Computação Quântica: Primeiros Passos para a Programação

Regina Melo Silveira

LARC- Laboratório de Arquitetura e Redes de Computadores

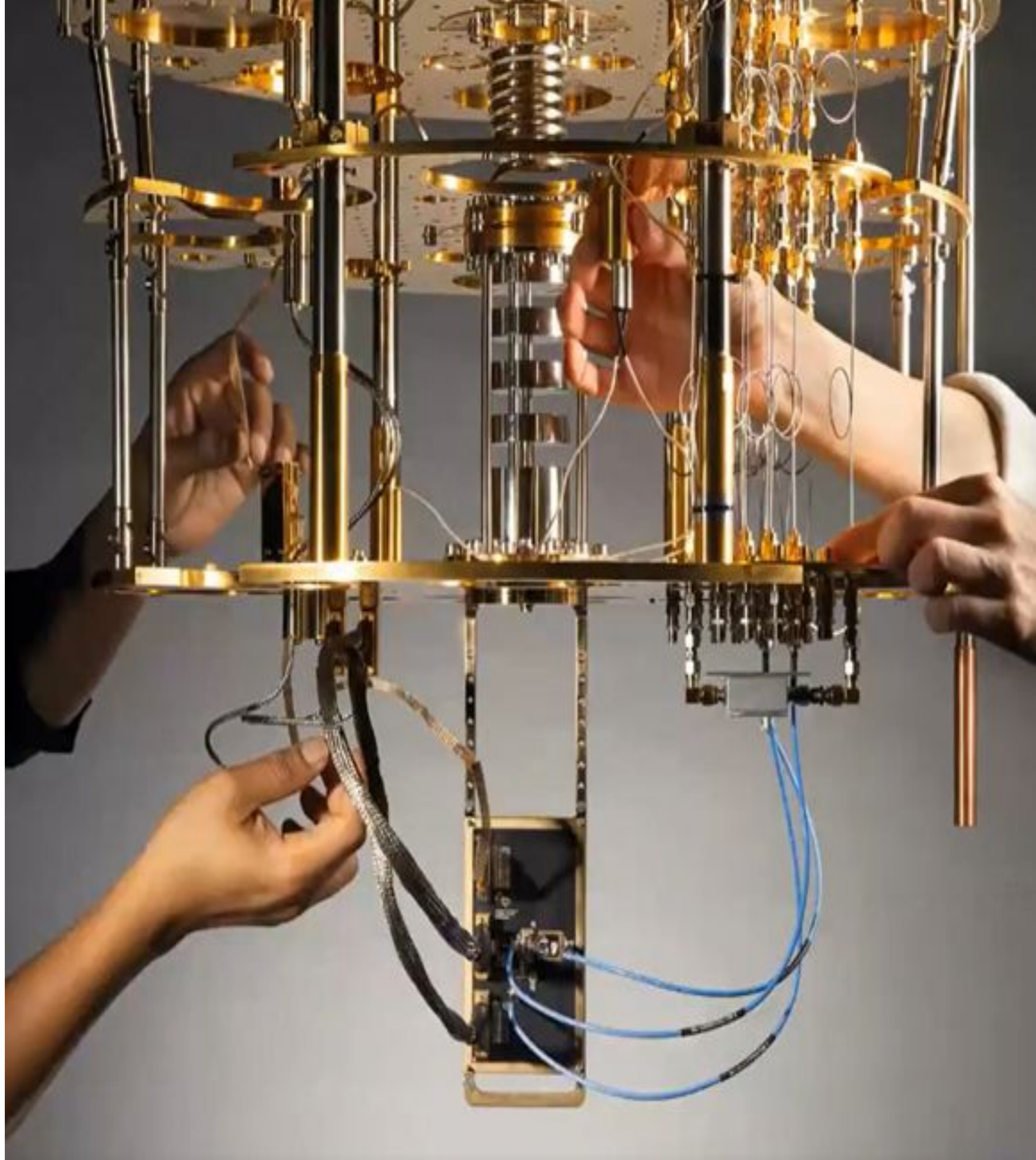
Escola Politécnica da Universidade de São Paulo

