Notes 8.370/18.435 Fall 2022

Lecture 30 Prof. Peter Shor

Last time we talked about the quantum Hamming code. This was based on the classical Hamming code. We have two matrices, G and H, with

$$G = \begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix} \quad \text{and} \quad H = \begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{pmatrix}$$

The quantum error correcting code they generated mapped $|0\rangle$ to

$$|0\rangle_L = \frac{1}{\sqrt{8}} \sum_{x \in H} |x\rangle \,,$$

and mapped $|1\rangle$ to

$$|1\rangle_L = \frac{1}{\sqrt{8}} \sum_{x \in H} |x \oplus 1111111\rangle \ .$$

Recall that a classical binary linear code is a subspace of $\mathbb{Z}_2^n$. If the minimum non-zero codeword has Hamming weight d, then the code is said to have distance d. Recall that a code with distance d can correct up to $t = \frac{d-1}{2}$ errors. A code that maps k bits into n bits and has distance d is called an $[n, k, d]$ code. For example, the Hamming code with generator matrix G is a $[7, 4, 3]$ code, and the code with generator matrix H is a $[7, 3, 4]$ code.

A CSS code is a quantum error-correcting code that is derived from two classical codes, C_1 and C_2, with $C_2 \subseteq C_1$. We will call this code CSS($C_1 : C_2$). If C_1 can correct t_1 errors and $C_2^\perp$, the dual code to C_2, can correct t_2 errors, then CSS($C_1 : C_2$) can correct t_1 bit-flip errors and t_2 phase-flip errors.

Let me remark that one can consider a quantum error-correcting code to either be a collection of codewords or the subspace generated by these words. Since quantum error-correcting codes must correct superpositions of codewords as well as the codewords themselves, they must correct every quantum state in the subspace. For CSS codes, it doesn't really matter which of these perspectives you use, but for some other quantum error-correcting codes, the subspace view is a more intuitive way to think about it.

Before we give the definition of a quantum CSS code, we need to define a few terms in classical error correcting codes. If $C_2 \subseteq C_1$, a *coset* of C_2 in C_1 is the set of vectors

$$x + C_2 = \{y | y = x + c,\ c \in C_2\}$$

for some $x \in C_1$. Two cosets $x_1 + C_2$ and $x_2 + C_2$ either are equal or are disjoint, and every element $x \in C_1$ is a member of some coset, so the number of cosets is $\frac{|C_1|}{|C_2|}$, i.e., the number of elements of C_1 divided by the number of elements of C_2.

Why are two cosets either equal or disjoint? Suppose that $y \in x + C_2$ Then $y = x + d$ for some $d \in C_2$. Now, if we add d to any other element in C_2, we still have an element in C_2, so

$$y{+}C_2 = \{z|z = y{+}c,\ c \in C_2\} = \{z|z = x{+}d{+}c,\ c \in C_2\} = \{z|z = x{+}c,\ c \in C_2\} = x{+}C_2.$$

For example, in the Hamming code, there are two cosets of H in G,

$$H \quad \text{and} \quad H + 1111111\,.$$

Now, we can give the codewords of the CSS code associated with two n-bit classical codes C_1 and C_2 with $C_2 \subset C_1$. The codewords of this code will be the associated with the cosets of C_2 in C_1. For each coset $x + C_2$, we have the codeword

$$|x + C_2\rangle = \frac{1}{|C_2|^{1/2}} \sum_{c \in C_2} |x + c\rangle\,.$$

The number of codewords is the number of cosets of C_2 in C_1, which is $\frac{|C_1|}{|C_2|}$. The dimension of the code subspace is

$$\dim \mathrm{CSS}(C_1 : C_2) = \log_2 \frac{|C_1|}{|C_2|} = \dim C_1 - \dim C_2$$

Why does CSS(C_1, C_2) correct t_1 bit-flip errors? We know that there is a classical error-correction procedure that will correct t_1 or fewer errors in any codeword c_1 of C_1. Now, the codewords of CSS(C_1, C_2) are composed of superposition of quantum states $|x + c\rangle$, each of which is a codeword of C_1. Thus, applying this classical error-correction procedure in quantum superposition will correct up to t_1 bit-flip errors in the quantum code.

We now need to explain why the dual code will correct t_2 phase errors. Recall that the dual of a subspace W of a vector space V was

$$W^\perp = \{x \in V | x \cdot w = 0\ \forall w \in W\}$$

If you're used to working with real and complex vector spaces, a non-intuitive fact about binary vector spaces is that the original space and its dual can overlap. For example, with the Hamming code above, $H = G^\perp \subset G$. Howerver, it is still true that if $W^\perp$ is the dual of W in vector space V, then

$$\dim V = \dim W + \dim W^\perp\,.$$

We won't prove it in these notes, but there is a proof of this along the lines of the Gaussian elimination manipulations we did in our discussion of Simon's algorithm.

