A Quantum Algorithm For Finding An ε-Approximate Mode

by

Aziz Kara

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Mathematics
in
Computer Science

Waterloo, Ontario, Canada, 2005

I hereby declare that I am the sole author of this thesis.

I authorize the University of Waterloo to lend this thesis to other institutions or individuals for the purpose of scholarly research.

Aziz Kara

I further authorize the University of Waterloo to reproduce this thesis by photocopying or other means, in total or in part, at the request of other institutions or individuals for the purpose of scholarly research.

Aziz Kara

Abstract

Combinatorial problems have long been a focal point of theoretical computer science, with much of the interest stemming from the fact that many naturally occurring problems can be viewed in combinatorial settings.

In this thesis, we explore the problems of almost uniform generation and approximate counting from the perspectives of quantum information. The aim is to show that treating these problems quantum mechanically offers improvements over conventional methods.

Almost uniform generation can be used to estimate set sizes by sampling, which is the main focus of the work by Jerrum, Valliant, and Vazirani [17]. They propose a fully polynomial randomized approximation scheme (FPRAS) for approximate counting, which is based on an almost uniform generator. In this thesis, we propose a *quantum FPRAS* that makes roughly quadratically fewer queries to the almost uniform generator in accomplishing the same task.

We also propose a quantum algorithm for estimating the statistical mode of a list of elements. Consider a list $\mathbf{S}$ of N integers drawn from a set $\mathbf{D}$ of M integers. The *statistical mode* of the list is the integer which occurs most often in $\mathbf{S}$. For example, if $\mathbf{D} = \{1,2,3\}$ and $\mathbf{S} = \{1,2,3,1,3,2,3,2,1,2,2\}$, then the mode of $\mathbf{S}$ is 2 as it occurs most often, five times. We define an ε-approximate mode of the list $\mathbf{S}$ as any element that occurs within ratio $(1+\varepsilon)$, $\varepsilon > 0$ of the frequency of the modal element.

The algorithm we propose is an approximation algorithm in that it finds an approximate mode of the list. Consequently, we refer to it as the approximate mode finding algorithm. We will also show that this scheme is indeed an $\{\varepsilon,\delta\}$-FPRAS in that it approximates the frequency of the

mode within ratio $(1+\varepsilon)$ with probability at least $1-\delta$, using a number of list queries that is a polynomial in $1/\varepsilon$, $\log(1/\delta)$, and M.

Counting in a quantum mechanical fashion is integral to the approximate mode finding algorithm, and in order for it to work, we must use a *uniformized* version of the quantum phase estimation algorithm. Standard quantum phase estimation approximates a phase parameter $0 \leq \omega \leq 1$ in the phase of an eigenvalue for some eigenstate of a given unitary operator, and returns an approximation whose accuracy is dependent on ω itself. This means that the distributions for the estimates of two different phase parameters might be significantly different. Uniformization is a technique which attempts to remove this dependency, thereby making such estimates independent of the unknown phase parameters. In turn, a uniformized phase estimation algorithm yields a *monotonized* counting algorithm which is a key ingredient of the mode finding $\{\varepsilon,\delta\}$-FPRAS.

Consider counts a_1 and a_2 such that $0 \leq a_2 < a_1 \leq N$, and let approximations to these quantities be $\widetilde{a_1}$ and $\widetilde{a_2}$ respectively. In comparing the distributions of these two approximations, we would like that $\Pr[\widetilde{a_1} > w] \geq \Pr[\widetilde{a_2} > w]$, where $0 \leq a_2 < a_1 \leq w \leq N$ is some threshold. If quantum counting is based on a non-uniformized version of the phase estimation algorithm, then such relations are not necessarily true. In fact, in some instances, this would be outright false. A *monotonized* version of the counting algorithm enables us to make such claims up to a small corrective error.

The mode finding FPRAS, and therefore the uniformized phase estimation algorithm, are used in order to quantize the approximate counting FPRAS of Jerrum et al. [17]. As such, these three topics are the focal points of this thesis.

We start in Chapter 1 by giving a number of motivations for pursuing research in quantum information science. We also list the common themes from computer science that will be encountered throughout. These topics will assume some background on the part of the reader; comprehensive references will be given where required.

In Chapter 2, we present a brief description of the underlying quantum mechanical framework used throughout. In Chapter 3, we describe quantum models of computation, which are based on the framework of Chapter 2. In Chapter 4, we give the reader a taste of quantum algorithms by describing Deutsch's algorithm and the Deutsch-Jozsa algorithm. We review the concepts of quantum searching and counting in Chapter 5, both of which are used extensively in Chapters 6 through 8.

In Chapter 6, we explore the technique of uniformization and give a uniformized phase estimation algorithm. The uniformized phase estimation algorithm leads to a *monotonized* counting algorithm which is also described in Chapter 6. With these tools in hand, we discuss the $\{\varepsilon, \delta\}$-FPRAS for mode finding in Chapter 7, and include a detailed analysis of its correctness. Finally, we present an application of the mode finding FPRAS in Chapter 8 by quantizing an FPRAS for approximate counting, originally proposed by Jerrum, Valliant, and Vazirani.

This thesis contains original results on my part that were greatly facilitated through the help of my supervisors, Michele Mosca and Prabhakar Ragde. The motivations in Chapter 1 are based on some personal articles I wrote in the fall of 2002. The information therein draws mostly from personal reflection and other similar pieces that aim to motivate research in QIP.

The contents of Chapter 2 are almost entirely based on an advanced quantum mechanics

course that was offered at the University of Waterloo in the fall of 2002 by Achim Kempf. The material in that chapter is drawn almost entirely from course lecture notes.

Chapters 3 to 5 are based on standard discussions of those topics with some added personal interpretations. Most of the material therein is based on course material from an Introductory Quantum Computing course taught by Michele Mosca, and from the standard text for QIP by Nielsen and Chuang [23]. I give my personal intuitive understanding in explaining the various quantum algorithms in chapters 4 and 5, which I believe to be a useful alternative point of view.

The material in Chapter 6 was largely motivated and introduced to me by Michele. My original contributions in that chapter include the introduction where I give a more extensive motivation for uniformized phase estimation. My derivation of the success probabilities in section 6.3.2 is also more extensive than the original discussion [21]. The material in the remaining sections are all original contributions as far as I know. In particular, the approximate monotonicity results and the monotonized counting algorithm are original contributions that I have not yet encountered.

Almost all of the material in Chapter 7 is original with the exception of the Maximum finding algorithm which is really a corollary of the minimum finding algorithm of Dürr and Høyer [14]. The analysis of the mode finding approximation schemes in section 7.2 is based on the work of Jerrum et al. [17] and the material in Motwani and Raghavan [22]. The original presentations did not include such a description and therefore these analyses are original contributions.

The material in Chapter 8 is based largely on the work of Jerrum et al. [17]. A significant portion of this chapter is devoted to explaining their work, while the remainder of the chapter is devoted to incorporating the quantum approximate mode finding algorithm in order to quantize

their result. The quantization is therefore an original contribution.

Acknowledgments

Writing this thesis has been more than just an eight month journey. Along the way, many people have selflessly lent me their support.

First and foremost, I would like to thank my supervisors Michele Mosca and Prabhakar Ragde for their outstanding academic support and constant encouragement. I would also like to thank them for their generous financial support without which, this degree would not have been possible. Both Michele and Prabhakar were extremely flexible and allowed me to find my own way to this end, despite it taking a little longer than expected.

I am also indebted to my parents Amina and Allauddin, my brother Azim, and Roxanna, all of whom sacrificed a lot for me to complete this thesis. Their constant support and reassurance were always good confidence builders. Despite not knowing what I was doing, they always held me in high regard and were sympathetic when I took longer than expected to finish.

This work has also benefited from the help of many IQC students that have sat through my otherwise incomprehensible descriptions of technical problems I faced throughout the compilation of this thesis. Their reassuring advice always made me feel that I was not too far off the typical path of a Master's student. I am also thankful to my thesis readers Ashwin Nayak and Richard Cleve who agreed to evaluate my thesis for correctness. A reader stands to gain very little from such a work while sacrificing their own research and personal time, and for this, I am very grateful to them.

I dedicate this work to these people as a small token of my appreciation.

Contents

1 Introduction **1**

1.1 Motivations 1

1.1.1 Information theoretic motivations 2

1.1.2 Industry motivations 3

1.1.3 Transistor technology will remain 5

1.1.4 Motivations for the Physical Sciences 5

1.1.5 Information and physics 10

1.2 Motivations for Thesis Topics 12

1.3 Thesis Topics 14

1.4 Computer Science Background 15

2 A Brief Introduction to Quantum Mechanics **19**

2.1 Hilbert Spaces 20

2.2 Quantum States that Encode Information 24

2.2.1 Single qubit states 24

2.2.2 Multiple qubit states 27

2.2.3 Multiple qubit states in large Hilbert spaces 28

2.2.4 Entangled states 28

2.2.5 Other relevant topics . . . 30
2.3 Operations on Quantum States . . . 31
2.3.1 Quantum evolution and unitary operations . . . 31
2.4 Measurements on Quantum Systems . . . 35

3 Quantum Mechanical Models of Computing 37
3.1 The Universal Quantum Turing Machine . . . 37
3.2 The Quantum Circuit Model . . . 38

4 Putting it all together: The First Quantum Algorithms 41
4.1 Deutsch's Algorithm . . . 41
4.2 The Deutsch-Jozsa Algorithm . . . 45

5 Quantum Searching and Quantum Counting 49
5.1 Quantum Searching . . . 49
5.1.1 Quantum amplitude amplification . . . 50
5.1.2 Quantum searching by amplitude amplification . . . 52
5.2 Quantum Counting . . . 56
5.2.1 The quantum Fourier transformation . . . 56
5.2.2 Quantum phase estimation . . . 57
5.2.3 Counting by phase estimation . . . 59

6 Uniformized Phase Estimation and Monotonized Counting 61
6.1 Introduction . . . 61
6.2 Uniformized Phase Estimation . . . 64
6.2.1 Understanding uniformization . . . 64
6.2.2 The algorithm . . . 67

6.3 Errors in the Uniformized Algorithm 69
6.3.1 Re-centering the estimator without affecting probabilities 69
6.3.2 Success probabilities 71
6.4 Approximate Monotonicity of the Uniformized Probability Function 75
6.4.1 Approximate monotonicity in continuously uniformized distributions . . 76
6.4.2 Uniformized distributions are close 83
6.5 Query Complexity of the Uniformized Phase Estimation Algorithm 86
6.6 The Operator $\mathbf{U_R}$ 86
6.7 Applications to Counting 87
6.7.1 Monotonized quantum counting 88
6.7.2 Approximate monotonicity of the monotonized quantum counting algorithm 90
6.7.3 Query complexity of the monotonized quantum counting algorithm . . . 92
6.8 Concluding Remarks and Open Questions 92

7 A Quantum $\{\varepsilon, \delta\}$-FPRAS for Finding the Statistical Mode 95
7.1 Introduction 95
7.2 A Fully Polynomial Randomized Approximation Scheme for Mode Finding . . . 97
7.2.1 A randomized approximation scheme 97
7.2.2 An $\{\varepsilon, \delta\}$-FPRAS for mode finding 100
7.3 Preliminary Details and Facts 102
7.3.1 Quantum searching 102
7.3.2 Quantum counting 102
7.3.3 A maximum finding algorithm 103
7.4 Replacing the Counting Oracle with a Quantum Counter 105
7.4.1 Why replacement with a quantum counter is not straight-forward 106
7.4.2 Using the uniformized counting algorithm 108

7.4.3 Correctly computing F . . . 110
7.4.4 Counts as ranks . . . 115
7.5 The Algorithm . . . 119
7.5.1 Finding the approximate mode . . . 119
7.5.2 Amplifying the success probability . . . 126
7.6 Concluding Remarks . . . 127

8 Approximate Counting by Almost Uniform Generation and Quantum Methods 129
8.1 Introduction . . . 129
8.2 The Results of Jerrum, Valiant, and Vazirani . . . 129
8.3 Definitions and Notation . . . 130
8.4 Randomized Approximate Counting . . . 133
8.4.1 Approximating $N_R(x)$ . . . 133
8.4.2 An FPRAS for approximate counting . . . 135
8.5 A Quantum FPRAS for Counting . . . 138
8.5.1 Counting by querying the generator . . . 139
8.5.2 Sufficient statistics . . . 140
8.5.3 Quantum mechanically estimating these statistics . . . 143
8.6 The Quantum FPRAS . . . 145
8.6.1 Query complexity . . . 147
8.7 Concluding remarks . . . 148

A Useful Facts 149
A.1 Proof of Fact 1 . . . 149
A.2 Derivative Bounds . . . 150
A.3 Approximate Integration . . . 153

A.4 A Variant of the Powering Lemma 156
A.5 Expected Value of a Monotone Increasing Probability Function 158
A.6 Probabilistic Recurrences 160

Chapter 1

Introduction

1.1 Motivations

Quantum Information Processing (QIP) is a relatively new area of research where the goal is to determine if a quantum-mechanical paradigm of information processing will lead to fundamentally new ways of manipulating information. QIP aims to manipulate a quantum physical system to perform information processing tasks. Examples of such systems include atoms or electrons, which have quantum mechanical properties that are exploited so that we may perform some computational task.

In this chapter, we outline some of the motivations for pursuing research in QIP. The three main themes that fuel this interest are: information-theoretic motivations, industry motivations, and motivations for the physical sciences. QIP is an interdisciplinary field that brings together mathematicians, physicists, and computer scientists, all of whom find some motivation from these three themes.

1.1.1 Information theoretic motivations

Many daily tasks require us to manipulate information, which we do by processing some input to produce an output. For example, given a map of a city along with an origin A and a destination B, one could process it by drawing a route from A to B. In this way, we have processed the map and the points A and B (the input) to produce a route from A to B (the output).

Another example is the well-known "Traveling Salesperson Problem" (TSP) where a salesperson must travel to different cities in a country to sell a product. To minimize traveling costs, we have to find a route that will minimize the total distance traveled. This problem falls into a class of optimization problems considered difficult to solve "efficiently". Such optimization problems are often considered the bane of computer science, and their efficient solution its holy grail.

If we had a quantum computer at our disposal, could we solve the TSP any faster than on a conventional computer? Addressing this question at this point would be premature, so instead, we examine the basic ideas of information manipulation.

For a human, it is natural to make a mental representation of a problem when attempting to solve it. Here lies the first problem for an abstract computing device: How do you represent a problem? More generally, how do you represent information? If a human were to solve the TSP, it would most likely involve a mental representation of a map and flight paths between the cities to be visited. But this is a complex model and, more importantly, hard to represent on a computing device.

The basis of all information representation on computers today is the binary number sys-

tem which consists of two numbers, 0 and 1. These two numbers are the building blocks of a representation system for everything that can be represented or modeled by current devices. Physically, the binary number system is modeled by voltage levels manipulated by transistors, which are the elementary data transformation units.

Through the advancement of manufacturing technology, the computer industry has been able to fabricate smaller transistors, thereby being able to fit more of them per square centimetre on a silicon-based computer chip. In addition to the greater density, the reduced size means that more transistors can be manipulated with a fixed amount of energy. All of this amounts to manipulating more binary information for the same resource cost.

A natural question that follows is "How small can we make these elementary units?" An obvious bound is the size of the smallest objects we know to exist in nature. In order to answer this question, we must turn to subatomic physics, which is best described by quantum mechanics. Thus, from an information processing standpoint, it makes sense to understand quantum mechanics if we are interested in answering these questions.

Quantum mechanics also presents some unintuitive behaviour of matter, which scientists have struggled to understand. How to harness such phenomena for the purposes of information processing is also another question that researchers aim to answer.

1.1.2 Industry motivations

The computer technology industry has been able to make smaller components with the advancement of manufacturing methods. This has led to the introduction of micro-computing devices that are ubiquitous today. At the heart of this revolution is the ability of manufacturers to make

increasingly smaller components. As a by-product, more computing power is possible per fixed amount of space.

Demand for this greater information processing capability has fueled a rapid advancement in the technology industry. The sustaining force behind this demand is the continual advancement of manufacturing methods which ultimately aim to do one thing – shrink transistor technology as much as possible.

In 1965, Dr. Gordon Moore noticed a trend in circuit technology which today is known as "Moore's Law". It is not a mathematical law, but a trend that has persisted, hence the resemblance to a law of nature. Moore's law (paraphrased) states that "The number of transistors that can fit into a fixed area roughly doubles every 18 months". This statement has two profound implications, the first of which is that roughly every two years, manufacturing advances enable us to make a transistor of roughly *half the size* of its predecessor. Thus in a fixed area, we can place roughly twice as many new transistors as we could place old transistors. This means that we could manipulate roughly twice the amount binary information that was previously possible, and being able to manipulate more binary information is precisely an increase in computational power.

The second implication is that as time goes on, manufacturing advances will enable us to create transistors at atomic scales. At this point, it would no longer be possible to use transistors in the intended ways due to quantum physical effects. Does this impose a limit to our computing power? Yes and no. It is a limit to computing power provided by transistors and semiconductor computer technology. So how do we get around this technological, or more appropriately scientific "brick wall"? The answer is to reconsider our elementary units of data manipulation; a

natural choice is to consider atoms and other atomic-scale objects as candidates.

1.1.3 Transistor technology will remain

It is quite common to cite Moore's law as motivation for considering quantum devices. After reading this sort of discussion, it seems that eventually quantum devices will completely replace conventional mechanisms for storing, processing, and communicating classical information. This is unlikely, as some form of classical control or assistance will always remain. For example, the current forms of quantum error correction require classical control in order to continually remove errors from quantum systems. In addition to this, classical operations are required for quantum state teleportation which can be used for distributed quantum computation. Finally, measurement-based quantum computing schemes incorporate classical control as an integral part. In light of such schemes, it is not even clear that eliminating these mechanisms is desirable. What seems clear is that a marriage of the two will prevail. But then again, no prediction about information technology ever seems safe.

1.1.4 Motivations for the Physical Sciences

All scientific theories are in some way due to experimentation and observation. They can either be formulated by postulating some theory and finding supporting physical evidence, or by inferring the theory from observations. Both methods of formulating scientific theories have been extremely successful and important. For example, Nicolaus Copernicus proposed the idea of planets orbiting around the sun, and subsequently, Galileo Galilei made observations through a telescope that supported this theory. Johannes Kepler on the other hand, made observations of planetary bodies and discovered the three laws of planetary motion. Kepler used the method of

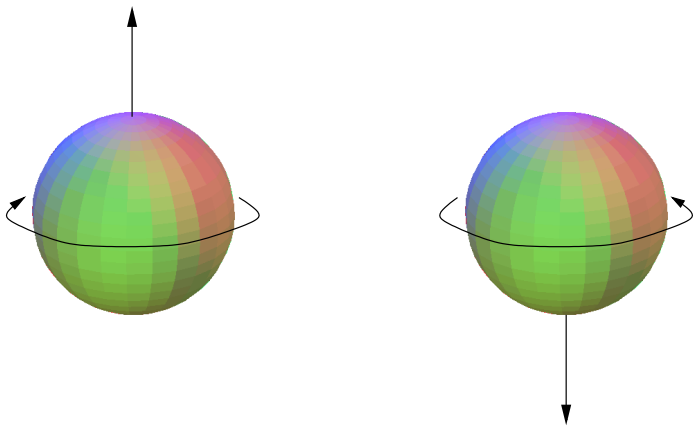
inferring laws from observations, which is a widely-used method in experimental physics.

A scientist will often carry out experiments and attempt to fit observed data to a mathematical model. Sometimes experiments are infeasible: Consider charting the orbit of a newly discovered planet. A scientist would have to chart planetary positions for one orbital cycle i.e., one year of this new planet. Understanding the physics of the universe at initial moments of the big bang cannot involve direct observation or experimentation. These are interesting scientific questions, and determining their answers are difficult, if not impossible. As a result, scientists have found clever and creative ways of trying to understand these phenomena. In the case of the big bang, scientists have used computers and simulation software to understand what must have happened at the initial moments of the universe.

The use of technology in experimentation has many benefits. In the big bang example, scientists hypothesize certain initial conditions of the universe and extrapolate them to current day observations. If the initial conditions and the extrapolation lead to what is observed today, then those initial conditions are deemed to be more plausible based on the observable data. In the same fashion, a quantum physicist can simulate a quantum system to formulate a theory, and then experiment to see if some consequence of the theory actually occurs in nature. If this is the case, then the theory is considered accurate based on what is observed.

Scientific theories attempt to predict behaviour observed in nature, and technology helps to find a starting point when experimentation is hard to do. But what happens when simulating a physical system by technology is not possible? Often, quantum physicists are interested in understanding the quantum interactions of a number of atoms. How do they affect one another? How does one atom's spin orientation change when another one is close by? An atom can spin

in a leftward or rightward direction. If it spins leftwards, then it induces an upward pole, while if it spins rightward, it induces a downward pole (see figure 1.1).



(a) (b)

Figure 1.1: An atom spinning in a leftward direction (a) induces an upward pointing pole, while an atom spinning in a rightward direction (b), induces a downward pointing pole. This correspondence is summarized by the popular "left-hand" rule often taught in high school level physics courses.

Now consider a line of atoms, each of which has its own induced spin pole - either upwards or downwards. As mentioned earlier, quantum physical effects allow for interesting things to start happening. For instance, in a *quantum superposition*, all of these atoms in the line can have a spin orientation upward, and downward at the same time. Any mix of orientations is also possible. In figure 1.2 we have illustrated all eight possible orientations for three atoms in a line. The superposition principle of quantum physics says that with three atoms in a line, all eight configurations can exist at once.

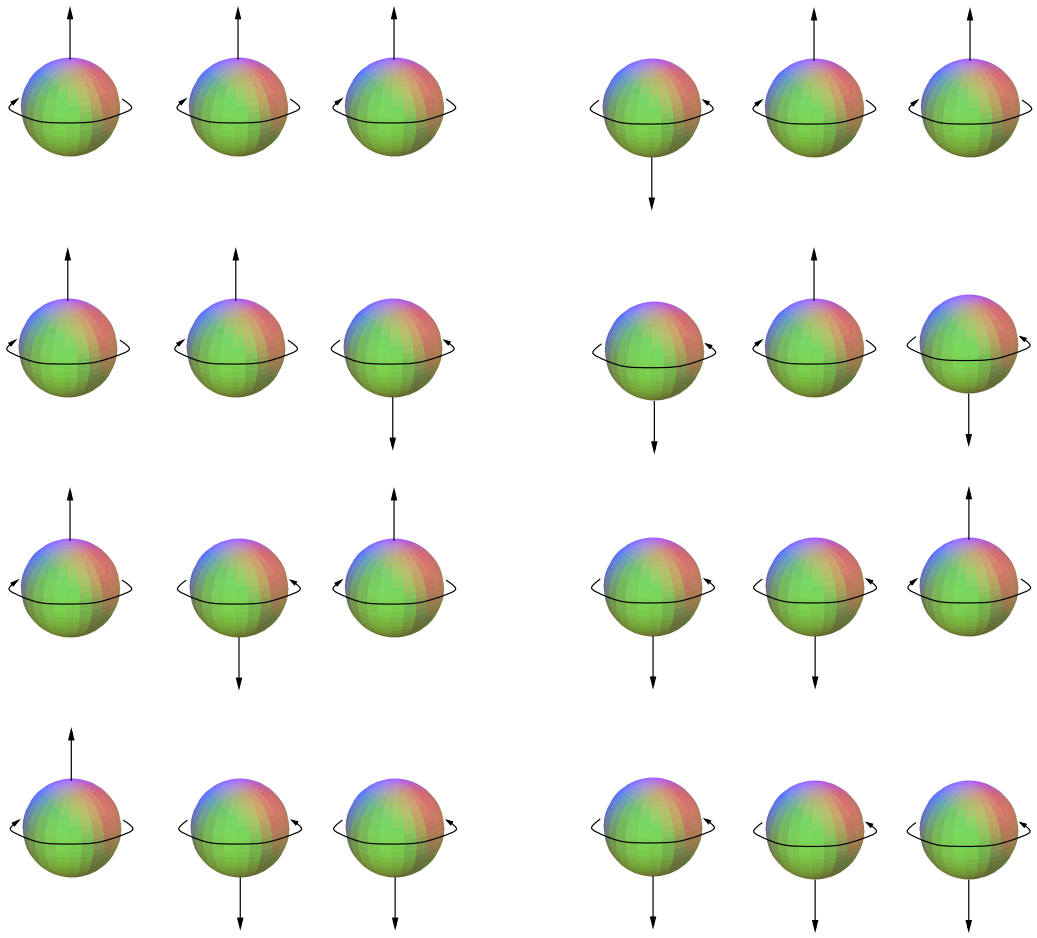


Figure 1.2: All eight possible pole configurations for three atoms in a line.

To simulate atomic activity at the quantum level, we would have to account for all possible configurations, and be able to predict what should happen to each configuration in order to be able to infer a global property at the end of the experiment. Now managing a small number of configurations is not hard, but consider a line of 300 atoms which is easy to visualize, yet hard to simulate. The number of configurations we would have to manage would be 2^{300} which is more configurations than the number of atoms in the universe. Clearly, our desktop computers could not perform this calculation (or anything close to it). With a quantum computer at our disposal, it is a conceivable experiment to line up 300 atoms and use the quantum mechanical powers of nature to do the computation for us, and in the end measure our desired property. We evolve the quantum system in nature instead of simulating its evolution. Intuitively, it is like learning something about fuel consumption in a car. We could write a computer program to monitor all the atoms of the car to see how many kilometers we get on a full tank, or we could simply drive the car until we run out of gas.

Quantum information processing in this light offers a way to hand a job off to nature and let it do the work for us. All that is left is to find a way to implement a robust quantum computer that can handle 300 (or more) atoms. The theme in the last example was the notion of uncomputability due to a limitation in resources. We do not have the computing power, much less the time, to simulate all possible configurations of a quantum state. Our current technology is effectively useless in simulating large-scale quantum experiments. Exploiting the quantum mechanical features of nature will enable us to learn more about it which is another reason why the field has attracted considerable attention.

1.1.5 Information and physics

The two most compelling reasons for pursuing quantum information, however, might be the ideas that *information is physical* and *physics is informational*. The first idea is due to to Rolf Landauer, who hypothesized that information is fundamentally physical. He believed that every physical system registered some amount of information. Particles capable of having upward and downward poles register one bit of information, for example.

More recently, in a November 2004 Scientific American article, Seth Lloyd and Jack Ng discussed the possibility that the entire universe is a large computer. In their article, they also reviewed Stephen Hawking's recent change of opinion on the matter of Hawking radiation from a black hole. The general understanding is that matter can enter a black hole, but not leave it. Stephen Hawking showed that black holes, should in principle, glow like a hot coal, and that the "glow" would be in the form of radiation, which has been named in honour of Hawking. Until recently, Hawking radiation was thought to be completely random as per Hawking's hypothesis, but under a certain set of assumptions (which seem viable to the authors), it is not, and thus a black hole too, can be used for manipulating information.

Additionally, there are some that believe that nature is fundamentally discrete, and a corollary of this would be that there exists a true atomic-sized object, i.e., some object that is the smallest possible object in nature. If Landauer is correct, then this hypothetical object too, can be used to compute. But since it is the smallest object, we once again run into the problem of being able to represent information with smaller and smaller physical systems. Such a "smallest" object, aside from representing the limit, would be a small bit of evidence that information is indeed physical.

The theory of computation has traditionally been studied almost entirely in the abstract, as a topic in pure mathematics. This is to miss the point of it. Computers

> *are physical objects, and computations are physical processes. What computers can or cannot compute is determined by the laws of physics alone, and not by pure mathematics.*
>
> *- David Deutsch*

One of Landauer's achievements was to show that the erasure of a bit required the dissipation of energy which is equivalent to an increase in physical entropy. More generally, any gain of information causes a decrease in physical entropy. This fact was used by Bennett [4] in 1987 to solve the Maxwell's daemon problem. Essentially, Maxwell's daemon was a hypothetical situation where a being in a closed system could decrease the entropy of a gas by knowing which particles were moving fast and which were moving slow. Having this information, the daemon could separate the fast particles from the slow ones, thereby reducing the entropy of the gas. This was an apparent violation of the second law of thermodynamics which claims that entropy never decreases in a closed system.

Bennett showed that entropy did not decrease since the daemon was *gaining information*! This result was one of the first examples of where a significant physical problem was solved by information-theoretic techniques. The Hawking radiation problem too, was solved by information theoretic techniques. These examples support the second idea, namely that *physics is informational*, and that we stand to gain a lot by treating physics from an information-theoretic point of view.

These are some of the major reasons why people have attempted to construct quantum informational devices. The resulting theory, known as quantum information science, has been successfully applied to solve major open questions in theoretical physics. Non-physicists too,

stand to gain a lot from the study of quantum information. Industry and those interested in practical computing could take advantage of computational speed-ups offered by nature, for example.

At this time, quantum devices are mostly theoretical, and those that have been built are rudimentary at best (with exception to quantum cryptographical devices, which are already being sold for commercial purposes). There remains the possibility that such devices will not be feasible in practice, and if researchers eventually conclude that this is indeed the case, then at least we will have benefited by answering some fundamentally important physical questions.

1.2 Motivations for Thesis Topics

Combinatorial problems have been a focal point of theoretical computer science for many years. Much of the interest stems from the fact that many naturally occurring problems can be viewed as combinatorial problems. Solving these problems thus requires some amount of exploration of an underlying combinatorial object space.

Consider the traveling salesperson problem (TSP), in which a salesperson must travel to different cities in a country to sell a product. To minimize traveling costs, we have to find a route that will minimize the total distance traveled. The TSP can be formulated as a graph theoretic problem if all of the cities of the country are considered as nodes of a graph, and flight routes are considered as edges. Each edge, or flight path, has an associated cost.

In graph theoretic terms, the problem is reduced to finding the shortest tour of the nodes i.e., a minimum cost path in which each city is visited exactly once. In this setting, solving the TSP requires some sampling of possible tours to determine which is the least costly. One may also

be interested in knowing the number of tours of a given cost; this is known as the associated counting problem i.e., "How many tours exist, having cost at most K?". Intimately related to the counting problem is the uniform generation problem: "Of all the tours having cost at most K, generate one uniformly at random".

The uniform generation of combinatorial structures is of great importance. Amongst its numerous applications is the ability to estimate set sizes by sampling. This is the main focus of the work by Jerrum, Valliant and Vazirani [17], which we explore in this thesis.

Consider a problem P and a problem instance x of P, having a solution set $Y = \{y_i\}$. To generate uniformly, we are required to generate some solution y_i with probability $1/|Y|$. The related counting problem asks for the size of the set Y, i.e. the number of solutions to x.

Quantum mechanics offers a new and potentially more powerful paradigm for information processing. There are certain indications that information is fundamentally physical and therefore quantum at some level. Two extremely important achievements in quantum information processing have been the development of quantum searching and quantum counting algorithms. Both use quantum physical phenomena to harness the power of nature in solving these problems. As mentioned above, counting and searching are intimately related to combinatorial problems, and hence, the study of such quantum algorithms has a direct impact on the future of practical computing.

The quantum searching problem is related to finding a sought-after element in an unstructured and unsorted database. With a quantum physical system encoding the database, one can theoretically apply quantum transformations and operations to find this element faster than any

possible classical deterministic or randomized process. Most combinatorial problems can be seen as database search problems in the absence of any structure in the underlying combinatorial object space. In this sense, tackling the abstract database search problem also implicitly deals with many interesting combinatorial problems.

1.3 Thesis Topics

The topics explored in this thesis are related to quantum searching and counting. In chapter 6, we explore a uniformized version of the quantum phase estimation algorithm which forms the basis of the quantum counting algorithm. Standard quantum phase estimation approximates a phase parameter $0 \leq \omega \leq 1$ in the phase of an eigenvalue for some eigenvector of a given unitary operator, and returns an approximation whose accuracy is dependent on ω itself. This means that the distributions for the estimates of two different phase parameters might be significantly different. Uniformization is a technique which attempts to remove this dependency, thereby making such estimates independent of the unknown phase parameters. In turn, a uniformized phase estimation algorithm yields a *monotonized* counting algorithm.

Consider counts a_1 and a_2 such that $0 \leq a_2 < a_1 \leq 1$, and let approximations to these quantities be $\widetilde{a_1}$ and $\widetilde{a_2}$ respectively. In comparing the distributions of these two approximations, we would like that $\Pr[\widetilde{a_1} > w] \geq \Pr[\widetilde{a_2} > w]$, where $0 \leq a_2 < a_1 \leq w \leq 1$ is some threshold. If quantum counting is based on a non-uniformized version of the phase estimation algorithm, then such relations are not trivially true. In fact, in some instances this would be outright false. A *monotonized* version of the counting algorithm enables us to make such claims up to a small corrective error.

