Quantum Computing Assignment 2 - Group 18

Rallabhandi, Anand Krishna   Mustafa, Syed Husain
, Mohammed Kamran

November 24, 2020

Exercise 2.1

(Basic quantum circuits)

(a) For a 2 qubit state $|\phi\rangle$ the following are valid transitions for a Swap Gate:

$$|0,0\rangle \mapsto |0,0\rangle$$

$$|0,1\rangle \mapsto |1,0\rangle$$

$$|1,0\rangle \mapsto |0,1\rangle$$

$$|1,1\rangle \mapsto |1,1\rangle$$

For Unitary Matrix $U\epsilon\ \mathbb{C}^{2n\times 2n}$, where $U(|a,b\rangle) \mapsto (|b,a\rangle$ implies U is given by

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}_{4\times 4}$$

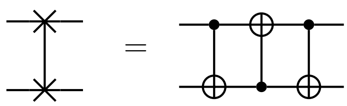
It can be shown that the Swap Gate is equivalent to 3 Controlled NOT Gates,

$$|a,b\rangle \mapsto |a,a\oplus b\rangle\ ,\ \ here\ a\ is\ control\ and\ b\ is\ target$$

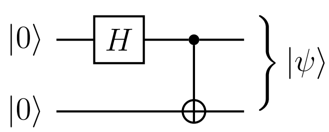
$$|a\oplus(a\oplus b), a\oplus b\rangle\ ,\ \ here\ a\oplus b\ is\ control\ and\ a\ is\ target$$

$$|b,(a\oplus b)\oplus b\rangle\ ,\ \ here\ b\ is\ control\ and\ a\oplus b\ is\ target$$

$$|b,a\rangle\ ,\ \ \text{the result is the swap of}\ \ |a,b\rangle$$

Therefore the circuits are equivalent 

(b)



The Hadamard Gate transforms the top qubit into a superposition state, this then acts as a control input to the CNOT Gate. The target get's inverted only when control is 1.

In general the outputs of the given circuit, ie; Hadamard Gate followed by CNOT with a 2 qubit input, are known as **Bell States**. They are given by

$$|\beta\rangle_{xy} = \frac{|0, y\rangle + (-1)^x |1, \overline{y}\rangle}{\sqrt{2}}$$

Thus for $|00\rangle$,

$$\beta_{00} = \frac{|0, 0\rangle + (-1)^0 |1, \overline{0}\rangle}{\sqrt{2}} = \frac{1}{\sqrt{2}}\Big(|00\rangle + |11\rangle \Big)$$

Alternatively, this can be shown as follows:

$$H|0\rangle = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}_{2\times 2} \begin{pmatrix} 1 \\ 0 \end{pmatrix}_{2\times 1} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 1 \end{pmatrix}_{2\times 1} = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$$

$$\Big(\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)\Big)|0\rangle \xrightarrow{\text{CNOT}} \frac{1}{\sqrt{2}}\Big(|00\rangle + |11\rangle \Big)$$

(c) We know that

$$|0\rangle \xrightarrow{\text{H}} \frac{1}{\sqrt{2}}\Big(|0\rangle + |1\rangle \Big) \quad \textit{and} \quad |1\rangle \xrightarrow{\text{H}} \frac{1}{\sqrt{2}}\Big(|0\rangle - |1\rangle \Big) \qquad \textbf{(A)}$$

For the Pauli-Z Unitary Operator

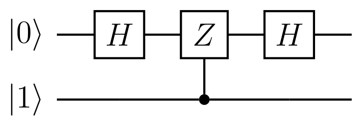
$$|0\rangle \xrightarrow{\text{Z}} |0\rangle \quad \textit{and} \quad |1\rangle \xrightarrow{\text{Z}} -|1\rangle \qquad \textbf{(B)}$$

Using the results from **(A)** & **(B)**

Consider $|\psi\rangle = |10\rangle$ where 1 is the control qubit and 0 is the target qubit, our desired output is $|\psi\rangle = |11\rangle$. We perform the following operations on the target qubit with Controlled-Z Gate switched on for control set to 1.

$$|0\rangle \xrightarrow{\text{H}} \frac{1}{\sqrt{2}}\Big(|0\rangle{+}|1\rangle \Big) \xrightarrow{\text{Z}} \frac{1}{\sqrt{2}}\Big(|0\rangle{-}|1\rangle \Big) \xrightarrow{\text{H}} \frac{1}{\sqrt{2}}\left(\Big(\frac{1}{\sqrt{2}}\Big(|0\rangle{+}|1\rangle \Big) - \Big(\frac{1}{\sqrt{2}}\Big(|0\rangle{-}|1\rangle \Big)\right) \rightarrow |1\rangle$$

The above operations can be realized in the below Quantum Circuit



Exercise 2.2

(IBM Q and Qiskit)

(a)

(b) IBM Quantum Lab Jupyter Notebook Code

```
%matplotlib inline
# Importing standard Qiskit libraries and configuring account
from qiskit import QuantumCircuit, execute, Aer, IBMQ
from qiskit.compiler import transpile, assemble
from qiskit.tools.jupyter import *
from qiskit.visualization import *

from qiskit import QuantumRegister, ClassicalRegister, QuantumCircuit
from numpy import pi

# Loading your IBM Q account(s)
provider = IBMQ.load_account()

qreg_q = QuantumRegister(2, 'q')
creg_c = ClassicalRegister(2, 'c')
circuit = QuantumCircuit(qreg_q, creg_c)

circuit.reset(qreg_q[0])
circuit.reset(qreg_q[1])
circuit.h(qreg_q[0])
circuit.cx(qreg_q[0], qreg_q[1])
circuit.measure(qreg_q[0], creg_c[0])
circuit.measure(qreg_q[1], creg_c[1])

circuit.draw()


simulator = Aer.get_backend('qasm_simulator')

result = execute(circuit, simulator).result()
counts = result.get_counts(circuit)
plot_histogram(counts, title='Results')
```