Lectures on Quantum Computing Physics 709

Lecture 1

Introduction, States, and Operators

1 Introduction

Why study quantum computing? The field has become a very active subject of research mainly because of the possibility of constructing new computing machines that would be vastly more powerful than our present-day computers, though only for certain types of problem. The most famous such problem is that of factoring large numbers, but it has also been shown that quantum computers (QCs) could search databases very quickly, and efficiently simulate large physical systems that obey quantum mechanics. There are several other interesting but more specialized algorithms as well. An up-to date list is at http://quantumalgorithmzoo.org/. I counted 61 algorithms last time I looked. It is a very exciting research problem to find new algorithms that would make full use of the power of QCs.

Beyond these quite specific problems, most people believe that if the manipulations necessary for quantum computation can be perfected, then there will be applications that are still not dreamed of. In the early days of classical computing in the 1940s and 50s, it was believed that computers would be of very limited usefulness. We know how that turned out. In fact the possibilities go well beyond computing. In fact, many believe that just as the technological advances of the 19th century came from thermodynamics, and the advances of the 20th century came from electricity, so will the advances of the 21st century come from the quantum.

The subject has tremendous intrinsic interest as well - we are utilizing the quantum character of the world in entirely new ways, and greatly deepening our understanding of quantum physics as we do so. One of the great intellectual advances of the later 20th century was the theoretical quantification of classical information mainly by Shannon. This went hand in hand with all the great advances in information technology. This has three important aspects: the physical storage of information (memory), the communication of information (cell phones, optical fibers,...), and the manipulation of information (computers). These three revolutionary discoveries depend on physical systems that have a classical description. At a deeper level, though, the world is not classical, it is quantum, and we need to change our description if we are to truly understand information storage, communication, and manipulation in the real world.

2 Stern-Gerlach Experiment

We are going to start with a famous experiment from 1922. Stern and Gerlach sent a beam of silver atoms through a region of magnetic field $\vec{B}$, arranging the configuration of the magnets so that there was a strong field gradient dB_z/dz. The silver atoms have a magnetic moment because of the unpaired spin of one of the electrons on the atom, (though they were not fully aware of that at the time). This spin $\vec{S}$ is an internal angular momentum. There is a force in the z-direction given by $F_z = \mu_z \, dB_z/dz$, where μ_z is the z-component of the magnetic moment of the unpaired electron. μ_z is proportional to S_z, and for electrons the z-component of the magnetic moment is given by $\mu_z = -(e/mc) S_z$, to a very good approximation.

In classical mechanics we would expect that the electrons have some random continuous distribution of magnetic moments in the z-direction, since nothing is done to align them before they are passed through the field. If this were true, the atoms would just spray out of the region of the magnetic field with a continuous distribution of momenta in the z-direction, since they have been subjected to a random force. What is observed does not fit this idea. Rather, the beam splits into just two beams with equal and opposite deflection angle in the z-direction. When the spin angular momentum is computed, we find that $S_z = \pm\hbar/2$. As we said, nothing was done to prepare the spins, which had a thermal distribution, and the two possible values of the z-component were observed with probability 1/2. This situation is shown in Fig. 1.

Let's go a step further. Let's pass the electron through a set of magnets sequentially, perhaps changing the spatial orientation of the magnets. See Fig. 2. If we stop one of the 2 beams, say the lower beam, and pass the remaining upper beam through a z-magnet we find that the electrons form only a single upper beam. If instead we pass that upper z-beam through an x-magnet the beam once again splits in 2, an upper and a lower x-beam. If we stop one of those x-beams and pass it through a z-magnet, then again it splits into 2.

Somehow the magnet affects the state of the electrons. How do we describe this mathematically? Call the state of the upper and lower beams after passing through a z-magnet z_+ and z_-, respectively. The beam stop for the z_- electrons gives a set of electrons all in the z_+ "quantum state". The original incoming beam was an equal mixture of electrons in the z_+ and z_- states and we have removed the latter. The strange part is that when we pass them through the x-magnet we find that the z_+ electrons are in a superposition of x_+ and x_- quantum states. Thus we have + and − states for any space direction. In the Dirac notation, the z_+ and z_- states are often denoted by $|\uparrow\rangle$ and $|\uparrow\rangle$ and the x_+ and x_- states by $|\rightarrow\rangle$ and $|\leftarrow\rangle$ where the direction of the arrow denotes the spin direction. We will go directly to the quantum computing notation, in which the z_+ and z_- states are written as $|0\rangle$ and $|1\rangle$. There are two reasons for the notation. The first is sort of obvious: the spin is a qubit and 0 and 1 are the usual symbols for the two states of a classical bit. The second is that there are many kinds of qubits and not all of them are spins with a spatial direction. So an arrow is not appropriate. The $|\rangle$ notation is what comes from Dirac.

It was his way of writing a vector and though it looks strange at first, it has many advantages. Dirac notation is the standard in the quantum computing literature.

3 Kets

Dirac called these states "kets". They are vectors and thus they are members of a vectors space.

Kets can be added

$$|\alpha\rangle + |\beta\rangle = |\gamma\rangle$$

or multiplied by a complex number

$$c\,|\alpha\rangle\,,$$

and the result is still a ket. In other words, the set of kets for a quantum system is closed under addition and scalar multiplication by a complex number and it is therefore a complex vector space. A key point is that ALL of the kets in the space represent allowable quantum states. Thus $|0\rangle$ and $|1\rangle$ are both allowable states, as we have seen, but the x_+ and x_- states are also in the space and are given by $\left(1/\sqrt{2}\right)(|0\rangle + |1\rangle)$ and $\left(1/\sqrt{2}\right)(|0\rangle - |1\rangle)$, as we shall see later. The states of a spin 1/2 particle are in a 2-dimensional space, and can all be written as linear combinations of $|0\rangle$ and $|1\rangle$.

The states of a spin 1 particle are members of a 3-dimensional space. The quantum states of translational motion of a particle in space are members of an infinite-dimensional space. Understanding what the space is our first task when confronted by a quantum system. All qubits live in a 2-dimensional space and the state of a quantum computer with N qubits is a member of a 2^N-dimensional space. We never have to deal with infinite-dimensional spaces in this course.

Actually, any multiple $c\,|\alpha\rangle$ also represents the same quantum state as $|\alpha\rangle$, but we will discuss that in more detail later.

4 Operators

We also define operators A:

$$A\,|\alpha\rangle$$

is a ket. Viewed purely mathematically, an operator is a mapping of the ket space into itself. Multiplication by a complex number is a special case of an operator. A *linear* operator also satisfies

$$A\,(c\,|\alpha\rangle + d\,|\beta\rangle) = cA\,|\alpha\rangle + dA\,|\beta\rangle\,.$$

It turns out that in quantum mechanics we are only concerned with linear operators (with the very occasional exception of antilinear operators, but we will not talk about them in this course).

An equation of the form

$$A\,|\alpha\rangle = a\,|\alpha\rangle$$

where a is a complex number is called an eigenvalue equation. If this equation holds, then we say that $|\alpha\rangle$ is an eigenket of the operator A, and that a is an eigenvalue of A belonging to the ket $|\alpha\rangle$.

5 Bras and the Inner Product

We now define the bra space. A bra is written as $\langle\alpha|$. Mathematically, bras are linear maps that take kets into complex numbers (unlike linear operators, which take kets into other kets). The set of bras is also a complex vector space and indeed the bras are in a one-to-one correspondence with the kets:

$$\langle\alpha| \longleftrightarrow |\alpha\rangle\,,$$

where the double-headed arrow denotes this correspondence. Thus it is consistent and sometimes useful to say that the bras also represent states of the quantum system and $\langle\alpha|$ represents the same state as $|\alpha\rangle$. The bra-ket correspondence preserves addition:

$$\langle\alpha| + \langle\beta| \longleftrightarrow \langle\alpha| + |\beta\rangle$$

but the correspondence under multiplication involves a complex conjugation:

$$\langle\alpha|\,c^* \longleftrightarrow c\,|\alpha\rangle\,.$$

The reason we introduce the bra space is that it allows us to define the inner product of a bra and ket, which is written as $\langle\alpha|\,|\beta\rangle$, (or sometimes as $\langle\alpha|\beta\rangle$) and the result is a complex number. Mathematically, the inner product is a mapping from the set of bra-ket pairs to the set of complex numbers. The inner product has linearity properties. For example

$$\begin{aligned}(\langle\alpha| + \langle\beta|)\ |\gamma\rangle &= \langle\alpha|\,|\gamma\rangle + \langle\beta|\,|\gamma\rangle \\ \langle\alpha|\ (|\gamma\rangle + |\delta\rangle) &= \langle\alpha|\,|\gamma\rangle + \langle\alpha|\,|\delta\rangle\end{aligned}$$

Also $\langle\alpha|\,|\beta\rangle = \langle\beta|\,|\alpha\rangle^*$. Usually the double vertical line is abbreviated and this equation is written as $\langle\alpha|\ \beta\rangle = \langle\beta\ |\alpha\rangle^*$

Now we can see why Dirac called these objects bras and kets - when you put them together to form an inner product you get a bracket. Those interested in mathematics may wish to know that the bra space is called the dual space to the ket space, and its existence, and that of the inner product, is guaranteed by the Riesz representation theorem. When the vectors are put together with the inner product, (plus a technical requirement or two in the case of infinite dimensions) the whole thing is called a Hilbert space.

It is reasonable to think of the inner product as being analogous to the ordinary dot product of vectors, but one must always keep in mind that the

inner product is a complex number, so the geometrical interpretation is not quite the same. This problem doesn't arise when we take the inner product of a state with itself, because

$$\langle\alpha|\,|\alpha\rangle = \langle\alpha|\,|\alpha\rangle^*$$

is real. It is also positive, but that is an axiom and cannot be proved from the definitions above. So $\sqrt{\langle\alpha|\,|\alpha\rangle}$ may be thought of as the norm of the ket vector $|\alpha\rangle$, and $\langle\alpha|\,|\alpha\rangle = 0$ if and only if $|\alpha\rangle$ is the zero vector. Furthermore, we say that $|\alpha\rangle$ and $|\beta\rangle$ are orthogonal if

$$\langle\alpha|\,|\beta\rangle = 0$$

and $|\alpha\rangle$ is normalized if

$$\langle\alpha|\,|\alpha\rangle = 1.$$

6 Operator Algebra

Our linear operators also form a vector space, that is if X and Y are linear operators, then $cX + dY$ is also a linear operator. The bra-ket correspondence also carries over to operators. It is

$$X\,|\alpha\rangle \longleftrightarrow \langle\alpha|\,X^\dagger,$$

which defines the adjoint operator: $X^\dagger$ is called the adjoint of X. If $X = X^\dagger$, then we say that X is self-adjoint, or Hermitian. Operators can be multiplied, so if X and Y are operators, then XY is also an operator. So

$$XY\,|\alpha\rangle$$

is the ket obtained by first applying Y to $|\alpha\rangle$ and then applying X to the result. Multiplication of operators is associative:

$$X\ \ (YZ)\,|\alpha\rangle = (X\ Y)\,Z\,|\alpha\rangle\,,$$

but not generally commutative. So it can happen that

$$X\ Y\ \ |\alpha\rangle \neq YX\ \ |\alpha\rangle\,.$$

The set of linear operators, in mathematical terms, is a non-commutative algebra. The commutator of two operators (which is again an operator) comes up quite often and it is denoted by a square bracket: $[X, Y] = XY - YX$. Note that $(XY)^\dagger = Y^\dagger X^\dagger$ - you should prove this for yourself. If an operator commutes with its Hermitian conjugate: $\left[A, A^\dagger\right] = 0$, then A is called a normal operator. All Hermitian operators are obviously normal.

It is possible to make operators from bras and kets by reversing their usual order. Let the operator X be defined by

$$X = |\beta\rangle\,\langle\alpha|\,.$$

This expression is defined by giving the action of X on an arbitrary ket:

$$X\left|\gamma\right\rangle = \left|\beta\right\rangle \quad \left\langle\alpha\right|\left|\gamma\right\rangle = \left\langle\alpha\right|\left|\gamma\right\rangle \quad \left|\beta\right\rangle,$$

is a complex number multiplying the ket $\left|\beta\right\rangle$, so it is a ket. A very useful operator that can be made in this way is $P_\alpha = \left|\alpha\right\rangle\left\langle\alpha\right|$. This is the projection operator onto the state $\left|\alpha\right\rangle$. It satisfies $P_a^2 = P_\alpha$, which is the definition of a projection operator.

Lecture 2
Hermitian Operators and Time Evolution

1 Properties of Hermitian Operators

We will now prove two very important theorems about Hermitian operators.

Theorem 1. The eigenvalues of a Hermitian operator are real.

Proof. Let $|\alpha\rangle$ be an eigenvector of A. Then

$$A|\alpha\rangle = a|\alpha\rangle,$$

where a is a complex number. Taking the inner product with $\langle\alpha|$ gives

$$\langle\alpha|A|\alpha\rangle = a\langle\alpha||\alpha\rangle$$

The adjoint correspondence tells us that we also have

$$\langle\alpha|A^\dagger = a^*\langle\alpha| = \langle\alpha|A,$$

and taking the inner product of this equation with $|\alpha\rangle$ gives

$$\langle\alpha|A^\dagger|\alpha\rangle = \langle\alpha|A|\alpha\rangle = a^*\langle\alpha||\alpha\rangle.$$

Comparing the two results yields $a = a^*$ so a is real.

Theorem 2. Eigenkets of a Hermitian operator belonging to different eigenvalues are orthogonal.

Proof. Let $|\alpha\rangle$ and $|\alpha'\rangle$ be two different eigenvectors of the Hermitian operator A. Then

$$\begin{aligned} A|\alpha\rangle &= a|\alpha\rangle \qquad (1)\\ A|\alpha'\rangle &= a'|\alpha'\rangle, \end{aligned}$$

where $a \neq a'$ are real numbers. The adjoint correspondence tells us that we also have

$$\begin{aligned} \langle\alpha|A &= a\langle\alpha|,\\ \langle\alpha'|A &= a'\langle\alpha'| \qquad (2) \end{aligned}$$

Taking the inner product of (1) with $\langle\alpha'|$, and of (2) with $|\alpha\rangle$,we have

$$\begin{aligned} \langle\alpha'|A|\alpha\rangle &= a\langle\alpha'||\alpha\rangle,\\ \langle\alpha'|A|\alpha\rangle &= a'\langle\alpha'||\alpha\rangle. \end{aligned}$$

Subtracting these equations gives

$$(a - a')\langle\alpha'||\alpha\rangle = 0,$$

so if a and a' are different we must have $\langle\alpha'|\,|\alpha\rangle = 0$. If $a = a'$, there is no reason that we must have $\langle\alpha'|\,|\alpha\rangle = 0$, and quantum states that correspond to the same eigenvalue may or may not be orthogonal. Such states are said to be degenerate, particularly when the operator in question is the Hamiltonian.

Hermitian operators form a real vector space: if X and Y are Hermitian, then $aX + bY$ is Hermitian if a and b are real. It is possible to form the product of Hermitian operators XY, which means first operating with Y and then with X. But most sets of Hermitian operators are not closed under multiplication: if X and Y are Hermitian, then XY is Hermitian if and only if X and Y commute.

2 Time Evolution

A very important Hermitian operator is the Hamiltonian H, since it determines the time evolution of the system through the time-dependent Schroedinger equation:

$$i\hbar\frac{\partial}{\partial t}\,|\psi\rangle = H\,|\psi\rangle\,.$$

H also determines an important set of states, the energy eigenkets that satisfy

$$H\,|\psi_n\rangle = E_n\,|\psi_n\rangle$$

where $n = 1, ..., D$, where D is the dimension of the Hilbert space. E_n are the energies of the states $|\psi_n\rangle$ and of course the E_n are real. The set $\{|\psi_n\rangle\}_{n=1,2,...,D}$ is often used as a basis for the Hilbert space.

3 Quantum Mechanics and Computation

Textbooks will give the following description of an experiment in quantum mechanics. There are 3 steps.

1. Preparation. We construct a device that supplies a quantum system in a definite state. In the 2-step Stern-Gerlach experiment, this would be the z_+ or $|0\rangle$ state. In general this state is written as $|\psi\,(t=0)\rangle$. It is the initial condition. This function $|\psi\,(t=0)\rangle$ might coincide with one of the eigenfunctions $|\psi_n\rangle$, but it need not.

2. Evolution. The state of the particle evolves according to the time-dependent Schrödinger equation

$$i\hbar\frac{\partial}{\partial t}\,|\psi\rangle = H\,|\psi\rangle$$

and since this equation is first-order in time, it only needs one initial condition, which was given in step 1. So the evolution is deterministic.

3. Measurement. Now we wish to extract information about the system at time t, at which point it is represented by a ket $|\psi\,(t)\rangle$. We have a *choice* of measurements. This is very important. Let's say we want the state of a

particle in motion. We might wish to measure the position with a camera. We might measure the energy with a bolometer. We might measure the velocity with a time-of-flight apparatus. The results are very different.

Now consider our spin 1/2 particle. Again we have a choice. Indeed, we can measure the spin in any spatial direction. *But only ever get the result* $+\hbar/2$ *or* $-\hbar/2$. That is what the Stern-Gerlach experiment tells us. Our histogram of results ends up with only two bins no matter which direction we measure in.

We'll see how to describe this situation mathematically soon. For now, I want to point out a curious analogy between this 3-step picture of a quantum experiment and a computation. At this point, I note a curious analogy. Think about calculating on a computer. Let us say we want to find the value of $3+5$. We prepare the computer, thought of as having 2 data registers, in the state $(3,5)$. This is the input (which at the machine level is converted into bits). The $+$ is like the Hamiltonian - it determines the time evolution of the data. This time evolution is the algorithm (which at the machine level is a recipe for manipulating bits). At a later time, we extract the output - the number 8. Thus the 3 stages of a computation (input, algorithm, output) are very similar to the 3 stages of a quantum experiment (preparation, evolution, measurement). Of course this is a classical computation, so 8 is obtained with certainty: we get the same result every time on a perfect classical computer. If instead we got a range of results over many trials this would start looking a lot like a quantum experiment.

4 Using Bases

Now we can use our knowledge of vector spaces to make some calculations in quantum mechanics. The main tool that we will use is a complete basis of orthonormal kets. This will allow us to turn the abstract object $|\alpha\rangle$ into a useful set of numbers. In ordinary three-dimensional space we express vectors $\vec{r}$ in terms of the unit vectors $\widehat{x}, \widehat{y}$, and $\widehat{z}$. $\vec{r} = r_x\widehat{x} + r_y\widehat{y} + r_z\widehat{z}$: the abstract object $\vec{r}$ is turned into ("encoded in", as we will sometimes say in this course) the three numbers r_x, r_y and r_z. The analogous equation in the ket space is

$$|\beta\rangle = \sum_{\alpha=1}^{D} c_\alpha |\alpha\rangle ,$$

where D (a finite positive integer) is the dimension of the Hilbert space. How do we get the c_α? If the set $\{|\alpha_1\rangle , |\alpha_2\rangle , ..., |\alpha_D\rangle\}$ has D members and satisfies

$$\langle\alpha|\alpha'\rangle = \delta_{\alpha,\alpha'} \equiv \begin{cases} 1, \text{ if } \alpha = a' \\ 0, \text{ if } \alpha \neq \alpha' \end{cases}$$

then we have the completeness relation

$$1 = \sum_{\alpha=1}^{D} |\alpha\rangle \langle\alpha| = \sum_{\alpha=1}^{D} P_\alpha,$$

where the P_a are the projection operators mentioned before. (In these notes I will use the $\equiv$ symbol for a definition). So

$$|\beta\rangle = \left(\sum_{\alpha=1}^{D} |\alpha\rangle\langle\alpha|\right)|\beta\rangle = \sum_{\alpha=1}^{D} \langle\alpha|\beta\rangle \quad |\alpha\rangle\,,$$

and we see that the coefficients c_α are just the inner products

$$c_\alpha = \langle\alpha|\beta\rangle\,.$$

So kets can be expanded in terms of basis vectors.

What about matrices? The expression $\langle\gamma| A |\alpha\rangle$ can be expanded by inserting the unit operator in two places:

$$\langle\gamma| A |\beta\rangle = \sum_{\alpha,\alpha'=1}^{D} \langle\gamma|\alpha\rangle\langle\alpha| A |\alpha'\rangle\langle\alpha'|\beta\rangle\,,$$

and now if we look at the pattern of the indices, we see that if $\langle\alpha| A |\alpha'\rangle$ is regarded as a matrix with α labeling the rows and α' labeling the columns, then a ket becomes a column vector and a bra becomes a row vector. More explicitly, the bra $\langle\gamma|$ is represented in this basis by the row vector

$$\langle\gamma|\alpha\rangle \to (\langle\gamma|\alpha_1\rangle, \langle\gamma|\alpha_2\rangle, ..., \langle\gamma|\alpha_D\rangle)\,,$$

the operator A is represented in this basis by the square matrix

$$\langle\alpha| A |\alpha'\rangle \to \begin{pmatrix} \langle\alpha_1| A |\alpha_1\rangle & \langle\alpha_1| A |\alpha_2\rangle & \langle\alpha_1| A |\alpha_3\rangle & ... \\ \langle\alpha_2| A |\alpha_1\rangle & \langle\alpha_2| A |\alpha_2\rangle & \langle\alpha_2| A |\alpha_3\rangle & ... \\ \langle\alpha_3| A |\alpha_1\rangle & \langle\alpha_3| A |\alpha_2\rangle & ... & ... \\ ... & ... & ... & \langle\alpha_D| A |\alpha_D\rangle \end{pmatrix}$$

and the ket $|\beta\rangle$ is represented in this basis by the column vector

$$\langle\alpha'|\beta\rangle \to \begin{pmatrix} \langle\alpha_1|\beta\rangle \\ \langle\alpha_2|\beta\rangle \\ ... \\ \langle\alpha_D|\beta\rangle \end{pmatrix},$$

and the actual numerical value of the whole inner product is

$$\begin{aligned} \langle\gamma| A |\beta\rangle \quad = \quad & (\langle\gamma|\alpha_1\rangle, \langle\gamma|\alpha_2\rangle, ..., \langle\gamma|\alpha_D\rangle) \times \\ & \begin{pmatrix} \langle\alpha_1| A |\alpha_1\rangle & \langle\alpha_1| A |\alpha_2\rangle & \langle\alpha_1| A |\alpha_3\rangle & ... \\ \langle\alpha_2| A |\alpha_1\rangle & \langle\alpha_2| A |\alpha_2\rangle & \langle\alpha_2| A |\alpha_3\rangle & ... \\ \langle\alpha_3| A |\alpha_1\rangle & \langle\alpha_3| A |\alpha_2\rangle & ... & ... \\ ... & ... & ... & \langle\alpha_D| A |\alpha_D\rangle \end{pmatrix} \begin{pmatrix} \langle\alpha_1|\beta\rangle \\ \langle\alpha_2|\beta\rangle \\ ... \\ \langle\alpha_D|\beta\rangle \end{pmatrix}. \end{aligned}$$

This has the form of a multiplication of a $1 \times D$ row vector by a $D \times D$ matrix by a $D \times 1$ column vector. By the same token the matrix representing the

operator $A^\dagger$ is the complex conjugate transpose of that representing A and the operator product AB is represented by the matrix product in any basis one chooses. (You should check these statements.)

The representation of linear operators by matrices is amazingly useful, since we can use all the very powerful tools of linear algebra to help us do calculations.

4.0.1 Qubits as Vectors

A complete orthonormal basis for the two-dimensional Hilbert space is given by the kets $|0\rangle$ and $|1\rangle$, (when the degree of freedom is spin, we'll often see the equivalent notation $|\uparrow\rangle$ and $|\downarrow\rangle$, as we said before, and the two states represent a particle in the z_+ and z_- beams). But we can describe an arbitrary qubit if we do not specify the degree of freedom of the quantum system. It is simply thought of as a "2-state system" or a "2-level system". So $D = 2$ and the unit operator I is

$$I = |0\rangle\langle 0| + |1\rangle\langle 1|,$$

and any vector $|\gamma\rangle$ in the space can be written as

$$|\gamma\rangle = c_0|0\rangle + c_1|1\rangle \rightarrow \begin{pmatrix} c_0 \\ c_1 \end{pmatrix}$$

while all operators are represented by two-by-two matrices. The

$$A \rightarrow \begin{pmatrix} \langle 0|A|0\rangle & \langle 0|A|1\rangle \\ \langle 1|A|0\rangle & \langle 1|A|1\rangle \end{pmatrix} = \begin{pmatrix} A_{00} & A_{01} \\ A_{10} & A_{11} \end{pmatrix}.$$

The last form is very convenient, since it saves a lot of writing. In the future, we will most often deal directly with the matrix representations of operators and vectors, which makes the calculations much simpler.

Lecture 3
The Uncertainty Principle

1 Projective Measurement

We talked about the relationship to actual experiments in the first lecture, but the discussion of measurement was incomplete. We need a set of postulates to make it precise. I will talk about a complete projective measurement. This is the kind of measurement usually discussed in undergraduate quantum mechanics courses and it is the type of measurement we will usually use. Partial measurements are also important in QC and we will discuss them later.

Postulate 1. Every measurement corresponds to a Hermitian operator A. It then follows that for each measurement there corresponds a complete set of orthonormal eigenkets $\{|\alpha_1\rangle, |\alpha_2\rangle, ..., |\alpha_D\rangle\}$, each with a corresponding real eigenvalue:

$$A\ |\alpha_n\rangle = a_n\ |\alpha_n\rangle .$$

Postulate 2. The only possible result of a measurement of A is one of the real numbers a_n drawn from the set of eigenvalues $\{a_1, a_2, ..., a_D\}$. If the system at the time of measurement is in the normalized state $|\beta\rangle$, then we can expand $|\beta\rangle$ in the eigenfunction basis

$$|\beta\rangle = \sum_{n=1}^{D} c_n |\alpha_n\rangle$$

and the probability of measuring a_1 is $|c_1|^2$, the probability of measuring a_2 is $|c_2|^2$, and so on.

Postulate 3. If we measure the result a_n, then immediately after the measurement the state of the system is $|\alpha_n\rangle$, the eigenstate corresponding to the measurement result a_n. If the eigenvalue is degenerate, then the system must be in the subspace corresponding to α_n, but the state is not completely determined.

This last postulate goes under the name of the "collapse of the wavefunction", since the wavefunction seems to go discontinuously from the (possibly) non-eigenstate $|\beta\rangle$ to the eigenstate $|\alpha_n\rangle$.

We repeat once more: In quantum mechanics, an experiment must be repeated many times to confirm the theory. The theory does not predict the result of an individual measurement - it only predicts what the table of probabilities - the histogram of the values of a - will look like after many repetitions are done. This is unlike classical mechanics. If we have a theory to calculate the trajectory $\vec{r}(t)$ (of a rocket, say), we need only observe the rocket's path once to know if the theory is right or wrong.

As we said in the previous lecture, a useful characteristic of such a histogram is the average value of a. If the probability of observing a_n is p_n, then the average value of a, determined by many observations, is

$$\overline{a} = \sum_{n=1}^{N_T} p_n a_n,$$

where N_T is the number of trials and a_n is the value observed at the n-th trial. Since p_n is given by postulate 2 as $p_n = |c_n|^2$, we have the identity

$$\begin{aligned}
\langle\beta| A |\beta\rangle &= \sum_{m=1}^{D} c_m^* \langle\alpha_m| A \sum_{n=1}^{D} c_n |\alpha_n\rangle \\
&= \sum_{m=1}^{D}\sum_{n=1}^{D} c_m^* c_n \langle a_m| A |a_n\rangle \\
&= \sum_{m=1}^{D}\sum_{n=1}^{D} c_m^* c_n a_n \delta_{m,n} \\
&= \sum_{n=1}^{D} |c_n|^2 a_n \\
&= \sum_{n=1}^{D} p_n a_n.
\end{aligned}$$

As the number of trials N_T becomes very large, then the two expressions converge to the same limit and we have that

$$\overline{a} = \langle\beta| A |\beta\rangle .$$

$\langle\beta| A |\beta\rangle$ is called the "expectation value of A in the state $|\beta\rangle$".

1.0.1 Measurement of a Qubit

Suppose we perform an experiment in which the Stern-Gerlach machine is used as a filter: first we throw away all the $S_z = -\hbar/2$ particles, which we call $|1\rangle$. We may then verify postulate 3 by repeating the experiment on the remaining particles, confirming that they are all in the state with $S_z = \hbar/2$ which we have defined to be the $|0\rangle$ state. We call the operator which measures the z-component of angular momentum S_z and we then have the eigenvalue equations

$$\begin{aligned}
S_z\ |0\rangle &= \frac{\hbar}{2}\ |0\rangle \\
S_z\ |1\rangle &= -\frac{\hbar}{2}\ |1\rangle .
\end{aligned}$$

We now do a different experiment. Here is where the choice of measurements comes in, that choice that we stressed above. Take the $|0\rangle$ beam and pass it

through the same apparatus except that the magnet is rotated such we now measure the x-component of spin. Again we will find two possible results, and the eigenvalue equations are

$$\begin{aligned} S_x \ |+_x\rangle &= \frac{\hbar}{2} \ |+_x\rangle \\ S_x \ |-_x\rangle &= -\frac{\hbar}{2} \ |-_x\rangle . \end{aligned}$$

These results are obtained with probability 1/2, which means that

$$|+_z\rangle = c_1 \ |+_x\rangle + c_2 \ |-_x\rangle ,$$

with $|c_1|^2 = |c_2|^2 = 1/2$. If we do the same except choosing the $S_z = -\hbar/2$ beam we get the same results when we pass it through the magnet that measures S_x, so

$$|-_z\rangle = d_1 \ |+_x\rangle + d_2 \ |-_x\rangle ,$$

with $|d_1|^2 = |d_2|^2 = 1/2$. $\langle 1|\,|0\rangle = 0$ since $|0\rangle$ and $|1\rangle$ correspond to different eigenvalues of the Hermitian operator S_z, so we also have

$$0 = \langle 1|\,|0\rangle = c_1^* d_1 + c_2^* d_2 .$$

The simplest solution is then $c_1 = c_2 = 1/\sqrt{2}$ and $d_1 = -d_2 = 1/\sqrt{2}$.

We find ket-bra forms for the operators:

$$S_z = \frac{\hbar}{2} \left(|0\rangle \langle 0| - |1\rangle \langle 1| \right)$$

and

$$S_x = \frac{\hbar}{2} \left(|0\rangle \langle 1| + |1\rangle \langle 0| \right) .$$

A similar calculation, together with the commutation relations (given below) for S_x, S_y, and S_z, gives

$$\begin{aligned} S_y &= \frac{\hbar}{2} \left(|+_y\rangle \langle +_y| - |-_y\rangle \langle -_y| \right) \\ &= \frac{-i\hbar}{2} \left(|0\rangle \langle 1| - |1\rangle \langle 0| \right) , \end{aligned}$$

and in matrix form we have $S_x = \hbar\sigma_x/2, S_y = \hbar\sigma_y/2$ and $S_z = \hbar\sigma_z/2$, where σ_x, σ_y, and σ_z are the famous Pauli matrices:

$$\sigma_x = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \quad \sigma_y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}, \quad \sigma_z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} .$$

These operators have the commutation relations

$$\begin{aligned} [S_x, S_y] &= i\hbar S_z \text{ or } [\sigma_x, \sigma_y] = 2i\sigma_z \\ [S_y, S_z] &= i\hbar S_x \text{ or } [\sigma_y, \sigma_z] = 2i\sigma_x \\ [S_z, S_x] &= i\hbar S_y \text{ or } [\sigma_z, \sigma_x] = 2i\sigma_y \end{aligned}$$

and these commutation relations hold for all angular momentum operators, including the orbital angular momentum operators, spin 1, etc.

The spin 1/2 operators also satisfy the special relations

$$\vec{S}^2 = S_x^2 + S_y^2 + S_z^2 = \frac{3}{4}\hbar^2,$$

a consequence of a special property of the Pauli matrices:

$$\sigma_x^2 = \sigma_y^2 = \sigma_z^2 = 1.$$

These last two equations do not hold for $S > 1/2$.

2 Compatible and Incompatible Observables

The pair S_x, S_z is an example of two *incompatible* observables: measuring one of them destroys information about the other, in the following sense. If we prepare a particle in the eigenstate $|+_z\rangle$ and measure S_z then we get $+\hbar/2$ with certainty. If instead we first measure S_x, then we will change the state of the particle into one of the eigenstates of S_x. A subsequent measurement of S_z will give $\pm\hbar/2$ with equal probability. S_x and S_z do not commute: $[S_x, S_z] \neq 0$, which is the mathematical expression of incompatibility.

If two operators A, B do commute: $[A, B] = 0$, however, they are compatible, and measuring one does not destroy the value of the other. We may see this as follows.

Let $[A, B] = 0$ and let $|a\rangle, |a'\rangle$ be eigenkets of A. We will take the case where there is no degeneracy, so the eigenvalues corresponding to these two eigenvectors are distinct. Then

$$\begin{aligned} 0 &= \langle\alpha| [A, B] |\alpha\rangle \\ &= \langle\alpha'| (AB - BA) |\alpha\rangle \\ &= \langle\alpha'| AB |\alpha\rangle - \langle\alpha'| BA |\alpha\rangle \\ &= (a' - a) \langle\alpha'| B |\alpha\rangle , \end{aligned}$$

but $a \neq a'$, so $\langle\alpha'| B |\alpha\rangle = 0$. If we express B in the $|\alpha\rangle$ basis, this is the statement that B is diagonal. The only way that this can be true for all pairs $|\alpha\rangle, |\alpha'\rangle$ is if both are eigenfunctions of B, so that $B |\alpha\rangle = b_\alpha |\alpha\rangle$. Now we know from Postulate 3 that if we measure A, this puts the system into an eigenstate of A. However, this is also an eigenstate of B and if we now measure B we will definitely get the result b_α. For $D = 2$ there are no nontrivial examples of two such commuting operators, but in higher dimensions there are plenty.

3 The Uncertainty Principle

We can quantify this incompatibility property and the resulting inequality is called the Heisenberg Uncertainty Principle. Given two Hermitian operators A

and B, define the uncertainty of A and B in the state $|\alpha\rangle$ as

$$\langle\alpha|\left(\Delta A\right)^2|\alpha\rangle \text{ and } \langle\alpha|\left(\Delta B\right)^2|\alpha\rangle,$$

where

$$\begin{aligned}\Delta A &= A - \langle\alpha|A|\alpha\rangle = A - \overline{a},\\ \Delta B &= B - \langle\alpha|B|\alpha\rangle = B - \overline{b},\end{aligned}$$

are the deviations from the mean. ΔA and ΔB are also Hermitian. As we have already seen, $\langle\alpha|\left(\Delta A\right)^2|\alpha\rangle$ and $\langle\alpha|\left(\Delta B\right)^2|\alpha\rangle$ are the variances in the distribution we get when we measure A or B many times in the state $|\alpha\rangle$. They are therefore natural measures of the uncertainty. Another natural, and more common, measure of uncertainty is the standard deviation, obtained by taking the square root of the variance. Note that $\langle\alpha|\left(\Delta A\right)^2|\alpha\rangle \geq 0$ for any Hermitian A and $|\alpha\rangle$. (You might want to prove this by yourself. Expand $|\alpha\rangle$ in the eigenket basis of A.)

The Uncertainty Principle states that

$$\langle\alpha|\left(\Delta A\right)^2|\alpha\rangle \ \ \langle\alpha|\left(\Delta B\right)^2|\alpha\rangle \geq \frac{1}{4}\ \left|\langle\alpha|\left[A,B\right]|\alpha\rangle\right|^2.$$

for all $|\alpha\rangle$.