Consider a list **S** of N integers drawn from a set **D** of M integers. The *statistical mode* of the list is the integer which occurs most often in **S**. For example, if $\mathbf{D} = \{1,2,3\}$ and $\mathbf{S} = \{1,2,3,1,3,2,3,2,1,2,2\}$, then the mode of **S** is 2 as it occurs most often, five times. An approximate mode of the list **S** is defined as any element that occurs within ratio $(1+\varepsilon)$, $\varepsilon > 0$ of the frequency of the modal element. Armed with a monotonized counting algorithm, we design an $\{\varepsilon, \delta\}$ fully-polynomial randomized approximation scheme (FPRAS) for finding an approximate mode of **S**.

Finally, we apply this FPRAS to the problem of approximate counting. This focuses on the work by Jerrum, Valliant, and Vazirani [17]. We use the algorithm in order to *quantize* their FPRAS for approximate counting which uses a postulated almost uniform generator. Sampling and counting are integral to these counters, and as such, we attempt to substitute quantum counting and quantum sampling wherever possible. We show that treating the problem quantum mechanically yields an almost-quadratic reduction in the number of generator queries.

1.4 Computer Science Background

The underlying themes in this thesis are based in theoretical computer science. The reader should therefore be comfortable with common topics such as deterministic, non-deterministic, and randomized models of computation. For the first two models, the book by Papadimitriou [24] is an excellent resource, while the book by Motwani and Raghavan [22] is recommended for randomized models.

Additionally, we will use common tools that are frequently used in the analysis of random-

ized processes, including Markov's inequality and Chernoff's bounding methods. These topics are covered in the book by Motwani and Raghavan [22] as well.

Some knowledge of common computational complexity classes will also be assumed. Knowing classes such as **P**, **NP**, **RP**, and **BPP** will be helpful as we make indirect references to them. For example, we mention bounded-error algorithms, and **BPP** is a bounded-error complexity class. Therefore understanding **BPP** will be useful when the term "bounded-error" is used. The definition of the quantum complexity class **BQP** is similar to the definition of **BPP** and may be found in the book by Nielsen and Chuang [23]. In dealing with approximate counting and uniform generation, we mention the complexity class #**P** and algorithms that make use of a #**P** oracle.

In addition to oracles, we will also use *black-box functions*. For example, a function f used in a quantum algorithm will be supplied as a black-box, i.e., f will be supplied as an implicit unitary operator whose implementation is not known. Consequently, the running time of such a function is also not known. In this setting, we count the number of *queries*, or black-box function evaluations, and consider these as the focal statistic in our analysis. In fact, most quantum algorithms are analyzed in terms of the number of queries posed to a black-box function.

All of the quantum algorithms discussed herein will be analyzed by considering the number of queries they make to an underlying black-box function. Alongside this *query complexity* measure, one can keep track of the costs of the other computational operations of the algorithm. This gives two aspects to the final analysis of the problem, which may be useful in some settings.

Often, such black-box functions are known to be efficiently computable, and so the query

complexity analysis yields the asymptotic running time of an algorithm in which the black-box has been replaced with an efficient implementation. Readers familiar with relativized models of computation will be familiar with the notions of query complexity briefly explained here.

Chapter 2

A Brief Introduction to Quantum Mechanics

Quantum mechanics is a mathematical framework capable of describing physical phenomena at the quantum level. In itself, quantum mechanics does not make any mention of physical states in the form of atoms or photons, but rather, it is a collection of abstract spaces and objects which succinctly characterize what is observed in nature.

In order to understand quantum information processing, it is necessary to understand the underlying framework to some extent. While quantum mechanics is a rich collection of tools dealing with infinite dimensional spaces and generalized operators on quantum states, we limit ourselves to finite-dimensional spaces, and unitary operators. A comprehensive survey of this material suitable for those interested in quantum information processing is found in the book by Nielsen and Chuang [23]. Readers that are interested in learning more about quantum mechanics are directed to other texts such as the one by Messiah [19].

2.1 Hilbert Spaces

The enveloping object space in which all quantum states reside is called the *Hilbert space*, named after David Hilbert (1862-1943). To define a Hilbert space in its most general form, we would need to consider infinite-dimensional vector spaces and therefore infinite dimensional vectors.

Definition 1. *A* Hilbert space $\mathcal{H}$ *is an inner product space having a norm defined with respect to the inner product such that every Cauchy sequence has a limit in* $\mathcal{H}$.

Mathematically, infinite dimensional vectors can have a divergent norm, and so the definition of a Hilbert space includes only those spaces in which vectors have converging norms. Intuitively, this is what is meant by every Cauchy sequence having a limit in $\mathcal{H}$. Furthermore, a divergent norm, or divergent vector length, is not consistent with physical observations. Hence, the notion of a vector space in which norms are finite, appropriately characterizes physical states. This is an intuitive reason for why Hilbert spaces are considered in quantum mechanics.

For the purposes of this work, however, we only consider finite-dimensional Hilbert spaces, and thus divergent norms are not an issue. This is because in any finite-dimensional vector space, the norm of a vector must be finite due to a finitely countable spanning set of basis vectors.

In what follows, we describe the notion of a unitary vector space which is a complex inner product space satisfying the conditions of the Hermitean conjugation map. Such a space is sufficient for modeling the type of quantum information processing that is to follow. We assume a basic understanding of linear algebra on the part of the reader.

Consider a vector space H over the complex numbers $\mathbb{C}$, with the operations of addition ($+ : H \times H \to H$) and scalar multiplication ($\cdot : \mathbb{C} \times H \to H$), which obey the standard conditions

and axioms of vector spaces. Such a vector space is called a *complex vector space H*.

Since we are dealing with quantum mechanics, we will use Dirac's Bra-Ket notation[1] throughout, and denote a vector ψ by $|\psi\rangle$, called 'ket ψ'.

Definition 2. *Consider a complex vector space H. The dual space H^* is a set of linear maps denoted by $\{\langle\chi|\}$ that map vectors from H to complex numbers $(\langle\chi| : H \to \mathbb{C})$. More specifically, for a vector $|\psi\rangle \in H$, such a linear map is defined as:*

$$\langle\chi| \quad : \quad |\psi\rangle \to \langle\chi|\psi\rangle \in \mathbb{C}.$$

The vector space H^* is the dual space of H and thus, vectors in H^* are dual vectors of those in H. For example, $\langle\psi|$ (called 'bra ψ') is the dual of $|\psi\rangle$, and is derived from $|\psi\rangle$ by the *Hermitean conjugation* map to be defined shortly.

Any vector space can be described by a set of orthonormal vectors $\{|b_i\rangle\}$, which are a set of unit-length vectors, where the inner product of any two such basis vectors is the Kronecker delta, i.e., $\langle b_i|b_j\rangle = \delta_{ij}$, where

$$\delta_{ij} \quad = \quad \begin{cases} 1 \text{ if } i = j \\ 0 \text{ if } i \neq j. \end{cases}$$

With a fixed choice of orthonormal basis, any state $|\psi\rangle \in H$, can be written as a linear combination of basis states:

[1] Paul Adrien Maurice Dirac (1902-1984) introduced his Bra-Ket notation in his 1930 book *Principles of Quantum Mechanics* [13], and ever since, it has been the de facto notation for quantum mechanics.

$$|\psi\rangle \quad = \quad \sum_i \alpha_i \, |b_i\rangle \tag{2.1}$$

where $\alpha_i = \langle b_i|\psi\rangle \in \mathbb{C}$ are the complex coefficients of the basis vectors.

Definition 3. *The Hermitean conjugation map, denoted* †*, is defined as:* $\dagger \ : \ H \to H^*$

Since H^* is also a complex vector space, it follows that this operator maps vectors from H^* back to H. In this way, $(|\psi\rangle)^\dagger = \langle\psi|$ and $(\langle\psi|)^\dagger = |\psi\rangle$, $\forall\, |\psi\rangle \in H$ (respectively $\forall\, \langle\psi| \in H^*$).

The Hermitean conjugation map also obeys the following axioms:

1. $(|\psi\rangle + |\varphi\rangle)^\dagger = \langle\psi| + \langle\varphi|$
2. $(\alpha\,|\psi\rangle)^\dagger = \alpha^* \langle\psi|$
3. $(\langle\psi|\varphi\rangle)^* = \langle\varphi|\psi\rangle$
4. $\langle\psi|\psi\rangle = 0$ *iff* $|\psi\rangle = \overrightarrow{0}$

where $|\psi\rangle, |\varphi\rangle \in H$, $\alpha \in \mathbb{C}$, and $\overrightarrow{0}$ is the additive identity vector in H. Note that the effect of this operator on complex scalars is exactly that of complex conjugation.

Note that the dual maps implicitly define an inner product operation between two vectors $|\psi\rangle, |\varphi\rangle \in H$:

$$\begin{aligned} \langle\psi|\varphi\rangle \quad &= \quad \sum_{i,j} \alpha_i^* \langle b_i|b_j\rangle \beta_j \\ &= \quad \sum_i \alpha_i^* \beta_i \end{aligned}$$

Since an inner-product operation exists, such a vector space is also classified as an *inner-product space*. In infinite-dimensional spaces, these expressions range over an infinite number of basis vectors and therefore become infinite summations which may or may not converge.

Definition 4. *The norm of a vector $|\psi\rangle$ is defined in terms of the inner-product operation as:*

$$\| |\psi\rangle \| \quad = \quad \sqrt{\langle\psi|\psi\rangle}$$

From definition 4, it follows that Hilbert spaces are exactly those spaces in which inner-product expressions converge. This is because a Hilbert space is defined to only contain vectors with converging norms, and here, the norm is defined in terms of the inner product. Thus, in a Hilbert space, the inner product expression will converge.

If a complex inner-product space H satisfies the conditions of the Hermitean conjugation map, then we call it a *unitary vector space*. For any unitary vector space H, one can apply the Hermitean conjugation map to its set of orthonormal basis vectors to get $\{\langle b_i|\}$, an orthonormal basis set for the dual space H^*.

In quantum algorithms, we will most often use the computational $\{0,1\}$ basis for a Hilbert space. There are however, other interesting bases that appear from time to time. One example is the $\{|+\rangle, |-\rangle\}$ basis where $|+\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$ and $|-\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$. It is not difficult to verify that the $\{|+\rangle, |-\rangle\}$ basis is indeed orthonormal.

Any such finite-dimensional unitary vector space H is thus a Hilbert space. This is evident from the finite norm expressions which are in turn related to the inner product operations that we

saw defined by the dual maps.

2.2 Quantum States that Encode Information

2.2.1 Single qubit states

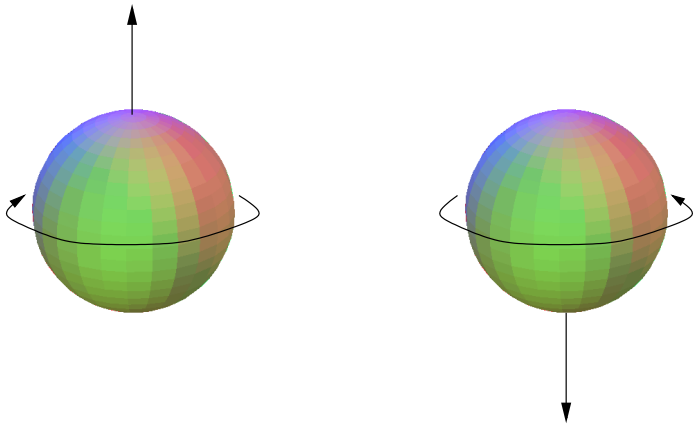
In order to compute using quantum states, all possible computational states must be represented by some quantum physical system. Furthermore, this quantum physical system must be capable of representing multiple modes of information. For example, a light switch can be in the ‘on’ or ‘off’ position, representing the logical 0 and 1 bits respectively. Since we wish to compute using a quantum system, we will consider a quantum bit, or *qubit* as any two-level quantum physical system with some *quantum property*.

While many useful quantum properties can be found in the numerous tiny particles of nature, we will limit ourselves to the spin of an atom. Another frequently used quantum property is the polarization of light particles (photons), where one can use horizontal and vertical polarization modes to represent information as well.

Consider an atom that can spin left or right.[2] Each of these spins induce either an upward or downward pointing pole as illustrated in figure 2.1.

The direction of the spin can be manipulated by radio waves as is done in nuclear magnetic resonance quantum computing. These are only two of many proposals for quantum information

[2]This is with respect to a chosen basis. In another basis, the atom may appear to spin upwards or downwards, inducing either a left or right pole respectively.



(a) (b)

Figure 2.1: A particle spinning in a leftward direction (a) induces an upward pointing pole, while a particle spinning in a rightward direction (b), induces a downward pointing pole. This correspondence is summarized by the popular “left-hand” rule often taught in high school level Physics courses.

processing. For a comprehensive summary, the reader is directed to the book by Nielsen and Chuang [23]. We label an upward pointing pole of an atom’s spin as the computational state 1 and the downward pointing pole as the computational state 0.

In the quantum mechanical framework, we can describe a one-particle system as living in a two-dimensional Hilbert space $\mathcal{H}$. Accordingly, the computational state 0 represented by the quantum spin down, is the column vector $(0\ 1)^T$. Similarly, the computational state 1 represented by spin up, is the column vector $(1\ 0)^T$:

$$|0\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

$$|1\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

According to physical experiments, any quantum state may exist in a superposition of basis states. For example, there exists a state $|\psi\rangle$ in nature which is an equally weighted superposition of $|0\rangle$ and $|1\rangle$. The superposition need not be an equally weighted one, and in its most general form, can be written as:

$$|\psi\rangle \quad = \quad \alpha|0\rangle + \beta|1\rangle$$

where $\alpha, \beta \in \mathbb{C}$ are the complex *amplitudes* of $|0\rangle$ and $|1\rangle$ respectively. Note that the quantum mechanical framework allows for the description of such a state $|\psi\rangle$ as being in a two-dimensional Hilbert space spanned by $|0\rangle$ and $|1\rangle$. In particular, $|\psi\rangle$ is a linear combination of the two basis vectors $|0\rangle$ and $|1\rangle$, with complex coefficients α and β respectively.

For the purposes of consistency, quantum states are required to have norm 1, i.e., they are required to be unit length vectors. As such, an additional constraint is placed on α and β, namely, $|\alpha|^2 + |\beta|^2 = 1$, which is derived from the expression of the norm: $\| |\psi\rangle \| = \sqrt{\langle\psi|\psi\rangle}$.

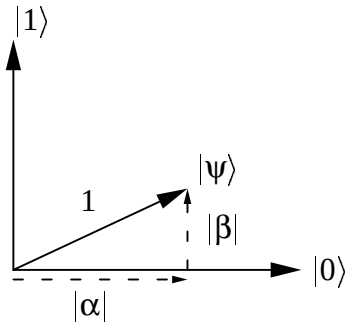


Figure 2.2: $|\psi\rangle$ is a unit length vector, and so the components of $|0\rangle$ and $|1\rangle$ obey Pythagoras' theorem which gives $|\alpha|^2 + |\beta|^2 = 1$.

2.2.2 Multiple qubit states

One can mathematically consider the joint system of many single qubit states by taking their *tensor product*.[3] Consider two Hilbert spaces $\mathcal{H}_A$ and $\mathcal{H}_B$, both of dimension 2, whose joint space is expressed as their tensor product: $\mathcal{H}_A \otimes \mathcal{H}_B$ with dimension four. More generally, the joint system of two such spaces of dimension *m* and *n* respectively, has dimension *mn*.

For two single qubit states $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ and $|\varphi\rangle = \gamma|0\rangle + \delta|1\rangle$, the joint system $|\psi\rangle \otimes |\varphi\rangle$ is :

$$|\psi\rangle \otimes |\varphi\rangle \quad = \quad \alpha\gamma(|0\rangle \otimes |0\rangle) + \alpha\delta(|0\rangle \otimes |1\rangle) + \beta\gamma(|1\rangle \otimes |0\rangle) + \beta\delta(|1\rangle \otimes |1\rangle) \qquad (2.2)$$

Often, the tensor product operator symbol is omitted where obviously implied. For example, the expression $|0\rangle \otimes |1\rangle$ may be written as $|0\rangle|1\rangle$ or $|01\rangle$ or $|0,1\rangle$. Note that the joint system is a superposition (or linear combination) of the four basis states of the joint Hilbert space. In this way, one can construct numerous single qubit systems and "concatenate" them to produce

[3] A summary of the tensor product formalism is found in the book by Nielsen and Chuang [23]

superpositions of arbitrarily large (but finite) Hilbert spaces.

2.2.3 Multiple qubit states in large Hilbert spaces

Extending the idea of such "concatenating", consider the joint system of n copies of $|\psi\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$:

$$|\psi\rangle^{\otimes^n} = \sum_{i=0}^{2^n-1} \frac{1}{\sqrt{2^n}} |i\rangle$$

where i is expressed in binary. Notice that this state is in an equally weighted (uniform) superposition of all possible binary strings from 0 to $2^n - 1$. Thus, concatenating n copies of $|\psi\rangle$ gives a desirable state which is frequently used in many quantum algorithms. Informally, the time required to create this state is $O(n) = O(\log 2^n)$ since there are n qubits, each of which are prepared in an identical fashion. Preparing any single qubit in this state requires $O(1)$ time. For example, this preparation process involves rotating the orientation of a particle appropriately which can be done by a single radio wave in the case of NMR quantum devices.

Equivalently, the joint space is a 2^n-dimensional Hilbert space, and $|\psi\rangle^{\otimes^n}$ is a single state in this space, expressed as a linear combination of the basis vectors $\{0,1\}^n$.

2.2.4 Entangled states

The multiple qubit states in the preceding section are known as *separable states* in that they can be written in a separably addressable manner by way of the tensor product notation. For example, the state in equation (2.2) can be written as $|\psi\rangle \otimes |\varphi\rangle$. Also, if a state such as the right-hand-side

of equation (2.2) can be "factored" into the form $|\psi\rangle \otimes |\varphi\rangle$, then it is called a separable state.

Multiple qubit states that cannot be expressed in such a separable way are known as *entangled states* since they cannot be "factored" into a tensor product of two or more single qubit states.

Entanglement is arguably one of the most important quantum properties offered by nature. To create entanglement between arbitrarily large quantum systems is not a trivial task. Consequently, quantum entanglement has often been treated as a resource, much in the same way that randomness has been treated as a resource in the study of randomized algorithms and complexity theory.

While entanglement is an important concept, we only give a brief notion of it here, and point the reader to the book by Nielsen and Chuang [23] for more details.

As mentioned, an entangled state of at least two qubits cannot be written in a separable manner by way of the tensor product notation. The most popular example of a two qubit entangled state is $|\Psi^+\rangle = \frac{1}{\sqrt{2}}(|0\rangle|1\rangle + |1\rangle|0\rangle)$ which is known as a Bell state. In fact, there is a Bell basis for a four-dimensional Hilbert space that can be used for the purposes of single-qubit teleportation [23].

Clearly, $|\Psi^+\rangle$ cannot be written as the tensor product of two single-qubit states, and thus, $|\Psi^+\rangle$ is considered to be an *entangled state*.

2.2.5 Other relevant topics

The story of quantum states does not end here. While there are numerous other interesting mathematical abstractions of physical phenomena, we make little use of them here, and so, in the interest of being concise, we limit ourselves to describing mixed states and the density operator, and direct the reader once again to the book by Nielsen and Chuang ([23]) for other topics.

So far, we have discussed states whose decomposition into basis states were known exactly. Sometimes, a state $|\psi\rangle$ might be in a probabilistic mixture of different such decompositions. When a state's decomposition is known with certainty, it is referred to as a *pure state*, while a state in a probabilistic mixture of pure states is referred to as a *mixed state*.

To deal with mixed states, we use the *density operator* formalism.

Definition 5. *Assume $|\psi\rangle$ is a mixed state such that with probability p_i, it is in the pure state $|\varphi_i\rangle$. Then the density operator for $|\psi\rangle$, denoted ρ_ψ is:*

$$\rho_\psi \quad = \quad \sum_i p_i \, |\varphi_i\rangle \langle\varphi_i|$$

where $|\varphi_i\rangle \langle\varphi_i|$ is a projector onto the pure state $|\varphi_i\rangle$.

Note that a density operator can also be constructed for a pure state. Consider some state $|\phi\rangle = |\varphi_j\rangle$. Since we know with certainty that $|\phi\rangle$ is in fact $|\varphi_j\rangle$, it follows that $\rho_\phi = |\varphi_j\rangle \langle\varphi_j|$.

Similarly, for two (possibly mixed) states with density operators ρ_ψ and ρ_{φ_k}, the density operator of the joint system is simply $\rho_\psi \otimes \rho_{\varphi_k}$. It is more common in the quantum mechanical literature to treat states as density operators, as they encompass both mixed and pure states.

2.3 Operations on Quantum States

In this section, we describe possible operations that one can apply to quantum states, and we do so by continuing to describe them in the quantum mechanical framework.

2.3.1 Quantum evolution and unitary operations

The driving force behind quantum evolution is the time evolution operator which is the solution to the differential equation:

$$i\hbar\frac{d}{dt}U(t)H(t) \quad = \quad U(t)H(t)$$

where $U(t_0) = \mathbb{I}$, the identity operator. A simplification that applies in our case occurs when the operator $H(t)$ (called the Hamiltonian), is time independent, i.e., $\frac{d}{dt}H(t) = 0$. In this case, $U(t)$ has a simple solution, namely:

$$\begin{aligned} U(t) &= e^{\frac{1}{i\hbar}\int_{t_0}^{t} H dt} \\ &= e^{\frac{1}{i\hbar}H(t-t_0)} \end{aligned}$$

The time evolution operator gives a general structure to most operations applicable to quantum states.

Frequently, one will notice in the literature that all possible quantum operations are limited to *linear maps*. While the quantum mechanical framework allows for non-linear maps, physical observations do not. In other words, non-linear maps predict physical outcomes that are inconsistent with experiments. As an interesting aside, some researchers have shown that if quantum

physics allows for non-linear maps, then under certain conditions, such maps can be used to prove that **NP** $\subseteq$ **BQP** or that #**P** $\subseteq$ **BQP** [1].

Consider a finite dimensional Hilbert space $\mathcal{H}$. Any linear map $L : \mathcal{H} \to \mathcal{H}$ can be expressed using the orthonormal basis vectors $\{b_i\}$ as:

$$L = \sum_{i,j} L_{i,j} |b_i\rangle \langle b_j|$$

where $L_{i,j} = \langle b_i| L |b_j\rangle$. When L is expressed in matrix form, $L_{i,j}$ is its $(i,j)^{\text{th}}$ entry. The action of L on any vector $|\psi\rangle \in \mathcal{H}$ is written as:

$$L|\psi\rangle = \sum_{i,j} L_{i,j} |b_i\rangle \langle b_j|\psi\rangle$$

Since $|\psi\rangle$ can also be expressed in this orthonormal basis, the expression simplifies to some new linear combination of basis vectors, which represents the action of L on $|\psi\rangle$.

Note that one can also consider the Hermitean conjugate, or *adjoint* of L, denoted $L^\dagger : \mathcal{H}^* \to \mathcal{H}^*$ as:

$$\begin{aligned} L^\dagger &= \left(\sum_{i,j} L_{i,j} |b_i\rangle \langle b_j| \right)^\dagger \\ &= \sum_{i,j} \left(L_{i,j} |b_i\rangle \langle b_j| \right)^\dagger \\ &= \sum_{i,j} L^*_{i,j} |b_j\rangle \langle b_i| \end{aligned}$$

When represented in matrix form, $L^\dagger$ is the complex conjugate transpose of L. Having defined the Hermitean conjugate for operators, we can now define a *unitary operator*:

Definition 6. *A unitary operator is any operator U that satisfies $U^\dagger U = \mathbb{I} = UU^\dagger$.*

Recall the time evolution operator $U(t)$, which can now be verified as a unitary operator, i.e., $U^\dagger(t)U(t) = \mathbb{I}$. Thus any operator that evolves a quantum state from one point in time to another must be unitary. In particular, operations on qubits will also be unitary operations. While it is still too early to introduce these operations, we direct the interested reader to Nielsen and Chuang [23], chapter 4.

A useful feature of unitary operators is that they are reversible in the sense that if one applies a unitary $U(t)$ to some state $|\psi\rangle$, then the action can be undone by applying the inverse of $U(t)$ to $U(t)|\psi\rangle$ which results in $U^{-1}(t)U(t)|\psi\rangle = |\psi\rangle$.

It should be noted that unitary evolution implies a closed system. Intuitively, this means that while applying unitary transformations, we gain no information from the underlying system. It can be shown that gaining any information corresponds to a transformation which does not preserve norm, and so, any norm-preserving transformation necessarily gives no information about the underlying system. Closed system evolution follows from the fact that unitary transformations are norm-preserving and that the time evolution operator is unitary.

Another frequently encountered basis for a linear operator L is the *eigenbasis* in which the basis vectors $\{|l_i\rangle\}$ are orthonormal eigenvectors of L. The *Spectral Theorem* [23] states that for any unitary operator, there exists at least one such basis in which an operator L can be expressed as:

$$L = \sum_i \lambda_i |l_i\rangle \langle l_j|$$

where $\{|l_i\rangle\}$ are orthonormal eigenvectors of L, and $\{\lambda_i\}$ are their corresponding eigenvalues. A matrix representation of L in this basis will be a diagonal matrix with the eigenvalues along the main diagonal. Frequently, the eigenbasis of an operator is a convenient one for analyzing quantum algorithms.

In most cases, linear maps on multiple qubit systems can be seen as the tensor product of linear maps on single qubit systems. As in the previous section, consider the joint state $|\psi\rangle \otimes |\varphi\rangle$, and an operator L_ψ on $|\psi\rangle$ and an operator L_φ on $|\varphi\rangle$. Then the operator on the joint system is simply $L_\psi \otimes L_\varphi$ and each operator acts solely on its part of the joint system.

The exceptions to this rule are multiple qubit entangling operations. For example, consider the quantum operation U that creates the entangled Bell state: $|\Psi^+\rangle = \frac{1}{\sqrt{2}}(|0\rangle |1\rangle + |1\rangle |0\rangle)$. If the process of creating such a state starts with two unentangled qubits, then the operator U *creates entanglement*. Creation of this state can require a controlled-operation. See Nielsen and Chuang [23] for a review on controlled-operations.

Entangling operators, as they are often referred to, are integral to quantum computers as they induce quantum interaction between two qubits [23]. More importantly, such operations along with single qubit operations have been shown to be universal, in the same sense as a universal set of logic gates for classical computation. That is, a universal set of quantum operations which are capable of simulating any unitary quantum operation can be constructed by using single-qubit and two-qubit entangling operators.

2.4 Measurements on Quantum Systems

For our purposes, we will consider a quantum measurement as an operator M that gives information about an underlying physical system in a particular basis. Inherent to this fact is that a measurement operator is basis-dependent, and so one can require the measurement to be carried out in a chosen basis. Common examples include measurements in the Bell basis, measurements in the $\{|+\rangle, |-\rangle\}$ basis, or in the $\{0, 1\}$ computational basis.

Consider the state $|\psi\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$. To "measure" this state in the computational basis means that we will observe $|1\rangle$ with probability $|1/\sqrt{2}|^2 = 1/2$, and therefore $|0\rangle$ with probability $|1/\sqrt{2}|^2 = 1/2$ also.

A measurement of $|\psi\rangle$ in the $\{|+\rangle, |-\rangle\}$ basis will yield $|\psi\rangle = |+\rangle$ with certainty, and so, the probability of observing $|-\rangle$ in this measurement is 0. Thus measurements are highly basis-dependent.

More generally, given a state $|\psi\rangle = \sum_i \alpha_i |i\rangle$, a measurement operator in the appropriate basis will yield any $|i\rangle$ with probability $|\alpha_i|^2$.

This description of quantum measurements, while not comprehensive, is sufficient for our needs. An excellent treatment is found in the book by Nielsen and Chuang [23].

Chapter 3

Quantum Mechanical Models of Computing

3.1 The Universal Quantum Turing Machine

Manipulating information quantum mechanically reveals an apparent violation of the strong Church-Turing thesis which states that *any "reasonable" model of computation can be efficiently simulated on a probabilistic Turing Machine*. Certain computational problems have been brought to light which can be solved more efficiently on a quantum device than *seems possible* on any (probabilistic) Turing machine. This was illustrated by David Deutsch in 1985 [10].

The addition of the probabilistic clause to the Church-Turing thesis was a modification to deal with the problems presented by randomized algorithmic processes. Such processes could seemingly solve some problems more efficiently than a *deterministic* Turing machine, as it was specified originally by Alan Turing. Does such a quantum device signal another required change to the strong Church-Turing thesis?

According to Paul Benioff in 1980 [3], the answer to this question was a resounding yes. His proposal was an abstract computing device, much like the Turing Machine, but which was based on the fundamental laws of nature, which to date, are quantum as far as physicists can tell. He therefore proposed the quantum Turing machine which incorporated the quantum physical features of nature known to date. This concept was later refined by Deutsch in [10] where he defined the notion of a "universal quantum computer". A comprehensive treatment of quantum Turing machines is found in the paper by Bernstein and Vazirani [5].

Since it seems that all physical objects are fundamentally quantum at some level, it follows that any classical deterministic or randomized algorithmic procedure can be simulated efficiently by a universal quantum machine that is "aware" of these quantum properties. In the realm of quantum algorithms, the universal quantum Turing machine is the fundamental computational device, much the same as the universal Turing machine is central to classical computation.

3.2 The Quantum Circuit Model

Using a quantum Turing machine to specify an algorithmic process would be too cumbersome, even more so than specifying classical algorithmic processes on a (probabilistic) Turing machine. Instead, we use the *quantum circuit model* which was first proposed by Deutsch in 1989 [11], and later refined by Yao [25]. In his paper, Yao also showed the equivalence between the quantum circuit model and the quantum Turing machine model. Roughly speaking, the quantum circuit model is the quantum analog of the classical circuit model which is equivalent to the Turing machine model.

In the quantum circuit model, logic gates are replaced with reversible quantum logic gates, and bits are replaced with qubits. Reversible quantum logic gates, or simply quantum gates, are the unitary quantum operations that were described previously. Most (if not all) quantum algorithms are stated in the quantum circuit model. In fact, it has been so successful, that physicists having nothing to do with quantum information processing have used it to describe quantum processes as well.

The major difference between the classical and quantum circuit models (apart from one being quantum) is the absence of a fanout gate. This is a result of the famous *no cloning theorem* which informally states that there exists no quantum circuit for cloning an unknown state $|\psi\rangle$.

Classically, a bit in a circuit may be cloned simply by knowing its value and then subsequently applying a logical not gate to a target register (if necessary). For example, we can clone a 1 bit by preparing a target register with a 0 bit and then applying a not gate. Quantum mechanically, however, a qubit may be in an arbitrary superposition $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$, and thus looking at it, or measuring it cannot be done without irreparably destroying the superposition. Nielsen and Chuang [23] show that no unitary quantum operation can clone an *unknown* quantum state. Non-unitary operations do not help much, either, as they introduce information loss in the "copied" state.

A comprehensive summary of the no cloning theorem as well as quantum circuits and elementary quantum gates can be found in the book by Nielsen and Chuang [23].

Chapter 4

Putting it all together: The First Quantum Algorithms

4.1 Deutsch's Algorithm

The following discussion, for the most part, is adapted from Cleve et al. [9]. We explore the details of Deutsch's algorithm in-depth, but we also direct the reader to the original papers of Deutsch [10] and Deutsch and Jozsa [12] for those interested. For basic gates such as the Hadamard gate, refer to chapter 4 of Nielsen and Chuang [23].

Consider a Boolean function $f : \{0,1\} \to \{0,1\}$. Clearly, there are only four such functions:

- Constant functions: $f(1) = f(0) = 0$ and $f(1) = f(0) = 1$
- Balanced functions: $f(1) = 1\,,\, f(0) = 0$ and $f(1) = 0\,,\, f(0) = 1$

Here, balanced means exactly one half of all outputs of f are 0 and exactly one half of all outputs are 1.

The problem Deutsch considered was the following: Given a function f which is either balanced or constant, decide which is the case with as few evaluations of f as possible.

Classically, one must query the function on both inputs $\{0,1\}$ to know with certainty which case f falls in. Quantum mechanically, however, one query suffices to answer the question with certainty. The quantum algorithm solving this problem was proposed by David Deutsch [10] and was the first example of the benefits of quantum parallelism. The following is an adaptation of Deutsch's algorithm:

1: Prepare the state:

$$\frac{1}{\sqrt{2}}|0\rangle\left(|0\rangle-|1\rangle\right)$$

2: Apply the Hadamard gate (H) to the first qubit register:

$$\frac{1}{2}\left(|0\rangle+|1\rangle\right)\left(|0\rangle-|1\rangle\right)$$

3: Using the first register as the input to f, compute the function in the second register using the unitary operator U_f:

$$\frac{1}{2}(-1)^{f(0)}\left(|0\rangle+(-1)^{f(0)\oplus f(1)}|1\rangle\right)\left(|0\rangle-|1\rangle\right)$$

4: Apply the Hadamard gate to the first qubit register to observe either a $|1\rangle$ or a $|0\rangle$

5: If $|1\rangle$ is observed, then f is balanced, else f is constant.

