Construindo circuitos Quânticos com Qiskit

Ismael Araujo

Revisão

Revisão

Qubits

- Estados quânticos:

$$|0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \;;\; |1\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix};\; \alpha|0\rangle + \beta|1\rangle = \begin{bmatrix} \alpha \\ \beta \end{bmatrix}$$

Revisão

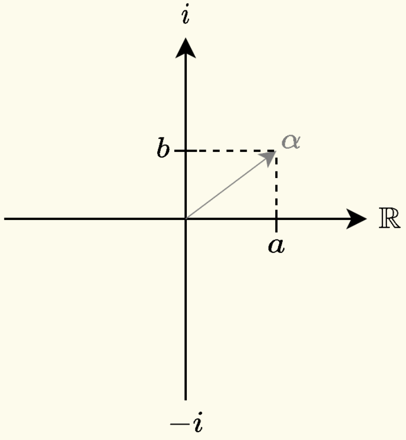
Qubits

- Estados quânticos:

$$|0\rangle = \begin{bmatrix}1\\0\end{bmatrix} \;;\; |1\rangle = \begin{bmatrix}0\\1\end{bmatrix};\; \alpha|0\rangle + \beta|1\rangle = \begin{bmatrix}\alpha\\\beta\end{bmatrix}$$

- Amplitudes complexas:

$$\alpha = a + ib$$



Revisão

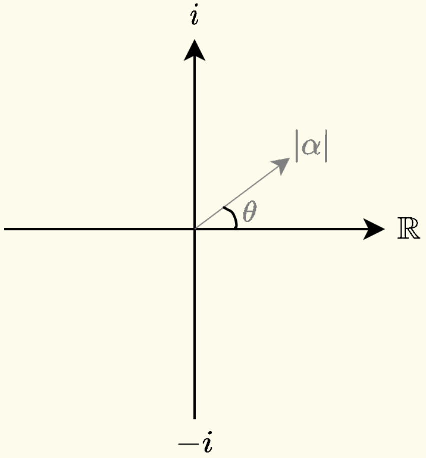
Amplitudes complexas

- Componentes complexos e reais:

$$\alpha = a + ib$$

- Forma polar:

$$\alpha = |\alpha| e^{i\theta}$$



Revisão

Amplitudes complexas

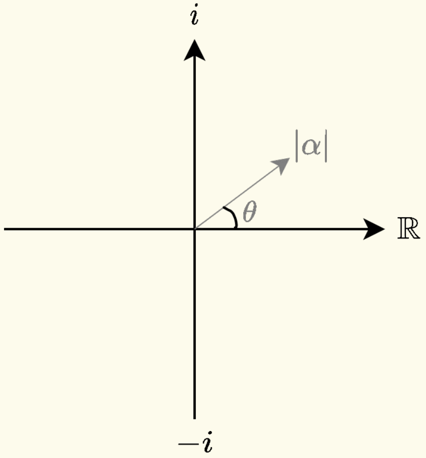
- Componentes complexos e reais:

$$\alpha = a + ib$$

- Forma polar:

$$\alpha = |\alpha| e^{i\theta}$$

$$e^{i\theta} = \cos(\theta) + i \sin(\theta)$$



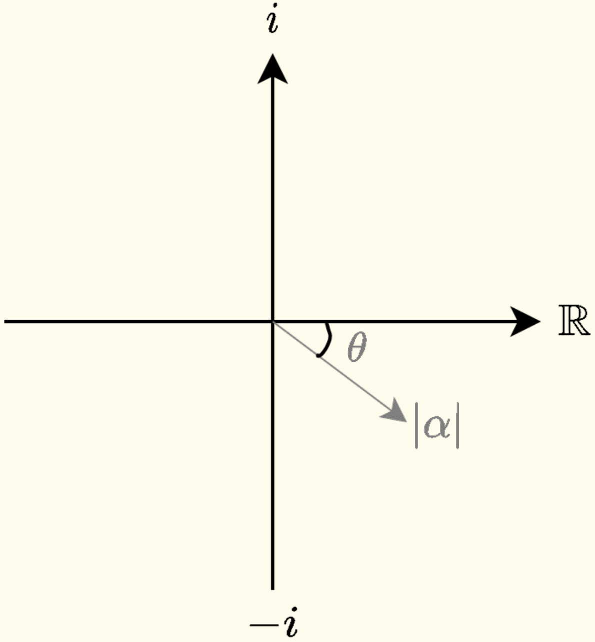
Revisão

Amplitudes complexas

- Conjulgado Complexo:

$$\alpha^{\star} = |\alpha| e^{-i\theta}$$

$$e^{i\theta} = \cos(\theta) - i \sin(\theta)$$



Revisão

Superposição

- Regra de Born

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

$$p(|0\rangle) = |\alpha|^2 \quad ; \quad p(|1\rangle) = |\beta|^2$$

$$|\alpha|^2 = |\alpha||\alpha|e^{i(\theta-\theta)}$$

Aplicando operadores quânticos

Operações elementares, Operações Parametrizadas, Decomposição ZY

Aplicando operadores quânticos

Os efeitos dos operadores nos estados

- Manipulação dos estados da base
- Gerar superposições de estados da base
- Operadores Unitários

$$UU^\dagger = U^\dagger U = I$$

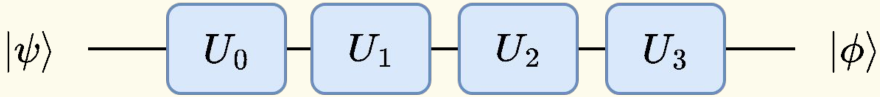
Operadores sobre 1-qubit

Notação

- Matricialmente:

$$U_3 U_2 U_1 U_0 |\psi\rangle = |\phi\rangle$$

- No circuito quântico:



Operadores sobre 1-qubit

Portas elementares

- Operadores de Pauli:

$$\sigma_1 = \sigma_x = X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \qquad \sigma_2 = \sigma_y = Y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix} \qquad \sigma_3 = \sigma_z = Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$$

- No circuito quântico:

$$|0\rangle \; -\boxed{X}- \; |1\rangle \qquad |0\rangle \; -\boxed{Y}- \; i|1\rangle \qquad |0\rangle \; -\boxed{Z}- \; |0\rangle$$

$$|1\rangle \; -\boxed{X}- \; |0\rangle \qquad |1\rangle \; -\boxed{Y}- \; -i|0\rangle \qquad |1\rangle \; -\boxed{Z}- \; -|1\rangle$$

Operadores sobre 1-qubit

Portas elementares

- Operador Hadamard:

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

- No circuito quântico:

$$|0\rangle \;-\!\boxed{H}\!-\; \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$$

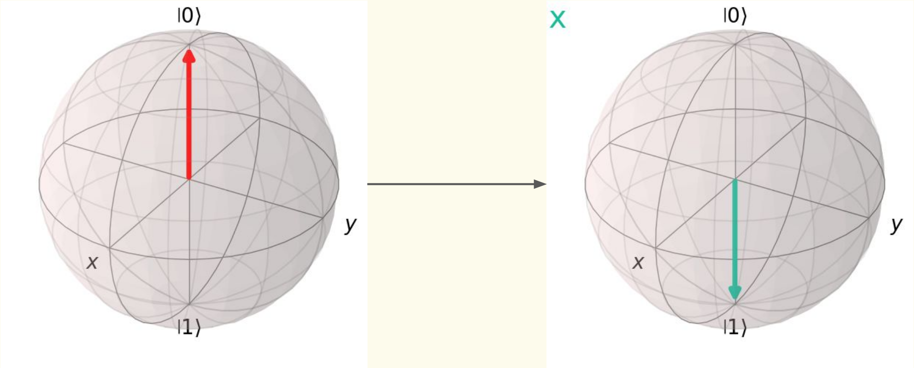
$$|1\rangle \;-\!\boxed{H}\!-\; \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$$

