Lecture 16: Quantum error correction

March 21, 2006

Any physical realization of a quantum computer is likely to be susceptible to errors (such as noise and inaccuracies)—we cannot build perfect physical systems and isolate them from their environments while still maintaining control over them. In this lecture and the next, we will discuss quantum error correction, which aims to protect quantum information against such errors.

Classical repetition codes

We will start with a very simple classical error correcting code, the *3 bit repetition code*. Initially, just for the sake of having a concrete error model to think about, we will consider errors described by the *binary symmetric channel*. Here, we have a channel from Alice to Bob that carries 1 bit at a time. There is noise on the channel, so the bit sent by Alice is not always received correctly by Bob. Specifically, we suppose that the noise is parameterized by some real number $p \in [0,1]$ that represents the probability of an error: if Alice sends the bit b, then Bob receives b with probability $1-p$ and $\neg b$ with probability p:

$$b \longmapsto \begin{cases} b & \text{with probability } 1-p \\ \neg b & \text{with probability } p. \end{cases}$$

Suppose that Alice needs to send a very important bit to Bob, and the two of them do not wish to accept that Bob will receive the wrong bit with probability p. What can they do? The answer is that they can use an error correcting code to decrease the probability of error.

Possibly the simplest example of an error correcting code is the 3 bit repetition code. The encoding is simply to repeat the bit to be transmitted three times, and decoding is just taking the most frequently appearing bit (which is the same as computing the majority function).

encoding	transmission (3 bits)	decoding
$0 \to 000$	$\longrightarrow$	$abc \to \mathrm{majority}(a,b,c)$
$1 \to 111$		

If 0 or 1 of the bits are flipped during transmission, then Bob will recover the correct bit sent by Alice. The probability of error shrinks from p to $3\,p^2 - 2\,p^3$:

$$\Pr[\text{correct transmission}] = (1-p)^3 + 3\,(1-p)^2 p = 1 - (3\,p^2 - 2\,p^3).$$

(Really this is only a reduction in error when $p < 1/2$. If $p = 1/2$, then the channel is useless for transmitting information, and if $p > 1/2$ then Bob can simply flip every bit which effectively replaces p with $1-p$.)

Another way of explaining what this code does (without making any reference to the binary symmetric channel) is to say that it encodes 1 bit into 3 in a way that protects against one bit-flip. Of course much more efficient (and more complicated) error correcting codes exist—this is indeed a very simple code.

Protecting quantum information against bit-flips

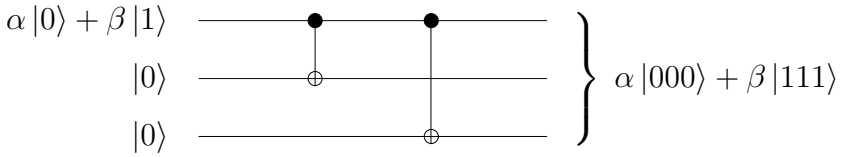
Can we encode quantum information in a similar way to the previous example? For instance, suppose Alice has a qubit in the pure state $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ that she wishes to send to Bob through a noisy quantum channel. We will have to discuss exactly what a noisy quantum channel is, but let us just think about this notion informally for now. Can something analogous to the repetition code be used?

First let us note that it is not acceptable to encode $|\psi\rangle$ as $|\psi\rangle|\psi\rangle|\psi\rangle$. In case Alice does not know α and β, this would violate the no-cloning theorem. It is also not clear how Bob would decode.

A more sensible analogue of the repetition code is to perform this encoding:

$$\alpha|0\rangle + \beta|1\rangle \mapsto \alpha|000\rangle + \beta|111\rangle\,.$$

This encoding could be performed by the following simple circuit:



Now, this encoding protects against some errors, but actually makes things worse for others as we will see. Let us start off with a type of error that it does correct against: bit-flip errors. Specifically, consider a *quantum* channel that is similar to the classical binary symmetric channel, again parameterized by a number $p \in [0, 1]$:

> With probability $1 - p$, the channel acts as the identity. With the remaining probability p, the channel applies a bit-flip error, meaning that the unitary operation
>
> $$\sigma_x = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$
>
> is performed.

