PROBLEM SOLVING USING QUANTUM COMPUTING AND QUANTUM CLIQUE ALGORITHM

A Thesis

Submitted in partial fulfillment of the
requirement for the award of degree
Of

Master of Engineering
In
Software Engineering

Under the Supervision of
Mr. Lalit Garg
Lecturer
Computer Science and Engineering Department
Thapar Institute Of Engineering and Technology, Patiala

Submitted By
Ms. Mandeep Kaur
(8023108)

Computer Science & Engineering Department
Thapar Institute Of Engineering & Technology
(Deemed University), Patiala-147004 (India)

May 2004

ABSTRACT

Quantum Computers are being proposed as new devices for computing. They are based on the principles of quantum mechanics. A key difference between the classical computer and a quantum computer is that a classical computer obeys the well-understood laws of classical physics whereas a quantum computer is a device that harnesses physical phenomenon unique to quantum mechanics to realize a fundamentally new mode of information processing. Quantum Computation and Quantum Information is the study of the information processing tasks that can be accomplished using quantum mechanical systems. The spectacular promise of quantum computers is to enable new algorithms (called quantum algorithms) that give solutions to problems, requiring exorbitant resources for their solution on a classical computer.

A comparison of computability and complexity of a quantum computer to that of a classical computer is done. Specifically we are interested in the questions: can a quantum computer compute any problems that are incomputable classically and can a quantum computer compute problems more efficiently than a classical computer. An attempt has been made to divide the problems into three broad categories from view of quantum computing. These are as follows:

- Problems that can be efficiently solved by quantum computers, i.e. they provide exponential speed-up as compared to their classical counterparts.
- Problems for which quantum computers provide polynomial solutions but not exponential speedups, than those provided by their classical counterparts.
- Problems that cannot be solved efficiently by either quantum computers or classical computers.

The fields of quantum information theory and quantum complexity have been expanding dramatically, with a number of new interesting and important theoretical results. Meanwhile, the development of algorithms has lagged behind, with barely any significant new algorithms discovered other than Shor's and Grover's algorithms. We present some guidelines that give insight into designing new quantum algorithms.

We choose an NP-Complete problem – Clique Problem and show that it is one of those problems that can gain speed-ups using quantum computing. The proposed quantum algorithm for clique problem reformulates it as a search problem and uses Grover's search algorithm and quantum-counting algorithms, to search for the solutions i.e. cliques in a given graph. It first estimates the number of cliques present in a given graph using the quantum counting algorithm and then apply Grover's search algorithm for finding those cliques. It offers a great speed-up in comparison with the classical algorithms.

DECLARATION

I hereby certify that the work which is being presented in the thesis entitled, "Problem Solving Using Quantum Computing and Quantum Clique Algorithm", in partial fulfillment of the requirements for the award of degree of Master of Engineering in Software Engineering submitted in Computer Science and Engineering Department of Thapar Institute of Engineering and Technology (Deemed University), Patiala, is an authentic record of my own work carried out under the supervision of Mr. Lalit Garg.

The matter presented in this thesis has not been submitted by me for the award of any other degree of this or any other University.

Ms. Mandeep Kaur

This is to certify that the above statement made by the candidate is correct and true to best of my knowledge.

Mr. Lalit Garg
Lecturer
Computer Sc. and Engg. Department
Thapar Institute Of Engg. And Technology, Patiala

Countersigned by

Ms. Seema Bawa
H.O.D
Computer Sc. and Engg. Department
Thapar Institute of Engg. and Technology
Patiala.

Dr. D.S Bawa

Dean of Academic Affairs,
Thapar Institute of Engg. and Technology
Patiala.

ACKNOWLEDGMENT

I express my gratitude to Mr. Lalit Garg under whose guidance, encouragement and inspiration I have prepared my thesis. I would like to thank him for introducing me to the problem and providing invaluable advice throughout the course of the thesis. He let me work with complete freedom, while strongly supporting my academic endeavors.

I am highly indebted to Ms. Seema Bawa, Head Of Department (Computer Science & Engineering Department), for providing me the requisite environment and for being a constant source of inspiration.

It was a pleasure working at Thapar Institute of Engineering and Technology, and is mostly due to the wonderful people who have sojourned here over the past years.

This work could never have been accomplished without the inspiration and support of innumerable colleagues.

Ms. Mandeep Kaur
(8023108)

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1 INTRODUCTION

Science offers the boldest metaphysics of the age. It is a thoroughly human construct, driven by the faith that if we dream, press to discover, explain, and dream again, thereby plunging repeatedly into new terrain, the world will some how come dearer and we will grasp the true strangeness of the universe. And the strangeness will all prove to be connected, and make sense.

- Edward O. Wilson

1.1 INTRODUCTION

Quantum computers are a proposed means of using quantum-mechanical effects to achieve efficient computation. The potential power of a QC stems from the fact that a quantum system can be in a superposition of states, allowing exponentially many computations to be done in parallel.

The indivisible unit of classical information is the bit, which can take one of two values {0, 1}. The corresponding unit of quantum information is the "quantum bit" or qubit, which can be represented as a ray in 2-d Hilbert space. We may denote an orthonormal basis for this Hilbert space $\{|0\rangle, |1\rangle\}$, any state of a qubit may be expressed as

$$a|0\rangle + b|1\rangle \qquad (1.1)$$

with $|a|^2+|b|^2 = 1$ (normalization).

When we measure qubit we get result either $|0\rangle$ with probability $|\alpha|^2$ or $|1\rangle$ with probability $|\beta|^2$, where $|\alpha|^2 + |\beta|^2 = 1$. After measurement the state of the system can be represented in terms of classical states.

The temporal evolution of the system is given by Unitary transformations U which is represented as an n x n matrix, which obeys the rule $U^{\dagger}U$= I(where I is the identity matrix and $U^{\dagger}$ is the conjugate transpose of U). This implies that unitary transformations are reversible in nature as opposed to classical transformations, which don't need to obey the reversibility condition. In fact, reversibility of quantum circuits comes as the effect of characteristics of the unitary transformations. The only non-unitary transformation allowed in quantum computing is the Measurement operation.

Quantum parallelism is the fundamental feature of many quantum algorithms. Quantum parallelism allows quantum computers to evaluate a function f(x) for many different values of x simultaneously. Quantum parallelism arises from the ability of a quantum memory register to exist in a superposition of base states. A quantum memory register can exist in a superposition of states, each component of this superposition may be thought of as a single argument to a function. A function performed on the register in a superposition of states is thus performed on each of the components of the superposition, but this function is only applied one time. Since the number of possible states is 2^n, where n is the number of qubits in the quantum register, we can perform in one operation on a quantum computer what would take an exponential number of operations on a classical computer. This is the main feature of quantum parallelism [25][23].

Unlike classical parallelism, where multiple circuits each built to compute f(x) are executed simultaneously, here a single f(x) circuit is employed to evaluate the function for multiple values of x simultaneously, by exploiting the ability of a quantum computer to be in superposition of different states. This is the main difference between a classical parallelism and a quantum parallelism.

Such a parallelism suggested that quantum computers are capable of providing exponential speed-ups. Interestingly it was thought whether quantum computers would be able to provide such a speed-up for NP problems. But the argument of Bennet, Bernstein, Brassard and Vazirani that a brute force approach to quantum computation results in only quadratic gains suggests that BQP does not contain NP [5]. Further, it has been known that there is no intrinsic property of quantum computation that will function like a black box to solve NP-complete problems [5]. Thus even quantum computers are not capable of efficiently solving NP-complete problems, although it does provide speed-ups for some of the NP-Complete problems. In this thesis we show that Clique problem is one such problem that can gain speed-ups from quantum computers but these speed-ups are not good enough to place it into BQP complexity class.

1.2 PROBLEM SPECIFICATION

In this thesis an attempt has been made to categorize problems from view of quantum Computation into three broad classes; first, problems that can be efficiently solved by quantum computers, i.e. they provide exponential speed-up as compared to their classical counterparts, second problems for which quantum computers provide polynomial solutions but not exponential speedups, than those provided by their classical counterparts and third problems that cannot be solved efficiently by either quantum computers or classical computers. Further, it has been shown that quantum computer is capable of providing speed-ups in the case of clique problem, which is a NP-complete problem.

Clique problem has been solved using quantum algorithms. Complexity of clique problem that uses a brute force search (on a classical algorithm) to find cliques is $O(2^n)$. Let

G(v, e) represents an undirected graph where v = {0,1,2,3,…,n} is the vertex set of G and $e \subseteq v$ x v is the edge set of G. A clique in graph G is *complete subgraph* of G. Clique problem, formulated as a decision problem, is to find whether there exist a k-clique i.e. clique of size k in graph G.

The proposed algorithm named Q-Clique algorithm uses quantum counting and quantum search algorithm. It attacks the Clique problem by reformulating it as a search problem. It uses a search space of $N = 2^n$ items. Here items are all possible combinations of n nodes. This will contain valid as well as invalid (because there can't be cliques between two or one nodes) combinations of cliques. Q-Clique algorithm's Oracle takes as input an item from search space and tells whether it denotes a clique in graph G or not. Q-clique algorithm uses quantum counting algorithm to estimate the number of solutions (i.e. cliques)M in a given graph G and then uses Grover's search algorithm to find the cliques. The complexity of Q-Clique algorithm is $O(\sqrt{N/M})$ where $N= 2^n$.

1.3 ORGANISATION OF THESIS

The **First chapter**, briefly introduces quantum computing and provides an insight into how it differs from other computational models and the promises it hold. It also gives a brief overview of the problem undertaken in this thesis. The **Second chapter** of the thesis is devoted to brief history of quantum computation and some background material. Although it would be possible to define quantum computation without reference to any more than elementary linear algebra, it is often far more fruitful to view it as an outgrowth of quantum mechanics or classical complexity theory. With this in mind, the **Third chapter** provides a brief overview of classical complexity theory and introduces the

quantum complexity class - BQP. In this chapter an attempt has been made to classify problems into three different categories – problems that can be efficiently solved by quantum computers, problems for which quantum computers cannot provide efficient solutions or exponential speedups than those provided by their classical counterparts and problems for which quantum computers provide inefficient solution. The **Fourth chapter** concerns itself with the quantum algorithms. It provides an insight into various quantum algorithms - Quantum Fourier transform, Shor's Factoring algorithm and Grover's search algorithm, and some guidelines on how to design quantum algorithms. The **Fifth chapter** opens with introduction to clique problem followed by the Q-Clique algorithm. Finally, concluding thesis in **Sixth chapter** with future scope and directions.

CHAPTER 2 QUANTUM COMPUTATION

"The memory of a classical computer is a string of 0s and 1s, and a classical computer can do calculations on only one set of numbers at once. The memory of a quantum computer is a quantum state, which can be in a superposition of many different numbers at once. A classical computer is made up of bits, and a quantum computer is made up of quantum bits, or qubits. A quantum computer can do an arbitrary reversible classical computation on all the numbers simultaneously, and also has some ability to produce interference, constructive or destructive, between various different numbers. By doing a computation on many different numbers at once, then interfering the results to get a single answer, a quantum computer has the potential to be much more powerful than a classical computer of the same size.... The 0 and 1 of a qubit might be the ground and excited states of an atom in a linear ion trap; they might be polarizations of photons that interact in an optical cavity; they might even be the excess of one nuclear spin state over another in a liquid sample in an NMR machine.... It may not ultimately be possible to make a quantum computer that can do a useful calculation before decohering, but if we can get the error rate low enough, we can use a quantum error-correcting code to protect the data even when the individual qubits in the computer decohere."

Daniel Gottesman Microsoft. October 29, 1997.

2.1 BRIEF HISTORY

Every now and then surprising new theory appears on the scientific stage that holds the promise of dramatic new technologies. Quantum computing is one of these. "Quantum computing begins where Moore's Law ends - about the year 2020, when circuit features are predicted to be the size of atoms and molecules," says Isaac L. Chuang. "Indeed, the basic elements of quantum computers are atoms and molecules." As electronic devices were made smaller and smaller in size, quantum effects started interfering in their functioning. The problem imposed by these results find a solution by moving on to new computing paradigm – Quantum Computing, which performs computing based on the principles of quantum mechanics. It turned out that quantum computers provide a speed advantage that no classical computer can ever provide with a conceivable amount of improvement in its power, over time.

Although most of what is presently known about basic quantum mechanics was worked out in the early twentieth century, we are far from having worked out all of the implications. The quantum mechanical picture of particles and wave functions is profoundly counter-intuitive and despite its universal acceptance, there are few people who are truly comfortable with it. One of the most striking examples of this quantum weirdness is superposition, the ability of quantum systems to exist in arbitrary linear combinations of allowable classical states. This accounts for, among other things, the fact that we observe interference patterns in the double-slit experiment even when only a single photon passes through at a time [1].

Rules of quantum mechanics are simple but even experts find them counter-intuitive. Generations of physicists since have wrestled with quantum mechanics in an effort to make its predictions more palatable. One of the goals of quantum computation is to develop tools, which sharpen our intuition about quantum mechanics, and its predictions more transparent to human minds. In early 1980's interest arose in whether it might be possible to use quantum effects to signal faster than light, this lead to the invention of the No-Cloning theorem, which stated that it is not possible to Clone an unknown quantum state.

Interest in quantum computation and quantum information lies back since 1970's when efforts were made to develop complete control over single quantum systems. One of the early challenges met by the quantum computation and quantum information was that the effects of realistic noise must be taken into account while evaluating the efficiency of a computational model, which was met with great success with the development of a theory of **quantum error-correcting codes** and fault-tolerant quantum computation. Another early

discovery in the field of quantum computation was **quantum cryptography** or quantum key distribution. Meanwhile came the strong Church-Turing thesis in mid 1970's. The first major challenge to strong church-Turing thesis came from randomized algorithms. In 1985 David Deutsch asked whether laws of physics could be used to derive an even stronger version of the strong Church-Turing thesis. This led Deutsch to consider computing devices based on the laws of quantum mechanics. He developed a notion of a Universal Quantum computer. It is still an open question whether this model is sufficient to efficiently simulate any arbitrary physical system. Deutsch's model of universal quantum computation posed a challenge to strong Church-Turing thesis, asking whether it is possible to efficiently solve computational problems which have no efficient solution on a classical computer, even a probabilistic Turing machine. He then constructed a simple example suggesting that, indeed, quantum computers might have computational powers exceeding those of classical computers. This result was further strengthened with the discovery of Peter Shor's Factoring algorithm and discrete algorithms using quantum computing in 1994. Further evidence for the power of quantum computer came when in 1995 Lov Grover showed that it is possible to speed up the problem of searching through an unstructured search space using quantum computers. Eventually now questions started arising as what other problems can quantum computers solve more quickly than classical computers? But till yet very little has been known to answer these questions.