Operadores sobre 1-qubit

- Na esfera de Bloch:

$$|0\rangle \; -\!\boxed{X}\!- \; |1\rangle$$

$$|1\rangle \; -\!\boxed{X}\!- \; |0\rangle$$

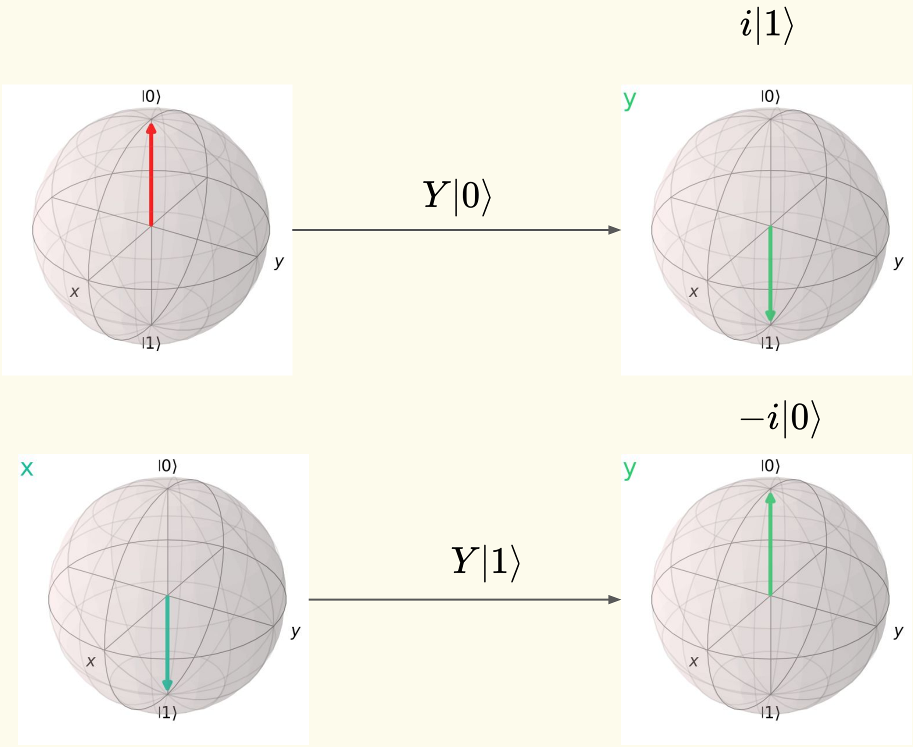


Operadores sobre 1-qubit

- Na esfera de Bloch:

$$|0\rangle \; - \boxed{Y} - \; i|1\rangle$$

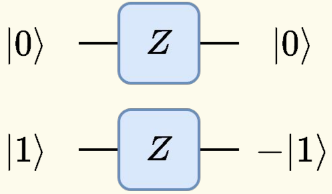
$$|1\rangle \; - \boxed{Y} - \; -i|0\rangle$$



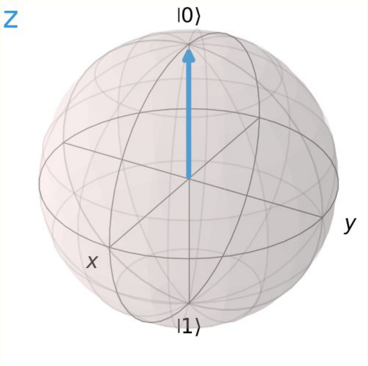
IBM

Operadores sobre 1-qubit

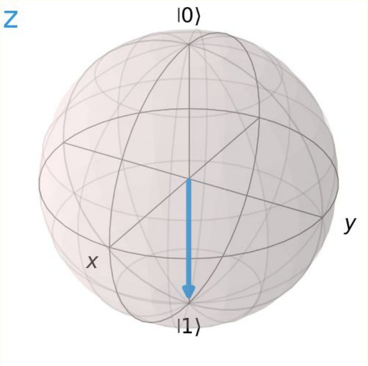
- Na esfera de Bloch:



$Z|0\rangle$



$Z|1\rangle$



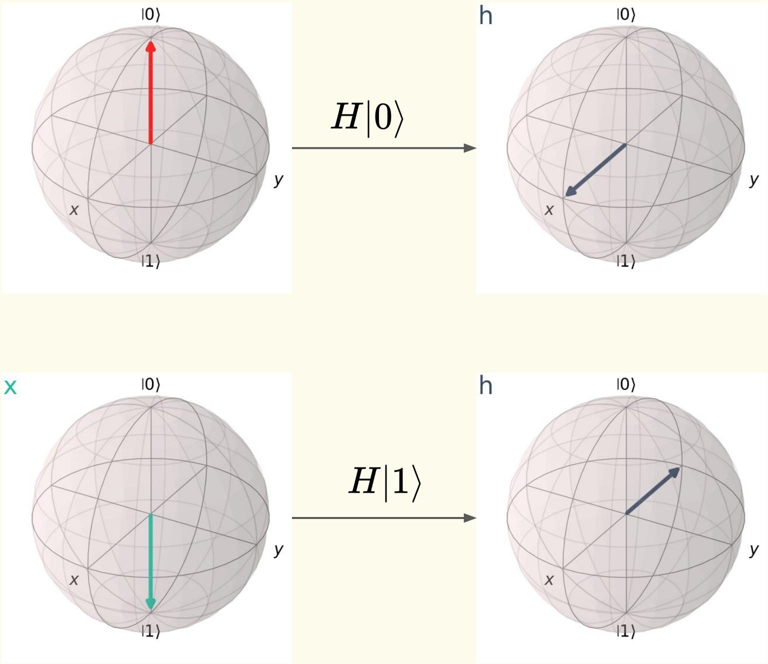
Operadores sobre 1-qubit

Portas elementares

- Na esfera de Bloch:

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

$$|1\rangle \; -\boxed{H}- \; \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$$



Operadores sobre 1-qubit

Portas elementares

- Rotações parametrizadas:

$$R_x(\theta) = \begin{bmatrix} \cos\left(\frac{\theta}{2}\right) & -i\sin\left(\frac{\theta}{2}\right) \\ -i\sin\left(\frac{\theta}{2}\right) & \cos\left(\frac{\theta}{2}\right) \end{bmatrix} \quad R_y(\theta) = \begin{bmatrix} \cos\left(\frac{\theta}{2}\right) & -\sin\left(\frac{\theta}{2}\right) \\ \sin\left(\frac{\theta}{2}\right) & \cos\left(\frac{\theta}{2}\right) \end{bmatrix} \quad R_z(\theta) = \begin{bmatrix} e^{-i\theta/2} & 0 \\ 0 & e^{i\theta/2} \end{bmatrix}$$

- No circuito quântico:

$$|0\rangle \; - \boxed{R_x(\theta)} - \; cos(\tfrac{\theta}{2})|0\rangle - isin(\tfrac{\theta}{2})|1\rangle \qquad |0\rangle \; - \boxed{R_y(\theta)} - \; cos(\tfrac{\theta}{2})|0\rangle + sin(\tfrac{\theta}{2})|1\rangle \qquad |0\rangle \; - \boxed{R_z(\theta)} - \; e^{-i\theta/2}|0\rangle$$

$$|1\rangle \; - \boxed{R_x(\theta)} - \; -isin(\tfrac{\theta}{2})|0\rangle + cos(\tfrac{\theta}{2})|1\rangle \qquad |1\rangle \; - \boxed{R_y(\theta)} - \; -sin(\tfrac{\theta}{2})|0\rangle + cos(\tfrac{\theta}{2})|1\rangle \qquad |1\rangle \; - \boxed{R_z(\theta)} - \; e^{i\theta/2}|1\rangle$$

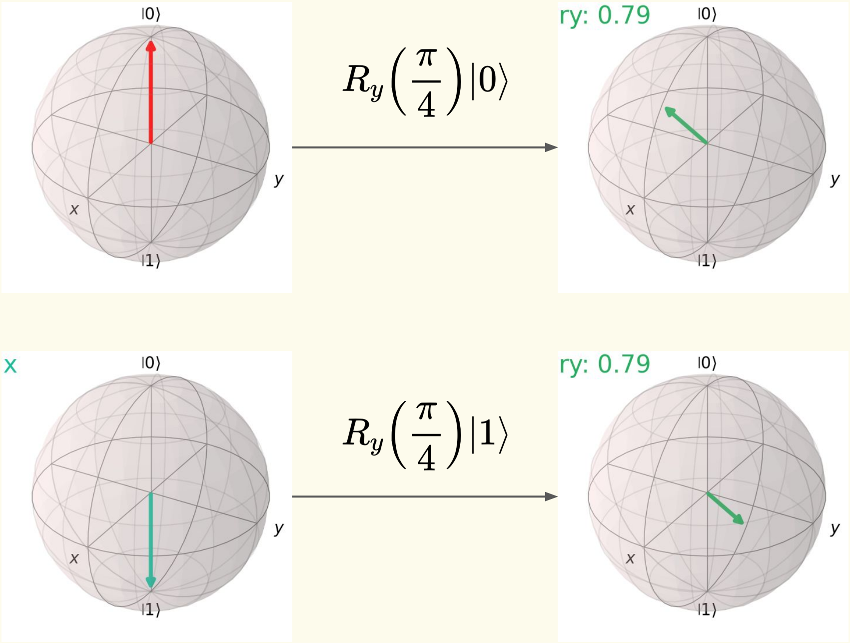
Operadores sobre 1-qubit

Portas elementares

- Na esfera de Bloch:

$$|0\rangle \; — \boxed{R_y(\theta)} — \; cos(\tfrac{\theta}{2})|0\rangle + sin(\tfrac{\theta}{2})|1\rangle$$

$$|1\rangle \; — \boxed{R_y(\theta)} — \; -sin(\tfrac{\theta}{2})|0\rangle + cos(\tfrac{\theta}{2})|1\rangle$$



Operadores sobre 1-qubit

Decomponndo operações

- E se o operador que precisamos não estiver implementado?

