Quantum Circuits and Quantum Programming

CPSC 4470/5470

Introduction to Quantum Computing

Instructor: Prof. **Yongshan Ding**

Computer Science, Applied Physics, Yale Quantum Institute

Mathematical Model of Quantum Computing

Four Principles to model quantum systems mathematically:

1. **Superposition:**
   The state of a qubit is a normalized complex vector in the two-dimensional Hilbert Space.
2. **Composition:**
   The joint state of many (independent) quantum systems is the tensor product of component states.
3. **Transformation:** (More in Lecture 5)
   Time evolution of a quantum system is a unitary process.
4. **Measurement:** (More in Lecture 6)
   Readout information from a quantum state causes the superposition state to collapse/project to one of its basis states randomly.

Richard Feynman: "*What I cannot create, I do not understand.*"

Programming Model of Quantum Computing

Quantum Programming:

1. **Base programming language:**
   E.g., Python
2. **Library interface and specifications:**
   Types (quantum/classical registers), methods (initializations, gates, measurements), constants
3. **Execution model:**
   Ensures the proper execution of quantum programs
   (Based on quantum circuit model.)
4. **Backend:**
   Classical Simulators or Quantum Devices

```python
def testCircuit():
    qc = QuantumCircuit(2,2)
    qc.h(0)
    qc.s(1)
    qc.cx(0,1)
    qc.measure([0],[0])
    # print(qc)
    outcome = qc.simulate()
```

Principle #1 – Superposition State

The **state of a qubit** is a two-dimensional complex vector in the **Hilbert Space** $\mathcal{H}$, described by two complex numbers, $\alpha, \beta \in \mathbb{C}$, satisfying that its 2-norm: $|\alpha|^2 + |\beta|^2 = 1$. In the Dirac notation:

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle = \begin{bmatrix} \alpha \\ \beta \end{bmatrix} \in \mathcal{H}$$

Defining and addressing qubits:

```
class Qubit(object):
    """Qubit object"""
    def __init__(self, arg, label='q'):
        super(Qubit, self).__init__()
        self.arg = arg
        self.label = label
```

arg: index of the qubit; **label**: name of the qubit

Initialize qubits: [q0, q1, q2, q3, q4]

```
qubit_array = [Qubit(i) for i in range(5)]
```

Principle #1 – Superposition State

The **state of a qubit** is a two-dimensional complex vector in the **Hilbert Space** $\mathcal{H}$, described by two complex numbers, $\alpha, \beta \in \mathbb{C}$, satisfying that its 2-norm: $|\alpha|^2 + |\beta|^2 = 1$. In the Dirac notation:

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle = \begin{bmatrix}\alpha \\ \beta\end{bmatrix} \in \mathcal{H}$$

Defining and addressing qubits:

```
class Qubit(object):
    """Qubit object"""
    def __init__(self, arg, label='q'):
        super(Qubit, self).__init__()
        self.arg = arg
        self.label = label
        self.state = np.array([1, 0], dtype=complex)
```

arg: index of the qubit; **label**: name of the qubit

state: numpy array to store the qubit's state vector. (optional: needed for classical simulator)

Initialized to $|0\rangle = \begin{bmatrix}1 \\ 0\end{bmatrix}$. Does it work for multiple qubits?

Principle #2 – Composition

Tensor Product and Entanglement

A **Quantum Register** for tracking multiple qubits (**size**: number of qubits):

```
class QuantumRegister(object):
    """QuantumRegister is where we keep track of qubits"""
    def __init__(self, num_q, label='qreg'):
        super(QuantumRegister, self).__init__()
        self.size = num_q
        self.label = label
        self.array = [Qubit(i) for i in range(num_q)]
```

The joint system is in $\mathcal{H}_{AB} = \mathcal{H}_A \otimes \mathcal{H}_B$.

E.g., for two separable qubits: $|\psi_A\rangle \in \mathcal{H}_A$ and $|\psi_B\rangle \in \mathcal{H}_B$:

$$|\psi_{AB}\rangle = |\psi_A\rangle \otimes |\psi_B\rangle = \begin{bmatrix}\alpha_0\\\alpha_1\end{bmatrix} \otimes \begin{bmatrix}\beta_0\\\beta_1\end{bmatrix} = \begin{bmatrix}\alpha_0\begin{bmatrix}\beta_0\\\beta_1\end{bmatrix}\\\alpha_1\begin{bmatrix}\beta_0\\\beta_1\end{bmatrix}\end{bmatrix} = \begin{bmatrix}\alpha_0\beta_0\\\alpha_0\beta_1\\\alpha_1\beta_0\\\alpha_1\beta_1\end{bmatrix}$$

Ordered array of qubits: [q0, q1, q2, q3, q4]

Joint state: q0.state $\otimes$ q1.state $\otimes \cdots \otimes$ q4.state

Why does ordering matter?

Example: $|\psi_{AB}\rangle = \begin{bmatrix}1\\0\end{bmatrix} \otimes \begin{bmatrix}1/\sqrt{2}\\1/\sqrt{2}\end{bmatrix} = \begin{bmatrix}1/\sqrt{2}\\1/\sqrt{2}\\0\\0\end{bmatrix}$. $|\psi_{BA}\rangle = \begin{bmatrix}1/\sqrt{2}\\1/\sqrt{2}\end{bmatrix} \otimes \begin{bmatrix}1\\0\end{bmatrix} = \begin{bmatrix}1/\sqrt{2}\\0\\1/\sqrt{2}\\0\end{bmatrix}$.

Principle #2 – Composition

Tensor Product and Entanglement

What about: $\frac{1}{\sqrt{2}}|00\rangle + \frac{1}{\sqrt{2}}|11\rangle = \begin{bmatrix} 1/\sqrt{2} \\ 0 \\ 0 \\ 1/\sqrt{2} \end{bmatrix}$? It belongs to $\mathcal{H}_A \otimes \mathcal{H}_B$, but no longer separable: ~~$|\psi_A\rangle \otimes |\psi_B\rangle$~~.

"Entangled state"

We need to track the **joint state** of the entire quantum register together, not qubit by qubit!

Entangled state or **product state**?

- $|000\rangle$
- $\frac{1}{2}|00\rangle + \frac{1}{2}|01\rangle + \frac{1}{2}|10\rangle + \frac{1}{2}|11\rangle$
- $\frac{1}{2}|00\rangle + \frac{1}{2}|01\rangle + \frac{1}{2}|10\rangle - \frac{1}{2}|11\rangle$
- $\frac{3}{5}|00\rangle - \frac{\sqrt{6}}{5}|01\rangle + \frac{\sqrt{6}}{5}|10\rangle - \frac{2}{5}|11\rangle$

```python
class Qubit(object):
    """Qubit object"""
    def __init__(self, arg, label='q'):
        super(Qubit, self).__init__()
        self.arg = arg
        self.label = label

class QuantumRegister(object):
    """QuantumRegister is where we keep track of qubits"""
    def __init__(self, num_q, label='qreg'):
        super(QuantumRegister, self).__init__()
        self.size = num_q
        self.label = label
        self.array = [Qubit(i) for i in range(num_q)]
        self.state = np.array([1] + [0] * (2 ** num_q - 1), dtype=complex)
```

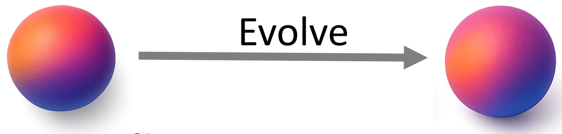
Principle #3 – Transformation

Unitary Evolution

A quantum state evolves by **unitary transformation**:

$$|\psi_1\rangle = U|\psi_0\rangle$$

(Unitary matrix: $U^{-1} = U^{\dagger}$)



$|\psi_0\rangle = \begin{bmatrix}\alpha \\ \beta\end{bmatrix}$ $\qquad$ $|\psi_T\rangle = \begin{bmatrix}\gamma \\ \delta\end{bmatrix}$

Assuming it evolves for T discrete steps:

$$|\psi_T\rangle = U_T \ldots U_2 U_1 |\psi_0\rangle$$

← time

Input state $\quad$ Gate #1 $\quad$ Gate #2 $\quad$ Output state

Our first (sequential) quantum circuit: $|0\rangle$ — H — S — $|\psi\rangle = ?$ $\quad$ $HS|0\rangle$ or $SH|0\rangle$?

time →

Hadamard gate:

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

Phase gate:

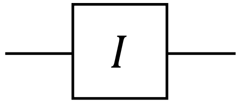
$$S = \begin{bmatrix}1 & 0 \\ 0 & i\end{bmatrix}$$

Multiplying unitary matrices to the state:

$$|\psi\rangle = SH|0\rangle = \begin{bmatrix}1 & 0 \\ 0 & i\end{bmatrix}\frac{1}{\sqrt{2}}\begin{bmatrix}1 & 1 \\ 1 & -1\end{bmatrix}\begin{bmatrix}1 \\ 0\end{bmatrix} = \begin{bmatrix}1 & 0 \\ 0 & i\end{bmatrix}\begin{bmatrix}1/\sqrt{2} \\ 1/\sqrt{2}\end{bmatrix} = \begin{bmatrix}1/\sqrt{2} \\ i/\sqrt{2}\end{bmatrix}$$

More Single-Qubit Gates

NOP (no-op)



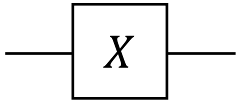
Identity matrix:

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

For any $|\psi\rangle$, $I|\psi\rangle = |\psi\rangle$

$I = I^\dagger = I^{-1}$

NOT (bit flip)



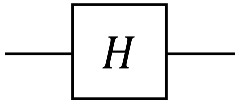
Pauli X matrix:

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

- $X|0\rangle = |1\rangle$
- $X|1\rangle = |0\rangle$
- $X(\alpha|0\rangle + \beta|1\rangle) = \alpha X|0\rangle + \beta X|1\rangle = \alpha|1\rangle + \beta|0\rangle$

$X = X^\dagger = X^{-1}$

Hadamard



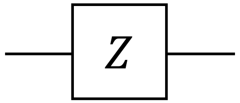
Hadamard matrix:

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

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

$H = H^\dagger = H^{-1}$

Phase flip



Pauli Z matrix:

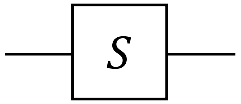
$\sigma_Z = Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$

- $Z|0\rangle = |0\rangle$
- $Z|1\rangle = -|1\rangle$
- $Z(\alpha|0\rangle + \beta|1\rangle) = \alpha Z|0\rangle + \beta Z|1\rangle = \alpha|0\rangle - \beta|1\rangle$

