

# Quantum Computing

## A Gentle Introduction to Grover's Algorithm

Juan Miguel Auñón García

September 13, 2019  
v1.0

# Outline

1. Grover's algorithm
  - 1.1. Motivation & Outline
  - 1.2. Steps
2. Implementation of Grover's algorithm: 2-Qubit States
  - 2.1. Quantum Circuit
  - 2.2. IBM Implementation
3. References

# Outline

## 1. Grover's algorithm

### 1.1. Motivation & Outline

### 1.2. Steps

## 2. Implementation of Grover's algorithm: 2-Qubit States

### 2.1. Quantum Circuit

### 2.2. IBM Implementation

## 3. References

# Outline

## 1. Grover's algorithm

### 1.1. Motivation & Outline

### 1.2. Steps

## 2. Implementation of Grover's algorithm: 2-Qubit States

### 2.1. Quantum Circuit

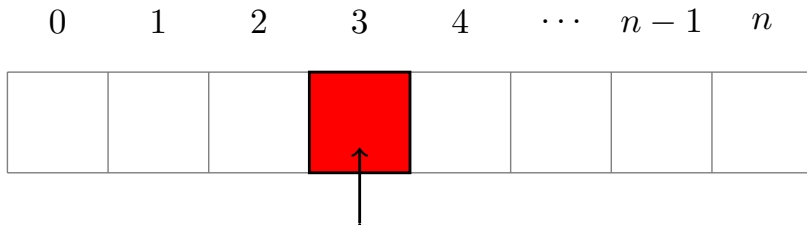
### 2.2. IBM Implementation

## 3. References

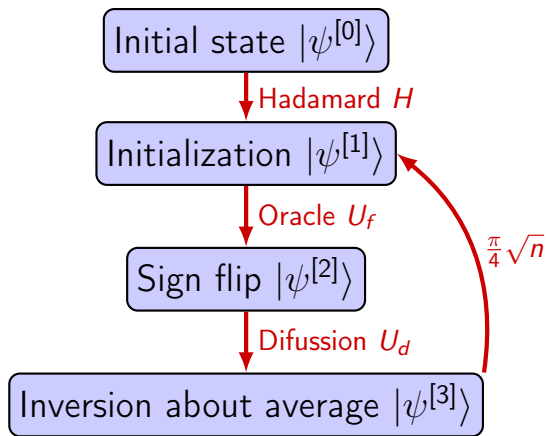
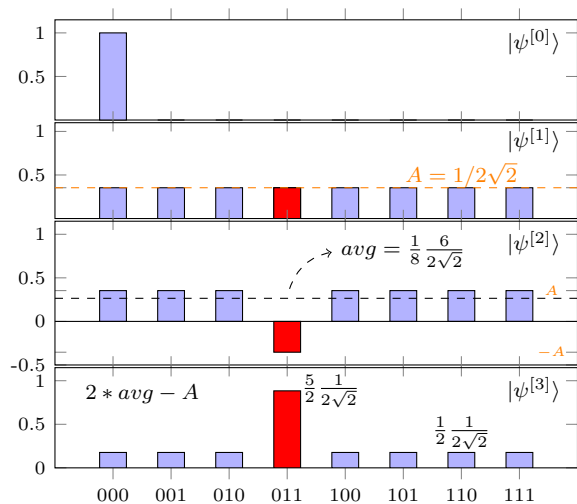
## Grover's algorithm: Motivation

Grover's algorithm performs a search over an unordered set of  $2^n$  items to find the unique element that satisfies some condition

- Classic approach:  $\sum_{i=1}^n \frac{1}{n} i = \frac{n+1}{2} \Rightarrow \mathcal{O}(n)$
- Quantum approach: (...)  $\Rightarrow \mathcal{O}(\sqrt{n})$