Another area of quantum computation and quantum information where development started taking place was - **Information theory**. In 1995, Ben Schumacher provided an analogue to Shannon's noiseless coding theorem, and in the process defined the "qubit" as a tangible physical resource. Theory of quantum error-correction was developed

which allowed quantum computers to compute effectively in the presence of noise, and also allows communication over noisy quantum channels to take place reliably. In 1996 an important class of quantum codes now known as CSS codes was discovered. These discoveries greatly facilitated a rapid understanding of quantum-error correcting codes and their application to quantum computation and quantum information.

In 1992 Charles Bennet and Stephen Wiesner explained how to transmit two classical bits of information, while only transmitting one quantum bit from sender to receiver, called superdense coding. The results of distributed quantum computation were even more interesting. It has been shown that quantum computers can require exponentially less communication to solve certain problems than would be required if the networked computers where classical. Unfortunately as yet these problems are not especially important in a practical setting and suffer from some undesirable technical restrictions.

2.2 AXIOMS OF QUANTUM MECHANICS

Quantum theory (as all physical theories) can be characterized by how it represents (physical) states, observables, measurements, and dynamics (evolution in time) [19].

1. **States**: A state is a complete description of a physical system. In quantum mechanics, a state is a ray in a Hilbert space. A Hilbert space is a vector space over the field of complex numbers C, with vectors denoted by $|\psi\rangle$ (Dirac's ket notation). A ket $|\psi\rangle$ is represented as $n \times 1$ matrix (or a n element vector), where n is the dimension of the Hilbert space, and its corresponding bra $\langle\psi|$ is the transpose conjugate of the ket. It has an inner product $\langle\psi|\varphi\rangle$ (may be seen as

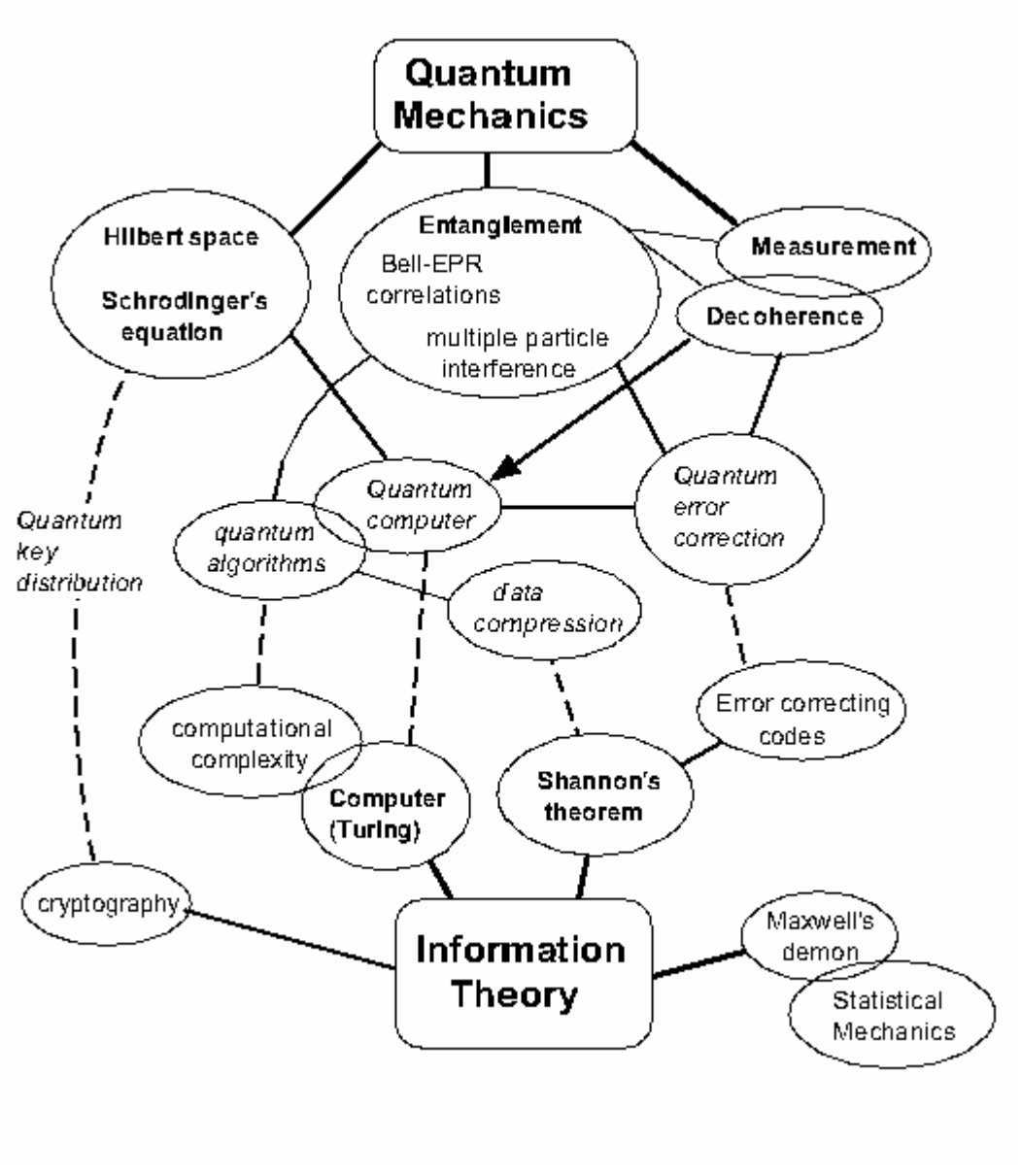

Figure 2.1: Relationships between information science (including computer science) and quantum mechanics

matrix multiplication) that maps an ordered pair of vectors to C, with the properties:

(a) Positivity $\langle\psi|\ \psi\rangle > 0$ for $|\ \psi\rangle \neq 0$

(b) Linearity $\langle\varphi|\ (a|\ \psi_1\rangle + b|\ \psi_2\rangle\ ) = a\langle\varphi|\ \psi_1\rangle + b\langle\varphi|\ \psi_2\rangle$

(c) Skew symmetry $\langle\varphi|\ \psi\rangle=\langle\psi|\ \varphi\rangle^{\dagger}$

The Hilbert space is complete in the norm $|\ |\ \psi|\ | =\langle\psi|\ \psi\rangle^{1/2}$

A ray in a Hilbert space is an equivalent class of vectors that differs by multiplication by a nonzero complex scalar. That is, a ray is represented by a given vector and all its (complex) multiples, and two different rays treated as vectors will never be "parallel" (in the Hilbert space). We represent rays (the equivalent classes) by a vector with unit norm $\langle\psi|\ \psi\rangle = 1$, so $e^{i\alpha}|\ \psi\rangle$ (where $\alpha\in\ \Re$) represent the same physical state for all α. The real number α can be seen as an overall phase, which is physically insignificant. We can form new states by superposition $a|\ \varphi\rangle + b|\ \varphi\rangle$, and here the relative phase between the two components are physically significant. That is, whereas $a|\ \varphi\rangle + b|\ \varphi\rangle$ and $e^{i\alpha}$ $(a|\ \varphi\rangle + b|\ \varphi\rangle)$ represent the same physical state, $a|\ \varphi\rangle + e^{i\alpha}\ b|\ \varphi\rangle$ is generally a different physical state.

2. **Observables:** An observable is a property of a physical system that in principle can be measured. In quantum mechanics, an observable is a self-adjoint operator (matrix). An operator is a linear map taking vectors to vectors, which can be represented by matrices. For an operator A, $A:|\ \psi\rangle\rightarrow A|\ \psi\rangle$ and $A(a|\ \psi\rangle + b|\ \varphi\rangle) = aA|\ \psi\rangle+bA|\ \varphi\rangle$. And the adjoint of an operator $A^{\dagger}$ is defined by $\langle\varphi|\ A\psi\rangle = \langle A^{\dagger}\varphi|\ \psi\rangle$, where $|\ A\psi\rangle$ denotes $A|\ \psi\rangle$, for all vectors $|\ \psi\rangle$, $|\ \varphi\rangle$. In matrix

representation of the adjoint is the transpose conjugate. A is self-adjoint if $A = A^{\dagger}$. The eigenstates of an observable (eigenvectors of the corresponding self-adjoint matrix) form a complete orthonormal basis in the Hilbert space H.

3. **Measurements:** In quantum mechanics, the numerical outcome of the measurement of the observable A is an eigenvalue (An eigenvector of a linear operator A on a vector space is a non-zero vector $|v\rangle$ such that $A|v\rangle = v|v\rangle$, where v is a complex number known as the eigenvalue of A corresponding to $|v\rangle$) of A, and right after the measurement, the quantum state becomes the eigenstate of A corresponding to the measurement result. If the quantum state before measurement is $|\psi\rangle$, then outcome a_n is obtained with probability $\text{Prob}(a_n) = \| P_n|\psi\rangle\|^2 = \langle\psi| P_n|\psi\rangle$, and the (normalized) quantum state becomes in this case

$$\frac{P_n|\psi\rangle}{\langle\psi| P_n|\psi\rangle^{1/2}} \qquad (2.1)$$

Which means if the measurement is immediately repeated, the same result would be obtained with probability one.

4. **Dynamics:** Time evolution of a quantum state is unitary (unitary transformation can be seen as a rotation in Hilbert space); it is generated by a self-adjoint operator, called the Hamiltonian of the system. In the Schrödinger picture of dynamics, the vector (state) describing the system evolves in time according to the Schrödinger equation

$$\frac{d}{dt}|\psi(t)\rangle = -iH|\psi(t)\rangle \qquad (2.2)$$

where H is the Hamiltonian. Using the definition of function derivatives, the equation can be expressed as $|\psi(t + dt)\rangle = (1 - iHdt)|\psi(t)\rangle$. The operator $U(dt) \equiv 1 - iHdt$ is unitary to linear order in dt, because $U(dt)^{\dagger}U(dt) = (1+iHdt)(1 - iHdt) = 1+(Hdt)^2 \approx 1$. Since a product of unitary operators is finite, time evolution over a finite interval is also unitary $|\psi(t)\rangle = U(t)|\psi(0)\rangle$. If the Hamiltonian is time-independent, the Schrödinger equation can be directly solved to give $U(t) = e^{-itH}$ (which is likewise unitary).

In the formulations in this section, there is an obvious dualism between how a quantum state evolves when "left to itself", and when it has been "measured" by "something". In the former case the state evolves according to the Schrödinger equation, which is deterministic; whereas in the latter case the evolution is probabilistic.

2.3 POSTULATES OF QUANTUM MECHANICS

Postulates are statements, which are not directly proven (like a mathematical proof). Their validity is considered to be either self-evident ("Heat flows spontaneously from a hot body to a cold body.") or the results which flow from the postulates agree with other observations and we conclude that the postulates must be correct. Quantum theory postulates are of the second kind. We can't prove them. We can only assert them and then demonstrate a huge body of evidence in agreement with those results.

POSTULATE 1: Associated to any isolated physical system is a complex vector space with inner product (that is Hilbert space) known as the state space of the system. The system is completely described by its state vector, which is a unit vector in the systems state space.

The simplest quantum mechanical system is the qubit. A qubit has a two-dimensional state space. $B = \mathbf{C}^2$. The state $|\psi\rangle$ of a qubit can be described by a linear combination (also called superposition) of just two basis states labeled $|0\rangle$ and $|1\rangle$.

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle \text{ with } \alpha,\beta \in \mathbf{C} \text{ and } |\alpha|^2 + |\beta|^2 = 1 \qquad (2.3)$$

POSTULATE 2: The evolution of a closed quantum system is described by a unitary transformation. That is, the state $|\psi\rangle$ of the system at time t_1 is related to the state $|\psi'\rangle$ of the system at time t_2 by a unitary operator U which depends only on the times t_1 and t_2,

$$|\psi'\rangle = U|\psi\rangle \qquad (2.4)$$

Operators are recipes for converting one function into another, in the same way that functions are recipes for converting one number into another. This postulate describes how the quantum states of a closed quantum system at two different times are related. A more refined version the postulate can be given which describes the evolution of quantum system in continuous time.

POSTULATE 2': The time evolution of the state of a closed quantum system is described by the Schrödinger equation,

$$i\hbar \frac{\partial|\psi\rangle}{\partial t} = H|\psi\rangle \qquad (2.5)$$

In this equation, $\hbar$ is a physical constant known as Planck's constant whose value must be experimentally determined. In practice it is common to absorb the factor $\hbar$ into H, effectively setting $\hbar = 1$. H is a fixed Hermitian operator known as Hamiltonian of the closed system.

If we know the Hamiltonian of a system, then (together with the general knowledge of $\hbar$) in principle, one can understand the dynamics of the system completely. In general figuring out the Hamiltonian needed to describe a particular physical system is a very difficult problem.

POSTULATE 3: Quantum measurements are described by a collection $\{M_m\}$ of measurement operators. These are operators acting on the state space of the system being measured. The index *m* refers to the measurement outcomes that may occur in the experiment. If the state of Quantum system is $|\psi\rangle$ immediately before the measurement then the probability that result *m* occurs is given by

$$p(m) = \langle\psi|M_m^\dagger M_m|\psi\rangle \qquad (2.6)$$

and the state of the system after the measurement is

$$\frac{M_m|\psi\rangle}{\sqrt{\langle\psi|M_m^\dagger M_m|\psi\rangle}} \qquad (2.7)$$

The measurement operators satisfy the completeness equation,

$$\sum_m M_m^\dagger M_m = I \qquad (2.8)$$

The completeness equation expresses the fact that probabilities sum to one:

$$\sum_m p(m) = \sum_m \langle\psi|M_m^\dagger M_m|\psi\rangle = 1 \qquad (2.9)$$

An example of a measurement is the measurement of a qubit in the computational basis. This is a measurement on a single qubit with two outcomes defined by the two measurement operators $M_0 = |0\rangle\langle 0|$, and $M_1 = |1\rangle\langle 1|$. Observe that each measurement operator is Hermitian, and that $M_0^2 = M_0$, $M_1^2 = M_1$. Thus the completeness relation is

obeyed. Suppose the state being measured is $|\psi\rangle = a|0\rangle + b|1\rangle$. Then the probability of obtaining measurement outcome 0 is

$$p(0) = \langle\psi| M_0^{\dagger} M_0 |\psi\rangle = \langle\psi| M_0 |\psi\rangle = |a|^2 \quad (2.10)$$

and the state after measurement becomes

$$\frac{M_0|\psi\rangle}{|a|} = \frac{a}{|a|}|0\rangle \quad (2.11)$$

It can be seen that multipliers like $a/|a|$, which have modulus one, can be effectively ignored, so the post measurement state is effectively $|0\rangle$ [25].