$Z = Z^\dagger = Z^{-1}$

More Single-Qubit Gates

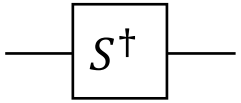
Phase gate



Phase matrix:
$S = \begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix}$

- $S|0\rangle = |0\rangle$
- $S|1\rangle = i|1\rangle$
- $S(\alpha|0\rangle + \beta|1\rangle) = \alpha S|0\rangle + \beta S|1\rangle = \alpha|0\rangle + i\beta|1\rangle$

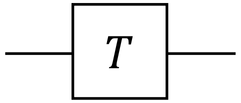
$Z = S^2$



$S^\dagger = \begin{bmatrix} 1 & 0 \\ 0 & -i \end{bmatrix}$

- $S^\dagger(\alpha|0\rangle + \beta|1\rangle) = \alpha|0\rangle - i\beta|1\rangle$

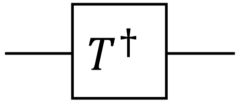
T gate



T matrix:
$T = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\pi/4} \end{bmatrix}$

- $T(\alpha|0\rangle + \beta|1\rangle) = \alpha|0\rangle + e^{i\pi/4}\beta|1\rangle$

$Z = S^2 = T^4$

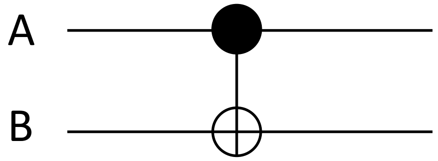


$T^\dagger = \begin{bmatrix} 1 & 0 \\ 0 & e^{-i\pi/4} \end{bmatrix}$

- $T^\dagger(\alpha|0\rangle + \beta|1\rangle) = \alpha|0\rangle + e^{-i\pi/4}\beta|1\rangle$

Interaction between Two Qubits

Controlled-X / CX / CNOT gate:



“Quantum if-else”:

- If A is 0: do nothing.
- If A is 1: flip B.

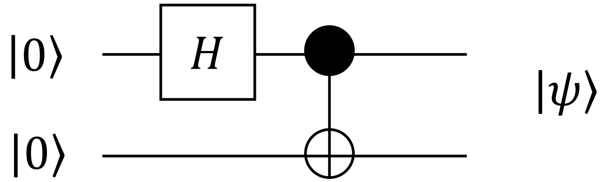
$$\mathrm{CX} = \begin{array}{c|cccc} & 00 & 01 & 10 & 11 \\ \hline 00 & 1 & 0 & 0 & 0 \\ 01 & 0 & 1 & 0 & 0 \\ 10 & 0 & 0 & 0 & 1 \\ 11 & 0 & 0 & 1 & 0 \end{array}.$$

Examples:

$$CX_{0,1}|00\rangle = |00\rangle$$
$$CX_{0,1}|01\rangle = |01\rangle$$
$$CX_{0,1}|10\rangle = |11\rangle$$
$$CX_{0,1}|11\rangle = |10\rangle$$

$$CX_{0,1}(0.6|00\rangle + 0.8|10\rangle) = 0.6|00\rangle + 0.8|11\rangle$$

A two-qubit quantum program:

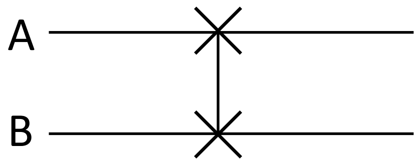


$$|\psi\rangle = CX_{0,1}H_0|00\rangle$$
$$= CX_{0,1}(|+\rangle \otimes |0\rangle)$$
$$= \frac{|00\rangle + |11\rangle}{\sqrt{2}}$$

“Entangling gate”

Interaction between Two Qubits

SWAP gate:



FROM

TO	00	01	10	11
00	1	0	0	0
01	0	0	1	0
10	0	1	0	0
11	0	0	0	1

Swapping A and B:

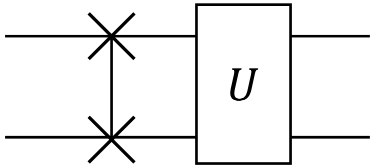
- If separable state:

$$\mathrm{SWAP}_{0,1}(|\psi_A\rangle \otimes |\psi_B\rangle) = |\psi_B\rangle \otimes |\psi_A\rangle$$

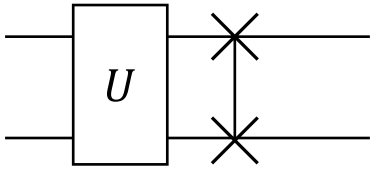
- If entangled state:

$$\mathrm{SWAP}_{0,1}(|00\rangle) = |00\rangle$$
$$\mathrm{SWAP}_{0,1}(|01\rangle) = |10\rangle$$
$$\mathrm{SWAP}_{0,1}(|10\rangle) = |01\rangle$$
$$\mathrm{SWAP}_{0,1}(|11\rangle) = |11\rangle$$

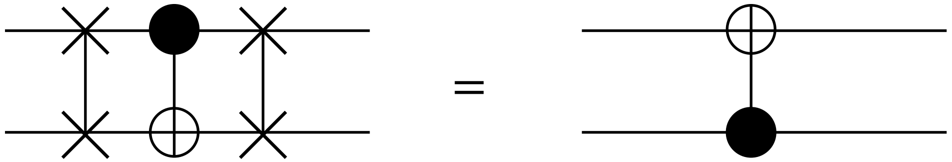
<u>Derive on board:</u> What are the following transformations?



$$U \cdot \mathrm{SWAP}_{0,1} = \begin{bmatrix} u_{00} & u_{02} & u_{01} & u_{03} \\ u_{10} & u_{12} & u_{11} & u_{13} \\ u_{20} & u_{22} & u_{21} & u_{23} \\ u_{30} & u_{32} & u_{31} & u_{33} \end{bmatrix}.$$



$$\mathrm{SWAP}_{0,1} \cdot U = \begin{bmatrix} u_{00} & u_{01} & u_{02} & u_{03} \\ u_{20} & u_{21} & u_{22} & u_{23} \\ u_{10} & u_{11} & u_{12} & u_{13} \\ u_{30} & u_{31} & u_{32} & u_{33} \end{bmatrix}.$$



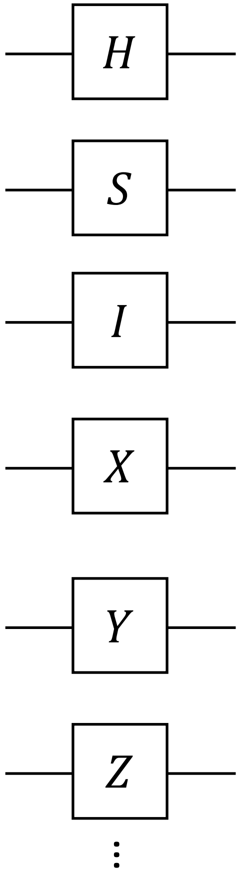
$$CX_{0,1} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}. \Rightarrow CX_{1,0} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}.$$

Instruction Set

```python
class Gate(object):
    """Gate object to describe its name, kind, and matrix"""
    def __init__(self, name, num_q, matrix):
        super(Gate, self).__init__()
        self.name = name
        self.num_q = num_q
        self.matrix = matrix
```

```python
# Define Gate by its name, kind (number of qubit), and matrix
HGate = Gate('h', 1, 1/np.sqrt(2) * np.array([[1,1],[1,-1]], dtype=complex))
```

Single-qubit gates:



Hadamard gate:

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

Phase gate:

$$S = \begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix}$$

Idle gate:

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

Pauli X gate (NOT):

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

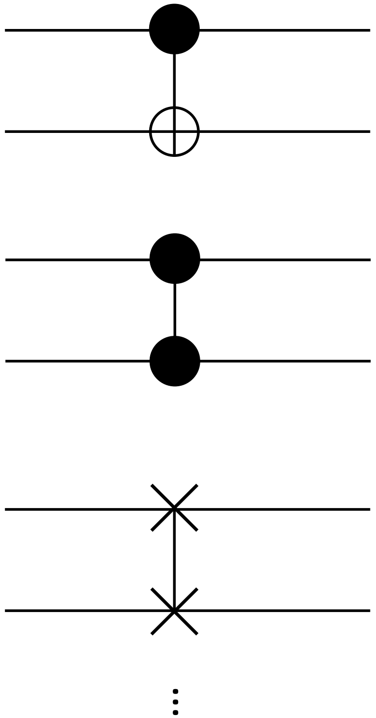
Pauli Y gate:

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

Pauli Z gate:

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

Two-qubit gates:



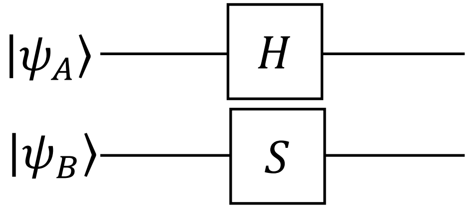
$$\mathrm{CX} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}.$$

$$\mathrm{CZ} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix}.$$

$$\mathrm{SWAP} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

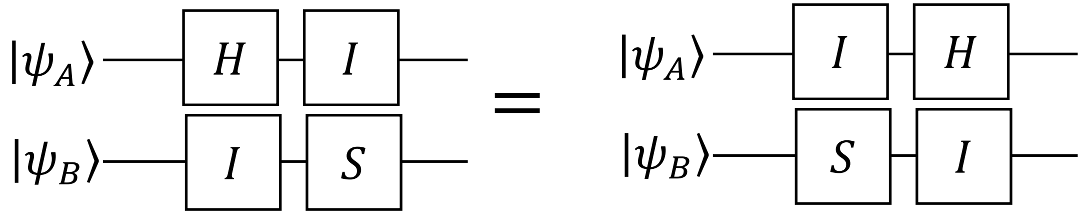
Parallel Execution of Quantum Gates

Parallel gates on two qubits:



What's the joint transformation? $H \otimes S = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & i & 0 & i \\ 1 & 0 & -1 & 0 \\ 0 & i & 0 & -i \end{bmatrix}$

||



$$H \otimes I = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & -1 & 0 \\ 0 & 1 & 0 & -1 \end{bmatrix}$$

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

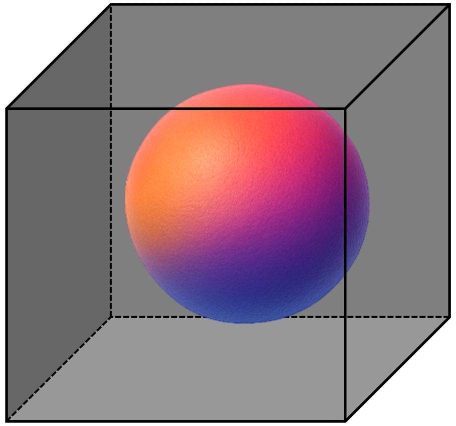
$$H \otimes S = (I \otimes S)(H \otimes I) = (H \otimes I)(I \otimes S)$$

Principle #4 – Measurement

by "probing/disturbing" its quantum state.