# Grover's algorithm: Outline



# Outline

## 1. Grover's algorithm

### 1.1. Motivation & Outline

### 1.2. Steps

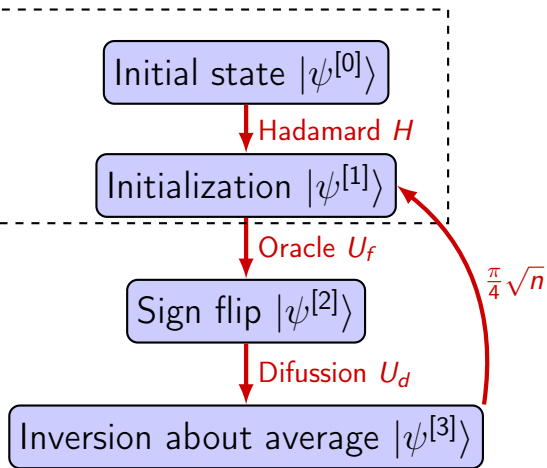
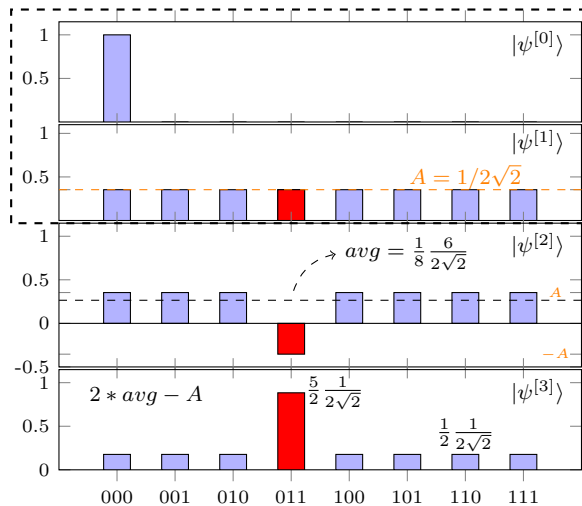
## 2. Implementation of Grover's algorithm: 2-Qubit States

### 2.1. Quantum Circuit

### 2.2. IBM Implementation

## 3. References

# Initialization





# Initialization

We find a method (unitary operator) to have all the states with the same probability (*principle of superposition*).

# Initialization

We find a method (unitary operator) to have all the states with the same probability (*principle of superposition*).

$$\begin{aligned}
 (I^{\otimes n} \otimes X) |0\rangle_{n+1} &= |0\rangle_n \otimes |1\rangle \\
 H^{\otimes(n+1)} [(I^{\otimes n} \otimes X) |0\rangle_{n+1}] &= H^{\otimes n} |0\rangle_n \otimes H |1\rangle \\
 &= \sum_{j \in \{0,1\}^n} \frac{1}{\sqrt{2^n}} |j\rangle_n \otimes \frac{1}{\sqrt{2}} (|0\rangle - |1\rangle) \\
 &= \sum_{j \in \{0,1\}^n} \alpha_j |j\rangle_n \otimes \frac{1}{\sqrt{2}} (|0\rangle - |1\rangle) \\
 &= |\psi^{[1]}\rangle
 \end{aligned}$$

# Initialization

3-qubit example: Set ancillary qubit to  $|1\rangle$

$$\begin{aligned}
 (I^{\otimes 3} \otimes X) |0\rangle_{3+1} &= I^{\otimes 3} |0\rangle_3 \otimes X |0\rangle \\
 &= \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} \otimes \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} \\
 &= |0\rangle_3 \otimes |1\rangle
 \end{aligned}$$

# Initialization

## 3-qubit example: Apply Hadamard gate

$$\begin{aligned}
 |\psi^{[1]}\rangle &= H^{\otimes(3+1)} [(I^{\otimes 3} \otimes X) |0\rangle_{n+1}] \\
 &= H^{\otimes 3} |0\rangle_3 \otimes H |1\rangle \\
 &= \frac{1}{\sqrt{2^3}} \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} \otimes \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix}
 \end{aligned}$$

