Quantum Computing for Engineers

Osama Muhammad Raisuddin

June 23, 2023

Contents

This is the Abstract hi

1 Introduction

Quantum computing is an emerging area of research with great potential for engineering and scientific computing applications. Endowed with quantum mechanical properties such as superposition, entanglement and tunneling, quantum computers can enable exponential speedups over classical computers for some problems. The advantage can be realized in both time and energy consumption.

All classical computation can be performed on quantum computers using reversible versions of logic gates. However, this approach is inefficient and required large overheads. To take advantage of quantum computers new algorithms have been developed and are an active area of research. Due to stark differences with classical computation, the engineering community has limited exposure to quantum computing and quantum algorithms.

In this review we aim to provide an introduction to quantum computing with demonstration of concepts and ideas with code. Quantum algorithms relevant to engineers are presented in detail with implementation.

2 Quantum Information

2.1 Quantum States

2.2 Operators

2.3 Measurements

2.4 Composite Systems

2.5 Density Operators

2.6 Born Rule

This section should give an introduction to the state spaces of the qubits, operators, the Kronecker products, and densoty operator representation.

The Born Rule needs to be introduced. The no-cloning theorem can also be introduced here.

2.7 Postulates of Quantum Mechanics

2.7.1 State Space

2.7.2 Evolution

2.7.3 Measurement

2.7.4 Composite Systems

3 Quantum Computation

In this section we abstract away details of quantum information and the underlying physics of a quantum computer and present the computational model for quantum computing. Since all gate-based quantum computers share the same computational model, this is a safe starting point for beginners. Implementation of quantum algorithms does not require knowledge beyond these fundamental elements.

3.1 Qubits and Qudits

A 'qubit', or a quantum bit, is the quantum analogue of a classical bit. It is the elementary information unit in quantum computing. A qubit is a two-dimensional quantum system with dimensions typically labelled '0' and'1' analogous to the states of a classical bit. A qubit is exists in the Hilbert space $\mathcal{H} : \mathbb{C}^2$ and is represented in bra-ket (or Dirac) notation as:

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle \tag{1}$$

where $|\alpha|^2 + |\beta|^2 = 1$ according to the Born rule and $|\psi\rangle \in \mathbb{C}^2$ denotes the quantum state of the qubit. $|0\rangle$ and $|1\rangle$ are the basis states in the $0-1$ basis, corresponding to the basis vectors $|0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$ and $|1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$. α and β are the 'probability amplitudes' of the state $|\psi\rangle$, corresponding to a probabilities of measuring the qubit $|\psi\rangle$ in states $|0\rangle$ and $|1\rangle$ with probailitities $|\alpha|^2$ and $|\beta|^2$ respectively. Note that once a measurement is performed on qubit $|\psi\rangle$ to obtain a measured state of either $|0\rangle$ or $|1\rangle$, the state $|\psi\rangle$ is destroyed and further measurements (without any other subsequent operations) will repeatedly yield the same state that was measured.

The state $|\psi\rangle$ is a 'ket'. The corresponding 'bra' $\langle\psi|$ is the adjoint, or the complex transpose of the 'ket'. As an example, the basis states form the kets $\langle 0| = \begin{pmatrix} 1 & 0 \end{pmatrix}$ and $\langle 1| = \begin{pmatrix} 0 & 1 \end{pmatrix}$.

A qudit is higher-dimensional generalization of of a qubit, s.t. $|\psi\rangle \in \mathbb{C}^d$, $d > 2$.

3.2 Registers of Qubits

A register of qubits is a collection of n qubits, whose combined state is represented as the quantum system $|\psi\rangle \in \mathbb{C}^{2^n}$, corresponding to a tensor product of the individual Hilbert spaces of the qubits. The combined state $|\psi\rangle$ of two individual qubits $|\psi_0\rangle = \alpha_1|0\rangle + \beta_1|1\rangle$ and $|\psi_1\rangle = \alpha_2|0\rangle + \beta_2|1\rangle$ may be represented in Dirac notation as a Kronecker product of the individual qubits with various equivalent notations:

$$\begin{aligned}
|\psi\rangle = |\psi_1\rangle \otimes |\psi_2\rangle = |\psi_1\rangle|\psi_2\rangle = |\psi_1\psi_2\rangle \\
= (\alpha_1|0\rangle + \beta_1|1\rangle)(\alpha_2|0\rangle + \beta_2|1\rangle) \\
= \alpha_1\alpha_2|0\rangle|0\rangle + \alpha_1\beta_2|0\rangle|1\rangle + \alpha_2\beta_1|1\rangle|0\rangle + \beta_1\beta_2|1\rangle|1\rangle \\
= \alpha_1\alpha_2|0\rangle \otimes |0\rangle + \alpha_1\beta_2|0\rangle \otimes |1\rangle + \alpha_2\beta_1|1\rangle \otimes |0\rangle + \beta_1\beta_2|1\rangle \otimes |1\rangle \\
= \alpha_1\alpha_2|00\rangle + \alpha_1\beta_2|01\rangle + \alpha_2\beta_1|10\rangle + \beta_1\beta_2|11\rangle \\
= \alpha_1\alpha_2 \begin{pmatrix}1\\0\end{pmatrix} \otimes \begin{pmatrix}1\\0\end{pmatrix} + \alpha_2\beta_1 \begin{pmatrix}1\\0\end{pmatrix} \otimes \begin{pmatrix}0\\1\end{pmatrix} + \alpha_1\beta_2 \begin{pmatrix}0\\1\end{pmatrix} \otimes \begin{pmatrix}1\\0\end{pmatrix} + \beta_1\beta_2 \begin{pmatrix}0\\1\end{pmatrix} \otimes \begin{pmatrix}0\\1\end{pmatrix} \\
= \alpha_1\alpha_2 \begin{pmatrix}1\\0\\0\\0\end{pmatrix} + \alpha_2\beta_1 \begin{pmatrix}0\\1\\0\\0\end{pmatrix} + \alpha_1\beta_2 \begin{pmatrix}0\\0\\1\\0\end{pmatrix} + \beta_1\beta_2 \begin{pmatrix}0\\0\\0\\1\end{pmatrix} = \begin{pmatrix}\alpha_1\alpha_2\\ \alpha_2\beta_1\\ \alpha_1\beta_2\\ \beta_1\beta_2\end{pmatrix}.
\end{aligned} \tag{2}$$

Additonal qubits will follow the same pattern, resulting in an exponentially large state space for the qubits. Note that the ordering of qubits is arbitrary, and a change in ordering shuffles the representation of the state corresponding the Kronecker product; typically one may choose the most conventient order for their application.

3.3 Gates

In the gate-based quantum computing model operations on qubits are represented as quantum gates, or simply gates. Some of the gates are analogous to classical gate operations. However, some quantum gates may not have a corresponding classical counterpart.

Gates can be conveniently represented as complex matrices, in line with the column vector representation of qubits. The most basic gates are single qubit gates represented as $\mathbb{C}^{2\times 2}$. Some important single-qubit gates are the Pauli gate set $\{I, X, Y, Z\}$ and the Hadamard gate H.

Gate can also act on multiple qubits, in which case they can be represented in $\mathbb{C}^{2^n \times 2^n}$. Multiple qubit gates typically arise as controlled versions of gates, of which $CNOT$ is a commonly used one. The $SWAP$ gate is another common gate which swaps the states between qubits.

All quantum gates are unitary operations, which ensures normalization of quantum states in line with the Born rule. Consequantly, all quantum gates are also reversible operations, with the reverse operation simply being the Hermitian transpoise or conjugate transpose of the quantum gate.

Gates acting on qubits are typically represented as left multiplication. As an example, consider a register of three qubits in the state $|000\rangle$. Applying a

Hadamard gate to the first qubit and a Pauli X gate to the last qubit (from the left) is represented as:

$$(H \otimes I \otimes X)|000\rangle = (H \otimes I \otimes I)|001\rangle = \frac{1}{\sqrt{2}}(|001\rangle + |101\rangle) \tag{3}$$

As another example, consider a register of two qubits with a Hadamard gate applied to the first qubit and a CNOT gate applied to the second qubit, controlled by the first qubit:

$$(|00\rangle\langle 00| + |01\rangle\langle 01| + |10\rangle\langle 11| + |11\rangle\langle 10|)(H \otimes I)|00\rangle = |00\rangle + |11\rangle \tag{4}$$

Note how the order of operations progresses from right to left. It is common practice not to include the I gate when the qubits on which the operation is applied is implied.

