Quantum Programming
Algorithms: Deutsch-Jozsa

Jens Palsberg

Apr 15, 2019

Outline

Hook: Our first quantum algorithm is the Deutsch-Jozsa algorithm. It solves an artificial problem, but does so faster on a quantum computer than we can ever do on a classical computer.

Purpose: Persuade you that you can understand this algorithm.

Preview:

1. We will make sure that everybody understands the problem.
2. We will begin with a simple version called Deutsch's algorithm.
3. We will cover full details of how and why the Deutsch-Jozsa algorithm works.

Transition to Body: Now let us talk about the problem that this algorithm solves.

Main Point 1: We will make sure that everybody understands the problem.
[Most of you have done the homework on solving the problem on a classical computer]
[We can represent the black-box function in multiple ways]
[We can program the solution in any classical language]

Transition to MP2: Now let us look at a solution on a quantum computer.

Main Point 2: We will begin with a simple version called Deutsch's algorithm.
[We make the black-box computation reversible]
[We construct a circuit that puts each of two qubits in superposition]
[The reasoning about correctness uncovers phase-kickback trick]

Transition to MP3: Now we are ready for the full solution.

Main Point 3: We will cover full details of how and why the Deutsch-Jozsa algorithm works.
[We can make a black box with n inputs reversible]
[The structure is the same as in Deutsch's algorithm]
[The reasoning about the phase-kickback trick is more complicated with n bits]

Transition to Close: So there you have it.

Review: The Deutsch-Jozsa algorithm makes a single call to the black-box function, which works because it tries all combinations of $|0\rangle$ and $|1\rangle$ at the same time.

Strong finish: The Deutsch-Jozsa algorithm is the first indication that a quantum computer can be faster than a classical computer. We will see more examples of that in this course.

Call to action: Learn how to reason about correctness by getting good at calculating with Dirac notation.

Detailed presentation

Hook: Our first quantum algorithm is the Deutsch-Jozsa algorithm. It solves an artificial problem, but does so faster on a quantum computer than we can ever do on a classical computer.

Purpose: Persuade you that you can understand this algorithm.

Preview:

1. We will make sure that everybody understands the problem.
2. We will begin with a simple version called Deutsch's algorithm.
3. We will cover full details of how and why the Deutsch-Jozsa algorithm works.

Transition to Body: Now let us talk about the problem that this algorithm solves.

Main Point 1: We will make sure that everybody understands the problem.

[Most of you have done the homework on solving the problem on a classical computer]
Here is the homework problem.

The Deutsch-Jozsa problem:
Input: a function $f : \{0,1\}^n \rightarrow \{0,1\}$.
Assumption: f is either constant or balanced.
Output: 1 if f is constant; 0 if f is balanced.
Notation: $\{0,1\}$ is the set of bits, and $\{0,1\}^n$ is the set of bit strings of length n.
Terminology:
Constant: f is constant if either f always outputs 0 or f always outputs 1.
Balanced: f is balanced if f outputs 0 on exactly half of the inputs.

The assignment:
On a classical computer, in a classical language of your choice (such as C, Java, Python, etc), program solutions to the Deutsch-Jozsa problem. Treat the input function f as black box that you can call but cannot inspect in any way at all. Each solution will be code that includes one or more calls of f.

[We can represent the black-box function in multiple ways]
For the case of $n = 1$, we have that f can be one of four functions, here called f_0, f_1, f_2, f_3:

Input	f_0	f_1	f_2	f_3
0	0	0	1	1
1	0	1	0	1

Notice that f_0 and f_3 are constant, while f_1 and f_2 are balanced.

How do we keep the black-box function at an arms length such that we cannot get tempted to inspect it? In C we can use a function pointer. In Java we can use a lambda expression. In Python we can use an anonymous function. Other languages have similar constructs that will enable us to call the function, while giving us no way to inspect it.

[We can program the solution in any classical language]
You wrote the solutions to the homework in several languages.

If we want an exact solution, we need $2^{n-1}+1$ queries in the worst case. If we can tolerate that we get the right answer with high probability, we can randomly choose k inputs; evaluate f on each of the k inputs, and if the outputs are all the same, then output *constant*, and otherwise output *balanced*. If the function really is constant, this method is correct every time, and if the function is balanced, this method will be wrong (and answer *constant*) with probability $2^{-(k-1)}$.

Transition to MP2: Now let us look at a solution on a quantum computer.

Main Point 2: We will begin with a simple version called Deutsch's algorithm.

