Denver Physics Group

Quantum Computing Notes

Mike Witt

msg2mw@gmail.com

Contents

Part I

Background

1 Introduction and Prerequisites

Introduction

These notes are intended to go along with the quantum computing "series" planned for the Denver physics study group in early 2017. We hope to alternate between sessions on QC hardware and sessions on algorithms. If all goes well, these notes will capture the information on algorithms.

The general plan

First we'll do a little bit of review of basic calculations with qbits and cover what comes up with that, just to make sure we're all on the same page. Then we'll spend some time on the density matrix (and related things) so that we're prepared to use the DM as a measure of entanglement. Some investigation of the effects of the most important quantum operators (gates) may also be a good idea.

Then, the plan is to look at (1) Grover's algorithm, (2) Simon's algorithm, and (3) Shor's algorithm.

Prerequisites

I'm assuming that everybody involved in this has seen the notes from the class that Tina and I taught in Portland. If you understand the material there, then you're good to go. If somehow *these* notes have fallen into your hands, but you haven't seen *those* ones, email me.

Basically, you'll need to know the fundamental concepts of (finite) complex vector spaces (particularly change of basis, eigenvalues and eigenvectors), the basic postulates of (discrete) quantum theory, how to deal with both (simplified) spin states and qbit states, and how to use tensor products to put bits together. All this needs to be done either using vector notation (that is, "rows" and "columns") or in Dirac notation ("bras" and "kets").

2 Review: Working in Dirac notation

Before going forward, let's review the various kinds of products in Dirac notation. You've probably been doing tensor products this way already. The outer product will be important when we get to density matrices. The inner product in Dirac notation is not very useful, but it's good to understand how it would be done.

When working in Dirac notation, we are basically just multiplying expressions containing states (bras and kets) in the same way we would multiply polynomial expressions. The only differences are: (1) we have to be careful not to interchange the order of the states, and (2) we must do the adjoints properly (change kets into bras) and conjugate the scalar coefficients when necessary.

We'll be working with two vectors (or states)

$$|a\rangle = \begin{pmatrix} a_1 \\ a_2 \end{pmatrix} = a_1|0\rangle + a_2|1\rangle$$

$$|b\rangle = \begin{pmatrix} b_1 \\ b_2 \end{pmatrix} = b_1|0\rangle + b_2|1\rangle$$

The Inner Product

The inner product $\langle a|b\rangle$ is a row times a column in vector notation:

$$\begin{pmatrix} a_1^* & a_2^* \end{pmatrix} \begin{pmatrix} b_1 \\ b_2 \end{pmatrix} = a_1^* b_1 + a_2^* b_2$$

In Dirac notation $|a\rangle$ is: $a_1|0\rangle + a_2|1\rangle$. So $\langle a|$ will be: $a_1^*\langle 0| + a_2^*\langle 1|$. Notice that we have to turn the kets into bras and also conjugate the coefficients. Then we multiply $|a\rangle$ and $|b\rangle$:

$$\begin{aligned} &\big(\, a_1^*\langle 0| + a_2^*\langle 1|\,\big)\,\big(\, b_1|0\rangle + b_2|1\rangle\,\big) \\ &= a_1^* b_1\langle 0|0\rangle + a_1^* b_2\langle 0|1\rangle + a_2^* b_1\langle 1|0\rangle + a_2^* b_2\langle 1|1\rangle \\ &= a_1^* b_1 \times 1 + a_1^* b_2 \times 0 + a_2^* b_1 \times 0 + a_2^* b_2 \times 1 \\ &= a_1^* b_1 + a_2^* b_2 \end{aligned}$$

As I mentioned above, doing the inner product this way is not generally very useful.

The Outer Product

First let's examine the result of some simple outer products between zero and one states. We'll do this in vector notation.

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

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

As you can see, if you're using zeros and ones as states, and if you use "zero based" indexing in constructing the matrices, then the ket represents the row number and the bra represents the column number. If you work it out, you will see that this pattern continues when you have multiple bits. For example $|11\rangle\langle 11|$ will generate a matrix that has a one in row 3, column 3 (the last row and column).

So now let's do the outer product of $|a\rangle$ and $|b\rangle$. In this case we need to transpose and conjugate $|b\rangle$, giving:

$$\begin{aligned} |a\rangle\langle b| &= \big(\, a_1|0\rangle + a_2|1\rangle \,\big)\big(\, b_1^*\langle 0| + b_2^*\langle 1| \,\big) \\ &= a_1\, b_1^*\, |0\rangle\langle 0| + a_1\, b_2^*\, |0\rangle\langle 1| + a_2\, b_1^*\, |1\rangle\langle 0| + a_2\, b_2^*\, |1\rangle\langle 1| \end{aligned}$$

The expression: $a_1\, b_1^*\, |0\rangle\langle 0| + a_1\, b_2^*\, |0\rangle\langle 1| + a_2\, b_1^*\, |1\rangle\langle 0| + a_2\, b_2^*\, |1\rangle\langle 1|$ *is* the matrix in Dirac form. In matrix form we would have:

$$a_1 b_1^* \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} + a_1 b_2^* \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} + a_2 b_1^* \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix} + a_2 b_2^* \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} a_1 b_1^* & a_1 b_2^* \\ a_2 b_1^* & a_2 b_2^* \end{pmatrix}$$

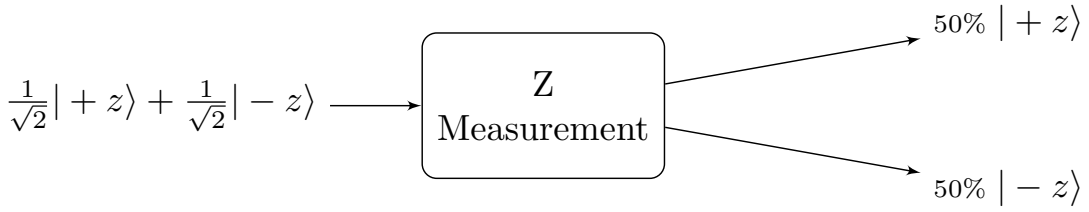
The Tensor Product

For the tensor product $|a\rangle \otimes |b\rangle$ you simply do the multiplication:

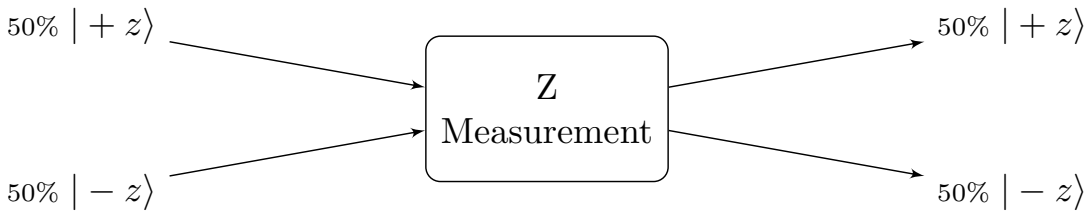
$$\big(\, a_1|0\rangle + a_2|1\rangle \,\big)\ \big(\, b_1|0\rangle + b_2|1\rangle \,\big) = a_1b_1|00\rangle + a_1b_2|01\rangle + a_2b_1|10\rangle + a_2b_2|11\rangle$$

3 Pure and mixed states

Consider the following measurement scenarios.