2.4 QUANTUM BASICS

Anybody who is not shocked by quantum theory has not understood it - Niels Bohr

Quantum mechanics, a branch of mathematical physics, holds that energy and matter exist in discrete, knowable quantities dubbed quanta (quantum singular). This supposition, called quantization, makes the field of quantum mechanics especially applicable to the study of interactions among elementary particles. It stands opposed to the tenet of classical physics, which declares energy to be a solely continuous phenomenon, and matter to move similarly in a precise region of space. According to quantum mechanics, energy is emitted and absorbed in the aforementioned units of quanta, which move about in some situations like matter particles. Quantum theory suggests a dual nature for waves and particles, and as such has become useful in explaining many properties of matter.

Some definitions in quantum mechanics are as follows:

1. **Hilbert Space:** It is a complete complex vector space. If we have a complex vector space of finite dimension n it is called complete. Hilbert space is denoted by H. The state of a quantum computer is represented by a vector in some vector space H.

   Let $|f\rangle, |g\rangle, |h\rangle \in H$ and $\alpha,\beta \in \mathbf{C}$, then the following operations are defined [21]:

   $$|f\rangle + |g\rangle \in H \qquad \text{linear combination} \qquad (2.12)$$

   $$\alpha|f\rangle \in H \qquad \text{scalar multiplication} \qquad (2.13)$$

   $$|f\rangle + |0\rangle = |f\rangle \qquad \text{zero-element} \qquad (2.14)$$

   $$|f\rangle + |-f\rangle = |0\rangle \qquad \text{inverse element} \qquad (2.15)$$

   The inner product $\langle . | . \rangle : H \times H \rightarrow \mathbf{C}$ is conjugate linear and meets the following conditions:

   $$\langle f | g + h\rangle = \langle f | g\rangle + \langle f | g\rangle \qquad (2.16)$$

   $$\langle f | \alpha g\rangle = \alpha\langle f | g\rangle \qquad (2.17)$$

   $$\langle f | g\rangle = (\langle g | f\rangle)^{\dagger} \qquad (2.18)$$

   $$\langle f | f\rangle = 0 \Leftrightarrow |f\rangle = |0\rangle \qquad (2.19)$$

   $$\| f \| \equiv \sqrt{\langle f | f\rangle} \geq 0 \qquad (2.20)$$

2. **Qubit:** Qubit is analogous to a classical bit. A qubit has states $|0\rangle$, $|1\rangle$ and superposition of these states. The states $|0\rangle$ and $|1\rangle$ are called computational basis states. The superposition state is represented as $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$. The numbers α and β are complex numbers, which represent the amplitude for states $|0\rangle$ and $|1\rangle$. The state of a qubit can be represented in a two-dimensional complex vector space. When we measure qubit $|\psi\rangle$ we get result either $|0\rangle$ with probability $|\alpha|^2$ and $|1\rangle$ with probability $|\beta|^2$, where $|\alpha|^2 + |\beta|^2 = 1$.

The value of a qubit is the observable N with the Hermitian operator $N\,|\,i\rangle = i\,|\,i\rangle$ over the Hilbert space $H = \mathbf{C}^2$, or in matrix representation

$$N = \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix}$$

The expectation value of N is given by

$$\langle N \rangle = \langle \Psi | N | \Psi \rangle = \begin{pmatrix} \alpha^* & \beta^* \end{pmatrix} \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} \alpha \\ \beta \end{pmatrix} = |\beta|^2 \qquad (2.21)$$

thus, $\langle N \rangle$ gives the probability to find the system in state $|\,1\rangle$ if a measurement is performed on the qubit.

If we combine two qubits, the general state of the resulting system is $|\,\psi\rangle = \alpha\,|\,00\rangle + \beta\,|\,01\rangle + \gamma\,|\,10\rangle + \delta\,|\,11\rangle$ with $|\,\alpha|^2 + |\,\beta|^2 + |\,\gamma|^2 + |\,\delta|^2 = 1$. We can define distinct observables $N^{(1)}$ and $N^{(2)}$ for the value of each qubit with the operators $N^{(1)}\,|\,ij\,\rangle = i\,|\,ij\,\rangle$ and $N^{(2)}\,|\,ij\,\rangle = j\,|\,ij\,\rangle$ their expectation values

$$\langle N^{(1)} \rangle = |\,\beta|^2 + |\,\delta|^2 \quad \text{and} \quad \langle N^{(2)} \rangle = |\,\gamma|^2 + |\,\delta|^2 \qquad (2.22)$$

But this doesn't allow us to reconstruct the actual probability distribution among the eigenvalues. To measure a qubit state in two-qubit state, first measure the first qubit. For example, measuring the first qubit gives 0 with probability $|\,\alpha|^2 + |\,\beta|^2$ and the post measurement state becomes

$$|\,\psi\rangle = \frac{\alpha\,|\,00\rangle + \beta\,|\,01\rangle}{(|\,\alpha|^2 + |\,\beta|^2)^{1/2}} \qquad (2.23)$$

3. **Bell State or EPR Pair:** It has the property that, measurement of second qubit always gives the same result as measurement of the first qubit. Some of the Bell states are given below:

$$|\beta_{00}\rangle = \frac{|00\rangle + |11\rangle}{(2)^{1/2}} \qquad (2.24)$$

$$|\beta_{01}\rangle = \frac{|01\rangle + |10\rangle}{(2)^{1/2}} \qquad (2.25)$$

$$|\beta_{10}\rangle = \frac{|00\rangle - |11\rangle}{(2)^{1/2}} \qquad (2.26)$$

$$|\beta_{11}\rangle = \frac{|01\rangle - |10\rangle}{(2)^{1/2}} \qquad (2.27)$$

4. **Operators:** Operators are represented as matrices in quantum mechanics. All valid states ψ can be expressed as a sum of eigenfunctions, i.e.

$$\psi(\vec{r}, t) = \sum_{i=0}^{\infty} c_i \psi_i(\vec{r}, t) \qquad (2.28)$$

If we use $\{\psi_0, \psi_1, \psi_2, \ldots\}$ as unit vectors, we can write the vectors of ψ as infinitely dimensional row- and column- vectors

$$\langle\psi| \equiv (c_0^{\dagger}, c_1^{\dagger}, c_2^{\dagger}, \ldots) \text{ and } |\psi\rangle \equiv \begin{pmatrix} c_0 \\ c_1 \\ \cdot \\ \cdot \\ \cdot \end{pmatrix}$$

5. **Physical Observables:** In quantum physics, a physics observable O is expressed as a linear operator O while the classical value of expectation value $\langle O\rangle$. Since the value of observable like momentum, position must be real so we have $\langle O\rangle \in \mathbf{R}$.

$O^{\dagger}$ is called adjoint operator to O if

$$\langle \hat{f} | g\rangle = \langle f | O | g\rangle \text{ with } |\hat{f}\rangle = O^{\dagger}|f\rangle \qquad (2.29)$$

If O is given in matrix form, the $O^{\dagger}$ is the conjugated transposition of O, i.e. $O^{\dagger} = (O^{T})^{*}$. An operator O with $O^{\dagger} = O$ is called self-adjoint or Hermitian. All quantum

observables are represented by Hermitian operators as we can reformulate the requirement $\langle O \rangle \in \mathbf{R}$ as $\langle O \rangle = \langle O \rangle^*$ or

$$\langle \psi \mid O \mid \psi \rangle = (\langle \psi \mid O \mid \psi \rangle)^* = \langle \psi \mid O^\dagger \mid \psi \rangle \qquad (2.30)$$

6. **Unitary Operators:** The operator of temporal evolution satisfies the condition

$$U^\dagger(t)U(t) = e^{\frac{i}{\hbar}Ht} e^{\frac{-i}{\hbar}Ht} = 1 \qquad (2.31)$$

Operators U and $U^\dagger$ with $UU^\dagger = I$ are called unitary. Since the temporal evolution of a quantum system is described by a unitary operator and $U^\dagger(t) = U(-t)$ it follows that the temporal behavior of a quantum system is reversible, as long a no measurement is performed. Unitary operators can also be used to describe abstract operations like rotations

$$R_z(\alpha) \mid n_1, n_2, n_3 \rangle = \cos(\alpha) \mid n_1, n_2, n_3 \rangle + i \sin(\alpha) \mid n_1, n_2, n_3 \rangle$$

or the flipping of eigenstates

$$\text{Not} \mid n \rangle = \begin{cases} \mid 1 \rangle \text{ if } n=0 \\ \mid 0 \rangle \text{ if } n=1 \\ \mid n \rangle \text{ otherwise} \end{cases}$$

without the need to specify how this transformations are actually performed or having to deal with time-dependent Hamilton operators. Mathematically, unitary operations can be described as base-transformations between 2 orthonormal bases. Let A and B be Hermitian operators with the orthonormal eigenfunctions ψ_n and ψ'_n and $\mid \psi \rangle = \sum_i c_i \mid \psi_i \rangle = \sum_i c'_i \mid \psi'_i \rangle$, then the Fourier coefficients c'_i are given by

$$\begin{pmatrix} \tilde{c}_0 \\ \tilde{c}_1 \\ \cdot \\ \cdot \\ \cdot \end{pmatrix} = U \begin{pmatrix} c_0 \\ c_1 \\ \cdot \\ \cdot \\ \cdot \end{pmatrix} \quad \text{with } U = \sum_{ij} \left|\tilde{\Psi}_i\right\rangle\left\langle\tilde{\Psi}_i\right|\left.\Psi_j\right\rangle\left\langle\Psi_j\right| \qquad (2.32)$$

Unitary transformations and measurements applied to only one subsystem don't affect the other subsystem. That is the unitary transformations applied to distinct subsets of qubits are independent. A unitary transformation U ' over the first m qubits also doesn't affect a measurement of the remaining qubits [4].

There are three major differences when we compare Boolean functions with unitary operators from functional point of view [4]

- Reversibility: Since unitary operators, by definitions, follow the condition $U^{\dagger}U = I$, that is for every unitary transformation there exists the inverse transformation $U^{\dagger}$, unitary transformations are restricted to reversible functions.
- Superposition: An eigenstate $|\psi\rangle = |k\rangle$ can be transformed into a superposition of eigenstates.

$$|\psi'\rangle = U|k\rangle = \sum_{k'} U_{k'k}\left|k\right\rangle \qquad (2.33)$$

- Parallelism: If the machine state $|\psi\rangle$ already is a superposition of several eigenstates, then a transformation U is applied to all eigenstates simultaneously.

$$U\sum_i c_i\left|i\right\rangle = \sum_i c_i U\left|i\right\rangle \qquad (2.34)$$

This feature is called quantum parallelism and is a consequence of the linearity of unitary transformations.

7. **Quantum register:** An m qubit quantum Register s is a sequence of mutually different zero-based qubit positions $\langle s_0, s_1, \ldots, s_{m-1}\rangle$ of some machine state $|\psi\rangle \in \mathbf{C}^{2^m}$ with $n \geq m$.

8. **Tensor Product:** The tensor product of an n dimensional vector *u* and an m dimensional vector *v* is an nm dimensional vector $u \otimes v$.

   If *A* and *B* are operators on n and m dimensional vectors, respectively, then $A \otimes B$ is an operator on nm dimensional vectors. If *H1* and *H2* are Hilbert spaces, then $H1 \otimes H2$ is also a Hilbert space. If *H1* and *H2* are finite dimensional with bases $\{u_1, u_2, \ldots, u_n\}$ and $\{v_1, v_2, \ldots, v_m\}$ respectively, then $H1 \otimes H2$ has dimension nm with basis $\{u_i \otimes v_j \mid 1 \leq i \leq n; 1 \leq j \leq m\}$.

   To give an example of the tensor product of two qubits suppose we have two vectors $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$, $\phi = \gamma|0\rangle + \delta|1\rangle \in H^2$. The resulting product state is

$$|\psi\phi\rangle = \alpha\gamma|00\rangle + \alpha\delta|01\rangle + \beta\gamma|10\rangle + \beta\delta|11\rangle \qquad (2.35)$$

   and lies in H^4. the standard basis of H^4 is $\{|00\rangle, |01\rangle, |10\rangle, |11\rangle\}$.

   Tensor products obey a number of rules. For matrices A, B, C, D, U, vectors u, v, w, and scalars a, b, c, d the following hold [38]:

$$(A \otimes B)(C \otimes D) = AC \otimes BD \qquad (2.36)$$

$$(A \otimes B)(u \otimes v) = Au \otimes Bv \qquad (2.37)$$

$$(u + v) \otimes w = u \otimes w + v \otimes w \qquad (2.38)$$

$$u \otimes (v + w) = u \otimes v + u \otimes w \qquad (2.39)$$

$$au \otimes bv = ab(u \otimes v) \qquad (2.40)$$

   Thus for matrices

$$\begin{pmatrix} A & B \\ C & D \end{pmatrix} \oplus U = \begin{pmatrix} A \oplus U & B \oplus U \\ C \oplus U & D \oplus U \end{pmatrix} \quad (2.41)$$

which specializes for scalars into

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix} \oplus U = \begin{pmatrix} aU & bU \\ cU & dU \end{pmatrix} \quad (2.42)$$

The conjugate transpose distributes over tensor products

$$(A \oplus B)^{\dagger} = A^{\dagger} \oplus B^{\dagger} \quad (2.43)$$

9. **Entanglement:** A quantum register is entangled if it can't be written as a tensor product of its parts. If the quantum system is entangled, it means that there are correlations between the subsystems. What we do with one part of the system will influence the other part. For example the state in H^4

$$|\psi'\rangle = \frac{1}{\sqrt{2}}(|01\rangle - |10\rangle) \quad (2.44)$$

This state is entangled because we cannot find two pure states $|\psi_1\rangle$ and $|\psi_2\rangle$ such that $|\psi'\rangle = |\psi_1\rangle \otimes |\psi_2\rangle$. Entanglement has strange properties. If we measure our qubit (on earth), we will obtain random answer (0 or 1), but we will be sure that if some one measures the other qubit, he will obtain the opposite answer. Although the answer we receive is random, the other answer is always its opposite. This strange "non-local" property led Einstein to criticize quantum mechanics in the famous EPR paper [8].

