Quantum Computation & Quantum Information Directed Reading Chapter Notes

Peter Karalekas
Advisor: Michael Fischer

April 30, 2015

1 Chapter 1: Introduction and Overview

1.1 History of quantum computing

There are four fields which have contributed to the fundamental ideas of quantum computation and quantum information—quantum mechanics, computer science, information theory, and cryptography. Knowing about the history of each field is essential to understanding the context of quantum computing.

1.1.1 Quantum mechanics

At the turn of the twentieth century, physicists were beginning to realize that there were holes in the classical theories of physics. These theories were predicting nonsensical phenomena like the *ultraviolet catastrophe* or the spiraling of electrons into the nucleus, and many of the ad hoc fixes to classical physics were unsettling with members of the scientific community. In the 1920s, the creation of quantum mechanics solved these issues, and has been the most complete description of reality ever since. Despite being a wonderful framework for physics, quantum mechanics has had its critics. Its best known critic, Albert Einstein, went to his grave without fully coming to terms with the theory that he had indirectly invented. Now, some of the most striking phenomena in quantum mechanics, like *entanglement* and the *no-cloning theorem*, are offering exicting new ways to develop algorithms in quantum computing.

1.1.2 Computer science

Although the algorithmic ideas at the heart of computer science have been around for centuries, the modern version of computer science began with Alan Turing's paper in 1936. Building on his ideas over the years has resulted in what is known as the *strong Church-Turing thesis*, which states that any algorithmic process can be simulated efficiently using a probablistic Turing machine. A Turing machine, which is analogous to our current notion of a programmable computer, is currently our benchmark for efficiently solving problems. However, with the arrival of quantum computing, some quantum algorithms have offered polynomial and even exponential speedups over the best known classical algorithms. Therefore, although it has yet to be proven, it is believed that once a quantum computer is successfully implemented, the Universal Quantum Computer will be our new benchmark for efficiency.

1.1.3 Information theory

During the computer science revolution in the early twentieth century, Claude Shannon published a pair of papers in 1948 that laid the foundation for the modern theory of information and communication. His goal was to mathematically describe the concept of information, and he was especially interested in how information could be transmitted over a communications channel while being protected against the channel's noise. Through his interest, Shannon developed the *noiseless channel coding theorem* and the *noisy channel coding theorem*, which quantified the resources necessary to transmit information and how reliable the transmission could be in the presence of noise. And, to allow for reliable communication over a noisy channel, Shannon showed that *error-correcting codes* could be implemented to undo errors brought about by interference. Due to the fact that information is at the heart of quantum computing, these developments in information theory have been invaluable for many aspects of the field, such as *quantum cryptography* and *quantum error-correction*.

1.1.4 Cryptography

Like algorithms and problem solving, cryptography has been around since the earliest written records of human existence. Today, cryptography is a rich field full of various cryptographic protocols for communicating between parties that may or may not trust one another. These protocols are split between that of *private key cryptosystems*, where two parties share a key for encryption and decryption that only they know, and *public key cryptosystems*, where one party publishes a key to the general public, but the nature of the encryption scheme makes it extremely difficult for anyone other than the key-publishing party to perform decryption. Currently, the best known form of public key cryptography is the *RSA cryptosystem*, which depends on the intractable problem of factoring to provide security. However, one of the most well known quantum algorithms, *Shor's algorithm*, can perform factoring in polynomial time, effectively rendering RSA useless. Although quantum computing poses problems for classical cryptosystems, it also offers ways to increase security. *Quantum key distribution*, which uses the postulates of quantum mechanics to exchange private keys securely, offers a new form of private key cryptography that can defend against the revolution of quantum computing.