Measuring a qubit collapses/projects the superposition state to a basis state randomly.

Measure $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle = \frac{|0\rangle + i|1\rangle}{\sqrt{2}}$



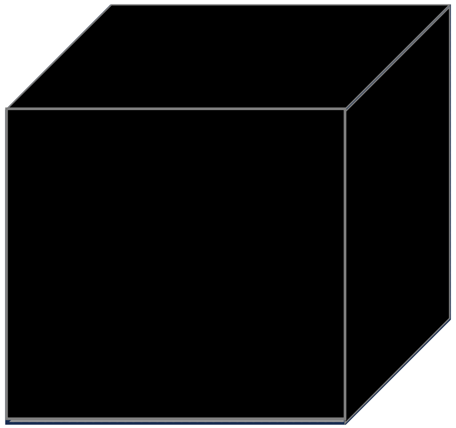
Yale

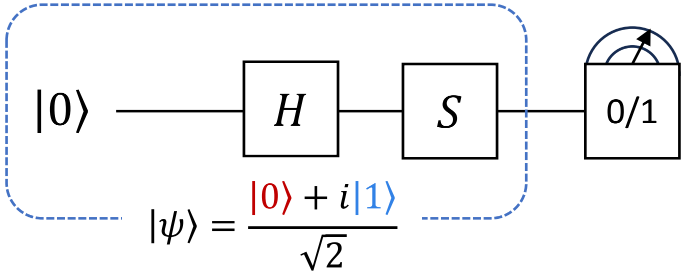
Principle #4 – Measurement

by “probing/disturbing” its quantum state.

Measuring a qubit collapses/projects the superposition state to a basis state randomly.

Measure $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle = \frac{|0\rangle + i|1\rangle}{\sqrt{2}}$





After: Becomes either $|0\rangle$ or $|1\rangle$ randomly.

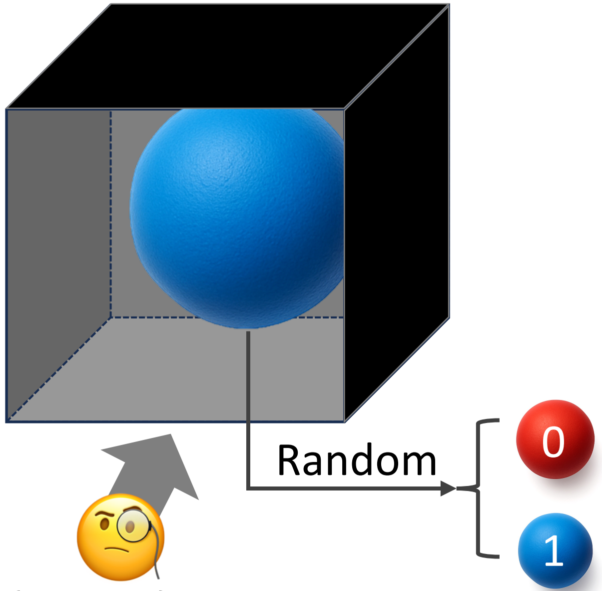
Principle #4 – Measurement

by "probing/disturbing" its quantum state.

Measuring a qubit collapses/projects the superposition state to a basis state randomly.

Measure $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle = \frac{|0\rangle + i|1\rangle}{\sqrt{2}}$

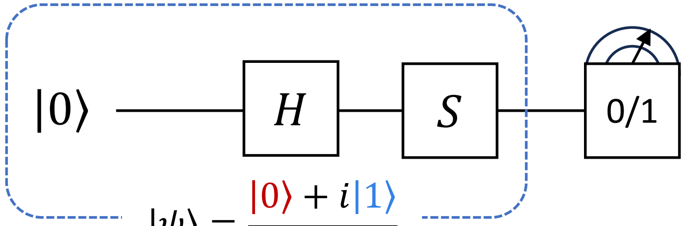
After measurement:



$p = |\alpha|^2$

$p = |\beta|^2$

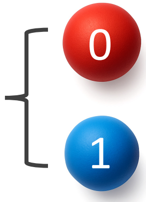
Basis: $\{|0\rangle, |1\rangle\}$



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

After: Becomes either $|0\rangle$ or $|1\rangle$ randomly.

Measurement outcome:



$p = 0.5$

$p = 0.5$

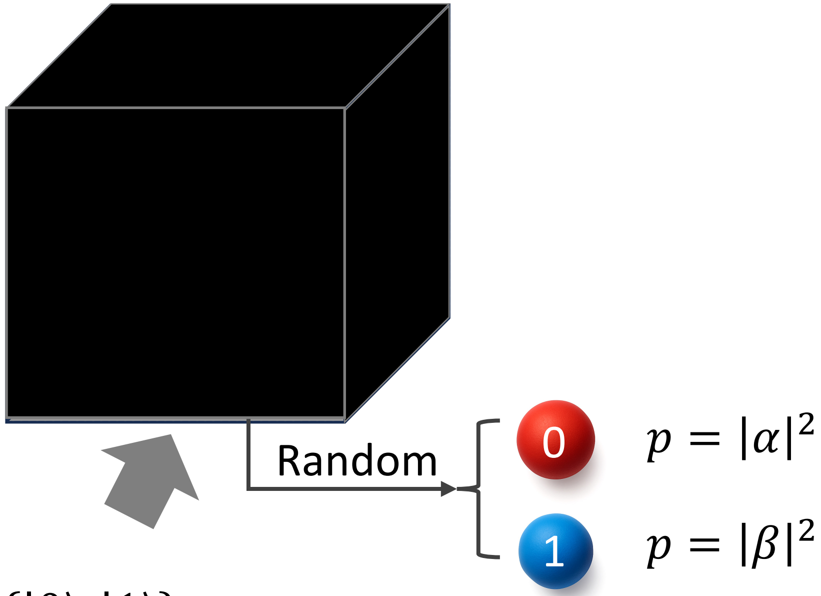
Principle #4 – Measurement

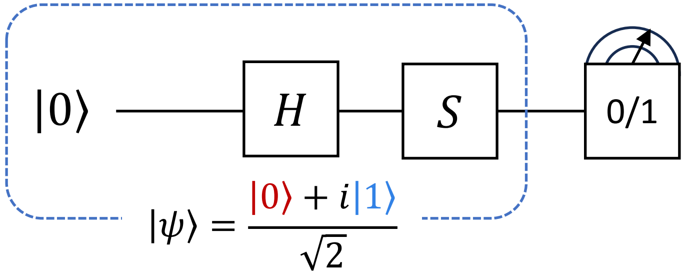
by “probing/disturbing” its quantum state.

Measuring a qubit collapses/projects the superposition state to a basis state randomly.

Measure $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle = \frac{|0\rangle + i|1\rangle}{\sqrt{2}}$

Start over and prepare the same $|\psi\rangle$:





After: Becomes either $|0\rangle$ or $|1\rangle$ randomly.

Measurement outcome:
- 0: $p = 0.5$
- 1: $p = 0.5$

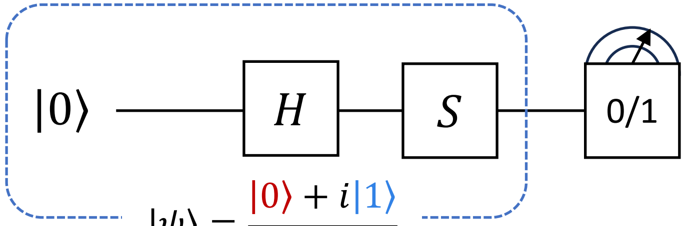
Basis: $\{|0\rangle, |1\rangle\}$

Principle #4 – Measurement

by "probing/disturbing" its quantum state.

Measuring a qubit collapses/projects the superposition state to a basis state randomly.

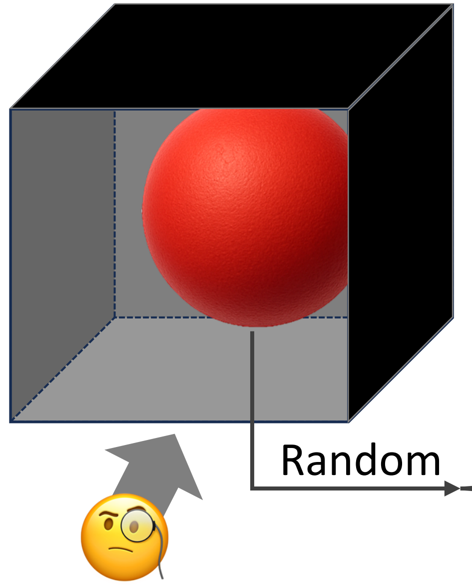
Measure $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle = \frac{|0\rangle + i|1\rangle}{\sqrt{2}}$



After: Becomes either $|0\rangle$ or $|1\rangle$ randomly.

Measurement outcome:
- 0: $p = 0.5$
- 1: $p = 0.5$

After measurement:



- 0: $p = |\alpha|^2$
- 1: $p = |\beta|^2$

Basis: $\{|0\rangle, |1\rangle\}$

Inherently probabilistic:
- Produces a random bit
- Collapses the qubit

```python
class QuantumRegister(object):
    """QuantumRegister is where we keep track of qubits"""
    def __init__(self, num_q, label='qreg'):
        super(QuantumRegister, self).__init__()
        self.size = num_q
        self.label = label
        self.array = [Qubit(i) for i in range(num_q)]
        self.state = np.array([1] + [0] * (2 ** num_q - 1), dtype=complex)

class ClassicalRegister(object):
    """ClassicalRegister is where we keep track of measurement outcomes"""
    def __init__(self, num_c, label='creg'):
        super(ClassicalRegister, self).__init__()
        self.size = num_c
        self.label = label
        self.state = np.array([bool(0) for _ in range(num_c)])
```

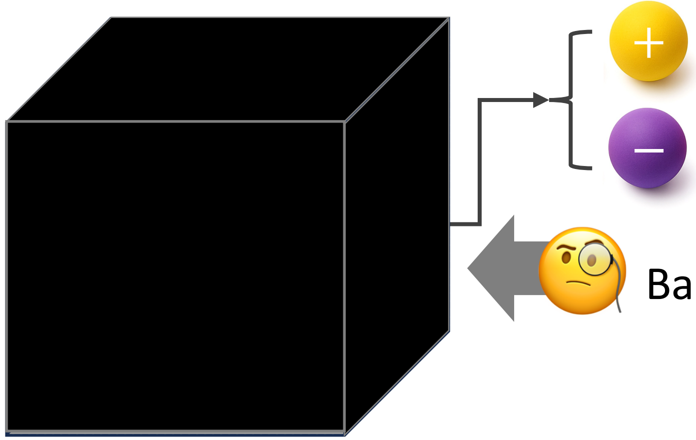
Principle #4 – Measurement

by “probing/disturbing” its quantum state.

