QC101 Documentation

Release 0.1

Dong Zhou

Oct 31, 2018

Contents:

Warning: This is an unfinished draft.

Quantum computing has become a buzzword nowadays. Both big tech companies and startups are competing to bring quantum computing services to the market. Despite the increasing R&D effort in industry and academia, quantum computing is mysterious because both quantum mechanics and computing theory are lesser-known topics. And most online materials are written for physics majors.

The purpose of this book is to introduce quantum computing to non-physicists. I try to focus on the principles and ideas to the best of my knowledge, and not get trapped too much in detailed calculations. If the quantum computing field is a state park, then this book is a map of a trail.

The maths are kept easy and analogies/comparisons are often used. The reader is assumed to know linear algebra and probability theory. Specifically, the following equations should look familiar

- $\mathbf{f} = m\mathbf{a}$
- $\frac{d}{dt}\mathbf{p} = R\mathbf{p}$
- $A\mathbf{x} = \lambda\mathbf{x}$
- $e^{i\theta} = \cos\theta + i\sin\theta$

What this book doesn't do:

- It doesn't teach quantum mechanics systematically.
- It doesn't describe physical implementations of quantum computers.

As for notation, I use upper case letter for matrix, bold and lower case letter for vector, and lower case letter for scalars. For simplicity, Planck constant is omitted in all formulas.

Note: I use these boxes to give heads up.

See also:

I use these boxes for optional materials.

The book should be read in linear order.

While preparing the materials, I follow several principles to keep my sanity

- Concise is better than verbose.
- Concrete is better than abstract.
- Goal-oriented is better than rambling.
- Comprehensive is better than referencing.

Please email me if you find errors, or have comments and suggestions. Feel free to create pull requests on GitHub with revision too.

CHAPTER 1

Overview

> Someone told me that each equation I included in the book would halve the sales. I therefore resolved not to have any equations at all. — Stephen Hawking

Note: The goal of this chapter is to demystify quantum computing without math. We will talk about what it is and why it is wanted.

1.1 comparison to classical computing

In general, "computing" means information processing. It can be thought of as all the possible functions (or transformations, or maps) that takes some input information and return some output information. The term "information" may sound both familiar and elusive. According to Wikipedia,

> Information is any entity or form that provides the answer to a question of some kind or resolves uncertainty.

which is closely related to information representation.

what kind of processing is allowed?

Before introducing quantum computing, we will first review classical computing.

A classical computer has the following components:

- processor
- memory
- input device: switches, keyboard, mouse, etc
- output device: light bulbs, speaker, screen, etc

This layout is known as the Von Neumann architecture.

of classical computer as a dumb but fast file clerk

Currently, the so-called quantum computers on the market are more of the nature of quantum processors, where the computation process is a quantum time evolution of the quantum bits. As far as I know, 'quantum memory' does not exist. Thus calculations need to be read out immediately.

For classical computing, increasing level of abstraction

- underlying physical processes
- logic gate
- machine code
- assembly language
- higher-level languages

Fortunately, as long as we do not worry about hardware implementations (superconducting circuits, quantum optics, nuclear magnetic resonance, etc), not much physics background is needed to get some sense of quantum computing. The same thing is true for classical computing. In fact, most computer scientists and programmers are not familiar with transistors, the basic building block of classical bit.

Fig. 1.1: Bloch sphere representation of single qubit states. Each point on the unit sphere corresponds to a valid single qubit state.

Table 1.1: Comparison of classical and quantum computer

	classical computer	quantum computer
bit	bit • two voltage states 0 and 1 • computation unit • storage unit	qubit • two quantum basis states • computation unit • storage unit ??
gate	• 1-bit: NOT • 2-bit: AND, OR, XOR, NAND, etc • 3-bit: Toffoli	• 1-qubit: X, Y, Z, etc • 2-qubit: CNOT, CPHASE, etc • 3-qubit: Toffoli
math foundations	Boolean algebra	Lie algebra, Lie group

Nowadays, the quantum computing industry all adopted the cloud based quantum computing. Thus a quantum programmer designs some kind of machine code or assembly-like language, uploads to the cloud. Due to the peculiar nature of quantum mechanics, initializing the quantum bits and reading out their states are hard. And I have dedicated chapters for them later.

For quantum computing, one still needs to work on lower levels. The optimal protocols, or even the best hardware implementations are not settled yet.

> Computer science… differs from physics in that it is not actually a science. It does not study natural objects. Neither is it, as you might think, mathematics; although it does use mathematical reasoning pretty extensively. Rather, computer science is like engineering; it is all about getting something to do something, rather than just dealing with abstractions, as in the pre-Smith geology. — Richard Feynman

1.2 motivations of quantum computing

killer applications

1.2.1 simulate quantum systems with quantum systems

In the early 80s, people started to think about simulating quantum systems using quantum systems, for example

- Yuri Manin, Computable and uncomputable (in Russian), Sovetskoye Radio (1980)
- Richard Feynman, Simulating Physics with Computers, IJTP 21, 467 (1982)

This is a very natural idea because simulating quantum systems on a classical computer is expensive. There are two aspects to this expense:

- the amount of **space** to store the quantum state, either in memory or on disk
- the effort to calculate the time evolution, i.e., the **time** complexity

In the most straightforward implementation, the number of bits needed to describe a quantum system on a classical computer grows exponentially with the number of atoms. For example, if 10 states are needed to describe an atom, then a 100-atom system would require a vector of size 10^{100} components. Note that the number of atoms in the whole universe is estimated to be about 10^{80}.

Time evolution of quantum systems is computed using matrix manipulations. On classical computers, straightforward implementation of matrix multiplication has computational complexity $O(n^3)$, where n is the matrix size. It is possible to make it slightly more efficient but not much: definitely not $O(n^2)$. Going back to our 100-atom system example, calculating its dynamics would then have time complexity $O(10^{300})$ in the worst case.

Thus in practice, such straightforward implementation on classical computers, the so-called full configuration interaction approach, can only be used to study very small molecules. Many approximated methods have been developed to deal with medium sized molecules with less computational burden. For large molecules such as proteins which can easily have more than 100,000 atoms, it is still extremely challenging if not impossible to simulate them.

On the other hand, if we have a simulator which is itself quantum mechanical, Nature will take care of the time evolution calculation: no more matrix multiplications. All we need to do is to set up the Hamiltonian of the system (i.e., describe how atoms interact), and then wait for the desired end time of the simulation. For example, if we use the quantum system of interest to 'simulate' itself and we are interested in the result at 1 second, then we just wait for 1 second and look at the system.

Nowadays there is a vague notion of quantum supremacy at 50 qubits. It basically says that a quantum computer with 50 qubits has more computational power than any classical computer. If we just count the size of the state space, 50 qubit amounts to a state vector with size $2^{50} \simeq 10^6 G$. Matrix multiplications on such vectors is indeed daunting. There are still controversies on whether this supremacy happens at 50 qubits. But it definitely gives strong incentives for the tech companies to make 50-qubit devices.

1.2.2 general-purpose quantum computer

It doesn't take long for the idea of universal quantum computer to appear, for example

- David Deutsch, Quantum theory, the Church-Turing principle and the universal quantum computer, Proc. R. Soc. London A 400, 97 (1985)

The goal here is to build a universal machine that can do all possible calculations, instead of building a specialized machine for each computational task. In terms of quantum simulators, it means that one would have a device that can simulate all possible quantum systems at least with some approximations. This line of thought is a direct analogy of the classical Church-Turing thesis.

CHAPTER 2

Quantum mechanics in a Nutshell

Note: The goal of this chapter is to introduce quantum mechanics. There are two aspects to it: the experiment and the theory. Specifically, I will

- describe the phenomenon of atoms/molecules absorb light.
- compare the math of quantum mechanics to the math of stochastic processes.

The precursor of quantum mechanics, known as classical mechanics, is a deterministic theory. Time evolution is described by differential equations such as Newton's second law

$$\mathbf{F} = m\mathbf{a} \equiv m\frac{d^2}{dt^2}\mathbf{x}$$

where $\mathbf{F}$ is force, m is mass, $\mathbf{a}$ is acceleration, and $\mathbf{x}$ is position of an object of interest, say a projectile. Given initial conditions of position and velocity, one can calculate the object's position at any future time (or even past time).

Although classical mechanics is a successful theory to describe 'big' (the physicists' jargon is macroscopic) objects, its prediction often fails on 'small' (microscopic) objects such as molecules and atoms. With the effort of many physicists, quantum mechanics was established in early 20'th century. It is intrinsically a probabilistic theory and is considered bizarre even by the experts. Yet it is a successful theory that seems to have no reported failures on experimental observations.

Overall, its difference from classical mechanics can be summarized as follows

1. State space could be finite.
2. Superposition of different states is allowed.
3. Measurement collapses superposition.
4. Dynamics (i.e., transitions between possible states) is probabilistic.

To be concrete, I will use atom as the prototypical quantum object.

2.1 absorption spectrum

To illustrate the first difference, let's revisit Newton's prism experiment originally recorded in the 17'th century, where a beam of white sunlight passes through a prism and becomes a rainbow of colors, see Fig. 2.1. From classical optics, we know that light is a wave phenomenon and its color is determined by the wave's wavelength (or equivalently, its frequency, the speed of light is the same for light of all colors and the light speed is equal to the product of its wavelength and frequency). For example, red light has frequency about 700 nm (460THz), and blue light has frequency about 470 nm (640THz).

Fig. 2.1: Newton's prism experiment.

This simple refraction experiment contains the basic idea of spectroscopy. The plot of light intensity versus light wavelength (or frequency) is called spectrum. An example of the sunlight spectrum is shown in Fig. 2.2.

Fig. 2.2: Solar spectrum. The absorption lines are denoted by arrows.