Experiment 1



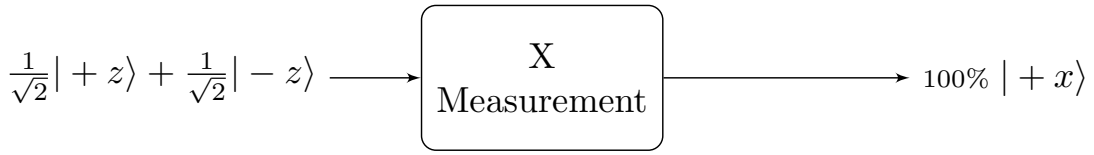
Experiment 2

Consider the two experiments above. In both cases we have a stream of electrons entering a device that measures spin along the z axis. In experiment 1 there is no ambiguity about the states. Every electron is in the same state, an equal superposition of plus and minus z. But the results of the measurement are probabilistic. For each electron we have a 50/50 chance of seeing plus or minus spin.

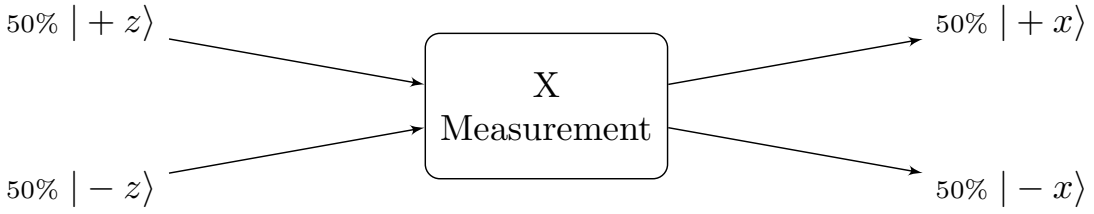
In experiment 2, half of the electrons coming into the measuring device have plus z spin and half of them have minus z spin. So we don't know the actual state of any given electron on input. But notice that we get exactly the same output results. 50% plus and 50% minus spin.

Now you might be a little suspicious about the claim that the inputs to these two experiments are actually different. After all, we get exactly the same measurement statistics. When we say that all the electrons in experiment 1 are in an "equal superposition" of plus and minus, might that not simply mean that half of them are plus and half are minus – and we just don't know which ones are which?

This concern can be addressed with the following two measurements.



Experiment 3



Experiment 4

For the two experiments above we orient our measuring device along the x axis rather than the z axis. Bearing in mind that $\frac{1}{\sqrt{2}}|+z\rangle + \frac{1}{\sqrt{2}}|-z\rangle = |+x\rangle$, you can see how experiments 3 and 4 do distinguish between the two input streams. This shows that a superposition really is different than simply not knowing the state.

The input to experiments 1 and 3 is called a *pure state* and the input to experiments 2 and 4 is called a *mixed state*.

$\frac{1}{\sqrt{2}}|+z\rangle + \frac{1}{\sqrt{2}}|-z\rangle =$ Pure State

50% $|+z\rangle$, 50% $|-z\rangle =$ Mixed State

In a pure state, we know exactly what the state is. The uncertainty about further measurement is a quantum effect. In a mixed state, there is "classical uncertainty" about the value of the state.

It is possible in principle to construct a measurement that would yield a single result for all electrons in any single pure state. But since, for a mixed state, the electrons are actually in *different* states, no single measurement will give the same result for all of them.

4 The Density Matrix

We now turn to the study of the density operator (or density matrix as it is typically called). This, like the operators that represent observables, is an object that doesn't actually operate on a state, but rather *represents* something.

The density matrix is used in formulating measures of entanglement, and it is a key component in the study of decoherence and 'open' quantum systems.

The density matrix for pure states

The density matrix for a pure state is simply the outer product of the state with itself:

$$\rho = |\psi\rangle\langle\psi|$$

Some examples:

$$\rho_{(+z)} = |+z\rangle\langle+z| = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \begin{pmatrix} 1 & 0 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}$$

$$\rho_{(+x)} = |+x\rangle\langle+x| = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 1 \end{pmatrix} \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \end{pmatrix} = \begin{pmatrix} 1/2 & 1/2 \\ 1/2 & 1/2 \end{pmatrix}$$

$$\rho_{(-y)} = |-y\rangle\langle-y| = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ -i \end{pmatrix} \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & i \end{pmatrix} = \begin{pmatrix} 1/2 & i/2 \\ -i/2 & 1/2 \end{pmatrix}$$

Here are some facts about the three density matrices above. As usual, all of the arithmetic is being done in the z basis. So all three matrices have their components in z. The main diagonal of each matrix gives the probabilities of getting $+z$ and $-z$. So for $\rho_{(+z)}$ we have certainty of getting $+z$, as expected. For both $\rho_{(+x)}$ and $\rho_{(-y)}$ we have a 50/50 chance of getting plus or minus z, which are also the values that you would calculate.

The off diagonal terms, which are sometimes called *interference terms*, do not come into the picture in terms of calculating probabilities. But they do, in some sense, signify the "degree of superposition" in the z basis.

The density matrix for mixed states

The density matrix for a mixed state is the sum of the density matrices for all of the pure states, weighted by the probability of seeing each of those states in the total mixture:

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

For example, a 50/50 mixture of $|+z\rangle$ and $|-z\rangle$ would have the density matrix:

$$\rho_{mix} = \frac{1}{2}|+z\rangle\langle+z| + \frac{1}{2}|-z\rangle\langle-z| = \frac{1}{2}\begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} + \frac{1}{2}\begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} \frac{1}{2} & 0 \\ 0 & \frac{1}{2} \end{pmatrix}$$

Compare this to the density matrix for an equal superposition of $|+z\rangle$ and $|-z\rangle$:

$$\rho_{sup} = \left(\frac{1}{\sqrt{2}}|+z\rangle + \frac{1}{\sqrt{2}}|-z\rangle\right)\left(\frac{1}{\sqrt{2}}\langle+z| + \frac{1}{\sqrt{2}}\langle-z|\right) = \begin{pmatrix} \frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & \frac{1}{2} \end{pmatrix}$$

(When working out the expression above, do the outer product in Dirac form, and then remember that $|+z\rangle = |0\rangle$ and $|-z\rangle = |1\rangle$, giving the indices in the resulting matrix.)

