Notes 8.370/18.435 Fall 2022

Lecture 15 Prof. Peter Shor

Today, we covered the Deutsch-Jozsa algorithm.

Before talking about the Deutsch-Jozsa algorithm, I'm going to explain the Hadamard transform, which is a necessary ingredient both for the Deutsch-Jozsa algorithm and Simon's algorithm, which we will do next. Recall the Hadamard gate $H = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$. The Hadamard transform is simply $H^{\otimes n}$. That is, you do a Hadamard gate on each of n qubits.

What does $H^{\otimes n}$ do to a quantum state?

First, let's do an easy example

$$\begin{aligned} H^{\otimes n}|0^n\rangle &= \frac{1}{2^{n/2}}\big(|0\rangle+|1\rangle\big)\big(|0\rangle+|1\rangle\big)\ldots\big(|0\rangle+|1\rangle\big) \\ &= \frac{1}{2^{n/2}}\sum_{j=0}^{2^n-1}|j\rangle. \end{aligned}$$

Here, $|j\rangle$ means the bitstring in $\{0,1\}^n$ that represents the integer j in binary. For example, if $n=4$, then $|12\rangle = |1100\rangle$ and $|7\rangle = |0111\rangle$. When we apply the distributive law to $\big(|0\rangle+|1\rangle\big)\big(|0\rangle+|1\rangle\big)\ldots\big(|0\rangle+|1\rangle\big)$, we get every n-bit binary string exactly once, and so we get every integer $|j\rangle$ from $|0\rangle$ to $|2^n-1\rangle$.

Now, what happens when we apply the Hadamard transform to an arbitrary bit string? For example,

$$\begin{aligned} H^{\otimes 5}|01011\rangle &= \frac{1}{\sqrt{32}}\big(|0\rangle+|1\rangle\big)\big(|0\rangle-|1\rangle\big)\big(|0\rangle+|1\rangle\big)\big(|0\rangle-|1\rangle\big)\big(|0\rangle-|1\rangle\big) \\ &= \frac{1}{\sqrt{32}}\big(|00000\rangle-|00001\rangle-|00010\rangle+|00011\rangle+|00100\rangle-\ldots-|11111\rangle\big) \end{aligned}$$

Here, for example, we see that $|11111\rangle$ has a minus sign in the expansion because there are three $-$'s in the five $|1\rangle$ terms we multiply to get $|11111\rangle$.

We see from the above that each binary string representing integers k between 0 and 31 appears in the sum with amplitude $2^{-n/1}$. The question is; what is the sign on $|k\rangle$. Let $j\cdot k$ be the dot product of j and k, i.e., the number of places where 1s appear in both j and k. I claim that

$$H^{\otimes n}|j\rangle = \frac{1}{2^{n/2}}\sum_{k=0}^{2^n-1}(-1)^{j\cdot k}|k\rangle.$$

Why do we get the phase $(-1)^{j\cdot k}$ in the sum above. Let's look at our example above. What would the phase on $|11001\rangle$ be in $H^{\otimes 5}|01011\rangle$? To get $|11001\rangle$, we take the $|1\rangle$ from the first, second, and fifth terms, and the $|0\rangle$ from the second and third terms. This gives

$$\frac{1}{\sqrt{32}}\big(|1\rangle\big)\big(-|1\rangle\big)\big(|0\rangle\big)\big(|0\rangle\big)\big(-|1\rangle\big) = \frac{1}{\sqrt{32}}|11001\rangle$$

We get a minus sign only when we choose a $|1\rangle$ (which means there is a 1 bit in that spot in $|k\rangle$ and when there is a $-$ sign on the $|1\rangle$ (which means there is a $|1\rangle$ in that position in $|j\rangle$. So the number of $-$ signs is just the number of places which have a 1 in both k and j. This is $j \cdot k$. Further, multiplying all the $-$ signs together gives the term $(-1)^{j \cdot k}$ given in our formula.

Next, we describe the Deutsch-Jozsa algorithm.

The Deutsch-Jozsa algorithm operates on a function mapping binary strings of length n into bits. First, let's give a couple of definition.

Definition 1 *A function is* constant *when* $f(x) = f(y)$ *for all* x, y *in* $\mathrm{Domain}(f)$.

Definition 2 *A function is* balanced *when there are an equal number of inputs that produce a* 0 *and that produce a* 1.

For example, the function on two bits:

$$f(00) = 0; \quad f(01) = 1 \quad f(10) = 0 \quad f(11) = 1$$

is balanced.

The Deutsch-Jozsa algorithm takes a function which is either constant or balanced, and tells which it is.

How many queries to the function does it take a classical computer to solve this problem? We will require it to solve it deterministically. This is equivalent to the problem: you have 2^n balls in an urn, and the balls are either black or white. You know that either all the balls are the same color, or half of them are each color. How many balls do you need to take to be know for certain which case holds?