In step one of the algorithm, $|0\rangle-|1\rangle$ is prepared by applying the Hadamard gate to $|1\rangle$. Step three now requires some clarification, for which we establish this fact: the evaluation of f by applying U_f to the state $|x\rangle\left(|0\rangle-|1\rangle\right)$ yields:

$$U_f |x\rangle (|0\rangle - |1\rangle) \rightarrow |x\rangle (|0 \oplus f(x)\rangle - |1 \oplus f(x)\rangle) \quad (4.1)$$

$$= (-1)^{f(x)} |x\rangle (|0\rangle - |1\rangle) \quad (4.2)$$

where $i \oplus f(x)$ is the bitwise addition (modulo 2) of i and $f(x)$. To see this, consider the possible values of $f(x) \in \{0,1\}$ and how $|0 \oplus f(x)\rangle - |1 \oplus f(x)\rangle$ changes with the possible substitutions.

Note as well that in the algorithm, $|x\rangle$ is a superposition of $|0\rangle$ and $|1\rangle$. In this sense, applying f to such a superposition evaluates f on *both* input values. Therefore, we have the following state (ignoring normalization factors):

$$\left((-1)^{f(0)} |0\rangle + (-1)^{f(1)} |1\rangle\right) (|0\rangle - |1\rangle)$$

If a factor of $(-1)^{f(0)}$ is divided out, then we are left with:

$$(-1)^{f(0)} \left(|0\rangle + (-1)^{f(0) \oplus f(1)} |1\rangle\right) (|0\rangle - |1\rangle)$$

Since $(-1)^{f(0)}$ is a global phase i.e., a phase that applies to the entire state, it can be ignored since it cannot be physically detected. To see this, consider a state $|\psi\rangle$ and a state $|\psi'\rangle = e^{\gamma} |\psi\rangle$, where e^{γ} is a global phase of $|\psi'\rangle$. It is not difficult to verify that the inner-product of $|\psi\rangle$ with itself, i.e., $\langle\psi|\psi\rangle$ is exactly the same as $\langle\psi'|\psi'\rangle$. Intuitively, since measurements involve an inner-product expression, it follows that the global phase will disappear, and therefore a measurement will not be able to distinguish between $|\psi\rangle$ and $|\psi'\rangle$.

Consider now the various possibilities for f: if it is balanced, then $f(0) \oplus f(1) = 1$ while if f is constant, $f(0) \oplus f(1) = 0$. If f is balanced, the $|1\rangle$ in the first register acquires a relative phase of -1, while if f is constant, $|1\rangle$ acquires no phase at all. Relative phases are physically

detectable [23], and so in principle, we can tell the difference between balanced and constant functions.

In the case of balanced functions, we have the state $|0\rangle - |1\rangle$ which under the action of the Hadamard gate is mapped to $|1\rangle$, while in the case of constant functions, we have the state $|0\rangle + |1\rangle$ which under the Hadamard gate is mapped to $|0\rangle$. Upon applying the Hadamard gate, as in step four of the algorithm, we can measure to know know if f is balanced or constant with only a single f evaluation.

At the heart of this algorithm is the constructive versus destructive interference in the phases $(-1)^{f(0)\oplus f(1)}$. A constant function f will cause destructive interference, while a balanced function will cause constructive interference. Thus using the principle of quantum superposition, we are able to quantum mechanically solve this balanced/constant decision problem by observing constructive or destructive interference patterns in the phases of these quantum states.

With his algorithm, David Deutsch found a reduction from this particular decision problem to a physical experiment whereby nature handles the information more effectively. In fact, this is the spirit of quantum information processing: we find ways to reduce information problems into physical ones where nature, or the laws of quantum physics, seem to offer superior manipulation capabilities.

4.2 The Deutsch-Jozsa Algorithm

Some may notice that the advantages of Deutsch's algorithm over classical methods are fairly slight. To amplify its power, we briefly present the Deutsch-Jozsa algorithm which clearly shows an exponential reduction in the number of function evaluations. Consider a function $f : \{0,1\}^n \to \{0,1\}$ which is again promised to either be balanced or constant. Once again, we are required to determine which category f belongs to with as few f-evaluations as possible.

By a simple adversary argument, it is clear that any classical deterministic or randomized method must make at least $N/2+1$, (where $N = 2^n$) function evaluations to know the answer with certainty. Using Deutsch's algorithm with larger registers, one can quantum mechanically determine the answer with a single evaluation.

Instead of the first register being a single qubit register, we start with a tensor product of n qubits, each in the state $|0\rangle$. To these registers, we apply the n-qubit Hadamard gate, or simply just a one qubit Hadamard gate to each of the n qubits. We then have the following state in step 2 of the new algorithm:

$$\frac{1}{\sqrt{2N}} \sum_{x \in \{0,1\}^n} |x\rangle \left(|0\rangle - |1\rangle\right)$$

The action of the n-qubit Hadamard gate is:

$$H^{\otimes^n} |x\rangle = \sum_{y \in \{0,1\}^n} (-1)^{x \cdot y} |y\rangle$$

When we now evaluate f, we do so for each $x \in \{0, \ldots, N-1\}$. Consequently, after applying the Hadamard gate again (as in step 4), we are left with the state:

$$\frac{1}{\sqrt{2}N}\sum_{x,y\in\{0,1\}^n}(-1)^{f(x)\oplus(x\cdot y)}|y\rangle\left(|0\rangle-|1\rangle\right)$$

Consider the amplitude of $|00\cdots0\rangle = \sum_{x\in\{0,1\}^n}\frac{(-1)^{f(x)}}{N}$. If f is balanced, then exactly one half of the values of $f(x)$ will be 0 while the other half will be 1. Clearly the amplitude of $|00\cdots0\rangle$ will be 0 in this case. If the function is constant, then all values of $f(x)$ are the same, and the amplitude of $|00\cdots0\rangle$ is exactly $(-1)^{f(00\cdots0)}$.

The amplitude expression for any state other than $|00\cdots0\rangle$ will be $\sum_{x,y\in\{0,1\}^n}\frac{(-1)f(x)\oplus(x\cdot y)}{N}$.

Definition 7. *The Hamming distance of two n-bit strings x and y is defined as the integer $h(x,y)$ which is exactly the number of bit locations in which x and y differ.*

Note that $h(x,y) \mod 2$ is exactly $x\cdot y \mod 2$. For any fixed x, the quantity $x\cdot y$ partitions the set $\{y \mid y\in\{0,1\}^n\}$ into two equally sized sets Y_0 and Y_1 where any string y in Y_0 has $x\cdot y = 0 \mod 2$ and any string y in Y_1 has $x\cdot y = 1 \mod 2$.

For any constant function f and non-zero state $|x\rangle$, there will be an equal number of $-1/N$ and $1/N$ terms in the relative phase, implying that the amplitude of such a state $|x\rangle$ is identically 0. Thus for a constant function f, the only remaining state is $|00\cdots0\rangle$. Once again, destructive interference in the case of constant f leads to this consequence. In the case of balanced f, destructive interference leads to the impossibility of observing $|00\cdots0\rangle$.

As the last step of the Deutsch-Jozsa algorithm, we measure the first n qubits and if the outcome is the state $|00\cdots0\rangle$, we conclude that the function was constant. If the outcome of

the measurement is any state but $|00\cdots0\rangle$, we conclude that the function was balanced. By this algorithm, we can answer the constant/balanced decision problem for functions f of larger domains – *with one evaluation of* f. This is a clear example of the exponential speedup offered by quantum mechanics.

Chapter 5

Quantum Searching and Quantum Counting

5.1 Quantum Searching

The quantum searching algorithm is one of the most exciting developments in quantum algorithms. Its repeated use as a subroutine has led to improved quantum and hybrid quantum-classical algorithms. The algorithm is set in the very generic language of searching a database which is a problem to which many interesting combinatorial problems can be reduced. Intuitively, there is a database with some sought-after, or *marked* elements, and this quantum algorithm finds one of these elements by quantum *amplitude amplification*.

Consider the popular 3-CNF-SAT problem, which is **NP**-complete. Given a formula F in conjunctive normal form, we must find a set of satisfying variable assignments under which F evaluates to true. Finding such an assignment can be seen as a database search problem, where the database elements are the possible variable assignments, and marked elements are assign-

ments under which F evaluates to true.

Most problems can be seen as database search problems in the absence of any structure. Furthermore, the best algorithms to date that attempt to solve such problems involve heuristically searching through some combinatorial object space. In this sense, tackling the abstract database search problem also implicitly deals with many interesting combinatorial search problems. Here, we review quantum amplitude amplification and then apply it to the searching problem.

5.1.1 Quantum amplitude amplification

Consider a prepared quantum state $|\Psi\rangle = \alpha|\Psi_1\rangle + \beta|\Psi_0\rangle$, where we are interested in observing $|\Psi_1\rangle$ as a result of our measurement, which happens with probability $|\alpha|^2$. If α is small, then we will seldom see $|\Psi_1\rangle$ in such a measurement process. Note that $|\Psi_1\rangle$ and $|\Psi_0\rangle$ are orthogonal states, and $|\alpha|^2 + |\beta|^2 = 1$.

Quantum amplitude amplification is a process whereby we boost the amplitude of $|\Psi_1\rangle$ to α' such that $\alpha' \gg \alpha$, thereby increasing the likelihood that $|\Psi_1\rangle$ will be the result of our measurement.

We assume the existence of a unitary operator $\mathbf{A}$ which, when applied to $|0\rangle$, creates the state $|\Psi\rangle = \alpha|\Psi_1\rangle + \beta|\Psi_0\rangle$. Essentially, since $|\Psi\rangle$ lives in a two dimensional Hilbert space, spanned by $|\Psi_1\rangle$ and $|\Psi_0\rangle$, we can apply a series of reflections to rotate $|\Psi\rangle$ to the state $|\Psi'\rangle$ such that the amplitude corresponding to $|\Psi_1\rangle$ is large in $|\Psi'\rangle$. These reflections are implemented by the Grover iterate:

$$\mathbf{G} \;=\; \left(\mathbf{A}\mathbf{S_0}\mathbf{A}^{-1}\right)\mathbf{S}_\chi \tag{5.1}$$

The operator $\mathbf{S}_\chi$ flips the amplitude of the $|\Psi_1\rangle$ portion of $|\Psi\rangle$, thereby reflecting the two-dimensional state vector $|\Psi\rangle$ about $|\Psi_0\rangle$. The reflected vector is labeled $|\Psi_a\rangle$ in figure 5.1.1.

The operator $\mathbf{S_0}$ flips the amplitude of the $|\Psi_0\rangle$ portion of $|\Psi\rangle$. When conjugated by $\mathbf{A}$ and $\mathbf{A}^{-1}$, the series of operators $\mathbf{A}\mathbf{S_0}\mathbf{A}^{-1}$ performs a reflection about the state $\mathbf{A}|0\rangle = |\Psi\rangle$. Thus these two operators perform reflections which are equivalent to a rotation of $|\Psi\rangle$ by angle 2θ, where θ was the original angle between the state vector $|\Psi\rangle$ and $|\Psi_0\rangle$. The second reflection about $|\Psi\rangle$ is represented by the vector $|\Psi_b\rangle$ in figure 5.1.1.

Intuitively, the conjugation by $\mathbf{A}$ and $\mathbf{A}^{-1}$ corresponds to a change of basis into the reference frame of $|\Psi\rangle$. In this new basis, $|\Psi\rangle$ is the new $|\Psi_0\rangle$, and $\mathbf{S_0}$ reflects the vector $|\Psi_a\rangle$ about $|\Psi\rangle$. The application of $\mathbf{A}$ changes the basis back to the original $\{|\Psi_0\rangle, |\Psi_1\rangle\}$ basis. Figure 5.1.1 may help in visualizing these rotations.

The operator $\mathbf{G}$ is applied m times to rotate the state vector $|\Psi\rangle$ towards $|\Psi_1\rangle$. Each application of $\mathbf{G}$ rotates $|\Psi\rangle$ by an angle of 2θ, and thus, after m applications, the amplitude of $|\Psi_1\rangle$ is $\sin((2m+1)\theta)$. Note that the probability of observing $|\Psi_1\rangle$ is $\sin^2((2m+1)\theta)$. Now, the correct choice of m depends directly on the value of θ, and in the absence of such knowledge, one can use existing techniques to estimate or guess this angle to determine the correct value of m [7]. Overall, an appropriate value of m will ensure that $\sin^2((2m+1)\theta)$ is close to 1, implying that with high probability, we will observe $|\Psi_1\rangle$.

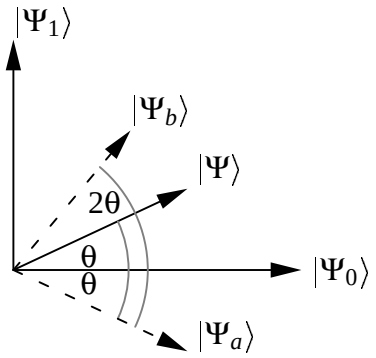


Figure 5.1: $|\Psi\rangle$ in a two-dimensional Hilbert space is rotated by an angle of 2θ with every application of **G**.

Brassard et al. [7] show that to amplify this amplitude sufficiently high, we require $\Theta\left(1/\sqrt{\sin^2(\theta)}\right)$ applications of **G**.

5.1.2 Quantum searching by amplitude amplification

Formally, the database search problem is as follows:

Definition 8 (taken from [6]).

- *Let* $X_N = \{0,\ldots,N-1\}$ *for some integer N*
- *Consider an arbitrary function* $\chi : X_N \to \{0,1\}$

The database search problem requires one to find an $i \in X_N$ *such that* $\chi(i) = 1$*, provided that such an i exists.*

Usually, χ is given as a black-box oracle and we can query the value of χ on any $i \in X_N$. Conventionally, such a query costs one unit, and in this regard, we usually consider the query complexity of any algorithm that solves the database search problem.

Essentially, querying χ for some $i \in X_N$ corresponds to a database lookup, where χ is used to determine if such an i is *marked*. Depending on the setting, χ can be deterministic or randomized, but a randomized function requires extra care as described by Høyer et al. [16].

The database search problem can be solved either by deterministic or randomized methods. To say that the database search problem is solved by a deterministic algorithm in $O(T)$ database queries means that after $O(T)$ database queries the algorithm either finds a marked element if one exists or it returns an output indicating that none was found. In either case, the algorithm is always correct.

Solving the problem by a randomized algorithm in $O(T)$ queries can imply various conditions. If the algorithm is a Las Vegas algorithm [22], then it will always answer correctly i.e., it will always find a marked element, or it will return an output indicating that none exist. In either case, the algorithm is always correct in its output. This is similar to the deterministic algorithm except that a Las Vegas algorithm will use random bits in determining the answer. A Las Vegas algorithm is referred to as a zero-error randomized algorithm.

In the context of decision problems, a Monte Carlo algorithm can either be a one-sided or two-sided error randomized algorithm . In these algorithms, there is some (bounded) probability that it will respond with an incorrect answer. In the context of database searching, a two-sided error algorithm will have a bounded probability of incorrectly responding to both the yes or no questions i.e., when the algorithm finds a “marked element”, there is some bounded probability that the algorithm is wrong. Similarly, when the algorithm claims that no marked elements exist, there is some bounded probability that it is wrong. In a one-sided error Monte Carlo algorithm, at least one of the yes or no questions have a zero probability of error. Once again, any one of

these algorithms solving the database search problem would use $O(T)$ queries before it answers.

Quantum algorithms can be one-sided or two-sided error – just like classical randomized algorithms. In addition to this, there is the concept of exact quantum algorithms which is a quantum counterpart to classical deterministic algorithms. For more information on computing models, see the book by Nielsen and Chuang [23].

In a two-sided error quantum computing model, solving the database problem means to prepare a state register $|\psi\rangle$ and apply to it, some sequence of unitary operators U with $O(T)$ such operators making database queries. After having applied these unitary operators, measurement of the state register should collapse it to a state representing a marked element with high probability. With low probability, it collapses to a state representing a non-marked element.

Given an unstructured/unsorted database of N elements, a simple adversary argument shows that the best possible deterministic or randomized algorithm requires at least $N/2$ database queries. The quantum search algorithm solves this problem with high probability using $O(\sqrt{N})$ database queries, which is a quadratic reduction. Note that the classical lower bound does not include any heuristics that may be used i.e., it implies exhaustive search through the database. The use of a heuristic would exploit some structure in the underlying database and would avoid exhaustive searching. As it turns out, any such heuristics can also be quantized [7].

As mentioned in the outset, it is possible to solve the database search problem by amplitude amplification. We prepare a uniform superposition of all database elements (ignoring normalization constants) $|\Psi\rangle = \sum_i |i\rangle$ which can be rewritten as $\alpha|\Psi_0\rangle + \beta|\Psi_1\rangle$, where $|\Psi_0\rangle$ (respectively $|\Psi_1\rangle$) is a uniform (equally-weighted) superposition of all unmarked (respectively marked)

database elements.

The abstract function χ, supplied as a black-box, is queried by $\mathbf{S}_\chi$ in determining which elements $|\Psi\rangle$ are marked. This enables $\mathbf{S}_\chi$ to flip the amplitude of the $|\Psi_1\rangle$ portion of $|\Psi\rangle$. Thus $\mathbf{S}_\chi$ flips the amplitudes of the *marked* elements by using χ.

Using the amplitude amplification algorithm, we amplify the amplitude of the set of marked elements close to a value of 1 by *m* applications of the Grover iterate. Once the amplitude of $|\Psi_1\rangle$ has been amplified in $|\Psi\rangle$, we measure it and observe $|\Psi_1\rangle$ with high probability. More specifically, since $|\Psi_1\rangle$ was a uniform superposition of the marked elements, it follows that it too will collapse to one of the marked elements of its superposition, upon measurement.

Since measurement is a probabilistic event, it follows that with some probability, we will observe a non-marked element i.e., with some probability, we will observe $|\Psi_0\rangle$ as a result of our measurement. This is where the two-sided error property of the algorithm is exemplified. Note that if we were able to amplify the amplitude of the marked states exactly to 1, then this algorithm becomes an exact quantum algorithm, and the probability of observing $|\Psi_0\rangle$ after amplification, is exactly 0.

This is a general consideration of Lov Grover's original database search algorithm [15] in which he considered a table of elements and was looking for a particular marked element *y*. In his formalism, $\mathbf{A}$ was the Hadamard transformation and χ was the equality function. Note as well that the database searching algorithm is essentially the amplitude amplification algorithm, and often in the literature, reference is made directly to amplitude amplification rather than database searching.

5.2 Quantum Counting

5.2.1 The quantum Fourier transformation

The quantum Fourier transformation (QFT) has arguably been one the most important tools for quantum information processing. It is the cornerstone of the factoring and discrete logarithm algorithms of Shor, and it has enabled theorists to find faster algorithms in the quantum realm. For our purposes, we simply describe the QFT and show its action on a simple state. For a more in-depth discussion, please refer again to the book by Nielsen and Chuang [23].

The quantum Fourier transformation of dimension M is the unitary operator $\mathbf{QFT}_M$ which acts on a state $|a\rangle$ of $\log M$ qubits to produce:

$$\mathbf{QFT}_M |a\rangle = \frac{1}{\sqrt{M}} \sum_{y=0}^{M-1} e^{2\pi i a \cdot y} |y\rangle \tag{5.2}$$

where $a \cdot y$ is the dot product of a and y modulo 2.

The *inverse* quantum Fourier transformation of dimension M is the unitary operator $\mathbf{QFT}_M^{-1}$ which acts on a state $|a\rangle$ of $\log M$ qubits to produce:

$$\mathbf{QFT}_M^{-1} |a\rangle = \frac{1}{\sqrt{M}} \sum_{y=0}^{M-1} e^{-2\pi i a \cdot y} |y\rangle \tag{5.3}$$

Note that $\mathbf{QFT}_M^{-1} (\mathbf{QFT}_M |a\rangle) = |a\rangle$, and this can be verified explicitly.

5.2.2 Quantum phase estimation

Given a unitary operator U as a black-box, and an eigenvector (eigenstate) $|\psi\rangle$ of U with eigenvalue $e^{2\pi i\omega}$, $0 \leq \omega \leq 1$, the phase estimation problem requires one to find the value of ω. In addition to being given U, it is commonly assumed that we are also given ways of computing controlled-U^{2^j} operations, for $j \geq 0$.[1] As well, we are only given a single copy of $|\psi\rangle$. This restriction highlights the concern that $|\psi\rangle$ may be hard to produce, and that we must find ω in a reversible manner so as to reuse the single copy of $|\psi\rangle$ supplied.

We define an operator $\Lambda(U)$ which acts on a two register state as: $|j\rangle|y\rangle \rightarrow |j\rangle\left(U^j|y\rangle\right)$, for the purposes of the phase estimation algorithm which we give here:

1: Prepare $|0\rangle|\psi\rangle$

2: Apply $\mathbf{QFT}_M$ to the first register:

$$\frac{1}{\sqrt{M}}\sum_{j=0}^{M-1}|j\rangle|\psi\rangle$$

3: Apply $\Lambda(U)$ to the second register:

$$\frac{1}{\sqrt{M}}\sum_{j=0}^{M-1}|j\rangle U^j|\psi\rangle$$

4: Apply $\mathbf{QFT}_M^{-1}$ to the first register:

$$\frac{1}{M}\sum_{k=0}^{M-1}\left(\sum_{j=0}^{M-1}e^{-2\pi ij\cdot k}\right)|k\rangle|\psi\rangle$$

5: Measure the first register to get an approximation of ω.

$$|\widetilde{\omega}\rangle|\psi\rangle$$

[1] For a review of controlled operations, see Nielsen and Chuang [23], chapter 4.

Note that the amplitudes of the state in step 4 simplify because the summation over j is a geometric sum.

To intuitively see how this algorithm works, assume that we have the state $|\omega\rangle$ at our disposal, to which we apply $\mathbf{QFT}_M$ to get:

$$\mathbf{QFT}_M |\omega\rangle = \frac{1}{\sqrt{M}} \sum_{j=0}^{M-1} e^{2\pi i \omega \cdot j} |j\rangle \tag{5.4}$$

Clearly, if we apply the inverse QFT to the state in (5.4), then we retrieve $|\omega\rangle$ exactly. Note that in the phase of this state, we have multiplied ω by j. We can construct a similar phase by applying U to $|\psi\rangle$, a total of j times. However, to construct the state in (5.4), we must apply U for all $j \in [0, M-1]$, which is why we defined $\Lambda(U)$.

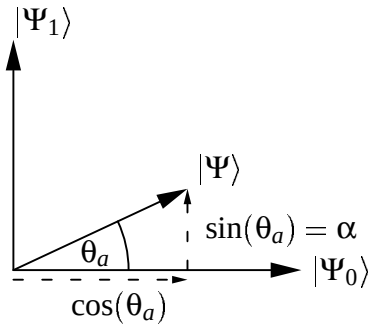
Intuitively, this is exactly what the phase estimation algorithm does: we apply $\mathbf{QFT}_M$ and $\Lambda(U)$ to build the *Fourier transformed image* of $|\omega\rangle$. Upon applying $\mathbf{QFT}_M^{-1}$, we learn the value of ω (approximately).

From the simplified amplitude expression in the phase estimation algorithm, note that if ω is exactly an integer fraction of M, then we retrieve it exactly by the inverse QFT; if not, we still get a good approximation [9, 7]. In particular, with probability at least $4/\pi^2$, we get the best m-bit estimate of ω [9], and with probability at least $8/\pi^2$, we get one of the two best m-bit estimates of ω [7].

5.2.3 Counting by phase estimation

The goal of counting by phase estimation is to estimate the quantity $a = t/N$ where t is the number of solutions to a Boolean function $\chi : \{0,\ldots,N-1\} \to \{0,1\}$, i.e. the number of elements j such that $\chi(j) = 1$. Formally, for some given unitary operator $\mathbf{A}$, $\mathbf{A}|0\rangle = |\Psi\rangle = \alpha|\Psi_1\rangle + \beta|\Psi_0\rangle$, where $|\Psi_1\rangle$ is the good component of $|\Psi\rangle$, i.e. an equally-weighted superposition of elements j such that $\chi(j) = 1$, and $|\Psi_0\rangle$ is defined similarly. Estimating the quantity a means to estimate $a = \langle\Psi_1|\alpha^*\alpha|\Psi_1\rangle = |\alpha|^2$, i.e., the probability that a measurement of $|\Psi\rangle$ results in a good state. Knowing a solves the counting problem since we also assume to know N.

There exists some angle θ_a, such that $a = \sin^2(\theta_a)$ which is true for any $a \in [0,1]$. Thus, an estimate for θ_a results in an estimate for a. We have the following picture again:



Obviously, $0 \leq a \leq 1$ due to the bounds of t, and from the sinusoidal property of α. Brassard et al. [7] have shown that the phase estimation algorithm gives an estimate $\widetilde{a}$ to a such that:

$$|\widetilde{a} - a| \quad \leq \quad 2\pi k \frac{\sqrt{a(1-a)}}{M} + k^2 \frac{\pi^2}{M^2}$$

with probability at least $8/\pi^2$ for $k = 1$ and probability at least $1 - \dfrac{1}{2(k-1)}$ for $k \geq 2$. Here, M is the dimension of the QFT applied in the phase estimation algorithm algorithm. It also controls

the quality of our estimate, and thus increasing M reduces the error in our estimate. In other words, the larger our value of M, the more accurate our estimate for a. If $M = N$, we get exact counting information in which case, there is no speedup over classical counting methods.

Counting is related to phase estimation by the Grover iterate $\mathbf{G}$ whose eigenvalues encode θ_a. By estimating the phases of the eigenstates of $\mathbf{G}$, we learn θ_a which we then relate to t. The specific Grover iterate used in the phase estimation algorithm is directly dependent on the Boolean function χ.

To find t, we estimate θ_a by the phase estimation algorithm and then compute $t = N\sin^2(\theta_a)$. Once again, if θ_a is not an integer multiple fraction of M, then we only get an approximation of it, and therefore an approximation of t. The approximation bounds for t follow from the bounds of the phase estimation algorithm [7]. To approximate t within a few standard deviations, we can set $M = \lceil\sqrt{N}\rceil$. These and more useful facts are found in the paper by Brassard et al. [7].

Chapter 6

Uniformized Phase Estimation and Monotonized Counting

6.1 Introduction

Recall from Chapter 5 that for a given unitary operator $\mathbf{U}$ and an eigenstate $|\Psi_\omega\rangle$ of $\mathbf{U}$ with eigenvalue $e^{2\pi i\omega}$, the phase estimation problem requires one to estimate the value $0 \leq \omega \leq 1$. It has been shown [9, 20] that the phase estimation algorithm outputs a estimate $\widetilde{\omega}$ that is the best m-bit approximation of ω with probability at least $4/\pi^2$, i.e., $\Pr\left(|\widetilde{\omega}-\omega| < 1/2^{m+1}\right) \geq 4/\pi^2$. With probability at least $8/\pi^2$, the algorithm outputs one of the *two* best estimates of ω, i.e., $\Pr\left(|\widetilde{\omega}-\omega| < 1/2^{m}\right) \geq 8/\pi^2$ [8, 7].

Here, $\widetilde{\omega}$ is the resulting outcome of a discrete random variable X having a distribution with much of its weight within a $1/M$ error window about ω where $M = 2^m$. It was previously shown that the distribution of the estimate is indeed dependent on ω [7, 20]. For another eigenstate $|\Psi_\gamma\rangle$ of the same unitary operator, the distribution of the discrete random variable Y associated with $\widetilde{\gamma}$

would also be γ-dependent.

If $\omega = x/M$ for some integer $0 \leq x \leq M$, the resulting distribution of X is a sharp peak at x, while, if $\gamma = \frac{x-1/2}{M}$, then the resulting distribution of Y is spread out over different values of x [7]. This is illustrated in figure 6.1.

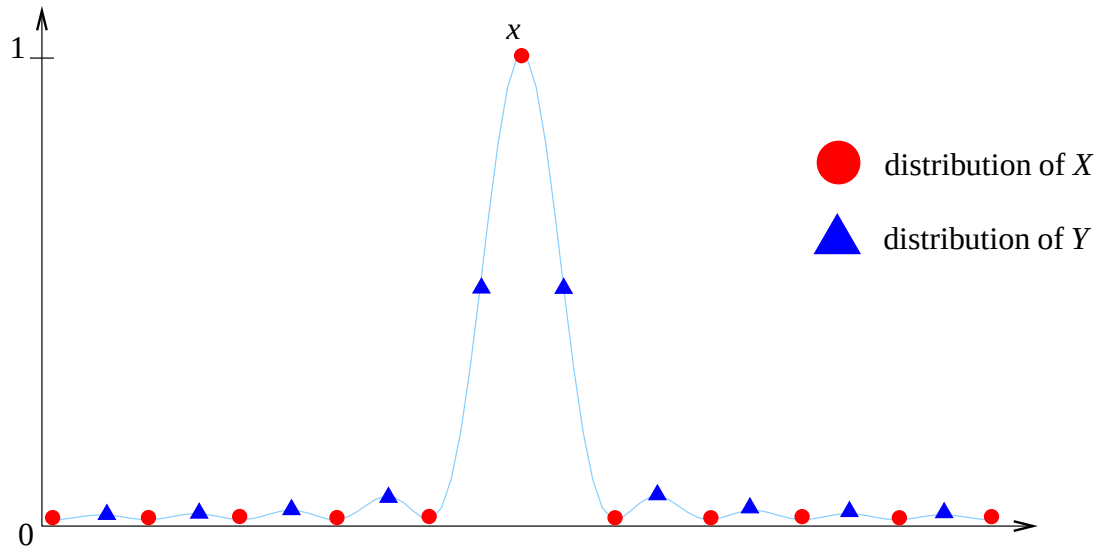


Figure 6.1: Discrete distributions for ω and γ represented by discrete random variables X and Y respectively. In this figure, they are illustrated on the same curve, while in actuality, they will centred at different locations.

If the distributions are shaped as illustrated, then the probability of observing a number which is distance j away from x is very different in the cases of X and Y. Clearly, this is a problem if we wish to compare the two estimates in certain ways. Typically, we would like the distributions of X and Y to be the same in the sense that $\Pr[X = y \mid x]$ should be a function of $x - y$, and similarly, $\Pr[Y = y \mid x]$ should be a function of $x - y$. If this is the case, and $\omega > \gamma$, then clearly,

$\Pr[X > z] > \Pr[Y > z]$ for $\forall z$. A priori however, this is not true, since the distributions of X and Y are shaped differently. Thus, in comparing the distributions of $\widetilde{\gamma}$ and $\widetilde{\omega}$, it may be the case that $\Pr[Y > z] > \Pr[X > z]$, as illustrated in figure 6.2.

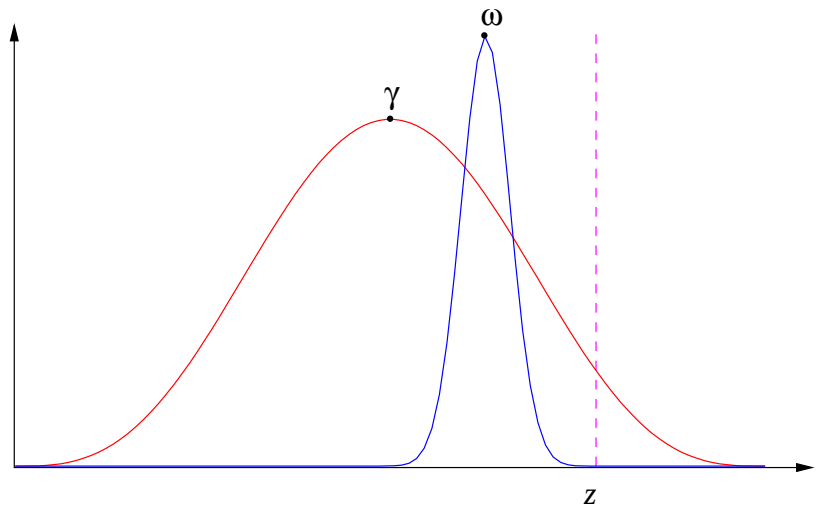


Figure 6.2: Sampling from these distributions might indicate that γ is greater than ω. The distribution curves are shown as continuous for illustration purposes.

Uniformization is a method by which we can smooth out these different distributions and make them independent of the phase parameters in the eigenvalues. Thus after uniformization, the probability of observing a number which is distance j away from x is approximately the same in both distributions. For a phase parameter such as ω, we are losing information by uniformizing its distribution, but we are gaining the ability to compare approximate quantities as described above.

The idea of uniformization was first used by Mosca and Zalka [21] to make the success

probability of the eigenvalue estimation algorithm independent of the eigenvalues. This was a necessary tool for constructing an exact quantum Fourier transform, and as such, the authors showed how the resulting probability functions were indeed independent of the phase parameters in the eigenvalues. As opposed to the general situation here, the authors knew the form of phases which allowed them to uniformize exactly. This notion will become clear shortly when we describe uniformization at an intuitive level.