Interestingly, the overall shape of this spectrum reveals the temperature of the Sun. There is a formula relating the light intensity to the light frequency and the temperature of the object, proposed by Max Plank in 1900 and known as Plank's law. It can also be used to estimate the temperature of a light bulb, or human body. In fact, the derivation of Plank's law requires quantum mechanics, and it is one of the pioneering work to demonstrate the validity of quantum mechanics. Classical physics gives a non-sensible result in this case, commonly known as the ultraviolet catastrophe of black-body radiation (here black-body simply means a hot object). It predicts that the light intensity approaches infinite at high frequency, for any temperature of the object.

I will not explain Plank's law here. Instead I would like to draw your attention to some ugly dips in the sunlight spectrum, as denoted by arrows in Fig. 2.2. These dips are known as Fraunhofer lines. A different view of them is shown in Fig. 2.3. What happens here is that some of the light is absorbed by various atoms and molecules in the Sun's atmosphere (actually for stars, it's called photosphere), causing deviations to Plank's formula. Interestingly, these absorption lines happens at special frequencies. In fact, they are characteristic of the material. For example, the lines for iron and the lines of oxygen molecules look nothing like each other. Thus by looking at these lines, we can deduce the composition of the Sun and even their abundance.

Fig. 2.3: Solar spectrum with Fraunhofer lines.

But what does it mean that iron (or any other atoms or molecules) has multiple absorption lines? One hypothesis could be that iron atom has multiple states and each state corresponds to one particular line. This is actually not too far from the full story. The more complete explanation is as follows

1. Each dark line means the absence of light with specific wavelength (or frequency);
2. That specific light carries a specific amount of energy;
3. The iron atom has two states whose energy difference equals to that specific light's energy, and it is able to absorb the light and jump from the low energy state to the high energy state.

The first point merely states experimental observation. The latter two points are both alien to the 19'th century physicists. Nowadays they are known to the general public. For example, sunscreens contain special molecules that absorb ultraviolet (UV) (high frequency/short wavelength) light. If the UV light is not blocked by sunscreen, molecules in our skin may absorb them and become something else.

One confusing subtlety of light can be summarized in this question: does a dim UV light carry more energy or less energy than a bright red (red is long wavelength/low frequency) light? One way to think about this is to imagine light as a flux of flying balls (the balls are known as photons). Two factors contributes to the light source's intensity, or brightness, or illuminance or radiance as coined by the pros:

- the number of balls hit a unit area in a unit time, i.e., the flux
- the energy carried by each ball: a purple ball has more energy than a red ball

Thus it is possible that a dim UV light (small flux) bombast one's skin (of a fixed area and in a fixed time span) with less energy than a bright red light. However, red light is much less likely to be absorbed by the skin thus it doesn't cause as much harm as a dimmer UV light even though it carries more energy.

Now let me illustrate why it's weird for an atom (or molecule) to be in a finite number of states. By that time, physicists already knew that atoms are made of nucleus with positive charge and electrons with negative charge. If one makes the analogy that nucleus is the Sun, an electron is a planet, and the attractive force between positive and negative charges plays the role of gravitational attraction, it is unclear why only a finite number of stable orbits exist. Note that in the Sun-and-planet case, there are infinite stable orbits, although only a few of them are occupied. If we launch a new satellite, we pre-calculate the stable orbit so that it doesn't hit into other things. And if the satellite's speed increase a little bit, it will be in a slightly different orbit.

Again the full explanation require quantum mechanics.

See also:

Spectroscopy is likely the most important experimental technique in modern science.

I don't known any convincing explanation about why microscopic objects often assume discrete states. For example, these atomic states are not energy minimums of some cost function. However, the evidence of them are quite strong, thus a successful theory has to incorporate them. As a result, quantum theory is bizarre.

> All models are wrong; some are useful. — George E. P. Box

2.2 analogy of coin tossing

Quantum mechanics is intrinsically a probabilistic theory, that is, if one repeats an experimental procedure in an idealized situation (no human mistake, no machine error, no noise) with a measurement at the end, the measurement results could still differ from different trials. At a superficial level, it is similar to probability theory where some information is not available.

The simplest quantum system has two states and the classical analogy is a coin with two sides. A coin toss has two outcomes: head and tail, and is represented by Bernoulli distribution:

$$\mathbf{p} = \begin{bmatrix} p_H \\ p_T \end{bmatrix}, \text{ with } p_H + p_T = 1.$$

Each coin toss has two outcomes and their probabilities can be described by a two-component vector $\mathbf{p}$. For example, fair coins have

$$\mathbf{p} = \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix}$$

Given such probability vectors, we can easily describe the tossing of the same coin many times, or more generally, the tossing of many coins with different biases. Take two coins for example, the outcome probability is given by the tensor product of the individual probability vectors, i.e.,

$$\mathbf{p} = \mathbf{p}_1 \otimes \mathbf{p}_2 \equiv \begin{bmatrix} p_{1H}\mathbf{p}_2 \\ p_{1T}\mathbf{p}_2 \end{bmatrix} = \begin{bmatrix} p_{1H}p_{2H} \\ p_{1H}p_{2T} \\ p_{1T}p_{2H} \\ p_{1T}p_{2T} \end{bmatrix}$$

With n different coins, there are 2^n possible states. However, the probabilities can be calculated from $2n$ numbers. This is the product rule of probability since we assume the coin tosses are independent events.

To make the situation more complicated, there are two ways to go:

1. make the coin tosses dependent events: maybe they hit each other as they are tossed (instead of being tossed one by one)
2. make the probability distribution time-dependent: maybe they are being deformed as they are tossed

The first complication breaks the product rule and we have to assign one probability to each outcome. In the two-coin example,

$$\mathbf{p} = \begin{bmatrix} p_{1H,2H} \\ p_{1H,2T} \\ p_{1T,2H} \\ p_{1T,2T} \end{bmatrix}$$

and no decomposition is possible anymore.

The second complication adds dynamics to the probabilities distributions. The simplest description one can give may be the Kolmogorov equation:

$\frac{d}{dt}\mathbf{p}(t) = R\mathbf{p}(t)$

where R is a transition rate matrix.

For simplicity, let's assume that R is time-independent. Then we have a formal solution

$\mathbf{p}(t) = e^{Rt}\mathbf{p}(0)$.

2.3 'quantum' coin tossing

If we magically force a coin to obey quantum mechanics, some of its behavior would appear identical to the classical coin. For example, if we 'toss' the quantum coin, there will still be only two outcomes, head or tail, just like the classical coin. This 'quantum' coin tossing is called von Neumann measurement in quantum mechanics, which is the equivalent of drawing one sample from a probability distribution. If we toss many quantum coins with identical states, the outcomes of head or tail also follow Bernoulli distribution.

Before we proceed to the difference between quantum coin and classical coin, let's first prepare ourselves with the notation for describe quantum states, i.e., the Dirac notation.

2.3.1 Dirac notation

The state of the quantum coin is described by a 2D complex vector

$$|\psi\rangle = c_H |H\rangle + c_T |T\rangle \tag{2.1}$$

where $c_H, c_T \in \mathbb{C}$ and they are called probability amplitudes. Here $|H\rangle$ and $|T\rangle$ are called basis states, and $|\psi\rangle$ is known as the wave function.

If we tossing the quantum coin many times, the probabilities to get head or tail are given by $\|c_H\|^2$ and $\|c_T\|^2$. And we have the normalization $\|c_H\|^2 + \|c_T\|^2 = 1$.

Here the half bracket notation is called the Dirac notation. In this example, they correspond to 2D vectors:

$$|H\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, |T\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, |\psi\rangle \equiv \mathbf{c} = \begin{bmatrix} c_H \\ c_T \end{bmatrix}.$$

These states with right bracket are called ket(s). Paul Dirac also defined the complex conjugate of these vectors

$$\langle\psi| \equiv \mathbf{c}^\dagger = \begin{bmatrix} c_H^* & c_T^* \end{bmatrix}.$$

Such states with left bracket are called bra(s). And overall Dirac notation is also called bra-ket notation. With this notation, probability normalization can be written succinctly as

$$\langle\psi|\psi\rangle = 1$$

One could argue that it is not really more convenient than $\mathbf{c}^\dagger\mathbf{c} = 1$, which I agree. I think its power is slightly more evident when there are uncountable infinite possible states. In the end, it is just a notation that physicists are used to.

For multiple quantum coins, their state vector is also tensor product of the individual ones. Take the two coin case for example,

$$|\psi\rangle = |\psi_1\rangle \otimes |\psi_2\rangle = \begin{bmatrix} c_{1H}\mathbf{c}_2 \\ c_{1T}\mathbf{c}_2 \end{bmatrix} = \begin{bmatrix} c_{1H}c_{2H} \\ c_{1H}c_{2T} \\ c_{1T}c_{2H} \\ c_{1T}c_{2T} \end{bmatrix}$$

Again, if the two quantum coins are somehow coupled together, such decomposition is not possible,

$$|\psi\rangle = \begin{bmatrix} c_{1H,1T} \\ c_{1H,2T} \\ c_{1T,2H} \\ c_{1T,2T} \end{bmatrix}$$

If we go back to the absorption spectrum example, $|\psi_1\rangle \otimes |\psi_2\rangle$ represent two atoms, each of which has two internal states. Thus at most two absorption lines can be observed. On the other hand, the state of coupled coins could also model an atom with four internal states. In that case at most $C(4, 2) = 6$ absorption lines can be observed.

Now if we identify $p_i = c_i^* c_i$, then it appears the quantum case maps to the classical case exactly. And it seems unnecessary to use complex numbers instead of real numbers. This superficial similarity will be examined more closely in the following sections.

2.3.2 superposition principle and quantum measurement

Before going to the time evolution of quantum mechanics, let me first reveal the differences between classical probability theory and quantum mechanics in static situations.

The analogy between probability theory and quantum mechanics is summarized in Table 2.1. Both the superposition principle and von Neumann measurement differ from their probabilistic counterpart at fundamental level.

Table 2.1: Analogy between probability theory and quantum mechanics.

probability theory	quantum mechanics
multiple possible states	superposition principle
drawing sample from a distribution	von Neumann measurement

The peculiarities of the superposition principle are

1. Different states can be linearly superimposed.
2. The coefficients of such superposition are complex numbers.

In probability theory, statistics, or ensemble average, is calculated as

$$\langle f\rangle \equiv \sum_i p_i f_i \tag{2.2}$$

where f_i is some numerical value assigned to the state. For example, we can assign f to 1 for head, and -1 for tail.