10. **No cloning:** Given an arbitrary qubit $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$, there does not exist an operator A and a state $|a\rangle$ such that $A(|\psi\rangle \otimes |a\rangle) = |\psi\rangle \otimes |\psi\rangle$.

This implies that we can't clone or copy arbitrary quantum states. Suppose that you have a qubit ϕ, and you'd like to make few identical copies of it. That's no problem:

measure ϕ to determine its state, and then make a bunch of new qubits in that state. But suppose instead that you want to make copies of ϕ without measuring it first. That is, you want to make more qubits in the same state as ϕ, but you don't want to determine the value of ϕ in the process. This would be helpful because you could make a few copies of ϕ and work with them while they were still in their superimposed state. If it took a long time to build up the state of ϕ, and you wanted to run several experiments, it would be handy to make some copies of this input so you wouldn't have to build it up from scratch each time. Unfortunately, cloning isn't possible. You simply can't make a perfect copy of a quantum particle without first determining its state. Although we can't clone an unmeasured particle, don't forget that we have no trouble making copies of a known state; just measure it and manufacture copies [15].

11. **Quantum Parallelism:** Suppose we have a function $f\ \{0,1\}\rightarrow\ \{0,1\}$ and a qubit in superposition of 0 and 1.We want to compute its output. Classically we have to compute $f(0)$ and $f(1)$ separately. If we use quantum computation, we can do this in one step, by computing $f(|\ 0\rangle + |\ 1\rangle) = f(|\ 0\rangle) + f(|\ 1\rangle)$. The two values are computed in parallel. This can be generalized to a superposition of any number of states and gives an exponential speedup over classical computation. The problem is that we can't extract both answers. If we measure the resulting qubit, we will obtain the answer $f(0)$ or $f(1)$ at random.
12. **Quantum Interference:** What makes quantum computation so powerful is that the different parallel computation can interfere, leading some answers to be more likely than others. Suppose we have the unitary transformation U defined by

$$U = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ -1 & 1 \end{pmatrix}$$

If we apply the transformation twice (without measuring in between) we are sure to obtain 0 if we measure because

$$UU|1\rangle = U\,\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$$

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

$$= \frac{1}{2}(|0\rangle - |1\rangle + |0\rangle + |1\rangle = |0\rangle$$

We also have that $UU|0\rangle = -|1\rangle$. Applying U once yields complete random but applying it twice gives a deterministic answer: the negation of the input (up to a constant factor, called global phase.) This is the work of quantum interference. In many quantum algorithms, interference plays a crucial role. The "good" computational paths interfere constructively and the "bad" ones interfere destructively so they will not be encountered. Interference, together with entanglement and parallelism are the weapons that give the quantum computer all his power.

2.5 QUANTUM GATES [40]:

X or Not gate: This is the gate that we know in classical computation with the additional characteristic that it respects the superposition of a qubit. X or quantum NOT gate reverses the states, and in case of superposition state $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ is changed to $\alpha|1\rangle + \beta|0\rangle$.

$$\textbf{Not}\,(\alpha|0\rangle + \beta|1\rangle) = \beta|0\rangle + \alpha|1\rangle \qquad (2.45)$$

$$X \equiv \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

Z or Phase Flip: The Flip or Z gate changes the phase of a qubit conditional on its value. It negates the sign of $|1\rangle$ but leaves the sign of state $|0\rangle$ unchanged.

$$\textbf{Flip}\ (\alpha|0\rangle + \beta|1\rangle) = \alpha|0\rangle - \beta|1\rangle \qquad (2.46)$$

$$Z \equiv \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$$

Hadamard transform or SQUARE-ROOT of NOT Gate: This transformation H maps the zero and one state to the following superpositions of the two basis states:

$$H|0\rangle = \frac{1}{\sqrt{2}}\left(|0\rangle + |1\rangle\right) \text{ and } H|1\rangle = \frac{1}{\sqrt{2}}\left(|0\rangle - |1\rangle\right) \qquad (2.47)$$

The Hadamard is its own inverse ($H^2 = I$).

$$H \equiv \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

Phase Rotation: A more general phase rotation is provided by the Phase operation, which has a free parameter ϕ that determines the angle of the phase change:

$$\textbf{Phase}\phi\ (\alpha|0\rangle + \beta|1\rangle) = \alpha|0\rangle + e^{i\phi}\beta|1\rangle \qquad (2.48)$$

$$\text{Flip} = \text{Phase}_{\Pi}$$

General Rotation: The general rotation R with angles α, θ, ϕ is the unitary one qubit transformation with eigenvectors $|\psi\rangle = \cos\frac{\theta}{2}|0\rangle + e^{i\phi}\sin\frac{\theta}{2}|1\rangle$ and $|\psi'\rangle = \sin\frac{\theta}{2}|0\rangle + e^{i\phi}\cos\frac{\theta}{2}|1\rangle$

The corresponding eigenvalues are indicated by the equalities

$$\mathbf{R}_{\alpha,\theta,\phi} | \psi\rangle = | \psi\rangle \text{ and } \mathbf{R}_{\alpha,\theta,\phi} | \psi'\rangle = e^{i\alpha} | \psi'\rangle \qquad (2.49)$$

and are therefore 1 and $e^{i\phi}$.

Controlled-Not: The controlled-not is a two-qubit operation that applies the Not gate to the target bit if the control bit equals "1"; otherwise it leaves the target unchanged:

$$\mathbf{CNot} | x, y\rangle = | x, y \oplus x\rangle \qquad (2.50)$$

for all $x, y \in \{0,1\}$.

Controlled-Flip: The controlled-flip is, like the CNot , a two-qubit operation. It applies the Flip gate if both bits equals "1"; otherwise it leaves the state unchanged:

$$\mathbf{CFlip} | x,y\rangle = (-1)^{xy} | x,y\rangle \qquad (2.51)$$

for all $x, y \in (0,1)$.

All quantum gates satisfy the property that the matrix representing the gate are unitary i.e. $U^{\dagger}U = I$, where $U^{\dagger}$ is the conjugate transpose of U and I is the identity matrix. If U contains only real numbers then conjugate of U remains U and hence $U^{\dagger}$ becomes the transpose of matrix U. All quantum gates are reversible in nature.

CHAPTER 3 CLASSIFYING PROBLEMS FROM VIEW OF QUANTUM COMPUTING

Quantum computers have gained widespread interest because some problems of practical interest are known to be in BQP. Currently, only three such problems are known-integer factorization, discrete logarithm, and simulation of quantum systems. BQP contains P and BPP and is contained in PP and PSPACE. BQP is complexity class defined for quantum computers. The corresponding class for an ordinary Turing machine plus a source of randomness is BPP. In the plethora of complexity classes, we try to categorize the problems into three categories: problems that can be efficiently solved by quantum computers, problems for which quantum computers cannot provide efficient solutions or exponential speedups, than those provided by their classical counterparts and problems for which quantum computers provide inefficient solution.

3.1 INTRODUCTION TO COMPUTATIONAL COMPLEXITY

Computational Complexity is the study of resources required during computation to solve a given problem. It proves a lower bound on the resources required by the best possible algorithm for solving the given problem. The most common resources are time (how many steps does it take to solve a problem), space (how much memory does it take to solve a problem) and energy (how much energy is expended in performing computation). Although, study of Energy requirements has been given less importance in computer science field, but it is related to the field of reversible computing, which plays a vital role in quantum computing. Another important result in the field of computational complexity theory is the Strong Church-Turing thesis, which states that if a problem cannot be solved efficiently on a probabilistic Turing machine then it cannot be solved efficiently on any machine.

STRONG Church-Turing Thesis: Any model of computation can be simulated on a probabilistic Turing Machine with at most a polynomial increase in the number of

elementary operations required.

But Quantum computers has put into doubt the Strong Church-Turing thesis, by enabling the efficient solution of a problem, which is believed to be intractable on all classical computers, including probabilistic Turing machines.

Computation theory divides problems into various complexity classes, which have been categorized based on essentially three properties - the resource of interest (time, space, …), the type of problem being considered(decision problem, optimization problem, …) and the underlying computational model(deterministic TM, probabilistic TM, Quantum computer, …) [25].

Following are some of the classes of problems considered in complexity theory, along with their rough definitions. Many of these classes have a 'Co' partner (i.e. NP and Co-NP), which consists of the complements of all languages in the original class. For example if L is in NP then the complement of L is in Co-NP. But this doesn't imply that the complement of NP is Co-NP - there are languages, which are known to be in both, and other languages, which are known to be in neither.

P	the set of languages accepted by deterministic Turing machines in polynomial time (Solvable in polynomial time)
NP	the set of languages accepted by nondeterministic Turing machines in polynomial time (YES answer checkable in polynomial time)
Co-NP	the set of languages rejected by nondeterministic Turing machines in polynomial time (NO answers checkable in polynomial time)
NP-complete	The hardest problems in NP
Co-NP-complete	The hardest problems in Co-NP
PSPACE	the set of languages accepted by deterministic Turing machines in polynomial space (Solvable with polynomial memory and unlimited time)

PSPACE-complete	The hardest problems in PSPACE
BQP	Solvable in polynomial time on a quantum computer (answer is probably right)
BPP	Solvable in polynomial time by randomized algorithms (answer is probably right)
PP	The class of decision problems solvable by an NP machine such that - if the answer is 'yes' then at least 1/ 2 of computation paths accept and if the answer is 'no' then less than 1/ 2 of computation paths accept.
NPI	Sometimes used to denote the set of decision problems in NP that are neither NP-complete nor in P. Is thought to contain (for example) decision versions of factoring and graph isomorphism.
ZPP	problem can certainly be solved by a PTM in polynomial time on average

Table 3.1 Table showing some of the complexity classes along with their rough definitions

The question whether P is the same set as NP is the most important open question in theoretical computer science. Assuming that P ≠ NP it is possible to prove that there is non-empty class of problems NPI (NP Intermediate) which denotes the set of decision problems in NP that are neither NP-complete nor in P. Indeed, under this assumption, it contains an infinite number of distinct polynomial-time equivalence classes. Although not proved, it is thought to contain (for example) decision versions of factoring and graph isomorphism. NPI will be nonempty if P does not equal NP [22]. NPI problems are of interest in the area of quantum computation and quantum information. First, it is desirable to find fast quantum algorithms to solve problems that are not in P. Second, many suspect that quantum computers will not be able to efficiently solve all problems in NP and NP-complete [25]. Thus it is natural to focus on the class of NPI. Indeed, a fast quantum

algorithm for factoring has motivated the search for fast quantum algorithms for other problems suspected to be in NPI.

Quantum computers have gained widespread interest because some problems of practical interest are known to be in BQP, but suspected to be outside P. BQP in complexity theory is Bounded-error, Quantum, Polynomial time. It denotes the class of problems solvable by a quantum computer in polynomial time, with an error probability of at most 1/ 4 for all instances. In other words, there is an algorithm for a quantum computer that is guaranteed to run in polynomial time, with the probability of at most ¼, that it will give the wrong answer. That is true irrespective of the answer - YES or NO. The choice of 1/ 4 in the definition is arbitrary. It can be any real number k such that $0 < k < 1/2$. In Quantum Computing, the number of qubits required to solve a problem is function of the instance size. For example, algorithms are known for factoring an n-bit integer using just over 2n qubits. The BQP class is defined for quantum computers, the corresponding class for an ordinary Turing machine plus a source of randomness is BPP. BQP contains P and BPP and is contained in PP and PSPACE.

3.2 CATEGORIZING PROBLEMS FROM VIEW OF QUANTUM COMPUTING

We categorize problems from view of quantum computing, into three categories: those that can be efficiently solved by quantum computers, those for which quantum computers cannot provide solutions more efficient than those provided by their classical counterparts, those for which quantum computers provide inefficient solution.

One advantage that quantum computers have over classical computers is that

quantum circuits are reversible, whereas classical circuits are irreversible in nature. But, it has also been shown that any classical circuit can be converted into an equivalent reversible circuit in polynomial time [3]. It has been proved that BQP can simulate any deterministic or probabilistic polynomial-time algorithm [7]. Thus quantum computer is capable of computing anything that can be computed on classical computer. These results imply that quantum computer is at least as powerful as a classical computer. But, it has also been proved that Turing machine can simulate each of the operations of a quantum circuit, and therefore compute anything that can be computed on a quantum circuit. However, this simulation is very inefficient, as it requires space and time, exponential in the number of qubits in the quantum circuit [12]. No one has yet been able to present an efficient classical simulation of a quantum computer, although a simulation that takes polynomial space, but exponential time has been presented [7]. This has contributed to the idea that a quantum computer may be inherently exponentially faster. But we must also note, that it has yet not been proved that such an efficient simulation does not exist. Hence the question, how much powerful is a quantum computer as compared to classical computer, remains unsolved.

3.2.1 Class of problems that can be efficiently solved by Quantum computer

Class of problems that have been efficiently solved by Quantum computer is BQP. Currently, only three problems (of practical interest) are known to be in BQP - integer factorization, discrete logarithm, and simulation of quantum systems. Another evidence from [33] shows that problem of distinguishing between two fairly natural classes of function, can be solved exponentially faster in the quantum model than in the classical probabilistic one, when the function is given as an oracle drawn equiprobably from the

uniform distribution on either class. Shor's algorithm proves that factoring is in BQP, while it is not known to be in BPP and NP-Complete. It has been showed that there exist oracles under which there exist problems that are in BQP but not BPP, which is now taken as some of the earliest evidence that QTMs are more powerful than probabilistic Turing Machines [5]. But no other problem, than these, has been reported to be in BQP. Recent research suggests that quantum computers does same amount of raw processing as a classical computer, they simply take advantage of unique quantum mechanical structures- like quantum interference [35]. This suggests, although quantum computers are capable of only quadratic speed-up from classical computers, there are certain problems like factoring, discrete logarithms, with certain characteristics that are capable of exploiting certain quantum mechanics properties and thus give exponential speed-ups. It is these problems that can be efficiently solved on quantum computers.