So, we have:

$$\rho_{sup} = \begin{pmatrix} \frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & \frac{1}{2} \end{pmatrix}, \qquad \rho_{mix} = \begin{pmatrix} \frac{1}{2} & 0 \\ 0 & \frac{1}{2} \end{pmatrix}$$

In both cases, the first and second numbers in the main diagonal tell us that we have a 50% chance of measuring $+z$ and $-z$ respectively. But ρ_{sup} also has the so called "interference terms" in the off diagonal elements. Thus, the density matrix "recognizes" that an equal superposition and an equal mixture are different things.

Here's another comparison. The density matrix of a mixture of one quarter $|0\rangle$ states and three quarters $|1\rangle$ states, along with the pure state that would exhibit the same probabilities:

$$\psi_{mix} = \frac{1}{4}|0\rangle, \ \ \frac{3}{4}|1\rangle \qquad \rho_{mix} = \begin{pmatrix} \frac{1}{4} & 0 \\ 0 & \frac{3}{4} \end{pmatrix}$$

$$\psi_{pure} = \frac{1}{2}|0\rangle + \frac{\sqrt{3}}{2}|1\rangle \qquad \rho_{pure} = \begin{pmatrix} \frac{1}{4} & \frac{\sqrt{3}}{4} \\ \frac{\sqrt{3}}{4} & \frac{3}{4} \end{pmatrix}$$

5 Purity

Using the density matrix there is a way to calculate "how pure" a state is. The purity of a state is given by the trace of the density matrix squared:

$$\text{purity}(\rho) = \text{Tr}(\rho^2)$$

The purity of a pure state is 1. For any mixed state the purity will be less than one. The minimum purity is one divided by the dimension of the state. 1/2 for a one-bit state, 1/4 for a two-bit state, and so on.

The state $|0\rangle$ is a pure state:

$$\rho = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}, \quad \rho^2 = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}, \quad \text{Tr}(\rho^2) = 1 + 0 = 1$$

The state $\frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle$ is pure:

$$\rho = \begin{pmatrix} \frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & \frac{1}{2} \end{pmatrix}, \quad \rho^2 = \begin{pmatrix} \frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & \frac{1}{2} \end{pmatrix}, \quad \text{Tr}(\rho^2) = \frac{1}{2} + \frac{1}{2} = 1$$

But a 50/50 mixture is "completely" mixed and therefore has the minimum purity:

$$\rho = \begin{pmatrix} \frac{1}{2} & 0 \\ 0 & \frac{1}{2} \end{pmatrix}, \quad \rho^2 = \begin{pmatrix} \frac{1}{4} & 0 \\ 0 & \frac{1}{4} \end{pmatrix}, \quad \text{Tr}(\rho^2) = \frac{1}{4} + \frac{1}{4} = \frac{1}{2} = .5$$

A 25/75 mixture is less mixed and so has somewhat higher purity:

$$\rho = \begin{pmatrix} \frac{1}{4} & 0 \\ 0 & \frac{3}{4} \end{pmatrix}, \quad \rho^2 = \begin{pmatrix} \frac{1}{16} & 0 \\ 0 & \frac{9}{16} \end{pmatrix}, \quad \text{Tr}(\rho^2) = \frac{10}{16} = .625$$

And a 1/999 mixture is just a "little bit" mixed and so its purity is almost one:

$$\rho = \begin{pmatrix} \frac{1}{100} & 0 \\ 0 & \frac{99}{100} \end{pmatrix}, \quad \rho^2 = \begin{pmatrix} \frac{1}{10,000} & 0 \\ 0 & \frac{9,801}{10,000} \end{pmatrix}, \quad \text{Tr}(\rho^2) = \frac{9,802}{10,000} \approx .98$$

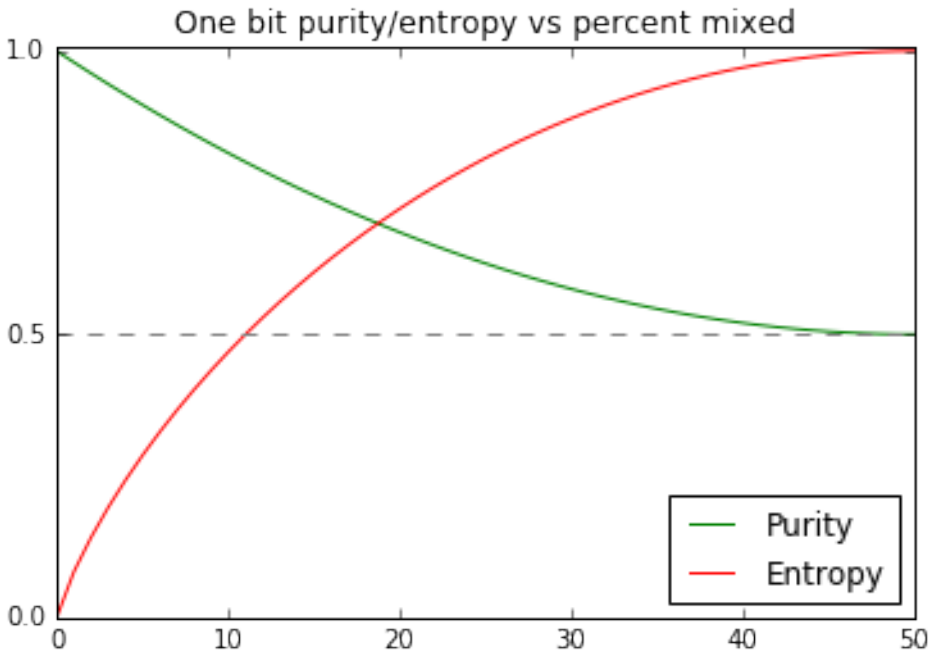
6 Comparison of purity and entropy

There is another metric called entropy, which is (I believe) more widely used than purity. Entropy is defined as:

$$\text{entropy}(\rho) = \sum_i \lambda_i \, log_2(\lambda_i)$$

where the λ are the eigenvalues of the density matrix. The entropy of any pure state will be zero. The maximum entropy (assuming you are taking the logs in base 2) will be the number of bits in the state.

I'm going with purity for the time being, simply because it's easier to calculate "by hand" and may serve us better for doing quick calculations on the board. I'll have more to say about entropy later, but for the moment I'll just provide this plot of purity (green) and entropy (red) for a one-bit system as the amount of the mixture ranges from zero (pure) to 50% (maximally mixed).



