The Deutsch-Josza Algorithm

The Problem

Our task is to write a function with:

- Input
  - A function $f : \{0,1\}^n \to \{0,1\}$, where f is either constant or balanced
    - f is **constant** if it always outputs 0 or always outputs 1.
    - f is **balanced** if it outputs 0 on exactly half of the inputs.
- Output
  - 0 if f is constant; 1 if f is balanced

Classical Solution

We know that if f ever outputs two different values (i.e. 0 and 1) for different inputs, then f must be constant by the nature of the problem. So, to solve this problem on a classical computer, we would need to call f on $2^{n-1}+1$ inputs, or half of the inputs plus one.

- If f always outputs 0 or 1, then we know if must be constant.
- Otherwise, we would encounter two different outputs and would know to conclude that f is balanced.

Therefore, on a classical computer, we need to make $2^{n-1}+1 = O(2^n)$ calls to f in the worst case.

Quantum Solution ($n = 1$)

To determine a quantum solution to this problem, we begin by considering the simplest case: when $n = 1$. That is, we now have $f : \{0,1\} \to \{0,1\}$. Consider the classical solution:

```
def deutsch_josza(f):
  return f(0) != f(1)
```

It is impossible to write a classical solution to this problem without making **at least 2 calls** to f. However, we will show that a quantum algorithm only needs to make **1 call** to f.

Making f Reversible

Recall that all quantum operations must be **reversible**. In general, however, given the output of a function $f(x)$ it is not always possible to invert f to obtain the input x. In order to achieve this, we define a gate U_f that uses an additional "helper bit" b:

$$U_f|x\rangle|b\rangle = |x\rangle|b \oplus f(x)\rangle$$

where $\oplus$ indicates addition modulo 2, or XOR.

Observe that U_f is reversible; if we apply U_f to its output, we get:

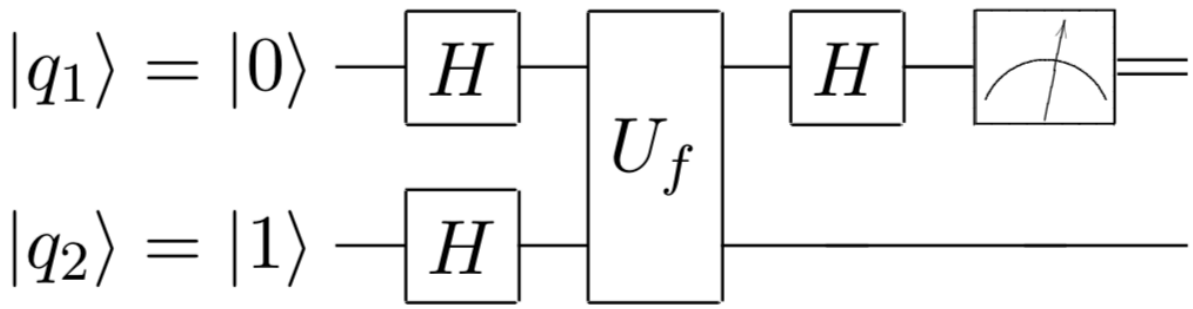
$$\begin{aligned} U_f|x\rangle|b \oplus f(x)\rangle &= |x\rangle|b \oplus f(x) \oplus f(x)\rangle \\ &= |x\rangle|b\rangle. \end{aligned}$$

Now that we have a quantum implementation of f, we have all the pieces we need to create a quantum solution for our problem (in the $n = 1$ case).

Note: The actual hardware implementation of U_f is beyond the scope of these notes. It certainly exists, but for now we will simply treat U_f as a black box that we can use. Using U_f in our quantum circuit is treated as one call of f.

