- Why QC ?

→ Quantum computers can solve search problem $\sqrt{N}$ steps.

- Haber - Bosch Process

→ This process is used to create Ammonia that will be used widely as a fertilizer. It is a critical chemical that helps us produce more food for larger human populations.

But it has some drawbacks. Infact, it uses about 1-2% of the entire world's energy supply.

③ How can we create fertilizer without using so much energy.

= Pea plant does this using a special molecule called nitrogenase to create its own fertilizer, without even using high T & P.

It is described by quantum information.

- Traveling Salesperson problem

→

It is an optimization problem about someone who wants to sell some goods at multiple cities. What they want to figure out is the shortest/fastest way possible for visiting all these cities one after another.

- Quantum physics

- It talks about very, very small things in our worlds like molecules, atoms & sub-atomic particles like electrons & protons!

~~All our so~~

- When we take photo using a digital camera, the light coming from our camera's flash or the environment goes toward the objects we are trying to capture in the image. The target object absorbs photons & reflects others. The photons are collected

by the lens. The lens focuses them on a sensor that causes the emission of electrons. The electrons will be emitted until the shutter of the camera closes. These emitted electrons are a -ve electric charges. The stored charges will then be converted to numbers or binary data with the help of a computer. When this binary data is printed out or sent to a computer, it will form an image that we can see.

It wont be possible without the concept of QP. It enables us to make electron- sensitive that can help us transfer photons to binary data.

- Scale of Quantum Objects

- Quantum information is contained in "qubits" which are the quantum version of a classical computer "bit". Things that are very cold, or very small, are described by qubits & can store qubits. for ex, superconductors

by the lens. The lens focuses them on a sensor that causes the emission of electrons. The electrons will be emitted until the shutter of the camera closes. These emitted electrons are a) -ve electric charges. The stored charges will then be converted to numbers or binary data with the help of a computer. When this binary data is printed out or sent to a computer, it will form an image that we can see.

It wont be possible without the concept of QP. It enables us to make electron-sensitive that can help us transfer photons to binary data.

- Scale of Quantum Objects

- Quantum information is contained in "qubits" which are the quantum version of a classical computer bit. Things that are very cold, or very small, are described by qubits & can store qubits. for ex, superconductors

can be used to store qubits. Single atoms even single electrons also store qubits.

- superfluid helium (watch video YT)
- superconductor liquid nitrogen (YT)

- The Double slit Experiment

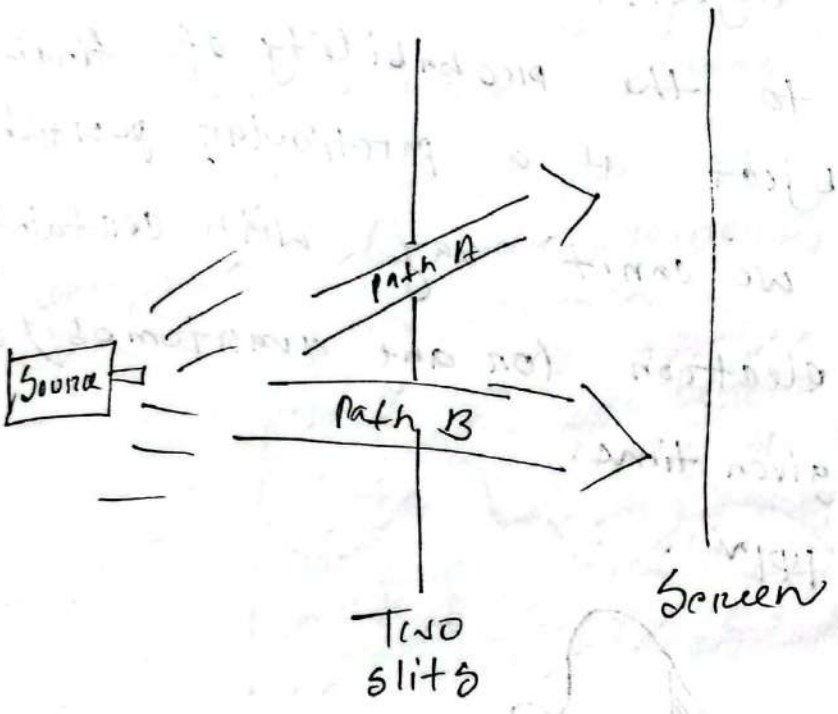
1. for particle -> random
2. waves -> interference
3. quantum objects -> interference
4. quantum objects but with adding an observer -> random

- It tells us quantum objects can have both wave-like & particle-like properties.

> [ This means that they exhibit particle like behavior if we try to find out which path the electron (or photon) took before it hit the screen ]

- Superposition

Think: If an electron (or any quantum object) passes through the double slit & reaches the screen, which path did it exactly take (path A or B)?



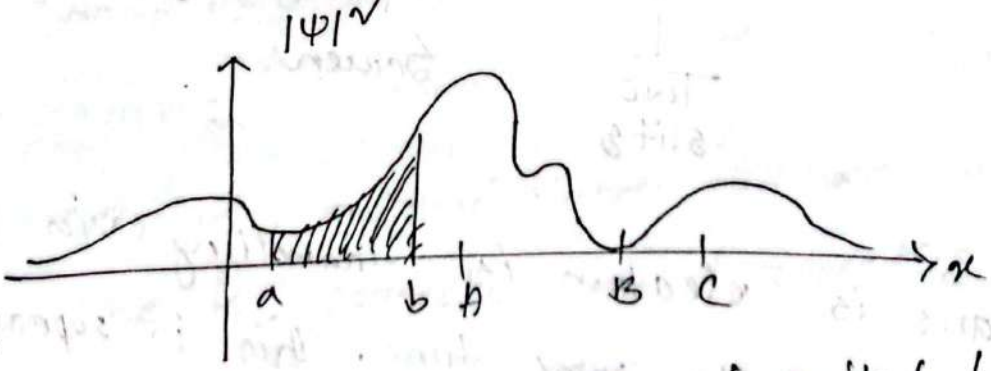

The ans is electron is travelling via both paths at the same time. This is superposition.

However things change if we place a detector to find out which path was actually taken by the electron. In this case electron suddenly starts behaving normally like a particle. The act of measurement changes the way the electron was behaving.

- Quantum wavefunctions

- Quantum objects follow the rules of quantum mechanics (like Schrodinger's equation)

This equation describes the wavefunctions of quantum objects. This wavefunctions are related to the probability of finding a quantum object at a particular position. This means we can't say with certainty where the electron (or any quantum obj) will be at any given time.

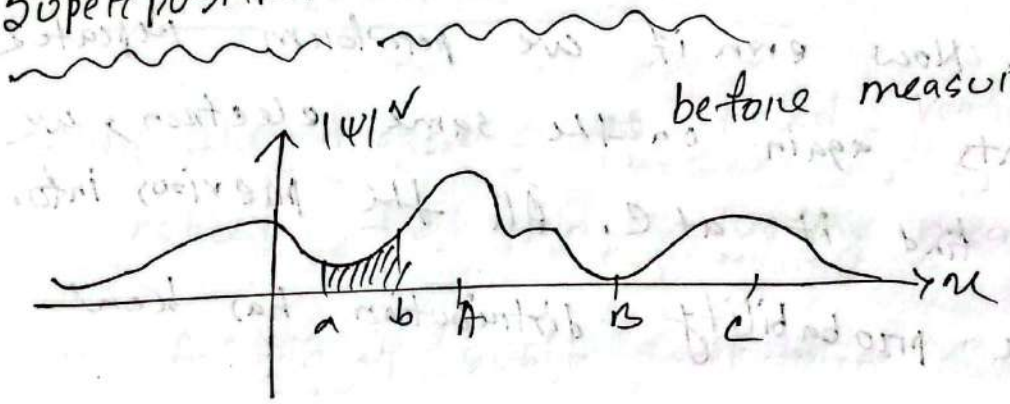


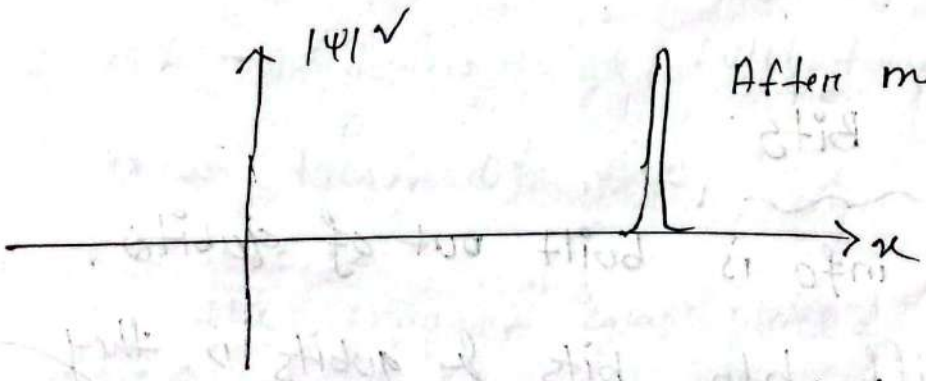

let a quantum object be described by a wavefunction ψ. The figure depicts the probability of finding the quantum object at any point along x axis. The shaded region represents the probability of finding the quantum objects btn a & b. This just depicts th

chance of finding the quantum object in the shaded region - there are no guarantee here!

"Heisenberg's uncertainty principle dictates that the more precisely determines a quantum objects position in, the less precise is its momentum (& vice versa)"

- Superposition & measurement




(Q) Where exactly was the electron just before it was measured to be in state C?

→ Before measurement, we were not sure where exactly the electron was. We say that this the electron is in a superposition state of all

possible points that are predicted by the wavefunction. The prob. of finding the electron at any specific point is determined by the wavefunction.

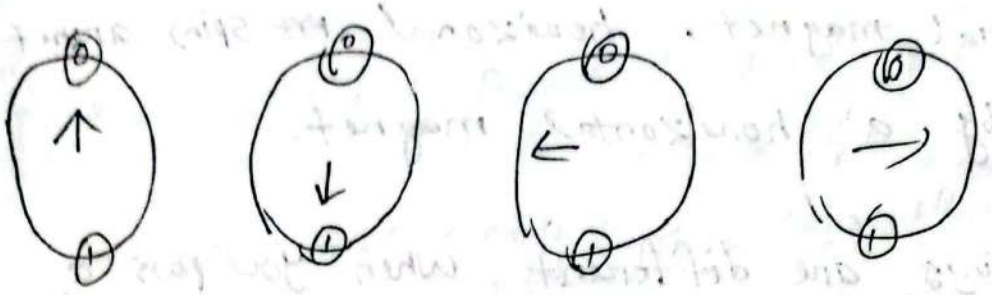
(3) What if we measure the position of the electron again? Will be still be at point C?

-> When we determined once, we say that the wavefunction has collapsed to that specific position. Now even if we perform repeated measurements again on the same electron, we will still find it at C. All the previous info. about the probability distribution has been erased from its memory!

- Quantum Bits

- quantum info is built out of qubits. The main diff btn bits & qubits is, they can be in more than 2 states. There are many ways to store a qubit. Some ex. are using photons, using superconductors, or using the spin of a small particle, like an

atom or electron. Spin is a property of particles that we can determine using magnets.



- Stern-Gerlach Observations

link: Stern-Gerlach Simulator

- When the magnet was placed vertically (Z axis) we noticed that upon conducting measurement

1. A spin up qubit will turn out to be 100% of the time.

2. A spin down qubit will turn out to be down 100% of the time.

- When the magnet was placed horizontally (X axis)

1. A left qubit will turn out to be left 100% of the time

2. A right qubit will turn out to be

right 100% of the time

So we can say, vertical spins are not affected by a vertical magnet, horizonal spins are affected by a horizontal magnet.

- However things are different when you pass a left qubit through a magnet along -Z direction. Now there's a 50-50 chance the qubit will go up or down!

In the quantum world, it is said that left state is a superposition of up & down. Superposition is a way to describe a particle being in a combination of known states. In the case of a right or right qubit going through a vertical magnet, we can think of "left" as being a bit of up & down. By measuring the qubit, we can force the qubit into either up or down.

- Superposition & Measurement

Left = UP - DOWN
Right = UP + DOWN
} we can do either one

- When we pass a spin through a magnet with a blocker, the spin can go in only one of two directions. finding out which direction it goes means we have performed a quantum measurement.

- A quantum logic gate is used to manipulate quantum info (the quantum state of a qubit or a collection of qubits) to enable us to perform tasks that require using concepts of superposition & measurement

- Quantum Circuit

There are 3 types of quantum circuit

1. Lines; Wires carrying qubits from place to place.
2. Gates: Gates do things to qubits
3. Measurements: These measure qubits, just like the

- Superposition & Measurement

Left = UP - Down
Right = UP + Down
} we can do either one

- When we pass a spin through a magnet with a blocker, the spin can go in only one of two directions. finding out which direction it goes means we have performed a quantum measurement.

- A quantum logic gate is used to manipulate quantum info (the quantum state of a qubit or a collection of qubits) to enable us to perform tasks that require using concepts of superposition & measurement

- Quantum Circuit

There are 3 types of quantum circuit

1. Lines; Wires carrying qubits from place to place.
2. Gates: Gates do things to qubits
3. Measurements: These measure qubits, just like the

magnets & blockers in the stern-Gerlach exp.

- Ibm circuit composer

⊕ → not gate

|0⟩ (initialize to be zero)

[H] (Hadamard gate, for making superposition)

in the end we should use [↗] (measurement

|0⟩ instead of spin-up qubit

|1⟩ " " " down "

superposition state of |0⟩ & plus |1⟩ instead of spin right qubit

|0⟩ & minus |1⟩ instead of spin left qubit.

imp note: The Q-sphere on IBM isn't related to spin direction. It simply shows what outcomes are possible after applying a measurement gate.

|0⟩ ⊕ [↗] (prob. of qubit being 1 is 100%)

|0⟩ [↗] ⊕ ( " " " " 0% )

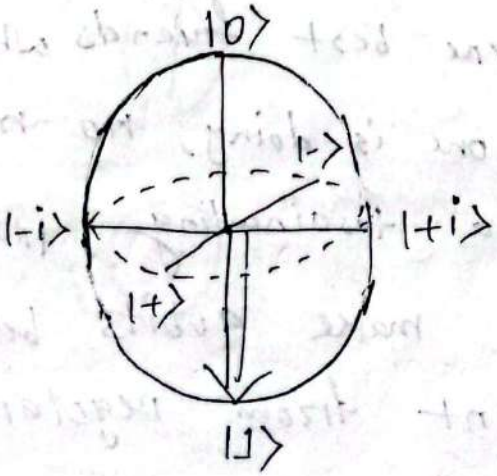
$|0\rangle$ ⊕ H $\wedge^2$ (Prob of qubit being 1 is 50%)

$|0\rangle$ H H $\wedge^2$ (Prob of qubit being 1 is 0%)

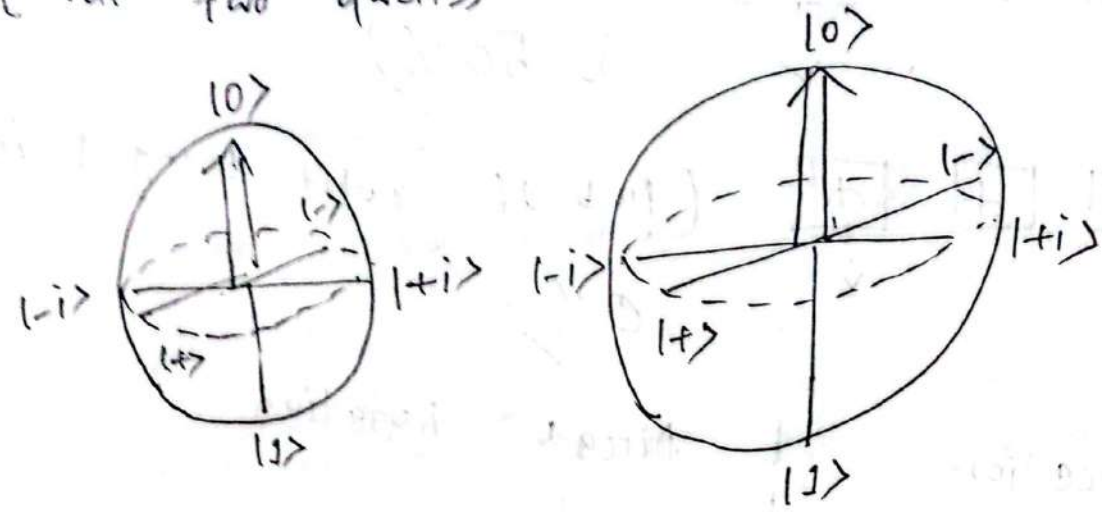
| Spin direction | Dirac notation |
|---|---|
| Up | $\|0\rangle$ |
| Down | $\|1\rangle$ |
| Right | a superposition of $\|0\rangle + \|1\rangle$ |
| Left | " " " $\|0\rangle - \|1\rangle$ |

- Two Qubits

$|1\rangle$ on a Block sphere.