2 Chapter 2: Quantum Mechanics

2.1 Linear algebra

2.1.1 Vectors and operators

In quantum mechanics, vertical vectors are represented by $|\psi\rangle$, known as a "ket", and horizontal vectors are represented by $\langle\psi|$, known as a "bra." However "bra" vectors are not just the transpose of "ket" vectors, they are the complex conjugate transpose, a transformation known as *adjoint* which we discuss later. This notation is known as Dirac notation and is extremely useful in quantum mechanics. The following are examples of typical vectors (also known as states):

$$\text{Ground state} = |0\rangle = \begin{pmatrix}1\\0\end{pmatrix} \qquad \text{Excited state} = |1\rangle = \begin{pmatrix}0\\1\end{pmatrix}$$

When two states form a basis, then any vector in the state space can be written as a linear combination of the basis states. For example, a vector on the Bloch sphere (which represents the state of a single qubit) can be expressed as:

$$|\psi\rangle = a\,|0\rangle + b\,|1\rangle$$

Where a and b are complex numbers and $a^2 + b^2 = 1$. Also, in Dirac notation operators are typically expressed as capital letters. Some common operators in quantum mechanics include the Pauli matricies, which are as follows:

$$\sigma_0 = I = \begin{pmatrix}1 & 0\\0 & 1\end{pmatrix} \qquad \sigma_x = X = \begin{pmatrix}0 & 1\\1 & 0\end{pmatrix} \qquad \sigma_y = Y = \begin{pmatrix}0 & -i\\i & 0\end{pmatrix} \qquad \sigma_z = Z = \begin{pmatrix}1 & 0\\0 & -1\end{pmatrix}$$

To apply an operator to a state vector, we typically use the notation $X\,|\psi\rangle$ to show that the X operator is being applied to the state ψ. Dirac notation is convenient, but we can easily check our answers in quantum by simply multiplying the matricies that correspond to operators and the vectors that correspond to states. For example, the X Pauli matrix is in effect a bit flip operator, and therefore $|1\rangle = X\,|0\rangle$. Multiplying the X matrix by the ground state vector will show that this is true.

Finally, there exists a special kind of operator transformation in quantum mechanics known as the *adjoint* or *Hermitian conjugate*. The adjoint of an operator A is denoted by $A^\dagger$, and is equivalent to $(A^T)^*$ or the complex conjugate of the transpose of A. An operator A is known as a *Hermitian* operator if $A^\dagger = A$, and an operator U is said to be *unitary* if $U^\dagger U = I$.

2.1.2 Products between states and operators

There are multiple ways to combine states in quantum mechanics that have clear parallels to linear algebra. For example, the *inner product* between two states, represented by $\langle\phi\,|\,\psi\rangle$, is equivalent to the dot product between the vectors $|\phi\rangle$ and $|\psi\rangle$. Therefore, an inner product between to states always results in a scalar. For a normalized state, the inner product with a state and itself will be 1. In addition to the inner product, states also have an *outer product*, represented by $|\phi\rangle\,\langle\psi|$. Because the outer product is a vertical vector multiplying a horizontal vector, the result of an outer product is always a matrix.

A third product, known as the *tensor product*, can be used both between states and between operators. Stated abstractly, the tensor product is a way of putting vector spaces together to form larger vector spaces. However, we can view tensor products with a convenient matrix representation known as the *Kronecker product* in order to better understand it. The tensor product of the vectors $|a\rangle = (1, 2)$ and $|b\rangle = (2, 3)$, denoted by $|a\rangle \otimes |b\rangle$, is:

$$\begin{pmatrix} 1 \\ 2 \end{pmatrix} \otimes \begin{pmatrix} 2 \\ 3 \end{pmatrix} = \begin{pmatrix} 1 \times 2 \\ 1 \times 3 \\ 2 \times 2 \\ 2 \times 3 \end{pmatrix} = \begin{pmatrix} 2 \\ 3 \\ 4 \\ 6 \end{pmatrix}$$

And, the tensor product of the Pauli matricies X and Y is:

$$X \otimes Y = \begin{pmatrix} 0 \cdot Y & 1 \cdot Y \\ 1 \cdot Y & 0 \cdot Y \end{pmatrix} = \begin{pmatrix} 0 & 0 & 0 & -i \\ 0 & 0 & i & 0 \\ 0 & -i & 0 & 0 \\ i & 0 & 0 & 0 \end{pmatrix}$$

Thus, the tensor product of two vectors or matricies results in a vector or matrix with dimensions equal to the product of the dimensions of the two items in the tensor product.