[We make the black-box computation reversible]

Here we bring in the technique from the previous lecture. Specifically, if we have an operation

$$f \quad : \quad \{0,1\} \rightarrow \{0,1\}$$

then we can represent it as an invertible operation U_f:

$$\begin{aligned} U_f \quad &: \quad \mathit{Qubit}^{\otimes 2} \rightarrow \mathit{Qubit}^{\otimes 2} \\ U_f|x\rangle|b\rangle \quad &= \quad |x\rangle|b \oplus f(x)\rangle \end{aligned}$$

We use $\mathit{Qubit}^{\otimes n}$ to denote the tensor product of n vectors spaces of single qubits. We use $\oplus$ to denote addition (mod 2).

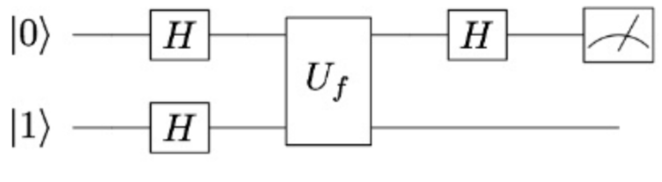
Let us check that for every one of the four possibilities of f, we have that U_f is a unitary function.

$$U_{f_0} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} = I \qquad U_{f_1} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} = CNOT$$

$$U_{f_2} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \qquad U_{f_3} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

Notice that each of the four matrices above is a permutation matrix: every entry is 0 or 1, every column has a single 1, and every row has a single 1. Every permutation matrix is unitary.

[We construct a circuit that puts each of two qubits in superposition]



If we measure 0, we conclude that the function is *constant*, and otherwise it is *balanced*.

[The reasoning about correctness uncovers the phase-kickback trick]

Lemma 1. $\forall a \in \{0,1\} : |0 \oplus a\rangle - |1 \oplus a\rangle \;=\; (-1)^a \; (|0\rangle - |1\rangle)$.

Proof. Let us try both possibilities for a in turn.
For $a = 0$, we have:

$$\begin{array}{rrcccc} \text{left-hand side:} & |0 \oplus a\rangle - |1 \oplus a\rangle & = & |0 \oplus 0\rangle - |1 \oplus 0\rangle & = & |0\rangle - |1\rangle \\ \text{right-hand side:} & (-1)^a \; (|0\rangle - |1\rangle) & = & (-1)^0 \; (|0\rangle - |1\rangle) & = & |0\rangle - |1\rangle \end{array}$$

For $a = 1$, we have:

$$\begin{array}{rrcccc} \text{left-hand side:} & |0 \oplus a\rangle - |1 \oplus a\rangle & = & |0 \oplus 1\rangle - |1 \oplus 1\rangle & = & |1\rangle - |0\rangle \\ \text{right-hand side:} & (-1)^a \; (|0\rangle - |1\rangle) & = & (-1)^1 \; (|0\rangle - |1\rangle) & = & |1\rangle - |0\rangle \end{array}$$

□

Lemma 2 (Phase kickback)**.** $\forall x \in \{0,1\}^n : \; U_f|x\rangle|-\rangle \;=\; (-1)^{f(x)}|x\rangle|-\rangle$.

Proof.

$$\begin{array}{cl} & U_f|x\rangle|-\rangle \\ = & \frac{1}{\sqrt{2}} \; (U_f|x\rangle|0\rangle - U_f|x\rangle|1\rangle) \\ = & \frac{1}{\sqrt{2}} \; (|x\rangle|f(x)\rangle - |x\rangle|1 \oplus f(x)\rangle) \\ = & |x\rangle \otimes \frac{1}{\sqrt{2}} \; (|0 \oplus f(x)\rangle - |1 \oplus f(x)\rangle) \\ = & |x\rangle \otimes \frac{1}{\sqrt{2}} \; (-1)^{f(x)} \; (|0\rangle - |1\rangle) \\ = & (-1)^{f(x)} \; |x\rangle \otimes \frac{1}{\sqrt{2}} \; (|0\rangle - |1\rangle) \\ = & (-1)^{f(x)} \; |x\rangle|-\rangle \end{array}$$

In the fourth step, we used Lemma 1. □

Lemma 3. $\forall a \in \{0,1\} : H\left(\frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}(-1)^a|1\rangle\right) \;=\; |a\rangle$.

Proof. Let us try both possibilities for a in turn.
For $a = 0$, we have:

$$\begin{array}{rcl} H\left(\frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}(-1)^a|1\rangle\right) & = & H\left(\frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}(-1)^0|1\rangle\right) \;=\; H\left(\frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle\right) \;=\; H|+\rangle \\ & = & |0\rangle \;=\; |a\rangle \end{array}$$

For $a = 1$, we have:

$$\begin{array}{rcl} H\left(\frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}(-1)^a|1\rangle\right) & = & H\left(\frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}(-1)^1|1\rangle\right) \;=\; H\left(\frac{1}{\sqrt{2}}|0\rangle - \frac{1}{\sqrt{2}}|1\rangle\right) \;=\; H|-\rangle \\ & = & |1\rangle \;=\; |a\rangle \end{array}$$

□

Theorem 4. *Deutsch's algorithm is correct.*

Proof. Initially, the state of the two qubits is $|01\rangle$.