For two qubits,




But there are actually more possibilities tha just each qubit being separately in one of the states up, down, left, right. These states ca only be described in reference to the other qubi When this happens, we say the two qubit are entangled.

** - When two qubits are in entangled state, it means that they are connected in a very special way. It's like they're best friends who always know what other one is doing, no matter how far we are. It's a fascinating property of quantom physic) that make qubits behave in a way that's different from regular bits in classical computers. When qubits are entangled thi states are connected in a special, mysterious

way & changes to one qubit instantly affect the other, no matter how far they are.

- Spins

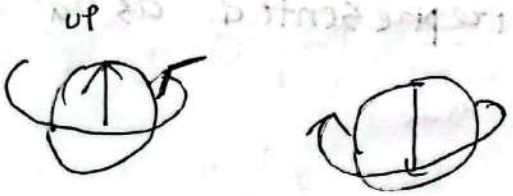

spin = $\pm \frac{\hbar}{2}$

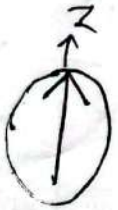
charge = $-1.6 \times 10^{-19} C$

mass = $9.1 \times 10^{-31}$ kg

spin = angular momentum

Total angular momentum = spin + "orbital" angular momentum




make spin measurement along z direction

momentum measurement along x direction

$\hat{S}_z |\psi\rangle$

↑ measurement operation (this one measures spin along z)

if it is already in measurement state, again after measurement it won't affect


$|\psi\rangle = |\uparrow\rangle$

spin up

("nice" = eigenstate of $\hat{S}_z$)

$\hat{S}_z|\uparrow\rangle = +\frac{\hbar}{2}|\uparrow\rangle$ ) Eigenvalue equation

↑ size of angular momentum $= \frac{\hbar}{2}$

$|\psi\rangle$ can be represented as a vector


spin up $= \begin{pmatrix} 1 \\ 0 \end{pmatrix}$


spin down $= \begin{pmatrix} 0 \\ 1 \end{pmatrix}$


superposition

$|\psi\rangle = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 \\ 1 \end{pmatrix}$ ← equal blend of spin up & spin down

after measurement

↓

$|\psi_{new}\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$

$|\psi\rangle = \frac{1}{\sqrt{26}} \begin{pmatrix} 5 \\ 1 \end{pmatrix}$ → more spin up in the blend

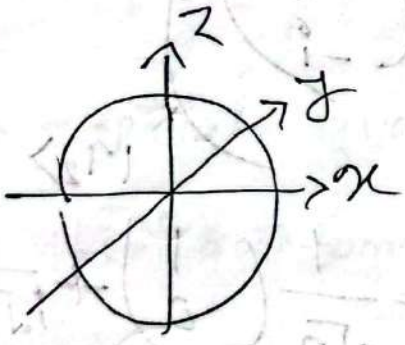
* Measurement operators (e.g $\hat{S}_z$) described by matrix & quantum states as vectors

$\hat{S}_z |\uparrow\rangle = +\frac{\hbar}{2}|\uparrow\rangle \rightarrow \hat{S}_z \begin{pmatrix} 1 \\ 0 \end{pmatrix} = +\frac{\hbar}{2}\begin{pmatrix} 1 \\ 0 \end{pmatrix}$

& $\hat{S}_z |\downarrow\rangle = -\frac{\hbar}{2}|\downarrow\rangle \rightarrow \hat{S}_z \begin{pmatrix} 0 \\ 1 \end{pmatrix} = -\frac{\hbar}{2}\begin{pmatrix} 0 \\ 1 \end{pmatrix}$

$\hat{S}_z = \frac{\hbar}{2}\begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$

z dir




we could measure by any axis. Whenever we make a measurement the wave function will either collapsed into either clockwise or anticlockwise ~~of~~ around the choosen ~~direction~~ axis.

$$\hat{S}_z = \frac{\hbar}{2}\begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$

"rotate" this

$$\hat{S}_x = \frac{\hbar}{2}\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$

$$\hat{S}_y = \frac{\hbar}{2}\begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}$$

"Eigenstates" of $\hat{S}_y$

i.e spin up & spin down in y-dir

$$\hat{S}_y|\psi\rangle = \pm\frac{\hbar}{2}|\psi\rangle$$

$$|\uparrow_y\rangle = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 \\ i \end{pmatrix}$$

$$|\downarrow_y\rangle = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 \\ -i \end{pmatrix}$$

where $\begin{pmatrix} 1 \\ 0 \end{pmatrix}$ & $\begin{pmatrix} 0 \\ 1 \end{pmatrix}$ are in z direction

$$|\uparrow_y\rangle = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 \\ i \end{pmatrix} = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 \\ 0 \end{pmatrix} + \frac{1}{i\sqrt{2}}\begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

$|\uparrow_z\rangle$ $|\downarrow_z\rangle$

z eigenstates

if we took z measurement here it will collapsed to either z eigenstates with same probability of spin up & down.

$$\sigma_x = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$
$$\sigma_y = \begin{pmatrix} 0 & i \\ i & 0 \end{pmatrix}$$
$$\sigma_z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$

} Pauli matrices

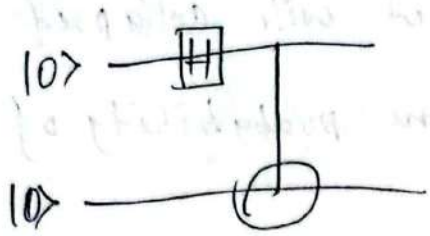
$$\hat{S}_x = \frac{\hbar}{2}\sigma_x, \quad \hat{S}_y = \frac{\hbar}{2}\sigma_y, \quad \hat{S}_z = \frac{\hbar}{2}\sigma_z$$

for "spin 1/2" particles (e.g. electrons)

- Bell state

It is a special quantum entangled state that describes the quantum correlation between 2 qubits.

There are 4 possible Bell states, famous one is "EPR pair" or "maximally entangled" Bell state.




$$|00\rangle \xrightarrow{H} \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)|0\rangle$$

$$= \frac{1}{\sqrt{2}}(|0\rangle|0\rangle + |1\rangle|0\rangle)$$

$$\xrightarrow{C\text{-}Not} \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle) \quad \text{"EPR"} \ \Phi^+$$

~~$\frac{1}{\sqrt{2}}(|0\rangle|0\rangle +$~~

$$\begin{matrix}0\\1\end{matrix} \rightarrow \frac{1}{\sqrt{2}}(|01\rangle + |10\rangle) \ \Psi^+$$

$$\begin{matrix}1\\0\end{matrix} \rightarrow \frac{1}{\sqrt{2}}(|00\rangle - |11\rangle) \ \Phi^-$$

$$\begin{matrix}1\\1\end{matrix} \rightarrow \frac{1}{\sqrt{2}}(|01\rangle - |10\rangle) \ \Psi^-$$

" ① If u measure qubit A & find it in the state $|0\rangle$, you instantly know the qubit B is also in the state $|0\rangle$. They are

perfectly correlated.

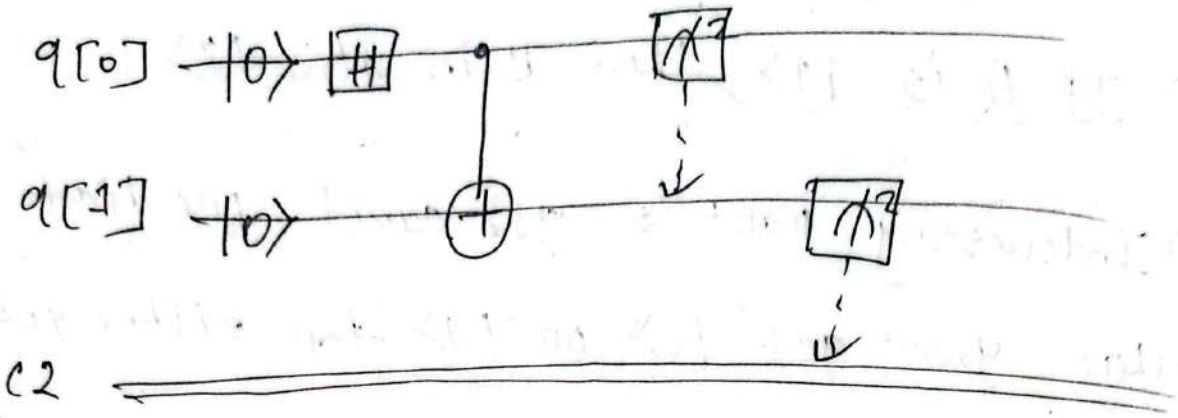
(ii) If A is $|1\rangle$, then B is also $|1\rangle$.

(iii) Interesting part is you can't predict whether you'll get $|0\rangle$ or $|1\rangle$ for either qubit A or B. But as soon as you measure one of them, you'll know the other one instantly, no matter how far they are.

Rules of entanglement :

(i) When the qubits are entangled, the quantum states of both qubits are always related to each other

(ii) An entangled pair remains entangled regardless of the measurement basis.




- Quantum Teleportation

let's use an example,

The year is 2101 & humans & otters are living harmoniously. Together they created a 2nd home on Mars! Joey (the otter) lives on Earth & would like to go visit Bob (the human), who lives on Mars.

Alice (the human) is also friends with Joey & Bob, & decides to help Joey to get to Mars. A major issue is that Joey is afraid of spaceship, so they have to find a diff way to get Joey to Mars.

To get to Mars, Alice & Joey first send one of qubits from a Bell state to Mars, so that there are entangled particles spread between the two planets.

We'll first need to turn Joey into a qubit. Others, like all living things can't turn into a qubit. However imagine a scenario where Joey can do this.

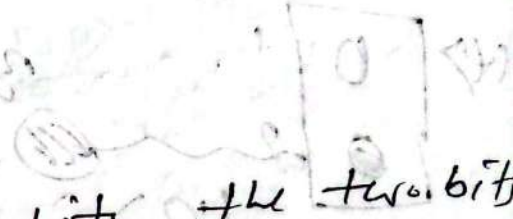
Joey is now recorded as the state of the qubit. It might be up, down, left or right state for Joey to safely reach on Mars, we need to make sure the contact of Joey state ends up at Mars.

Alice has 2 qubits: one holds the Joey state & the other is one half of the entangled Bell pairs shared with Bob (the other half is on Mars!) The last step to teleport the Joey state to Mars is for Alice to perform a special type of measurement called a

Bell measurement on her qubits. A bell measurement is done on 2 qubits at once. & Alice will get some measurement outcomes which are a pair of classical bits, for example 00 or 01.

After Alice does this measurement, the qubit over on Mars will be one of the states up, down, left, right. We know from the rules of entanglement that the state of Bob's qubit on Mars will be related to the state of Alice's qubit on Earth, cz they were part of entangled bell pair. But, just bcz the 2 states are related doesn't mean they're exactly the same. Maybe Alice's qubit is in the opposite state to Bob's. If that the case, we'll need to fix it so that Alice & Bob end up with exactly the same state.

We can fix Bob's qubit on Mars by using classically controlled gates. Remember that a classically controlled gate applies a quantum gate to a qubit depending on the value of a classical bit. In this case we'll use Alice's measurement outcomes to control the gates that Bob applies to his qubits. So Alice send her bits over to Bob. If Bob applies the right gate, Joey will be on Mars. If not... who knows what fate could befall Joey!



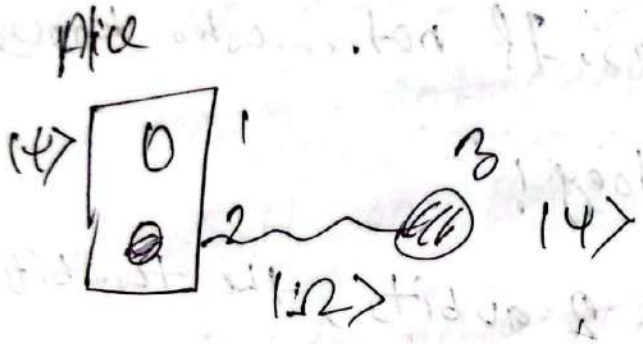
When Alice measures the 2 qubits, the two bits she gets out are:

- 00 with 25% chance
- 01 " " " "
- 10 " " " "
- 11 " " " "

For each diffrent outcome, Bob will have to do diffrent gates to fix up the Joey state.

Bob applies gates on his qubit using the results of Alice's measurements:

00 → No gate
01 → Apply X gate
10 → Apply Z gate
11 → " ZX "



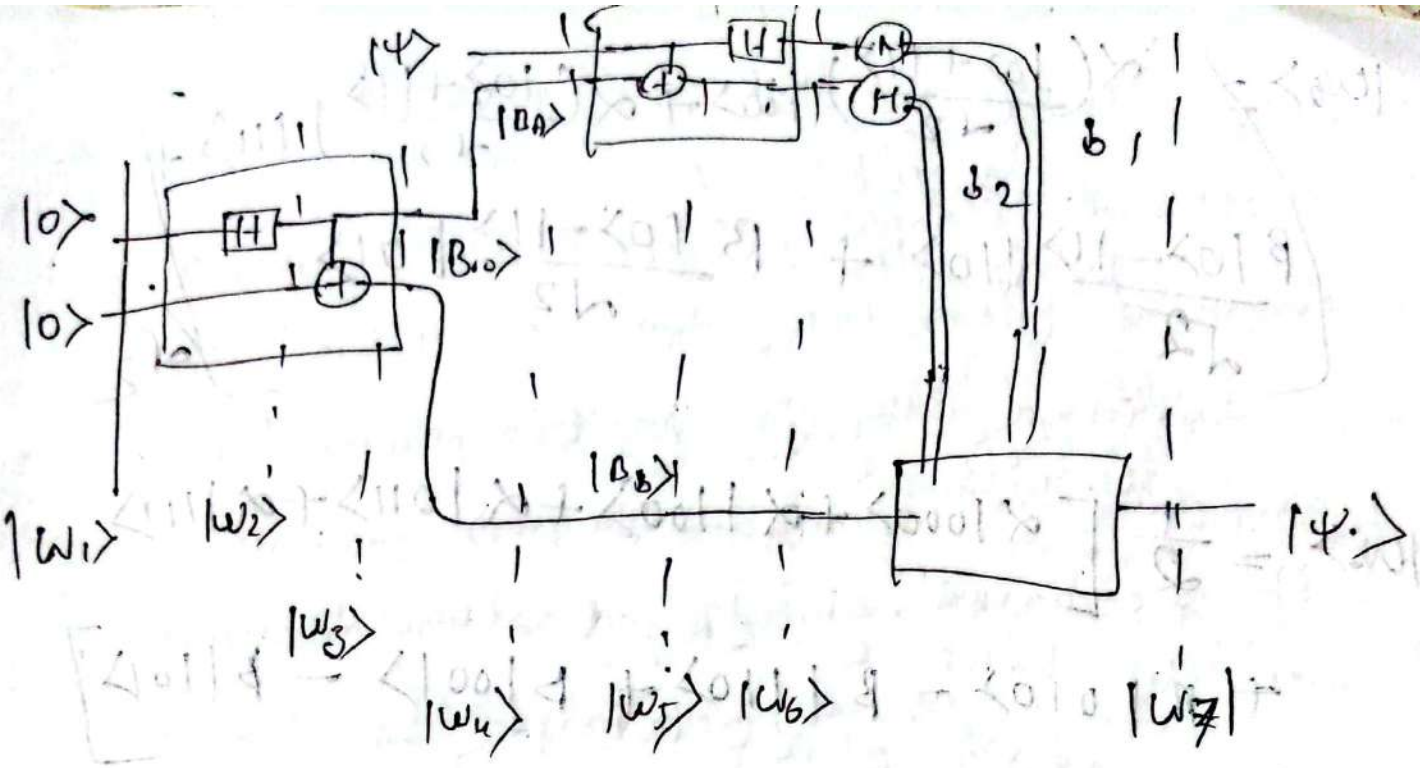

$(\alpha|0\rangle + \beta|1\rangle)\frac{1}{\sqrt{2}}(|00\rangle + |11\rangle) =$