2.1.3 Commutators

The *commutator* between two operators A and B is definted to be:

$$[A, B] = AB - BA$$

If $[A, B] = 0$, we say that A commutes with B. In quantum, whether or not an operator commutes with another operator tells us many important things about how those two are related. Specifically, if two Hermitian operators commute, then it is possible to simultaneously diagonalize those operators. More abstractly, this means that two commuting operators that represent observables in a quantum system can be measured and known at the same time. On the other hand, the fact that many observables in fact do not commute leads to many of the strange features of quantum mechanics.

2.2 Postulates of quantum mechanics

In this subsection, we will cover the four postulates of quantum mechanics and how and when they can be applied to some of the strange features of the quantum world.

2.2.1 State space

Postulate 1: Associated to any isolated physical system is a complex vector space with inner product known as the state space of the system, and the system is completely described by its state vector, which is a unit vector in the system's state space.

A complex vector space with inner product is also often referred to as a *Hilbert space*, and comes up frequently in the discussion of quantum mechanics.

2.2.2 Evolution

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 t_2 by a unitary operator U that depends only on the times t_1 and t_2.

$$|\psi'\rangle = U\,|\psi\rangle$$

This postulate describes how the quantum states of a closed quantum system at two different times are related, but a more refined version of this postulate can be given that describes the evolution of the quantum system in continuous time.

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

$$i\hbar\frac{d\,|\psi\rangle}{dt} = H\,|\psi\rangle$$

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 of $\hbar$ into H, effectively setting $\hbar = 1$. H is a fixed Hermitian operator known as the *Hamiltonian* of the closed system.

The Hamiltonian of a system describes its dynamics completely. In addition, the eigenstates and eigenvalues of the Hamiltonian are special. These eigenstates $|e\rangle$ are known as the *energy eigenstates* or *stationary states* of the system, and the eigenvalues E are the energies corresponding to each eigenstate. In addition, we can relate our discrete and continuous versions of system evolution using a simple trick. By exponentiating a Hamiltonian operator, we can produce a unitary operator as follows:

$$U(t_1, t_2) = exp\left[\frac{-iH(t_2 - t_1)}{\hbar}\right]$$

2.2.3 Quantum measurement

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 experiement. If the state of the quantum system is $|\psi\rangle$ immediately before the measurement then the probability that result m occurs is given by:

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

And the state of the system after the measurement is:

$$\frac{M_m\,|\psi\rangle}{\sqrt{\left\langle\psi\,\middle|\,M_m^\dagger M_m\,\middle|\,\psi\right\rangle}}$$

2.2.4 Composite systems

Postulate 4: The state space of a composite physical system is the tensor product of the state spaces of the component physical systems. Moreover, if we have systems 1 through n, and system number i is prepared in the state $|\psi_i\rangle$, then the joint state of the total system is $|\psi_1\rangle \otimes |\psi_2\rangle \otimes \ldots \otimes |\psi_n\rangle$.