# Initialization

## 3-qubit example: Apply Hadamard gate

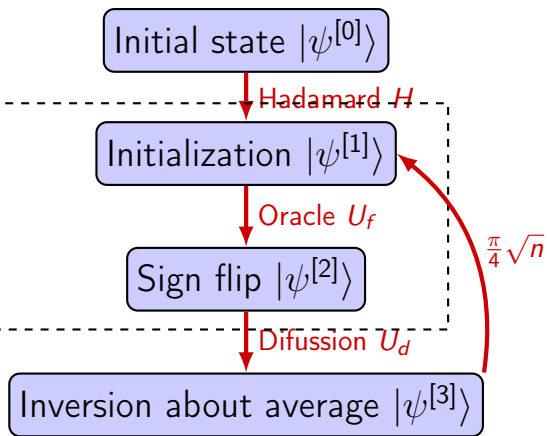
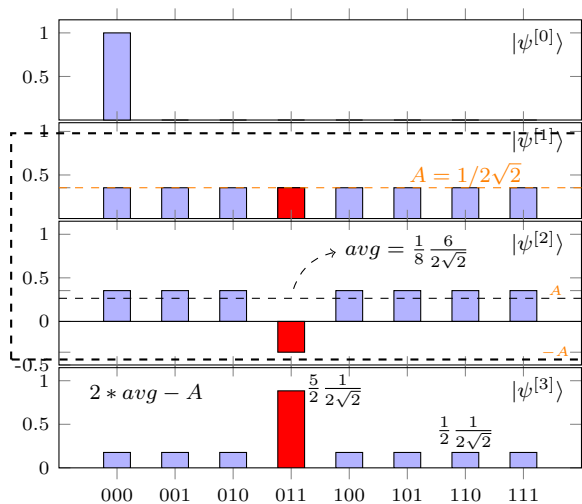
$$\begin{aligned}
 |\psi^{[1]}\rangle &= \frac{1}{\sqrt{2^3}} (11111111)^\dagger \otimes \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ -1 \end{pmatrix} \\
 &= \left[ \frac{1}{2\sqrt{2}} |000\rangle + \frac{1}{2\sqrt{2}} |001\rangle + \frac{1}{2\sqrt{2}} |010\rangle + \frac{1}{2\sqrt{2}} |011\rangle \right. \\
 &\quad \left. + \frac{1}{2\sqrt{2}} |100\rangle + \frac{1}{2\sqrt{2}} |101\rangle + \frac{1}{2\sqrt{2}} |110\rangle + \frac{1}{2\sqrt{2}} |111\rangle \right] \\
 &\quad \otimes \frac{1}{\sqrt{2}} (|0\rangle - |1\rangle)
 \end{aligned}$$

# Initialization

## 3-qubit example: Summary

$$\begin{aligned}
 |\psi^{[1]}\rangle &= H^{\otimes(3+1)} [(I^{\otimes 3} \otimes X) |0\rangle_{n+1}] \\
 &= \left[ \frac{1}{2\sqrt{2}} |000\rangle + \frac{1}{2\sqrt{2}} |001\rangle + \frac{1}{2\sqrt{2}} |010\rangle + \frac{1}{2\sqrt{2}} |011\rangle \right. \\
 &\quad \left. + \frac{1}{2\sqrt{2}} |100\rangle + \frac{1}{2\sqrt{2}} |101\rangle + \frac{1}{2\sqrt{2}} |110\rangle + \frac{1}{2\sqrt{2}} |111\rangle \right] \\
 &\quad \otimes \frac{1}{\sqrt{2}} (|0\rangle - |1\rangle) \\
 &= \sum_{j \in \{0,1\}^n} \frac{1}{\sqrt{2^n}} |j\rangle_n \otimes \frac{1}{\sqrt{2}} (|0\rangle - |1\rangle) = |\psi^{[1]}\rangle
 \end{aligned}$$

## Sign flip



## Sign flip

We find a method (unitary operator) which flip the sign of the state of interest.



## Sign flip

We find a method (unitary operator) which flip the sign of the state of interest.

### Quantum Oracle

It is defined the operator  $U_f$ :

$$U_f : |j\rangle_n \otimes |y\rangle_1 \rightarrow |j\rangle_n \otimes |y \oplus f(j)\rangle_1,$$

where  $\oplus$  is the sum operator in mod 2, and  $f(j) = \begin{cases} 1 & j = I \\ 0 & j \neq I \end{cases}$

A	B	XOR
0	0	0
0	1	1
1	0	1
1	1	0

## Sign flip

We apply  $U_f$  (*Quantum Oracle*) to the previous state  $\psi^{[1]}$

$$\begin{aligned}
 |\psi^{[2]}\rangle &= U_f |\psi^{[1]}\rangle \\
 &= U_f \left( \sum_{j \in \{0,1\}^n} \alpha_j |j\rangle_n \otimes \frac{1}{\sqrt{2}} (|0\rangle - |1\rangle) \right) \\
 &= U_f \left( \alpha_l |l\rangle_n \otimes \frac{1}{\sqrt{2}} (|0\rangle - |1\rangle) + \sum_{j \in \{0,1\}^n; j \neq l} \alpha_j |j\rangle_n \otimes \frac{1}{\sqrt{2}} (|0\rangle - |1\rangle) \right) \\
 &= \left( -\alpha_l |l\rangle_n + \sum_{j \in \{0,1\}^n; j \neq l} \alpha_j |j\rangle_n \right) \otimes \frac{1}{\sqrt{2}} (|0\rangle - |1\rangle)
 \end{aligned}$$

