ICT Solutions for Brilliant Minds

Introduction to quantum computing and FiQCI – Deutsch Algorithm

Olli Mukkula – Application specialist at CSC

Deutsch Algorithm

- Deutsch algorithm was invented by David Deutch in 1985. It was the first algorithm developed for quantum computers

Prof. David Deutsch - Creator of the first quantum algorithm

Source: [https://www.daviddeutsch.org.uk/]

Deutsch Algorithm

- Deutsch algorithm was invented by David Deutch in 1985. It was the first algorithm developed for quantum computers
- Deterministic algorithm with 100% success rate on noiseless quantum computer

Prof. David Deutsch - Creator of the first quantum algorithm

Source: [https://www.daviddeutsch.org.uk/]

Deutsch Algorithm

- Deutsch algorithm was invented by David Deutch in 1985. It was the first algorithm developed for quantum computers
- Deterministic algorithm with 100% success rate on noiseless quantum computer
- Solves the specific problem (Deutsch problem) exponentially faster than any classical algorithm

Prof. David Deutsch - Creator of the first quantum algorithm

Source: [https://www.daviddeutsch.org.uk/]

Deutsch Algorithm

- The Deutsch problem:

Consider a function $f : \{0,1\} \to \{0,1\}$

Is it true that $f(0) = f(1)$

Deutsch Algorithm

- The Deutsch problem:

Consider a function $f : \{0,1\} \to \{0,1\}$

Is it true that $f(0) = f(1)$

- There are four possible functions:

Deutsch Algorithm

- The Deutsch problem:

Consider a function $f : \{0,1\} \to \{0,1\}$

Is it true that $f(0) = f(1)$

- There are four possible functions:

Always zero

$$f(0) = 0$$

$$f(1) = 0$$

Deutsch Algorithm

- The Deutsch problem:

Consider a function $f : \{0,1\} \to \{0,1\}$

Is it true that $f(0) = f(1)$

- There are four possible functions:

Always zero

$$f(0) = 0$$
$$f(1) = 0$$

Always one

$$f(0) = 1$$
$$f(1) = 1$$

Deutsch Algorithm

- The Deutsch problem:

Consider a function $f : \{0,1\} \to \{0,1\}$

Is it true that $f(0) = f(1)$

- There are four possible functions:

Always zero

$$f(0) = 0$$
$$f(1) = 0$$

Identity-function

$$f(0) = 0$$
$$f(1) = 1$$

Always one

$$f(0) = 1$$
$$f(1) = 1$$

Deutsch Algorithm

- The Deutsch problem:

Consider a function $f : \{0,1\} \to \{0,1\}$

Is it true that $f(0) = f(1)$

- There are four possible functions:

Always zero

$f(0) = 0$

$f(1) = 0$

Identity-function

$f(0) = 0$

$f(1) = 1$

Always one

$f(0) = 1$

$f(1) = 1$

NOT-function

$f(0) = 1$

$f(1) = 0$

Deutsch Algorithm Introduction

- The Deutsch problem:

Consider a function $f : \{0,1\} \to \{0,1\}$

Is it true that $f(0) = f(1)$

- There are four possible functions:

Is the function constant or varied?

Constant functions

Always zero

$$f(0) = 0$$
$$f(1) = 0$$

Always one

$$f(0) = 1$$
$$f(1) = 1$$

Varied functions

Identity-function

$$f(0) = 0$$
$$f(1) = 1$$

NOT-function

$$f(0) = 1$$
$$f(1) = 0$$

Deutsch Algorithm
Introduction

- Deutsch problem is a black box problem

Deutsch Algorithm

- Deutsch problem is a black box problem
- Black box oracle:

Deutsch Algorithm

- Deutsch problem is a black box problem
- Black box oracle:
  - Has an input and an output

Deutsch Algorithm

- Deutsch problem is a black box problem
- Black box oracle:
  - Has an input and an output
  - Internal workings are unknown