In the same spirit, a quantum ensemble average could take the same form

$$\langle\psi| f |\psi\rangle = \sum_i p_i f_i = \sum_i c_i^* f_i c_i \tag{2.3}$$

The acute reader may find Eq. (2.3) puzzling. In the coin example, $|\psi\rangle$ is a 2-by-1 vector, thus it appears that f should be a 2-by-2 matrix. This is indeed true. Since I use capital letters to denote matrix, the quantum ensemble average should take the form of

$$\langle\psi| F |\psi\rangle = \sum_{ij} c_i^* F_{ij} c_j \tag{2.4}$$

where F_{ii} may correspond to f_i. But what is the meaning of the off diagonal terms F_{ij}? Actually they don't have classical correspondence in probability theory. They are put in there to model wave interference, which takes care of the discrepancy to the predictions of classical mechanics. It is an intrinsic nature of quantum mechanics (note that quantum mechanics is also known as wave mechanics and the state vector is known as wave function).

These off-diagonal terms together with the c_i being complex numbers provide more degrees of freedom to explain experimental observations that deviate from classical expectations in Eq. (2.2). Note that I didn't explain why the quantum ensemble takes the form of Eq. (2.4). In fact, I don't know any high-level plausible ideas why it should be, except that the formalism works.

The value of the off-diagonal terms is not completely arbitrary. There is controversy of whether complex numbers are physical (i.e., whether they can be measured from an experiment) or they are only mental constructions to simplify notations (I tend to take the latter view). For the sake of argument, let's assume that measurement can only yield real numbers. This requirement puts on extra constraint on the form of F since $\langle\psi| F |\psi\rangle$ has to be real for any vector $|\psi\rangle$. The qualified class is called Hermitian matrix.

As a summary, the main difference introduced in the superposition principle is the off-diagonal terms in the observation, which models interference.

Measurement in quantum mechanics is radically different: **after a quantum coin toss, the coin's probability amplitude (thus probability distribution) changes**.

Suppose the coin is in state of Eq. (2.1) before the measurement. After the measurement, its state changes to either $|H\rangle$ or $|T\rangle$ with the corresponding probability. Any subsequent measurement gives deterministic result. In other words, tossing the same quantum coin multiple times doesn't work because the quantum coin is not the same after a toss (unless it's in one of the basis states to start with). This phenomenon is known as wave function collapse. In our analogy, we could also call it probability distribution collapse.

On the other hand, if we toss an ensemble of quantum coins with identical state (i.e., probability amplitude or probability distribution), we will observe the same head and tail counts as in probability theory. In general, if we want to observe some physical quantity as in Eq. (2.4), an ensemble of quantum states is needed. This ensemble can be generated either by repeatedly preparing and measuring the same quantum object, or preparing the same state for many quantum objects and measuring them.

2.3.3 dynamics

There are two

- von Neumann measurement
- time evolution

the mathematics to describe the coin will change quite dramatically.

$$i\frac{d}{dt}\mathbf{c}(t) = H\mathbf{c}(t)$$

Similar to the classical case, in the case when H is time independent, Schrodinger equation has explicit solution

$$\mathbf{c}(t) = e^{-iHt}\mathbf{c}(0)$$

In general, the solution can be very complicated when the Hamiltonian is time dependent.

$$\mathbf{c}(t) = U(t)\mathbf{c}(0)$$

where $U(t)$ absorbs all the complications in it and is simply called the time evolution.

Probability conservation $\mathbf{c}^\dagger\mathbf{c} = 1$ puts on requirement on the time evolution matrix

$$U^\dagger U = 1$$

This type of matrices are called unitary matrix.

The simplest time-dependent Hamiltonian may be one that is piecewise constant. Suppose during time interval Δt_i, the Hamiltonian is H_i, then we have

$$U(t) = \Pi_i e^{-iH_i\Delta t_i} = e^{-iH_1\Delta t_1}e^{-iH_2\Delta t_2}\dots$$

This is a common scheme to build time evolution out of a handful Hamiltonians. Here the control variables are the ordering of the available Hamiltonians and their time intervals.

A side-by-side comparison of quantum mechanics and stochastic processes is shown in Table 2.2.

Table 2.2: Comparison of stochastic process and quantum mechanics. Here R and H are assumed to be time-independent to allow simple solutions.

	stochastic process	quantum mechanics
state vector	probabilities $\mathbf{p}$	probability amplitudes $\mathbf{c}$
normailization	$\|\mathbf{p}\|_1 = 1$	$\mathbf{c}^\dagger\mathbf{c} = 1$
dynamics	**Kolmogorov equation** $\frac{d}{dt}\mathbf{p}(t) = R\mathbf{p}(t)$	**Schrödinger equation** $i\frac{d}{dt}\mathbf{c}(t) = H\mathbf{c}(t)$
solution	$\mathbf{p}(t) = e^{Rt}\mathbf{p}(0)$	$\mathbf{c}(t) = e^{-iHt}\mathbf{c}(0)$

See also:

Here I only presented the equations and notations used in quantum mechanics, but didn't explain how they come into being. The

CHAPTER 3

Quantum bit (qubit)

Note: The goal of this chapter includes

- introducing 3 different ways to represent a qubit state
  - wave function $|\psi\rangle$ as a 2D complex vector
  - Bloch vector $[x, y, z]^T$ as a 3D real vector
  - density matrix ρ as a 2×2 matrix with complex entries
- calculting time evolution (i.e., quantum gate) using time-independent Hamiltonian

A qubit is a quantum two-state system. It is an abstraction with all hardware implementation details hidden, just like its classical counterpart of bit. To be concrete, one can think of the two qubit states to be two states of an atom, or two sides of a coin.

The single-qubit wave function has the parametrization

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

where α, β are complex numbers and the probability interpretation requires $\|\alpha\|^2 + \|\beta\|^2 = 1$. Two complex numbers with one normalization constraint gives rise to three degrees of freedom.

There is one more dispensable degree of freedom: $|\psi\rangle$ and $e^{i\eta}\,|\psi\rangle$ are equivalent for any real-valued η. This is because these two states have the same measurement probability distribution ($p_0 = \|\alpha\|^2$ and $p_1 = \|\beta\|^2$) and ensemble averages in Eq. (2.4), as

$$\langle\psi|\, e^{-i\eta} F e^{i\eta}\,|\psi\rangle = \langle\psi|\, F\,|\psi\rangle$$

Without loss of generality, we can let α be real and positive, and parametrize the wave function with only two real values $\theta \in [0, \pi]$, and $\phi \in [0, 2\pi)$:

$$|\psi\rangle = \cos\frac{\theta}{2}\,|0\rangle + e^{i\phi}\sin\frac{\theta}{2}\,|1\rangle \tag{3.1}$$

It may look strange to use $\theta/2$ instead of θ. This choice will be convenient later.

3.1 a geometric picture and the density matrix

There is a geometric picture called Bloch sphere representation for quantum two-state systems, which may be first formulated in studies of nuclear magnetic resonance. It is a must-know for hardware implementations, but may be less useful for quantum algorithm designs.

There are only three ensemble averages for a single-qubit state:

$$x = \langle\psi|X|\psi\rangle \tag{3.2}$$
$$y = \langle\psi|Y|\psi\rangle \tag{3.2}$$
$$z = \langle\psi|Z|\psi\rangle \tag{3.2}$$

where X, Y, Z are Pauli matrices:

$$X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \tag{3.2}$$
$$Y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix} \tag{3.3}$$
$$Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \tag{3.4}$$

This three dimensional real vector is called Block vector. Note also that the Pauli matrices are Hermitian and traceless.

Plugging the state Eq. (3.1), we have

$$x = \begin{bmatrix} \cos\frac{\theta}{2} & \sin\frac{\theta}{2}e^{-i\phi} \end{bmatrix} \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} \cos\frac{\theta}{2} \\ \sin\frac{\theta}{2}e^{i\phi} \end{bmatrix} = \sin\theta\cos\phi \tag{3.5}$$
$$y = \sin\theta\sin\phi \tag{3.6}$$
$$z = \cos\theta \tag{3.7}$$

This is the same parametrization of a vector on the sphere of radius 1 (thanks to our judicious choice of $\theta/2$). The two parameters θ and ϕ in Eq. (3.1) can be interpreted as polar and azimuthal angles. And one can visualize qubit state as a 3D vector, as in Fig. 3.1.

Fig. 3.1: Bloch sphere representation of single qubit states. Pure states are on the unit sphere, whereas mixed states are inside the unit sphere.

Another way to introduce the Bloch vector is via the so-called density matrix. Instead of writing the quantum state as a complex vector $|\psi\rangle$, we could write out a matrix

$$\rho = |\psi\rangle\langle\psi| \tag{3.8}$$

If we know $|\psi\rangle$, it's trivial to get ρ. However, if know ρ, we cannot get a unique $|\psi\rangle$ because $e^{i\eta}|\psi\rangle$ with any real-valued η is also valid. Again, this is not a problem since the overall phase factor does not affect any measurement result.

The diagonal entries of ρ denote the probability of the states. And the off-diagonal entries denote the quantum interference that has no classical counterpart. We can easily see that the following identity holds

$$tr(\rho) \equiv \sum_i \rho_i = 1 \tag{3.9}$$

where tr is called the trace operation, which sums the diagonal entries.

For a single qubit, the density matrix can be decomposed as

$$\rho = \frac{1}{2}(I + xX + yY + zZ)$$

and the Bloch vector components in Eq. (3.2) can be expressed as

$$\begin{aligned} x &= tr(\rho X) \\ y &= tr(\rho Y) \\ z &= tr(\rho Z) \end{aligned} \tag{3.10}$$

It will be helpful to check the Bloch vector components in Eq. (3.2) and Eq. (3.10) are equal. In fact, it is possible to show that

$$tr(|\psi\rangle \langle\psi| F) = \langle\psi| F |\psi\rangle$$

for arbitrary matrix F.