# Sign flip

We apply  $U_f$  (*Quantum Oracle*) to the previous state  $\psi^{[1]}$

$$\begin{aligned}
 |\psi^{[2]}\rangle &= U_f |\psi^{[1]}\rangle \\
 &= U_f \left( \sum \alpha_j |j\rangle_n \otimes \frac{1}{\sqrt{2}} (|0\rangle - |1\rangle) \right) \\
 &= U_f \left( \alpha_I |I\rangle_n \otimes \frac{1}{\sqrt{2}} (|0\rangle - |1\rangle) + \sum_{j \in \{0,1\}^n; j \neq I} \alpha_j |j\rangle_n \otimes \frac{1}{\sqrt{2}} (|0\rangle - |1\rangle) \right) \\
 &= \left( -\alpha_I |I\rangle_n + \sum_{j \in \{0,1\}^n; j \neq I} \alpha_j |j\rangle_n \right) \otimes \frac{1}{\sqrt{2}} (|0\rangle - |1\rangle)
 \end{aligned}$$

*Note: A red arrow points from the term  $\alpha_I |I\rangle_n$  in the second line to the term  $-\alpha_I |I\rangle_n$  in the fourth line, indicating a sign flip.*

# Sign flip

We apply  $U_f$  (Quantum Oracle) to the previous state  $\psi^{[1]}$

$$\begin{aligned}
 |\psi^{[2]}\rangle &= U_f |\psi^{[1]}\rangle \\
 &= U_f \left( \sum_{j \neq l} \alpha_j |j\rangle_n \otimes \frac{1}{\sqrt{2}} (|0\rangle - |1\rangle) \right) \text{sign flip!} \text{Extra qubit!} \\
 &= U_f \left( \alpha_l |l\rangle_n \otimes \frac{1}{\sqrt{2}} (|0\rangle - |1\rangle) + \sum_{j \in \{0,1\}^n; j \neq l} \alpha_j |j\rangle_n \otimes \frac{1}{\sqrt{2}} (|0\rangle - |1\rangle) \right) \\
 &= \left( -\alpha_l |l\rangle_n + \sum_{j \in \{0,1\}^n; j \neq l} \alpha_j |j\rangle_n \right) \otimes \frac{1}{\sqrt{2}} (|0\rangle - |1\rangle)
 \end{aligned}$$

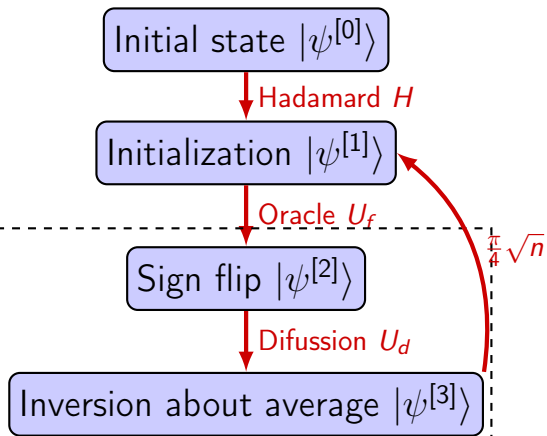
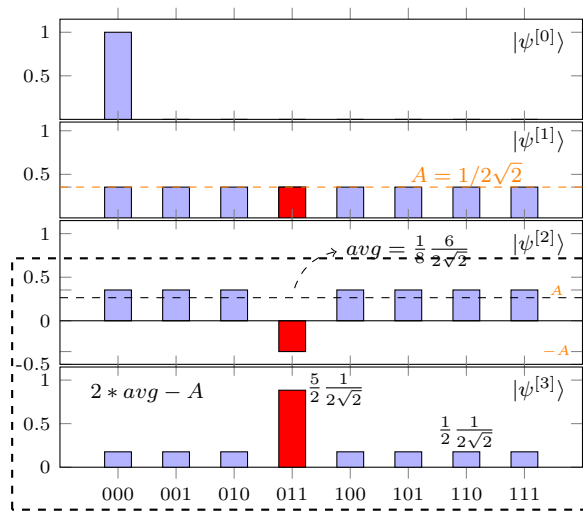
# Sign flip<sup>1</sup>

## 3-qubit example: Quantum Oracle

$$\begin{aligned}
 |\psi^{[2]}\rangle &= U_f |\psi^{[1]}\rangle \\
 &= \left[ \frac{1}{2\sqrt{2}} |000\rangle + \frac{1}{2\sqrt{2}} |001\rangle + \frac{1}{2\sqrt{2}} |010\rangle - \frac{1}{2\sqrt{2}} |011\rangle \right. \\
 &\quad \left. + \frac{1}{2\sqrt{2}} |100\rangle + \frac{1}{2\sqrt{2}} |101\rangle + \frac{1}{2\sqrt{2}} |110\rangle + \frac{1}{2\sqrt{2}} |111\rangle \right] \\
 &\quad \otimes \frac{1}{\sqrt{2}} (|0\rangle - |1\rangle)
 \end{aligned}$$

<sup>1</sup>This slide shows the result of applying  $U_f$  but not how is applied. This is because this step of the algorithm depends on the specific problem.

