands repeating the process extra times to achieve the final result. The QPE algorithm has a success probability of at least $\frac{8}{\pi^2}$ [KLM07]. Phase estimation using AQFT instead, with phase-shift operators up to degree m where $m > \log_2(n) + 2$, has success probability at least $\frac{4}{\pi^2} - \frac{1}{4n}$ [BES96, Che04].

On the other hand, Kitaev's original approach requires only the first phase-shift operator (as a single qubit gate not controlled). Comparing the existing methods, there is a gap between Kitaev's original approach and QPE with AQFT in terms of the degree of phase-shift operators needed. In this dissertation our goal is to fill this gap and introduce a more general phase estimation algorithm such that it is possible to realize a phase estimation algorithm with any degree of phase-shift operators in hand. In physical implementation of the phase estimation algorithm, the depth of the circuit should be small to avoid decoherence. Also, higher degree phase-shift operators are costly to implement and in many cases it is not physically feasible.

In this dissertation, we assume only one copy of the eigenvector $|u\rangle$ is available. This implies a restriction on the use of controlled-U gates that all controlled-U gates should be applied on one register. Thus, the entire process is a single circuit that can not be divided into parallel processes. Due to results by Griffiths and Niu, who introduced semi classical quantum Fourier transform [GN96], quantum circuits implementing different approaches discussed in this dissertation would require the same number of qubits.

2. Additive Approximation of the Tutte Polynomial

The Tutte polynomial $T(G; x, y)$ of a graph G is a two variable generalization of the chromatic polynomial and was first introduced by William T. Tutte in 1954 [Tut01]. An important feature of the Tutte polynomial is that it captures a lot of information about the graph G. For example, $T(G; 1, 1)$ counts the number of spanning trees of a connected graph G and $T(G; 2, 1)$ counts the number of forests in G. Reference [Wel93] provides a more extensive collection of graph properties that can be simply read off by evaluating the Tutte polynomial at suitable points.

The complexity of Tutte polynomial has been studied by many authors. We give some examples of the results that are relevant to our study. Jaeger, Vertigan and Welsh showed in [JVW90] that evaluating the Tutte polynomial exactly is #P-hard except for the points on the hyperbola $(x-1)(y-1) = 1$ and the four points $(x, y) \in \{(1, 1), (0, -1), (-1, 0), (-1, -1)\}$. Goldberg and Jerrum have recently shown in [GJ08] that for rational numbers (x, y) with $x < -1$ or $y < -1$ and not on the hyperbolas $H_n : (x - 1)(y - 1) = n$ where $n = 0, 1, 2$, there is no fully polynomial randomized approximation scheme (FPRAS) for approximately

computing $T(G; x, y)$ for general graphs G. They have also shown that for some other points there is no FPRAS; for more details see [GJ08].

There are also some results on efficient algorithms for approximately evaluating the Tutte polynomial of some special types of graphs. For example, Alon, Frieze and Welsh [AFW95] obtained FPRAS for dense graphs G for points (x, y) where $x > 1$ and $y > 1$. We refer the interested reader to [GJ08] for a review of such algorithms.

The above discussion shows that the problem of exactly and even approximately evaluating the Tutte polynomial is classically hard. In this dissertation we relate the problem of approximately evaluating the Tutte polynomial of some special types of graphs at certain points to quantum computing.

We consider two types of graphs, referred to as triangular and circular. We prove that the problem of providing an additive approximation for the evaluation of the Tutte polynomial of triangular graphs is BQP-complete. Roughly speaking, the complexity class BQP (Bounded error Quantum Polynomial time) is the class of problems that can be solved efficiently on a quantum computer. DQC1 is a quantum complexity class that is contained in BQP. The difference between DQC1 and BQP is that in DQC1 only one qubit can be initialized in the state $|0\rangle$ and all other qubits are in a completely random (maximally mixed) state [DFC05]. This "one clean qubit model" was first introduced by Knill and Laflamme in [KL98]. Our proof establishes that it suffices to consider only triangular graphs to achieve BQP-hardness. We also show that the problem of providing an additive approximation of the Tutte polynomial for circular graphs is in DQC1 at the point $(e^{2\pi i/5}, e^{-2\pi i/5})$ and in BQP for

points $(q^k, 1 + \frac{1-q^{-k}}{(q^{1/2}-q^{-1/2})^2})$, where q is a root of unity. To prove these results, we establish a connection between the problems of approximately evaluating the Tutte polynomial of triangular and circular graphs and that of approximately evaluating the Jones and HOMFLY polynomial of plat and trace closures of braids, respectively. It is known that the latter are related to the quantum complexity classes BQP [AJL06, FKW02, FLW02, KL07a, KL07b, LK06, WY08] and DQC1 [SJ08].

More precisely, we establish this connection as follows:

- Given an arbitrary braid b we show how to efficiently construct a braid $\tilde{b}$ such that its plat closure $\tilde{b}^{\mathrm{p}lat}$ is an alternating link and

$$J(b^{\mathrm{p}lat}; q) = J(\tilde{b}^{\mathrm{p}lat}; q)\,, \tag{1.2}$$

  where $J(L; q)$ denotes the evaluation of the Jones polynomial of the link L at q. This construction relies upon the fact that the images of braid group generators under all irreducible Jones-Wenzl representations of the braid group B_n have finite order.

- We construct a triangular graph G such that the alternating link $D(G) = \tilde{b}^{\mathrm{p}lat}$ corresponds to the medial graph $M(G)$ of G. Using the connection between the Tutte and Jones polynomials [Thi87], we obtain

$$T(G; q, 1/q) = \alpha(G) J(\tilde{b}^{\mathrm{p}lat}; q)\,, \tag{1.3}$$

where $\alpha(G)$ is complex number of modulus one that is easily computed. These arguments establish that the ability to approximately evaluate the Tutte polynomial of triangular graphs implies the ability to approximately evaluate the Jones polynomial of plat closure of braids. Since the latter problem is already known to be BQP-hard, we see that the approximate evaluation of the Tutte polynomial is also BQP-hard.

- The other direction, i.e., the proof that the problem of approximately evaluating the Tutte polynomials of triangular graphs is in BQP, is obtained using the above arguments.

- In the case of DQC1, we have to consider the trace closure instead of the plat closure. The structure of the proofs remains the same. We make use of the result by Shor and Jordan [SJ08] that evaluating the Jones polynomial of the trace closure of braids at the fifth root of unity is DQC1-complete.

3. Quantum Algorithms for Black-box Structures

In the final chapter of the dissertation (Chapter 5), we provide quantum algorithms in order to find the structure of black-box modules and algebras. Suppose we are given black-box access to a finite module M or algebra over a finite ring R, and a list of generators for M and R. Here we present an oracle for finding structure constants for black-box mathematical structures consisting an Abelian group. Finding the structure constants is classically hard and needs too many queries to the black-box. Moreover we find quantum algorithms for

several problems regarding finite black-box modules, which need not be commutative. All of the algorithms run in time scaling polylogarithmically in the size of the module.

A module is normally specified by a set of elements that generate the module via linear combination of its elements and multiplication by ring elements. To apply the known quantum techniques for abelian groups we find sets that generate rings and modules *as Abelian groups*, that is, by linear combination only. The problem of finding such a generating set for rings has been already solved by Arvind *et al.*[ADM06]. Our solution for modules and algebras generalizes their result.

As shown in [KS05], both integer factorization and graph isomorphism reduce to the problem of counting automorphisms of rings. The decision version of this counting problem is contained in AM∩coAM. Therefore it is unlikely to be NP-hard. Integer factorization also reduces to the problem of finding nontrivial automorphisms of rings and to the problem of finding isomorphisms between two rings. Furthermore, graph isomorphism reduces to ring isomorphism for commutative rings. Thus these ring automorphism and isomorphism problems are attractive targets for quantum computation. Perhaps the quantum algorithms given in this dissertation can serve as steps toward efficient quantum algorithms for some of these problems.

Many quantum algorithms, including those for factoring, discrete logarithms, and Pell's equation are based on the efficient solution to the Abelian hidden subgroup problem. Significant effort has been devoted to finding efficient quantum algorithms for non-Abelian generalizations of the hidden subgroup problem. In particular, the graph isomorphism prob-

lem reduces to the hidden subgroup problem for the symmetric group. Our algorithms for modules and algebras can be viewed as a new approach to obtain quantum algorithms for non-Abelian problems. Although our quantum algorithms at core rely on the efficient solution to the Abelian hidden subgroup problem by quantum Fourier transforms, they nevertheless efficiently solve several problems involving the potentially non-Abelian multiplicative structure of the ring.

CHAPTER 2
PRELIMINARIES

2.1 Quantum Computing

Quantum computers can outperform classical computers by executing special algorithms. One of the best examples is Shor's efficient quantum algorithm for factoring integers. Integer factorization is known to be an NP problem in classical computation, which means finding a solution efficiently is difficult. Therefore, the major challenge in quantum computation is finding new quantum algorithm which can be used to solve problems significantly faster than classical computation.

A *bit* is the basic unit of information and the fundamental concept of classical computation and classical information. An analogous concept used in quantum computation and quantum information is the quantum bit (*qubit*).

A classical bit is always in one of the two states 0 or 1. Assume a two-dimensional Hilbert space with basis $\{|0\rangle, |1\rangle\}$ and complex coefficients. A qubit also has a state but is a unit vector inside this vector space, which means its a linear combination of basis $|0\rangle$ and $|1\rangle$ with complex coefficients α and β such as

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle \tag{2.1}$$

The states $|0\rangle$ and $|1\rangle$ are known as computational basis states, and form an orthonormal basis for this vector space. The classical case has only one computational basis, while in the quantum case any orthonormal basis of the space can be a computational basis too. For example the following two states also form a basis,

$$|+\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle), \qquad |-\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle). \tag{2.2}$$

In classical computation we can always read the true value of a bit without disturbing it but in quantum computation we cannot examine a qubit to determine its quantum state that means we cannot read the values α and β without destroying the superposition. We can only read the value of a qubit by measuring it which is projecting it onto one of the computational basis. When we measure a qubit we get either the result 0, with probability $|\alpha|^2$, or the result 1, with probability $|\beta|^2$. Therefore, $|\alpha|^2 + |\beta|^2 = 1$, since the probabilities must sum to one. Thus, in general a qubit is a unit vector in a two-dimensional Hilbert space.

Now lets consider multiple qubits. In the classical case two bits would be in one of the four possible states, 00, 01, 10, and 11. Correspondingly, a two qubit system has four computational basis states such as $|00\rangle$, $|01\rangle$, $|10\rangle$, $|11\rangle$. Therefore a two qubit state is a superposition of these four states. Hence a two qubit state can be described as the vector

$$\alpha_{00}|00\rangle + \alpha_{01}|01\rangle + \alpha_{10}|10\rangle + \alpha_{11}|11\rangle, \tag{2.3}$$

where the coefficients α_{ij} are complex numbers and called amplitudes.

Similar to the case for a single qubit, if we measure a two qubit state the post measurement state $|x\rangle$ will be one of the four possible choices $\{|00\rangle, |01\rangle, |10\rangle, |11\rangle\}$ which occurs with probability $|\alpha_x|^2$. Similarly, the condition that probabilities must sum to one should also be satisfied in this case therefore

$$\sum_{x\in\{0,1\}^2} |\alpha_x|^2 = 1. \tag{2.4}$$

Quantum gates act on qubits and transfer a quantum state to another quantum state. A quantum gate on a single qubit can be represented by a unitary matrix. The normalization condition requires $|\alpha|^2 + |\beta|^2 = 1$ for a quantum state $\alpha|0\rangle + \beta|1\rangle$. This must also be true of the quantum state $|\psi\rangle = \alpha'|0\rangle + \beta'|1\rangle$ after the gate has acted. This means that the quantum gate should preserve the norm of the vector which acts on. Its easy to show that this is a sufficient and necessary condition for a quantum gate to be unitary. By other means a quantum gate U should satisfy the property $U^\dagger U = I$, where $U^\dagger$ is the complex conjugate transpose of the matrix U, and I is the two by two identity matrix.

A commonly used quantum gate is the Hadamard gate H,

$$H = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}. \tag{2.5}$$

Other frequently used gates are Pauli gates,

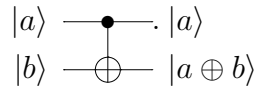
$$X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \quad Y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}, \quad Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}. \tag{2.6}$$

The gate X is also known as the NOT-gate.

Similarly, a multi qubit quantum gate is nothing other than a unitary matrix that acts on a multi qubit. The prototypical multi-qubit quantum logic gate is the controlled-NOT or CNOT gate. This gate has two input qubits, known as the control qubit and the target qubit, respectively. The matrix representative for the CNOT is

$$CNOT = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{2.7}$$

and the corresponding circuit would be as follows:



Unitary quantum gates are always invertible, since the inverse of a unitary matrix is also a unitary matrix, and thus a quantum gate can always be inverted by another quantum gate. So the action of any gate on a quantum state is always reversible.

It has been shown that single quantum gates and the CNOT gate are universal for quantum computation. This means we can approximate any unitary transformation to arbitrary accuracy with a quantum circuit using only single qubit gates and the CNOT gate.

Measuring a qubit is not always projecting it on a computational basis. Quantum measurements can be described by a collection $\{M_m\}$ of operators that act on the state space of the system. This collection should satisfy the following condition

$$\sum_m M_m^\dagger M_m = I. \tag{2.8}$$

This condition is called the completeness equation.

The probability of an outcome m for a measurement M_m on the quantum system at the state $|\phi\rangle$ is given by

$$\Pr(m) = \langle\phi|M_m^\dagger M_m|\phi\rangle \tag{2.9}$$

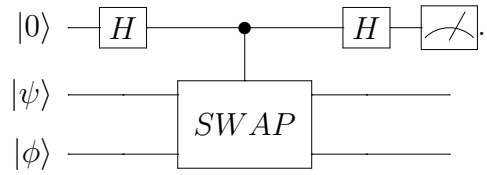
and the state of the system after the measurement is

$$\frac{M_m|\phi\rangle}{\langle\phi|M_m^\dagger M_m|\phi\rangle}. \tag{2.10}$$

By the completeness equation we see that the probability of all the possible outcomes sum to one:

$$1 = \sum_m \Pr(m) = \sum_m \langle\phi|M_m^\dagger M_m|\phi\rangle. \tag{2.11}$$

For comparing two quantum states we can estimate their inner product by sampling the swap test sufficient many number of times. If the two states are exactly the same their inner product will be one and if they are orthogonal their inner product is zero. The circuit of the SWAP test is as follows:



The resulting state $|f\rangle$ before the measurement is

$$|f\rangle = \frac{1}{2}\Big[|0\rangle\big(|\phi\rangle|\psi\rangle + |\psi\rangle|\phi\rangle\big) + |1\rangle\big(|\phi\rangle|\psi\rangle - |\psi\rangle|\phi\rangle\big)\Big]. \tag{2.12}$$

Therefore, the probabilities for getting either 0 or 1 from measuring the first qubit is

$$\Pr(0) = \frac{1}{2}(1 + |\langle\phi|\psi\rangle|^2), \tag{2.13}$$

$$\Pr(1) = \frac{1}{2}(1 - |\langle\phi|\psi\rangle|^2). \tag{2.14}$$

So, as we see, if both states $|\phi\rangle$ and $|\psi\rangle$ are exactly the same the SWAP test will output 0 with probability 1, and if not it might output the wrong value. Therefore, with repeating this process sufficiently many times with high probability, we are able to decide whether these two states are the same or not.