Deutsch Algorithm

- If function is $f(x) = \{0,1\}$ is a black box, it is non-reversible
  - Can't solve x based on the output

Deutsch Algorithm

- If function is $f(x) = \{0,1\}$ is a black box, it is non-reversible
  - Can't solve x based on the output
- One can, however, determine what the function does by observing the output for different inputs

Deutsch Algorithm

- If function is $f(x) = \{0,1\}$ is a black box, it is non-reversible
  - Can't solve x based on the output
- One can, however, determine what the function does by observing the output for different inputs
- Classically, one needs to evaluate function twice
  - Once for $x = 0$
  - Once for $x = 1$

Deutsch Algorithm

- If function is $f(x) = \{0,1\}$ is a black box, it is generally non-reversible
  - Can't solve x based on the output
- One can, however, determine what the function does by observing the output for different inputs
- Classically, one needs to evaluate function twice
  - Once for $x = 0$
  - Once for $x = 1$
- Quantum computer only needs <u>one evaluation!</u>

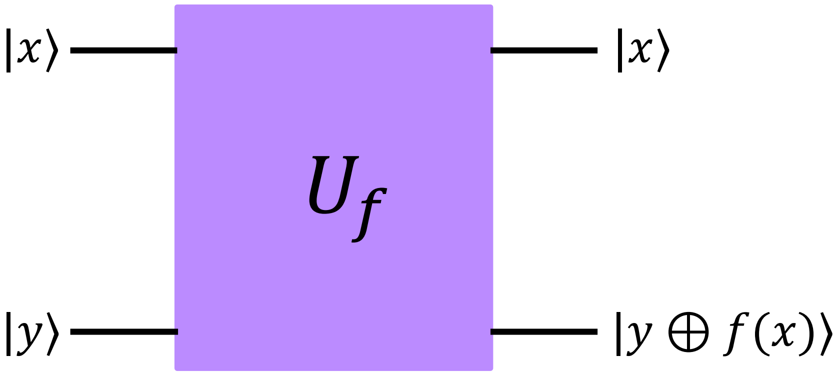
Deutsch Algorithm

- Function $f(x)$ is not directly realizable with quantum computers

Deutsch Algorithm

- Function $f(x)$ is not directly realizable with quantum computers

- Using two qubits, we can build Quantum oracle U_f ('black box' quantum gate) that implements $f(x)$

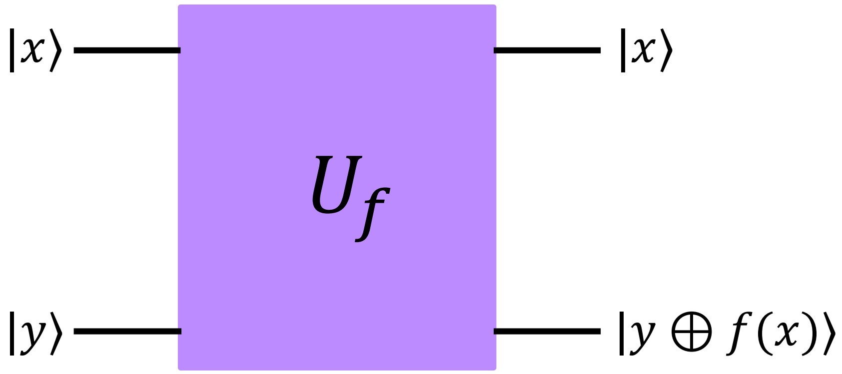


Deutsch Algorithm

- Function $f(x)$ is not directly realizable with quantum computers

- Using two qubits, we can build Quantum oracle U_f ('black box' quantum gate) that implements $f(x)$

- Let's analyze the output state $|x\ \ y \oplus f(x)\rangle$ to see what it can tell us about the oracle