# Inversion about average



## Inversion about the average

We find a method (unitary operator) to invert the amplitude about the average

## Inversion about the average

We find a method (unitary operator) to invert the amplitude about the average

$$\sum_{j \in \{0,1\}^n} \alpha_j |j\rangle_n \xrightarrow{U_d} \sum_{j \in \{0,1\}^n} \left( 2 \left( \sum_{k \in \{0,1\}^n} \frac{\alpha_k}{2^n} \right) - \alpha_j \right) |j\rangle_n,$$

where  $\sum_{k \in \{0,1\}^n} \frac{\alpha_k}{2^n}$  is the average.

$$\sum_{k \in \{0,1\}^n} \frac{\alpha_k}{2^n} = \frac{1}{2^3} \frac{6}{2\sqrt{2}}$$



# Inversion about the average

## Difussion operator

$$U_d = \begin{pmatrix} \frac{2}{2^n} & \frac{2}{2^n} & \cdots & \frac{2}{2^n} \\ \frac{2}{2^n} & \frac{2}{2^n} & \cdots & \frac{2}{2^n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{2}{2^n} & \frac{2}{2^n} & \cdots & \frac{2}{2^n} \end{pmatrix} - I^{\otimes n} = \cdots = -H^{\otimes n} D H^{\otimes n}, \quad (1)$$

where  $D = \text{diag}(-1, 1, 1, \dots, 1)$

# Inversion about the average

## 3-qubit example: Difussion operator in python

```
>>> import numpy as np
>>> H1 = 1/np.sqrt(2)*np.array([[1,1],[1, -1]]) # Hadamard operator 1 qubit
>>> H2 = np.kron(H1,H1) # Hadamard operator 2 qubit
>>> H3 = np.kron(H2,H1) # Hadamard operator 3 qubit

>>> D = np.eye(8) # Diagonal operator
>>> D[0,0] = -1

>>> Ud = -np.dot(np.dot(H3,D),H3) # Difussion operator
>>> psi_2 = 1/(2*np.sqrt(2))*np.array([1,1,1,-1,1,1,1,1]) # psi_2

>>> psi_3 = np.dot(Ud,psi_2)
>>> print(psi_3)
array([0.176 , 0.176 , 0.176 , 0.883, 0.176, 0.176, 0.176, 0.176])
```

# Inversion about the average

## 3-qubit example: Difussion operator

$$|\psi^{[3]}\rangle = U_d |\psi^{[2]}\rangle$$

$$|\psi^{[3]}\rangle = -\frac{1}{8} \frac{1}{2\sqrt{2}} \begin{pmatrix} 6 & -2 & -2 & -2 & -2 & -2 & -2 & -2 \\ -2 & 6 & -2 & -2 & -2 & -2 & -2 & -2 \\ -2 & -2 & 6 & -2 & -2 & -2 & -2 & -2 \\ -2 & -2 & -2 & 6 & -2 & -2 & -2 & -2 \\ -2 & -2 & -2 & -2 & 6 & -2 & -2 & -2 \\ -2 & -2 & -2 & -2 & -2 & 6 & -2 & -2 \\ -2 & -2 & -2 & -2 & -2 & -2 & 6 & -2 \\ -2 & -2 & -2 & -2 & -2 & -2 & -2 & 6 \end{pmatrix} \begin{pmatrix} 1 \\ 1 \\ 1 \\ -1 \\ 1 \\ 1 \\ 1 \\ 1 \end{pmatrix} = \frac{1}{4\sqrt{2}} \begin{pmatrix} 1 \\ 1 \\ 1 \\ 5 \\ 1 \\ 1 \\ 1 \\ 1 \end{pmatrix}$$