For example, if we create a two qubit system, the state vector for the composite system will be:

$$|\psi\rangle = \alpha_{00}\,|00\rangle + \alpha_{01}\,|01\rangle + \alpha_{10}\,|10\rangle + \alpha_{11}\,|11\rangle$$

Where the basis vectors are:

$$|00\rangle = |0\rangle \otimes |0\rangle \quad |01\rangle = |0\rangle \otimes |1\rangle \quad |10\rangle = |1\rangle \otimes |0\rangle \quad |11\rangle = |1\rangle \otimes |1\rangle$$

2.3 Entanglement

One of the most counterintuitive yet important aspects of quantum mechanics is the idea of entanglement. Entanglement occurs when a composite quantum system is generated or interacts in such a way that the state vectors of the individual systems cannot be described independently. Instead, a state vector can only be given for the composite system. In terms of classical intuition, entanglement can be thought of as a perfect correllation between measurement outcomes of the individual quantum systems within a composite system.

2.3.1 Superdense coding

Although entanglement is a physical phenomenon, it can also be thought of as a physical resource, capable of storing additional information. One application of entanglement that uses this feature is *superdense coding*, which allows for the transfer of two classical bits of information by only sending one qubit. More formally, we suppose that Alice is in possession of two classical bits of information that she wants to send to Bob, but can only send Bob a single qubit. To do this, Alice and Bob share a pair of qubits in the entangled state:

$$|\psi\rangle = \frac{|00\rangle + |11\rangle}{\sqrt{2}}$$

Now, each has a single qubit from the entangled pair in his or her possession, and the protocol can begin. If Alice wants to send Bob the bit string '00', she does nothing to her qubit. If she wants to send '01', she instead applies the phase flip gate Z to the her qubit. If she wants to send '10' she applies the bit flip gate X to her qubit. And, if she wants to send '11', then she applies the iY gate to her qubit. The four resulting states of these choices are as follows:

$$00 : |\psi\rangle \rightarrow \frac{|00\rangle + |11\rangle}{\sqrt{2}}$$

$$01 : |\psi\rangle \rightarrow \frac{|00\rangle - |11\rangle}{\sqrt{2}}$$

$$10 : |\psi\rangle \rightarrow \frac{|10\rangle + |01\rangle}{\sqrt{2}}$$

$$11 : |\psi\rangle \rightarrow \frac{|01\rangle - |10\rangle}{\sqrt{2}}$$

These four states form an orthonormal basis, and therefore once Alice sends her qubit to Bob, he can measure in that basis and determine the string that Alice wished to send.

2.3.2 EPR and Bell

The four states written above are known as the *Bell states* or *EPR pairs* in honor of some of the first scientists who wrestled with the idea of entanglement. In the famous EPR paradox paper, Einstein, Podolsky, and Rosen set out to prove that something was missing in the formulations of quantum mechanics. They recognized that entanglement was completely against their current understanding of the physical world, and therefore believed that the only other option was that *hidden parameters* described all the information in quantum mechanics. Rather than accept the idea of entanglement, they believed that these hidden parameters meant that quantum mechanics was an incomplete and incorrect theory of reality. However, many years later, the paradox was resolved experimentally, using what is known as the *Bell inequality*. By working out the mathematics for both entanglement and hidden parameters, and comparing these formulas to experimental results, it was determined that entanglement is in fact an observable physical phenomenon and that quantum mechanics is the correct description of reality.

3 Chapter 3: Computer Science

3.1 The analysis of computational problems

In order to understand computational problems, it is necessary to understand the resources required to solve them. The amount of time, space, and energy that an algorithm requires to solve a problem is known as the *computational complexity* of that algorithm, and provides an important framework for thinking about how to design algorithms, both classical and quantum alike.

3.1.1 Asymptotic notation

One way that computer scientists simplify the study of computational complexity is through the use of *asymptotic notation*. The first tool in asymptotic notation, known as *big O notation*, sets upper bounds on the behavior of a function. If we say that a function $f(n)$ is $O(g(n))$, we know that all values in $f(n)$ are necessarily bounded above by $g(n)$. In addition, there exists *big Omega notation* that sets lower bounds on the behavior of a function. If we say that a function $f(n)$ is $\Omega(g(n))$, we know that all values in $f(n)$ are necessarily bounded below by $g(n)$. Finally, our third tool, known as *big Theta notation*, combines the properties of the first two tools. That is, if we say that a function $f(n)$ is $\Theta(g(n))$, we know that $f(n)$ and $g(n)$ behave the same asymptotically.