This can be understand in terms of a linear space of matrices. From Eq. (3.8) the density matrix ρ is a Hermitian matrix, i.e. $\rho^\dagger = \rho$. And any 2×2 Hermitian matrix can be expanded with the basis set $\{I, X, Y, Z\}$. The inner product for this linear space is the trace operation tr, i.e., two matrices A, B are orthogonal if

$$tr(AB) = 0$$

And the Bloch vector components are coefficients with respect to this basis set. Note that the coefficient of the identity matrix is fixed by Eq. (3.9).

Overall, the density matrix is a more succinct representation of the quantum state. It automatically gets rid of the global phase factor and connects to physical observables directly.

See also:

Eq. (3.8) is not the most general form of a quantum state. For example, if a colleage asks a quantum state from us repeated, and we accidentally give him the wrong state half of the time, then the average state he gets can be described as

$$\rho = \sum_i p_i |\psi_i\rangle \langle\psi_i|$$

These states are called mixed states and the ones described by Eq. (3.8) are called pure states. Besides being result of mistakes, mixed states emerges in physical implementations whenever imprefection occurs. In other words, pure states are idealizations that almost never occur in real life.

For mixed states,

$$tr(\rho^2) \neq tr(\rho)$$

For a single qubit, one can show that $tr(\rho^2) = 1$ is equivalent to the Bloch vector has unit length. Thus mixed states are represented by vectors inside the unit ball, as in Fig. 3.1.

3.2 time evolution

Recall from the previous chapter that quantum time evolution has the form of

$$|\psi(t)\rangle = e^{-iHt} |\psi(0)\rangle$$

where $i = \sqrt{-1}$, H is the Hamiltonian matrix and it determines how the state vector (probability distribution) changes. For simplicity, we have assumed that H is time-independent.

For a single qubit, the most general form of H is

$$H = aX + bY + cZ + dI \tag{3.10}$$

where I is 2-by-2 identity matrix, X, Y, Z are Pauli matrices, and a, b, c, d are real-valued coefficients. In an experimental setup, one may have control over these coefficients. Thus by tuning these coefficients as well as time, one controls the time evolution $U(t) = \exp(-iHt)$ of the qubit.

Usually one drops the I term because it only gives rise to an overall phase factor $\exp(-idt)$, which has no consequence for measurement.

It is illuminating to see the action of the unitary time evolution in the Bloch sphere representation, i.e.,

$$\begin{aligned} x(t) &= \langle\psi(t)| X |\psi(t)\rangle \\ y(t) &= \langle\psi(t)| Y |\psi(t)\rangle \\ z(t) &= \langle\psi(t)| Z |\psi(t)\rangle \end{aligned} \tag{3.11}$$

To make the math simple, let's consider Hamiltonians with only one Pauli matrix.

3.2.1 Z rotation

The simplest case is to have only the Z term in the Hamiltonian, since it is already diagonal.

$$\begin{aligned} U(t) =& e^{-icZt} \\ =& \begin{bmatrix} e^{-ict} & 0 \\ 0 & e^{ict} \end{bmatrix} \\ =& e^{-\frac{i\phi_t}{2}} \begin{bmatrix} 1 & 0 \\ 0 & e^{i\phi_t} \end{bmatrix} \end{aligned} \tag{3.14}$$

where $\phi_t = 2ct$.

With this time evolution, the state vector becomes

$$|\psi(t)\rangle = U(t) |\psi(t=0)\rangle = \begin{bmatrix} \cos\frac{\theta}{2} \\ \sin\frac{\theta}{2} e^{i(\phi+\phi_t)} \end{bmatrix}$$

Here I have omitted the overall phase factor of $\exp(-i\phi_t/2)$. In terms of the Bloch vector,

$$\begin{aligned} x(t) &= \sin\theta\cos(\phi + \phi_t) \\ y(t) &= \sin\theta\sin(\phi + \phi_t) \\ z(t) &= \cos\theta \end{aligned} \tag{3.14}$$

Thus the Bloch vector rotates about the z-axis with angular velocity $2c$.

3.2.2 X and Y rotation

The time evolution matrix $U(t) = \exp(-iHt)$ for the other two cases is more difficult to calculate since H is not diagonal. We will need to diagonalize the Hamiltonian first. Take X for example, it has the decomposition

$$\begin{aligned} X \equiv& \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \\ =& V Z V^\dagger \\ =& \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \end{aligned} \tag{3.17}$$

One can think of this decomposition as singular-value decomposition. Since X and all Hamiltonian matrices are Hermitian, only one orthonormal basis is needed.

Then the time evolution is given by

$$\begin{aligned} U(t) =& e^{-iaXt} \\ =& V e^{iaZt} V^\dagger \\ =& \begin{bmatrix} \cos\frac{\phi_t}{2} & -i\sin\frac{\phi_t}{2} \\ -i\sin\frac{\phi_t}{2} & \cos\frac{\phi_t}{2} \end{bmatrix} \end{aligned} \tag{3.20}$$

The expression of the Bloch vector with this time evolution matrix is unfortunately quite a mess, although the underlying interpretation is simple. The answer is given by

$$\begin{aligned} x(t) &= \sin\theta\cos\phi \\ y(t) &= \sin\theta\sin\phi\cos\phi_t - \cos\theta\sin\phi_t \\ z(t) &= \sin\theta\sin\phi\sin\phi_t + \cos\theta\cos\phi_t \end{aligned} \tag{3.23}$$

In fact, it is more illuminating to avoid the calculation of the Bloch vector and calculate the time evolution matrix on the Bloch vector instead.

Using eq. (3.10), we have

$$\begin{aligned} x(t) =& tr\left(\rho(t)X\right) \\ =& tr\left(U(t)\rho_0 U^\dagger X\right) \\ =& \frac{1}{2} tr\left(x_0 UXU^\dagger X + y_0 UYU^\dagger X + z_0 UZU^\dagger X\right) \end{aligned} \tag{3.26}$$

Here the subscript 0 denotes $t = 0$. And the other two Bloch vector components can be calculated similarly.

Further simplification comes from the realization that

$$U(t) = \cos\frac{\phi_t}{2} I - i\sin\frac{\phi_t}{2} X$$

Putting various terms together, we have

$$\begin{bmatrix} x(t) \\ y(t) \\ z(t) \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi_t & -\sin\phi_t \\ 0 & \sin\phi_t & \cos\phi_t \end{bmatrix} \begin{bmatrix} x_0 \\ y_0 \\ z_0 \end{bmatrix}$$

It is obvious that the quantum dynamics with only the Pauli X term gives rise to 3D rotation on the Bloch vector along the x-axis.

I will leave the calculation of Y-rotation as an exercise for you.

3.2.3 arbitrary rotation

As you may have guessed, the general Hamiltonian in Eq. (3.10) causes the Bloch vector to rotate about the axis of $[a, b, c]$ with the angular velocity of $2t\sqrt{a^2 + b^2 + c^2}$.

3.3 one-qubit quantum gate

In the chapter, we have seen a few important single-qubit gates.

The Z rotation essentially gives rise to the so-called phase shift gate

$$U_\phi = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\phi} \end{bmatrix}$$

When we diagonalize the Pauli X matrix, the auxiliary V matrix is a very useful gate called Hadamard gate. Conventionally it is denoted as

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

Its common usage is to turn $|0\rangle$ to $\frac{1}{2}(|0\rangle + |1\rangle)$.

The Pauli X matrix is actually a gate since it is unitary (actually all Pauli matrices are both Hermitian and unitary, thus could be gates)

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

It is the equivalent of NOT gate in classical computing since it swaps $|0\rangle$ and $|1\rangle$ states.

Note that NOT gate is the only logic gate for single bit in classical computing. Quantum computing contains infinitely more gates because its state space is continuous.

We will see these gates over and over again in later chapters.

CHAPTER 4

Two qubits and quantum entanglement

A two-qubit state can be parametrized as

$$|\psi\rangle = \alpha_0 |00\rangle + \alpha_1 |01\rangle + \alpha_2 |10\rangle + \alpha_3 |11\rangle \tag{4.1}$$

where α_i are complex numbers. The probabilistic interpretation requires $\sum_i \|\alpha_i\|^2 = 1$. And again we can get rid of the overall phase factor by requiring α_0 to be real. Thus there are six independent real parameters for two-qubit pure states.

Often times, the binary state labels are converted to decimal labels to simplify the notation

$$|\psi\rangle = \alpha_0 |0\rangle + \alpha_1 |1\rangle + \alpha_2 |2\rangle + \alpha_3 |3\rangle$$

This is helpful when we deal with many qubits.

4.1 two-qubit gates

It's possible to make two-qubit gates out of single-qubit gates. For example,

$$X \otimes X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \otimes \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \tag{4.2}$$

$$= \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \tag{4.3}$$

where X is the Pauli X gate. However, such gates are less interesting than the ones that cannot be expressed as tensor products, because they don't bring the two qubits into interaction.

As an example of truly two-qubit gates, let's generalize the XOR gate in classical computing to the quantum case. The truth table of XOR gate is

Inputs		Output
A	B	A XOR B
0	0	0
0	1	1
1	0	1
1	1	0

In classical computing, three bits are involved in XOR gate: two input bits and one output bit. In quantum computing, all (single-qubit and multiple-qubit) gates are unitary matrices that map the input states to the output states. In other words, the input and output qubits are the same qubits.