Deutsch's Algorithm

The circuit for Deutsch's Algorithm is as follows:



It is not at all obvious why this works, however let's simply work through it and prove that it does work. Working from left to right, and considering the underlying matrix of each operation, this circuit boils down to the following equation:

$$(H \otimes I)U_f(H^{\otimes 2}|01\rangle)$$

- We can read this equation from right to left:
  - We begin with the initial state $|01\rangle$.
  - We apply the Hadamard gate to both qubits ($H^{\otimes 2}$).
  - We apply U_f to the result.
  - We apply the Hadamard gate to the first qubit ($H \otimes I$), where I is the identity matrix and $\otimes$ is the Kronecker product or tensor product.

We begin by expanding this equation by brute force.

A Brute Force Approach

We have:

$$\begin{aligned}(H \otimes I)U_f(H^{\otimes 2}|01\rangle) &= (H \otimes I)U_f(|+-\rangle) \\ &= (H \otimes I)U_f\left(\frac{1}{2}(|0\rangle + |1\rangle)(|0\rangle - |1\rangle)\right) \\ &= (H \otimes I)U_f\left(\frac{1}{2}(|00\rangle - |01\rangle + |10\rangle - |11\rangle)\right)\end{aligned}$$

Now, we apply the definition of U_f, namely $U_f|x\rangle|b\rangle = |x\rangle|b \oplus f(x)\rangle$:

$$\begin{aligned}&= (H \otimes I)\left(\frac{1}{2}\Big(|0\rangle|0 \oplus f(0)\rangle - |0\rangle|1 \oplus f(0)\rangle + |1\rangle|0 \oplus f(1)\rangle - |1\rangle|1 \oplus f(1)\rangle\Big)\right) \\ &= (H \otimes I)\left(\frac{1}{2}\Big(|0\rangle|f(0)\rangle - |0\rangle|1 \oplus f(0)\rangle + |1\rangle|f(1)\rangle - |1\rangle|1 \oplus f(1)\rangle\Big)\right)\end{aligned}$$

Before applying the final Hadamard gate to the first qubit, we use case analysis to simplify the equation further.

Case 1 (f is constant):

If f is constant, then we must have $f(0) = f(1)$. Therefore, we will replace every $f(1)$ in the above equation with $f(0)$:

$$\begin{aligned}&= (H \otimes I)\left(\frac{1}{2}\Big(|0\rangle|f(0)\rangle - |0\rangle|1 \oplus f(0)\rangle + |1\rangle|f(0)\rangle - |1\rangle|1 \oplus f(0)\rangle\Big)\right) \\ &= (H \otimes I)\left(\frac{1}{2}\Big((|0\rangle + |1\rangle)|f(0)\rangle - (|0\rangle + |1\rangle)|1 \oplus f(0)\rangle\Big)\right) \\ &= (H \otimes I)\left(\frac{1}{2}\Big((|0\rangle + |1\rangle)(|f(0)\rangle - |1 \oplus f(0)\rangle)\Big)\right) \\ &= (H \otimes I)\left(\frac{1}{\sqrt{2}}\Big(|+\rangle(|f(0)\rangle - |1 \oplus f(0)\rangle)\Big)\right)\end{aligned}$$

The first qubit is clearly in state $|+\rangle$, which makes the application of the Hadamard gate easy:

$$= \frac{1}{\sqrt{2}}|0\rangle\Big(|f(0)\rangle - |1 \oplus f(0)\rangle\Big).$$

Therefore, measuring qubit 1 (in the standard basis) **yields outcome $|0\rangle$ with probability 1**. //

Case 2 (f is balanced):

If f is balanced, then we must have $f(0) \neq f(1)$ (since $n = 1$). In addition, since f is a binary function we must also have $1 \oplus f(0) = f(1)$ and equivalently $1 \oplus f(1) = f(0)$. We can use this to simplify again:

$$
\begin{aligned}
&= (H \otimes I) \left(\frac{1}{2} \Big(|0\rangle|f(0)\rangle - |0\rangle|f(1)\rangle + |1\rangle|f(1)\rangle - |1\rangle|f(0)\rangle \Big) \right) \\
&= (H \otimes I) \left(\frac{1}{2} \Big((|0\rangle - |1\rangle)|f(0)\rangle - (|0\rangle - |1\rangle)|f(1)\rangle \Big) \right) \\
&= (H \otimes I) \left(\frac{1}{2} \Big((|0\rangle - |1\rangle)(|f(0)\rangle - |f(1)\rangle) \Big) \right) \\
&= (H \otimes I) \left(\frac{1}{\sqrt{2}} |-\rangle (|f(0)\rangle - |f(1)\rangle) \right)
\end{aligned}
$$