7 The partial trace operation

The partial trace operation is used to "eliminate" one or more systems from a multi-system state, so that we can examine the remaining system(s) in isolation. We are going to use it, in conjunction with the density matrix, to examine the "degree" of entanglement of various states we encounter in the course of running quantum algorithms.

We'll need to follow two rules to help understand how the partial trace is done:

(1) *The outer product of the tensor products equals the tensor product of the outer products.* In other words:

$$|a\,b\rangle\langle a\,b| = |a\rangle\langle a| \otimes |b\rangle\langle b|$$

This implies that the density matrix of a composite system is the tensor product of the density matrices of the individual components.

$$\rho_{ab} = \rho_a \otimes \rho_b$$

(2) *The trace of an outer product equals the (reversed) inner product*

$$\mathrm{Tr}(|x\rangle\langle y|) = \langle y|x\rangle$$

Applying these two rules, we can take the density matrix for a composite system and "trace out" one of the components, leaving the density matrix of the other one. Say we have two systems a and b where the DM of the composite system is:

$$\rho_{ab} = |a\,b\rangle\langle a\,b|$$

Then we can obtain the DM for system a by tracing out system b like this:

$\rho_a = \mathrm{Tr_b}(\rho_{ab})$

$= \mathrm{Tr_b}(|ab\rangle\langle ab|)$ - This outer product is equivalent to ρ_{ab}

$= \mathrm{Tr_b}(|a\rangle\langle a| \otimes |b\rangle\langle b|)$ - We applied rule (1)

$= |a\rangle\langle a|\mathrm{Tr_b}(|b\rangle\langle b|)$ - By linearity of the trace operator

$= |a\rangle\langle a|\langle b|b\rangle$ - We applied rule (2)

Now at this point it may not *look* like we've eliminated b, but the inner product in the last line will drop out via orthogonality, as we'll see when we work a couple of examples.

Example one: a product state

Let the two states be:

$$|a\rangle = \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle, \quad \rho_a = \begin{pmatrix} \frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & \frac{1}{2} \end{pmatrix} \quad |b\rangle = \frac{\sqrt{3}}{2}|0\rangle + \frac{1}{2}|1\rangle \quad \rho_b = \begin{pmatrix} \frac{3}{4} & \frac{\sqrt{3}}{4} \\ \frac{\sqrt{3}}{4} & \frac{1}{4} \end{pmatrix}$$

The composite state is:

$$|ab\rangle = \frac{\sqrt{6}}{4}|00\rangle + \frac{\sqrt{2}}{4}|01\rangle + \frac{\sqrt{6}}{4}|10\rangle + \frac{\sqrt{2}}{4}|11\rangle$$

And its density matrix is:

$$\begin{aligned}
\rho_{ab} = &\frac{3}{8}|00\rangle\langle 00| + \frac{\sqrt{3}}{8}|00\rangle\langle 01| + \frac{3}{8}|00\rangle\langle 10| + \frac{\sqrt{3}}{8}|00\rangle\langle 11| \\
&+ \frac{\sqrt{3}}{8}|01\rangle\langle 00| + \frac{1}{8}|01\rangle\langle 01| + \frac{\sqrt{3}}{8}|01\rangle\langle 10| + \frac{1}{8}|01\rangle\langle 11| \\
&+ \frac{3}{8}|10\rangle\langle 00| + \frac{\sqrt{3}}{8}|10\rangle\langle 01| + \frac{3}{8}|10\rangle\langle 10| + \frac{\sqrt{3}}{8}|10\rangle\langle 11| \\
&+ \frac{\sqrt{3}}{8}|11\rangle\langle 00| + \frac{1}{8}|11\rangle\langle 01| + \frac{\sqrt{3}}{8}|11\rangle\langle 10| + \frac{1}{8}|11\rangle\langle 11|
\end{aligned}$$

Now let's try to retrieve system a by tracing out system b. Remember that everything is linear, so we simply do the scheme on the last page to each term individually. I'll take the first two terms in detail. For the first term:

$$\begin{aligned}
&\mathrm{Tr_b}\left(\frac{3}{8}|00\rangle\langle 00|\right) \text{ - First term} \\
&= \mathrm{Tr_b}\left(\frac{3}{8}|0\rangle\langle 0| \otimes |0\rangle\langle 0|\right) \text{ - Rule (1)} \\
&= \frac{3}{8}|0\rangle\langle 0|\,\mathrm{Tr_b}(\,|0\rangle\langle 0|\,) \text{ - Linearity} \\
&= \frac{3}{8}|0\rangle\langle 0|\langle 0|0\rangle \text{ - Rule (2)} \\
&= \frac{3}{8}|0\rangle\langle 0| \text{ - Since it turns out that } \langle 0|0\rangle = 1
\end{aligned}$$

For the second term:

$$\mathrm{Tr_b}\left(\frac{\sqrt{3}}{8}\left|00\right\rangle\left\langle 01\right|\right) \text{ - Second term}$$

$$= \mathrm{Tr_b}\left(\frac{\sqrt{3}}{8}\left|0\right\rangle\left\langle 0\right| \otimes \left|0\right\rangle\left\langle 1\right|\right) \text{ - Rule (1)}$$

$$= \frac{\sqrt{3}}{8}\left|0\right\rangle\left\langle 0\right| \mathrm{Tr_b}(\left|0\right\rangle\left\langle 1\right|) \text{ - Linearity}$$

$$= \frac{\sqrt{3}}{8}\left|0\right\rangle\left\langle 0|1\right\rangle \text{ - Rule (2)}$$

$$= \text{(nothing) - The whole term drops out since } \left\langle 0|1\right\rangle = 0$$

So at this point I hope it's obvious that you can just look at ρ_{ab} in Dirac form and strike out all the terms where the two bits of system b don't match (which will be half of the terms), reducing it to:

$$\begin{aligned}\rho_{ab} = &\frac{3}{8}\left|00\right\rangle\left\langle 00\right| + \frac{3}{8}\left|00\right\rangle\left\langle 10\right| \\ &+ \frac{1}{8}\left|01\right\rangle\left\langle 01\right| + \frac{1}{8}\left|01\right\rangle\left\langle 11\right| \\ &+ \frac{3}{8}\left|10\right\rangle\left\langle 00\right| + \frac{3}{8}\left|10\right\rangle\left\langle 10\right| \\ &+ \frac{1}{8}\left|11\right\rangle\left\langle 01\right| + \frac{1}{8}\left|11\right\rangle\left\langle 11\right|\end{aligned}$$