## Measurement & Repetition

For the first iteration we measure

$$\begin{aligned}\alpha_{011} &= \frac{5}{2} \frac{1}{2\sqrt{2}} \longrightarrow \|\alpha_{011}\|^2 \simeq 78,12\% \\ \alpha_j &= \frac{1}{2} \frac{1}{2\sqrt{2}} \longrightarrow \|\alpha_j\|^2 \simeq 3,12\% \quad (j \neq |011\rangle)\end{aligned}$$

The optimal number of repetitions  $R \simeq \frac{\pi}{4} \sqrt{n} \simeq 2.2$ . For the second iteration (Oracle & Diffusion) we have:

$$\begin{aligned}\alpha_{011} &= \frac{11}{4} \frac{1}{2\sqrt{2}} \longrightarrow \|\alpha_{011}\|^2 \simeq 94,5\% \\ \alpha_j &= \frac{-1}{4} \frac{1}{2\sqrt{2}} \longrightarrow \|\alpha_j\|^2 \simeq 0,78\% \quad (j \neq |011\rangle)\end{aligned}$$

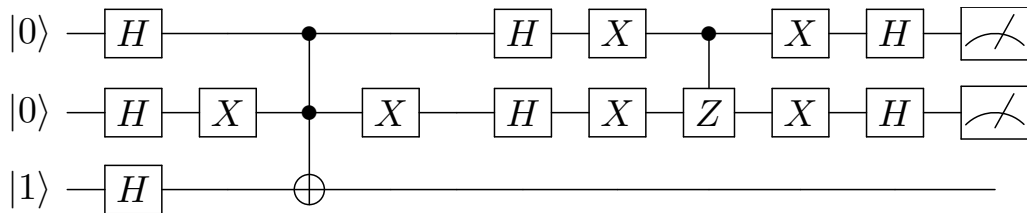
# Outline

1. Grover's algorithm
  - 1.1. Motivation & Outline
  - 1.2. Steps
2. Implementation of Grover's algorithm: 2-Qubit States
  - 2.1. Quantum Circuit
  - 2.2. IBM Implementation
3. References

# Outline

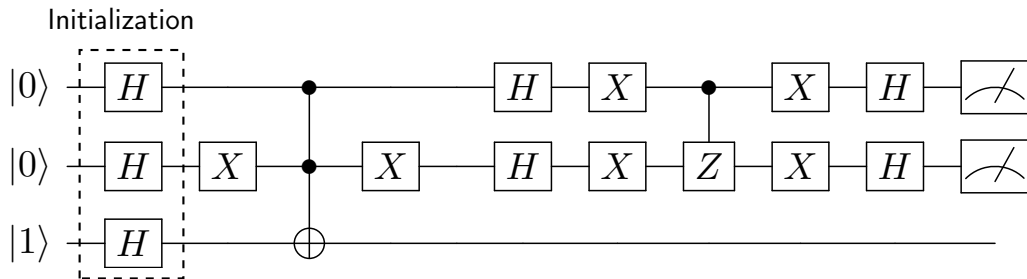
1. Grover's algorithm
  - 1.1. Motivation & Outline
  - 1.2. Steps
2. Implementation of Grover's algorithm: 2-Qubit States
  - 2.1. Quantum Circuit
  - 2.2. IBM Implementation
3. References

## 2-Qubit Quantum Circuit



## 2-Qubit Quantum Circuit

### Initialization





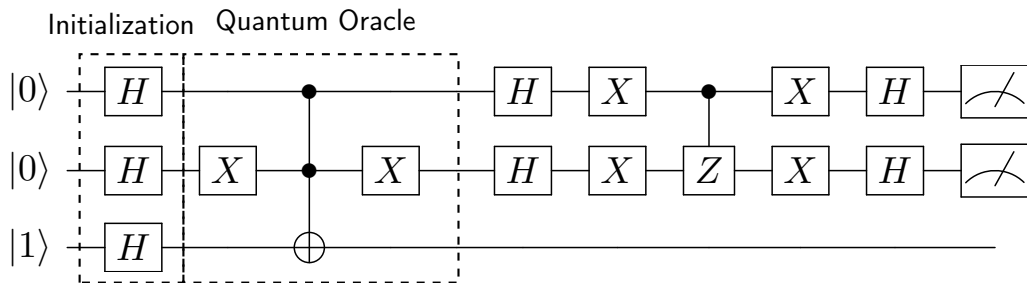
## 2-Qubit Quantum Circuit

### Initialization

$$\begin{aligned}\psi^{[0]} &= |001\rangle \\ \psi^{[1]} &= H^{\otimes 3} \psi^{[0]} \\ &= \frac{1}{\sqrt{8}} (|000\rangle + |010\rangle + |100\rangle + |110\rangle - |001\rangle - |011\rangle - |101\rangle - |111\rangle)\end{aligned}$$

