

Course	Deutsch Algorithm
Instructor	Maurizio Ferrari Dacrema
Academic Year	2025/26

1 Deutsch Algorithm (1985)

The problem tackled by the Deutsch algorithm is as follows, we have a function that takes as input one classical bit and returns as output another classical bit:

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

Four possible functions exist, that can be grouped in two:

Constant: $f(b) = 0, f(b) = 1$;

Balanced: $f(b) = b, f(b) = \neg b$;

The *constant* functions have an output value that is always the same regardless of the input. The *balanced* functions are called like this because the number of times they return 0 and 1 are equal.

Crucially, function f is provided to us as a *black box*, meaning that we can use it as part of a quantum circuit to do computation, but we cannot know how it is implemented. In quantum computing terminology, black box functions are also called *oracles*. The goal of the Deutsch algorithm is, given a quantum circuit that contains function f , to identify if the function is constant or balanced.

Computational Complexity: Given that the goal of the Deutsch algorithm is to identify whether a function is constant or balanced, the way we measure computational complexity is by counting how many times we have to query the black-box function f before we are able to identify it.

In classical computing we have to query function f two times, since we need to know its output for both $f(0)$ and $f(1)$. The decision tree is as follows:

$$\text{Query } f(0) = \begin{cases} 0 & \text{if } f(b) = 0 \text{ or } f(b) = b, \\ 1 & \text{if } f(b) = 1 \text{ or } f(b) = \neg b, \end{cases} \quad \text{Query } f(1) = \begin{cases} 0 & \text{if } f(b) = 0 \\ 1 & \text{if } f(b) = b \\ 0 & \text{if } f(b) = \neg b \\ 1 & \text{if } f(b) = 1 \end{cases}$$

We will see that the Deutsch algorithm allows to identify the function in only 1 query, exhibiting a small but provable quantum speedup.

A single-qubit function oracle is not enough: In order to understand the reasoning behind the Deutsch algorithm, the first thing we have to do is to discuss how to create the quantum circuits implementing the four oracle functions. While the Deutsch algorithm does not use knowledge of how the function is implemented, we nonetheless must be able to implement all four of them as a quantum circuit. The formalism we adopt is of a quantum circuit with a single qubit that will have as initial state the input of our function, b , on which we will apply a unitary operation that implements function f , denoted as U_f :

$$|b\rangle \longrightarrow \boxed{U_f} \longrightarrow |f(b)\rangle$$

It is easy to see how the two balanced functions should be implemented:

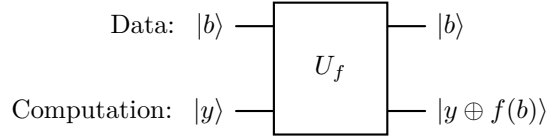
- For function $f(b) = b$ the unitary U_f is the identity
- For function $f(b) = \neg b$ the unitary U_f is the not gate

The problem arises when trying to implement the constant functions. The constant functions are *not invertible*, meaning that it is impossible to create a quantum circuit by using this formalism.

$$|b\rangle \longrightarrow |b\rangle$$

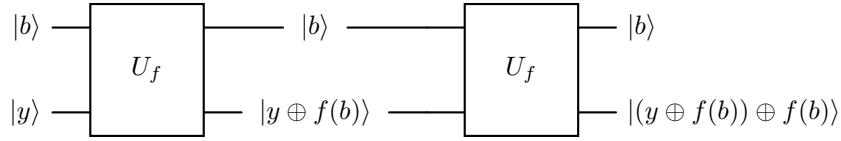
$$|b\rangle \longrightarrow \boxed{X} \longrightarrow |-b\rangle$$

How to make constant functions invertible: As we have seen the constant functions are *not invertible*, meaning that we cannot implement them as-is in a quantum computer. This issue arises in a number of scenarios, for example for logical operations such as AND, OR. It is possible to take a function that is non invertible and transform it into an invertible one by using additional qubits. For the purposes of the Deutsch algorithm we need to use two qubits. The first qubit is the *data qubit*, that contains the data provided as input to the function and remains unchanged. The second qubit is the *computation qubit* that contains a combination of the result of function f and the initial qubit state y . The general formulation is this:



with $b, y \in \{0, 1\}$ refer to classical bits, $|b\rangle$ and $|y\rangle$ are the corresponding quantum states, and $\oplus$ is the bit-wise XOR (also known as the addition modulo 2). Remember that computations are performed in the computational basis $|0\rangle, |1\rangle$. Even if $|b\rangle$ is a superposition we will apply U_f to each one of the basis states that constitutes the superposition, hence $|0\rangle$ or $|1\rangle$, and on those the bit-wise XOR behaves like in the classical case.