This channel can therefore be described by the admissible operation

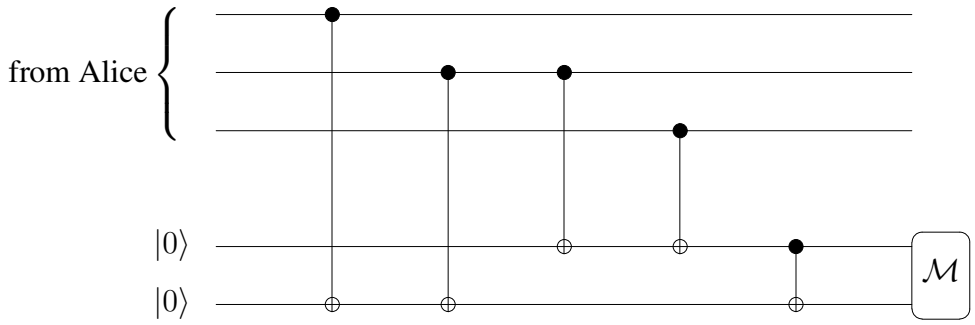
$$\Phi(\rho) = (1-p)\,\rho + p\,\sigma_x\rho\sigma_x^\dagger = (1-p)\,\rho + p\,\sigma_x\rho\sigma_x.$$

Suppose that Alice encodes $\alpha|0\rangle + \beta|1\rangle$ as $\alpha|000\rangle + \beta|111\rangle$, and sends the three qubits through the channel (one at a time). Imagine that it happens that the second qubit experiences a bit-flip error, but the first and third are unaffected. The pure state of the three qubits therefore becomes

$$\alpha|010\rangle + \beta|101\rangle\,.$$

How would Bob decode? Of course he does not know that the bit-flip occurred, and he does not want to measure the three qubits and compute the majority because that would ruin the qubit sent by Alice. Bob's goal is to recover the qubit $\alpha|0\rangle + \beta|1\rangle$.

Consider the following circuit:



Consider what this circuit does in the present case (where a bit-flip error occurred on the second qubit):

$$\begin{aligned}(\alpha|010\rangle + \beta|101\rangle)|00\rangle &= \alpha|010\rangle|00\rangle + \beta|101\rangle|00\rangle \\ &\mapsto \alpha|010\rangle|10\rangle + \beta|101\rangle|10\rangle \\ &= (\alpha|010\rangle + \beta|101\rangle)|10\rangle\,.\end{aligned}$$

The measurement would give outcome 10 (with certainty). This output string is called the *syndrome*; in this case it tells us that a bit-flip error occurred on qubit number 2 (or 10 in binary). So, Bob corrects the error by applying σ_z to qubit number 2:

$$\alpha|010\rangle + \beta|101\rangle \mapsto \alpha|000\rangle + \beta|111\rangle\,.$$

Now, by applying the inverse of the encoding procedure he recovers $\alpha|0\rangle + \beta|1\rangle$:

$$\alpha|000\rangle + \beta|111\rangle \mapsto (\alpha|0\rangle + \beta|1\rangle)|00\rangle\,.$$

The same procedure works in the case that the first or third qubit experiences a bit-flip as well. To see this, consider the syndrome that would result from any classical state:

Classical state	Syndrome
$\|000\rangle$	00
$\|001\rangle$	11
$\|010\rangle$	10
$\|011\rangle$	01
$\|100\rangle$	01
$\|101\rangle$	10
$\|110\rangle$	11
$\|111\rangle$	00

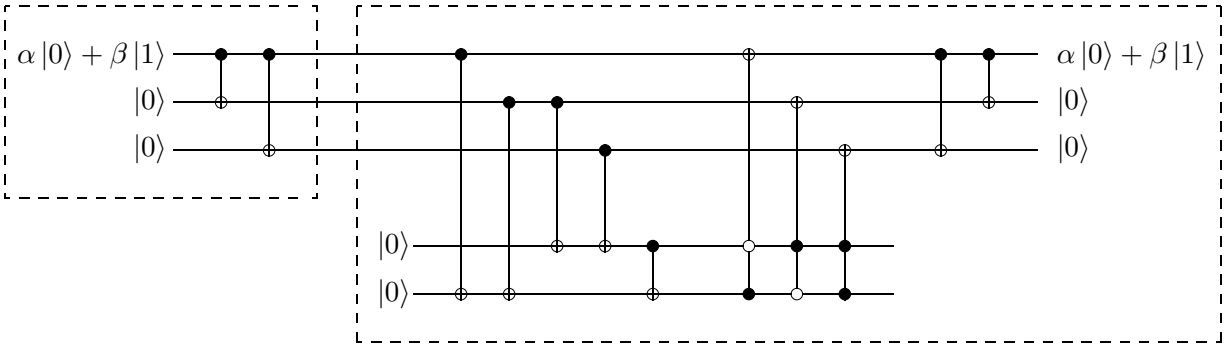
If no errors occur, or a single bit-flip occurs, the syndrome will correctly diagnose the errors (or lack of errors):