Deutsch Algorithm

- Operation $\oplus$ is called exclusive-OR (XOR

x	y	$x \oplus y$
0	0	0
0	1	1
1	0	1
1	1	0

Table of XOR operations

Deutsch Algorithm

- Operation $\oplus$ is called exclusive-OR (XOR)
- We can calculate all the possible output states

x	y	$x \oplus y$
0	0	0
0	1	1
1	0	1
1	1	0

Table of XOR operations

	Output state: $\mathrm{U_f}\lvert\mathrm{x\ y}\rangle = \lvert x\ \ y \oplus f(x)\rangle$			
Initial state	$f(0)=0$ $f(1)=0$	$f(0)=1$ $f(1)=1$	$f(0)=0$ $f(1)=1$	$f(0)=1$ $f(1)=0$
$\lvert 00\rangle$	$\lvert 00\rangle$	$\lvert 01\rangle$	$\lvert 00\rangle$	$\lvert 01\rangle$
$\lvert 01\rangle$	$\lvert 01\rangle$	$\lvert 00\rangle$	$\lvert 01\rangle$	$\lvert 00\rangle$
$\lvert 10\rangle$	$\lvert 10\rangle$	$\lvert 11\rangle$	$\lvert 11\rangle$	$\lvert 10\rangle$
$\lvert 11\rangle$	$\lvert 11\rangle$	$\lvert 10\rangle$	$\lvert 10\rangle$	$\lvert 11\rangle$

Deutsch Algorithm

x	y	$x \oplus y$
0	0	0
0	1	1
1	0	1
1	1	0

Table of XOR operations

- Operation $\oplus$ is called exclusive-OR (XOR)
- We can calculate all the possible output states

	Output state: $\mathrm{U_f}\lvert \mathrm{x\ y}\rangle = \lvert x\ \ y \oplus f(x)\rangle$			
Initial state	$f(0) = 0$ $f(1) = 0$	$f(0) = 1$ $f(1) = 1$	$f(0) = 0$ $f(1) = 1$	$f(0) = 1$ $f(1) = 0$
$\lvert 00\rangle$	$\boxed{\lvert 00\rangle}$	$\lvert 01\rangle$	$\lvert 00\rangle$	$\lvert 01\rangle$
$\lvert 01\rangle$	$\boxed{\lvert 01\rangle}$	$\lvert 00\rangle$	$\lvert 01\rangle$	$\lvert 00\rangle$
$\lvert 10\rangle$	$\boxed{\lvert 10\rangle}$	$\lvert 11\rangle$	$\lvert 11\rangle$	$\lvert 10\rangle$
$\lvert 11\rangle$	$\boxed{\lvert 11\rangle}$	$\lvert 10\rangle$	$\lvert 10\rangle$	$\lvert 11\rangle$

- $y \oplus f(x)$ is identity on both qubits

Deutsch Algorithm

x	y	$x \oplus y$
0	0	0
0	1	1
1	0	1
1	1	0

Table of XOR operations

- Operation $\oplus$ is called exclusive-OR (XOR)
- We can calculate all the possible output states

	Output state: $\mathrm{U_f}\lvert\mathrm{x\ y}\rangle = \lvert x\ \ y \oplus f(x)\rangle$			
Initial state	$f(0) = 0$ $f(1) = 0$	$f(0) = 1$ $f(1) = 1$	$f(0) = 0$ $f(1) = 1$	$f(0) = 1$ $f(1) = 0$
$\lvert 00\rangle$	$\lvert 00\rangle$	$\lvert 01\rangle$	$\lvert 00\rangle$	$\lvert 01\rangle$
$\lvert 01\rangle$	$\lvert 01\rangle$	$\lvert 00\rangle$	$\lvert 01\rangle$	$\lvert 00\rangle$
$\lvert 10\rangle$	$\lvert 10\rangle$	$\lvert 11\rangle$	$\lvert 11\rangle$	$\lvert 10\rangle$
$\lvert 11\rangle$	$\lvert 11\rangle$	$\lvert 10\rangle$	$\lvert 10\rangle$	$\lvert 11\rangle$