## 2-Qubit Quantum Circuit

### Quantum Oracle



## 2-Qubit Quantum Circuit

### Quantum Oracle

$$U_f = (I \otimes X \otimes I) \cdot T \cdot (I \otimes X \otimes I) = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

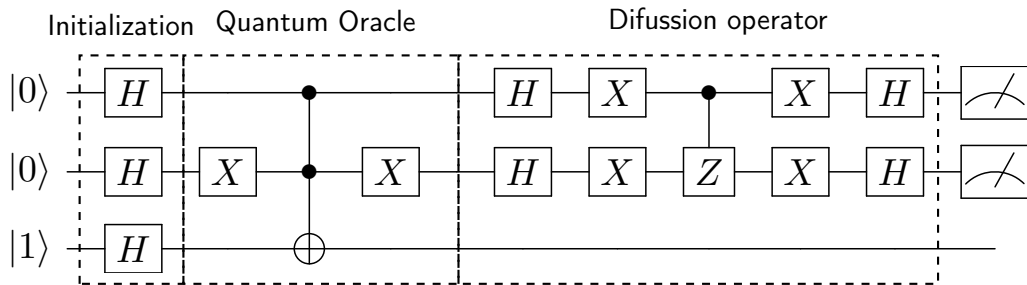
## 2-Qubit Quantum Circuit

### Quantum Oracle

$$\begin{aligned}
 \psi^{[1]} &= \frac{1}{\sqrt{8}} (|000\rangle + |010\rangle + |100\rangle + |110\rangle - |001\rangle - |011\rangle - |101\rangle - |111\rangle) \\
 \psi^{[2]} &= U_f |\psi^{[1]}\rangle \\
 &= \frac{1}{\sqrt{8}} (|000\rangle + |010\rangle + |101\rangle + |110\rangle - |001\rangle - |011\rangle - |100\rangle - |111\rangle) \\
 &= \frac{1}{\sqrt{8}} (|00\rangle + |01\rangle - |10\rangle + |11\rangle) \otimes (|0\rangle - |1\rangle) \\
 &= \frac{1}{2} (|00\rangle + |01\rangle - |10\rangle + |11\rangle) \otimes \frac{1}{\sqrt{2}} (|0\rangle - |1\rangle)
 \end{aligned}$$

## 2-Qubit Quantum Circuit

### Difussion Operator



## 2-Qubit Quantum Circuit

### Diffusion Operator

$$U_d = (H \otimes H) \cdot (X \otimes X) \cdot (CZ) (X \otimes X) \cdot (H \otimes H) = \frac{1}{4} \begin{pmatrix} 2 & -2 & -2 & -2 \\ -2 & 2 & -2 & -2 \\ -2 & -2 & 2 & -2 \\ -2 & -2 & -2 & 2 \end{pmatrix}$$

$$\psi^{[3]} = U_d \psi^{[2]}$$

$$\frac{1}{8} \begin{pmatrix} 2 & -2 & -2 & -2 \\ -2 & 2 & -2 & -2 \\ -2 & -2 & 2 & -2 \\ -2 & -2 & -2 & 2 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} = \frac{1}{8} \begin{pmatrix} 2 & -2 & 2 & -2 \\ -2 & 2 & 2 & -2 \\ -2 & -2 & -2 & -2 \\ -2 & -2 & 2 & 2 \end{pmatrix} \rightarrow \begin{pmatrix} 0 \\ 0 \\ -1 \\ 0 \end{pmatrix} = -|10\rangle$$

# Outline

1. Grover's algorithm
  - 1.1. Motivation & Outline
  - 1.2. Steps
2. Implementation of Grover's algorithm: 2-Qubit States
  - 2.1. Quantum Circuit
  - 2.2. IBM Implementation
3. References

# Implementation

Next slides show the implementation of the algorithm in two different ways:

- 1 Using *qiskit* framework. In this case we have a simulated quantum backend. The reader can copy & run the code in python in order to reproduce the experiment
- 2 Building the gates directly in Quantum IBM computer

In both cases we will observe a histogram with a unique state ( $|10\rangle$ )