- No errors $\Rightarrow$ Syndrome = 00.
- σ_x applied to qubit number $j \in \{1, 2, 3\} \Rightarrow$ Syndrome = j (in binary).

In general, the decoding procedure is as follows.

1. If the syndrome is 00, do nothing. Otherwise, if the syndrome is $j \in \{1, 2, 3\}$ (in binary), apply a σ_x operation to qubit number j.
2. Apply the reverse of the encoding procedure.

The entire code can be represented by a diagram as follows:



The code corrects up to one bit-flip error. If two or more bit-flip errors occurred, there are no guarantees...

You could analyze the precise behavior for the channel described above, but we will not do this. You would find (assuming $p < 1/2$) that the probability of a correct transmission would increase. (There are more precise technical ways of characterizing how close the output density matrix would be to $|\psi\rangle\langle\psi|$, but this would be too much of a digression at this point in the course.)

Other quantum errors

Although the above code protects against bit-flips, this is not good enough in general. The problem is that there are different types of quantum errors. For example, a *phase-flip* error could occur:

$$|a\rangle \mapsto (-1)^a |a\rangle$$

for $a \in \{0, 1\}$. This error is therefore represented by the σ_z matrix:

$$\sigma_z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}.$$

The previous code does nothing to protect against phase-flips. For instance, we have

$$\alpha|000\rangle + \beta|111\rangle \mapsto \alpha|000\rangle - \beta|111\rangle$$

if any odd number of phase-flips occur. If, say, each phase-flip occurs with probability $p \in (0, 1/2)$, then the probability that an odd number occur is

$$3p(1-p)^2 + p^3 > p.$$

In other words, the code is making things *worse* rather than better.

Suppose that we are interested in protecting against phase errors. We could change the encoding slightly in order to do this:

$$\alpha |0\rangle + \beta |1\rangle \mapsto \alpha |+\rangle |+\rangle |+\rangle + \beta |-\rangle |-\rangle |-\rangle .$$

This is easily done by encoding $\alpha |0\rangle + \beta |1\rangle$ as $\alpha |000\rangle + \beta |111\rangle$ just like before, followed by a Hadamard transform on each qubit. The effect of a phase-flip on the basis $\{|+\rangle , |-\rangle\}$ is similar to the effect of a bit-flip on the standard basis:

$$\sigma_z |+\rangle = |-\rangle , \quad \sigma_z |-\rangle = |+\rangle .$$

Therefore, Bob can easily correct against a phase-flip on a single qubit by first applying Hadamard transforms to all three qubits, and then correcting as before. For instance, if a phase-flip happens on qubit number 1, then the encoding