Similar to classical Boolean logic, quantum gates can also form a universal gate set. This implies that any arbitrary quantum gate can be approximated using a universal gate set. The Solovay-Kitaev theorem is a central theorem in quantum computing which shows that the approximation error using a universal gate set scales as $O(\log^c(\frac{1}{\epsilon}))$ for a single-qubit gate where $c \approx 2$ and $O(m \log^c(\frac{m}{\epsilon}))$ for a a set of m $CNOT$s and single-qubit unitaries. These correspond to a polylogarithmic increase in the original number of gates.

Underlying hardware implementations can have a variety of gate sets. Algorithms are typically agnostic to the hardware implementation, since the gates are converted to the target hardware's gate set in a process called transpilation.

3.4 Measurement

A quantum state is defined using probability amplitudes. In order to read a state a series of measurements of the state need to be performed, and the measurement statistics will correspond to the probability amplitudes of the quantum state.

In gate-based quantum computing projective measurements are typically used and are represented by a 'meter' symbol. The measurement operation is a projection onto the $|0\rangle$ $|1\rangle$ basis. As an example, consider a two-qubit system in the state $\frac{1}{2}(|00\rangle + |01\rangle + |10\rangle + |11\rangle)$, with a measurement being performed on the first qubit. If the first qubit is measured in the state $|0\rangle$, the final state is:

$$(|0\rangle\langle 0| \otimes I)(\frac{1}{2}(|00\rangle + |01\rangle + |10\rangle + |11\rangle)) = \frac{1}{\sqrt{2}}(|00\rangle + |01\rangle) \tag{5}$$

Similarly, if the first qubit is measured in the state $|1\rangle$ the final state is:

$$(|1\rangle\langle 1| \otimes I)(\frac{1}{2}(|00\rangle + |01\rangle + |10\rangle + |11\rangle)) = \frac{1}{\sqrt{2}}(|10\rangle + |11\rangle) \tag{6}$$

Measurements may be performed to either read out an entire quantum register or as a flag to indicate successful operation by measuring ancilla qubits.

Note that measurements will provide the squared moduli (amplitudes) of the complex numbers defining the probability amplitude of a quantum state. In order to recover additonal information, like the arguments (phases), a process known as quantum state tomography [ref] is performed. Quantum states also have an overall phase. The overall phase is not measurable; only relative phase between two states is measurable.

3.5 Circuits

Working with algebraic forms of algorithms can be unwieldy and difficult to visualize. Quantum circuits are a convenient representation of qubits and the operation sequence.

As an example, consider a register of three qubits $|ABC\rangle$, with an H gate applied to the A and an X gate applied to C, followed by a $CNOT$ gate controlled by the A applied to B, and finally measurements on all the qubits. These operations can conveniently be represented in circuit form as:

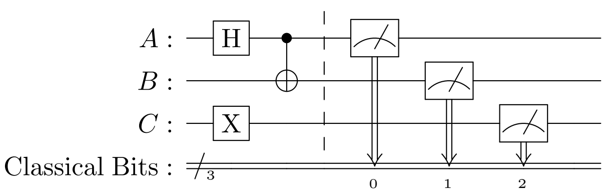

Figure 1: Circuit with gates and measurements.

Note that the operations are ordered from left to right, unlike operations ordered right to left in algebraic notation. A controlled operation can be applied conditioned on the control qubit being either in the state 1 or 0. This in indicated in a quantum circuit with a filled or empty circle respectively, as shown in Figure 2.

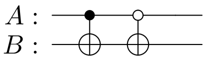

Figure 2: $CNOT$ gates conditioned on A in states 0 and 1.

Quantum registers can be ordered using either little-endian or big-endian notation. Little-endian ordering corresponds to the right-most qubit in algeraic notation indexed as 0 and big-endian corresponds to the left-most qubit indexed as 0. Qiskit uses little-endian ordering.

As an example, we repeat the example in Figure 1 but with $|ABC\rangle$ being denoted as $|\psi\rangle$. Consistent with Qiskit, in little-endian notation $|A\rangle$, $|B\rangle$, and $|C\rangle$ correspond to $|\psi_2\rangle$, $|\psi_1\rangle$, and $|\psi_0\rangle$. We provide an example code: starting with big-endian notation and converting to little-endian at the end.

```
#!/usr/bin/python3

import qiskit

# Create a register of 3 qubits
myQRegister = qiskit.QuantumRegister(3, '\psi')

# Create a register of 3 classical bits
myCRegister = qiskit.ClassicalRegister(3,'Classical Bits')

# Create a quantum circuit with using myRegister
myCircuit = qiskit.QuantumCircuit(myQRegister, myCRegister)

# Start working in big-endian order

# Add the gates in order of operation
myCircuit.h(0)
myCircuit.x(2)
myCircuit.cnot(0,1,ctrl_state='1')

# Insert a barrier to keep circuit organized
myCircuit.barrier()

# Measure all the qubits in myQRegister and store state in myCRegister
myCircuit.measure(myQRegister,myCRegister)

# Convert from big-endian to little-endian by reversing bits
myCircuit = myCircuit.reverse_bits()

# Draw the circuit
myCircuit.draw('latex')
```