3.1.2 Classical complexity classes

When talking about the resources necessary to perform an algorithmic process, we typically use O notation to set upper bounds on the time or space requirements of a computation. With this notation, complexity theorists have devised a number of complexity classes that are comprised of problems with the same characteristics. The most common classes are **L**, **P**, **NP**, and **EXP**. The most basic are **L**, which is comprised of problems that can be solved in logarithmic time, **P**, which is comprised of problems that can be solved in polynomial time, and **EXP**, which is comprised of problems that can be solved in exponential time. The less intuitive class is **NP**, which is comprised of problems that have no known polynomial time solution, but whose correct solutions can be verified in polynomial time. These problems are typically cast as *decision problems*, meaning that they have either a "yes" or "no" answer. Thus, the "yes" solutions to decision problems in **NP** can be verified in polynomial time.

The most important tool in studying the **NP** complexity class is the idea of *reduction*, a process in which one problem in **NP** can be transformed into another problem whose complexity class is not known, and by doing so prove that the new problem must be at least as hard as the first (and therefore in **NP**). This can also be done for the special subset of problems denoted **NP**-hard, which is comprised of problems that are at least as hard as every problem in **NP**. One of the biggest unsolved problems in computer science is the $\mathbf{P} \neq \mathbf{NP}$ problem, which aims to either prove or disprove that the **P** and **NP** complexity classes do not encompass the same set of problems. Although most computer scientists believe that $\mathbf{P} \neq \mathbf{NP}$, if polynomial-time algorithms could be developed for intractable problems like factoring or the *Traveling Salesman Problem*, they would provide revolutionary advances in computer science. But, for now, the hierarchy of complexity classes is as follows:

$$\mathbf{L} \subseteq \mathbf{P} \subseteq \mathbf{NP} \subseteq \mathbf{EXP}$$

As a side note, you may be wondering why there is so much focus on algorithms that run in polynomial time. Although extremely high-degree polynomials could be worse than small ex-

ponentials, for all intents and purposes polynomial time is the best benchmark for algorithmic efficiency. There is no theoretical basis for such a benchmark, but time has shown that polynomial time seems to be the goal that computer scientists should be striving for when designing algorithms.

3.1.3 Quantum computational complexity

Finally, the biggest question is where does quantum computing fit into all of this? One more class of problems that we didn't discuss earlier is **BPP**, or *bounded-error probabilistic polynomial time*, which describes the set of decision problems that can be solved by a probablisitic Turing machine in polynomial time with error probability bounded at 1/3. With this as a reference, the class of problems we care about in quantum computing is known as **BQP**, for *bounded-error quantum polynomial time*, which describes the set of problems that can be solved by a quantum computer in polynomial time with error probability bounded at 1/3. Not much is known about the relationship between these complexity classes and the more common ones other than that:

$$\mathbf{P} \subseteq \mathbf{BPP} \subseteq \mathbf{BQP}$$

Thus, until more developments are made in quantum computing, it remains to be proven that quantum computers are necessarily more powerful that classical computers, a result which many in the scientific community believe (and hope) to be true.

4 Chapter 4: Quantum Circuits

4.1 Quantum algorithms

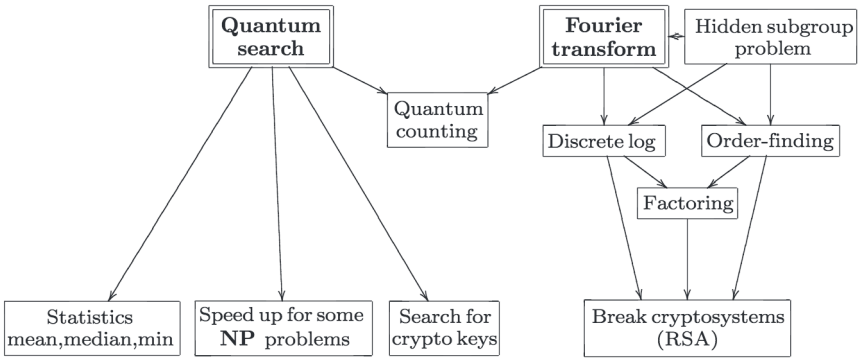

Figure 1: Some of the main algorithms and applications of quantum computing.