3.2.2 Class of problems for which quantum computers either cannot provide efficient solutions or exponential speedups, than those provided by their classical counterparts

Since anything that can be computed on a classical machine can be computed on a quantum machine with little overhead, it is known that BPP $\subseteq$ BQP. However, it is not known whether or not this containment is strict. While there are problems such as factoring and computing a discrete log that are in BQP but not known to be in BPP, no one has actually *proven* that these problems are not in BPP. Similarly, the relation between NP and BQP is unknown. The argument of Bennet, Bernstein, Brassard and Vazirani that a brute force approach to quantum computation results in only quadratic gains suggests that BQP does not contain NP [5]. However, the relationship between

NP and BPP is currently unknown, so whether or not NP contains BQP is also unknown [25]. It is currently known that BQP sits between BPP and PSPACE in the complexity hierarchy. Thus BQP contains all of P and BPP, and potentially some problems in NP but probably none that are NP-complete, and perhaps some problems in PSPACE that are not in NP [36]. Although the latter two postulations have not yet been proven. Thus *P, ZPP and BPP are the class of problems that can be solved on quantum computer with little overhead, and efficiency comparable to that of classical computers.* These categories of problems differ from the first category of problems (e.g. factoring, discrete logarithms), as these don't provide exponential speed-ups. In other words, these are the problems that don't have the characteristics to exploit certain quantum mechanic properties, which have been taken by problems like Shor's factoring algorithm etc, to provide exponential gains. Lov Grover's Quantum search algorithm is an example of one such problem that has provided quadratic speedups but is not in the complexity class BQP.

3.2.3 Class of problems that cannot be efficiently solved by quantum computers

By efficiency we mean that these problems require more than polynomial time (and/ or polynomial space) to be solved on quantum computers. It has been demonstrated that relative to an oracle chosen uniformly at random, the class NP can never be solved in less than $\sqrt{2^n}$ time. Further, it has been known that there is no intrinsic property of quantum computation that will function like a black box to solve NP-complete problems [5]. These results do establish the fact that quantum computers are not capable of efficiently solving NP-complete problems, but they do not prove that NP cannot be contained in BQP. Thus, it is believed that quantum computers

cannot efficiently solve NP-complete problems, as these problems are supposed to have no particular structure or the structure present is too complex that they has not yet been discovered. If structure of any one problem in NP-complete complexity class could be found, it would become possible to find relationship between NP-complete and BQP class.

With the growing interest in quantum computation field it becomes necessary to categorize problems from quantum computation point of view. We have divided problems into three simple categories - problems that can be efficiently solved on quantum computers - BQP, problems that can be solved on quantum computers but does not provide exponential speed-ups and problems that cannot be solved efficiently on quantum computers. This categorization has provided division of problems into three abstract categories with respect to quantum computers. So far our knowledge of algorithms for solving certain problems has been the main drive in categorizing problems into various complexity classes. This suggests that, categorization of problems is limited by our knowledge of the solutions available for the problem. Need of the hour is to devise algorithms or procedures which categorize problems into various complexity classes. This would be possible only if, every problem is reformulated in mathematical terms.

CHAPTER 4 QUANTUM ALGORITHMS

If computers that you build are quantum,
Then spies everywhere will all want them.
Our codes will all fail,
And they'll read our email,
Till we get crypto that's quantum, and daunt' em.

Jennifer and Peter Shor

4.1 INTRODUCTION

The spectacular promise of quantum computers is to enable new algorithms, which render feasible, problems requiring exorbitant resources for their solution on a classical computer. So far, only two broad classes of algorithms are known which fulfill this promise. The first class is based upon Shor's quantum Fourier transform, and includes remarkable algorithms for solving the factoring and discrete logarithm problems, providing a striking exponential speedup over the best known classical algorithms. The second class of algorithms is based upon Grover's search algorithm for performing quantum searching. These provide a quadratic speedup over the best possible classical algorithms.

Relationship between main quantum algorithms and some of their applications are shown in figure 4.1 [25]. At the core of the diagram is the quantum Fourier transform and the quantum search algorithm. Quantum counting algorithm is a combination of the quantum searching and Fourier transform algorithms, which can be used to estimate the number of solutions to a search problem more quickly than is possible on a classical computer.

Quantum searching algorithms have many applications some of which are - extracting statistics, such as the minimal element, from an unordered data set, to speed up algorithms for some problems in NP (those for which a straight forward search for a solution is the best algorithm known), to speed up the search for keys to cryptosystems such as widely used DES.

Quantum Fourier Transform has many interesting applications. It can be used to solve the discrete logarithm and factoring problems, which in turn enable a quantum computer to break many of the most popular cryptosystems now in use. It is related to the hidden subgroup problem. In fact factoring, discrete logarithm and order finding and many other problems are instances of hidden subgroup problem.

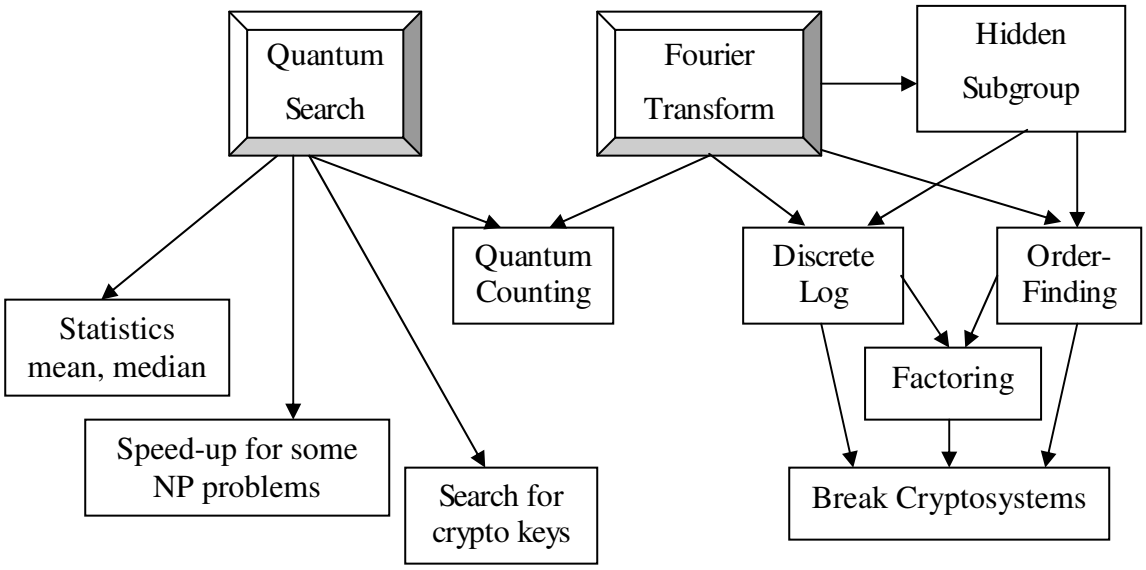

Figure 4.1 The main quantum algorithms and their relationships, including some notable applications

4.2 QUANTUM FOURIER TRANSFORM

One of the most useful ways of solving a problem in mathematics or computer science is to transform it into some other problem for which a solution is known. One such transform is the discrete Fourier transform, which takes as input a vector of complex numbers *x* and outputs the transformed data, another vector of complex numbers *y*. The discrete Fourier transform for a N (=2^n) dimensional vector | ψ⟩, is defined as

$$DFT : |x\rangle \rightarrow \frac{1}{\sqrt{N}} \sum_{y=0}^{N-1} e^{\frac{2\Pi i}{N} xy} |y\rangle \qquad (4.1)$$

The classical fast Fourier Transform (FFT) uses a binary decomposition of the exponent to perform the transformation in O(n2^n) steps.

As suggested by Coppersmith [9], the same principle could be adapted to quantum computers by using a combination of Hadamard transformations *H* and conditional phase gates *V* . The indices below indicate the qubits operated on:

$$DFT' = \prod_{i=1}^{n-1} \left(H_{n-i-1} \left(\frac{\pi}{2} \right) \prod_{j=0}^{i-1} V_{n-i-1,n-j-1} \left(\frac{2\pi}{2^{i-j+1}} \right) \right) H_{n-1} \qquad (4.2)$$

DFT′ iterates the qubits form the MSB to the LSB, "splits" the qubits with the Hadamard transformation and then conditionally applies phases according to their relative binary position ($e^{\frac{2\pi i}{2^{i-j+1}}}$) to each already split qubit. The base-vectors of the transformed state | ψ ')⟩ = *DFT′* | ψ⟩ are given in reverse bit order, so to get the actual DFT, the bits have to be flipped [4].

The Fourier transform is the key to a general procedure known as phase estimation. Suppose a unitary operator U has an eigenvector $|u\rangle$ with eigenvalue $e^{2\pi i\varphi}$, where the value of φ is unknown. The goal of the phase estimation algorithm is to estimate φ.

The quantum phase estimation procedure uses two registers. The first register contains t qubits initially in the state $|0\rangle$. How to choose t depends upon two things: the number of digits of accuracy we wish to have in our estimate φ, and with what probability one wishes the phase estimation procedure to be successful. The second register begins in the state $|u\rangle$, and contains as many qubits as is necessary to store $|u\rangle$.

Quantum phase estimation algorithm takes a Black Box which performs the controlled-U^j operation, for integer j, an eigenstate $|u\rangle$ of U with eigenvalue $e^{2\pi i\varphi_u}$ and $t = n + \lceil \log(2+\frac{1}{2\epsilon}) \rceil$ qubits initialized to $|0\rangle$, as input. It produces an n-bit approximation $\tilde{\varphi}_u$ to φ_u as output in $O(t^2)$ operations and one call to controlled-U^j black box. It succeeds with probability at least $1 - \epsilon$.

Steps for Quantum Phase Estimation

1) $|0\rangle|u\rangle$ initial state

2) $\rightarrow \frac{1}{\sqrt{2^t}} \sum_{j=0}^{2^t-1} |j\rangle|u\rangle$ create superposition

3) $\rightarrow \frac{1}{\sqrt{2^t}} \sum_{j=0}^{2^t-1} |j\rangle U^j|u\rangle$ apply black box

$= \frac{1}{\sqrt{2^t}} \sum_{j=0}^{2^t-1} e^{2\pi ij\varphi_u} |j\rangle|u\rangle$ result of black box

4) $\rightarrow |\tilde{\varphi}_u\rangle|u\rangle$ apply inverse Fourier transform

5) $\rightarrow \tilde{\varphi}_u$ measure first register

Phase estimation is useful, as it solves the problem of estimating the eigenvalue associated to a given eigenvector of a unitary operator. Its real value though comes from the observation that other interesting problems can be reduced to it. Quantum Phase estimation has been applied to solve the order finding and factoring problem.

4.3 SHOR'S FACTORING ALGORITHM

Shor's Algorithm is a quantum algorithm for factoring a number N in $O((\log N)^3)$ time and $O(\log N)$ space, named after Peter Shor. Shor's algorithm is viewed as important because the difficulty of finding prime factors of large numbers is relied upon for most cryptography systems. If an efficient method of factoring large numbers were to be discovered most of the current encryption schemes would be easily compromised. While it has not been proven that factoring large numbers cannot be archived on a classical computer in polynomial time, the fastest published algorithm for factoring large number runs in exponential time. In contrast, Shor's algorithm runs in polynomial time.

Like all quantum computer algorithms, Shor's algorithm is probabilistic: it gives the correct answer with high probability, and repeating the algorithm can decrease the probability of failure. Shor's algorithm was demonstrated in 2001 by a group at IBM, which factored 15 into 3 and 5, using a quantum computer with 7 qubits.

Shor's algorithm hinges on a result from number theory, which states that function $F(a) = x^a \bmod n$ is a periodic function, where x is an integer coprime to n. In the context of Shor's algorithm n will be the number we wish to factor. When two numbers are coprime it means that their greatest common divisor is 1. Calculating this function for an

exponential number of a's would take exponential time on a classical computer. Shor's algorithm utilizes quantum parallelism to perform the exponential number of operations in one step. The reason why this function is of utility in factoring large numbers is because, since $F(a)$ is a periodic function, it has some period r. We know that $x^0 \bmod n = 1$, so $x^r \bmod n = 1$, and $x^{2r} \bmod n = 1$ and so on since the function is periodic.

Given this information and through the following algebraic manipulation:

$$x^r \equiv 1 \bmod n$$

$$(x^{r/2})^2 = x^r \equiv 1 \bmod n$$

$$(x^{r/2})^2 - 1 \equiv 0 \bmod n$$

and if r is an even number

$$(x^{r/2} - 1)(x^{r/2} + 1) \equiv 0 \bmod n$$

We can see that the product $(x^{r/2} - 1)(x^{r/2} + 1)$ is an integer multiple of n, the number to be factored. So long as $x^{r/2}$ is not equal to ± 1, then at least one of $(x^{r/2} - 1)$, $(x^{r/2} + 1)$ must have a nontrivial factor in common with n. So by computing $gcd(x^{r/2} - 1, n)$ and $gcd(x^{r/2} + 1, n)$, we will obtain a factor of n, where gcd is the greatest common denominator function [23].

Steps of Shor's Algorithm:

Shor's algorithm for factoring a given integer n can be broken into some simple steps.

1) Determine if the number n is a prime, an even number, or an integer power of a prime number. If it is Shor's algorithm is not used. There are efficient classical methods for determining if an integer n belongs to one of the above groups, and providing factors for it if it is. This step would be performed on a classical computer.

2) Pick a integer q that is the power of 2 such that $n^2 \leq q < 2n^2$. This step would be done on a classical computer.

3) Pick a random integer x that is coprime to n. There are efficient classical methods for picking such an x. This step would be done on a classical computer.

4) Create a quantum register and partition it into two regions, register 1 and register 2. Thus the state of the quantum computer is given by: $|$ reg1, reg2$\rangle$. Register 1 must have enough qubits to represent integers as large as q-1. Register 2 must have enough qubits to represent integers as large as n-1. The calculations for how many qubits are needed would be done on a classical computer.