$\frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)(\alpha|0\rangle + \beta|1\rangle) \rightarrow 1$

$+ \frac{1}{\sqrt{2}}(|00\rangle - |11\rangle)(\alpha|0\rangle - \beta|1\rangle) \rightarrow 2$

$+ \frac{1}{\sqrt{2}}(|01\rangle + |10\rangle)(\alpha|1\rangle + \beta|0\rangle) \rightarrow X$

$+ \frac{1}{\sqrt{2}}(|01\rangle - |10\rangle)(\alpha|1\rangle - \beta|0\rangle) \rightarrow ZX$




$$|w_1\rangle = |0\rangle|0\rangle$$

$$|w_2\rangle = |H|0\rangle|0\rangle = \frac{|0\rangle+|1\rangle}{\sqrt{2}}|0\rangle = \frac{|00\rangle+|10\rangle}{\sqrt{2}}$$

$$|w_3\rangle = \frac{|00\rangle+|11\rangle}{\sqrt{2}} = |B_{00}\rangle$$

$$|w_4\rangle = |\psi\rangle|B_{00}\rangle = (\alpha|0\rangle+\beta|1\rangle)\left(\frac{|00\rangle+|11\rangle}{\sqrt{2}}\right)$$

$$= \frac{\alpha|000\rangle+\alpha|011\rangle+\beta|100\rangle+\beta|111\rangle}{\sqrt{2}}$$

$$|w_5\rangle = \frac{\alpha|000\rangle+\alpha|011\rangle+\beta|110\rangle+\beta|101\rangle}{\sqrt{2}}$$

$$|\omega_0\rangle = \left[\alpha\left(\frac{|0\rangle+|1\rangle}{\sqrt{2}}\right)|00\rangle + \alpha\left(\frac{|0\rangle+|1\rangle}{\sqrt{2}}\right)|11\rangle + \frac{\beta|0\rangle-|1\rangle}{\sqrt{2}}|10\rangle + \beta\frac{|0\rangle-|1\rangle}{\sqrt{2}}|01\rangle\right]\frac{1}{\sqrt{2}}$$

$$|\omega_1\rangle = \frac{1}{2}\left[\alpha|000\rangle + \alpha|100\rangle + \alpha|011\rangle + \alpha|111\rangle + \beta|010\rangle - \beta|110\rangle + \beta|001\rangle - \beta|101\rangle\right]$$

$$= \frac{1}{2}\left[|00\rangle(\alpha|0\rangle+\beta|1\rangle) + |10\rangle(\alpha|0\rangle-\beta|1\rangle) + |01\rangle(\alpha|1\rangle+\beta|0\rangle) + |11\rangle(\alpha|1\rangle-\beta|0\rangle)\right]$$

| (A) | Prob | Resultant state | Operation |
|---|---|---|---|
| 00 | 1/4 | $\alpha\|0\rangle+\beta\|1\rangle$ | NO |
| 10 | 1/4 | $\alpha\|0\rangle-\beta\|1\rangle$ | $Z\|B_0\rangle = \alpha\|0\rangle + \beta\|1\rangle$ |
| 01 | 1/4 | $\alpha\|1\rangle+\beta\|0\rangle$ | $X\|B_0\rangle = \alpha\|1\rangle + \beta\|1\rangle$ |
| 11 | 1/4 | $\alpha\|1\rangle-\beta\|0\rangle$ | $ZX\|B_0\rangle = \alpha\|0\rangle - \beta\|1\rangle$ |

$ZX|B_0\rangle = \alpha|0\rangle + \beta|1\rangle$

- Coding Platforms

↳ Quantum computers are made of qubits, unlike bits, qubits can only not easily be represented using semiconductors or other materials used in classical computers. Instead scientists use superconductors or photons. Depending on the method of representing the qubit, the quantum computer will behave slightly differently. However, they will all have the same overarching systems. for ex, they will all use gates like we learned about. Quantum gates will also come together to shape quantum circuits & in a larger scale quantum hardware like Quantum Processing Units (QPUs)

- Quantum Programming

Q# → low level, directly works with hardware or very close to it.

Ocean → allows to write programs in Python

Qiskit → can be used 2 ways (on IBM's quantum computers, on a simulator that behaves like a quantum computer)

- Quantum Circuits with Qiskit

The workflow of using Qiskit consists of 3 high level steps: Build, Execute, Analyze

Build: first, we need to design the circuit based on the specifications given to us. Once the design is ready we need to translate it into the Qiskit language.

Execute:

Analyze:

- Qiskit Program

✓ !pip install qiskit

we need libraries like,

1. Python libraries ( numpy, scipy, matplotlib)

2. Qiskit package used to build quantum circuit:
To initialize & store qubits (QuantumRegister), store the measurement to classical bits (ClassicalRegister), & build a circuit (QuantumCircuit)

3. Qiskit package used to simulate the quantum circuit: we will need to run the quantum circuit we built on a simulators on a quantum computer (execute). Here, we will use the basic backend provided in Qiskit (Aer)

4. Qiskit packages to visualize & analyze results; we will use the most basic histogram plot (plot_histogram) from the qiskit.visualization package.

✓ import numpy as np
from qiskit import QuantumRegister, ClassicalRegi
from qiskit import QuantumCircuit
from qiskit import execute, Aer
from qiskit.visualization import plot_histogram

To build a circuit,

1. initialize variables (quantum & classical registers, circuit itself)
2. add gates (operations & measurements)
3. visualize the circuit.

✓ # creat quantum register to store qubit
qreg_q = QuantumRegister(1, 'q')
# create classical register to store the regist results
creg_c = ClassicalRegister(1, 'c')
# Initialize the circuit
circuit = QuantumCircuit(qreg_q, creg_c)
# Initialize all qubits to |0>

```
circuit.reset(qreg_q)
# Apply the hadamard gate
circuit.h(qreg_q)
# Apply measurement
circuit.measure(qreg_q, creg_c)
# visualize the circuit
circuit.draw()
```

✓ !pip install qiskit-aer

Qiskit Aer is a high performance simulation framework for quantum circuits. We can change the backend of Aer to achieve different simulation goals. We can even include an actual quantum computer as the backend for more complicated circuits.

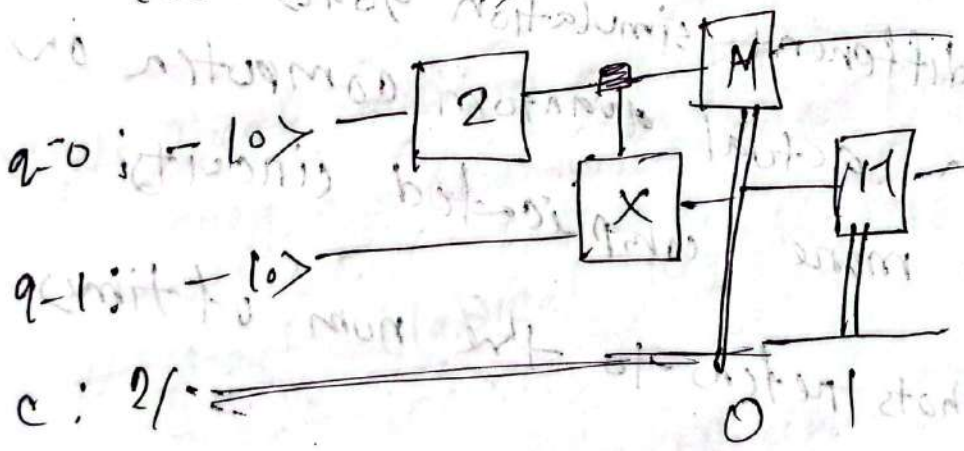
Note: Num of shots refers to the num of times the circuit is run

```
✓ simulator = Aer.get_backend('qasm_simulator')
  job = execute(circuit, simulator, shots=1000)
  result = job.result()
  counts = result.get_counts(circuit)
```

print("\n Output counts: ", counts)

another one /

```
qreg_q = QuantumRegister(2, 'q')
creg_c = ClassicalRegister(2, 'c')
circuit = QuantumCircuit(qreg_q, creg_c)
circuit.reset(qreg_q)
circuit.z(qreg_q[0])
circuit.cx(qreg_q[0], qreg_q[1])
circuit.measure(qreg_q, creg_c)
circuit.draw()
```



• Spin Qubits

↳ There are various ways qubits can be created for real-world quantum computers. (spin, atoms, photons etc).

A qubit made out of spin of electrons, aka spin qubit, can be in the ↑ state or ↓ state or a combination of both states.

↑ - qubit state = $|0\rangle$

↓ - " " = $|1\rangle$

↳ Key notation & Vectors:

The symbols $|\rangle$ used to describe the qubits are just a fashionable way of describing simple vectors as given below,

$$|0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

$$|1\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

We can define the transpose of the $|0\rangle$ & $|1\rangle$ vectors as

$\langle 0| = [1 \quad 0]$

$\langle 1| = [0 \quad 1]$

Defining quantum states using such symbols is called bra-ket notation where $|\rangle$ is the ket vectors & $\langle|$ is the bra vector.

↳ Inner products

$$\langle 0|1\rangle = [1 \quad 0] \cdot \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

$$= (1)(0) + (0)(1)$$

$$= [0]$$

$$\langle 0|0\rangle = [1 \quad 0] \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

$$= (1)(1) + (0)(0) = 1$$

$$\langle 1|0\rangle = [0 \quad 1] \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

$$= [0)(1) + 1(0)$$

$$= 0$$

- Superposition

↳ orthonormal basis

→ $\langle 0|1\rangle$ or $\langle 1|0\rangle$ results in 0. This property is called orthogonality.

$\langle 0|0\rangle$ or $\langle 1|1\rangle$ results in 1. This property is called normality.

Combining these 2 properties we get the most crucial characteristics of the quatum state vectors — orthonormality.

→ The quantum states $|0\rangle$ & $|1\rangle$ are special vectors, called basis vectors. What that means is that you can construct any quantum state by a combination (such as addition, subtraction, multiplication by a number) of the $|0\rangle$ & $|1\rangle$. Moreover $|0\rangle$ & $|1\rangle$ belong to a special class of basis vectors called orthonormal basis vectors, due to a characteristic property of orthonormality.

## superposition

Any linear combination of $|0\rangle$ & $|1\rangle$ gives rise to a different quantum states. Some examples of qubits formed by linear combination are,

$$|q_1\rangle = 0.6|0\rangle + 0.8|1\rangle$$

$$|q_2\rangle = \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle$$

$$|q_3\rangle = \frac{1}{2}|0\rangle + \frac{\sqrt{3}}{2}|1\rangle$$

The linear combination of $|0\rangle$ & $|1\rangle$ resulting in a completely new qubit, is called superposition, We can also convert this into conventional vector form. for instance;

$$|q_1\rangle = 0.6|0\rangle + 0.8|1\rangle$$

$$= 0.6\begin{bmatrix}1\\0\end{bmatrix} + 0.8\begin{bmatrix}0\\1\end{bmatrix}$$

$$= \begin{bmatrix}0.6\\0.8\end{bmatrix}$$

How to make qubits with superposition

$$|q\rangle = a|0\rangle + b|1\rangle$$

Here a & b are any 2 real numbers. In the bra notation, the qubit becomes $\langle q| = a\langle 0| + b\langle 1|$. Note that if we expand in terms of matrices (vectors), we will realise that $\langle q|$ is the transpose of $|q\rangle$. The inner product $\langle 1|q\rangle$ can be calculated as

$$\langle q|q\rangle = [a\langle 0| + b\langle 1|] \cdot [a|0\rangle + b|1\rangle]$$

$$= a^2\langle 0|0\rangle + ab\langle 0|1\rangle + ba\langle 1|0\rangle + b^2\langle 1|1\rangle$$

$$= a^2\langle 0|0\rangle + b^2\langle 1|1\rangle$$

$$= a^2 + b^2$$

If we evaluate all the ~~final~~ inner products $\langle q_1|q_2\rangle$, $\langle q_2|q_2\rangle$, $\langle q_3|q_3\rangle$, from the above example you will get all $= 1$.

To build any qubit using superposition we need to ensure that the inner prod. of the qubit with itself is 1. A simple form to remember for a qubit $|q\rangle = a|0\rangle + b|1\rangle$,

$a^2 + b^2 = 1$

This rule must be followed while constructing any qubit using the principle of superposition.

- More superposition

In order to construct a qubit using the superposition principle, there is no rule which states a & b have to be real numbers.

for an ex, $|q\rangle = \frac{1}{\sqrt{2}}|0\rangle + \frac{i}{\sqrt{2}}|1\rangle$.

$\langle \eta | \eta \rangle = \left[\frac{1}{\sqrt{2}}|0\rangle - \frac{i}{\sqrt{2}}|1\rangle\right]$

$= \left[\frac{1}{\sqrt{2}}\langle 0| - \frac{i}{\sqrt{2}}\langle 1|\right] \cdot \left[\frac{1}{\sqrt{2}}|0\rangle + \frac{i}{\sqrt{2}}|1\rangle\right]$

$= \left(\frac{1}{\sqrt{2}}\right)^2 \langle 0|0\rangle + \left(\frac{1}{\sqrt{2}}\left(\frac{i}{\sqrt{2}}\right)\langle 0|1\rangle + \left(\frac{-i}{\sqrt{2}}\right)\left(\frac{1}{\sqrt{2}}\right)\right.$

$\left.\langle 1|0\rangle + \left(\frac{-i}{\sqrt{2}}\right)\left(\frac{i}{\sqrt{2}}\right)\langle 1|1\rangle\right.$

$= \left(\frac{1}{\sqrt{2}}\right)^2 \langle 0|0\rangle - \left(\frac{i}{\sqrt{2}}\right)^2 \langle 1|1\rangle$

$= \left(\frac{1}{\sqrt{2}}\right)^2 + \left(\frac{1}{\sqrt{2}}\right)^2$

$= 1$

imp note

In general in the presence of complex, we state that, $|a|^2 + |b|^2 = 1$, where $|a|$ & $|b|$ denote the modulus of complex numbers a & b.

## • Measurement of Qubit States

probabilities are useful for when there are many possible outcomes & we don't have enough info to work out which will happen. like a dice roll, or a coin toss. We give each outcome a probability & use these to work out the likelihood of something occurring. Probabilities work extremely well for things we usually see the world around us, but with "quantum" things (like qubits), this approach fails,

so what's the solution? To describe quantum mechanics we use probability amplitudes. Prob. amplitudes are similiar to normal prob. in that:

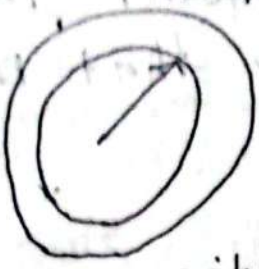
- amplitudes have a magnitude.
- each possible outcome has a probability amplitude
- & the magnitude of that outcome's amplitude tells us how likely that outcome is to occur.