There are two major classes of algorithms in quantum computing. The first is based off of Shor's *quantum Fourier transform* and provides an exponential speedup over the best classical algorithms for solving the factoring and discrete log problems. Solving these problems would enable a quantum computer to break the most popular cryptosystems used today, including RSA. The second class is based off of Grover's algorithm for *quantum searching* and provides a quadratic speedup over the best classical algorithms. Quantum searching can be used to extract statistics, speed up some problems in **NP**, and search for keys in cryptosystems such as DES much more quickly.

4.2 Operations on qubits

4.2.1 Single qubit operations

In order to begin talking about quantum circuits, we must begin with the smallest building block—single qubit operations. We have seen some of the single qubit gates previously, such as the bit flip gate X and the phase flip gate Z, but in this subsection we will introduce a few more that are essential to building complex quantum circuits. The first, called the Hadamard gate and denoted H, takes the qubit from a state that is in the z-basis to a state in the x-basis. The second, called the phase gate and denoted S, rotates the state vector $\pi/2$ around the z-axis. The third, called the $\pi/8$ gate and denoted T, rotates the state vector $\pi/4$ around the z-axis. The matrix representation of these three are as follows:

$$H = \frac{1}{\sqrt{2}}\begin{pmatrix}1 & 1\\ 1 & -1\end{pmatrix} \qquad S = \begin{pmatrix}1 & 0\\ 0 & i\end{pmatrix} \qquad T = \begin{pmatrix}1 & 0\\ 0 & e^{i\pi/4}\end{pmatrix}$$

4.2.2 Controlled operations

In addition to single qubit operations, controlled operations of the form "If A is true, then do B" are extremely useful both in classical and quantum computing. The most common controlled operation is the controlled-NOT or CNOT. The CNOT gate takes as input two qubits, known as the *control qubit* and the *target qubit*. The action of the CNOT is such that if the control qubit

is set to $|1\rangle$ then the target qubit is flipped. Otherwise, the target qubit is left alone. In Dirac notation, the operation of the CNOT on the state $|c\rangle\,|t\rangle$ results in the state $|c\rangle\,|t \oplus c\rangle$. Therefore in the $|c, t\rangle$ computational basis, the matrix representation of CNOT is as follows:

$$\text{CNOT} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

In addition to the CNOT, we can form the more general controlled-U operation that only applies the single qubit unitary operator U to the target qubit if the control qubit is in the state $|1\rangle$. In Dirac notation, the operation of the controlled-U on the state $|c\rangle\,|t\rangle$ results in the state $|c\rangle\,U^c\,|t\rangle$. While the CNOT gate is simply an X gate acting on the target qubit conditional on the state of the control qubit, it is not as clear how an arbitrary controlled-U operation would be implemented. Fortunately, there is an extremely useful decomposition $U = e^{i\alpha}AXBXC$ which allows any arbitrary unitary operation U to be decomposed into only single qubit operations and the CNOT gate—things we know how to implement.