Proof. Because ΔA and ΔB are Hermitian, we have that $\left[\left(\Delta A\right)\left(\Delta B\right)\right]^\dagger = \left(\Delta B\right)\left(\Delta A\right)$. This implies that $\langle\alpha|\left(\Delta A\right)\left(\Delta B\right)|\alpha\rangle = \langle\alpha|\left(\Delta B\right)\left(\Delta A\right)|\alpha\rangle^*$, so if we write $\langle\alpha|\left(\Delta A\right)\left(\Delta B\right)|\alpha\rangle = x + iy$, we find

$$\begin{aligned}\langle\alpha|\left[\left(\Delta A\right)\left(\Delta B\right) + \left(\Delta B\right)\left(\Delta A\right)\right]|\alpha\rangle &= 2x\\ \langle\alpha|\left[\left(\Delta A\right)\left(\Delta B\right) - \left(\Delta B\right)\left(\Delta A\right)\right]|\alpha\rangle &= \langle\alpha|\left[\left(\Delta A\right),\left(\Delta B\right)\right]|\alpha\rangle = 2iy.\end{aligned}$$

Hence

$$\begin{aligned}\left|\langle\alpha|\left[\left(\Delta A\right)\left(\Delta B\right) + \left(\Delta B\right)\left(\Delta A\right)\right]|\alpha\rangle\right|^2 + \left|\langle\alpha|\left[\left(\Delta A\right),\left(\Delta B\right)\right]|\alpha\rangle\right|^2 &= 4\left(x^2+y^2\right) &(1)\\ &= 4\ |x+iy|^2 &(2)\\ &= 4\ \left|\langle\alpha|\left(\Delta A\right)\left(\Delta B\right)|\alpha\rangle\right|^2 &(3)\end{aligned}$$

The Schwartz inequality is

$$\langle\beta||\beta\rangle \ \ \langle\gamma||\gamma\rangle \geq \left|\langle\beta||\gamma\rangle\right|^2$$

which is a purely mathematical theorem. It's easy to remember because in a real vector space it is the statement that $\cos^2\theta \leq 1$ where θ is the angle between $|\beta\rangle$ and $|\gamma\rangle$. We apply it to the states $|\beta\rangle = \Delta A|\alpha\rangle$ and $|\gamma\rangle = \Delta B|\alpha\rangle$ and find

$$\langle\alpha|\left(\Delta A\right)^2|\alpha\rangle \ \ \langle\alpha|\left(\Delta B\right)^2|\alpha\rangle \geq \left|\langle\alpha|\left(\Delta A\right)\left(\Delta B\right)|\alpha\rangle\right|^2. \qquad (4)$$

Putting Eqs. 4 and 1 together we have

$$\begin{aligned}\langle\alpha|\left(\Delta A\right)^2|\alpha\rangle\ \ \langle\alpha|\left(\Delta B\right)^2|\alpha\rangle &\geq \frac{1}{4}\ \left|\langle\alpha|\left[\Delta A,\Delta B\right]|\alpha\rangle\right|^2+\frac{1}{4}\left|\langle\alpha|\left[\left(\Delta A\right)\left(\Delta B\right)+\left(\Delta B\right)\left(\Delta A\right)\right]|\alpha\rangle\right|^2\\ &\geq \frac{1}{4}\ \left|\langle\alpha|\left[\Delta A,\Delta B\right]|\alpha\rangle\right|^2\\ &= \frac{1}{4}\ \left|\langle\alpha|\left[A,B\right]|\alpha\rangle\right|^2,\end{aligned}$$

where the last step follows because $\Delta A = A-\overline{a}$, $\Delta B = B-\overline{b}$, and the constants $\overline{a}$ and $\overline{b}$ commute with all operators. The square root of this equation is

$$\sqrt{\langle\alpha|\left(\Delta A\right)^2|\alpha\rangle}\ \sqrt{\langle\alpha|\left(\Delta B\right)^2|\alpha\rangle}\geq\frac{1}{2}\ \left|\langle\alpha|\left[A,B\right]|\alpha\rangle\right|, \tag{5}$$

which is more closely related to typical examples of the Uncertainty Principle given in elementary texts. In this form it states that the product of the standard deviation in the histograms for repeated measurements of A and B in the state $|\alpha\rangle$ exceeds $1/2$ the magnitude of expectation value of the commutator of A and B in the state $|\alpha\rangle$.

There is no useful converse to the Uncertainty Principle. If A and B commute, then there will be states for which the product of the variances vanish, but there may be others for which it does not.

The experimental interpretation of the Uncertainty Principle needs to be carefully discussed. The Principle holds for a different experiment than the one described above in which S_z and S_x are measured in sequence. Rather, it refers to an experiment in which the state $|\alpha\rangle$ is prepared repeatedly and then either A or B is measured. For example, we could prepare the state $|\alpha\rangle = |+_z\rangle$ by passing an unpolarized beam through a Stern-Gerlach magnet in the z-direction and discarding the $S_z = -\hbar/2$ states. (Notice, by the way, that measurement is also a good means of preparation!). On this beam we could measure $A = S_x$ many times, then repeat the experiment except that we measure $B = S_y$ many times. Then when we multiply the widths of the two distributions, we would find that they satisfy

$$\begin{aligned}\langle 0|\left(\Delta S_x\right)^2|0\rangle\ \ \langle 0|\left(\Delta S_y\right)^2|0\rangle &\geq \frac{1}{4}\ \left|\langle 0|\left[S_x,S_y\right]|0\rangle\right|^2\\ &= \frac{1}{4}\ \left|\langle 0|\, i\hbar S_z\,|0\rangle\right|^2\\ &= \frac{1}{4}\left(\frac{\hbar^2}{2}\right)^2\\ &= \frac{\hbar^4}{16}.\end{aligned}$$

You should compute the left-hand side of this equation directly from what we already know and confirm the correctness of this inequality. To show another aspect of the Uncertainty Principle, prepare the state $|\alpha\rangle = |+_x\rangle$ by rotating the

magnets to the x-direction and discard the $S_x = -\hbar/2$ outgoing beam. Then we get

$$\langle +_x | \left(\Delta S_x\right)^2 | +_x \rangle \;\; \langle +_x | \left(\Delta S_y\right)^2 | +_x \rangle \geq \frac{\hbar^2}{4} \;\; \left| \langle +_x | S_z | +_x \rangle \right|^2 ,$$

but $\langle +_x | S_z | +_x \rangle = 0$. So the inequality does not tell you anything nontrivial. $\langle +_x | \left(\Delta S_x\right)^2 | +_x \rangle = 0$ for this state, since it is an eignenstate of S_x, and we get the same result every time, and the width of the histogram is zero.

In the case of a sequential measurement of A, then B, the system is in an eigenstate of A (call it $|a\rangle$) when B is measured. In this state $\langle a | \left(\Delta A\right)^2 | a \rangle = 0$, so the inequality is not very helpful.

You may have seen the Uncertainty Principle in the special form

$$\Delta x \; \Delta p \geq \frac{1}{2}\hbar,$$

which seems to follows from Eq. 5 since $[x, p] = i\hbar$. However, it contains no reference to the state $|\alpha\rangle$. This difference is due to the fact that the $\Delta x \; \Delta p$ form of the Uncertainty Principle applies in an infinite-dimensional Hilbert space (in which there is no state-independent upper bound on standard deviations), while our derivation was for a finite-dimensional space.

Lectures on Quantum Computing Physics 709

Lecture 4: Bases and Time Evolution

1 Changing Bases

We have seen that to do actual computations in quantum mechanics, we need the explicit matrix representations of operators, and for this one needs to choose a basis. An orthonormal basis is almost always the most convenient. Often we need to change the basis, which is similar to rotating Cartesian coordinates in ordinary mechanics.

As in rotations in classical mechanics, there are two ways of viewing the change of basis: a passive change, where the physical system is left alone but the numbers that describe it are changed, and the active change where the object is rotated but the coordinate axes remain fixed, so again the numbers change. They are equivalent mathematically.

Let $\left|\alpha^{(n)}\right\rangle$ and $\left|\beta^{(n)}\right\rangle$ be two orthonormal bases:

$$\begin{aligned}\left\langle\alpha^{(m)}|\alpha^{(n)}\right\rangle &= \delta_{m,n}\\ \left\langle\beta^{(m)}|\beta^{(n)}\right\rangle &= \delta_{m,n}.\end{aligned}$$

Now we know that any vector can be expanded in the $\left|\alpha^{(n)}\right\rangle$ basis, so let us do this for the vectors $\left|\beta^{(m)}\right\rangle$. This defines a matrix U that connects the two:

$$\left|\beta^{(m)}\right\rangle = \sum_n U_{mn}\left|\alpha^{(n)}\right\rangle.$$

As an operator, then U is given by

$$U = \sum_n \left|\beta^{(n)}\right\rangle\left\langle\alpha^{(n)}\right|$$

Theorem. $UU^\dagger = I$, which we express by saying that U is unitary. (I is the identity operator: $I\ |\alpha\rangle = |\alpha\rangle$ for all $|\alpha\rangle$.)

Proof:

$$UU^\dagger = \sum_n \left|\beta^{(n)}\right\rangle\left\langle\alpha^{(n)}\right| \sum_m \left|\alpha^{(m)}\right\rangle\left\langle\beta^{(m)}\right| = \sum_{n,m}\left|\beta^{(n)}\right\rangle\ \delta_{m,n}\ \left\langle\beta^{(m)}\right| = \sum_m \left|\beta^{(m)}\right\rangle\ \left\langle\beta^{(m)}\right| = I.$$

Hence $U^\dagger = U^{-1}$. U is analogous to a rotation in real space. A unitary matrix is the complex analog of an orthogonal matrix in real linear algebra. The most

important respect in which it resembles a rotation is that it preserves the norms of a vector. The norm of the vector $U|\alpha\rangle$ is

$$\langle\alpha| U^\dagger U |\alpha\rangle = \langle\alpha| U^{-1} U |\alpha\rangle = \langle\alpha|\alpha\rangle .$$

This property implies immediately that the eigenvalues of a unitary operator are of the form $e^{i\theta}$ - they are pure phases (you should prove this).

In the $|\alpha\rangle$ basis, the matrix representation of U is

$$U = \begin{pmatrix} \langle\alpha^{(1)}|\beta^{(1)}\rangle & \langle\alpha^{(1)}|\beta^{(2)}\rangle & \langle\alpha^{(1)}|\beta^{(3)}\rangle & ... \\ \langle\alpha^{(2)}|\beta^{(1)}\rangle & \langle\alpha^{(2)}|\beta^{(2)}\rangle & ... & ... \\ \langle\alpha^{(3)}|\beta^{(1)}\rangle & ... & ... & ... \\ ... & ... & ... & ...\langle\alpha^{(D)}|\beta^{(D)}\rangle \end{pmatrix}.$$

If we write a ket $|a\rangle$ as a column vector in the $|\alpha\rangle$ basis we get:

$$|a\rangle = \begin{pmatrix} \langle\alpha^{(1)}|a\rangle \\ \langle\alpha^{(2)}|a\rangle \\ ... \\ \langle\alpha^{(D)}|a\rangle \end{pmatrix},$$

whereas in the $|\beta\rangle$ basis

$$|a\rangle = \begin{pmatrix} \langle\beta^{(1)}|a\rangle \\ \langle\beta^{(2)}|a\rangle \\ ... \\ \langle\beta^{(D)}|a\rangle \end{pmatrix},$$

but

$$\langle\alpha^{(1)}|a\rangle = \langle\alpha^{(1)}| \sum_{n=1}^{D} \langle\beta^{(n)}|a\rangle \ |\beta^{(n)}\rangle$$

and, more generally

$$\begin{pmatrix} \langle\alpha^{(1)}|a\rangle \\ \langle\alpha^{(2)}|a\rangle \\ ... \\ \langle\alpha^{(D)}|a\rangle \end{pmatrix} = \begin{pmatrix} \langle\alpha^{(1)}|\beta^{(1)}\rangle & \langle\alpha^{(1)}|\beta^{(2)}\rangle & \langle\alpha^{(1)}|\beta^{(3)}\rangle & ... \\ \langle\alpha^{(2)}|\beta^{(1)}\rangle & \langle\alpha^{(2)}|\beta^{(2)}\rangle & ... & ... \\ \langle\alpha^{(3)}|\beta^{(1)}\rangle & ... & ... & ... \\ ... & ... & ... & ...\langle\alpha^{(D)}|\beta^{(D)}\rangle \end{pmatrix} \begin{pmatrix} \langle\beta^{(1)}|a\rangle \\ \langle\beta^{(2)}|a\rangle \\ ... \\ \langle\beta^{(D)}|a\rangle \end{pmatrix}.$$

So we see that a ket expressed as a vector in the $|\alpha\rangle$ (old) basis is equal to U (in the $|\alpha\rangle$ basis) times the vector of the ket expressed in the $|\beta\rangle$ (new) basis and a ket expressed as a vector in the $|\beta\rangle$ (new) basis is equal to $U^{-1} = U^\dagger$ (in the $|\alpha\rangle$ basis) times the vector of the ket expressed in the $|\alpha\rangle$ (old) basis. Often this

kind of equation is written as $|\alpha'\rangle = U|\alpha\rangle$. Thus U is the matrix that allows us to go back and forth between bases.

The transformation of operators between bases is not much more complicated. We want objects with a direct physical interpretation as the result of a measurement like $\langle\alpha|X|\beta\rangle$ to be invariant. If we transform this to the new basis we would have $\langle\alpha'|X'|\beta'\rangle$. These must be the same so

$$\begin{aligned}\langle\alpha|X|\beta\rangle &= \langle\alpha'|X'|\beta'\rangle \\ &= \langle\alpha|UX'U^{-1}|\beta\rangle,\end{aligned}$$

so we must have $X = UX'U^{-1}$ or $X' = U^{-1}XU$ as the transformation rule.

An easy way to see whether a matrix is unitary is to think about its columns as a collection of vectors. Each vector must be normalized and must be orthogonal to all the others.

2 Time evolution

Now we wish to ask how the state of a quantum system changes when the particles in it are subjected to forces. Let the state at time $t = t_0$ be $|\alpha(t_0\rangle$ and at some later time it is $|\alpha(t_1\rangle$. It satisfies the Schrödinger equation:

$$i\hbar\frac{\partial}{\partial t}|\alpha(t)\rangle = H|\alpha(t)\rangle, \tag{1}$$

with the initial condition $|\alpha(t = t_0)\rangle = |\alpha(t_0)\rangle$. Here H is the Hamiltonian, which is Hermitian and may depend on time. The initial state and the final state are connected by an operator called the evolution operator $U(t_1, t_0)$:

$$|\alpha(t_1\rangle = U(t_1, t_0)|\alpha(t_0\rangle.$$

U must be unitary because it cannot change the norm of the state, which must always be equal to one - recall that the sum of the probabilities for any measurement is equal to the norm of the vector, so the unitarity of U is sometimes called the conservation of probability. So U must satisfy

$$U^\dagger(t_1, t_0) = U(t_1, t_0)$$

for all t_1 and t_0. It also satisfies the composition rule

$$U(t_2, t_0) = U(t_2, t_1)U(t_1, t_0)$$

and the inversion rule

$$U(t_0, t_1) = U^{-1}(t_1, t_0),$$

both of which follow from the definition of U.

We can also write a Schrödinger equation for U without reference to any particular quantum state:

$$i\hbar\frac{\partial}{\partial t}U(t, t_0) = HU(t, t_0),$$

where H is the Hamiltonian. We can then multiply this on the right by $|\alpha(t_0\rangle$, and we obtain

$$i\hbar\frac{\partial}{\partial t}U(t,t_0)\left|\alpha(t_0)\right\rangle = HU(t,t_0)\left|\alpha(t_0)\right\rangle,$$

which is the same as

$$i\hbar\frac{\partial}{\partial t}\left|\alpha(t)\right\rangle = H\left|\alpha(t)\right\rangle,$$

and when the ket is a wavefunction, this is the form of the original Schrödinger equation that we first wrote down.

Solving the Schrödinger equation presents 3 levels of difficulty, depending on the complexity of the time dependence of the Hamiltonian H.

1. When H is time independent, then we can write a formal solution for U :

$$U(t_1,t_0) = \exp\left[-iH\left(t_1-t_0\right)/\hbar\right].$$

The exponential of a matrix can be given a rigorous meaning by using the power series expansion for the exponential.

2. If the Hamiltonian depends on time, but the H operators at different times commute: $[H(t),H(t')]=0$ for all t,t', then

$$U(t_1,t_0) = \exp\left[-\frac{i}{\hbar}\int_{t_0}^{t_1}H\left(t\right)dt\right].$$

This case is important in some quantum computing implementations. The most common way for it to happen is when the Hamiltonian takes the form

$$H(t) = g(t)H_0,$$

where $g(t)$ is a number and H_0 is independent of time. We then get

$$U(t_1,t_0) = \exp\left[-\frac{i}{\hbar}H_0\int_{t_0}^{t_1}g\left(t\right)dt\right]$$

3. If $[H(t),H(t')]\neq 0$ for some $t\neq t'$ then the problem becomes more complicated, and we will not go into it here. Google "time-ordered exponentials" if you are interested.

Lectures on Quantum Computing
Physics 709
Lecture 5: Multiple Qubits

1 Multiple Qubits and Tensor Products

If we are dealing with two or more particles, then we need to generalize the Dirac formalism slightly, but in a fairly obvious way. We will now specialize to the case of a set of spin 1/2 particles, or any set of particles, as long as the total system acts like a set of two-level systems. Since this is how quantum computers are made, this will be a very important example for us. From now on I will write qubit rather than particle. In the QC world, the $\{|0\rangle, |1\rangle\}$ basis for qubits is called the 'computational basis'.

Let us now suppose that we have two qubits. Call them A and B. A possible state of such a 2-particle system is given by a tensor product of two kets:

$$|\Psi_p\rangle^{(2)} = |\alpha\rangle_A \otimes |\beta\rangle_B ,$$

which just says that particle A is in the state $|\alpha\rangle$ and particle B is in the state $|\beta\rangle$. A tensor product of two kets gives rise to a new ket in a higher dimensional space. If the original kets are in spaces of dimensions D_1 and D_2, then the new ket is in a space of dimension $D_1 \times D_2$. For the tensor product of two qubits, $D_1 D_2 = 4$ and the tensor product represents a ket in a space of dimension 4. The tensor multiplication symbol acts like an ordinary product in that it distributes over addition of kets just like ordinary multiplication distributes over ordinary addition:

$$|\alpha\rangle_A \otimes [|\alpha\rangle_B + |\beta\rangle_B] = |\alpha\rangle_A \otimes |\alpha\rangle_B + |\alpha\rangle_A \otimes |\beta\rangle_B .$$

$\otimes$ is the tensor product symbol, and it is usually written explicitly in the math literature. However, the $\otimes$ symbol is often omitted as being redundant in the physics literature and I will follow the physics convention, unless it produces ambiguity. This product can of course be generalized to an N-particle system:

$$|\Psi\rangle^{(N)} = |\alpha_1\rangle \ \ |\alpha_2\rangle \cdots |\alpha_N\rangle_N$$

meaning that particle 1 is in the $|\alpha_1\rangle$ state, particle 2 is in the $|\alpha_2\rangle$ state, *etc.*

Now here is the key point. The dimension of the Hilbert space of the N-qubit system is 2^N, not $2N$. The general wavefunction for a a collection of N qubits is the linear combination

$$|\Psi\rangle^{(N)} = \sum_{\alpha_1=0}^{1} \sum_{\alpha_2=0}^{1} \cdots \sum_{\alpha_N=0}^{1} c_{\alpha_1,\alpha_2,\ldots\alpha_N} \ \ |\alpha_1\rangle \ \ |\alpha_2\rangle \cdots |\alpha_N\rangle$$

where the coefficients satisfy

$$\sum_{\alpha_1=0}^{1}\sum_{\alpha_2=0}^{1}\cdots\sum_{\alpha_N=0}^{1}|c_{\alpha_1,\alpha_2,\ldots\alpha_N}|^2=1$$

and there are 2^N of them, since that is how many ways there are to choose $\alpha_1 = 0$ or 1, $\alpha_2 = 0$ or 1, etc.

As an aside to those who have studied many-particle physics, you are no doubt aware that if the particles are indistinguishable, then these coefficients need to satisfy

$$c_{P(\alpha_1,\alpha_2,\ldots\alpha_N)}=(\pm 1)^{S_P}\,c_{\alpha_1,\alpha_2,\ldots\alpha_N},$$

where $P(\alpha_1,\alpha_2,\ldots\alpha_N)$ is any permutation of $\alpha_1,\alpha_2,\ldots\alpha_N$ and S_P is the sign of P, i.e., +1 if P is even and -1 if P is odd. Bosons need the "+1" and fermions need the "-1". However, in QCs, the qubits are spatially separated and therefore distinguishable, so there is no restriction on the $c_{\alpha_1,\alpha_2,\ldots\alpha_N}$. So there are 4 basis states and therefore 4 coefficients in a 2-qubit system, 8 in a 3-qubit system, and so on. Similarly, operators in an N-qubit system are represented by $2^N\times 2^N$ matrices.

2 Partial Measurement

Once we begin to consider composite systems, then the question of measurement becomes more complicated. We can do a "partial measurement": measure just one (or a few) of the qubits. This must be possible since the qubits can be spatially separated. What is then the state of the other qubits? We need an additional postulate to govern this situation.

Here it is.

We perform a measurement of the operator σ_z on the first qubit. σ_z is a Hermitian operator, it has a complete set of 2 eigenstates $|0\rangle$ and $|1\rangle$ that satisfy $\sigma_z|0\rangle = |0\rangle$ and $\sigma_z|1\rangle = |1\rangle$. This is called measuring in the computational basis since the eigenstates of σ_z *are* the computational basis. When we measure σ_z on qubit 1 we obtain one of two possible results. Call this result b_1 and the eigenstate $|b_1\rangle$. The the state of the whole system immediately after the measurement is

$$|\Psi\rangle^{(N)}=C\sum_{\alpha_2=0}^{1}\cdots\sum_{\alpha_N=0}^{1}a_{b_1,\alpha_2,\ldots\alpha_N}\;\;|b_1\rangle\;\;|\alpha_2\rangle\cdots|\alpha_N\rangle_N\,,$$

so qubit 1 is projected onto an eigenstate of σ_z. (There is of course nothing special about qubit 1. The formulas are just harder to write and uglier for qubit i.) This projection can have dramatic effects, since half of the coefficients have been erased: the sum now has only 2^{N-1} terms. $C \geq 1$ is a normalization constant.

Is there a physical significance to whether or not the state is just a simple product rather than a linear combination? Yes, there is, and it has to do with inter-qubit correlations and partial measurements.

First recall the definition of correlated random variables a and b in probability theory. If $p(a,b) = p(a) \times p(b)$, then a and b are not correlated.

In quntum mechanics the question is a little bit subtle because of the possibility of change of basis. Take the simple (one term in the sum) state

$$|\Psi\rangle^{(2)} = |0\rangle_A \ \ |0\rangle_B \tag{1}$$

For this state it is clear that the 2 qubits are not really correlated - measurement of B in the $\{|0\rangle_B, |1\rangle_B\}$ basis will always give $|0\rangle_B$ and leave qubit A in the state $|0\rangle_A$. $p(0,0) = 1 = p(0) \times p(0)$ with $p(a=0) = p(b=0) = 1$. The same statement holds if we measure A first. The same lack of correlation is present in

$$\begin{aligned} |\Psi\rangle^{(2)} &= \frac{1}{\sqrt{2}} |0\rangle \ \ |0\rangle + \frac{1}{\sqrt{2}} |1\rangle \ \ |0\rangle \\ &= \frac{1}{\sqrt{2}} (|0\rangle + |1\rangle) \ \ |0\rangle \end{aligned}$$

since, though a measurement of A in the $\{|0\rangle_A, |1\rangle_A\}$ basis will give $|0\rangle_A$ only half the time, the result has no correlation to the result of a measurement on B. This example makes the important point that correlation is not affected by basis changes that are done separately on each qubit, but such "local unitary operations" can change a state from a simple product state into one which is not a simple product, if we are dealing with a fixed basis. Clearly, any state of the product form:

$$|\Psi\rangle^{(2)} = |\psi\rangle \left|\psi'\right\rangle \tag{2}$$

has this uncorrelated character.

3 Density matrix

What if we do not know precisely the state of a quantum system, perhaps because our preparation method is imperfect? Let's say that the probability of the state $|\alpha\rangle$ is p_α where, as usual $0 \leq p_\alpha \leq 1$ and $\sum p_\alpha = 1$. p_α represents *classical* uncertainty. Now we measure the Hermitian operator A, which satisfies $A|\alpha_n\rangle = a_n \ |\alpha_n\rangle$. Here $\{|\alpha_n\rangle\}_{n=1,2,\ldots,D}$ is a complete orthonormal basis. Then the probability of measuring a_n is p_n and the average value of a is

$$\overline{a} = \sum_n p_n a_n. \tag{3}$$

What if we measure the Hermitian operator B, satisfying $B|\beta_n\rangle = b_n \ |\beta_n\rangle$, so $B = \sum_n b_n \ |\beta_n\rangle \langle\beta_n|$? Recalling that the quantum-mechanical probability for

obtaining the results b_m when the system is in the state $|\alpha_n\rangle$ is $|\langle\alpha_n|\beta_m\rangle|^2$ we find

$$\begin{aligned}\overline{b} &= \sum_{n,m} p_n b_m \left|\langle\alpha_n|\beta_m\rangle\right|^2 \\ &= \sum_{n,m} p_n b_m \ \langle\alpha_n|\beta_m\rangle \ \langle\beta_m|\alpha_n\rangle \,.\end{aligned}$$

This leads us to define the density matrix, also sometimes called the density operator

$$\rho = \sum_n p_n \ |\alpha_n\rangle \ \langle\alpha_n| \,. \tag{4}$$

and we see that in the α-basis ρ is diagonal:

$$\rho = \begin{pmatrix} p_1 & 0 & ... & 0 \\ 0 & p_2 & ... & 0 \\ ... & ... & ... & 0 \\ 0 & 0 & 0 & p_D \end{pmatrix}$$

and we can write

$$\overline{a} = Tr\left(\rho A\right). \tag{5}$$

where the trace of an operator C is defined in analogy with their matrix representation (here in the basis $\{|\alpha_n\rangle\}_{n=1,...,D}$ as

$$\begin{aligned} Tr\left(C\right) &= Tr \sum_{mn} |\alpha_m\rangle \, c_{mn} \, \langle\alpha_n| \\ &= \sum_n c_{nn}. \end{aligned}$$

We also have

$$\overline{b} = \sum_{n,m} p_n b_m \, \langle\beta_m|\alpha_n\rangle \ \langle\alpha_n|\beta_m\rangle \tag{6}$$

$$= \sum_m b_m \, \langle\beta_m| \left[\sum_n p_n \, |\alpha_n\rangle \ \langle\alpha_n|\right] |\beta_m\rangle \tag{7}$$

$$= Tr\left(\rho B\right). \tag{8}$$

So there is a general rule that the average of a measurement of A in a state given by a density matrix ρ is the trace of the product of ρ and A : $\overline{A}$ =Tr(ρA) .

Note that the trace is invariant under unitary transformations, since

$$Tr\left(C'\right) = Tr\left(UCU^{-1}\right) = Tr\left(CUU^{-1}\right) = Tr\left(C\right) \tag{9}$$

because of the invariance of the trace under cycling of matrix products.

ρ has three important properties: (1) ρ is Hermitian; (2) $\mathrm{Tr}(\rho) = 1$; (3) ρ is a positive operator $\langle\alpha|\rho|\alpha\rangle \geq 0$ for all $|\alpha\rangle$.

Considering ρ as an operator, we get an equation of motion for ρ:

$$\begin{aligned} i\hbar\frac{d\rho}{dt} &= \sum_n p_n \left(i\hbar\frac{d}{dt}|\alpha_n\rangle\right) \langle\alpha_n| + \sum_n p_n |\alpha_n\rangle \left(i\hbar\frac{d}{dt}\langle\alpha_n|\right) \\ &= \sum_n (H|\alpha_n\rangle) a_n \langle\alpha_n| - \sum_n |\alpha_n\rangle a_n (\langle\alpha_n| H) \qquad (10) \\ &= [H,\rho]. \end{aligned}$$

In this derivation I used the result that

$$\frac{d}{dt}\langle\alpha_n| = \frac{i}{\hbar}\langle\alpha_n| H.$$

This comes from the fact that the magnitude of $|\alpha_n\rangle$ is time-independent:

$$\begin{aligned} 0 &= \frac{d}{dt}1 \\ &= \frac{d}{dt}\langle\alpha_n|\alpha_n\rangle \\ &= \left(\frac{d}{dt}\langle\alpha_n|\right)|\alpha_n\rangle + \langle\alpha_n|\left(\frac{d}{dt}|\alpha_n\rangle\right) \\ &= \left(\frac{d}{dt}\langle\alpha_n|\right)|\alpha_n\rangle - \frac{i}{\hbar}\langle\alpha_n| H|\alpha_n\rangle \end{aligned}$$

and for this to be true for all kets requires

$$\frac{d}{dt}\langle\alpha_n| = \frac{i}{\hbar}\langle\alpha_n| H.$$

This evolution of ρ is unitary because the differential equation $i\hbar\, d\rho/dt = [H,\rho]$ corresponds to $\rho(t) = U\rho U^{-1}$ where U is the same evolution operator as for the kets.

If H is time-independent, then we can use the simple form denoted by 1. above for U and we have

$$\rho(t) = e^{-iHt/\hbar}\rho(0)\, e^{iHt/\hbar}.$$

Lectures on Quantum Computing
Physics 709
Lecture 6: Entanglement and its Uses

1 Entangled states

The state

$$|\Psi\rangle_{AB} = \frac{1}{\sqrt{2}} |0\rangle_A |0\rangle_B + \frac{1}{\sqrt{2}} |1\rangle_A |1\rangle_B \tag{1}$$

behaves differently from a product state. If we have $|\Psi\rangle_{AB}$ and measure qubit B and find that it is in the $|0\rangle_B$ state, then a future measurement of qubit A gives $|0\rangle_A$ with certainty, while if we measure B and find that it is in the $|1\rangle_B$ state, then a future measurement of A gives $|1\rangle_A$ with certainty. No change of basis will reduce this state to a simple product state. There is no basis in which I can define a $p(a)$ and $p(b)$ such that $p(a,b) = p(a)\,p(b)$. This non-product property is not necessarily evident from inspection. If a state cannot be written as a product state in *any* basis it is called an entangled state.

Can we come up with a simple characterization of entanglement? Let us look at the problem as a geometrical problem in Hilbert space. A completely general 2-qubit state is

$$|\Psi\rangle_{AB} = c_{00} |0\rangle_A |0\rangle_B + c_{01} |0\rangle_A |1\rangle_B + c_{10} |1\rangle_A |0\rangle_B + c_{11} |1\rangle_A |1\rangle_B \tag{2}$$

with

$$|c_{00}|^2 + |c_{01}|^2 + |c_{10}|^2 + |c_{11}|^2 = 1. \tag{3}$$

Considered in the 8-dimensional space of the real and imaginary parts of the c_{ij}, this is the unit 7-sphere. If the state is unentangled (combed?), then it can be written as a product state and it satisfies

$$c_{00} |0\rangle_A |0\rangle_B + c_{01} |0\rangle_A |1\rangle_B + c_{10} |1\rangle_A |0\rangle_B + c_{11} |1\rangle_A |1\rangle_B = (a_0 |0\rangle_A + a_1 |1\rangle_A)(b_0 |0\rangle_B + b_1 |1\rangle_B) \tag{4}$$

so that

$$\begin{aligned} c_{00} &= a_0 b_0 \\ c_{01} &= a_0 b_1 \\ c_{10} &= a_1 b_0 \\ c_{11} &= a_1 b_1. \end{aligned}$$

Dividing, we get

$$c_{00}/c_{01} = b_0/b_1 = c_{10}/c_{11} \tag{5}$$

and so

$$c_{00}c_{11} = c_{01}c_{10}. \tag{6}$$

This *nonlinear* constraint is what we need to have an unentangled state. The set of all entangled 2-qubit states form a complicated manifold in the 8-dimensional space, not a linear subspace. The fact that the constraint is nonlinear is what makes entanglement difficult to deal with. Normally, the only subsets of Hilbert space that we talk about in quantum theory are linear subspaces that are themselves vector spaces. Thus the formalism of quantum theory is often said to be entirely linear.

This is not true for the concept of entanglement. Neither the set of unentangled states, nor the set of entangled states, form a vector space. This has meant in practice that there are several measures of entanglement, each with its own advantages and disadvantages. We will talk about these measures later in the course.

2 QC Notation

We have said that we will change to the notation used in the QC community. We have already started this process by using

$$\begin{aligned} |+_z\rangle &= |0\rangle \\ |-_z\rangle &= |1\rangle . \end{aligned}$$

This has two advantages: $|+_z\rangle$ gives the impression that we are dealing with spin, but many two-level systems can be used for QC that are not spin systems; also, we want the particles to represent qubits, and the states of a classical bit are always written as 0 and 1. As column vectors we have

$$\begin{aligned} |0\rangle &= \begin{pmatrix} 1 \\ 0 \end{pmatrix} \\ |1\rangle &= \begin{pmatrix} 0 \\ 1 \end{pmatrix} \end{aligned}$$

in the computational basis. We will go a bit farther in the notational capitulation to the QC community and write the Pauli matrices with capital letters: $X = \sigma_x, Y = \sigma_y, Z = \sigma_z$. This does save a subscript. We then get

$$\begin{aligned} X|0\rangle &= |1\rangle \\ X|1\rangle &= |0\rangle \\ Y|0\rangle &= i|1\rangle \\ Y|1\rangle &= -i|0\rangle \\ Z|0\rangle &= |0\rangle \\ Z|1\rangle &= -|1\rangle . \end{aligned}$$

(You should write out these equations in the computational basis as well.)

These equations have a geometric interpretation. If we think of the states as 3-vectors lying on a unit sphere, with $|0\rangle$ at the North Pole $(0,0,1)$, $|1\rangle$ at the South Pole $(0,0,-1)$, $(|0\rangle + |1\rangle)/\sqrt{2}$ and $(|0\rangle - |1\rangle)/\sqrt{2}$ as antiparallel vectors with their end points on the equator $(1,0,0)$ and $(-1,0,0)$, then X is a rotation by an angle $\pi = 180^\circ$ about the x-axis, Y is a rotation by an angle $\pi = 180^\circ$ about the y-axis, and Z is a rotation by an angle $\pi = 180^\circ$ about the z-axis. This is called the Bloch sphere.

It is also conventional to define the unitary Hadamard matrix

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

which gives

$$\begin{aligned} H|0\rangle &= \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \\ H|1\rangle &= \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) \end{aligned}$$

and

$$\begin{aligned} H\left[\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)\right] &= |0\rangle \\ H\left[\frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)\right] &= |1\rangle . \end{aligned}$$

Actually X, Y, Z and H are all both unitary *and* Hermitian, (which is a little unusual). We change language and say these operations are "1-qubit gates" (rather than "single-particle unitary operators"): they change the state of a single qubit. They are quantum logic gates. For example, X is like a NOT gate on a bit since it changes TRUE (0) to FALSE (1) and FALSE (1) to TRUE (0). Y acts like a NOT gate followed by a change in phase (that is, a multiplication by a complex number of unit magnitude), Z only changes the phase of $|1\rangle$. So Y and Z do not have obvious classical analogs, since the phase of a state has no significance in a classical computer.

If we have 2 qubits then there are 4 basis states. We will go a little further with QC notation and write the tensor product in an even more abbreviated form as

$$|\alpha\rangle \otimes |\beta\rangle = |\alpha\rangle \ \ |\beta\rangle = |\alpha\beta\rangle .$$

Operators in a 2-qubit system are represented by 4×4 matrices and are called 2-qubit gates. The most important one is the cNOT gate, which is called C and acts as

$$\begin{aligned} C|00\rangle &= |00\rangle \\ C|01\rangle &= |01\rangle \\ C|10\rangle &= |11\rangle \\ C|11\rangle &= |10\rangle . \end{aligned}$$

C is unitary. It flips qubit B (the target qubit) if qubit A (the control qubit) is in the 1 state - otherwise qubit B is left alone. The control qubit is always left unchanged by C. cNOT is short for "controlled NOT", since the state of qubit B is controlled by that of qubit A. Once the operator is defined in the computational basis it is defined by linearity for all states. This is even more obvious if we write it as a matrix:

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

in the $|00\rangle, |01\rangle, |10\rangle, |11\rangle$ basis. (Notice that when we specify a basis to define a matrix representation of an operator we need to specify the order of the states in the basis.) We can check that this is a unitary operator by noticing that its columns are orthonormal. Unitarity is an essential component of any quantum logic gate.