If we put two of these U_f gates in series we can see that they are invertible regardless for whether $f(b)$ is invertible or not. In particular, we can see that $U_f = U_f^{-1}$.



If we look at the output, we can see that we obtain $|b, y\rangle$:

$$U_f U_f |b, y\rangle = U_f |b, y \oplus f(b)\rangle = |b, y \oplus f(b) \oplus f(b)\rangle = |b, y \oplus 0\rangle = |b, y\rangle$$

Two-qubit function oracle: We can now write the circuits to implement all the four functions by using two qubits. The formalism is rather general but we can simplify it, in particular we can set the initial value of the computation qubit to $|y\rangle = |0\rangle$, meaning that after the application of U_f it will simply be $|0 \oplus f(b)\rangle = |f(b)\rangle$, see Figure 1.

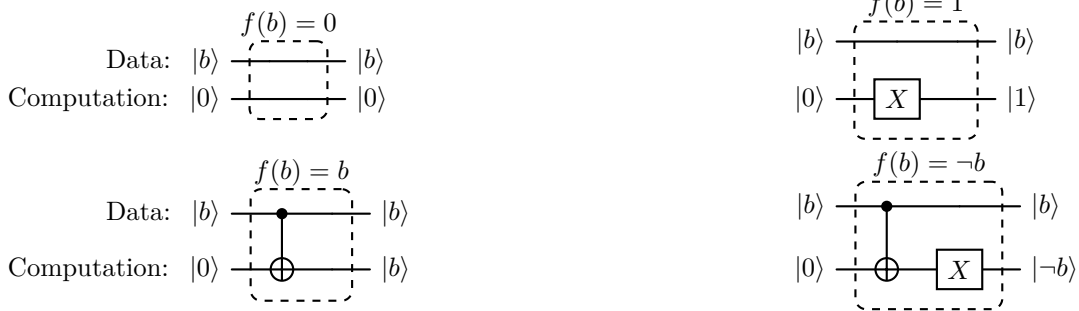
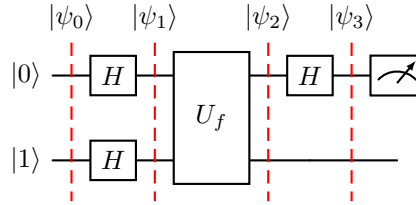


Figure 1: Circuits implementing the four functions.

Algorithm intuition: Notice how all balanced functions have a C_X gate, while the constant ones do not and we will use this fact to our advantage. The X gate however may act as a confounding factor. Remember that function f is provided to us as an oracle, meaning that we can use it to do computation but we cannot see how it is implemented. The solution is to build an algorithm that will be able to detect, via phase kickback, if function f contains a C_X gate. To do so we need to:

- Put the data qubit in a state that will allow us to produce the phase kickback (superposition), for example $|+\rangle$ or $|-\rangle$;
- Put the computation qubit in a state that is an eigenvector of the X gate that will kickback a phase we can detect (must not be a phase of 1, but a -1 would do). We already know that both $|+\rangle$ and $|-\rangle$ are eigenvectors of X, but $|+\rangle$ has eigenvalue $+1$ which is not useful, $|-\rangle$ on the other hand has eigenvalue -1 which will produce an effect on the control qubit.
- At this point we need to be able to distinguish the $|+\rangle$ and $|-\rangle$ states on the control qubit with a measurement. This can be done by applying a Hadamard gate to transform the superposition into a state we can measure with certainty $H|+\rangle = |0\rangle$, $H|-\rangle = |1\rangle$

The final circuit is as follows:



Let's now consider all possible cases:

Constant functions If function f is constant the algorithm will proceed as follows:

$$\begin{aligned} |\psi_0\rangle &= |0, 1\rangle \\ |\psi_1\rangle &= (H \otimes H) |0, 1\rangle = |+, -\rangle \end{aligned}$$