The first qubit is clearly in state $|-\rangle$, again making the application of the Hadamard gate easy:

$$
= \frac{1}{\sqrt{2}} |1\rangle (|f(0)\rangle - |f(1)\rangle)
$$

Therefore, measuring qubit 1 (in the standard basis) **yields outcome $|1\rangle$ with probability 1**. $//$

By enumerating both cases, we have proven that if f is constant our circuit always outputs $|0\rangle$ and if f is balanced our circuit always outputs $|1\rangle$! **The crux is that Deutsch's algorithm decides whether f is constant or balanced using just a single call to f ($O(1)$).**

This analysis used brute force, but there exists a more intuitive view which simplifies the calculation greatly.

The Phase Kickback Trick

Consider the quantity $U_f|x\rangle|-\rangle$ for $x \in \{0,1\}^n$:

$$
\begin{aligned}
U_f|x\rangle|-\rangle &= U_f \left(\frac{1}{\sqrt{2}} (|x\rangle|0\rangle - |x\rangle|1\rangle) \right) \\
&= \frac{1}{\sqrt{2}} U_f \left(|x\rangle|0\rangle - |x\rangle|1\rangle \right) \\
&= \frac{1}{\sqrt{2}} \Big(U_f|x\rangle|0\rangle - U_f|x\rangle|1\rangle \Big) \\
&= \frac{1}{\sqrt{2}} \Big(|x\rangle|0 \oplus f(x)\rangle - |x\rangle|1 \oplus f(x)\rangle \Big) \quad (\text{by the definition of } U_f) \\
&= \frac{1}{\sqrt{2}} \Big(|x\rangle|f(x)\rangle - |x\rangle|1 \oplus f(x)\rangle \Big) \\
&= \frac{1}{\sqrt{2}} \Big(|x\rangle(|f(x)\rangle - |1 \oplus f(x)\rangle) \Big)
\end{aligned}
$$

Remembering that f is a binary function, we know that:

$$
1 \oplus f(x) = \begin{cases} 1 & \text{if } f(x) = 0, \\ 0 & \text{if } f(x) = 1. \end{cases}
$$

Further:

$$\frac{1}{\sqrt{2}}\Big(|f(x)\rangle - |1 \oplus f(x)\rangle\Big) = \begin{cases} \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) = |-\rangle & \text{if } f(x) = 0, \\ -\frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) = -|-\rangle & \text{if } f(x) = 1. \end{cases}$$

Therefore, we can finally arrive at the compact expression:

$$U_f|x\rangle|-\rangle = (-1)^{f(x)}|x\rangle|-\rangle$$

The term "phase" comes from the -1 "phase" factor produced when $f(x) = 1$. Intuitively, this equation says that there is a way to "extract" $f(x)$ from the application of U_f, as long as the input is in the correct form.

Applying the Phase Kickback Trick

Let's revisit our calculations above from the beginning:

$$\begin{aligned}(H \otimes I)U_f(H^{\otimes 2}|01\rangle) &= (H \otimes I)U_f(|+-\rangle) \\ &= (H \otimes I)U_f\left(\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)|-\rangle\right) \\ &= (H \otimes I)U_f\left(\frac{1}{\sqrt{2}}|0\rangle|-\rangle + \frac{1}{\sqrt{2}}|1\rangle|-\rangle\right) \\ &= (H \otimes I)\frac{1}{\sqrt{2}}(U_f|0\rangle|-\rangle + U_f|1\rangle|-\rangle)\end{aligned}$$

Now, we can apply the phase kickback track:

$$= (H \otimes I)\frac{1}{\sqrt{2}}\Big((-1)^{f(0)}|0\rangle|-\rangle + (-1)^{f(1)}|1\rangle|-\rangle\Big)$$