But amplitudes also have an extra property which we call "phase"

probability

magnitude: 0.66

Amplitude



magnitude: 0.75
phase: 45°

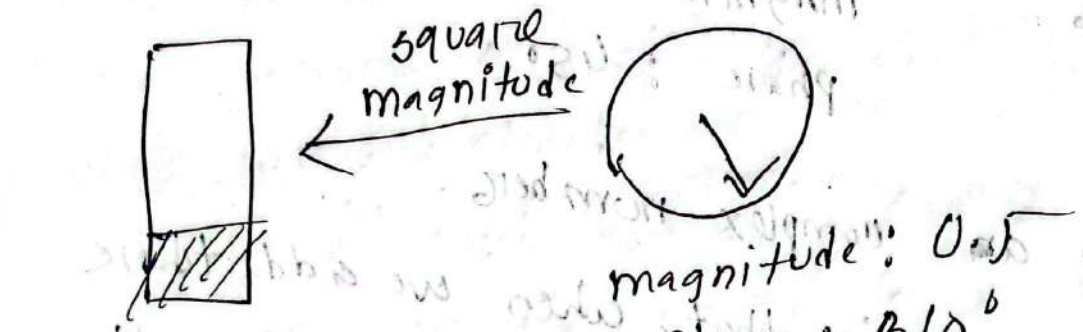
The amplitude is a complex number.
The result of phase is that when we add these amplitudes together, they can cancel each other out, just like +ve & -ve numbers do. This behavior is called interference & explains all the behavior specific to quantum mechanics that we don't see in classical mechanics.

magnitude: 0.7
phase: 25

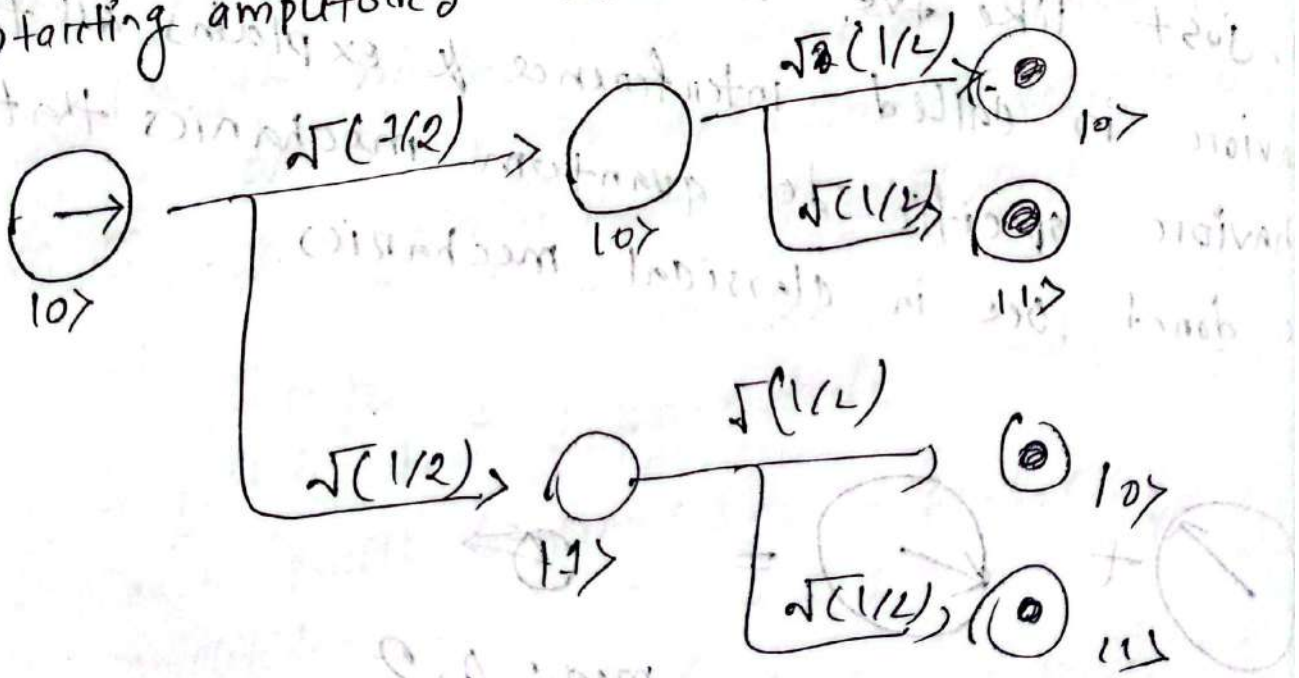
mag: 0.6
Phase: 190

mag: 0.2
phase: 77

To find the probability of measuring an outcome, we square the magnitude of that outcome's amplitude. This is a mathematical trick that makes everything add up nicely




Starting amplitudes



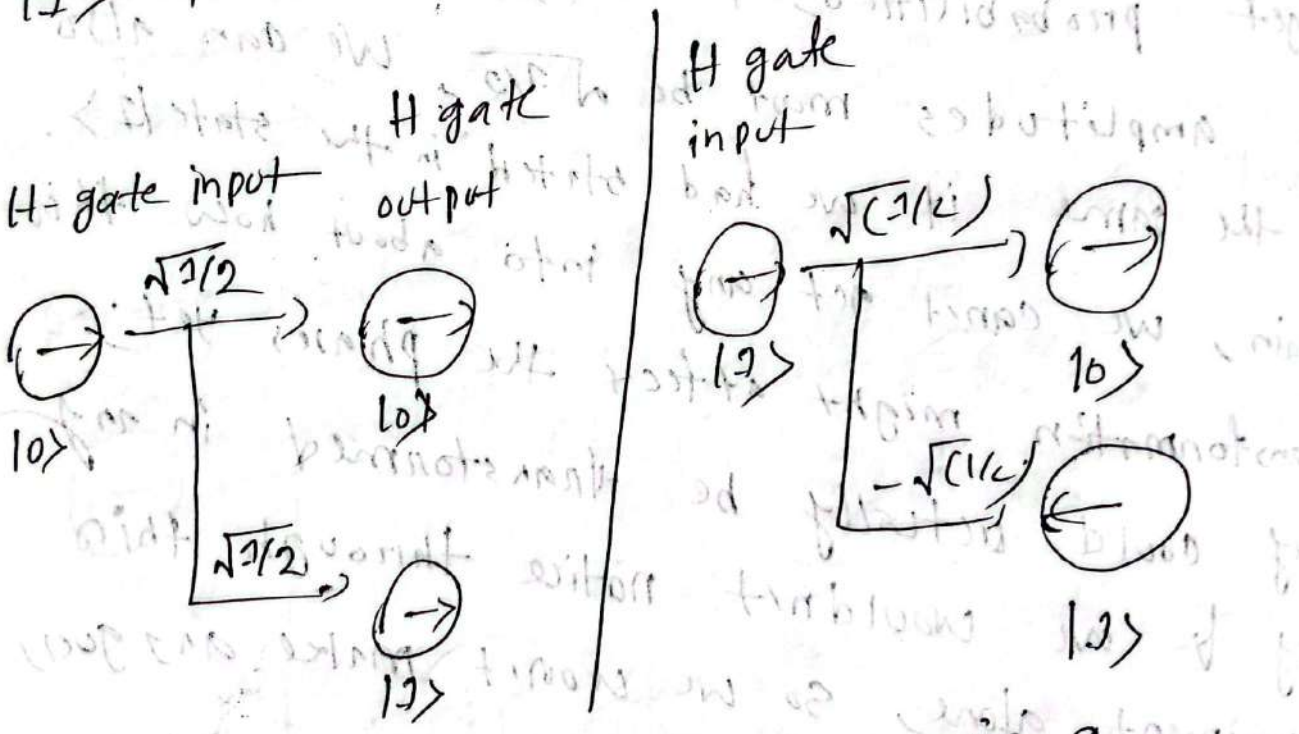
✓ Before the H gate the chance of measuring the state $|0\rangle$ was 1, so the magnitude of the amplitude must be 1 too. We can't tell what the phase of this information

✓ After we apply H gate we will measure either $|0\rangle$ or $|1\rangle$ with probability 1/2. Since we square the ~~amplitudes~~ magnitudes of our amplitudes to get probabilities, the magnitudes of these two amplitudes must be $\sqrt{1/2}$. We can also say the same if we had started in the state $|1\rangle$.

Again, we can't get any into about how this transformation might affect the phases yet; they could actually be transformed in any way & we wouldn't notice through this experiment alone, so we can't make any such

✓ But when we apply two H gates, we can start to make statements about the phases our amplitudes can have. At the end of our amplitude tree, we have four different branches that lead

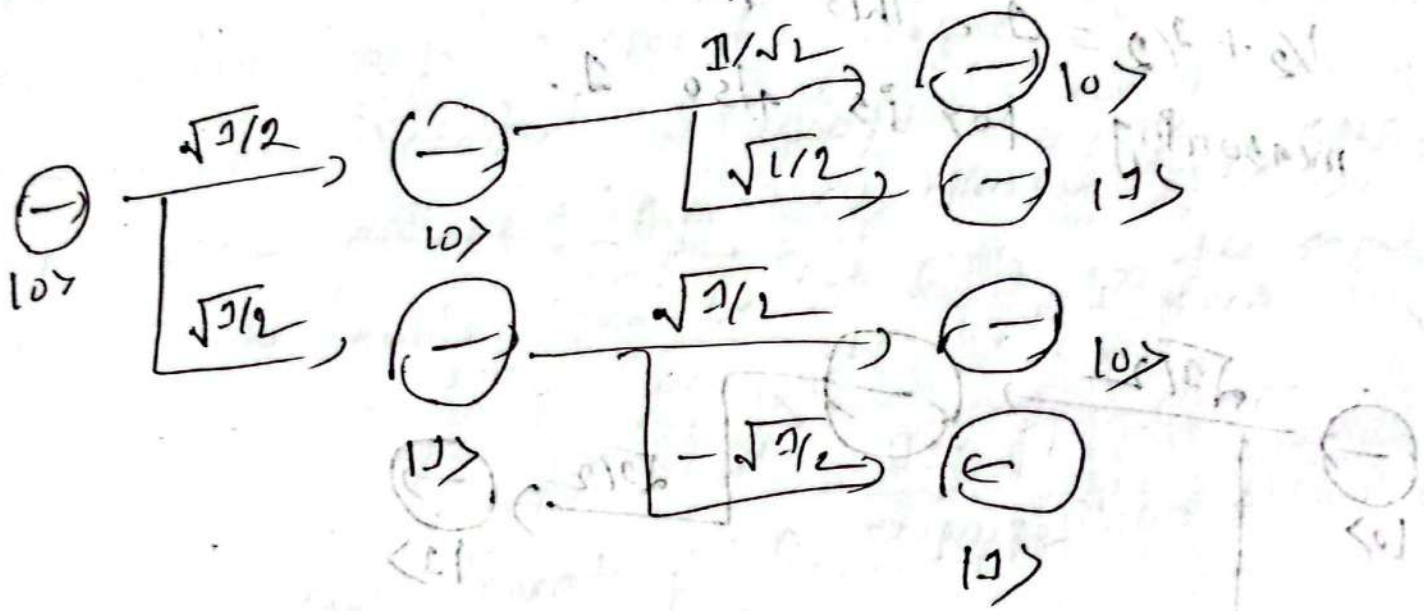
only 2 possible outcomes. To work out the final amplitudes for these outcomes, we need to multiply along those branches & add the amplitudes together, just as we would a probability tree. To give a 0% chance of measuring $|1\rangle$, we need the amplitudes on the 2 branches that lead to $|1\rangle$ to have opposite phases.



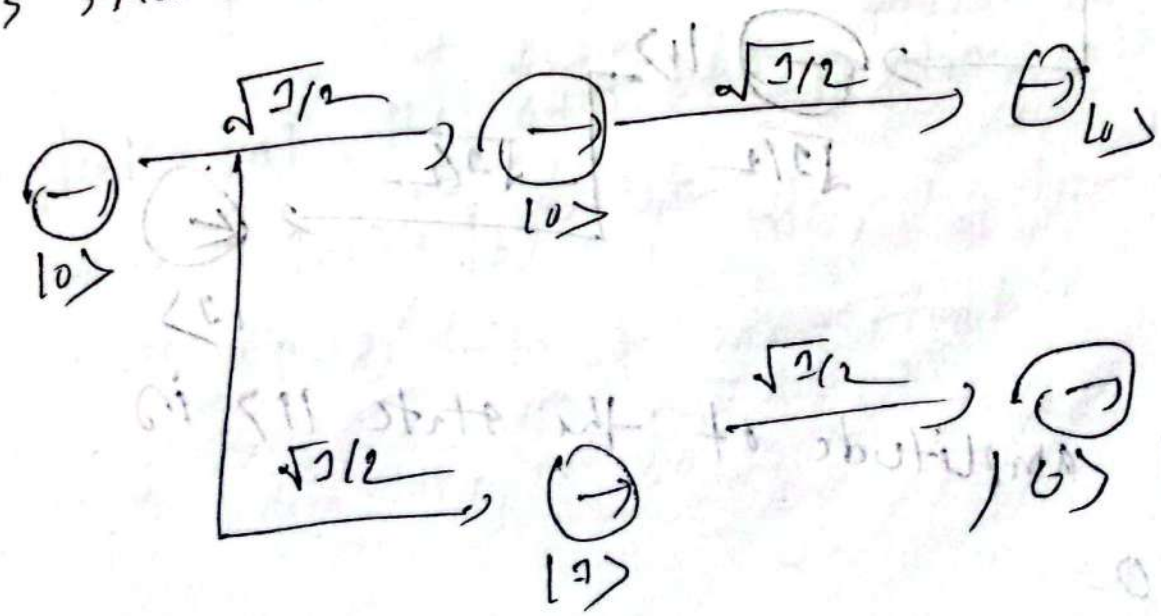

Since 2 H gates in sequence give a deterministic result, the H gate must always behave in the same way given the same input. Hence, one guess for what might happen: Acting on the $|0\rangle$ state, all outputs amplitude

point in the same direction, but acting on the $|1\rangle$ state, the phase of the $|1\rangle$ state is rotated 180°

✓ When we chain these together, we multiply along the branches & add up the final amplitudes for each state.




✓ Let's start



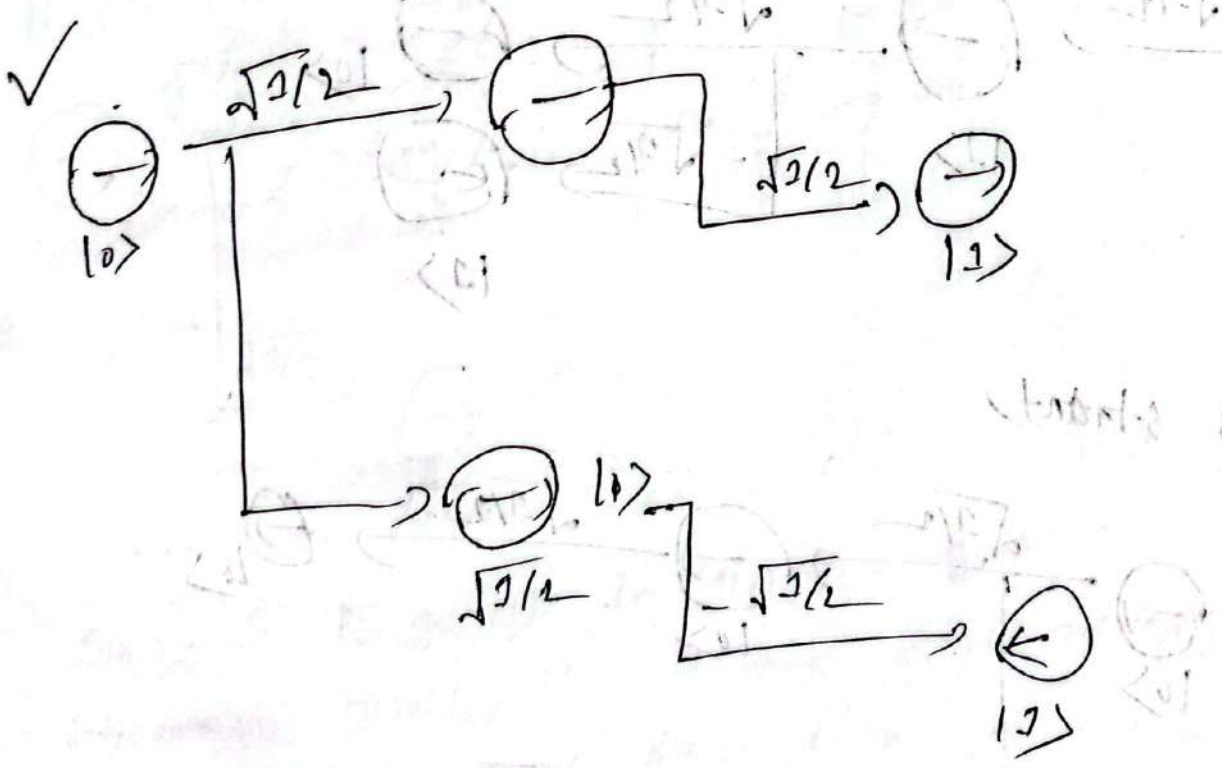
For top branch,

$$\sqrt{1/2} \times \sqrt{1/2} = 1/2$$

other branch,

$$\sqrt{1/2} \times \sqrt{1/2} = 1/2$$

Adding these 2 together, the final magnitude of the amplitude of the state $|0\rangle$ is $1/2 + 1/2 = 1$. This means the prob. of measuring $|0\rangle$ is also 1.




final amplitude of the state $|1\rangle$ is

0.

Here something counter intuitive has happened. If we toss a coin, it could be head or tails with equal probability. The results is that the coin will be in one of these states & we use prob. to deal with the fact that we don't know which.