# IBM Implementation: Qiskit

## Backend & Quantum Registers

---

```

from qiskit import *
from qiskit import QuantumCircuit
from qiskit.circuit.quantumcircuit import QuantumCircuit
from qiskit.visualization import plot_histogram

backend = BasicAer.get_backend('qasm_simulator')
#backend = BasicAer.get_backend('statevector_simulator')

'''Quantum register
https://quantumcomputing.stackexchange.com/questions/4907/
qiskit-is-there-any-way-to-discard-the-results-of-a-measurement'''

q = QuantumRegister(2, 'q')
a = QuantumRegister(1, 'a')
c = ClassicalRegister(2, 'c')

```

---

# IBM Implementation: Qiskit

## Quantum Circuit

---

```
#Circuit
circ = QuantumCircuit(q,a,c) # type: qiskit.circuit.quantumcircuit.QuantumCircuit

# ===== prepare the states:  $\psi^{\{0\}}$  =====
circ.iden(q[0])
circ.iden(q[1])
circ.x(a[0])

# ===== Initialization:  $\psi^{\{1\}}$  =====
circ.h(q[0])
circ.h(q[1])
circ.h(a[0])
circ.barrier(q)

# ===== Sign flip:  $\psi^{\{2\}}$  =====
circ.x(q[0])
circ.ccx(q[0], q[1], a[0])
circ.x(q[0])

circ.barrier(q)

# ===== Inversion about average:  $\psi^{\{3\}}$  =====
circ.h(q[0])
circ.h(q[1])
circ.x(q[0])
circ.x(q[1])
```

# IBM Implementation: Qiskit

## Measurement

---

```
##### Measurement #####
circ.measure(q,c)

#job
job = execute(circ, backend=backend, shots=1000)
result = job.result()

#circuit
figure = circ.draw(output='mpl')
figure.savefig('../data/images/qiskit-circuit.png')

#histogram
counts = result.get_counts(circ)
print("Total counts:")
print(counts)

figure = plot_histogram(counts)
figure.savefig('../data/images/qiskit-histogram.png')
```

---

# IBM Implementation: Qiskit

## Measurement

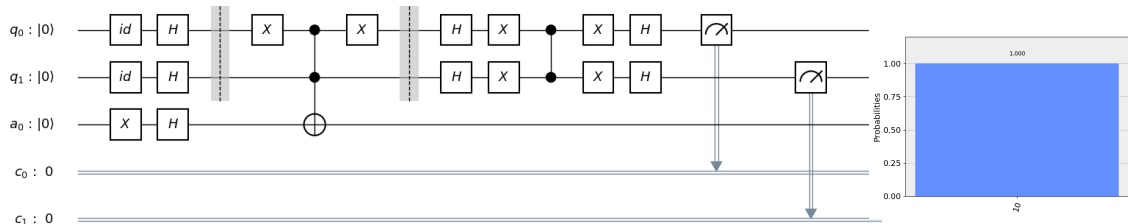


Figure: Histogram

Figure: Quantum Circuit

# IBM Implementation: Quantum Experience

<https://www.research.ibm.com/ibm-q/technology/experience/>

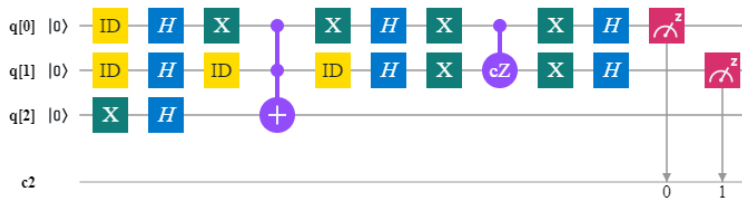


Figure: Quantum Circuit







Figure: Histogram

# Outline

1. Grover's algorithm
  - 1.1. Motivation & Outline
  - 1.2. Steps
2. Implementation of Grover's algorithm: 2-Qubit States
  - 2.1. Quantum Circuit
  - 2.2. IBM Implementation
3. References

# References

-  M. A. Nielsen and I. Chuang, “Quantum computation and quantum information,” 2002.
-  G. Nannicini, “An introduction to quantum computing, without the physics,” *CoRR*, vol. abs/1708.03684, 2017. [Online]. Available: <http://arxiv.org/abs/1708.03684>
-  M. van der Lans, “Quantum algorithms and their implementation on quantum computer simulators,” 2018. [Online]. Available: <https://repository.tudelft.nl/islandora/object/uuid:f05164dd-b853-41a1-a9e9-394cb7a1105e>
-  H. Abraham and I. Y. A. et al., “Qiskit: An open-source framework for quantum computing,” 2019. [Online]. Available: <https://doi.org/10.5281/zenodo.2562111>