$$\alpha |+\rangle |+\rangle |+\rangle + \beta |-\rangle |-\rangle |-\rangle$$

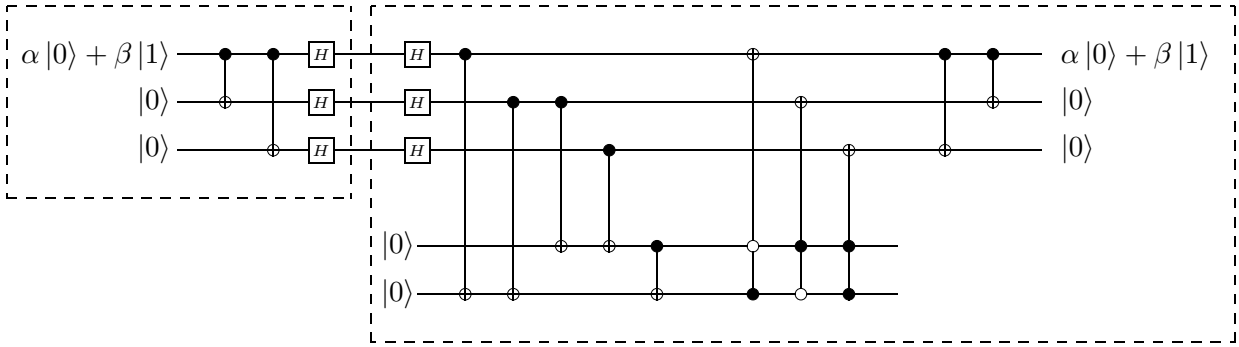
changes to

$$\alpha |-\rangle |+\rangle |+\rangle + \beta |+\rangle |-\rangle |-\rangle .$$

Bob applies Hadamard transforms to all three qubits to obtain

$$\alpha |100\rangle + \beta |011\rangle .$$

He then corrects just as before to obtain $\alpha |0\rangle + \beta |1\rangle$. This process can be described by the following circuit:

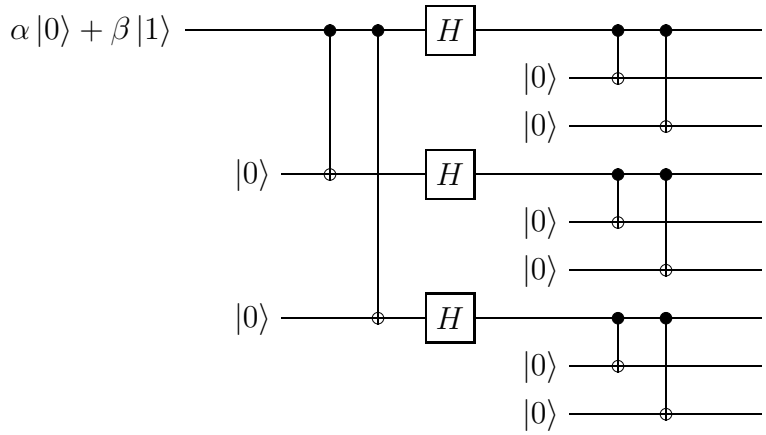


As you would expect, although the new code protects against phase-flips, it fails to protect against bit-flips (and in fact makes matters worse for bit-flips just as happened previously for phase-flips).

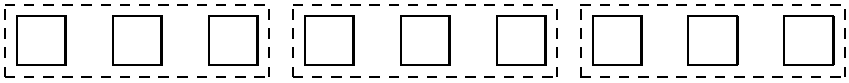
Is there any way to protect against both bit flips and phase flips simultaneously? The answer is yes, but we need to *concatenate* the codes. In other words, we first apply the code that protects against phase-flips, and then encode *each* of the three resulting qubits using the code that protects against bit-flips. In other words, we will encode $\alpha|0\rangle + \beta|1\rangle$ as

$$\alpha \frac{(|000\rangle + |111\rangle)(|000\rangle + |111\rangle)(|000\rangle + |111\rangle)}{2\sqrt{2}}$$
$$+\beta \frac{(|000\rangle - |111\rangle)(|000\rangle - |111\rangle)(|000\rangle - |111\rangle)}{2\sqrt{2}}.$$

This is known as Shor's 9 qubit code, because it was first discovered by Shor. A circuit for the encoding procedure looks like this:



Now, how does Bob decode? Intuitively it is easy—he first imagines that the 9 qubits form 3 blocks of 3 qubits:

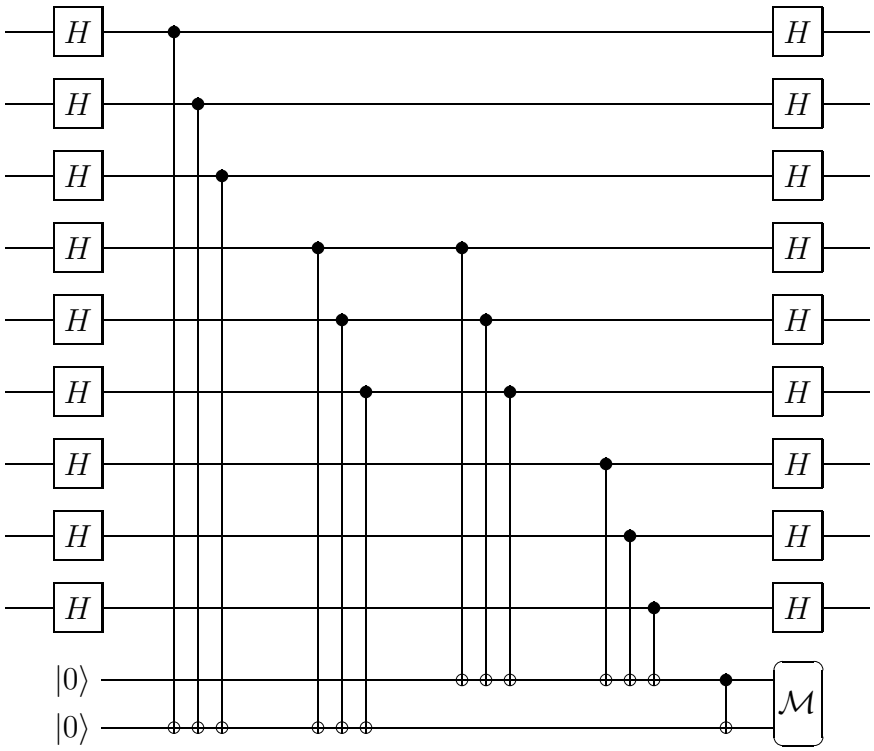


Each block represents an encoding of one of the three qubits of the state

$$\alpha|+\rangle|+\rangle|+\rangle + \beta|-\rangle|-\rangle|-\rangle,$$

where the encoding is with respect to the bit-flip code. He corrects any bit flips as before, and decodes each block to give a total of three qubits (the first in each block). Assuming at most one bit-flip occurred in each block, the remaining 6 qubits will have all returned to the state $|0\rangle$, and can be discarded or re-used. When it is time to correct phase flips, the process is again as before.

Sometimes it is convenient to separate the decoding phase into two parts—the first being to correct errors so that the original *encoding* is recovered, and the second being to transform the encoding back to the original qubit (which is obtained by running the encoding procedure in reverse). In the present case this is done by correcting bit-flip errors in each block as above and then running a slightly modified procedure to correct phase errors:



The measurement gives a syndrome that specifies, in binary, which of the three blocks has received a phase-flip error. Any phase-flip error can be corrected by applying a σ_z operation to **any one** of the three qubits in that block (or all three if you prefer).

Fact. The Shor 9 qubit code can simultaneously protect against up to 1 bit-flip error in each block and any number of phase-flip errors contained in any one block. In particular, it can protect against any one of the four "errors"

$$I,\ \sigma_x,\ \sigma_z,\ \sigma_x\sigma_z$$

occurring on a single qubit.

Example 1. Suppose the error $\sigma_x\sigma_z$ occurs on qubit number four. The encoded state becomes

$$\alpha\,\frac{(|000\rangle+|111\rangle)(|100\rangle-|011\rangle)(|000\rangle+|111\rangle)}{2\sqrt{2}}$$
$$+\beta\,\frac{(|000\rangle-|111\rangle)(|100\rangle+|011\rangle)(|000\rangle-|111\rangle)}{2\sqrt{2}}.$$

The bit-flip code correcting procedure is applied to each block. This results in three 2-bit syndromes, which in this case are 00, 01, and 00, respectively—the first block has no bit-flip errors, the second has a bit-flip error in position 1, and the third block has no bit-flip errors. The appropriate correction for each block results in the state

$$\alpha\,\frac{(|000\rangle+|111\rangle)(|000\rangle-|111\rangle)(|000\rangle+|111\rangle)}{2\sqrt{2}}$$
$$+\beta\,\frac{(|000\rangle-|111\rangle)(|000\rangle+|111\rangle)(|000\rangle-|111\rangle)}{2\sqrt{2}}.$$

Now the phase-flip code correcting procedure is applied, and this gives a single 2-bit syndrome; you can check that the procedure above gives syndrome 10 in this case. This implies that a phase-flip happened somewhere in block 2. Applying a σ_z gate to any qubit in this block results in

$$\alpha \frac{(|000\rangle + |111\rangle)(|000\rangle + |111\rangle)(|000\rangle + |111\rangle)}{2\sqrt{2}}$$
$$+\beta \frac{(|000\rangle - |111\rangle)(|000\rangle - |111\rangle)(|000\rangle - |111\rangle)}{2\sqrt{2}},$$

which is the original encoding.

Next time we will see how arbitrary errors on a single qubit can be corrected by this code (and more generally by any code that can correct the four errors represented by the Pauli matrices: $I,\ \sigma_x,\ \sigma_z,\ \sigma_y = -i\sigma_x\sigma_z$).