But with this amplitude model, we can't say the qubit took a specific route ("0 → 0 → 0 or 0 → 1 → 0") because we would not see the interference effect. This leads people to say things like "the qubit can be 0 & 1 at the same time"

✓ let us again create a superposition state: $|q\rangle = a|0\rangle + b|1\rangle$.

let's look at the odds of getting "heads". We know that $|a|^2 + |b|^2 = 1$ must be true. And we need the odds to be same for both heads & tails — a coin toss is 50:50!

We can write this as,

$$|q\rangle = \frac{1}{\sqrt{2}}|0\rangle + \frac{i}{\sqrt{2}}|1\rangle$$

Let's create use inner product to calculate that the $|0\rangle$ state actually does have a probability of 50%. We will call, $|q_0\rangle$

Now,

$$\langle 0||q_0\rangle = \frac{1}{\sqrt{2}}\langle 0||0\rangle + \frac{i}{\sqrt{2}}\langle 0||1\rangle$$

$$\langle 0||q_0\rangle = \frac{1}{\sqrt{2}} * 1 + \frac{i}{\sqrt{2}} * 0$$

$$\langle 0||q_0\rangle = \frac{1}{\sqrt{2}}$$

$\because \langle 0||1\rangle = 0, \quad \langle 0||0\rangle = 1$

So, $\langle 0||q_0\rangle = \frac{1}{\sqrt{2}}$

Now we square both sides & take the absolute value to find the probability,

$$|\langle 0||q_0\rangle|^2 = \left|\frac{1}{\sqrt{2}}\right|^2 = 1/2$$

as expected 50%

* probability of a "quantum coin toss" can be calculated using the inner products.

So,
* The probability of getting $|x\rangle$ in superposition state $|\psi\rangle$ is

$$|\langle x|\psi\rangle|^2$$

## • Visual Representation of Qubits

### Bloch Sphere

- A generic qubit which is expressed as a superposition of $|0\rangle$ & $|1\rangle$, can be written as

$$|q\rangle = a|0\rangle + b|1\rangle$$

where, $|a|^2 + |b|^2 = 1$ is a requirement

- $|a|^2$ is the probability of measuring qubit in the state $|0\rangle$ & $|b|^2$ is the probability

of measuring qubit in the state $|1\rangle$.

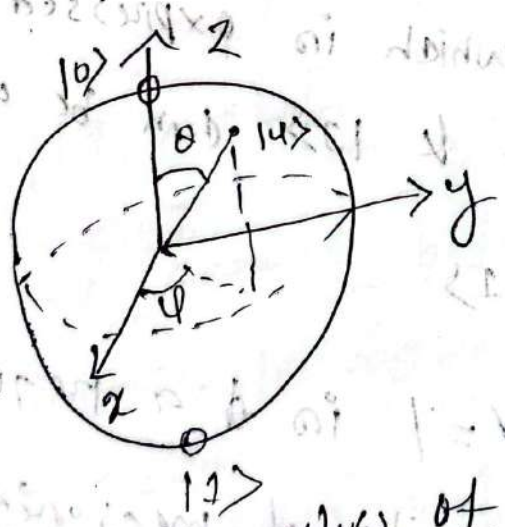
- Write $a = \cos(\theta/2)$
- Write $b = \sin(\theta/2) \times e^{i\phi}$

[$\phi$ are wanna include superposition with complex phases]

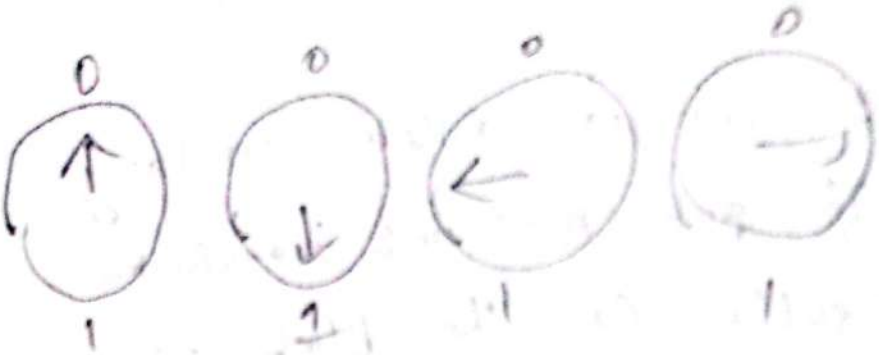
- the most general qubit which is superposition of $|0\rangle$ & $|1\rangle$

$$|q\rangle = \cos\left(\frac{\theta}{2}\right)|0\rangle + e^{i\phi}\sin\left(\frac{\theta}{2}\right)|1\rangle$$

- The qubit can be represented on a sphere called the Bloch sphere.




- different values of $\theta$ & $\phi$ give different superposition qubit state.



- Single Qubit Quantum Gates

We know that X gate on $|0\rangle$ will result in state $|1\rangle$, $X|0\rangle = |1\rangle$

$$|0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \quad |1\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix}.$$

This means that any quantum gate we apply is also a matrix. Multiplying the "gate matrix" with the state vector, gives us the output state.

So,

$$X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

$$X|0\rangle = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix} = |1\rangle$$

So, all gates can be written as the sum of matrices.

These gate matrices are also called unitary transformations. The gate matrices transform the state of the qubit. On the Bloch sphere these matrices change the orientation of the a... Since the Bloch sphere has radius of 1, the gate matrices that transforms the qubit states are called unitary transformations.

$$X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

$$Y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}$$

$$Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$$

$$H = \begin{bmatrix} 1/\sqrt{2} & 1/\sqrt{2} \\ 1/\sqrt{2} & -1/\sqrt{2} \end{bmatrix}$$

"Single qubit gates are unitary transformations"

$Y|0\rangle = i|1\rangle$
$Y|1\rangle = -i|1\rangle$
→ phase factors

→ if qubit 0 does nothing.
" " 1 $Z(|1\rangle) = -|1\rangle$

$$H|0\rangle = \begin{bmatrix} 1/\sqrt{2} & 1/\sqrt{2} \\ 1/\sqrt{2} & -1/\sqrt{2} \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

$$= \begin{bmatrix} 1/\sqrt{2} \\ 1/\sqrt{2} \end{bmatrix} = \frac{1}{\sqrt{2}} (|0\rangle + |1\rangle)$$

- Why does multiple qubit state/operation matter?

Typically in a real hardware quantum processor, we mostly implement quantum gates on one or two qubits. This gives us the ability to make quantum circuits & operations on a great number of qubits. This is possible using one & two qubit gates given to us by the hardware.

- Matrix representation of two qubits

Assuming we are talking about two qubits that are both in the spin up state, we can represent each of them as follow:

qubit_1 = $q_1$ = spin up = $|0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$

qubit_2 = $q_2$ = spin up = $|0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$

But imagine if we want to represent hundred of qubits each in a separate line! So we shrink this notation a bit by putting them beside each other..

qubit_1 qubit_2 = spin up spin up = $\uparrow\uparrow = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \otimes \begin{bmatrix} 1 \\ 0 \end{bmatrix}$

$00 = q_1 q_2 = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}$ $01 = q_1 q_2 = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}$

$10 = q_1 q_2 = \uparrow\downarrow = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}$ $11 = q_1 q_1 = \downarrow\downarrow = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$

And to describe the general state of two qubits we can store it in a 4D vector

$$\psi = \begin{bmatrix} a_{00} \\ a_{01} \\ a_{10} \\ a_{11} \end{bmatrix}$$

If we use ket notation;-

$00 = q_1 q_2 = \uparrow\uparrow = |0\rangle|0\rangle = |00\rangle$

$01 = q_1 q_2 = \uparrow\downarrow = |0\rangle|1\rangle = |01\rangle$

$10 = q_1 q_2 = \downarrow\uparrow = |1\rangle|0\rangle = |10\rangle$

$11 = q_1 q_1 = \downarrow\downarrow = |1\rangle|1\rangle = |11\rangle$

So we can represent a general 2 qubit states as follows;

$|a\rangle|b\rangle = |ab\rangle$

After mapping the matrix representation to Dirac notations we get;

$|00\rangle = \uparrow\uparrow = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}$ $|01\rangle = \uparrow\downarrow = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}$ $|10\rangle = \downarrow\uparrow = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}$

$|11\rangle = \downarrow\downarrow = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$

We can do this mapping for a general qubit states as follows:

$|a\rangle \otimes |b\rangle = |ab\rangle = \begin{bmatrix} a_1 b_1 \\ a_1 b_2 \\ a_2 b_1 \\ a_2 b_2 \end{bmatrix} = a_1 b_1 \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} +$

$$a_1 b_2 \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} + a_2 b_1 \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix} + a_2 b_2 \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

$$= a_1 b_1 |00\rangle + a_1 b_2 |01\rangle + a_2 b_1 |10\rangle + a_2 b_2 |11\rangle$$

where, $|a\rangle = \begin{bmatrix} a_1 \\ a_2 \end{bmatrix}$ & $|b\rangle = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}$

## L> Orthonormal basis set and inner prod

orthonormal basis set

$\{|00\rangle, |01\rangle, |10\rangle, |11\rangle\}$

Orthogonality can be tested by calculating the inner product of the vectors. inner prod of two general 2D vectors is,

$$\langle a|b\rangle = [a_1\langle 0| + a_2\langle 1|] \cdot [b_1|0\rangle + b_2|1\rangle]$$

$$= a_1 b_1 \langle 0|0\rangle + a_1 b_2 \langle 0|1\rangle + a_2 b_1 \langle 1|0\rangle + a_2 b_2 \langle 1|1\rangle$$

$$= a_1 b_1 \langle 0|0\rangle + a_2 b_2 \langle 1|1\rangle = [a_1\ a_2] \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}$$

$= a_1 b_1 + a_2 b_2$

where, $|a\rangle = \begin{bmatrix} a_1 \\ a_2 \end{bmatrix}$ & $|b\rangle = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}$

similarly the inner prod of 2 general 4-D vectors.

$$\langle a|b\rangle = a_1 b_1 \langle 00|00\rangle + a_1 b_2 \langle 01|01\rangle + a_2 b_1 \langle 10|10\rangle + a_4 b_4 \langle 11|11\rangle$$

$$= [a_1\ a_2\ a_3\ a_4] \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \end{bmatrix}$$

$$= a_1 b_1 + a_2 b_2 + a_3 b_3 + a_4 b_4$$

where, $|a\rangle = \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \end{bmatrix}$ & $|b\rangle = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \end{bmatrix}$

- $\langle 00|00\rangle = \langle 01|01\rangle = \langle 10|10\rangle = \langle 11|11\rangle = 1$
- $\langle 00|01\rangle = \langle 00|10\rangle = \langle 00|11\rangle = 0$
- $\langle 01|00\rangle = \langle 01|10\rangle = \langle 01|11\rangle = 0$
- $\langle 10|00\rangle = \langle 10|01\rangle = \langle 10|11\rangle = 0$
- $\langle 11|00\rangle = \langle 11|01\rangle = \langle 11|10\rangle = 0$

These conditions satisfy orthonormality of the above mentioned basis set

↳ superposition & vector decomposition

In general 2 qubit state,

$$|a\rangle \otimes |b\rangle = |ab\rangle = \begin{bmatrix} a_1 b_1 \\ a_1 b_2 \\ a_2 b_1 \\ a_2 b_2 \end{bmatrix}$$

Orthonormal 2 qubit basis sets $\{|00\rangle, |01\rangle, |10\rangle, |11\rangle\}$. So now let's try to decompose our general 2 qubit as a function of our two qubit state.

$$\psi = |a\rangle \otimes |b\rangle = |ab\rangle = \begin{bmatrix} a_1 b_1 \\ a_1 b_2 \\ a_2 b_1 \\ a_2 b_2 \end{bmatrix} = a_1 b_1 \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

$$+ a_1 b_2 \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} + a_2 b_1 \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix} + a_2 b_2 \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

$$= a_1 b_1 |00\rangle + a_1 b_2 |01\rangle + a_2 b_1 |10\rangle + a_2 b_2 |11\rangle$$

Here we tried to decompose a two qubit state into an orthonormal basis set. Clearly we can see that a general state can be written as a superposition of 2 qubit states,

$$\psi = a|00\rangle + b|01\rangle + c|10\rangle + d|11\rangle.$$

Similarly, coefficients here can be any real numbers that following the rule.

$$a^2 + b^2 + c^2 + d^2 = 1$$

& the state is in superposition.

## 4) Measurement of qubit state & measurement probability

When we want to measure the state of the qubit the mathematical operation that does this for us is the inner product.

$$\psi = a|00\rangle + b|01\rangle + c|10\rangle + d|11\rangle$$

Measuring this state could end up in any of

these states $|00\rangle, |01\rangle, |10\rangle, |11\rangle$ with the following probabilities & mathematical step:

$|\langle 00|\psi\rangle|^2 = |a\langle 00|00\rangle + b\langle 00|01\rangle + c\langle 00|10\rangle + d\langle 00|11\rangle|^2 = |a\langle 00|00\rangle|^2 = a^2$

$|\langle 01|\psi\rangle|^2 = |a\langle 01|00\rangle + b\langle 01|01\rangle + c\langle 01|10\rangle + d\langle 01|11\rangle|^2 = |b\langle 01|01\rangle|^2 = b^2$

$|\langle 10|\psi\rangle|^2 = |a\langle 10|00\rangle + b\langle 01|01\rangle + c\langle 10|10\rangle + d\langle 10|11\rangle|^2 = |c\langle 10|10\rangle|^2 = c^2$

$|\langle 11|\psi\rangle|^2 = |a\langle 11|00\rangle + b\langle 01|01\rangle + c\langle 10|10\rangle + d\langle 11|11\rangle|^2 = |d\langle 11|11\rangle|^2 = d^2$

L> tensor product of 2 1D vectors

$$|00\rangle = |0\rangle \otimes |0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \otimes \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

$\otimes$ is another product type in mathematics. This product is useful when you are dealing with more than one qubit. You can think of it this way: qubits are living in different worlds where they can't interact with each other. Once you want them to be interactive, you should put them in a new common world. To put qubits in this new world, you use the tensor product. Now that this new space hosts more qubits, it should be bigger in size. That's what you see in the matrix size. The same description applies when you want to operate on 2 qubit states using gates.

$$\begin{bmatrix} a \\ b \end{bmatrix} \otimes \begin{bmatrix} c \\ d \end{bmatrix} = \begin{bmatrix} a\begin{bmatrix} c \\ d \end{bmatrix} \\ b\begin{bmatrix} c \\ d \end{bmatrix} \end{bmatrix} = \begin{bmatrix} ac \\ ad \\ bc \\ bd \end{bmatrix}$$

$$\begin{bmatrix} a_1 & b_1 \\ c_1 & d_1 \end{bmatrix} \otimes \begin{bmatrix} a_2 & b_2 \\ c_2 & d_2 \end{bmatrix} = \begin{bmatrix} a_1\begin{bmatrix} a_2 & b_2 \\ c_2 & d_2 \end{bmatrix} & b_1\begin{bmatrix} a_2 & b_2 \\ c_2 & d_2 \end{bmatrix} \\ c_1\begin{bmatrix} a_2 & b_2 \\ c_2 & d_2 \end{bmatrix} & d_1\begin{bmatrix} a_2 & b_2 \\ c_2 & d_2 \end{bmatrix} \end{bmatrix}$$

- Single Qubit Gates on two Qubits

k Circuit Diagram

$X|0\rangle = |1\rangle$

$X|1\rangle = |0\rangle$

$X|a\rangle = \begin{bmatrix} b \\ a \end{bmatrix}$

Identity matrix does nothing on the qubit

$$I = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$I|0\rangle = |0\rangle \quad I|1\rangle = |1\rangle$

But it may not be clear how an X gate would act on a qubit in a two-qubit or generally a multiqubit vector. The rule is simple, just as we used the tensor product to calculate multiqubit state vectors, we use the tensor product to calculate matrices that act on these statevectors. For example consider a case where we want to apply an X gate to the 1st qubit & no gate to the 2nd qubit. This can mathematically be written as,

$$X^{(1)}|q_1 q_2\rangle = X^{(1)} \otimes I^{(2)} |q_1 q_2\rangle$$

$$= \left(X^{(1)}|q_1\rangle\right) \otimes \left(I^{(2)}|q_2\rangle\right) = \begin{bmatrix} b_1 \\ a_1 \end{bmatrix} \otimes \begin{bmatrix} a_2 \\ b_2 \end{bmatrix}$$

$$= \begin{bmatrix} b_1 a_2 \\ b_1 b_2 \\ a_1 a_2 \\ a_1 b_2 \end{bmatrix}$$

This is same as,

$$(X \otimes I)(|q_1\rangle \otimes |q_2\rangle) = \begin{bmatrix} 0\begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix} & 1\begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix} \\ 1\begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix} & 0\begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix} \end{bmatrix}$$

$$\begin{bmatrix} a_1 a_2 \\ a_1 b_2 \\ b_1 a_2 \\ b_1 b_2 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} a_1 a_2 \\ a_1 b_2 \\ b_1 a_2 \\ b_1 b_2 \end{bmatrix} = \begin{bmatrix} b_1 a_2 \\ b_1 b_2 \\ a_1 a_2 \\ a_1 b_2 \end{bmatrix}$$

So, if we want to apply a gate to only one qubit at a time (), we describe this using tensor product with the identity matrix,

Let's take another example. Now apply a Hadamard gate on the 1st qubit & an X gate on the 2nd one.