4.2.3 Toffoli gate and multiple control qubits

Because operations on qubits are represented by unitary operators U, and the reverse operation of U is simply its Hermitian conjugate $U^\dagger$, then all of the circuits that implement quantum operations must use only reversible quantum gates. The classical logic gates like AND and OR are not reversible because they destroy information, but the *Toffoli gate* (also known as the CCNOT gate) is a reversible logic gate that can be used to implement the irreversible classical ones. The Toffoli gate has two control qubits and a single target qubit, and only flips the target qubit if both of the control qubits are in the state $|1\rangle$. In Dirac notation, the operation of the Toffoli gate on the state $|c_1, c_2, t\rangle$ results in the state $|c_1, c_2, t \oplus c_1 \cdot c_2\rangle$. The matrix representation of the Toffoli gate is as follows:

$$\text{Toffoli gate} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}$$

Fortunately, the Toffoli gate can be implemented as a quantum circuit using only Hadamard, phase, controlled-NOT, and $\pi/8$ gates, which means it can be used for quantum computation. And, with the Toffoli gate in our toolkit, we can also create more complex controlled circuits that have mutiple control qubits. Such a gate, denoted $C^n(U)$, applies the single qubit unitary operation U to the target qubit if all n control qubits are set to $|1\rangle$. Using the Toffoli gate $2(n-1)$ times with $n-1$ work qubits, we can in a sense "combine" the states of all the control qubits and then perform a controlled-U on the target qubit with the combined state as the control qubit for the operation. Then, all the work qubits are reversed to their original states. This clever circuit can be seen in Figure 2.

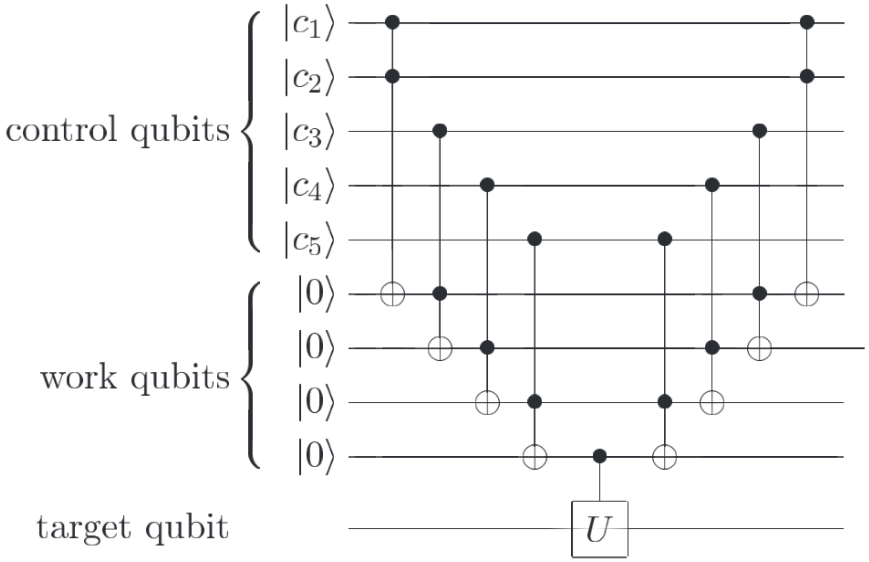

Figure 2: Quantum circuit that implements $C^n(U)$ using Toffoli and controlled-U gates.

4.2.4 Measurement

One of the most important operations in quantum circuits is the measurement operation. Relating to quantum measurement, there are two principles that are universal for all quantum circuits.

Principle of deferred measurement: Measurements can always be moved from an intermediate stage of a quantum circuit to the end of the circuit; if the measurement results are used at any stage of the circuit then the classically controlled operations can be replaced by conditional quantum operations.

Principle of implicit measurement: Without loss of generality, any unterminated quantum wires (which are not measured) at the end of a quantum circuit may be assumed to be measured.

The first principle states that if a circuit performs a measurement and then uses the result of a measurement to classically control a future operation, the measurement can be moved to the end of the circuit and the classical control operation can be replaced with a controlled quantum gate. The second principle states that we can assume all wires are measured, because measuring one wire should not have any influence on the measurement of another wire.