- Combinando diferentes portas:

$$U_{ZY} = e^{\alpha} R_z(\beta) R_y(\theta) R_z(\delta) = \begin{bmatrix} e^{i(\alpha-\beta/2-\delta/2)} \cos\left(\frac{\theta}{2}\right) & -e^{i(\alpha-\beta/2+\delta/2)} \sin\left(\frac{\theta}{2}\right) \\ e^{i(\alpha+\beta/2-\delta/2)} \sin\left(\frac{\theta}{2}\right) & e^{i(\alpha+\beta/2+\delta/2)} \cos\left(\frac{\theta}{2}\right) \end{bmatrix}$$

- Como obter α, β, θ e δ ?

Operadores sobre 1-qubit

Decompondo operações

- Se U for composto somente de entradas reais:

$$U_{ZY} = e^{\alpha} R_z(\beta) R_y(\theta) R_z(\delta) = \begin{bmatrix} e^{i(\alpha - \beta/2 - \delta/2)} \cos\left(\frac{\theta}{2}\right) & -e^{i(\alpha-\beta/2+\delta/2)} \sin\left(\frac{\theta}{2}\right) \\ e^{i(\alpha+\beta/2-\delta/2)} \sin\left(\frac{\theta}{2}\right) & e^{i(\alpha+\beta/2+\delta/2)} \cos\left(\frac{\theta}{2}\right) \end{bmatrix} \qquad U = \begin{bmatrix} x_0 & x_1 \\ x_2 & x_3 \end{bmatrix}$$

- Então podemos definir $\alpha, \beta \text{ e } \delta = 0$

- Obtemos θ calculando: $\theta = -2 \arcsin\left(x_1\right)$

Operadores sobre 1-qubit

Decompondo operações

$$U_{ZY} = \begin{bmatrix} e^{i(\alpha-\beta/2-\delta/2)}\cos\left(\frac{\theta}{2}\right) & -e^{i(\alpha-\beta/2+\delta/2)}\sin\left(\frac{\theta}{2}\right) \\ e^{i(\alpha+\beta/2-\delta/2)}\sin\left(\frac{\theta}{2}\right) & e^{i(\alpha+\beta/2+\delta/2)}\cos\left(\frac{\theta}{2}\right) \end{bmatrix}$$

- Se U for composto somente de entradas reais: $U = \begin{bmatrix} x_0 & x_1 \\ x_2 & x_3 \end{bmatrix}$
- Só precisamos setar: $\alpha, \beta \text{ e } \delta = 0$

$$U_{ZY} = \begin{bmatrix} \cos\left(\frac{\theta}{2}\right) & -\sin\left(\frac{\theta}{2}\right) \\ \sin\left(\frac{\theta}{2}\right) & \cos\left(\frac{\theta}{2}\right) \end{bmatrix} = R_y(\theta)\,;$$

Operadores sobre 1-qubit

Decompondo operações

$$U_{ZY} = \begin{bmatrix} e^{i(\alpha-\beta/2-\delta/2)}\cos\left(\frac{\theta}{2}\right) & -e^{i(\alpha-\beta/2+\delta/2)}\sin\left(\frac{\theta}{2}\right) \\ e^{i(\alpha+\beta/2-\delta/2)}\sin\left(\frac{\theta}{2}\right) & e^{i(\alpha+\beta/2+\delta/2)}\cos\left(\frac{\theta}{2}\right) \end{bmatrix}$$

- Se U for composto somente de entradas reais: $U = \begin{bmatrix} x_0 & x_1 \\ x_2 & x_3 \end{bmatrix}$
- Só precisamos setar: $\alpha, \beta \text{ e } \delta = 0$

$$U_{ZY} = \begin{bmatrix} \cos\left(\frac{\theta}{2}\right) & -\sin\left(\frac{\theta}{2}\right) \\ \sin\left(\frac{\theta}{2}\right) & \cos\left(\frac{\theta}{2}\right) \end{bmatrix} = R_y(\theta)\,; \qquad \theta = -2\arcsin(x_1)$$

Operadores sobre 1-qubit

Decompondo operações

$$U_{ZY} = \begin{bmatrix} e^{i(\alpha-\beta/2-\delta/2)}\cos\left(\frac{\theta}{2}\right) & -e^{i(\alpha-\beta/2+\delta/2)}\sin\left(\frac{\theta}{2}\right) \\ e^{i(\alpha+\beta/2-\delta/2)}\sin\left(\frac{\theta}{2}\right) & e^{i(\alpha+\beta/2+\delta/2)}\cos\left(\frac{\theta}{2}\right) \end{bmatrix}$$

- Se U for composto entradas complexas: $U = \begin{bmatrix} x_0 & x_1 \\ x_2 & x_3 \end{bmatrix} = \begin{bmatrix} r_0 e^{ia} & r_1 e^{ib} \\ r_2 e^{ic} & r_3 e^{id} \end{bmatrix}$
- Podemos setar: $\theta = 2\arcsin(r_1)$

Operadores sobre 1-qubit

Decompondo operações

$$U_{ZY} = \begin{bmatrix} e^{i(\alpha-\beta/2\,-\,\delta/2)}\cos\left(\frac{\theta}{2}\right) & -e^{i(\alpha-\beta/2\,+\,\delta/2)}\sin\left(\frac{\theta}{2}\right) \\ e^{i(\alpha+\beta/2\,-\,\delta/2)}\sin\left(\frac{\theta}{2}\right) & e^{i(\alpha+\beta/2\,+\delta/2)}\cos\left(\frac{\theta}{2}\right) \end{bmatrix}$$

- Se U for composto entradas complexas: $U = \begin{bmatrix} x_0 & x_1 \\ x_2 & x_3 \end{bmatrix} = \begin{bmatrix} r_0e^{ia} & r_1e^{ib} \\ r_2e^{ic} & r_3e^{id} \end{bmatrix}$
- Podemos setar: $\theta = 2\arcsin\left(r_1\right)$
- Unitariedade:

$$\cos\left(\theta\right)\sin\left(\theta\right)e^{i(a-c)} - \sin\left(\theta\right)\cos\left(\theta\right)e^{i(b-d)} = 0$$

$$UU^\dagger = \begin{bmatrix} e^{ia}\cos\left(\theta\right) & -e^{ib}\sin\left(\theta\right) \\ \sin\left(\theta\right)e^{ic} & e^{id}\cos\left(\theta\right) \end{bmatrix} \begin{bmatrix} e^{-ia}\cos\left(\theta\right) & \sin\left(\theta\right)e^{-ic} \\ -e^{-ib}\sin\left(\theta\right) & e^{-id}\cos\left(\theta\right) \end{bmatrix} ;$$

$$a - c = b - d$$
$$d = b + c - a$$

Operadores sobre 1-qubit

Decompondo operações

$$U_{ZY} = \begin{bmatrix} e^{i(\alpha-\beta/2-\delta/2)}\cos\left(\frac{\theta}{2}\right) & -e^{i(\alpha-\beta/2+\delta/2)}\sin\left(\frac{\theta}{2}\right) \\ e^{i(\alpha+\beta/2-\delta/2)}\sin\left(\frac{\theta}{2}\right) & e^{i(\alpha+\beta/2+\delta/2)}\cos\left(\frac{\theta}{2}\right) \end{bmatrix}$$

- Se U for composto entradas complexas: $U = \begin{bmatrix} x_0 & x_1 \\ x_2 & x_3 \end{bmatrix} = \begin{bmatrix} r_0 e^{ia} & r_1 e^{ib} \\ r_2 e^{ic} & r_3 e^{id} \end{bmatrix}$
- Podemos setar: $\theta = 2\arcsin\left(r_1\right)$

Operadores sobre 1-qubit

Decompondo operações

$$U_{ZY} = \begin{bmatrix} e^{i(\alpha-\beta/2\,-\,\delta/2)}\cos\left(\frac{\theta}{2}\right) & -e^{i(\alpha-\beta/2\,+\,\delta/2)}\sin\left(\frac{\theta}{2}\right) \\ e^{i(\alpha+\beta/2\,-\,\delta/2)}\sin\left(\frac{\theta}{2}\right) & e^{i(\alpha+\beta/2\,+\delta/2)}\cos\left(\frac{\theta}{2}\right) \end{bmatrix}$$