At this point all the system b bits will turn into inner products that equal one. So you can just look at the system a bits and write:

$$\frac{3}{8}\left|0\right\rangle\left\langle 0\right| + \frac{3}{8}\left|0\right\rangle\left\langle 1\right| + \frac{1}{8}\left|0\right\rangle\left\langle 0\right| + \frac{1}{8}\left|0\right\rangle\left\langle 1\right| + \frac{3}{8}\left|1\right\rangle\left\langle 0\right| + \frac{3}{8}\left|1\right\rangle\left\langle 1\right| + \frac{1}{8}\left|1\right\rangle\left\langle 0\right| + \frac{1}{8}\left|1\right\rangle\left\langle 1\right|$$

Finally, gathering like terms together gives us back the correct DM for system a:

$$\rho_a = \frac{1}{2}\left|0\right\rangle\left\langle 0\right| + \frac{1}{2}\left|0\right\rangle\left\langle 1\right| + \frac{1}{2}\left|1\right\rangle\left\langle 0\right| + \frac{1}{2}\left|1\right\rangle\left\langle 1\right| = \begin{pmatrix} \frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & \frac{1}{2} \end{pmatrix}$$

Example two: an entangled state

Consider the first Bell state:

$$|\phi^+\rangle \;=\; \frac{1}{\sqrt{2}}|00\rangle + \frac{1}{\sqrt{2}}|11\rangle$$

And its density matrix:

$$\frac{1}{2}|00\rangle\langle 00| + \frac{1}{2}|00\rangle\langle 11| + \frac{1}{2}|11\rangle\langle 00| + \frac{1}{2}|11\rangle\langle 11|$$

Doing a partial trace on *either* bit will result in the one-bit density matrix:

$$\rho = \begin{pmatrix} \frac{1}{2} & 0 \\ 0 & \frac{1}{2} \end{pmatrix}$$

Now consider what has just happened. We had an entangled state. It would not have been possible to write the individual "state" for one of the bits, because they have no individual state vectors. But by tracing out one of the two bits, we *can* derive a density matrix for the other one. This matrix shows the correct probabilities (50/50 for getting a zero or one) and it shows something else. A "maximally mixed" state (check the purity).

Above, when we traced out the individual DMs for the product state, we got back the originals from the states before they were multiplied together. The purity metric would have told us that these were pure states (if we hadn't known already). *But when we isolate the DM of a bit that is entangled with one or more other bits, then a purity calculation will indicate that it is mixed.* For a "maximally entangled" state such as the Bell state above, the purity calculation will show that it is maximally mixed. This is one way we can measure the "degree" of entanglement.

Exercises: Investigate the entanglement of

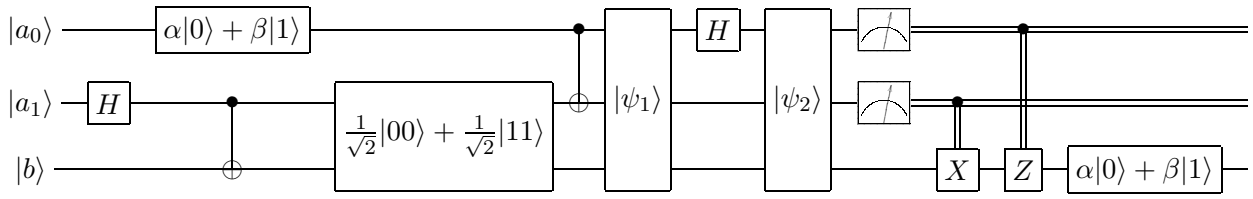
$\psi_1 = \frac{1}{\sqrt{3}}|00\rangle + \frac{1}{\sqrt{3}}|01\rangle + \frac{1}{\sqrt{3}}|10\rangle$

$\psi_2 = \frac{1}{2}|00\rangle + \frac{1}{2}|01\rangle + \frac{1}{2}|10\rangle - \frac{1}{2}|11\rangle$ (Note the minus sign in the last term!)

8 Investigating entanglement. Example: Teleportation.

Now we'll use what we've learned so far to take a closer look at the entanglement in different steps of the quantum teleportation algorithm that we previously covered.

First I'll summarize the algorithm. This time I'll do it as a "circuit" and include the step that entangles Alice's 2nd bit with Bob's bit:



Take a look at the "Portland notes" and make sure you can see how the circuit above corresponds to Section 12 (starting around page 61, I believe). a_0 is Alice's unknown state. The one she wants to transmit to Bob. a_1 and b are Alice and Bob's entangled bits. In the discussion in the old notes, we didn't talk about how a_1 and b originally got entangled. This happens via the Hadamard on a_1 and the CNOT(a_1, b).

On the circuit I have two boxes marked ψ_1 and ψ_2. ψ_1 is the state (of all three bits) just after Alice does her CNOT (step 2 in the old notes) and ψ_2 comes after her Hadamard (step 3). These are the states:

$$\psi_1 = \frac{1}{\sqrt{2}}(\ \alpha|000\rangle + \alpha|011\rangle + \beta|101\rangle + \beta|110\rangle$$

$$\psi_2 = \frac{1}{2}(\ \alpha|000\rangle + \alpha|011\rangle + \alpha|100\rangle + \alpha|111\rangle + \beta|001\rangle + \beta|010\rangle - \beta|101\rangle - \beta|110\rangle$$

We know that prior to Alice doing CNOT(a_0, a_1) there wasn't any entanglement between her two bits. Logically that CNOT must have been what entangled them, since it was the only operation that involved both bits. But some questions could still be asked: Are all three bits entangled with each other? What is the degree of the entanglement? And do questions like this even have answers?

Let's see what we can find out with the density matrix.

I'm not going to do all the math here. We should probably do it on the board together. But if you take ψ_1 and compute its density matrix, trace out b, and then trace out a_1. You should get the following DM for a_0:

$$\rho_{a_0} = \begin{pmatrix} \alpha\alpha^* & 0 \\ 0 & \beta\beta^* \end{pmatrix}, \quad \rho_{a_0}^2 = \begin{pmatrix} \left(\alpha\alpha^*\right)^2 & 0 \\ 0 & \left(\beta\beta^*\right)^2 \end{pmatrix}, \quad \text{purity} = \left(\alpha\alpha^*\right)^2 + \left(\beta\beta^*\right)^2$$

What can we make of this? Bear in mind that α and β are simply what we chose to name the (unknown) probability amplitudes for Alice's bit 0. Therefore it must be true that $\alpha\alpha^* + \beta\beta^* = 1$. (This is just another way of writing $|\alpha|^2 + |\beta|^2$).

Given this, you should be able to work out that the purity of this bit will be one when α is either zero or one and it will decrease to .5 as α either increases or decreases to $1/\sqrt{2}$. So, the greater the superposition of Alice's original bit, the more the bit will become entangled (with some combination of the other two bits).