5) Load register 1 with an equally weighted superposition of all integers from 0 to q-1. Load register 2 with all zeros. This operation would be performed by the quantum computer. The total state of the quantum memory register at this point is:

$$\frac{1}{\sqrt{q}}\sum_{a=0}^{q-1}|a,0\rangle \qquad (4.3)$$

6) Now apply the transformation $x^a \bmod n$ to each number stored in register 1 and store the result in register 2. Due to quantum parallelism this will take only one step, as the quantum computer will only calculate $x^{|a\rangle} \bmod n$, where $|a\rangle$ is the superposition of states created in step 5. This step is performed on the quantum computer. The state of the quantum memory register at this point is:

$$\frac{1}{\sqrt{q}}\sum_{a=0}^{q-1}\left|a, x^a \bmod n\right\rangle \qquad (4.4)$$

7) Measure the second register, and observe some value k. This has the side effect of collapsing register 1 into a equal superposition of each value a between 0 and q-1 such that

$x^a \bmod n = k$. This operation is performed by the quantum computer. The state of the quantum memory register after this step is:

$$\frac{1}{\sqrt{\| A \|}} \sum_{a'=a' \in A} |a', k\rangle \qquad (4.5)$$

Where A is the set of a's such that $x^a \bmod n$, and $\| A \|$ is the number of elements in that set.

8) Now compute the discrete Fourier transform on register one. The discrete Fourier transform when applied to a state $|a\rangle$ changes it in the following manner:

$$|a\rangle = \frac{1}{\sqrt{q}} \sum_{c=0}^{q-1} |c\rangle * e^{2\pi i a c / q} \qquad (4.6)$$

This step is performed by the quantum computer in one step through quantum parallelism. After the discrete Fourier transform our register is in the state:

$$\frac{1}{\sqrt{\| A \|}} \sum_{a' \in A} \frac{1}{\sqrt{q}} \sum_{c=0}^{q-1} |c, k\rangle * e^{2\pi i a' c / q} \qquad (4.7)$$

9) Measure the state of register one, call this value m, this integer m has a very high probability of being a multiple of q/r, where r is the desired period. This step is performed by the quantum computer.

10) Take the value m, and on a classical computer do some post processing which calculates r based on knowledge of m and q. There are many ways to do this post processing. This post processing is done on a classical computer.

11) Once r has been attained, a factor of n can be determined by taking $gcd(x^{r/2} - 1, n)$ and $gcd(x^{r/2} + 1, n)$. If a factor of n has been found, then stop, if not go to step 4. This final step is done on a classical computer.

Step 11 contains a provision for what to do if Shor's algorithm failed to produce factors of *n*. There are a few reasons why Shor's algorithm can fail, for example the discrete Fourier transform could be measured to be 0 in step 9, making the post processing in step 10 impossible. The algorithm will sometimes find factors of 1 and *n*, which is not useful either. For these reasons step 11 must be able to jump back to step four to start over [23].

4.4 GROVER'S SEARCH ALGORITHM

Grover's algorithm is a quantum algorithm that searches space with N elements, for exactly M solutions, in $O(\sqrt{N/M})$ time. This search space can be an unsorted database of size N or in more general terms, N possible solutions available for a problem out of which exactly M are the solutions with $1 \leq M \leq N$.

It was invented by Lov Grover in 1996. Classically, searching a space with N elements requires a linear search, which is O(N). Grover's algorithm provides only quadratic speedup, unlike other quantum algorithms which are thought to provide exponential speedup over their classical counterparts. However, it is the fastest possible quantum algorithm for searching an unsorted database, and even quadratic speedup is considerable when N is large.

Like all quantum computer algorithms, Grover's algorithm is *probabilistic* in the sense that it gives the correct answer with high probability. The probability of failure can be decreased by repeating the algorithm.

If P(x) is a boolean function for $0 \leq x < N$, classical search algorithms take on the order of $N/2$ operations to find an item x_0 for which $P(x_0) = 1$. Grover's algorithm takes on the order of $\sqrt{N}$ operations. Although Grover's algorithm is often described as *searching*

a database, it would be more accurate to describe it as *inverting a function*. If a function $y = f(x)$ can be computed by an algorithm for a quantum computer, then Grover's algorithm can calculate x, given y. Grover's algorithm could also be used to search for a key that decrypts an encrypted message. If the function for computing $f(x)$ is given as a black box, then Grover's algorithm is the fastest possible algorithm for inverting it.

Grover's algorithm can be used for mean and median estimation, and solving the collision problem. It can be used to solve NP-complete problems by performing exhaustive searches over all possible solutions. This will result in considerable speedups over classical methods, but won't achieve polynomial runtime for NP-complete problems.

To solve the problem, Grover starts by setting a quantum register to a superposition of all possible items in the search space. The quantum state contains the right answer, but if the register were observed at this point, the odds of picking the right answer would be as small as if one picked the item by random. Grover's discovery involves a sequence of simple quantum operations on the register's state. He describes the process in terms of wave phenomena. Grover explains, "All the paths leading to the desired results interfere constructively, and the other ones interfere destructively and cancel each other out," [2].

Steps for Grover's Search Algorithm (for known number of solutions):

1) Let n be such that $2^n \geq N$, and prepare a register containing a superposition of all $x_i \in [0 \ldots 2^n - 1]$.

2) Apply a unitary transformation that computes $P(x_i)$ on this register:

$$U_P : \frac{1}{\sqrt{2^n}} \sum_{x=0}^{2^n-1} |x,0\rangle \rightarrow \frac{1}{\sqrt{2^n}} \sum_{x=0}^{2^n-1} |x, P(x)\rangle \qquad (4.8)$$

For any x_0 such that P(x_0) is true, $|x_0,1\rangle$ will be part of the resulting superposition, but since its amplitude is $\frac{1}{\sqrt{2^n}}$, the probability that a measurement produces x_0 is only $\frac{1}{2^n}$.

3) Change amplitude a_j to $-a_j$ for all x_j such that P(x_j)=1.

4) Apply inversion about the average to increase amplitude of x_j with P(x_j)=1 and decrease other amplitudes.

5) Repeat steps 2 through 4, $\frac{\pi}{4}\sqrt{2^n}$ times (in case of M=1 otherwise $\frac{\pi}{4}\sqrt{2^n/M}$ times, where M is the number of solutions).

6) Measure the last qubit of the quantum state, representing P(x). Because of the amplitude change, there is a high probability that the result will be 1. If this is the case, the measurement has projected the state onto the subspace $\frac{1}{\sqrt{2^k}}\sum_{i=1}^{2^k}|x_i,1\rangle$, where k is the number of solutions. Further measurement of the remaining bits will provide one of these solutions.

An interesting feature of this algorithm is that repeating steps 2 through 4 $\frac{\pi}{4}\sqrt{2^n} \approx O\left(\sqrt{N}\right)$ times is optimal. In particular, if the process is repeated more times, the probability of a successful measurement decreases back toward zero [2].

One drawback of the Grover's algorithm described above is that, in case if M (number of solutions to an N item search) is unknown, then we cannot apply the algorithm directly. It is crucial to know M as it is required in the algorithm for determining the

number of iterations to perform, in order to get the result. This problem can be alleviated by using the Quantum counting algorithm to first estimate the number of solutions M to high accuracy and then applying the quantum search algorithm. Quantum counting is an application of phase estimation procedure to estimate the eigenvalues of the Grover iteration G, which in turn enable us to determine the number of solutions M to the search problem. Function of phase estimation algorithm in Quantum counting is to estimate θ to m-bits of accuracy, with a probability of success at least 1 - ∈. Quantum counting algorithm uses two registers, the first one contains $t \equiv m + \lceil \log(2 + 1/2\epsilon) \rceil$ qubits as per the phase estimation algorithm and second register contains n+1 qubits, enough to implement the Grover iteration on the augmented search space and is initialized to an equal superposition of all possible inputs by a Hadamard transform, as shown in figure 4.2. This gives us the estimate of θ to accuracy 2^{-m} with probability 1-∈ [25].

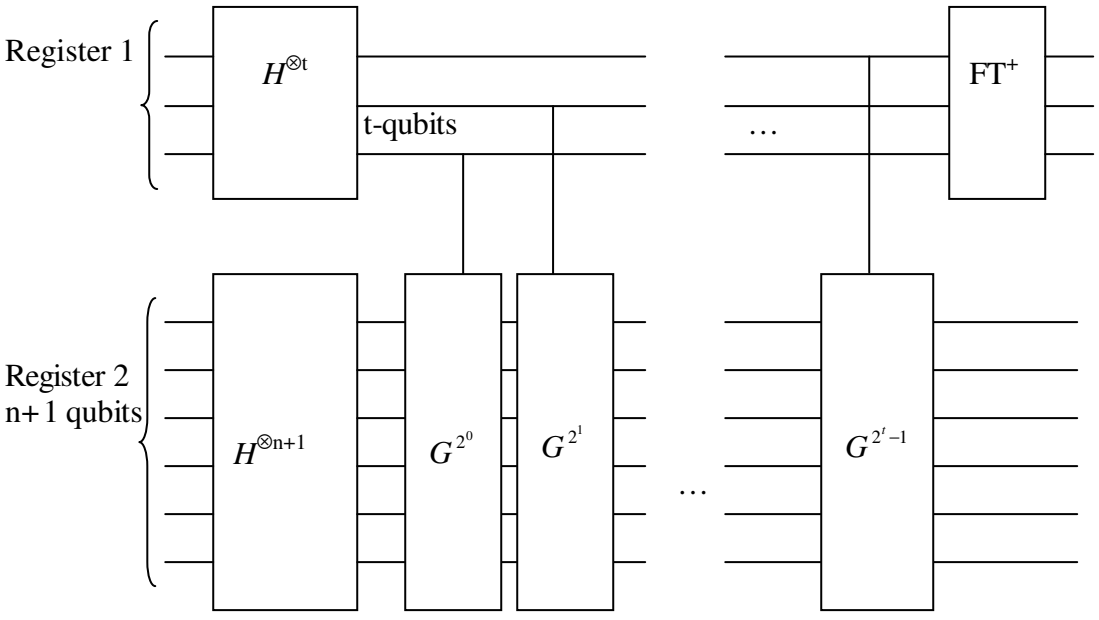

Figure 4.2 Circuit for performing approximate quantum counting on a quantum computer

Chapter 5 GUIDELINES FOR DESIGNING QUANTUM ALGORITHMS

The history of quantum algorithms is very brief. There are two main approaches that have resulted in descriptions of efficient quantum computational algorithms: the first is estimates of periodicity that resulted in the factorization algorithm, and the second is amplitude amplification that has led to Grover's quantum search and related algorithms

With discovery of the quantum factoring algorithm in 1994 a great excitement prevailed among theoretical computer scientists. Quantum computers provided a completely new paradigm for the theory of computation, and this was the first time it had been shown that quantum computation could efficiently solve a problem that had already been established as important in this field. Many people expected a succession of other interesting algorithms to follow. The reality has been disappointing, especially compared with the progress of the rest of the field of quantum information processing. So far, all the quantum algorithms known to offer substantial speed-up over classical algorithms, for the same problems, fall into one of three classes. The first class uses the Fourier transform to find periodicity. This class contains the factoring and discrete logarithm algorithms [31], Simon's algorithm [34] (the first member of this class to be discovered), and Hallgren's algorithms for Pell's equation and certain other number theory problems [17]. There is, in fact a different way of looking at the factoring algorithm that, although it yields basically the same algorithm, puts it into a setting that emphasizes spectral methods rather than periodicity [20], but this approach has not yet yielded any new algorithms. The second class contains Grover's search algorithm, which can perform an exhaustive search of N items in $\sqrt{N}$ time [16], and a number of extensions of this algorithm (see [29]). These extensions all

have the general flavor of giving a square root improvement in the speed of optimization or search problems. The third class consists of algorithms for simulating or solving problems in quantum physics. This class contains Feynman's original idea [13] of using quantum computers to speed up simulations of quantum physics. While not many theoretical papers have yet been written on this class of algorithms, it is clear that if quantum computers are ever developed, this class will be extremely useful in practice. Feynman came up with his idea of using quantum computers to simulate quantum physics in 1982, Simon's algorithm and the factoring algorithm were developed in 1993 and 1994, and Grover came up with his original search algorithm in 1995. Since then, there have been further theoretical developments within each of these classes of algorithms, but no new classes of quantum algorithms have been discovered [32]. In the following discussion we present some guidelines for designing new quantum algorithms (based on views of [1][30]).

Certain properties of quantum computation that are useful for constructing quantum algorithms [30]:

Fact 1: a deterministic computation is performable on a quantum computer if and only if it is reversible [6]. From results on reversible computation [3], this means that we can compute any polynomial time function $f(a)$ as long as we keep the input, a, on the machine. To erase a and replace it with $f(a)$ we need in addition that f is one-to-one and that a is computable in polynomial time from $f(a)$; i.e. both f and f^{-1} are polynomial.

Fact 2: Any polynomial size unitary matrix can be approximated using a polynomial number of elementary unitary transformations [10][6][41] and thus can be approximated in polynomial time on a quantum computer. Further this approximation is good enough so as to introduce at most a bounded probability of error into the results of the computation.

Another interesting result in field of quantum computation is that one obtains quantum search algorithm using quantum simulation. This method works in two steps. First, one make a guess as to a Hamiltonian H which solves the search problem, that is H depends on the solution x and initial state $|\psi\rangle$ such that a quantum system evolving according to H will change from $|\psi\rangle$ to $|x\rangle$ after some prescribed time. Once such an Hamiltonian is found we can simulate the action of such an Hamiltonian using a quantum circuit. Amazingly following this procedure leads very quickly to the quantum search algorithm! This is different from the more conventional approach in two respects: one guesses a Hamiltonian rather than a quantum circuit, and there is no analogue to the simulation step in conventional approach. Thus one see that quantum search algorithm can be re-derived from a deifferent point of view, the point of view of quantum simulation. This opens questions whether this approach can be used to find other fast quantum algorithms? No one has a definitive answer for these questions but it seems certain that this 'quantum algorithms as quantum simulations' point of view offers a useful alternative viewpoint to simulate in the development of quantum algorithms [25].

The following guidelines represent an initial attempt to characterise a methodology for the design and development of quantum algorithms taken from [1].

1. The problem should be expressed in a numerical form, and if it is not, then a method should be employed to convert it into such a form.
2. The initial configuration should be determined.
3. The terminating condition should be defined concisely.
4. The problem should be amenable to being divided into smaller subproblems.
5. The number of universes required should be identified.

6. Each subproblem is assigned its own universe.

7. Computations in the different universes occur in parallel.

8. There must be some form of interaction between all of the universes. The interference must either yield a solution, or new information for the universes to utilise in locating a solution.

