Quantum Algorithms

Miguel Sozinho Ramalho

November, 2018

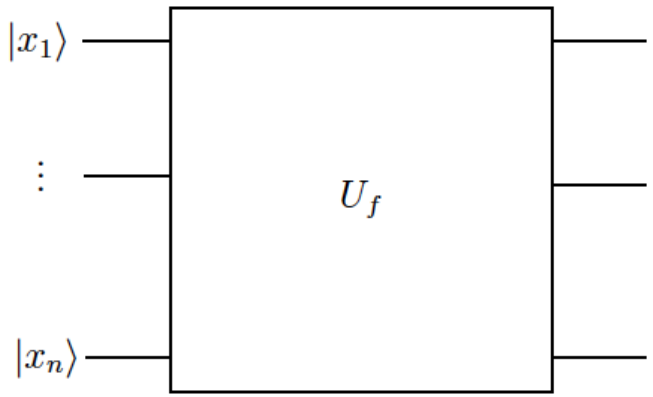
Table of contents

Introduction

Time has come for us to effectively show how a quantum computer might beat a classical one. For this, we will explore the Deutsch's algorithm (and its general case Deutsch-Jozsa). It does not solve a particularly interesting problem in classical computer science, but it does highlight the *quantum power*, so to speak. Some previous notions on **Quantum Oracles** and **Reversible Computation** are also presented.

Quantum Oracles

No, we are not about to delve into the Matrix's Oracle. Quantum Oracles are actually a useful abstractions for writing quantum algorithms. An Oracle hides a function that takes a number of qubits as input, performs a transformation, and produces the same number of qubits as output. It can be seen as black box (reversible):



Quantum Oracles

Some of the reasons for using oracles are:

- Conceptual simplification of circuits
- Allows for the comparison between classical and quantum algorithms
- Eases the visual interpretation of quantum circuits

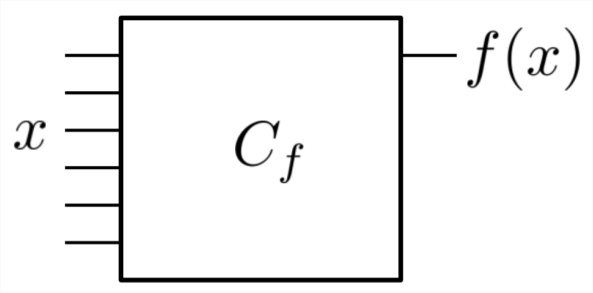
As we will see, this concept is quite used in the specification of new quantum algorithms, as it allows us to abstract from the unnecessary, following the lines of the computer science Abstraction Principle.

Quantum Reversible Computation

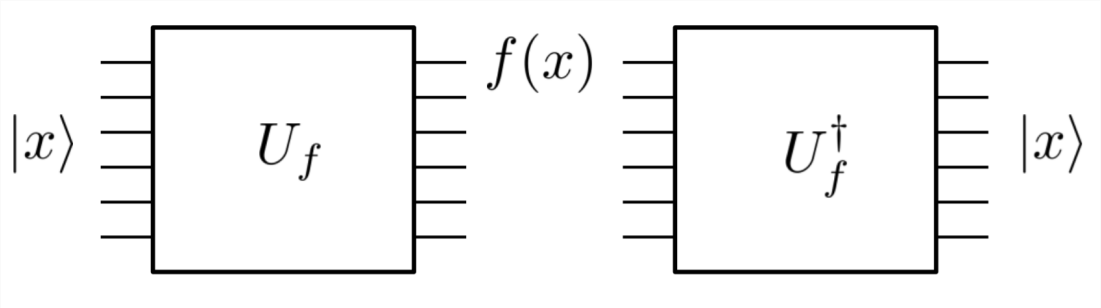
We want quantum computations to be **reversible**, and that is an intended consequence of the **unitary** nature of quantum gates. However, we must also be able to extrapolate this notion of **reversibility to oracles**, so that we can be sure (through a sort of implicit induction) that we do not lose the property of reversibility regardless of the complexity of our circuit.