Thus to extend XOR gate to the quantum case, we need to expand the output a two-bit string. The easiest way is to overwrite the output to qubit B and let qubit A remain its input state. Writing out explicitly, we have

$$|00\rangle \rightarrow |00\rangle$$
$$|01\rangle \rightarrow |01\rangle$$
$$|10\rangle \rightarrow |11\rangle$$
$$|11\rangle \rightarrow |10\rangle$$

If we choose the representation

$$|00\rangle = \begin{bmatrix}1\\0\\0\\0\end{bmatrix}, |01\rangle = \begin{bmatrix}0\\1\\0\\0\end{bmatrix}, |10\rangle = \begin{bmatrix}0\\0\\1\\0\end{bmatrix}, |11\rangle = \begin{bmatrix}0\\0\\0\\1\end{bmatrix},$$

then the quantum equivalent of the classical XOR gate takes the form

$$CNOT = \begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&0&1\\0&0&1&0\end{bmatrix} \tag{4.4}$$
$$= |0\rangle\langle 0| \otimes I + |1\rangle\langle 1| \otimes X \tag{4.4}$$

In fact, it is an important 2-qubit quantum gate called Controlled-NOT or CNOT gate, and it will be used many times later in this book. Its name comes from the following interpretation: if the first qubit is in state $|0\rangle$, don't do anything to the second qubit; if the first qubit is in state $|1\rangle$, flip the second qubit state.

Here we pick XOR gate as an example because it is a reversible gate. Other classical two-bit gates such as AND gate and OR gate are irreversible, and their extension to the quantum region will involve three qubits. We will see that in the next chapter.

CNOT gate belongs to an important class of quantum gates: the controlled-unitary gates. In the two-qubit case, controlled-unitary gates have an explicit matrix form

$$C(U) = \begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&u_{00}&u_{01}\\0&0&u_{10}&u_{11}\end{bmatrix}$$

where u_{ij} are the matrix components of the single-qubit unitary gate.

4.2 separable states and entangled states

Some of the two-qubit states can be constructed from two single-qubit states, i.e.,

$$|\psi\rangle = |\psi_1\rangle \otimes |\psi_2\rangle \tag{4.4}$$

These states are called separable states. The states that cannot be decomposed to tensor product of single-qubit states are called entangled states. Recall from the previous chapter that each single-qubit pure state has only two real parameters. Thus two-qubit separable states live in a four dimensional subspace of the six dimensional two-qubit state space. In other words, there are a lot more entangled states than separable states.

See also:

Here I only talk about pure state. The identification and quantification of entanglement in mixed states is more complicated. In short, a separable two-qubit state can be put in the form of

$$\rho = \sum_{i=1}^{K} p_i \rho_A \otimes \rho_B$$

for some K and $\{p_i\}$. You can see that it's a much harder problem than the pure state separability, because of the extra parameters $\{p_i\}$.

- Schmacher

4.2.1 Bell states

The nominal examples of two-qubit entangled states are the so-called Bell states:

$$\begin{aligned}
\left|\Phi^+\right\rangle &= |00\rangle + |11\rangle \\
\left|\Phi^-\right\rangle &= |00\rangle - |11\rangle \\
\left|\Psi^+\right\rangle &= |01\rangle + |10\rangle \\
\left|\Psi^-\right\rangle &= |01\rangle - |10\rangle
\end{aligned} \tag{4.5}$$

Here I omit the normalization factor $1/\sqrt{2}$ to save typing. Note that the four Bell states are orthogonal to each other thus form a basis set. We will see later that it is a very useful basis set to use.

Bell states can be transformed to the computational basis states $|00\rangle$, $|01\rangle$, $|10\rangle$, $|11\rangle$ by two gate operations

$$\begin{aligned}
|00\rangle &= H \otimes I \cdot CNOT \left|\Phi^+\right\rangle \\
|01\rangle &= H \otimes I \cdot CNOT \left|\Psi^+\right\rangle \\
|10\rangle &= H \otimes I \cdot CNOT \left|\Phi^-\right\rangle \\
|11\rangle &= H \otimes I \cdot CNOT \left|\Psi^-\right\rangle
\end{aligned} \tag{4.6}$$

and equivalently,

$$\begin{aligned}
CNOT \cdot H \otimes I \,|00\rangle &= \left|\Phi^+\right\rangle \\
CNOT \cdot H \otimes I \,|01\rangle &= \left|\Psi^+\right\rangle \\
CNOT \cdot H \otimes I \,|10\rangle &= \left|\Phi^-\right\rangle \\
CNOT \cdot H \otimes I \,|11\rangle &= \left|\Psi^-\right\rangle
\end{aligned}$$

The significance of the computational bases is that they represent states on hardware and in reality one can only measure these states. Thus if we know some computation results are in one of the four Bell states, we can apply CNOT and Hadamart gate to convert the results to computational basis states first, and then make the measurement.

Separable state is the quantum analogy of statistical independent random variables. Entangled states have the peculiar feature that the total system is less complex than the subsystems. To fully understand this statement, one needs to know the concept of entropy in classical information theory. For simplicity, I will use $|\Phi^+\rangle$ as example.

If we measure $|\Phi^+\rangle$ in the two-qubit computational bases, the measurement outcome follows Bernoulli distribution with $p = 0.5$, just like tossing a fair coin.

See also:

Here is a short introduction to entropy. **The essense of entropy is state counting. The more states the bigger the entropy.** And entropy quantifies the amount of information, or uncertainty, or possibilities. Suppose the system could be in N states with equal probability, then its entropy is

$$S = \log N.$$

Here the logrithm function is a historical convention, any monotonic function could be used. Its base is also arbitrary as long as used consistently. In classical information theory, $N = 2$ is of special importance because that is the number of states for a bit. In that case, base 2 is used and we have $S = \log_2 2 = 1$, i.e., one bit of information. This is also the entropy of a fair coin.

What if the coin is not fair? Here we cannot count the states directly since they are not of equal probability. Instead, we can count something else that

- both characterizes the coin bias and
- is made of individual events of equal probabilities.

This quantity is the number of configurations with certain heads given a fixed number of coin tosses. For example, suppose we make four tosses and focus on configurations with two heads. There are six such configurations: HHTT, HTHT, HTTH, THHT, THTH, TTHH. Here HTHT means the first and third tosses end up with heads. Note that these configurations are of the same probability, no matter what the coin bias is.

There are still the questions of how many tosses to do and how many heads to focus on. Obviously any finite number of tosses doesn't make sense. And the law of large numbers automatically fixes the head counts for us, i.e., with M tosses, the number of heads is

$$M_H = Mp_H$$

in the limit of large M, and p_H is the head probability. This is where the coin bias enters the picture.

This idea can be expressed as

$$S = \lim_{M\to\infty} \frac{\log_2 C(M, M_H)}{M}$$

where $C(n, k)$ is the n-choose-k function, and the denominator is there to prevent the numerator to blow up.

Using Stirling's approximation,

$$\log M! = M \log M - M + O(\log M)$$

we get the binary entropy function

$$S = -p_H \log p_H - p_T \log p_T$$

It is consistent with the fair coin case. Also the fair coin case has the largest entrpoy, which is 1.

It turns out that entanglement is a useful resource for quantum communication. I will give two examples here - quantum superdense coding and quantum teleportation. Usually people think entanglement is also important to get speedup in quantum algorithms than their classical counterparts. But this topic is a bit controversial. It turns out that entanglement is not the only non-classical correlation possessed by qubits (see quantum discord for example), and there are quantum algorithms providing exponential speedup over best-known classical algorithms and the computation processes do not contain entanglement.

4.2.2 quantum superdense coding

The purpose of quantum superdense coding is to send one qubit which carries four messages, i.e., two bits of classical information. This work is published in

- C. H. Bennett and Stephen J. Wiesner, Phys. Rev. Lett. 69, 2881 (1992)

It may appear that one qubit could encode infinite messages since its state space is continuous (recall the qubit state is a 2D complex vector in terms of wave function, or a 3D real vector on unit sphere in terms of Bloch vector). However, encoding messages in non-orthogonal quantum states messes up the decoding, due to the wave function collapse of measurement.

For example, suppose we encode four messages in

$$|0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \tag{4.7}$$

$$|1\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \tag{4.8}$$

$$|+\rangle = |0\rangle + \tag{4.9}$$

$$|-\rangle = |0\rangle - \tag{4.10}$$

and we receive an encoded qubit from someone. No matter how we measure it, we won't be sure about which state it is in. In fact, there is a so-called Holevo's bound which says one qubit can at most carry the information of one classical bit.

The trick of quantum superdense coding is to first share a Bell state before transmitting the message-carrying qubit. For example, share $|\Psi^-\rangle$ between Alice and Bob.

The encoding can be done by applying one of the four gates $\{I, X, Y, Z\}$ on Alice's qubit,

$$I\,|\Psi^-\rangle = |\Psi^-\rangle \tag{4.11}$$

$$X\,|\Psi^-\rangle = \tag{4.12}$$

$$Y\,|\Psi^-\rangle = i\,\tag{4.13}$$

$$Z\,|\Psi^-\rangle = \tag{4.14}$$

(4.15)

Then Alice can send her qubit to Bob, and Bob can decode the message by converting the Bell state to computational basis state as in Eq. (4.6) and making a measurement.

Note that quantum superdense coding does not break Holevo's bound because two qubits are sent. It's just that one qubit can be sent at an earlier time and it alone does not carry information.

4.2.3 quantum teleportation

The purpose of quantum teleportation is to instantaneously destroy a quantum state on Alice's side and create it on Bob's side without sending any qubits between Alice and Bob. This work is published in

- C. H. Bennett, G. Brassard, C. Crépeau, R. Jozsa, A. Peres, W. K. Wootters Teleporting an Unknown Quantum State via Dual Classical and Einstein-Podolsky-Rosen Channels, Phys. Rev. Lett. 70, 1895 (1993).

Again, the prerequisite is to have Alice and Bob share an entangled state, say $|\Phi+\rangle$. The initial state can be written as

$$|\xi\rangle_A \otimes (|0\rangle_A |0\rangle_B + |1\rangle_A |1\rangle_B) = (a\,|0\rangle + b\,|1\rangle)_A \otimes (|0\rangle_A |0\rangle_B + |1\rangle_A |1\rangle_B)$$

where the subscripts denote the qubits ownership.