The above guidelines provide useful pointers for designing quantum algorithms but they do not help identify the initial configuration. Perhaps there is a direct relationship between the general type of problem and the type of configuration to be used, but this is currently not known. Separating the problem into suitable subproblem components is the most challenging element in the design of quantum algorithms.

On the other hand, there are some problems to be overcome if quantum algorithms are to be as robust and reliable as their classical counterparts.

(a) Quantum factoring is not always guaranteed to work (e.g. the repeating frequency may be odd), but there are known fast algorithms for taking an arbitrary number and determining if it is a factor of another number. Hence, quantum algorithms are probabilistic. The exact relationship between quantum algorithms and currently known classes of probabilistic computation is not fully understood.

(b) Constructing sufficiently large quantum registers which allow for coherent superpositions and entanglement may not be technologically feasible. There must be no outside interference (e.g. heat, noise, vibration) otherwise coherent superpositions will decohere. Simulations of Shor's algorithms on classical machines in the Department of Computer Science at the University of Exeter show that there is an exponential growth required in the number of universes as the number to be factored grows larger (over two

million universes for a 10 digit number, 12 million for a 13 digit number). Deutsch [11] uses this exponential growth within Shor's quantum algorithm to argue that the algorithm provides evidence for the parallel universes interpretation of quantum mechanics. However, this claim is premature, for two reasons. First, Shor's quantum algorithm has not been realised on quantum hardware, except for claims that NMR technology using bulk spin resonances has been used for implementing extremely small scale versions of the algorithm [14]. Secondly, there is the problem of possible exponential growth within a universe in terms of the time required in order for a pattern to repeat itself. One may find that no vertical frequency is identifiable because it is not technologically possible to run a quantum computation for more than a few seconds, or even minutes, whereas for an arbitrary universe to find a repeating pattern may take several hours or even years. All these issues are highly controversial and uncertain, leading to the more pragmatic view that quantum algorithms, if they work, may only work in limited domains and therefore tells nothing about how best to interpret quantum mechanics. Nevertheless, one may be able to perform useful calculations, which are currently beyond the most powerful of classical machine.

(c) There are issues of error correction because of multiple states of a quantum computer, thereby raising question of reliability and repeatability.

(d) Finally, as pointed out above, there is no clear methodology for expressing problems in the NP-hard class as problems of repeating frequencies.

CHAPTER 6 QUANTUM CLIQUE ALGORITHM

A good idea has a way of becoming simpler and solving problems other than that for which it was intended.

Robert Tarjan.

6.1 INTRODUCTION TO CLIQUE PROBLEM

Throughout this chapter G(*v, e*) represents an undirected graph where v = {0,1,2,3,…,n} is the vertex set of G(we use the term *vertex* and *node* synonymously throughout) and $e \subseteq v$ x v is the edge set of G. The symmetric n x n matrix $M_G = (a_{ij})_{(i,j)\in v\times v}$ where $a_{ij} = 1$ if (i, j) $\in e$ is an edge of G and $a_{ij} = 0$ if (i, j) $\notin e$, is called the adjacency matrix of G.

The graph G(*v, e*) is said to be complete if $\forall i, j \in v$ with i ≠ j, we have (i, j) $\in e$. A clique $\boldsymbol{C}$ of graph G(*v, e*) is the $v' \subseteq v$ such that $\forall u, w \in v'$ we have $(u, w) \in e$. i.e. clique is a complete subgraph of G(*v, e*). For example, the following graph have two cliques C_1 and C_2 , each of size 3.

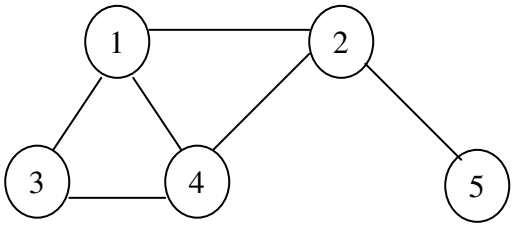

Figure 6.1 Graph showing cliques C_1= 1,3,4 and C_2 = 1,4,2

Clique as a decision problem is defined as

Clique(G, k) :A Graph G = (v, e) with a set v of vertices and a set $e \subseteq v \times v$ of edges, $|v| = n$, $k \in N$, $k \leq n$. Does G have a k-clique, i.e., does there exist a set $v' \subseteq v$ with $|v'| = k$, so that for every pair $v_1, v_2 \in v'$, $(v_1, v_2) \in e$?

Clique problem is one of the first problems shown to be NP-Complete, which means that unless P= NP, a fact that is widely believed to be false, exact algorithms are guaranteed to return a solution only in a time which increases exponentially with the number of vertices in the graph [18]. For example, complexity of algorithms that uses *k-interchange* heuristics is given by $O(n^k)$, where k represents the neighborhood size. Larger the neighborhood size, better will be the quality of results [26]. In fact, Clique(G, k) is strong NP-complete as there do not exist any pseudo-polynomial algorithms unless P = NP [39]. The PCP-theory proves the clique problem as not solvable by approximation with reasonable results [24]. The strong NP-completeness and the hopeless outlook for a good result by means of approximation show the hardness of finding cliques in graphs by sequential algorithms.

6.2 Q-CLIQUE ALGORITHM

Quantum Clique algorithm (Q-Clique Algorithm) uses Grover's search algorithm and quantum counting algorithm. For a graph G with n nodes our search space will be the superposition of n-qubit, that is it will contain all possible combinations of n nodes. This will contain both valid and invalid cliques (because there cannot be any clique with less than 3 nodes, all such combinations will act as invalid cliques.). In terms of search problem, Clique problem can be stated as

For G(v, e) with n nodes, search space contains $N=2^n$ possible combinations of nodes. We have to find all the cliques (i.e. solutions M) in G.

We will first use quantum-counting algorithm to get an estimate of M, i.e. number of cliques, then we apply Grover's quantum search algorithm to get the cliques. In order to find M solutions in an N-item search space Grover's search algorithm requires $O(\sqrt{\frac{N}{M}})$ queries to the Oracle. The oracle will be an efficient quantum circuit that implements the function $f(x) = 1$ if x is a clique and $f(x) = 0$ otherwise. A classical circuit that implements this function is first designed. Using the techniques of reversible computation, its corresponding classical reversible circuit is designed, which takes (x , q) - representing an input register initially set to x and a one bit output register initially set to q - to (x, q $\oplus$ $f(x)$) . Now this reversible circuit can immediately be translated into a quantum circuit that takes $|x\rangle|$ q$\rangle$ to $|x\rangle|$ q $\oplus$ $f(x)\rangle$, as required by the oracle.

Oracle of Q-Clique Algorithm uses AND function for determining whether the x represents clique G(x), where $x = \sum_{i=0}^{n-1} x_i$ is an n-bit sequence of 0's and 1's and ith bit denotes v_i , the ith node of graph G(v ,e). AND function performs the column wise logical $\wedge$ (***and***) on the rows $R_{i_1}\ R_{i_2}\ \ldots R_{i_m}$ for indexes i_1, i_2, …, i_m specified in the AND function as an argument. This operation can be represented as

$$AND(R_{i_1}, R_{i_2}, \ldots, R_{i_m}) = C_0C_1C_2\ldots C_{n-1},$$

Where $C_j = a_{i_1 j} \wedge a_{i_2 j} \wedge \ldots \wedge a_{i_m j} \quad \forall j = 0,1,2,\ldots,n-1$ where n is the number of columns and $\wedge$ is the logical ***and***. $a_{i_m j}$ is the j[th] element of row R_{i_m} and i_m represents the index of the row and can be anything from 0 to n-1. Classical function that implements this oracle is as follows:

1. Store the bit positions of x for which $x_i = 1$. Let's denote these bit positions as i_1, i_2, i_3, …, i_m where m represents the number of 1's in x. If $m < 3$ then function outputs 0 (because there cannot be any clique between nodes less than 3) and exits, otherwise move on to next step.
2. Apply $AND(R_{i_1}, R_{i_2}, \ldots, R_{i_m})$. Column wise compare the result of AND with x. There equality indicates a clique and thus function returns 1 as output, otherwise function returns 0.

The time complexity of this classical function is $O(n^2)$. From this we can estimate the classical circuit complexity for this function. Although circuit complexity is in some sense incomparable with time complexity because, for each language L (even non-recursive L) there is a constant c such that $C_L(n) \leq c^n$. However, there is a basic relationship in one direction between these two notions of complexity: circuit complexity provides a related lower bound on time complexity [37]. [28] showed that if $L \in \text{DTIME}(T(n))$ then $C_L(n) = O(T(n)^2)$. [27] improved this to $C_L(n) = O(T(n) \log T(n))$. Thus the circuit complexity of our oracle function becomes $O(n^2 \log n)$ (because $\log(n^2) = 2\log(n)$). The complexity of the reversible circuit constructed from this circuit will be almost the same as that of the irreversible circuit. This circuit can then be implemented using quantum circuits.

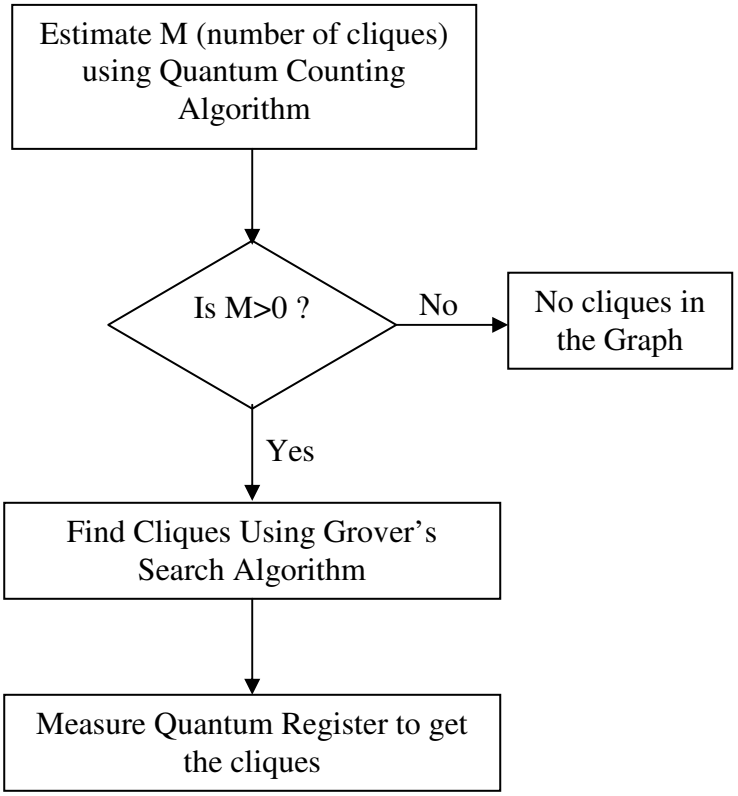

Figure 6.2 Flowchart of Q-Clique Algorithm

Steps for Q-Clique Algorithm

1. Apply Quantum Counting algorithm as shown in the figure 4.2.
    a. Take a t-qubit register where $t \equiv m + \lceil \log(2+1/2\epsilon \rceil$ qubits where m is the bits of accuracy and ϵ is the probability of failure.
    b. Take a second register of $n+1$ qubits, where n is the number of nodes in graph G and one extra qubit is used as oracle qubit. Grover's iteration $G=2(|\psi\rangle\langle\psi| - I)O$, will be performed on this register.
    c. Initialize the first register as $|0\rangle^{\otimes t}$. Initialize second register as superposition of all possible inputs $\sum_x |x\rangle$ by using Hadamard Transform.

d. Apply the Phase Estimation Procedure (as given in section 4.2). This will give us the estimate of θ to *m*-bits of accuracy.

e. Using equation 6.1 calculate M.

$$\sin^2 \theta/2 = M/2N \qquad (6.1)$$

2. Calculate R which denotes number of Grover's iterations needed to be performed.

$$R = CI\left(\frac{\arccos\sqrt{M/N}}{\theta}\right) \qquad (6.2)$$

where *CI*(x) denotes the integer closest to the real number x.

3. Apply the Grover's Search Algorithm

a. Start with the initial state $|0\rangle^{\otimes n}$.

b. Apply Hadamard transform $H^{\otimes n}$ to first n qubits and *HZ* to the last qubit. This puts the system in the following state

$$|\psi\rangle = \frac{1}{\sqrt{N}}\sum_{x=0}^{N-1}|x\rangle\left[\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right] \qquad (6.3)$$

c. Apply the following Grover's iteration *R* times (as calculated in step 2.)

i. Apply the Oracle O.

ii. Apply the Hadamard transform $H^{\otimes n}$.

iii. Perform a conditional phase shift on computer with every computational basis state except $|0\rangle$ receiving a phase shift of –1,

$$|x\rangle \rightarrow -(-1)^{\delta_{x0}}|x\rangle \qquad (6.4)$$

iv. Apply the Hadamard transform $H^{\otimes n}$.

Above steps (from i to iv) can be represented by the following unitary operation

$$\left[\left(2|\psi\rangle\langle\psi| - I\right)O\right]^{\otimes R} \qquad (6.5)$$

4. Measure the last qubit of the quantum state, representing $f(x)$. Because of the amplitude change, there is a high probability that the result will be 1. If this is the case, the measurement has projected the state onto the subspace $\frac{1}{\sqrt{2^M}}\sum_{i=1}^{2^M}|x_i,1\rangle$, where M is the number of solutions. Further measurement of the remaining bits will provide one of these solutions.

Complexity of this algorithm is $O(\sqrt{N/M})$ where $N = 2^n$. In Grover's search algorithm it is assumed that $M \leq N/2$. When $M > N/2$, it can be shown that number of iterations increases as M varies from $N/2$ to N. That is the reason for including both valid and invalid entries of the cliques so that M is always less than $N/2$. For $M = N/4$, which is a special case of Grover's algorithm, a solution is found with certainty in a single iteration.

6.3 RESULTS

We compare the number of iterations required by the classical algorithm and Q-clique algorithm for solving Felt, Moon-Moser and Tarjan graphs.

Definition A felt graph $G_n = (V_n, E_n)$ of complexity n contains n components S_1, S_2, ..., S_n with (1) every component S_i is a circle with 5 nodes, $1 \leq i \leq n$,

(2) $\forall i \forall j \forall u \forall v (1 \leq i \neq j \leq n \wedge u \in V_i \wedge v \in V_j \rightarrow (u, v) \in E_n)$.

