PUP IEC Quantum Computing Outline

**Unit 4: Introduction to Quantum Computing**

August 10, 2023

Around August 20, nadagdagan yung content ng outline ko

By Ross Mariano

# Quantum Computation

Tungkol saan to lmfao. Hindi ba ito na rin yung pangalan ng chapter/unit ??

Ang pwede kong ilagay dito ay about quantum information and introducing the concept of qubits. Pero nasa Unit 3 na siya (by Ren)

Kung kukuha ako ng content from the book references (bukod sa qubits)

From McMahon

Chapter 1 – Information content in a signal, entropy and Shannon’s information theory

From Nielsen and Chuang

Chapter 1.1 – Global perspectives

Chapter 1.6 – Quantum information

Some quick recap about qubits

McMahon Chapter 2

Nielsen Ch 1.2

Wong Ch 2

# Quantum Gates

**Definition**

Quantum logic gates or quantum gates are the quantum computing equivalents of the classical logic gates in classical computing. Similar to the classical logic gates, quantum gates are used to manipulate qubits. These are the components used for creating quantum circuits.

**Some math/properties**

Mathematically speaking, a quantum gate *transforms* the quantum state of a qubit into another quantum state. For general quantum gates, we use U.

Example

### Some properties

A quantum gate must be linear, i.e. we can distribute the operation

And must still follow that the total probability is 1

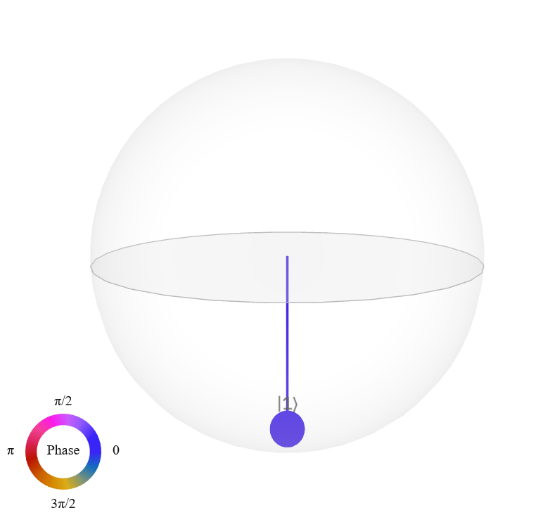
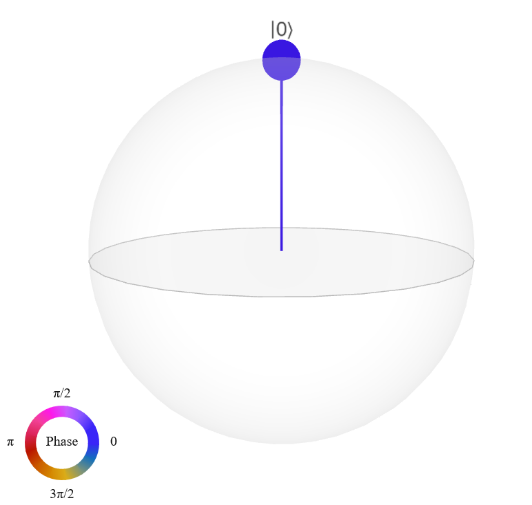
Quantum Logic gates are reversible.

### One qubit quantum gates

There are the “basic” quantum gates that we use in quantum computing.

* Identity gate. This gate preserves the quantum state. We can also say that it does “nothing” to the quantum state. Denoted by I.
* Pauli X gate, X gate, or NOT gate. This gate performs a *bit flip* on the quantum state.

Bloch sphere representation of effect of X gate. This is a 180deg rotation along the x-axis.



* Pauli Y gate, or Y gate.

This is the first gate we’ve seen that has a coefficient. Let us check if this is a valid quantum gate. We check by calculating if the total probability is 1.

Probability is

Y gate is a valid quantum gate

Bloch sphere representation. This is a 180deg rotation about the y-axis. Notice that we get a different color representation. This is due to the i coefficient which represents the angle and direction applied.