As before, prior to applying the final Hadamard gate we can use case analysis:

Case 1 (f is constant):

If f is constant, then we must have $f(0) = f(1)$. Therefore, we will replace every $f(1)$ in the above equation with $f(0)$:

$$\begin{aligned}&= (H \otimes I)\frac{1}{\sqrt{2}}\Big((-1)^{f(0)}|0\rangle|-\rangle + (-1)^{f(0)}|1\rangle|-\rangle\Big) \\ &= (H \otimes I)\frac{1}{\sqrt{2}}(-1)^{f(0)}\Big(|0\rangle|-\rangle + |1\rangle|-\rangle\Big) \\ &= (H \otimes I)(-1)^{f(0)}|+\rangle|-\rangle\end{aligned}$$

We can apply the final Hadamard to the first qubit (which is in state $|+\rangle$) regardless of the value of $(-1)^{f(0)}$, to obtain:

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

As before, measuring qubit 1 (in the standard basis) **yields 0 with probability 1**. $//$

Case 2 (f is balanced):

When f is balanced, then we must have $f(0) \neq f(1)$. That means:

$$\frac{1}{\sqrt{2}}\left((-1)^{f(0)}|0\rangle|-\rangle + (-1)^{f(1)}|1\rangle|-\rangle\right) = \begin{cases} \frac{1}{\sqrt{2}}(|0\rangle|-\rangle - |1\rangle|-\rangle) = |-\rangle|-\rangle & \text{if } f(0) = 0, \\ \frac{1}{\sqrt{2}}(-|0\rangle|-\rangle + |1\rangle|-\rangle) = -|-\rangle|-\rangle & \text{if } f(0) = 1. \end{cases}$$

Compactly, we can write:

$$\frac{1}{\sqrt{2}}(-|0\rangle|-\rangle + |1\rangle|-\rangle) = \pm|-\rangle|-\rangle.$$

Picking up our calculations where we left off:

$$= (H \otimes I) \pm |-\rangle|-\rangle$$

Again, we can apply the final Hadamard to the first qubit (which is in state $|-\rangle$) regardless of the sign in front to obtain:

$$= \pm|1\rangle|-\rangle.$$

Finally, we conclude that measuring qubit 1 in the standard basis **yields 1 with probability 1.** //

Generalizing to Any n: The Deutsch-Josza Algorithm

Deutsch's algorithm works in the simple case where $n = 1$ and f acts on a single input bit. In reality, however, we are interested in algorithms working on n input bits, and indeed there exists an n-bit generalization of Deutsch's algorithm known as the Deutsch-Josza Algorithm.

We return to our original problem statement; our task is to write a function with:

- Input
  - A function $f : \{0,1\}^n \to \{0,1\}$, where f is either constant or balanced
    - f is **constant** if it always outputs 0 or always outputs 1.
    - f is **balanced** if it outputs 0 on exactly half of the inputs.
- Output
  - 0 if f is constant; 1 if f is balanced

It turns out that Deutsch's algorithm generalizes in an easy manner to this setting; however, its analysis is a bit more tricky, and **crucially uses the phase kickback trick**.

First, as before, consider the classical solution to this problem:

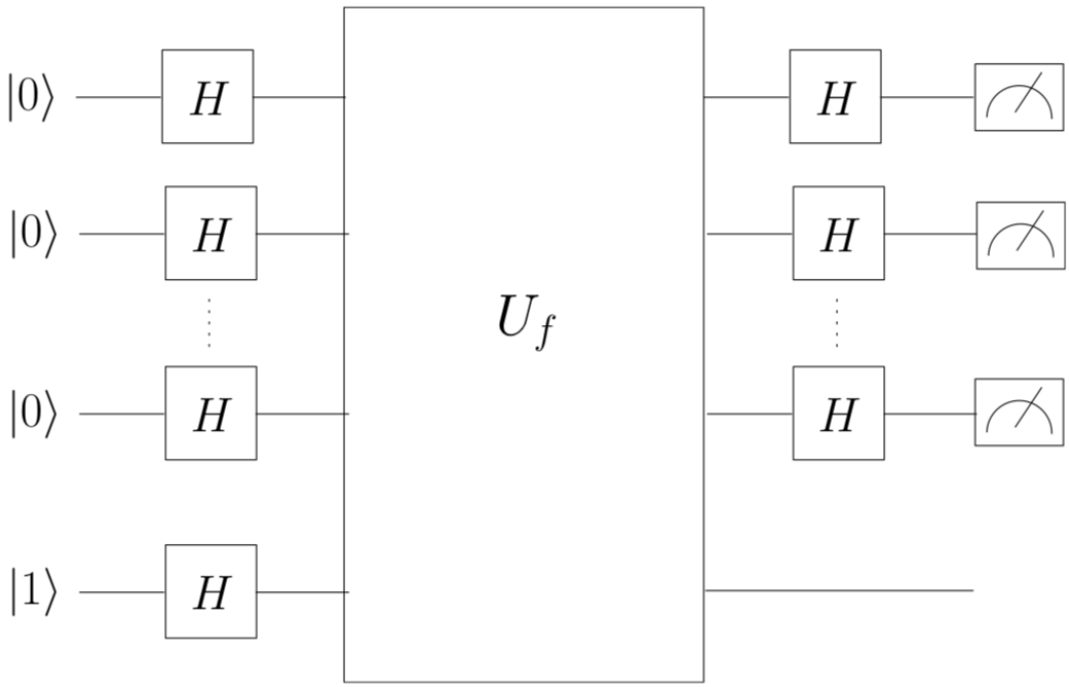
```
def deutsch_jozsa(f, n):
    # Generate the first 2^(n-1) + 1 bit strings of length n.
    bit_strings = makeBitStrings(2**(n-1) + 1)

    # Iterate through generated bit strings and check f(x) for each.
    first = f(bit_strings[0])
```

```
    for i in range(1, len(bit_strings)):
        res = f(bit_strings[i])
        if res != first:
            # We found f(x) that differs from f(0...0); f is balanced.
            return 0
    # f(x) returned the same value for the first 2^(n-1) + 1 possible bitstrings
of
    # length n; f is constant.
    return 1
```

We need to make $O(2^n)$ calls to f as aforementioned. However, we will show that the Deutsch-Josza algorithm requires only **1 call** to f, leading to an exponential speedup.

The circuit for the Deutsch-Josza algorithm is as follows:



- There are n input qubits initialized to $|0\rangle$.
- The $n+1^{\text{th}}$ qubit is initialized to $|1\rangle$.

We again begin by considering the equation represented by this circuit:

$$(H^{\otimes n} \otimes I)U_f(H^{\otimes n+1}|0\rangle^{\otimes n}|1\rangle)$$

Applying the first Hadamard, we obtain:

$$= (H^{\otimes n} \otimes I)U_f(|+\rangle^{\otimes n}|-\rangle)$$

U_f is still defined as $U_f|x\rangle|b\rangle = |x\rangle|b \oplus f(x)\rangle$; x is just now of length n. In order to apply U_f now, however, we must **convert** the expression $|+\rangle^{\otimes n}|-\rangle$ to the standard basis. First, note:

$$
\begin{aligned}
|+\rangle^{\otimes n} &= \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \otimes \cdots \otimes \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \\
&= \frac{1}{2^{n/2}}(|0\rangle + |1\rangle) \otimes \cdots \otimes (|0\rangle + |1\rangle)
\end{aligned}
$$

The tensor product $(|0\rangle + |1\rangle) \otimes \cdots \otimes (|0\rangle + |1\rangle)$, when expanded, yields the sum of every possible bit string of length n. Thus, we obtain:

$$
|+\rangle^{\otimes n} = \frac{1}{2^{n/2}} \sum_{x \in \{0,1\}^n} |x\rangle
$$

And, further (using the distributive property of the tensor product):

$$
|+\rangle^{\otimes n}|-\rangle = \frac{1}{2^{n/2}} \sum_{x \in \{0,1\}^n} |x\rangle|-\rangle.
$$