Measuring a qubit collapses/projects the superposition state to a basis state randomly.

Measure $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle = \alpha'|+\rangle + \beta'|-\rangle$

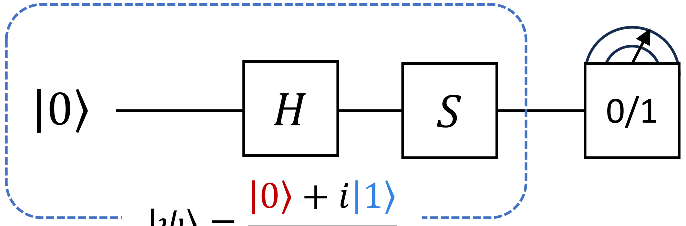
Measure in a different (orthonormal) basis:



$p = |\alpha'|^2$

$p = |\beta'|^2$

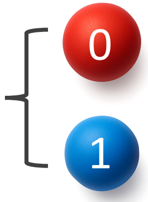
Basis #2: $\{|+\rangle, |-\rangle\}$



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

After: Becomes either $|0\rangle$ or $|1\rangle$ randomly.

Measurement outcome:



$p = 0.5$

$p = 0.5$

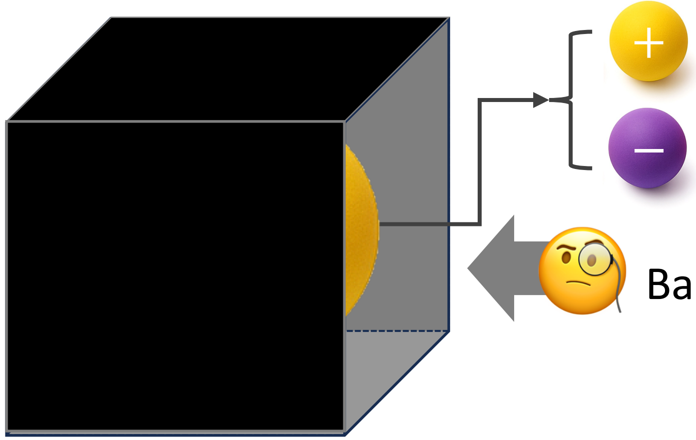
Principle #4 – Measurement

by "probing/disturbing" its quantum state.

Measuring a qubit collapses/projects the superposition state to a basis state randomly.

Measure $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle = \alpha'|+\rangle + \beta'|-\rangle$

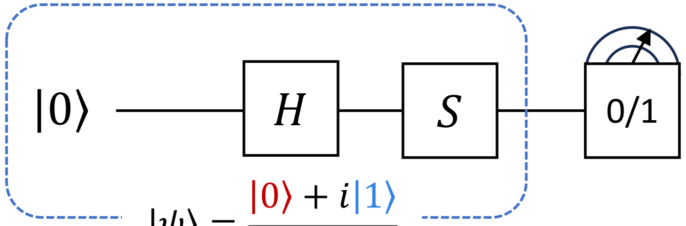
Measure in a different basis:



$p = |\alpha'|^2$

$p = |\beta'|^2$

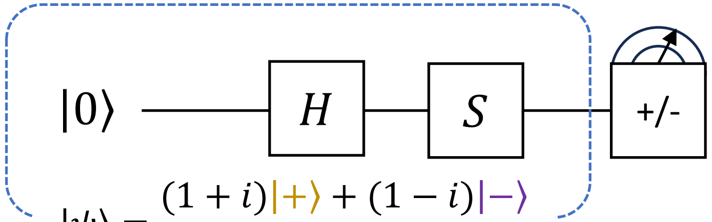
Basis #2: $\{|+\rangle, |-\rangle\}$



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

After: Becomes either $|0\rangle$ or $|1\rangle$ randomly.

Measurement outcome:
- 0: $p = 0.5$
- 1: $p = 0.5$



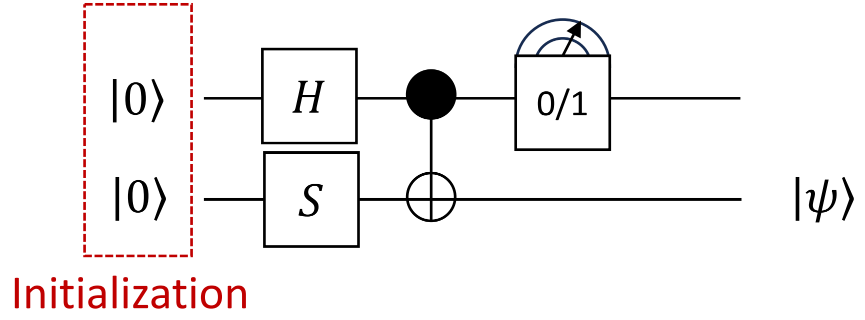
$|\psi\rangle = \frac{(1+i)|+\rangle + (1-i)|-\rangle}{2}$

After: Becomes either $|+\rangle$ or $|-\rangle$ randomly.

Measurement outcome:
- +: $p = 0.5$
- –: $p = 0.5$

Putting it all together – QuantumCircuit

Quantum circuit:



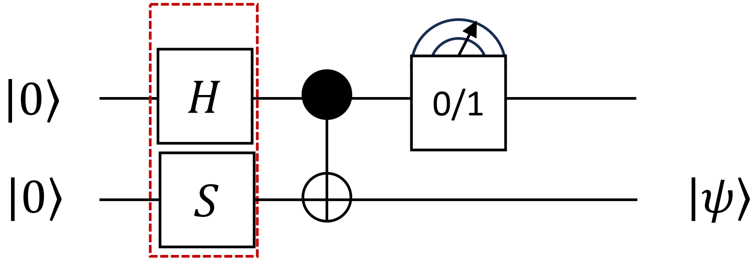
Quantum program:

pc →

```
def testCircuit():
    qc = QuantumCircuit(2,2)
    qc.h(0)
    qc.s(1)
    qc.cx(0,1)
    qc.measure([0],[0])
    # print(qc)
    outcome = qc.simulate()
```

Execution of QuantumCircuit

Quantum circuit:



Parallel Gates: $H \otimes S$

Joint transformation: $H \otimes S = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & i & 0 & i \\ 1 & 0 & -1 & 0 \\ 0 & i & 0 & -i \end{bmatrix}$

Quantum program:

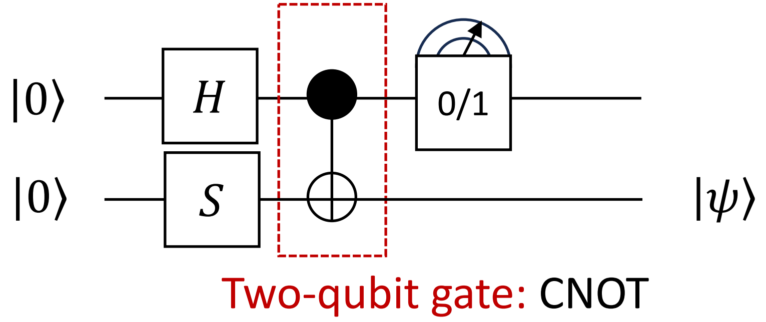
```
def testCircuit():
    qc = QuantumCircuit(2,2)
    qc.h(0)
    qc.s(1)
    qc.cx(0,1)
    qc.measure([0],[0])
    # print(qc)
    outcome = qc.simulate()
```

pc → `qc.h(0)`

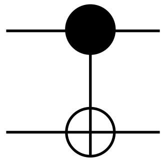
Instruction-level parallelism (ILP) → `qc.h(0)`, `qc.s(1)`

Putting it all together – QuantumCircuit

Quantum circuit:



$CX_{0,1}$ **Gate:**



$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}.$$

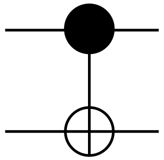
Quantum program:

pc →

```
def testCircuit():
    qc = QuantumCircuit(2,2)
    qc.h(0)
    qc.s(1)
    qc.cx(0,1)
    qc.measure([0],[0])
    # print(qc)
    outcome = qc.simulate()
```

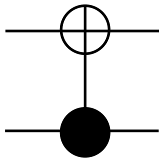
Gates in a Multi-Qubit Circuit

$CX_{0,1}$ Gate:



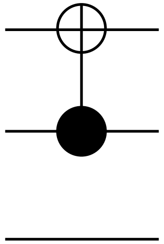
$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}.$$

$CX_{1,0}$ Gate:



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

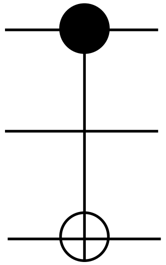
What about the following gates in a qc=QuantumCircuit(3,3)?



	000	001	010	011	100	101	110	111
000	1							
001		1						
010							1	
011								1
100					1			
101						1		
110			1					
111				1				

qc.cx(1,0): $CX_{1,0} \otimes I$

Identity on untouched qubit(s).



	000	001	010	011	100	101	110	111
000	1							
001		1						
010			1					
011				1				
100						1		
101					1			
110								1
111							1	

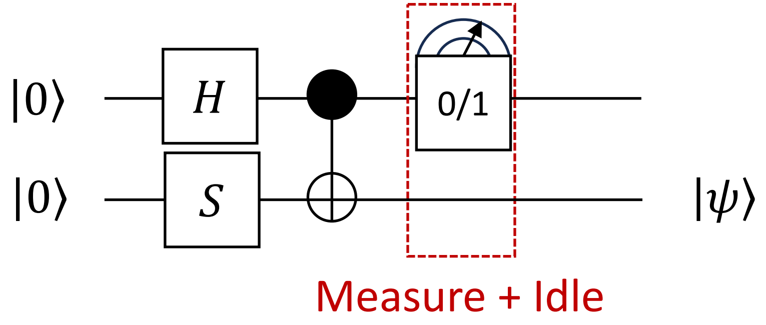
qc.cx(0,2): $\mathrm{SWAP}_{1,2} \cdot \left(CX_{0,1} \otimes I\right) \cdot \mathrm{SWAP}_{1,2}$

Homework: tensorizeGate().