N-qubit operators are represented by $2^N \times 2^N$ matrices, but we will deal only with 1- and 2-qubit operators.

3 Uses of Entanglement

Everything we discussed up until this point was well known in the 1930s. Now we are starting to talk about more recent discoveries. We will look at two simple uses of entanglement, simple because you don't need many bits. These two experiments have been carried out at some level.

3.0.1 Quantum Dense Coding

This is a communication scheme. In classical communication, we send a string of 0s and 1s down a line. We can do the same thing with quantum states - just send a stream of particles in states $|0\rangle$ and $|1\rangle$. But there is a way to do better, in a certain sense. The idea is that if Alice and Bob share entangled pairs of qubits, then this allows Alice to send information faster to Bob - twice as fast, to be precise. So it would be useful if Alice expects that at some point she will need to say something in a hurry to Bob, but she doesn't know what it is yet.

We'll now show how to send two bits of information with the transfer of only one physical qubit. Alice prepares the entangled state

$$|\Psi\rangle = \frac{1}{\sqrt{2}} \left(|00\rangle + |11\rangle\right),$$

keeps the first qubit and sends to second one to Bob. Then the time comes that Alice wants to send me 2 bits of classical information: the messages 00,01,10, or 11. (I'm Bob.) She does this by first transforming her qubit according to the message she wants to send: if she wants to send 00,01,10,or 11, then she applies

I, X, Z, or ZX, to the qubit she has, which is the A qubit. Then the shared state becomes one of these 4:

$$\begin{aligned}
00 \quad &: \quad I_A \ |\Psi\rangle = \frac{1}{\sqrt{2}} \left(|00\rangle + |11\rangle\right) \\
01 \quad &: \quad X_A \ |\Psi\rangle = \frac{1}{\sqrt{2}} \left(|10\rangle + |01\rangle\right) \\
10 \quad &: \quad Z_A \ |\Psi\rangle = \frac{1}{\sqrt{2}} \left(|00\rangle - |11\rangle\right) \\
11 \quad &: \quad Z_A X_A \ |\Psi\rangle = \frac{1}{\sqrt{2}} \left(-|10\rangle + |01\rangle\right).
\end{aligned}$$

Then she sends the first (A) qubit to me. Please notice that she only has to send me ONE qubit. I've got them both now and I do a cNOT with the one I just received with A as control qubit and B as the target qubit. To be specific call this operation C_{AB} and find

$$\begin{aligned}
00 \quad &: \quad C_{AB} I \ |\Psi\rangle = \frac{1}{\sqrt{2}} \left(|00\rangle + |10\rangle\right) = \frac{1}{\sqrt{2}} \left(|0\rangle + |1\rangle\right) |0\rangle \\
01 \quad &: \quad C_{AB} X \ |\Psi\rangle = \frac{1}{\sqrt{2}} \left(|11\rangle + |01\rangle\right) = \frac{1}{\sqrt{2}} \left(|1\rangle + |0\rangle\right) |1\rangle \\
10 \quad &: \quad C_{AB} Z \ |\Psi\rangle = \frac{1}{\sqrt{2}} \left(|00\rangle - |10\rangle\right) = \frac{1}{\sqrt{2}} \left(|0\rangle - |1\rangle\right) |0\rangle \\
11 \quad &: \quad C_{AB} ZX \ |\Psi\rangle = \frac{1}{\sqrt{2}} \left(-|11\rangle + |01\rangle\right) = \frac{1}{\sqrt{2}} \left(-|1\rangle + |0\rangle\right) |1\rangle .
\end{aligned}$$

Finally, a Hadamard on the one I was sent gives me

$$\begin{aligned}
00 \quad &: \quad H_A CI \ |\Psi\rangle = \frac{1}{\sqrt{2}} H_A \left(|0\rangle + |1\rangle\right) |0\rangle = |00\rangle \\
01 \quad &: \quad H_A CX \ |\Psi\rangle = \frac{1}{\sqrt{2}} H_A \left(|1\rangle + |0\rangle\right) |1\rangle = |01\rangle \\
10 \quad &: \quad H_A CZ \ |\Psi\rangle = \frac{1}{\sqrt{2}} H_A \left(|0\rangle - |1\rangle\right) |0\rangle = |10\rangle \\
11 \quad &: \quad H_A CZX \ |\Psi\rangle = \frac{1}{\sqrt{2}} H_A \left(-|1\rangle + |0\rangle\right) |1\rangle = -|11\rangle .
\end{aligned}$$

Measuring both qubits in the computational basis now tells me what Alice wanted me to know. It's important for you to go through this computation line by line, since we will be doing many similar things in this course.

This scheme is only useful if on-site processing is cheap and qubit information transfer between sites at a high rate is expensive or otherwise undesirable. I get one extra bit of information for each pair. It's not extremely dramatic. Still, it seems to suggest that quantum effects can really improve information processing in specific circumstances.

The entangled pair is destroyed at the end. Entanglement was created in order to make the process possible and that entanglement is then used up. We can think of entanglement as a resource for communication.

3.0.2 Teleportation

This denotes the process of communicating an arbitrary and in fact unknown quantum state

$$|\psi_U\rangle = \alpha\,|0\rangle_U + \beta\,|1\rangle_U$$

from Alice to Bob through a channel that opens briefly. Perhaps the two get together at time $t = 0$ and can share qubits, but later they are separated and have no quantum channel. But we assume that they have a channel through which they can communicate classical information, like a telephone. Neither Alice nor Bob knows what the state $|\psi_U\rangle$ is, meaning they have no knowledge of the values of α and β. Alice does have another pair of qubits A and B at the start.

Alice starts with the total state $|\psi_1\rangle$

$$|\psi_1\rangle = \frac{1}{\sqrt{2}}\left(\alpha\,|0\rangle_U + \beta\,|1\rangle_U\right)\left(|0\rangle_A\,|0\rangle_B + |1\rangle_A\,|1\rangle_B\right),$$

a product of the unknown state and an entangled state that she prepares, so she has a 3-qubit state. Now she gives me qubit B, so we share an entangled pair

$$\frac{1}{\sqrt{2}}\left(|0\rangle_A\,|0\rangle_B + |1\rangle_A\,|1\rangle_B\right).$$

I now leave with my qubit. The question is whether Alice can somehow tell me how to put my qubit in the state $|\psi_U\rangle$.

Now Alice applies a cNOT with the U qubit as the control and her A qubit as the target. This gate is C_{UA}. Then the total state is $|\psi_2\rangle$:

$$\begin{aligned}
|\psi_2\rangle &= C_{UA}\,|\psi_1\rangle \\
&= \frac{1}{\sqrt{2}}\alpha\,|0\rangle_U\left(|0\rangle_A\,|0\rangle_B + |1\rangle_A\,|1\rangle_B\right) \\
&\quad + \frac{1}{\sqrt{2}}\beta\,|1\rangle_U\left(|1\rangle_A\,|0\rangle_B + |0\rangle_A\,|1\rangle_B\right).
\end{aligned}$$

Then she applies a Hadamard to the U qubit, obtaining $|\psi_3\rangle$:

$$\begin{aligned}
|\psi_3\rangle &= H_U |\psi_1\rangle \\
&= \frac{1}{2}\alpha \left(|0\rangle_U + |1\rangle_U\right)\left(|0\rangle_A |0\rangle_B + |1\rangle_A |1\rangle_B\right) \\
&\quad + \frac{1}{2}\beta \left(|0\rangle_U - |1\rangle_U\right)\left(|1\rangle_A |0\rangle_B + |0\rangle_A |1\rangle_B\right) \\
&= \frac{1}{2}|0\rangle_U |0\rangle_A \left(\alpha |0\rangle_B + \beta |1\rangle_B\right) + \\
&\quad \frac{1}{2}|0\rangle_U |1\rangle_A \left(\alpha |1\rangle_B + \beta |0\rangle_B\right) + \\
&\quad \frac{1}{2}|1\rangle_U |0\rangle_A \left(\alpha |0\rangle_B - \beta |1\rangle_B\right) + \\
&\quad \frac{1}{2}|1\rangle_U |1\rangle_A \left(\alpha |1\rangle_B - \beta |0\rangle_B\right).
\end{aligned}$$

Alice now measures qubits U and A, obtaining the results 00,01,10, or 11 with equal probability 1/4. If she gets 00, she phones me and says "Bob, you do not need to do anything". She has successfully sent me the state $|\psi_U\rangle = \frac{1}{\sqrt{2}}\left(\alpha |0\rangle_B + \beta |1\rangle_B\right)$ and we are done. If she measures 01, then I have $\frac{1}{\sqrt{2}}\left(\alpha |1\rangle_B + \beta |0\rangle_B\right)$, so she tells me to apply an X gate for which

$$X\frac{1}{\sqrt{2}}\left(\alpha |1\rangle_B + \beta |0\rangle_B\right) = \frac{1}{\sqrt{2}}\left(\alpha |0\rangle_B + \beta |1\rangle_B\right) = |\psi_U\rangle .$$

If she gets 10, she tells me to apply a Z gate, and if she gets 11 she tells me to apply a ZX gate; in all cases I end up with $|\psi_U\rangle$.

This protocol is useful if we do not have a channel in which arbitrary qubit states like $|\psi_U\rangle$ could be sent directly - it is possible to imagine channels like this. I do not know why this is called teleportation. It seems a silly name (and I am as big a Star Trek fan as anyone.)

In both these protocols, some interesting task is achieved which requires that we have an entangled pair to start with. At the end of the game, the entanglement is completely gone - we are in fact left with product states when all is said and done. Again, this gives rise to the idea of thinking of entanglement as a resource that is consumed during some kinds of information processsing.

Lectures on Quantum Computing
Physics 709
Lecture 7: Bell Inequalities

1 Correlations

We have seen how to use entanglement to do some simple but still surprising information process tasks. But its significance is much more fundamental. Entanglement is one of the ways (maybe the most important way!) in which the real world that is described by quantum mechanics is different from other worlds that could exist and that Einstein thought *did* exist. In the end, entanglement was the aspect of quantum mechanics that most bothered Einstein. The famous paper of Einstein, Podolsky, and Rosen in the 30s first clearly pointed out what was paradoxical about entanglement. In the 60s Bell showed that the kinds of physics theories (local realistic theories) that Einstein felt had to be correct necessarily implied certain inequalities between measurements on correlated systems. He also showed that entangled states would violate those inequalities if quantum mechanics was correct. Later, experiments verified that nature does indeed violate Bell's inequalities.

In classical physics we often cannot measure everything about a system. For example, we cannot measure the positions of all the air molecules in a lecture room. This may lead to uncertainty in some of the things we *can* measure: the measurement of the pressure in a small subvolume of a lecture hall will not give the same result every time. So there is nothing strange about the fact that physical systems are probabilistic *in practice*. But we might argue, and Einstein did, that even though we cannot measure everything about a system, nevertheless quantities that are in principle measurable still do have some definite values.

Some of the most interesting differences between quantum and classical physics come about when we consider composite systems, defined as a system consisting of two or more subsystems on which separate measurements can be made. A probabilistic composite system is characterized by a joint probability distribution $p_{12}(x_1, x_2)$, where the two subsystems are 1 and 2. System 1 can be in any of the states x_1. The various possible x_1 are defined as any possible states that can be distinguished by measurements on system 1. The same definition is made for x_2, the states of system 2. $0 \leq p_{12}(x_1, x_2) \leq 1$ and $\sum_{x_1,x_2} p_{12}(x_1, x_2) = 1$: these equation are ture for all probabilities. If the two systems are uncorrelated, then there exist functions $p_1(x_1)$ and $p_2(x_2)$ such that p_{12} has a product form: $p_{12}(x_1, x_2) = p_1(x_1)\ p_2(x_2)$, where $p_1(x_1)$ and $p_2(x_2)$ are separate probability distributions for the separate systems; $0 \leq p_1(x_1) \leq 1$ and $\sum_{x_1} p_1(x_1) = 1$ and $0 \leq p_2(x_2) \leq 1$ and $\sum_{x_2} p_2(x_2) = 1$.

At first sight this classical framework is very analogous to the two-particle quantum description with a general 2-particle ket having the form

$$|\Psi\rangle^{(2)} = \sum_{x_1,x_2} a_{12} |x_1\rangle \otimes |x_2\rangle$$

with probabilities $\sum_{x_1,x_2} |a_{12}|^2 = 1$. This wavefunction is uncorrelated when there exist wavefunctions $|\psi\rangle_1$ and $|\psi\rangle_2$ such that

$$|\Psi\rangle^{(2)} = |\psi\rangle_1 \otimes |\psi\rangle_2 = \left(\sum_{x_1} a_1(x_1) |x_1\rangle\right) \otimes \left(\sum_{x_2} a_2(x_2) |x_2\rangle\right)$$

with $\sum_{x_1} |a_1|^2 = 1$ and $\sum_{x_2} |a_2|^2 = 1$.

The analogy is very close in the case of uncorrelated systems because measurements on product wavefunctions are uncorrelated: we can write $p_1 = |a_1|^2$ and $p_2 = |a_2|^2$ and $p_{12} = |a_1|^2 |a_2|^2$. But when correlations are present the two formalisms can produce very different results, and the differences are very non-intuitive: entanglement has implications that are weird. The best way to see this weirdness is to look at Bell inequalities. There are many Bell inequalities. I will do a version due to Clauser, Horne, Shimony, and Holt called the CHSH inequality.

2 Experiment

Let's take a 2-qubit system, and imagine for definiteness that they are two electron spins. We can choose to measure the spins of the qubits along any axes in space, but let's focus on two directions for each, making four possible experiments we can do. Do the experiments at different locations but simultaneously (or with timelike separation), so they don't influence one another. Measure the spin component of particle 1 along $\widehat{n}_1$ or $\widehat{n}_2$ with results $n_1, n_2 = \pm 1$ and the spin component of particle 2 along $\widehat{m}_1$ or $\widehat{m}_2$ also with results ± 1. Notice that we take as input for the Bell argument the observed fact that there are only two output beams in a Stern-Gerlach experiment for *all* angles. Once we have limited ouselves to these directions there are 4 possible measurement schemes $(\widehat{n}_1, \widehat{m}_1), (\widehat{n}_1, \widehat{m}_2), (\widehat{n}_2, \widehat{m}_1), (\widehat{n}_2, \widehat{m}_2)$, and for each scheme there are 4 possible outcomes $(\pm 1, \pm 1)$. We will actually do a measurement and then take the product of the results, obtaining such things as $n_1 m_1 = \pm 1$.

3 Classical Result

We now make two assumptions:

1. There exists a total classical probability distribution prior to the measurement, which we call $p(n_1, n_2, m_1, m_2)$. which can include correlations between the two systems, since the two systems were at one time causally connected. This function exists if Einstein's view is correct: that if we choose to measure n_1 and do not choose

to measure n_2 nevertheless n_2 has some value. In quantum mechanics, there is no reason to believe in the existence of such a function. $p(n_1, n_2, m_1, m_2)$ is a probability function so it is a real positive function specified by giving 16 real numbers each satisfying $0 \leq p(n_1, n_2, m_1, m_2) \leq 1$ and $\sum_{n_1=\pm1, n_2=\pm1, m_1=\pm1, m_2=\pm1} p(n_1, n_2, m_1, m_2) = 1$. Repetitions of the experiment will give average values computable by the usual formulas of classical probability theory. The system is in one of the 16 states in each repetition. This assumption that p exists is called "Realism". We need probability for the usual reason in classical statistical physics: we don't completely know what that state is.

2. Measuring system 1 does not affect system 2, so $p(n_1, n_2, m_1, m_2)$ is fixed at the time the two systems become causally disconnected and does not change thereafter. This assumption is called "Locality".

We measure $(n_1 m_1)_{av}$ by setting up the Stern-Gerlach magnets for electron 1 in the $\widehat{n}_1$ position for electron 2 in the $\widehat{m}_1$ position and repeat many times. Rearrange the magnets so as to also get the same for the other three choices: $(n_2 m_1)_{av}, (n_2 m_2)_{av}$ and $(n_1 m_2)_{av}$.

Now we look at the quantity R,

$$R = n_1 m_1 + n_2 m_1 + n_2 m_2 - n_1 m_2 = (n_1 + n_2) m_1 + (n_2 - n_1) m_2 = \pm 2.$$

The last equality comes from the fact that if $n_1 + n_2 = \pm 2$ then $(n_2 - n_1) = 0$ and vice versa. The average value of R is

$$\begin{aligned} R_{av} &= \sum p(n_1, n_2, m_1, m_2) R, \text{ so} \\ |R_{av}| &\leq \sum p(n_1, n_2, m_1, m_2) |R| \\ &= 2 \sum p(n_1, n_2, m_1, m_2) \\ &= 2 \end{aligned}$$

but averaging is linear, so we also have

$$R_{av} = (n_1 m_1)_{av} + (n_2 m_1)_{av} + (n_2 m_2)_{av} - (n_1 m_2)_{av} \leq 2.$$

This is a Bell inequality - it is important to note that any Bell inequality is really a statement about what we would think of as *classical* systems, since the hypotheses of realism and locality come from our classical intuition.

4 Quantum Result

Now we're going to do the experiment in the actual quantum world. Prepare the state

$$|\Psi_s\rangle = \frac{1}{\sqrt{2}} (|0\rangle |1\rangle - |1\rangle |0\rangle),$$

which is an entangled state. That is not so strange, but here comes the tricky part. We choose

$$\begin{aligned}
\widehat{n}_1 &= \widehat{z}, \\
n_2 &= \widehat{x}, \\
\widehat{m}_1 &= \left(-\widehat{z}-\widehat{x}\right)/\sqrt{2}, \text{ and} \\
\widehat{m}_2 &= \left(\widehat{z}-\widehat{x}\right)/\sqrt{2}.
\end{aligned}$$

This specifies a set of four experiments. (This choice is somewhat mysterious at first sight. That is one reason why the Bell inequalities were not easy to discover.)

We now do our first experiment. We have 2 particles in the state $|\Psi_s\rangle$. Separate particles 1 and 2. Set up two Stern-Gerlach magnets in the orientations that give $\widehat{n}_1 = \widehat{z}$ and $\widehat{m}_1 = \left(-\widehat{z}-\widehat{x}\right)/\sqrt{2}$. Send particle number 1 through apparatus $\widehat{n}_1$ and particle 2 through apparatus $\widehat{m}_1$. Record the results n_1 and m_1, and their product $n_1 m_1$. Repeat many times and average. We know what we will get since the state $|\Psi_s\rangle$ has been given. One finds

$$\begin{aligned}
\left(n_1 m_1\right)_{av} &= \frac{-1}{2\sqrt{2}}\left(\langle 0|\langle 1| - \langle 1|\langle 0|\right) Z_1 \left(Z_2 + X_2\right)\left(|0\rangle|1\rangle - |1\rangle|0\rangle\right) \\
&= \frac{-1}{2\sqrt{2}}\left(\langle 0|\langle 1| - \langle 1|\langle 0|\right)\left(Z_2 + X_2\right)\left(|0\rangle|1\rangle + |1\rangle|0\rangle\right) \\
&= \frac{-1}{2\sqrt{2}}\left(\langle 0|\langle 1| - \langle 1|\langle 0|\right)\left(-|0\rangle|1\rangle + |1\rangle|0\rangle + |0\rangle|0\rangle + |1\rangle|1\rangle\right) \\
&= \frac{-1}{2\sqrt{2}}\left(-2\right) \\
&= \frac{1}{\sqrt{2}}.
\end{aligned}$$

Now do the other three experiments in the same way, changing only the orientation of the magnets. We find

$$\left(n_2 m_1\right)_{av} = \frac{1}{2\sqrt{2}}, \;\; \left(n_2 m_2\right)_{av} = \frac{1}{\sqrt{2}}, \;\; \left(n_1 m_2\right)_{av} = -\frac{1}{\sqrt{2}},$$

so that

$$\left(n_1 m_1\right)_{av} + \left(n_2 m_1\right)_{av} + \left(n_2 m_2\right)_{av} - \left(n_1 m_2\right)_{av} = 2\sqrt{2} > 2.$$

Quantum mechanics violates the Bell inequality. No classical system that satisfies assumptions 1 and 2 can do that. It can be shown that $2\sqrt{2}$ is the maximum value that can be achieved, $R_{av} \leq 2\sqrt{2}$ for any choice of wavefunction, a result due to Tsirelson.

It is customary to see in the literature that entanglement is a form of correlation that is beyond classical correlation. This is true, but the additional effect is normally one of ANTI-correlation. If we compute the correlation function that is the average value of $X_1X_2 + Y_1Y_2 + Z_1Z_2$ in the entangled state $|\Psi_s\rangle$, we find

$$\langle\Psi_s| \;\; \left(X_1X_2 + Y_1Y_2 + Z_1Z_2\right) \;\; |\Psi_s\rangle = -3,$$

which is the *minimum* value over all 2-particle states. $|\Psi_s\rangle$ is called the "singlet" state in representation theory and it is essentially unique in having this much anti-correlation. The *maximum* for the expectation value is obtained by choosing any 2-particle correlation function is obtained by choosing $|\Psi_t\rangle = |0\rangle\,|0\rangle$ and we find

$$\langle\Psi_t|\ \ (X_1X_2 + Y_1Y_2 + Z_1Z_2)\ \ |\Psi_t\rangle = 1,$$

and the state is not unique: $\langle 1|\,\langle 1|\ \ (X_1X_2 + Y_1Y_2 + Z_1Z_2)\ \ |1\rangle\,|1\rangle = 1$. $|\Psi_t\rangle$ is of course not an entangled state. The +1 result looks very classical, but the -3 not at all. This shows that minus signs are very important when we come to discuss the difference between classical correlations and quantum correlations. In the CHSH inequality there are minus signs both in the quantity R that is involved in the inequality and in the choice of measurement directions. This is true of all Bell inequalities.

Lectures on Quantum Computing Physics 709

Lecture 8: Classical Computing

1 Classical Computers

Let's start with some terminology. A classical bit is a physical device that stores a 0 or a 1. This may be a capacitor, magnetic domain, etc. A set of bits that stores some convenient unit of information, such as a number or a word, is called a register. A register of N bits can store 2^N different numbers. Conversely, the number of bits required to store a number x is $\log_2 x$. An algorithm is a function takes a given input register or set of input registers to an output register.

Let us consider a computer that can do a calculation such as $2 + 3 = 5$. It must have two input registers that can store numbers from 0 to 3 and an output register that can store numbers from 0 to 7. This means 2 2-bit registers and 1 3-bit register. The computation $2 + 3 = 5$, now in binary notation, is

$$10 + 11 = 101. \tag{1}$$

The addition function is a mapping from a set of four input bits to a set of three output bits:

$$f_{add} : \{\{0,1\}_1^{in}, \{0,1\}_2^{in}, \{0,1\}_3^{in}, \{0,1\}_4^{in}\} \rightarrow \{\{0,1\}_1^{out}, \{0,1\}_2^{out}, \{0,1\}_3^{out}\} \tag{2}$$

from the input registers to the output register. The input space contains $2^4 = 16$ discrete points and the output space contains $2^3 = 8$ discrete points. An algorithm (such as addition) is a function whose argument can take on 16 possible values and which is then equal to one of 8 possible values. This means there are a finite number of possible algorithms on this computer. This is the number of ways of associating 2^3 outputs with each of 2^4 inputs, which is $2^{4+3} = 2^7 = 128$. If we define "algorithm" in this way, then the number of algorithms is is finite on any machine.

Clearly, no matter how big the input and output, we can repeat the above argument and calculate the number of possible algorithms. The finiteness of the possibilities gives us hope that we might be able to construct every algorithm by putting some very simple standard algorithms together in a finite chain. This turns out to be possible. The very simple standard algorithms are called gates. Some of the easiest to understand are the OR and AND gates, which take 2 bits into 1:

$$z = f_{OR}(x, y) = x + y - xy \pmod 2, \tag{3}$$

$$z = f_{AND}(x, y) = xy \pmod 2. \tag{4}$$

The nomenclature comes from the idea of associating 0 with a false statement and 1 with a true statement. Then the OR gate says that z is true if either x or y is true, while the AND gate says that z is true if both x and y are true. Another simple and important gate is the COPY gate, which takes 1 bit into 2 bits.

$$f_{COPY}(x) = (x, x). \tag{5}$$

The collection of these 3 gates is a universal gate set. This means that any of the 128 algorithms on our microcomputer can be constructed by a sequence of such gates acting on a set of bits. The basic gates act on 2 bits i and j, so they can be written as $f^{ij}_{OR}(x, y)$, $f^{ij}_{AND}(x, y)$, and $f^{ij}_{COPY}(x)$, or, generically, as f^{ij}_a for $a = 1, 2, 3$. A chain of gates acting on the input registers has the form $f^{i_T j_T}_{a_T}\left(f^{i_{T-1} j_{T-1}}_{a_{T-1}}\left(f^{i_{T-2} j_{T-2}}_{a_T}(...)\right)\right)$. Now we see that we can estimate the time taken for a calculation. If a single gate takes a time τ, and there are T gates in the sequence, then than the total time for the computation is $T\tau$. $1/\tau$ is the "clock speed" of the computer.

We could also write the state of the computer as a big column vector and then each gate can be formulated as a matrix. Then the sequence that constitutes the algorithm becomes just a product of T matrices. This is very closely analogous to the operation of a quantum computer, as we shall see.

This set of 3 gates is sufficient to construct any finite algorithm on a classical computer. Mathematically, this statement can be reformulated by saying that any algorithm is a huge matrix and it can be written as a product of 3 simpler matrix types.

Of course, as you are aware, not all algorithms are as straightforward as addition. The input might affect the application of the f's, one might measure a register and use the information so obtained to modify the subsequent operations, and so on. All these complications go under the heading of "models of computation". But the fundamental idea of doing sequences of very simple operations underlies all computations.

2 Hardness of Problems

I'm going to give a very simplified discussion of this very important topic, omitting models of computation, deterministic and non-deterministic Turing machines, and so on.

We define the size of a problem N as the number of bits needed to specify the input completely. Adding 2 numbers, each of magnitude n, needs $N = 2\log_2 n$ bits for the input. The "time" it takes to solve the problem is the number of gate operations T. The time required for an algorithm to turn the input into the output for a given problem defines some function $T(N)$. The hardness of a problem is defined by the growth of the function $T(N)$ when the fastest algorithm is chosen. Usually we do not try to determine T in a completely explicit way. Instead we try to determine its asymptotic behavior as $N \to \infty$. The big O (for "is the order of") notation is common: $T(N) = O(N^x)$ if there

are constants C and N_0 such $T(N) \leq CN^x$ for some x, C and all $N > N_0$. In this case we say that the problem has a "polynomial-time algorithm". The other very important case is when $T(N) = O\left(x^N\right)$, that is there are constants x, C and N_0 such that $T(N) \leq Cx^N$ for some x, C and all $N > N_0$ In this case we say that the problem has an "exponential-time algorithm". Polynomial-time algorithms are sometimes called "efficient" algorithms, while exponential-time algorithms are called "inefficient". This is because exponential functions grow so much faster then polynomial ones as N increases. Theoretical computer scientists don't care too much about C because, even though it is important in figuring how much actual time it takes to solve a problem, C is dependent on the actual computer used. From a practical standpoint, however, reducing C might be important. The same holds for x, except even more so.

The jargon of complexity theory is a little weird, for historical reasons. If the fastest algorithm for a problem is polynomial, then we say that the problem is in P. If a given solution to a problem can be checked in polynomial time, then it is in NP. It is believed that $P \neq NP$, i.e., that there are problems whose solutions can be checked efficiently but that cannot be solved efficiently. An NP-complete problem is one in NP whose efficient solution would guarantee the efficient solution of all problems in NP. (This is shown by means of an appropriate efficient mapping from one problem to another.) NP-intermediate problems are in NP but are neither NP-complete nor in P. Factoring is believed to be in this class. NP-intermediate is known to non-empty if $P \neq NP$. Finally, NP-hard problems are those that are as least as hard as any NP-complete problem. NP-hard problems need not be in NP.

Turning from jargon to what is actually known for sure, it has never been proved that $P \neq NP$. Also, factoring has never been proved to be NP-intermediate - in fact no problem has been proved to be NP-intermediate.

We can clearly make similar classes for quantum computers by replacing bits by qubits and gate operations by quantum gate operations, and hardness of classical algorithms by hardness of quantum algorithms. This is not quite the end of the story and we'll come back to it later.

Lectures on Quantum Computing Physics 709

Lecture 9: Quantum Computers and the Deutsch Algorithm

1 Hardware for Quantum Computers

The basic element of a quantum computer is the quantum bit, or qubit, a physical device that can be prepared in any linear combination of two quantum states $|0\rangle$ and $|1\rangle$. These states might be the up and down states of an electron or spin-1/2 nucleus, two states of an atom, *etc.* We may write the wavefunction of the qubit as

$$\psi = c_0|0\rangle + c_1|1\rangle, \tag{1}$$

where

$$|c_0|^2 + |c_1^2| = 1. \tag{2}$$

A qubit stores far more information than a classical bit can. It stores the ratio $|c_0|/|c_1|$, which is an arbitrary nonnegative real number, and the relative phase ϕ, defined by $\exp(i\phi) = (c_0/c_1)/(|c_0|/|c_1|)$. Furthermore, if there are N qubits, a quantum register, then we can store 2^N complex numbers:

$$\begin{aligned} |\Psi\rangle &= c_0|0\rangle_1|0\rangle_2\cdots|0\rangle_N + c_1|0\rangle_1|0\rangle_2\cdots|1\rangle_N \\ &\quad + \cdots + a_{2^N-1}|1\rangle_1|1\rangle_2\cdots|1\rangle_N \\ &= \sum_0^{2^N-1} c_X|X\rangle. \end{aligned} \tag{3}$$

X is an n-digit binary number. The last line shows that we can write the state of a quantum computer in a rather compact way.

The c_X are subject only to the normalization constraint

$$\sum_0^{2^N-1} |c_X|^2 = 1.$$

There is also an overall phase that has no physical significance: $|\Psi\rangle$ and $\exp(i\theta)\,|\Psi\rangle$ represent the same physical state.

But, how much of the information stored in the 2^N complex numbers c_X is really available to us? If we measure $|\Psi\rangle$, we do NOT learn the values of each of the c_X. Instead will find a definite state $|Y\rangle$ with probability $|c_Y|^2$. We can only get a handle on the values of $|c_Y|$ for different Y by doing repeated

measurements, as we have seen. The register contains a lot of information, but the measurement seems to destroy most of it. This is what makes the construction of quantum algorithms very difficult. It also means that when we use the word "store", as above, we use it in a different sense than when we use it for a classical computer, since the *availability* of the "stored" information is not the same.

Before we continue, let us reiterate that there is a one-to-one correspondence between the 2^N basis states

$$|X\rangle = |x_1\rangle_1 |x_2\rangle_2 \cdots |x_N\rangle_N$$

where the x_i equal 0 or 1 and the 2^N numbers $X = x_1 x_2 ... x_N$, where the x_i are the digits of X in binary notation. (This is usually called the computational basis). This correspondence is used repeatedly in the whole subject of quantum computing: it allows us to think of the state of a quantum computer with N qubits as a linear superposition of the numbers from 0 to $2^N - 1$.

2 Quantum Logic Gates

A quantum algorithm is a unitary transformation U on a set of qubits:

$$U|X\rangle_{input} = |X\rangle_{output}. \tag{4}$$

For a single qubit this looks like

$$U|x\rangle_{input} = |x\rangle_{output}, \tag{5}$$

which written out in matrix form is

$$\begin{pmatrix} u_{00} & u_{01} \\ u_{10} & u_{11} \end{pmatrix} \begin{pmatrix} c_0 \\ c_1 \end{pmatrix} = \begin{pmatrix} u_{00}c_0 + u_{01}c_1 \\ u_{10}c_0 + u_{11}c_1. \end{pmatrix} \tag{6}$$

The unitary condition: $UU^\dagger = 1$ can be written out as

$$\sum_j u_{ij} u_{kj} = \delta_{ik}. \tag{7}$$

This is a 1-qubit gate, which we write as U_1, the rotation of qubit 1. If $|X\rangle$ has N qubits, then its total dimension is 2^N and U_1 is a $2^N \times 2^N$ matrix. One interesting difference between a quantum computer and a classical computer is that the dimension of the input register and the output register is always the same. This is a characteristic of a unitary transformation. A classical program normally generates bits that are used and then thrown away. This does not occur in a quantum program. This also implies that a quantum program is reversible. Thermodynamics tells us that we can make a reversible process free from dissipation (in principle!).

The question of how to construct a universal gate set for a quantum computer has a very simple answer. Only the one-qubit gate described above and one type of 2-qubit gate are enough, the controlled-R gate. This has the action:

$$R_{12}|0\rangle_1|y\rangle_2 = |0\rangle_1|y\rangle_2, \tag{8}$$

$$R_{12}|1\rangle_1|y\rangle_2 = |1\rangle_1 R_2|y\rangle_2. \tag{9}$$

Here R_{12} is a unitary 4×4 matrix and R_2 is a fixed unitary 2×2 matrix. It rotates the second bit (the target bit), or not, depending on whether the first bit (the control bit) is 1 or 0. R_2 can be almost any 2×2 unitary matrix. Let us take it to be

$$R_2 = \begin{pmatrix} \cos\theta & \sin\theta \\ \sin\theta & -\cos\theta \end{pmatrix},$$

where θ is a fixed angle that is not a rational multiple of π.

In order to get a little more insight, let's calculate the matrix for R_{12} in the computational basis $\{|00\rangle, |01\rangle, |10\rangle, |11\rangle\}$.

$$R_{12} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \cos\theta & \sin\theta \\ 0 & 0 & \sin\theta & -\cos\theta \end{pmatrix}.$$

R_2 can be anything except the unit matrix and any matrix satisfying $R_2^n = 1$ for some integer n.

A very important special case is the CNOT, or controlled-NOT gate

$$C_{12} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

which flips the second (target) qubit if the first (control) qubit is in the 1 state, and leaves number 2 alone if number 1 is in the 0 state. This is obtained by taking $\theta = \pi/2$, so it is not in the universal set mentioned above, but is it a very common gate in quantum algorithms.

If our computer can perform arbitrary 1-qubit rotations U_i on the i-th qubit for all i from 1 to N and the controlled-R operation R_{ij} for all pairs i, j for $i \neq j$ from 1 to N, then an arbitrary $2^N \times 2^N$ unitary matrix can be written as a product

$$U = U_{i_1} R_{i_2 i_3} U_{i_4} \cdots$$

for some ordering of the U's and R's and some choice of the indices.

Most courses in quantum computing spend considerable time on quantum logic gates, but we will proceed to other things. I also stress that this discussion of U and R gates is rather abstract. I mainly used them as an example of a universal set. There are many such universal sets. Furthermore, in a real quantum computer, other gates would more likely be used, and which gates are the best depends on the physics of the particular computer.

3 Quantum Algorithms

How does a quantum algorithm work? For the purposes of this course, we want the quantum computer to solve classical problems. This means that we are given a set of input numbers and we wish to obtain the answer to the problem in terms of an output number or numbers. So we need to be able to translate classical numbers to quantum states. We already know one way to do this. The input $X = x_n x_{n-1} x_{n-2} ... x_0$, an n-digit binary number, is translated into the n-qubit state $|X\rangle = |x_n\rangle \otimes |x_{n-1}\rangle \otimes |x_{n-2}\rangle \otimes \cdots \otimes |x_0\rangle$. Here $x_i = 0$ or 1 and $|x_0\rangle$ is the corresponding state in the computational basis. We may also wish to add some other quantum states as working space for the algorithm. As output we want a number $Z = z_m z_{m-1} z_{m-2} ... z_0$, an m-digit binary number. So we need to get the quantum state $|Z\rangle = |z_m\rangle \otimes |z_{m-1}\rangle \otimes |z_{m-2}\rangle \otimes \cdots \otimes |z_0\rangle$, perhaps along with some other qubits that we do not need to measure. For a decision problem with a "yes" or "no" answer, we could have $m = 1$. The whole algorithm can be written as

$$|Z\rangle \otimes |B\rangle = U\left(|X\rangle \otimes |A\rangle\right)$$

where $|B\rangle$ contains the qubits that we do not need at the end, and $|A\rangle$ contains the "workspace" qubits that do not encode input. The dimension of $|Z\rangle \otimes |B\rangle$ must be the same as that of $|X\rangle \otimes |A\rangle$ since U is a square matrix. U is a sequence of T logic gates and the algorithm time is $T\tau$, where again $1/\tau$ is the clock rate.