Another interesting result regarding black-box groups is due to Watrous[Wat01]. The results of his work comes in the form of the following theorem.

Theorem 1 ([Wat01]). *There exists a quantum algorithm operating as follows (relative to an arbitrary group oracle). Given generators $g_1, \ldots, g_k$ such that $G = \langle g_1, \ldots, g_k \rangle$ is solvable, the algorithm outputs the order of G with probability of error bounded by ϵ in time polynomial in $n + log(1/\epsilon)$ (where n is the length of the strings representing the generators). Moreover, the algorithm produces a quantum state ρ that approximates the pure state $|G\rangle = |G|^{-1/2} \sum_{g \in G} |g\rangle$ with accuracy ϵ (in the trace norm metric).*

2.2 Knot Theory

A knot is an embedding of a circle in 3-dimensional Euclidean space $\mathbb{R}^3$ (by some definitions $\mathbb{S}^3$). Two knots are equivalent if one can be transformed into the other via an ambient isotopy which is a deformation of $\mathbb{R}^3$ upon itself. In other words, these transformations correspond to moving the knot in 3-dimensional space which does not involve cutting the string or passing the string through itself.

A mathematical knot is the same as the knot we know in real world. Pick a peace of string, wrap it around itself arbitrarily and then connect the two ends together to form a closed loop. The easiest way to create a knot is the same way we know knots in real world. A knot is created by beginning with a one-dimensional line segment, wrapping it around itself arbitrarily, and then fusing its two free ends together to form a closed loop [Ada00, Sos02].

The most important problem in knot theory is determining the equivalence of two knots. There are many algorithms that solve this problem, the first algorithm is due to Wolfgang

Haken in 1960 [Has98]. The running time of an algorithm for knots is measured by the number of its crossing as the input. All algorithms known to date have running times of at least NP-hard, which makes them extremely time-consuming. One of the most important problems in knot theory is to understand how hard this problem really is.

The most common way to visualize knots is to project the knot onto a 2-dimensional space similar to the shadow of an object over a table. Assume there is a source of light above a table. If the knot is between the source and the table it will create a shadow on the table. With moving the knot a little, it is easy to create a one to one correspondence between the knot and the shadow except at double points. Points in which the knot crosses itself called crossings [Rol90]. At each crossing there is a over-strand and under-strand, by creating a break in the strand going underneath the two strands can be distinguished from each other.

Two knot diagrams are equivalent (belong to the same knot) if one can be transformed into the other by a sequence of three kinds of moves called the Reidemeister moves. These moves are depicted in Figure 2.2

Twist and untwist in either direction. Move one strand completely over another. Move a strand completely over or under a crossing.

A knot invariant is a *quantity* that is the same for equivalent knots [Ada00, Lic97, Rol90]. This means, if two knot diagrams are equivalent the value given by the invariant should be equal for both knots. note that the invariant is equal for same knots but it does not necessarily mean that it would be different for inequivalent knots. There are examples of

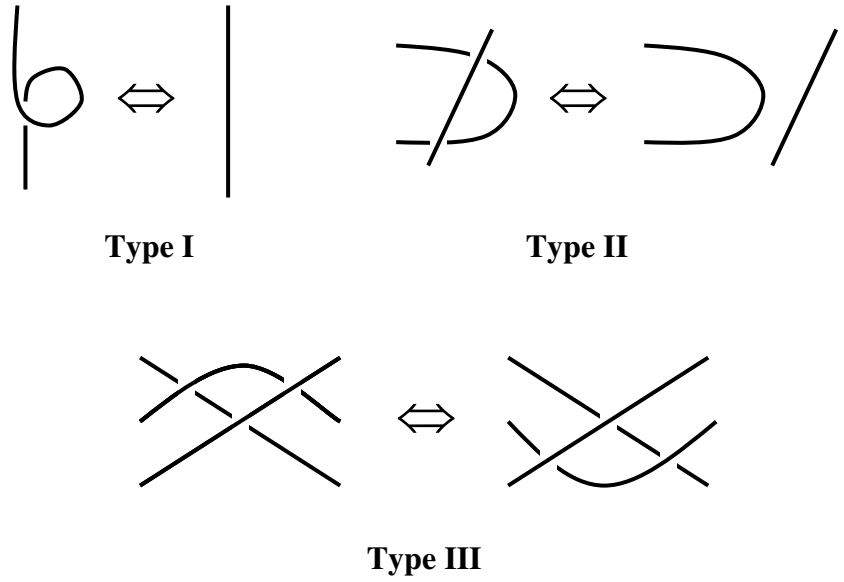


Figure 2.1: Reidemeister moves

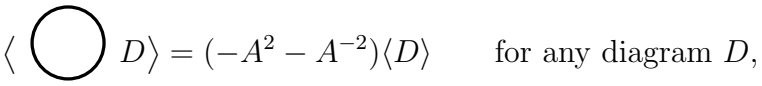
knots and invariants that the knots are not equivalent but have the same invariant. In other words, invariants are incapable to distinguish all knots.

The very first known knot invariants are the knot group and the Alexander polynomial. The knot group is the fundamental group of the knot complement and the Alexander polynomial can be computed from the Alexander invariant which is a module constructed from the infinite cyclic cover of the knot complement [Lic97, Rol90]. More recent knot invariants are quantum knot polynomials, Vassiliev invariants and hyperbolic invariants.

2.2.1 Polynomial invariants of knots

A knot polynomial is a knot invariant that is a polynomial. The most well known and used knot polynomial is the Jones polynomials. Here we first introduce the Kauffman bracket which is the building block of the Jones polynomial.

The *Kauffman bracket* of unoriented link diagrams is defined by the following recursive relations,



$\langle$the empty diagram $\emptyset\rangle = 1$,

where three pictures in the first formula imply three links diagrams, which are identical except in one small region where they differ by the crossing changes. The Jones polynomial can be defined using the Kauffman bracket. The Jones polynomial $V_L(t)$ (which is a Laurent polynomial in the variable $t^{1/2}$) of an oriented link L is defined by

$$V_L(t) = (-A^2 - A^{-2})^{-1}(-A^3)^{-w(D)}\langle D\rangle\Big|_{A^2=t^{-1/2}} \in \mathbb{Z}[t^{1/2}, t^{-1/2}],$$

where D is a diagram of L, $w(D)$ is the writhe of D, and $\langle D\rangle$ is the Kauffman bracket of D with its orientation forgotten. The Jones polynomial is an isotopy invariant of oriented

links uniquely characterized by

$$t^{-1}V_{L_+}(t) - tV_{L_-}(t) = (t^{1/2} - t^{-1/2})V_{L_0}(t), \tag{2.15}$$
$$V_O(t) = 1,$$

where O denotes the trivial knot, and L_+, L_-, and L_0 are three oriented links, which are identical except in one small region where they differ by the crossing changes or smoothing as shown in the Figure 2.2. It is shown, by (2.15), that for any knot K, its Jones polynomial $V_K(t)$ belongs to $\mathbb{Z}[t, t^{-1}]$.

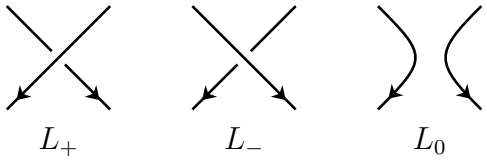


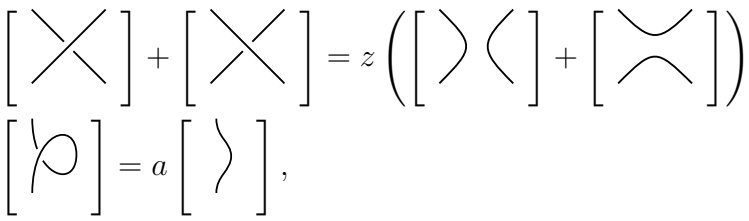
Figure 2.2: Three links L_+, L_-, L_0

Another interesting knot polynomial is the HOMFLY polynomial. The *skein polynomial* (or the *HOMFLY polynomial*) $P_L(l, m) \in \mathbb{Z}[l^{\pm 1}, m^{\pm 1}]$ of an oriented link L is uniquely characterized by

$$l^{-1}P_{L_+}(l, m) - lP_{L_-}(l, m) = mP_{L_0}(l, m),$$
$$P_O(l, m) = 1,$$

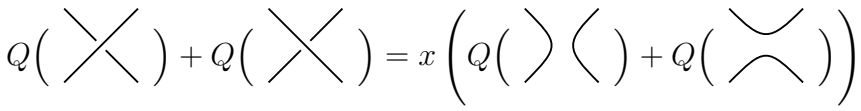
where O denotes the trivial knot, and L_+, L_-, and L_0 are three oriented links, which are identical except in one small region where they differ by the crossing changes or smoothing

as shown in the Figure 2.2. For a knot K, $P_K(l,m) \in \mathbb{Z}[l^{\pm 2}, m]$. The *Kauffman polynomial* $F_L(a,z) \in \mathbb{Z}[a^{\pm 1}, z^{\pm 1}]$ of an oriented link L is defined by $F_L(a,z) = a^{-w(D)}[D]$ for an unoriented diagram D presenting L (forgetting its orientation), where $[D]$ is uniquely characterized by



$$[O] = 1.$$

For a knot K, $F_K(a,z) \in \mathbb{Z}[a^{\pm 1}, z]$. The *Q polynomial* $Q_L(x) \in \mathbb{Z}[x^{\pm 1}]$ of an unoriented link L is uniquely characterized by



$$Q(O) = 1.$$

It is known that

$$V_L(t) = P_L(t, t^{1/2} - t^{-1/2}) = F_L(-t^{-3/4}, t^{1/4} + t^{-1/4}),$$

$$\Delta_L(t) = P_L(1, t^{1/2} - t^{-1/2}),$$

$$Q_L(z) = F_L(1, z),$$

where $\Delta_L(t)$ denotes the Alexander polynomial of L. The variable m of $P_L(l, m)$ is called the *Alexander variable* [Kaw96, Lic97].

2.2.2 Tutte Polynomial of a Graph

A graph is a set of vertices V and edges E, where vertices are points and edges are line segments in which connect vertices together. In this section, we introduce a polynomial corresponding to graphs called the Tutte Polynomial [Tut04]. The Tutte polynomial can be defined as

$$T_G(x, y) = \sum_{F \subseteq E} (x-1)^{c(F)-c(E)} (y-1)^{c(F)+|F|-|V|}. \tag{2.16}$$

Here, G is a graph with vertex set V and edge set E; The number of connected components in the graph with vertex set V and edge set F is shown by $c(F)$. A connected component of a graph is a subgraph in which any two vertices are connected to each other by an edge.

The Tutte polynomial can also be defined using a deletion-contraction recurrence. The edge contraction G/uv of graph G is the graph obtained by merging the vertices u and v and removing the edge uv. If only the edge uv is removed from the graph we write denote it by $G - uv$. The Tutte polynomial is defined by the following recurrence relation. $T_G = 1$ if G contains no edges. If G contains i bridges and j loops and no other edges

$$T_G(x, y) = x^i y^j \tag{2.17}$$

otherwise, if e is neither a loop nor a bridge

$$T_G = T_{G-e} + T_{G/e}. \tag{2.18}$$

CHAPTER 3
QUANTUM PHASE ESTIMATION WITH ARBITRARY PHASE SHIFT OPERATORS

Quantum Phase Estimation (QPE) plays a core role in many quantum algorithms [Hal07, Sho94, Sho97, Sze04, WCN09]. Some interesting algebraic and theoretic problems can be addressed by QPE, such as prime factorization [Sho94], discrete-log finding [Sho97], and order finding.

Problem. ***[Phase Estimation]*** *Let U be a unitary matrix with eigenvalue $e^{2\pi i\varphi}$ and corresponding eigenvector $|u\rangle$. Assume only a single copy of $|u\rangle$ is available, the goal is to find $\widetilde{\varphi}$ such that*

$$\Pr(|\widetilde{\varphi} - \varphi| < \frac{1}{2^n}) > 1 - c, \tag{3.1}$$

where c is a constant less than $\frac{1}{2}$.

In this chapter we investigate a more general approach for the QPE algorithm. This approach completes the transition from Kitaev's original approach that requires no controlled phase shift operators, to QPE with approximate quantum Fourier transform (AQFT). The standard QPE algorithm utilizes the complete version of the inverse QFT. The disadvantage of the standard phase estimation algorithm is the high degree of phase shift operators required. Since implementing exponentially small phase shift operators is costly or physically not feasible, we need an alternative way to use lower precision operators.

The structure of this chapter is organized as follows. In Section 3.1 we give a brief overview on existing approaches, such as Kitaev's original algorithm and standard phase estimation algorithm based on QFT and AQFT. In Section 3.2 we introduce our new approach and discuss the requirements to achieve the same performance output (success probability) as the methods above. Finally, we make our conclusion and compare with other methods.

This chapter is based on the paper *Quantum Phase Estimation with Arbitrary Constant-precision Phase Shift Operators* co-authored with Chen-Fu Chiang [AC].

3.1 Quantum Phase Estimation Algorithms

3.1.1 Kitaev's Original Approach

Kitaev's original approach is one of the first quantum algorithms for estimating the phase of a unitary matrix [KSV02]. Let U be a unitary matrix with eigenvalue $e^{2\pi i\varphi}$ and corresponding eigenvector $|u\rangle$ such that

$$U|u\rangle = e^{2\pi i\varphi}|u\rangle. \tag{3.2}$$

In this approach, a series of Hadamard tests are performed. In each test the phase $2^{k-1}\varphi$ $(1 \leq k \leq n)$ will be computed up to precision $1/16$. Assume an n-bit approximation is desired. Starting from $k = n$, in each step the kth bit position is determined consistently from the results of previous steps.

For the kth bit position, we perform the Hadamard test depicted in Figure 3.1, where the gate $K = I_2$. Denote $\varphi_k = 2^{k-1}\varphi$, the probability of the post measurement state is

$$\Pr(0|k) = \frac{1 + \cos(2\pi\varphi_k)}{2}, \quad \Pr(1|k) = \frac{1 - \cos(2\pi\varphi_k)}{2}. \tag{3.3}$$

In order to recover φ_k, we obtain more precise estimates with higher probabilities by iterating the process. But, this does not allow us to distinguish between φ_k and $-\varphi_k$. This can be solved by the same Hadamard test in Figure 3.1, but instead we use the gate

$$K = \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix}. \tag{3.4}$$

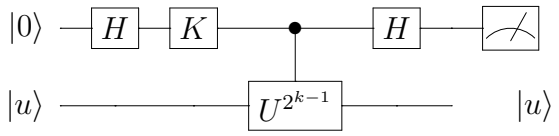


Figure 3.1: Hadamard test with extra phase shift operator.

The probabilities of the post-measurement states based on the modified Hadamard test become

$$\Pr(0|k) = \frac{1 - \sin(2\pi\varphi_k)}{2}, \quad \Pr(1|k) = \frac{1 + \sin(2\pi\varphi_k)}{2}. \tag{3.5}$$

Hence, we have enough information to recover φ_k from the estimates of the probabilities.