Additionally, we discuss error bounds on estimates after uniformization by considering the corresponding probability density functions for uniformized distributions. We also consider the common tasks of comparing such quantities with the eventual goal of being able to compare them in a quantum way, i.e., within a quantum algorithm. Finally, as an application of this technique, we *monotonize* the quantum counting algorithm.

6.2 Uniformized Phase Estimation

6.2.1 Understanding uniformization

Consider the following four discrete random variables, and their respective probabilities defined on the set $\{1,2,3,4,5\}$:

Random Variable	1	2	3	4	5
X_1	0.1	**0.5**	0.2	0.1	0.1
X_2	0.15	0.2	**0.3**	0.2	0.15
X_3	**0.7**	0.1	0.1	0.05	0.05
X_4	0.1	0.1	0.1	0.3	**0.4**
$\sum_i \frac{\Pr(X_i=k)}{4}$	0.2625	0.225	0.175	0.1675	0.175

Clearly, the distributions of these random variables are different. A first attempt at uniformization may be to use the averaged probability distribution, i.e., $\Pr(X_i = k) = \sum_i \frac{\Pr(X_i=k)}{4}$. However, this changes the peak point of each distribution. For our purposes, we will need to keep the distribution peaked at its original value, and so we must consider the corresponding centre points s_i, which are indicated in bold above.

If we uniformize these distributions while keeping the centre points intact, then for any random variable X_i, the probability of observing a value that is distance j away from the centre s_i, is exactly this quantity averaged *over all* distributions. In the case of X_1:

$$\begin{aligned} \Pr(X_1 = 1) &= \frac{1}{4}(\Pr(X_1 = 1) + \Pr(X_2 = 2) + \Pr(X_3 = 5) + \Pr(X_4 = 4)) \\ &= 0.1625 \end{aligned}$$

Notice that for each X_i, we are incorporating the probability of measuring a value that is one unit to the left of its centre point. The process of uniformization is as follows: assuming that we have a method for sampling from any of these distributions, we randomly choose one, and sample a value from it. We succeed in sampling a correct value if the outcome is a distance j away from the peak of the chosen distribution.

To recover the corresponding value in the case of X_1, we subtract the difference of the centre points of X_1 and the randomly chosen distribution. Assume that X_2 was the randomly chosen distribution. If we sampled a value k from X_2, then we say that the outcome of sampling from X_1 was $k - f(s_1, s_2)$, where $f(s_i, s_j)$ gives the directional shift between the centre points of distribution i and distribution j. For example, $f(s_1, s_2) = -1$, while $f(s_2, s_4) = 2$. Subtracting off the

directional shift given by f, "re-centers" the distribution about the peak value of X_1. Accordingly, the probability of success is a generalized form of the above expression:

$$\Pr[X_1 = k] \quad = \quad \frac{1}{4}\sum_{j=1}^{4} \Pr[X_j = k + f(s_1, s_j)]. \tag{6.1}$$

Essentially, such a process will be used to uniformize the distributions of estimated phase parameters. The eigenvalue spectrum of **U** gives a set of unknown phases, each of which has an associated phase parameter. For some phase parameter ω_i, we estimate it by applying a quantum Fourier transform of dimension M in the phase estimation algorithm. The resulting distribution of the estimator for ω_i is centred at the best m-bit ($m = \log M$) estimate of ω_i, denoted s_i. Accordingly, the remainder of this chapter is set in the context of these centre-points as being the best m-bit estimates of their corresponding phase parameters.

In the example above, knowing the centre-points was essential, and so how does one learn these unknown quantities? The eigenvalue spectrum of **U** is a priori unknown, and this presents a challenge in learning the required peak points. Furthermore, one does not have a directional shift function such as f. Even if the respective peak points were known, how does one "re-centre" the sampled outcome in the absence of such a function?

Intuitively, we will choose a larger set of distributions over which to uniformize. Thus, instead of uniformizing over the set of phase parameters in the eigenvalue spectrum of **U**, we uniformize over R distributions by choosing the set $\{r/R \mid r = 0 \ldots R-1\}$ and concocting R distributions, each centred at r/R. Essentially, we create R phase parameters which give R distributions over which we uniformize. If R is large enough, then we hope to cover the set of phase parameters in the eigenvalue spectrum of **U**. Furthermore, this set of phases forms a regular mesh, which enables us to easily "re-centre" distributions as will be shown in later sections. In

other words, this mesh gives a very simple form to the directional shift function f. Finally, note that in the work of Mosca and Zalka [21], they knew the form of the phases which enabled them to cover the eigenspectrum exactly. Knowing the form for the phases effectively gave them the function f. Here, we do not assume to know this, and therefore, our treatment is more general in this regard.

6.2.2 The algorithm

An algorithm for uniformized phase estimation follows quite easily from the standard phase estimation algorithm. Consider as before, a unitary operator $\mathbf{U}$ and an eigenstate $|\Psi_\omega\rangle$ of $\mathbf{U}$ with eigenvalue $e^{2\pi i\omega}$. The standard phase estimation algorithm can be found in Chapter 5.

Ideally, we wish to uniformize over all phase parameters that occur in the eigenvalue spectrum of $\mathbf{U}$, but unfortunately, this set is a priori unknown. We do know, however, that the phase parameter values occur between 0 and 1 which we will use to our advantage. As suggested in the previous section, we choose R values to average over instead of the set of phase parameters. The hope is that if R is large enough, then we will effectively cover the eigenvalue spectrum of U.

For the uniformized version of this algorithm, we require an extra unitary operator c-$\mathbf{U_R}$ with eigenstate $|\varphi\rangle$. The operator c-$\mathbf{U_R}$ is an addition modulo R operator that adds some integer k modulo R to the target register. Since $|\varphi\rangle$ is an eigenstate of this operator, it follows that adding any integer to it results in the same state $|\varphi\rangle$. Here, we show how it is used and explain it in more detail at a later point.

$$\text{c-}\mathbf{U_R}\,|r\rangle\,|j\rangle\,|\varphi\rangle \;=\; e^{2\pi i(\frac{jr}{R} \mod R)}\,|r\rangle\,|j\rangle\,|\varphi\rangle \tag{6.2}$$

The operator c-$\mathbf{U_R}$ adds jr to $|\varphi\rangle$ and this causes a phase "kick-back" of $e^{\frac{2\pi i jr}{R}}$.

The operator c-$\mathbf{U_R}$ is denoted as such since it is a controlled operator. We also require an additional controlled unitary subtraction operator c-$\mathbf{S_R}$ which acts as c-$\mathbf{S_R}\,|r\rangle\,|k\rangle \;\rightarrow\; |r\rangle\,|k-\frac{r}{R}\rangle$. The algorithm now proceeds as follows (normalization constants omitted):

$$|0\rangle\,|0\rangle\,|\Psi_\omega\rangle\,|\varphi\rangle \overset{QFT_R\otimes QFT_M}{\rightarrow} \sum_{r=0}^{R-1}|r\rangle\sum_{j=0}^{M-1}|j\rangle\,|\Psi_\omega\rangle\,|\varphi\rangle \tag{6.3}$$

$$\overset{\text{c-}\mathbf{U}^j}{\rightarrow} \sum_{r=0}^{R-1}|r\rangle\sum_{j=0}^{M-1}e^{2\pi i\omega j}\,|j\rangle\,(|\Psi_\omega\rangle\,|\varphi\rangle) \tag{6.4}$$

$$\overset{\text{c-}\mathbf{U_R}^r}{\rightarrow} \sum_{r=0}^{R-1}|r\rangle\sum_{j=0}^{M-1}e^{2\pi i j(\omega+\frac{r}{R})}\,|j\rangle\,(|\Psi_\omega\rangle\,|\varphi\rangle) \tag{6.5}$$

$$\overset{QFT_M^{-1}}{\rightarrow} \sum_{r=0}^{R-1}|r\rangle\left|\widetilde{\omega+\frac{r}{R}}\right\rangle|\Psi_\omega\rangle\,|\varphi\rangle \tag{6.6}$$

$$\overset{\text{c-}\mathbf{S_R}}{\rightarrow} \sum_{r=0}^{R-1}|r\rangle\left|\widetilde{\omega+\frac{r}{R}}-\frac{r}{R}\right\rangle|\Psi_\omega\rangle\,|\varphi\rangle \tag{6.7}$$

We have abused notation above in $\left|\widetilde{\omega+\frac{r}{R}}-\frac{r}{R}\right\rangle$, where the $\widetilde{\omega+\frac{r}{R}}$ portion of the state is a superposition over estimates for $\omega+\frac{r}{R}$. Subtracting r/R from this implies that r/R is subtracted from each term of the superposition. Unfortunately, explicitly showing this would complicate the presentation and therefore, we will continue to use this notation despite it not being entirely correct.

From the algorithm, we are left with the (normalized) state:

$$\frac{1}{\sqrt{R}}\sum_{x=0}^{R-1}|r\rangle\left|\widetilde{\omega+\frac{r}{R}}-\frac{r}{R}\right\rangle.$$

Upon tracing out r, we are left with the mixed state

$$\sum_{r=0}^{R-1}\frac{1}{R}\left|\widetilde{\omega+\frac{r}{R}}-\frac{r}{R}\right\rangle\left\langle\widetilde{\omega+\frac{r}{R}}-\frac{r}{R}\right| \tag{6.8}$$

Notice that the probability of obtaining any element of the mixture is exactly $1/R$. This corresponds to sampling from a randomly chosen distribution as in section 6.2.1. Subtracting r/R corresponds to subtracting the difference of the peak points of the distributions of $\widetilde{\omega}$ and $\widetilde{\omega+r/R}$. In other words, subtracting r/R is the equivalent of removing the directional shift between these two distributions.

If we have uniformized approximations for two phase parameters ω_1 and ω_2, then the number of possible estimates for these parameters is directly dependent on R. In particular, if ω_1 and ω_2 are not congruent modulo 1, then the set of values over which we uniformize will be completely disjoint. Thus, increasing R brings these sets closer together. With larger values of R, we have a denser subset of the curve in figure 6.1.

6.3 Errors in the Uniformized Algorithm

6.3.1 Re-centering the estimator without affecting probabilities

Consider some phase parameter ω. In the standard phase estimation algorithm, an estimator $|\widetilde{\omega}\rangle$ takes on discrete values in the set $[0,1)$. The uniformized version of the algorithm yields an

estimator $\left|\widetilde{\omega + r/R}\right\rangle$ which also takes on discrete values in the set $[0,1)$. Since the addition by r/R in the uniformized algorithm is just a re-phasing, it follows that $\left|\widetilde{\omega + r/R}\right\rangle$ yields a good estimate of $\omega + r/R$, i.e. the amplitudes (probabilities) are peaked around the best estimates of $\omega + r/R$. Subtracting r/R does not alter the amplitudes, but rather it "re-centers" the distribution approximately about ω.

We know that measuring $\left|\widetilde{\omega + r/R}\right\rangle$ will give us the best m-bit approximation to $\omega + r/R$ with probability at least $4/\pi^2$ [9]. Thus,

$$\begin{aligned} \omega + \frac{r}{R} - 1/M \quad &\leq \widetilde{\omega + \tfrac{r}{R}} \leq \quad \omega + \frac{r}{R} + 1/M \\ \omega - 1/M \quad &\leq \widetilde{\omega + \tfrac{r}{R}} - \tfrac{r}{R} \leq \quad \omega + 1/M \end{aligned}$$

showing that subtracting r/R "re-centers" the distribution approximately about ω and still gives us the best m-bit approximation of ω, again with probability at least $4/\pi^2$.

By tracing through the algorithm, it can be shown that the subtracting operator does not change any of the amplitudes. One can see this by considering the application of the inverse QFT. It therefore follows that $\left|\widetilde{\omega + r/R} - r/R\right\rangle$ yields the best m-bit estimate of ω with probability at least $4/\pi^2$. Since amplitudes are not altered, the subtraction can be a classical post-processing operation.

6.3.2 Success probabilities

In this section, we will consider the probability of measuring any value $0 \leq \phi < 1$. Consider the resulting state after having applied the inverse quantum Fourier transform in the uniformized phase estimation algorithm:

$$\begin{aligned} \overset{QFT_M^{-1}}{\rightarrow} \quad & \frac{1}{\sqrt{R}} \sum_{r=0}^{R-1} |r\rangle \frac{1}{M} \sum_{j,k=0}^{M-1} e^{2\pi i j(\omega+\frac{r}{R})} e^{-2\pi i \frac{jk}{M}} |k\rangle \left(|\Psi_\omega\rangle |\varphi\rangle \right) \\ = \quad & \frac{1}{\sqrt{R}} \sum_{r=0}^{R-1} |r\rangle \sum_{k=0}^{M-1} \left(\frac{1}{M} \sum_{j=0}^{M-1} e^{2\pi i(\omega+\frac{r}{R}-\frac{k}{M})j} \right) |k\rangle \left(|\Psi_\omega\rangle |\varphi\rangle \right) \end{aligned}$$

Notice that the amplitudes for any value of $k \in \{0,\ldots,M-1\}$ form a geometric sum. Let the random variable $X_{r,\omega}$ correspond to the outcome of measuring this state. Thus for any fixed value of r, we have that the probability of measuring k from $X_{r,\omega}$ is:

$$\Pr[X_{r,\omega} = k] = \left| \frac{1}{M} \frac{1 - e^{2\pi i(\omega+\frac{r}{R}-\frac{k}{M})M}}{1 - e^{2\pi i(\omega+\frac{r}{R}-\frac{k}{M})}} \right|^2 \tag{6.9}$$

The following fact, proved in appendix A.1, will be useful in simplifying this expression:

Fact 1.

$$\left| \frac{1}{M} \frac{1 - e^{2iMx}}{1 - e^{2ix}} \right|^2 = \frac{\sin^2(Mx)}{M^2 \sin^2(x)}$$

Using fact 1, the probability of observing k from a distribution parameterized by ω and r simplifies to:

$$\Pr[X_{r,\omega} = k] = \frac{\sin^2(M\pi(\omega + \frac{r}{R} - \frac{k}{M}))}{M^2 \sin^2(\pi(\omega + \frac{r}{R} - \frac{k}{M}))}. \tag{6.10}$$

For any fixed r, the possible values of k come from a discrete set contained in $[0, M-1]$. The measured value k, when re-scaled by M as k/M, is the best m-bit estimate of $\omega + \frac{r}{R}$, with probability at least $4/\pi^2$. Subsequently, we subtract the random shift r/R away from k/M. Thus, when these two events are jointly considered, the possible resulting values from this algorithm come from a discrete set of size $\mathrm{LCM}(R, M)$. If M divides R, then this discrete set is of size R. Henceforth, we assume that $R = TM$, for $T \geq 1$. Any possible outcome of the algorithm must therefore have the form:

$$\frac{k}{M} - \frac{r}{TM} = \frac{kT - r}{TM}. \tag{6.11}$$

This implies that $\phi = k/M - r/TM = l/TM$ where $l = kT - r \mod TM$. Note that ϕ is restricted to a discrete set of R values in $[0, 1]$. Now consider the probability of observing $\phi = l/TM$ as the measured outcome (with subtraction post-processing), given that the phase parameter being estimated is ω:

$$\begin{aligned} \Pr[X = \phi \mid \omega] &= \frac{1}{TM} \Pr\left[\frac{k}{M} \mid \omega + \frac{r}{TM}\right] && (6.12) \\ &= \frac{1}{TM} \Pr[X_{r,\omega} = k] && (6.13) \\ &= \frac{1}{TM} \frac{\sin^2(M\pi(\omega + \frac{r}{TM} - \frac{k}{M}))}{M^2 \sin^2(\pi(\omega + \frac{r}{TM} - \frac{k}{M}))}. && (6.14) \end{aligned}$$

Here we have incorporated the probability function in equation (6.10). Note that the factor of $1/TM$ arises from choosing the distribution corresponding to r; the remainder of the expression corresponds to measuring k/M from the distribution parameterized by ω and r.

Consider a continuous interval $[a, b]$; we can extend this probability expression as:

$$\Pr[a \leq X \leq b \mid \omega] \quad = \quad \sum_{j=1}^{R} \frac{1}{TM} \frac{\sin^2(M\pi(\omega + \frac{r_j}{TM} - \frac{k_j}{M}))}{M^2 \sin^2(\pi(\omega + \frac{r_j}{TM} - \frac{k_j}{M}))} \tag{6.15}$$

Here, we have partitioned $[a,b]$ into R intervals, where consecutive points are separated by $1/R$. More specifically, the point $r_j/TM - k_j/M$ is some point in the j^{th} interval in the partition of $[a,b]$.

As R tends to infinity (but is still finite), this expression implies that the mesh of possible values for X gets finer and finer and we begin to cover more and more of the curve illustrated in figure 6.1.

In the limit as R tends to infinity, the uniformized distributions for two different phase parameters will be exactly the same, which corresponds exactly to an ideal, continuous uniformization. In the limit as R tends to infinity, the random variable X corresponding to the phase parameter ω is a continuous random variable, and so the probability of observing any single value ϕ is identically 0. When we now consider the range $[a,b]$, the associated probability function becomes:

$$\Pr[a < X < b] \quad = \quad \lim_{n\to\infty} \sum_{j=1}^{n} \frac{1}{n} \frac{\sin^2(M\pi(\omega + \frac{r_j}{R} - \frac{k_j}{M}))}{M^2 \sin^2(\pi(\omega + \frac{r_j}{R} - \frac{k_j}{M}))} \tag{6.16}$$

This is exactly the definition of the Riemann integral, and thus in the limit as R tends to infinity:

$$\Pr[a < X < b] \quad = \quad \int_a^b \frac{\sin^2(M\pi(\omega - x))}{M^2 \sin^2(\pi(\omega - x))} dx \tag{6.17}$$

Consider now two phase parameters ω_1 and ω_2. It is clear from the continuously uniformized distributions that for the ranges $[a - \omega_1, b - \omega_1]$ and $[a - \omega_2, b - \omega_2]$:

$$\Pr[a-\omega_1 < X_{\omega_1} < b-\omega_1] - \Pr[a-\omega_2 < X_{\omega_2} < b-\omega_2] \quad = \quad 0$$

which shows that in the limit as R tends to infinity, the distributions for ω_1 and ω_2 are identical. This is also an expression of the independence from the underlying phase parameters being estimated.

Over a discrete mesh, some dependency remains, but it is reduced considerably by this process. This is evident from the derivation above: namely, any measured value must be of the form l/R, and so, for $\phi-\omega_1 = \phi-\omega_2 \mod 1/R$ to be true, it must be the case that $\omega_1 = \omega_2 \mod 1/R$ which will not be true in general. We can only conclude that the discretely uniformized distributions for ω_1 and ω_2 are identical, contingent on $\omega_1 = \omega_2 \mod 1/R$, which is an expression of the remaining dependence on the underlying phase parameters. As R increases, this dependence is further dampened, and this dependency can be made arbitrarily small by choosing R appropriately. This analysis is found in the next section.

Immediately, one may ask for the probability of measuring the best m-bit estimate of ω in this uniformized phase estimation algorithm. A closed form expression would be possible by evaluating the integral in equation (6.17). However, there does not seem to be a closed form for the solution of this integral. One possible approach is to integrate its Taylor expansion at a suitable point.

Considering the probability of the best m-bit estimate, one only needs to note that the distribution obtained by averaging over all possible values of the phase parameter will always be at least as good as the worst possible distribution. We know that obtaining an estimate $\widetilde{\omega}$ such that $|\widetilde{\omega}-\omega| \leq 1/2M$ is possible with probability at least $4/\pi^2$ which is a worst-case lower bound.

Additionally, we get one of the two best m-bit estimates ($|\widetilde{\omega} - \omega| \leq 1/M$) with probability at least $8/\pi^2$, again, in the worst-case distribution. Thus these bounds follow through in the uniformized phase estimation algorithm, which gives an averaged distribution.

6.4 Approximate Monotonicity of the Uniformized Probability Function

In this section, we consider the remaining dependence on the phase parameters after a discrete-point uniformization. As stated previously, full independence is only achieved in the limit as R tends to infinity. Here, we show that this dependence is very small.

Additionally, we consider the task of comparing the continuously uniformized distributions of two phase parameters ω_1 and ω_2. If the midpoint of some continuous interval $[a,b]$ is closer to ω_1, then we we would expect that $\Pr[a \leq \widetilde{\omega_1} \leq b] \geq \Pr[a \leq \widetilde{\omega_2} \leq b]$. Since the underlying probability functions fluctuate frequently, it follows that such a relation is not necessarily true. We analyze these functions and derive a concept of *approximate monotonicity*. In particular, for phase parameters ω_1 and ω_2 and interval $[a,b]$ whose midpoint is closer to ω_1, we show that this probability relation holds up to a minor error of $O(1/M)$.

More specifically, for two phase parameters ω_1 and ω_2, and interval $[a,b]$ such that the midpoint of $[a,b]$ is closer to ω_1, we show that:

$$\Pr[a < \widetilde{\omega_1} < b] \quad \geq \quad \Pr[a < \widetilde{\omega_2} < b] - \frac{3}{M} \tag{6.18}$$

6.4.1 Approximate monotonicity in continuously uniformized distributions

The continuous function:

$$f(x) = \frac{\sin^2(M\pi x)}{M^2 \sin^2(\pi x)} \tag{6.19}$$

is not monotonic, and thus comparing two continuously uniformized distributions for ω_1 and ω_2 will not be as straight-forward as comparing two unimodal distributions (e.g., the normal distribution).

It is easy to see that $f(x)$ is indeed an even function, and therefore is symmetric about its peak. This means that there is an axis of symmetry at the peak point. Without loss of generality, we wish to argue that for two disjoint $1/M$-sized windows to the left of the peak, separated by a distance of k/M, where $k \in [0, \frac{M-1}{2} - 2]$, the area under the window closer to the peak will be greater than that of the window that is further away from the peak. Let A_1 be the window that is closer to the peak, and A_2 be the window that is further away.

Notice that $f(x)$ is a quotient of two sinusoids: the $\sin^2(M\pi x)$ portion has a period of $1/M$, while the $\sin^2(\pi x)$ has a period of 1. The denominator thus has the effect of scaling the numerator, which is why at each $1/M$, we observe a new period of the $\sin^2(M\pi x)$ sinusoid.[1] The combination of the two functions is illustrated in figure 6.3.

If the intervals A_1 and A_2 have a distance of k/M between them, then it follows that each interval contains the same period of $\sin^2(M\pi x)$. Each window is then scaled by some amount, dictated by $M^2 \sin^2(\pi x)$. Since $M^2 \sin^2(\pi x)$ is monotone increasing in the direction of the global

[1] an exception is at the highest peak which spans a window of size $2/M$.

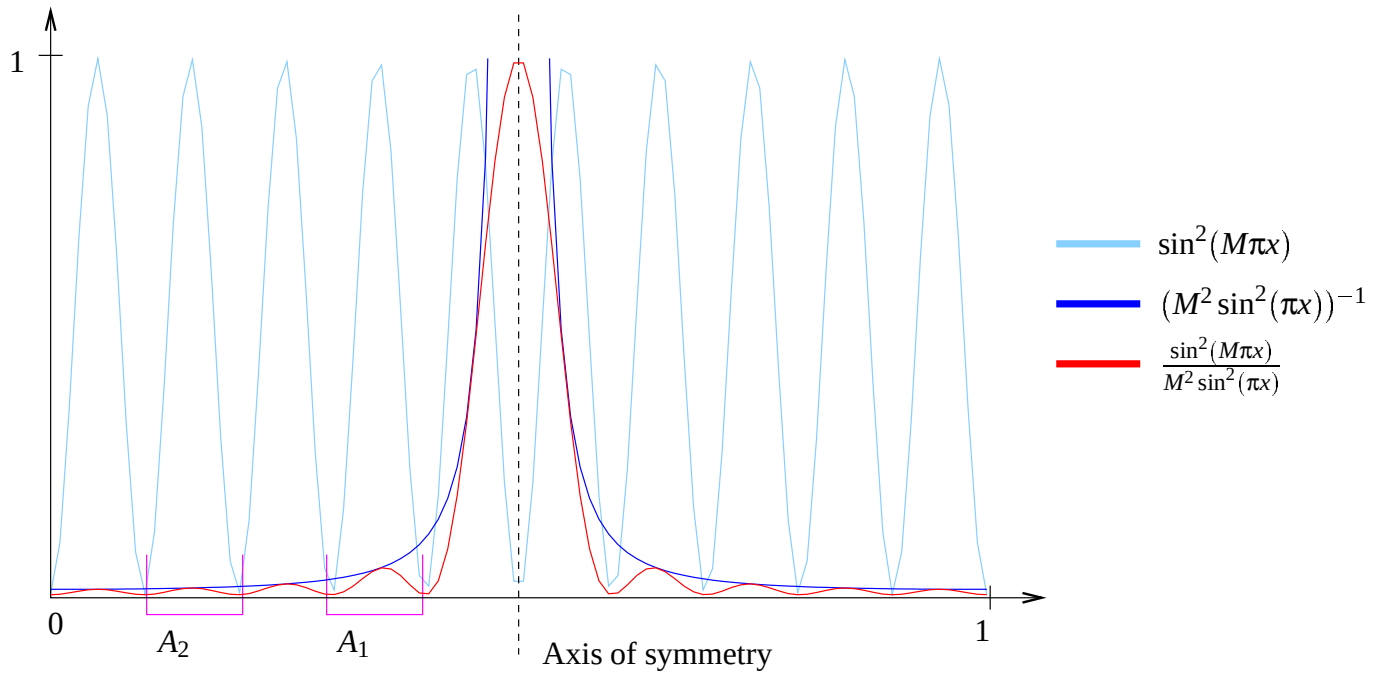


Figure 6.3: The function $f(x)$ is the ratio of two sinusoidal functions. The intervals A_1 and A_2 are also illustrated here.

peak of $f(x)$, it follows that the period contained in A_1 will be scaled by a larger amount than the period in A_2, and therefore the area in A_1 will be greater than the area in A_2.

Note that if the intervals A_1 and A_2 were of size j/M, for $j > 1$, then the same result would apply because each of the corresponding j sub-intervals (of size $1/M$) of both A_1 and A_2 could be compared in this way.

A canonical form

Consider two continuously uniformized distributions for ω_1 and ω_2. We re-label $f(x)$ as $f_{\omega_i}(x)$ to indicate that the function f has its peak shifted to ω_i.

Additionally, consider a closed interval $[a,b]$ over which we will integrate $f_{\omega_1}(x)$ and $f_{\omega_2}(x)$ to study the quantities $\Pr[a \leq \widetilde{\omega_1} \leq b]$ and $\Pr[a \leq \widetilde{\omega_2} \leq b]$. Let the distance between ω_1 and ω_2 be δ. Then clearly, in the context of $f_{\omega_1}(x)$, we have two intervals, $[a,b]$ and $[a+\delta, b+\delta]$ which is our canonical form. In other words, when comparing two distributions, we consider the two intervals in the reference frame of ω_1. This means that we will consider:

$$\int_a^b f_{\omega_1}(x) - f_{\omega_2}(x)dx \quad = \quad \int_a^b f_{\omega_1}(x)dx - \int_{a+\delta}^{b+\delta} f_{\omega_1}(x)dx \tag{6.20}$$

The plot of $f_{\omega_1}(x)$ along with the two intervals is illustrated in figure 6.4. Note that it is more convenient to consider f on a circle, due to its periodic nature.

Immediately, we can conclude that in the canonical form, we will always have two disjoint intervals. If $\delta > b-a$, then intervals will clearly be disjoint. If $\delta \leq b-a$, then the overlapping domain will cancel out in the integral expression; this is evident in equation (6.20). If canceling occurs, then the two disjoint intervals are $[a, a+\delta]$ and $[b, b+\delta]$. This scenario is illustrated in figure 6.5.

Depending on the location of the interval $[a,b]$, the expression in equation (6.20) may be positive or negative. More specifically, if the midpoint of $[a,b]$ is closer to ω_1, then this expression will be positive, while if its midpoint is closer to ω_2, then it will be negative.

Case	Location of Interval	$\Pr[a < \widetilde{\omega_1} < b] - \Pr[a < \widetilde{\omega_1} < b]$
1	Midpoint of $[a,b]$ closer to ω_1	> 0
2	Midpoint of $[a,b]$ closer to ω_2	< 0

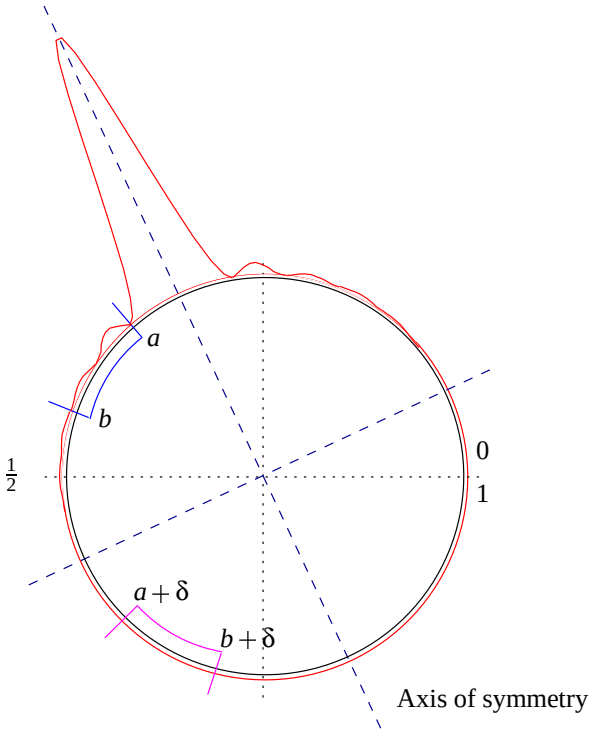


Figure 6.4: An illustration of the canonical form

In both cases, one can apply a series of reflections about the axis of symmetry, which, along with canceling portions of overlapping domain intervals, will transform any case into the canonical form. We will show how this is done in case one. Note that in the canonical form, case one implies that the midpoint of $[a,b]$ is closer to ω_1, more so than the midpoint of $[a+\delta,b+\delta]$.

The consequences of extending an interval

At the beginning of this section, we argued for monotonicity with intervals A_1 and A_2 which were each of size j/M. A priori however, the interval $[a,b]$ may not be of this size. By extending the

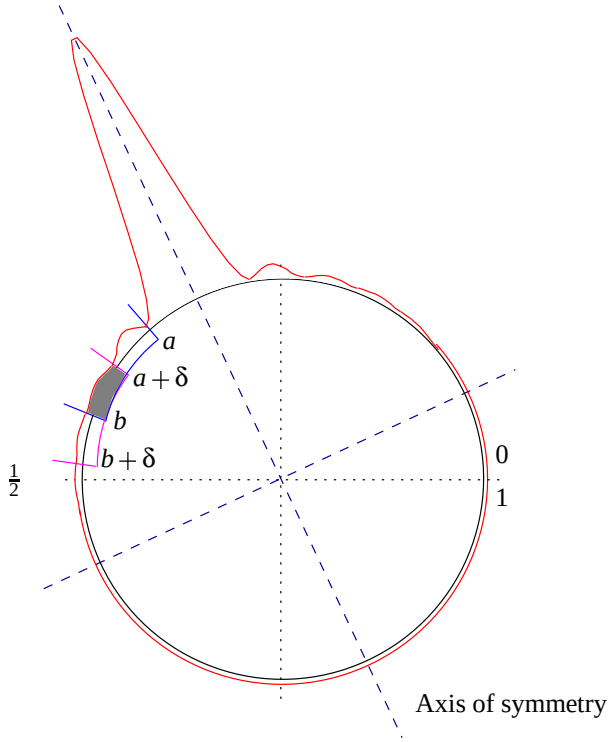


Figure 6.5: An intuitive argument for disjoint intervals

interval in the direction of a, i.e., subtracting some amount from a, we can resize it so that the interval has width which is an integer fraction of M. Suppose now that we subtract some amount from a so that the new interval is $[a', b]$.

The integral expression in equation (6.20) changes as follows:

$$\begin{aligned} \left| \int_a^b f_{\omega_1}(x) - f_{\omega_2}(x) dx \right| &= \left| \int_{a'}^b f_{\omega_1}(x) - f_{\omega_2}(x) dx - \int_{a'}^a f_{\omega_1}(x) - f_{\omega_2}(x) dx \right| \\ &\leq \left| \int_{a'}^b f_{\omega_1}(x) - f_{\omega_2}(x) dx \right| + \frac{1}{M} \end{aligned}$$

Here, we have used the fact that f has a global maximum at 1 and that $f \geq 0$, and that $a - a' \leq 1/M$. Thus, integrating f from a' to a contributes at most $1/M$ and therefore, extending the interval to ensure that its width is an integer fraction of M implies an error of at most $1/M$.

Henceforth, we will always assume that $[a,b]$ has a width that is an integer fraction of M and we will accept an error of $1/M$. For simplicity, we say that a *nice width* for an interval is a width that is an integer fraction of M.