Expand a gate into a $2^n \times 2^n$ matrix for an n-qubit circuit.

Putting it all together – QuantumCircuit

Quantum circuit:



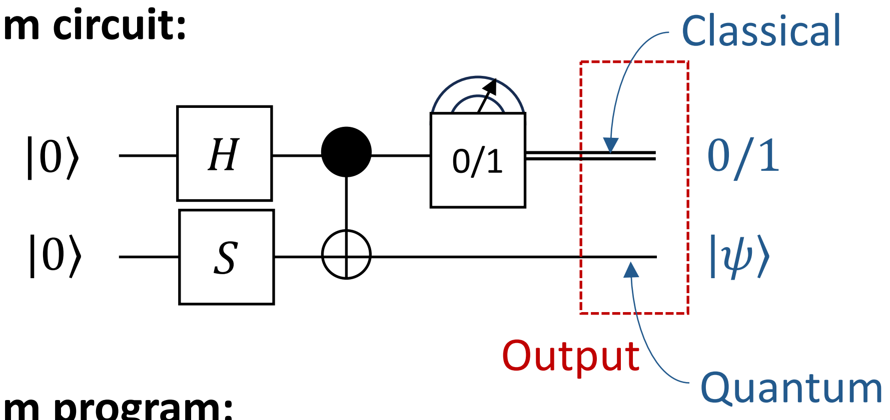
Quantum program:

pc →

```
def testCircuit():
    qc = QuantumCircuit(2,2)
    qc.h(0)
    qc.s(1)
    qc.cx(0,1)
    qc.measure([0],[0])
    # print(qc)
    outcome = qc.simulate()
```

Putting it all together – QuantumCircuit

Quantum circuit:



Quantum program:

```
def testCircuit():
    qc = QuantumCircuit(2,2)
    qc.h(0)
    qc.s(1)
    qc.cx(0,1)
    qc.measure([0],[0])
    # print(qc)
    outcome = qc.simulate()
```

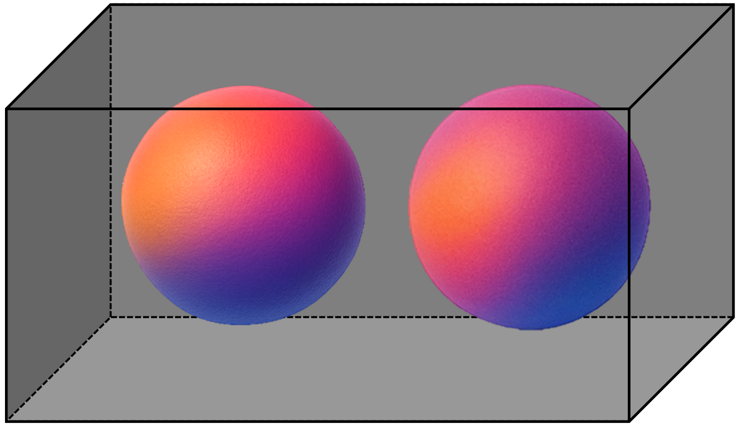
pc → `outcome = qc.simulate()`

```
class QuantumCircuit(object):
    """QuantumCircuit"""
    def __init__(self, num_q, num_c):
        super(QuantumCircuit, self).__init__()
        self.num_q = num_q
        self.qubits = QuantumRegister(num_q) # initialized qubits
        self.num_c = num_c
        self.cbits = ClassicalRegister(num_c) # initialized cbits
        self.circuit = [] # sequence of instructions
        self.pc = 0 # program counter
        self.curr_state = self.qubits.state # state up to the point of program counter

    def _append(self, operation, q_array, c_array):
        # Add new instruction to circuit
        instruction = [operation, q_array, c_array]
        self.circuit.append(instruction)

    # Hadamard gate
    def h(self, qubit):
        # Define Gate by its name, kind (number of qubit), and matrix
        HGate = Gate('h', 1, 1/np.sqrt(2) * np.array([[1,1],[1,-1]], dtype=complex))
        self._append(HGate, [qubit], [])
        return

    # Measure qubits in array 'qubits' and store classical outcome in 'cbits'
    # Note: Action of measurement will be defined in simulate function.
    def measure(self, qubits, cbits):
        assert(len(qubits) == len(cbits))
        Measure = Gate('measure', len(qubits), None)
        self._append(Measure, qubits, cbits)
        return
```

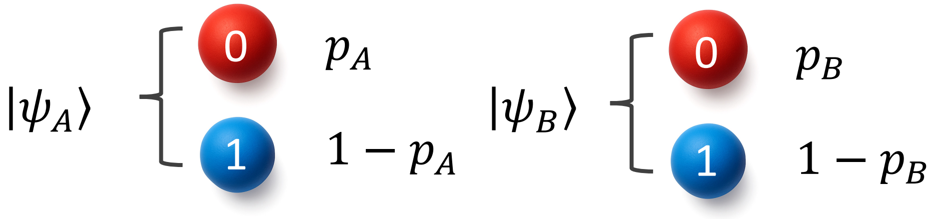
COMING UP NEXT

- Distinguishing two qubits
- Visualizing one qubit

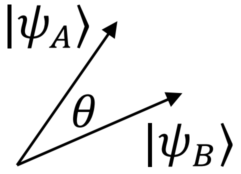
Distinguishing Two Qubits



$|\psi_A\rangle = \begin{bmatrix}\alpha_0 \\ \alpha_1\end{bmatrix}$ $\qquad |\psi_B\rangle = \begin{bmatrix}\beta_0 \\ \beta_1\end{bmatrix}$



Assume $|\psi_A\rangle, |\psi_B\rangle \in \mathcal{H}$ with real amplitudes.



Length: $\langle\psi_A|\psi_A\rangle = \langle\psi_B|\psi_B\rangle = 1$
Angle: $\cos\theta = \langle\psi_A|\psi_B\rangle$
Qubits are the same if $\theta = 0$

Can we tell if $\theta = 0$ by measuring the qubits?

Measurement in the standard basis:

$$p_A = |\langle 0|\psi_A\rangle|^2 = |\alpha_0|^2, \qquad p_B = |\langle 0|\psi_B\rangle|^2 = |\beta_0|^2$$

Measurement Strategy:

- Receive multiple copies of the two qubits.
- Repeat the measurement experiment.
- If $p_A \neq p_B$ then **Different!**

Distinguishing Two Qubits

Trial #1

Trial #2

Repeat many times

…

We can tell if two qubits are different by collecting **measurement statistics** of each qubit.

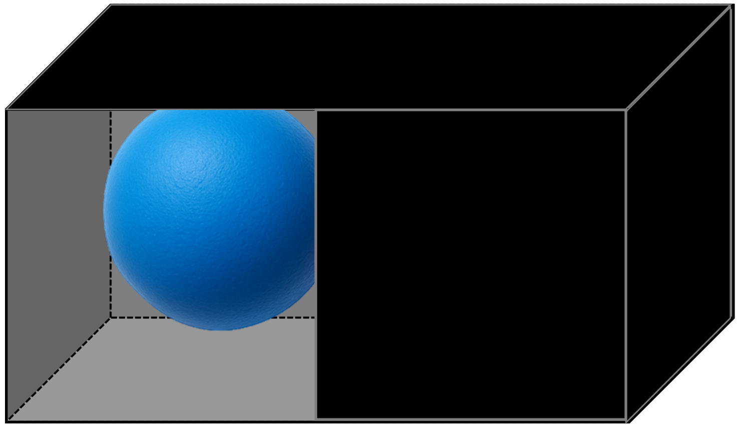
Example #1:

- $|\psi_A\rangle = |1\rangle$
  - 0: $p = ?$
  - 1: $p = ?$
- $|\psi_B\rangle = |+\rangle$
  - 0: $p = ?$
  - 1: $p = ?$

Different!

Distinguishing Two Qubits

Trial #1



Trial #2

Repeat many times

…

We can tell if two qubits are different by collecting **measurement statistics** of each qubit.

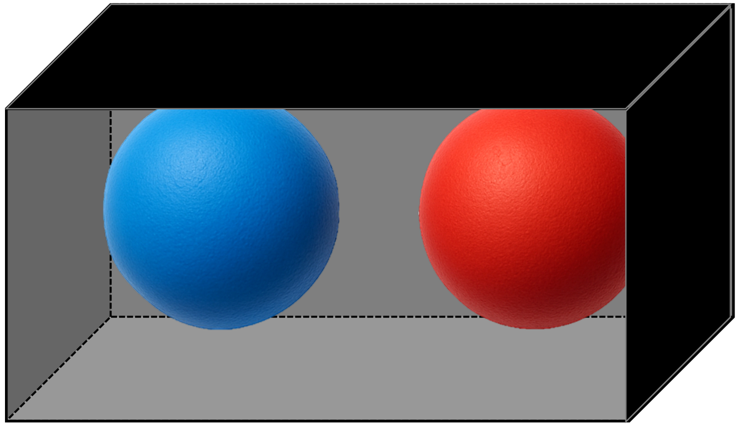
Example #1:

- $|\psi_A\rangle = |1\rangle$
  - 0: $p = 0$
  - 1: $p = 1$
- $|\psi_B\rangle = |+\rangle$
  - 0: $p = 0.5$
  - 1: $p = 0.5$

Different!

Distinguishing Two Qubits

Trial #1



Trial #2

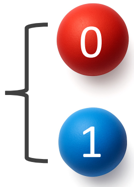
Repeat many times

...

We can tell if two qubits are different by collecting **measurement statistics** of each qubit.