In Kitaev's original approach, after performing the Hadamard tests, some classical post processing is also necessary. Suppose $\varphi = 0.x_1x_2\ldots x_n$ is an exact n-bit. If we are able to

determine the values of φ, $2\varphi, \ldots,$ $2^{n-1}\varphi$ with some constant-precision (1/16 to be exact), then we can determine φ with precision $1/2^n$ efficiently [Kit96, KSV02].

Starting with φ_n we increase the precision of the estimated fraction as we proceed toward φ_1. The approximated values of $\varphi_k\,(k = n, \ldots, 1)$ will allow us to make the right choices.

For $k = 1, \ldots, n$ the value of φ_k is replaced by β_k, where β_k is the closest number chosen from the set $\{\frac{0}{8}, \frac{1}{8}, \frac{2}{8}, \frac{3}{8}, \frac{4}{8}, \frac{5}{8}, \frac{6}{8}, \frac{7}{8}\}$ such that

$$|\varphi_k - \beta_k|_{\mathrm{mod}\ 1} < \frac{1}{8}. \tag{3.6}$$

The result follows by a simple iteration. Let $\beta_n = \overline{0.x_n x_{n+1} x_{n+2}}$ and proceed by the following iteration:

$$x_k = \begin{cases} 0 & \text{if } |\overline{0.0x_{k+1}x_{k+2}} - \beta_k|_{\mathrm{mod}\ 1} < 1/4 \\ 1 & \text{if } |\overline{0.1x_{k+1}x_{k+2}} - \beta_k|_{\mathrm{mod}\ 1} < 1/4 \end{cases} \tag{3.7}$$

for $k = n-1, \ldots, 1$. By using simple induction, the result satisfies the following inequality:

$$|\overline{0.x_1 x_2 \ldots x_{n+2}} - \varphi|_{\mathrm{mod}\ 1} < 2^{-(n+2)}. \tag{3.8}$$

In Eq. 3.6, we do not have the exact value of φ_k. So, we have to estimate this value and use the estimate to find β_k. Let $\widetilde{\varphi_k}$ be the estimated value and

$$\epsilon = |\widetilde{\varphi_k} - \varphi_k|_{\mathrm{mod}\ 1} \tag{3.9}$$

be the estimation error. Now we use the estimate to find the closest β_k. Since we know the exact binary representation of the estimate $\widetilde{\varphi_k}$, we can choose β_k such that

$$|\widetilde{\varphi_k} - \beta_k|_{\text{mod } 1} \leq \frac{1}{16}. \tag{3.10}$$

By the triangle inequality we have,

$$|\varphi_k - \beta_k|_{\text{mod } 1} \leq |\widetilde{\varphi_k} - \varphi_k|_{\text{mod } 1} + |\widetilde{\varphi_k} - \beta_k|_{\text{mod } 1} \leq \epsilon + \frac{1}{16}. \tag{3.11}$$

To satisfy Eq. 3.6, we need to have $\epsilon < 1/16$, which implies

$$|\widetilde{\varphi_k} - \varphi_k|_{\text{mod } 1} < \frac{1}{16}. \tag{3.12}$$

Therefore, it is required for the phase to be estimated with precision 1/16 at each stage.

In the first Hadamard test (Eq. 3.3), in order to estimate $\Pr(1|k)$ an iteration of Hadamard tests should be applied to obtain the required precision of 1/16 for φ_k. This is done by counting the number of states $|1\rangle$ in the post measurement state and dividing that number by the total number of iterations performed.

The Hadamard test outputs $|0\rangle$ or $|1\rangle$ with a fixed probability. We can model an iteration of Hadamard tests as Bernoulli trials with success probability (obtaining $|1\rangle$) being p_k. The best estimate for the probability of obtaining the post measurement state $|1\rangle$ with t samples

is

$$\widetilde{p_k} = \frac{h}{t}, \tag{3.13}$$

where h is the number of ones in t trials. This can be proved by Maximum Likelihood Estimation (MLE) methods [HS98].

In order to find $\sin(2\pi\varphi_k)$ and $\cos(2\pi\varphi_k)$, we can use estimates of probabilities in Eq. 3.3 and Eq. 3.5. Let s_k be the estimate of $\sin(2\pi\varphi_k)$ and t_k the estimate of $\cos(2\pi\varphi_k)$. It is clear that if

$$|\widetilde{p_k} - p_k| < \epsilon_0, \tag{3.14}$$

then

$$|s_k - \sin(2\pi\varphi_k)| < 2\epsilon_0, \quad |t_k - \cos(2\pi\varphi_k)| < 2\epsilon_0. \tag{3.15}$$

Since the inverse tangent function is more robust to error than the inverse sine or cosine functions, we use

$$\widetilde{\varphi_k} = \frac{1}{2\pi} \arctan\left(\frac{s_k}{t_k}\right) \tag{3.16}$$

as the estimation of φ_k. By Eq. 3.12 we should have

$$\left|\varphi_k - \frac{1}{2\pi} \arctan\left(\frac{s_k}{t_k}\right)\right|_{\text{mod } 1} < \frac{1}{16}. \tag{3.17}$$

The inverse tangent function can not distinguish between the two values φ_k and $\varphi_k \pm 1/2$. However, because we find estimates of the sine and cosine functions as well, it is easy to

determine the correct value. The inverse tangent function is most susceptible to error when φ_k is in the neighborhood of zero and the reason is that the derivative is maximized at zero. Thus, if

$$|s_k - \sin(2\pi\varphi_k)| = \epsilon_1 \quad \text{and} \quad |t_k - \cos(2\pi\varphi_k)| = \epsilon_2, \tag{3.18}$$

considering the case where $\varphi_k = 0$, then we have

$$\frac{1}{2\pi}\left|\arctan\left(\frac{\epsilon_1}{1 \pm \epsilon_2}\right)\right| < \frac{1}{16}. \tag{3.19}$$

By simplifying the above inequality, we have

$$\left|\frac{\epsilon_1}{1 \pm \epsilon_2}\right| < \tan(\frac{\pi}{8}). \tag{3.20}$$

With the following upper bounds for ϵ_1 and ϵ_2, the inequality above is always satisfied when

$$|\epsilon_1| < 1 - \frac{1}{\sqrt{2}} \quad \text{and} \quad |\epsilon_2| < 1 - \frac{1}{\sqrt{2}}. \tag{3.21}$$

Therefore, in order to estimate the phase φ_k with precision 1/16, the probabilities in Eq. 3.3 and Eq. 3.5 should be estimated with error at most $(2-\sqrt{2})/4$ which is approximately 0.1464. In other words, it is necessary to find the estimate of $\Pr(1|k)$ such that

$$\left|\Pr(1|k) - \frac{h}{t}\right| < \frac{2 - \sqrt{2}}{4} \approx 0.1464. \tag{3.22}$$

There are different ways we can guarantee an error bound with constant probability. The first method, used in [KSV02], is based on the Chernoff bound. Let $X_1, \ldots, X_m$ be Bernoulli random variables, by Chernoff's bound we have

$$\Pr\left(\left|\frac{1}{m}\sum_{i=0}^{m} X_i - p_k\right| \geq \delta\right) \leq 2e^{-2\delta^2 m}, \tag{3.23}$$

where in our case the estimate is $\widetilde{p_k} = \frac{1}{m}\sum_{i=0}^{m} X_i$. Since we need an accuracy up to 0.1464, we get

$$\Pr\left(|\widetilde{p_k} - p_k| > 0.1464\right) < 2e^{-(0.0429)m}. \tag{3.24}$$

In order to obtain

$$\Pr\left(|\widetilde{p_k} - p_k| < 0.1464\right) > 1 - \frac{\varepsilon}{2}, \tag{3.25}$$

a minimum of m_1 trials is sufficient when

$$\begin{aligned} m_1 &\approx 24 \ln \frac{4}{\varepsilon} \\ &\approx 33 + 24 \ln \frac{1}{\varepsilon} \end{aligned} \tag{3.26}$$

This is the number of trials for each Hadamard test, as we have two Hadamard tests at each stage. Therefore, in order to have

$$\Pr\left(|\widetilde{\varphi_k} - \varphi_k| < \frac{1}{16}\right) > 1 - \varepsilon. \tag{3.27}$$

we require a minimum of

$$\begin{aligned} m &= 2m_1 \\ &\approx 47 \ln \frac{4}{\varepsilon} \\ &\approx 66 + 47 \ln \frac{1}{\varepsilon} \end{aligned} \tag{3.28}$$

many trials.

In the analysis above, we used the Chernoff bound, which is not a tight bound. If we want to obtain the result with a high probability, we need to apply a large number of Hadamard tests. In this case, we can use an alternative method to analyze the process by employing methods of statistics [SS06].

Iterations of Hadamard tests have a Binomial distribution which can be approximated by a normal distribution. This is a good approximation when p is close to $1/2$ or $mp > 10$ and $m(1-p) > 10$, where m is the number of iterations and p the success probability. In other words, if we see 10 successes and 10 fails in our process, we can use this approximation to obtain a better bound.

In Kitaev's algorithm each Hadamard test has to be repeated a sufficient number of times to achieve the required accuracy with high probability. Because only one copy of $|u\rangle$ is available, all controlled-U gates have to be applied to one register. Therefore, all the Hadamard tests have to be performed in sequence, instead of parallel, during one run of the

circuit. A good example for this case is the order finding algorithm. We refer the reader to [NC00] for more details.

In Kitaev's approach, there are n different Hadamard tests that should be performed. Thus, if the probability of error in each Hadamard test is ε_0, by applying the union bound, the error probability of the entire process is $\varepsilon = n\varepsilon_0$. Therefore, in order to obtain

$$\Pr(|\varphi - \widetilde{\varphi}| < \frac{1}{2^n}) > 1 - \varepsilon, \tag{3.29}$$

for approximating each bit we need m trials where

$$m = 47 \ln \frac{4n}{\varepsilon}. \tag{3.30}$$

Since, all of these trials have to be done in one circuit, the circuit consists of mn Hadamard tests. Therefore the circuit involves mn controlled-U^{2^k} operations. As a result, if a constant success probability is desired, the depth of the circuit will be $O(n \log n)$.

3.1.2 Approach Based on QFT

One of the standard methods to approximate the phase of a unitary matrix is QPE based on QFT. The structure of this method is depicted at Figure 3.2. The QPE algorithm requires two registers and contains two stages. If an n-bit approximation of the phase φ is desired, then the first register is prepared as a composition of n qubits initialized in the

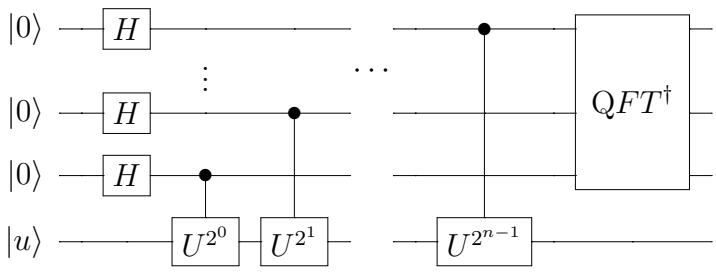


Figure 3.2: Standard Quantum Phase Estimation.

state $|0\rangle$. The second register is initially prepared in the state $|u\rangle$. The first stage prepares a uniform superposition over all possible states and then applies controlled-U^{2^k} operations. Consequently, the state will become

$$\frac{1}{2^{n/2}} \sum_{k=0}^{2^n-1} e^{2\pi i \varphi k} |k\rangle. \tag{3.31}$$

The second stage in the QPE algorithm is the QFT† operation.

There are different ways to interpret the inverse Fourier transform. In the QPE algorithm, the post-measurement state of each qubit in the first register represents a bit in the final approximated binary fraction of the phase. Therefore, we can consider computing each bit as a step. The inverse Fourier transform can be interpreted such that at each step (starting from the least significant bit), using the information from previous steps, it transforms the state

$$\frac{1}{\sqrt{2}}(|0\rangle + e^{2\pi i 2^k \varphi}|1\rangle) \tag{3.32}$$

to get closer to one of the states

$$\begin{aligned} \frac{1}{\sqrt{2}}(|0\rangle + e^{2\pi i 0.0}|1\rangle) &= \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \\ \text{or} & \\ \frac{1}{\sqrt{2}}(|0\rangle + e^{2\pi i 0.1}|1\rangle) &= \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle). \end{aligned} \tag{3.33}$$

Assume we are at step k in the first stage. By applying controlled-U^{2^k} operators due to phase kick back, we obtain the state

$$\frac{|0\rangle + e^{2\pi i 0.x_{k+1}x_{k+2}\ldots x_n}|1\rangle}{\sqrt{2}}. \tag{3.34}$$

Shown in Figure 3.3, each step (dashed-line box) uses the result of previous steps, where phase shift operators are defined as

$$R_k \equiv \begin{bmatrix} 1 & 0 \\ 0 & e^{2\pi i/2^k} \end{bmatrix} \tag{3.35}$$

for $2 \leq k \leq n$.

By using the previously determined bits $x_{k+2}, \ldots, x_n$ and the action of corresponding controlled phase shift operators (as depicted in Figure 3.3) the state in Eq. 3.34 becomes

$$\frac{|0\rangle + e^{2\pi i 0.x_{k+1}0\ldots 0}|1\rangle}{\sqrt{2}} = \frac{|0\rangle + (-1)^{x_{k+1}}|1\rangle}{\sqrt{2}}. \tag{3.36}$$

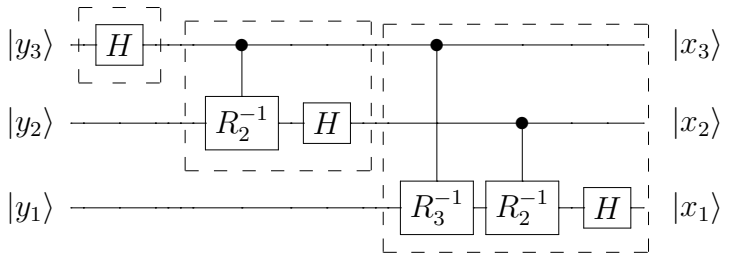


Figure 3.3: 3-qubit inverse QFT where $1 \leq i \leq 3$, $|y_i\rangle = \frac{1}{\sqrt{2}}(|0\rangle + e^{2\pi i(0.x_i \ldots x_3)} |1\rangle)$.

Thus, by applying a Hadamard gate to the state above we obtain $|x_{k+1}\rangle$. Therefore, we can consider the inverse Fourier transform as a series of Hadamard tests.

If φ has an exact n-bit binary representation the success probability at each step is 1. While, in the case that φ cannot be exactly expressed in n-bit binary fraction, the success probability P of the post-measurement state, at step k, is

$$P = \cos^2(\pi\theta) \quad \text{for} \quad |\theta| < \frac{1}{2^{k+1}} \tag{3.37}$$

Detailed analysis obtaining similar probabilities are given in Section 3.2.

Therefore, the success probability increases as we proceed. The following theorem gives us the success probability of the QFT algorithm.