In addition to A_1 and A_2 having nice widths, they were also separated by distance k/M. Again, this may not be the case with $\omega_1 - \omega_2 = \delta$. Similar to extending $[a,b]$ to $[a',b]$, we modify δ to δ' such that $\delta' = k/M$. Once again, $|\delta - \delta'| \leq 1/M$, and so, the maximum error introduced by this change is $1/M$. Combining the two extensions, we get that:

$$\begin{aligned}
\left| \int_a^b f_{\omega_1}(x) - f_{\omega_2}(x) dx \right| &\leq \left| \int_{a'}^b f_{\omega_1}(x) - f_{\omega_2}(x) dx \right| + \frac{1}{M} \\
&= \left| \int_{a'}^b f_{\omega_1}(x) dx - \int_{a'+\delta}^{b+\delta} f_{\omega_1}(x) dx \right| + \frac{1}{M} \\
&\leq \left| \int_a^b f_{\omega_1}(x) dx - \int_{a'+\delta'}^{b+\delta'} f_{\omega_1}(x) dx \right| + \frac{2}{M} \\
&= \left| \int_{a'}^b f_{\omega_1}(x) - f_{\omega_2'}(x) dx \right| + \frac{2}{M}
\end{aligned}$$

where ω_2' is ω_2 shifted by $\delta' - \delta$. The net result is that by resizing the interval and extending δ, we incur an error of at most $2/M$.

Approximate monotonicity in case 1

Assume that the interval $[a,b]$ has a nice width; for now, we do not make any assumptions on δ. Consider case 1 again, namely that the midpoint of $[a,b]$ is closer to ω_1.

Since there is no assumed condition on δ, it follows that in the canonical form, the two intervals may not be on the same side of the axis of symmetry. This is illustrated in figure 6.6.

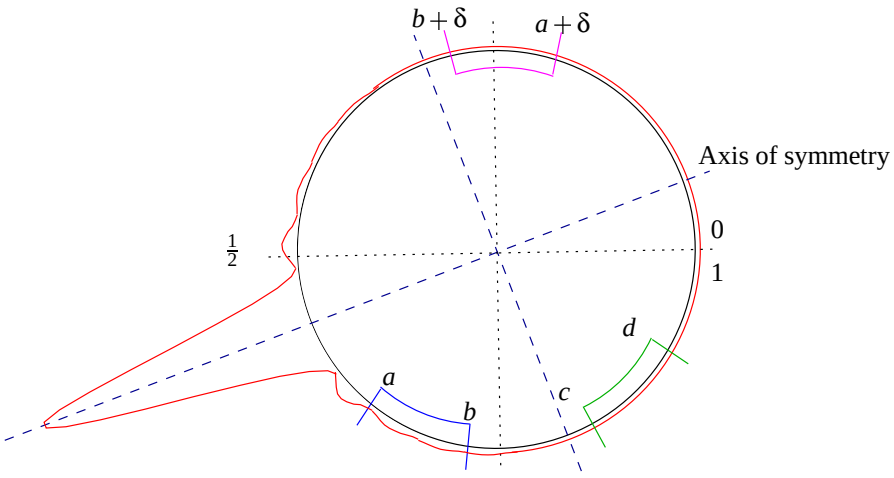


Figure 6.6: Phase parameters which are sufficiently separated may imply that the intervals $[a,b]$ and $[a+\delta,b+\delta]$ lie on opposite sides of the axis of symmetry. The shifted interval $[a+\delta,b+\delta]$ can be reflected about the axis of symmetry since f is even. The reflected interval is labeled $[c,d]$.

Note that the shifted interval $[a+\delta,b+\delta]$ can be reflected about the axis of symmetry since f is odd. The reflected interval is labeled $[c,d]$ in figure 6.6, and it will most likely not be separated from $[a,b]$ by a distance which is some integer multiple of $1/M$. This means that the intervals $[a,b]$ and $[c,d]$ may not contain the same period of the $\sin^2(M\pi x)$ portion of f. Once again, we

can shift the reflected interval appropriately at a cost of $1/M$ to to ensure an integer multiple of $1/M$ distance between the two intervals. This corresponds in some way to shifting δ, i.e., the type of shift involved is the same shift when aligning the two distributions by increasing/decreasing δ.

Now that case one has been transformed into the canonical form, it follows that:

$$\left|\int_a^b f_{\omega_1}(x) - f_{\omega_2}(x)dx\right| \quad \leq \quad \left|\int_a^b f_{\omega_1}(x)dx - \int_{c'}^{d'} f_{\omega_1}(x)dx\right| + \frac{2}{M}$$

where c and d have been shifted to c' and d' to align the two intervals at a distance which is an integer multiple of $1/M$. It follows that ,

$$\int_a^b f_{\omega_1}(x)dx - \int_{c'}^{d'} f_{\omega_1}(x)dx \quad \geq \quad 0$$

and thus,

$$\int_a^b f_{\omega_2}(x)dx - \frac{2}{M} \quad \leq \int_a^b f_{\omega_1}(x)dx \leq \quad \int_a^b f_{\omega_2}(x)dx + \frac{2}{M}$$

which is an expression of approximate monotonicity. In other words:

$$\Pr[a < \widetilde{\omega_2} < b] - \frac{2}{M} \quad \leq \Pr[a < \widetilde{\omega_1} < b] \leq \quad \Pr[a < \widetilde{\omega_2} < b] + \frac{2}{M}.$$

6.4.2 Uniformized distributions are close

Consider the following two definitions:

Definition 9. *For the phase parameter ω, let $D_{R,\omega}(x)$ be the probability that the measurement of the state $|\widetilde{\omega}\rangle$ results in $x = k/M - r/R \mod R$, under a discrete, R-point uniformization. More specifically, consider the discrete random variable X corresponding to the measurement of $|\widetilde{\omega}\rangle$. Formally,*

$$D_{R,X}(x) = \frac{1}{R}\frac{\sin^2(M\pi(\omega + r/R - k/M))}{M^2\sin^2(\pi(\omega + r/R - k/M))}. \tag{6.21}$$

As per section 6.3.2, for an interval $[a,b]$, it follows that,

$$\Pr_R[a \leq X \leq b] = \sum_{j=1}^{R} D_{R,X}(k_j/M - r_j/R). \tag{6.22}$$

Probabilities in discrete uniformizations are given a subscript R to highlight the granularity of the discrete mesh.

Definition 10. *For some phase parameter ω, let $C_\omega(x)$ be the continuously uniformized probability function. Thus, for the continuous random variable X corresponding to ω:*

$$\Pr_\infty[a < X < b] = \int_a^b C_\omega(x)dx$$

where

$$C_\omega(x) = \frac{\sin^2(M\pi(\omega - x))}{M^2\sin^2(\pi(\omega - x))}.$$

Probabilities in continuous uniformizations are given a subscript ∞ to highlight a continuous mesh.

From appendix A.3, it follows that:

$$\int_a^b C_\omega(x)dx - \frac{c}{6T^2} \leq \sum_{j=1}^{n} D_{R,X}(k_j/M - r_j/R) \leq \int_a^b C_\omega(x)dx + \frac{c}{6T^2}. \tag{6.23}$$

Consider the phase parameters ω_1 and ω_2 as before, and let the random variables X_1 and X_2 correspond to the measurement of the estimators $|\widetilde{\omega_1}\rangle$ and $|\widetilde{\omega_2}\rangle$ respectively. Also consider case one in section 6.4.1: since the midpoint of $[a,b]$ is closer to ω_1, we should expect that $\Pr_R[a < X_1 < b] \geq \Pr_R[a < X_2 < b]$. Under a continuous uniformization, we showed that this was true up to an error of $2/M$.

Since X_1 is discrete, it follows that $\Pr_R[a < X_1 < b] = \sum_{j=1}^{R} D_{R,X_1}(k_j/M - r_j/R)$. Similarly for X_2, we have that $\Pr_R[a < X_2 < b] = \sum_{j=1}^{R} D_{R,X_2}(k_j/M - r_j/R)$. Using the bounding expression in equation (6.23), we have that:

$$\left| \sum_{j=1}^{R} \left(D_{R,X_1}\left(\frac{k_j}{M} - \frac{r_j}{R}\right) - D_{R,X_2}\left(\frac{k_j}{M} - \frac{r_j}{R}\right)\right)\right| \leq \left(\int_a^b C_{\omega_1}(i) - C_{\omega_2}(i)dx + \frac{c}{3T^2}\right) \tag{6.24}$$

$$\leq \frac{2}{M} + \frac{c}{3T^2}. \tag{6.25}$$

Thus the probability difference in the discretely uniformized distributions of $\widetilde{\omega_1}$ and $\widetilde{\omega_2}$ is at most $c/3T^2$ away from the probability difference in the ideal, continuous uniformization. Once again, since T can be chosen as desired, the error in the monotonicity of the discretely uniformized distributions can be made as close to $2/M$ as required. For convenience, we choose $T = \sqrt{cM/3}$ as a minimum choice for T. With this as a lower bound for T (and therefore R), it follows that comparisons of discretely uniformized estimates are at most $3/M$ away from comparisons of continuously uniformized estimates.

In summary, for the random variables X_1 and X_2 corresponding to the estimates of two phase parameters ω_1 and ω_2, and an interval $[a,b]$ such that the midpoint of $[a,b]$ is closer to the peak of the distribution of X_1, we have shown that:

$$\Pr_R[a < X_1 < b] \quad \geq \quad \Pr_R[a < X_2 < b] - \frac{3}{M} \tag{6.26}$$

with an appropriate choice of $T \in \Omega(\sqrt{M})$.

6.5 Query Complexity of the Uniformized Phase Estimation Algorithm

The key parts of this algorithm include a QFT (and inverse QFT) of dimensions R and M, $O(M)$ applications of $\mathbf{U}$, one application of $\mathbf{U_R}$, and one application of $\mathbf{S_R}$. Since $\mathbf{U}$ is an abstract operator, we can only consider the query complexity of this algorithm. Once the running time of $\mathbf{U}$ is known, then the asymptotic running time of the algorithm is related to the query complexity by the runtime cost of $\mathbf{U}$. This is similar to the standard, non-uniformized version of the phase estimation algorithm. As such, we require $O(M)$ applications of $\mathbf{U}$, which matches the query complexity of the non-uniformized algorithm.

6.6 The Operator $\mathbf{U_R}$

Recall that $\mathbf{U_R}$ is the addition modulo R operator that has an eigenstate $|\varphi\rangle$. Previously, we showed how $\mathbf{U_R}$ was used, but we did not explain it in detail. Here, we consider how $|\varphi\rangle$ can be easily constructed and how $\mathbf{U_R}$ causes the desired phase kick-back. The idea of $\mathbf{U_R}$ is taken from

Cleve et al. [9], and the full details are found therein.

Consider the state:

$$|\varphi\rangle = \sum_{y=0}^{R-1} e^{-\frac{2\pi i}{R}y} |y\rangle$$

which can be constructed by applying the inverse QFT to the state encoding the binary number 1, i.e., $|000\cdots01\rangle$. Creating $|\varphi\rangle$ requires time proportional to preparing each qubit in the $|0\rangle$ state and then changing the least significant qubit to $|1\rangle$. This takes $O(1)$ time in conventional models of quantum computation. Upon adding jr modulo R to each y in the superposition, we get:

$$\begin{aligned}
\mathbf{U_R}\,|\varphi\rangle &= \sum_{y=0}^{R-1} e^{-\frac{2\pi i}{R}y} |y + jr \mod R\rangle && (6.27)\\
&= e^{\frac{2\pi i jr}{R}} \sum_{y=0}^{R-1} e^{-\frac{2\pi i}{R}(y+jr)} |y + jr \mod R\rangle && (6.28)\\
&= e^{\frac{2\pi i jr}{R}} \sum_{y=0}^{R-1} e^{-\frac{2\pi i}{R}y} |y\rangle && (6.29)\\
&= e^{\frac{2\pi i jr}{R}} |\varphi\rangle . && (6.30)
\end{aligned}$$

Thus adding jr modulo R to the eigenstate $|\varphi\rangle$ causes a desired phase kick-back of $e^{\frac{2\pi i jr}{R}}$ while leaving $|\varphi\rangle$ unchanged. By this examination, $\mathbf{U_R}$ is extremely efficient which means that an exponential setting of R will imply a polynomial runtime for $\mathbf{U_R}$.

6.7 Applications to Counting

One of the immediate applications of uniformized phase estimation is monotonized counting. If we estimate the phases of the eigenvalues of a suitable Grover iterate, then we gain count-

ing information about some related combinatorial problem. Here, we use the uniformized phase estimation algorithm to produce counting statistics, which leads to a monotonized counting algorithm. Quantum counting was reviewed in Chapter 5.

6.7.1 Monotonized quantum counting

The uniformized phase estimation algorithm can be applied to a suitably chosen Grover iterate **G** (as described in Chapter 5) to reveal counting information about an underlying Boolean function. The resulting count approximations are dependent on the value of the count itself, and this is evident in the counting bounds of Brassard et al. [7]:

$$|\widetilde{a} - a| \quad \leq \quad 2\pi k \frac{\sqrt{a(1-a)}}{M} + k^2 \frac{\pi^2}{M^2}$$

Consider counts a_1 and a_2 such that $a_1 > a_2$. In comparing the distributions of estimates $\widetilde{a_1}$ and $\widetilde{a_2}$, we would like that $\Pr[\widetilde{a_1} > w] \geq \Pr[\widetilde{a_2} > w]$, where $0 \leq a_2 < a_1 \leq w \leq 1$ is some threshold. Since the quantum counting algorithms considered here are based on phase estimation algorithms, then ultimately such counts arise from phase parameters ω_1 and ω_2. In terms of phase estimation, we would like $\Pr[\widetilde{a_1} > w] \geq \Pr[\widetilde{a_2} > w]$ if and only if $\Pr[u < \widetilde{\omega_1} < v] \geq \Pr[u < \widetilde{\omega_2} < v]$where $[u,v]$ is an appropriate continuous interval. In the following sections, we will assume counts a_1 and a_2 having exactly this relation and we will subsequently derive the relationship between the corresponding phase parameters. Using the conclusions of the previous section, we will conclude that $\Pr[\widetilde{a_1} > w] \geq \Pr[\widetilde{a_2} > w]$ for two such counts.

The phases in the eigenvalues of the Grover iterate contain the angle θ_a, and thus, the eigenvalue estimation algorithm reveals something about a as described in Chapter 5 (see the paper by Mosca [20] for a detailed explanation). Here, we apply the techniques of section 6.2.1 to

the counting algorithm from Chapter 5. As before, we start off in the state $|0\rangle|0\rangle\mathbf{A}|0\rangle|\varphi\rangle$ and proceed as follows:

$$
\begin{aligned}
&|0\rangle|0\rangle\mathbf{A}|0\rangle|\varphi\rangle \\
&\overset{QFT_R\otimes QFT_M}{\rightarrow} \quad \frac{-i}{\sqrt{2RM}}\sum_{r=0}^{R-1}|r\rangle\sum_{j=0}^{M-1}|j\rangle\left(|\Psi_+\rangle-|\Psi_-\rangle\right)|\varphi\rangle \\
&\overset{\text{c-}\mathbf{G}^j}{\rightarrow} \quad \frac{-i}{\sqrt{2RM}}\sum_{r=0}^{R-1}|r\rangle\sum_{j=0}^{M-1}|j\rangle\left(e^{2ij\theta_a}|\Psi_+\rangle-e^{-2ij\theta_a}|\Psi_-\rangle\right)|\varphi\rangle \\
&\overset{\text{c-}\mathbf{U_R}^r}{\rightarrow} \quad \frac{-i}{\sqrt{2RM}}\sum_{r=0}^{R-1}|r\rangle\sum_{j=0}^{M-1}|j\rangle\left(e^{2ij\theta_a}e^{\frac{2ijr}{R}}|\Psi_+\rangle-e^{-2ij\theta_a}e^{\frac{2ijr}{R}}|\Psi_-\rangle\right)|\varphi\rangle \\
&= \quad \frac{-i}{\sqrt{2R}}\sum_{r=0}^{R-1}|r\rangle\left(\frac{1}{\sqrt{M}}\sum_{j=0}^{M-1}e^{2ij(\frac{r}{R}+\theta_a)}|j\rangle|\Psi_+\rangle-\frac{1}{\sqrt{M}}\sum_{j=0}^{M-1}e^{2ij(\frac{r}{R}-\theta_a)}|j\rangle|\Psi_-\rangle\right)|\varphi\rangle \\
&\overset{QFT_M^{-1}}{\rightarrow} \quad \frac{-i}{\sqrt{2R}}\sum_{r=0}^{R-1}|r\rangle\left(\left|\widetilde{\frac{r}{R}+\omega}\right\rangle|\Psi_+\rangle-\left|\widetilde{\frac{r}{R}-\omega}\right\rangle|\Psi_-\rangle\right)|\varphi\rangle \\
&\overset{\text{c-}\mathbf{S_R}}{\rightarrow} \quad \frac{-i}{\sqrt{2R}}\sum_{r=0}^{R-1}|r\rangle\left(\left|\widetilde{\frac{r}{R}+\omega}-\frac{r}{R}\right\rangle|\Psi_+\rangle-\left|\widetilde{\frac{r}{R}-\omega}-\frac{r}{R}\right\rangle|\Psi_-\rangle\right)|\varphi\rangle\,.
\end{aligned}
$$

Once the first register is traced out, we are left with the following reduced system:

$$\sum_{r=0}^{R-1}\frac{1}{R}|\Phi_r\rangle\langle\Phi_r| \tag{6.31}$$

where

$$|\Phi_r\rangle \;=\; \left|\widetilde{\frac{r}{R}+\omega}-\frac{r}{R}\right\rangle|\Psi_+\rangle-\left|\widetilde{\frac{r}{R}-\omega}-\frac{r}{R}\right\rangle|\Psi_-\rangle\,. \tag{6.32}$$

Upon tracing out the second register in the eigenvector basis (in this now reduced state), we are left with:

$$\sum_{r=0}^{R-1} \frac{1}{R} \left(|\Phi_r^+\rangle \langle \Phi_r^+| + |\Phi_r^-\rangle \langle \Phi_r^-| \right) \tag{6.33}$$

where

$$|\Phi_r^{\pm}\rangle \quad = \quad \left| \widetilde{\frac{r}{R} \pm \omega} - \frac{r}{R} \right\rangle \tag{6.34}$$

It is clear from the state in equation (6.33), that each r is picked with uniform probability i.e., $1/R$, and in turn we have an equally weighted mixture of $|\Phi_r^+\rangle$ and $|\Phi_r^-\rangle$. It follows that measuring either $|\Phi_r^+\rangle$ or $|\Phi_r^-\rangle$ is sufficient to obtain an estimate for a [7] . Furthermore, by the discussion in Chapter 5, we get the best m-bit approximation of a with probability at least $4/\pi^2$ or higher if we wish by tweaking the algorithm as described by Cleve et al. [9].

6.7.2 Approximate monotonicity of the monotonized quantum counting algorithm

Consider two counts $a_1 = t_1/N$ and $a_2 = t_2/N$ such that $a_1 > a_2$, which we wish to estimate by quantum counting. The discrete uniformized counting algorithm yields estimators for a_1 and a_2 that have distributions that are very close together. This distance is dictated by our choice of R.

Since a_1 and a_2 are produced by computing the sine squared of the angles θ_1 and θ_2 respectively, it follows that we can learn something about a_1 and a_2 by using a phase estimation algorithm to estimate θ_1 and θ_2. Note that while θ_1 and θ_2 are angles, they correspond to phase

parameters ω_1 and ω_2 by $\theta_1 = \pi\omega_1$ and $\theta_2 = \pi\omega_2$. Thus the phase estimation algorithm estimates ω_1 and ω_2 which tell us something about a_1 and a_2.

Consider some threshold $0 \leq w \leq 1$; we would like to compare ω_1 and ω_2 to determine which *count* has a higher probability of being greater than w. Let X_1 and X_2 be discrete random variables corresponding to the measurement outcome of the state $|\widetilde{\omega_1}\rangle$ and $|\widetilde{\omega_2}\rangle$ respectively.

For any value $0 \leq w \leq 1$, there are two angles $\theta_{w,1}$ and $\theta_{w,2}$ such that $\sin^2(\theta_{w,1}) = \sin^2(\theta_{w,2}) = w$. These values fall in the range $[0, \pi]$, and thus to obtain the corresponding phase parameter, we divide them by π.

Let $u = \theta_{w,1}/\pi$ be one of the two values. It follows, therefore, that the second such value is $v = 1 - u$. Thus, we have two values u and v such that $\sin^2(\pi u) = \sin^2(\pi v) = w$.

The approximate monotonicity results of section 6.4 are now applicable: Since $a_2 < a_1 < w$, it follows that the midpoint of $[u, v]$ will be closer to the peak of the distribution of X_1. This scenario is therefore like case one of section 6.4.1, and as argued therein , $\Pr[u < X_1 < v] - \Pr[u < X_2 < v] \geq -2/M$ under an ideal, continuous uniformization. Therefore in the discretely uniformized case, it follows that:

$$\Pr[u < X_1 < v] \quad \geq \Pr[u < X_2 < v] - \tfrac{3}{M}. \tag{6.35}$$

With an appropriate choice for T (recall that $R = TM$). Thus for respective estimates $\widetilde{a_1}$ and $\widetilde{a_2}$ of a_1 and a_2:

$$\Pr[\widetilde{a_1} > w] \quad \geq \Pr[\widetilde{a_2} > w] - \tfrac{3}{M} \tag{6.36}$$

6.7.3 Query complexity of the monotonized quantum counting algorithm

Since the monotonized counting algorithm is just the uniformized phase estimation algorithm applied to a chosen Grover iterate, it follows that the query complexity will match that of the phase estimation algorithm. Thus, we conclude that the query complexity of the monotonized counting algorithm is $O(M)$ applications of **G**, the Grover iterate, which matches that of the non-monotonized version. Note that **G** uses three applications of the underlying Boolean function f, and thus the number of f-queries is three times the number of applications of **G**.

6.8 Concluding Remarks and Open Questions

We have explained in detail how to uniformize the phase estimation algorithm, a technique which was first used by Mosca and Zalka [21]. We also attempted to derive a probability density function for an ideal, continuously uniformized phase estimation algorithm, but as we noted, the integral to be evaluated does not seem to have a closed form solution. Any steps towards this end should improve the lower bound probability for obtaining the best m-bit estimate of a phase parameter and numerical simulations might shed some evidence in this regard. Essentially, Cleve et al. [9] have given worst-case lower bounds for phase estimation and counting, and (ideal) uniformization makes the worst-case equal to the average case which should have better probabilistic guarantees.

We also applied this technique to quantum counting and described a monotonized quantum counting algorithm with all the benefits and probabilistic guarantees of uniformized phase

estimation. We were able to dampen the dependency effects present in the standard counting algorithms by applying a uniformized version of the phase estimation algorithm.

This technique may be useful in algorithms that compare approximate quantum information in a quantum coherent way. As such, uniformization should be considered as a basic tool for standardizing quantum algorithms that give approximate quantities with unknown distributions.

Chapter 7

A Quantum $\{\varepsilon, \delta\}$-FPRAS for Finding the Statistical Mode

7.1 Introduction

Consider a list **S** of N integers drawn from a set **D** of M integers. The *statistical mode* of the list is the number which occurs most often in **S**. For example, if $\mathbf{D} = \{1, 2, 3\}$ and $\mathbf{S} = (1, 2, 3, 1, 3, 2, 3, 2, 1, 2, 2)$, then the mode of **S** is 2, as it occurs most often, five times.

A simple way to find the statistical mode is to examine each number in the list and record its frequency count. Once this information is available, the task of finding the mode is reduced to finding the maximum amongst the frequencies. This method obviously requires $\Omega(N)$ time since we have to produce the counting statistics from the set. As well, it finds the mode exactly. An $\{\varepsilon, \delta\}$-approximation scheme for mode finding can be devised by simple Monte Carlo sampling: Given the list **S**, we sample T elements and declare the most commonly occurring sample as the mode of the list. The analysis of this method is found in the next section.

The following definition of an approximate mode will be used throughout:

Definition 11. *Let the mode of the list* **S** *be w, and let* $\text{count}_{\mathbf{S}} : \{0,\ldots,M-1\} \to \{0,\ldots,N\}$ *be the frequency function, such that* $\text{count}_{\mathbf{S}}(i)$ *denotes the number of times i occurs in* **S**. *An* ε*-approximate mode is any element i whose frequency in the list is an* ε *approximation to that of the mode, i.e.,*

$$(1+\varepsilon)^{-1}\frac{\text{count}_{\mathbf{S}}(w)}{N} \;\leq\; \frac{\text{count}_{\mathbf{S}}(i)}{N} \;\leq\; (1+\varepsilon)\frac{\text{count}_{\mathbf{S}}(w)}{N}.$$

Using quantum mechanics, can we find a more efficient algorithm? In particular, is it possible to design a quantum ε-approximation algorithm for mode finding that does any better than simple Monte Carlo sampling?

In this chapter, we design a quantum $\{\varepsilon,\delta\}$-randomized approximation algorithm for mode finding which quantum-mechanically counts statistics for all M elements of the set. Having these prepared counts in a superposition, we use a variant of the minimum finding algorithm of Dürr and Høyer [14] to find an approximate mode. Furthermore, this algorithm is polynomial in M, $1/\varepsilon$, and $\log 1/\delta$. We find that the quantum method gives a quadratic reduction in the number of list queries in $1/\varepsilon$, and a less-than-quadratic reduction in M, over the classical Monte Carlo sampling schemes presented here.

7.2 A Fully Polynomial Randomized Approximation Scheme for Mode Finding

In this section, we describe a classical fully-polynomial randomized approximation scheme (FPRAS) for mode finding. The results in this section are adapted from the discussions in Motwani and Raghavan [22], Theorem 11.1, and Jerrum et al. [17]. The correctness of this approximation scheme has been adapted from the proof of Theorem 6.4 of Jerrum et al., which implicitly defines an ε-approximation scheme for mode finding that is brought to light here.

7.2.1 A randomized approximation scheme

In what follows, we assume a list **S** of size N with elements drawn from a set **D** of size M. The approximation scheme for mode finding follows from sampling the list for T elements. This scheme is adapted from Jerrum et al. [17].

Definition 12 (adapted from [22, 17]). *Consider a function $f \in \Sigma^* \to \mathbb{N}$. An $\{\varepsilon, \delta\}$-randomized approximation scheme (RAS) for f is a probabilistic Turing machine $\mathcal{A}$ which for all inputs $\langle x, \varepsilon \rangle \in \Sigma^* \times \mathbb{R}^+$ approximates f with output $\mathcal{A}(x, \varepsilon)$ such that:*

$$\Pr[\mathcal{A}(x,\varepsilon) \text{ approximates } f(x) \text{ within ratio } (1+\varepsilon)] \geq 1-\delta.$$

Such a randomized approximation scheme is fully polynomial (FPRAS) if its running time is bounded above by a polynomial in $|x|$, $1/\varepsilon$, and $\log 1/\delta$.

According to this definition, the function f that we aim to approximate is the mode function. The input is given as an oracle and the size of a query is $O(\log N)$. The mode function f will re-

turn the frequency of a modal element of the list. Any $\{\varepsilon,\delta\}$-FPRAS for f must make a number of list queries which is a polynomial in $\log N$, $1/\varepsilon$, $\log 1/\delta$.

In an explicit model, the list would be provided in some data structure, in which case the input would be of size $\Omega(N)$. In such a model, any FPRAS would have make a number of list queries which is at most a polynomial in N (amongst other parameters). In order to compare the quantum black-box FPRAS to the classical explicit-model FPRAS, we limit the quantum procedure by the same rules as the classical procedure. We assume therefore that any classical FPRAS would have the input provided by a black-box oracle which given an index query will return the list element at that index. Accordingly, a query will require $O(\log N)$ bits to write out onto the query tape of the Turing machine.

It will be shown that that quantum FPRAS uses a number of queries which is a polynomial in M, $1/\varepsilon$ and $\log(1/\delta)$. Thus implicitly, we consider the case when $M \in O(\log N)$. This may not always be the case and special consideration must be given to when $M \in \omega(\log N)$.

The procedure for estimating the mode by Monte Carlo sampling is as follows: sample T items from the list, and declare the approximate mode as the one that occurs most often. The accuracy of this procedure is dependent on T, i.e., how close the random variable X_w is to its expected value depends on T, where w is the modal element of the list. Consider the following choice for T:

$$T = \frac{M^3}{\delta\varepsilon^2}, 0 < \delta \leq 1.$$

Notice that T is a polynomial in M and $1/\varepsilon$. It will be shown that the frequency of the mode is approximated within these bounds with probability $1-\delta$. Since T is not a polyno-

mial in $\log(1/\delta)$, it follows that this approximation scheme using T queries will not be a fully-polynomial $\{\varepsilon,\delta\}$-scheme.

We now show that with T samples, the random variable X_u corresponding to some element u will be at most ε/M away from its mean with high probability. Intuitively, this means that for any $u \in \mathbf{D}$, the frequency of u in the sample will be very close to its frequency in the list.

Consider the random variables $Y_{u,i}$ corresponding to the i^{th} sample such that $Y_{u,i} = 1$ if the i^{th} trial was u, and 0 otherwise. Define the random variable $X_u = \sum_{i=0}^{T-1} \frac{Y_{u,i}}{T}$. The expectation μ_u, of X_u is exactly the frequency of u in the list, and it can be shown that the standard deviation of X_u is less than $1/\sqrt{T}$, i.e., $\sigma_u \leq 1/\sqrt{T}$. With these facts, we apply Chebyshev's inequality ([22], theorem 3.3):

$$\begin{aligned} \Pr[|X_u - \mu_u| \leq \varepsilon/M] &= 1 - \Pr[|X_u - \mu_u| \geq \varepsilon/M] \\ &\geq 1 - \frac{\delta}{M} \end{aligned}$$

which shows that with high probability, X_u is very close to its mean value. In particular, this implies that with high probability, no element u in the sample will be mistaken as occurring much more (or less) frequently than it does in the list. As such, the frequency of the modal element will also be close to its value in the list. Note however, that since this close proximity is a probabilistic event, there will always be some (small) chance that the frequency of any element u in the sample will deviate considerably from its expected value. If this happens, one might incorrectly conclude that an outlier element u is the mode of the list.

Following Jerrum et al. [17], we show that the probability of the event in which *all* sampled frequencies are close to their mean values, is high:

$$\begin{aligned}\Pr[|X_u - \mu_u| \geq \varepsilon/M \text{ for any } u] &\leq \sum_u \Pr[|X_u - \mu_u| \geq \varepsilon/M] \\ &\leq \sum_u \frac{\delta}{M} \\ &\leq \delta.\end{aligned}$$

Accordingly,

$$\Pr[|X_u - \mu_u| \leq \varepsilon/M \ \forall u] \geq 1 - \delta.$$

Finally, we note that frequency of the modal element w, is at least $1/M$, and so, the absolute error bound on X_w can be converted to a relative error bound. Thus, we convert the absolute error bound to a relative error bound to get that the frequency of the modal element of the list is approximated by the modal element of the sample within ratio $(1+\varepsilon)$, with probability at least $1-\delta$. The modal element of the sample, which might not be the mode of the list, will be considered an ε-*approximate mode*.

7.2.2 An $\{\varepsilon, \delta\}$-FPRAS for mode finding

The approximation scheme of Jerrum et al. was not fully polynomial since it used $1/\delta$ queries to reduce error probabilities. Here, we modify their approximation scheme and apply a variant of the powering lemma for randomized approximation schemes. The original powering lemma is found in Jerrum et al. [17]. The statement and proof of this variant lemma can be found in appendix A.4.

Consider sampling T items once again, and declaring the most commonly occurring sampled element as the mode; previously, $T = M^3/\delta\varepsilon^2$ samples were used for this purpose. In the analysis above, the use of Chebyshev's inequality demanded $O(M^2/\varepsilon^2)$ samples to ensure that any element was within ε/M of its mean with constant probability. This process was repeated M/δ times to amplify the constant probability of success to be at least $1-\delta/M$. Subsequently, the probability with which all elements are close to their mean value was shown to be at least $1-\delta$.

By the variant powering lemma in appendix A.4, we can amplify the constant probability of success to be at least $1-\delta/M$ using $O(\log(M/\delta))$ repeated rounds of sampling. Thus, with $O\left(M^2\log(M/\delta)\varepsilon^{-2}\right)$ samples, the frequency of any element is within ε/M of its mean with probability at least $1-\delta/M$.

However, we note that a single round of sampling with $O\left(M^2\log(M/\delta)\varepsilon^{-2}\right)$ queries cannot be used to gather statistics about each $i \in D$. This is because the frequencies of these elements in one round of sampling are not independent of each other, and the variant powering lemma requires independent trials. Thus, due to the complex dependence in a single round of sampling, it follows that we must take $O\left(M^2\log(M/\delta)\varepsilon^{-2}\right)$ samples *for each* element $i \in D$, which implies that $T \in O\left(M^3\log(M/\delta)\varepsilon^{-2}\right)$.