Example #1:

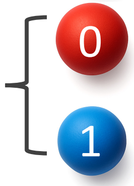
- $|\psi_A\rangle = |1\rangle$



$p = 0$

$p = 1$

- $|\psi_B\rangle = |+\rangle$



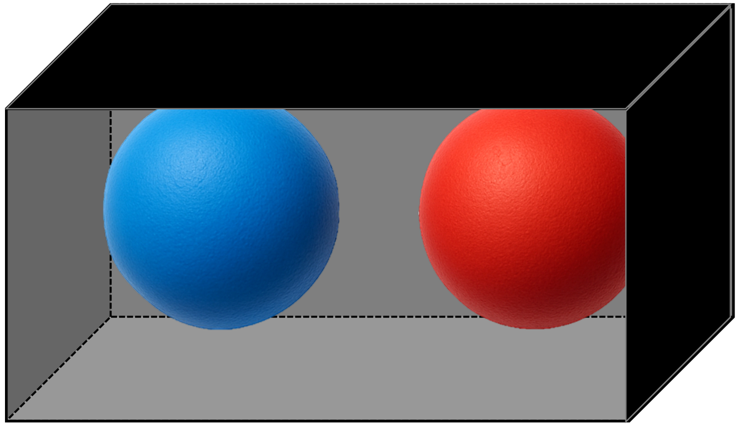
$p = 0.5$

$p = 0.5$

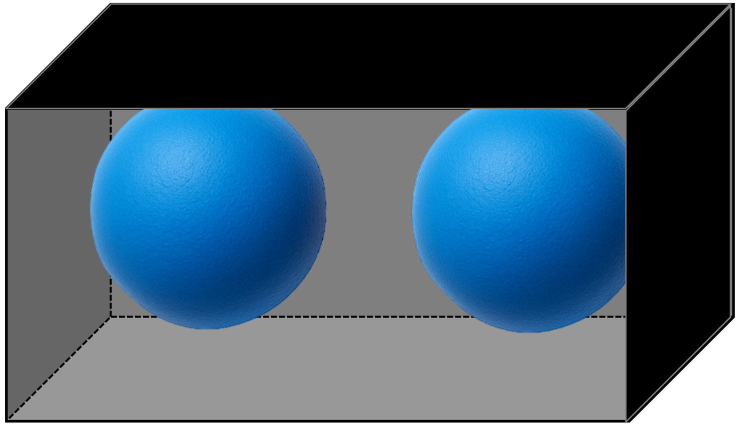
Different!

Distinguishing Two Qubits

Trial #1



Trial #2



Repeat many times

...

We can tell if two qubits are different by collecting **measurement statistics** of each qubit.

Example #1:

- $|\psi_A\rangle = |1\rangle$
  - 0: $p = 0$
  - 1: $p = 1$
- $|\psi_B\rangle = |+\rangle$
  - 0: $p = 0.5$
  - 1: $p = 0.5$

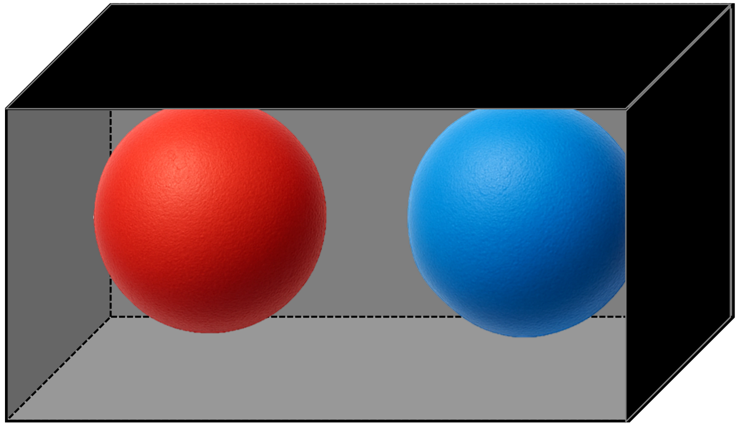
Different!

One-sided error strategy to tell the difference:

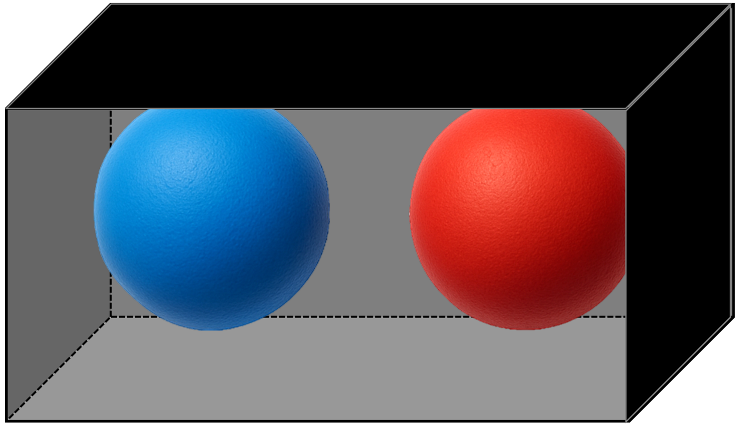
- If measured red(1), I know it's $|\psi_B\rangle$.
- If measured blue(0), I guess it's $|\psi_A\rangle$.

Distinguishing Two Qubits

Trial #1



Trial #2



Repeat many times

…

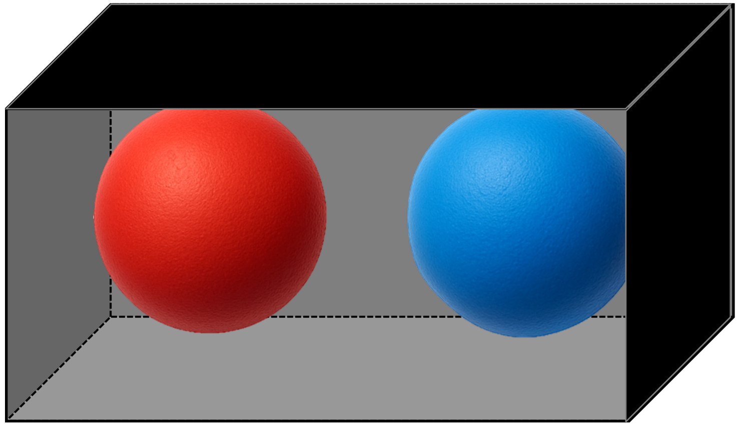
We can tell if two qubits are different by collecting **measurement statistics** of each qubit.

Example #2:

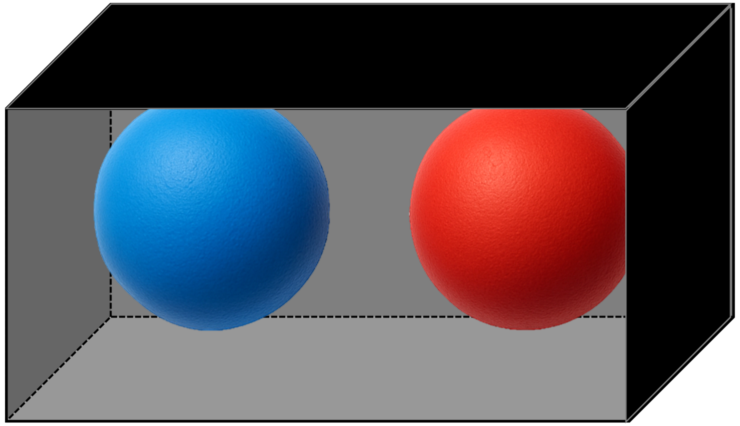
- $|\psi_A\rangle = |+\rangle$
  - 0 $p = ?$
  - 1 $p = ?$
- $|\psi_B\rangle = |-\rangle$
  - 0 $p = ?$
  - 1 $p = ?$

Distinguishing Two Qubits

Trial #1



Trial #2



Repeat many times

...

We can tell if two qubits are different by collecting **measurement statistics** of each qubit.

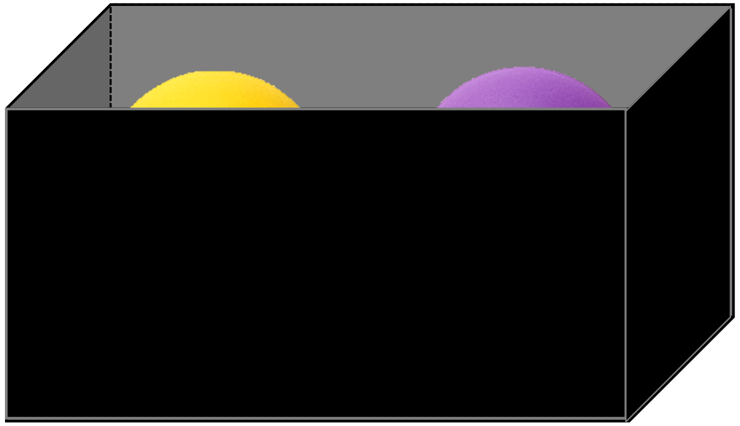
Example #2:

- $|\psi_A\rangle = |+\rangle$
  - 0: $p = 0.5$
  - 1: $p = 0.5$
- $|\psi_B\rangle = |-\rangle$
  - 0: $p = 0.5$
  - 1: $p = 0.5$

Same?

Distinguishing Two Qubits

Trial #1



Trial #2

Repeat many times

...

We can tell if two qubits are different by collecting **measurement statistics** of each qubit.

Example #2:

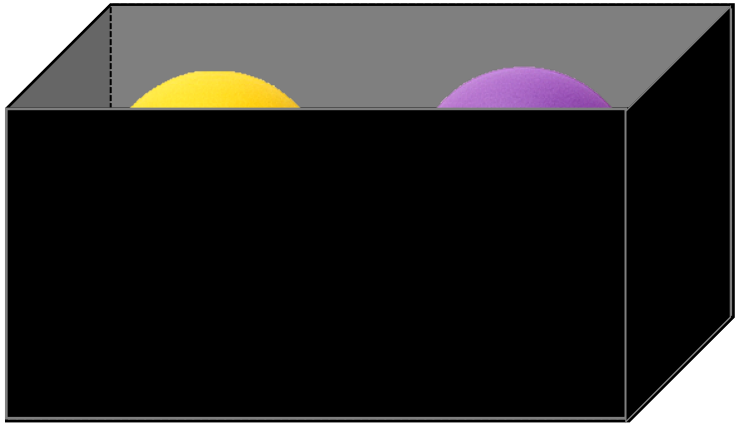
- $|\psi_A\rangle = |+\rangle$ — 0: $p = 0.5$; 1: $p = 0.5$
- $|\psi_B\rangle = |-\rangle$ — 0: $p = 0.5$; 1: $p = 0.5$

Same?

- $|\psi_A\rangle = |+\rangle$ — +: $p = ?$; −: $p = ?$
- $|\psi_B\rangle = |-\rangle$ — +: $p = ?$; −: $p = ?$

Distinguishing Two Qubits

Trial #1



Trial #2

Repeat many times

...

We can tell if two qubits are different by collecting **measurement statistics** of each qubit.