5 Chapter 12: Quantum Information Theory

5.1 Quantum cryptography

Although the discovery of the Quantum Fourier Transform has led to revolutionary advances in factoring and the discrete log problem, and subsequently the knowledge that a quantum computer could break the strongest classical cryptosystems, quantum computing also has potential to increase security. Through a procedure known as quantum cryptography or quantum key distribution, the principles of quantum mechanics can allow us to securely distribute private information over a public channel. In a classical key distribution protocol, the goal is to have Alice transmit a private key to Bob over a public channel without an eavesdropper Eve getting her hands on the key. Because classical information can be copied without consequences, this often poses problems for those designing such protocols. But, as we will see in quantum key distribution, the principles of quantum mechanics can be used to eliminate such problems.

5.1.1 Quantum key distribution

Quantum key distribution (QKD) is a secure protocol that allows to parties to determine a private shared key while communicating all the necessary information for the protocol over a public channel. If Alice is trying to transmit qubits to Bob, the no-cloning theorem tells us that Eve cannot make a copy of these qubits. More generally, any attempt at information gain by Eve implies the disturbance of the quantum systems she is trying to learn more about. With this in mind, the QKD protocols that have been developed use a series of checks as well as *privacy amplification* and *information reconciliation* to verify that Eve has not disturbed the qubits and that the final key bits that Alice and Bob have are in fact identical.

5.1.2 Privacy amplification

In order to effectively eliminate the partial information that Eve has about Alice and Bob's shared key, Alice and Bob need to transform their key bits in such a way that they still have identical information, but Eve's information is reduced. One possible method is to use a *universal hash function*, which takes the binary key values of Alice and Bob and returns a shorter key, determined only by the input and hash function. Alice and Bob can choose this hash function based off of how much information they believe that Eve could have extracted from their communication. The more information they posit that Eve has, the more Alice and Bob will want to shorten the resulting key. By choosing a hash function that sufficiently shortens their keys, they can reduce the probability that Eve has any knowledge of the new shorter key to a negligible value, and be sure that their shared keys are secure.

5.1.3 Information reconciliation

In addition to minimizing the amount of information that Eve can garner from Alice and Bob's communication over a public channel, Alice and Bob also want to verify that the key bits they both end up with are in fact identical. One common method of information reconciliation is the *cascade protocol*, which recursively divides the keys into blocks and compares the parities of those blocks until bit-flip errors can be honed in on and corrected. Although the exchange of parity information over the public channel allows Eve to gain more information, Alice and Bob can choose a security parameter s such that Eve's total information is negligible in comparison to the final m key bits that Alice and Bob distill from the results of their key-sharing protocol (which relates back to privacy amplification).

5.1.4 The BB84 protocol

Combining the principles of quantum mechanics which privacy amplification and information reconciliation, the BB84 QKD protocol allows for an arbitrarily secure exchange of a shared key between Alice and Bob.

1: Alice chooses $(4 + \delta)n$ random data bits.

2: Alice chooses a random $(4 + \delta)n$-bit string b, encoding each data bit as $\{|0\rangle, |1\rangle\}$ if the corresponding bit in b is 0, or as $\{|+\rangle, |-\rangle\}$ if the corresponding bit is 1.

3: Alice sends the resulting state to Bob.

4: Bob receives the $(4+\delta)n$ qubits, announces this fact, and measures each qubit in the X or Z basis at random.

5: Alice announces b.

6: Alice and Bob discard any bits where Bob measured a different basis than Alice prepared. With high probability, there are at least $2n$ bits left to keep. Otherwise, abort the protocol.

7: Alice selects a subset of n bits that will serve as a check on Eve's interference, and tells Bob which bits she selected.

8: Alice and Bob announce and compare the values of the n check bits. If more than an acceptable number disagree, they abort the protocol.

9: Alice and Bob perform information reconciliation and privacy amplification on the remaining n bits to obtain m shared key bits.