Theorem 1 ([KLM07])**.** *If* $\frac{x}{2^n} \leq \varphi \leq \frac{x+1}{2^n}$*, then the phase estimation algorithm returns one of* x *or* $x+1$ *with probability at least* $\frac{8}{\pi^2}$*.*

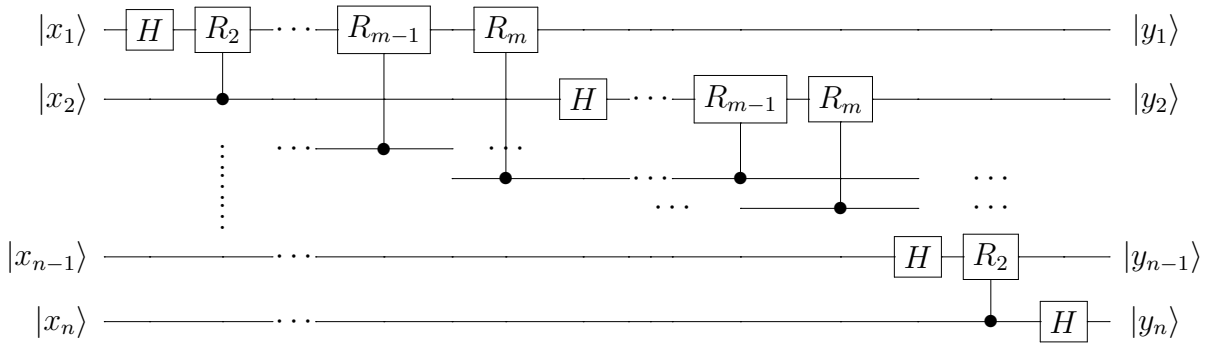


Figure 3.4: Quantum circuit for AQFT.

3.1.3 Approach Based on AQFT

AQFT was first introduced by Barenco, et al [BES96]. It has the advantage in algorithms that involve periodicity estimation. Its structure is similar to regular QFT but differs by eliminating higher precision phase shift operators. The circuit of AQFT is shown in Figure 3.4. At the RHS of the circuit, for $n - m < i \leq n$

$$|y_i\rangle = \frac{1}{\sqrt{2}}(|0\rangle + e^{2\pi i(0.x_i \ldots x_n)} |1\rangle) \tag{3.38}$$

and for $1 < i \leq n - m$,

$$|y_i\rangle = \frac{1}{\sqrt{2}}(|0\rangle + e^{2\pi i(0.x_i \ldots x_{i+m-1})} |1\rangle). \tag{3.39}$$

Let $0.x_1x_2 \ldots x_n$ be the binary representation of eigenphase φ. For estimating each x_p, where $1 \leq p \leq n$, AQFT$_m$ requires at most m phase shift operations. Here m is defined as the degree of the AQFT$_m$.

Therefore, phase shift operations in AQFT_m requires precision up to $e^{2\pi i/2^m}$. The probability P of gaining an accurate output using AQFT_m, when $m \geq \log_2 n + 2$, is at least [BES96]

$$P \geq \frac{8}{\pi^2}(\sin^2(\frac{\pi}{4}\frac{m}{n})). \tag{3.40}$$

The accuracy of AQFT_m approaches the lower bound for the accuracy of the full QFT, which is $\frac{8}{\pi^2}$. A better lower bound is also achieved by Cheung in [Che04]

$$P \geq \frac{4}{\pi^2} - \frac{1}{4n}. \tag{3.41}$$

Moreover, this indicates the logarithmic-depth AQFT provides an alternative approach to replace the regular QFT in many quantum algorithms. The total number of the phase shift operator invocations in AQFT_m is $O(n \log_2 n)$, instead of $O(n^2)$ in the QFT. The phase shift operator precision requirement is only up to $4n$, instead of 2^n.

By using the AQFT instead of the QFT we trade off smaller success probability with smaller degrees of phase shift operators and a shorter circuit.

3.2 New Approach With Arbitrary Degree Phase Shift Operators

In this section we introduce our new approach for QPE. Our approach draws a trade-off between the highest degree of phase shift operators being used and the depth of the circuit.

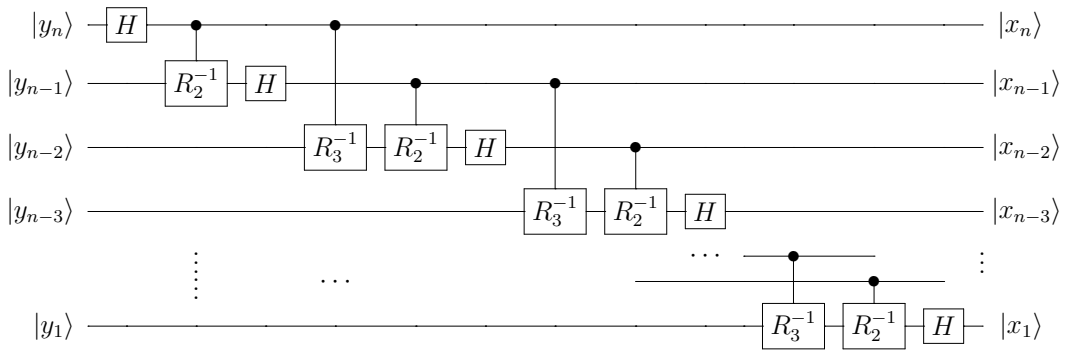


Figure 3.5: QPE with only two controlled phase shift operations.

As a result, when smaller degrees of phase shift operators are used, the depth of the circuit increases and vice versa.

As pointed out in Section 3.1.2, by using information of previous qubits, the full-fledged inverse QFT transforms the phase such that the phase of the corresponding qubit gets closer to one of the states $|+\rangle$ or $|-\rangle$. For our approach, we first consider the case where only the controlled phase shifts operators R_2 and R_3 are used (Eq. 3.35). In this case, we only use the information of the two previous qubits (see Figure 3.5). In such a setting, we show that it is possible to perform the QPE algorithm with arbitrary success probability.

The first stage of our algorithm is similar to the first stage of QPE based on QFT. Assume the phase is $\varphi = 0.x_1x_2x_3\ldots$ with an infinite binary representation. At step k, the phase after the action of the controlled gate U^{2^k} is $2^k\varphi = 0.x_{k+1}x_{k+2}\ldots$ and the corresponding state is

$$|\psi_k\rangle = \frac{1}{\sqrt{2}}(|0\rangle + e^{2\pi i 2^k \varphi}|1\rangle). \tag{3.42}$$

By applying controlled phase shift operators R_2 (controlled by the $(k-1)$th qubit) and R_3 (controlled by the $(k-2)$th qubit) to the state above, we obtain

$$\left|\widetilde{\psi_k}\right\rangle = \frac{1}{\sqrt{2}}(|0\rangle + e^{2\pi i\widetilde{\varphi}}|1\rangle), \tag{3.43}$$

where

$$\widetilde{\varphi} = 0.x_{k+1}00x_{k+4}\ldots. \tag{3.44}$$

It is easy to see that

$$|\widetilde{\varphi} - 0.x_{k+1}| < \frac{1}{8}. \tag{3.45}$$

Hence, we can express

$$\widetilde{\varphi} = 0.x_{k+1} + \theta \tag{3.46}$$

where $|\theta| < \frac{1}{8}$. Therefore, the state $\left|\widetilde{\psi_k}\right\rangle$ can be rewritten as

$$\left|\widetilde{\psi_k}\right\rangle = \frac{1}{\sqrt{2}}(|0\rangle + e^{2\pi i(0.x_{k+1}+\theta)}|1\rangle). \tag{3.47}$$

In order to approximate the phase φ at this stage (kth step), we need to find the value of x_{k+1} by measuring the kth qubit. In this regard, we first apply a Hadamard gate before the measurement to the state $\left|\widetilde{\psi_k}\right\rangle$. The post-measurement state will determine the value of x_{k+1} correctly with high probability. The post measurement probabilities of achieving $|0\rangle$

or $|1\rangle$ in the case where $x_{k+1} = 0$ is

$$\begin{aligned} \Pr(0|k) &= \cos^2(\pi\theta) \\ \Pr(1|k) &= \sin^2(\pi\theta). \end{aligned} \tag{3.48}$$

Therefore,

$$\begin{aligned} \Pr(0|k) &\geq \cos^2(\frac{\pi}{8}) \approx 0.85 \\ \Pr(1|k) &\leq \sin^2(\frac{\pi}{8}) \approx 0.15 \end{aligned} \tag{3.49}$$

In the case where $x_{k+1} = 1$, the success probability is similar.

By iterating this process a sufficient number of times and then letting the majority decide, we can achieve any desired accuracy. The analysis is similar to Section 3.1.1. In this case, all we require is to find the majority. Therefore, by a simple application of the Chernoff's bound

$$\Pr\left(\frac{1}{m}\sum_{i=0}^{m} X_i \leq \frac{1}{2}\right) \leq e^{-2m(p-\frac{1}{2})^2}, \tag{3.50}$$

where in this case $p = \cos^2(\pi/8)$. It is easy to see that if a success probability of $1 - \varepsilon$ is required, then we need at least

$$m = 4\ln(\frac{1}{\varepsilon}) \tag{3.51}$$

many trials for approximating each bit.

By comparing Eq. 3.30 and Eq. 3.51 (Table 3.1), we see that while preserving the success probability, our new algorithm differs by a constant and scales about 12 times better than Kitaev's original approach in terms of the number of Hadamard tests required (Figure 3.6). In physical implementations this is very important, especially in the case where only one copy of the eigenvector $|u\rangle$ is available and all Hadamard tests should be performed during one run of the circuit.

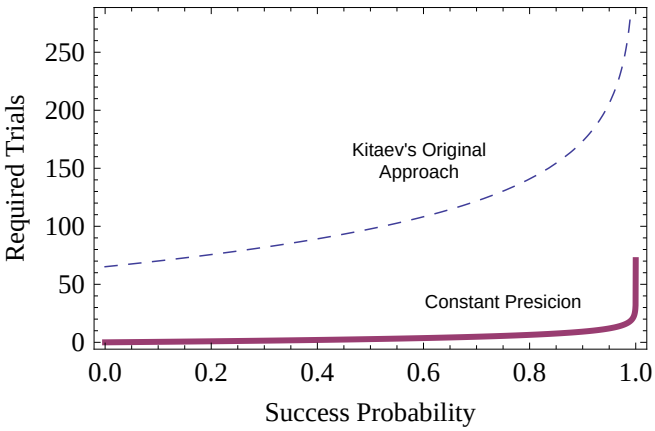


Figure 3.6: Required trials for estimating each bit in Kitaev's original approach and our new approach.

In the algorithm introduced above, only phase shift operators R_2 and R_3 are used. When higher phase shift operators are used in our algorithm, the success probability of each Hadamard test will increase. As a result, fewer trials are required in order to achieve similar success probabilities. As pointed out in Section 3.1.3, the QPE based on AQFT requires phase shift operators of degree at least $2 + \log n$. With this precision of phase shift operators in hand, the success probability at each step would be high enough such that there

is no need to iterate each step. In such scenario, one trial is sufficient to achieve an overall success probability of a constant.

Success Probability	**Kitaev's Original Approach**	**Constant Precision**
0.50000	98	3
0.68269	120	5
0.95450	211	13
0.99730	344	24
0.99993	515	39

Table 3.1: Required trials for estimating each bit by using Chernoff's bound.

Recall the phase estimation problem stated in the introduction. If a constant success probability greater than $\frac{1}{2}$ is required, the depth of the circuit for all the methods mentioned in this chapter (except the QPE based on full fledged QFT, which is $O(n^2)$), would be $O(n \log n)$ (assuming the cost of implementing the controlled-U^{2^k} gates are all the same). This means the depth of the circuits differ only by a constant. However, the disadvantage of Kitaev's original approach to our new approach is the large number of Hadamard tests required for each bit in the approximated fraction.

Therefore, the new method introduced in this chapter provides the flexibility of using any available degree of controlled phase shift operators while preserving the success probability and the length of the circuit up to a constant.

CHAPTER 4
QUANTUM COMPLEXITY OF EVALUATING THE TUTTE POLYNOMIAL

The problem of exactly and even approximately evaluating the Tutte polynomial is classically hard. In this chapter we relate the problem of approximately evaluating the Tutte polynomial of some special types of graphs at some points to quantum computing.

The Tutte polynomial $T(G; x, y)$ of a graph G is a two variable generalization of the chromatic polynomial and was first introduced by William T. Tutte in 1954 [Tut01]. An important feature of the Tutte polynomial is that it captures a lot of information about the graph G. For example, $T(G; 1, 1)$ counts the number of spanning trees of a connected graph G and $T(G; 2, 1)$ counts the number of forests in G. Reference [Wel93] provides a more extensive collection of graph properties that can be simply read off by evaluating the Tutte polynomial at suitable points.

The complexity of Tutte polynomial has been studied by many authors. We give some examples of the results that are relevant to our study. Jaeger, Vertigan and Welsh showed in [JVW90] that evaluating the Tutte polynomial exactly is #P-hard except for the points on the hyperbola $(x-1)(y-1) = 1$ and the four points $(x, y) \in \{(1, 1), (0, -1), (-1, 0), (-1, -1)\}$. Goldberg and Jerrum have recently shown in [GJ08] that for rational numbers (x, y) with $x < -1$ or $y < -1$ and not on the hyperbolas $H_n : (x - 1)(y - 1) = n$ where $n = 0, 1, 2$, there is no fully polynomial randomized approximation scheme (FPRAS) for approximately

computing $T(G; x, y)$ for general graphs G. They have also shown that for some other points there is no FPRAS; for more details see [GJ08].

There are also some results on efficient algorithms for approximately evaluating the Tutte polynomial of some special types of graphs. For example, Alon, Frieze and Welsh [AFW95] obtained FPRAS for dense graphs G for points (x, y) where $x > 1$ and $y > 1$. We refer the interested reader to [GJ08] for a review of such algorithms.

The above discussion shows that the problem of exactly and even approximately evaluating the Tutte polynomial is classically hard. In this chapter we relate the problem of approximately evaluating the Tutte polynomial of some special types of graphs at certain points to quantum computing.

We consider two types of graphs, referred to as triangular and circular. We prove that the problem of providing an additive approximation for the evaluation of the Tutte polynomial of triangular graphs is BQP-complete. Roughly speaking, the complexity class BQP (Bounded error Quantum Polynomial time) is the class of problems that can be solved efficiently on a quantum computer. DQC1 is a quantum complexity class that is contained in BQP. The difference between DQC1 and BQP is that in DQC1 only one qubit can be initialized in the state $|0\rangle$ and all other qubits are in a completely random (maximally mixed) state [DFC05]. This "one clean qubit model" was first introduced by Knill and Laflamme in [KL98]. This presents an alternative proof for results by Aharonov, Arad, Eban and Landau [AAE07]. Our proof establishes that it suffices to consider only triangular graphs to achieve BQP-hardness. We also show that the problem of providing an additive approximation of the

Tutte polynomial for circular graphs is in DQC1 at the point $(e^{2\pi i/5}, e^{-2\pi i/5})$ and in BQP for points $(q^k, 1 + \frac{1-q^{-k}}{(q^{1/2}-q^{-1/2})^2})$, where q is a root of unity. To prove these results, we establish a connection between the problems of approximately evaluating the Tutte polynomial of triangular and circular graphs and that of approximately evaluating the Jones and HOMFLY polynomial of plat and trace closures of braids, respectively. It is known that the latter are related to the quantum complexity classes BQP [AJL06, FKW02, FLW02, KL07a, KL07b, LK06, WY08] and DQC1 [SJ08].