Returning to our circuit equation, we now have:

$$
= (H^{\otimes n} \otimes I) U_f \left(\frac{1}{2^{n/2}} \sum_{x \in \{0,1\}^n} |x\rangle|-\rangle \right)
$$

And we are in a perfect position to apply the phase kickback trick (which, as proved above, works for x of any length n):

$$
= (H^{\otimes n} \otimes I) \left(\frac{1}{2^{n/2}} \sum_{x \in \{0,1\}^n} (-1)^{f(x)} |x\rangle|-\rangle \right)
$$

The n-bit Hadamard Lemma

To apply the final set of Hadamard gates, we need to understand what $H^{\otimes n}|x\rangle$ equals for arbitrary $|x\rangle$.

We can begin by considering what H does to a single qubit $\psi \in \{0, 1\}$. Namely, we know that $H|0\rangle = |+\rangle$ and that $H|1\rangle = |-\rangle$. Equivalently, we can compactly write:

$$
H|\psi\rangle = \frac{1}{\sqrt{2}} \sum_{z \in \{0,1\}} (-1)^{\psi z} |z\rangle
$$

where ψz is regular multiplication. Now, we can generalize. Consider a bit string $x = x_1 \cdots x_n$:

$$
\begin{aligned}
H^{\otimes n}|x\rangle &= H|x_1\rangle \otimes \cdots \otimes H|x_n\rangle \\
&= \frac{1}{\sqrt{2}} \sum_{z_1 \in \{0,1\}} (-1)^{x_1 z_1} |z_1\rangle \otimes \cdots \otimes \frac{1}{\sqrt{2}} \sum_{z_n \in \{0,1\}} (-1)^{x_n z_n} |z_n\rangle \\
&= \frac{1}{2^{n/2}} \sum_{z_1 \in \{0,1\}} (-1)^{x_1 z_1} |z_1\rangle \otimes \cdots \otimes \sum_{z_n \in \{0,1\}} (-1)^{x_n z_n} |z_n\rangle
\end{aligned}
$$

Since each $z_i \in \{0, 1\}$, the tensor product above clearly evaluates to a sum of every possible bit string of length n. However, what is the sign in front of an arbitrary string z? Well, each $z_i \in z$ contributes a coefficient of $(-1)^{x_i z_i}$. By exponent rules, we can obtain:

$$= \frac{1}{2^{n/2}} \sum_{z \in \{0,1\}^n} (-1)^{x_1 z_1 + \cdots + x_n z_n} |z\rangle$$

$x_1 z_1 + \cdots + x_n z_n$ is merely a linear combination, so simplifying further we finally arrive at:

$$H^{\otimes n}|x\rangle = \frac{1}{2^{n/2}} \sum_{z \in \{0,1\}^n} (-1)^{x \cdot z} |z\rangle$$

where $x \cdot z$ is the bitwise inner product modulo 2 of x and z (The modulo 2 comes from the fact that the exponent base is (-1), so all we care about is if $x \cdot z$ is even or odd).

Applying the Final Hadamard

We can now complete our calculations. Returning from where we left off and using the generalized n-bit Hadamard result:

$$\begin{aligned}
&= \frac{1}{2^{n/2}} \sum_{x \in \{0,1\}^n} \left((-1)^{f(x)} \left(\frac{1}{2^{n/2}} \sum_{z \in \{0,1\}^n} (-1)^{x \cdot z} |z\rangle \right) |-\rangle \right) \\
&= \frac{1}{2^n} \sum_{x \in \{0,1\}^n} \left((-1)^{f(x)} \left(\sum_{z \in \{0,1\}^n} (-1)^{x \cdot z} |z\rangle \right) |-\rangle \right) \\
&= \frac{1}{2^n} \sum_{x \in \{0,1\}^n} \sum_{z \in \{0,1\}^n} (-1)^{f(x)} (-1)^{x \cdot z} |z\rangle |-\rangle \\
&= \frac{1}{2^n} \sum_{x \in \{0,1\}^n} \sum_{z \in \{0,1\}^n} (-1)^{f(x) + x \cdot z} |z\rangle |-\rangle \\
&= \frac{1}{2^n} \sum_{z \in \{0,1\}^n} \sum_{x \in \{0,1\}^n} (-1)^{f(x) + x \cdot z} |z\rangle |-\rangle.
\end{aligned}$$