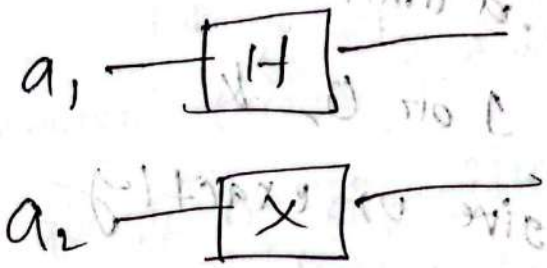
$$H^{(1)} \otimes X^{(2)} |a_1, a_2\rangle = H^{(1)}|a_1\rangle \otimes X|a_2\rangle$$

In matrix form,

$$H|0\rangle \otimes X^{(2)} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1\begin{bmatrix}0&1\\1&0\end{bmatrix} & 1\begin{bmatrix}0&1\\0&0\end{bmatrix} \\ 1\begin{bmatrix}0&1\\1&0\end{bmatrix} & -1\begin{bmatrix}0&1\\1&0\end{bmatrix} \end{bmatrix} \begin{bmatrix} a_1 a_2 \\ a_1 b_2 \\ b_1 a_2 \\ b_1 b_2 \end{bmatrix}$$

$$= \frac{1}{\sqrt{2}} \begin{bmatrix} 0&1&0&1 \\ 1&0&1&0 \\ 0&1&0&-1 \\ 1&0&-1&0 \end{bmatrix} \begin{bmatrix} a_1 - a_2 \\ a_1 b_2 \\ b_1 a_2 \\ b_1 b_2 \end{bmatrix} = \frac{1}{\sqrt{2}} \begin{bmatrix} a_1 b_2 + b_1 b_2 \\ a_2 b_1 + a_1 a_2 \\ a_1 b_2 - b_1 b_2 \\ a_1 a_2 - a_2 b_1 \end{bmatrix}$$

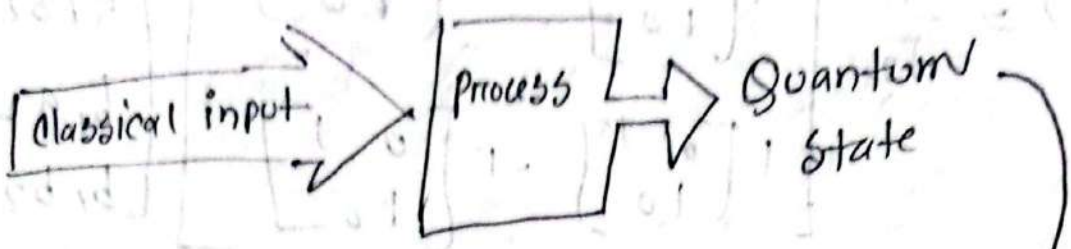
circuit diagram;




- Quantum Algorithms

Quantum computers have potential to solve certain problems faster than classical computers due to their use of quantum properties.

A general Quantum algorithm,




Some are

- The Deutsch Algorithm; which tells us if a Boolean (a true or false statement) is "constant" or "balanced". A constant statement would give us all either 1 or 0, & balanced function would give us exactly 50% 1s & 50% 0s.

- Simon's problem, which was the 1st algo to show us exponential speed up in computational time compared to the best classical algorithm. In simons problem we are given an unknown function

(called a Blackbox function) that can work in one of two ways. It can either be 1:1 function (have exactly one output for any input) or a 2:1 function, which has exactly one output for every 2 inputs:

- The Bernstein-Vazirani Algo: which is similar to the Deutsch Algorithm, except it utilizes a blackbox function with an unknown string instead of an unknown boolean.

. Quantum Oracle is a blackbox used extensively in quantum algorithms for the estimation of functions using qubits.

We often consider a quantum computer to behave like a blackbox, or an oracle - we don't care about the specifics of the circuit it implements, we just assume that it will perform some operation that we specify & return as an output in the form of qubits.

- Deutsch - Jozsa Problem

we are given a hidden Boolean function $f$, which takes an input a string of bits, and returns either 0 or 1, that is:

$$f(x_0, x_1, x_2, \ldots) \rightarrow 0 \text{ or } 1,$$ where $x_n$ is 0 or 1 ..

The property of the given Boolean function is that it is guaranteed to either be balanced or constant.

» constant function
return all 0's or all 1's for any input

» Balanced function
return 0's for exactly half of all inputs & 1's for the other half.

Our task is to determine whether the given function is balanced or constant.

1.2 The classical solution

→ Classically, in the best case two queries to the oracle can determine if the hidden Boolean function $f(x)$ is balanced: e.g if we get both $f(0,0,0,\ldots) \to 0$ & $f(1,0,0,\ldots) \to 1$ then we know the function is balanced as we have obtained the 2 different outputs.

In the worst case, if we continue to see the same output for each input, we will have to check exactly half of all possible inputs plus one in order to be certain that $f(x)$ is constant. Since the total number of possible inputs is $2^n$, this implies that we need $2^{n-1}+1$ trial inputs to be certain that $f(x)$ is constant in the worst case. for example, for a 4 bit string, if we checked 8 out of 16 possible combinations, getting all 0's it is still possible that the 9th input returns a 1 & $f(x)$ is balanced. Probabilistically, this is a very

unlikely event. In fact, if we get the same result continually in succession, we can express the probability that the function is constant as a function of k inputs as:

$$P_{constant}(k) = 1 - \frac{1}{2^{k-1}}$$

Realistically, we could opt to truncate our classical algorithm early, say if we were x% confident. But if we want to be 100% confident, we would need to check $2^{n-1}+1$ inputs.

## 1.3. quantum solution

Only 1 query of the function f is needed if we implement f as the quantum oracle that maps

$$|x\rangle|y\rangle \longrightarrow |x\rangle|y \oplus f(x)\rangle$$

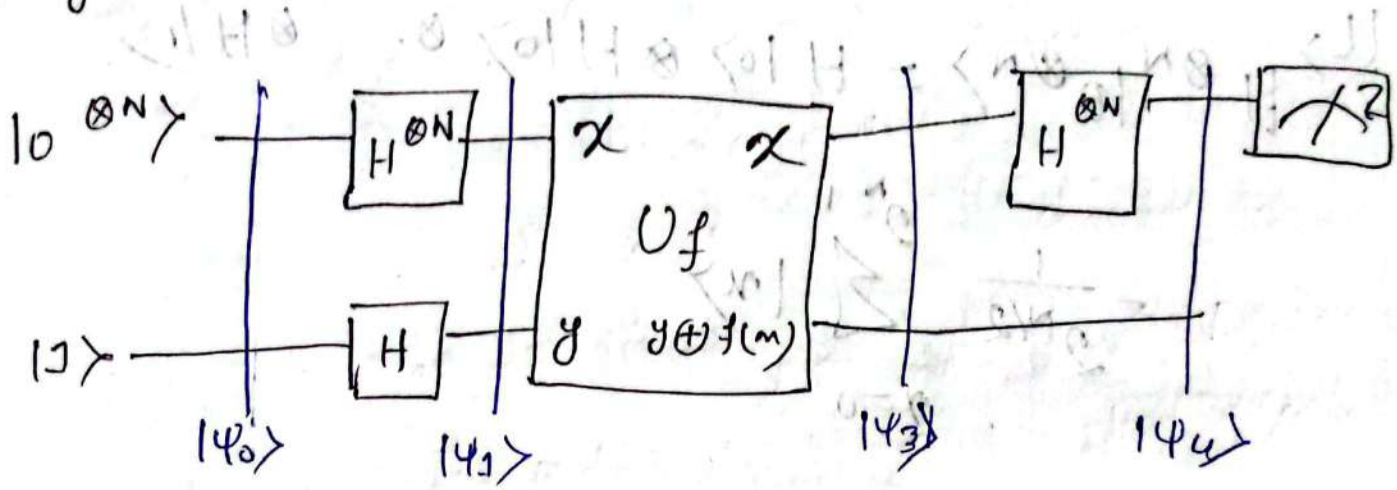
U_f

(⊕ : Addition modulo 2)

[ Addition modulo 2 :->

1. $0 \oplus 0 = 0$
2. $0 \oplus 1 = 1$
3. $1 \oplus 0 = 1$
4. $1 \oplus 1 = 0$ ]

The generic circuit,




$|\psi_0\rangle = |0^{\otimes N}\rangle \otimes |1\rangle$

$|\psi_1\rangle = H^{\otimes N} |0^{\otimes N}\rangle \otimes H \cdot |1\rangle$ side step to Hadamard Transform!

we know,

$H|0\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$ , $H|1\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$

**N=2**

$H^{\otimes 2}|0^{\otimes 2}\rangle = H|0\rangle \otimes H|0\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \otimes \frac{1}{\sqrt{2}}|0\rangle +$

2 bit adding

$$= \left(\frac{1}{\sqrt{2}}\right)^2 \left[ |00\rangle + |01\rangle + |10\rangle + |11\rangle \right]$$

$$= \left(\frac{1}{\sqrt{2}}\right)^2 \left[ |0\rangle + |1\rangle + |2\rangle + |3\rangle \right]$$

$$= \left(\frac{1}{\sqrt{2}}\right)^N \sum_{x=0}^{2^N-1} |x\rangle$$

$$|1\rangle \quad H^{\otimes N} |0^{\otimes N}\rangle = H|0\rangle \otimes H|0\rangle \otimes \cdots \otimes H|0\rangle$$

$$= \frac{1}{2^{N/2}} \sum_{x=0}^{2^N-1} |x\rangle$$

$$|\psi_1\rangle = H^{\otimes N} |0^{\otimes N}\rangle \otimes H|1\rangle$$

$$= \frac{1}{2^{N/2}} \sum_{x=0}^{2^N-1} |x\rangle \otimes \left(\frac{1}{\sqrt{2}} \left(|0\rangle - |1\rangle\right)\right)$$

$$|\psi_2\rangle = \frac{1}{\sqrt{2}} \frac{1}{2^{N/2}} \sum_{x=0}^{2^N-1} |x\rangle \left( |0\rangle \oplus f(x)\rangle - |1 \oplus f(x)\rangle \right)$$

$$= \frac{1}{\sqrt{2}} \frac{1}{2^{N/2}} \sum_{x=0}^{2^N-1} |x\rangle \Big( |f(x)\rangle - |1 \oplus f(x)\rangle \Big)$$

$$= \begin{cases} |0\rangle - |1\rangle & \text{if } f(x)=0 \\ -(|0\rangle - |1\rangle) & \text{if } f(x)=1 \end{cases}$$

$$= (-1)^{f(x)} [|0\rangle - |1\rangle]$$

$$= \frac{1}{2^{N/2}} \sum_{x=0}^{2^N-1} (-1)^{f(x)} |x\rangle \otimes \left( \frac{|0\rangle - |1\rangle}{\sqrt{2}} \right)$$

1st register

$|\psi_3\rangle$ = Hadamard to each qubit in first register

$$= H^{\otimes N} \left[ \frac{1}{2^{N/2}} \sum_{x=0}^{2^N-1} (-1)^{f(x)} |x\rangle \right] \otimes \left( \frac{|0\rangle - |1\rangle}{\sqrt{2}} \right)$$

First consider;

$$H^{\otimes N} |x\rangle = \frac{1}{2^{N/2}} \sum_{y=0}^{2^N-1} (-1)^{x \cdot y} |y\rangle$$

where, $x \cdot y = x_0 y_0 \oplus x_1 y_1 \oplus \cdots \oplus x_{N-1} y_{N-1}$

fx with $N=2$

$H^{\otimes 2}|x\rangle$ where $x$ can be :

| | $x_1$ | $x_0$ |
|---|---|---|
| 0 | 0 | 0 |
| 1 | 0 | 1 |
| 2 | 1 | 0 |
| 3 | 1 | 1 |

$$H \otimes H |2\rangle = H \otimes H |10\rangle = H|1\rangle \otimes H|0\rangle$$

$$= \left(\frac{1}{\sqrt{2}}\right)^2 [|00\rangle + |01\rangle + |10\rangle - |11\rangle]$$

$$= \left(\frac{1}{\sqrt{2}}\right)^2 [|0\rangle + |1\rangle - |2\rangle - |3\rangle]$$

$x \odot y$ when $N = 2$ & $x = 2$

$$\rightarrow x \odot y = x_0 y_0 \oplus x_1 y_1 = 0 y_0 \oplus 1 y_1$$

$$= 0 \oplus y_1 = y_1$$

$$\rightarrow \frac{1}{2^{N/2}} \sum_{y=0}^{2^N - 1} (-1)^{x \odot y} |y\rangle = \frac{1}{2^{N/2}} \sum_{y=0}^{3} (-1)^{y_1} |y\rangle$$

$$= \left(\frac{1}{\sqrt{2}}\right)^2 [|0\rangle + |1\rangle - |2\rangle - |3\rangle]$$

$$\Rightarrow H^{\otimes n}|x\rangle = \frac{1}{2^{n/2}} \sum_{y=0}^{2^n-1} (-1)^{x \cdot y} |y\rangle$$

$$H^{\otimes n}\left[\frac{1}{2^{n/2}} \sum_{x=0}^{2^n-1} (-1)^{f(x)}|x\rangle\right]$$

$$= \frac{1}{2^n} \sum_{x=0}^{2^n-1} (-1)^{f(x)} \left[\sum_{y=0}^{2^n-1} (-1)^{x \cdot y} |y\rangle\right]$$

$$= \frac{1}{2^n} \sum_{y=0}^{2^n-1} \left[\sum_{x=0}^{2^n-1} (-1)^{f(x)} (-1)^{x \cdot y}\right] |y\rangle$$

Measure the int register

probability to obtain outcome $|y=0\rangle = |0^{\otimes n}\rangle$

$$\left|\frac{1}{2^n} \sum_{x=0}^{2^n-1} (-1)^{f(x)} (-1)^{x \cdot 0}\right|^2 = \left|\frac{1}{2^n} \sum_{x=0}^{2^n-1} (-1)^{f(x)}\right|^2$$

$\Rightarrow f(x)$ is constant: $\left|\frac{1}{2^n} (-1)^{f(x)} \sum_{x=0}^{2^n-1}\right|^2 = \left|(-1)^{f(x)}\right|$

$$\left|\frac{2^n}{2^n}\right| = 1$$

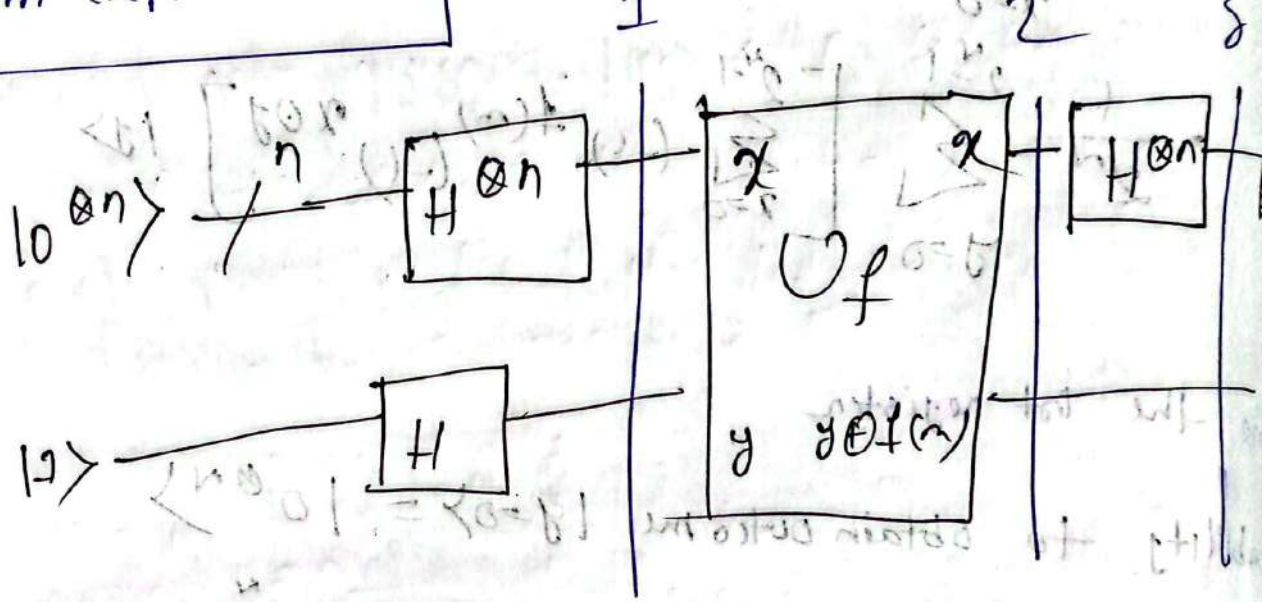
$\Rightarrow f(x)$ is balanced: $\left|\frac{1}{2^n}(\underbrace{1+1+1+\cdots}_{\text{half time}} \underbrace{-1-1-1-1\cdots}_{\text{half time}})\right|$

$= 0$

Thus if we measure the outcome $|y=0\rangle$ then $f(x)$ is CONSTANT.