Again one must stress that this is the simplest picture. For example, U itself may contain input information (this is actually the more common case.)

We not only want the quantum computer to work, we want it to solve problems that a classical computer cannot. Our review of classical complexity theory tells us that this question has a precise meaning: are there problems that have no efficient classical algorithm but do have an efficient quantum algorithm?

The hope for quantum computation is to use quantum parallelism. A classical parallel computer performs computations on many inputs simultaneously on different processors, then combines the results at the end. This does not reduce the number of operations performed relative to a purely sequential computation, but it cuts down the time required for a given task. Now a quantum computer, unlike a classical computer, can be in a superposition of states. So we could prepare the input register in a superposition of different inputs. Then the unitary transformation, the quantum algorithm, acts on all these states at once. Unlike the parallel classical computer, however, it does so on a single processor. Then at the end we combine all these results to get the answer. That is the idea. However, there are serious obstacles. So we will look first at a problem for which quantum computation gives no advantage even though we can apply a quantum parallel algorithm.

3.1 Problem One (Not the Deutsch Algorithm)

Let us suppose that we have a very complicated function f that takes many steps to compute classically. The function solves a decision problem. A decision problem is a problem with a yes or no answer, a $|0\rangle$ or a $|1\rangle$. This is a 1-bit output. Decision problems are very important in computer science because nearly any problem can be reduced to a sequence of decision problems. Let us also suppose that the input is extremely simple, also one bit. So f has a 1-bit input and a 1-bit output. That is all we know about the function but we have a subroutine that computes the function. Writing out the input and output of the subroutine: $f(0) = 0$ or 1, and $f(1) = 0$ or 1. The first problem (which is NOT the Deutsch problem) is to completely determine the function. There are only 4 possibilities. The obvious way to answer this question is to call the subroutine twice: once for each of the 2 possible inputs. So the total time required is essentially twice the time that it takes to compute f. Clearly we cannot do better using a classical computer.

What about the quantum computer? We need two qubits to do a unitary computation; the input is

$$|X\rangle_{input} = |x\rangle\ |y\rangle, \tag{10}$$

where $x = 0$ or 1 and $y = 0$ or 1. I am dropping the indices on the qubit kets: it is understood that the first ket refers to qubit number one and the second to qubit number 2. We now construct some complicated unitary operator U_f that computes the function and gives the output

$$|X\rangle_{output} = U_f\ |x\rangle\ |y\rangle = |x\rangle\ |f(x) + y\rangle, \tag{11}$$

where addition is understood to be mod (2) addition, so $1 + 1 = 0$. Thus $|y\rangle$ is flipped if $|f(x)\rangle = 1$, and $|y\rangle$ is left alone if $|f(x)\rangle = 0$ (a slightly more complicated version of a CNOT gate). More explicitly,

$$U_f\ |0\rangle\ |0\rangle = |0\rangle\ |f(0)\rangle \tag{12}$$

and

$$U_f\ |1\rangle\ |0\rangle = |1\rangle\ |f(1)\rangle. \tag{13}$$

To determine f, we set the value of the first bit and set the second bit to 0. The final measurement reads out the answer from the second. However, we still need to do 2 calculations, one for each of the two inputs. So the quantum computer offers no advantage over the classical computer.

Can we get the answer more efficiently using quantum parallelism? We might try the following, using the notion of parallelism introduced above. The quantum computer offers the possibility that we can prepare the input bit in a superposition of $|0\rangle$ and $|1\rangle$:

$$|X\rangle_{input} = \frac{1}{\sqrt{2}}\left(|0\rangle + |1\rangle\right)|0\rangle. \tag{14}$$

To get the output, just remember that the unitary transformation from input to output is linear and use Eqs. 12 and 13. This yields

$$|X\rangle_{output} = \frac{1}{\sqrt{2}} U_f \ (|0\rangle + |1\rangle) \ |0\rangle = \frac{1}{\sqrt{2}} \left[|0\rangle \ |f(0)\rangle + |1\rangle \ |f(1)\rangle \right]. \tag{15}$$

Now we measure the first bit. There is a probability 1/2 that it is in the state $|0\rangle$ and we find out the value of $f(0)$. With probability 1/2 it is in the state $|1\rangle$ and we find out the value of $f(1)$. This is even worse than the classical case! Even if we repeat the process twice, we might not find out what the function f is. So we conclude that the quantum computer offers no advantage over the classical computer for discovering *all* the values of f.

3.2 Problem Two (The Deutsch Algorithm)

Now, however, let us ask a subtly different question. Let us say we do not care about all the values of f but only whether $f(0) = f(1)$. If we want to answer this with a classical computer, we have no option but to calculate $f(0)$ and $f(1)$ separately and compare them. In other words, on a classical computer this problem is just as hard as the previous one and would take just as much time.

With the quantum computer we do the problem in 2 steps. We prepare the input state

$$|X\rangle_{input} = \frac{1}{2}(|0\rangle + |1\rangle) \ (|0\rangle - |1\rangle). \tag{16}$$

Then apply U_f. The output follows from linearity. It is

$$\begin{aligned} |X\rangle^1 &= U_f \ |X\rangle_{input} \\ &= \frac{1}{2}|0\rangle \ |f(0)\rangle + \frac{1}{2}|1\rangle \ |f(1)\rangle - \frac{1}{2}|0\rangle \ |1 + f(0)\rangle - \frac{1}{2}|1\rangle \ |1 + f(1)\rangle. \end{aligned} \tag{17}$$

If $f(0) = f(1) = c$, the

$$\begin{aligned} |X\rangle^1_{=} &= \frac{1}{2}|0\rangle \ |c\rangle + \frac{1}{2}|1\rangle \ |c\rangle - \frac{1}{2}|0\rangle \ |1 + c\rangle - \frac{1}{2}|1\rangle \ |1 + c\rangle \\ &= \frac{1}{2} \left(|0\rangle + |1\rangle \right) \ \left(|c\rangle - |1 + c\rangle \right) \end{aligned}$$

while if $c = f(0) \neq f(1) = 1 + c$, then

$$|X\rangle^1_{\neq} = \frac{1}{2}|0\rangle \ |f(0)\rangle + \frac{1}{2}|1\rangle \ |1 + c\rangle - \frac{1}{2}|0\rangle \ |1 + c\rangle - \frac{1}{2}|1\rangle \ |c\rangle \tag{18}$$

$$= \frac{1}{2} \left(|0\rangle - |1\rangle \right) \ \left(|c\rangle - |1 + c\rangle \right). \tag{19}$$

Now we apply a Hadamard to the first qubit and we get

$$\begin{aligned} H_1 \, |X\rangle^1_{=} &= \frac{1}{\sqrt{2}} \, |0\rangle \ \left(|c\rangle - |1 + c\rangle \right) \text{ and} \\ H_1 \, |X\rangle^1_{\neq} &= \frac{1}{\sqrt{2}} \, |1\rangle \ \left(|c\rangle - |1 + c\rangle \right). \end{aligned}$$

Now we measure the first qubit only and we find $|0\rangle$ for $f(0) = f(1)$ and $|0\rangle$ for $f(0) \neq f(1)$. And we have only done 1 computation of f, saving a factor of two in computer time! So in a certain sense it seems that the quantum computation beats classical computation. We have applied U_f once, not twice. This assumes that application of U_f is faster than two classical computations of f, which is imaginable, but it seems to me not likely, since the definition of U_f involves both $f(0)$ and $f(1)$. But we have certainly shown that two problems that are equivalent in the classical case are quite different in the quantum case. We were able to do this only by choosing the problem very carefully, and by designing the input state and measurement in a very clever way.

Classical Computers

Let's start with some terminology. A classical bit is a physical device that stores a 0 or a 1. This may be a capacitor, magnetic domain, etc. A set of bits that stores some convenient unit of information, such as a number or a word, is called a register. A register of N bits can store 2^N different numbers. Conversely, the number of bits required to store a number x is $\log_2 x$. An algorithm is a function takes a given input register or set of input registers to an output register.

Let us consider a computer that can do a calculation such as $2 + 3 = 5$. It must have two input registers that can store numbers from 0 to 3 and an output register that can store numbers from 0 to 7. This means 2 2-bit registers and 1 3-bit register. The computation $2 + 3 = 5$, now in binary notation, is

$$10 + 11 = 101. \tag{20}$$

The addition function is a mapping from a set of four input bits to a set of three output bits:

$$f_{add} : \{\{0,1\}_1^{in}, \{0,1\}_2^{in}, \{0,1\}_3^{in}, \{0,1\}_4^{in}\} \rightarrow \{\{0,1\}_1^{out}, \{0,1\}_2^{out}, \{0,1\}_3^{out}\} \tag{21}$$

from the input registers to the output register. The input space contains $2^4 = 16$ discrete points and the output space contains $2^3 = 8$ discrete points. An algorithm (such as addition) is a function whose argument can take on 16 possible values and which is then equal to one of 8 possible values. This means there are a finite number of possible algorithms on this computer. This is the number of ways of associating 2^3 outputs with each of 2^4 inputs, which is $2^{4+3} = 2^7 = 128$. If we define "algorithm" in this way, then the number of algorithms is is finite on any machine.

Clearly, no matter how big the input and output, we can repeat the above argument and calculate the number of possible algorithms. The finiteness of the possibilities gives us hope that we might be able to construct every algorithm by putting some very simple standard algorithms together in a finite chain. This turns out to be possible. The very simple standard algorithms are called gates. Some of the easiest to understand are the OR and AND gates, which take 2 bits into 1:

$$z = f_{OR}(x, y) = x + y - xy \pmod 2, \tag{22}$$

$$z = f_{AND}(x, y) = xy \pmod 2 . \tag{23}$$

The nomenclature comes from the idea of associating 0 with a false statement and 1 with a true statement. Then the OR gate says that z is true if either x or y is true, while the AND gate says that z is true if both x and y are true. Another simple and important gate is the COPY gate, which takes 1 bit into 2 bits.

$$f_{COPY}(x) = (x, x). \tag{24}$$

The collection of these 3 gates is a universal gate set. This means that any of the 128 algorithms on our microcomputer can be constructed by a sequence of such gates acting on a set of bits. The basic gates act on 2 bits i and j, so they can be written as $f^{ij}_{OR}(x, y)$, $f^{ij}_{AND}(x, y)$, and $f^{ij}_{COPY}(x)$, or, generically, as f^{ij}_a for $a = 1, 2, 3$. A chain of gates acting on the input registers has the form $f^{i_T j_T}_{a_T}\left(f^{i_{T-1} j_{T-1}}_{a_{T-1}}\left(f^{i_{T-2} j_{T-2}}_{a_T}(...)\right)\right)$. Now we see that we can estimate the time taken for a calculation. If a single gate takes a time τ, and there are T gates in the sequence, then than the total time for the computation is $T\tau$. $1/\tau$ is the "clock speed" of the computer.

We could also write the state of the computer as a big column vector and then each gate can be formulated as a matrix. Then the sequence that constitutes the algorithm becomes just a product of T matrices. This is very closely analogous to the operation of a quantum computer, as we shall see.

This set of 3 gates is sufficient to construct any finite algorithm on a classical computer. Mathematically, this statement can be reformulated by saying that any algorithm is a huge matrix and it can be written as a product of 3 simpler matrix types.

Of course, as you are aware, not all algorithms are as straightforward as addition. The input might affect the application of the f's, one might measure a register and use the information so obtained to modify the subsequent operations, and so on. All these complications go under the heading of "models of computation". But the fundamental idea of doing sequences of very simple operations underlies all computations.

4 Hardness of Problems

I'm going to give a very simplified discussion of this very important topic, omitting models of computation, deterministic and non-deterministic Turing machines, and so on.

We define the size of a problem N as the number of bits needed to specify the input completely. Adding 2 numbers, each of magnitude n, needs $N = 2\log_2 n$ bits for the input. The "time" it takes to solve the problem is the number of gate operations T. The time required for an algorithm to turn the input into the output for a given problem defines some function $T(N)$. The hardness of a problem is defined by the growth of the function $T(N)$ when the fastest algorithm is chosen. Usually we do not try to determine T in a completely explicit way. Instead we try to determine its asymptotic behavior as $N \to \infty$.

The big O (for "is the order of") notation is common: $T(N) = O(N^x)$ if there are constants C and N_0 such $T(N) \leq CN^x$ for some x, C and all $N > N_0$. In this case we say that the problem has a "polynomial-time algorithm". The other very important case is when $T(N) = O\left(x^N\right)$, that is there are constants x, C and N_0 such that $T(N) \leq Cx^N$ for some x, C and all $N > N_0$ In this case we say that the problem has an "exponential-time algorithm". Polynomial-time algorithms are sometimes called "efficient" algorithms, while exponential-time algorithms are called "inefficient". This is because exponential functions grow so much faster then polynomial ones as N increases. Theoretical computer scientists don't care too much about C because, even though it is important in figuring how much actual time it takes to solve a problem, C is dependent on the actual computer used. From a practical standpoint, however, reducing C might be important. The same holds for x, except even more so.

A very rough but useful rule of thumb is that if the time to do a problem is proportional to the number of bits in the input, then it is easy. If the time is proportional to the numbers in the input, then it is hard.

The jargon of complexity theory is a little weird, for historical reasons. If the fastest algorithm for a problem is polynomial, then we say that the problem is in P. If a given solution to a problem can be checked in polynomial time, then it is in NP. It is believed that $P \neq NP$, i.e., that there are problems whose solutions can be checked efficiently but that cannot be solved efficiently. An NP-complete problem is one in NP whose efficient solution would guarantee the efficient solution of all problems in NP. (This is shown by means of an appropriate efficient mapping from one problem to another.) NP-intermediate problems are in NP but are neither NP-complete nor in P. Factoring is believed to be in this class. NP-intermediate is known to non-empty if $P \neq NP$. Finally, NP-hard problems are those that are as least as hard as any NP-complete problem. NP-hard problems need not be in NP.

Turning from jargon to what is actually known for sure, it has never been proved that $P \neq NP$. Also, factoring has never been proved to be NP-intermediate - in fact no problem has been proved to be NP-intermediate.

We can clearly make similar classes for quantum computers by replacing bits by qubits and gate operations by quantum gate operations, and hardness of classical algorithms by hardness of quantum algorithms. This is not quite the end of the story and we'll come back to it later.

Lectures on Quantum Computing
Physics 709
Lecture 10: Group Theory

1 Groups

Before we start on our next quantum algorithm, we will need a little more mathematical background. In this section, I will introduce the idea of a group, a subgroup, and an important connection between groups and their subgroups.

Here are the group axioms for a set G with members $a, b, c, ..., g...$ and a binary operation in G. The binary operation could be anything we define. The only requirement is that G is closed under this operation. I use a multiplication notation for the operation because this is the most common in the physics literature, where matrices and matrix multiplication is very important. But one could just as well use addition.

1. Associativity: $(ab)c = a(bc)$.
2. Identity: there is an element $1 \in G$, such that $1g = g1 = g$ for all $g \in G$.
3. Inverses: every element g has an inverse g^{-1} such that $gg^{-1} = g^{-1}g = 1$.

If $ab = ba$ for all a and b in G, then the group is called a commutative group or an abelian group. There are many interesting groups that are not commutative, like the group of rotations in three dimensions, but most of the groups we will need in this course are commutative. $|G|$ is the number of elements in G and it is called the order of G.

A subgroup of G is a subset of G that is itself a group with the same operations of G. For example, the integers Z form a group under addition, with $-a$ being the inverse of a. A subgroup of Z is the even integers.

2 Lagrange's Theorem

Now we need one theorem, Lagrange's theorem, which states that: the order of a subgroup H of G divides the order of G: $|G| / |H|$ is an integer. This clearly only applies if $|G|$ is a finite number, in which case G is called a finite group. The statement is proved as follows.

Let G be of order N, $|G| = N$ and let the subgroup H be of order M, $|H| = M$. Then H is a set $H = \{h_0, h_1, ..., h_{M-1}\}$ of distinct elements of G and since H is a subgroup it must contain 1 so let $h_0 = 1$.

If $H = G$ then we are done since then $|G| / |H| = 1$. If not then take some element $g \in G$ but $g \notin H$. Form the set of M elements $gH = \{gh_0, gh_1, ..., gh_{M-1}\}$. These are all distinct from each other and in one-to-one correspondence to the elements of

H since each may be multiplied by g^{-1} to get to an element of H. Also, H and gH do not share any elements. For if $gh_i = h_j$, then $g = h_j h_i^{-1} \in H$, in contradiction to our choice of g. Thus the two sets H and gH have the same size: $|H| = |gH| = M$. If these two sets exhaust all members of G, then $|G| / |H| = 2$ and the theorem holds. If not, then we take some element g' that has not already appeared (does not belong to H or gH), and do the same thing again. Again $g'H$ shares no elements with H or gH and $|g'H| = |gH| = |H| = M$. This process eventually ends, since G is finite group. Hence M, the order of H divides N, the order of G: $|G| / |H|$ =integer.

We get an important bonus from this proof. Whenever we find a subgroup H of a group G we also get a natural partition of G into subsets, each of size $|H|$. The N/M subsets $H, gH, ...$, are called the cosets of G under the subgroup H.

Here's a simple example of a group, a subgroup, and its cosets. Take $G = Z_6 = \{0, 1, 2, 3, 4, 5\}$ with the group operation being addition mod 6, so $1 + 3 = 4$ and $2 + 5 = 1$, and so on. The identity element is 0 (not 1 !). There is a subgroup $H = \{0, 3\}$ of order $M = 2$. The cosets of Z_6 under H are $\{0, 3\}, \{1, 4\}, \{2, 5\}$. Each coset is of order 2 and there are 3 of them. A function that is constant on the cosets and takes on distinct values on each coset would be $f(0) = 1,\ f(1) = 5,\ f(2) = 4,\ f(3) = 1,\ f(4) = 5,\ f(5) = 4$.

If the group is not abelian we must distinguish between left and right cosets, but we will mainly talk about abelian groups in this course.

For the Simon algorithm we need the group $(Z_2)^n$. $Z_2 = \{0, 1\}$ with addition mod 2 as the group operation. Z_2 is the simplest non-trivial group. $G = (Z_2)^n$ is the set of all n-digit binary numbers and the group operation is bitwise addition modulo 2 (which is different from addition of binary numbers). So for this particular choice of operation we have (for $n = 3$) the operation $101 + 110 = 011$. Our notation is that $X = (x_0, x_1, ..., x_{n-1})$ is an n-bit string that belongs to this group, and the addition operation is

$$\begin{aligned} X + a &= (x_0, x_1, ..., x_{n-1}) + (a_0, a_1, ..., a_{n-1}) \\ &= ((x_0 + a_0) \,(\mathrm{mod}\, 2), (x_1 + a_1) \,(\mathrm{mod}\, 2), .., (x_{n-1} + a_{n-1}) \,(\mathrm{mod}\, 2)). \end{aligned}$$

$0 = (0, 0, ..., 0)$ is the identity element. The inverse of any element X is X itself: $X + X = 0$ so $X = X^{-1}$. Any element a defines a subgroup $A = \{0, a\}$. This subgroup partitions G into $2^n - 1$ distinct sets, each containing 2 elements. For if we take an arbitrary nonzero element $g_1 \neq a$ of the group, then there is also $g_1 + a$, which is distinct from $0, a$, and g_1. So there is the coset $\{g, g + a\}$. Now choose an elements g_2 not equal to $0, a, g_i$ or $g_1 + a$. Add a to it to get $g_2 + a$ and so on, just as in the proof above. The set of 2^{n-1} cosets is $\{g_0, g_0 + a\}, \{g_1 + a\}, \{g_2, g_2 + a\}, ... \left\{g_{(2^{n-1}-1)}, g_{(2^{n-1}-1)} + a\right\}$where $g_0 = 0$.

Physics 709
Quantum Computing
Lecture 11: Quantum Algorithms

1 Simon problem

The Deutsch algorithm is interesting, but purely as a matter of principle - the problem it solves is so simple that it is not of much use. We are now going to build up some more interesting algorithms step-by-step. Simon is next. As in the Deutsch algorithm, the Simon algorithm involves a very specific (and slightly artificial) class of functions. For the Simon algorithm we once more assume that we have a subroutine that calculates a function f, but this time the input and output are a bit more complicated. Both input and output have n bits, that is $f : \{0,1\}^n \rightarrow \{0,1\}^n$. Thus f takes values in a set with 2^n elements and we identify it with $(Z_2)^n$ and the "+" sign means bitwise addition.

Here is the defintion of the problem that the algorithm solves. We restrict our class of function to have a *very* special structure: $f(X) = f(Y)$ if and only if $X = Y$ or $Y = X + a$, but we do not know beforehand what a is. We do, however, have a subroutine available that will calculate $f(X)$ for any input X. Put another way, f is periodic but we do not know the period. Our notation is that $X = (x_0, x_1, ..., x_{n-1})$ is an n-bit string that belongs to this group.

As can be seen by looking at the set of cosets given above, we can sum over the whole group by summing over 2^{n-1} distinct elements g_i as long as we also sum over $g_i + a$. $\sum_{g \epsilon G/A}$ denotes the sum over g_i from $i = 0$ up to $2^{n-1} - 1$. We may summarize the properties of f by saying that it is constant on the cosets, but takes distinct values on different cosets.

Here's a very simple example for $n = 2$. $G = \{(00), (01), (10), (11)\}$. (00) is the identity element. Here is a posssible f :

$$\begin{aligned} f(00) &= 10 \\ f(01) &= 11 \\ f(10) &= 10 \\ f(11) &= 11 \end{aligned}$$

In this example, $a = (10)$.

Simon's problem is: given f satifying the contsaints, find a. This is an example of a what is called a hidden subgroup problem, the subgroup here being $A = \{0, a\}$.

This is a generalization to an n-digit number from a 1-digit number for the Deutsch problem: there X and Y were 0 or 1 and a had to be 0 or 1. However, this is a highly non-trivial generalization, because there are 2^n n-digit numbers.

A naive classical algorithm would choose an $h_1 \in G$, compute $f(h_1)$, choose another $h_2 \neq h_1$, compute $f(h_2)$ and compare it to $f(h_1)$. If $f(h_1) = f(h_2)$ then $a = h_2 - h_1$ and the algorithm stops. If not, then compute $f(h_3)$ and compare to the earlier values until finally we get an equality. This algorithm is guaranteed to work after 2^{n-1} steps, and on average it will do rather better than this, but the run time is still of order $\sqrt{2^n} = 2^{n/2}$. There are somewhat better algorithms, but even the best known classical algorithms are still exponential in n. *So this is a hard problem classically!*

2 Algorithm

Simon's algorithm is a quantum algorithm to find a much faster. Let us recall that in our notation the basis states are $|X\rangle = |x_0\rangle |x_1\rangle \cdots |x_{n-1}\rangle$ (the computational basis) and each basis state corresponds to the binary number $X = (x_0, x_1, ..., x_{n-1}) = x_0 \times 2^0 + x_1 \times 2^1 + x_2 \times 2^2 + ... + x_{n-1} \times 2^{n-1}$.

The one-qubit Hadamard H_i in the computational basis is given by a two-by-two unitary matrix

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

applied to the i-th qubit, and the n-qubit Hadamard H is defined by applying this transformation to each qubit. The product is then given by

$$H|X\rangle = H_0|x_0\rangle H_1|x_1\rangle \cdot \cdot H_{n-1}|x_{n-1}\rangle = \frac{1}{2^{n/2}} \sum_{Y=0}^{2^n-1} (-1)^{X \cdot Y} |Y\rangle ,$$

where $X \cdot Y = x_0y_0 + x_1y_1 + ... + x_{n-1}y_{n-1} \bmod 2$ is the mod 2 scalar product. (You should check this equation!).

Our black-box program for computing f does the unitary transformation

$$U_f \ |X\rangle |Y\rangle = |X\rangle \ \ |Y + f(X)\rangle$$

The algorithm is as follows.

1. Initialize two n-qubit registers in the state

$$|\psi_1\rangle = \ |0\rangle^n |0\rangle^n .$$

The first register is the input register and the second register is called the "ancillary" register.

2. Apply the n-qubit Hadamard to the input register.

$$|\psi_2\rangle = \ (H \ |0\rangle^n) |0\rangle^n = \frac{1}{2^{n/2}} \sum_{X=0}^{2^n-1} \ |X\rangle \ \ |0\rangle^n .$$

3. Apply U_f.

$$\begin{aligned}
|\psi_3\rangle &= U_f \ |\psi_2\rangle \\
&= \frac{1}{2^{n/2}} \sum_{X=0}^{2^n-1} \ |X\rangle \ \ |f(X)\rangle \\
&= \frac{1}{2^{n/2}} \sum_{X \in G/A} (|X\rangle \ \ |f(X)\rangle + |X+a\rangle \ \ |f(X+a)\rangle) \\
&= \frac{1}{2^{n/2}} \sum_{X \in G/A} (|X\rangle + |X+a\rangle) \ \ |f(X)\rangle \, ,
\end{aligned}$$

where the sum runs over the 2^{n-1} values of X such that no two elements of the sets are related by addition of a.

4. Apply the Hadamard again to the input register

$$\begin{aligned}
|\psi_4\rangle &= \frac{1}{2^{n/2}} \sum_{X \in G/A} \sum_{Y=0}^{2^n-1} \left[(-1)^{X \cdot Y} |Y\rangle + (-1)^{X \cdot Y + a \cdot Y} |Y\rangle \right] \ \ |f(X)\rangle \\
&= \frac{1}{2^{n/2-1}} \sum_{X \in G/A} \sum_{a \cdot Y = 0} (-1)^{X \cdot Y} |Y\rangle \, |f(X)\rangle \, .
\end{aligned}$$

5. Now measure the input register, obtaining some result Y. There are many possible values for the binary number Y , but each satisfies $a \cdot Y = 0$.

6. Repeat $O(n)$ times obtaining $n-1$ linearly independent values $Y_1, Y_2, ...Y_n$ of Y that are orthogonal to a. (To prove that we need only $O(n)$ tries, we need to note that 2^{n-1} values of Y satisfy this relation and that the probability of measuring the various Y's in this set is uniform. but this seems pretty clear.)

7. Solve the set of $n-1$ equations $a \cdot y_1 = 0, ..., a \cdot y_n = 0$ for a. This can be done by a method which works somewhat similarly to the Gaussian elimination method for sets of linear equations. It runs in polynomial time.

We will not give the details of the proof that this can be done in polynomial time, but we can give a plausibility argument rather easily. If we wish to find a vector $\vec{a}$ in 3-space and we have a vector $\vec{y}$ that satisfies $\vec{a} \cdot \vec{y} = 0$, this means that $\vec{a}$ lies in a certain plane. In our problem $\vec{a}$ has components that are zeros and ones so it lies on the corner of a cube in 3-dimensional space. Applying the condition $\vec{a} \cdot \vec{y}_1$ we find that $\vec{a}$ lies now in a plane. Applying the condition $\vec{a} \cdot \vec{y}_2$ we find that $\vec{a}$ lies now in a line. Each equation reduces the dimension of space where a can be by 1. So we only need 2 vectors $\vec{y}$ to solve the problem. In the general case, $n-1$ linearly independent vectors will work.

Notice that quantum interference is used to get the final result in step 4, while quantum entanglement also seems to be necessary since we need 2 registers, not 1, and the wavefunction is not a product wavefunction in the middle of step 3. So quantum mechanics appears to enter in 2 ways.

Lectures on Quantum Computing
Physics 709

Lecture 12: Shor Algorithm (1)

1 Introduction

Most of the excitement about quantum computing comes from the fact that one can use it to break certain "unbreakable" codes. These codes are the ones based on factoring of large numbers. The most common one is the RSA code. I will go through the code and show why it is difficult to break using classical computers. Then I will show how it can be broken using the Shor algorithm. We will need considerable background in number theory first.

2 Division and Factoring

An integer a is divisible by an integer d if there is another integer c such that $a = cd$. When this is true then we write $d|a$. When it is not true we write $d \nmid a$. For example, 10285 is divisible by 121 since $10285 = 121 \times 85$, but not by 2, since $2 \times 5142 < 10285$ and $2 \times 5143 > 10285$. So $121|10285$ but $2 \nmid 10285$.

Another way of looking at this is that the remainder r when 10285 is divided by 121 is 0, but the remainder r when 10285 is divided by 2 is 1. A zero remainder is the same thing as divisibility. When a is divided by d, the remainder always lies between 0 and $d-1$ inclusive. More systematically, we may say that

$$a = cd + r \tag{1}$$

where c can be anything but

$$0 \leq r \leq d-1. \tag{2}$$

There is another notation that is often used for Eq. 1. This is

$$a = r \pmod{d}. \tag{3}$$

This comes from thinking of the numbers from 0 to $d-1$ as themselves forming a number system. These numbers may be added and then reduced by subtracting out multiples of d. They then form a group. It is called the cyclic group of order d, or Z_d for short. We already met Z_2 in the Simon problem.

Any number a has the trivial divisors 1 and a. A number which has *only* these divisors is called a prime number. (Most authors do not include 1 as a prime number.) All other numbers are called composite.

Factoring a number means finding all the divisors, or factors. This problem is well-defined because of the fundamental theorem of arithmetic, which says that the decomposition of a number into its prime factors is unique up to the ordering of the prime factors:

$$a = (p_1)^{n_1} (p_2)^{n_2} (p_3)^{n_3} \cdots (p_x)^{n_x}, \tag{4}$$

where $p_1, p_2, ..., p_x$ are distinct primes. Thus $10285 = 5 \times 11 \times 11 \times 17$, so that, in this case, $p_1 = 5$, $n_1 = 1$, $p_2 = 11$, $n_2 = 2$, and $p_3 = 17$, $n_3 = 1$. $x = 3$. The divisors are $1, 5, 11, 17, 55, 85, 121, 187, 605, 935, 2057, 10285$. All of these are obtained by multiplying the prime factors together and therefore has the form

$$a = (p_1)^{m_1} (p_2)^{m_2} (p_3)^{m_3} \cdots (p_x)^{m_x},$$

with $0 \leq m_i \leq n_i$. So if we want to find all divisors, it is sufficient to find all the prime factors.

If we can find even one factor, we have made a lot of progress. Let's try a simple algorithm to do that. We want to factor the number a. We start with 2, see if $a/2$ is an integer. If so then 2 is a factor. If not then go on to 3, and finally check all numbers up to $N^{1/2}$. The runtime is $O(N^{1/2})$. This is not very clever, but even the best classical algorithm known has a runtime of about $O(a^{1/3})$. Since $a = 2^n$, where n is the number of bits, the complexity of this problem is about $O\left(2^{n/3}\right) = O\left(e^{(2/3)\ln 2}\right)$, which is exponential.

There is no known classical efficient algorithm for finding the prime factors of an arbitrary number. It has not been proved that there is no such algorithm, but many knowledgeable people do not believe it exists.

3 Mutual Divisibility

We can extend the concept of divisibility to mutual divisibility. Two numbers a and b may share divisors, and the largest such common divisor d is called the greatest common divisor, or $\gcd(a, b) = d$. Any common divisor of a and b divides d. Amazingly, there is a polynomial-time algorithm for finding the gcd. Even more amazingly, it has been known for over 2000 years. It is called the Euclidean algorithm. Here it is.

Suppose $a > b$. Then we divide a by b and compute the remainder r_2. Then we divide b by r_2 and compute the remainder r_3. Continue until there is no remainder. Let $r_0 = a$ and $r_1 = b$. We have the series of equations

$$\begin{aligned}
r_0 &= k_1 r_1 + r_2, \quad 0 \leq r_2 < r_1 && (5)\\
r_1 &= k_2 r_2 + r_3, \quad 0 \leq r_3 < r_2 && (6)\\
&\cdot && (7)\\
&\cdot && (8)\\
r_{n-2} &= k_{n-1} r_{n-1} + r_n, \quad 0 \leq r_n < r_{n-1} && (9)\\
r_{n-1} &= k_n r_n + r_{n+1}, \quad r_{n+1} = 0. && (10)
\end{aligned}$$

This is clearly a deterministic algorithm that is finite, since the r_i form a strictly decreasing sequence.

We claim that $r_n = \gcd(r_0, r_1) = \gcd(a, b)$.

Here is the proof, which has two parts.

1. We show that $r_n \mid a$ and $r_n \mid b$. The last line shows that r_n divides r_{n-1}. Then looking at the next-to-last line shows that r_n divides r_{n-2} and so on and we conclude that r_n divides r_0 and r_1. So r_n is a common divisor of r_0 and r_1.

2. To see that r_n is the *greatest* common divisor we show that all common divisors of a and b divide r_n. If r_n were not the biggest, then there would be a larger one that would not divide r_n. So take any common divisor d of r_0 and r_1. So $d \mid r_0$ and $d \mid r_1$. Then note that, from the first equation $r_2 = r_0 - k_1 r_1$, which is a multiple of d so $d \mid r_2$. The second equation shows that $d \mid r_3$ and this continues down the set of equations until we finally conclude that $d \mid r_n$. Since d was an arbitrary divisor we have that $r_n = \gcd(r_0, r_1)$, which completes the proof.

How many steps are required to find r_n? Notice that in a division-with-remainder equation

$$a = kb + r \tag{11}$$

we have that $r \leq kb$, so $r \leq a/2$. Applying this to our equations we have that $r_{n+2} \leq r_n/2$. In base 2, this means we shorten the number by at least one digit every two steps. Hence the number of steps is only $O(\log_2 a)$, and this is a polynomial-time algorithm.

We might think we could factor a number a by using the Euclidean algorithm. For each a , choose a b and find $\gcd(a, b)$. Then the question is: how many values of b would we need to test? The problem is that we might have that $\gcd(a, b) = 1$, so that a and b have no nontrivial divisors in common. Such pairs of numbers are said to be relatively prime. If a and b are relatively prime, then we get no closer to finding a divisor of a.

This leads to the question: given a, how many numbers less than a are relatively prime to it? This function is denoted by $\phi(a)$ and it is called called Euler's function. For example, $\phi(5) = 4 = |\{1, 2, 3, 4\}|$; $\phi(8) = |\{1, 3, 5, 7\}| = 4$, and so on. ϕ is of course a very wildly varying function, since a prime number p has $\phi(p) = p - 1$, and it can right next to a number that has lots of divisors: $\phi(p + 1) << p - 1$. However, it is possible to consider instead the running average $F(n) = \sum_{a=1}^{n} \phi(a)$, which is much smoother. In fact, there is an asymptotic formula for this function

$$F(n) \to \frac{6}{\pi^2} \frac{n(n+1)}{2} + O(n \log n). \tag{12}$$

Since $n(n+1)/2$ is the number of pairs of numbers in which each member is less than n, this implies that the probability that two numbers chosen at random are relatively prime is $6/\pi^2$.

This tells us that we can factor most numbers a using the Euclidean algorithm, since the chance of finding a factor this way is $1 - 6/\pi^2 \approx 0.392$, and we can take a

large number of b's. However, if $\phi(a)$ is close to a, as it might be, then a becomes hard to factor. We say that *on average*, factoring is not not a hard problem.

However, as we will soon see, we are actually interested not in typical numbers but rather certain numbers that are hard to factor. In physics, we are usually interested in typical cases, but in cryptography we are interested in worst cases!

Lectures on Quantum Computing
Physics 709

Lecture 13: Shor Algorithm (2)

1 Linear combinations

One final different take on the greatest common denominator is necessary. We can make linear combinations of our pair of numbers a and b, forming a set of numbers $C(x, y)$:

$$xa + yb = C. \tag{1}$$

Here x and y can be any integers, positive or negative, but let us only consider positive answers C.

Theorem. We claim that the smallest such number $c = \min_{x,y} C(x, y) = \gcd(a, b)$.