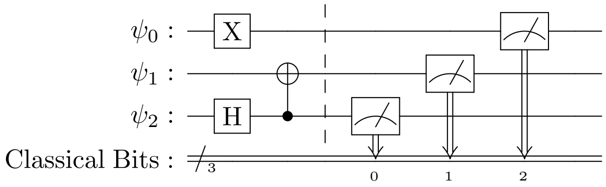

Figure 3: Output of Example code

3.6 Superposition

Superposition is a central property of quantum computing. A quantum state can exist in a superposition of states. As an example, a single qubit $|\psi\rangle$ can exist in a superposition of $|0\rangle$ and $|1\rangle$. A register of n qubits can exist in an exponentially large superposition of $N = 2^n$ states. The Hadamard gate puts qubits in superposition. As an example, we provide the following code to put qubits in uniform superposition and measure their states:

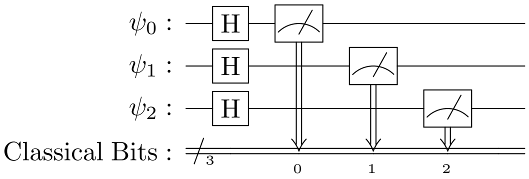

Figure 4: Circuit for uniform superposition.

```
#!/usr/bin/python3

import qiskit
from qiskit import Aer
from qiskit.tools.visualization import plot_histogram

# Create a register of 3 qubits
myQRegister = qiskit.QuantumRegister(3, '\psi')

# Create a register of 3 classical bits
myCRegister = qiskit.ClassicalRegister(3,'Classical␣Bits')

# Create a quantum circuit with using myRegister
myCircuit = qiskit.QuantumCircuit(myQRegister, myCRegister)

# Hadamard gates on al qubits
myCircuit.h(myQRegister)

# Measure all the qubits in myQRegister and store state in myCRegister
myCircuit.measure(myQRegister,myCRegister)


# Simulate the circuit
mySimulator = Aer.get_backend('aer_simulator')
result = mySimulator.run(myCircuit,shots=5120).result()

# Plot a bar chart of all the results
plot_histogram(result.get_counts(), title='Uniform␣Superposition')
```

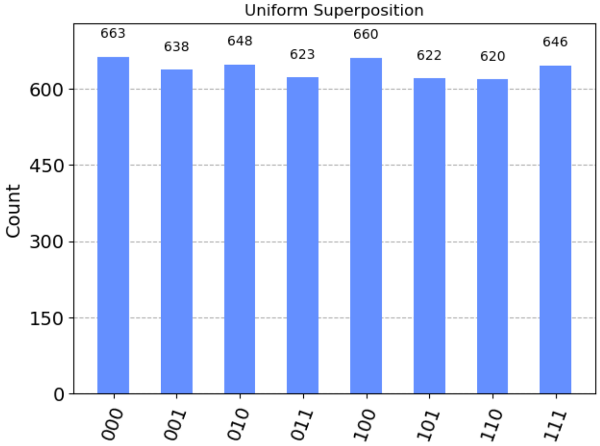

Figure 5: Output of Uniform Superposition code

3.7 Entanglement

Entanglement is another property unique to quantum computation. Entangled qubits have a correlated state.

Note that entangled qubits cannot be seperated into a Kronecker product, at least not without an auxiliary variable.

3.8 Reversible Computation

This is the most detailed section. Over here we abstract away all the details about physics etc. and will only work with qubits and circuits.

We will introduce single and multiple qubits, and gate for single and multiple qubits. Controlled gates will also be introduced.

Then we talk about measurements. We will use projective measurements in the standard basis.

The ideas of superposition, entanglement, (and interference?) will be introduced. We will also mention that all operations are reversible, and reversible versions of classical operations can be done in reversible circuits, but this has lage overheads.

Talk about data representation, amplitude encoding is the important one.

Talk about the big limitations.

Inroduce some

4 Quantum Algorithms for Engineering