Only other measurement result means that $f(x)$ is BALANCED

## Ibm explanation




1. Prepare 2 quantum registers. The 1st n qubit register initialized to $|0\rangle$ & 2nd one to $|1\rangle$

$$|\psi_0\rangle = |0\rangle^{\otimes n} |1\rangle$$

2. Apply a Hadamard gate to each

$$|\psi_1\rangle = \frac{1}{\sqrt{2^{n+1}}} \sum_{x=0}^{2^n-1} |x\rangle (|0\rangle - |1\rangle)$$

3. Apply the quantum oracle $|x\rangle|y\rangle$ to $|x\rangle|y \oplus f(x)\rangle$

$$|\psi_2\rangle = \frac{1}{\sqrt{2^{n+1}}} \sum_{x=0}^{2^n-1} |x\rangle \left( |f(x)\rangle - |1 \oplus f(x)\rangle \right)$$

$$= \frac{1}{\sqrt{2^{n+1}}} \sum_{x=0}^{2^n-1} (-1)^{f(x)} |x\rangle (|0\rangle - |1\rangle)$$

since for each $x$, $f(x)$ is either 0 or 1

4. At this point the 2nd qubit register may be ignored. Apply a Hadamard gate to each qubit in the 1st register:

$$|\psi_3\rangle = \frac{1}{2^n} \sum_{x=0}^{2^n-1} (-1)^{f(x)} \left[ \sum_{y=0}^{2^n-1} (-1)^{x \cdot y} |y\rangle \right]$$

$$= \frac{1}{2^n} \sum_{y=0}^{2^n-1} \left[ \sum_{x=0}^{2^n-1} (-1)^{f(x)} (-1)^{x \cdot y} \right] |y\rangle$$

where, $x \cdot y = x_0 y_0 \oplus x_1 y_1 \oplus \cdots \oplus x_{n-1} y_{n-1}$ is the sum of the bitwise product

5. Measure the 1st register. Notice that the probability of measuring $|0\rangle^{\otimes n} = \left| \frac{1}{2^n} \sum_{x=0}^{2^n-1} (-1)^{f(x)} \right|^2$ which evaluates to 1 if $f(x)$ is constant & 0 if

monitor (job)

f(x) is balanced:

- Constant oracle

When the oracle is constant, it has no effect (up to a global phase) on the input qubits, & the quantum states before & after querying the oracle are the same. Since the H gate is its own inverse, in step 4 we reverse step 2 to obtain the initial quantum state $|00\cdots0\rangle$ in the 1st register.

$$H^{\otimes n}\begin{bmatrix}1\\0\\\vdots\\0\end{bmatrix} = \frac{1}{\sqrt{2^n}}\begin{bmatrix}1\\1\\\vdots\\1\end{bmatrix} \xrightarrow[U_f]{\text{after}} H^{\otimes n}\frac{1}{\sqrt{2^n}}\begin{bmatrix}1\\1\\\vdots\\1\end{bmatrix} = \begin{bmatrix}1\\0\\0\\\vdots\\0\end{bmatrix}$$

- Balanced oracle.

After step 2, our input register is in equal superposition of all the states, i.e. the computational basis. When the oracle is

balanced, phase kickback adds a negative phase to exactly half these states

$$U_f \frac{1}{\sqrt{2^n}} \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix} = \frac{1}{\sqrt{2^n}} \begin{bmatrix} -1 \\ \vdots \\ 1 \end{bmatrix}$$

The quantum state after querying the oracle is orthogonal to the quantum state before querying the oracle. Thus, in step 4, when applying the H-gates, we must end up with a quantum state that is orthogonal to $|00\ldots0\rangle$. this means we should never measure the all-zero state.

- code

```
! pip install qiskit
! pip install qiskit-aer
```

Given: 1-bit function $f: \{0,1\} \to \{0,1\}$

function outputs:

| f | f(0) | f(1) |
|---|---|---|
| Constant-Zero | 0 | 0 |
| Constant-One | 1 | 1 |

```
from qiskit import Job_Monitor
```

Balanced : 1 : 0 1

" : not 1 : 0

problem: find if the function is Consistent or Balanced in minimum number of queries

A: All f(x) are equal f(0) = f(1)

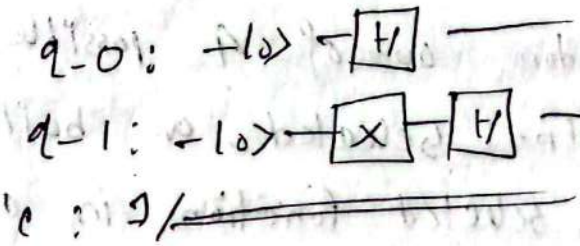
B: Equal num of 0's & 1's: f(0) ≠ f(1)

✓

```
import numpy as np
import matplotlib as mpl
import matplotlib.pyplot as plt
import time
import qiskit
from qiskit. import QuantumRegister, ClassicalRe
    QuantumCircuit, execute, Aer, IBMQ
from qiskit.compiler import transpile, assemble
from qiskit.tools.jupyter import *
from qiskit.visualization import *
```

Step 1 - Initialization: Create quantum circuit with initial state of for function input. Prepare equal superposition (apply Hadamard gate) of both qubits.

```
q = QuantumRegister(2, 'q')
c = ClassicalRegister(1, 'c')
circuit = QuantumCircuit(q, c)
circuit.reset(q)
circuit.x(q[1])
circuit.h([0,1])
circuit.draw()
```




We are given a black box that we know implements one of: constant-zero, constant-one, blanced-id, balanced-not. We are allowed to query the box. (i.e ask for f(x) for either x=0, 1) as many time as we need. How many queries are needed to determine with certainty which function the box implements?

Classical solution:

We must query twice. For example, if we query f(0) & recieve 0, we know we must have constant-zero or balanced-id, but we still can't pin down the

```
--monitor(job)
```

without querying 2nd time

Quantum Solution:

We can perform this task with only a single query using two qubits.

Step 2 - The Oracle : Apply the oracle circuit on the prepared initial states.

In this step we are selecting a function at random out of 4 possible options. Depending on function selected, we build the oracle circuit the selected function is constant zero. then nothing.

✓ function_names = ['constant_zero', 'constant_one', 'balanced_id', 'balanced_not']

oracle = function_names[np.random.randint(4)]

circuit.barrier()

if oracle == "constant_one":

    circuit.x(q[1])

elif oracle == 'balanced_id':

    circuit.cnot(q[0], q[1])

```
elif oracle == "balanced_not":
    circuit.x(q[0])
circuit.cnot(q[0], q[1])
```

At this moment we don't know which function was selected. To find out what function comprises the oracle, we perform measurement.

Measurement: We perform measurement on one of the qubits after applying a Hadamard gate.

```
✓ circuit.barrier()
  circuit.h(q[0])
# Apply measurement to q[0] & map it to c[0]
circuit.measure(q[0], c[0])

✓# Execute the circuit a single time to get the
  measurement outcome
backend = Aer.get_backend('qasm_simulator')
result = execute(circuit, backend, shots=1).result()
counts = result.get_counts()
print("Measurement output:", counts)

# checks the measurement results & states if the
  ...monitor(job)
```

...mport job_monitor

function is constant or balanced:

```
if list(counts.keys())[0] == '0':
    print("Measured qubit in state |0>, fo
          constant")
else:
    print("Measured qubit in state |1>, fo
          balanced")
```

we just needed to measure one qubit to fin
what type of function we have!
we can easily check if we were right. We ca
thing,

✓ `print("selected oracle was", oracle)`
`circuit.draw()`

Deutsch-Jozza Algo

Here we create a circuit to simulate Deuts
Algo with balanced f(n) output for 3 q

✓ `qreg_q = QuantumRegister(3, 'q')`
`creg_c = ClassicalRegister(3, 'c')`

```
circuit1 = QuantumCircuit(qreg_q, creg_c)

# Reset
circuit1.reset(qreg_q[0])
circuit1.reset(qreg_q[1])
circuit1.reset(qreg_q[2])

circuit1.h(qreg_q[0])
circuit1.h(qreg_q[1])
circuit1.h(qreg_q[2])

# oracle for balanced function
circuit1.barrier(qreg_q[0])
circuit1.barrier(qreg_q[1])
circuit1.barrier(qreg_q[2])
circuit1.z(qreg_c[0])
circuit1.barrier(qreg_q[0])
circuit1.h(qreg_q[2])
circuit1.cx(qreg_q[1], qreg_q[2])
circuit1.barrier(qreg_q[1])
circuit1.h(qreg_q[2])
circuit1.barrier(qreg_q[2])
circuit1.h(qreg_q[0]) # Applying to all qubits.
circuit1.h(qreg_q[1])
circuit1.h(qreg_q[2])
...-- monitor(job)
```

...ion import job_monitor

```
circuit.measure_all()
circuit.draw()

# qasm simulator backend
simulator = Aer.get_backend('qasm_simulator')
job = execute(circuit1, simulator, shots=50)
result = job.result()
print("Success?", result.success)
print("Status:", result.status)
counts = results.get_counts(circuit1)
plot_histogram(counts)

# statevector_simulator backend
simulator = Aer.get_backend('statevector_
                            simulator')
job = execute(circuit1, simulator, shots=50)
result = job.result()
counts = result.get_counts(circuit1)
plot_bloch_multivector(result.get_statevector())
```

# To run on actual Ibm computer

✓ !pip install qiskit-ibmq-provider

```
IBMQ.save_account("My API", overwrite=True)
IBMQ.load_account()
# Run on quantum computer
start_time = time.time()
from qiskit.providers.ibmq import least_busy
shots = 500
# Get the least busy backend
provider = IBMQ.get_backend(hub='ibm-q')
backend = least_busy(provider.backends(filters=
  lambda x: x.configuration().n_qubits >=2
and not x.configuration().simulator and
x.status().operational == True))

print("Least busy backend: ", backend)
job = execute(circuit1, backend = backend,
                shots = shots)
from qiskit.tools.monitor import job_monitor
job_monitor(job)
```

circuit m...

result = job.result()

plot_histogram(result.get_counts(

circuit))

- Exploring other quantum algo & applicati

↳ Quantum Annealer:

→ way of using intrinsic effects of quan

physics that help to solve certain types

like, Optimization problem, probability sa

there are 3 problems one can solve

D-waves Quantum Annealer

. Lattice Optimization Game

. Sudoku Solver

. Nurse scheduling

• fundamentals of Entanglement

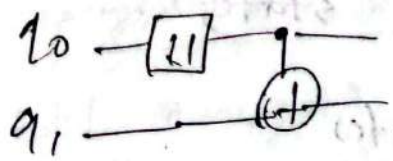
1> A 2 qubit gate : the CNOT gate ;-

The CNOT gate performs an X gate on the target qubit if the state of the control qubit is $|1\rangle$. If the state of the control qubit is $|0\rangle$, then the CNOT gate doesn't do anything to the target qubit,

$$CNOT(|0\rangle_c \otimes |0\rangle_t) = |0\rangle_c \otimes |0\rangle_t$$

$$CNOT[(|1\rangle_c \otimes |0\rangle_t) = |1\rangle_c \otimes X|0\rangle_t = |1\rangle_c \otimes |1\rangle_t$$

Things get more interesting when we apply the CNOT gate to some qubits that are in a superposition.




first, $\frac{(|0\rangle + |1\rangle)|0\rangle}{\sqrt{2}} = \frac{|00\rangle + |10\rangle}{\sqrt{2}}$

then CNOT $\frac{|00\rangle + |11\rangle}{\sqrt{2}} = |\Phi^+\rangle$

This is called bell state,

## ↳ Entangled states, Bell states

Bell state, $|\Phi^+\rangle = \frac{|00\rangle + |11\rangle}{\sqrt{2}}$

to compare which is not ~~bell state~~, entangled.

$$|\psi\rangle = \frac{|00\rangle + |01\rangle}{\sqrt{2}}$$

$$= \frac{|0\rangle \otimes |0\rangle + |0\rangle \otimes |1\rangle}{\sqrt{2}}$$

$$= \frac{|0\rangle \otimes (|0\rangle + |1\rangle)}{\sqrt{2}}$$

1st qubit — 2nd qubit

Entangled states can't be factored i
product of single-qubit states

There are 4 bell states

$$|\Phi^+\rangle = \frac{1}{\sqrt{2}} (|0\rangle_A \otimes |0\rangle_B + |1\rangle_A \otimes$$

$$|\Phi^-\rangle = \frac{1}{\sqrt{2}} (|0\rangle_A \otimes |0\rangle_B - |1\rangle_A \otimes$$

$$|\psi^+\rangle = \frac{1}{\sqrt{2}} (|0\rangle_A \otimes |1\rangle_B + |1\rangle_A \otimes |0\rangle_B)$$

$$|\psi^-\rangle = \frac{1}{\sqrt{2}} (|0\rangle_A \otimes |1\rangle_B - |1\rangle_A \otimes |0\rangle_B)$$

Entangled states are special from physical prospective bcz measurement of an entangled state will always be related to each other.

$$|\Phi^+\rangle = \frac{|00\rangle + |11\rangle}{\sqrt{2}}$$

Here, $|0\rangle$ for the 1st qubit only appears next to $|0\rangle$ for the 2nd qubit, & same for $|1\rangle$. The measurement result from the 1st qubit will always match the 2nd qubit.

