Boolean Function Oracles

Practical Quantum Computing using Qiskit and IBMQ

Jothishwaran C.A.

Department of Electronics and Communication Engineering
Indian Institute of Technology Roorkee

September 27, 2020

Outline

Boolean Strings as Basis Vectors

- A single-qubit computational basis vector can be represented as $|x\rangle$, where $x \in \{0, 1\}$.
- If there n such qubits, each with computational basis state $|x_i\rangle$ where $x_i \in \{0, 1\}$ and $i \in [0, n-1]$.
- Computational basis vector for the n-qubit system can simply be defined as $|x_0\rangle \otimes |x_1\rangle \otimes \cdots \otimes |x_{n-2}\rangle \otimes |x_{n-1}\rangle$. There are 2^n such vectors.
- Each of these basis vectors can now be represented as $|x_0 x_1 \ldots x_{n-1}\rangle \equiv |x\rangle$, where $x \in \{0, 1\}^n$.
- This notation shall be adopted for defining multi-qubit basis vectors for all further discussions.

The X_n and H_n gates

- If the X gate is applied to each of the n qubits of the system. The combined operator can be represented as:

$$\underbrace{X \otimes X \otimes \cdots \otimes X}_{n\text{-times}} \equiv X^{\otimes n}$$

- When the H gate is applied to each of the n qubits of the system. The combined operator can be represented as:

$$\underbrace{H \otimes H \otimes \cdots \otimes H}_{n\text{-times}} \equiv H^{\otimes n}$$

- When these operators are applied to the n-qubit basis state $|0_n\rangle$ the results are as follows:

$$X^{\otimes n} |0_n\rangle = |1_n\rangle$$
$$H^{\otimes n} |0_n\rangle = \frac{1}{\sqrt{2^n}} \sum_{x \in \{0,1\}^n} |x\rangle$$

Toffoli gate: a formal introduction

- The Toffoli gate (CCX) is a three qubit gate and has the following actions on the three-qubit computational basis:

$$
\begin{aligned}
CCX\,|000\rangle &= |000\rangle \;;\; CCX\,|100\rangle = |100\rangle \\
CCX\,|001\rangle &= |001\rangle \;;\; CCX\,|101\rangle = |101\rangle \\
CCX\,|010\rangle &= |010\rangle \;;\; CCX\,|110\rangle = |111\rangle \\
CCX\,|011\rangle &= |011\rangle \;;\; CCX\,|111\rangle = |110\rangle
\end{aligned}
$$

- The matrix form is given as:

$$
CCX = \begin{pmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0
\end{pmatrix}
$$

The Toffoli Circuit

- Combining the circuit conventions defined previously and the definitions of Boolean functions The circuit corresponding and the actions to the Toffoli gate is:

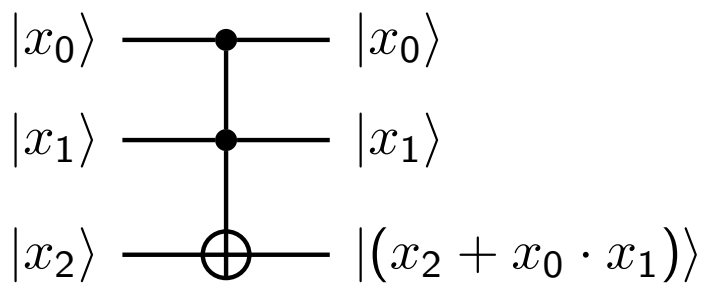


Figure 1: The Toffoli gate

- Here, each state $|x_i\rangle$ is a computational basis vector for the single-qubit state.
- Since there were no phase factors in the actions defined before, the circuit also defines the resultant states for the input state vectors.