Proof. Again we show that c is the gcd of a and b in 2 steps. First we show that we show that any divisor of a and b divides c. Then we show that $\mid a$ and $c \mid b$ and second

From Eq. 1 and the fact that if a and b have a common divisor d we have

$$xa + yb = x\alpha d + y\beta d = (x\alpha + y\beta)\, d = c$$

so $d|c$ as well.

But it is also true that $c|a$, since we can write a as

$$a = ct + r, \tag{2}$$

where $0 \leq r \leq c - 1$. But then

$$a = t(xa + yb) + r \tag{3}$$

and

$$(1 - xt)a + (-yt)b = r, \tag{4}$$

so r can be written as a linear combination of a and b. But c is the smallest positive linear combination and $0 \leq r \leq c - 1$. The only possibility is that $r = 0$. (This last step is a little tricky, but think about it: c is positive but r is non-negative.) Thus $a = ct$ and so $c|a$. The same argument shows that $c|b$. But if c divides both a and b and any divisor of both a and b divides c, then $c = \gcd(a, b)$.

The most important application of this for our purposes is that if (and only if) a and b are relatively prime, then there exist integers x and y such that

$$xa + yb = 1. \tag{5}$$

An example is $a = 10, b = 7$. Then using $x = 10$ and $y = -7$, we have $10 \times 10 - 7 \times 7 = 1$.

1.1 Multiplication mod n

We have shown previously that, given a number n, the numbers $\{0, 1, ..., n-1\}$ form a group under the operation of addition modulo n. Now we ask the question, do they form a group under the operation of multiplication modulo n? Recall the group axioms for a set G and a closed binary operation

1. Associativity: $(ab)c = a(bc)$
2. Identity: there is an element $1 \in G$, such that $1g = g1 = g$ for all $g \in G$.
3. Inverses: every element g has an inverse g^{-1} such that $gg^{-1} = g^{-1}g = 1$.

(Reminder: our groups are always commutative: $ab = ba$.)

Let us look at the set $\{0, 1, 2, 3, 4, 5, 6, 7\}$ with the operation of multiplication modulo 8. It clearly satisfies 1 and 2. However, there is a problem with axiom 3. The number 2 has no inverse:

$$\begin{aligned} 2 \cdot 0 &= 0 && (6)\\ 2 \cdot 1 &= 2 && (7)\\ 2 \cdot 2 &= 4 && (8)\\ 2 \cdot 3 &= 6 && (9)\\ 2 \cdot 4 &= 8 = 0 && (10)\\ 2 \cdot 5 &= 10 = 2 && (11)\\ 2 \cdot 6 &= 12 = 4 && (12)\\ 2 \cdot 7 &= 14 = 6. && (13) \end{aligned}$$

There is no element that satisfies $2 \cdot g = 1$. This is not a group. The problem is clear: $2|8$, and we can never get anything but even numbers as the result of multiplying by 2 and taking multiples of 8 away. This suggests that we restrict ourselves to the numbers that are relatively prime to 8, the set $Z_8^* = \{1, 3, 5, 7\}$ with the operation of multiplication modulo 8. This is indeed a group, called the reduced residue classes modulo 8. (You should prove that the set is closed under multiplication.) Here are the inverses:

$$\begin{aligned} 1 \cdot 1 &= 1 && (14)\\ 3 \cdot 3 &= 9 = 1 && (15)\\ 5 \cdot 5 &= 25 = 1 && (16)\\ 7 \cdot 7 &= 49 = 1. && (17) \end{aligned}$$

The fact that each number is its own inverse is not common among the Z_n^* groups, but it is true for $n = 8$.

This way of constructing the group works for any n and any group element g. g and n are relatively prime, so we know from the previous section that there exist x and y such that

$$xg + yn = 1. \tag{18}$$

Notice in this equation that we must have $\gcd(x, n) = 1$. If $\gcd(x, n) = s > 1$, then the left-hand side of the equation is divisible by s but the right-hand side is not. So $x \in Z_n^*$.

But we can write the equation as

$$xg = 1 (\mathrm{mod}\, n). \tag{19}$$

Hence $x = g^{-1}$ and there is always an inverse.

Note that if n is prime, then all the numbers less than p are in the group. In general, the order of Z_n^* is $\phi(n)$, by definition.

1.2 Euler's theorem

Lagrange's theorem was that the order of a subgroup H of G divides the order of G.

This allows us to prove a crucial theorem for the Shor algorithm, named after Euler.

Lemma.

If N is the order of G, and a is any element of G, then

$$a^N = 1. \tag{20}$$

Here is the proof, which after what we have studied is now rather simple. Form the sequence of elements a, a^2, a^3, etc. Since there are only N elements in the group, this sequence must repeat after some point, say $a^x = a$, where $x \leq N+1$. Multiplying by a^{-1}, we find $a^y = 1$, where $y = x-1 \leq N$. However, consider the set $\{a, a^2, ...a^z\}$, which has z elements, where z is the *smallest* positive integer satisfying $a^z = 1$. The set of elements $\{a, a^2, a^3, ..., a^z = 1\}$ forms a group (isomorphic to Z_z), which is of order z. By Lagrange's theorem, $z|N$, or there is a w satisfying $N = wz$. Finally $a^N = a^{zw} = 1^w = 1$, and the theorem is proved.

Euler's Theorem.

Applying this to the group Z_n^*, we find the amazing Euler's theorem, which states that

$$a^{\phi(n)} = 1 (\mathrm{mod}\, n) \tag{21}$$

if a and n are relatively prime.

Incidentally, Euler's theorem gives a simple test for primality. If p is prime, then $\phi(p) = p - 1$ and all the numbers greater than 1 but less than p are relatively prime to p. So for all primes p and all $1 < a < p$ we have

$$a^{p-1} = 1 (\mathrm{mod}\, p). \tag{22}$$

We may test a given number p for primality by choosing some a and trying out the above equation. If it fails, then p is definitely composite. If it succeeds then we do not learn anything useful. This is a rather simple test and not the best one. In general, it is much easier to test whether a number is composite than to factor it.

2 Factoring large numbers using period-finding

The code that we will soon be looking at depends on having a large number N that is hard to factor using the Euclidean algorithm. The best choice is to take the product of 2 distinct primes: $N = pq$. If we wish to break the code, we need to factor N, that is, find p and q. If we try to factor a number like this using the Euclidean algorithm, we are very unlikely to be successful, since $\phi(N) = (p-1)(q-1) = N - O(N^{1/2})$, which means that a random number less than N is overwhelmingly likely to be relatively prime to N.

The key point now is that we can reduce the factorizing N to the problem of finding the period of a certain function. The strategy is to choose a number $a < N$ at random. First, we compute $\gcd(a, N)$. If $\gcd(a, N) = w \neq 1$, then we are done, since w is a factor of N. (Very unlikely when N is a product of 2 primes.) If $\gcd(a, N) = 1$, then we define the function $F(x)$ by the equation

$$F(x) = a^x \pmod{N}. \tag{23}$$

As we noted in the proof of Euler's theorem, this function is periodic. But the period may be long, $O(N)$, and classically we need that many evaluations of F. Indeed finding the period classically is felt to be a hard problem. Let us ignore that for the moment and imagine that the period is r:

$$F(x + r) = a^{r+x} \pmod{N} = F(x) = a^x \pmod{N}. \tag{24}$$

Hence

$$a^r = 1 \pmod{N} \tag{25}$$

or

$$a^r - 1 = 0 \pmod{N}. \tag{26}$$

Just to remind you, this means that $a^r - 1 = 0$ or $a^r - 1 = N$, or $a^r - 1 = 2N$...

Now let us assume that r is even. Then we can also write Eq. 26 as

$$(a^{r/2} - 1)(a^{r/2} + 1) = 0 \pmod{N}. \tag{27}$$

What this says, in ordinary language, is that the product on the left is a multiple of N. This decomposition only works if r is even. If it is odd, then we need to go back and choose another a and start over. [Prof. Kao points out that we might be able to use the factorization $(a-1)(a^{r-1} + a^{r-2} + ... + 1)$ instead.] Hence p must divide $(a^{r/2} - 1)$ or $(a^{r/2} + 1)$ and q must divide $(a^{r/2} - 1)$ or $(a^{r/2} + 1)$. There are

altogether four possibilities. However, we can rule out one right away. p and q cannot both divide $(a^{r/2}-1)$ since then $pq = N$ divides $a^{r/2}-1$ or $a^{r/2} = 1 \pmod N$, which would mean that the period of F is $r/2$, not r, contrary to the definition of r. If only one of p and q divides $(a^{r/2}+1)$, then we are done: we just find $\gcd(N,(a^{r/2}+1))$ and this will give us the desired factor. If both p and q divide $(a^{r/2}+1)$ then our algorithm fails, since finding $\gcd(N,(a^{r/2}+1))$ would only give us the product of p and q, which is N and that we already know. If this is the case, we have to go back, choose another value of a and start over again. The point, however, is that we have a good chance of getting the answer - more precisely, the probability of getting the answer is independent of N. So the fact that the algorithm only works part of the time does not lead to a great increase in the computer time. We will quickly find a value of a that works.

To take a simple example, let $N = 15$. We want to find its prime factors. Take, at random, $a = 7$. $\gcd(15,7) = 1$. We first find the period:

$$F(1) = 7^1 \pmod{15} = 7, \tag{28}$$

$$F(2) = 7^2 \pmod{15} = 49 \pmod{15} = 4, \tag{29}$$

$$F(3) = 7^3 \pmod{15} = 7 \times 4 \pmod{15} = 13, \tag{30}$$

$$F(4) = 7^4 \pmod{15} = 7 \times 13 \pmod{15} = 91 \pmod{15} = 1. \tag{31}$$

Hence $r = 4$. and Eq. 27 is

$$(7^2-1) \times (7^2+1) = 48 \times 50 = 0 \pmod{15}, \tag{32}$$

which is certainly true, since

$$48 \times 50 = 2400 = 160 \times 15. \tag{33}$$

Now we compute $\gcd(15,50) = 5$, which is indeed one of the factors of 15.

Lectures on Quantum Computing
Physics 709
Lecture 14:
Shor Algorithm (Quantum Part)

1 Classical Fourier Transform

You are familiar with the ordinary Fourier transform (FT), and with its many uses. I will just review it here to establish notation. We start with a complex-valued function $f(z)$ defined on a finite interval $[0, 1]$. Then its Fourier transform is

$$\widetilde{f}(k) = \int_0^1 f(z)\, e^{2\pi i k z} dz, \tag{1}$$

where k is any integer. The computer can't store a function of a real variable, so for numerical purposes the interval is split into some large but finite number N of points, say $0, 1/N, 2/N, ..., (N-1)/N$. Let the value of the function at these points be x_j, where j goes from 0 to $N-1$. Then we define the discrete FT of this function as

$$y_k = \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} x_j e^{2\pi i j k/N} \tag{2}$$

where k also runs from 0 to $N-1$. Notice that this can be written as a matrix multiplication

$$\vec{y} = M\vec{x}$$

or in index notation

$$y_k = \sum_{j=0}^{N-1} M_{kj} x_j$$

where $M_{kj} = e^{2\pi i j k/N}/N$. Clearly a brute force calculation of all of the y_k from the x_j takes $O(N^2)$ operations, since there are N terms in the sum for each of the N calculations to be performed. There is actually quite a lot of redundancy in doing it that way, since all of the phases in the matrix M are not independent. One can eliminate that redundancy using some clever tricks, which gives the (classical) fast FT, with a runtime of $O(N\ \ln N)$.

2 Quantum Fourier Transform (QFT)

The QFT does much better as far as run time is concerned. It takes $O(\ln N)^2$ operations. However, it unfortunately does not yield quite the same information as the classical FT.

We have n qubits, so there are $N = 2^n$ states in the computational basis. Each state of the computational basis can be written in binary as

$$|j\rangle = |j_1 j_2 ... j_n\rangle \,, \tag{3}$$

where $j_1 = 0$ or 1, $j_2 = 0$ or 1, etc

$$j = \sum_{k=1}^{n} j_k 2^{n-k}. \tag{4}$$

Also, we will denote binary numbers less than 1 by

$$0.j_1 j_2 ... j_n = \sum_{k=1}^{n} j_k 2^{-k} \tag{5}$$

The input to the QFT is the binary number j,

$$|\Psi_{in}\rangle = |j_1 j_2 ... j_n\rangle$$

the QFT unitary transformation is

$$|\Psi_{out}\rangle = U_{FT} \; |j\rangle \tag{6}$$

and the output is

$$|\Psi_{out}\rangle = \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} e^{2\pi i j k/N} \, |k\rangle \,. \tag{7}$$

There is a simple trick that makes this possible, based on the astonishing identity

$$\frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} e^{2\pi i j k/N} \, |k\rangle \;\; = \;\; 2^{-n/2} \left(|0\rangle + e^{2\pi i 0.j_n} \, |1\rangle \right) \times \tag{8}$$

$$\left(|0\rangle + e^{2\pi i 0.j_{n-1} j_n} \, |1\rangle \right) \times \cdots \times \left(|0\rangle + e^{2\pi i 0.j_1 j_2 ... j_n} \, |1\rangle \right). \tag{9}$$

This shows that the transformed state is *not* entangled: it is a product state.

Here is the proof, each step of which is simple.

$$\frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} e^{2\pi i j k/N} \, |k\rangle \;\; = \;\; 2^{-n/2} \sum_{k_1=0}^{1} \sum_{k_2=0}^{1} \cdots \sum_{k_n=0}^{1} e^{\left(2^{-n}\right) 2\pi i j k} \; |k_1 k_2 ... k_n\rangle \tag{10}$$

$$= \;\; 2^{-n/2} \sum_{k_1=0}^{1} e^{2\pi i j 2^{-1} k_1} \sum_{k_2=0}^{1} e^{2\pi i j 2^{-2} k_2} \times \tag{11}$$

$$\cdots \times \sum_{k_n=0}^{1} e^{2\pi i j 2^{-n} k_n} |k_1 k_2 ... k_n\rangle \tag{12}$$

$$= 2^{-n/2} \sum_{k_1=0}^{1} e^{2\pi i j 2^{-1} k_1} |k_1\rangle \sum_{k_2=0}^{1} e^{2\pi i j 2^{-2} k_2} |k_2\rangle \otimes \tag{13}$$

$$\cdots \otimes \sum_{k_n=0}^{1} e^{2\pi i j 2^{-n} k_n} |k_n\rangle \tag{14}$$

$$= 2^{-n/2} \left(|0\rangle + e^{2\pi i j 2^{-1}} |1\rangle\right) \otimes \left(|0\rangle + e^{2\pi i j 2^{-2}} |1\rangle\right) \otimes \cdots \tag{15}$$

$$\otimes \left(|0\rangle + e^{2\pi i j 2^{-n}} |1\rangle\right) \tag{16}$$

$$= 2^{-n/2} \left(|0\rangle + e^{2\pi i 0.j_n} |1\rangle\right) \otimes \left(|0\rangle + e^{2\pi i 0.j_{n-1} j_n} |1\rangle\right) \otimes \cdot \tag{17}$$

$$\otimes \left(|0\rangle + e^{2\pi i 0.j_1 j_2 ... j_n} |1\rangle\right) \tag{18}$$

We have made use of the fact that

$$\frac{j}{2} = \frac{j_1 j_2 ... j_n}{2} = \text{integer} + \frac{j_n}{2} = \text{integer} + 0.j_n$$

and $\exp(2\pi i \times \text{integer}) = 1$ so

$$e^{2\pi i j/2} = e^{2\pi i \cdot (0.j_n)}.$$

The QFT circuit is rather simple, owing to the fact that the output state is not entangled. For the circuit we need a new kind of gate, a controlled R_m. If the control qubit is in the state $|0\rangle$ we do notthing to the target qubit, while if it is in the state $|1\rangle$ we apply the matrix

$$R_m = \begin{pmatrix} 1 & 0 \\ 0 & e^{2\pi i/2^m} \end{pmatrix}. \tag{19}$$

The qubits are prepared in the known state $|j\rangle = |j_1 j_2 ... j_n\rangle$. We then do a Hadamard on the first one

$$|j_1\rangle \rightarrow \frac{1}{\sqrt{2}} \left(|0\rangle + e^{2\pi i 0.j_1} |1\rangle\right), \tag{20}$$

then do a controlled R_2 :

$$R_2 = \begin{pmatrix} 1 & 0 \\ 0 & e^{2\pi i/4} \end{pmatrix} \tag{21}$$

where $|j_2\rangle$ is the control bit. This produces

$$|j_1\rangle \rightarrow \frac{1}{\sqrt{2}} \left(|0\rangle + e^{2\pi i 0.j_1 j_2} |1\rangle\right), \tag{22}$$

and then we do R_3 with $|j_3\rangle$ as the control bit, which gives

$$|j_1\rangle \rightarrow \frac{1}{\sqrt{2}} \left(|0\rangle + e^{2\pi i 0.j_1 j_2 j_3} |1\rangle\right), \tag{23}$$

and so on, finally getting

$$|j_1\rangle \rightarrow \frac{1}{\sqrt{2}}\left(|0\rangle + e^{2\pi i 0.j_1 j_2 j_3 \ldots j_n}\,|1\rangle\right) \tag{24}$$

for the first bit. Then we do H on the second bit (which is unchanged to this point), then R_3 with $|j_3\rangle$ as the control bit, and so on until we get

$$|j_2\rangle \rightarrow \frac{1}{\sqrt{2}}\left(|0\rangle + e^{2\pi i 0.j_2 j_3 \ldots j_n}\,|1\rangle\right) \tag{25}$$

and, finally

$$|j_n\rangle \rightarrow \frac{1}{\sqrt{2}}\left(|0\rangle + e^{2\pi i 0.j_n}\,|1\rangle\right). \tag{26}$$

This is the desired state, except that we must reverse the order of the qubits using a series of swaps.

In terms of time this is a fantastic improvement even on the FFT, since there are only n controlled R operations. Note that it uses the tensor product character of the bits in a very basic way, as did the Simon algorithm. But the quantum phases in the Simon and the Deutsch algorithms are only ± 1. The quantum interference in the QFT is much more elaborate.

All that being said, we must note that the QFT is not actually an algorithm for determining the Fourier transform of a function. This is because if we actually measure the final state

$$|\Psi_f\rangle = \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} e^{2\pi i j k/N}\,|k\rangle$$

in the computational basis, all we would get is a *random* n-digit binary number, picked uniformly from all of the k's. This is not useful information at all. This is why the QFT is not a real quantum algorithm. We can regard it as a quantum subroutine.

Lectures on Quantum Computing
Physics 709
Lecture 15: Shor Algorithm Concluded

1 Quantum Phase Estimation

Quantum phase estimation is a real quantum algorithm because it does give a useful answer when we do the final MCB. In fact it gives directly a certain desired phase.

The problem that this algorithm solves is this.

We are given an $N \times N$ unitary matrix U and an N-dimensional eigenvector $|u\rangle$:

$$U|u\rangle = e^{2\pi i\phi}|u\rangle . \tag{1}$$

$0 \leq \phi < 1$. We wish to calculate $\phi = 0.\phi_1\phi_2\cdots\phi_t$.

The algorithm is as follows. As input we take the tensor product

$$|\Psi_{in}\rangle = |0\rangle_1 \otimes |0\rangle_2 \otimes |0\rangle_3 \otimes \cdots |0\rangle_t \otimes |u\rangle . \tag{2}$$

Thus in this algorithm there are two registers. The first is the working register and second contains the input. We first perform a Hadamard on all of the first t qubits to get

$$\begin{aligned} |\Psi_1\rangle &= H_1 \otimes H_2 \otimes \cdots \otimes H_t |\Psi_{in}\rangle & (3)\\ &= 2^{-t/2}(|0\rangle_1 + |1\rangle_1) \otimes & (4)\\ &\quad (|0\rangle_2 + |1\rangle_2) \otimes (|0\rangle_3 + |1\rangle_3) \otimes \cdots (|0\rangle_t + |1\rangle_t) \otimes |u\rangle . & (5)\end{aligned}$$

This once again follows the quntum parallelism philosophy: there is a superposition of inputs.

Then we perform a controlled U on $|u\rangle$, with the t-th qubit as the control bit. This produces

$$\begin{aligned} |\Psi_2\rangle &= 2^{-t/2}(|0\rangle_1 + |1\rangle_1) \otimes & (6)\\ &\quad (|0\rangle_2 + |1\rangle_2) \otimes (|0\rangle_3 + |1\rangle_3) \otimes \cdots \left(|0\rangle_t |u\rangle + e^{2\pi i\phi}|1\rangle_t |u\rangle\right) & (7)\\ &= 2^{-t/2}(|0\rangle_1 + |1\rangle_1) \otimes & (8)\\ &\quad (|0\rangle_2 + |1\rangle_2) \otimes (|0\rangle_3 + |1\rangle_3) \otimes \cdots \left(|0\rangle_t + e^{2\pi i\phi}|1\rangle_t\right) \otimes |u\rangle & (9)\end{aligned}$$

Then we perform a controlled U^2 on $|u\rangle$, with the $t-1$-th qubit as the control bit. This produces

$$\begin{aligned} |\Psi_3\rangle &= 2^{-t/2}(|0\rangle_1 + |1\rangle_1) \otimes (|0\rangle_2 + |1\rangle_2) & (10)\\ &\quad \otimes (|0\rangle_3 + |1\rangle_3) \otimes \cdots \otimes \left(|0\rangle_{t-1} + e^{2\times 2\pi i\phi}|1\rangle_{t-1}\right)\left(|0\rangle_t + e^{2\pi i\phi}|1\rangle_t\right) \otimes |u\rangle \end{aligned}$$

and so on, until we get

$$\begin{aligned}
|\Psi_{t+1}\rangle &= 2^{-t/2}\left(|0\rangle_1 + e^{2^{t-1}\times 2\pi i\phi}|1\rangle_1\right)\otimes\left(|0\rangle_2 + e^{2^{t-2}\times 2\pi i\phi}|1\rangle_2\right) \\
&\quad \otimes\left(|0\rangle_3 + e^{2^{t-3}\times 2\pi i\phi}|1\rangle_3\right)\otimes\cdots\otimes\left(|0\rangle_{t-1} + e^{2\times 2\pi i\phi}|1\rangle_{t-1}\right)\left(|0\rangle_t + e^{2\pi i\phi}|1\rangle_t\right)\otimes|u\rangle \\
&= 2^{-t/2}\left(|0\rangle_1 + e^{2^{t-1}\times 2\pi i 0.\phi_t}|1\rangle_1\right)\otimes\left(|0\rangle_2 + e^{2^{t-2}\times 2\pi i 0.\phi_{t-1}\phi_t}|1\rangle_2\right) \quad (11) \\
&\quad \otimes\left(|0\rangle_3 + e^{2^{t-3}\times 2\pi i 0.\phi_{t-2}\phi_{t-1}\phi_t}|1\rangle_3\right)\otimes\cdots \quad (12) \\
&\quad \otimes\left(|0\rangle_{t-1} + e^{2\times 2\pi i 0.\phi_2\cdots\phi_t}|1\rangle_{t-1}\right)\left(|0\rangle_t + e^{2\pi i 0.\phi_1\cdots\phi_t}|1\rangle_t\right)\otimes|u\rangle \quad (13)
\end{aligned}$$

Once again we have sued the fact that $\exp(2\pi i\times\text{integer}) = 1$.

Now we use the identity from the QFT derivation to see that

$$|\Psi_{t+1}\rangle = 2^{-t/2}\sum_{j=0}^{N-1} e^{2\pi ij\phi}|\phi\rangle\otimes|u\rangle \quad (14)$$

and performing the inverse QFT on the ϕ part, we get

$$|\Psi_{out}\rangle = \text{QFT}\,|\Psi_{t+1}\rangle = |\phi\rangle\otimes|u\rangle\,, \quad (15)$$

or in tensor product form

$$|\Psi_{out}\rangle = |\phi_1\rangle_1\otimes|\phi_2\rangle_2\otimes|\phi_3\rangle_3\otimes\cdots\otimes|\phi_t\rangle_t\otimes|u\rangle\,. \quad (16)$$

So now we do a MCB and determine the phase to t places.

2 Order finding

Quantum phase estimation can be used to find the order of a number in a multiplicative group. We want the group Z_N^*. Choose a number x in Z_N^*.

To find the order r of x using phase estimation, define the unitary operator U_x by

$$U_x|y\rangle = |xy(\text{mod}\,N)\rangle\,. \quad (17)$$

U_x is unitary because it just permutes the numbers $y\in Z_N^*$. U_x has eigenstates

$$|u_s\rangle = \frac{1}{\sqrt{r}}\sum_{k=0}^{r-1}\exp\left[\frac{-2\pi isk}{r}\right]\left|x^k(\text{mod}\,N)\right\rangle \quad (18)$$

and the eigenvalues are $e^{2\pi is/r}$. Here s is an integer such that $s = 0, 1, 2, ..., r-1$.

The proof that these are indeed eigenstates with eigenvalues $e^{2\pi is/r}$ is:

$$U_x|u_s\rangle = \frac{1}{\sqrt{r}}\sum_{k=0}^{r-1}\exp\left[\frac{-2\pi isk}{r}\right]\left|x^{k+1}(\text{mod}\,N)\right\rangle\text{; now set } l = k+1 \quad (19)$$

$$= \frac{1}{\sqrt{r}} \sum_{l=1}^{r} \exp\left[\frac{-2\pi i s(l-1)}{r}\right] \left|x^l (\mathrm{mod}\, N)\right\rangle \quad (20)$$

$$= \frac{1}{\sqrt{r}} e^{2\pi i s/r} \sum_{l=1}^{r} \exp\left[\frac{-2\pi i s l}{r}\right] \left|x^l (\mathrm{mod}\, N)\right\rangle \quad (21)$$

$$= \frac{1}{\sqrt{r}} e^{2\pi i s/r} \sum_{l=0}^{r-1} \exp\left[\frac{-2\pi i s l}{r}\right] \left|x^l (\mathrm{mod}\, N)\right\rangle \quad (22)$$

$$= e^{2\pi i s/r} \left|u_s\right\rangle . \quad (23)$$

In going from the third to the fourth line we used the fact that $\exp(2\pi i s) = 1$ and $x^r = x^0 = 1$.

Applying phase estimation to U_x and $|u_s\rangle$ gives us the output state $|\phi_s\rangle |u_s\rangle$ and measurement gives the phase $\phi_s = s/r$. We'll see in a moment that this is enough to find r.

The problem with this procedure as it stands is that, for the phase estimation algorithm, we need to know the eigenstate in advance, and constructing the $|u_s\rangle$ appears to involve knowledge of the unknown quantity r. However, it turns out that we can start with the state $|u\rangle = |1\rangle$, i.e.,

$$|\Psi_{in}\rangle = |0\rangle_1 \otimes |0\rangle_2 \otimes |0\rangle_3 \otimes \cdots |0\rangle_t \otimes |0\rangle_1 \otimes |0\rangle_2 \otimes |0\rangle_3 \otimes \cdots |1\rangle_n \quad (24)$$

and still get, in a somewhat roundabout way, the desired answer. This follows from the identity

$$\frac{1}{\sqrt{r}} \sum_{s=0}^{r-1} |u_s\rangle = \frac{1}{r} \sum_{s=0}^{r-1} \sum_{l=0}^{r-1} \exp\left(\frac{-2\pi i s l}{r}\right) \left|x^l \,(\mathrm{mod}\, N)\right\rangle \quad (25)$$

$$= \frac{1}{r} \sum_{l=0}^{r-1} \sum_{s=0}^{r-1} \exp\left(\frac{-2\pi i s l}{r}\right) \left|x^l \,(\mathrm{mod}\, N)\right\rangle \quad (26)$$

$$= \frac{1}{r} \sum_{l=0}^{r-1} \frac{1 - \exp(-2\pi i l)}{1 - \exp(-2\pi i l/r)} \left|x^l \,(\mathrm{mod}\, N)\right\rangle \quad (27)$$

$$= \frac{1}{r} \sum_{l=0}^{r-1} r\ \delta_{0,l}\ \left|x^l \,(\mathrm{mod}\, N)\right\rangle \quad (28)$$

$$= |1\rangle \quad (29)$$

Here I have used the geometric series identity

$$\sum_{n=0}^{N-1} a^n = \frac{1 - a^N}{1 - a},$$

valid for $a \neq 1$.

So what we do is to follow the phase estimation algorithm with

$$|\Psi_{in}\rangle = |0\rangle_1 \otimes |0\rangle_2 \otimes |0\rangle_3 \otimes \cdots |0\rangle_t \otimes |0\rangle_1 \otimes |0\rangle_2 \otimes |0\rangle_3 \otimes \cdots |1\rangle_n$$

$$= |0\rangle_1 \otimes |0\rangle_2 \otimes |0\rangle_3 \otimes \cdots |0\rangle_t \otimes \frac{1}{\sqrt{r}} \left(\sum_{s=0}^{r-1} |u_s\rangle\right)$$

as the input state. This does not require knowledge of r. Since the algorithm, being a unitary transformation, is linear, the resulting final state is

$$|\Psi_{out}\rangle = \frac{1}{\sqrt{r}} \sum_{s=0}^{r-1} |\phi_s\rangle \otimes |u_s\rangle$$

and an MCB on the first t qubits yields the result $\phi_s = s/r$ to an accuracy of t bits, but each value of s has equal probability in the measurement. Since $0 \leq s \leq r-1$, the probability of a given value of s occurring in the result is $1/r$. It may therefore appear that we are no further along in our quest to determine r. Actually, we are almost done. We know that $0 \leq \phi = r/s < 1$. Also, we know that s and r are integers. We take the best rational approximation to ϕ, which is the continued fraction expansion of ϕ and a set of R repetitions yields $s_1/r, s_2/r, ..., s_R/r$.

I will skip the discussion of the continued fraction expansion, but a simple example should make clear what can be done. Say we wish to factor the number 15. We choose at random the number 8. We first compute $\gcd(15, 8) = 1$. This computation does not produce a factor. Since

$$\begin{aligned} 8^2 &= 8 \times 8 = 64 = 4 \mod 15 && (30) \\ 8^3 &= 8^2 \times 8 = 4 \times 8 = 32 = 2 \mod 15 && (31) \\ 8^4 &= 8^3 \times 8 = 2 \times 8 = 16 = 1 \mod 15, && (32) \end{aligned}$$

the period of 8 is 4.

We now apply the phase estimation algorithm to the operator $U_8 |y\rangle = |8y \pmod{15}\rangle$ using the input state

$$\begin{aligned} |\Psi_{in}\rangle &= |0\rangle_1 \otimes |0\rangle_2 \otimes |0\rangle_3 \otimes \cdots |0\rangle_t \otimes |1\rangle \\ &= |0\rangle_1 \otimes |0\rangle_2 \otimes |0\rangle_3 \otimes \cdots |0\rangle_t \otimes \\ &= \frac{1}{\sqrt{r}} \sum_{s=0}^{3} e^{-2\pi i s/4} \ |\phi_s\rangle |u_s\rangle . \end{aligned}$$

The output state is

$$|\Psi_{out}\rangle = \frac{1}{\sqrt{r}} \sum_{s=0}^{3} e^{-2\pi i s/4} \ |\phi_s\rangle |u_s\rangle$$

The MCB on the first register produces $0, 1/4, 1/2$, or $3/4$, each with probability $1/4$. The point is that we can check the denominators in turn, verifying that $8^4 = 1 \mod 15$, while $8^2 \neq 1 \mod 15$. Hence the order of $8 \mod 15$ is $r = 4$.

Let's finish the job. Since $8^4 = 1 \mod 15$, $8^4 - 1 = 0 \mod 15$, and $(8^2 - 1)(8^2 + 1) = 0 \mod 15$, and so either $8^2 - 1 = 63 = 3 \mod 15$ or $8^2 + 1 = 65 = 5 \mod 15$ or both is a factor of 15. Checking these possibilities, we find that both are factors, and we are, at long last, finished.

Lectures on Quantum Computing
Physics 709
Lecture 16: The RSA Code

1 Cryptography using Number Theory

Mathematically, a code can be described quite simply. M, the actual message, is a bit string. The encoding method (key) is a function $f(M)$ applied by the sender. $f(M)$ is the string that is actually sent. The decoding method is the function f^{-1} applied by the receiver. She reconstructs M according to the formula $f^{-1}(f(M)) = M$.

The codes used for centuries were private key codes. For example the classical substitution codes where, for example, a is substituted by r,b by w, c by u and so on, is a private key method in which the encoding method f is defined by a permutation P. Anyone who knows P can easily invert the function f and decode the message, so P must be kept secret (private).

The idea of public key cryptography is that the encoding (or encryption) method f is actually known to everyone (public) but the decoding (or decryption) method is not. This greatly simplifies communication, since we need not send out the encoding method every time someone wants to send a coded message. For the same reason, public key cryptography is also in practice more secure. Your QR code is encrypted in this way when it is sent and the encryption method can even be hard-wired. However, only the bank can decrypt it. For a public key code everyone knows f, but *we must ensure that finding f^{-1} is hard.*

The problem of factoring large numbers serves as the basis for the most widely used public key schemes.

The most popular such scheme is the RSA (Rivest–Shamir–Adleman) code first invented by Cocks, which works as follows.

1.1 Encryption method

I am the codemaster. I choose 2 large prime numbers p and q. By "large", I mean $p, q \sim 10^{100}$. Their product $N = pq$ is a very large composite number: $N \tilde{} O(10^{200})$. Also $\phi(N) = (p-1)(q-1) \tilde{} 10^{200}$. I choose a number c, where $c < \phi(N)$, and c does not have any common factors with $\phi(N)$: $\gcd(c, \phi(N)) = 1$. Thus $c \in Z^*_{\phi(N)}$. I now reveal the values of N and c, but *not* the values of p, q or $\phi(N)$. This defines the encoding method f. You have a message, maybe your credit card number a, that you wish to convey to me. It is important that $\gcd(a, N) = 1$, but this is overwhelmingly likely. You compute a new number

$$b = f(a) = a^c \pmod{N}. \tag{1}$$

You send it to me. Someone else might intercept the signal b and we do not want them to able to find a, that is, we do not want them to possess the function f^{-1} for which $a = f^{-1}(b)$.

As a side note, the computation of b is efficient in spite of the fact that computing a^c would appear to involve $O(N)$ multiplications. What we do is to compute $a^2 \pmod N$, then $(a^2)^2 = a^4 \pmod N$, then $(a^4)(a^2) = a^8 \pmod N$ and so on. Then if c is not a power of 2, we fix things up when we have computed a number a^x, where x is close to c. So we only need $O(\log_2 N)$ multiplications, not $O(N)$.

1.2 Decryption Method

For decryption, we need one additional mathematical fact. We saw in our lecture about number theory that

$$a^{p-1} = 1 \pmod p$$

for any integer not divisible by p. This means that also

$$\begin{aligned} \left(a^{q-1}\right)^{p-1} &= 1 \pmod p \text{ or} \\ \left(a^{q-1}\right)^{p-1} - 1 &= 0 \pmod p \end{aligned}$$

since a^{q-1} is also not divisible by p. Similarly

$$\begin{aligned} \left(a^{p-1}\right)^{q-1} &= 1 \pmod q \text{ or} \\ \left(a^{p-1}\right)^{q-1} - 1 &= 0 \pmod q. \end{aligned}$$

But p and q are distinct primes, so if $(a^{p-1})^{q-1} - 1$ is divisible by both p and q then

$$\left(a^{p-1}\right)^{q-1} - 1 = 0 \pmod{pq}.$$

Hence

$$a^{s(p-1)(q-1)} = 1^s = 1 \pmod{pq}$$

for any integer k, and multiplying by a we have

$$\begin{aligned} a^{1+k(p-1)(q-1)} &= a \pmod{pq} \text{ or} \\ a^{1+k\phi(N)} &= a \pmod N \end{aligned}$$

(The previous paragraph was inadvertently omitted when I gave the lecture. All is now OK.)

Now we can get to the decryption procedure.

We know that there is a unique number $d = c^{-1}$ in the group $Z^*_{\phi(N)}$ such that $1 < d < \phi(N)$ and

$$cd = 1 \pmod{\phi(N)}, \tag{2}$$

which may also be stated that d is the multiplicative inverse of $c \pmod{\phi(N)}$: $dc = 1 \pmod{\phi(N)}$. The codemaster (me) can find d because he knows $\phi(N) = (p-1)(q-1)$ but no one else can, because they don't know p and q. This equation can also be written as

$$cd = 1 + k\phi(N), \tag{3}$$

where k is some integer.