- $y \oplus f(x)$ is identity on both qubits
- $y \oplus f(x)$ is NOT (X-gate) on 2nd-qubit

Deutsch Algorithm

- Operation $\oplus$ is called exclusive-OR (XOR)
- We can calculate all the possible output states

x	y	$x \oplus y$
0	0	0
0	1	1
1	0	1
1	1	0

Table of XOR operations

	Output state: $\mathrm{U_f}\lvert\mathrm{x\ y}\rangle = \lvert x\ \ y \oplus f(x)\rangle$			
Initial state	$f(0) = 0$ $f(1) = 0$	$f(0) = 1$ $f(1) = 1$	$f(0) = 0$ $f(1) = 1$	$f(0) = 1$ $f(1) = 0$
$\lvert 00\rangle$	$\lvert 00\rangle$	$\lvert 01\rangle$	$\lvert 00\rangle$	$\lvert 01\rangle$
$\lvert 01\rangle$	$\lvert 01\rangle$	$\lvert 00\rangle$	$\lvert 01\rangle$	$\lvert 00\rangle$
$\lvert 10\rangle$	$\lvert 10\rangle$	$\lvert 11\rangle$	$\lvert 11\rangle$	$\lvert 10\rangle$
$\lvert 11\rangle$	$\lvert 11\rangle$	$\lvert 10\rangle$	$\lvert 10\rangle$	$\lvert 11\rangle$

- $y \oplus f(x)$ is identity on both qubits
- $y \oplus f(x)$ is NOT (X-gate) on 2nd-qubit
- $y \oplus f(x)$ is CNOT targeted to 2nd qubit

Deutsch Algorithm

- Operation $\oplus$ is called exclusive-OR (XOR)
- We can calculate all the possible output states

x	y	$x \oplus y$
0	0	0
0	1	1
1	0	1
1	1	0

Table of XOR operations

	Output state: $U_f\vert x\ y\rangle = \vert x\ \ y \oplus f(x)\rangle$			
Initial state	$f(0)=0$ $f(1)=0$	$f(0)=1$ $f(1)=1$	$f(0)=0$ $f(1)=1$	$f(0)=1$ $f(1)=0$
$\vert 00\rangle$	$\vert 00\rangle$	$\vert 01\rangle$	$\vert 00\rangle$	$\vert 01\rangle$
$\vert 01\rangle$	$\vert 01\rangle$	$\vert 00\rangle$	$\vert 01\rangle$	$\vert 00\rangle$
$\vert 10\rangle$	$\vert 10\rangle$	$\vert 11\rangle$	$\vert 11\rangle$	$\vert 10\rangle$
$\vert 11\rangle$	$\vert 11\rangle$	$\vert 10\rangle$	$\vert 10\rangle$	$\vert 11\rangle$

- $y \oplus f(x)$ is identity on both qubits
- $y \oplus f(x)$ is NOT (X-gate) on 2nd-qubit
- $y \oplus f(x)$ is CNOT targeted to 2nd qubit
- $y \oplus f(x)$ is NOT-CNOT targeted to 2nd qubit

Deutsch Algorithm

- Quantum oracle is reversible even if $f(x)$ is non-reversible

Deutsch Algorithm

- Quantum oracle is reversible even if $f(x)$ is non-reversible
- Quantum oracle <u>alone is not enough</u> to tell if $f(0) = f(1)$

Deutsch Algorithm

- Quantum oracle is reversible even if $f(x)$ is non-reversible
- Quantum oracle alone is not enough to tell if $f(0) = f(1)$
- Deutsch algorithm uses quantum superposition and entanglement to solve the problem

Deutsch Algorithm

- Quantum oracle is reversible even if $f(x)$ is non-reversible
- Quantum oracle alone is not enough to tell if $f(0) = f(1)$
- Deutsch algorithm uses quantum superposition and entanglement to solve the problem
  - Oracle acts on superposition states with different phases.