This chapter is organized as follows. In Section 4.1 we define formally triangular and circular graphs. In Section 4.2 we recall the formal definition of additive approximation and state the three main theorems. In Section 4.3 we prove all the results that we use in proofs of the theorems.

This chapter is based on the paper *On the quantum complexity of evaluating the Tutte polynomial* co-authored with Pawel Wocjan [AW10].

4.1 Definition of Triangular and Circular Graphs

Figure 4.1 presents an example of a triangular graph. The formal definition is given below.

Definition 1. *A triangular graph is a graph without loops that is constructed by the following steps:*

1. *Draw an $n \times m$ grid.*

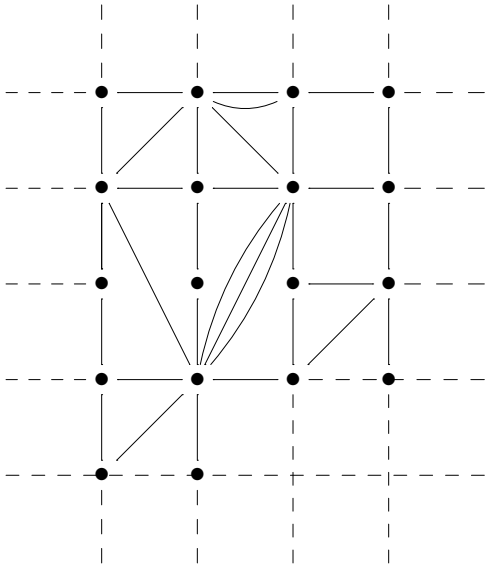


Figure 4.1: Triangular graph

2. *Label the intersection points of the horizontal and vertical lines by* (i, j) *where* $i = 0, \ldots, n-1$ *denotes the row index and* $j = 0, \ldots, m-1$ *denotes the column index. The point* $(0, 0)$ *is the upper-left corner of the grid.*

3. *The vertex set* V *of the triangular graph* G *is an arbitrary subset of the intersections points that satisfies the following condition: if* (i, j) *is a vertex with* $i > 0$ *then* $(i-1, j)$ *is also a vertex.*

4. *The edge set* E *satisfies the following two properties:*

5. *If* (i, j) *is a vertex with* $i > 0$*, then there is exactly one edge between* (i, j) *and the vertex* $(i-1, j)$*.*

6. *Two vertices* (i, j) *and* (i', j') *when* $j \neq j'$*, can be connected only if* $j' = j + 1$*.*

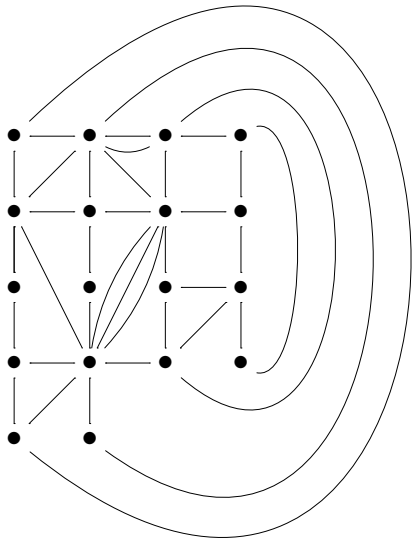


Figure 4.2: Circular graph

Figure 4.2 presents an example of a circular graph. The formal definition is given below.

Definition 2. *A circular graph is a graph that is obtained as follows.*

1. *Start with an arbitrary triangular graph on an $n \times m$ grid.*

2. *For each $j = 0, \ldots, n$, add a new edge between the vertices $\{(0, j)$ and (m_i, j) where $m_i = \max\{a : (a, j)$ is a vertex of the triangluar graph$\}$.*

We refer to n in the above definitions as the width $w(G)$ of the graph G and to m as the length $\ell(G)$.

4.2 Additive Approximation of the Tutte Polynomial of Triangular and Circular Graphs

To state our results we have to recall the definition of additive approximation as formalized in [BFL05]. Let $\mathcal{I}$ denote a set of problem instances, and suppose we are given a function $f : \mathcal{I} \to \mathbb{K}$ ($\mathbb{K} = \mathbb{R}$ or $\mathbb{K} = \mathbb{C}$) that is potentially difficult to evaluate exactly. The main idea is to approximate the function f with respect to some positive normalization function $g : \mathcal{I} \to \mathbb{R}$. An additive approximation for the normalized function f/g associates a random variable $Z(I)$ to any problem instance $I \in \mathcal{I}$ and $\epsilon > 0$ satisfying

$$\Pr\left(\left|\frac{f(I)}{g(I)} - Z(I)\right| \leq \epsilon\right) \geq 3/4\,. \tag{4.1}$$

The process is required to run in time that is polynomial in $1/\epsilon$ and the input size of $\mathcal{I}$.

In case where $\mathcal{I}$ is the set of all triangular graphs, we consider

$$f(G) = |T_G(q, 1/q)| \tag{4.2}$$

$$g(G) = [2]_\ell^{w(G)}\,, \tag{4.3}$$

where $[m]_\ell$ is the quantum integer $[m]_\ell = \frac{q^{\frac{m}{2}} - q^{\frac{-m}{2}}}{q^{\frac{1}{2}} - q^{\frac{-1}{2}}}$ when $q = e^{2\pi i/\ell}$.

In the case where $\mathcal{I}$ is the set of circular graphs, we consider

$$f(G) = \text{Re } T_G(q, 1/q) \tag{4.4}$$

or

$$f(G) = \text{Im } T_G(q, 1/q) \tag{4.5}$$

and

$$g(G) = [2]_\ell^{w(G)}. \tag{4.6}$$

We defined two types of graphs in the previous chapter, referred to as triangular and circular. We prove that the problem of providing an additive approximation for the evaluation of the Tutte polynomial of triangular graphs is BQP-complete.

Theorem 2. *The problem of additively approximating $T(G; q, 1/q)$ for triangular graphs G is BQP-complete for $q = e^{2\pi i/\ell}$.*

Our proof establishes that it suffices to consider only triangular graphs to achieve BQP-hardness. We also show that the problem of providing an additive approximation for circular graphs is in DQC1 at $(e^{2\pi i/5}, e^{-2\pi i/5})$ and in BQP for $(q^k, 1 + \frac{1-q^{-k}}{(q^{1/2}-q^{-1/2})^2})$.

Theorem 3. *The problem of additively approximating $T(G; q^k, 1 + \frac{1-q^{-k}}{(q^{1/2}-q^{-1/2})^2})$ for circular graphs G is in BQP for $q = e^{2\pi i/\ell}$.*

Theorem 4. *The problem of additively approximating $T(G; e^{2\pi i/5}, e^{-2\pi i/5})$ for circular graphs G is DQC1-complete.*

To prove these results, we establish a connection between the problems of approximately evaluating the Tutte polynomial of triangular and circular graphs and that of approximately evaluating the Jones and HOMFLY polynomial of plat and trace closures of braids, respectively. It is known that the latter are related to the quantum complexity classes BQP [AJL06, FKW02, FLW02, KL07a, KL07b, LK06, WY08] and DQC1 [SJ08].

More precisely, we establish this connection as follows:

- Given an arbitrary braid b we show how to efficiently construct a braid $\tilde{b}$ such that its plat closure $\tilde{b}^{\mathrm{p}lat}$ is an alternating link and

$$J(b^{\mathrm{p}lat}; q) = J(\tilde{b}^{\mathrm{p}lat}; q)\,, \tag{4.7}$$

  where $J(L; q)$ denotes the evaluation of the Jones polynomial of the link L at q. This construction relies upon the fact that the images of braid group generators under all irreducible Jones-Wenzl representations of the braid group B_n have finite order.

- We construct a triangular graph G such that the alternating link $D(G) = \tilde{b}^{\mathrm{p}lat}$ corresponds to the medial graph $M(G)$ of G. Using the connection between the Tutte and Jones polynomials [Thi87], we obtain

$$T(G; q, 1/q) = \alpha(G) J(\tilde{b}^{\mathrm{p}lat}; q)\,, \tag{4.8}$$

  where $\alpha(G)$ is complex number of modulus one that is easily computed. These arguments establish that the ability to approximately evaluate the Tutte polynomial of

triangular graphs implies the ability to approximately evaluate the Jones polynomial of plat closure of braids. Since the latter problem is already known to be BQP-hard, we see that the approximate evaluation of the Tutte polynomial is also BQP-hard.

- The other direction, i.e., the proof that the problem of approximately evaluating the Tutte polynomials of triangular graphs is in BQP, is obtained using the above arguments.

- In the case of DQC1, we have to consider the trace closure instead of the plat closure. The structure of the proofs remains the same. We make use of the result by Shor and Jordan [SJ08] that evaluating the Jones polynomial of the trace closure of braids at the fifth root of unity is DQC1-complete.

4.3 Connection Between Tutte and Jones Polynomials

We associate a link diagram $D(G)$ to a planar graph G (Figure 4.3) such that the graph G can be recovered by the following way: Color the link diagram as a white/black checkerboard with the outer region colored as white. Assign a vertex to each black region and connect the vertices by an edge for any crossing the corresponding regions share. (for more details see Ref. [Kau01])

It is easy to construct a graph G having $D(G)$, but its not always easy to find a link diagram such that the link becomes $D(G)$ associated to the planar graph G. Here we give an

algorithm for constructing braids such that the plat or trace closure of it becomes $D(G)$ of our special type of graph G. In these algorithms we use the following subroutines: (We labeled each vertex with (i, j) where i and j are the corresponding row and column, respectively.)

Vertex(i,j):

1. If the vertex $(i-1, j)$ is marked exit.
2. If $i > 1$ call **Vertex($i-2$,j)**.
3. If $i > 0$, For each vertex (k, ℓ) connected to the vertex $(i-1, j)$ call **Edge**$((i-1, j), (k, \ell))$.
4. Mark vertex $(i-1, j)$ if it is not the last vertex in column j.
5. Add σ_{2j+1} to the braid word b.

Edge$((i, j), (k, \ell))$**:**

1. If the edge $\big((i, j), (k, \ell)\big)$ is marked OR $\ell = j$ exit.
2. If $k > 0$ call **Vertex(k,ℓ)**.
3. Mark the edge $\big((i, j), (k, \ell)\big)$.
4. Add $\sigma_{j+\ell+1}^{-\alpha}$ to the braid word b, where α is the weight of the edge $\big((i, j), (k, \ell)\big)$.

In the following algorithms we do not consider vertical edges of input graphs and for multiple edges between two vertices we replace them by one edge, weighted by their number.

The algorithms for triangular and circular graphs are as follows:

Algorithm for finding a braid such that its plat closure becomes $D(G)$ of the input graph G

1. input: a triangular graph
2. Assume an empty braid word b.
3. For $i = 0$ to $n - 1$ do
   - If $i > 0$ and vertex (i, j) is NOT the last vertex in column j and unmarked do
     - Mark the vertex (i, j).
     - Add σ_{2j+1} to the braid word b.
   - For $k = i$ to $n - 1$ do
     - Call **Edge** for all edges between row i and k.
4. Output: braid word b.

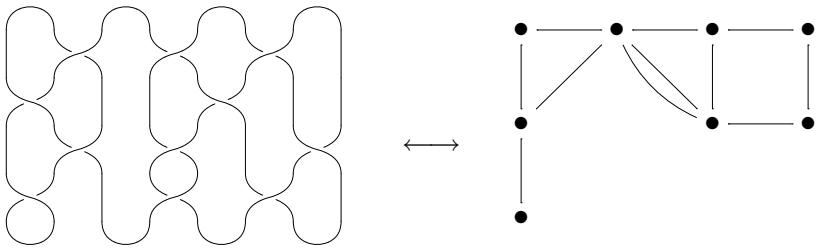


Figure 4.3: The plat closure of the braid on the left which is $D(G)$ corresponding to the triangular graph G on the right.

Algorithm for finding a braid such that its trace closure becomes $D(G)$ corresponding to the input graph G

1. Input: a circular graph
2. Assume an empty braid word b.
3. For $i = 0$ to $n - 1$ do
    - For each unmarked vertex (i, j) that is NOT the last vertex in column j
        - Mark the vertex (i, j)
        - Add σ_{2j+1} to the braid word b.
    - For $k = i$ to $n - 1$ do
        - Call **Edge** for all edges between row i and k.
4. Output: braid word b.

Observe that both algorithms always output braid words such that their respective closures are alternating links. This is because they have the special form in Lemma 1 below.

Lemma 1. *If any braid word b in the n-strand Braid group B_n with generators $\{\sigma_1, \ldots, \sigma_{n-1}\}$ does not contain σ_i^{-1} for odd i's and does not contain σ_i for even i's, then the trace and plat closure of b is an alternating link.*

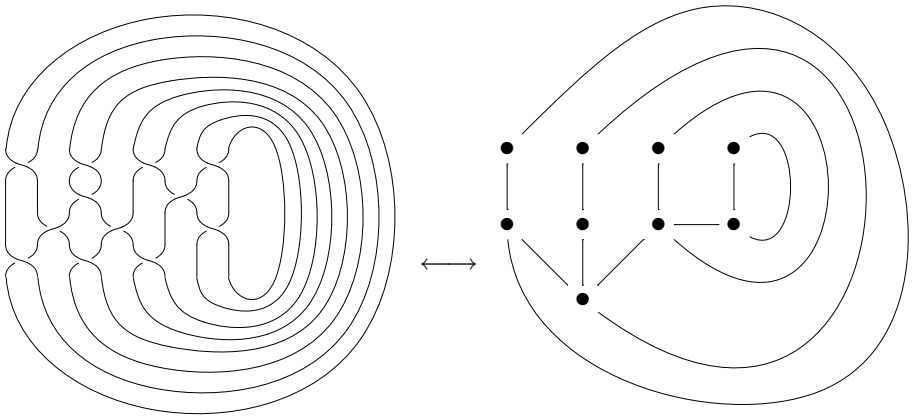


Figure 4.4: The trace closure of the braid on the left which is $D(G)$ corresponding to the circular graph G on the right.

Since the links are alternating, we can use the correspondence between the Jones (and HOMFLY) and the Tutte polynomials given by the following theorems.

Theorem 5 ([Thi87])**.** *Let $D(G)$ be a connected alternating oriented link diagram with a A-regions, b B-regions and writhe ω. Then the Jones polynomial of $D(G)$ is given by the Tutte polynomial of G,*

$$J(D(G), t) = (-1)^{\omega} t^{(b-a+3\omega)/4} T_G(-t, -1/t). \tag{4.9}$$

Theorem 6 ([Jae88])**.** *Let $G = (V, E)$ be a connected plane graph. Then for all nonzero numbers t and x, the HOMFLY polynomial of the corresponding alternating link $D(G)$ is:*

$$H(D(G), t, x) = (-tx)^{-|V(G)|+1}(-x/t)^{|E(G)|} T(G; t^2, 1 + ((1 - t^{-2})/x^2)). \tag{4.10}$$

4.3.1 Unitary Representation of The Braid Groups

Assume the n-strand Braid group [Art91] with generators $\{\sigma_i\}$. We can represent the braid group inside a Hecke Algebra $H_n(q)$ [Jon86, Wen88] by mapping each generator σ_i of B_n to the generator g_i of $H_n(q)$. The Hecke Algebra $H_n(q)$ for any $q \in \mathbb{C}^\times$ of type A is a free complex algebra generated by 1 and $\{g_1, \ldots, g_{n-1}\}$ with relations:

$$g_i^2 = g_i(q-1) + q \tag{4.11}$$

$$g_i g_j g_i = g_j g_i g_j \qquad |i-j| = 1 \tag{4.12}$$

$$g_i g_j = g_j g_i \qquad |i-j| > 1. \tag{4.13}$$

By defining $e_i = (q - g_i)/(1+q)$ we can rewrite the above relations of the Hecke algebra $H_n(q)$:

$$e_i^2 = e_i \tag{4.14}$$

$$e_i e_j e_i - \tau e_i = e_j e_i e_j - \tau e_j, \quad |i-j| = 1 \tag{4.15}$$

$$e_i e_j = e_j e_i, \qquad |i-j| > 1, \tag{4.16}$$

where for $q = e^{2\pi i/\ell}$ and $\tau = [2]_\ell^{-2}$.