- Se U for composto entradas complexas: $U = \begin{bmatrix} x_0 & x_1 \\ x_2 & x_3 \end{bmatrix} = \begin{bmatrix} r_0 e^{ia} & r_1 e^{ib} \\ r_2 e^{ic} & r_3 e^{id} \end{bmatrix}$
- Podemos setar: $\theta = 2\arcsin\left(r_1\right)$
- Unitariedade:

$$UU^{\dagger} = \begin{bmatrix} e^{ia}\cos\left(\theta\right) & -e^{ib}\sin\left(\theta\right) \\ \sin\left(\theta\right)e^{ic} & e^{id}\cos\left(\theta\right) \end{bmatrix} \begin{bmatrix} e^{-ia}\cos\left(\theta\right) & \sin\left(\theta\right)e^{-ic} \\ -e^{-ib}\sin\left(\theta\right) & e^{-id}\cos\left(\theta\right) \end{bmatrix} ;$$

Operadores sobre 1-qubit

Decompondo operações

$$U_{ZY} = \begin{bmatrix} e^{i(\alpha-\beta/2-\delta/2)}\cos\left(\frac{\theta}{2}\right) & -e^{i(\alpha-\beta/2+\delta/2)}\sin\left(\frac{\theta}{2}\right) \\ e^{i(\alpha+\beta/2-\delta/2)}\sin\left(\frac{\theta}{2}\right) & e^{i(\alpha+\beta/2+\delta/2)}\cos\left(\frac{\theta}{2}\right) \end{bmatrix}$$

- Se U for composto entradas complexas: $U = \begin{bmatrix} x_0 & x_1 \\ x_2 & x_3 \end{bmatrix} = \begin{bmatrix} r_0 e^{ia} & r_1 e^{ib} \\ r_2 e^{ic} & r_3 e^{id} \end{bmatrix}$
- Podemos setar: $\theta = 2\arcsin\left(r_1\right)$
- Unitariedade:

$$\cos(\theta)\sin(\theta)e^{i(a-c)} - \sin(\theta)\cos(\theta)e^{i(b-d)} = 0$$

$$UU^\dagger = \begin{bmatrix} e^{ia}\cos(\theta) & -e^{ib}\sin(\theta) \\ \sin(\theta)e^{ic} & e^{id}\cos(\theta) \end{bmatrix} \begin{bmatrix} e^{-ia}\cos(\theta) & \sin(\theta)e^{-ic} \\ -e^{-ib}\sin(\theta) & e^{-id}\cos(\theta) \end{bmatrix};$$

Qiskit | Fall Fest 2023

Operadores sobre 1-qubit

Decompondo operações

$$U_{ZY} = \begin{bmatrix} e^{i(\alpha-\beta/2-\delta/2)}\cos\left(\frac{\theta}{2}\right) & -e^{i(\alpha-\beta/2+\delta/2)}\sin\left(\frac{\theta}{2}\right) \\ e^{i(\alpha+\beta/2-\delta/2)}\sin\left(\frac{\theta}{2}\right) & e^{i(\alpha+\beta/2+\delta/2)}\cos\left(\frac{\theta}{2}\right) \end{bmatrix}$$

- Se U for composto entradas complexas: $U = \begin{bmatrix} x_0 & x_1 \\ x_2 & x_3 \end{bmatrix} = \begin{bmatrix} r_0 e^{ia} & r_1 e^{ib} \\ r_2 e^{ic} & r_3 e^{id} \end{bmatrix}$
- Podemos setar: $\theta = 2\arcsin(r_1)$
- Unitariedade:

$$\cos(\theta)\sin(\theta)e^{i(a-c)} - \sin(\theta)\cos(\theta)e^{i(b-d)} = 0$$

$$UU^\dagger = \begin{bmatrix} \boxed{e^{ia}\cos(\theta) \quad -e^{ib}\sin(\theta)} \\ \sin(\theta)e^{ic} \quad e^{id}\cos(\theta) \end{bmatrix} \begin{bmatrix} e^{-ia}\cos(\theta) & \sin(\theta)e^{-ic} \\ -e^{-ib}\sin(\theta) & e^{-id}\cos(\theta) \end{bmatrix};$$

$$a - c = b - d$$
$$d = b + c - a$$

Qiskit | Fall Fest 2023

Operadores sobre 1-qubit

Decompondo operações

$$U_{ZY} = \begin{bmatrix} e^{i(\alpha-\beta/2-\delta/2)}\cos\left(\frac{\theta}{2}\right) & -e^{i(\alpha-\beta/2+\delta/2)}\sin\left(\frac{\theta}{2}\right) \\ e^{i(\alpha+\beta/2-\delta/2)}\sin\left(\frac{\theta}{2}\right) & e^{i(\alpha+\beta/2+\delta/2)}\cos\left(\frac{\theta}{2}\right) \end{bmatrix}$$

- Com: $d = b + c - a$
- Computar os outros ângulos resolvendo o sistema:

$$\begin{cases} \alpha - \beta/2 - \delta/2 &= a \\ \alpha - \beta/2 + \delta/2 &= b + \pi \\ \alpha + \beta/2 - \delta/2 &= c \end{cases} \longrightarrow \begin{cases} \alpha &= (a + b + \pi)/2 \\ \beta &= c - a \\ \delta &= b - a + \pi \end{cases}$$

Operadores sobre 1-qubit

Requisitos de implementação

- Linguagem de programação:

`Versão 3.10`

- Ambiente de execução:

`https://colab.research.google.com`

Operadores sobre 1-qubit

Requisitos de implementação

- Instalando bibliotecas e importando módulos

```
!pip install qiskit pylatexenc --quiet
```

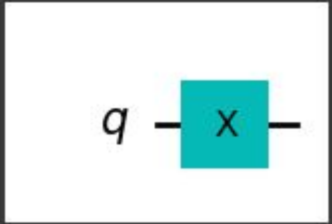
```
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 162.6/162.6 kB 3.3 MB/s eta 0:00:00
  Preparing metadata (setup.py) ... done
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 6.2/6.2 MB 47.6 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 2.0/2.0 MB 55.0 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 49.6/49.6 kB 3.7 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 115.3/115.3 kB 9.9 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 49.6/49.6 kB 1.6 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 37.5/37.5 MB 14.2 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 112.7/112.7 kB 10.6 MB/s eta 0:00:00
  Building wheel for pylatexenc (setup.py) ... done
```

```
[2] import numpy as np
    from qiskit import QuantumCircuit
```

Operadores sobre 1-qubit

Aplicando Operadores Elementares

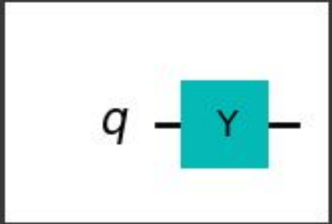
```
qc = QuantumCircuit(1)

qc.x(0)
qc.draw("mpl")
```



Operadores sobre 1-qubit

Aplicando Operadores Elementares

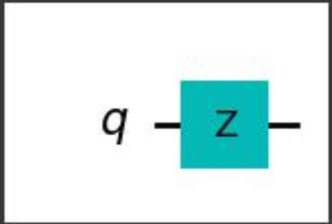
```
qc = QuantumCircuit(1)

qc.y(0)
qc.draw("mpl")
```



Operadores sobre 1-qubit

Aplicando Operadores Elementares

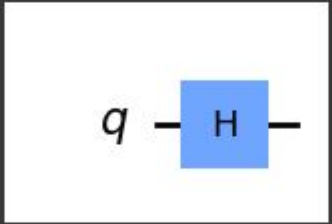
```
qc = QuantumCircuit(1)

qc.z(0)
qc.draw("mpl")
```



Operadores sobre 1-qubit

Aplicando Operadores Elementares

```
qc = QuantumCircuit(1)

qc.h(0)
qc.draw("mpl")
```