Deutsch Algorithm

- Quantum oracle is reversible even if $f(x)$ is non-reversible
- Quantum oracle alone is not enough to tell if $f(0) = f(1)$
- Deutsch algorithm uses quantum superposition and entanglement to solve the problem
  - Oracle acts on superposition states with different phases.
  - If $f(x)$ is varied, the quantum oracle is two-qubit operation (CNOT or NOT-CNOT). Phase kickback will reverse the phase of the 1st qubit.

Deutsch Algorithm

- Quantum oracle is reversible even if $f(x)$ is non-reversible
- Quantum oracle alone is not enough to tell if $f(0) = f(1)$
- Deutsch algorithm uses quantum superposition and entanglement to solve the problem
  - Oracle acts on superposition states with different phases.
  - If $f(x)$ is varied, the quantum oracle is two-qubit operation (CNOT or NOT-CNOT). Phase kickback will reverse the phase of the 1st qubit.
  - Reversing the superposition of the 1st qubit causes it flip if its phase was reversed

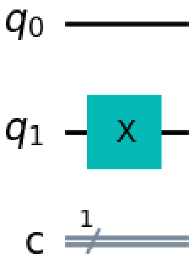
Deutsch Algorithm

- Quantum oracle is reversible even if $f(x)$ is non-reversible
- Quantum oracle alone is not enough to tell if $f(0) = f(1)$
- Deutsch algorithm uses quantum superposition and entanglement to solve the problem
  - Oracle acts on superposition states with different phases.
  - If $f(x)$ is varied, the quantum oracle is two-qubit operation (CNOT or NOT-CNOT). Phase kickback will reverse the phase of the 1st qubit.
  - Reversing the superposition of the 1st qubit causes it flip if its phase was reversed
  - Whether the 1st qubit was flipped reveals if $f(x)$ is constant or varied

Deutsch Algorithm

- Deutsch algorithm implementation:

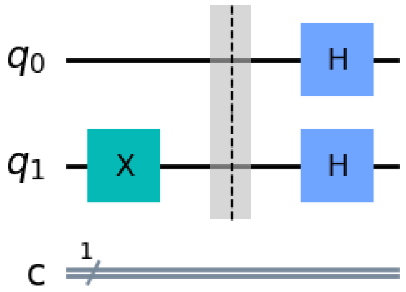
1. Allocate qubits (Flip the 2nd qubit with X-gate): $|0\rangle|1\rangle$



Deutsch Algorithm

- Deutsch algorithm implementation:

1. Allocate qubits (Flip the 2nd qubit with X-gate): $|0\rangle|1\rangle$
2. Apply Hadamard gates to create superposition: $\rightarrow \left[\frac{|0\rangle + |1\rangle}{\sqrt{2}}\right]\left[\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right]$



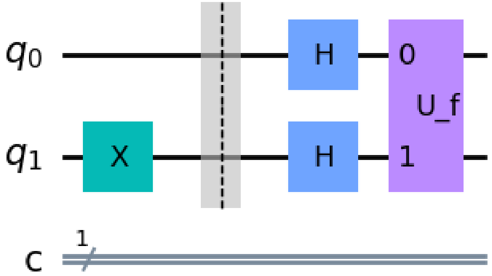
Deutsch Algorithm

- Deutsch algorithm implementation:

1. Allocate qubits (Flip the 2nd qubit with X-gate): $|0\rangle|1\rangle$

2. Apply Hadamard gates to create superposition: $\rightarrow \left[\frac{|0\rangle + |1\rangle}{\sqrt{2}}\right]\left[\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right]$

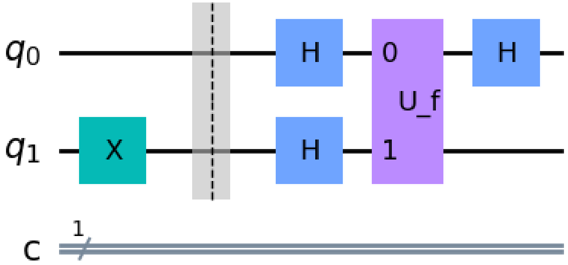
3. Apply our quantum oracle gate: $\rightarrow \left[\frac{(-1)^{f(0)}|0\rangle + (-1)^{f(1)}|1\rangle}{\sqrt{2}}\right]\left[\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right]$