The Jones-Wenzl irreducible representation π_λ of the Hecke algebra $H_n(q)$ are enumerated by admissible Young diagrams λ. The image of g_i under the irreducible representation π_λ [Jon86, Wen88] is as follows:

$$\pi_\lambda(g_i) = q1_{V_\lambda} - (1+q)\pi_\lambda(e_i) \tag{4.17}$$

where V_λ is the vector space on which π_λ acts.

For our purposes, it is important that $\pi_\lambda(e_i)$ is an orthogonal projector for all $1 \leq i \leq n$. For more details we refer the reader to [Jon86, Wen88, WY08].

For any integer $k \geq 2$ we can define the one-variable HOMFLY polynomial for a link L as

$$H_L^{(k)}(q) = H_L(q^{k/2}, q^{1/2} - q^{-1/2}). \tag{4.18}$$

By [Jon86, Wen88] a representation-theoretic formula for the HOMFLY polynomial of the trace closure of a braid is

$$H_{b^{tr}}^{(k)}(q) = [k]_\ell^{n-1} q^{-\frac{k+1}{2}e(b)} \sum_\lambda s_\lambda \mathrm{Tr}\pi_\lambda(b). \tag{4.19}$$

where s_λ are the Markov weights and for the special case $k = 2$, we will have the Jones polynomial of the trace closure of a braid.

There is also a relation with the Jones polynomial of the plat closure of a braid,

$$|J(b^{plat}, q)| = [2]_\ell^{n-1} |\mathrm{Tr}(\pi_\mu(b)P)| \tag{4.20}$$

where μ is a particular Young diagram and P is some projector acting on V_μ. (The exact formula is given in Eq. (5.39) in [WY08]).

Lemma 2. *The image of the generator g_i of the Hecke Algebra $H_n(q)$ for $q = e^{2\pi i/\ell}$ and integers $2 \leq k < \ell < \infty$ under the Jones-Wenzl representation π_λ is of finite order $lcm(2, \ell)$.*

Proof. The irreducible Jones-Wenzl representation π_λ is an orthogonal projector therefor, let $\mathrm{P} = \pi_\lambda(e_i)$ where $\mathrm{P}^2 = \mathrm{P}$ and $1_{V_n} = \mathrm{P} + \mathrm{P}^\perp$.

$$\begin{aligned} \pi_\lambda(g_i) &= q(\mathrm{P} + \mathrm{P}^\perp) - (1+q)\mathrm{P} && (4.21) \\ &= q\mathrm{P}^\perp - \mathrm{P} && (4.22) \end{aligned}$$

Since P and $P^\perp$ are orthogonal ($PP^\perp$=0) we have,

$$(\pi_\lambda(g_i))^n = q^n\mathrm{P}^\perp + (-1)^n\mathrm{P}. \quad (4.23)$$

Hence wherever n is even and a multiple of ℓ we have,

$$\begin{aligned} (\pi_\lambda(g_i))^n &= \mathrm{P}^\perp + \mathrm{P} && (4.24) \\ &= 1_{V_n}. && (4.25) \end{aligned}$$

Hence the order of $\pi_\lambda(g_i)$ is $lcm(2, \ell)$. □

Theorem 7. *For any $b \in B_n$ there is $\tilde{b} \in B_n$ such that the plat and trace closure of $\tilde{b}$ is an alternating link and we have*

$$H^{(k)}_{b^{tr}}(e^{2\pi i/\ell}) = H^{(k)}_{\tilde{b}^{tr}}(e^{2\pi i/\ell}) \tag{4.26}$$

$$J(b^{tr}, e^{2\pi i/\ell}) = J(\tilde{b}^{tr}, e^{2\pi i/\ell}). \tag{4.27}$$

Proof. By Lemma 2 we know that $\pi^{(2,\ell)}_{\lambda}(\sigma_i)$ has finite order $p = lcm(2, \ell)$. Assume $b = \sigma_{i_1}^{e_1}\sigma_{i_2}^{e_2}\dots\sigma_{i_m}^{e_m}$ with $0 < |e_j| < p$. Apply the following to the braid word b:

- For $\sigma_{i_j}^{e_j}$ if i_j is odd and $e_j < 0$ then replace $\sigma_{i_j}^{e_j}$ by $\sigma_{i_j}^{p+e_j}$.
- For $\sigma_{i_j}^{e_j}$ if i_j is even and $e_j > 0$ then replace $\sigma_{i_j}^{e_j}$ by $\sigma_{i_j}^{-p+e_j}$.

By Eq. (4.19), Lemma 2 and Lemma 1 proof is completed. □

CHAPTER 5
QUANTUM PROCESSING OF FINITE BLACK-BOX MODULES AND ALGEBRAS

Suppose we are given black-box access to a finite module M or algebra over a finite ring R, and a list of generators for M and R. Here we present an oracle for finding structure constants for black-box mathematical structures consisting an Abelian group. Finding the structure constants is classically hard and needs too many queries to the black box. Moreover we find quantum algorithms for several problems regarding finite black box modules, which need not be commutative. All of the algorithms run in time scaling polylogarithmically in the size of the module.

This chapter is organized as follows, In Section 5.1 we provide some necessary algebraic definitions used in this chapter.

In Section 5.2 we introduce quantum algorithms for finding linear basis and structure constants of mathematical structures consisting an Abelian group.

In Section 5.3 we discuss how using our quantum algorithms for finding the structure constants together with additional polynomial time quantum and classical processing we can solve several module and ring theoretic problems: decomposing ring elements in terms of generators, testing equality and finding intersection of two ideals, solving systems of linear equations over R, finding the multiplicative and additive identities in R, computing the multiplication tensor for R, computing quotient ideals, finding the annihilator of an ideal, finding the order of an ideal, testing surjectivity and injectivity of ring homomorphisms,

testing whether a given ring element is a unit, and testing whether a given two-sided ideal is prime.

In Section 5.4 we show how having the structure constant we can decompose associative algebras into direct sum for simple algebras. Also, how to solve the module isomorphism problem in the special case of modules over finite rings. Moreover, we discuss some problems regarding algebras that can be solved efficiently on a quantum computer.

This chapter is based on the paper *Efficient quantum processing of ideals in finite rings* co-authored with Pawel Wocjan, Stephen Jordan and Joseph Brennan [WJA].

5.1 Definitions

Let M be a finite left R-Module over the ring R with identity, which need not be commutative. A left R-module over the ring R consists of an Abelian group $(M, +)$ and an operation $\bullet : R \times M \mapsto M$. For convenience we write $r \bullet m = rm$ for $r \in R$ and $m \in M$. For all r,s in R, x,y in M, the multiplication operation satisfies the following relations

1. $r(x + y) = rx + ry$
2. $(r + s)x = rx + sx$
3. $(rs)x = r(sx)$
4. $1_R x = x$ if R has multiplicative identity 1_R.

Let $\tilde{M} = \{m_1, \ldots, m_n\}$ be a subset of M such that each element of M can be obtained by a linear combination of elements of $\tilde{M}$ with multiplicative scalars from R. We say that $\tilde{M}$ is a generating set for M. Note that there always exists a generating set $\tilde{M}$ such that $|\tilde{M}| = O(\log |M|)$. We assume here that our quantum computer has only blackbox access to M. That is, the module and ring elements are assigned arbitrary bit strings by injective maps η and γ, respectively. We are given a list of bit strings $\{\eta(m_1), \ldots, \eta(m_n)\}$ corresponding to a generating set, and access to blackboxes implementing $f_+(\eta(a), \eta(b)) = \eta(a + b)$ and $f_\bullet(\gamma(a), \eta(b)) = \eta(a \cdot b)$.

The most important structure theorem in group theory is the structure theorem for finitely generated Abelian groups.

Theorem 8 (Structure theorem for Abelian groups)**.** *Let G be a finitely generated Abelian group. Then there is a unique expression of the form:*

$$G \stackrel{\text{sim}}{=} \mathbb{Z}^r \oplus \mathbb{Z}/n_1\mathbb{Z} \oplus \mathbb{Z}/n_2\mathbb{Z} \oplus \ldots \oplus \mathbb{Z}/n_s\mathbb{Z}$$

for some integers r, n_i satisfying:

$$r \geq 0; \quad \forall i, n_i \geq 2; \quad n_{i+1} \mid n_i \text{ for } 1 \leq i \leq s-1.$$

Any module M has an Abelian group structure $(M, +)$ under addition. Any generating set $\{a_1, \ldots, a_\ell\}$ for an Abelian group A yields a homomorphism from $\mathbb{Z}_{s_1} \times \ldots \times \mathbb{Z}_{s_\ell}$ to A where $s_1, \ldots, s_l$ are the orders of $a_1, \ldots, a_\ell$. In additive notation, this homomorphism takes

the integers $z_1, \ldots, z_\ell$ to $\sum_{j=1}^{\ell} z_j a_j$. The structure theorem for finite Abelian groups states that there exists a generating set for A such that this homomorphism is an isomorphism. We call this a *generating set of the invariant factors*, or i. f. generating set for short.

Cheung and Mosca have shown that its possible to determine the structure of a finite Abelian black box group efficiently on a Quantum computer[CM01].

A module consists an Abelian group but it also has an extra multiplication by ring elements. In Section 5.2 we provide an efficient quantum algorithm that extracts all the information needed to multiply random elements of R and M and write it as a linear combination of generators of M. Let $\{h_1, \ldots, h_\ell\}$ be an generating set for M. The multiplication of R and M can be fully specified by the structure constants in the tensor T_{ij}^k defined by

$$r_i h_j = \sum_{k=1}^{\ell} T_{ij}^k h_k. \tag{5.1}$$

Note that a ring R is also an R-module. Therefore, the i. f. generators for R, their orders, and the structure constants are called a *basis representation* for R. The previous work of Arvind *et al.* shows how to efficiently quantum compute a basis representation in the special case of rings. [ADM06].

The case of algebras are very similar to modules. An algebra A over a finite field F is a vector space over F, together with a F-bilinear operation(multiplication). Therefore, algebras also consist of an Abelian group with an multiplication operation that can be

specified by structure constants. Be believe our algorithm would work for any mathematical structure consisting an Abelian group and other operations.

Because of the similarities between modules and algebras, for convenience we only refer to modules and leave the algebra case to where its needed.

5.2 Efficient Quantum Algorithm for Finding Structure Constants

In this section we provide a quantum algorithm for finding structure constants of mathematical structures consisting an Abelian group in particular modules. We first provide an efficient algorithm in Oracle I to calculate an i. f. generating set for the module M. This task can also be done by a probabilistic algorithm[ADM06]. This is a necessary step for our quantum algorithm. The second step would be a quantum algorithm for Oracle II that decomposes given elements into a linear combination of the generators. Oracle II will provide us enough tools to compute the structure constants. Our algorithm provide an exponential speed up with regard to classical algorithms.

We compute all l^3 entries of T_{ij}^k by taking each pair r_i, h_j, using the multiplication oracle to find the bit string encoding their product, and then applying the Oracle in step 2. The best existing classical algorithm for finding structure constants requires order $|M|$ queries[ZMR08]. Many problems in the field of rings and modules can be solved by simple linear algebra together with these oracles.

5.2.1 Oracle I: Finding I. F. Generators

Our method for finding an i. f. generating set for $(M, +)$ proceeds in two steps. First we find a generating set for $(M, +)$. Although the elements of $\tilde{M}$ generate M, they do not generate M as an Abelian group, that is, by addition only with no left-multiplication by R elements. After finding a generating set for $(M, +)$ we then convert it to an i. f. generating set for $(M, +)$ using the quantum algorithms of [CM01, Wat01].

To find a generating set for $(M, +)$, let $\tilde{B}_1 = \tilde{M}$ and apply the following iteration. Let B_k be the Abelian group additively generated by $\tilde{B}_k$. At the k^{th} step we search for an element $i \in M$ not contained in B_k. If we find one, we let $\tilde{B}_{k+1} = \tilde{B}_k \cup \{i\}$. For some sufficiently large k, $B_k = M$, at which point the search for i fails and the process terminates. We now show in detail how this works and that we need at most $\log_2 |M|$ iterations.

Suppose we know $\tilde{B}_k$. To find an element of M not contained in B_k, we choose any generator $r \in \tilde{R}$ of R. Let $rB_k = \{r \cdot x | x \in B_k\}$. We create the superpositions

$$|B_k\rangle = \frac{1}{\sqrt{|B_k|}} \sum_{x \in B_k} |x\rangle \tag{5.2}$$

and

$$|rB_k\rangle = \frac{1}{\sqrt{|B_k|}} \sum_{x \in B_k} |rx\rangle. \tag{5.3}$$

Because B_k and rB_k are Abelian groups whose generators we know, these states can be created efficiently to polynomial precision using the results of [Wat01, CM01].

To determine the intersection of B_k and rB_k we use the swap test to estimate the inner product $\langle B_k|rB_k\rangle$. Polynomially many applications of the swap test yield $\langle B_k|rB_k\rangle$ to 1/poly precision. $B_k \cap rB_k$ is a subgroup of B_k. Thus by Lagrange's theorem, either $\frac{|B_k \cap rB_k|}{|B_k|} = 1$ or $\frac{|B_k \cap rB_k|}{|B_k|} \leq \frac{1}{2}$. These two cases can be distinguished with high reliability by swap tests, because

$$\langle B_k|rB_k\rangle = \frac{|B_k \cap rB_k|}{|B_k|}. \tag{5.4}$$

If we find that $\frac{|B_k \cap rB_k|}{|B_k|} \leq \frac{1}{2}$ then we choose an element $i \in rB_k$ uniformly at random. We can do this using the techniques of [Wat01, CM01] to find an i. f. generating set for B_k and then sampling uniformly from the product of cyclic groups to which B_k is isomorphic. Thus, along with i we get an expression for i as r times some linear combination of the elements of $\tilde{B}_k$. i is definitely contained in M, and with probability at least 1/2, i is not contained in B_k. If $i \in B_k$ then $\langle B_k|i + B_k\rangle = 1$, otherwise $\langle B_k|i + B_k\rangle = 0$. Thus, to determine whether $i \in B_k$ we create the states $|B_k\rangle$ and $|i + B_k\rangle$ and use the swap test. If $i \in B_k$ we choose a different random element of rB_k and try again. With probability $1 - \epsilon$, this process terminates in $O(\log(1/\epsilon))$ time. Once it does, we let $\tilde{B}_{k+1} = \tilde{B}_k \cup \{i\}$.