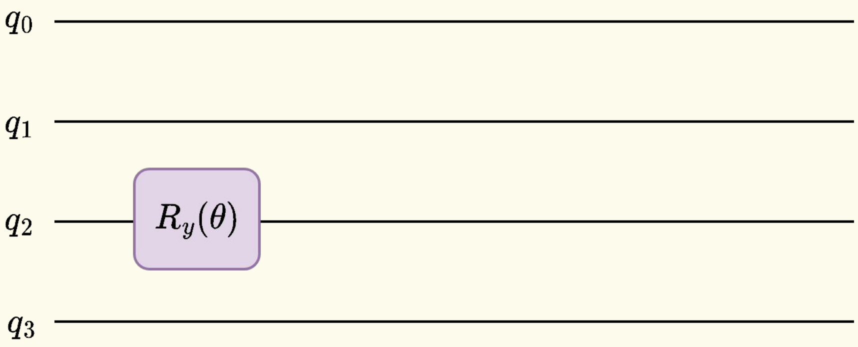


Operadores sobre 1-qubit

Aplicando Parametrizadas

- Índice do qubit
- Parametro a ser utilizado

```
qc.ry(theta,2)
```



Operadores sobre 1-qubit

Aplicando Parametrizadas

- Decomposição ZY

$$U = e^{\alpha} R_z(\beta) R_y(\theta) R_z(\delta) = \begin{bmatrix} e^{i(\alpha - \beta/2 - \delta/2)} \cos\left(\frac{\theta}{2}\right) & -e^{i(\alpha-\beta/2+\delta/2)} \sin\left(\frac{\theta}{2}\right) \\ e^{i(\alpha+\beta/2-\delta/2)} \sin\left(\frac{\theta}{2}\right) & e^{i(\alpha+\beta/2+\delta/2)} \cos\left(\frac{\theta}{2}\right) \end{bmatrix}$$

Operadores sobre 1-qubit

Aplicando Parametrizadas

- Funções auxiliares para teste

```
ry_op = lambda t: np.array([[np.cos(t/2), -np.sin(t/2)],
                            [np.sin(t/2), np.cos(t/2)]])
rz_op  = lambda xhi : np.array([[np.e**(-1j*xhi/2),  0             ],
                                [0,                 np.e**(1j*xhi/2)]])

scalar = lambda s: np.e**(1j*s)
```

Operadores sobre 1-qubit

Aplicando Parametrizadas

- Matriz unitária aleatória

```
column = np.random.rand(2) + 1j*np.random.rand(2)
column = column / np.linalg.norm(column)

matrix = np.array([[column[0], -column[1].conj()],
                   [column[1], column[0].conj()]])
matrix
```

```
array([[ 0.33368289+0.51020004j, -0.36277345+0.70480286j],
       [ 0.36277345+0.70480286j,  0.33368289-0.51020004j]])
```

Operadores sobre 1-qubit

Aplicando Parametrizadas

- Recriando matriz utilizando parâmetros

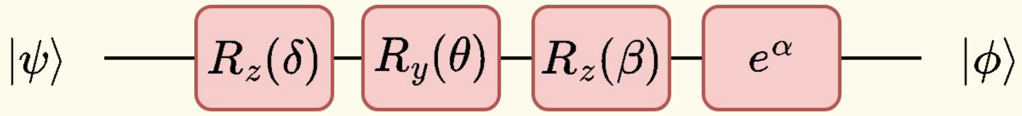
```
[177] theta = 2*np.arcsin(np.abs(matrix[1, 0]))

      a, b, c = np.angle(matrix[0, 0]), np.angle(matrix[0, 1]), np.angle(matrix[1, 0])

      alpha = (c + b + np.pi) / 2
      beta = c - a
      delta = b - a + np.pi

      m_2 = scalar(alpha)*(rz_op(beta) @ ry_op(theta) @ rz_op(delta))
      m_2

      array([[ 0.33368289+0.51020004j, -0.36277345+0.70480286j],
             [ 0.36277345+0.70480286j,  0.33368289-0.51020004j]])

[178] np.allclose(m_2, matrix)

      True
```

$$\begin{bmatrix} r_0 e^{ia} & r_1 e^{ib} \\ r_2 e^{ic} & r_3 e^{id} \end{bmatrix}$$

Operadores sobre 1-qubit

Aplicando Parametrizadas

- Aplicando operadores
- Matricialmente: $e^{\alpha} R_z(\beta) R_y(\theta) R_z(\delta) |\psi\rangle$
- No circuito quântico:

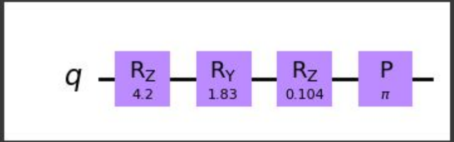


Operadores sobre 1-qubit

Aplicando Parametrizadas

- Implementando no Qiskit

```
[176] qc = QuantumCircuit(1)
      qc.rz(delta, 0)
      qc.ry(theta, 0)
      qc.rz(beta, 0)
      qc.p(alpha, 0)
      qc.draw("mpl")
```



Operadores sobre 2-qubits

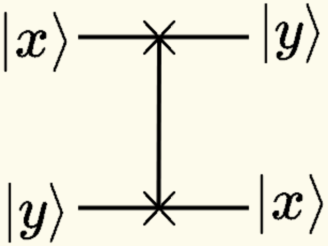
SWAP, CNOT e Operadores Controlados

Operadores sobre 2-qubits

- Operador swap

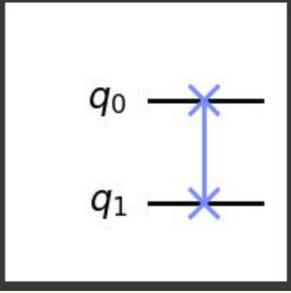
$$SWAP = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

- No circuito quântico



- Implementando no Qiskit

```
qc = QuantumCircuit(2)
qc.swap(0, 1)
qc.draw("mpl")
```



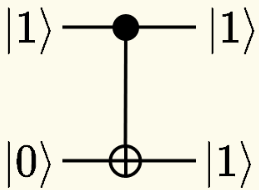
Operadores sobre 2-qubits

Operadores controlados

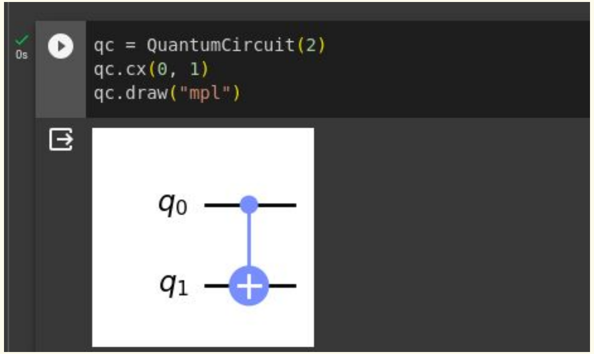
- Not controlado - CNOT

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

- No circuito quântico



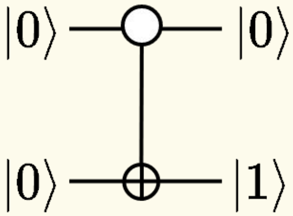
- Implementando no Qiskit:



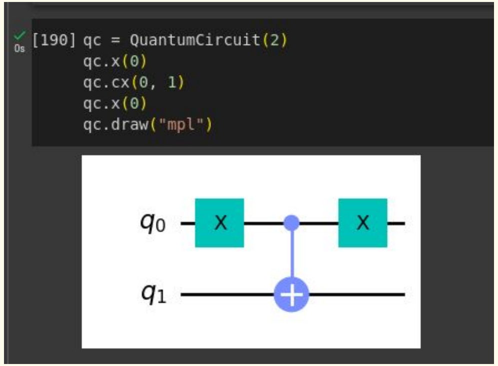
Operadores sobre 2-qubits

Operadores controlados

- Controle Aberto:



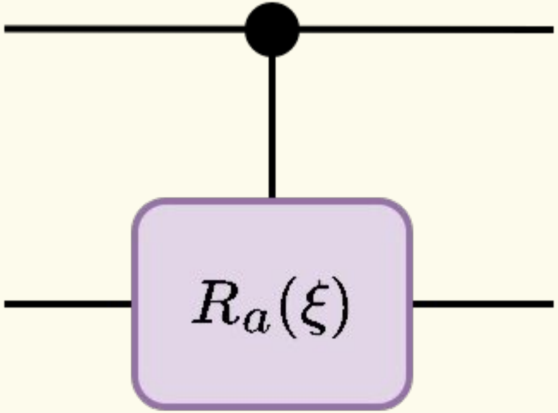
- Implementando no Qiskit:



Operadores sobre 2-qubits

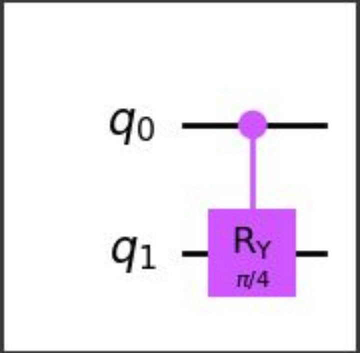
Operadores controlados

- Rotações Controladas:



$$a \in \{x, y, z\}$$

```
[189] qc = QuantumCircuit(2)
      qc.cry(np.pi/4, 0, 1)
      qc.draw("mpl")
```



Operadores sobre 2-qubits

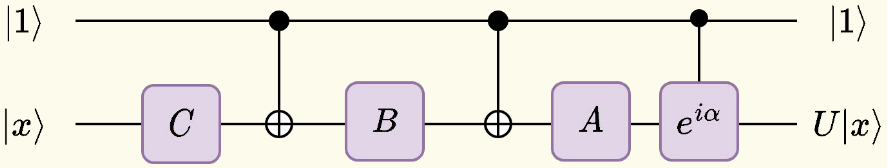
Operadores controlados

- E se operador que eu precisar não estiver disponível?