The answer is that you need to take $2^{n-1} + 1$, or one more than half the balls. Suppose you start drawing balls and you draw 2^{n-1} of them that are black. You can't know for certain that all the other balls in the urn are white, and that you were very unlucky in which balls you chose. Thus, classically, it may take evaluating the function on $2^{n-1} + 1$ different inputs to be sure you have the right answer.

Now, we will assume that the function f is given to us in the form of some circuit or black box (we will call it an *oracle*) that takes

$$O_f |x\rangle = \begin{cases} |x\rangle & \text{if } f(x) = 0 \\ -|x\rangle & \text{if } f(x) = 1 \end{cases}$$

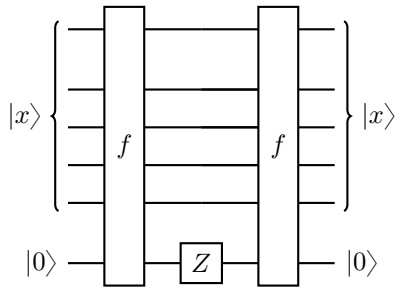
This is a *phase oracle* because it encodes the information about the function in the phase. A *bit oracle* computes the result of the function in a separate register.

$$O_f |x\rangle |0\rangle = |x\rangle |f(x)\rangle \ .$$

Here the first register has n qubits and the second has one qubit.

We will show that if you have a bit oracle for a function f, you can construct a phase oracle for this function. It actually turns out that these two kinds of oracles are equivalent, but we will not prove that now.

How can we construct a phase oracle if we are given a bit oracle? What we do is first compute $|f(x)\rangle$ in a second register. We then apply a Z gate to this register to get $(-1)^{f(x)}$, and then uncompute $|f(x)\rangle$. The quantum circuit is as below:



After the first gate, we have $|x\rangle\,|f(x)\rangle$. After the Z gate, we get $(-1)^{f(x)}\,|x\rangle\,|f(x)\rangle$. The last gate uncomputes f, so we get $(-1)^{f(x)}\,|x\rangle\,|0\rangle$, which is the phase oracle (aside from the work bit that ends up in the same state as it started in, and thus can be disregarded).

The Deutsch algorithm is fairly straightforward:

Step 1: Start with $|0^n\rangle$,

Step 2: Apply the Hadamard transform $H^{\otimes n}$,

Step 3: Apply the phase oracle O_f,

Step 4: Apply the Hadamard transform $H^{\otimes n}$,

Step 5: Measure the quantum state.

Now let's see what happens in this algorithm. We've already computed much of what we need for these manipulations in our discussion of the Hadamard transform.

$$H^{\otimes n}\,|0\rangle = \frac{1}{2^{n/2}} \sum_{j=0}^{2^n-1} |j\rangle$$

$$O_f H^{\otimes n}\,|0\rangle = \frac{1}{2^{n/2}} \sum_{j=0}^{2^n-1} (-1)^{f(j)}\,|j\rangle$$

$$H^{\otimes n} O_f H^{\otimes n}\,|0\rangle = \frac{1}{2^n} \sum_{j=0}^{2^n-1} \sum_{k=0}^{2^n-1} (-1)^{f(j)} (-1)^{j\cdot k}\,|k\rangle$$

Now, let's compute the probability of seeing the state $|0^n\rangle$ (i.e., $k = 0$) after the measurement. This probability is just the square of the amplitude on the $|0^n\rangle$ state.

This amplitude is just

$$\frac{1}{2^n} \sum_{j=0}^{2^n-1} (-1)^{f(j)}$$

because $j \cdot 0 = 0$.

If f is constant, then $f(j) = 1$ for all j or $f(j) = -1$ for all j. The sum is then either $+1$ or -1, so the probability of seeing $|0^n\rangle$ is 1.

If the function is balanced, then an equal number of $+1$ and -1 terms appear in the sum $\sum_{j=0}^{2^n-1}(-1)^{f(j)}$, so this sum is 0, and the probability of seeing $|0^n\rangle$ is 0.

So if the output is $|0^n\rangle$, we conclude the function is constant, and if the output is anything else we conclude the function is balanced. This computation took one call to the oracle and a linear number of gates in n, while the best deterministic classical algorithm takes around $2^{n/2}$ gates. Thus, for this problem, the best deterministic quantum algorithm is exponentially faster.

The Deutsch-Jozsa algorithm was not a very convincing argument that quantum computation was more powerful than classical computation. While you need around $2^{n/2}$ queries to tell with absolute certainty whether a function is balanced or constant, if you make 20 or 30 queries, the chances that you will be wrong are very, very small. And randomized algorithms are a well-studied area of computer science, and are generally considered almost as good as deterministic algorithms. The reason that the Deutsch-Jozsa algorithm is important is that it led to Simon's algorithm, which makes a much more convincing argument that quantum computation is more powerful.