If we instead find that $\frac{|B_k \cap rB_k|}{|B_k|} = 1$, we choose a different $r \in \tilde{R}$ and swap test again. We keep repeating this process until we find some $r \in \tilde{R}$ such that $\frac{|B_k \cap rB_k|}{|B_k|} \neq 1$ or we exhaust $\tilde{R}$. If $\frac{|B_k \cap rB_k|}{|B_k|} = 1$ for all $r \in \tilde{R}$ we are done, because $B_k = M$. We can prove this with the following lemma.

Lemma 1. *Let M be a left module generated by $\{i_1, \ldots, i_m\}$ over a finite ring R. Let $\tilde{B}_k$ be a subset of M containing $\{i_1, \ldots, i_m\}$. The set of elements B_k additively generated by $\tilde{B}_k$ is equal to M if and only if $rB_k \subseteq B_k$ $\forall r \in \tilde{R}$.*

Proof. If $rB_k \subseteq B_k$ for all $r \in \tilde{R}$ then, because $\tilde{R}$ is a generating set for R, $rB_k \subseteq B_k$ for all $r \in R$. Thus, B_k is a left module in R. By construction, B_k contains $i_1, \ldots, i_m$. By the definition of generators, M is the smallest left module over R containing $i_1, \ldots, i_m$. B_k is also contained in M. Thus $B_k = M$. The converse follows immediately from the fact that M is a module. □

Remark: *This lemma can be easily extended to any finite mathematical structure consisting an Abelian group with an multiplication operation.*

In the above procedure, the time needed to obtain each additive generator is poly$(\log |R|)$. Furthermore, every time we add another generator, we increase the size of the generated group by at least a factor of two. Thus, we need to perform the above iteration at most $\log_2 |M|$ times. We can also in polynomial time obtain expressions for the elements of this set in terms of the original generators for M by recursively composing the expressions we obtained at each step for i in terms of the preceding generators B_k.

Once we have a set B_k of elements that generate M as an Abelian group, we can efficiently find an i. f. generating set for $(M, +)$, as well as expressions for the i. f. generators as linear combinations of B_k using the techniques of [CM01, Wat01]. These techniques also efficiently yield the additive orders of the i. f. generators.

5.2.2 Oracle II: Decomposing Into Linear Combination of Generators

After finding an i. f. generating set for $(M, +)$, one would like to have a procedure to take a given element $i \in M$ and decompose it as a linear combination of these generators. Note that i is given as an arbitrary bit string from the encoding η, so initially we know nothing about i. We can efficiently perform this decomposition as described below.

Let $G = \mathbb{Z}_{s_1} \times \mathbb{Z}_{s_2} \times \ldots \times \mathbb{Z}_{s_\ell} \times \mathbb{Z}_s$, where $s_1, \ldots, s_\ell$ are the orders of the i. f. generators $h_1, \ldots, h_\ell$ and s is the order of i. Let

$$f(n_1, n_2, \ldots, n_\ell, m) = \eta \left(\sum_{j=1}^{\ell} n_j h_j + mi \right). \tag{5.5}$$

This function hides the cyclic subgroup of G generated by

$$(n_1(i), n_2(i), \ldots, n_\ell(i), -1), \tag{5.6}$$

where $n_1(i), \ldots, n_\ell(i)$ is the decomposition of i in terms of the i. f. generators:

$$i = \sum_{j=1}^{\ell} n_j(i) h_j. \tag{5.7}$$

Using the polynomial time quantum algorithm for the Abelian hidden subgroup problem [NC00], we thus recover this decomposition.

From the algorithm of Oracle I we also obtain expressions for i. f. generators in terms of the original generators of M. Thus one can efficiently convert the expression for r as a linear combination of generators of M into an expression for r in terms of the original generators for M.

Note that Oracle II is also a membership test, if the given element is not in the module the hidden subgroup will be empty.

5.3 Algorithms for Module-Theoretic Problems

In this section we provide a list of problems that can be solved using Oracle I and Oracle II. As mentioned through out the chapter M is an R-Module over the ring R. Generators of R and M are provided and we have black-box access to M. Having the structure constants of a module M the solution to many problems will be reduced to a system of linear Diophantine equations. Other problems can be solved using well known quantum techniques and a combination of Oracle I and II or similar techniques used in their algorithms. The problems that can be solved are as follows:

1. **Membership test**

   Here we give an alternative way to test whether a given element is in the module. After constructing $|M\rangle$ and being given a element m, we can use the addition black-box to construct the co-set state $|m + M\rangle$. If $m \in M$ then the inner product of these states

is one, and otherwise it is zero. Thus, the swap test on $|M\rangle$ and $|m+M\rangle$ tells us whether $m \in M$.

2. **Testing equality of modules**

Assume we are given two separate generating set for modules $M_1 = \langle a_1, \ldots, a_n \rangle$ and $M_2 = \langle b_1, \ldots, b_m \rangle$, we want to see if they are equal. Using Oracle I and II we compute structure constants of M_1 and M_2. Then using oracle II we test if $b_i \in M_1$ for $1 \leq i \leq m$ and $a_i \in M_2$ for $1 \leq j \leq n$. If all are true $M_1 = M_2$.

3. **Inverse of a unit**

If r is a unit, then we can find its inverse using the quantum order finding algorithm[Sho97]. If $r^c = \mathbb{1}$ then $r^{-1} = r^{c-1}$.

4. **Maximal and prime ideal test**

We now show how to efficiently determine whether a given two-sided ideal I is prime. Recall that an ideal I is prime if $ab \in I$ implies that $a \in I$ or $b \in I$ for all $a, b \in R$, which is equivalent to the fact that the quotient ring $S = R/I$ does not have any zero-divisors. This already implies that S is a division ring (i.e., each non-zero element has a multiplicative inverse) since S is finite. Wedderburn's theorem shows that all finite division rings are finite fields [LN97]. R/I a field implies I is maximal, thus I is prime implies I is maximal. The converse is also true.

5. **Field test**

Let S^* denote the group of units of the quotient ring S. We choose an element r

uniformly at random in R. With probability at least $1/2$ we have $r \notin I$. Once we obtain such r we determine the size of the (additively generated) cyclic subgroup $\langle \bar{r} \rangle$ of S, where $\bar{r}$ denotes the image of r in S under the canonical projection. This can be done by applying Shor's period finding algorithm to the state $(1/\sqrt{q}) \sum_{x=0}^{q} |x\rangle|xr+I\rangle)$ where q is a power of 2 with $|S|^2 < q \leq 2|S|^2$. This state can be prepared efficiently.

If S is a field, then with probability at least $\varphi(|S|-1)/|S| \geq \Omega(1/\log|S|)$ we have $\langle \bar{r} \rangle = S^*$ where φ denotes Euler's totient function. This follows from the fact that the group of units $\mathbb{F}_d^*$ of an arbitrary finite field $\mathbb{F}_d$ with d element is cyclic of order $d-1$ and $\varphi(m)/m = \Omega(1/\log m)$ for integers m [HW08]. If S is not a field, then S^* cannot have order $|S|-1$ (otherwise every non-zero element would have a multiplicative inverse, implying that S is a field). If we find that S is a field then we know I is prime, otherwise I is not prime. The above procedure for determining whether the quotient ring S is a field can be applied to any finite black-box ring, offering a simpler alternative to the algorithm in [ADM06].

6. **Problems that can be reduced to systems of linear Diophantine equations**

Many problems regarding rings and modules can be reduced to a system of linear Diophantine equations. Moreover, these problems can be solved using known techniques of Quantum computing as Abelian hidden subgroup problem, SWAP test and order finding and results of [CM01, Wat01].

One of these problems is finding a generating set for the intersection of two given modules. We first show how this problem can be reduced. The other problems can also be reduced to a system of linear equations in a similar fashion:

- **Linear equations**

  Suppose we wish to solve a linear equation $ax = b$ over R. To do this we find an i. f. generating set $\{h_1, \ldots, h_\ell\}$ for R, and decompose a and b in terms of these generators

  $$a = \sum_{i=1}^{\ell} a_i h_i \qquad b = \sum_{i=1}^{\ell} b_i h_i. \tag{5.8}$$

  Let

  $$A_{ij} = \sum_k a_k M_{kj}^i \tag{5.9}$$

  where M_{kj}^i the structure constants. Parametrize x as $x = \sum_{i=1}^{\ell} x_i h_i$ for integers $x_1, \ldots, x_\ell$. Then, in an i. f. generating set, $ax = b$ if and only if

  $$\sum_{j=1}^{\ell} A_{ij} x_j \equiv b_i \mod s_i, \tag{5.10}$$

  for each $i = 1, 2, \ldots, \ell$. (Here s_i is the additive order of h_i.) We can introduce additional integer unknowns $k_1, \ldots, k_\ell$ and rewrite this as a system of linear Diophantine equations:

  $$\sum_{j=1}^{\ell} A_{ij} x_j + k_i s_i = b_i, \quad i = 1, 2, \ldots, \ell. \tag{5.11}$$

A solution to a system of m Diophantine equations in n variables can be found in poly(n, m) time using the classical algorithms of [CC82]. Thus we can classically find an integer solution to Eq. 5.11, which has ℓ equations and 2ℓ unknowns, in poly(ℓ) time. Equation 5.11 is undetermined because the original system of equations 5.10 is modular.

- **Intersection of two modules:**

  Suppose we are given generating sets for two submodules $I = \langle a_1, \ldots, a_s \rangle$ and $J = \langle b_1, \ldots, b_t \rangle$ of module M. We wish to find a basis for $I \cap J$.

  The goal is to reduce this problem to a system of linear Diophantine equations. Using Oracle I and II we can find the structure constants of $M = \langle m_1, \ldots, m_\ell \rangle$. Using Oracle II we write generators of I and J as a linear combination of generators of M. Hence,

$$a_\alpha = \sum_{j=1}^{\ell} I_{\alpha j} m_J, \tag{5.12}$$

$$b_\beta = \sum_{j=1}^{\ell} J_{\beta j} m_J \tag{5.13}$$

  where $1 \leq \alpha \leq s$ and $1 \leq \beta \leq t$. Assume the following equation,

$$v_1 a_1 + \cdots + v_s a_s = u_1 b_1 + \cdots u_t b_t, \tag{5.14}$$

  where v_i and u_i's are integers.

Now using the relations Eq. 5.12 the equation above becomes a linear system of modular equations,

$$v_1 I_{1i} + \cdots + v_s I_{si} = u_1 J_{1i} + \cdots + u_t u_{ti} \pmod{|m_i|} \tag{5.15}$$

where $1 \leq i \leq \ell$ and $|m_i|$ is the order of m_i that can be found using Shor's order finding algorithm.

Using similar techniques of problem 1 the system above becomes a system of linear equations.

The solution to this system are elements of $I \cap J$. Using techniques of Oracle I we can find an additive generating set for $I \cap J$. Hence applying Oracle I and II to the additive generating set we can find generators together with structure constants of $I \cap J$.

- **Similar Problems:**

  The following are examples of problems that can be reduced to a system of linear Diophantine equations.

  - Computing the colon ideal:

    If I and J are two ideals of the ring R, one defines $(I : J) = \{x \in R | xJ \subseteq I\}$. $(I : J)$ is an ideal, and is called an ideal quotient or a colon ideal. $(I : J)$ is a subgroup of $(R, +)$.

– Computing the Annihilator:

  The left annihilator A_S of $S = \{s_1, \ldots, s_n\} \subseteq R$ is defined as $A_S = \{x \in R | xs_1 = 0, \ldots, xs_n = 0\}$. A_S forms a subgroup of $(R, +)$.

– Injectivity and surjectivity of homomorphisms:

  Given a black-box implementing a homomorphism $\rho : R \to R'$ between two rings. Determining whether ρ is injective or surjective.

– Unit test

  Given $r \in R$, we want to test if r is a unit. let Rr be the left ideal in R generated by r. $Rr = R$ if and only if r is a unit.

– Finding the multiplicative and additive identity

5.4 Decomposition of Algebras

One of the fundamental problems in algebra is decomposing mathematical structures into direct sum of simpler structures. For algebras and modules, in most cases a set of generators and structure constants are assumed to be given. While having this information in hand many authors were able to achieve important results regarding associative algebras and modules. But if the algebra or module was given in a black-box setting, or the set of generators were not given as a linear basis. Then there is no efficient algorithm to find the structure constants for these structures. Therefore our algorithms will provide the necessary tools for these results.

Its known that the structure of Abelian black-box groups as in the structure theorem of Abelian groups, can be determined efficiently on a quantum computer[CM01, Wat01]. Moreover, other properties of these groups such as the order of the group and order of its elements can also be computed using Shor's algorithm. As we know, there are other structure theorems for mathematical structures such as Modules and Algebras. Therefore In this section we show that we can generalize this result to more complicated algebraic structures such as modules and associative algebras. We expect these results can be generalized to any mathematical structure containing an Abelian group. In this section we consider decomposition of associative algebras and modules over finite fields.

An algebra A over a finite field F is a vector space over F, together with a F-bilinear operation(multiplication). The algebra A with the multiplication forms a ring (not necessarily commutative). If the multiplication is associative the algebra is called an *associative algebra*.

An element $x \in A$ is called *nilpotent* if there is a positive integer m such that $x^m = 0$. An element $x \in A$ is called *strongly nilpotent* if for every $y \in A$, xy is nilpotent. An algebra is called *semi-simple* if it has no strongly nilpotent element except zero and called *simple* if it has no nontrivial two sided ideals. The following theorem due to Wedderburn [Wed08] provides a decomposition for semi-simple algebras.

Theorem 9 (Wedderburn Structure theorem)**.** *Let A be a finite dimensional semi-simple algebra over the field F. Then A can be expressed by a direct sum of simple algebras*

$$A = A_1 \oplus A_2 \oplus \cdots \oplus A_k, \tag{5.16}$$

where the A_i are the only minimal nontrivial ideals of A. Moreover, each A_i is isomorphic to a full matrix algebra $M_{n_i}(F_i)$ where F_i is a not necessarily commutative extension field of F.

The first known algorithms computing the structure of associative algebras are due to Friedl and Rónyai [FR85], who gave an efficient classical algorithm to compute the Jacobson radical and to decompose a semi-simple algebra over a finite field as a direct sum of simple algebras (Wedderburn Structure theorem). In [Ron87] Rónyai gives polynomial algorithms for problems in associative algebras such as computing the Jacobson radical and finding zero divisors. In a subsequent paper [IR93] Ivanyos and Rónyai show that constructing a maximal order in a semisimple algebra over an algebraic number field can be solved by a efficient classical algorithm if there is an oracle for integer factorization. Were in the quantum case by Shor's algorithm we know that this is tractable. Another interesting result is due to Gianni [GMT89] who provides an efficient algorithm for decomposing Abelian algebras into local algebras. Chistov, Ivanyos and Karpinski in [CIK97] present a polynomial algorithm for decomposing Modules over finite fields as direct sums of indecomposable modules.

As you see, many problems can be solved by deterministic or probabilistic methods for algebras and modules when they are given as generators and structure constants. Therefore the Oracles I and II gives a good primitive to find generators and the structure constants of algebras and modules if given as black boxes. Hence, Oracles I and II introduced in this chapter together with known quantum algorithms such as integer factorization, Abelian HSP and swap test, we can compute many properties of finite associative algebras and modules.