Subsequently, the probability with which all elements are close to their mean value is at least $1-\delta$, and this aspect of the analysis remains the same. Since T is bounded above by a polynomial in M, $1/\varepsilon$ and $\log 1/\delta$, it follows that this procedure is an $\{\varepsilon,\delta\}$-FPRAS for mode finding.

In the following sections, we construct a quantum $\{\varepsilon,\delta\}$-FPRAS for mode finding that uses $O\left(\dfrac{M^{3/2}\log M\log 1/\delta}{\varepsilon}\right)$ list queries.

7.3 Preliminary Details and Facts

Throughout, we will use quantum counting and quantum searching methods as key ingredients in the quantum $\{\varepsilon, \delta\}$-FPRAS. Here, we state facts related to these methods, and refer the reader to appropriate sources.

7.3.1 Quantum searching

Consider a database of N items and a function $f : \{0,...,N-1\} \to \{0,1\}$. We wish to find an element i in the database that satisfies $f(i) = 1$.

Fact 2. *There exists a quantum algorithm for finding such a "marked" i which uses an expected* $\Theta(\sqrt{N})$ *applications of* f *[6].*

Henceforth, we will assume the existence of such an algorithm, where the number of marked elements is not required to be known beforehand. Details of this algorithm can be found in Chapter 5, as well as other publications [6, 7].

7.3.2 Quantum counting

Consider the same database of N items and a function $f : \{0,...,N-1\} \to \{0,1\}$. The counting problem requires one to find the size of the pre-image of 1, i.e., the size of the set of elements over which f evaluates to 1. Let the size of this set be t, and define $a = t/N$ as the relative size of the pre-image of 1.

Fact 3. *There exists a quantum algorithm for estimating a by $\widetilde{a}$ such that:*

$$|\widetilde{a}-a| \quad \leq \quad 2\pi k\frac{\sqrt{a(1-a)}}{L}+k^2\frac{\pi^2}{L^2} \tag{7.1}$$

which uses $O(L)$ applications of f. For $k=1$, these bounds hold with probability at least $8/\pi^2$ and for $k\geq 2$, with probability at least $1-1/2(k-1)$ [7].

In the case of $k=1$, one could boost the success probability further by using Chernoff bounding techniques [17]. Thus, $\widetilde{a}$ approximates a within these bounds with probability at least $1-\delta$, using an additional $O(\log(1/\delta))$ repetitions of the counting algorithm. Once again, the variant powering lemma of appendix A.4 is being used here.

For the purposes of the results in this chapter, we will require a uniformized phase estimation algorithm which was described in Chapter 6.

7.3.3 A maximum finding algorithm

Given a list **S**, let T be an implicitly supplied table-lookup function which returns the value of the j^{th} item of **S** on input j. Also, for two numbers $x,y\in\mathbb{R}$, consider the following function $F(x,y)$:

$$F(x,y) \quad = \quad \begin{cases} 1 & \text{if } x<y \\ 0 & \text{otherwise} \end{cases} \tag{7.2}$$

The maximum finding algorithm presented here, follows from the minimum finding algorithm of Dürr and Høyer [14]. We explain how the algorithm intuitively works shortly.

Algorithm(FindMax($\mathcal{A},\chi_{\text{comp}}$))

1: Choose threshold index $0 \leq y \leq N-1$ uniformly at random.
2: **while** Total running time is less than $22.5\sqrt{N} + 1.4\log^2 N$. **do**
3: Initialize memory as $\frac{1}{\sqrt{N}} \sum_j |j\rangle |T[j]\rangle |y\rangle |T[y]\rangle |0\rangle$.
4: Apply the quantum searching algorithm [6] by supplying it with $\mathcal{A}$ and a comparison operator χ_{comp}.
5: **end while**
6: Observe the first register; let y' be the outcome. If $F(T[y], T[y']) = 1$, set $y = y'$ and goto step 2.
7: return y.

Theorem 1. *The algorithm* **FindMax(**$\mathcal{A}, \chi$**)** *finds the index of the maximum value with probability at least* $\frac{1}{2}$ *and requires an expected* $O(\sqrt{N})$ *Grover iterations.*

Proof. The theorem is proven by similar methods that are found in Dürr and Høyer, [14]. □

In **FindMax**, χ_{comp} is a comparison operator, while $\mathcal{A}$ is the operator that creates the state $\frac{1}{\sqrt{N}} \sum_j |j\rangle |T[j]\rangle |y\rangle |T[y]\rangle |0\rangle$ starting from $|0\rangle |0\rangle |y\rangle |T[y]\rangle |0\rangle$. Essentially, this algorithm performs a binary search through the table elements. Step four of the algorithm amplifies the amplitude of all indices whose corresponding table values are greater than $T[y]$, the threshold value. The amplification process requires χ_{comp} to set the last register to 1 for such elements.

Once the amplitude of these elements has been amplified, we measure in step 6 to obtain a new threshold index y'. Assume for a moment that y' is chosen from the set of elements $\{z \mid T[z] > T[y]\}$. The key observation is that y' will be chosen according to a uniform distribution and the expected value of a random variable having a uniform distribution is the median range value. We expect that the algorithm will choose a new threshold index that has the median

table value in $\{z \mid T[z] > T[y]\}$. Thus in every iteration, we expect the algorithm to cut the number of possible threshold indices for the subsequent iteration by an expected factor of 2.

Corollary 1. *There exists a quantum algorithm for finding the statistical mode, equipped with a #P-oracle P, which, given an index i into the set, returns the number of times the i^{th} element of the set occurs in* **S**. *Moreover, this algorithm finds the index of the modal element with probability at least $\frac{1}{2}$ and requires an expected $O(\sqrt{M})$ Grover iterations.*

Proof. The list, in this case, is the set **D** of M elements. We also replace T in **FindMax** with P. Now, the altered **FindMax** gives the index y (into the set **D**) of the modal element. Having this index y, we use an additional table query to learn the modal element.

The algorithm succeeds with probability at least $\frac{1}{2}$ and requires an expected $O(\sqrt{M})$ Grover iterations, and this is clear from Theorem 1. □

7.4 Replacing the Counting Oracle with a Quantum Counter

In this section, we replace the #P-oracle in the algorithm of corollary 1 with a quantum counter. While the replacement is straightforward, the error analysis is not since quantum counters may not give exact counting information. Moreover, to obtain the claimed improvements, we *must* use these counters in a bounded-error fashion since extracting exact information offers no speedup over classical counting methods [2]. Since we deal with bounded error quantum counters, we relax the goal of finding the exact mode and accept finding the *approximate mode* as previously defined. Hence, this is where the notion of an ε-approximation algorithm arises.

7.4.1 Why replacement with a quantum counter is not straight-forward

Since we use quantum counting algorithms in a bounded-error fashion, it follows that any resulting estimated counts will contain some uncertainty. Furthermore, from the maximum finding algorithm in section 7.3.3, we compare counts of all elements to the count of a threshold element, with the eventual goal of selecting a new element whose count is higher than that of the current threshold.

Since estimated counts are inexact, it follows that the count of the threshold element too, will be inexact. Thus, any comparison with the count of the threshold element will be a comparison with an inexact quantity.

To elaborate further, let the threshold element be y, and let its count in the list be a_y. From the counting bounds in fact 3, we have that $|\widetilde{a_y} - a_y| \leq \varepsilon$ with probability at least $8/\pi^2$. For any other element u with count a_u, we have that $|\widetilde{a_u} - a_u| \leq \varepsilon$, also with probability at least $8/\pi^2$. Thus, if $a_u \leq a_y$, then the accuracy bounds imply the following inequality:

$$\widetilde{a_u} - \varepsilon < a_u \leq a_y < \widetilde{a_y} + \varepsilon$$

$$\implies \widetilde{a_u} < \widetilde{a_y} + 2\varepsilon$$

with probability at least $64/\pi^4$. Similarly, if $a_u > a_y + 4\varepsilon$, then we have the following inequality:

$$\widetilde{a_u} + \varepsilon > a_u > a_y + 4\varepsilon > \widetilde{a_y} + 3\varepsilon$$

$$\implies \widetilde{a_u} > \widetilde{a_y} + 2\varepsilon$$

with probability at least $64/\pi^4$. From these inequalities, we have a rule about $\widetilde{a_y} + 2\varepsilon$ such that if the estimates of a_u and a_y are within the stated bounds, then we will be able to correctly compare them with probability at least $64/\pi^4$.

Note that we have no probabilistic guarantees for our comparison when a_u occurs in the range $(a_y, a_y + 4\varepsilon]$. This means that for all elements whose counts fall in this range, we have no indication regarding the outcome of such a comparison. Since $a_u > a_y$ in this range, then ideally we should expect that $\widetilde{a_u} > \widetilde{a_y}$ with high probability. At worst, one might assume that as we get closer to the $a_y + 4\varepsilon$ boundary, our comparisons have a better chance of being correct. A priori however, this is not true since the distributions of $\widetilde{a_y}$ and $\widetilde{a_u}$ are not necessarily monotonic or similarly shaped.

If we do not know anything about the distribution of correct comparisons in this range, then it may lead to problematic cases as the one illustrated in figure 7.1: In (a), the distribution in the $(\widetilde{a_y}, \widetilde{a_y} + 2\varepsilon]$ is peaked around the first $\sqrt{M}$ counts, and thus, we expect to move at most, $\sqrt{M}$ in to this interval which is illustrated in (b). Accordingly, we would need $\sqrt{M}$ rounds to get to the front of the list, i.e., to the modal element, when we would like to do it in $O(\log M)$ rounds, as is done in the maximum finding algorithm.

A priori, we have no reason to believe that such a case will not occur. If we had a monotonic increasing distribution of selection probabilities, then this problem could be avoided. More generally, we can make use of any distribution that has an expected selection probability that is at least the median of selection probabilities in this range. Such a distribution would ensure that we avoid this case by moving through the problematic range quickly.

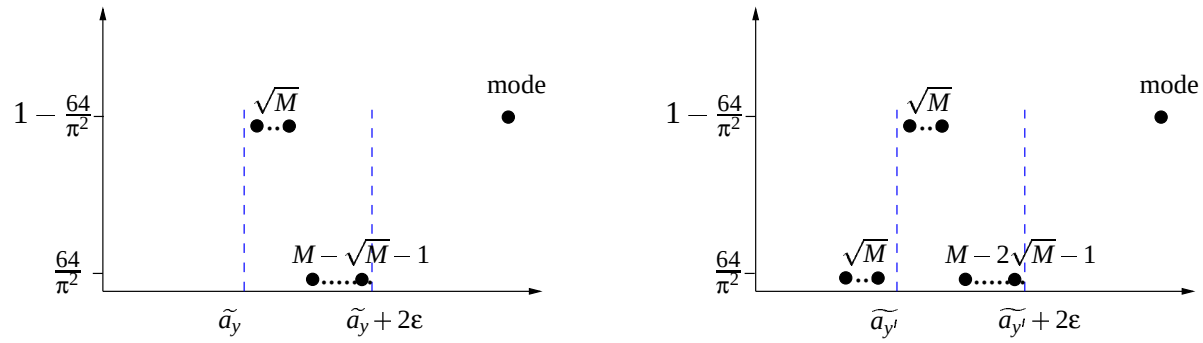


Figure 7.1: A hypothetical distribution of frequency counts. In (a), the distribution in the $(\widetilde{a_y}, \widetilde{a_y}+2\varepsilon]$ is peaked around the first $\sqrt{M}$ counts, and thus, we expect to move at most $\sqrt{M}$ in to this interval which is illustrated in (b). The vertical axes represent the probability of these frequency counts being identified as greater than the frequency of the threshold element. In (b), y' represents the updated threshold index as a result of the previous iteration.

A monotonic distribution is one such useful distribution and the remainder of this section is devoted to showing such an approximate monotonicity result: as we get closer to the $a_y + 4\varepsilon$ boundary, our comparisons have a better chance of being correct up to a small error term. In other words, the selection probabilities increase as we approach the $a_y + 4\varepsilon$ boundary. We attempt to change the distributions for approximate counts so that we may compare them with greater ease; this requires a uniformized version of the phase estimation algorithm which was explained in Chapter 6.

7.4.2 Using the uniformized counting algorithm

For a general Boolean function $f : \{0,\ldots,N-1\} \to \{0,1\}$, the quantum counting algorithm gives an estimate of the quantity $a = t/N$. In our setting, we are interested in the number of times an

element $i \in \{0,\ldots,M-1\}$ occurs in **S**. Thus, our Boolean functions are dependent on i, call them f_i, and for each of the N numbers j in **S**, $f_i(j) = 1$ *iff* the j^{th} element in **S**, $\mathbf{S}[j] = i$. By using these Boolean functions in the quantum counting algorithm, we can find the number of times i occurs in **S** as a fraction of N. We denote this quantity by $C(i)$.

In this sense, we can create a state $\left|\widetilde{C(i)}\right\rangle$ in superposition, for each $i \in \mathbf{D}$. In other words, we start with the state $\sum_{i=0}^{M-1} |i\rangle |0\rangle$ and use the uniformized counting algorithm in quantum parallel to build the state $\sum_{i=0}^{M-1} |i\rangle \left|\widetilde{C(i)}\right\rangle$. This step essentially requires an extra Hadamard (Fourier) transformation on the first register to put it into a superposition of all possible M list elements. Note that $\left|\widetilde{C(i)}\right\rangle = \sum_j \alpha_{i,j} |j\rangle$, i.e., each approximate count is a distribution over possible values of $C(i)$.

Recall that the relativized mode finding algorithm prepared the following state (omitting normalization factors):

$$\sum_{i=0}^{M-1} |i\rangle |P(i)\rangle |y\rangle |P(y)\rangle |F(P(y),P(i))\rangle \tag{7.3}$$

Note that the state $\sum_{i=0}^{M-1} |i\rangle \left|\widetilde{C(i)}\right\rangle$ is really the same as $\sum_{i=0}^{M-1} |i\rangle \left|\widetilde{\frac{P(i)}{N}}\right\rangle$. With a procedure to prepare such a state, we are in a position to replace the oracle. Thus, the eventual algorithm will create the state:

$$\sum_{i=0}^{M-1} |i\rangle \sum_j \alpha_{i,j} |j\rangle |y\rangle |s_y\rangle |F(s_y,j)\rangle \tag{7.4}$$

in place of the state in equation (7.3). Note that s_y is not a superposition of estimates for $C(y)$, but instead, is the measured outcome upon observing $\left|\widetilde{C(y)}\right\rangle$. Also note that F is evaluated on s_y and all possible values of $\widetilde{C(i)}$; this is an example of quantum parallelism.

7.4.3 Correctly computing F

Our goal in each iteration of the algorithm is to find a new element i such that $C(i) > C(y)$, which is achieved by computing F on frequency counts and then selecting some element i for which $F(s_y, \widetilde{C(i)}) = 1$.

Since $\left|\widetilde{C(i)}\right\rangle = \sum_j \alpha_{i,j} |j\rangle$ is a distribution over approximations to $C(i)$, it follows that $\sum_j \alpha_{i,j} |F(s_y, j)\rangle$ will also be a distribution of F-evaluations over possible values of $\widetilde{C(i)}$. Measuring this state will give the evaluation of F on s_y and the best estimate of $C(i)$ with the probabilistic guarantees of the counting algorithm.

As noted above, it is desirable to have similar distributions for all count estimates, and in order to get this, we must use the uniformized version of the quantum phase estimation algorithm. Since the dependence on the true values of the underlying phases is mostly eliminated, it follows that the accuracy of the algorithm can be chosen as desired. Therefore, we choose bounds that look like the standard counting bounds from fact 3. Accordingly, the number of list queries L, should be chosen so that:

$$\left|\widetilde{C(i)} - C(i)\right| \leq \frac{\varepsilon}{4M}$$

for any element $i \in D$. Consider the accuracy bounds from fact 3:

$$\left|\widetilde{C(i)} - C(i)\right| \leq \frac{2\pi k\sqrt{C(i)(1-C(i))}}{L} + \frac{\pi^2 k^2}{L^2}. \tag{7.5}$$

For the modal element w, we know immediately that $1/M \leq C(w) \leq 1$, which implies that $0 \leq \sqrt{C(w)(1-C(w))} \leq \sqrt{1-1/M}$.

With $L = 10\pi M/\varepsilon$ list queries, we approximate the frequency of w within the above bounds ($\varepsilon/4M$) with probability at least $8/\pi^2$, i.e.:

$$\begin{aligned}\left|\widetilde{C(w)} - C(w)\right| &\leq \frac{\varepsilon}{5M} + \frac{\varepsilon^2}{100M^2} \\ &\leq \frac{\varepsilon}{4M}.\end{aligned}$$

We repeat the algorithm $6\log M$ times to boost the constant probability of success to at least $1 - 1/16M^2$. Once again, this is an application of the variant powering lemma in appendix A.4. Note that $4 + 2\log M$ repetitions suffice, but we increase it to $6\log M$ to simplify the analysis.

Thus, the number of list queries required to approximate the frequency of the mode within the $\varepsilon/4M$ of its true value with probability at least $1 - 1/16M^2$ is $60\pi M \log M\varepsilon^{-1}$.

Since the phase estimation algorithm is uniformized, we are guaranteed to approximate the frequency of *any* element i within these bounds with probability at least $1 - 1/16M^2$ by using L list queries.

The unitary U_F which computes F should not mark any element whose frequency is less than that of y. If such an element is marked, then with some non-zero probability, we could select less

frequently occurring elements in successive iterations of the algorithm.

Thus, we require F to be evaluated to 0 for all i such that $C(i) \leq s_y$, and we require F to be evaluated to 1 for all i such that $C(i) > s_y$, both of which must occur with high probability. To understand the reasoning behind this discussion, recall that the goal of the algorithm is to choose more frequently occurring elements in successive iterations. The reason for having the case of equality evaluated to 0 above, is illustrated in figure 7.2.

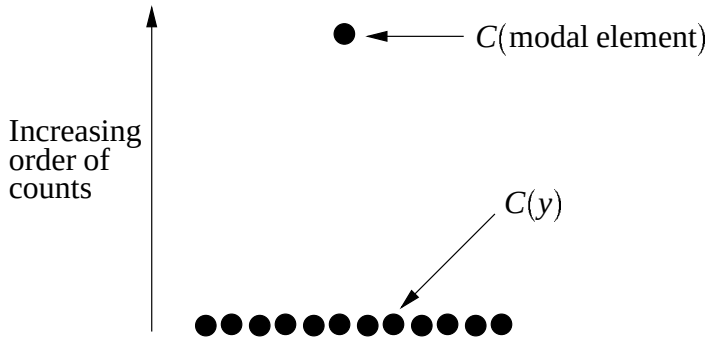


Figure 7.2: A bad scenario for the algorithm: Since the frequencies of many elements are close to that of y, they will all be marked, and with high probability, the algorithm will spend most of its time amongst these "close" elements.

With the approximation bounds in mind for $C(i)$, we consider the case when $C(i) \leq C(y)$. The accuracy bounds imply the following inequality:

$$\widetilde{C(i)} - \varepsilon/4M < C(i) \leq C(y) < s_y + \varepsilon/4M$$

$$\Longrightarrow \widetilde{C(i)} < s_y + \varepsilon/2M.$$

Similarly, if $C(i) > C(y) + \varepsilon/M$, then we have the following inequality:

$$\widetilde{C(i)} + \varepsilon/4M > C(i) > C(y) + \varepsilon/M > s_y + 3\varepsilon/4M$$

$$\implies \widetilde{C(i)} > s_y + \varepsilon/2M.$$

From these inequalities, we have a rule about $s_y + \varepsilon/2M$ such that if used by U_F to compute F, then we will correctly mark elements i such that $C(i) \leq C(y)$ or $C(i) > C(y) + \varepsilon/M$ with probability at least $1 - 1/8M^2$.

In particular, elements i such that $C(i) > C(y) + \varepsilon/M$ are marked with probability at least $1 - 1/8M^2$, i.e., the probability distribution according to which we select any i whose count occurs in this range is lower bounded by the uniform probability function $1 - 1/8M^2$. For elements i such that $C(i) \leq C(y)$, we have an upper bound on the marking probabilities. Namely, an element whose count occurs in this range is marked with probability at most $1/8M^2$.

In the range $C(y) < C(i) \leq C(y) + \varepsilon/M$, we have no guarantees for how F is evaluated on counts for i and y. Furthermore, it would be useful to have a lower bound on the marking probabilities for elements i such that $C(i) \leq C(y) + \varepsilon/M$. This is where the uniformized phase estimation algorithm is essential: For two counts $C(i_1)$ and $C(i_2)$, the standard phase estimation algorithm may return states $\left|\widetilde{\theta_{C(i_1)}}\right\rangle$ and $\left|\widetilde{\theta_{C(i_2)}}\right\rangle$ having different distributions, while the uniformized algorithm will return these states with similar distributions.

The reader will note that this discussion is very similar to that of section 7.4.1. Essentially, we have derived the same arguments as before, except that we have now incorporated the implications of uniformized phase estimation. We now proceed to use the approximate monotonicity

results in Chapter 6.

Assume that $C(i_2) < C(i_1)$. Then by the results in Chapter 6, it follows that:

$$\Pr[\widetilde{C(i_2)} > s_y + \varepsilon/2M] - \frac{3}{M} \quad \leq \quad \Pr[\widetilde{C(i_1)} > s_y + \varepsilon/2M] \tag{7.6}$$

Consider the range $[0, s_y + \varepsilon/M]$, and let the first element whose count occurs in this range be j_1. For every element i such that $C(j_1) < C(i)$, we can consider an inequality of the type in equation (7.6). Thus, $\Pr[\widetilde{C(j_1)} > s_y + \varepsilon/2M] - 3/M$ represents a lower bound on the selection probabilities for all elements occurring in this range.

Let j_2 be the next element in the range such that $\Pr[\widetilde{C(j_2)} > s_y + \varepsilon/2M] > \Pr[\widetilde{C(j_1)} > s_y + \varepsilon/2M]$: We can once again consider an inequality of the type in equation (7.6); thus, for every element i such that $C(j_2) < C(i)$, $\Pr[\widetilde{C(j_2)} > s_y + \varepsilon/2M] - 3/M$ represents a new lower bound on the selection probabilities.

In this way, every new point j_l such that $\Pr[\widetilde{C(j_l)} > s_y + \varepsilon/2M] > \Pr[\widetilde{C(j_{l-1})} > s_y + \varepsilon/2M]$ presents a new lower bound for the selection probabilities of elements i such that $C(j_l) < C(i)$. We can therefore define a function $h(i)$ which is a monotone non-decreasing lower bound function on the selection probabilities in this range. Intuitively, $h(i)$ looks like a step function where the lower bound probability jumps at the new j_l border points. Consider the illustration in figure 7.3

In this sense, the probability function according to which we select an element whose count is in the range $[0, C(y) + \varepsilon/M]$, is lower bounded by a monotone non-decreasing function.

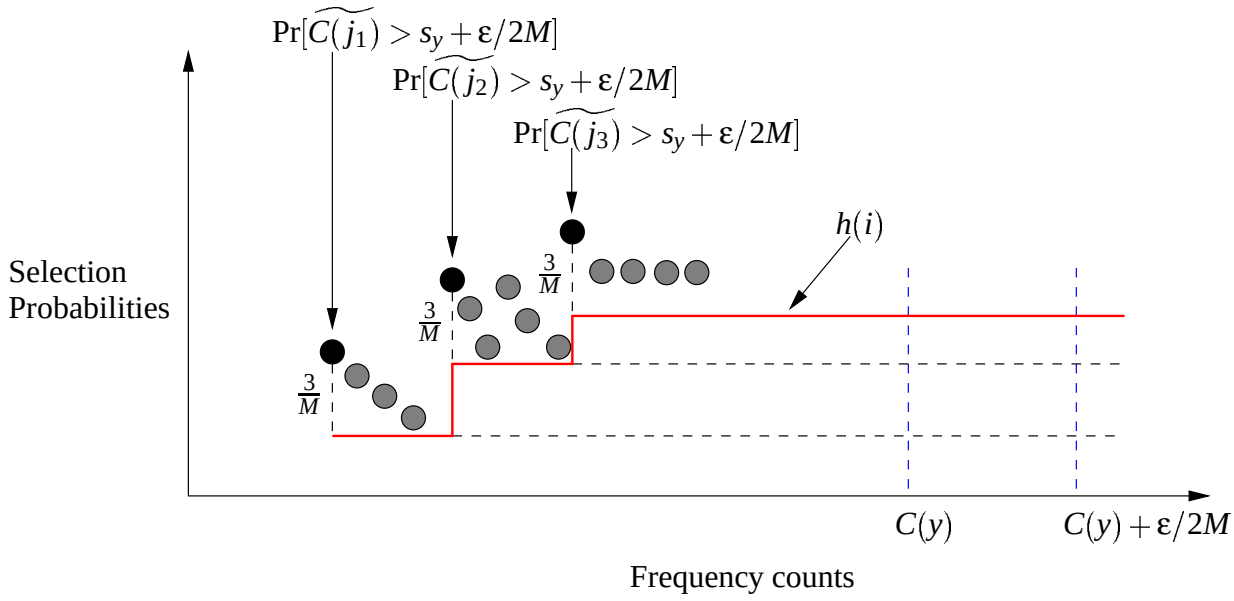


Figure 7.3: An illustrated example of the lower bounding step function $h(i)$.

7.4.4 Counts as ranks

Let the least frequently occurring element be the element of rank M, and let the modal element of the list be the element of rank 1. In any iteration, the algorithm will select a new threshold index from amongst the marked elements. Let the current threshold element y, be of rank m. For each i in the range $[M,1]$, let $f(i)$ be the probability with which i is marked. Also, let $P = \sum_i f(i)$. It follows that the probability with which i is chosen as the next threshold index is $p(i) = f(i)/P$.

Consider $f(i)$ which we decompose as $f(i) = h(i) + g(i)$, such that $h(i)$ is monotone non-decreasing and $0 \leq g(i) \leq 3/M$. Here, $h(i)$ is exactly the lower-bound for the selection probabilities, which justifies the decomposition of f in this way. This decomposition can be substituted into the expression of $p(i)$ to get:

$$\begin{aligned} p(i) &= \frac{f(i)}{P} \\ &= \frac{h(i)}{P} + \frac{g(i)}{P} \end{aligned}$$

Clearly, $h(i)/P$ is still monotone non-decreasing, and and $0 \leq g(i)/P \leq 3/MP$. Since $g(i)$ is non-negative, we can consider the amount of probability weight it contributes to the distribution. An upper bound on the number of elements in this range is clearly M, and thus $\sum_i g(i)/P$ adds probability weight at most $3/P$.

Recall that elements whose counts were at least $C(y) + \varepsilon/M$ were marked with probability at least $1 - 1/8M^2 > 7/8$ for $M > 0$. Thus, assuming that at least one such element exists, we have that $\sum_i h(i)/P \geq 7/8P$.

Consider the following:

$$\sum_i p(i) = \sum_i \frac{h(i)}{P} + \sum_i \frac{g(i)}{P} \tag{7.7}$$

$$= H\left(\frac{\sum_i h(i)}{H}\right) + G\left(\frac{\sum_i g(i)}{G}\right) \tag{7.8}$$

where $H = \sum_i h(i)/P$ and $G = \sum_i g(i)/P$. Here, we have renormalized $\sum_i h(i)/P$ and $\sum_i g(i)/P$ so that we may treat them as separate probability distributions.

For any fixed element i, equation (7.8) says that i is chosen as the next threshold index from either one of two distributions: With probability H, we choose i as the next threshold index according to the distribution $X_h = \sum_i h(i)/H$, while with probability G, we choose it according to

the distribution $X_g = \sum_i g(i)/G$.

We now find a lower bound on the frequency of choosing from X_h. Consider the following inequalities:

$$\frac{7}{8P} \leq H \leq \frac{1}{P} \tag{7.9}$$

and,

$$0 \leq G \leq \frac{3}{P} \tag{7.10}$$

which allows us to conclude that:

$$\frac{H}{H+G} \geq \frac{7}{8P} \cdot \frac{P}{4} \tag{7.11}$$

$$\geq \frac{1}{5} \tag{7.12}$$

and so at worst, we will sample from X_h one in every five times. Note that assuming the existence of at least one element whose count is at least $C(y) + \varepsilon/M$ is a valid assumption, since if such an element does not exist, then by definition, the current threshold index is an approximate mode.

Also note that sampling from X_g is not detrimental, i.e., it only serves to increase the marking probability of i. Assume that each time we sample from X_g, we do nothing; this is clearly worse than sampling from X_h and X_g. In other words, the performance of a procedure that samples from both distributions is at least as good as a procedure that only samples from X_h.

Since X_h has a monotone non-decreasing probability function, it follows that its expected rank is at least that of a uniform distribution. Underlying this fact is that the expected value of a random variable having a monotone non-decreasing probability distribution is at least that of a random variable having a uniform distribution; the proof of this is found in appendix A.5.

Note however, that this does not guarantee forward movement: Formally, let Y be the random variable corresponding to the selection of a new threshold index; let X_1 be the random variable corresponding to the selection of elements from $[M,m]$, and let X_2 correspond to the selection of elements from $[m-1,1]$. Since we are interested in how the rank of the threshold index changes, it follows that X_1 takes on values in $[m-M,0]$ and X_2 takes on values in the range $[1,m-1]$.

Clearly then, $\mathrm{E}[Y] = q\mathrm{E}[X_1] + (1-q)\mathrm{E}[X_2]$. If $\mathrm{E}[X_1] \geq (m-M)/2$ and $\mathrm{E}[X_2] \geq m/2$, then

$$\mathrm{E}[Y] \quad \geq \quad \frac{1}{2}(m - qM)$$

which does not ensure that we move forward. Thus, simply applying F to these counts and selecting according to a single evaluation, may not result in forward movement. To circumvent this, we will amplify the probability of selecting from X_2 by quantum searching, which will ensure that we sample from X_2 with some constant probability. A priori, this does not guarantee forward movement, and thus, other techniques will have to be used to prevent sampling from X_1.

Note that if we were to sample from X_1, the algorithm would be wise not to move. This requires a mechanism by which to know that we have sampled from X_1. Ultimately, we will construct a mechanism such that the expected change in rank is positive with extremely low probability. Thus, with high probability, we either stay in the same place (threshold index does

not change), or we move forward (sample from X_2).

Here, the expected ranks of X_1 and X_2 assumed sampling from a monotone non-decreasing distribution. Since this happens one in five times on average, it follows that the expected value of Y must be re-scaled by $1/5$. So by the arguments above, along with a re-scaling factor, we have that:

$$\mathrm{E}[Y] \quad \geq \quad \frac{1}{10}(m - qM).$$

7.5 The Algorithm

7.5.1 Finding the approximate mode

Let A_{count} be the unitary operator that creates the state in equation (7.4) from section 7.4.2. For simplification, we ignore the extra registers introduced by the uniformized phase estimation algorithm. Thus, A_{count} creates the state:

$$\sum_{i=0}^{M-1} |i\rangle \sum_j \alpha_{i,j} |j\rangle |y\rangle |s_y\rangle |F(s_y, j)\rangle$$

starting from $|0\rangle|0\rangle|y\rangle|s_y\rangle|0\rangle$. We apply a Hadamard or Fourier transform of dimension M to the first register, to create a superposition of all possible M list elements. We then compute the counts $\widetilde{C(i)} = \sum_j \alpha_{i,j} |j\rangle$ in the second register, followed by computing F using U_F in the fifth register. Once again, preparing each count requires $60\pi M \log M \varepsilon^{-1}$ list queries, which will be used in the query complexity analysis of the algorithm.

The following algorithm finds an element whose sampled frequency is an ε-approximation to that of the mode. We consider a list $\mathbf{S}$ of N numbers drawn from a set of M numbers. Note that the required unitary operators A_{count} and χ_{comp} are supplied implicitly.

Algorithm(Find_Mode (ε))

1: Choose a random element y, and compute an estimate s_y of its frequency.
2: **while** total number of iterations is less than $120 \log M$ **do**
3: Initialize memory as $|\Psi\rangle = |0\rangle\,|0\rangle\,|y\rangle\,|s_y\rangle\,|0\rangle$
4: Apply A_{count} with ε/M as the counting accuracy to $|\Psi\rangle$:

$$\sum_{i=0}^{M-1} |i\rangle \sum_j \alpha_{i,j}\,|j\rangle\,|y\rangle\,|s_y\rangle\,|F(s_y,j)\rangle$$

5: Apply the quantum searching algorithm, requiring A_{count}, ε/M, and χ_{comp}.
6: Measure all registers, and let z be the output of the first register, and let s_z be the outcome of the second register.
7: **if** $s_z > s_y + \varepsilon/2M$ **then**
8: set $y = z$ and $s_y = s_z$
9: **end if**
10: **end while**
11: Query the list for the element at index y, and return it as the mode.

Theorem 2. Find_Mode *finds an element whose sampled frequency is within ratio* $1+\varepsilon$ *of the frequency of the mode with probability at least* $1/2$*, using* $O\left(\frac{M^{3/2}\log M}{\varepsilon}\right)$ *list queries.*

Proof. Initially, a threshold element y is chosen at random. From this point onwards, the algorithm attempts to find a new threshold element whose frequency is higher than that of y; this is

one iteration of the algorithm, and it continues in this way until it has found some element whose frequency is an ε-approximation to that of the modal element.