Operadores sobre 2-qubits

Operadores controlados

- Combinando operadores



- Então:

$$AXBXC = U$$

- Consequentemente:

$$ABC = I$$

Operadores sobre 2-qubits

Operadores controlados

- Ângulos da decomposição ZY:

$$\alpha, \beta, \theta \text{ e } \delta$$

- Definindo:

$$A \equiv R_z(\beta)R_y(\theta/2) \qquad B \equiv R_y(-\theta/2)R_z(-(\delta+\beta)/2) \qquad C \equiv R_z((\delta-\beta)/2)$$

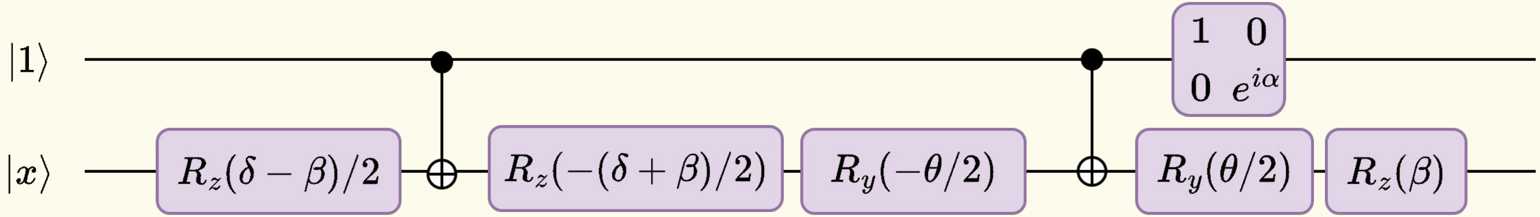
- E também:

$$\begin{bmatrix} 1 & 0 \\ 0 & e^{i\alpha} \end{bmatrix}$$

Operadores sobre 2-qubits

Operadores controlados

- No circuito quântico:



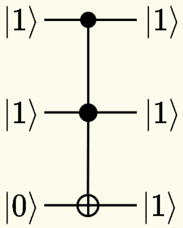
Operadores sobre Mais de um qubit

Toffoli, Swap Controlado

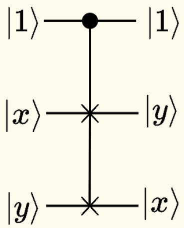
Operadores sobre 2-qubits

Operadores controlados

- Porta Toffoli:



- Porta CSWAP:



- Utilizando no Qiskit

```
[54] qc = QuantumCircuit(3)

     qc.ccx(0, 1, 2)
```

```
qc = QuantumCircuit(3)
qc.cswap(0, 1, 2)
```

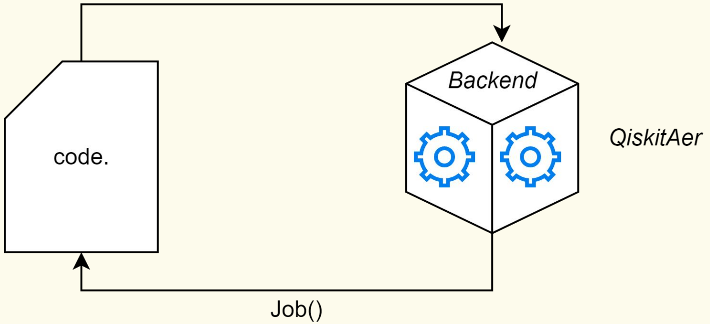
Simulando Circuitos Quânticos

Simulando Circuitos Quânticos

- Front end

  `qc.x, qc.y, qc.z, qc.h, qc.rx, qc.ry, qc.rz, …`

- Backend



Simulando Circuitos Quânticos

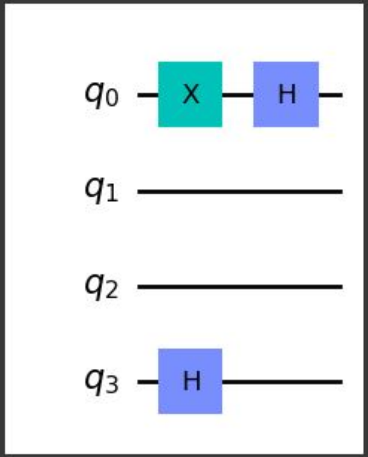
- Utilizando Módulo Aer:

```
[67] !pip install qiskit_aer --quiet

[72] from qiskit import Aer
```

Simulando Circuitos Quânticos

- Circuito Quântico:

```
qc = QuantumCircuit(4)
qc.x(0)
qc.h(0)
qc.h(3)
qc.draw("mpl")
```



Simulando Circuitos Quânticos

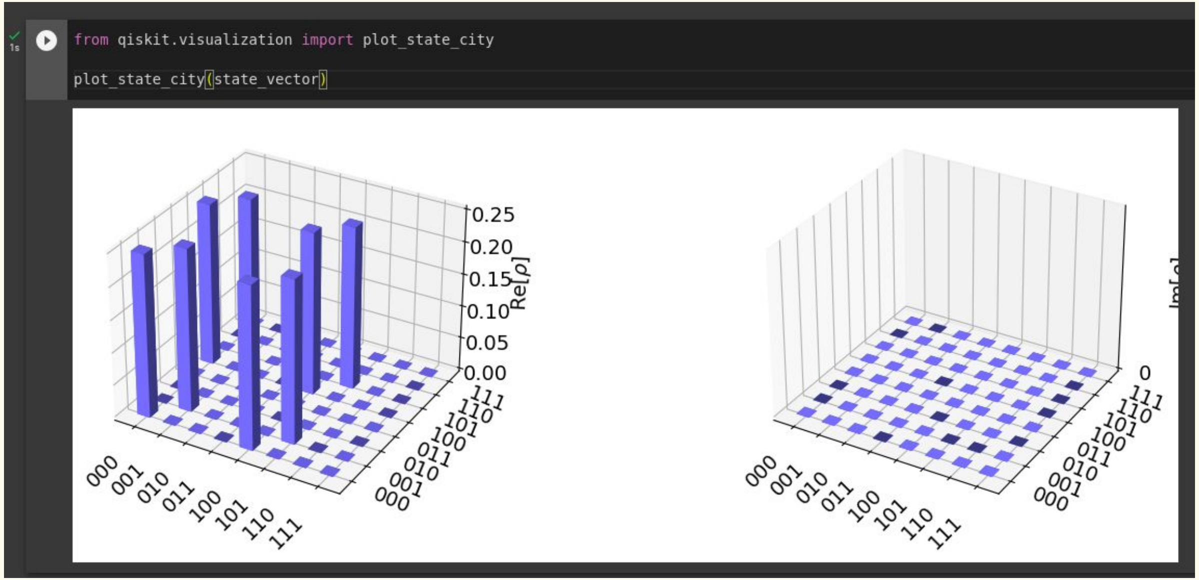
- Simulando Vetor de Estados:

```
sv_simulator = Aer.get_backend("statevector_simulator")
job = sv_simulator.run(qc)
result = job.result()
state_vector = result.get_statevector()
state_vector.draw("latex")
```

$$\frac{1}{2}|000\rangle - \frac{1}{2}|001\rangle + \frac{1}{2}|100\rangle - \frac{1}{2}|101\rangle$$

Simulando Circuitos Quânticos

- Visualizando:



Simulando Circuitos Quânticos

- Simulação unitária:

```
[102] from qiskit.visualization import array_to_latex
```

```
u_simulator = Aer.get_backend("unitary_simulator")
job = u_simulator.run(qc)
result = job.result()
unitary_matrix = result.get_unitary()
array_to_latex(unitary_matrix)
```