Deutsch Algorithm

- Deutsch algorithm implementation:

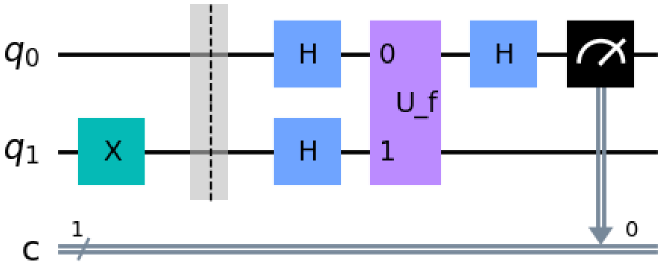
1. Allocate qubits (Flip the 2nd qubit with X-gate): $|0\rangle|1\rangle$
2. Apply Hadamard gates to create superposition: $\rightarrow \left[\frac{|0\rangle+|1\rangle}{\sqrt{2}}\right]\left[\frac{|0\rangle-|1\rangle}{\sqrt{2}}\right]$
3. Apply our quantum oracle gate: $\rightarrow \left[\frac{(-1)^{f(0)}|0\rangle+(-1)^{f(1)}|1\rangle}{\sqrt{2}}\right]\left[\frac{|0\rangle-|1\rangle}{\sqrt{2}}\right]$
4. Apply Hadamard gate on 1st qubit: $\rightarrow (-1)^{f(0)}\left[\frac{(1+(-1)^{f(0)\oplus f(1)})|0\rangle+(1-(-1)^{f(0)\oplus f(1)})|1\rangle}{2}\right]\left[\frac{|0\rangle-|1\rangle}{\sqrt{2}}\right]$



Deutsch Algorithm

- Deutsch algorithm implementation:

1. Allocate qubits (Flip the 2nd qubit with X-gate): $|0\rangle|1\rangle$
2. Apply Hadamard gates to create superposition: $\rightarrow \left[\frac{|0\rangle+|1\rangle}{\sqrt{2}}\right]\left[\frac{|0\rangle-|1\rangle}{\sqrt{2}}\right]$
3. Apply our quantum oracle gate: $\rightarrow \left[\frac{(-1)^{f(0)}|0\rangle+(-1)^{f(1)}|1\rangle}{\sqrt{2}}\right]\left[\frac{|0\rangle-|1\rangle}{\sqrt{2}}\right]$
4. Apply Hadamard gate on 1st qubit: $\rightarrow (-1)^{f(0)}\underbrace{\left[\frac{(1+(-1)^{f(0)\oplus f(1)})|0\rangle+(1-(-1)^{f(0)\oplus f(1)})|1\rangle}{2}\right]}\left[\frac{|0\rangle-|1\rangle}{\sqrt{2}}\right]$



5. Measure:
   - $|0\rangle$ if $f(0) = f(1)$
   - $|1\rangle$ if $f(0) \neq f(1)$

Deutsch Algorithm

- To conclude: the Deutsch algorithm is a demonstration of a quantum computer solving a specific problem faster than what is possible with a classical computer

Deutsch Algorithm

- To conclude: the Deutsch algorithm is a demonstration of a quantum computer solving a specific problem faster than what is possible with a classical computer

- Generalization for n-qubits is called *Deutsch-Jozsa algorithm*

Deutsch Algorithm

- To conclude: the Deutsch algorithm is a demonstration of a quantum computer solving a specific problem faster than what is possible with a classical computer

- Generalization for n-qubits is called *Deutsch-Jozsa algorithm*

- It has no practical use, but it inspired the development of quantum algorithms for practically relevant problems