A particular setup

- Consider the Toffoli gate with a condition the target qubit is initially fixed in the $|0\rangle$ state:

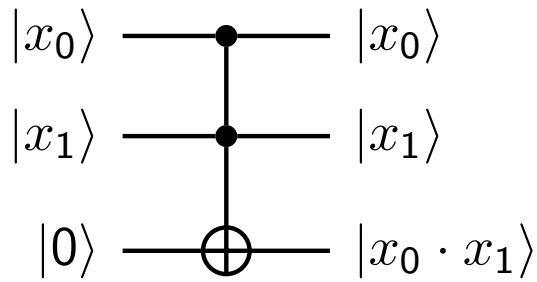


Figure 2: The Toffoli gate

- The effect of the Toffoli gate on these initial states can be summarized as:

$$CCX\,|x_0\rangle\,|x_1\rangle\,|0\rangle \;=\; |x_0\rangle\,|x_1\rangle\,|x_0 \cdot x_1\rangle$$

The AND Oracle

- The *CCX* gate with the initial state as shown before transforms the target qubit from $|0\rangle$ to $|x_0 \cdot x_1\rangle$
- Therefore for any initial control state represented by by x_0x_1 the circuit transforms the input state into the output state $|x_0\rangle\,|x_1\rangle\,|x_0 \cdot x_1\rangle$
- This circuit is referred to as the Quantum AND Oracle: The circuit transforms an input state corresponding to the inputs of a classical AND gate, with a target qubit set to $|0\rangle$ into an output state where the AND operation is performed and stored in the target qubit.
- The two-bit classical AND gate now has an equivalent three-qubit quantum oracle.

Simpler Oracles

- If a two-bit classical gate has a three-qubit quantum oracle, then it maybe possible to define a single bit classical gate with a two-qubit oracle. Consider the following circuit:

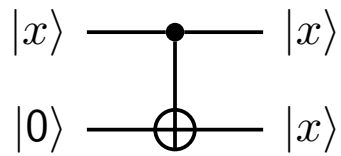


Figure 3: A single bit oracle

- This is the equivalent oracle for the single bit function $F(x) = x$. The other single bit oracle maybe defined as follows:

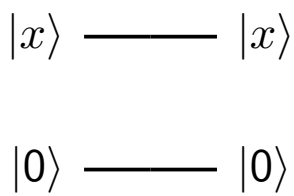


Figure 4: $F(x) = 0$

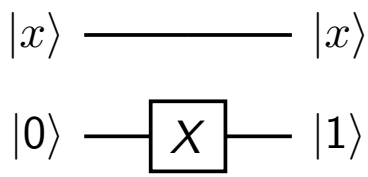


Figure 5: $F(x) = 1$

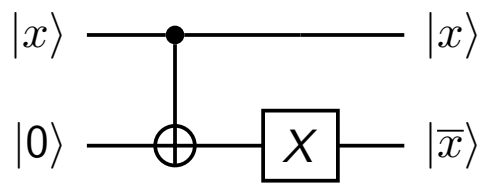


Figure 6: $F(x) = \overline{x}$

Oracles with *CNOT* gates

- As discussed before, the *CNOT* gate acts on a two-qubit basis state $|x_0\rangle\,|x_1\rangle$ as shown below

$$\mathit{CNOT}\,|x_0\rangle\,|x_1\rangle \;=\; |x_0\rangle\,|(x_0 + x_1)\rangle$$

- While this is equivalent to the XOR gate, it is not in the oracle form like the case with the *CCX* gate.
- However, consider the following circuit:

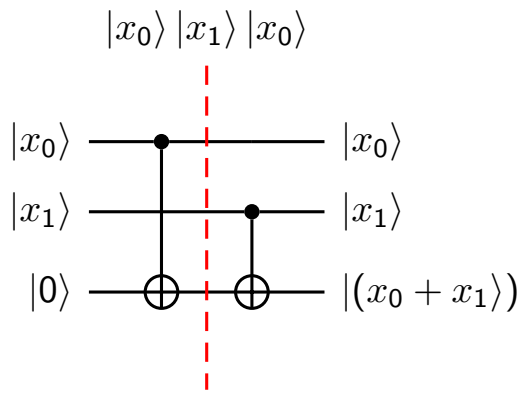


Figure 7: The Toffoli gate

- This circuit is the three-qubit oracle equivalent to the classical XOR gate.

Oracles with *CNOT* gates

- A general n-bit classical Boolean function has an equivalent $(n+1)$-qubit quantum oracle representation U_F that acts as follows.

$$U_F \left|x\right\rangle \left|0\right\rangle = \left|x\right\rangle \left|F(x)\right\rangle$$

here, $x \in \{0,1\}^n$ and $\left|x\right\rangle$ is an element of the n-qubit computational basis.