The final step is to take a measurement of the first n qubits, and the trick of proving the algorithm's correctness will be to determine the amplitude on the all zeros state, $|0\rangle^{\otimes n}$. As with Deutsch's algorithm, we will now use case analysis to simplify the equation further.

Case 1 (f is constant):

If f is constant, we can factor out the $(-1)^{f(x)}$ term in the summand above to obtain:

$$= \frac{1}{2^n} (-1)^{f(x)} \sum_{z \in \{0,1\}^n} \sum_{x \in \{0,1\}^n} (-1)^{x \cdot z} |z\rangle |-\rangle.$$

Rearranging (and adding some brackets):

$$= (-1)^{f(x)} \sum_{z \in \{0,1\}^n} \left(\frac{1}{2^n} \sum_{x \in \{0,1\}^n} (-1)^{x \cdot z} \right) |z\rangle |-\rangle.$$

The term $\frac{1}{2^n} \sum_{x \in \{0,1\}^n} (-1)^{x \cdot z}$ is clearly the amplitude (multiplied by a phase of $(-1)^{f(x)}$) of each possible n-qubit state z. As previously mentioned, we want to consider the amplitude on the all zeros state, $\alpha_{|0\rangle^{\otimes n}}$. Setting $|z\rangle = |0\rangle^{\otimes n}$:

$$\begin{aligned}
\alpha_{|0\rangle^{\otimes n}} &= \frac{1}{2^n} \sum_{x \in \{0,1\}^n} (-1)^{x \cdot \vec{0}} \\
&= \frac{1}{2^n} \sum_{x \in \{0,1\}^n} (-1)^0 \\
&= \frac{1}{2^n} \sum_{x \in \{0,1\}^n} 1 \\
&= \frac{1}{2^n} \times 2^n \\
&= 1.
\end{aligned}$$

This astonishing result means that if f is constant and we measure the first n qubits, **we will obtain the state $|0\rangle^{\otimes n}$ with probability 1.** //

Case 2 (f is balanced):

When f is balanced, we can no longer simply factor out the $(-1)^{f(x)}$ term. However, we will again consider the amplitude of the all zeros state and reason about what we get:

$$\begin{aligned}
\alpha_{|0\rangle^{\otimes n}} &= \frac{1}{2^n} \sum_{x \in \{0,1\}^n} (-1)^{f(x) + x \cdot \vec{0}} \\
&= \frac{1}{2^n} \sum_{x \in \{0,1\}^n} (-1)^{f(x) + 0} \\
&= \frac{1}{2^n} \sum_{x \in \{0,1\}^n} (-1)^{f(x)}
\end{aligned}$$

Since f is balanced, we must have that $f(x) = 0$ for half of the inputs and $f(x) = 1$ for the other half. Therefore, we must also have that $(-1)^{f(x)} = 1$ for half of the inputs and $(-1)^{f(x)} = -1$ for the other half, meaning the total value of the sum above is exactly zero! Concretely, we have:

$$\alpha_{|0\rangle^{\otimes n}} = 0.$$

This means if f is balanced and we measure the first n qubits, **we will never obtain the state $|0\rangle^{\otimes n}$.** //

Combining our observations, we now have the following rule: when the final measurement is completed, if the outcome is $|0\rangle^{\otimes n}$, then we output 0 or "constant"; for any other result, we output 1 or "balanced".

We have now proven that the Deutsch-Josza algorithm only makes $O(1)$ calls to f in order to solve this problem, compared to the classical solution which requires $O(2^n)$.