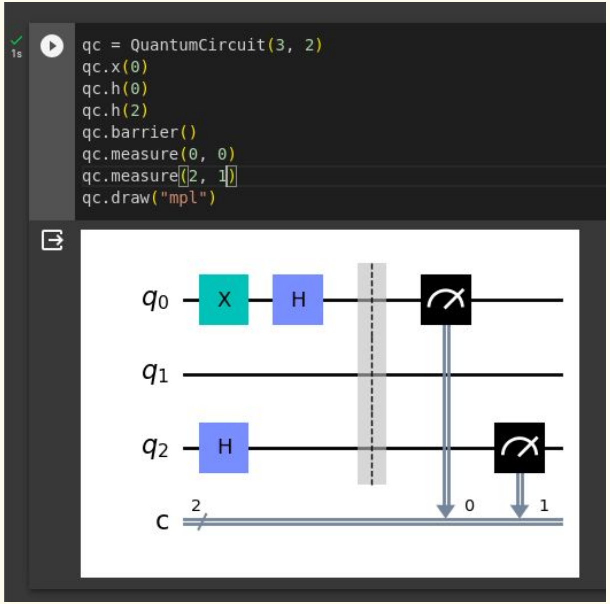
$$\begin{bmatrix} \frac{1}{2} & \frac{1}{2} & 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 \\ -\frac{1}{2} & \frac{1}{2} & 0 & 0 & -\frac{1}{2} & \frac{1}{2} & 0 & 0 \\ 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & \frac{1}{2} & \frac{1}{2} \\ 0 & 0 & -\frac{1}{2} & \frac{1}{2} & 0 & 0 & -\frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & \frac{1}{2} & 0 & 0 & -\frac{1}{2} & -\frac{1}{2} & 0 & 0 \\ -\frac{1}{2} & \frac{1}{2} & 0 & 0 & \frac{1}{2} & -\frac{1}{2} & 0 & 0 \\ 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & -\frac{1}{2} & -\frac{1}{2} \\ 0 & 0 & -\frac{1}{2} & \frac{1}{2} & 0 & 0 & \frac{1}{2} & -\frac{1}{2} \end{bmatrix}$$

Simulando Circuitos Quânticos

- Vetor de estado e Matriz Unitária
  - Saídas ideais
  - Conferir se estado produzido corresponde ao esperado
  - Conferir se o circuito implementa o operador planejado
- Experimentos reais
  - Medição dos qubits diversas vezes
  - Saída probabilística

Simulando Circuitos Quânticos

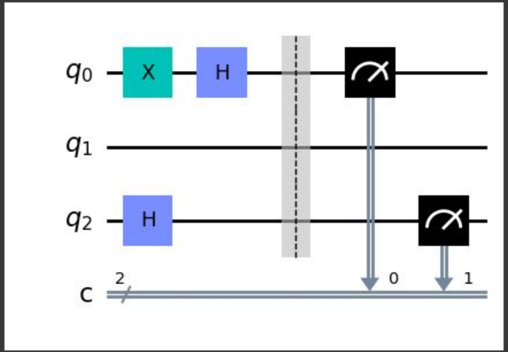
- *QasmSimulator*
  - Simula saída probabilística
  - Necessita da utilização de registradores clássicos
- Implementação do circuito:



Simulando Circuitos Quânticos

- Implementação do circuito:

```python
qc = QuantumCircuit(3, 2)
qc.x(0)
qc.h(0)
qc.h(2)
qc.barrier()
qc.measure(0, 0)
qc.measure(2, 1)
qc.draw("mpl")
```



Simulando Circuitos Quânticos

- Executando o simulador:

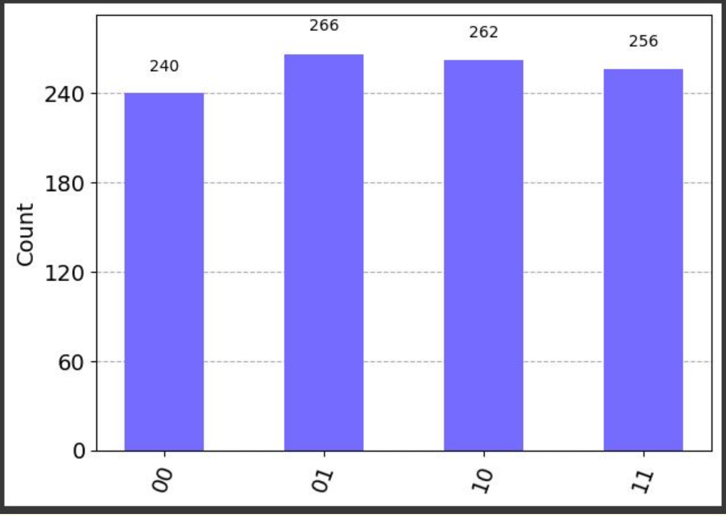
```
qasm_simulator = Aer.get_backend("qasm_simulator")
job = qasm_simulator.run(qc, shots=1024)
result = job.result()
counts = result.get_counts()
counts
```

```
{'01': 247, '11': 262, '10': 256, '00': 259}
```

Simulando Circuitos Quânticos

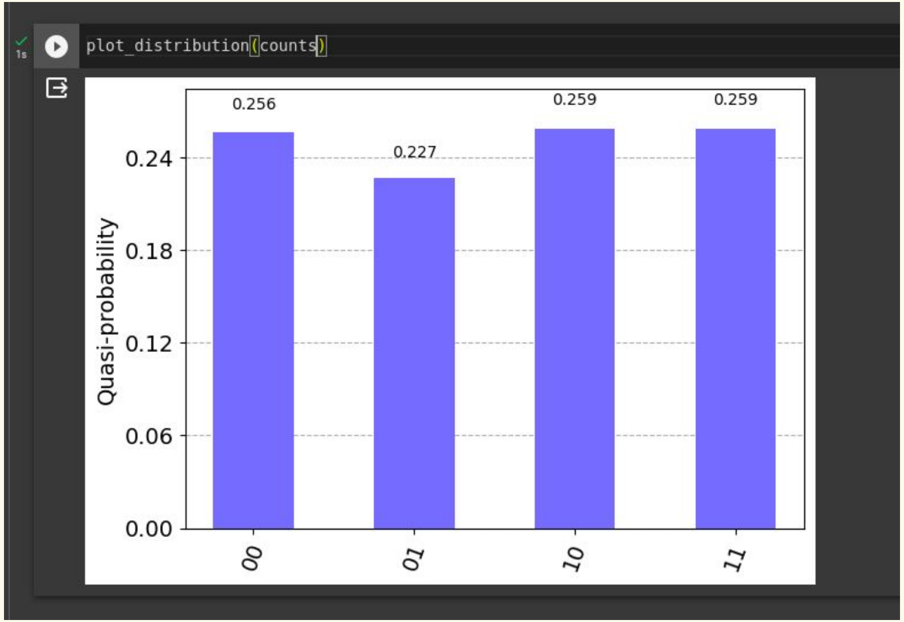
- Visualizando contagens

```
from qiskit.visualization import plot_histogram

plot_histogram(counts)
```



Simulando Circuitos Quânticos

- Visualizando Frequência relativa



Aplicação: Circuitos de Bell

Preparando estados de Bell utilizando Qiskit

Aplicação: Circuitos de Bell

Preparando estados de Bell utilizando Qiskit

- Estados de Bell

$$\beta_{00} = (|00\rangle + |11\rangle)/\sqrt{2}$$

$$\beta_{01} = (|01\rangle + |10\rangle)/\sqrt{2}$$

$$\beta_{10} = (|00\rangle - |11\rangle)/\sqrt{2}$$

$$\beta_{11} = (|01\rangle - |10\rangle)/\sqrt{2}$$

- Teleporte Quântico
- Codificação Superdensa

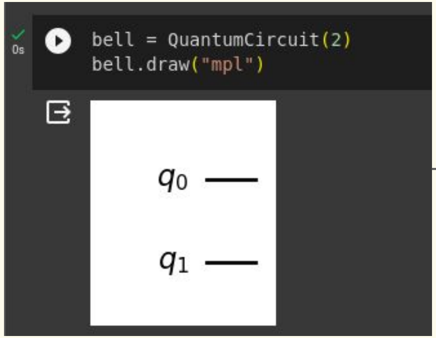
Aplicação: Circuitos de Bell

Preparando estados de Bell utilizando Qiskit

- Preparando

$$\beta_{00} = (|00\rangle + |11\rangle)/\sqrt{2}$$

- Inicialmente



$$\rightarrow |00\rangle$$

Aplicação: Circuitos de Bell

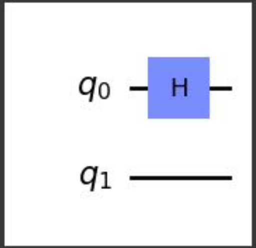
Preparando estados de Bell utilizando Qiskit