Quantum Reversible Computation

Consider a classical circuit for computing a Boolean operator:

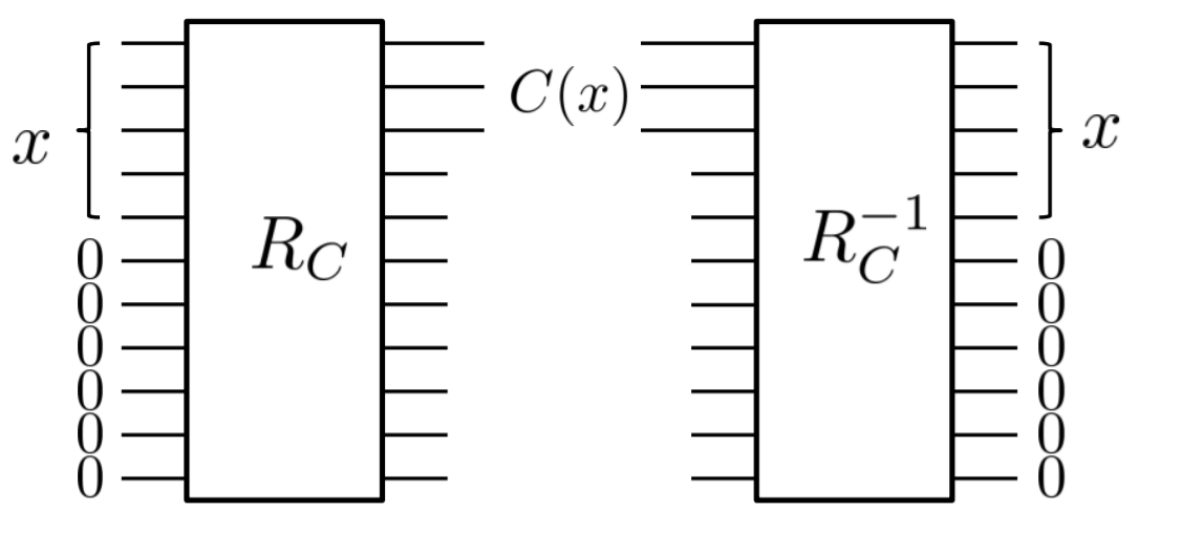


If we were to do this in quantum circuits, we would lose all the information about x and, therefore, destroy any chance of reversibility. What we really want is a way we can apply the reverse gate of U_f, $U_f^\dagger$:

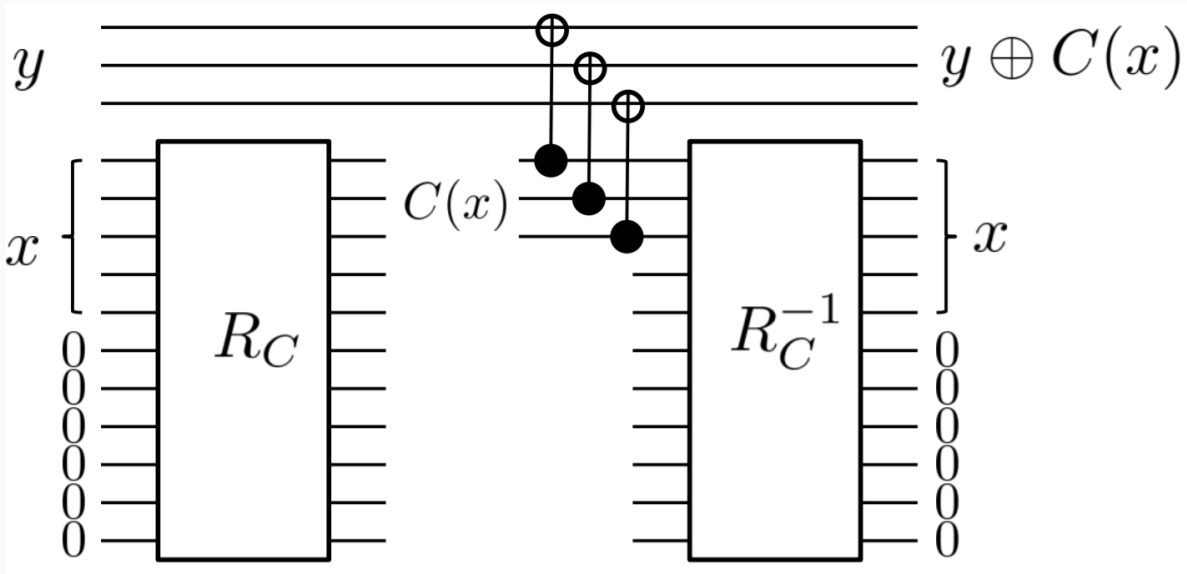


Quantum Reversible Computation

Achieving this is easier than it looks. We already have the tools! We are already able to compute any function and then reverse it back to the original state, as we only use unitary operators. But doing this means going back to the start. However, if we use some *helper qubits*, we may be on to something...