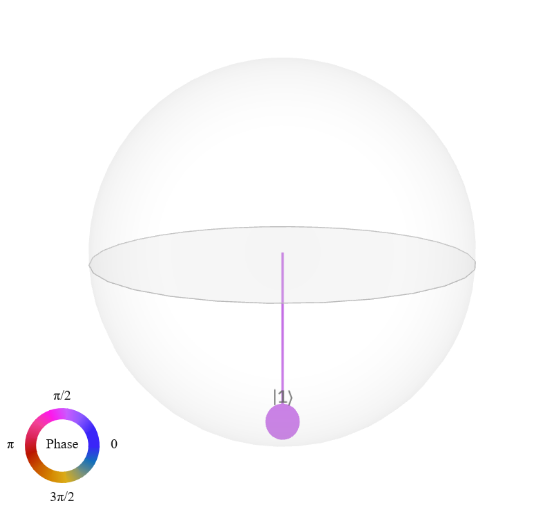
A diagram of a sphere with a blue dot

Description automatically generatedA white sphere with a pink circle and a purple circle with a pink circle

Description automatically generated

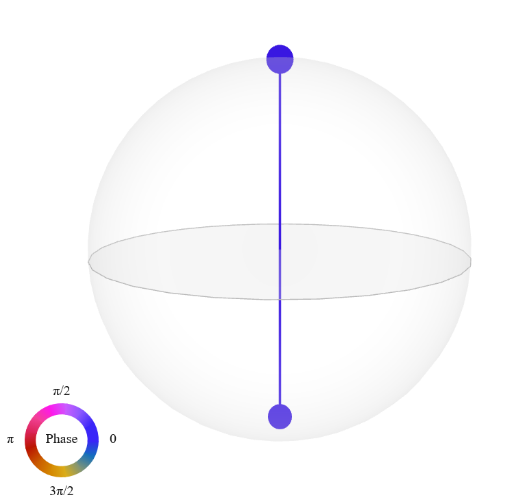
* Pauli Z gate or Z gate. This gate performs a *phase flip* on the quantum state.

A diagram of a sphere with a blue ball

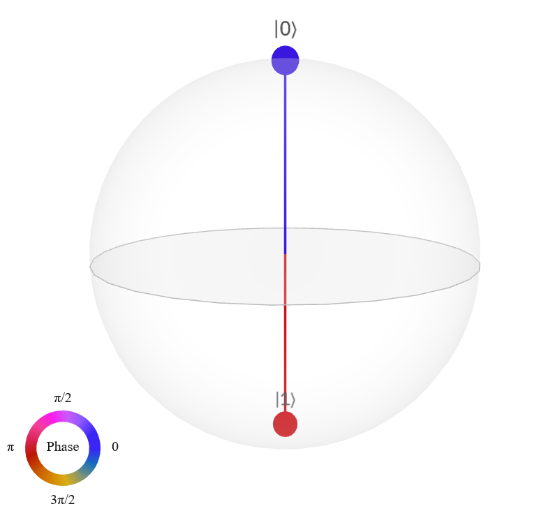
Description automatically generated

* Hadamard gate, or H gate. This gate creates a superposition between states. This is a rotation about the x and z axis by 180 deg.

A diagram of a sphere with a blue dot

Description automatically generated

A diagram of a sphere with a blue ball

Description automatically generated

* We can combine these gates to achieve our desired transformations (more on this sa quantum circuits)

There are other single qubit gates such as the S, T, RX, RY, RZ, U, etc. gates which all manipulate the qubit state by rotating.

### Sample calculation

Ngayong napakita ko na yung basics gates. Let us see lang yung mathematics behind. Essentially, this is just matrix multiplication. Kung marunong ka mag matmul, kaya mo na to.

Recall matrix multiplication.

Let us verify .

Let us check reversibility.

What if we apply many single qubit gates? For example,

Let us determine the result without *doing math*.

We know that , we can reduce to . We also know the result of this, . Let us now verify by calculating.

We calculate the result of many single qubit gates via matrix multiplication. We call operations of gates in succession *gates in series*.

### Multiple qubits

So far we have only examined one qubit. What if we have multiple qubits? Let’s start with 2.

Suppose we have two qubits and . We get the quantum state of both qubits using the *tensor product*.

We will use the notation for multiqubit states, where n is the nth qubit.

For more qubits, we apply the tensor product multiple times.

We call this operation *gates in parallel*.