I now wish to decode your message. To do so, I must turn the number $b = f(a)$ into $a = f^{-1}(b)$. I do this by computing $b^d \pmod{N}$:

$$\begin{aligned} b^d \pmod{N} &= a^{cd} \pmod{N} \\ &= a^{1+k\phi(N)} \pmod{N} \\ &= a \pmod{N} \end{aligned} \tag{4}$$

Thus $f^{-1}(b) = b^d$.

The conclusion of all this is very simple: *if we can invent an algorithm for solving the order-finding problem efficiently, then we can break the RSA code.* This is the single biggest reason that quantum computing became an active field of research.

Lectures on Quantum Computing
Physics 709

Lecture 17: Grover Algorithm

1 Quantum Searching

Another application of quantum computing that has attracted attention is search algorithms. The standard example is searching a database of elements (names, for example) arranged in random order. Imagine a phone book that is not alphabetized, and I am looking, perhaps somewhat narcissistically, for the name Joynt. If I have a computer, I need some program that can recognize "Joynt". Also it must be able to process each item sequentially. In a completely unstructured database, the time it takes to find "Joynt" is clearly roughly the number of entries in the database, since the only thing I can do is to scan down until I find it. In other words the classical time $T_{cl} = O(N)$ where N is the number of items in the datbase. Obviously, if the database is structured in some way, alphabetized for example, the time can be shorter. There has been quite a bit of research on modifications of the Grover algorithm for structures databases. In this course,we'll only deal with the unstructured case.

2 Oracles, classical and quantum

Now let us try to quantify the problem a little more precisely. I have a long $[O(N)]$ list of distinct n-digit binary numbers x_i. $N < 2^n$ so all of the numbers can be encoded in states $|x_i\rangle$. The desired number x_J occurs only once. Our recognition software takes the form of a function $f(x)$ and a subroutine that gives $f(x_J) = 1$ and $f(x) = 0$ if $x \neq x_J$. Clearly, we need to have such a program in any search algorithm. It is called an oracle. A classical oracle may be thought of as a solution to a decision problem. Each input leads to a yes (1) or no (0) answer. A naive search would just try the x_i in sequence until we find $f(x_i) = 1$.

A quantum oracle must be implemented by a unitary transformation U_f. The function $f(x)$ above is clearly not invertible [$f^{-1}(0)$ is not unique] and therefore will not work, because a unitary transformation is invertible by definition. However, a simple modification is as follows. The information in the number x is encoded in n qubits, where $n = \log_2 N$. (If N is not a power of 2, we can always add some zeros to make it up to the next power of 2.) In addition, we have an indicator qubit $|q\rangle$. The transformation U_f is then given by

$$U_f \ |x\rangle\,|q\rangle = |x\rangle \ |q + f(x)\rangle\,, \tag{1}$$

where the addition is mod 2, as usual. The unitary transformation U_f is a controlled-NOT, but where there are many control bits, since x is an n-digit number. In spin language, this is just a conditional spin flip. It is always convenient to take

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

for then the result of the oracle U_f is just

$$U_f\left[|x\rangle \frac{1}{\sqrt{2}}\left(|0\rangle - |1\rangle\right)\right] = (-1)^{f(x)}\left[|x\rangle \frac{1}{\sqrt{2}}\left(|0\rangle - |1\rangle\right)\right]. \tag{3}$$

Thus the oracle produces a sign change if $|x\rangle$ is true $[f(x) = 1]$, and no change if $|x\rangle$ is false $[f(x) = 0]$. Note that $|q\rangle$ is unchanged if we make this choice. In subsequent discussions, we will omit the indicator qubit, and just take the operation of O as a sign change for $|x\rangle : O|x\rangle = (-1)^{f(x)}|x\rangle$. Note that O itself, acting on an n-qubit state, is also unitary. So our computer will be an n-qubit device.

The question of whether there is an *efficient* quantum oracle is dependent on the problem in question. Clearly, if there is no efficient oracle, then we cannot achieve any quantum speedup. We will leave this question aside for the purposes of this course, and just consider the consequences of assuming that such an efficient oracle exists. The cost of an algorithm is the total number of calls to the oracle, both in the classical and in the quantum case.

3 The Grover algorithm for 1 object

To illustrate the algorithm, we will assume that there is only one Joynt in the book, only one value of x for which $f(x) = 1$: $f(x_J) = 1$, but $f(x) = 0$ if $x \neq x_J$.

We start with the computer in the state

$$|\Psi_0\rangle = |0\rangle = |0\rangle_1 \otimes |0\rangle_2 \otimes |0\rangle_3 \otimes \cdots \otimes |0\rangle_n \tag{4}$$

and apply the Hadamard to each qubit, a product of single-qubit gates which I will write in the long notation $H^{\otimes n}$. This gives

$$|\Psi_1\rangle = H^{\otimes n}|\Psi_0\rangle = H^{\otimes n}|0\rangle = \frac{1}{\sqrt{N}}\sum_{x=0}^{N-1}|x\rangle \tag{5}$$

$$= \frac{1}{\sqrt{N}}\sum_{x=0, x\neq x_J}^{N-1}|x\rangle + \frac{1}{\sqrt{N}}|x_J\rangle. \tag{6}$$

$|\Psi_1\rangle$ is a very important state that we have met many times before, the equally weighted combination of all states in the computational basis. Once again, we are using quantum parallelism.

The aim is to use the oracle to "steer" the state $|\Psi_1\rangle$ to the state $|x_J\rangle$. First we apply the oracle, which gives

$$\begin{aligned} |\Psi_2\rangle &= O\,|\Psi_1\rangle & (7)\\ &= \frac{1}{\sqrt{N}}\sum_{x=0}^{N-1}(-1)^{f(x)}\;|x\rangle & (8)\\ &= \frac{1}{\sqrt{N}}\left[\sum_{x=0,x\neq x_J}^{N-1}|x\rangle - |x_J\rangle\right] & (9)\\ &= |\Psi_1\rangle - \frac{2}{\sqrt{N}}\,|x_J\rangle\,. & (10)\end{aligned}$$

What we have done so far is just to tag the desired item with a minus sign. But of course measuring now would do no good. Because the probability of measuring a state is the square of its coefficient, all states are equally likely to be observed. We need to amplify the coefficient of the x_J state.

We achieve this by an iterative procedure. The iteration consists of three additional steps. First, another application of $H^{\otimes n}$, then the conditional phase shift operator $P\,|x\rangle = -(-1)^{\delta_{x,0}}\,|x\rangle = 2\,|0\rangle\,\langle 0| - I$, where I is the identity and finally $H^{\otimes n}$ once more. This sequence can be written simply as

$$\begin{aligned} H^{\otimes n}PH^{\otimes n} &= H^{\otimes n}\left(2\,|0\rangle\,\langle 0| - I\right)H^{\otimes n} & (11)\\ &= 2H^{\otimes n}\,|0\rangle\;\;\langle 0|\,H^{\otimes n} - \left(H^{\otimes n}\right)^2 & (12)\\ &= 2\,|\Psi_1\rangle\,\langle\Psi_1| - I. & (13)\end{aligned}$$

I have used the facts that $H^{\otimes n}\,|0\rangle = |\Psi_1\rangle$ and so $\langle 0|\,H^{\otimes n} = \langle\Psi_1|$ as well, and $\left(H^{\otimes n}\right)^2 = \left(H_1\right)^2 \otimes \left(H_2\right)^2 \cdots \left(H_n\right)^2 = I \otimes I \cdots = I$. (Recall that

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

and squaring this gives $H^2 = I$.

Thus the complete Grover iteration G is

$$G = H^{\otimes n}PH^{\otimes n}O = \left[2\,|\Psi_1\rangle\,\langle\Psi_1| - I\right]O. \qquad (14)$$

The strategy of the Grover algorithm is to apply G repeatedly to the state $|\Psi_1\rangle$, getting closer to the state $|x_J\rangle$ with each iteration.

The analysis may be simplified by defining the normalized state $|\alpha\rangle$ that is equally weighted over all states except $|x_J\rangle$:

$$|\alpha\rangle = \sqrt{\frac{1}{N-1}}\sum_{x=0,x\neq x_J}^{N-1}|x\rangle\,. \qquad (15)$$

It is orthogonal to $|x_J\rangle$:

$$\langle\alpha|x_J\rangle = 0 \qquad (16)$$

and satisfies

$$|\Psi_1\rangle = \frac{\sqrt{N-1}}{\sqrt{N}}|\alpha\rangle + \frac{1}{\sqrt{N}}|x_J\rangle \quad (17)$$

$$\equiv \cos\frac{\theta}{2}|\alpha\rangle + \sin\frac{\theta}{2}|x_J\rangle\,, \quad (18)$$

where the last equation serves to define the angle θ :

$$\sin\frac{\theta}{2} \equiv \frac{1}{\sqrt{N}}, \quad (19)$$

which for large N is

$$\theta \approx \frac{2}{\sqrt{N}} \quad (20)$$

to an extremely good approximation.

Now let us look at the effect of the first iteration:

$$G|\Psi_1\rangle = H^{\otimes n}PH^{\otimes n}O|\Psi_1\rangle \quad (21)$$

$$= [2|\Psi_1\rangle\langle\Psi_1| - I]\left[\cos\frac{\theta}{2}|\alpha\rangle - \sin\frac{\theta}{2}|x_J\rangle\right] \quad (22)$$

$$= \left\{2\left[\cos\frac{\theta}{2}|\alpha\rangle + \sin\frac{\theta}{2}|x_J\rangle\right]\left[\cos\frac{\theta}{2}\langle\alpha| + \sin\frac{\theta}{2}\langle x_J|\right] - I\right\}\times \quad (23)$$

$$\left[\cos\frac{\theta}{2}|\alpha\rangle - \sin\frac{\theta}{2}|x_J\rangle\right] \quad (24)$$

$$= 2\left[\cos\frac{\theta}{2}|\alpha\rangle + \sin\frac{\theta}{2}|x_J\rangle\right]\left[\cos^2\frac{\theta}{2} - \sin^2\frac{\theta}{2}\right] - \left[\cos\frac{\theta}{2}|\alpha\rangle - \sin\frac{\theta}{2}|x_J\rangle\right] \quad (25)$$

$$= 2\cos\theta\,\left[\cos\frac{\theta}{2}|\alpha\rangle + \sin\frac{\theta}{2}|x_J\rangle\right] - \left[\cos\frac{\theta}{2}|\alpha\rangle - \sin\frac{\theta}{2}|x_J\rangle\right] \quad (26)$$

$$= (2\cos\theta - 1)\cos\frac{\theta}{2}|\alpha\rangle + (2\cos\theta + 1)\sin\frac{\theta}{2}|x_J\rangle \quad (27)$$

$$= \cos\frac{3\theta}{2}|\alpha\rangle + \sin\frac{3\theta}{2}|x_J\rangle\,. \quad (28)$$

The last step follows from

$$(2\cos\theta - 1)\cos\frac{\theta}{2} = \cos\theta\cos\frac{\theta}{2} + (\cos\theta - 1)\cos\frac{\theta}{2} \quad (29)$$

$$= \cos\theta\cos\frac{\theta}{2} - 2\sin^2\frac{\theta}{2}\cos\frac{\theta}{2} \quad (30)$$

$$= \cos\theta\cos\frac{\theta}{2} - \sin\frac{\theta}{2}\sin\theta \quad (31)$$

$$= \cos\left(\frac{3\theta}{2}\right), \quad (32)$$

and similarly for the other term. Thus $|\Psi_1\rangle = \cos\frac{\theta}{2}|\alpha\rangle + \sin\frac{\theta}{2}|x_J\rangle$ is transformed into $G|\Psi_1\rangle = |\Psi_2\rangle = \cos\frac{3\theta}{2}|\alpha\rangle + \sin\frac{3\theta}{2}|x_J\rangle$.

We have succeeded in our goal, which was to to amplify the coefficient in front of $|x_J\rangle$.

Each iteration adds θ to the angle, and so after k iterations, we have

$$G^k|\Psi_1\rangle = \cos\frac{(2k+1)\theta}{2}|\alpha\rangle + \sin\frac{(2k+1)\theta}{2}|x_J\rangle. \tag{33}$$

Our aim is to produce $|x_J\rangle$ as output. Hence we want

$$\frac{(2k+1)\theta}{2} = \frac{\pi}{2}, \tag{34}$$

whic would give

$$G^k|\Psi_1\rangle = |x_J\rangle$$

So we choose k to be

$$k = \frac{\pi}{2\theta} - \frac{1}{2}\theta \approx \frac{\pi}{4}N^{1/2} + O\left(N^{-1/2}\right).$$

We now do a measurement and find x_J with high probability. Hence we only need $O\left(\sqrt{N}\right)$ calls to the oracle, rather than $O(N)$ as needed by the classical algorithm. So the quantum run time is $T_q = O\left(\sqrt{N}\right)$. This is a considerable speedup. Curiously, unlike most iteration procedures, continuing beyond $\pi\sqrt{N}/4$ iterations worsens the result.

4 The Grover algorithm for many objects

That is the most basic algorithm. A simple generalization is to the case where there are M Joneses in the book. Then $f(x) = 1$ if $|x\rangle \in S$, where S is a subset of $\{|x_i\rangle\}_{i=1,2,\ldots,N}$ and S has M members. $f(x) = 0$ if $|x\rangle \notin S$. Then we define

$$|\alpha\rangle = \sqrt{\frac{1}{N-M}}\sum_{x\notin S}|x\rangle \tag{35}$$

and

$$|\beta\rangle = \sqrt{\frac{1}{M}}\sum_{x\in S}|x\rangle \tag{36}$$

with

$$\langle\alpha|\beta\rangle = 0 \tag{37}$$

so that the two states are orthogonal. The algorithm proceeds exactly as for $M = 1$. The equally weighted state can still be written as

$$|\Psi_1\rangle = H^{\otimes n}|0\rangle = \frac{\sqrt{N-M}}{\sqrt{N}}|\alpha\rangle + \sqrt{\frac{M}{N}}|\beta\rangle \quad (38)$$

$$= \cos\frac{\theta}{2}|\alpha\rangle + \sin\frac{\theta}{2}|\beta\rangle \quad (39)$$

so for this case we have

$$\sin\frac{\theta}{2} = \sqrt{\frac{M}{N}}. \quad (40)$$

and

$$\theta \approx 2\sqrt{\frac{M}{N}},$$

which is the main change from the 1-solution case.

Once more

$$O|\Psi_1\rangle = OH^{\otimes n}|0\rangle = \cos\frac{\theta}{2}|\alpha\rangle - \sin\frac{\theta}{2}|\beta\rangle$$

and

$$G^k|\Psi_1\rangle = \cos\frac{(2k+1)\theta}{2}|\alpha\rangle + \sin\frac{(2k+1)\theta}{2}|\beta\rangle. \quad (41)$$

Again the algorithm rotates the state by an angle θ at each iteration towards the state $|\beta\rangle$. This time the optimum number of iterations is $(\pi/4)\sqrt{N/M}$, less than in the simplest case. When we make the measurement, we obtain one of the Joneses with high probability. Which one we get is completely undetermined: each is equally likely to be observed.

5 Counting solutions

There is another intriguing extension of the algorithm. Let us suppose that we do not know whether or not there are solutions. Then we can use the phase estimation algorithm (PSA) to determine whether or not a solution exists without actually finding it, and indeed to count how many there are.

Recall that the input for the phase estimation algorithm is U, a unitary operator, and an eigenstate $|u\rangle$. The algorithm finds the eigenvalue. We set $U = G = H^{\otimes n}PH^{\otimes n}O$.

We get eigenstates as follows. We start with the easily prepared state

$$|\Psi_1\rangle = \cos\frac{\theta}{2}|\alpha\rangle + \sin\frac{\theta}{2}|\beta\rangle, \quad (42)$$

where $|\alpha\rangle$ and $|\beta\rangle$ are defined in Eqs. 35 and 36. Here $\theta = 2\sqrt{M/N}$ if there are M solutions. We do not know M or θ : they could even be 0. But we do know G. In the $\{|\alpha\rangle, |\beta\rangle, ...\}$ basis, $U = G$ is a rotation that affects only $|\alpha\rangle$ and $|\beta\rangle$ and acts as

the identity on all other states. The 2×2 submatrix for G in the $|\alpha\rangle, |\beta\rangle$ subspace is

$$G = U = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix} \tag{43}$$

and the eigenvalues ω of U satisfy

$$\begin{aligned} (\omega - \cos\theta)^2 + \sin^2\theta &= \omega^2 - 2\omega\cos\theta + 1 && (44) \\ &= (\omega - e^{i\theta})(\omega - e^{-i\theta}) && (45) \\ &= 0. && (46) \end{aligned}$$

The eigenvalue $e^{i\theta}$ belongs to the eigenvector $|a\rangle = (|\alpha\rangle - i\,|\beta\rangle)/\sqrt{2}$, while the eigenvalue $e^{-i\theta}$ belongs to the eigenvector $|b\rangle = (|\alpha\rangle + i\,|\beta\rangle)/\sqrt{2}$. So $|a\rangle$ and $|b\rangle$ are the desired eigenvectors of U. Note that

$$\begin{aligned} |\Psi_1\rangle &= \frac{\cos\theta/2}{\sqrt{2}}(|a\rangle + |b\rangle) + \frac{i\sin\theta/2}{\sqrt{2}}(|a\rangle - |b\rangle) && (47) \\ &= \frac{1}{\sqrt{2}}\left[e^{i\theta/2}|a\rangle + e^{-i\theta/2}|b\rangle\right]. && (48) \end{aligned}$$

Thus if we apply the phase estimation algorithm to $|\Psi_1\rangle$, we find $\theta/2$ or $-\theta/2$, each with probability $1/2$. Which one we find is a matter of chance, but it doesn't matter, since in either case $M = N\theta^2/4$ is the number of solutions.

In a problem where we need to actually find the solution, but do not know the number of solutions in advance, we can first do the counting algorithm to find M, then do the prescribed $(\pi/4)\sqrt{N/M}$ rotations to get a solution.

Lectures on Quantum Computing
Physics 709

Lecture 18: Optimality of Grover

1 Optimality

In this lecture we show that the Grover algorithm is pretty much optimal. The run time is measured by $t(N)$, the total number of calls to the oracle needed to find a given entry x_J in a database with N entries. For the Grover algorithm $t(N) = (\pi/4) N^{1/2} = 0.785\ N^{1/2}$. We will demonstrate that $t(N) \geq c\ N^{1/2}$ with $c = 0.324$ for any quantum algorithm that works more than half the time. Again, we stress that the prefactor is not so important. It is the power of N that matters.

I like to give this proof for a couple of reason. First, it shows that quantum computers have limits. Second, it is generally pretty hard to prove that no algorithm exists to solve a problem better than an existing algorithm, which is why we do not have a proof that $P \neq NP$. This proof does accomplish this dificult task for a very interesting case.

2 Proof strategy

We wish to prove that *no* other quantum algorithm can reduce the power of N, so we need to think about what the most general algorithm could be. We have at our disposal the oracle O_x that recognizes the sought object x_J. (I remind you that $O_x |x\rangle = |x\rangle$ if $|x\rangle \neq |x_J\rangle$ and $O_x |x\rangle = -|x\rangle$ if $|x\rangle = |x_J\rangle$.) We can apply O_x repeatedly. In between applications of O_x we can do an arbitrary unitary operation. Thus the most general algorithm is $U_t O_x U_{t-1} O_x \cdots U_1 O_x$, which leads to the state

$$|\gamma_t(x)\rangle = U_t O_x U_{t-1} O_x \cdots U_1 O_x |\Psi\rangle \tag{1}$$

after t operations. (I remind you that $|\Psi\rangle = \left(1/\sqrt{N}\right) \sum_x |x\rangle$.)

We can define the U_t any way we like, so unlike the Grover agorithm in which $U_t = 2|\Psi\rangle\langle\Psi| - I$ for all t, the U_t need not be identical. It is important to note that the U_t cannot depend on x, because any access to x comes only through the oracle. I will compare $|\gamma_t(x)\rangle$ to the state

$$|\phi_t\rangle = U_t U_{t-1} \cdots U_1 |\Psi\rangle\,, \tag{2}$$

which is the same, except without the calls to the oracle. The point here is that $|\phi_t\rangle$ knows nothing about x_J, so the distance between $|\gamma_t(x)\rangle$ and $|\phi_t\rangle$ is a measure of how

well our "steering" process relative to a process that cannot possibly go systematically toward $|x_J\rangle$.

The proof is a little complicated and depends on some vector space inequalities. First of all, we need a distance measure, also called a metric. A Hilbert space has an inner product that automatically defines a metric. The squared distance between two vectors (kets) u and v is defined by a short-hand notation $|(u-v)|^2$ in these notes. Written out in bra-ket notation it is

$$\begin{aligned} |(u-v)|^2 &= \langle (u-v) \,|\, (u-v) \rangle \\ &= \langle u|u\rangle + \langle v|v\rangle - 2\,\mathrm{Re}\,\langle u|v\rangle\,. \end{aligned}$$

If $|u-v|$ is small, then measurements on u and v should give similar results. $|u-v| = 0$ if and only if $u = v$. If u and v are unit vectors, then $\langle u|u\rangle = \langle v|v\rangle = 1$ and

$$|(u-v)|^2 = 2 - 2\,\mathrm{Re}\,\langle u|v\rangle\,.$$

Also if there is an orthonormal basis $\{|x_i\rangle\}_{i=0,\ldots,N-1}$ and $|v\rangle$ is a unit vector then

$$\sum_{i=0}^{N-1} |\langle x_i|v\rangle|^2 = 1$$

and this will be abbreviated as

$$\sum_x |\langle x|v\rangle|^2 = 1.$$

For the proof we need some auxiliary quantities. The distance between γ_t (the guided walk endpoint) and ϕ_t (the unguided walk endpoint) averaged over the unknown x, is proportional to

$$D_t = \sum_x |\gamma_t(x) - \phi_t|^2\,. \tag{3}$$

We average over x because the algorithm needs to work for all x. In addition, we shall define a couple of convenient quantities

$$E_t = \sum_x |\gamma_t(x) - x|^2 \tag{4}$$

which is the distance between γ_t (the guided walk endpoint) and the goal x averaged over x and

$$F_t = \sum_x |\phi_t - x|^2 \tag{5}$$

which is the distance between ϕ_t (the un guided walk endpoint) and the goal x averaged over x.

First note that, in order to have a probability of success greater than or equal to $1/2$ as stipulated at the beginning, we need

$$|\langle x|\gamma_t(x)\rangle|^2 \geq \frac{1}{2}, \text{ for all } x. \tag{6}$$

We could take a number smaller than $1/2$ at the cost of running the algorithm multiple times, but optimizing this parameter would change c but would not change the main conclusion.

3 Some inequalities

Eq. 6 has implications for E_t :

$$E_t = \sum_x |\gamma_t(x) - x|^2 = \sum_x [2 - 2\,\mathrm{Re}\,\langle x\,|\gamma_t(x)\rangle] \tag{7}$$

$$= \sum_x [2 - \mathrm{Re}\,|\langle x|\gamma_t(x)\rangle|] \leq \left(2 - \sqrt{2}\right) N, \tag{8}$$

since without loss of generality we may take $|\gamma_t(x)\rangle$ to be real.

On the other hand

$$F_t = \sum_x |\phi_t - x|^2\,, \tag{9}$$

and note that

$$F_t = 2N - 2\,\mathrm{Re} \sum_x \langle \phi_t\,|x\rangle\,, \tag{10}$$

because ϕ_t and x are unit vectors. Now if there is no correlation between ϕ_t and x, then the average value of $\langle \phi_t\,|x\rangle$ is zero. If we take into account Eq. 2 which allows for maximum downward motion of this quantity, then we still find

$$F_t \geq 2N - 2\sqrt{N}. \tag{11}$$

There is one further preliminary step, which is to remind ourselves of the Cauchy-Schwarz (CS) inequality:

$$\left|\vec{a}\cdot\vec{b}\right| \leq |\vec{a}|\,\left|\vec{b}\right|\,, \tag{12}$$

which can also take the forms

$$\left|\vec{a}+\vec{b}\right|^2 \geq |\vec{a}|^2 + \left|\vec{b}\right|^2 - 2\,|\vec{a}|\,\left|\vec{b}\right| \tag{13}$$

and

$$\left|\vec{a}+\vec{b}\right|^2 \leq |\vec{a}|^2 + \left|\vec{b}\right|^2 + 2\,|\vec{a}|\,\left|\vec{b}\right|\,. \tag{14}$$

4 Bounds on $\mathbf{D}_t$

Now we are coming to the heart of the proof, which is to put upper and lower bounds on D_t.

First the lower bound:

$$D_t = \sum_x |\gamma_t(x) - x + x - \phi_t|^2 \tag{15}$$

$$\geq \sum_x |\gamma_t(x) - x|^2 - 2\sum_x |\gamma_t(x) - x|\,|x - \phi_t| + \sum_x |x - \phi_t|^2 \tag{16}$$

$$= E_t + F_t - 2\sum_x |\gamma_t(x) - x|\,|x - \phi_t|\,. \tag{17}$$

We can define a dot product between the N-component vectors $\vec{a}$ and $\vec{b}$ whose components are given by $a_x = |\gamma_t(x) - x|$ and $b_x = |x - \phi_t|$ (*not* the Hilbert space inner product) and apply the CS inequality:

$$\sum_x |\gamma_t(x) - x|\,|x - \phi_t| \le \sqrt{\sum_x |\gamma_t(x) - x|^2}\sqrt{\sum_x |x - \phi_t|^2} = \sqrt{E_t}\sqrt{F_t}, \tag{18}$$

so

$$D_t \ge E_t + F_t - 2\sqrt{E_t}\sqrt{F_t} = (\sqrt{F_t} - \sqrt{E_t})^2. \tag{19}$$

Substituting in the inequalities (Eqs. 11 and 7) for F_t and E_t derived earlier, we find

$$\begin{aligned}
D_t &\ge \left[\sqrt{2N - 2\sqrt{N}} - \sqrt{2N - \sqrt{2}N}\right]^2 && (20)\\
&= 2N - 2\sqrt{N} + 2N - \sqrt{2}N - 2\sqrt{2N - 2\sqrt{N}}\sqrt{2N - \sqrt{2}N} && (21)\\
&= (4 - \sqrt{2})N - 2\sqrt{2}N(1 - 1/\sqrt{N})^{1/2}(2 - \sqrt{2})^{1/2} && (22)\\
&= \left[4 - \sqrt{2} - 2\sqrt{2}(2 - \sqrt{2})^{1/2}\right] N + O(N^{1/2}) && (23)\\
&= 0.421\,N + O(N^{1/2}). && (24)
\end{aligned}$$

Now for the upper bound. We shall show that $D_t \le 4t^2$, by induction on t. The first step is therefore to show that $D_1 \ge 4$.

$$\begin{aligned}
D_1 &= \sum_x |\gamma_1(x) - \phi_1|^2\\
&= \sum_x \langle (U_1 O_x - U_1)\,\Psi|\,((U_1 O_x - U_1)\,\Psi)\rangle\\
&= \sum_x \langle (O - I)\,\Psi|\,((O - I)\,\Psi)\rangle
\end{aligned}$$

but

$$|(O - I)\,\psi\rangle = -\frac{2}{\sqrt{N}}\,|x\rangle\,,$$

using the definitions of O_x and $|\Psi\rangle$, so $D_t = 4$. The second step in the induction proof is that $D_t \le 4t^2$ implies $D_{t+1} \le 4(t+1)^2$.

$$\begin{aligned}
D_{t+1} &= \sum_x |O_x\gamma_t(x) - \phi_t|^2 && (25)\\
&= \sum_x |[O_x\gamma_t(x) - O_x\phi_t] + [O_x\phi_t - I\phi_t]|^2 && (26)\\
&\le \sum_x \left\{|O_x\gamma_t(x) - O_x\phi_t|^2 + |O_x\phi_t - I\phi_t|^2 + 2\,|O_x\gamma_t(x) - O_x\phi_t|\,|O_x\phi_t - I\phi_t|\right\}, && (27)
\end{aligned}$$

where the last line is the triangle inequality. Now note that because of unitarity we have $|O_x\gamma_t(x) - O_x\phi_t|^2 = |O_x[\gamma_t(x) - \phi_t]|^2 = |\gamma_t(x) - \phi_t|^2$. The definition of O_x gives $O_x\phi_t - I\phi_t = (O_x - I)\,\phi_t = -2\,\langle x\,|\phi_t\rangle\,|x\rangle$. Hence

$$D_{t+1} \le \sum_x \left\{|\gamma_t(x) - \phi_t|^2 + 4\,|\langle x\,|\phi_t\rangle|^2 + 4|\gamma_t(x) - \phi_t|\,|\langle x\,|\phi_t\rangle|\right\}. \tag{28}$$

The completeness relation is

$$\sum_x |\langle x \,|\phi_t\rangle|^2 = 1, \tag{29}$$

which allows us to simplify the second term. Now we apply the Cauchy-Schwartz inequality to the vectors

$$\sum_x |\gamma_t(x) - \phi_t| \,|x\rangle \tag{30}$$

and

$$|\phi_t\rangle \tag{31}$$

to get

$$\sum_x |\gamma_t(x) - \phi_t| \,|\langle x \,|\phi_t\rangle| \le \sqrt{\sum_x |\gamma_t(x) - \phi_t|^2}\, |\phi_t|^2 = \sqrt{D_t}. \tag{32}$$

Now recall that

$$D_t = \sum_x |\gamma_t(x) - \phi_t|^2 \tag{33}$$

and this, together with the inductive hypothesis, all gives

$$\begin{aligned} D_{t+1} &\le \sum_x \left\{ |\gamma_t(x) - \phi_t|^2 + 4\,|\langle x \,|\phi_t\rangle|^2 + 4|\gamma_t(x) - \phi_t| \,|\langle x \,|\phi_t\rangle| \right\} && (34)\\ &= D_t + 4\sqrt{D_t} + 4 && (35)\\ &\le 4t^2 + 8t + 4 && (36)\\ &= 4(t+1)^2, && (37) \end{aligned}$$

which completes the inductive proof of the inequality $D_t \le 4t^2$.

5 Conclusion

Putting all this together, we have

$$0.421N \le D_t \le 4t^2 \tag{38}$$

which implies

$$t \ge 0.324\ N^{1/2}. \tag{39}$$

Any algorithm requires at least $0.324\ N^{1/2}$ oracle calls. The Grover algorithm requires $t_G = (\pi/4)\, N^{1/2} = 0.785\ N^{1/2}$ calls. Note that $t < t_G$, as it must be.

So, up to a constant factor, the Grover algorithm is optimal.

Lectures on Quantum Computing
Physics 709

Lecture 19: Classical Error Correction

1 Introduction

Quantum computers are particularly susceptible to random errors. Here is an example. After a Hadamard gate performed on the $|0\rangle$ state of a spin qubit, the new state is

$$|\psi(t=0)\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle).$$

Now suppose that a random electromagnetic signal acts on the qubit for a very short time δt. Its magnetic field will have a component in the B_z in the z-direction. So there is a Hamiltonian $H(t) = -\mu B_z \sigma_z$, where μ is the gyromagnetic ratio. The corresponding unitary operator is $U = \exp(i\mu B_z t/\hbar) = \exp(i\omega t) = \cos\omega t + i\sigma_z \sin\omega t$, where $\hbar\omega = \mu B_z$. After the short time δt, the state is

$$|\psi(t=\delta t)\rangle = \frac{1}{\sqrt{2}}\left(e^{i\omega(\delta t)}|0\rangle + e^{-i\omega(\delta t)}|1\rangle\right) \quad (1)$$

$$= \frac{1}{\sqrt{2}} e^{-i\omega(\delta t)}\left(e^{2i\omega(\delta t)}|0\rangle + |1\rangle\right) \quad (2)$$

The relative phase of the $|0\rangle$ and $|1\rangle$ states is not 1 anymore, it is $\exp[2i\omega(\delta t)]$ and ω and δt are unknown and therefore random numbers.

In a quantum computation *everything* depends on the relative phase of the components. In fact this is what distinguishes quantum computation from classical computation. This random process therefore represents an error that will render our result meaningless. Notice that this particular error is one that can't occur in a classical bit, where there is no concept of relative phase.

The continuous character of the quantum information renders it more vulnerable to errors than classical information. For this reason, even after the discovery of the Shor algorithm, there was little effort for several years to construct quantum computers. This was because physicists believed that the qubits would not be able to remain in their quantum states long enough to perform interesting calculations. This changed in the late 1990s with the discovery of quantum error correction.

We will deal with different types of errors in turn. The first, in the 3-qubit bit flip code, will be errors of the type that flip a qubit, that is, changes $|0\rangle$ into $|1\rangle$ and $|1\rangle$ into $|0\rangle$. In other words, this error changes $|\psi\rangle$ into $X|\psi\rangle$. The second will be qubit flips *and* phase errors that changes $|0\rangle$ into $|0\rangle$ and $|1\rangle$ into $-|1\rangle$. Both qubit flip and phase errors are discrete errors. Finally we will deal with continuous errors such as the one that happened in Eq. 1.

2 Classical Error Correction

Let us return to a task we have already discussed in these lectures, that of sending a classical message. For classical bits, the only possible error is a bit flip: $0 \rightarrow 1$ or $1 \rightarrow 0$. How do we protect against such errors? We do the same thing when we are speaking and are afraid that the listener may not hear us properly: we repeat ourselves. In more technical terms we use *redundancy*. Alice encodes 0 as 000 and 1 as 111 and sends a string, 3 times as long as the original string, through the noisy channel. We characterize this by saying that one "logical" bit has been encoded into 3 "physical" bits. Bob decodes by mapping 000, 100, 010, and 001 into 0 and 111, 011, 101, and 110 into 1. This is sometimes called "majority rules" since it resembles a vote. It is an example of a repetition code. This procedure clearly protects against single bit flips but not against 2 or 3 bit flips. To see this more quantitatively, let the probability of any single physical bit flipping be given by p. As for any probability $0 \leq p \leq 1$. If $p << 1$, then the channel is good but as p approaches $1/2$ then the channel is bad (noisy). In our procedure the probability of getting the wrong answer for the final logical bit is $p^3 + 3p^2(1-p)$, which is the sum of the probability of 3 flips and 2 flips. This is $O(p^2)$. The probability of getting the wrong answer if we do no encoding is $p^3 + 3p^2(1-p) + 3p(1-p)^2 = O(p)$. So if $p << 1$, as it is if the channel is not too noisy, then the encoding gains Alice and Bob quite a bit. Mathematically, the key is to change the ultimate error probability from $O(p)$ to $O(p^2)$.

Notice that error correction is costly: The message sent is 3 times as long as the uncoded message.

A very slightly different task arises if Bob wants not only to decode the message by error detection but also to send the message on. Then he corrects errors by mapping 000, 100, 010, and 001 back into 000 and 111, 011, 101, and 110 back into 111 and transmits the corrected message on. So he needs to *locate* the error and flip the appropriate bit. The same thing needs to be done if instead of communication we are doing a computation with logic bits that are encoded into physical bits at all times during the computation. One imagines stopping the computation occasionally, performing an error detection *and* correction, and then continuing. It's that task that we need to do for quantum computing.

Lectures on Quantum Computing
Physics 709
Lecture 20: Bit Flip Code and Pauli Operators

1 Bit flip error correction code

We will start our quantum error correction discussion by constructing the analog of the above classical code. Imagine a single qubit initially in the general state

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

and, after a time T elapses, this state can get corrupted by unpredictable outside influences. Alternatively, we can imagine sending a physical object, such as a photon in a definite polarization state, from one point to another. In either case, an error can occur. For the moment we need to restrict (rather unphysically) the kind of error that can occur: in this lecture we will only look at bit flip errors. They send $|0\rangle$ to $|1\rangle$ and $|1\rangle$ to $|1\rangle$.

The result is that there is a probability p that $|\psi\rangle \to \alpha\,|1\rangle + \beta\,|0\rangle = X\,|\psi\rangle$. We may or may not know p but it must be true that $p << 1$ or there is no hope for error correction. Is there some way that we can reduce the error probability in the reconstructed state to $O\left(p^2\right)$?