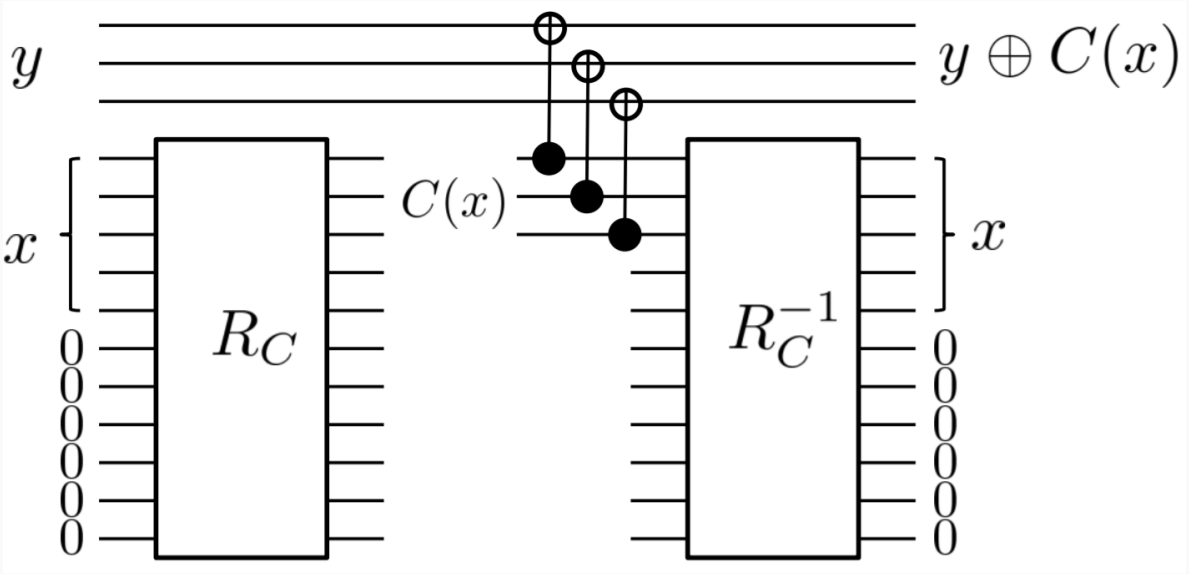


Indeed! Have you noticed the CNOT gate's equivalent in logical gates is the exclusive OR (XOR), and it can be used to copy the result of $C(x)$ onto the y qubits, if each of those qubits in $y = |0\rangle$ (0 XOR A = A).