**Relevant references**

**Wong ch2.6, 3.3, 4.4, 4.6**

# Quantum Circuit Model

Now that we know quantum gates, we can now construct a quantum circuit. A quantum circuit is the quantum equivalent of a classical circuit. This is a sequence of quantum gates being applied to the qubits.

### Circuit diagram

We draw a quantum circuit diagram as follows.

A blue squares with black letters and a black line

Description automatically generated

We denote gates as boxes/squares with their respective labels. X for X gate, Y – Ygate, Z – Zgate, H – Hadamard. The horizontal lines, also called, *quantum wires*, dictate the flow of time. Time flows from left to right. In quantum computing software, qubits are initialized as .

Since time flows from left to right, this means that we apply X to the qubit. Then Z, then H, thus for this quantum circuit, the equation is

Multiple qubits

**A diagram of a number of squares

Description automatically generated with medium confidence**

For multiple qubits, we label where n is the nth qubit. Notice that on q1, under the H gate, there is “nothing”. This implies that there is an identity gate there.

Since we follow the flow of time from left to right, we calculate the gates on the same timestep. We do this via tensor product.

For

### Multi-qubit gates

There are quantum gates that act on multiple qubits *simultaneously*. We introduce the two qubit gates. An example of this is the Controlled-NOT gate, or CNOT gate, or CX gate. This gate acts on two qubits, named the *control qubit* and the *target qubit*.

What does this gate do? This gate performs an *if-else* like operation. The output of the target qubit is dependent on the control qubit. If the control qubit is 0, we **DO NOT** apply the NOT/X gate, in other words, do nothing. If the control qubit is 1, we apply the NOT/X gate on the target qubit, flip the target.

The diagram is as follows. On q0 is the control qubit, designated as a small dot. On q1 is the target qubit, which is the X gate. It is an alternative symbol of the X gate (compared to the box with X). The vertical line implies that this is a controlled gate.

*A diagram of a connection

Description automatically generated with medium confidence*

How does this work? For 2 qubits, we can have 4 initial states . Let us examine the outputs for each initial state.

Notice for states we get the same output. This is because the control qubit is 0, thus we do not apply anything to the target qubit.

For states , we get different outputs. This is because the control qubit is 1, we flip the target qubit.

Mathematically, the CNOT gate is

Alternatively, in Dirac notation using outer products.

There are numerous control gates available, not just CNOT. There is also CY, CZ, etc. in which Y/Z/other gate is applied to the target qubits. There is also a gate with multiple controls, called the Toffoli gate of the CCNOT gate.

### Bell state/entanglement

Now that we can construct circuits with multiple qubits and gates and multiqubit gates. Let us try creating a circuit that creates entanglement.

A quantum state is separable/not entangled if for a state

iff

This means that we can separate or factor a quantum state into its basis states.

Consider the quantum state

We can see that it has coefficients of +-½, thus it satisfies ad=bc

Let’s try to separate them

We see that we can separate them.

Now let’s see an example of an entangled state.

Can we separate ?

We can see that

However, if we set any of the coefficients to 0, then we won’t be able to get 1/sqrt2. Thus, this state is entangled. This specific state is called a Bell state.

How do we construct a Bell state?

We use a Hadamard which produces superposition and CNOT which entangles the qubits

A diagram of a circuit

Description automatically generated

This circuit that produces entanglement is used in quantum algorithms such as quantum teleportation and superdense coding.

# Quantum Circuit Complexity

Ang kailangan kong gawin ay **circuit complexity** hindi **computational complexity**

From Wong

Chapter 1.7 – Computational complexity

**Complexity Classes**

In computer science, an algorithm is called *efficient* if it takes polynomial time or less. Constant time, logarithmic, loglinear time are also efficient.

An algorithm is *inefficient* if it takes more than polynomial time, called *superpolynomial time*. This includes algorithms that take *exponential time* such as 2^n or e^(n/1000). It also includes algorithms that take *subexponential time*, which are less than exponential time but greater than polynomial time such as 2^n^1/3.

Problems are *easy* if solutions are efficient, *hard* if solutions are inefficient.

Chapter 7.1 – Circuit vs Query complexity

**Circuit complexity** refers to the least number of quantum gates, relative to some universal set of quantum gates.

An *efficient* quantum algorithm is one with a polynomial circuit complexity.

**Size** – refers to number of gates it contains

**Depth** – maximal length of a path from an input gate to the output gate

Why does this matter?