1.1 Naive Method

Let's first try to imitate the classical method. Use three physical qubits for the single logical bit. Map $|0\rangle \to |0\rangle\,|0\rangle\,|0\rangle$ and $|1\rangle \to |1\rangle\,|1\rangle\,|1\rangle$ so that the encoded state is

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

We have encoded one "logical" qubit into 3 "physical" qubits. This multiplies the size of our computer by 3, but this a price we have to pay. Let the noise act on this state. If we are lucky, then we get the uncorrupted state $|\Psi\rangle$ back again. This happens with probability $(1-p)^3 \approx 1-3p$ if $p << 1$.

If we are not so lucky, then the state is corrupted. The whole picture is:

$$\begin{aligned}
|\Psi\rangle &= \alpha\ |0\rangle\,|0\rangle\,|0\rangle + \beta\,|1\rangle\,|1\rangle\,|1\rangle \text{ with probability } 1-3p \\
X_1\,|\Psi\rangle &= \alpha\ |1\rangle\,|0\rangle\,|0\rangle + \beta\,|0\rangle\,|1\rangle\,|1\rangle \text{ with probability } p \\
X_2\,|\Psi\rangle &= \alpha\ |0\rangle\,|1\rangle\,|0\rangle + \beta\,|1\rangle\,|0\rangle\,|1\rangle \text{ with probability } p \\
X_3\,|\Psi\rangle &= \alpha\ |0\rangle\,|0\rangle\,|1\rangle + \beta\,|1\rangle\,|1\rangle\,|0\rangle \text{ with probability } p.
\end{aligned}$$

The right state $|\Psi\rangle$ occurs with probability $(1-p)^3 \approx 1-3p$ if $p << 1$ and the wrong states occur with probability $p\,(1-p)^2 \approx p$ if $p << 1$ but I have only given

the probabilities to order p. There are worse things that can happen with probability $O(p^2)$ and $O(p^3)$ and our method will not fix those.

Now we want to measure our 3 qubits and use the results of the measurement to correct any errors that have occurred: that is, we want to reconstruct the state $|\Psi\rangle$ from the corrupted state.

We will again try to follow the classical procedure. We perform a sequence of measurements: first measure Z_1 on qubit 1, then Z_2 on qubit 2, and then Z_3 on qubit 3 in the computational basis. This will yield one of the 8 results in the set $(\pm 1, \pm 1, \pm 1)$. Recall that the $|0\rangle$ state gives +1 and the $|1\rangle$ state gives -1.

Let's see what happens. The whole system has 3 qubits but we measure only qubit 1 to start with. We get +1 with probability

$$\begin{aligned} P_{Z_1}(+1) &= (1-3p)|\alpha|^2 + 2p|\alpha|^2 + p|\beta|^2 \\ &= (1-p)|\alpha|^2 + p|\beta|^2 \end{aligned}$$

and -1 with probability

$$P_{Z_1}(-1) = (1-p)|\beta|^2 + p|\alpha|^2 .$$

If we get +1 then the new set of states is

$$\begin{aligned} |\Psi_{1+}\rangle &= |0\rangle|0\rangle|0\rangle \text{ with probability } (1-3p) \\ |\Psi_{2+}\rangle &= |0\rangle|1\rangle|1\rangle \text{ with probability } p \\ |\Psi_{3+}\rangle &= |0\rangle|1\rangle|0\rangle \text{ with probability } p \\ |\Psi_{4+}\rangle &= |0\rangle|0\rangle|1\rangle \text{ with probability } p. \end{aligned} \tag{1}$$

while if we get -1 then the new set of states is

$$\begin{aligned} |\Psi_{1-}\rangle &= |1\rangle|1\rangle|1\rangle \text{ with probability } (1-3p) \\ |\Psi_{2-}\rangle &= |1\rangle|0\rangle|0\rangle \text{ with probability } p \\ |\Psi_{3-}\rangle &= |1\rangle|0\rangle|1\rangle \text{ with probability } p \\ |\Psi_{4-}\rangle &= |1\rangle|1\rangle|0\rangle \text{ with probability } p. \end{aligned} \tag{2}$$

We continue this by measuring qubits 2 and 3. Let's say we get the result +1,-1+1 in our measurements. The probability of this result is $p|\alpha|^2$, as we could have guessed at the start. Similar calculations would give the probabilities for the other 7 possible results. So the final state is

$$|\Psi_{+1,-1,+1}\rangle = |0\rangle|1\rangle|0\rangle ,$$

since at each stage we project a qubit into some given state. After measuring all 3 qubits, we project the total state into one of the 8 computational basis states, each occurring with some probability.

Now we would like to perform the error correction, still following the classical method. We know that the final state is $|0\rangle|1\rangle|0\rangle$. So again following the classical

protocol, it looks like qubit 2 got flipped, so we flip it back: apply X_2 which flips qubit 2 and we get $|0\rangle|0\rangle|0\rangle$. But this is NOT what we want. We were trying to get back to $|\Psi\rangle = \alpha|0\rangle|0\rangle|0\rangle + \beta|1\rangle|1\rangle|1\rangle$ but what we actually got is $|0\rangle|0\rangle|0\rangle$ and all of the quantum information in α and β has been lost. So the situation looks hopeless.

What is the problem? It comes from the quantum-mechanical fact that if we measure a result whose probability depends on a probability amplitude, then information about that amplitude is lost. Actually, we destroyed any possibility of reconstructing $|\Psi\rangle$ by our very first measurement. When we measured Z_1 the probability of getting ± 1 depended on α and information about α was lost forever.

1.2 Quantum Bit Flip Error Correction

But here's the trick. Instead of measuring Z_1, Z_2, and Z_3 we instead measure the product operators Z_1Z_2, then Z_2Z_3. Let's see what happens. We get $+1$ with probability

$$P_{Z_1Z_2}(+1) = (1-2p)$$

and -1 with probability

$$P_{Z_1}(-1) = 2p.$$

If we get $+1$ then the new set of states is

$$\begin{aligned} |\Psi_{1+}\rangle &= \alpha\ |0\rangle|0\rangle|0\rangle + \beta|1\rangle|1\rangle|1\rangle \text{ with probability } (1-3p) \\ |\Psi_{2+}\rangle &= \alpha\ |0\rangle|0\rangle|1\rangle + \beta|1\rangle|1\rangle|0\rangle \text{ with probability } p \end{aligned} \tag{3}$$

while if we get -1 then the new set of states is

$$\begin{aligned} |\Psi_{1-}\rangle &= \alpha\ |1\rangle|0\rangle|0\rangle + \beta|0\rangle|1\rangle|1\rangle \text{ with probability } p \\ |\Psi_{2-}\rangle &= \alpha\ |0\rangle|1\rangle|0\rangle + \beta|1\rangle|0\rangle|1\rangle \text{ with probability } p. \end{aligned} \tag{4}$$

Now measure Z_2Z_3. Let's look at the results for all 4 possibilities. We get $Z_1Z_2 = +1, Z_2Z_3 = +1$ with probability $1-3p$ and so on. Summarizing, we find

$$\begin{aligned} |\Psi_{++}\rangle &= \alpha\ |0\rangle|0\rangle|0\rangle + \beta|1\rangle|1\rangle|1\rangle \text{ with probability } 1-3p \\ |\Psi_{+-}\rangle &= \alpha\ |0\rangle|0\rangle|1\rangle + \beta|1\rangle|1\rangle|0\rangle \text{ with probability } p \\ |\Psi_{-+}\rangle &= \alpha\ |1\rangle|0\rangle|0\rangle + \beta|0\rangle|1\rangle|1\rangle \text{ with probability } p \\ |\Psi_{-+}\rangle &= \alpha\ |0\rangle|1\rangle|0\rangle + \beta|1\rangle|0\rangle|1\rangle \text{ with probability } p. \end{aligned}$$

We note that $|\Psi_{++}\rangle = |\Psi\rangle$, $|\Psi_{+-}\rangle = X_3|\Psi\rangle$, etc. Our calculations give the following table:

SYNDROME	$+1,+1$	$+1,-1$	$-1,+1$	$-1,-1$
PROBABILITY	$1-3p$	p	p	p
FINAL STATE	$\|\Psi_c\rangle$	$\|X_3\Psi_c\rangle$	$\|X_1\Psi_c\rangle$	$\|X_2\Psi_c\rangle$
CORRECTIVE ACTION	I	X_3	X_1	X_2

The other 4 states have probability $O(p^2)$ or $O(p^3)$ so they are not listed. In fact if they occur then our error correction protocol does not work. For the measurement results I have used the term "syndrome" which is standard jargon in error correction.

Inspecting the table we see that we have NOT lost the quantum information in α and β, since in all cases $|\Psi\rangle = \alpha|0\rangle|0\rangle|0\rangle + \beta|1\rangle|1\rangle|1\rangle$ is still there. In fact all we need to do to recover $|\Psi\rangle$ is to consult our measurement result. If it is $+1,+1$ we do nothing, if it is $+1,-1$ we apply an X_3 operator and the final state is $X_3X_3|\Psi\rangle = |\Psi\rangle$, if it is $-1,+1$ we apply an X_1 operator and the final state is $X_1X_1|\Psi\rangle = |\Psi\rangle$,if it is $-1,-1$ we apply an X_2 operator and the final state is $X_2X_2|\Psi_c\rangle = |\Psi\rangle$.

We have performed a successful error correction! There are boxes with probability $O(p^2)$ or $O(p^3)$ that we left out of our table. For these our procedure would fail, but if p is small enough, the possibilities are unlikely. We say that this procedure corrects all errors that have probabilities of $O(p)$.

2 General Errors

Why did the measurement of correlation functions work, and the naive imitation of the classical method did not? The point is that the errors created by the actions of X_1, X_2, or X_3 rotate the state $|\Psi\rangle$ rigidly into another subspace, and this rigid rotation does not destroy the information in α and β. Furthermore, each of the corrupted states $X_1|\Psi\rangle$, $X_2|\Psi\rangle$, and $X_3|\Psi\rangle$ is an eigenfunction of Z_1Z_2, Z_2Z_3, and measuring these operators in the corrupted states did not destroy the quantum information in α and β. To see this explicitly, compares Eqs. 1 and 2 which do not contain α and β, with Eqs. 3 and 4, which do. The remarkable thing is that there is enough information in the measurement results of Z_1 and Z_2 that we could tell which state (out of a possible 4) that we have. That information allows us to recover the original state.

2.1 Stabilizer Group

We want next to construct a code that will correct for all errors, not just bit flips. To do this we think more generally about error correction.

Let's start by looking at all these operators we have been using. They have all been products of the identity and Pauli matrices. For an n-qubit system they all look like $I_1X_2Z_3, ..., X_n$, *i.e.,* a string of n operators each drawn from the set $\{I, X, Y, Z\}$. The order does not count, since operators that act on different qubits commute. So there are 4^n operators in this set. This is a very useful collection of operators in several contexts. For one thing, all Hermitian operators H that can act on vectors in the n-qubit space can be written as a real linear combination of these operators

$$H = \sum_{a=0}^{4^n-1} h_a \lambda_a$$

where a is an n-digit base-4 number and the notation is that $\lambda_{0213} = I_1 Y_2 X_3 Z_4$ for $n = 4$, that is, I is the 0th operator, X is the 1st, Y is the 2nd, and Z is the 3rd. The h_a are an arbitrary set of *real,* not complex, numbers. Actually, the Hermitian operators form a real vector space and the λ_a are basis vectors. Like the usual Pauli matrices, the λ_a are unitary and Hermitian. The existence of such a useful set of operators is a consequence of the digital character of our quantum computer. If the dimension of the Hilbert space is not a power of 2, then there is no set of operators quite as nice.

If I take out λ_{0000} the the λ_a form a set of generators for the Lie algebra of $SU(2^n)$, by which I mean the set of all unitary operators U that can act in the n-qubit space. These unitary operators can be written as $U = e^{iH}$, where it is conventional to set $h_{0000} = 0$ in this expression, since this term only gives an overall phase. All the λ_a have the convenient property that $\lambda_a^2 = I = \lambda_{0000}$, which in turn implies that the eigenvalues of the λ_a are ± 1. Each operator (except the identity) partitions the 2^n-dimensional Hilbert space into two orthogonal subspaces, one with eigenvalue +1 and one with eigenvalue -1. Since Tr $\lambda_a = 0$ for $a \neq 0000$ and the the trace is the sum of the eigenvalues, the eigenvalues ± 1 must be 2^{n-1}-fold. So the two orthogonal subspaces each have dimension 2^{n-1}. The projector onto the subspace with eigenvalue ± 1 is $\Pi_a^{(\pm 1)} = (I + \lambda_a)/2$. Furthermore, since Pauli matrices commute or anticommute, all pairs of the λ_a's commute or anticommute: $[\lambda_a, \lambda_b] = \lambda_a \lambda_b - \lambda_b \lambda_a = 0$ or $\{\lambda_a, \lambda_b\} = \lambda_a \lambda_b + \lambda_b \lambda_a = 0$, depending on the choice of a and b.

Now if we consider the set $G_n = \{\pm \lambda_a, \pm i\lambda_a\}_{a=0,\ldots,4^n-1}$, then we have a group of transformations on the 2^n-dimensional qubit space. The group operation is matrix multiplication. G_n has 4^{n+1} elements. G_n is called the Pauli group. For example, $G_1 = \{\pm I, \pm iI, \pm X, \pm iX, \pm Y, \pm iY, \pm Z, \pm iZ\}$. We need the $\pm i$ because otherwise G_n would not be closed under multiplication: for example $XYZ = iZ^2 = iI$.

In our 3-qubit code, the error operators $X_1 I_2 I_3$, $I_1 X_2 I_3$, and $I_1 I_2 X_3$ all belong to G_n, as do the measurement operators $Z_1 Z_2 I_3$, $I_1 Z_2 Z_3$, and $Z_1 I_2 Z_3$. In fact the measurement operators together with the identity $I_1 I_2 I_3$ is closed under matrix multiplication, so it forms a group S, which is a subgroup of G_3. This subgroup has the interesting and extremely important property that all vectors in the code space $C =$ span $\{|000\rangle, |111\rangle\}$ are left invariant by the operators in S. This was precisely the property of the measurements that was crucial: it meant that the measurement did not destroy the information about α and β.

When this occurs, we say that C is the vector space stabilized by S or that S is the stabilizer of C. It is easy to see that, in any group of linear transformations on a vector space, the set of operators that leave a set of vectors invariant is a subgroup, and the set of vectors stabilized by a subgroup is itself a subspace. This can also be expressed in terms of eigenvectors and eigenvalues: C is the intersection of the eigenspaces belonging to eigenvalue 1 for all the operators in S.

Not all subgroups of G_n have a nontrivial stabilized vector subspace, but it turns out that if and only if $S \subset G_n$ is abelian and does not contain $-I$, then there is a nontrivial space of vectors stabilized by S.

Lectures on Quantum Computing
Physics 709
Lecture 21: Stabilizer Codes

1 Stabilizer Codes

How do stabilizer groups relate to error correction? Here is the idea.

We have a set of k logical qubits. For our bit-flip code $k = 1$, since we were trying to preserve a single qubit's worth of information, which amounts to 2^k complex numbers. Thus we started with

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

and w want to keep hold of α and β even after the error.

We now wish to encode this qubit or qubits in a larger set of n physical qubits. Thus the dimension of the physical Hilbert space H, the one for the states of the actual quantum computer, is 2^n. For our bit-flip code $n = 3$.

These n qubits are subject to errors $E_j \in G_n$ and we wish to protect the string of k qubits against them. Note that the E_j need not be single qubit operators, though things get complicated quickly if they are not. We choose a 2^k-dimensional code space $C \subset H$. It is a linear subspace of H and it must be 2^k-dimensional in order to hold k qubits of information. For our bit-flip code $k = 1$ and we chose $C = span\{|000\rangle, |111\rangle\}$.

We also choose a stabilizer group for C generated by $g_1, g_2, ..., g_{n-k}$. $g_i |\Psi\rangle = |\Psi\rangle$ for all $|\Psi\rangle \in C$. When I say generate, I mean that all the elements of S can be expressed as products of the G_n, and that the set $\{g_i\}_{i=1,2,...,n-k}$ is a minimal set of such operators. For our bit-flip code the generators were $Z_1Z_2I_3$ and $I_1Z_2Z_3$, and the group C was $\{I_1I_2I_3, Z_1Z_2I_3, I_1Z_2Z_3, Z_1I_2Z_3\}$ Stabilizer groups are commutative, so $[g_i, g_j] = 0$.

One advantage of choosing a stabilizer group is that we can easily construct a projector Π_C onto C. This is

$$\Pi_C = 2^{k-n} (I + g_1)(I + g_2) \cdots (I + g_{n-k}).$$

The set of independent generators should have $n-k$ elements because each projection operator $(I + g_i)/2$ reduces the dimensionality of the space by a factor of 2. To see this take a general vector $|\phi\rangle \in H$. Then $(1/2)(I + g_{n-k})|\phi\rangle$ is the projection of $|\phi\rangle$ onto the part of $|\phi\rangle$ that has eigenvalue $+1$ for g_{n-k}. $(1/4)(I + g_{n-k-1})(I + g_{n-k})|\phi\rangle$ is the projection of $|\phi\rangle$ onto the part of $|\phi\rangle$ that has eigenvalue $+1$ for g_{n-k} and $+1$ for g_{n-k-1}, and so on. At each step the dimensionality is reduced by a factor of 2. Since we need to get from a dimensionality of 2^n for H down to a dimensionality of 2^k for C, we need $n - k$ generators.

Finally we need to choose the g_i so that there is a certain pattern of commutation of the g_i with the E_j. This is the tricky part. We will elucidate it below. For now, let's just summarize a general quantum error correcting code.

1. Build in redundancy by increasing the number of qubits from k to n.

2. Choose a 2^k-dimensional subspace C of the whole 2^n-dimensional space H and write the state in that subspace.

3. Wait for the error E_j.

4. Measure the set of generators $\{g_i\}_{i=1,2,\ldots,n-k.}$

5. Use the information gained to correct the error.

Lectures on Quantum Computing
Physics 709
Lecture 22: The 5-Qubit Code

The 3-qubit code is quite limited in the kind of errors it can correct. There are many codes that do better: they can correct general single qubit errors. The ones usually presented are the 5-qubit code, since it involves the least number of qubits, the 7-qubit code since the circuits to implement the code are fairly simple, and the 9-qubit Shor code since it has a certain symmetry to it as well as some historical importance, being the first code discovered. I'm going with the 5-qubit code.

Before we start, it's interesting to note that there is a simple proof that 5 is the smallest number of qubits that can possibly work in correcting general single-qubit errors for a single logical qubit. ($k = 1$). If the code has n physical qubits and can correct general errors, then the $3n$ error operators are $X_1, Y_1, Z_1, X_2, Y_2, Z_2, X_3, ..., Z_n$. Notice that the disadvantage of increasing n is that there are more errors that can happen. On the plus side, the space H of the physical qubits is 2^n-dimensional, so we are getting more room in our vector space to take care of errors. We start with a state in the 2-dimensional code space C. If a nontrivial error occurs, then the code state is sent into one of the $3n$ orthogonal error subspaces. We need to make measurements to determine which of the $3n + 1$ subspaces we are in. (The $+1$ takes care of the no-error possibility). Each of the error subspaces (including C) is 2-dimensional. So the total Hilbert space must have at least $6n + 2$ dimensions: $2^n \geq 6n + 2$, and the smallest integer that satisfies this condition is $n = 5$ so $n \geq 5$, which completes the proof.

We start with the logical qubit so

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

and we want to protect this state against errors, so we encode it in a set of n physical qubits

$$|\psi\rangle = \alpha \left|\overline{0}\right\rangle + \beta \left|\overline{1}\right\rangle$$

and as time goes on this state can get corrupted by unpredictable outside influences. $\left|\overline{0}\right\rangle$ and $\left|\overline{1}\right\rangle$ are called the logical zero and the logical one, respectively. They span the space $C \subset H$. C is 2-dimensional and H is 32-dimesnional. Now we want to correct arbitrary single-qubit errors, of all the 15 types given above.

The complication in the 5-qubit code is that the coded state $|\psi\rangle$ is rather complicated if expressed in the computational basis. We will get to that later. It's actually easier to think about the measurement operators g_i. We define

$$\begin{aligned} g_1 &= Z_2 X_3 X_4 Z_5 \\ g_2 &= Z_3 X_4 X_5 Z_1 \\ g_3 &= Z_4 X_5 X_1 Z_2 \\ g_4 &= Z_5 X_1 X_2 Z_3. \end{aligned}$$

Note that, since $(X_4)^2 = I_4$ and operators with different subscripts commute, we have $(g_i)^2 = I_i$. Let's do a commutator:

$$\begin{aligned}
[g_1, g_2] &= Z_2X_3X_4Z_5Z_3X_4X_5Z_1 - Z_3X_4X_5Z_1Z_2X_3X_4Z_5 \\
&= Z_2Z_1\left(X_3Z_5Z_3X_5 - Z_3X_5X_3Z_5\right) \\
&= Z_2Z_1\left[(-iY_3)\left(iY_5\right) - \left(iY_3\right)(-iY_5)\right] \\
&= 0
\end{aligned}$$

and similarly $[g_i, g_j] = 0$ for all $i \neq j$. We can also verify that we cannot get -1 by multiplying together the g_i. So this set generates a stabilizer group.

Now all we need to do is to compute the commutator or anticommutator to find the β_{ij} table. $\beta_{ij} = 1$ if $[g_i, E_j] = 0$ and $\beta_{ij} = -1$ if $\{g_i, E_j\} = 0$. Carrying this out we can make the β_{ij} table.

	I	X_1	Y_1	Z_1	X_2	Y_2	Z_2	X_3	Y_3	Z_3	X_4	Y_4	Z_4	X_5	Y_5	Z_5
g_1	$+1$	$+1$	$+1$	$+1$	-1	-1	$+1$	$+1$	-1	-1	$+1$	-1	-1	-1	-1	$+1$
g_2	$+1$	-1	-1	$+1$	$+1$	$+1$	$+1$	-1	-1	$+1$	$+1$	-1	-1	$+1$	-1	-1
g_3	$+1$	$+1$	-1	-1	-1	-1	$+1$	$+1$	$+1$	$+1$	-1	-1	$+1$	$+1$	-1	-1
g_4	$+1$	$+1$	-1	-1	$+1$	-1	-1	-1	-1	$+1$	$+1$	$+1$	$+1$	-1	-1	$+1$

We chose the strange form of the g_i so that the columns are all distinct from one other, so we can determine E_j from 4 measurements.

Now that we have some understanding of how all these operators work, it is actually not hard to work out the logical 0 and 1. They are

$$\begin{aligned}
\left|\overline{0}\right\rangle &= \frac{1}{16}\left(1+g_1\right)\left(1+g_2\right)\left(1+g_3\right)\left(1+g_4\right)\left|00000\right\rangle \\
&= \Pi_{g_1}^{(+1)}\,\Pi_{g_2}^{(+1)}\,\Pi_{g_3}^{(+1)}\,\Pi_{g_4}^{(+1)}\,\left|00000\right\rangle \\
\left|\overline{1}\right\rangle &= \frac{1}{16}\left(1+g_1\right)\left(1+g_2\right)\left(1+g_3\right)\left(1+g_4\right)\left|11111\right\rangle
\end{aligned}$$

and these states form a basis for the code subspace C. Because the projections commute, $\left|\overline{0}\right\rangle$ and $\left|\overline{1}\right\rangle$ and indeed all states in C are eigenstates of all the g_i with eigenvalues $+1$. When a state in C is corrupted by an error, it rotates into one of the 15 2-dimensional subspaces $X_1C, Y_1C, ..., Z_5C$. Each of the 15 possible subspaces is an eigenspace of all 4 of the g_i. The key point is that the measurements do not disturb the coded information contained in α and β!

One should note, finally, that it is also necessary to be able to do the usual gate operations in the code space. That is, we have a logical "computational basis" $\left\{\left|\overline{0}\right\rangle, \left|\overline{1}\right\rangle\right\}$ and we need to be able to do single qubit operations such as a bit flip which acts as $\overline{X}\left|\overline{0}\right\rangle = \left|\overline{1}\right\rangle$ and $\overline{X}\left|\overline{1}\right\rangle = \left|\overline{0}\right\rangle$ on the code state computational basis, and 2-qubit operations in the 2-qubit tensor product space, *e.g.*, the CNOT. For the 3-qubit error correction code, it is not too hard to see what to do. We need $\overline{X}\left|000\right\rangle = \left|111\right\rangle$

and $\overline{X}\,|111\rangle = |000\rangle$ and obviously $\overline{X} = X_1X_2X_3$ fits the bill. It is not quite so obvious, but we can use $\overline{Z} = Z_1Z_2Z_3$ as well. It is necessary that that logical operation has the desired effect on the basis states, but one must also check that any logical operation $\overline{O}$ commutes with all the elements of the stabilizer group: $\left[\overline{O}, g_i\right] = 0$ for all i. Only then can we be certain that error correction works on the state $|\psi\rangle$ but also on the state $\overline{O}_1\overline{O}_2\cdots\overline{O}_N\ |\psi\rangle$ that might arise after N steps in the algorithm. For the 3-qubit code, the $\overline{O}$ operators are quite easy to construct, and it is also easy to see that, for example $\left[\overline{X}, g_1\right] = [X_1X_2X_3, Z_1Z_2] = 0$,and $\left[\overline{Z}, g_1\right] = [Z_1Z_2Z_3, Z_1Z_2] = 0$. But for the 5-qubit code the logic operators are unwieldy, and so I'll avoid going into them. The 7-qubit code has simpler logic operators which accounts for its popularity.

Not all quantum error correction codes work like this. Those that do are called "stabilizer codes". Pretty much all the codes you will find in textbooks fall in this class. Finally, the elements of G_n are not all the possible kinds of errors, and in that case something slightly different needs to be done, but we will not go into that.

Lectures on Quantum Computing
Physics 709
Lecture 22a: Continuous Errors

But still we are not quite done with quantum error correction. Indeed, at the start we insisted that the main problem for quantum computers was that errors could grow continuously from zero, very different from the bit-flip-only error for classical bits. We have shown how to correct for the single qubit X, Y, and Z errors but these still appear to be discrete errors. What about small errors that do not take the original state into an orthogonal state, but into a state with an arbitrary overlap with the original state? That's the kind of error we considered way back at the beginning of our discussion of quantum error correction.

An arbitrary 1-qubit error is the unitary transformation

$$\begin{aligned} E_{\theta,\widehat{n}} &= I\cos\frac{\theta}{2} + i\widehat{n}\cdot\sigma\sin\frac{\theta}{2} \\ &= I\cos\frac{\theta}{2} + i\sin\frac{\theta}{2}\left(xX_1 + yY_1 + zZ_1\right), \end{aligned}$$

where x, y, and z are real and $x^2 + y^2 + z^2 = 1$. It looks like we have a problem because this is not of the form X, Y, or Z except for very special values of $\widehat{n}$ and θ. In the 5-qubit code, the uncorrupted wavefunction is

$$|\Psi\rangle = \alpha\left|\overline{0}\right\rangle + \beta\left|\overline{1}\right\rangle,$$

and the wavefunctions after possible continuous errors are

$$\begin{aligned} &|\Psi\rangle \text{ with probability } 1-5p \\ &E^{(1)}_{\theta_1,\widehat{n_1}}|\Psi\rangle \text{ with probability } p \\ &E^{(2)}_{\theta_2,\widehat{n_2}}|\Psi\rangle \text{ with probability } p \\ &E^{(3)}_{\theta_3,\widehat{n_3}}|\Psi\rangle \text{ with probability } p \\ &E^{(4)}_{\theta_4,\widehat{n_4}}|\Psi\rangle \text{ with probability } p \\ &E^{(5)}_{\theta_5,\widehat{n_5}}|\Psi\rangle \text{ with probability } p. \end{aligned}$$

Now the trick is to do *just exactly what we did for the discrete errors.*

Let us say that the error is on qubit 1. We measure g_1, g_2, g_3, g_4 on the state $E^{(1)}_{\theta_1,\widehat{n_1}}|\Psi\rangle$ and obtain a sequence β_i of 4 numbers, each of which is ± 1. It is easy to compute the probabilities: since the g_i commute, the probability of any sequence

β_i is the product of the four individual probabilities. Performing the computations (which you should do) gives

$$\begin{aligned}
\Pr(1,1,1,1) &= 1-p+p\cos^2\frac{\theta}{2} \\
\Pr(1,-1,-1,1) &= p\sin^2\frac{\theta}{2}x^2 \\
\Pr(1,-1,-1,-1) &= p\sin^2\frac{\theta}{2}y^2 \\
\Pr(1,1,-1,-1) &= p\sin^2\frac{\theta}{2}z^2
\end{aligned}$$

and all others 0: the other sequences would indicate an error on qubits 2,3,4, or 5. Note that the sum of these probabilities is 1 and that θ measures the "size" of the error: if $\theta \longrightarrow 0$ then there is no error at all. Again, since θ is a continuous variable, the actual error is not drawn from a discrete set. However, the error syndrome is drawn from a discrete set.

If we look back at the table for β_{ij}, we see that the 4 possible measurement results are the sequences for no error, an X_1 error, a Y_1 error, and a Z_1 error, respectively. The table gives us unambiguous instructions what to do: apply I, X_1, Y_1, Z_1, respectively. Here, however, comes the absolutely crucial point. If we obtain, for example, $(1,-1,-1,1)$ for our error syndrome (as we would find for an X_1 error), then even though before the measurement we have $E^{(1)}_{\theta_1,\widehat{n_1}}|\Psi\rangle$, *after* the measurement the wavefunction is $X_1|\Psi\rangle$, because the measurement *has projected it into this state.* Hence the wavefunction is just exactly the one we would have if *there actually had been an* X_1 *error*! The projection erases all the other parts of the error. Hence we simply apply X_1 and we have corrected the (continuous) error. Clearly the same holds for all the other possible measurements. This is sometimes called the "discretization" of the errors. The collapse of the wavefunction by a measurement is what makes quantum error correction possible.

Lectures on Quantum Computing
Physics 709
Lecture 23
Introduction to Quantum Cryptography

No Cloning

The most famous achievement of quantum information theory, the Shor algorithm, provides a way to break a classical code. The *first* really well-known achievement (in 1984 by Bennett and Brassard) in quantum information theory was to show how to make a secure quantum communication scheme. That is the aim of quantum cryptography. The basic idea is pretty simple. A quantum measurement disturbs the system. So if Eve (eavesdropper - cute, eh?) intercepts a message from Alice to Bob and reads it, she cannot necessarily send it on to Bob completely undisturbed, so Bob may be able to figure out if Eve has been at work. We need to arrange things so as to take advantage of this peculiar aspect of quantum mechanics, that is, to find a protocol that both sends a message and detects efforts to intercept it.
Cryptography, the making of codes, is to be distinguished from cryptanalysis, which is the breaking of codes. Cryptanalysis is a subject we already talked about - it is what the Shor algorithm does for the RSA code. Together, the two subjects are sometimes called cryptology.
So let's try see how quantum mechanics might be of some use in cryptography, making use of the idea that it is hard to eavesdrop on quantum information without being detected. Let us say that Alice wants to send string of 0s and 1s to Bob. In between them, Eve is listening in. If the message is sent using classical bits, Eve just records the bits as they pass by and lets them go on to Bob, or intercepts them, copies them and then resends them to Bob. There is not that much one can do about this. However, if the message is sent using quantum bits, then the no-cloning theorem prevents Alice from doing these things. I mentioned the no-cloning theorem previously but did not discuss it in detail. Here is the proof of the theorem, adapted to this situation. Suppose Alice sends a $|0\rangle$ state to Bob. Along the way, it is intercepted by Eve. Her detector includes a memory register in the state $|r\rangle$ and a measuring device in the state $|d\rangle$. Hence the initial state is $|0\rangle \otimes |r\rangle \otimes |d\rangle$. When the detector interacts with the message to make a copy we have a unitary transformation

$$U\left(|0\rangle \otimes |r\rangle \otimes |d\rangle\right) = |0\rangle \otimes |0\rangle \otimes |d_0\rangle\,,$$

where the second $|0\rangle$ is the memory qubit, $|d_0\rangle$ is the final state of the measuring apparatus. U, $|r\rangle$, and $|d\rangle$ can contain no information about the received qubit since they have to be set up before the message get there. Thus Eve has copied ("cloned") the $|0\rangle$ state. If Alice sends a $|1\rangle$ state, then we have

$$U\left(|1\rangle \otimes |r\rangle \otimes |d\rangle\right) = |1\rangle \otimes |1\rangle \otimes |d_1\rangle\,.$$

So Eve has also copied ("cloned") the $|1\rangle$ state. So if Alice always sends a $|0\rangle$ or a $|1\rangle$, then Eve is OK: she has not disturbed the information so she can send it on. She also has a copy, which is what she wants. But here of course is where the possibility of changing bases comes in: Alice sends a state $\alpha|0\rangle + \beta|1\rangle$. Then the two equations imply that

$$U\left[(\alpha|0\rangle + \beta|1\rangle) \otimes |r\rangle \otimes |d\rangle\right] = \alpha|0\rangle \otimes |0\rangle \otimes |d_0\rangle + \beta|1\rangle \otimes |1\rangle \otimes |d_1\rangle,$$

because U is a linear operator. Now Eve needs to measure and send on the result. She will measure and send a $|0\rangle$ with probability $|\alpha|^2$ and her detector goes into the state $|0\rangle \otimes |d_0\rangle$ or measure and send a $|1\rangle$ with probability $|\beta|^2$ and her detector goes into the state $|1\rangle \otimes |d_1\rangle$. But the quantum information is in the coefficients α and β, which she still knows nothing about. The reason that we call this "no cloning" is that even though Eve has complete access to the state $\alpha|0\rangle + \beta|1\rangle$ she cannot make a copy of it.

There are 2 important lessons from this.

The first is that it is absolutely crucial that Alice and Bob use the freedom to choose different bases. If they always use the $\{|0\rangle, |1\rangle\}$ basis and Eve knows this, then Eve can succeed - the communication scheme would be essentially classical.

The second is that if they do use the basis freedom, then Alice and Bob can detect the doings of Eve. If Bob knows that Alice has prepared a known state in some basis other than $\{|0\rangle, |1\rangle\}$, then he will (at least sometimes) measure something other than what was sent. For example, Alice can use the $\{|+\rangle, |-\rangle\}$ basis. So sometimes Alice prepares and sends $H|0\rangle = |+\rangle = \left(1/\sqrt{2}\right)(|0\rangle + |1\rangle)$. Bob applies an H to the received qubit, then measures it in the computational basis, and he gets a $+1$. But if Eve has been copying and resending in the computational basis, then he will get $+1$ and -1 with equal probability.

(As an aside at this point, it is noteworthy that the proof of the no cloning theorem does not make use of the unitarity of U, only the linearity of U.)

So the basic idea of quantum cryptography is that the freedom to choose bases - a freedom that is absent in classical physics - can help to send secure information. It is very interesting that this basis choice was also at the heart of the Bell inequalities.

This may be a good time to point out a notational difference in the quantum computing and quantum cryptography literatures. Actual implementations of cryptographic schemes are mostly done using optical communication, *i.e.*, photons, and the quantum information is usually stored in the photon polarization. The computational basis states are the horizontal $|0\rangle$ and vertical $|1\rangle$ states, and the other basis that is most often used is the $45°$ $|+\rangle$ and $135°$ states $|-\rangle$. So one often sees notations such as $|h\rangle$ and $|v\rangle$, and so on, and the explanations are often given using the polarization language. For consistency with the rest of the lectures, I will stick with the spin language and just use $|0\rangle$ and $|1\rangle$ for the computational basis states, and $|+\rangle = \left(1/\sqrt{2}\right)(|0\rangle + |1\rangle)$, and $|-\rangle = \left(1/\sqrt{2}\right)(|0\rangle - |1\rangle)$ for the other most common basis. So as usual, $\langle 0|1\rangle = 0$, $\langle +|-\rangle = 0$, $\langle 0|\pm\rangle = 1/\sqrt{2}$, $\langle 1|\pm\rangle = \pm 1/\sqrt{2}$. $Z|0\rangle = |0\rangle$, $Z|1\rangle = -|1\rangle$, $X|+\rangle = |+\rangle$, $X|-\rangle = -|-\rangle$.