If $f(b) = 0$

$$\begin{aligned} |\psi_2\rangle &= (I \otimes I) |+, -\rangle = |+, -\rangle \\ |\psi_3\rangle &= (H \otimes I) |+, -\rangle = |0, -\rangle \end{aligned}$$

If $f(b) = 1$

$$\begin{aligned} |\psi_2\rangle &= (I \otimes X) |+, -\rangle = |+\rangle \otimes \frac{1}{\sqrt{2}} (-|0\rangle + |1\rangle) = -|+, -\rangle \\ |\psi_3\rangle &= -(H \otimes I) |+, -\rangle = -|0, -\rangle \end{aligned}$$

Notice how in both cases the control qubit has remained in the $|+\rangle$ state, the only difference between the two functions is a global phase, which as we know has no effect. At the end, if the function is constant we will measure 0 with 100% probability.

Balanced functions If function f is balanced the algorithm will proceed as follows:

$$\begin{aligned} |\psi_0\rangle &= |0, 1\rangle \\ |\psi_1\rangle &= (H \otimes H) |0, 1\rangle = |+, -\rangle \end{aligned}$$

If $f(b) = b$

$$\begin{aligned}
|\psi_2\rangle &= C_X |+, -\rangle = C_X \frac{1}{\sqrt{2}} (|0\rangle + |1\rangle) \otimes \frac{1}{\sqrt{2}} (|0\rangle - |1\rangle) \\
&= C_X \frac{1}{2} (|0, 0\rangle - |0, 1\rangle + |1, 0\rangle - |1, 1\rangle) \\
&= \frac{1}{2} (|0, 0\rangle - |0, 1\rangle + |1\rangle \otimes X|0\rangle - |1\rangle \otimes X|1\rangle) \\
&= \frac{1}{2} (|0, 0\rangle - |0, 1\rangle + |1, 1\rangle - |1, 0\rangle) \\
&= \frac{1}{2} (|0\rangle \otimes (|0\rangle - |1\rangle) + |1\rangle \otimes (|1\rangle - |0\rangle)) \\
&= \frac{1}{\sqrt{2}} (|0, -\rangle - |1, -\rangle) = \frac{1}{\sqrt{2}} (|0\rangle - |1\rangle) \otimes |-\rangle = |-, -\rangle
\end{aligned}$$

Note that we could have just used the phase kickback definition to compute the result without going through the entire set of calculations.

$$|\psi_3\rangle = (H \otimes I) |-, -\rangle = |1, -\rangle$$

If $f(b) = \neg b$

$$|\psi_2\rangle = (I \otimes X) C_X |+, -\rangle$$

We have already computed $C_X |+, -\rangle$ for $f(b) = b$ so we take that result and apply the additional gates on it:

$$\begin{aligned}
|\psi_2\rangle &= (I \otimes X) |-, -\rangle = |-\rangle \otimes X \frac{1}{\sqrt{2}} (|0\rangle - |1\rangle) \\
&= |-\rangle \otimes (-|-\rangle) = -|-, -\rangle \\
|\psi_3\rangle &= -(H \otimes I) |-, -\rangle = -|1, -\rangle
\end{aligned}$$

Notice how in both cases the control qubit has remained in the $|-\rangle$ state, the only difference between the two functions is a global phase, which as we know has no effect. At the end, if the function is balanced we will measure 1 with 100% probability.

General formulation: Now let's use a more general formulation:

$$\begin{aligned}
|\psi_2\rangle &= U_f |+, -\rangle = U_f \frac{1}{\sqrt{2}} (|0\rangle + |1\rangle) \frac{1}{\sqrt{2}} (|0\rangle - |1\rangle) \\
&= U_f \frac{1}{2} (|0, 0\rangle - |0, 1\rangle + |1, 0\rangle - |1, 1\rangle) \\
&= \frac{1}{2} (|0, 0 \oplus f(0)\rangle - |0, 1 \oplus f(0)\rangle + |1, 0 \oplus f(1)\rangle - |1, 1 \oplus f(1)\rangle)
\end{aligned}$$