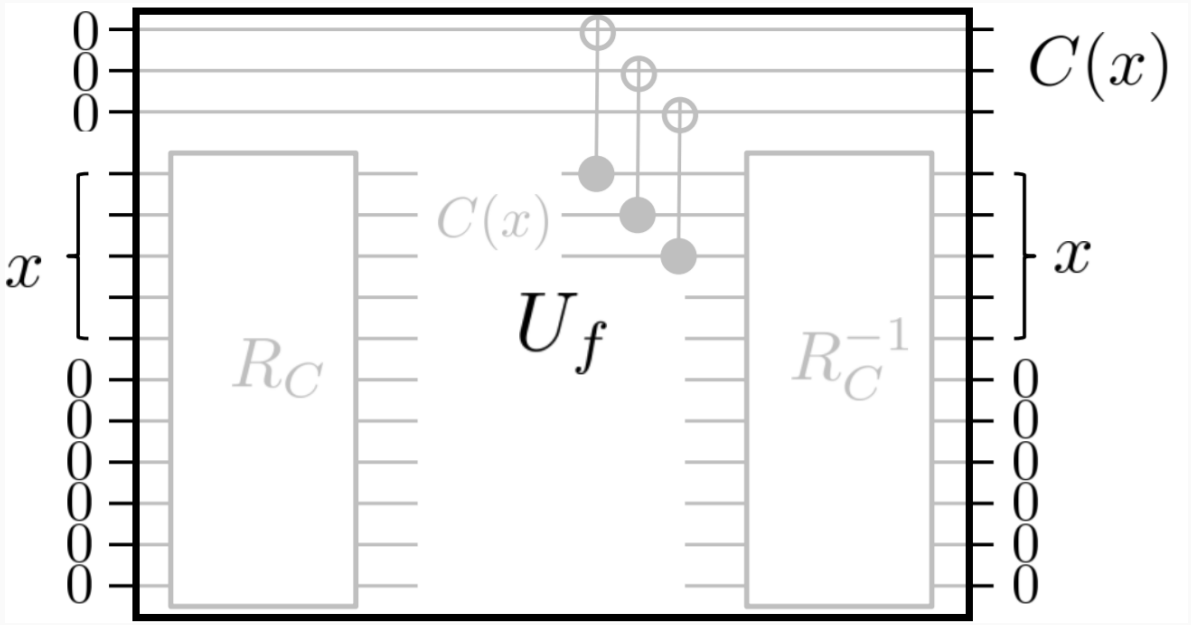
Quantum Reversible Computation

This means we can obtain our inductive reversibility notion if we apply the operations we want on our input qubits, save it using as many CNOT gates as the number of qubits in the result, and reverting the qubits we obtained by applying the inverse operations to go back to the original input. The difference being we also have the result!



Quantum Reversible Computation

In the end, even though we need some extra zeroed qubits, we can abstract our circuit onto a reversible mapping, since we never loose, only gain, information! *et voila*:



Deutsch-Jozsa Problem Formulation

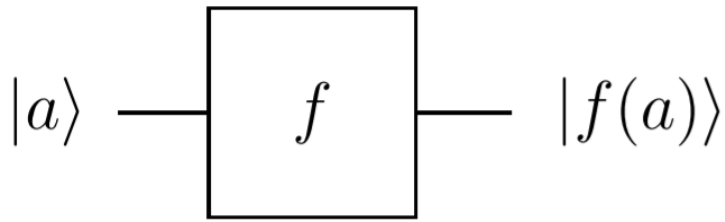
Consider a function $f : \{0,1\}^n \rightarrow \{0,1\}$ that maps an array of n bits into either 0 or 1. We do not know the logic behind it. We know that it is either constant or balanced:

Constant: its output is always 0 or always 1

Balanced: outputs 0 for half the input value and 1 for the other half

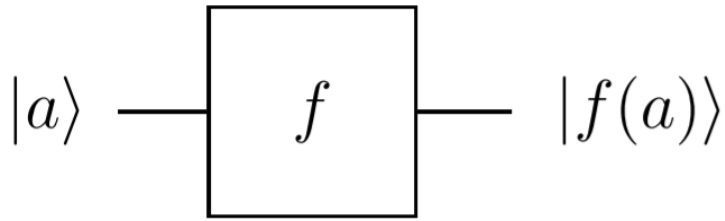
Deutsch's Problem Formulation

For the case that $n = 1$ we have $f : \{0, 1\} \to \{0, 1\}$ that maps a single bit into either 0 or 1. If we are given a black box, an **oracle**, that takes as input this two bits and outputs the unknown value:



To answer this question classically, we would always need two function invocations. We could do $f(0)$ and $f(1)$ and see if it is either constant or balanced.

Classical Algorithm



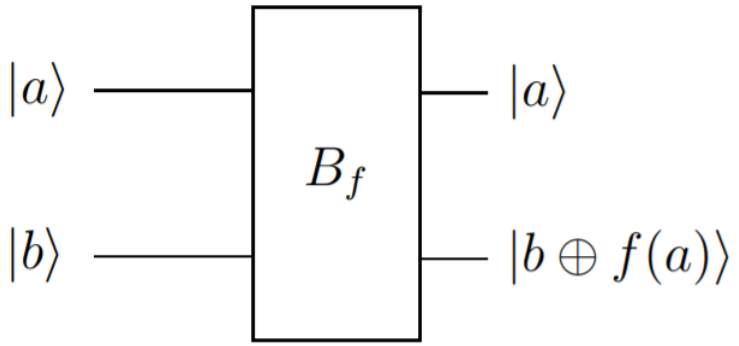
An alternative would be to check what the value of $f(0)$ XOR $f(1)$ would produce:

$$\begin{cases} f(0) \text{ XOR } f(1) = 0 \rightarrow \textit{constant} \\ f(0) \text{ XOR } f(1) = 1 \rightarrow \textit{balanced} \end{cases}$$

Yet, this would still require two invocations of the black box.