To prove this theorem, we study the random variable T, which counts the number of iterations of the algorithm. Subsequently, we will show that with high probability T does not deviate far from its expected value, which will be shown to be $O(\log M)$. Having a bound on T, we will then calculate the total number of list queries by studying the amount of work involved in one iteration.

Consider step 6 of **Find_Mode** , where we select a new threshold index element. From section 7.4.3, we choose z from one two regions: $\{z \mid C(z) \leq C(y)\}$ or $\{z \mid C(z) > C(y)\}$.

Let the element corresponding to the lowest count be the element of rank M. Similarly, let the rank of the modal element be 1; the rank of the next threshold element will be in $[M,1]$.

Define the random variable X_1 as taking on values in $[m-M,0]$ and define X_2 as taking on values in $[1,m-1]$. Here, X_1 and X_2 are random variables defined on the ranks of the elements about the threshold index y. Thus, when we choose a new threshold index, its rank will either be in the range of $m-X_1$ or $m-X_2$.

Note that quantum searching amplifies the amplitude of selecting an element whose rank is in X_2. In particular, with probability at least $3/4$, we select an element from X_2, which can be achieved by repeating the searching algorithm $4/3$ times on average.[1] In the formalism of sec-

[1]This is an application of Markov's rule.

tion 7.4.3, the expected rank of the selected element in step 9 is $\mathrm{E}[Y] = (1/4)\mathrm{E}[X_1] + (3/4)\mathrm{E}[X_2]$, which implies that $\mathrm{E}[Y] \geq (1/10) \cdot (m - M/4)$. Obviously, this does not guarantee that we move forward.

To ensure forward movement with high probability, we test the chosen element z to ensure that it occurs more frequently than y. This is done by measuring a good quality count for z and testing it with $s_y + \varepsilon/2M$. Accordingly, if z occurs less frequently than y, then the threshold index remains y, and we start a new iteration of the algorithm.

Note that post-amplification measurement and the test give four possible cases: (1) we measure from X_2 and pass the test; (2) we measure from X_2 and fail the test; (3) we measure from X_1 and pass the test; or (4) we measure from X_1 and fail the test.

Consider once again, the random variable $Y = (1/4)X_1 + (3/4)X_2$. It is clear that $E[Y]$ is not zero in cases 1 and 3. These events happen with probability $(3/4) \cdot (1 - 1/8M^2)$ and $1/32M^2$ respectively. Using the lower bounds on $\mathrm{E}[X_1]$ and $\mathrm{E}[X_2]$ from section 7.4.3, it follows that:

$$\mathrm{E}[Y] \geq \frac{1}{32M^2}\frac{m}{20} + \frac{3}{4}\left(1 - \frac{1}{8M^2}\right)\frac{m-M}{20} \tag{7.13}$$

$$\mathrm{E}[Y] \geq \frac{m}{60}. \tag{7.14}$$

It follows that the rank of the next threshold index is therefore $m - Y$, and that the expected next rank is a constant fraction of the current threshold element's rank. In essence, the algorithm trims the problem size down by some constant factor in each iteration; the expected decay is $\mathrm{E}[Y]$, which is non-negative.

We now turn our attention to the random variable T, which counts the number of iterations of the algorithm before the threshold index is of rank 1. To analyze $\mathrm{E}[T]$, we use a probabilistic recurrence [18, 22]. The decay of subsequent problem sizes is modeled by a particle that starts at position n and proceeds to position 1. If the current position is m, then the next position is $m - Y$ where Y is a random variable (as above) defined on $[m - M, m - 1]$. All that we know about Y is that $\mathrm{E}[Y] \geq g(x)$, where g is a monotone non-decreasing function. Consider the random variable T in this setting, which denotes the number of steps before the particle reaches position 1.

Theorem 3 (adapted from [18, 22]). *Let T be a random variable denoting the number of steps before a particle which starts at position n, reaches position 1. If $\mathrm{E}[Y] > g(m) > 0$, then*

$$\mathrm{E}[T] \leq \int_1^n \frac{dx}{g(x)}$$

where Y is the random variable denoting the particle's change in position.

Proof. See appendix A.6 □

Using theorem 3, we can upper bound the expected number of iterations for the algorithm to reach the front of the list of counts. Analogous to the theorem, the position of the particle is the size of subsequent subproblems, and the random variable Y is the expected decay, as described above. Note that this is slightly misleading in that we will not start at the element of rank M with high probability. However, treating the problem as such re-enforces the upper-bound on $\mathrm{E}[T]$.

From above, we have that $\mathrm{E}[Y] \geq m/60$, and therefore, we set $g(x) = x/60$, which is clearly monotone non-decreasing when x is increasing. It follows by theorem 3 that $\mathrm{E}[T] \leq 60 \log M$.

Note that in theorem 3, there were no restrictions on the range of Y . In the context of this algorithm, some elements (accordingly some ranks) may not be reachable because they fall within

the $\varepsilon/2M$ window of the current threshold index. This is not a problem, so long as there are elements having frequency greater than $s_y + \varepsilon/2M$.

This problem arises when the frequency of y is within $\varepsilon/2M$ of the mode. In this case however, y is an approximate mode by definition. In other words, there may be some problem instances where we may not hit the element of rank 1. However, we will certainly end up on an element whose frequency is an ε-approximation to that of the mode. Note that if the next-to-modal element occurs at least $\varepsilon/2M$ less frequently than the mode, then there is a good chance that we will actually land on the mode. In other words, depending on the frequencies of the elements, we may or may not find the true mode.

In the end, the threshold element y will be declared as the mode of the list. By the above arguments:

$$\left|s_y - C(w)\right| \leq \frac{\varepsilon}{2M}$$

We also know from the bounds on counting that:

$$\left|s_y - C(y)\right| \leq \frac{\varepsilon}{4M}$$

thus,

$$\left|s_y - C(w)\right| \leq \frac{3\varepsilon}{4M}$$

Since $C(w) \geq 1/M$, it follows that this bound can be converted into a relative error bound by dividing the inequality by $C(w)$:

$$\begin{aligned} s_y &\leq \left(1+\frac{3\varepsilon}{4}\right) C(w) \\ &\leq (1+\varepsilon) C(w) \end{aligned}$$

Thus, y is an approximate mode, and its sampled frequency is an ε-approximation to that of the mode.

Next, we consider the number of list queries used to find y. At the start of this section, we noted that A_{count} was the unitary operator that created the state:

$$\sum_{i=0}^{M-1} |i\rangle \sum_j \alpha_{i,j} |j\rangle |y\rangle |s_y\rangle |F(s_y, j)\rangle$$

Furthermore, A_{count} used $60\pi M \log M \varepsilon^{-1}$ list queries in creating this state.

Quantum mechanically searching in a list of size K with t marked elements requires $O(\sqrt{K/t})$ applications of such a unitary operator to succeed with probability at least $1/2$ [6]. Thus, in one iteration of the algorithm, we require:

$$\sqrt{\frac{K}{t}} \cdot \frac{60M \log M}{\varepsilon}$$

list queries. Since the size of the list decreases by an expected constant factor in each iteration, it follows that the total amount of list queries is given by the following sum:

$$\begin{aligned}\sum_{i=0}^{60\log M-1}\sqrt{30\cdot 2^i}\cdot\frac{60M\log M}{\varepsilon} &= \frac{60M\log M}{\varepsilon}\cdot\sum_{i=0}^{60\log M-1}\sqrt{30\cdot 2^i}\\ &\leq \frac{60M\log M}{\varepsilon}\cdot 60\sqrt{M}\\ &\in O\left(\frac{M^{3/2}\log M}{\varepsilon}\right)\end{aligned}$$

Overall, this process requires $O(M^{3/2}\log M\varepsilon^{-1})$ list queries.

Finally, we note that by Markov's inequality, $\Pr(T \leq 2\mathrm{E}[T]) \geq 1/2$ which completes the proof.

□

7.5.2 Amplifying the success probability

Here, we boost the success probability of **Find_Mode** by Chernoff methods. We wish to find an approximate mode, having a sampled frequency which is within ratio $(1+\varepsilon)$ of the frequency of the mode, with probability at least $1-\delta$.

Theorem 4. *There exists a quantum* $\{\varepsilon,\delta\}$*-approximation scheme for mode finding that approximates the frequency of the mode within ratio* $1+\varepsilon$ *with probability at least* $1-\delta$*, using* $O\left(\frac{M^{3/2}\log M}{\varepsilon}\cdot\log(1/\delta)\right)$ *list queries.*

Proof. The procedure consists of repeating **Find_Mode** a total of $O(\log(1/\delta))$ times, and declaring the approximated frequency of the modal element as the *median* of the approximations returned by the independent runs.

This is an application of the powering lemma for ε-approximation schemes found in Jerrum et al. [17] – see appendix A.4. □

7.6 Concluding Remarks

We have described a quantum $\{\varepsilon, \delta\}$-FPRAS for finding the mode of a list **S** of N numbers drawn from a set **D** of M numbers. Compared to classical Monte Carlo sampling, the quantum method gives a quadratic reduction in $1/\varepsilon$, and a less-than-quadratic reduction in M, in the number of required list queries - compared to the classical schemes presented here. Recall that these complexities were $O\left(M^3/\delta\varepsilon^2\right)$ and $O\left(M^3\log(M/\delta)\varepsilon^{-2}\right)$ respectively.

The evaluation of F on approximate counts leads to a distribution over the outputs 0 and 1, i.e., for some element z with $C(z) > s_y$, there is always some non-zero probability of observing a 0 in the evaluation of F on $\widetilde{C(z)}$ and s_y. In this sense, we are dealing with bounded-error inputs. Searching over such bounded error inputs can lead to an incorrect answer which was circumvented by repeating the quantum search $O(\log M)$ times.

By treating the inputs as bounded-error inputs, and using the methods of Høyer et al. [16], we can eliminate this logarithmic factor. Thus, in the end, we would require $O\left(M^{3/2}\log(1/\delta)\varepsilon^{-1}\right)$ list queries which is a true quadratic reduction in M.

Chapter 8

Approximate Counting by Almost Uniform Generation and Quantum Methods

8.1 Introduction

In this chapter, we review the results of Jerrum, Valiant, and Vazirani [17], and we quantize their FPRAS for approximately counting the number of solutions to a combinatorial problem. Sampling from a postulated *almost-uniform generator* is integral in approximating the number of elements y such that $\langle x, y \rangle \in R$, where y is a solution to a problem instance x and R is a relation, and as such, we use quantum sampling and quantum counting where possible, to reduce the number of queries to the almost-uniform generator.

8.2 The Results of Jerrum, Valiant, and Vazirani

Jerrum et al. [17] consider the problems of *uniform generation* and *counting* applied generally to combinatorial problems. Consider a relation R that assigns to each problem instance $x \in \Sigma^*$, a

set of solutions $\{y \in \Sigma^* | \langle x,y \rangle \in R\}$. For some problem instance x, the counting problem related to x asks for the size of the set $R(x) = \{y \in \Sigma^* \mid \langle x,y \rangle \in R\}$, i.e. the number of solutions to x. The uniform generation problem for x requires one to generate a y chosen uniformly at random from $R(x)$.

In addition to these results, Jerrum et al. reduce uniform generation to approximate counting. Thus being able to count approximately is sufficient to generate uniformly. Note that such an approximate counting scheme is within the prescribed approximation bounds with certainty. The authors also note that reducing approximate counting to uniform generation is not possible since the randomness in the uniform generator can not guarantee the deterministic counting approximation bounds with certainty, i.e., there is always some non-zero probability that such a scheme will fail to approximate the required count to within the prescribed bounds due to the randomness of the uniform generator.

Finally, the authors relax the definition of approximate counting and consider a randomized approach; in this sense, a randomized scheme for approximate counting is guaranteed to be within the prescribed approximation bounds *most of the time*. With the deterministic approximation accuracy requirement removed, the problems of *randomized approximate counting* and *almost uniform generation* are now inter-reducible.

8.3 Definitions and Notation

Definition 13. *Let Σ be an alphabet. A relation $R \subseteq \Sigma^* \times \Sigma^*$ is a p-relation if:*

1. *there exists a polynomial p such that $\langle x,y \rangle \in R \Rightarrow |y| \leq p(|x|)$*
2. *$\langle x,y \rangle \in R$ can be tested in deterministic polynomial time*

The first part of the definition limits the length of the solution y to a polynomial in the size of the input, while the second part requires y to be efficiently verifiable.

Let x be a problem instance and $R(x)$ be the set of solutions to x. As an example, the DNF-SAT problem admits a p-relation: Given a Boolean formula F in disjunctive normal form (DNF) over variables $x_1, \ldots, x_n$, we are required to find a set of satisfying assignments $c_1, \ldots, c_n$ to the variables such that F evaluated at these assignment substitutions evaluates to 1 (true). Clearly, for any pair $\langle x, y \rangle \in R$, $|x| = |y|$ and this can be tested in deterministic polynomial time by implementing the Boolean circuit encoding F on a deterministic Turing machine.

In particular, we are interested in self-reducible relations as they have a structure that will allow us to recursively reduce a problem in the following sense: If we have a problem instance x and a partial solution w for x, then we may incorporate this partial solution and derive a new instance for the same problem, but which may be smaller than the original instance. Note that the DNF-SAT problem is a self-reducible problem. Consider the following, more formal definition from [17]:

Definition 14. *A relation $R \subseteq \Sigma^* \times \Sigma^*$ is* self-reducible *iff*

1. *there exists a polynomial time computable function $g \in \Sigma^* \to \mathbb{N}$ such that $\langle x, y \rangle \in R \Rightarrow |y| = g(x)$*
2. *there exist polynomial time computable functions $\psi \in \Sigma^* \times \Sigma^* \to \Sigma^*$ and $\sigma \in \Sigma^* \to \mathbb{N}$ satisfying*
   (a) $\sigma(x) = O(\log |x|)$,
   (b) $g(x) > 0 \Rightarrow \sigma(x) > 0 \ \ \forall x \in \Sigma^*$,

(c) $|\psi(x,w)| \leq |x| \ \ \forall x,w \in \Sigma^*$

such that, for all $x \in \Sigma^*$, $y = y_1 \ldots y_n \in \Sigma^*$,

$\langle x, y_1 \ldots y_n \rangle \in R \ \Leftrightarrow \ \langle \psi(x, y_1 \ldots y_{\sigma(x)}), y_{\sigma(x)+1} \ldots y_n \rangle \in R.$

Part one of this definition makes the polynomial relation between a solution and instance concrete, i.e., one can efficiently compute the length of a solution to a given problem instance. In the case of DNF-SAT, if F is a formula on n variables, then $g(x)$ for any such instance x is the constant function n, i.e., $g(x) = n$ for any such instance x.

Part two of this definition formalizes the meaning of self-reducibility. The most important of the two functions is ψ, which given an instance and corresponding partial solution, will return the self-reduced problem instance. In the case of DNF-SAT, for any instance x and partial assignment w, we could replace the assigned variables according to w by logical literals and be left with a reduced DNF-SAT instance x' on a smaller number of variables. This notion is succinctly conveyed by the following part of the definition of self-reducibility: $\langle x, y_1 \ldots y_n \rangle \in R \ \Leftrightarrow \ \langle \psi(x, y_1 \ldots y_{\sigma(x)}), y_{\sigma(x)+1} \ldots y_n \rangle \in R.$

One can consider the general counting function $N_R(x) = |\{y \in \Sigma^* : \langle x, y \rangle \in R\}|$ as the number of solutions to the instance x. Consider the extended definition of the counting function which can also count partial solutions in the following sense:

Definition 15. *The extension counting function* $Ext_R(x,w) = |\{z \in \Sigma^* : \langle x, wz \rangle \in R\}|$ *counts the number of solutions to x that are prefixed by w.*

The extension counting function was defined in Jerrum et al. [17], and will be used in the FPRAS for approximate counting. Finally, consider the following definition of *almost uniform generation* which will also be used in subsequent sections:

Definition 16. *A probabilistic Turing machine M is an* almost uniform generator *for the relation $R \subseteq \Sigma^* \times \Sigma^*$ iff*

1. *there exists a function $\varphi \in \Sigma^* \to (0,1]$ such that, for all inputs $\langle x, \varepsilon \rangle \in \Sigma^* \times \mathbb{R}^+$ to M and for all words $y \in \Sigma^*$,*

   (a) $\langle x,y \rangle \notin R \Rightarrow \Pr(M \text{ outputs } y) = 0$,

   (b) $\langle x,y \rangle \in R \Rightarrow (1+\varepsilon)^{-1}\varphi(x) \leq \Pr(M \text{ outputs } y) \leq (1+\varepsilon)\varphi(x)$

2. *for all inputs $\langle x, \varepsilon \rangle$ such that $\{y \in \Sigma^* : xRy\}$ is nonempty,* $\Pr(M \text{ accepts } \langle x, \varepsilon \rangle) \geq \frac{1}{2}$.

The second part of this definition implies that with probability at least $1/2$, the probabilistic Turing machine M will return a value y such that $\langle x,y \rangle \in R$.

8.4 Randomized Approximate Counting

In this section, we review the fully polynomial randomized approximation scheme (FPRAS) of Jerrum et al., that counts the number of solutions to a combinatorial problem. As before, R is a p-relation with problem instance x that has a corresponding set of solutions $\{y \mid \langle x,y \rangle \in R\}$, such that $N_R(x) = |\{y \mid \langle x,y \rangle \in R\}|$. For the definition of a FPRAS, see Chapter 7.

8.4.1 Approximating $N_R(x)$

The first ingredient is a simple consequence of sampling: From a population P of size N, assume we wish to know how many members satisfy a certain property Q, and let this quantity be N_Q. We can estimate the ratio N_Q/N by uniformly sampling t elements from P and determining the number of sampled members that satisfy Q. Note that when sampling from P, we sample a Q

satisfying member with probability N_Q/N. Thus in a sample of t members, we expect tN_Q/N satisfying samples, and so the required ratio of the sample is N_Q/N. If we sample almost uniformly, then we approximate N_Q/N within the bounds prescribed by our almost uniform sampling procedure.

Let the problem instance of interest be x_0. The second ingredient is the self-reducible property of x_0, i.e. given some partial solution w_0 of a full solution $y = w_0 z$, we can incorporate w_0 into x_0 and construct a smaller-sized problem instance x_1. Recall from the definition of self-reducibility, that x_1 is constructed by the postulated function $\psi \in \Sigma^* \times \Sigma^* \to \Sigma^*$ where $x_1 = \psi(x_0, w_0)$. Also, recall that $Ext_R(x_0, w_0)$ denotes the number of solutions to x_0 that are prefixed by w_0.

We use the sampling technique to estimate the quantity $N_R(x_0)/Ext_R(x_0, w_0)$. Note that equivalently, $Ext_R(x_0, w_0)$ can be expressed as $N_R(\psi(x_0, w_0)) = N_R(x_1)$, and so, we are actually estimating the quantity $N_R(x_0)/N_R(x_1)$. Recursively, we estimate the quantity $N_R(x_1)/Ext_R(x_1, w_1)$ and multiply the two estimates together. We now have an estimate of $N_R(x_0)/Ext_R(x_1, w_1)$. Continuing in this fashion, we accumulate a series of terms that we multiply together, and in the end, we are left with an estimate of $N_R(x_0)$ since:

$$\frac{N_R(x_0)}{N_R(x_1)} \cdot \frac{N_R(x_1)}{N_R(x_2)} \cdots \frac{N_R(x_{n-2}}{N_R(x_{n-1})} \cdot \frac{N_R(x_{n-1}}{N_R(x_n)} \quad = \quad \frac{N_R(x_0)}{N_R(x_n)} \quad = \quad N_R(x_0).$$

We now explain why $N_R(x_n) = 1$: For an instance x with partial solution w, the solutions to the *reduced instance* $\psi(x, w)$ are exactly those strings which when concatenated with w, form a solution to x [17]. Recall that in the last round of sampling, we have completely specified the solution to x which was to have a set length of $g(x)$. Thus, once the solution has been specified, the only possible string that can be concatenated to the solution is the empty string (since otherwise the solution would be longer than $g(x)$). This implies that on the last round of sampling,

$N_R(x_n) = 1$ indicating that the single solution is the empty string.

In order for this scheme to be an FPRAS, it must approximate $N_R(x)$ within ratio $(1+\varepsilon)$ for some chosen ε, and it must do so with probability at least $3/4$.

Finally, we note that the prefix which maximizes $Ext_R(x,w)$ will be guaranteed to be an above average prefix whose frequency is lower bounded by $N_R(x)/2^{\sigma(x)}$. The more frequent the prefix, the higher the accuracy of the method, and so the *modal* prefix intuitively should work the best. Thus we aim to find w, the modal prefix in the set of solutions to x, and therefore a sampling scheme that estimates the frequency of the most commonly occurring prefix within ratio $(1+\varepsilon)$ would be an ε-approximation algorithm for mode finding.

8.4.2 An FPRAS for approximate counting

Here, we give the FPRAS for approximate counting of Jerrum et al., which implements the intuitive scheme described in the preceding section. Their scheme assumes the existence of an almost uniform generator $\mathcal{G}$ which takes as input, $\langle x, \varepsilon \rangle$ where ε is the fixed error tolerance desired in the counting scheme.

The constant c in the algorithm depends on the size of $R(x)$ and is chosen so that $2^{\sigma(x)} \leq |x|^c$. Recall that $\sigma(x)$ is from the definition of self-reducibility and is defined as $\sigma(x) = O(\log|x|)$. The variable $m = g(x)$ is an upper bound on the size of any solution for x. The function $g(x)$ is a polynomial time computable function assumed to exist by the definition of self-reducibility (definition 14) and gives the size of any solution to x; clearly, $m = g(x)$ is a valid choice for m.

1: $\Pi = 1$
2: $t := 180|x|^{3c}m^3/\varepsilon^2$
3: **while** $g(x) > 0$ **do**
4: make $3t$ calls to $\mathcal{G}$ with input $\langle x, \varepsilon/11m \rangle$
5: **if** at least t of the $3t$ trials yield an output **then**
6: let $S = \{y_1, ..., y_t\}$ be the first t outputs of $\mathcal{G}$
7: **else**
8: return 0
9: **end if**
10: Let $w \in \Sigma^{\sigma(x)}$ be a most commonly occurring prefix in S of length $\sigma(x)$
11: $\alpha := |y \in S : w$ is an initial segment of $y|/|S|$
12: $x := \psi(x, w)$
13: $\Pi := \Pi/\alpha$
14: **end while**
15: output Π

It was shown by Jerrum et al., that with probability at least $3/4$, this scheme approximates $N_R(x)$ within ratio $(1+\varepsilon)$. The key step in the proof is to show that each iteration of the loop estimates the ratio $N_R(x)/Ext_R(x,w)$ within ratio $(1+\varepsilon/2m)$. Since there are at most m loop iterations, it follows that $N_R(x)$ (for the original instance x) is estimated within ratio $(1+\varepsilon/2m)^m$ which is less than $1+\varepsilon$ for $0 < \varepsilon < 1$.

With this choice for t, it follows that the frequency X_u of any prefix u in the sample is within $\varepsilon/6|x|^c m$ of its frequency in the population i.e., its mean value, with probability at least $1 - 1/5|x|^c m$. Since this analysis is generic for any u, it therefore also covers the modal prefix w.

Note however, that the close proximity of X_u to its mean value is a probabilistic event, and so, there is some (small) chance that the frequency of some u might deviate considerably far from its mean in any particular sample. The event in which the frequencies of *all* prefixes u are close to their mean values occurs with probability at least $1 - 1/5m$ since there are at most $|x|^c$ possible prefixes. From this fact, it follows that with high probability, the frequency of the modal element of the sample will be within ratio $(1+\varepsilon)$ of the frequency of the modal element of the population.

Note that we have a lower bound on the frequency of w in the sample. In particular, α is guaranteed to be at least $|x|^{-c}$, and so, the absolute error bound of $\varepsilon/6|x|^c m$ can be converted to a relative error bound such that α approximates the frequency of w in $N_R(x)$ within ratio $(1+\varepsilon/5m)$ with probability at least $1 - 1/5m$.

Finally, they show that the probability with which this sampling procedure succeeds on every round is at least $59/60$. Furthermore, the probability that $N_R(x)/Ext_R(x,w)$ is approximated within ratio $(1+\varepsilon/2m)$ on every iteration is at least $(1-1/5m)^m > 4/5$. It follows that the probability of both events happening is at least $59/60 \cdot 4/5 > 3/4$.

The reader will note that the randomized approximation scheme for estimating the frequency of w is exactly that which was described in Chapter 7, where we showed that $O(M^3\varepsilon_q^{-2}\delta^{-1})$ generator queries (list samples) were used to estimate the frequency of the mode within ratio $(1+\varepsilon_q)$ with probability at least $1-\delta$. In this setting, the frequency of the modal prefix w is being approximated to within $(1+\varepsilon/5m)$ with probability at least $1 - 1/5m$. Thus, $\delta = 1/5m$ and $\varepsilon_q = \varepsilon/5m$. Furthermore, $M = |x|^c$, and so it follows that $O(|x|^{3c}m^3\varepsilon^{-2})$ generator queries are required in one iteration of the algorithm. Note that this matches the variable t, the number

of samples required in the approximation scheme of Jerrum et al.

We now replace the randomized approximation scheme for estimating the frequency of w with the $\{\varepsilon,\delta\}$-FPRAS for mode finding of Chapter 7. Informally, the quantum FPRAS reduces the number of queries by an almost-quadratic factor in $|x|^c$ and a quadratic factor in $1/\varepsilon$. Furthermore, it has an exponential reduction in $1/\delta$, over the classical FPRAS *used here*. The reader should note that there exists a classical approximation scheme that will also have this exponential reduction, namely the one presented in Chapter 7. It may, however, require an increased number of samples in other parameters such as $|x|^c$.

Overall, replacing the randomized approximation scheme with the quantum $\{\varepsilon,\delta\}$-FPRAS will result in a quadratic reduction in $1/\varepsilon$ and an almost-quadratic-reduction in $|x|^c$ and m in the FPRAS for estimating $N_R(x)$. In particular, the scheme of Jerrum et al. uses $O\left(|x|^{3c}m^4/\varepsilon^2\right)$ queries, and we will construct a quantum FPRAS that will use $O\left(|x|^{3c/2}\log|x|^c m^2\log m/\varepsilon\right)$

8.5 A Quantum FPRAS for Counting

Given a list **S** of N integers drawn from a set **D** of M integers, the quantum FPRAS finds an approximate mode an element whose frequency is an ε_q-approximation to the that of the mode with probability at least $1-\delta$, using $O\left(\frac{M^{3/2}\log M}{\varepsilon_q}\cdot\log(1/\delta)\right)$ list queries. In this setting, list queries are exactly the outputs of an almost uniform generator. Furthermore, M, ε_q, and δ, all have the same meaning as above.

The algorithm is not limited to integers and can easily be adapted to other domains. In par-

ticular, one can consider a list of k-length strings drawn from Σ^k, where the quantum procedure finds the most commonly occurring k-length string. We will use this algorithm to find w and an estimate of its frequency in $R(x)$.

8.5.1 Counting by querying the generator

Since the generator is a randomized process, it requires a string of random bits each time it is used, and so repeatedly querying the generator with the same parameters including the same string of random bits results in identical outputs. Therefore querying the generator for the j^{th} item, corresponds to querying it with the j^{th} lexicographically occurring random bit string, or simply just the binary number j.

The counting algorithm is the application of the phase estimation algorithm to a suitably chosen Grover iterate [7]. Recall that the Grover iterate is:

$$\mathbf{G} \quad = \quad -\mathbf{A}\mathbf{S}_0\mathbf{A}^{-1}\mathbf{S}_\chi \tag{8.1}$$

and is applied to the state $A|0\rangle = \sum_j |j\rangle$. The operator $\mathbf{S}_\chi$ flips the amplitude of all states $|j\rangle$ for which $\chi(j) = 1$. Thus, all such items j are considered to be marked. For any given prefix i, we wish to count the number of solutions prefixed by i. This is done by choosing $\chi = f_i$ where the Boolean function $f_i(j) = 1$ *iff* the generator's output under the j^{th} random bit string is prefixed by i.

Essentially, the generator is queried for the j^{th} item which is produced by the j^{th} random bit string and $\mathbf{S}_{f_i}$ checks if it is prefixed by i. Each time we use the operator $\mathbf{S}_{f_i}$, we are querying

the generator. Equivalently, each time we apply **G**, we are querying the generator. Note that the mechanics of the counting algorithm have not changed; we have only specialized the operation of $\mathbf{S}_\chi$.

The $\{\varepsilon, \delta\}$-FPRAS for mode finding will use this counting procedure by creating a superposition of all possible prefixes of a fixed (logarithmic) length, and for each prefix i, counting the number of bit strings j that lead the generator to an i-prefixed solution.

8.5.2 Sufficient statistics

As stated above, we will prepare a superposition of all possible prefixes, and for each prefix i , we will prepare its count, i.e., the number of bit strings j that lead the generator to a solution prefixed by i. The count will be over the entire set of solutions, and not a small sample as is done classically [17]. This is possible because quantum mechanically, we can use the principle of superposition to gather statistics about the *entire population* as opposed to a small sample.

Note that such a statistic would be slightly misleading, since there are many bit strings that would cause the generator to not output any solution. In particular, this would be a ratio of the number of bit strings that lead to an i-prefixed solution (denoted s_i) to the total number of bit strings.

Let the set of all random bit strings be H, and let the set of random bit strings that lead the generator to produce *any valid output* be H_g. Similarly, let $H_b = H - H_g$. The statistic mentioned above will then be $s_i/|H|$, which we can appropriately re-scale by multiplying it with $|H|/|H_g|$. The resulting statistic will now be a ratio of the number of bit strings that lead to a i-prefixed

solution (s_i) to the number of *good bit strings*. Note, however, that each valid solution may have many random bit strings that cause the generator to produce it, and so it still is not clear that this statistic is useful.

We can use the fact that the generator is approximately uniform to see that each valid solution will be produced by roughly the same number of good bit strings.

Claim 1. *Assume the existence of an almost uniform generator $\mathcal{G}$ as stated in definition 16. Let the set of solutions be $R(x)$ with size $N_R(x)$. For a fixed prefix i, let the number of times i occurs as a prefix in $R(x)$ be $Ext_R(x,i)$. Then the quantity $\frac{s_i}{|H_g|}$ approximates $\frac{Ext_R(x,i)}{N_R(x)}$ to within ratio $(1+\varepsilon/11m)^2$.*

Proof. Note that since the generator is approximately uniform, each solution is "represented" by roughly the same number of random bit strings in H_g. In particular, from the definition of approximate uniform generation we have that for any solution y,

$$(1+\varepsilon/11m)^{-1}\varphi(x) \leq \quad \Pr(M \text{ outputs } y) \quad \leq (1+\varepsilon/11m)\varphi(x)$$

where $0 < \varepsilon < 1$ is the error tolerance supplied to the counting procedure.

Immediately, we know that the number of random bit strings that produce y is at least $(1+\varepsilon/11m)^{-1}\varphi(x)|H_g|$ and at most $(1+\varepsilon/11m)\varphi(x)|H_g|$. It follows that for any prefix i:

$$(1+\varepsilon/11m)^{-1}\varphi(x)\cdot|H_g|\cdot Ext_R(x,i) \leq \quad s_i \quad \leq (1+\varepsilon/11m)\varphi(x)\cdot|H_g|\cdot Ext_R(x,i) \qquad (8.2)$$

and

$$(1+\varepsilon/11m)^{-1}\varphi(x)\cdot|H_g|\cdot N_{R(x)} \leq \quad |H_g| \quad \leq (1+\varepsilon/11m)\varphi(x)\cdot|H_g|\cdot N_{R(x)}. \tag{8.3}$$

Thus,

$$(1+\varepsilon/11m)^{-2}\frac{Ext_R(x,i)}{N_{R(x)}} \leq \quad \frac{s_i}{|H_g|} \quad \leq (1+\varepsilon/11m)^2\frac{Ext_R(x,i)}{N_{R(x)}} \tag{8.4}$$

□

which shows that $s_i/|H_g|$ approximates $Ext_R(x,i)/N_{R(x)}$ sufficiently well for our purposes.

Since we do not know $|H_g|$, we can approximate it by quantum counting. Recall that in the definition of an almost uniform generator, of all possible random bit strings required, at least $1/2$ lead to a solution; this is evident from part 2 of the definition. Consider a uniform superposition of all possible random bit strings:

$$\sum_j |j\rangle \quad = \quad \sqrt{\frac{|H_g|}{|H|}}\sum_{j\in H_g}|j\rangle + \sqrt{\frac{|H_b|}{|H|}}\sum_{j\in H_b}|j\rangle\,. \tag{8.5}$$

Since at least half of all possible random bit strings lead to a valid generator output, it follows that $|H_g|/|H| \geq 1/2$, and therefore, this estimation occurs extremely quickly. This is done once in each loop iteration, and the quantumly estimated statistic $s_i/|H|$ is rescaled by an approximation to $|H_g|/|H|$.