Using photons for quantum computation is challenging because it is hard (though not impossible) to get photons to interact with each other, and it is hard to store them for a long time, (though this depends on what you mean by a long time). However, communication protocols don't usually need inter-qubit interactions or qubit storage. Preparing polarization states is very easy, and carefully constructed optical fibers preserve polarization. The main trick is to get exactly one photon at a time in a bunch and then detecting that single photon. But this is getting to be possible with current technology.

Lectures on Quantum Computing
Physics 709
Lecture 24
Quantum Key Distribution

One-Time Pad

We will only need one result from classical cryptography, and it is a very simple one: that the only provably unbreakable code is a "one-time pad". The simplest form of this is as follows. Classical Alice and Classical Bob have in their possession, in advance of their communication, a shared identical secret string of random 0's and 1's that is just as long as their message, which will also be sent as a string of 0's and 1's. Alice adds the message and the random string together, addition + being the usual bitwise addition mod 2. (This operation is also variously called XOR, the "exclusive or"; in English, "x or y but not both".) She sends the result to Bob. So if the message is $m = 00110$ and the pad is $p = 11000$, Alice sends $m + p = 00110 + 11000 = 11110 = c$ to Bob. Bob has the string 11000 already, so he adds this to what he has been sent and gets $p + c = 11000 + 11110 = 00110 = m$, the original message. The sent string is completely random, so no one who intercepts it will be able to decode it. What this means is that if Alice and Bob *share* a string of random secret bits they can send an unbreakable coded message (which can even be sent over a public classical channel!)

In the context of the RSA algorithm, we have already discussed the question of private vs. public key systems. The one-time pad is a private key system. If the one-time pad key is sent out publicly, it is immediately useless, since the key can be as easily used to decode as to encode. In mathematical terms, it represents a function that is easy to invert.

Key Distribution

The idea of quantum cryptography is to *send* a one-time pad (a random bit string) using qubits over a public channel. We can then follow it up with un unbreakable message using classical bits, securely. The particular protocol is called BB84. There are many others, and we will talk about one of them at the end. BB are Bennett and Brassard and they proposed their method in 1984. It was the first.

We assume that Eve can intercept the communication of the pad, but BB84 ensures that the interception can always be detected by Alice and Bob. We will give the genral protocol, and take an illustrative example where the pad is of length 4.

Here are the steps for sending the pad.

1. Alice chooses a sequence of N states chosen uniformly at random from the set of four states $|0\rangle$,$|1\rangle$, $|+\rangle$, $|-\rangle$. We can think of Alice's procedure as first choosing the basis, then choosing the state. Let us call the $|0\rangle$,$|1\rangle$ basis the c-basis and $|+\rangle$, $|-\rangle$ the h-basis. Alice sends these qubits to Bob. The length N of the string should be about 4 times the length

of the desired one-time pad. For our example, $N = 16$. It is this transmission that might be intercepted by Eve.
Alice's transmission:

$$\begin{pmatrix} \text{BASIS} & h & c & h & h & h & c & c & h & h & c & h & c & c & h & h & c \\ \text{STATE} & +1 & +1 & -1 & -1 & -1 & +1 & +1 & +1 & -1 & +1 & -1 & -1 & -1 & +1 & +1 & -1 \end{pmatrix}$$

2. Bob now measures all the qubits that have been sent to him, but he chooses at random the h-basis and the c-basis in which to measure each qubit. Half the time his basis choice agrees with Alice's and half the time not. Bob records all his results, both the string of bases he chose and the string of measurements of ± 1. Then he destroys his qubits so no further spying is possible. If no eavesdropping has occurred, then the 9 qubits that were measured in the 'right' basis now give a string of +1s and -1s.
Bob's results with no eavesdropping:

$$\begin{pmatrix} \text{BASIS} & h & c & c & c & h & c & h & h & c & c & c & c & h & h & c & c \\ \text{Result} & +1 & +1 & -1 & +1 & -1 & +1 & -1 & +1 & -1 & +1 & -1 & -1 & +1 & +1 & -1 & -1 \\ \text{AGREE?} & \surd & \surd & X & X & \surd & \surd & X & X & X & \surd & X & \surd & \surd & \surd & X & \surd \end{pmatrix}$$

The results with a check mark agree with the ones that Alice sent. The ones with an "X" are random. That still leaves (in our example) 9 meaningful results.
3. Bob announces publicly the bases in which he has done each of his measurements, (the top row) but not the results. Note that this announcement comes after any possible activities of Eve.
4. Alice lets him know, again publicly, for which of the qubits in the sequence he made the 'right' choice of basis. Bob discards the measurement results for which he chose the 'wrong' basis (the ones marked with an X), and preserves the results for the 'right' ones (marked with a $\surd$). But he does not say what these results are.
5. Alice and Bob now compare SOME of the information they have publicly, let's say the second half of the qubits. (Del points out that at this point they can choose at random which qubits to compare.) They find that there are two possible cases.
Case 1. No eavesdropping.
They discard all preparation and measurement results on the 'wrong' qubits (those that were prepared in a different basis from the one they were measured in). So they lose about half the data, which is a price they are willing to pay. For our example, they map the +1 to a zero and the -1 to a 1 so they now have a string 001001001 (which only Alice and Bob know), and they (along with everybody else) know that this string came from qubits 1,2,5,6,10,12,13,14,16. Now they share the results 01001 publicly and find they agree. This means there was no eavesdropping, as we shall see in a moment. Alice and Bob are now finished, since they can send a classical message using the random sequence 0010 from the first half of the qubits. That's their secret pad, which contains the required 4 bits of information.
Case 2. Eavesdropping. Eve has been active, so she also knows which were the right qubits, because this was announced publicly. It's just 1,2,5,6,10,12,13,14,16. But Eve has measured the qubits herself in some basis. It does not matter whether she chooses the h and c bases at random or measures all qubits in one of them. Alice has chosen the bases at

random so Eve will choose the 'right' basis only half the time, so her measurement results on half the qubits are random.
Eve's results:

$$\begin{pmatrix} \text{BASIS} & h & h & h & c & h & h & c & h & h & c & h & h & c & h & c & h \\ \text{STATE} & +1 & -1 & -1 & +1 & -1 & -1 & +1 & +1 & -1 & +1 & -1 & -1 & -1 & +1 & -1 & -1 \end{pmatrix}$$

Thus Eve does NOT have the 001001001 string; she has a completely garbled version of it. These measurements actually did not do her any good . But now the key point is that Bob can add an extra step that will detect the presence of Eve. The message Bob received will be affected by Eve's measurements, since she projected the states every time she measured.
Bob's results:

$$\begin{pmatrix} \text{BASIS} & h & c & c & c & h & c & h & h & c & c & c & c & h & h & c & c \\ \text{STATE} & +1 & -1 & +1 & +1 & -1 & +1 & +1 & +1 & -1 & +1 & +1 & -1 & -1 & +1 & -1 & -1 \end{pmatrix}$$

These results disagree with those for case 1, the previous table of Bob's results. We compare the checked results in the second half. In case 1 we got +1,-1,+1,+1,-1. In case 2, we got +1,–1,+1,+1,-1. They will disagree about 1/4 of the time, since when the bases agree (1/2 the time) the results will agree, and when the bases disagree, the results will agree 1/2 the time.
Thus Alice and Bob, using this public data analysis, decide whether there has been anyone eavesdropping on their communication. If there is a 25% disagreement rate on the compared results, then their message is compromised.
There are 2 interesting points. First, in case 2, Eve has actually destroyed the message, so Alice and Bob have not succeeded in sending anything - they have only detected an eavesdropper. They will have to get a different means of communication. Second, they do NOT send a pad that Alice decides on. It's actually a RANDOM sequence, consisting of the 1/4 of the qubits for which they randomly got the right basis AND for which they compared results.

No Cloning Again

It's possible to prove that Eve cannot do better, using a very simple variant of the no-cloning theorem, as we intuited at the start. What she would really like to do is to start with her quantum processing unit in an initial state $|\Phi\rangle$, receive a qubit in state $|n\rangle$, where $|n\rangle$ can be any one of the 4 states $|0\rangle$,$|1\rangle$, $|+\rangle$, $|-\rangle$, use $|n\rangle$ to convert $|\Phi\rangle$ to $|\Phi_n\rangle$, and send $|n\rangle$ on to Bob. So Eve must come up with some unitary operator U that does the operation

$$U\,|\Phi\rangle\,|n\rangle = |\Phi_n\rangle\,|n\rangle \;\text{ for all }\; |n\rangle\,.$$

That would enable her to intercept the message without being detected. But let's look at the quantity C_{mn}, defined as

$$\begin{aligned} C_{mn} &= \langle\Phi|\,\langle m|\;\;|\Phi\rangle\,|n\rangle \;\; = \\ &= \langle\Phi|\,|\Phi\rangle\,\langle m|\,|n\rangle \\ &= \langle m|\,|n\rangle\,. \end{aligned}$$

We have that $C_{00} = C_{11} = C_{++} = C_{--} = 1$, $C_{01} = C_{10} = C_{+-} = C_{-+} = 0$, while $C_{0+} = C_{+0} = C_{0-} = C_{-0} = C_{1+} = C_{+1} = 1/\sqrt{2}$ and $C_{1-} = C_{-1} = -1/\sqrt{2}$. On the other hand, we have that

$$\begin{aligned} C_{mn} &= \langle\Phi| \langle m| U^\dagger U |\Phi\rangle |n\rangle \\ &= \langle\Phi_m| \langle m| |\Phi_n\rangle |n\rangle \\ &= \langle\Phi_m| |\Phi_n\rangle \langle m| |n\rangle \\ &= \langle\Phi_m| |\Phi_n\rangle C_{mn} \end{aligned}$$

and when C_{mn} takes on nonzero values, we must have that $\langle\Phi_m| |\Phi_n\rangle = 1$. This can only be the case if $|\Phi_m\rangle = |\Phi_n\rangle$ for all 4 choices of m and n. But if this is true, then Eve has no way to distinguish the 4 possibilities. This calculation can be used as the basis of a more general theory of distinguishing non-orthogonal bases.

An Alternative

There is another way to communicate a random string, this time using a source of entangled qubits. Let us imagine that a third party, Tom, has a machine that constructs the 2-qubit state

$$|\Psi\rangle = \frac{1}{\sqrt{2}} (|00\rangle + |11\rangle) .$$

This state has the key property that

$$|\Psi\rangle = \frac{1}{\sqrt{2}} (|++\rangle + |--\rangle) .$$

Then Tom sends the resulting 2 qubits repeatedly, the first to Alice and the second to Bob. They measure independently and randomly in the c-basis and the h-basis. When their bases agree, then so do their measurements. When their bases do not agree, then the measurements will agree only 1/2 the time. That is the nice property of $|\Psi\rangle$.
Now Alice announces her sequence of measurement bases, but not her results. Bob keeps the results from the qubits when the bases agreed and discards the others, as in the BB84 scheme. The record of Alice's results and Bob's results are identical if there is no tampering. So if everything agrees, then they have a one-time pad. But again they sacrifice some of their bits for the security check. Again if there was an interception attempt, 1/4 of the results will disagree.

Quantum Money

Here is another very simple example of how to use the basis freedom. The goal is to prevent counterfeiting of money. The national bank wants to issue a currency that is completely secure against copying. It puts a serial number on the note, and also a quantum memory register in the state $|\psi_1\rangle \otimes |\psi_2\rangle \otimes |\psi_3\rangle \otimes ... |\psi_N\rangle$. The states $|\psi_i\rangle$ are chosen uniformly at random from the set of 4 possible states $\{|0\rangle, |1\rangle, |+\rangle, |-\rangle\}$. The bank keeps a secret record of the sequence of states (so both the eigenvalue and the basis), and each different

serial number is paired with a state sequence. When the note is presented for payment at the bank, the serial number is read, which gives the sequence of states. The states are measured in this secret basis. If the right sequence of +1s and −1s is obtained, then the note is genuine. If someone wants to copy the note, they will need to guess the basis of each state, measure, and copy. For half of the qubits, they will guess the right basis and prepare the proper state. For the other half of the qubits they will guess the wrong basis and prepare the wrong state. For example, let us say that the stored state is $|+\rangle$. Then the counterfeiters guess the $\{|+\rangle, |-\rangle\}$ basis, the 'right' basis. They measure +1 and prepare a $|+\rangle$ state, which will fool the bank. This happens 50% of the time. But the counterfeiters guess the $\{|0\rangle, |1\rangle\}$ basis also 50% of the time. Then they measure and obtain +1, which happens with 50% probability. Then they prepare the state $|0\rangle$. The bank measures in the right basis, which is the $\{|+\rangle, |-\rangle\}$ basis. The bank will measure +1 with 50% probability, and -1 with 50% probability. In the first case they are fooled again, but in the second case, they know something is wrong: when the bank measures the qubits for which the guess was wrong, they will get an answer half the time that does not agree with the stored secret answer. Hence if the bank finds that their secret stored sequence of +1s and -1s disagrees 1/4 of the time with the new measurement, they know that the note is counterfeit. N should be large enough that the chance of getting agreement by chance, which is about $(3/4)^N$, is a small number.

Lectures on Quantum Computing
Physics 709
Lecture 25: The Density Matrix

Introduction

We have been talking about errors in a very abstract way so far. We simply postulated a very simple error model and then asked what we could do about the errors. Now we are going to actually ask more carefully about errors: how to describe them and how to measure them.

We might try to run an algorithm and include error correction steps and see if the algorithm gets the right answer. But at the present stage of quantum technology, this will not work so well - any algorithm run on a realistic problem will fail badly. We are still very much in the developmental era. So we need to be able to figure out how good our qubits and gates are even though we are not very close to a real computer.

The way to do this is to measure the density matrix ρ of our set of qubits. This is called quantum tomography.

We will first define ρ, then develop a method to determine ρ for a collection of N qubits to some predetermined level of accuracy with the least number of measurements, assuming that we can produce as many copies of ρ as we want.

Definition

What if we do not know precisely the state of a quantum system, perhaps because our preparation method is imperfect? Let's say that the probability of the state $|\alpha\rangle$ is p_a where, as usual $0 \leq p_a \leq 1$ and $\sum p_a = 1$. p_a represents *classical* uncertainty. Now we measure the Hermitian operator A, which satisfies $A\,|\alpha_n\rangle = a_n\ |\alpha_n\rangle$. $A = \sum_n a_n\ |\alpha_n\rangle\langle\alpha_n|$. Here $\{|\alpha_n\rangle\}_{n=1,2,\ldots,D}$ is a complete orthonormal basis, the eigenbasis of A.

Then the probability of measuring a_n is p_n. If we prepare and measure many times, the average value of a is

$$\overline{a} = \sum_n p_n a_n. \tag{1}$$

What if we measure a different Hermitian operator B, satisfying $B\,|\beta_n\rangle = b_n\ |\beta_n\rangle$, so $B = \sum_n b_n\ |\beta_n\rangle\langle\beta_n|$? Recalling that the quantum-mechanical probability for obtaining the results b_m when the system is in the state $|\alpha_n\rangle$ is $|\langle\alpha_n|\beta_m\rangle|^2$ we find

$$\begin{aligned}
\overline{b} &= \sum_{n,m} p_n b_m\, |\langle\alpha_n|\beta_m\rangle|^2 \\
&= \sum_{n,m} p_n b_m\ \langle\alpha_n|\beta_m\rangle\ \langle\beta_m|\alpha_n\rangle\,.
\end{aligned}$$

This leads us to define the density matrix, also sometimes called the density operator

$$\rho = \sum_n p_n \ |\alpha_n\rangle \ \ \langle\alpha_n| \, . \tag{2}$$

and we see that in the α-basis ρ is diagonal:

$$\rho = \begin{pmatrix} p_1 & 0 & ... & 0 \\ 0 & p_2 & ... & 0 \\ ... & ... & ... & 0 \\ 0 & 0 & 0 & p_D \end{pmatrix}$$

and we can write

$$\overline{a} = \mathrm{Tr}\left(\rho A\right). \tag{3}$$

The trace of a matrix c_{mn} is the sum of the diagonal elements:

$$Tr\left(c_{mn}\right) = \sum_n c_{nn}.$$

The trace of an operator C in a basis $\{|\alpha_n\rangle\}_{n=1,...,D}$ is defined the same way:

$$\begin{aligned} \mathrm{Tr}\left(C\right) &= \mathrm{Tr}\left(\sum_{mn} |\alpha_m\rangle \, C_{mn} \, \langle\alpha_n|\right) \\ &= \mathrm{Tr}\left(\sum_{n} |\alpha_n\rangle \, C_{nn} \, \langle\alpha_n|\right) \\ &= \sum_n C_{nn}. \end{aligned}$$

We also have

$$\overline{b} = \sum_{n,m} p_n b_m \, \langle\beta_m|a_n\rangle \ \ \langle a_n|\beta_m\rangle \tag{4}$$

$$= \sum_m b_m \, \langle\beta_m| \left[\sum_n p_n \, |\alpha_n\rangle \ \ \langle\alpha_n|\right] |\beta_m\rangle \tag{5}$$

$$= Tr\left(\rho B\right). \tag{6}$$

So there is a general rule that the average of a measurement of an arbitrary operator O in a state given by a density matrix ρ is the trace of the product of ρ and O : $\overline{O}$ =Tr(ρO) . Note that the trace is invariant under unitary transformations, since

$$Tr\left(C'\right) = Tr\left(UCU^{-1}\right) = Tr\left(CUU^{-1}\right) = Tr\left(C\right) \tag{7}$$

because of the invariance of the trace under cycling of matrix products. This implies that the trace can be taken in any orthonormal basis

$$Tr\left(\rho\right) = \sum_n \langle\alpha_n \, |\rho| \, \alpha_n\rangle = \sum_n \langle\beta_n \, |\rho| \, \beta_n\rangle \, .$$

ρ has three important properties: (1) ρ is Hermitian; (2) Tr$(\rho) = 1$; (3) ρ is a positive operator $\langle\alpha|\rho|\alpha\rangle \geq 0$ for all $|\alpha\rangle$.

Time Evolution

Considering ρ as an operator, we get an equation of motion for ρ :

$$i\hbar\frac{d\rho}{dt} = \sum_n p_n \left(i\hbar\frac{d}{dt}\left|\alpha_n\right\rangle\right) \ \left\langle\alpha_n\right| + \sum_n p_n \ \left|\alpha_n\right\rangle \left(i\hbar\frac{d}{dt} \ \left\langle\alpha_n\right|\right)$$

$$= \sum_n \left(H\left|\alpha_n\right\rangle\right) \ a_n \ \left\langle\alpha_n\right| - \sum_n \left|\alpha_n\right\rangle \ a_n \left(\left\langle\alpha_n\right| H\right) \tag{8}$$

so

$$i\hbar\frac{d\rho}{dt} = [H,\rho]\,.$$

This is to be compared to

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

for wavefunctions. In this derivation I used the result that

$$\frac{d}{dt} \ \left\langle\alpha_n\right| = \frac{i}{\hbar}\left\langle\alpha_n\right| H.$$

This comes from the fact that the magnitude of $\left|\alpha_n\right\rangle$ is time-independent:

$$\begin{aligned} 0 &= \frac{d}{dt}1 \\ &= \frac{d}{dt}\left\langle\alpha_n \left|\alpha_n\right.\right\rangle \\ &= \left(\frac{d}{dt} \ \left\langle\alpha_n\right|\right)\left|\alpha_n\right\rangle + \ \left\langle\alpha_n\right|\left(\frac{d}{dt}\left|\alpha_n\right\rangle\right) \\ &= \left(\frac{d}{dt} \ \left\langle\alpha_n\right|\right)\left|\alpha_n\right\rangle - \frac{i}{\hbar} \ \left\langle\alpha_n\right| H\left|\alpha_n\right\rangle \end{aligned}$$

and for this to be true for all kets requires

$$\frac{d}{dt} \ \left\langle\alpha_n\right| = \frac{i}{\hbar}\left\langle\alpha_n\right| H.$$

This evolution of ρ is unitary because the differential equation $i\hbar \ d\rho/dt = [H,\rho]$ corresponds to $\rho\left(t\right) = U\rho U^{-1}$ where U is the same evolution operator as for the kets.
If H is time-independent, then we can use the simple form denoted by 1. above for U and we have

$$\rho\left(t\right) = e^{-iHt/\hbar}\rho\left(0\right)e^{iHt/\hbar}.$$

This is to be compared to

$$\psi\left(t\right) = e^{-iHt/\hbar}\psi\left(0\right)$$

for wavefunctions.

Pure states and mixed states

Let's go back to our initial motivation for introducing the density matrix, which is the presence of classical uncertainty. What if there is no such uncertainty? Then the system is in some definite quantum state, say $|a_1\rangle$ and $p_1 = 1$. There is a Hermitian operator A such that $A|a_1\rangle = a_1 |a_1\rangle$ and again $A = \sum_n a_n \ |a_n\rangle \langle a_n|$ and $\{|a_n\rangle\}_{n=1,2,...,D}$ is a complete orthonormal basis, the eigenbasis of A.
Then the probability of measuring a_1 is 1. If we prepare and measure many times, the average value of a is

$$\overline{a} = \sum_n p_n a_n = a_1. \tag{9}$$

In this basis

$$\rho = \begin{pmatrix} 1 & 0 & ... & 0 \\ 0 & 0 & ... & 0 \\ ... & ... & ... & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}.$$

If there exists a basis in which ρ has this form, then ρ is called a pure state. The state is the first state in the basis. We can also write

$$\rho = |a_1\rangle \langle a_1|$$

A pure state satisfies the equation $\rho^2 = \rho$, so ρ is a projection operator, projecting along the direction $|a_1\rangle$. Of course in some other orthonormal basis, this density matrix $\rho' = U\rho U^{-1}$ will not be diagonal. But

$$\rho'^2 - \rho' = U\rho U^{-1} U\rho U^{-1} - U\rho U^{-1} = U\rho^2 U^{-1} - U\rho U^{-1} = U\rho U^{-1} - U\rho U^{-1} = 0,$$

so the equation $\rho^2 = \rho$ is a good test for purity in any basis. It implies also $Tr\,(\rho^2) = 1$ so we can think of $P = \mathrm{Tr}(\rho^2)$ as a measure of purity. $P \leq 1$ with $P = 1$ for a pure state. Any state that is not pure is called a "mixed" state. The maximally mixed state is

$$\rho = \begin{pmatrix} 1/D & 0 & ... & 0 \\ 0 & 1/D & ... & 0 \\ ... & ... & ... & 0 \\ 0 & 0 & 0 & 1/D \end{pmatrix} = \frac{1}{D} I.$$

Note that this ρ, being proportional to the identity, is the same in all orthonormal bases. $P = 1/D$ for this state, which is the minimum value. It can also be written as

$$\rho = \sum_{i=1}^{D} |a_i\rangle \langle a_i|.$$

Remarks

Quantum theory has traditionally been presented beginning from wavefunctions (pure states) and talking about density matrices much later. That's how I did it in this course. Some authors these days introduce density matrices at or near the beginning. Nielsen and Chuang

do this, for example. For them a "quantum state" is a density matrix, not a wavefunction. The latter point of view is obviously more general, since it includes pure states as a special case. The density matrix formulation also has the (relatively minor) advantage that the global phase ambiguity in the wavefunction formulation (the fact that $|a\rangle$ and $e^{i\phi}|a\rangle$ represent the same physical state) does not exist in the density matrix formulation. This pure state is given by a density matrix $\rho = e^{-i\phi}|a\rangle\langle a|e^{i\phi} = |a\rangle\langle a|$.
The density matrix is the key quantity in the study of qubit decoherence. Decoherence means exactly that the quantum computer evolves away from a perfectly prepared and manipulated state to an imperfectly known state. This can only be described by a density matrix. This will be the single biggest topic in the next semester.

Lectures on Quantum Computing
Physics 709
Lecture 26: Quantum Tomography

Single qubits (N=1)

We write the 2×2 density matrix ρ for a single qubit as

$$\rho = \frac{1}{2} I + \frac{1}{2} \sum_{i=x,y,z} n_i \sigma_i, \tag{1}$$

where $\sigma_{x,y,z}$ are the Pauli matrices. Eq. 1 yields ρ when given $\vec{n} = (n_x, n_y, n_z)$. To invert this equation multiply both sides by σ_j, take the trace, and use the fact that $\mathrm{Tr}(\sigma_i \sigma_j) = 2\delta_{ij}$:

$$\begin{aligned} \mathrm{Tr}\ \sigma_j \rho &= \frac{1}{2} \mathrm{Tr}\ \sigma_j + \frac{1}{2} \sum_{i=x,y,z} n_i \mathrm{Tr}\ \sigma_j \sigma_i \\ &= n_j. \end{aligned}$$

Eq. 1 looks a lot like the expansion of a vector (ρ) in terms of a basis set (σ_i) in terms of expansion coefficients n_i. In fact this is just what it is. We can define an inner product $(A, B) = \mathrm{Tr}\left(A^\dagger B\right)/2$ on any set of matrices. Here we are concerned with 2×2 Hermitian matrices. Since $\mathrm{Tr}(\sigma_i \sigma_j) = 2\delta_{ij}$ the Pauli matrices form an orthonormal set. Together with the unit matrix they are also complete. It is often useful to have this geometric structure on the space of all ρ.

It should be noted that while the Hermitian matrices form a vector space V, the set of density matrices, a subset of V, does not. It is not closed under addition: the sum of two density matrices has a trace of 2, and so it is not a density matrix.

We have finished the first step on our path to quantum tomography, which is to parameterize ρ. The n_i are the parameters to be determined by the tomography process.

How do we determine the n_i? We prepare ρ many, many times. We measure σ_i many times, average the result and use Eq. **??**. The eigenvalues of the σ_i are ± 1, so we get $+1$ or -1 every time. We will call such a repeated measurement of one of the σ_i a "Pauli measurement" and Eq. **??** defines the 3 possible Pauli measurements for $N = 1$. After averaging over many measurements we get an estimate of $n_i = p_i(1) - p_i(-1)$, where $p_i(\pm 1)$ is the relative frequency of ± 1. Since $0 \leq p_i \leq 1$ and $p_i(1) + p_i(-1) = 1$ we have $-1 \leq n_i \leq 1$. Substituting the n_i back into Eq. **??** yields ρ. More measurements yield better n_i's. If we want a given error with a given level of certainty, we can use the biased coin formula, discussed below, to tell us how many measurements we need to make. Once we have the n_i's we also have ρ to within some specified error.

As a reminder, the Pauli matrices have the following properties:

$$\begin{aligned} \text{Tr } \sigma_i &= 0 && (2)\\ \sigma_i^2 &= I && (3)\\ \sigma_i\sigma_j &= i\varepsilon_{ijk}\sigma_k \text{ for } i \neq j && (4)\\ \sigma_i\sigma_j + \sigma_j\sigma_i &= 0 \text{ for } i \neq j. && (5) \end{aligned}$$

ρ must be positive: its 2 eigenvalues $\lambda_<$ and $\lambda_>$ satisfy $0 \leq \lambda \leq 1$. In terms of $\vec{n}$, a simple calculation gives $\lambda_< = 1/2 - |\vec{n}|/2$ and $\lambda_> = 1/2 + |\vec{n}|/2$. Hence we have $0 \leq |\vec{n}| \leq 1$. Thus the set of allowed $\vec{n}$ lie in a sphere that is called the Bloch sphere. Pure states satisfy $\rho^2 = \rho$:

$$\begin{aligned} \rho^2 &= \left(\frac{1}{2}I + \frac{1}{2}\sum_i n_i\sigma_i\right)\left(\frac{1}{2}I + \frac{1}{2}\sum_j n_j\sigma_j\right) && (6)\\ &= \frac{1}{4}I + \frac{1}{2}\sum_i n_i\sigma_i + \frac{1}{4}\sum_{ij} n_in_j\sigma_i\sigma_j && (7)\\ &= \frac{1}{4}I + \frac{1}{2}\sum_i n_i\sigma_i + \frac{1}{4}I\sum_i n_in_i && (8)\\ &= \frac{1}{4}I\left(1 + \vec{n}^2\right) + \frac{1}{2}\sum_i n_i\sigma_i && (9)\\ &= \rho \text{ if and only if } |\vec{n}| = 1. && (10) \end{aligned}$$

So the surface of the Bloch sphere consists of the pure states. Another way to characterize a pure state that we used in the previous lecture is that it has one eigenvalue equal to one and all others zero, which is to say that ρ is of rank one for a pure state.

Bayes' Theorem

Having understood how ρ is properly parametrized and measured, we now try to understand in little more detail the statistics of the measurements. In particular, we'd like to know how many measurements we need to make to determine ρ to some desired level of accuracy. We'll start with a theorem from probability theory.

Since ρ contains uncertainty, our problem is that of determining a probability distribution by measurement. A simple example would be to determine whether a coin is "fair". A fair coin gives heads 50% of the time and tails 50% of the time. To discover whether a coin is fair we need to flip it many times to collect the desired information. It turns out that this famous example is just the one we need in quantum tomography. Let s be the "fairness" of the coin, defined as follows. Counting "heads" as $+1$ and "tails" as -1, and s as the average result, then $s = 0$ is a fair coin and $s > 0$ is a coin biased towards heads and $s < 0$ is a coin biased towards tails. We start flipping the coin. The first try comes up "heads", an event E which gives us a little information about s but obviously does not determine it. How do we deal with this situation?

There is a general method for such problems, which is called Bayes' theorem. It gives the right method for the problem of updating probability distributions in the presence of new

information. Let us say that we are trying to determine the value of some quantity s. We have some "prior" probability distribution $P_0(s)$. At the beginning, we might not know much about s, so $P_0(s)$ would be a broad distribution. We would like to turn $P_0(s)$ into a narrow distribution by collecting information about s. A delta function for P_0 would be the ideal result, since that would represent certainty. Our information is the result of a measurement which we call the event E. E is connected in some way to s, and we quantify this connection by the function $P(E|s)$, the probability that E would happen if the value of our quantity is s. Then the new distribution is $P(s|E)$ (the probability of s given E) and it is given by

$$P(s|E) = CP(E|s)P(s),$$

where C is a normalization constant and $P(E|s)$ is the probability of E given s. This is almost the theorem. Summing both sides of the equation over all results for s, and using $\sum_s P(s|E) = 1$, we see that C is given by

$$C^{-1} = \sum_s P(E|s)P(s) = P(E),$$

so Bayes' theorem is

$$P(s|E) = \frac{P(E|s)P(s)}{P(E)}.$$

Measuring σ_x

Let us apply Bayes' theorem to the problem of determining the density matrix of a qubit. The idea is simply to measure the quantity E repeatedly and update P after each measurement. Let's start with determining $n_x = \langle \sigma_x \rangle$ for a single qubit.

We first need P_0, the prior distribution. We know that $-1 \leq n_x \leq 1$ but we do not know anything else about n_x. So the prior distribution for n_x is the uniform distribution $P_0(n_x) = \frac{1}{2}\Theta(1 - |n_x|)$. $\Theta(x)$ is the step function defined by $\Theta(x = 0)$ for $x < 0$ and $\Theta(x = 1)$ for $x \geq 0$. The prior average value of n_x is $\overline{n_x} = \int_{-\infty}^{+\infty} n_x P_0(n_x)\, dn_x = \frac{1}{2}\int_{-1}^{+1} n_x\, dn_x = 0$ and the prior variance is

$$\begin{aligned} \overline{n_x^2} &= \frac{1}{2}\int_{-1}^{+1} n_x^2\, dn_x \\ &= \frac{1}{3}. \end{aligned}$$

The prior standard deviation is $\sqrt{1/3} \approx 0.58$.

Now let us make one measurement, obtaining the event $E = +1$. n_x is the average value of the set of +1's and -1's, so $n_x = P(1|n_x) - P(-1|n_x)$, and since $1 = P(1|n_x) + P(-1|n_x)$ we get $P(\pm 1|n_x) = (1 \pm n_x)/2$. Applying the theorem we have that

$$P_1(n_x| + 1) = C\frac{1 + n_x}{2}\frac{1}{2}\Theta(1 - |n_x|),$$

and normalizing we find $C = 2$. So our knowledge of n_x is slightly improved. The new probability distribution is skewed toward +1. The updated estimate for the average is

$$\overline{n_x} = \int_{-1}^{+1} n_x \frac{1 + n_x}{2} dn_x = \frac{1}{3}$$

and

$$\begin{aligned}\overline{(n_x-\overline{n_x})^2} &= \int_{-1}^{+1}\left(n_x-\frac{1}{3}\right)^2\frac{1+n_x}{2}dn_x \\ &= \frac{7}{36}.\end{aligned}$$

The standard deviation is $\sqrt{7/36}=0.44$, somewhat reduced from the 0.58 that we had to start with. This is a good start but obviously we need to make many measurements.
Now let us make N_x measurements, obtaining $+1$ H_x times. Using the fairness analogy, let's denote $s=(1+n_x)/2$. Then each time we measure there is a chance $h_x=H_x/N_x$ that we obtain $+1$, and a chance $1-h_x=1-H_x/N_x$ that we obtain -1. Taking into account also the multiplicities of the total number of observations, we have

$$\begin{aligned}P_{N_x}(H_x|n_x) &= \binom{N_x}{H_x}\left[(1+n_x)/2\right]^{H_x}\left[(1-n_x)/2\right]^{N_x-H_x}\Theta(1-|n_x|) \\ &= \binom{N_x}{H_x}s^{H_x}(1-s)^{N_x-H_x}\ \Theta(1-|n_x|) \\ &= \frac{N_x!}{H_x!\,(N_x-H_x)!}s^{H_x}(1-s)^{N_x-H_x}\ \Theta(1-|n_x|).\end{aligned}$$

Then, using $\ln(N!)\approx N\ln N-N$, we find

$$\frac{1}{N_x}\ln P_{N_x}(H_x|n_x)=-h_x\ln h_x-(1-h_x)\ln(1-h_x)+h\ln s+(1-h)\ln(1-s)$$

and we may expand this around its maximum at $h_x=p$ and undo the logarithm to find

$$P_{N_x}(H_x|n_x)=C\exp\left[-\frac{1}{2}\frac{N_x(h_x-s)^2}{s(1-s)}\right]\ \Theta(1-|n_x|)$$

Since we are on a finite interval, C, a constant, is a complicated expression involving an error function (erfc). Our new probability distribution for n_x after N_x measurements is

$$P_{N_x}(n_x)=C\ \Theta(1-|n_x|)\exp\left[-\frac{1}{2}\frac{N_x(h_x-s)^2}{s(1-s)}\right].$$

so that $C=2$. After N_x observations

$$\overline{n_x}=\int_{-1}^{+1}n_xP_{N_x}(n_x)\,dn_x=\frac{1+h_x}{2}$$

and

$$\overline{(n_x-\overline{n_x})^2}\ \sim\frac{s(1-s)}{N_x}=\frac{1-h_x^2}{4N_x}.$$

The standard deviation is proportional to $\left[N_x/(1-h_x^2)\right]^{-1/2}$, which can be reduced as much as we want by increasing N_x. Notice that the distribution narrows faster if $h_x\approx\pm1$. In this case the gaussian is cut off at the edge of the interval.
Our knowledge, as measured by the width of the distribution, only increases as the square root of the number of observations N_x. This is the reason that experiments are eventually shut down: the cost usually increases linearly with N_x.

Conclusion

If we now move on to determine σ_y, then the prior distribution is

$$P(n_y) = P_{H_x}(n_x)\,\Theta\left(1 - n_x^2 - n_y^2\right),$$

since we must have that $n_x^2 + n_y^2 \leq 1$ by positivity. The remainder of the analysis is identical. We will need fewer measurements N_y to narrow the distribution in the y direction, particularly if $\overline{n_y} \approx \sqrt{1 - \overline{n_x^2}}$. Similarly for the z direction.

For N qubits the procedure is, in principle, the same. One must keep in mind, however, that all of the $4^N - 1$ Pauli operators must be measured. Furthermore, the positivity constraints are much more complicated. A very active field of research at the moment is how to reconstruct the density matrix given only a subset of the 4^N measurements.