After the two uses of H, followed by the use of U_f, followed by the single use of H, the state of the two qubits is:

$$\begin{array}{rl}
& (H \otimes I) \circ U_f \circ (H \otimes H)(|01\rangle) \\
= & (H \otimes I) \circ U_f \ (|+\rangle \otimes |-\rangle) \\
= & (H \otimes I) \circ U_f \ (\frac{1}{\sqrt{2}} \ (|0\rangle + |1\rangle) \otimes |-\rangle) \\
= & (H \otimes I) \circ U_f \ (\frac{1}{\sqrt{2}} \ (\Sigma_{x \in \{0,1\}} |x\rangle) \otimes |-\rangle) \\
= & (H \otimes I) \frac{1}{\sqrt{2}} \ \Sigma_{x \in \{0,1\}} \ (-1)^{f(x)} |x\rangle \otimes |-\rangle \\
= & (H \otimes I)(\frac{1}{\sqrt{2}} \ ((-1)^{f(0)}|0\rangle \ + \ (-1)^{f(1)}|1\rangle)) \ \otimes \ |-\rangle \\
= & (H \otimes I)((-1)^{f(0)} \ \frac{1}{\sqrt{2}} \ (|0\rangle \ + \ (-1)^{f(0) \oplus f(1)}|1\rangle)) \ \otimes \ |-\rangle \\
= & (H \otimes I)((-1)^{f(0)} \ (\frac{1}{\sqrt{2}} \ |0\rangle \ + \ \frac{1}{\sqrt{2}} \ (-1)^{f(0) \oplus f(1)}|1\rangle)) \ \otimes \ |-\rangle \\
= & ((-1)^{f(0)} \ |f(0) \oplus f(1)\rangle) \ \otimes \ |-\rangle
\end{array}$$

In the fourth step, we use Lemma 2. In the last step, we use Lemma 3. So, if we measure the first qubit, we will get $|f(0) \oplus f(1)\rangle$ with probability

$$|((-1)^{f(0)}|^2 \ = \ 1^2 \ = \ 1$$

Notice that

$$f(0) \oplus f(1) \ = \ \begin{cases} 0 & f \text{ is constant} \\ 1 & f \text{ is balanced} \end{cases}$$

□

Transition to MP3: Now we are ready for the full solution.

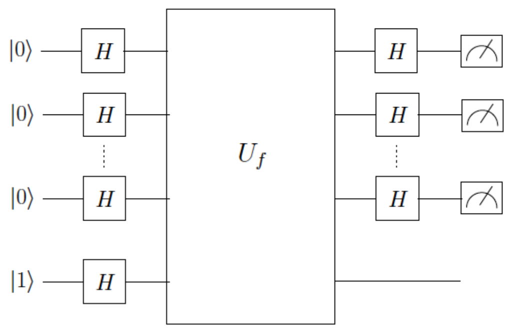
Main Point 3: We will cover full details of how and why the Deutsch-Jozsa algorithm works.

[We can make a black box with n inputs reversible]

The above construction of permutation matrices to support reversible computing generalizes to functions f from n bits to m bits.

[The structure is the same as in Deutsch's algorithm]

Like in the case of Deutsch's algorithm, we use a single call to U_f.



The idea is that we get n bits from the measurements. If all those bits are 0, we conclude that the function is *constant*, and otherwise it is *balanced*.

[The reasoning about the phase-kickback trick is more complicated with n bits]

Lemma 5. $\forall x \in \{0,1\} : \; H|x\rangle \; = \; \frac{1}{\sqrt{2}} \; \Sigma_{y\in\{0,1\}} \; (-1)^{x\cdot y} \; |y\rangle$.

Proof. The left-hand side:

$$H|x\rangle \quad = \quad \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}(-1)^x|1\rangle$$

Here, we use Lemma 3.

The right-hand side:

$$\frac{1}{\sqrt{2}} \; \Sigma_{y\in\{0,1\}} \; (-1)^{x\cdot y} \; |y\rangle \quad = \quad \frac{1}{\sqrt{2}} \; ((-1)^{x\cdot 0} \; |0\rangle + (-1)^{x\cdot 1} \; |1\rangle) \quad = \quad \frac{1}{\sqrt{2}} \; (|0\rangle + (-1)^x|1\rangle)$$

We see that the two sides are equal. □

We use $H^{\otimes n}$ to denote $H \otimes H \otimes \ldots \otimes H$ (n occurrences of H). If x and y are bit-strings of the same length, then $x \cdot y$ denotes the dot-product (mod 2) of x and y.