5.5 Directions for Future Work

We conjecture that our quantum algorithms apply to any category possessing a faithful functor to the category of Abelian groups.

It would be interesting to find efficient quantum algorithms for deciding whether a given ideal I is principal and computing the group of units R^* of R.

It is not obvious that the above algorithms extend to arbitrary algebras and modules over infinite fields. However, it seems likely that the above algorithms could be extended to a black-box ring R which is endowed with a grading by Abelian groups $R_0, R_1, R_2, \ldots$ and each component R_g is finite. Additionally, we would need a promise, making it possible to do all the computations in a component R_g for some g. For example, such a situation occurs for polynomial rings over a finite fields when the number of indeterminates is fixed. The complexity of the algorithms would then depend on the growth of the Hilbert function, which measures the dimension of the graded components R_g as R_0-modules.

LIST OF NOTATIONS

$\langle\psi\|$	bra vector in Dirac's notation.
$\langle\psi\|\phi\rangle$	The scalar (inner) product of $\|\psi\rangle$ and $\|\phi\rangle$
$\mathbb{C}^n$	n-dimensional vector space over the filed of complex numbers.
$\mathcal{H}_n$	n-dimensional Hilbert space.
$\Delta_L(t)$	Alexander polynomial of link L
δ_{ij}	Kronecker's delta function
$\|\psi\rangle$	ket vecor in Dirac's notation.
$\|\psi\rangle\langle\phi\|$	The outer product of $\|\psi\rangle$ and $\|\phi\rangle$
$\|\psi\rangle \otimes \|\phi\rangle$	The tensor product of $\|\psi\rangle$ and $\|\phi\rangle$
$\mathbb{I}, \sigma_x, \sigma_y, \sigma_z$	The identity and the Pauli matrices: $\mathbb{I} \equiv \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$; $\sigma_x \equiv \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$; $\sigma_y \equiv \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}$; $\sigma_z \equiv \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$
$\mathbb{I}_n$	The $n \times n$ identity matrix
$\Pr(m)$	Probability of obtaining m
$\mathbb{R}$	The field of real numbers
$\sum_m$	Summation over index m

$\mathbb{Z}$	The field of integers
A^T	Transpose of matrix A_{mn}
$A^\dagger$	Complex conjugation of matrix A_{mn}. For instance, if $A = \begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \end{pmatrix}$, then $A^\dagger = \begin{pmatrix} \alpha_1^* & \alpha_2^* & \alpha_3^* \end{pmatrix}$
$c(G)$	Number of connected components of graph G
$CNOT$	The controlled-NOT gate, its matrix representation is $CNOT = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$
$e = 2.718\ldots$	Euler's number
E	Edge set of a graph
$e^{i\alpha}$	Euler's formula: $e^{i\alpha} = \cos\alpha + i\sin\alpha$
$F_L(a, z)$	Kauffman polynomial of link L
$G - e$	Graph obtained by removing edge e
G/e	Graph obtained by contracting edge e

H, S, T	The Hadamard (H), phase (S), and $\pi/8$ (T) matrices for one qubit gates: $H \equiv \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$; $S \equiv \begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix}$; $T \equiv \begin{bmatrix} 1 & 0 \\ 0 & \exp(i\pi/8) \end{bmatrix}$
$i = \sqrt{-1}$	Imaginary number. Square root of -1.
M_m	Measurement operator
$P_L(\ell, m)$	HOPMFLY polynomial of link L
$T_G(x, y)$	Tutte polynomial of graph G
V	Vertex set of a graph
$V_L(t)$	Jones polynomial of link L
$w(G)$	Width of graph G
AQFT	Approximate quantum Fourier transform
BQP	Bounded error quantum polynomial time
DQC1	Deterministic quantum computation with one quantum bit
FPRAS	Fully Polynomial Randomized Approximation Scheme
QFT	Quantum Fourier transform
QPE	Quantum phase estimation

LIST OF REFERENCES

[AAE07] D. Aharonov, I. Arad, E. Eban, and Z. Landau. "Polynomial Quantum Algorithms for Additive approximations of the Potts model and other Points of the Tutte Plane.", 2007.

[AC] H. Ahmadi and C. Chiang. "Quantum Phase Estimation with Arbitrary Constant-precision Phase Shift Operators." *Quantum Information & Computation*. to appear.

[Ada00] C.C. Adams. *The Knot Book: An Elementary Introduction to the Mathematical Theory of Knots*. Henry Holt and Co., 2000.

[ADM06] V. Arvind, Bireswar Das, and Partha Mukhopadhyay. "The complexity of black-box ring problems." In *Computing and combinatorics*, volume 4112 of *Lecture Notes in Computer Science*, pp. 126–135. Springer, Berlin, 2006.

[AFW95] N. Alon, A. Frieze, and D. Welsh. "Polynomial time randomized approximation schemes for Tutte-Gröthendieck invariants: the dense case." *Random Structures & Algorithms*, **6**(4):459–478, 1995.

[AJL06] D. Aharonov, V. Jones, and Z. Landau. "A polynomial quantum algorithm for approximating the Jones polynomial." In *Proceedings of the thirty-eighth annual ACM symposium on Theory of computing*, STOC '06, pp. 427–436, New York, NY, USA, 2006. ACM.

[Art91] M. Artin. *Algebra*. Prentice Hall Inc., Englewood Cliffs, NJ, 1991.

[AW10] H. Ahmadi and P. Wocjan. "On the quantum complexity of evaluating the Tutte polynomial." *Journal of Knot Theory and its Ramifications*, **19**(6):727–737, 2010.

[BES96] A. Barenco, A. Ekert, K. "Approximate quantum Fourier transform and decoherence." *Physical Review A. Third Series*, **54**(1):139–146, 1996.

[BFL05] M. Bordewich, M. Freedman, L. Lovász, and D. Welsh. "Approximate counting and quantum computation." *Combinatorics, Probability and Computing*, **14**(5-6):737–754, 2005.

[CC82] T. J. Chou and G. E. Collins. "Algorithms for the solution of systems of linear Diophantine equations." *SIAM Journal on Computing*, **11**(4):687–708, 1982.

[Che04] D. Cheung. "Improved bounds for the approximate QFT." In *Proceedings of the winter international synposium on Information and communication technologies*, WISICT '04, pp. 1–6. Trinity College Dublin, 2004.

[CIK97] A. Chistov, G. Ivanyos, and M. Karpinski. "Polynomial time algorithms for modules over finite dimensional algebras." In *Proceedings of the 1997 International Symposium on Symbolic and Algebraic Computation (Kihei, HI)*, pp. 68–74 (electronic), New York, 1997. ACM.

[CM01] K. Cheung and M. Mosca. "Decomposing finite abelian groups." *Quantum Information & Computation*, **1**(3):26–32, 2001.

[DFC05] A. Datta, S. Flammia, and C. Caves. "Entanglement and the Power of One Qubit." *Physical Review A*, **72**(4):22, 2005.

[FKW02] M. H. Freedman, A. Kitaev, and Z. Wang. "Simulation of topological field theories by quantum computers." *Communications in Mathematical Physics*, **227**(3):587–603, 2002.

[FLW02] M. H. Freedman, M. Larsen, and Z. Wang. "A modular functor which is universal for quantum computation." *Communications in Mathematical Physics*, **227**(3):605–622, 2002.

[FR85] K. Friedl and L. Rónyai. "Polynomial time solutions of some problems of computational algebra." In *Proceedings of the seventeenth annual ACM symposium on Theory of computing*, STOC '85, pp. 153–162, New York, NY, USA, 1985. ACM.

[GJ08] L. Goldberg and M. Jerrum. "Inapproximability of the Tutte polynomial." *Information and Computation*, **206**(7):908–929, 2008.

[GMT89] P. Gianni, V. Miller, and B. Trager. "Decomposition of algebras." In *Symbolic and algebraic computation (Rome, 1988)*, volume 358 of *Lecture Notes in Computer Science*, pp. 300–308. Springer, Berlin, 1989.

[GN96] R. Griffiths and C. Niu. "Semiclassical Fourier transform for quantum computation." *Physical Review Letters*, **76**(17):3228–3231, 1996.

[Hal07] S. Hallgren. "Polynomial-time quantum algorithms for Pell's equation and the principal ideal problem." *Journal of the ACM*, **54**(1), 2007.

[Has98] J. Hass. "Algorithms for recognizing knots and 3-manifolds." *Chaos, Solitons and Fractals*, **9**(4-5):569–581, 1998. Knot theory and its applications.

[HS98] J. W. Harris and H. Stocker. "Maximum Likelihood Method." In *Handbook of Mathematics and Computational Science*, p. 824. Springer-Verlag, New York, 1998.

[HW08] G. H. Hardy and E. M. Wright. *An introduction to the theory of numbers*. Oxford University Press, Oxford, sixth edition, 2008.

[IR93] G. Ivanyos and L. Rónyai. "Finding maximal orders in semisimple algebras over Q." *Computational Complexity*, **3**:245–261, 1993.

[Jae88] F. Jaeger. "Tutte polynomials and link polynomials." *Proceedings of the American Mathematical Society*, **103**(2):647–654, 1988.

[Jon86] V. F. R. Jones. "Braid groups, Hecke algebras and type II_1 factors." In *Geometric methods in operator algebras (Kyoto, 1983)*, volume 123 of *Pitman Res. Notes Math. Ser.*, pp. 242–273. Longman Sci. Tech., Harlow, 1986.

[JVW90] F. Jaeger, D. L. Vertigan, and D. J. A. Welsh. "On the computational complexity of the Jones and Tutte polynomials." *Mathematical Proceedings of the Cambridge Philosophical Society*, **108**(1):35–53, 1990.

[Kau01] L. H. Kauffman. *Knots and physics*, volume 1 of *Series on Knots and Everything*. World Scientific Publishing Co. Inc., River Edge, NJ, third edition, 2001.

[Kaw96] A. Kawauchi. *A survey of knot theory*. Birkhäuser Verlag, Basel, 1996.

[Kit96] A. Kitaev. "Quantum Measurements and the Abelian Stabilizer Problem." Technical report, 1996.

[KL98] E. Knill and R. Laflamme. "Power of one bit of quantum information." *Physical Review Letters*, **81**(25):5672–5675, 1998.

[KL07a] L. H. Kauffman and S. J. Lomonaco. "A 3-stranded quantum algorithm for the Jones Polynomial." volume 6573, p. 65730T. SPIE, 2007.

[KL07b] L. H. Kauffman and S. J. Lomonaco. "q-deformed spin networks, knot polynomials and anyonic topological quantum computation." *Journal of Knot Theory and its Ramifications*, **16**(3):267–332, 2007.

[KLM07] P. Kaye, R. Laflamme, and M. Mosca. *An introduction to quantum computing*. Oxford University Press, Oxford, 2007.

[KS05] N. Kayal and N. Saxena. "On the ring isomorphism automorphism problems." In *Computational Complexity, 2005. Proceedings. Twentieth Annual IEEE Conference on*, pp. 2 – 12, June 2005.

[KSV02] A. Kitaev, A. H. Shen, and M. N. Vyalyi. *Classical and quantum computation*, volume 47 of *Graduate Studies in Mathematics*. American Mathematical Society, Providence, RI, 2002.

[Lic97] W. B. R. Lickorish. *An introduction to knot theory*, volume 175 of *Graduate Texts in Mathematics*. Springer-Verlag, New York, 1997.

[LK06] S. J. Lomonaco and L. H. Kauffman. "Topological quantum computing and the Jones polynomial." volume 6244, p. 62440Z. SPIE, 2006.

[LN97] R. Lidl and H. Niederreiter. *Finite fields*, volume 20 of *Encyclopedia of Mathematics and its Applications*. Cambridge University Press, Cambridge, second edition, 1997.

[NC00] M. A. Nielsen and I. L. Chuang. *Quantum computation and quantum information*. Cambridge University Press, Cambridge, 2000.

[Rol90] D. Rolfsen. *Knots and links*, volume 7 of *Mathematics Lecture Series*. Publish or Perish Inc., Houston, TX, 1990.

[Ron87] L. Ronyai. "Simple algebras are difficult." In *Proceedings of the nineteenth annual ACM symposium on Theory of computing*, STOC '87, pp. 398–408, New York, NY, USA, 1987. ACM.

[Sho94] P. W. Shor. "Algorithms for quantum computation: discrete logarithms and factoring." In *35th Annual Symposium on Foundations of Computer Science (Santa Fe, NM, 1994)*, pp. 124–134. Los Alamitos, CA, 1994.

[Sho97] P. W. Shor. "Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer." *SIAM Journal on Computing*, **26**(5):1484–1509, 1997.

[SJ08] P. W. Shor and S. P. Jordan. "Estimating Jones polynomials is a complete problem for one clean qubit." *Quantum Information & Computation*, **8**(8-9):681–714, 2008.

[Sos02] A. B. Sosinskiĭ. *Knots: mathematics with a twist*. Harvard University Press, 2002.

[SS06] D.S. Sivia and J. Skilling. *Data analysis: a Bayesian tutorial*. Oxford science publications. Oxford University Press, 2006.

[Sze04] M. Szegedy. "Quantum speed-up of Markov chain based algorithms." In *Foundations of Computer Science, 2004. Proceedings. 45th Annual IEEE Symposium on*, pp. 32 – 41, oct. 2004.

[Thi87] M. B. Thistlethwaite. "A spanning tree expansion of the Jones polynomial." *Topology. An International Journal of Mathematics*, **26**(3):297–309, 1987.

[Tut01] W. T. Tutte. *Graph theory*, volume 21 of *Encyclopedia of Mathematics and its Applications*. Cambridge University Press, Cambridge, 2001.

[Tut04] W. T. Tutte. "Graph-polynomials." *Advances in Applied Mathematics*, **32**(1-2):5–9, 2004.

[Wat01] J. Watrous. "Quantum algorithms for solvable groups." In *Proceedings of the Thirty-Third Annual ACM Symposium on Theory of Computing*, pp. 60–67 (electronic), New York, 2001. ACM.

[WCN09] P. Wocjan, C. Chiang, D. Nagaj, and A. Abeyesinghe. "Quantum algorithm for approximating partition functions." *Physical Review A*, **80**:022340, Aug 2009.

[Wed08] J. H. M. Wedderburn. "On Hypercomplex Numbers." *Proceedings of the London Mathematical Society*, **s2-6**(1):77–118, 1908.

[Wel93] D. Welsh. *Complexity: knots, colourings and counting*, volume 186 of *London Mathematical Society Lecture Note Series*. Cambridge University Press, Cambridge, 1993.

[Wen88] H. Wenzl. "Hecke algebras of type A_n and subfactors." *Inventiones Mathematicae*, **92**(2):349–383, 1988.

[WJA] P. Wocjan, S. Jordan, H. Ahmadi, and J. Brennan. "Efficient quantum processing of ideals in finite rings." arXiv:0908.0022.

[WY08] P. Wocjan and J. Yard. "The Jones polynomial: quantum algorithms and applications in quantum complexity theory." *Quantum Information & Computation*, **8**(1-2):147–180, 2008.

[ZMR08] J. Zumbragel, G. Maze, and J. Rosenthal. "Efficient recovering of operation tables of black box groups and rings." In *Information Theory, 2008. ISIT 2008. IEEE International Symposium on*, pp. 639 –643, July 2008.