Expressing Alice's two qubits in Bell state basis, we get

$$\begin{aligned} &\left|\Psi^+\right\rangle_A \otimes (a\left|0\right\rangle + b\left|1\right\rangle)_B \\ +&\left|\Psi^-\right\rangle_A \otimes (a\left|0\right\rangle - b\left|1\right\rangle)_B \\ +&\left|\Phi^+\right\rangle_A \otimes (b\left|0\right\rangle + a\left|1\right\rangle)_B \\ +&\left|\Phi^-\right\rangle_A \otimes (-b\left|0\right\rangle + a\left|1\right\rangle)_B \end{aligned} \tag{4.16}$$

Thus by measuring her qubits in the Bell state basis, Alice can project Bob's qubit into one of the four states. So if Alice further informs Bob about her measurement result, say from classical communication such as phone call, Bob can recover the original state $\left|\xi\right\rangle_A$. For example, if Alice's measurement result is $\left|\Psi-\right\rangle$, then Bob can apply Z gate on his qubit.

One may wonder why not sending $\left|\xi\right\rangle$ directly. One potential usage of teleportation is to send half of the Bell state when quantum communication cost is low and when the cost is high, one can use cheap classical communication to effectively transfer quantum state.

4.3 entanglement measure

Since quantum entanglement can be used as communication resource, it is of practical interest to quantify entanglement.

For two-qubit pure states, verifying whether a state is separable is simple. Given state in Eq. (4.1), we can check the validity of the following equalities

$$\begin{cases} \frac{\alpha_0}{\alpha_2} = \frac{\alpha_1}{\alpha_3} \\ \frac{\alpha_0}{\alpha_1} = \frac{\alpha_2}{\alpha_3} \end{cases}$$

A natural question to ask is: **are some entangled states more entangled than others?** For example, it seems plausible that the state

$$\left|\psi_\epsilon\right\rangle = \epsilon\left|00\right\rangle + (1-\epsilon)\left|\Psi^-\right\rangle$$

is less entangled than the Bell state and more entangled than $\left|00\right\rangle$.

There are a lot of open questions in entanglement measure. And many interesting results exist for small systems: two-qubits (dimension 4), qubit-qutrit (dimension 6), three-qubits (dimension 8). Since this is an introductory book, maybe I stop here.

- Quantum entanglement, Ryszard Horodecki, Pawel Horodecki, Michal Horodecki, Karol Horodecki, Review of Modern Physics, 81, 865-942 (2009)

CHAPTER 5

Quantum gates and quantum circuit

The purpose of computing is insight, not numbers. — Richard Hamming

5.1 quantum circuit

So far we have seen several important one-qubit and two-qubit gates, such as X gate (the equivalent of NOT gate), Hadamard gate, and CNOT gate (the equivalent of XOR gate)

$$X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix},$$
$$H = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix},$$
$$CNOT = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}.$$

A quantum circuit is built with many qubit and many quantum gates. To reveal the inner working of quantum circuit, graphic representations are created for quantum gates. For example, the Hadamard gate and CNOT gate are denoted by

Here the horizontal lines denote qubits, just like the classical case. For the CNOT gate, or any controlled-U gate, the black dot denotes the controlling qubit.

With a finite set of quantum gates, we can build all unitary operation on any number of qubits, just like in the classical computing, NOT, AND, OR gates are enough to construct any Boolean function

$$f : \{0,1\}^{\times n} \longrightarrow \{0,1\}$$

See also:

In classical computing, one gate is universal. It could be either NAND, or NOR gate. It could also be a three-bit gate called Toffli gate, which is Controlled-Controlled-NOT gate.

- David P. DiVincenzo, Quantum Gates and Circuits, Proc. Roy. Soc. Lond. A 454 (1998) 261

It's easy to show that quantum computer can do all what classical computer can do, and we will start from there.

5.2 ancilla qubits and classical logic gates

There is a gap between universal quantum gates which are low-level and general unitary operations which are high-level. This same conceptual gap exists in classical computing as well. For example, when we think of floating point multiplication, which is high-level, we don't bother to think about its low-level implementation such as

- binary representations of floating point numbers;
- bit by bit multiplications in terms of NOT, AND, OR gates;
- further encodings of the binary strings for error correction in case electronic noises screw up bits here and there.

Obviously, such high-level thinking is essential for any practical project. This poses an question of how to implement classical high-level operations in terms of quantum gates, or at least in terms of high-level quantum operations (unitary operation). This is of concern because classical operation (even the basic logic operations) are not reversible in general.

When it comes to two-bit logic gates, the situation is a little different: there are two input bits and only one output bit. Thus we can take two approaches

1. overwrite one input bit by the output whereas keep the other input bit intact 1. involve three bits in the computation: keep the input bits intact and write the output to the third bit.

As shown in the previous chapter (put in link), the first approach is sufficient to construct a unitary matrix for the XOR gate. Suppose we order the input and output as 00, 01, 10, 11 (they can be viewed as binary strings for 0, 1, 2, 3), and overwrite the second bit, then we have the CNOT gate.

5.2.1 AND gate

You can easily check that the first approach doesn't work for the AND gate: the corresponding matrix is rank-deficient because AND gate is not [logically reversible][reverse]. Thus we have to use the second approach. Note that although there are 8 input states, only 4 are meaningful. In other words, the initial state of the output bit is arbitrary. Thus we can always set it to 0 initially in real computations. This convention will fix 4 rows of the unitary matrix, corresponding to input states 000, 010, 100, and 110. Here the first two bits are input and the last one is output.

Note that since 110 is turned to 111, 111 has to be turned to 110. The remaining 3 rows are undetermined. Since they are not used in real computation, any choice will do as long as it makes the matrix unitary. The simplest choice is to have these input states map to themselves, i.e.,

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

In fact, this is an important gate in both classical and quantum computing called Controlled-Controlled NOT (CCNOT) or Toffoli gate, proposed by Dr. Tommaso Toffoli in 1980. It is known that Toffoli gate is universal for classical

computing: any boolean function can be decomposed into Toffoli gates with ancilla bits. However, to achieve universal quantum computing, extra single-qubit gates are needed.

I will leave the construction of OR gate as an exercise for you.

5.2.2 arbitrary boolean functions

Actually, there is a recipe to make arbitrary boolean functions reversible and the corresponding matrix is unitary. Without loss of generality, we can consider functions of the form

$$f : \{0,1\}^n \longrightarrow \{0,1\}$$

where n is the length of the input bit string. Boolean functions with multiple bits as output can be broken down into such 1-bit output functions.

The recipe is essentially our second approach, i.e.,:

$$(x, y) \longrightarrow (x, f(x) \oplus y)$$

where x is the input string and $\oplus$ is the XOR gate. Here the reversibility/unitariness of the gate is explicitly taken care of by the XOR gate. In real computations we can always set $y = 0$ then the last bit is simply the output $f(x)$. Applying this recipe to the AND gate, you will get Toffoli gate.

This trick is often called Controlled-f gate.

5.3 quantum no-cloning theorem

There is an important theorem in quantum computing that reveals fundamental limitations to quantum state manipulations such as state preparation and state readout. The so-called no-cloning theorem says that **it is impossible to copy an unknown quantum state**.

At high-level, this theorem appears obvious because copying is not a reversible operation, thus cannot be implemented as a unitary operation. However, copying is a valid logic operation, thus can be implemented as a unitary matrix using the controlled-f gate trick. Is there a contradiction?

Using the trick of controlled-f gate, classical state copy can be written as

$$(p, q) \longrightarrow (p, q \oplus p)$$

To copy bit p to bit q, we initialize bit q in the 0 state. The corresponding matrix is the CNOT gate.

Applying CNOT gate to the quantum initial state

$$CNOT(\alpha\,|0\rangle + \beta\,|1\rangle) \otimes |0\rangle = \alpha\,|00\rangle + \beta\,|11\rangle$$

Note that the result of cloning is

$$(\alpha\,|0\rangle + \beta\,|1\rangle) \otimes (\alpha\,|0\rangle + \beta\,|1\rangle).$$

Thus the action of CNOT gate creates entanglement in the qubits, but does not fulfill quantum cloning. There is no contradiction here because the controlled-f gate trick provides unitary matrices whose action on the classical bit space fulfills the classical logic operations. It does not guarantee the same operation to work on a general quantum state, which has no classical analogy.

5.4 quantum parallelism

The power of quantum computer can be demonstrated in a simple setup. Suppose we are interested in a Boolean calculation on n bits

$$f : \{0,1\}^{\times n} \longrightarrow \{0,1\}^{\times m}$$

and the corresponding controlled-f version on quantum computer is U_f.

We can prepare a special input state

$$H^{\otimes n} |0\rangle^{\otimes n} = (|0\rangle + |1\rangle)^{\otimes n} \tag{5.1}$$
$$= |0\rangle + |1\rangle + |2\rangle + \cdots + |2^n\rangle \tag{5.2}$$

Here I mix two notations and $|0\rangle$ and $|1\rangle$ refer to either single qubit or n qubit states. The interesting properties of this input state include

- It includes all input states of n bits.
- It is not an entangled state.

Then one application of U_f gate gives rise to all output states of the classical Boolean calculation. If one were to compute them classically, 2^n application of f gate would be needed. Thus there appears to be an exponential speedup in the quantum case. This feature is called quantum parallelism.

However, the superimposed output state is not directly useful because the readout collapses the output state on the computational basis. Only special computational task can fully utilize this exponential speedup, and sometimes the exponential speedup becomes quadratic speedup. We will see examples in the algorithm chapter and state readout chapter.

5.5 quantum gate design

Finally I will talk about how one designs quantum gate in reality.

There are many qubit implementations nowadays and their Hamiltonians are all somewhat different (recall that Hamiltonian is a matrix to describe a quantum system's dynamics). Thus the gate implementation differs on different hardware platforms. Even for the same hardware, it is possible to implement the same quantum gate using different strategies.

Typically part of the Hamiltonian can be controlled. And quantum gate design is formulated as an optimization problem. Suppose the controllable degrees of freedoms are denoted as $\lambda(t)$, then we have

$$argmin_{\lambda(t)} \|U(t) - U_{gate}\|$$

where U_{gate} is the desired gate.