Lemma 6. $H^{\otimes n}|x\rangle \; = \; \frac{1}{\sqrt{2^n}} \; \Sigma_{y\in\{0,1\}^n} \; (-1)^{x\cdot y} \; |y\rangle$

Proof.

$$\begin{array}{ll} & H^{\otimes n}|x\rangle \\ = & H|x_1\rangle \otimes \ldots \otimes H|x_n\rangle \\ = & \left(\frac{1}{\sqrt{2}} \; \Sigma_{y_1\in\{0,1\}} \; (-1)^{x_1\cdot y_1} \; |y_1\rangle\right) \otimes \ldots \otimes \left(\frac{1}{\sqrt{2}} \; \Sigma_{y_n\in\{0,1\}} \; (-1)^{x_n\cdot y_n} \; |y_n\rangle\right) \\ = & \frac{1}{\sqrt{2^n}} \; \Sigma_{y_1\in\{0,1\}} \; \ldots \Sigma_{y_n\in\{0,1\}} \; (-1)^{x_1\cdot y_1} \; \ldots \; (-1)^{x_n\cdot y_n} \; |y_1\rangle \ldots \; |y_n\rangle \\ = & \frac{1}{\sqrt{2^n}} \; \Sigma_{y\in\{0,1\}^n} \; (-1)^{x\cdot y} \; |y\rangle \end{array}$$

In the second step, we use Lemma 5 a total of n times. □

Theorem 7. *The Deutsch-Jozsa algorithm is correct.*

Proof. Initially, the state of the $n+1$ qubits is $|00\ldots01\rangle$.

After the $n+1$ uses of H followed by the use of U_f followed by the n uses of H, the state of the $n+1$ qubits is:

$$\begin{array}{cl} & (H^{\otimes n} \otimes I) \circ U_f \circ H^{\otimes n+1}(|00\ldots01\rangle) \\ = & (H^{\otimes n} \otimes I) \circ U_f \ \frac{1}{\sqrt{2^n}} \ \Sigma_{x \in \{0,1\}^n} \ |x\rangle \otimes |-\rangle \\ = & (H^{\otimes n} \otimes I) \ \frac{1}{\sqrt{2^n}} \ \Sigma_{x \in \{0,1\}^n} \ (-1)^{f(x)} \ |x\rangle \otimes |-\rangle \\ = & \frac{1}{\sqrt{2^n}} \ \Sigma_{x \in \{0,1\}^n} \ (-1)^{f(x)} \ (\frac{1}{\sqrt{2^n}} \ \Sigma_{y \in \{0,1\}^n} \ (-1)^{x \cdot y} \ |y\rangle) \otimes |-\rangle \\ = & \frac{1}{2^n} \ \Sigma_{x \in \{0,1\}^n} \ \Sigma_{y \in \{0,1\}^n} \ (-1)^{x \cdot y + f(x)} \ |y\rangle \otimes |-\rangle \end{array}$$

In the first step, we used Lemma 6; in the second step, we used Lemma 2; and in the third step, we used Lemma 6.

Now we measure the first n qubits. Turns out, the state $|00\ldots0-\rangle$ is important:

$$\text{the amplitude of } |00\ldots0-\rangle \quad = \quad \frac{1}{2^n} \ \Sigma_{x \in \{0,1\}^n} \ (-1)^{f(x)}$$

Let us consider the two cases of f in turn.

First, suppose f is constant.

$$\text{the amplitude of } |00\ldots0-\rangle \quad = \quad \begin{cases} 1 & \text{if } f(x) = 0 \text{ for all } x \\ -1 & \text{if } f(x) = 1 \text{ for all } x \end{cases}$$

We conclude that if f is constant, a measurement of the first n qubits will give us $00\ldots0$ with probability 1.

Second, suppose f is balanced.

$$\text{the amplitude of } |00\ldots0-\rangle \quad = \quad 0$$

The reason is that $(-1)^{f(x)}$ will be 1 half of the time and -1 half of time; they will cancel either other out. We conclude that if f is balanced, a measurement of the first n qubits will give us $00\ldots0$ with probability 0

The grand conclusion is that if we measure $00\ldots0$, we output "constant", and if we measure anything else, we output "balanced". □

Transition to Close: So there you have it.

Review: The Deutsch-Jozsa algorithm makes a single call to the black-box function, which works because it tries all combinations of $|0\rangle$ and $|1\rangle$ at the same time.

Strong finish: The Deutsch-Jozsa algorithm is the first indication that a quantum computer can be faster than a classical computer. We will see more examples of that in this course.

Call to action: Learn how to reason about correctness by getting good at calculating with Dirac notation.