Let's apply the Hadamard transform to the codewords $|x + C_2\rangle$. It turns out that what we get are superpositions of the codewords of the code CSS($C_2^\perp : C_1^\perp$). Before we prove this, let me make a little side comment about why this makes sense. because $C_2 \subseteq C_1$, any vector that is perpendicular to everything in C_1 is also perpendicular

to everything in C_2, so $C_1^\perp \subseteq C_2^\perp$. Further, $\dim C_1^\perp = n - \dim C_1$ and $\dim C_2^\perp = n - \dim C_2$, so

$$\dim \mathrm{CSS}(C_1 : C_2) = \dim \mathrm{CSS}(C_2^\perp : C_1^\perp)\,.$$

Now, let's show that the Hadamard transform of a codeword $|x + C_2\rangle$ of CSS($C_1 : C_2$) is a superposition of codewords of CSS($C_2^\perp : C_1^\perp$). Taking the Hadamard transform, we find:

$$\begin{aligned} H^{\otimes n} |x + C_2\rangle &= \frac{1}{\sqrt{2^{\dim C_2}}} H^{\otimes n} \sum_{c_2 \in C_2} |c_2 + x\rangle \\ &= \frac{1}{\sqrt{2^{\dim C_2}}} \frac{1}{\sqrt{2^n}} \sum_{y \in Z_2^n} \sum_{c_2 \in C_2} (-1)^{(c_2+x)\cdot y} |y\rangle \\ &= \frac{1}{\sqrt{2^{\dim C_2}}} \frac{1}{\sqrt{2^n}} \sum_{y \in Z_2^n} (-1)^{x\cdot y} \sum_{c_2 \in C_2} (-1)^{c_2 \cdot y} |y\rangle \end{aligned}$$

Now I claim that

$$\sum_{c_2 \in C_2} (-1)^{c_2 \cdot y} = \begin{cases} |C_2| & y \in C_2^\perp \\ 0 & y \notin C_2^\perp \end{cases}.$$

Why? if $y \in C_2^\perp$, all the elements of the sum are 1, so we get $|C_2$. If not, then there is some $d \in C_2$ such that $(-1)^{d\cdot yi} = -1$, and we can pair the elements of C_2 into pairs $(c_2, c_2 + d)$. Since $(-1(^{c_2} + (-1)^{c_2+d} = 0$, the sum for each of these pairs is 0, so the whole sum is 0. Thus, we have

$$\begin{aligned} H^{\otimes n} |x + C_2\rangle &= \frac{1}{\sqrt{2^{\dim C_2}}} \frac{1}{\sqrt{2^n}} \sum_{y \in C_2^\perp} (-1)^{x\cdot y} |C_2|\, |y\rangle \\ &= \frac{\sqrt{2^{\dim C_2}}}{\sqrt{2^n}} \sum_{y \in C_2^\perp} (-1)^{x\cdot y} |y\rangle \\ &= \frac{1}{\sqrt{2^{\dim C_2^\perp}}} \sum_{y \in C_2^\perp} (-1)^{x\cdot y} |y\rangle \end{aligned}$$

Now, we are nearly done. We claim that if y_1 and y_2 are in the same coset of $C_1^\perp$ in $C_2^\perp$, then $(-1)^{x\cdot y_1} = (-1)^{x\cdot y_2}$. Why is this? Because $y_1 - y_2 \in C_1^\perp$ and $x \in C_1$, so $x \cdot (y_1 - y_2) = 0$. So now let R be a set of representatives of the cosets of $C_1^\perp$ in $C_2^\perp$; this means there is one element of each coset in R. We can group the sum over y above into elements from each coset of $C_1^\perp$ in $C_2^\perp$. This gives

$$\begin{aligned} H^{\otimes n} |x + C_2\rangle &= \frac{1}{\sqrt{2^{\dim C_2^\perp}}} \sum_{y \in R} (-1)^{x\cdot y} \sum_{z \in y + C_1^\perp} |z\rangle \\ &= \frac{\sqrt{2^{\dim C_1^\perp}}}{\sqrt{2^{\dim C_2^\perp}}} \sum_{y \in R} (-1)^{x\cdot y} \left|y + C_1^\perp\right\rangle, \end{aligned}$$

showing that the Hadamard transform of a codeword in CSS($C_1 : C_2$) is a superposition of codewords in CSS($C_2^\perp : C_1^\perp$), as we wanted.

The last thing I want to talk about in these notes is shifting a code. We will need this definition for the proof of the security of BB84, which we present in the next lecture. We can take a CSS code and shift it in bit space or in phase space. A codeword for a code shifted by s in bit space is

$$|x + C_2\rangle = \frac{1}{|C_2|^{1/2}} \sum_{c \in C_2} |s + x + c\rangle$$

and one shifted by t in phase space is

$$|x + C_2\rangle = \frac{1}{|C_2|^{1/2}} \sum_{c \in C_2} (-1)^{t \cdot (x+c)} |x + c\rangle$$

And of course, you can shift in phase space and bit space (the order makes a difference) to get

$$|x + C_2\rangle = \frac{1}{|C_2|^{1/2}} \sum_{c \in C_2} (-1)^{t \cdot (x+c)} |s + x + c\rangle$$

All of these shifted error correcting codes can correct the same number of errors as the unshifted ones. What we will need to know for the proof of security of quantum key distribution is that if you shift in both bit space and phase space by a random vector, and average over the resulting codes, you get the completely uniform distribution — the density matrix is I_{2^n}. I'm not going to prove that in these lecture notes, but if you want to prove it, it would make a good exercise to make sure you understand CSS codes well.