Now look at Bob's bit:

$$\rho_b = \begin{pmatrix} \dfrac{\alpha\alpha^* + \beta\beta^*}{2} & 0 \\ 0 & \dfrac{\alpha\alpha^* + \beta\beta^*}{2} \end{pmatrix} = \begin{pmatrix} \frac{1}{2} & 0 \\ 0 & \frac{1}{2} \end{pmatrix}$$

The DM for Bob's bit shows a 50/50 mixture, which in this context means that it's maximally entangled. This is true regardless of the values of α and β.

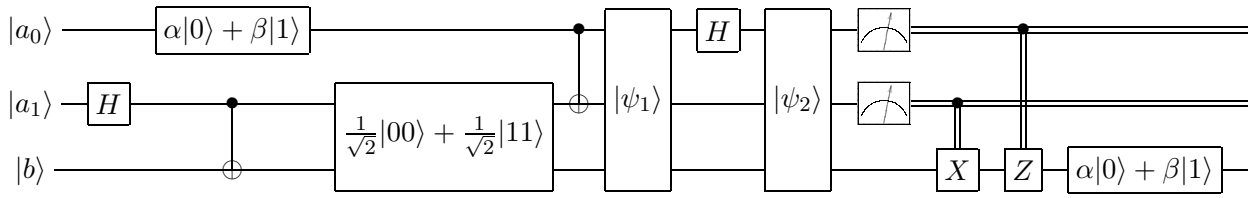
If the degree of entanglement of Alice's zero bit can vary, and Bob's bit is definitely maximally entangled, then it seems like something interesting must be happening with Alice's bit 1. However, that DM comes out to be exactly the same as the one for Bob's bit. So it's fully entangled too, regardless of the value of a_0.

The story actually appears to be in the DM for Alice's 1 bit and Bob's bit combined. If you trace out just Alice's 0 bit you get:

$$\begin{aligned} \rho \;=\; & \frac{\alpha\alpha^*}{2}\left|00\right\rangle\left\langle 00\right| + \frac{\alpha\alpha^*}{2}\left|00\right\rangle\left\langle 11\right| + \frac{\beta\beta^*}{2}\left|01\right\rangle\left\langle 01\right| + \frac{\beta\beta^*}{2}\left|01\right\rangle\left\langle 10\right| \\ & + \frac{\beta\beta^*}{2}\left|10\right\rangle\left\langle 01\right| + \frac{\beta\beta^*}{2}\left|10\right\rangle\left\langle 10\right| + \frac{\alpha\alpha^*}{2}\left|11\right\rangle\left\langle 00\right| + \frac{\alpha\alpha^*}{2}\left|11\right\rangle\left\langle 11\right|, \\ & \text{purity} = \left(\alpha\alpha^*\right)^2 + \left(\beta\beta^*\right)^2 \end{aligned}$$

This is the same purity we got for Alice's 0 bit. So the joint state of bits a_1 and b has the same degree of entanglement as a_0 regardless of the original state of a_0. The numbers work out, but that still seems to leave a little bit of ambiguity about exactly "what's entangled with what."

Now looking at the state after Alice does a Hadamard on a_0:



$$\psi_2 = \frac{1}{2}(\,\alpha|000\rangle + \alpha|011\rangle + \alpha|100\rangle + \alpha|111\rangle + \beta|001\rangle + \beta|010\rangle - \beta|101\rangle - \beta|110\rangle$$

I worked these out on the computer. I'll post the notebooks if anybody wants to see them. It doesn't look like the "entanglement" story has changed by doing that Hadamard. Should we have figured that out in advance? Logically, you can't *entangle* bits by only operating on one of them, but can you *break entanglement* with a unitary one-bit operator? Anyway, nothing changed here. a_0 has a different DM but the same purity:

$$\text{Bit 0}: \quad \rho = \begin{pmatrix} \dfrac{\alpha\alpha^* + \beta\beta^*}{2} & \dfrac{\alpha\alpha^* - \beta\beta^*}{2} \\ \dfrac{\alpha\alpha^* - \beta\beta^*}{2} & \dfrac{\alpha\alpha^* + \beta\beta^*}{2} \end{pmatrix}, \quad \text{purity} = \left(\alpha\alpha^*\right)^2 + \left(\beta\beta^*\right)^2$$

$a1$ and b come out exactly the same as they did for ψ_1.

9 Extension and generalization of CNOT

Suppose you want to do something like a CNOT, but flipping the target bit if the control bit is zero rather than one. Or maybe the program calls for changing a certain bit if two or three other bits are in a given state. The CNOT operation can be easily generalized to do these things.

Consider the operation of a simple CNOT. Below we have it in matrix form operating on a vector that represents the tensor product of the control and target bits. (c_0, c_1 and t_0, t_1 are the elements of the control and target bits respectively.) What this matrix does is simply to leave the top two elements of the vector alone, and interchange the bottom two. For "classical" states, the only two values that have a one in the control bit are $|10\rangle$ and $|11\rangle$:

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} c_0\, t_0 \\ c_0\, t_1 \\ c_1\, t_0 \\ c_1\, t_1 \end{pmatrix} = \begin{pmatrix} c_0\, t_0 \\ c_0\, t_1 \\ c_1\, t_1 \\ c_1\, t_0 \end{pmatrix}, \quad \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} \rightarrow \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}, \quad \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} \rightarrow \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix}$$

The general CNOT operation, $\quad |10\rangle \rightarrow |11\rangle, \quad |11\rangle \rightarrow |10\rangle$

If the control bit is in a superposition of one and zero, the swapping of the bottom two elements will have a more complicated effect on the outcome. But the operation is probably best understood by thinking about its effect on classical bits and then working out everything else in Dirac notation using linearity.

If more than one control bit is needed, all you need to do is to make the CNOT matrix larger and put ones in the main diagonal. The "lower right" four bits will always remain in the NOT pattern. For example:

$$\text{CNOT}_3 = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}$$

This 8×8 matrix will perform a CNOT with two control bits and one target bit

$$|110\rangle \rightarrow |111\rangle, \quad |111\rangle \rightarrow |110\rangle$$

Suppose we want to flip the target bit if the control bit is zero rather than one. We create an operator to do this by multiplying CNOT, on both sides, by X (the regular one-bit NOT) tensored with I (the identity operator):

$$\text{NEW\_OPERATOR} = (X \otimes I)\text{CNOT}(X \otimes I)$$

Think of it this way. We are tensoring X with I because we need a two-bit matrix to multiply CNOT. We want to invert the sense of the control bit, to enable the operation on zero rather than one. So the X is operating on the first bit. We are leaving the operation that is *done* to the target bit alone, so we have I operating on the second bit.