Example #2:

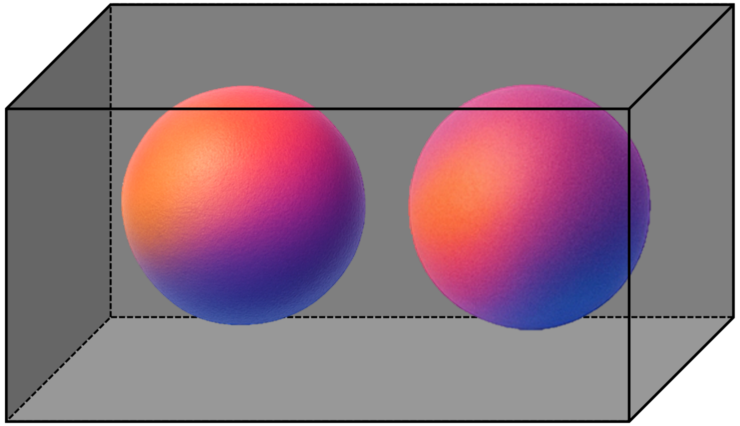
- $|\psi_A\rangle = |+\rangle$ — 0: $p = 0.5$; 1: $p = 0.5$
- $|\psi_B\rangle = |-\rangle$ — 0: $p = 0.5$; 1: $p = 0.5$

Same?

- $|\psi_A\rangle = |+\rangle$ — +: $p = 1$; −: $p = 0$
- $|\psi_B\rangle = |-\rangle$ — +: $p = 0$; −: $p = 1$

Different!

Improved Measurement Strategy



$|\psi_A\rangle = \begin{bmatrix}\alpha_0 \\ \alpha_1\end{bmatrix}$ $|\psi_B\rangle = \begin{bmatrix}\beta_0 \\ \beta_1\end{bmatrix}$

Change of basis:

- $|\psi_A\rangle = \alpha_0|0\rangle + \alpha_1|1\rangle = \alpha'_0|v_0\rangle + \alpha'_1|v_1\rangle$
- $|\psi_B\rangle = \beta_0|0\rangle + \beta_1|1\rangle = \beta'_0|v_0\rangle + \beta'_1|v_1\rangle$

Improved Measurement Strategy:

- Receive multiple copies of the two qubits.
- Choose a measurement basis.
- Repeat the measurement experiment.
- If $p_A \neq p_B$ then **Different!**

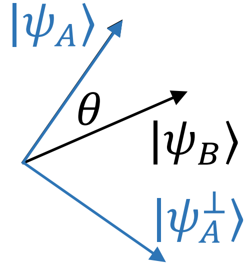
Measurement in the basis $\{|v_0\rangle, |v_1\rangle\}$:

$$p_A = |\langle v_0|\psi_A\rangle|^2, \qquad p_B = |\langle v_1|\psi_B\rangle|^2$$

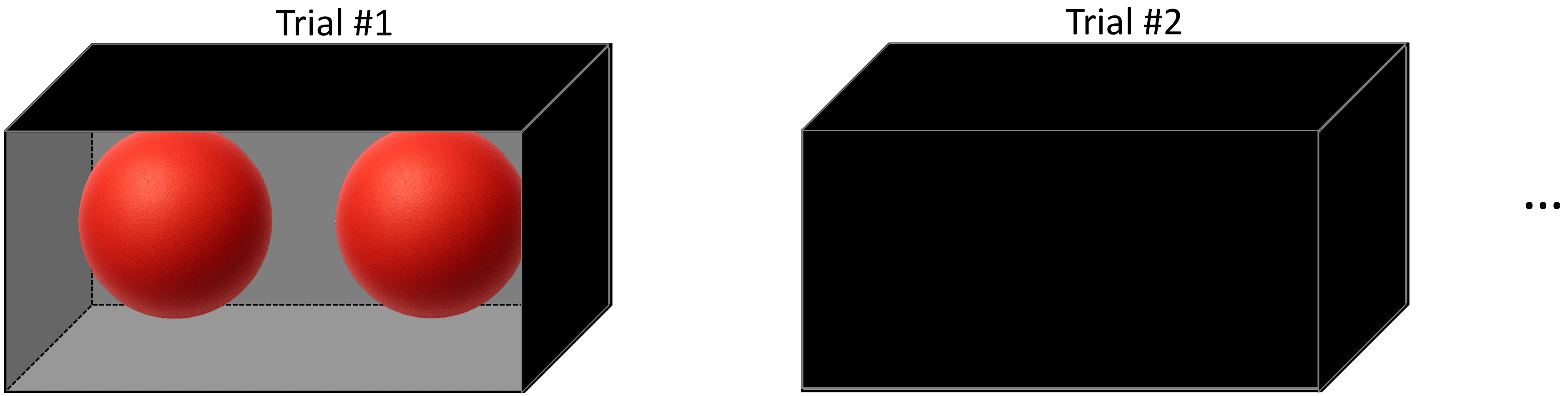
$$p_A = |\alpha'_0|^2, \qquad p_B = |\beta'_0|^2$$

For example , we choose basis $\{|\psi_A\rangle, |\psi_A^\perp\rangle\}$:

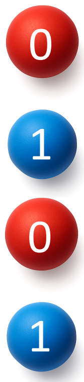
One-sided error: $\begin{cases} \text{For } |\psi_A\rangle, \Pr[|\psi_A^\perp\rangle] = 0 \\ \text{For } |\psi_B\rangle, \Pr[|\psi_A^\perp\rangle] = \sin^2\theta \end{cases}$



Distinguishing Two Qubits



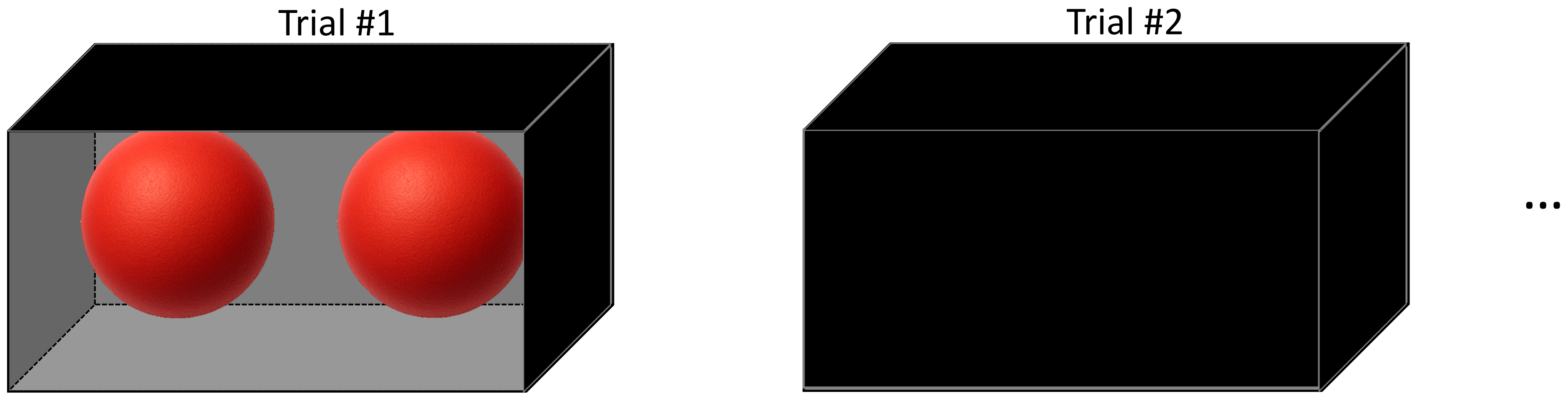
We can tell if two qubits are different by collecting **measurement statistics** of each qubit.



Example #3:

- $|\psi_A\rangle = |0\rangle$
  - 0: $p = ?$
  - 1: $p = ?$
- $|\psi_B\rangle = -|0\rangle$
  - 0: $p = ?$
  - 1: $p = ?$

Distinguishing Two Qubits



We can tell if two qubits are different by collecting **measurement statistics** of each qubit.

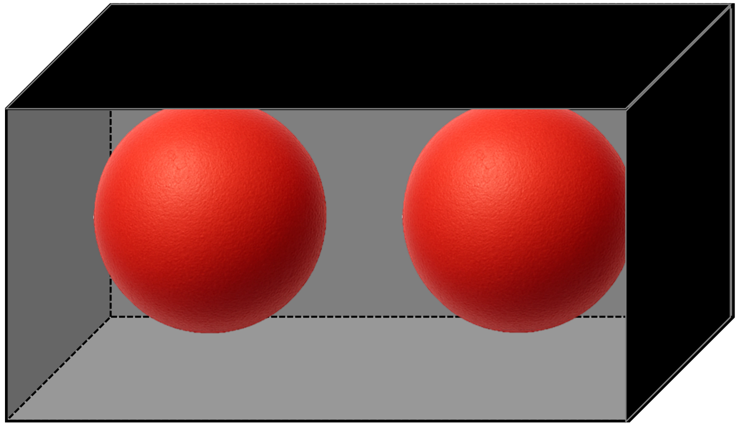
Example #3:

- $|\psi_A\rangle = |0\rangle$
  - 0: $p = 1$
  - 1: $p = 0$
- $|\psi_B\rangle = -|0\rangle$
  - 0: $p = 1$
  - 1: $p = 0$

Same?

Distinguishing Two Qubits

Trial #1



Trial #2

…

We can tell if two qubits are different by collecting **measurement statistics** of each qubit.

Example #3:

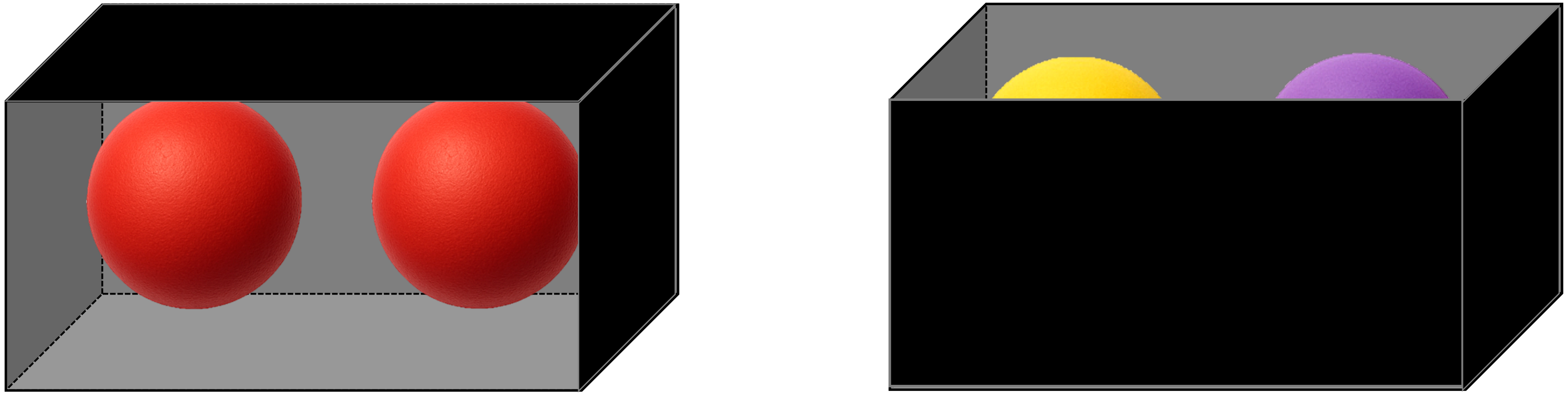
- $|\psi_A\rangle = |0\rangle$ — 0: $p = 1$; 1: $p = 0$
- $|\psi_B\rangle = -|0\rangle$ — 0: $p = 1$; 1: $p = 0$

Same?