Quantum Algorithm

Before transforming it into a quantum problem, we need our black box to be an oracle which allows for reversible computation, like so:



Deutsch's Algorithm

Let us imagine the following procedure:

1. We begin with two qubits, q0 in state $|0\rangle$ and q1 in state $|1\rangle$ ($|01\rangle$).
2. We apply a Hadamard to each qubit, the result is $\frac{1}{2}(|00\rangle - |01\rangle + |10\rangle - |11\rangle)$
3. We now call our oracle, which maps $|ab\rangle$ or $|a\rangle\,|b\rangle$ (easier to interpret) into $|a\rangle\,|b \oplus f(a)\rangle$ the result is:

$$\frac{1}{2}(|0\rangle\,|0 \oplus f(0)\rangle - |0\rangle\,|1 \oplus f(0)\rangle + |1\rangle\,|0 \oplus f(1)\rangle - |1\rangle\,|1 \oplus f(1)\rangle) \tag{1}$$

Deutsch's Algorithm

Simplifying:

$$\frac{1}{2}(|0\rangle\left[|0 \oplus f(0)\rangle - |1 \oplus f(0)\rangle\right] + |1\rangle\left[(|0 \oplus f(1)\rangle - |1 \oplus f(1)\rangle\right])$$

We can now use the following equivalence:

$$|0 \oplus a\rangle - |1 \oplus a\rangle = (-1)^a(|0\rangle - |1\rangle)$$

To replace above and get:

$$\frac{1}{2}(|0\rangle\left[(-1)^{f(0)}(|0\rangle - |1\rangle)\right] + |1\rangle\left[(-1)^{f(1)}(|0\rangle - |1\rangle)\right])$$

Separating back into the 2 qubit shape:

$$[\frac{1}{\sqrt{2}}(-1)^{f|0\rangle}\,|0\rangle + \frac{1}{\sqrt{2}}(-1)^{f(|1\rangle}\,|1\rangle][\frac{1}{\sqrt{2}}\,|0\rangle - \frac{1}{\sqrt{2}}\,|1\rangle]$$

Our second qubit : $(\frac{1}{\sqrt{2}}\,|0\rangle - \frac{1}{\sqrt{2}}\,|1\rangle)$ can be ignored, and what remains is our first qubit: $(\frac{1}{\sqrt{2}}(-1)^{f|0\rangle}\,|0\rangle + \frac{1}{\sqrt{2}}(-1)^{f(|1\rangle}\,|1\rangle)$ which contains both $f(0)$ and $f(1)$! both images of f with a single pass over the oracle. This can further be simplified as:

$$(-1)^{f(0)}(\frac{1}{\sqrt{2}}\,|0\rangle + \frac{1}{\sqrt{2}}(-1)^{f(0)\oplus f(1)}\,|1\rangle)$$

Deutsch's Algorithm

Lastly, we apply a Hadamard gate on our qubit (go ahead and do this on paper) and we arrive at:

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

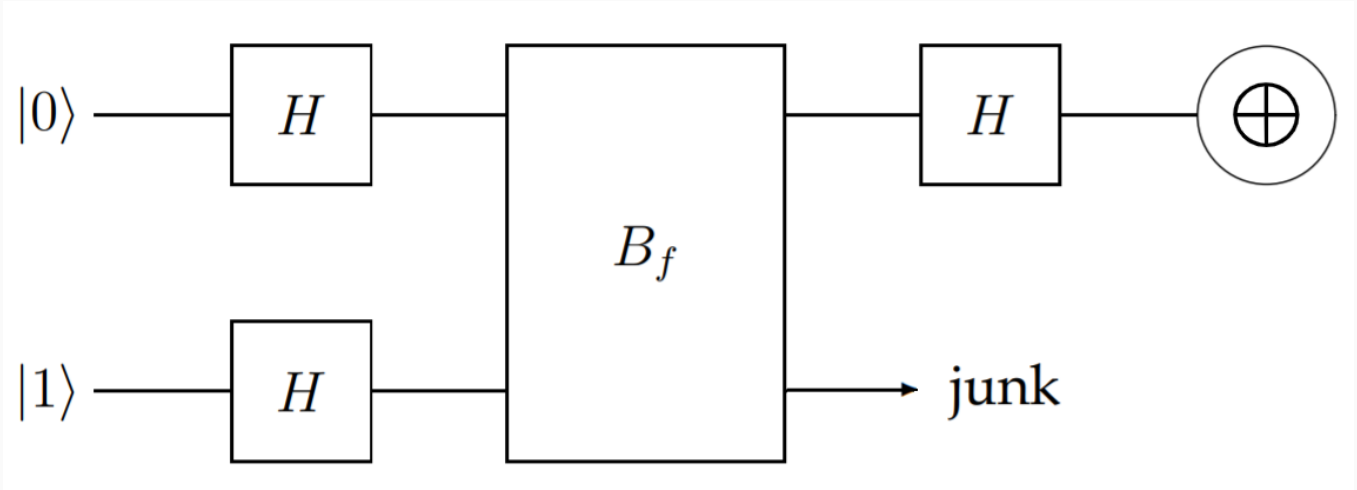
What is the meaning of this? See for yourself!