- The circuit representation of the same is as shown below. It should be noted that the upper bundle of qubit wires represent the n-qubit state.

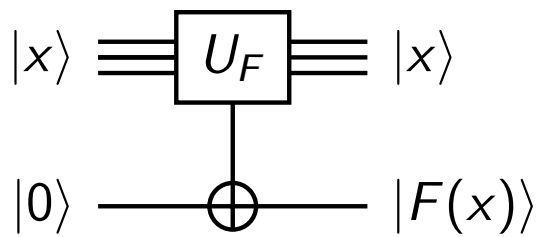


Figure 8: The general quantum Boolean oracle

- It is possible to build any Boolean oracle with the *CNOT*, *CCX* and *X* gates.

Varying the inputs of the Oracles

- Since the oracle is a unitary transformation, they are transformations since the inverse of U_F is simply the adjoint, $U_F^\dagger$.
- If the target qubit of an oracle is fixed to the $|1\rangle$ state, then applying the oracle U_F on such a state yields the following:

$$U_F |x\rangle |1\rangle = |x\rangle |1 + F(x)\rangle$$

- The oracle is also, by its definition a linear transformation and so, if we consider a state $|x_1\rangle + |x_2\rangle$ (normalization is disregarded) where $x_1, x_2 \in \{0,1\}^n$. Then, applying U_F on this state gives:

$$U_F(|x_1\rangle + |x_2\rangle) |0\rangle = |x_1\rangle |F(x_1)\rangle + |x_2\rangle |F(x_2)\rangle$$

  note that the separable input state may not be separable after U_F is applied.
- This is ability of a quantum oracle to evaluate multiple instances of the same function is a result of the linearity of the oracle. This feature is famously know as *Quantum Parallelism* and it has no classical equivalent.

Varying the inputs of the Oracles

- If the target state of the qubit is set to the $|-\rangle$ state, the action of U_F yields the following:

$$\begin{aligned} U_F |x\rangle |-\rangle &= U_F |x\rangle \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) \\ &= |x\rangle \frac{1}{\sqrt{2}} (|0 + F(x)\rangle - |1 + F(x)\rangle) \end{aligned}$$

- The resultant target qubit is in the state $|-\rangle$ when $F(x) = 0$ and $-|-\rangle$ when $F(x) = 1$. The combined result can be written as,

$$U_F |x\rangle |-\rangle = (-1)^{F(x)} |x\rangle |-\rangle$$

- In this case, it can be seen that the state of the target qubit is the same but the output sate gains a phase that corresponds to the value of $F(x)$. This variation of the oracle is referred to as the Phase Oracle implementation of the Boolean function.

The Deutsch Problem

Practical Quantum Computing using Qiskit and IBMQ

Jothishwaran C.A.

Department of Electronics and Communication Engineering
Indian Institute of Technology Roorkee

September 27, 2020

Outline

Boolean function varieties

- Consider an n-bit Boolean function $F(x)$. The input string x can have 2^n possible values. The output $F(x)$ for each of these inputs can be either '0' or '1'.
- A Boolean function $F(x)$ is *constant* if for all input strings, the output either '0' or '1'.
- The constant functions are either $F(x) = 0$ or $F(x) = 1$.
- A Boolean function $F(x)$ is *balanced* if the outputs '0' and '1' occur an equal number of times.
- Since there are 2^n different inputs for $F(x)$, in the balanced case the output '0' or '1' occurs $2^n/2 = 2^{n-1}$ times.
- The two-bit XOR operation and the single bit NOT operation are balanced functions.
- It should be noted that there can be Boolean functions that are neither constant nor balanced, e.g. the two-bit AND gate.

Classical Boolean Oracles

- It is possible to construct a classical versions a Boolean function oracle too.
- It is also possible to design them in a way that anyone using the oracle will only be able to get the value of the function $F(x)$ for the particular input x.
- Such an input is known as a *black box* implementation of the oracle and it does not reveal the form of the function.
- Simple example for black box oracle is the output file generated by a computer program or a sealed circuit implementation of the function.