For a time-dependent Hamiltonian $H(t)$, it is tricky to calculate its time evolution $U(t)$. Note that the formula $U(t) = \exp(-iHt)$ only works when H is time-independent. The full treatment of the problem requires quantum control theory. A simplified view is to discretize the time into small intervals, and assume constant Hamiltonian within each interval. The interested reader could start from the following paper of the GRAPE algorithm

- Navin Khaneja, Timo Reiss, Cindie Kehlet, Thomas Schulte-Herbrügge, Steffen J. Glaser, Optimal control of coupled spin dynamics: design of NMR pulse sequences by gradient ascent algorithms, Journal of Magnetic Resonance 172(2), 296 (2005)

CHAPTER 6

State preparation

6.1 preprare the $|0\rangle$ state

Most quantum algorithms assume the initial state to be $|0\rangle^{\otimes n}$, i.e., all qubits are in the $|0\rangle$ state. But how do we prepare the qubits in that state?

State prepration is strongly tied to the physical implementation. For different systems, the qubit states $|0\rangle$ and $|1\rangle$ mean different things. For simplicity, let's assume that they corresponds to the ground state and some excited state of a physical system. In this case, an easy but potentially expensive way to prepare the $|0\rangle$ state is to cool the qubit towards absolute zero, i.e., -273.15 degree Celsius.

20mK dilution refrigerator

6.2 prepare arbitrary state

After getting a high quality $|0\rangle$ state, other states can be prepared by applying quantum gates.

In practice, this approach has its limitations because

bath engineering

The disadvantage of the previous method is that applying quantum gates is costly. Current quantum processor can only retain its quantumness for a short period of time, which translates to a limited number of gates ().

Due to the no-cloning theorem It is impossible to copy an unknown quantum state.

quantum teleportation can be seen as state preparation.

CHAPTER 7

State readout

- many copies of the same quantum state
- prior information on what the quantum state could be

Suppose we do not have any prior information about a quantum state, it is impossible to know what it is. Take 1-qubit for example, an arbitrary state takes the form

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

A von-Neumann measurement collapses the state to either of the two basis state probabilistically. Thus with a single copy of $|\psi\rangle$, no information on α or β can be retrieved.

7.1 state tomography

Due to the no-cloning theorem It is impossible to copy an unknown quantum state.

The situation is different when we have multiple copies of the same quantum states. Due to the *quantum no-cloning theorem*, one needs to repeat the state preparation to create the copies.

$|\psi\rangle$

7.2 phase kickback

Its main idea can be demonstrated with two qubits. Suppose we are given a one-qubit unitary gate U (the term 'gate' is interchangeable with 'time evolution' or 'operator') and one of its eigenstates $|\psi\rangle$, i.e.,

$$U\,|\psi\rangle = e^{i\phi}\,|\psi\rangle$$

and our task is to find out what ϕ is (up to 2π wraps).

Note that there are two ways to think of this unitary gate

- white box: U is known but too difficult to diagonalize
- black box: U is not known but we are given a button to click which applies it on the qubit

The word 'known' is a bit tricky. Does it mean knowing at high-level what U is, or knowing every entry of the matrix if it has a matrix representation, or knowing every entry of the corresponding Hamiltonian? Fortunately, for our purposes, this detail can be omitted.

Recall that pure phase factor on a quantum state is not measurable (commonly known as $U(1)$ symmetry), only relative phase on different states is. Thus given only the unitary gate and the eigenstate, there is no hope to get ϕ. With some extra resources, it becomes possible, and it is exactly the aim of phase kickback. The three extra resources in phase kickback are

- an extra qubit, commonly known as ancilla (the latin word for 'maid') qubit
- a way to do Hadamard gate on the ancilla qubit
- a way to do controlled-unitary gate on the two qubits

In the more general case where U acts on multiple qubits, more ancilla qubits may be needed. If we have a general-purpose quantum computer, all these resources are available.

Before we proceed, we can also think about whether less resources can be used to extract ϕ. For example, what if we only have all possible single-qubit gates at hand, i.e., we can render any unitary evolution on $|\psi\rangle$? Depending on how much we know about $|\psi\rangle$, it could be possible to get ϕ. I will leave these details to you to think about.

The concept of controlled-unitary gate may also need some explanation. In our two-qubit example, it is a two-qubit gate whose action on the controlled qubit depends on the state of the control qubit. Usually, the ancilla qubit (or qubits) is used as control. If the control qubit is in state $|0\rangle$, then nothing happens to the controlled qubit. If the control qubit is in state $|1\rangle$, then the single-qubit unitary gate is applied to the controlled qubit. Also if the control qubit is in a superposition state, the superposition of the action happens. We are only interested in controlled-unitary gates instead of general controlled gates since the time evolution of quantum systems is unitary.

Now we are ready to describe the phase kickback protocol: start with $|0\rangle|\psi\rangle$, apply the Hadamard gate on the ancilla qubit and then the controlled-unitary gate on the two qubits. One can easily verify the resultant state as

$$C(U)H \otimes I\,|0\rangle\,|\psi\rangle = \frac{|0\rangle + e^{i\phi}\,|1\rangle}{\sqrt{2}}\,|\psi\rangle$$

where I is the identity matrix, $\otimes$ is the tensor product operation, and the one-qubit Hadamard gate is

Note that the overall effect is to add a phase shift to the control (ancilla) qubit. This is opposite to the common sense that the control bit remains intact and the controlled bit changes. And this is why it is called phase kickback.

For example, the controlled-unitary gate could be a controlled-phase (CPhase) gate and the eigenstate could be $|\psi\rangle = |1\rangle$. Then we have

$$C_\phi H \otimes I\,|0\rangle\,|1\rangle = \frac{|0\rangle + e^{i\phi}\,|1\rangle}{\sqrt{2}}\,|1\rangle$$

where the quantum gates are given by

$$C_\phi = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & e^{i\phi} \end{bmatrix}$$

To further extract the phase ϕ on the ancilla qubit, there are various options. The most straightforward one is to generate many copies of this state, and keep measuring the three physical observables

$$\langle\sigma_x\rangle = \cos\phi, \quad \langle\sigma_y\rangle = \sin\phi, \quad \langle\sigma_z\rangle = 0$$

where σ_i's are the Pauli matrices. This is basically the Bloch sphere representation of quantum two-level systems. Thus in principle ϕ can be determined as accurate as one wishes. However, it is not efficient to estimate ϕ this way (unless in special situations, say ϕ is known to be one of a few possible values) due to the cost of generating the copies. There are other **efficient** ways to measure ϕ, for example, using [quantum Fourier transform].

It turns out that **many quantum algorithms boil down to somehow encode the answer in the phase of the ancilla qubits, with the help of controlled-unitary gates**. Thus it is very helpful to think in this phase kickback framework.

7.3 quantum Fourier transform (QFT)

7.3.1 discrete Fourier transform (DFT)

QFT is closely related to discrete Fourier transform (DFT), an important tool in digital signal processing. And we will start from there.

As the name indicates, DFT is the discrete version of Fourier transform. Most commonly, the input sequence is time series data or spatial samples, and the output sequence is frequency data, i.e., the Fourier spectrum. Overall, it is

- a linear map between two sequences of complex numbers;
- a non-degenerate map with an inverse, i.e., the inverse discrete Fourier transform (IDFT);
- a map that's cheap to compute: the fast Fourier transform (FFT) algorithm has computational complexity $O(N \log N)$ (instead of $O(N^2)$ for a general linear map)

In practice, DFT is widely used because

- In signal processing, often times the signals are band-limited. Then the Fourier spectrum provides a more succinct representation of the signals;
- In physics problem, often times the symmetry can be more easily utilized in the Fourier domain;
- It is useful for convolution calculations;
- It is efficient to calculate;
- ...

In 1D, DFT can be defined as

$$X_k = \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} \omega_N^{-jk} x_j$$

where $\omega_N \equiv \exp(2\pi i/N)$ is the N'th root of unity and $i = \sqrt{-1}$. And the IDFT is given by

$$x_j = \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} \omega_N^{jk} X_k$$

Generalization to higher dimensions is straightforward.

Different authors use different normalization conventions for DFT and IDFT. I have the impression that physicists and engineers prefer to have 1 in DFT and $1/N$ in IDFT, whereas mathematicians prefer this $1/\sqrt{N}$ normalization. For QFT, $1/\sqrt{N}$ is better as it normalizes the wave functions correctly.

There is also arbitrariness in the range of the summation index j: any N consecutive integers will do. Sometimes it is more convenient to include both positive and negative values. To complicate things even more, some authors swap the definitions of DFT and IDFT. Thus one needs to be very cautious with other people's formulas.

The DFT transformation has explicit matrix form. For example,

$$DFT_2 = \frac{1}{\sqrt{2}}\begin{bmatrix}1 & 1\\ 1 & -1\end{bmatrix}, \quad DFT_4 = \frac{1}{2}\begin{bmatrix}1 & 1 & 1 & 1\\ 1 & -i & -1 & i\\ 1 & -1 & 1 & -1\\ 1 & i & -1 & -i\end{bmatrix}$$

In general, the DFT matrix looks like

$$DFT_N = \frac{1}{\sqrt{N}}\begin{bmatrix}1 & 1 & 1 & \cdots & 1\\ 1 & \omega_N^{-1} & \omega_N^{-2} & \cdots & \omega_N^{-(N-1)}\\ 1 & \omega_N^{-2} & \omega_N^{-4} & \cdots & \omega_N^{-2(N-1)}\\ \vdots & \vdots & \vdots & \cdots & \vdots\\ 1 & \omega_N^{-(N-1)} & \omega_N^{-2(N-1)} & \cdots & \omega_N^{-(N-1)(N-1)}\end{bmatrix}$$

You can easily check that the rows are all perpendicular to each other (remember to take complex conjugate). Thus each component of the DFT output, i.e., X_k, is a projection of the input to one of a set of orthogonal directions. If you visualize the ω_N^j in the complex plane, then each row can be seen as a rotating unit vector with different angular velocity. And the action of each row is to extract the component of a specific angular velocity.