- if f is constant (00 or 11) $\rightarrow$ output is 0 (xor is 0)
- if f is balanced (01 or 10) $\rightarrow$ output is ± 1 (xor is 1)

Which, in fact, means that we can do a **single pass** over the oracle gate discover whether it is constant or balanced, an impossible feat in classical computing.

Deutsch's Algorithm

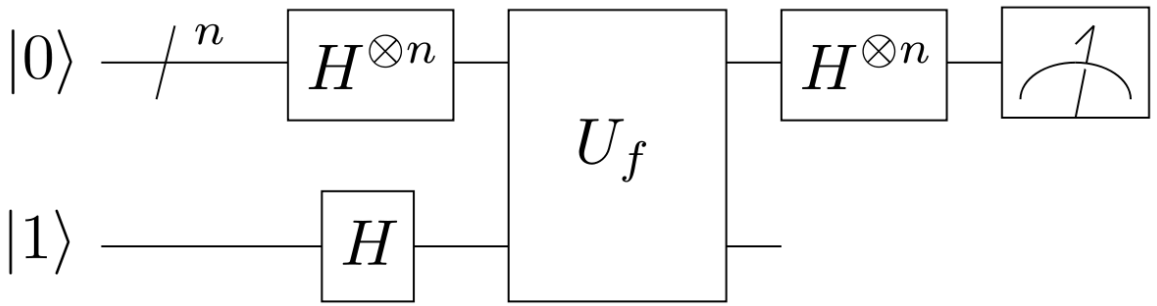
All in all, what we did was follow the following circuit:



As you can see it is one of the simplest quantum circuits we could have designed but, nonetheless, it gave us some hard time understanding how and why it worked. Do not fear, for quantum algorithms are usually hard at the beginning as they demand a new way of tackling problems!

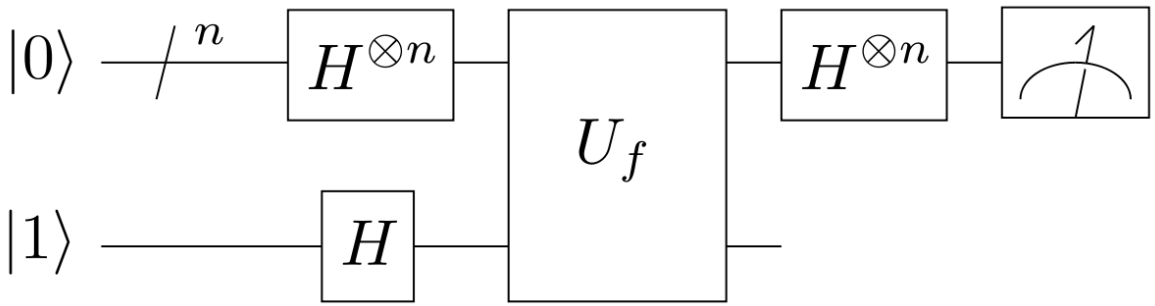
Deutsch-Jozsa Algorithm

If we go back to our function $f : \{0,1\}^n \to \{0,1\}$ it should be known that the solution to the $n = 1$ instance is generalizes to answer the same question on an n-dimensional array with... can you guess how many passes over the black box..? (Do note that a classical algorithm would need $2^{n-1} + 1$ passes)



ONE!

Just one pass. By using n qubits initialized to 0 and 1 to 1, by performing the same operations, but on more qubits, again:



Hands-on

Although Qiskit does not allow for oracles, as it would require quite a challenging approach (remember it should be able to interact with real hardware), we can simply choose a couple of operations instead of a black box and implement our Deutsch's algorithm around them, and that is precisely what you will get to do on this week's exercises!

Where to learn more?

- Deutsch-Jozsa on Youtube
- Deutsch's on Youtube
- The original Deutsch-Jozsa paper