At this point we can use these equivalences:

$$\begin{aligned}
|b, 0 \oplus f(b)\rangle &= |b, f(b)\rangle \\
|b, 1 \oplus f(b)\rangle &= |b, \neg f(b)\rangle \\
|\psi_2\rangle &= \frac{1}{2} (|0, f(0)\rangle - |0, \neg f(0)\rangle + |1, f(1)\rangle - |1, \neg f(1)\rangle) \\
&= \frac{1}{2} (|0\rangle \otimes (|f(0)\rangle - |\neg f(0)\rangle) + |1\rangle \otimes (|f(1)\rangle - |\neg f(1)\rangle))
\end{aligned}$$

Now let's look at what happens for the four functions:

$$|\psi_2\rangle = \frac{1}{2} (|0\rangle \otimes (|0\rangle - |1\rangle) + |1\rangle \otimes (|0\rangle - |1\rangle)) = |+, -\rangle$$

If $f(b) = 1$

$$\begin{aligned}
|\psi_2\rangle &= \frac{1}{2} (|0\rangle \otimes (|1\rangle - |0\rangle) + |1\rangle \otimes (|1\rangle - |0\rangle)) \\
&= \frac{1}{\sqrt{2}} ((-1)|0\rangle \otimes |-\rangle + (-1)|1\rangle \otimes |-\rangle) = -|+, -\rangle
\end{aligned}$$

If $f(b) = b$

$$\begin{aligned} |\psi_2\rangle &= \frac{1}{2} (|0\rangle \otimes (|0\rangle - |1\rangle) + |1\rangle \otimes (|1\rangle - |0\rangle)) \\ &= \frac{1}{\sqrt{2}} (|0\rangle \otimes |-\rangle + (-1) |1\rangle \otimes |-\rangle) = |-, -\rangle \end{aligned}$$

If $f(b) = -b$

$$\begin{aligned} |\psi_2\rangle &= \frac{1}{2} (|0\rangle \otimes (|1\rangle - |0\rangle) + |1\rangle \otimes (|0\rangle - |1\rangle)) \\ &= \frac{1}{\sqrt{2}} ((-1) |0\rangle \otimes |-\rangle + |1\rangle \otimes |-\rangle) = -|-, -\rangle \end{aligned}$$

We can see how $f(b)$ controls the sign, based on this we can represent $|\psi_2\rangle$ with a single equation:

$$|\psi_2\rangle = \frac{1}{\sqrt{2}} \left((-1)^{f(0)} |0, -\rangle + (-1)^{f(1)} |1, -\rangle \right)$$

Since $f(b)$ is applied on each basis state, we can write the equation in a more compact way by using $f(e)$ to represent the value of function f when applied on the first qubit e in the computational basis, i.e., $|0\rangle$ or $|1\rangle$:

$$|\psi_2\rangle = \frac{1}{\sqrt{2}} \left((-1)^{f(0)} |0, -\rangle + (-1)^{f(1)} |1, -\rangle \right) = \frac{1}{\sqrt{2}} \sum_{e \in \{0,1\}} (-1)^{f(e)} |e\rangle \otimes |-\rangle$$

Summing up, if the function is constant we will measure 0 while if it is balanced we will measure 1. The effect of the X gate ends up as a global phase which is irrelevant for us.

Deutsch's algorithm shows that a quantum computer can solve this problem with a small speed-up compared to the classical case. In particular, with Deutsch's algorithm we only need to query the black-box function (oracle) $f(b)$ once, by using its corresponding quantum operator, while classically we need to query $f(b)$ twice. In this sense, Deutsch's algorithm is the first simple example of a quantum query advantage.

Even though this speed-up is small, the algorithm is still important because it is one of the simplest examples where superposition, interference and phase kickback are used together to extract a global property of a black-box function, without checking all outputs one by one. Because of this, Deutsch's algorithm is usually seen as a basic prototype for more advanced quantum algorithms.

This same idea appears again in the *Deutsch-Jozsa algorithm*, which considers black-box functions $f : \{0,1\}^n \rightarrow \{0,1\}$. In that case, the quantum algorithm still needs only one query to the black-box function, while the classical counterpart may require up to $2^{n-1} + 1$ queries in the worst case. A related example is the *Bernstein-Vazirani algorithm*, where the goal is to determine an unknown bit string hidden in a black-box function. Also in that case, the quantum algorithm can recover the whole string with a single query, while classically one would need n queries. So, even if Deutsch's algorithm by itself is a very small example, it already contains the main intuition behind a broader class of quantum algorithms where a quantum computer can determine a global property of a function directly, instead of reconstructing all of its values separately.