Definition A Moon-Moser graph $G_n = (V_n, E_n)$ of complexity n contains n components S_1, S_2, ..., S_n with (1) every component S_i has 3 isolated nodes, $1 \leq i \leq n$

(2) $E_n = \{(u, v) : u \in S_i \wedge v \in S_j \wedge i \neq j\}$.

Definition A Tarjan graph $G_{m,n} = (V, E)$ contains 2n crowds K^0_1 , K^1_1, K^0_2, K^1_2, … K^0_n , K^1_n, each with m nodes. Two nodes $u \in V^x_i$ and $v \in V^y_j$ are not adjacent if i = j and x ≠ y.

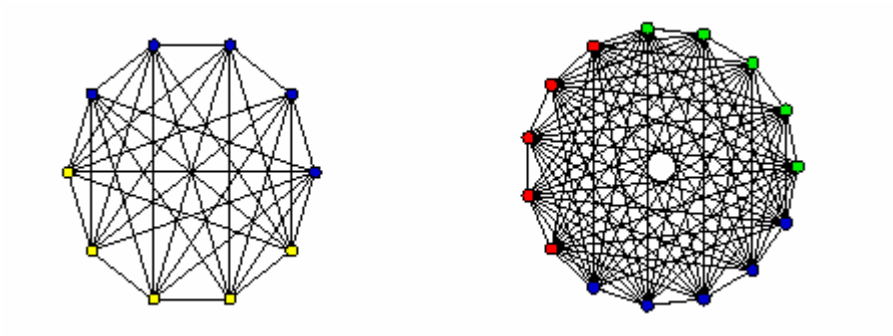

Figure 6.3 Felts with complexity 2 and 3

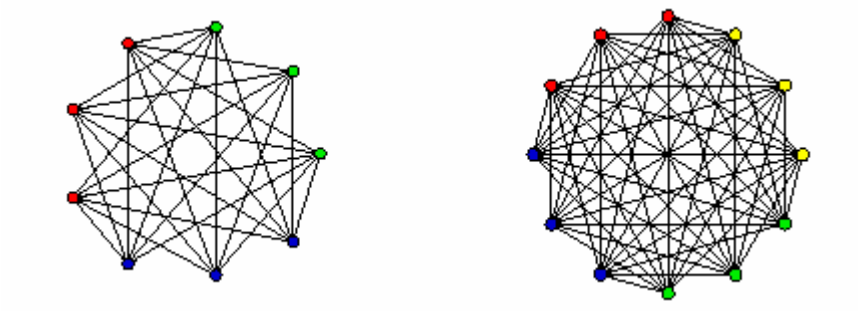

Figure 6.4 Moon-Moser Graphs with complexity 3 and 4

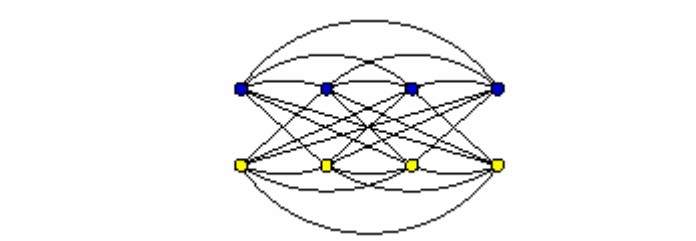

Figure 6.5 Tarjan Graph $G_{m,4}$. Every node in $G_{m,4}$ is a crowd with m vertices

Conclusion: For all n > 0

(1) a felt G_n contains 5^n cliques,

(2) a Moon-Moser graph G_n contains 3^n cliques,

(3) a Tarjan graph $G_{m,n}$ contains 2^n cliques.

	Nodes	**Cliques in Graph G**	**Classical Algorithm (2^n) (no. of iterations)**	**Q-Clique Algorithm ($2^n/ M)^{0.5}$ (no. of iterations)**
Tarjan n′ = 4	8	16	256	4
Moon-Moser n′ = 3	9	27	512	5
Felt n′ = 2	10	25	1024	7
Moon-Moser n′ = 4	12	81	4096	8
Felt n′ = 3	15	125	32768	17

Table 6.1 Comparison of Classical and Quantum algorithm for Clique problem

The graphs chosen are those in which its hard to find cliques. The above table shows theoretical results based on the complexity of the algorithms to give an idea of the speed-up provided by the Q-clique algorithm. As can be seen from the above table there is a great speed-up provided by the Q-Clique algorithm as compared to the classical algorithm in terms of number of iterations required to find the given number of cliques.

CHAPTER 7 CONCLUSION

"Nature uses only the longest threads to weave her patterns, so that each small piece of her fabric reveals the organization of the entire tapestry."

Richard Feynman

7.1 CONCLUSION

In the plethora of complexity classes, an attempt has been made to classify the problems into three categories, from view of quantum computing – problems that can be efficiently solved on quantum computers – Shor's Factoring algorithm, Discrete Logarithms, Quantum Simulation, problems for which quantum computers either cannot provide efficient solutions or exponential speedups, than those provided by their classical counterparts – P, ZPP, BPP complexity classes, Grover's search algorithm and problems which cannot be solved efficiently on quantum computers, that is they require exponential resources even on quantum computers – NP-complete problems. Although NP-complete problems cannot be solved efficiently on quantum computer, it can still provide speed-ups for some of the NP-complete problems with respect to their classical counterparts [25]. We recognize one such problem in the thesis and show how it can be solved on quantum computer.

The field of quantum computation and information theory has seen significant developments but till now only few quantum algorithms like Shor's algorithm and Grover's algorithm, have been designed that take advantage of features of quantum computers. We present some guidelines on how to design quantum algorithms and features to take into account while designing them.

We show that Clique problem can be solved on quantum computer with complexity $O(\sqrt{N/M})$ where $N = 2^n$ as compared to their complexity $O(2^n)$ on classical computers. Although it does offer a speed-up from its classical counterpart, it still requires exponential resources. Hence we place this problem into the third category of our complexity class – Class of problems, which cannot be solved efficiently on quantum computers.

7.2 FUTURE SCOPE

The field of quantum information theory and quantum complexity theory has expanded significantly. But the development of quantum algorithms lagged behind, with barely any new significant algorithms having been discovered in the last five years. There can be two reasons for this lag; the first possible reason is that quantum computers operate in a manner so different from classical computers that our techniques for designing algorithms and our intuitions for understanding the process of computation no longer work. The second reason is that there really might be relatively few problems for which quantum computers can offer a substantial speed-up over classical computers, and we may have already discovered many or all of the important techniques for constructing quantum algorithms. But this seems quite unlikely, as any quantum algorithm offering a speed-up over classical computation must use interference: a phenomenon which is unknown in classical computer science, and most theoretical computer scientists are not used to reasoning about it. Thus, it seems quite likely that several new and significant quantum algorithmic techniques have yet to be discovered [32].

Much of the research into new quantum computer algorithms has been spent looking for super-polynomial speed-ups. While these do not occur in Grover's algorithm

and its extensions, they can occur in the classes of quantum algorithms that use periodicity finding and that simulate quantum physics. Super-polynomial speed-ups cannot arise from problems that have polynomial-time classical algorithms, so researchers have been concentrating on problems that are not in the classical computational class P. The first class of problems not in P that come to mind are the NP-complete ones [32]. A quantum algorithm solving NP-complete problems in polynomial time would be a momentous discovery, but it is believed that the most likely scenario is, that this is not possible.

It has been proved that there is an oracle, relative to which NP-complete problems cannot be solved in polynomial time on quantum computers [5]. If we assume that NP-complete problems are not solvable efficiently on a quantum computer, then in order to achieve a super-polynomial speed-up, we must look within the class of problems which are neither NP-hard nor in P. There are only a relatively small number of well-studied problems that are suspected to be in this class. No general theory is known for these problems, and relatively few of them are known to be reducible to each other, so they all must be considered individually. It may be that many of these problems do not indeed have polynomial-time algorithms on quantum computers. People have to date concentrated their efforts on those problems that appear to have structure related to periodicity, thus providing a possible means of attack. These include the problems of graph isomorphism and that of approximating short vectors in a lattice. Neither of these problems has yet yielded to a quantum attack [32].

One research area that might be worth exploring is to try to find faster quantum algorithms for problems already known to be classically solvable in polynomial time. This approach is limited to providing polynomial factor speed-ups. While this is certainly less

exciting than finding super-polynomial speed-ups, it could nevertheless yield new techniques for designing quantum algorithms. At this point, any new techniques have the potential to be of great value in further exploration of quantum algorithms, and, if this avenue can help discover such new techniques, it should be pursued [32].

REFERENCES

[1] Ajit Narayanan, "Quantum Algorithms", Department of Computer Science, University of Exeter, UK.

[2] Bassam Aoun, Mohamad Tarifi, "Quantum Artificial Intelligence", Jan 2004.

[3] Bennet C. H., "Logical Reversibility of Computation", IBM J. Res. Develop., vol 17 pp525-532, 1973.

[4] Bernhard Omer, "Quantum Programming in QCL", pp 15 – 30, Jan 2000.

[5] Bennet C., Bernstein E., Brassard G., and Vazirani U., "Strengths and weaknesses of quantum computation", Special issue on Quantum Computation of the Siam Journal of Computing, Oct. 1997.

[6] Bernstein E., Vazirani U., "Quantum complexity theory", Proceedings of the 25th Annual Symposium on the Theory of Computing, ACM, New York, pp11-20, 1993.

[7] Bernstein, E. and Vazirani, U., ``Quantum complexity theory,'' Special issue on Quantum Computation of the Siam Journal of Computing, Oct. 1997.

[8] Chsritian Paquin, "Computing in the Quantum World", Universite de Montreal, July 1998.

[9] Coppersmith D., "An Approximate Fourier Transform Useful in Quantum Factoring", IBM Research Report No. RC19642, 1994.

[10] Deutsch D., "Quantum Computational Networks", Proc. R. Soc. Lond., Vol. A425 pp 73-90, 1989

[11] Deutsch D., "The Fabric of Reality", Allen Lane, 1997.

[12] Eric Benjamin, Kenny Huang, Amir Kamil, Jimmy Kittiyachavalit, "Quantum Computability and Complexity and the Limits of Quantum Computation", University of California, Berkeley, Dec 2003.

[13] Feynman R., "Simulating physics with computers", Internat. J. Theoret. Phys. 21, pp 467–488, 1982.

[14] Gershenfeld N., and Chung I., "Bulk spin resonance quantum computation", Science, 275 :350, January 17, 1997.

[15] Glassner A., "Quantum computing, Part 3", Andrew Glassner's Notebook, Nov/ Dec 2001. website: www.glassner.com.

[16] Grover L. K., "Quantum mechanics helps in searching for a needle in a haystack.", Phys. Rev. Lett. 78, pp 325–328, 1997.

[17] Hallgren S., "Polynomial-time quantum algorithms for Pell's equation and the principal ideal problem", In Proceedings of the 34th Annual ACM Symposium on Theory of Computing. ACM, New York, pp 653–658, 2002.

[18] Johnson D., Gary M., "Computers and Intractability – a guide to the Theory of NP-Completeness", Freeman, 1979.

[19] Jyh Ying Peng, "Quantum Computation Lecture Notes (based on lecture notes by John Preskill 1998)", 2003.

[20] Kitaev A. Yu., "Quantum measurements and the Abelian stabilizer problem", ECCC Report TR96-003. Los Alamos archive, e-print quant-ph/ 9511026, 1996.

[21] Kummer W., Trausmuth R., "Quantentheorie", Skriptum zur Vorlesung 131.869, 1988.

[22] Ladner R., "On the structure of polynomial time reducibility", Journal of the ACM 22:155-171, 1975.

[23] Matthew Hayward, "Quantum Computing and Shor's Algorithm", 2002. website: http:/ / alumni.imsa.edu/ ~ matth/ quant/

[24] Mayr E. W., Prömel H. J., and Steger A., Lectures on Proof Verification and Approximation Algorithms.", Springer, 1998.

[25] Michael A. Nielsen, Isaac L. Chuang, "Quantum Computation and Quantum Information", Cambridge University Press.

[26] Panos M. Pardalos, Marcello Pelillo, Marco Budinich, Immanuel M. Bomze, "Maximum Clique Problem", Kluwer Academic Publishers, 1999.

[27] Rabin M. O., Fischer M. J., "Super-exponential complexity of Presburger arithmetic", In Complexity of Computation, SIAM-AMS Proceedings, vol. 7, R. Karp, Ed. American Mathematical Society, Providence, R.I., pp 27-42, 1974.

[28] Savage J. E., "Computational work and time on finite machines.", J. ACM 19, pp 660 – 674, 1972.

[29] Sengupta A. M., Grover L. K., "From coupled pendulums to quantum search" In Mathematics of Quantum Computation. R. K. Brylinski and G. Chen, Eds. Chapman & Hall/ CRC, Boca Raton, FL, pp 119–134, 2002.

[30] Shor P. W., "Algorithms for quantum computation: Discrete Log and Factoring", extended abstract, AT& T Bell Labs, USA.

[31] Shor P. W., "Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer", *SIA M J. Comput. 26*, pp1484–1509,1997.

[32] Shor P. W., “Why Haven’t More Quantum Algorithms Been Found?”, Journal of the ACM, Vol. 50, No. 1, January 2003.

[33] Simon D., ``On the power of quantum computation," Proceedings of the 35th Annual IEEE Symposium on Foundations of Computer Science, pp. 116-123, 1994.

[34] Simon D., “On the power of quantum computation”, *SIA M J. Comput. 26*, pp1474–1483, 1997.

[35] Sipser M., “Introduction to the Theory of Computation”. PWS Publishing Company, 1997.

[36] Steane M., “A quantum computer only needs one universe," lanl e-print quant-ph/ 0003084, March 2003.

[37] Stockmeyer L., Meyer A. R., "Cosmological Lower Bound on the Circuit Complexity of a Small Problem in Logic", Journal of the ACM, Vol. 49, No. 6, pp. 753-784, November 2002.

[38] Tom Carter, “A brief overview of quantum computing or, Can we compute faster in a multiverse ?” , May 1999. website: http:/ / cogs.csustan.edu/ ~ tom/ quantum.

[39] Wegener I., “Theoretische Informatik”, Teubner, 1993.

[40] Wim van Dam, “On Quantum Computation Theory”, 2002.

[41] Yao A., “Quantum circuit complexity”, Proc. 34th IEEE Symp. on Foundations of Computer Science, 1993.