The Deutsch problem: Statement

- The aim of the problem is to find out if a given black box oracle represents a constant function or a balanced function.
- It is also assured that the given oracle definitely represents either constant or balanced functions and not one correspond to any other variety.
- It is enough to know what variety the oracle represents, the form of the function need not be found.
- In a realistic scenario, this problem also applies to Boolean functions over a large number of variables (n), where knowing the form might still not enable us to decide if a function is constant or balanced.

The Deutsch problem: Classical Approach

- The nature of the oracle can be found out by passing different input values to the oracle and observing the output. This process is called querying.
- If the querying results in a varying output even once, then it maybe concluded that the function is balanced.
- However one has to perform further queries to ensure that the function is constant.
- If the oracle is representing a balanced function, then it will give the output 0 (or 1) for a total of 2^{n-1} times for different input strings.
- Therefore if the oracle gives the output 0 (or 1) for even one time, then the function is constant.
- By querying the oracle for a maximum of $2^{n-1}+1$ times, it is possible to determine if the oracle is constant or balanced.
- This means that the classical solution to this problem has an exponential time complexity.

The single variable problem

- Consider the Deutsch problem single variable Boolean functions, the oracle represents one of the four Boolean functions shown below.

$$F_0(x) = 0 \;\; ; \;\; F_1(x) = 1$$
$$\& \;\; F_2(x) = x \;\; ; \;\; F_3(x) = \bar{x} \;\; :$$

- The classical solution to this problem requires $2^{1-1} + 1 = 2$ queries of the classical oracle.
- While this is a simple enough task in reality it should be noted that, single variable functions have only 2 possible inputs and so the oracle is completely queried.

A quantum solution

- Consider the following quantum circuit containing an unknown one variable Boolean function oracle

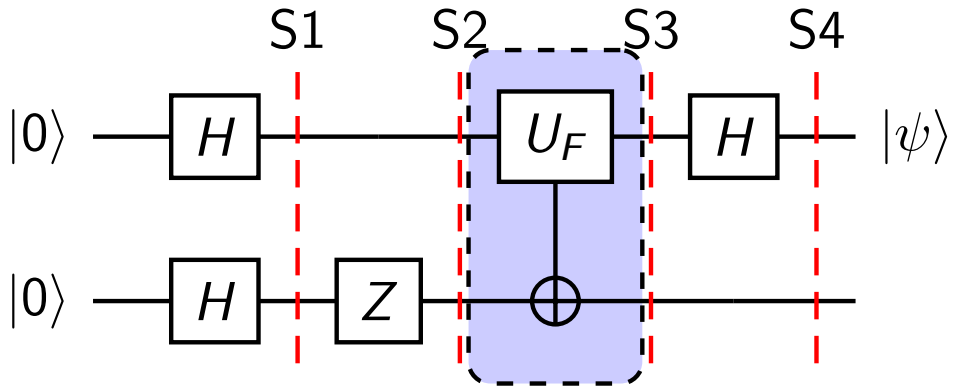


Figure 1: Quantum circuit with the oracle highlighted

- This circuit can be used to solve the single variable Deutsch problem. The state of the two qubits at every stage of the circuit is given as follows (The big endian notation is adopted).

The Deutsch Algorithm: Stage 1

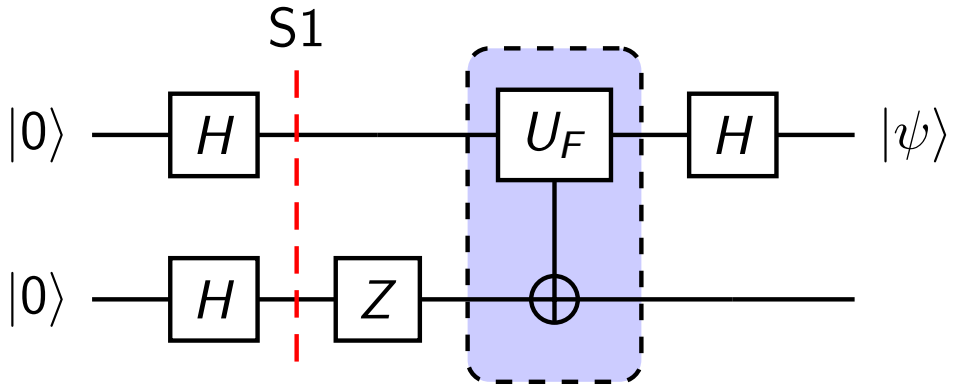


Figure 2: Circuit for the Deutsch Algorithm

- At S1, the transformation $H \otimes H$ is applied to the initial state $|0\rangle\,|0\rangle$:

$$|0\rangle\,|0\rangle \xrightarrow{H\otimes H} |+\rangle\,|+\rangle$$

The Deutsch Algorithm: Stage 2

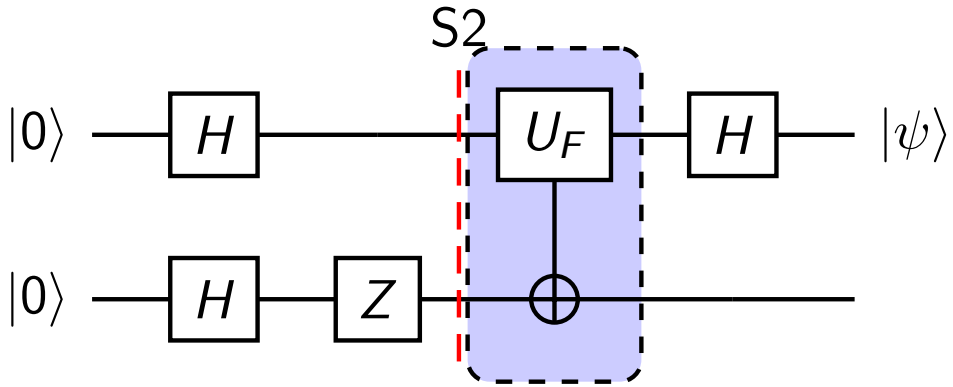


Figure 3: Circuit for the Deutsch Algorithm

- At S2, the transformation $I \otimes Z$ is applied to the state $|+\rangle |+\rangle$:

$$|+\rangle |+\rangle \xrightarrow{I \otimes Z} |+\rangle |-\rangle$$

- Since the target qubit is in the $|-\rangle$ state, we use the phase oracle implementation

The Deutsch Algorithm: Stage 3

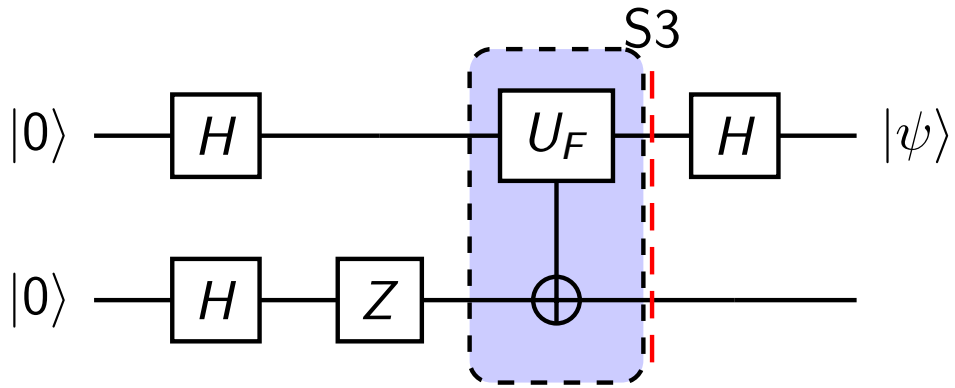


Figure 4: Circuit for the Deutsch Algorithm

- At S3, the transformation U_F is applied to the state $|+\rangle|-\rangle$ this can be expanded using the actions defined previously:

$$\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)|-\rangle \xrightarrow{U_F} \frac{1}{\sqrt{2}}\left((-1)^{F(0)}|0\rangle + (-1)^{F(1)}|1\rangle\right)|-\rangle$$

- If $F(0) = F(1)$, then the phase factor can be neglected as a global phase. However, if $F(0) \neq F(1)$ then there is a relative sign difference.