- Preparando

$$\beta_{00} = (|00\rangle + |11\rangle)/\sqrt{2}$$

- Passo 1

```
[28] bell.h(0)
     bell.draw("mpl")
```



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

Aplicação: Circuitos de Bell

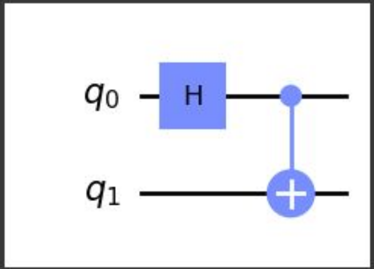
Preparando estados de Bell utilizando Qiskit

- Preparando

$$\beta_{00} = (|00\rangle + |11\rangle)/\sqrt{2}$$

- Passo 2

```
bell.cx(0, 1)
bell.draw("mpl")
```



$$\rightarrow (|00\rangle + |11\rangle)/\sqrt{2}$$

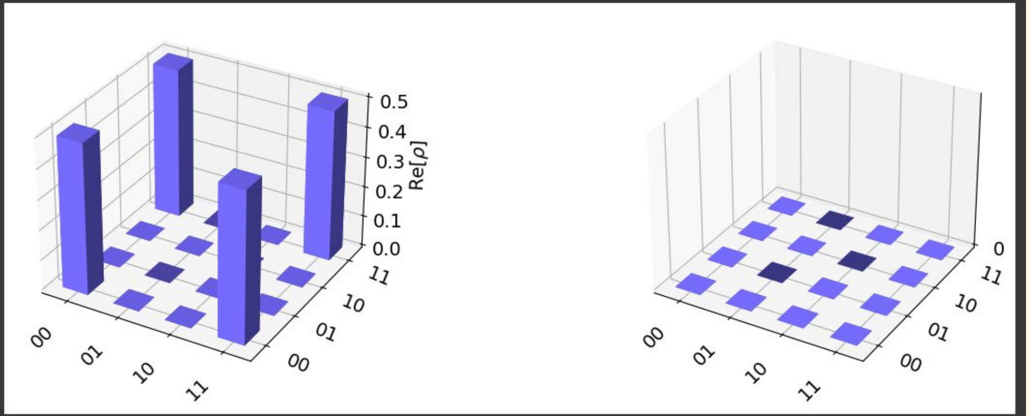
Aplicação: Circuitos de Bell

Preparando estados de Bell utilizando Qiskit

- Preparando

$$\beta_{00} = (|00\rangle + |11\rangle)/\sqrt{2}$$

```
job = sv_simulator.run(bell)
result = job.result()
bell_state = result.get_statevector()

plot_state_city(bell_state)
```



Desafio

Desafio

- Escolha um estado de Bell

$$\beta_{01} = (|01\rangle + |10\rangle)/\sqrt{2}$$

$$\beta_{10} = (|00\rangle - |11\rangle)/\sqrt{2}$$

$$\beta_{11} = (|01\rangle - |10\rangle)/\sqrt{2}$$

- Implementar um circuito que carrega o estado escolhido

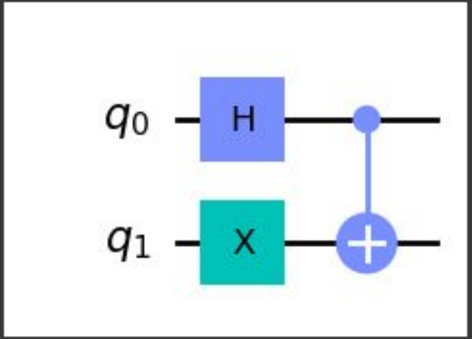
Desafio

Resposta

- Escolha um estado de Bell

$$\beta_{01} = (|01\rangle + |10\rangle)/\sqrt{2}$$

```python
bell_01 = QuantumCircuit(2)
bell_01.x(1)
bell_01.h(0)
bell_01.cx(0, 1)
bell_01.draw("mpl")
```



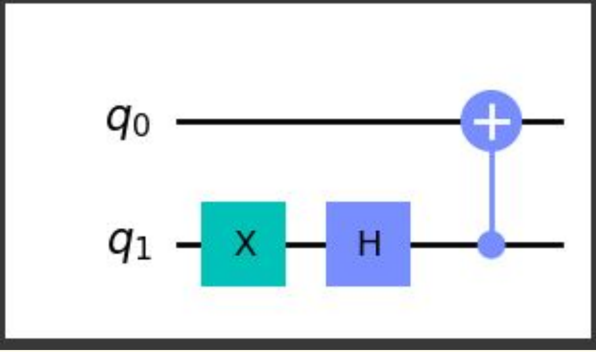
Desafio

Resposta

- Escolha um estado de Bell

$$\beta_{10} = (|00\rangle - |11\rangle)/\sqrt{2}$$

```
bell_10 = QuantumCircuit(2)
bell_10.x(1)
bell_10.h(1)
bell_10.cx(1, 0)
bell_10.draw("mpl")
```

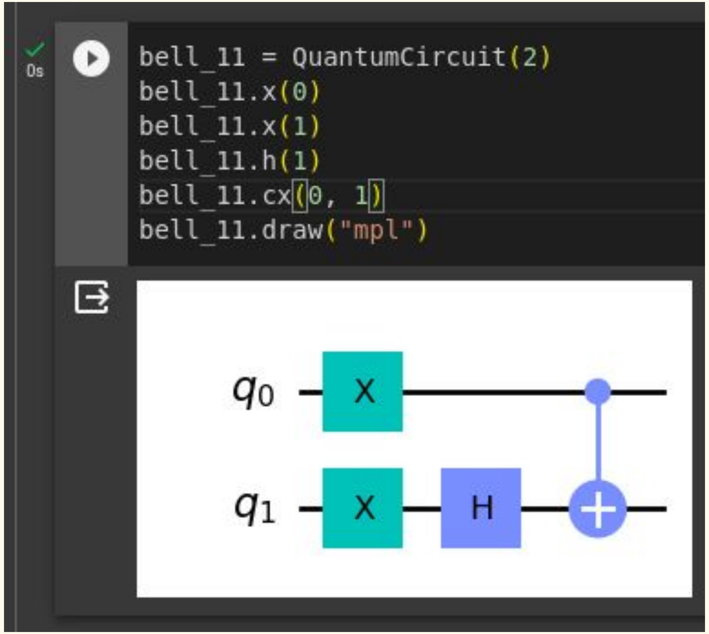


Desafio

Resposta

- Escolha um estado de Bell

$$\beta_{11} = (|01\rangle - |10\rangle)/\sqrt{2}$$



Obrigado

Name of presentation

Firstname Lastname
Job titles

Qiskit | Fall Fest 2023

Name of presentation

Firstname Lastname

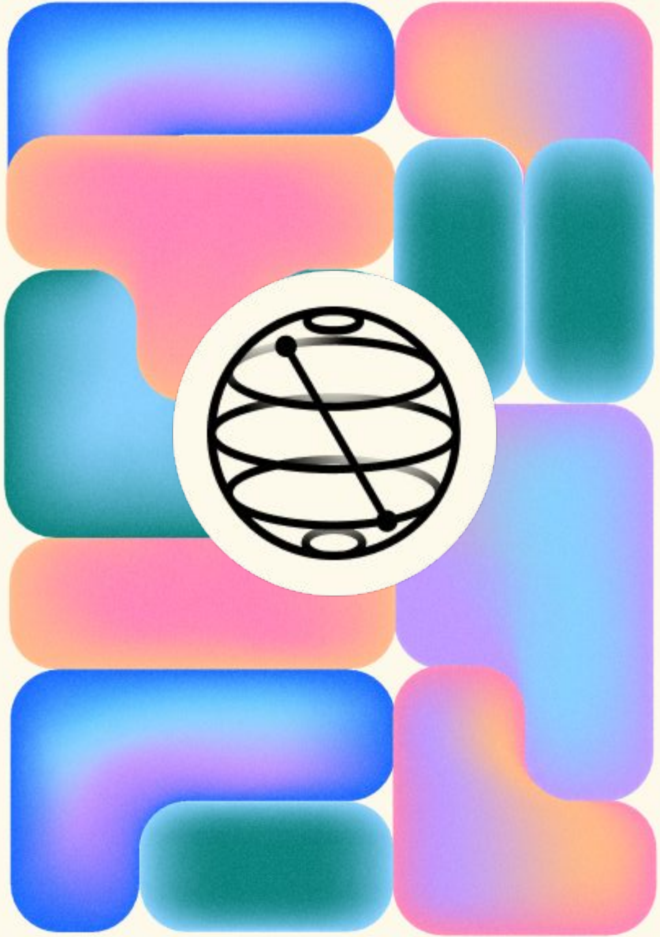
Job titles

Qiskit | Fall Fest 2023

Name of presentation

Firstname Lastname

Job titles



Qiskit | Fall Fest 2023

Thank you

Thank you