- $|\psi_A\rangle = |0\rangle$ — +: $p = ?$; −: $p = ?$
- $|\psi_B\rangle = -|0\rangle$ — +: $p = ?$; −: $p = ?$

Distinguishing Two Qubits

Trial #1 Trial #2



...

We can tell if two qubits are different by collecting **measurement statistics** of each qubit.

Example #3:

- $|\psi_A\rangle = |0\rangle$ — 0: $p = 1$; 1: $p = 0$
- $|\psi_B\rangle = -|0\rangle$ — 0: $p = 1$; 1: $p = 0$

Same?

- $|\psi_A\rangle = |0\rangle$ — +: $p = 0.5$; −: $p = 0.5$
- $|\psi_B\rangle = -|0\rangle$ — +: $p = 0.5$; −: $p = 0.5$

Same?

Angle: $\cos\theta = \langle\psi_A|\psi_B\rangle = -1$

$\theta = \pi$

Global Phase

Two states differing by a global factor:
$|\psi_A\rangle = \alpha_0|0\rangle + \alpha_1|1\rangle$ and $|\psi_B\rangle = -(\alpha_0|0\rangle + \alpha_1|1\rangle)$

No matter how we choose our measurement basis $\{|v_0\rangle, |v_1\rangle\}$:

$$p_A = |\langle v_0|(\alpha_0'|v_0\rangle + \alpha_1'|v_1\rangle)|^2 = |\alpha_0'|^2, \qquad p_B = |\langle v_0| - (\alpha_0'|v_0\rangle + \alpha_1'|v_1\rangle)|^2 = |-\alpha_0'|^2$$

In fact, for any "global phase":

- $\alpha|0\rangle + \beta|1\rangle \equiv e^{i\phi}(\alpha|0\rangle + \beta|1\rangle)$ for $\phi \in \mathbb{R}$
- No experiment can distinguish $|\psi\rangle$ from $e^{i\phi}|\psi\rangle$. They are **identical** quantum states.

$$p_A = |\langle v_0|(\alpha_0'|v_0\rangle + \alpha_1'|v_1\rangle)|^2 = |\alpha_0'|^2, \qquad p_B = \left|\langle v_0|e^{i\phi}(\alpha_0'|v_0\rangle + \alpha_1'|v_1\rangle)\right|^2 = \left|e^{i\phi}\alpha_0'\right|^2$$

Note: relative phase still matters!

- $\alpha|0\rangle + e^{i\phi}\beta|1\rangle$ for some $\phi \in \mathbb{R}$

A Qubit – The Bloch Sphere

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle \in \mathbb{C}^2$$

- Normalized: $|\alpha|^2 + |\beta|^2 = 1$
- **Global phase does not matter:**
  $|\psi\rangle$ and $e^{i\phi}|\psi\rangle$ not distinguishable

Derive on board (in terms of two real numbers?):

$$\begin{bmatrix}\alpha\\ \beta\end{bmatrix} = \begin{bmatrix}?\\ ?\end{bmatrix}$$

$$= \begin{bmatrix}a\\ e^{i\phi}\sqrt{1-a^2}\end{bmatrix} \qquad 0 \leq a \leq 1, 0 \leq \phi < 2\pi$$

We can visualize two real numbers!

A Qubit – The Bloch Sphere

$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle \in \mathbb{C}^2$

- Normalized: $|\alpha|^2 + |\beta|^2 = 1$
- Global phase does not matter:
  $|\psi\rangle$ and $e^{i\phi}|\psi\rangle$ not distinguishable

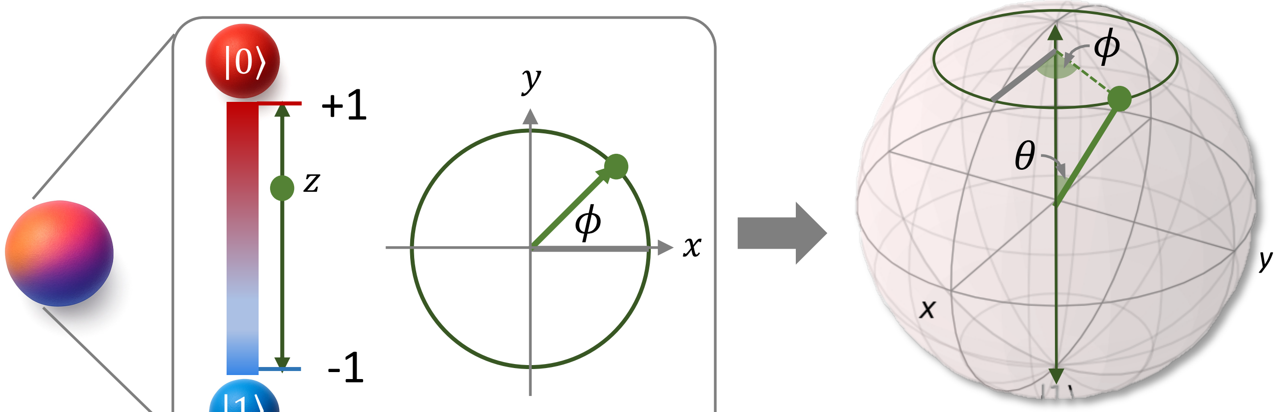
$$\begin{bmatrix}\alpha\\ \beta\end{bmatrix} = \begin{bmatrix} a \\ e^{i\phi}\sqrt{1-a^2}\end{bmatrix}$$

$$(0 \le a \le 1, 0 \le \phi < 2\pi)$$

$$= \begin{bmatrix} \cos\frac{\theta}{2} \\ e^{i\phi}\sin\frac{\theta}{2}\end{bmatrix} \quad (0 \le \theta \le \pi, 0 \le \phi < 2\pi)$$

We can visualize two real numbers:

- $a^2 \in [0,1]$: Probability of measuring $|0\rangle$
  - $z = 2\left(a^2 - \frac{1}{2}\right) \in [-1,1]$
- $e^{i\phi}$: relative phase, $0 \le \phi < 2\pi$



Spherical Coordinate: (θ, ϕ)

Where are $|0\rangle$ and $|1\rangle$ on Bloch sphere?

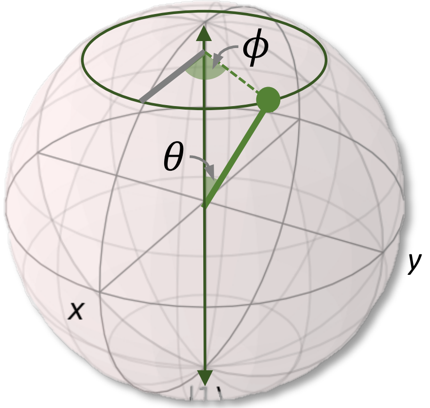
A Qubit – The Bloch Sphere

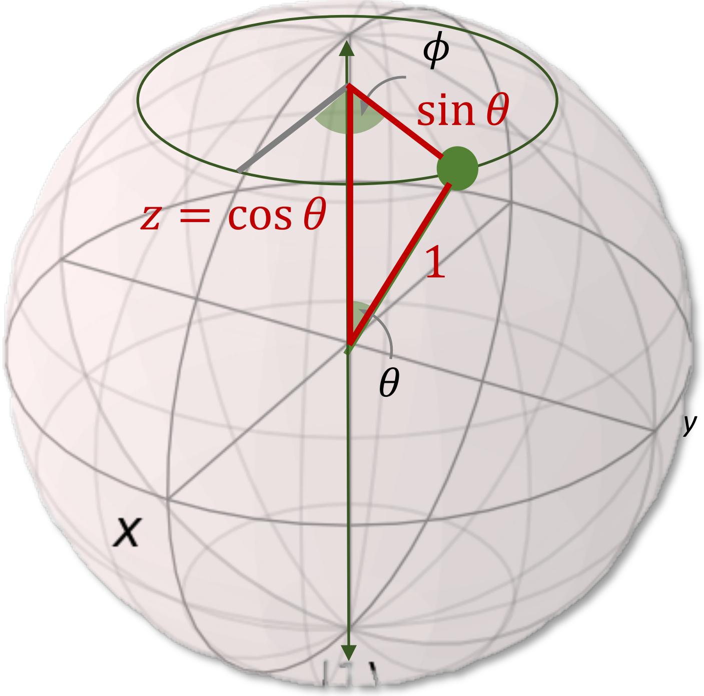
$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle \in \mathbb{C}^2$$

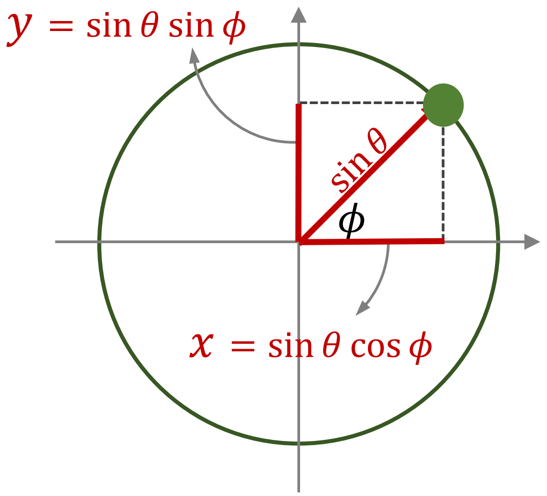
$$= \begin{bmatrix} \cos\frac{\theta}{2} \\ e^{i\phi}\sin\frac{\theta}{2} \end{bmatrix}$$

$$(0 \le \theta \le \pi, 0 \le \phi < 2\pi)$$

Spherical Coordinate: (θ, ϕ)







What about its **Cartesian coordinate**: (x, y, z)?

$$(x, y, z) = (\sin\theta\cos\phi, \sin\theta\sin\phi, \cos\theta)$$

Density operator: $|\psi\rangle\langle\psi| = \frac{1}{2}(\sigma_I + x\cdot\sigma_X + y\cdot\sigma_Y + z\cdot\sigma_Z)$