The Deutsch Algorithm: Stage 4

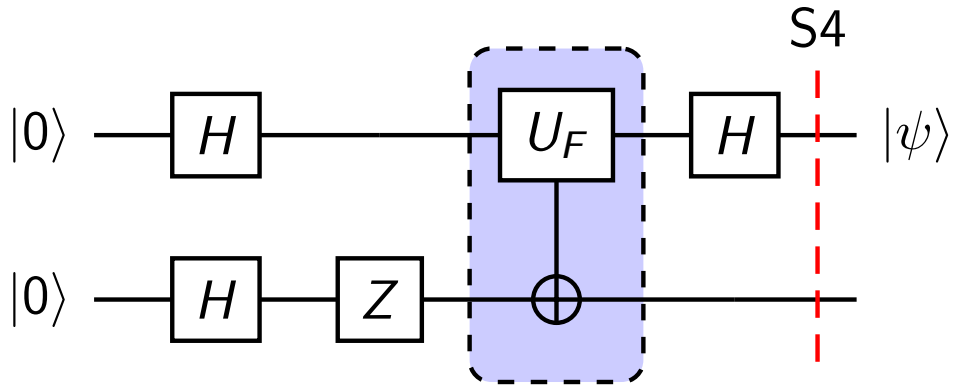


Figure 5: Circuit for the Deutsch Algorithm

- At S4, the transformation $H \otimes I$ is applied. If $F(0) = F(1)$, then the input state is $|+\rangle\,|-\rangle$ and the result is:

$$|+\rangle\,|-\rangle \xrightarrow{H\otimes I} |0\rangle\,|-\rangle$$

If $F(0) \neq F(1)$, then the input state is $|-\rangle\,|-\rangle$

$$|-\rangle\,|-\rangle \xrightarrow{H\otimes I} |1\rangle\,|-\rangle$$

- If $F(0) = F(1)$, then the function is a constant and the state $|\psi\rangle = |0\rangle$. If $F(0) \neq F(1)$ the function is balanced and the state $|\psi\rangle = |1\rangle$. Therefore a measurement of the first qubit in the computational basis will reveal the variety of the function.

Quantum Algorithms on Qiskit

Tuesday, September 8, 2020 9:10 AM

$f_0, f_3 \rightarrow$ constant

$f_1, f_2 \rightarrow$ balanced

1-bit Boolean functions

f_0

x	f(x)
0	0
1	0

f_1

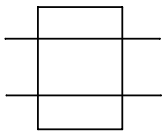
x	f(x)
0	0
1	1

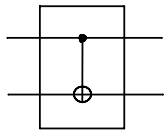
f_2

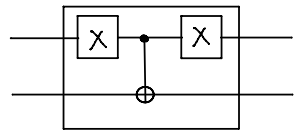
x	f(x)
0	1
1	0

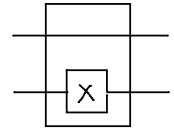
f_3

x	f(x)
0	1
1	1



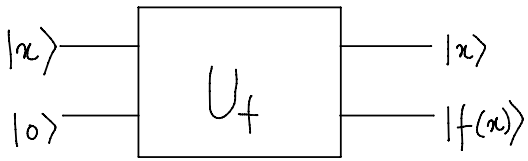


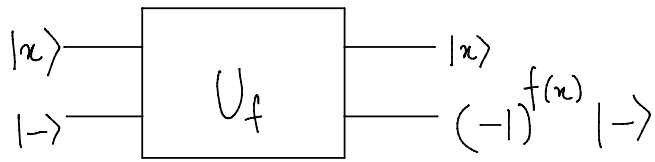




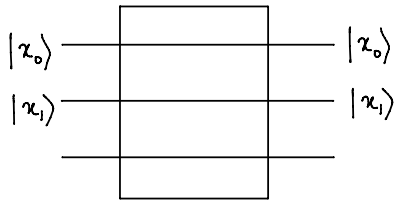
Bit Oracle

Phase Oracle





2-bit Boolean functions



Constant

x_1	x_0	f(x)
0	0	0
0	1	0
1	0	0
1	1	0

Balanced

x_1	x_0	f(x)
0	0	0
0	1	1
1	0	0
1	1	1