- **Entanglement coding**

```
import numpy as np
import matplotlib as mpl
import matplotlib.pyplot as plt
import time
import qiskit
```

```
from qiskit import QuantumRegister, Cl
    QuantumCircuit, execute, Aer, IBMQ
from qiskit.compiler import transpile, asse
from qiskit.tools.jupyter import *
from qiskit.visualization import *
```

↳ $|\psi_0\rangle = \frac{1}{\sqrt{2}} (|00\rangle + |11\rangle)$

```
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()
```

### Bloch sphere

```
simulator = Aer.get_backend("statevector_simulator")
job = execute(circuit, simulator, shots=500)
result = job.result()
counts = result.get_counts(circuit)
plot_bloch_multivector(result.get_statevector())
```

$\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$

### qasm_simulator

```
simulator = Aer.get_backend("qasm_simulator")
job = execute(circuit, simulator, shots=1200)
result = job.result()
counts = result.get_counts(circuit)
print("\nTotal count for 00 & 11 are", counts)
plot_histogram(counts)
```

↳ $|\psi_1\rangle = \frac{1}{\sqrt{2}}(|10\rangle + |10\rangle)$

```
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.x(qreg-q[0])
circuit.measure(qreg-q[0], creg-c[0])
circuit.measure(qreg-q[1], creg-c[1])

↳ $|\psi_2\rangle = \frac{1}{\sqrt{2}}(|00\rangle - |11\rangle)$

→ line 1-7

circuit.z(qreg-q[1])

↳ $|\psi_3\rangle = \frac{1}{\sqrt{2}}(|01\rangle - |10\rangle)$

→ line 1-7

circuit.x(qreg-q[0])
circuit.z(qreg-q[1])

# running on actual IBM quantum computer

→ same as previous

If we compare the results of the histograms generated by the quantum computer with the histograms generated using just the simulator we get more than just the expected states in the hist above. This is bc2 of quantum noise that comes up in the quantum computer.

Suppose one of the "connect" results we expected to state "11", then the accuracy of the quantum comp can be calculated,

→ Ibm qe code so far

connected_results = result.get_counts(circuit)["11"]

accuracy = (connected_results/shots) * 100

print("Accuracy = %.2f%%" % accuracy)

time_for_run = (time.time() - start_time)/60

print("Time for run:", time_for_run)

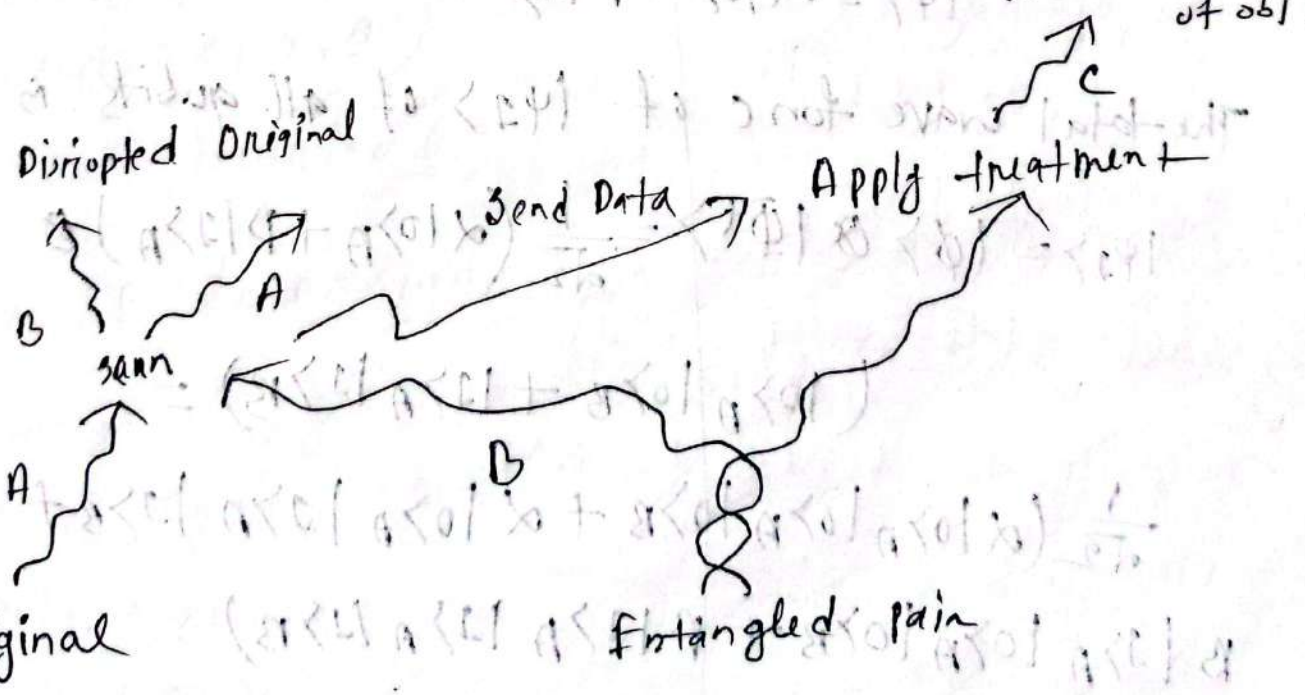
- Teleportation

Teleportation refers making an object or person disintegrate in one place while a perfect replica appears somewhere else. The general idea is the original object is scanned in such a way as to

all the info from it. Then this information is transmitted to the receiving location & used to construct the replica, not necessarily from the actual material of the original, but perhaps from atoms of the same kinds, arranged in exactly the same pattern as the original. A teleportation machine would be like a fax machine, except that it would work on 3-d objects as well as documents, it would produce an exact copy rather than an approximate facsimile, & it would destroy the original in the process of scanning it. Teleportation promises to be quite useful as an information processing primitive, facilitating long range quantum communication, and making it much easier to build a working quantum computer.

The idea of teleportation was not taken very seriously by scientists, as it was thought to violate the uncertainty principle of quantum mechanics, which forbids any measuring or scanning process from extracting all the info in an atom or other object. According to the

uncertainty principle, the more accurately an object is scanned, the more it is distributed by the scanning process, until one reaches a point where the object is original state has been completely disrupted, still without having extracted enough info to make a perfect replica. this sounds like a solid argument against ~~the~~ teleportation. But 6 scientists found a way to make an end around this logic, using a celebrated & paradoxical feature of quantum mechanics known as the Einstein-Podolsky-Rosen effect. In brief, they found a way to scan out part of the info from an object A, which one wishes to teleport, while causing the remaining, unscanned, part of the info to pass, via the Einstein-Podolsky-Rosen effect, into another object C which has never been in contact with A.



Later, by applying to C a treatment depending on scanned-out information, it is possible to make into exactly the same state as A was in before was scanned. A itself is no longer in that having been thoroughly disrupted by the scan so what has been achieved is teleportation. Not

## • Quantum Teleportation-Code

1. Let us Assume that Alice & Bob each have from a shared entangled pair whose wav

$$|\Phi^+\rangle = \frac{1}{\sqrt{2}} (|0\rangle_A |0\rangle_B + |1\rangle_A |1\rangle_B)$$

Alice wants to teleport another qubit in an u quantum state $|\phi\rangle$ to Bob

$$|\phi\rangle = \alpha|0\rangle + \beta|1\rangle$$

The total wave fun c of $|\psi_1\rangle$ of all qubits is

$$|\psi_1\rangle = |\phi\rangle \otimes |\Phi^+\rangle = \frac{1}{\sqrt{2}} (\alpha|0\rangle_A + \beta|1\rangle_A) \otimes$$

$$(|0\rangle_A |0\rangle_B + |1\rangle_A |1\rangle_B) =$$

$$\frac{1}{\sqrt{2}} (\alpha|0\rangle_A |0\rangle_A |0\rangle_B + \alpha|0\rangle_A |1\rangle_A |1\rangle_B +$$

$$\beta|1\rangle_A |0\rangle_A |0\rangle_B + \beta|1\rangle_A |1\rangle_A |1\rangle_B)$$

✓ from qiskit import *

#QUBIT ORDERING

# q0 = state |psi> that we want to teleport

# q1 = Alice's half of the Bell pair

# q2 = Bob's half of the Bell pair, the destination of the teleportation

# step 0: Create the state to be teleported in qubit 0

circuit.x(0) # qubit 0 is now in state |1>, & thus in the state that we wanna teleport

circuit.barrier()

# step 1: create an entangled Bell pair btw Alice & Bob (qubits 1 & 2)

circuit.h(1)

circuit.cx(1,2)

circuit.barrier()

# step 2: Alice applies a series of operations between the state to teleport (q-0) & her half of the Bell pair (q-1)

circuit.cx(0,1)

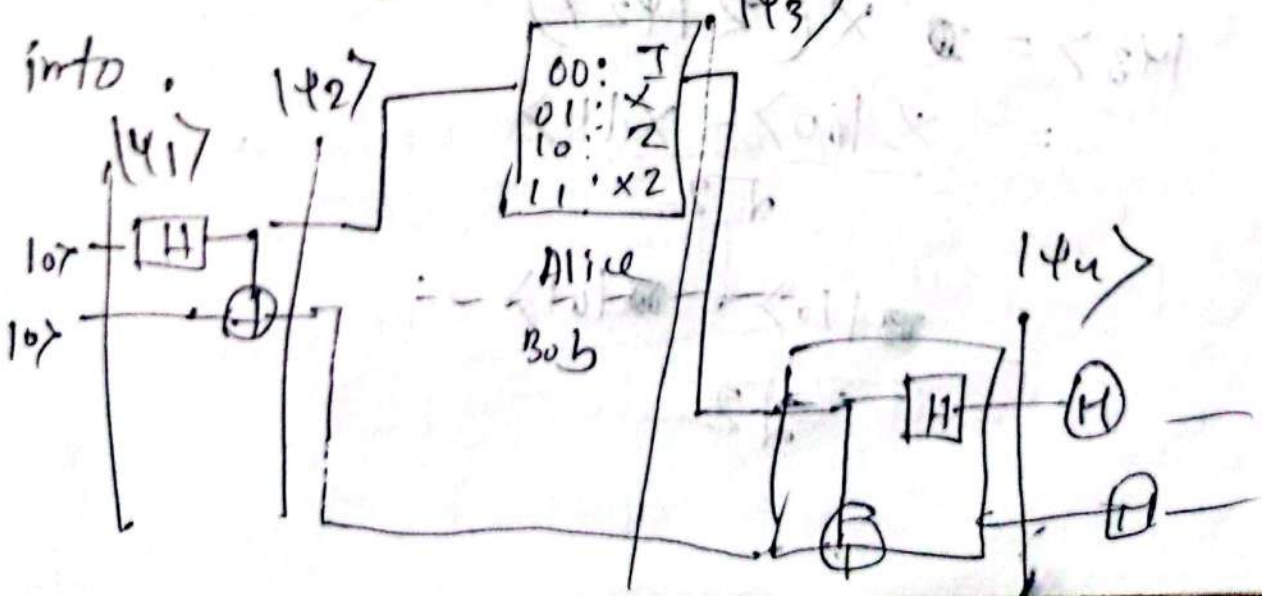
circuit.h(0)

```
circuit.barrier()

# Step 3: Alice measures both qubits 0 & 1
circuit.measure([0,1],[0,1]) # bits stored in
                              classical bits 0, 1
circuit.barrier()

# Step 4: Now that Alice has measured the
  their states have collapsed to 0 & 1!
# Bob can do operations conditioned on the
  qubits to his half of the Bell pair
# Note that while we're conditioning Bob
  operation on the collapsed qubits 0 & 1, we
  do teleportation over long distance by trans
  the classical info in classical bits 0 &1.
  circuit.cx(1,2)
  circuit.cz(0,2)

# step 5: Measure Bob's qubit
  circuit.measure([2],[2])
```

```
%matplotlib inline
circuit_draw(output='mpl')
```

## • Superdense coding

Quantum teleportation is the process by which the state of qubit $|\psi\rangle$ can be transmitted from one location to another, using two bits of classical info communication & a Bell pair. In other words, we can say it is a protocol that destroys the quantum state of a qubit in one location & recreates it on a qubit at a distant location, with the help of shared entanglement. Superdense coding is a procedure that allows someone to send two classical bits to another party using just a single qubit of info.



$$|\psi_1\rangle = |00\rangle$$

$$|\psi_2\rangle = \frac{|00\rangle + |11\rangle}{\sqrt{2}}$$

→ If Alice wishes to send 00,

$$|\psi_3\rangle = i\,\frac{|00\rangle + |11\rangle}{\sqrt{2}}$$

→ 01

$$|\psi_3\rangle = X|\psi_2\rangle$$

$$= \frac{|10\rangle + |01\rangle}{\sqrt{2}}$$

→ 10

$$|\psi_3\rangle = Z|\psi_2\rangle$$

$$= \frac{|00\rangle - |11\rangle}{\sqrt{2}}$$

→ 11

$$|\psi_3\rangle = [illegible] \times (Z \cdot |\psi_2\rangle)$$

$$= \frac{X|00\rangle - X|11\rangle}{\sqrt{2}}$$

$$= \frac{[illegible]|10\rangle - [illegible]|01\rangle}{\sqrt{2}}$$

→ Now Alice will send her portion ~~to Bob~~ of bell state to Bob

→ If 00,

$|\psi_4\rangle$

$$= \text{CNOT}\left(\frac{|00\rangle + |11\rangle}{\sqrt{2}}\right) \quad \rightarrow \text{Applying } H$$

$$= \frac{|10\rangle + |01\rangle}{\sqrt{2}} = \frac{|0\rangle + |1\rangle}{\sqrt{2}}|0\rangle + \frac{|0\rangle - |1\rangle}{2}|0\rangle$$

$$+ \frac{|00\rangle + |10\rangle + |00\rangle - |10\rangle}{2}$$

$$= \frac{2|00\rangle}{2} = |00\rangle$$

→ If 01,

$|\psi_4\rangle$

$$= H\left(\frac{|11\rangle + |01\rangle}{\sqrt{2}}\right)$$

$$= \frac{|0\rangle - |1\rangle}{2}|1\rangle + \frac{|0\rangle + |1\rangle}{2}|1\rangle$$

$$= \frac{|01\rangle - |11\rangle + |01\rangle + |11\rangle}{2} = |01\rangle$$

If $|0\rangle$,

$|\psi_4\rangle = |10\rangle$

If $|11\rangle$

$|\psi_4\rangle = |11\rangle$

- Superdense coding code

```
from qiskit import QuantumCircuit
from qiskit import IBMQ, Aer, transpi
from qiskit.visualization import plot_

def create_bell_pair():
    qc = QuantumCircuit(2)
    qc.h(1)
    qc.cx(1,0)
    return qc

def encode_message(qc, qubit, msg):
    """ Encodes a 2-bit msg on qc usin
    the superdense coding protocol
    Args:
        qc : Circuit to encode msg on
        qubit(int): Which qubit to add the
                                    to
```

msg (str) : Two-bit msg to send

Returns :

Quantum Circuit : Circuit that, when decoded, will produce msg

Raise :

ValueError if msg is wrong length or contains invalid characters

"""

if len(msg) != 2 or not set(msg).issubset({"0", "1"}):

raise ValueError(f"message '{msg}' is invalid")

if msg[1] == "1":

qc.x(1qbit)

if msg[0] == "1":

qc.z(1qbit)

return qc

```
def decode_message(qc):
    qc.cx(1,0)
    qc.h(1)
    return qc

# Charlie creates the entangled pair b/w
qc = create_bell_pair()

# We'll add a barrier
qc.barrier()
# At this point qubit 0 goes to Alice & qu
# goes to Bob
# Next, Alice encodes her msg onto qubi
# In this case.
# We wanna send msg '10'
message = '10'
qc = encode_message(qc, 1, message)
qc.barrier()
# Alice then sends her qubit to Bob
# After receiving qubit 0, Bob applies reco
protocol
```

```
qc = decode_message(qc)
qc.measure_all()
qc.draw()
# visualizing
aer_sim = Aer.get_backend('aer_simulator')
result = aer_sim.run(qc).result()
counts = result.get_counts(qc)
print(counts)
plot_histogram(counts)
```

- Environmental & societal impact

  - Climate modeling & weather forecasting

    - Quantum computers has the potential to greatly improve climate modeling & weather forecasting by increasing the capability of solving complex fluid dynamics-based simulations

  - Energy grids

    Quantum annealing can be used to optimize the scheduling & dispatching of resources to match

supply & demand. Additionally quantum Machine learn techniques can be used to analyze large amount of data generated by the grid & identify patte that can help operators make more informed decisions. Finally, quantum computing can also he designing & optimizing the energy storage syste which is becoming an imp part of the grid management

## L, Quantum Chemistry

with quantum computers, researchers can simulate th electronic structure & dynamics of molecules much greater accuracy & speed than classical com This can lead to the discovery of new mater with optimized properties for photovoltaic & battery technologies, allowing for more efficient energy production & storage. Furthermore, quan computing can also improve strategies for climate change mitigation, such as carbon capture.

- Application in Healthcare sector

1. Drug Discovery
2. Medical imaging
3. Personalized Medicine
4. Genomics
5. Optimization of Healthcare Process

- Applications in machine learning

1. Improved portfolio optimization
2. Enhanced machine learning (for more accurate predictions of market trends & behavior)
3. faster transaction processing
4. Improved encryption & security
5. Option pricing

- Applications in the Travel & Transportation sector

1. Optimizing route planning
2. Improved traffic flow
3. Enhancing logistics & supply chain management
4. Improving safety & security

- Some notable companies

1. IBM Quantum
2. Google Quantum AI
3. Rigetti Computing
4. Microsoft Quantum
5. Honeywell Quantum Solutions
6. D-wave systems
7. IonQ
8. PsiQuantum
9. Xanadu
10. Zapata Computing