This generalizes easily. To create a three-bit operator which will flip the target bit if the two control bits are 10 we would do:

$$\text{NEW\_OPERATOR} = (I \otimes X \otimes I)\text{CNOT}_3(I \otimes X \otimes I)$$

As you can see, this provides a way to have a generalized control operation that will flip a target bit given any number of control bits, representing any desired number. Let's give this kind of operation a name. We'll call it $\text{CTL}_b(n, X)$, where b is the number of bits in the operation, n is the value that will cause the operation to be applied to the target bit, and X is the operation that will potentially be done. So, for example, $\text{CTL}_4(5, X)$ would be an operator with three control bits and one target bit. It would be a "four-bit" matrix (which is 16×16) and it would flip the target bit if the three control bits represent the number five (that is, the bit pattern 101).

We would create the operator like this:

$$\text{CTL}_4(5, X) = (I \otimes X \otimes I \otimes I)\text{CNOT}_4(I \otimes X \otimes I \otimes I)$$

Finally, it may be of interest to look at the operation of a CNOT worked out in Dirac notation:

$$\begin{aligned}
\text{CNOT}\,|00\rangle &= \big(\,|00\rangle\langle 00| + |01\rangle\langle 01| + |10\rangle\langle 11| + |11\rangle\langle 10|\,\big)\,|00\rangle \\
&= |00\rangle\langle 00||00\rangle + |01\rangle\langle 01||00\rangle + |10\rangle\langle 11||00\rangle + |11\rangle\langle 10||00\rangle \\
&= |00\rangle \;\text{ (All the inner products but the first one are zero.)}
\end{aligned}$$

$$\begin{aligned}
\text{CNOT}\,|10\rangle &= \big(\,|00\rangle\langle 00| + |01\rangle\langle 01| + |10\rangle\langle 11| + |11\rangle\langle 10|\,\big)\,|10\rangle \\
&= |00\rangle\langle 00||10\rangle + |01\rangle\langle 01||10\rangle + |10\rangle\langle 11||10\rangle + |11\rangle\langle 10||10\rangle \\
&= |11\rangle \;\text{ (This time only that last inner product survives.)}
\end{aligned}$$

10 General control operations

Suppose we want to exercise control over some other operation than NOT. A more general control operation is constructed in the same way as the generalized CNOT in the last section. All that needs to be done is to replace the "lower right" four entries in the matrix with that operation. Here are two examples, controlled H (Hadamard) and controlled Z (phase flip):

$$\mathrm{CTL}_2(1,H) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ 0 & 0 & \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{pmatrix}, \quad \mathrm{CTL}_2(1,Z) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix}$$

A word about the notation I'm using for control operations. The subscript indicating the number of bits does not refer to the number of control bits, but rather the number of bits in the "space" we're working in. This will be one greater than the number of control bits, because it includes the target bit. So there is no CTL_1. The subscript may be omitted if the number of bits is obvious from the context, or if it's a basic two-bit controlled operation.

Just as with the controlled not operations, we can trigger the operation on any desired number by surrounding the CTL operation with Xs for the bits we want to be zero instead of one. Some examples:

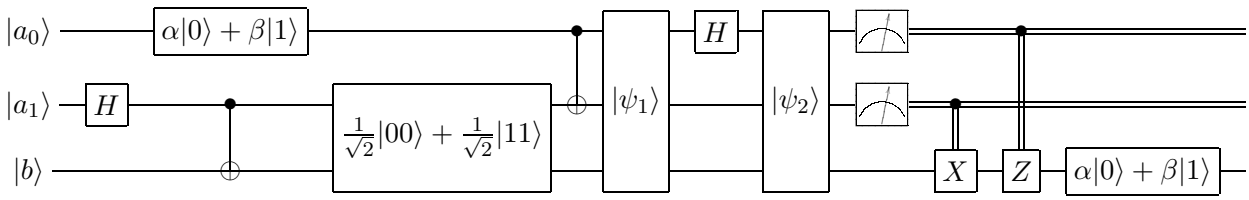
$$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \; Y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}, \; Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$

$$\mathrm{CTL}_2(0,H) = \big(X \otimes I\big)\ \mathrm{CTL}_2(1,H)\ \big(X \otimes I\big)$$

$$\mathrm{CTL}_3(2,Y) = \big(I \otimes X \otimes I\big)\ \mathrm{CTL}_3(1,Y)\ \big(I \otimes X \otimes I\big)$$

$$\mathrm{CTL}_5(9,Z) = \big(I \otimes X \otimes X \otimes I \otimes I\big)\ \mathrm{CTL}_5(1,Z)\ \big(I \otimes X \otimes X \otimes I \otimes I\big)$$

We have actually used CTL(1,Z) already (along with CTL(1,X) which is nothing but a CNOT) in the last step of the quantum teleportation program, where both operations are used to adjust Bob's state based on Alice's measurement of her two bits:



11 The effects of CNOT and Hadamard on entanglement

Almost any quantum algorithm starts right off by using Hadamard and CNOT gates to put bits into superposition and entangle them. We looked at how to extend CNOT (and other control operations). Now we'll a spend little more time to understand the details of how CNOT and Hadamard gates work together to produce entanglement between bits.

The basic CNOT operation

The most obvious use of a CNOT is to "flip bits." (Actually, in more general terms what a NOT does is to interchange the probability amplitudes of the state it operates on.) But there is another reason why the CNOT is arguably the most important quantum gate. That is because *it has the potential to entangle bits.*

Consider these circuits:

(1) $|1\rangle \otimes |0\rangle = |10\rangle \rightarrow \text{CNOT} \rightarrow |11\rangle = |1\rangle \otimes |1\rangle$

(2) $|1\rangle \otimes |1\rangle = |11\rangle \rightarrow \text{CNOT} \rightarrow |10\rangle = |1\rangle \otimes |0\rangle$

(3) $|1\rangle \otimes \left(\frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle \right) = \frac{1}{\sqrt{2}}|10\rangle + \frac{1}{\sqrt{2}}|11\rangle \rightarrow \text{CNOT} \rightarrow \frac{1}{\sqrt{2}}|11\rangle + \frac{1}{\sqrt{2}}|10\rangle = |1\rangle \otimes \left(\frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle \right)$

(4) $|1\rangle \otimes \left(\frac{1}{2}|0\rangle + \frac{\sqrt{3}}{2}|1\rangle \right) = \frac{1}{2}|10\rangle + \frac{\sqrt{3}}{2}|11\rangle \rightarrow \text{CNOT} \rightarrow \frac{1}{2}|11\rangle + \frac{\sqrt{3}}{2}|10\rangle = |1\rangle \otimes \left(\frac{\sqrt{3}}{2}|0\rangle + \frac{1}{2}|1\rangle \right)$