There is a natural connection between circuit complexity and time complexity.

Since yung gates ay matrices, and sa quantum circuit we multiply matrices, then circuit complexity and time complexity are related.

Simple example

We want to get from |0> to -|1>

Attempt 1

We use 3 operations

Attempt 2

Here, we only use 2 operations to get the same result.

For a more practical example, let us consider the Toffoli gate. Suppose we are only allowed to use one and two qubit quantum gates.

A diagram of a circuit

Description automatically generated

Here we can see that the Toffoli gate can be decomposed into 16 smaller gates, thus we have a circuit size of 16.

However, we can still reduce this to 7 gates.

A diagram of a circuit

Description automatically generated

This means that we can restate algorithms and gates to have reduced circuit complexity and thus reduced time complexity. However circuit complexity is generally hard to find. There is no surefire way in reducing circuit complexity.

To quantify this, we say an *efficient* quantum algorithm is one with a polynomial circuit complexity. Consider the quantum adder circuit. This circuit uses 4n-2 Toffoli gates and 4n CNOT gates. Since we know we can express the Toffoli gates into 7 smaller one/two qubit gates we get 7(4n-2)+4n=32n-14 qubit gates, which is polynomial (specifically linear) in n. Therefore, we can say that the quantum adder circuit is an *efficient* algorithm.

# Alternative Models of Quantum Computation

Here we focus on using quantum circuits and their representations

1. Tensor networks
2. Decision diagrams

I need to ask DD peeps pano intindihin yung matmul using decision diagrams.

# Exercises

Ang gagawin ko lang ay one question per topic ko lmfao kasi im lazy like that

Kokopya lang din ako sa libro ng questions lmfao

**Quantum Computation**

Question is inspired by McMahon exercise 1.1

**Question:** Suppose a quantum state will represent uppercase characters in the English alphabet. What quantum state will represent the letter Q?

**Solution:** Since there are 26 letters in the English alphabet then we will need

We will need 5 qubits to represent all characters.

The letter Q is the 17th letter in the alphabet. But recall that we use 0 as the “first”. That means the Q will be the will be state

Converting 16 to binary is 10000, thus the quantum state representing Q is

**Quantum Gates**

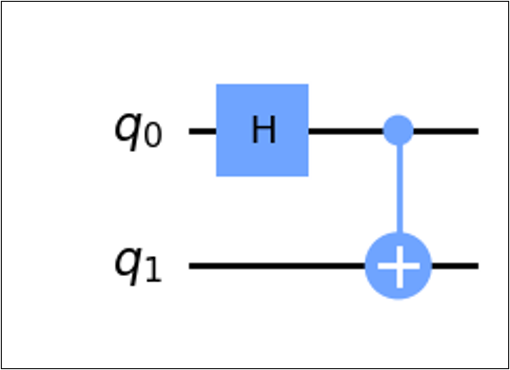
From Wong exercise 2.26

**Question:** Calculate

**Solution:** Recall that the Pauli gates are 180 degree rotations along their respective axes. This means that if we apply the same Pauli gate twice (e.g. XX, YY, ZZ) then we essentially get our original state. In general, if we apply the same gates an even number of times, we get the original state.

**Quantum Circuits**

Derive the four Bell states



**Solution:** Soln 1. Pwedeng mag compute ng unitary matrix then multiply the input statevector

Soln 2. Mas Madali to for me

Use yung known results na

Use the same idea for other Bell states

**Quantum Circuit Complexity**

**Question:** What is the difference between computational complexity and circuit complexity?

**Solution**:

Computational complexity – amount of resources required to perform an operation, typically time and space

Circuit complexity – Least number of quantum gates needed to implement a quantum circuit

**Alternative methods of quantum computation**

**Question:** In what scenarios are tensor network and decision diagrams effective

**Solution:**

Tensor network – Exploiting topological structures

Decision diagrams – exploiting redundancies

# References

**McMahon ch8**

**Nielsen ch1.3, ch4**

**Wong ch2.7**

# References

1. McMahon, D. (2007). Quantum Computing explained. Wiley-IEEE Computer Society Press.
2. Nielsen, M. A., & Chuang, I. L. (2010). Quantum computation and quantum information: 10th Anniversary Edition. Cambridge University Press.
3. Wong, T. (2022). Introduction to classical and quantum computing.