There are a lot of subtleties in DFT. For example, given a continuous function (analog signal), should I do Fourier transform first, then sample the frequency domain function? Or sample the continuous function first, and then do DFT on the time domain samples? My favorite theorem along these lines is the Shannon sampling theorem, which answers the following questions:

- Does a continuous function have infinite degrees of freedom since there are infinite input values (like in classical field theory)? This is plausible but is also complicated by the continuity requirement.
- If a function is band-limited, does it still have infinite degrees of freedom?

7.3.2 QFT, Hadamard transform, and measurement

In fact, QFT is exactly DFT, with the input/output vectors being probability amplitudes of quantum states. Thus it is a transformation between quantum states, i.e., a quantum gate.

For quantum computing, the state vector is of dimension $N = 2^n$ where n is the number of qubits. Following the arguments in the previous section, let's look at two sets of special states, both of which consists of only separable states:

- states corresponds to the rows of the Walsh-Hadamard matrix: $\prod_{j=0}^{n-1} \otimes (|0\rangle + (-1)^{r_j} |1\rangle)$
- states corresponds to the rows of the IQFT/IDFT matrix: $\prod_{j=0}^{n-1} \otimes \left(|0\rangle + \omega_{2^n}^{jr} |1\rangle\right)$

where $r = 0, 1, .., 2^n - 1$ is the row index of the n-qubit transform matrix, r_j is the j'th digit of r's binary representation, and $\otimes$ denotes tensor product. This QFT tensor product decomposition can be easily derived from small n cases and induction.

Note that the Walsh-Hadamard gate is particularly easy to implement since it can be decomposed as 1-qubit Hadamard gates, i.e., $H_n = H_1^{\otimes n}$. In the QFT case, the qubits do not fully decouple (note the coefficient before $|1\rangle$ depends on r instead of a single binary digit of r), thus a naive QFT implementation requires $n - 1$ 1-qubit controlled-phase gate on each qubit, which is already efficient ($O(n^2)$ 1-qubit gates) on a quantum computer. A more careful examination will reveal that many of them can be avoided.

Recall that classical FFT has complexity $O(N \log N)$ where $N = 2^n$, i.e., $O(n2^n)$. Thus it appears that quantum computer can calculate DFT exponentially faster. This is actually not completely true: the catch is that the DFT result (i.e., the probability amplitudes) is not accessible in general due to the measurement/readout problem.

CHAPTER 8

Quantum algorithms

- amplitude amplification
- period finding

8.1 Deutsch-Jozsa algorithm

The complete version of Deutch algorithm solves the following problem: given a binary function f defined on binary strings of n digits,

$$f : \{0,1\}^n \to \{0,1\},$$

with the guarantee that f is either constant or balanced (meaning that half of the answer is 0), determine whether it is constant or balanced.

On a classical computer, we will have to evaluate $2^{n-1}+1$ inputs to know the answer. In contrast, Deutch algorithm needs only one measurement on n qubits: an exponential speedup! For simplicity, I will only consider the $n = 1$ case. Generalizing to the n-qubit case is somewhat straightforward. Again, in this simplified situation, classically we need to evaluate two inputs. With a quantum computer, we only need one readout on one qubit.

The first thing we need to worry about is how to implement f. Obviously it cannot be applied on quantum states directly since its domain is binary strings. Without much surprise, it is implemented as a controlled unitary gate, with the action

$$|x\rangle\,|y\rangle \longrightarrow |x\rangle\,|f(x) \oplus y\rangle$$

where $\oplus$ is the XOR gate. This action is obviously reversible, thus there must be a way to implement it on a quantum computer as a unitary time evolution. Let's call it U_f. Note that its action on $|y\rangle$ is controlled by the value of $f(x)$ instead of the ancilla qubit state directly. It is a CNOT gate gate controlled by the value of $f(x)$. Let me also assure you that in the n-qubit case, such U_f can be implemented on a quantum computer without exponential trouble. Thus the exponential speed up is real.

With a hardware implementation of U_f, we can apply it on a special initial state, i.e.,

$$C(U_f)H \otimes H\,|0\rangle\,|1\rangle$$

where the first qubit is the ancilla qubit, and the controlled qubit is in the eigenstate of the NOT gate (or X-gate since it's the Pauli x matrix) $H|1\rangle$ with eigenvalue -1, or equivalently $\phi = \pi$. After simplification, the ancillar qubit is given by

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

Recall that f is either constant or balanced. Thus the phase kickback is either 0 or π, which is particularly easy to distinguish since they are eigenstates of σ_x. One can measure σ_x just once to tell them apart, with only one copy of the input state. In the quantum computing framework, the equivalent procedure is to first apply [Hadamard gate][h] and then check whether the state is $|0\rangle$ or $|1\rangle$.

Admittedly, Deutch algorithm is quite artificial. But it opens up the possibility of exponential speedup on quantum computers.

8.2 phase estimation algorithm (PEA)

In special occasions, the exponential speedup materializes. It happens when the answer corresponds to one of the QFT basis states. Then the application of QFT followed by a measurement can tell us the answer. This is the main idea of the quantum phase estimation algorithm (PEA), a quantum algorithm for eigenvalues and eigenvectors.

A simplified version of it can be stated as follows. Suppose we are given an eigenstate of a unitary gate/operator satisfying

$$U|\psi\rangle = e^{i\phi}|\psi\rangle$$

where ϕ is guaranteed to be smaller than 2pi, estimate ϕ.

For now, let's assume that ϕ happens to be ω_N^k with an unknown k (here we assume we somehow know N and again $N = 2^n$). Then we can use controlled-U gates and N ancilla qubits to do the *phase kickback* trick with the output state

$$\prod_{j=0}^{n-1} \otimes \left(|0\rangle + \omega_N^{jk}|1\rangle\right)$$

In other words, we can transform it to be one of the QFT basis states and the unknown value $$k$$ is exactly the row index. Then it only takes a QFT transformation and one measurement to reveal k.

What if ϕ is not ω_N^k? A moment of reflection will show that the aforementioned protocol produces the best N-bit approximation of ϕ (in this case one also needs to repeat this protocol several times to be sure).

- Quantum algorithms revisited by Richard Cleve, Artur Ekert, Chiara Macchiavello, Michele Mosca (1998)
- Quantum Algorithm Implementations for Beginners (2018)

8.3 quantum Approximate Optimization Algorithm (QAOA)

8.4 quantum random walk

CHAPTER 9

Obstacles

So far we have been pretending that qubits are two-level artificial atoms. In reality, they are implemented from

- superconducting circuits
- atoms or molecules
- quantum dots
- ion traps
- photonss (light particles)

and many other systems.

Most of these qubits have more than two states. Thus using them as qubits are like using two sides of a die to do coin flipping. Sooner or later, we will see the other four sides. This is a problem known as the qubit going outside the computational states.

The second problem is the imperfection of quantum control, which manifests both in the state preparation and gate implementation.

The third and probably the most damaging problem is decoherence. This is an umbrella term for the irreversible change of the qubits even if the experimentalist does not do anything. For example, suppose a qubit is implemented by an atom and the two qubit states correspond to two energy states of the atom. The higher energy state has some probability to spontaneous lose energy and drop to the lower energy state.

Another common scenario is that the experimentalist does not know or have control over certain interactions between the qubit and its environment.

See also:

Density matrix ρ and other types of equations for dynamics are needed to describe decoherence process. Wave function $|\psi\rangle$ and Schrodinger's equation is not sufficient in the situtation because they only describe reversible dynamics.

9.1 DiVincenzo criteria

Back in the 90's, Dr. Daniel Loss and Dr. David DiVincenzo proposed five criteria for a quantum system to be eligible for quantum computer. It is commonly known as the DiVincenzo's criteria.

- D. Loss and D. P. DiVincenzo, Quantum computation with quantum dots, Phys. Rev. A 57, 120 (1998)

The criteria are

1. A scalable physical system with well characterized qubits.
2. The ability to initialize the state of the qubits to a simple fiducial state.
3. Long relevant decoherence times.
4. A universal set of quantum gates.
5. A qubit-specific measurement capability.

- quantum decoherence: This is about how things could go wrong and it is the reason why we don't have an awesome quantum computer yet.
- quantum error correction: This is the counterpart of classical error correction and it is meant to fight decoherence.
- decoherence
- scalability
- gate fidelity
- state prep

CHAPTER 10

What's next

> The future is already here. It's just not very evenly distributed. — William Gibson

In this book we only have a superficial acquitance of limited topics. Apart from hiding all hardware implementations, we also fully ignored many important topics.

- books
  - Quantum Computation and Quantum Information (2000) by Michael Nielsen and Isaac Chuang: This is a comprehensive textbook, often used in undergraduate and graduate quantum computing courses.
  - Quantum Computer Science (2007) by N. David Mermin (age 83 this year)
  - Quantum Computing since Democritus (2013) by Scott Aaronson: This book talks about many interesting topics. You can take a glimpse on Scott Aaronson's website.
- lecture notes
  - Professor John Preskill's Physics229 at Caltech
  - Professor David Mermin's CS483 at Cornell
- news feed
  - quantum computing report
- quantum computer and simulator on the cloud
  - IBM's quantum experience
  - Google's playground
- papers: most quantum computing papers can be found on arxiv quant-ph Below is a list of papers that are particularly pedagogical:

misc

- Building logical qubits in a superconducting quantum computing system by Jay M. Gambetta, Jerry M. Chow, Matthias Steffen ???

CHAPTER 11

News

- **Oct.2017**
  - Microsoft released a preview of its quantum computing development kit
  - Intel announced 17-qubit processor
  - IBM demonstrated simulation of 56 qubits on classical computer with 3TB memory cost
  - Google announced an open source chemistry package called OpenFermion
- **Nov.2017**
  - IBM announced 20-qubit processor for clients and 50-qubit prototype
- **Jan.2018**
  - Intel announced 49-qubit processor
- **Mar.2018**
  - Google announced the new Bristlecone quantum processor with 72 qubits
- **Jun.2018**
  - IBM released ACQUA (Algorithms and Circuits for QUantum Applications) library
- **Jul.2018**
  - Google released an open source framework for NISQ algorithms