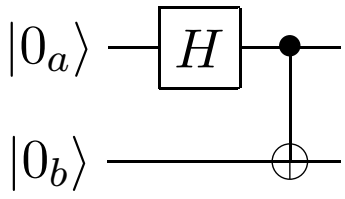
(5) $\left(\frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle \right) \otimes |0\rangle = \frac{1}{\sqrt{2}}|00\rangle + \frac{1}{\sqrt{2}}|10\rangle \rightarrow \text{CNOT} \rightarrow \frac{1}{\sqrt{2}}|00\rangle + \frac{1}{\sqrt{2}}|11\rangle = \text{entangled!}$

(6) $\left(\frac{1}{2}|0\rangle + \frac{\sqrt{3}}{2}|1\rangle \right) \otimes |1\rangle = \frac{1}{2}|01\rangle + \frac{\sqrt{3}}{2}|11\rangle \rightarrow \text{CNOT} \rightarrow \frac{1}{2}|01\rangle + \frac{\sqrt{3}}{2}|10\rangle = \text{entangled!}$

When the control bit is a classical zero, the CNOT doesn't do anything. When the control bit is a classical one (circuits 1 and 2 above) it flips the target bit as expected. If the target bit is in a superposition (3 and 4 above), each part of the superposition gets its bit flipped, which results in interchanging the probability amplitudes of the target state.

In any case, as long as the control bit is either a classical zero or one, the resulting state can be factored and no entanglement takes place. However, if the control bit itself is in a superposition (5 and 6 above) then the resulting state cannot be factored, and entanglement results.

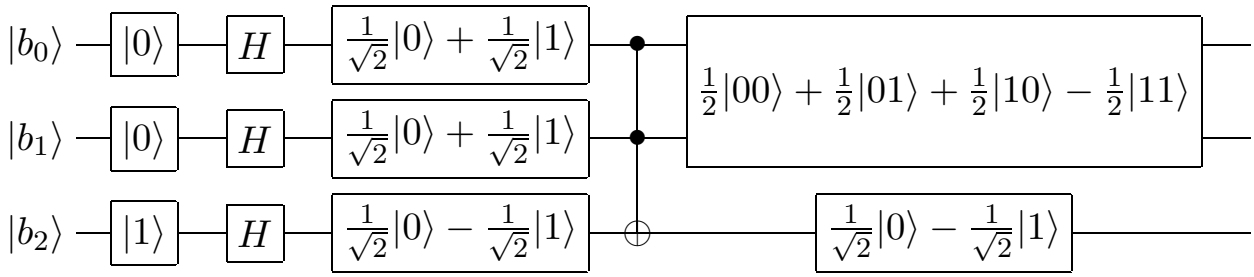
Mixing CNOTs and Hadamards



In the beginning of the teleportation scenario, Alice and Bob are physically in the same place. Therefore Alice can operate on both of their bits. Both of their bits start out as zeros. The initial entanglement is accomplished by Alice doing a Hadamard on her bit to put it into an equal superposition. Then she does a CNOT using her bit as the control and Bob's as the target. This is how the entangled state they share is initially created. After that they can separate, taking their respective bits with them, and during the remainder of the process they can operate at any distance from one another.

This combination of Hadamard followed by CNOT is a common way to begin a quantum program. Often *all* of the bits are put into superposition with Hadamards right at the start, followed by some kind of extended CNOT. But the results may not always be what you might expect.

Consider the following circuit:



If you think about a CNOT "classically" then you might suppose that only the "target" bit would be modified. But, in the circuit above for example, it turns out that the $\mathrm{CTL}_3(3, X)$ (a three-bit CNOT that will trigger if the two control bits are both one) ends up *entangling the two control bits* and not changing the target bit at all.

Here's how this happens. First note that the target bit b_2 starts off as a one, so the phase of its superposition is minus rather than plus. Then write the joint state of all three bits after the Hadamards:

$$\frac{\sqrt{2}}{4}\Big(|000\rangle - |001\rangle + |010\rangle - |011\rangle + |100\rangle - |101\rangle + |110\rangle - |111\rangle\Big)$$

Then do the CNOT. Remember that it triggers when the first two bits are 11. It will change only the last two terms.

$$\frac{\sqrt{2}}{4}\Big(|000\rangle - |001\rangle + |010\rangle - |011\rangle + |100\rangle - |101\rangle + |111\rangle - |110\rangle\Big)$$

Switch the last two terms, putting them back in increasing order of bit values.

$$\frac{\sqrt{2}}{4}\Big(|000\rangle - |001\rangle + |010\rangle - |011\rangle + |100\rangle - |101\rangle - |110\rangle + |111\rangle\Big)$$

As you can see, all that the operation has actually done is to flip the phase of the part of the superposition where the control bits were 11. This gives a hint about how to factor out the target bit.

Take an equal superposition of the control bits:

$$\frac{1}{2}\big(\,|00\rangle + |01\rangle + |10\rangle + |11\rangle\,\big)$$

Flip the phase when the bits equal 3:

$$\frac{1}{2}\big(\,|00\rangle + |01\rangle + |10\rangle - |11\rangle\,\big)$$

Assume the target bit hasn't changed at all:

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

And there you have it:

$$\frac{1}{2}\big(\,|00\rangle + |01\rangle + |10\rangle - |11\rangle\,\big) \otimes \frac{1}{\sqrt{2}}\big(\,|0\rangle - |1\rangle\,\big)$$

So the main point here is, when you deal with various kinds of superpositions and control operations, you will generally get some kind of entanglement. But the details will be dependent on exactly what's going on in the circuit, and may be counter-intuitive.

Part II

Algorithms

12 Grover's Algorithm

Why are quantum computers expected to be so much more powerful than classical computers? I think, if there's a short answer to this question, it must be that quantum computers can (in some sense) *run an algorithm on all the inputs at the same time.*

How does this happen? It goes something like this:

1. Put all the input bits into an equal superposition
2. Run the program
3. Pick out the answer from the resulting superposition

Number 3 is the tricky part. Grover's algorithm gives a reasonably easy to understand example of how this can be done. There are basically two parts. First there is an operation that somehow "flags" the part of the superposition that you want. This part is usually referred to as the "oracle" and is dependent on the specific problem. So this is the hard part. Then there is what's called the "diffusion transform" which increases the probability amplitude of the flagged part of the superposition, so that when you finally do a measurement, that's the result that you will (most likely) see.

The Oracle

The Diffusion Transform

Grovers_Algorithm.ipynb

13 Simon's Algorithm

14 Shor's Algorithm