From standard counting bounds [7], we can approximate $|H_g|/|H|$ within ratio $(1+\varepsilon/11m)$, with probability at least $8/\pi^2$, using $22\sqrt{2}\pi m\varepsilon^{-1}$ applications of a suitable Grover iterate. The

success probability can be boosted to $1-1/8m$, with an additional $\log(8m)$ repetitions of this estimation algorithm; this is an application of the variant powering lemma found in appendix A.4. Thus, with probability at least $1-1/8m$, we approximate $|H_g|/|H|$ within ratio $(1+\varepsilon/11m)$ using $\log(8m)\cdot 22\sqrt{2}\pi m\varepsilon^{-1}$ applications of a suitable Grover iterate:

$$\left|\frac{\widetilde{H_g|/|H|}-|H_g|/|H|}{|H_g|/|H|}\right| \leq \varepsilon/11m$$

where $\widetilde{|H_g|/|H|}$ is an ε-approximation to $|H_g|/|H|$. We have implicitly used the fact that $|H_g|/|H| \geq 1/2$.

Note that we additionally will estimate $s_i/|H|$, which means that there will be yet another added layer of ε-approximation, which will be explored in the next section.

8.5.3 Quantum mechanically estimating these statistics

The ratio $s_i/|H_g|$ should become the focal statistic for the quantum FPRAS since it approximates $Ext_R(x,i)/N_R(x)$ sufficiently well.

The mode finding FPRAS of Chapter 7 will find a prefix w whose frequency approximates that of the modal prefix within ratio $(1+\varepsilon/11m)$ with probability at least $1-1/8m$. Thus, the element w found by the mode finding algorithm is effectively the modal prefix sought by this process. Accordingly, we will refer to w as the modal prefix.

Along with identifying w, the approximate mode finding algorithm will also give an approximation to the frequency of w, i.e., $s_w/|H|$. This quantity will then be rescaled by our approxi-

mation of $|H_g|/|H|$.

We showed in Chapter 7 that there exists a quantum $\{\varepsilon,\delta\}$-FPRAS for finding an approximate mode of a list of elements; we restate that theorem here for reference. In what follows, we assume a list of N elements, drawn from a set of M elements:

Theorem 5 (Chapter 7, theorem 4). *There exists a quantum $\{\varepsilon,\delta\}$-approximation scheme for mode finding that approximates the frequency of the mode within ratio $1+\varepsilon$ with probability at least $1-\delta$, using $O\left(\frac{M^{3/2}\log M}{\varepsilon}\cdot\log(1/\delta)\right)$ list queries.*

In particular, with $19800|x|^{3c/2}\log|x|^{c}m\log(8m)\varepsilon^{-1}$ generator queries in the quantum FPRAS, the frequency of the modal prefix, $s_w/|H|$ is approximated within ratio $(1+\varepsilon/11m)$ with probability at least $(1-1/8m)$. The constant 19800 arises from the product of two constants, the first being 1800 in the specific details of theorem 4, and the second constant being 11, from the $\varepsilon/11m$ accuracy required in this application.

When rescaled by $\widetilde{|H|/|H_g|}$ the estimated quantity $\widetilde{s_w/|H|}\cdot\widetilde{|H|/|H_g|}$ approximates $s_w/|H_g|$ within ratio $(1+\varepsilon/11m)^2$.

Since $s_w/|H_g|$ approximates $Ext_R(x,w)/N_{R(x)}$ within ratio $(1+\varepsilon/11m)^2$, it follows that $\widetilde{s_w/|H|}\cdot\widetilde{|H|/|H_g|}$ approximates $Ext_R(x,w)/N_{R(x)}$ within ratio $(1+\varepsilon/11m)^4$ which is less than $1+\varepsilon/2m$ for $0<\varepsilon<1$. Finally, we note that this happens with probability at least $(1-1/8m)^2 > (1+1/4m)$.

8.6 The Quantum FPRAS

Here, we give the quantized FPRAS for approximate counting and account for the total number of generator queries required throughout. We also show the correctness of the algorithm by a proof which is in the spirit of Jerrum et al., [17].

As before, we assume the existence of an almost uniform generator $\mathcal{G}$ which takes as input, $\langle x, \varepsilon \rangle$ where ε is the fixed error tolerance desired in the counting procedure. For the following, recall that $\sigma(x)$ is from the definition of self-reducibility and has the property that $\sigma(x) \in O(\log |x|)$. The variable $m = g(x)$ is an upper bound on the size of any solution to x. The function $g(x)$ is a polynomial time computable function assumed to exist by the definition of self-reducibility (definition 14) and gives the size of any solution to x.

1: $\Pi = 1$
2: $t := 19800|x|^{3c/2}\log|x|^{c} m \log(8m)\varepsilon^{-1}$
3: **while** $g(x) > 0$ **do**
4: Using the quantum $\{\varepsilon, \delta\}$-FPRAS for mode finding with error tolerance $\varepsilon/11m$, and t queries, determine w
5: From the appropriate registers of the mode finding algorithm, determine $\widetilde{\frac{s_w}{|H|}}$.
6: Estimate $|H|/|H_g|$ using error tolerance $\varepsilon/11m$, and $\log(8m) \cdot 22\sqrt{2}\pi m/\varepsilon$ applications of a suitable Grover iterate.
7: Let $\alpha = \widetilde{\frac{s_w}{|H|}} \cdot \widetilde{|H|/|H_g|}$.
8: $x := \psi(x, w)$ (self-reduce)
9: $\Pi := \Pi/\alpha$
10: **end while**

11: output Π

Claim 2. *The above procedure is a fully-polynomial randomized approximation scheme for approximately counting the number of solutions to some fixed problem instance x.*

Proof. We must show that upon termination, this procedure will estimate $N_R(x)$ within ratio $(1+\varepsilon)$ with probability at least $3/4$. We note that the program variable Π will keep a running product of estimated quantities, which are of the form $Ext_R(x_i, w_i)/N_{R(x_i)}$, where i denotes a particular iteration of the algorithm. We will focus on an arbitrary iteration, and therefore the subscripts i will be dropped.

Previously, we showed that the quantum FPRAS for mode finding estimated $s_w/|H|$ within ratio $(1+\varepsilon/11m)$ with probability at least $(1-1/8m)$. We also estimated $|H|/|H_g|$ within ratio $(1+\varepsilon/11m)$ with probability at least $(1-1/8m)$. As stated previously, the rescaled estimates when multiplied together, approximate $Ext_R(x,w)/N_{R(x)}$ within ratio $(1+\varepsilon/11m)^4 < (1+\varepsilon/2m)$, with probability at least $(1-1/8m)^2$.

Note that this procedure can fail in one of two places during any particular iteration of the algorithm. Namely, we may get a bad estimate of $s_w/|H|$ from the quantum FPRAS, or we may may get a bad estimate of $|H|/|H_g|$. Since we have probability bounds on these events, it follows that in one round, the probability that we do not fail is at least $(1-1/8m)^2 > (1-1/4m)$. Since there are at most m rounds, the probability that we do not fail in each of the m rounds is at least $(1-1/4m)^m > 3/4$.

Finally, note that the value of Π returned by this procedure estimates $N_R(x)$ within ratio $(1+\varepsilon/2m)^m$ which is less than $(1+\varepsilon)$ for $\varepsilon < 1$.

□

8.6.1 Query complexity

Here, we calculate the number of queries required throughout the approximate counting process. For this, we fix a problem instance x, and consider the number of queries used in one iteration of the FPRAS. Subsequently, we will use this expression in a sum on the number of loop iterations to determine the total number of generator queries throughout. Finally, we will compare this quantity to that of Jerrum et al., to see what query reductions we get by treating this problem quantum mechanically.

For a fixed instance x, we use the quantum FPRAS for mode finding to find w. In one loop iteration, we require $t := 19800|x|^{3c/2}\log|x|^c m\log(8m)\varepsilon^{-1}$ generator queries. Note that in subsequent iterations, the size of x will reduce. Furthermore, an upper bound on the number of iterations is obviously m. Thus:

$$\sum_{i=0}^{m} \frac{19800|x_i|^{3c/2}\log|x_i|^c m\log(8m)}{\varepsilon} \tag{8.6}$$

$$\leq \frac{19800|x|^{3c/2}\log|x|^c m^2\log(8m)}{\varepsilon} \tag{8.7}$$

$$\in O\left(\frac{|x|^{3c/2}\log|x|^c m^2\log m}{\varepsilon}\right) \tag{8.8}$$

where x is the original problem instance supplied to the approximation scheme. In each iteration, there is an added $\log(8m)\cdot 22\sqrt{2}\pi m\varepsilon^{-1}$ queries required to estimate $|H_g|/|H|$. Overall however, the number of required generator queries does not change, i.e., we still require

$O\left(|x|^{3c/2}\log|x|^c m^2 \log m\varepsilon^{-1}\right)$ queries to estimate $N_R(x)$.

We now compare this number of queries to that of the scheme found in Jerrum et al. Recall that their scheme used $180|x|^{3c}m^3/\varepsilon^2$ queries per loop iteration with a maximum of m such iterations. Thus, the total number of queries required in that setting was: $180|x|^{3c}m^4/\varepsilon^2 \in O\left(|x|^{3c}m^4/\varepsilon^2\right)$.

Upon comparing the two quantities, there is a quadratic reduction in $1/\varepsilon$ and $|x|^c$, although, we gain $\log|x|^c$ factors, and so it is not quite a quadratic reduction in $|x|^c$. It should be noted that this additional logarithmic factor can be eliminated by treating the sampled prefix frequencies as bounded-error inputs. Essentially, this extra logarithmic factor is required in the mode approximation scheme to ensure a good quality count. By treating the approximated frequencies as bounded-error inputs, one can eliminate the logarithmic factor by using a quantum search for bounded-error inputs as mentioned in the concluding remarks of Chapter 7. The bounded-error searching algorithm is due to Høyer et al. [16].

8.7 Concluding remarks

We have attempted to incorporate quantum methods into the approximate counting scheme of Jerrum et al. [17], with the goal of reducing the number of generator queries in order to approximate the size of a combinatorial solution set. Alternatively, one can get a better (closer) approximation of this quantity by investing the same number of generator queries required in the Jerrum et al. scheme, but by using the quantum FPRAS for counting instead. Essentially, this means that we get a better estimate for the same number of queries.

Appendix A

Useful Facts

A.1 Proof of Fact 1

The fact is restated here for reference:

Fact 4 (statement of fact 1).

$$\left|\frac{1}{M}\frac{1-e^{2iMx}}{1-e^{2ix}}\right|^2 = \frac{\sin^2(Mx)}{M^2\sin^2(x)} \tag{A.1}$$

Proof. Note that

$$\begin{aligned}\left|1-e^{2ix}\right| &= \left|e^{-ix}-e^{ix}\right| \\ &= |\cos(x)-i\sin(x)-\cos(x)-i\sin(x)| \\ &= 2\sin(x).\end{aligned}$$

Therefore,

$$\left|\frac{1}{M}\frac{1-e^{2iMx}}{1-e^{2ix}}\right|^2 = \left|\frac{1}{M}\frac{2\sin(Mx)}{2\sin(x)}\right|^2 = \frac{sin^2(Mx)}{M^2\sin^2(x)}$$

□

A.2 Derivative Bounds

Consider the non-uniformized phase estimation algorithm. In estimating the phase parameter ω, the algorithm gives a state $|\widetilde{\omega}\rangle$, the measurement of which can be considered as a random variable X taking on values in $[0, M-1]$. The parameter M is the dimension of the quantum Fourier transform applied in the phase estimation algorithm.

The probability function associated with X is :

$$\Pr[X = x] = \frac{\sin^2(M\pi(\omega - x/M))}{M^2\sin^2(\pi(\omega - x/M))}$$

The generalized function corresponding to this distribution is $f(x) = \sin^2(M\pi x)/M^2\sin^2(\pi x)$ which is illustrated in figure A.1

Consider the derivative of $f(x)$ with respect to x:

$$\frac{d}{dx}f(x) = \frac{2\pi\sin(M\pi x)\cos(M\pi x)}{M\sin^2(\pi x)} - \frac{2\pi\sin^2(M\pi x)\cos(\pi x)}{M^2\sin^3(\pi x)}$$

In this appendix, we prove the following lemma:

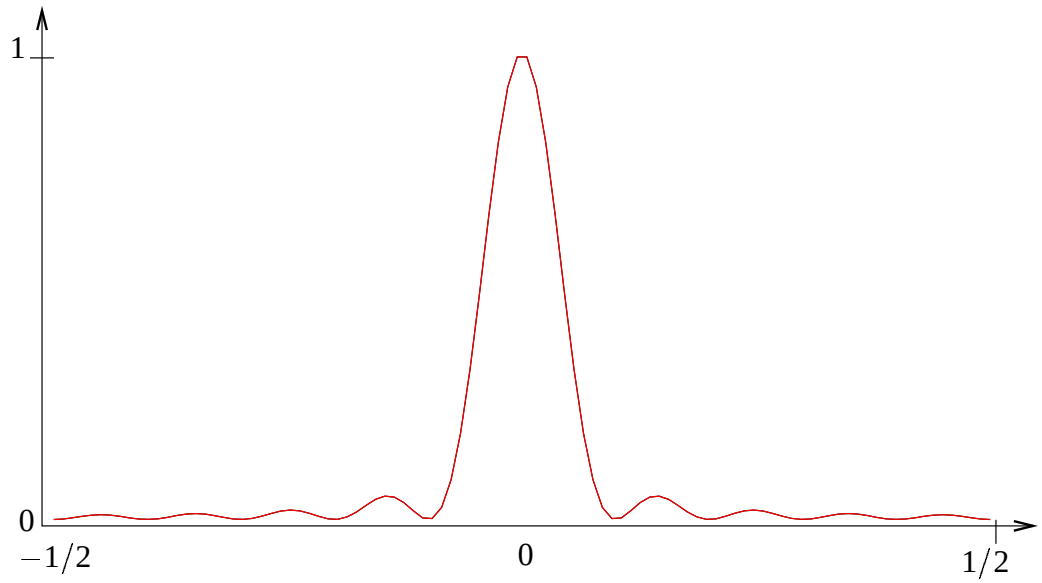


Figure A.1: The function $\sin^2(M\pi x)/M^2\sin^2(\pi x)$

Lemma 1. *Let* $f(x) = \sin^2(M\pi x)/M^2\sin^2(\pi x)$*, then* $f'(x) \leq cM$*, where c is some constant.*

Proof. Consider partitioning $f(x)$ into intervals of size $1/M$. With exception of the largest peak in figure A.1, we have a period of the $\sin^2(M\pi x)$ portion of f scaled appropriately by $\sin^2(\pi x)$. The largest peak covers a window of size $2/M$ due to the dramatic scaling of $\sin^2(M\pi x)$ by $\sin^2(\pi x)$ as x approaches 0.

Consider the global peak which occurs at 0 in the figure above. Since this is a global maximum, it follows that $f'(x)$ will also have global maxima around this area. In particular, one can consider $f''(x)$ to note that $f'(x)$ is maximized at the mid point of the $1/M$ windows surrounding the global maximum. More generally, if the interval being considered is $[a/M,(a+1)/M]$, then $f'(x)$ will have a local maximum at $(a+1/2)/M$. Since this is true for any a, we consider the windows surrounding the global peak. Without loss of generality, consider $f'(x)$ evaluated at

$|f'(1/2M)|$:

$$\begin{aligned} f'\left(\frac{1}{2M}\right) &= \frac{2\pi}{M}\left(\frac{\cos(\pi/2M)}{\sin^3(\pi/2M)}\right) \\ &= \frac{2\pi}{M}g(M) \end{aligned}$$

Since $f'(1/2M)$ tends to infinity as M tends to infinity, we consider the following equation: $M^u = g(M)$, and consider what value u approaches as M tends to infinity. Accordingly, $u = \log_M g(M)$, and in the limit as M tends to infinity, $u = 3$. Thus:

$$\begin{aligned} f'(x) &\leq f'(1/2M) \\ &= \frac{2\pi}{M}g(M) \\ &\leq 2\pi M \end{aligned}$$

□

In a similar fashion, one can prove the following lemma:

Lemma 2. *Let $f(x) = \sin^2(\pi x)/M^2\sin^2(\pi x/M)$, then $f''(x) \leq dM^2$, where d is some constant.*

Proof. The proof of this lemma follows the proof of lemma 1. The key details include:

- In a window of $1/M$ of f, the derivative $f'(x)$ can change from at most 0 to M
- A local maximum of $f''(x)$ occurs at the mid-point of these $1/M$ windows

□

A.3 Approximate Integration

In this section, we state some facts about approximate integration by Riemann sums. In particular, we are interested in the error introduced by such an approximation.

Consider some function $f(x)$ and its integral over a closed interval $[a,b]$ as $\int_a^b f(x)dx$. For our purposes, $0 \leq a \leq b \leq 1$. The formal definition of this integral is:

$$\int_a^b f(x)dx \quad = \quad \lim_{\|P\|\to 0} \sum_{i=1}^{n} f(x_i^*)\Delta x_i$$

where $\|P\| = \max\{\Delta x_i\}$. Here, the continuous interval $[a,b]$ is partitioned by $P = \{x_1, ..., x_n\}$ such that $x_1 = a$ and $x_n = b$. Without loss of generality, assume that that $\Delta x_i = 1/n \, \forall i$, so that $\|P\| = 1/n$. Above, x_i^* is some point in the i^{th} interval. Thus, the integral of $f(x)$ over $[a,b]$ is defined as

$$\int_a^b f(x)dx \quad = \quad \lim_{n\to\infty} \frac{1}{n} \sum_{i=1}^{n} f(x_i^*)$$

For any fixed value of n, the corresponding sum is an approximation of $\int_a^b f(x)dx$.

If x_i^* is taken as the left endpoint of the i^{th} interval, then the corresponding sum is known as a *left endpoint approximation*, while if x_i^* is taken as the right endpoint of the i^{th} interval, then the corresponding sum is known as a *right endpoint approximation*. Yet another approximation is the *midpoint approximation*, where x_i^* is taken as midpoint of the i^{th} interval. It is known that the midpoint approximation yields a lower error than either of the left or right endpoint approximations.

Another approximation is known as the *trapezoid rule*, which considers the average error of the left and right endpoint approximations. Let E_L and E_R be the approximation errors corresponding to the left and right endpoint approximations respectively. Then, the approximation error in the trapezoid rule is exactly $E_T = 1/2(E_L + E_R)$. The following error bound for E_T is found in advanced textbooks on Calculus:

Theorem 6. *Suppose that* $|f''(x)| < K$ *for* $a \leq x \leq b$. *Then,*

$$|E_T| \leq \frac{K(b-a)^3}{12n^2}$$

Since $b - a < 1$, it follows that for our considerations, this bound is: $|E_T| \leq K/12n^2$.

Now let $f(x) = \sin^2(M\pi x)/M^2 \sin^2(\pi x)$. Thus by a Riemann sum, we will approximate $\int_a^b \sin^2(M\pi x)/M^2 \sin^2(\pi x)dx$ by a left endpoint approximation. Before we proceed, we need to have a bound E_L. For this, we consider $E_L - E_R$:

$$E_L - E_R = \int_0^1 f(x)dx - \frac{1}{n}\sum_{i=1}^{n} f(x_{i-1}) - \int_0^1 f(x)dx + \frac{1}{n}\sum_{i=1}^{n} f(x_i) \quad \text{(A.2)}$$

$$= \frac{1}{n}\sum_{i=1}^{n} f(x_i) - f(x_{i-1}) \quad \text{(A.3)}$$

In the setting of uniformization, we take $n = R$ and $x_i = i/R$. Upon substituting these into equation (A.3), we get:

$$E_L - E_R = \frac{1}{R}\sum_{i=0}^{R-1} f\left(\frac{i+1}{R}\right) - f\left(\frac{i}{R}\right) \quad (A.4)$$

$$= \frac{1}{R}\left(f\left(\frac{R}{R}\right) - f\left(\frac{0}{R}\right)\right) \quad (A.5)$$

$$= 0 \quad (A.6)$$

Thus, for the function f, we have that $E_L = E_R$, which translates to a bound on E_L, namely that: $|E_L| \leq \frac{K}{6R^2}$.

Finally, we note that from the results in appendix A.2, it follows that $K \leq cM^2$, where c is some constant. Accordingly, we have that $|E_L| \leq \frac{cM^2}{6R^2}$. When $R = TM$, then $|E_L| \leq \frac{c}{6T^2}$.

A.4 A Variant of the Powering Lemma

Consider the following definition:

Definition 17 (taken from Jerrum et al. [17]). *Suppose $f \in \Sigma^* \to \mathbb{N}$. A randomized approximation scheme for f is a probabilistic Turing machine M which for all inputs $\langle x, \varepsilon \rangle \in \Sigma^* \times \mathbb{R}^+$ produces an output $M(x,\varepsilon)$ such that*

$$\Pr(M(x,\varepsilon,\delta) \text{ approximates } f(x) \text{ within ratio } 1+\varepsilon\ ) \quad \geq \quad \frac{3}{4}$$

A randomized approximation scheme is fully-polynomial if its runtime is bounded above by a polynomial in $|x|$ and $1/\varepsilon$.

We find the following powering lemma for randomized approximation schemes in Jerrum et al.:

Lemma 3 (Lemma 6.1, [17]). *Let $f \in \Sigma^* \to \mathbb{N}$, and suppose that there is a fully-polynomial randomized approximation scheme for f. Then there exists a probabilistic Turing machine M which on input $\langle x, \varepsilon, \delta \rangle \in \Sigma^* \times \mathbb{R}^+ \times \mathbb{R}^+$ produces an output $M(x,\varepsilon,\delta)$ (a random variable) such that:*

$$\Pr[M(x,\varepsilon,\delta) \text{ approximates } f(x) \text{ within ratio } 1+\varepsilon\ ] \quad \geq \quad 1-\delta$$

where $\delta < 1$. Moreover, the execution time of M is bounded by a polynomial in $|x|$, $1/\varepsilon$, and $\log 1/\delta$

Proof. (sketch)

We run the postulated randomized approximation scheme for $t = 12\lceil -\log\delta\rceil + 1$ times, and output the median of the t results. It follows that the probability that the median fails to approximate $f(x)$ within the required bounds is bounded above by:

$$\sum_{i=\frac{t+1}{2}}^{t} \binom{t}{i} \left(\frac{1}{4}\right)^{i} \left(\frac{3}{4}\right)^{t-i}$$

which was proved to be bounded above by $e^{-t/12}$ [17].

□

Now consider an approximation scheme which approximates $f(x)$ within an absolute error tolerance of ε with the same probabilistic guarantees as in definition 17:

Definition 18. *Suppose $f \in \Sigma^* \to \mathbb{N}$. An absolute error randomized approximation scheme for f is a probabilistic Turing machine M which for all inputs $\langle x, \varepsilon\rangle \in \Sigma^* \times \mathbb{R}^+$ produces an output $M(x,\varepsilon)$ such that*

$$\Pr[|M(x,\varepsilon,\delta) - f(x)| \leq \varepsilon] \quad \geq \quad \frac{3}{4}$$

An absolute error randomized approximation scheme is fully-polynomial if its runtime is bounded above by a polynomial in $|x|$ and $1/\varepsilon$.

Having defined an absolute error randomized approximation scheme, the variant powering lemma can be deduced:

Lemma 4. *Let $f \in \Sigma^* \to \mathbb{N}$, and suppose that there is a fully-polynomial absolute error randomized approximation scheme for f. Then there exists a probabilistic Turing machine M which on input $\langle x, \varepsilon, \delta\rangle \in \Sigma^* \times \mathbb{R}^+ \times \mathbb{R}^+$ produces an output $M(x,\varepsilon,\delta)$ (a random variable) such that:*

$$\Pr[|M(x,\varepsilon,\delta) - f(x)| \leq \varepsilon] \quad \geq \quad 1-\delta$$

where $\delta < 1$. *Moreover, the execution time of M is bounded by a polynomial in* $|x|$, $1/\varepsilon$, *and* $\log 1/\delta$

Proof. Similar to the proof of the powering lemma for randomized approximation schemes, we run the postulated absolute-error randomized approximation scheme for $t = 12\lceil -\log\delta\rceil + 1$ times, and output the median of the t results. It follows that the probability that the median fails to approximate $f(x)$ within the required bounds is bounded above by:

$$\sum_{i=\frac{t+1}{2}}^{t} \binom{t}{i} \left(\frac{1}{4}\right)^{i} \left(\frac{3}{4}\right)^{t-i}$$

which is less than $e^{-t/12}$ and hence less than δ by the choice of t. □

This proof easily follows since the failure probability expression is the same in both cases. In particular, it has nothing to do with the fact that one approximation scheme deals with relative errors, while the other deals with absolute errors.

A.5 Expected Value of a Monotone Increasing Probability Function

In this appendix, we show that a random variable having a monotone non-decreasing probability distribution defined on $[0,N]$ has an expected value at least that of a random variable having a uniform distribution, also defined on $[0,N]$.

Lemma 5. *Consider a discrete random variable X taking on values $x \in [0,N]$ with probability function $p(x)$ which is monotone non-decreasing. The expected value of X, $\mathrm{E}[X] \geq N/2$.*

Proof. Without loss of generality, assume that N is a power of 2. Consider the expected value of X:

$$\begin{aligned}
\mathrm{E}[X] &= \sum_{x=0}^{N} xp(x) && \text{(A.7)}\\
&= \sum_{x=0}^{N/2} xp(x) + (N-x)p(N-x) && \text{(A.8)}\\
&\geq \sum_{x=0}^{N/2} \frac{N}{2}p(x) + \frac{N}{2}p(N-x) && \text{(A.9)}\\
&= \frac{N}{2} && \text{(A.10)}
\end{aligned}$$

To prove the inequality in equation (A.9), consider difference between the expressions of equation (A.8) and equation (A.9):

$$\begin{aligned}
&\sum_{x=0}^{N/2} xp(x) + (N-x)p(N-x) - \sum_{x=0}^{N/2} \frac{N}{2}p(x) + \frac{N}{2}p(N-x) && \text{(A.11)}\\
&= \sum_{x=0}^{N/2} \left(x - \frac{N}{2}\right) p(x) + \left(\frac{N}{2} - x\right) p(N-x) && \text{(A.12)}\\
&= \sum_{x=0}^{N/2} \left(\frac{N}{2} - x\right) (p(N-x) - p(x)) && \text{(A.13)}\\
&\geq 0 && \text{(A.14)}
\end{aligned}$$

□

A.6 Probabilistic Recurrences

Consider a particle which begins a position n and proceeds to position 1 by taking steps of size X, where X is a discrete random variable. In particular, if m is the current position of the particle, then X is defined on $[1, m-1]$. Accordingly, this random variable is denoted as $X(m)$ to indicate the dependence on m. Consequently, the next position of the particle is $m - X(m)$. Consider the random variable T denoting the number of steps before the particle reaches position 1. Clearly, T is dependent on X in some way, and all we know about X is that $\mathrm{E}[X] > g(x)$ where g is a monotone non-decreasing function.

In this appendix, we prove the following theorem:

Theorem 7 (adapted from [22, 18]). *Let T be a random variable denoting the number of steps before a particle which starts at position n reaches position 1. If $\mathrm{E}[X] > g(m)$, then*

$$\mathrm{E}[T] \leq \int_1^n \frac{dx}{g(x)}$$

Proof. (Proof of theorem 1.3 [22])

The proof is by induction on n. Suppose that the result holds true for values of m smaller than n, and let $f(m) = \int_1^m dx/g(x)$ for $m \geq 1$. The inductive proof will show that $\mathrm{E}[T] \leq f(n)$.

Consider the first step of the particle which is given by the random variable X where $\mathrm{E}[X] \geq g(n)$. Then:

$$
\begin{aligned}
\mathrm{E}[T] &\leq 1+\mathrm{E}[f(n-X)] && \text{(A.15)}\\
&= 1+\mathrm{E}\left[\int_1^n \frac{dy}{g(y)} - \int_{n-X}^n \frac{dy}{g(y)}\right] && \text{(A.16)}\\
&= 1+f(n)-\mathrm{E}\left[\int_{n-X}^n \frac{dy}{g(y)}\right] && \text{(A.17)}\\
&\leq 1+f(n)-\mathrm{E}\left[\int_{n-X}^n \frac{dy}{g(n)}\right] && \text{(A.18)}\\
&\leq 1+f(n)-\frac{\mathrm{E}[X]}{g(n)} && \text{(A.19)}\\
&\leq f(n) && \text{(A.20)}
\end{aligned}
$$

□

Note that the range of X implies that the particle always moves forward. We now prove a slightly generalized version of this theorem in which X takes on values in $[m-n, m-1]$

Theorem 8. *Let T be a random variable denoting the number of steps before a particle which starts at position n reaches position 1. If $\mathrm{E}[X] > g(m) > 0$, then*

$$\mathrm{E}[T] \leq \int_1^n \frac{dx}{g(x)}$$

Proof. Essentially, the proof used in 7 applies in this case also. The only added requirement is that $\mathrm{E}[X] > 0$ so that we expect to move forward on every round. Thus, the absolute requirement for always moving forward is weakened. Intuitively, if we move forward most of the time, then the occasional backwards step should not dramatically affect the deviation of T from its expected value. In particular, it can be shown that a large deviation of T from $\mathrm{E}[T]$ happens with low probability by using Markov's inequality. □

Bibliography

[1] D. S. Abrams and S. Lloyd. Nonlinear quantum mechanics implies polynomial-time solution for **NP**-complete and #**P** problems. *Phys. Rev. Lett.*, 81:3992–3995, 1998.

[2] R. Beals, H. Buhrman, R. Cleve, and M. Mosca. Quantum lower bounds by polynomials. *J. ACM*, 48(4):778–797, 2001.

[3] P. Benioff. The computer as a physical system: A microscopic quantum mechanical hamiltonian model of computers as represented by turing machines. *J. Stat. Phys.*, 22:563–591, 1980.

[4] C. H. Bennett. Demons, engines and the second law. *Scientific American*, 259(5):108–116, November 1987.

[5] E. Bernstein and U. Vazirani. Quantum complexity theory. *SIAM J. Comp.*, 26:1411–1478, 1997.

[6] M. Boyer, G. Brassard, P. Høyer, and A. Tapp. Tight bounds on quantum searching. *Fortschritte der Physik*, 46:493–505, 1998.

[7] G. Brassard, P. Høyer, M. Mosca, and A. Tapp. Quantum amplitude amplification and estimation. *quant-ph/0005055*, 2000.

[8] D. Cheung. Using generalized quantum fourier transforms in quantum phase estimation algorithms. Master's thesis, University of Waterloo, 2003.

[9] R. Cleve, A. Ekert, C. Macchiavello, and M. Mosca. Quantum algorithms revisited. *quantph/9708016*, 1997.

[10] D. Deutsch. Quantum theory, the Church-Turing principle and the universal quantum computer. *Proc. R. Soc. Lond. A*, 400:97–117, 1985.

[11] D. Deutsch. Quantum computational networks. *Proc. R. Soc. Lond. A*, 425:73, 1989.

[12] D. Deutsch and R. Jozsa. Rapid solution of problems by quantum computation. *Proc. R. Soc. Lond. A*, 439:553–558, 1992.

[13] P. A. M. Dirac. *The principles of quantum mechanics*. The international series of monographs on physics. Clarendon Press, Oxford, 4 edition, 1947.

[14] C. Dürr and P. Høyer. A quantum algorithm for finding the minimum. *quant-ph/9607014*, 1996.

[15] L. K. Grover. A fast quantum mechanical algorithm for database search. In *Proc. 28th Annual ACM Symposium on the Theory of Computing*, pages 212–219, 1996.

[16] P. Høyer, M. Mosca, and R. de Wolf. Quantum search on bounded-error inputs. *quant-ph/0304052*, 2003.

[17] M. Jerrum, L. Valiant, and V. Vazirani. Random generation of combinatorial structures from a uniform distribution. *Theoretical Computer Science*, 43:169–188, 1986.

[18] R. M. Karp, E. Upfal, and A. Wigderson. The complexity of parallel search. *J. Comput. Syst. Sci.*, 36(2):225–253, 1988.

[19] A. Messiah. *Quantum Mechanics*. Interscience Publishers, New York, 1961.

[20] M. Mosca. Counting by quantum eigenvalue estimation. *Theoretical Computer Science*, 264:139–153.

[21] M. Mosca and C. Zalka. Exact quantum fourier transforms and discrete logarithm algorithms. *quant-ph/0301093*, 2003.

[22] R. Motwani and P. Raghavan. *Randomized algorithms*. Cambridge University Press, 1995.

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

[24] C. H. Papadimitriou. *Computational Complexity*. Addison-Wesley, 1994.

[25] A. C. Yao. Quantum circuit complexity. In *Proc. of the 34th Ann. IEEE Symp. on Foundations of Computer Science*, pages 352–361. IEEE Press, New York, 1993.