regina@larc.usp.br

Agradecimento ao
Nic.Br pelo convite

Agradecimento especial
a equipe do
LARC/PCS/EPUSP

Wilson Ruggiero
Graça Bressan
Waldemir Cambiucci
Jonatas Rossetti



Agenda

A Computação Quântica

Características quânticas, qubits e as evoluções alcançadas

Aspectos da Mecânica Quântica

Espaço de Hilbert

Representação matemática

Representação Computacional

Esfera de Bloch

Portas quânticas

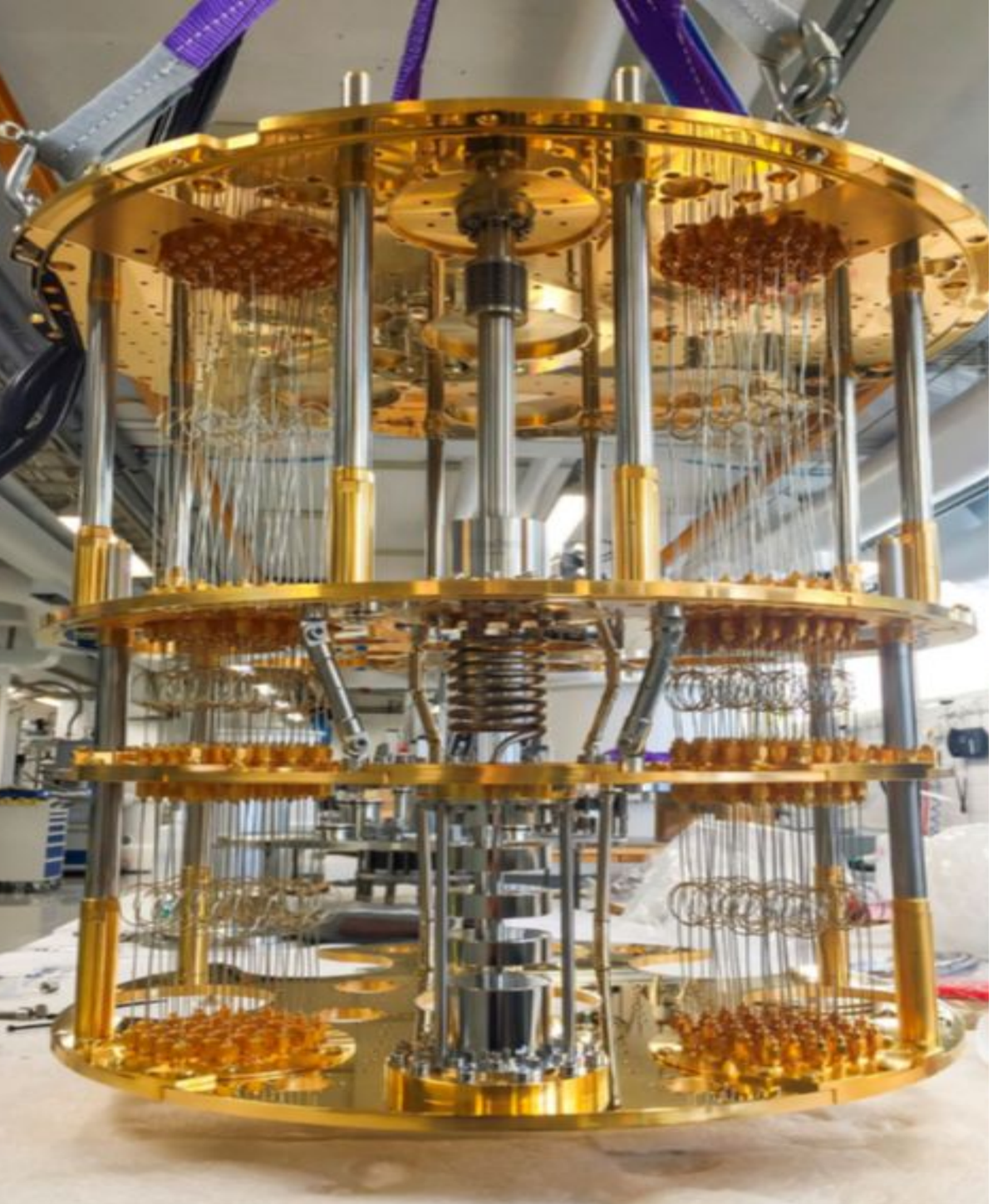
Circuitos Quânticos

Algoritmo Quântico

Exemplos

Uso do QisKit da IBM

A Computação Quântica



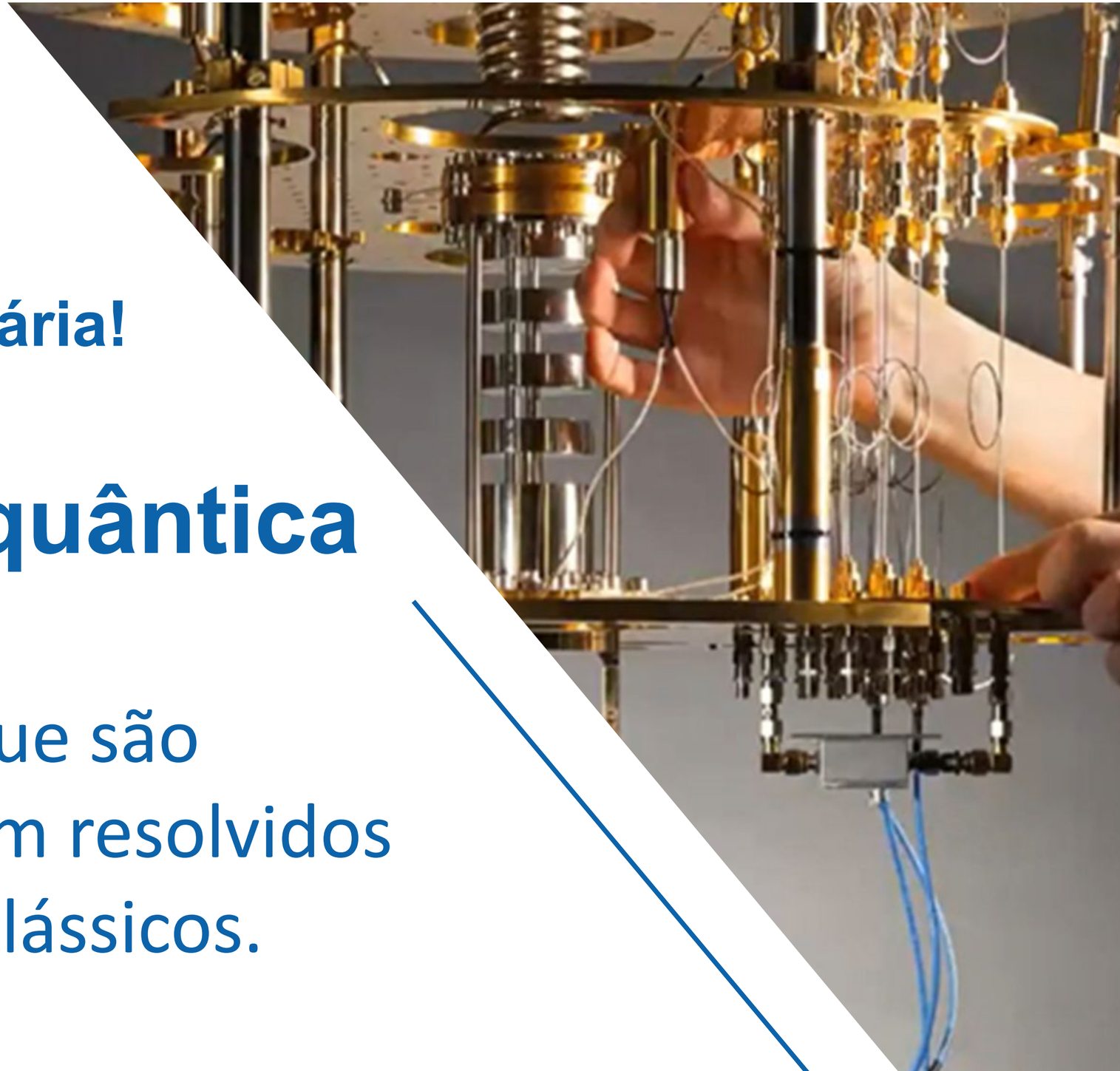
O que é Computação Quântica?

É um novo **modelo computacional** baseado em **princípios da mecânica quântica**, que pode resolver problemas muito complexos para computadores clássicos.

A Natureza não é binária!

A Natureza é quântica

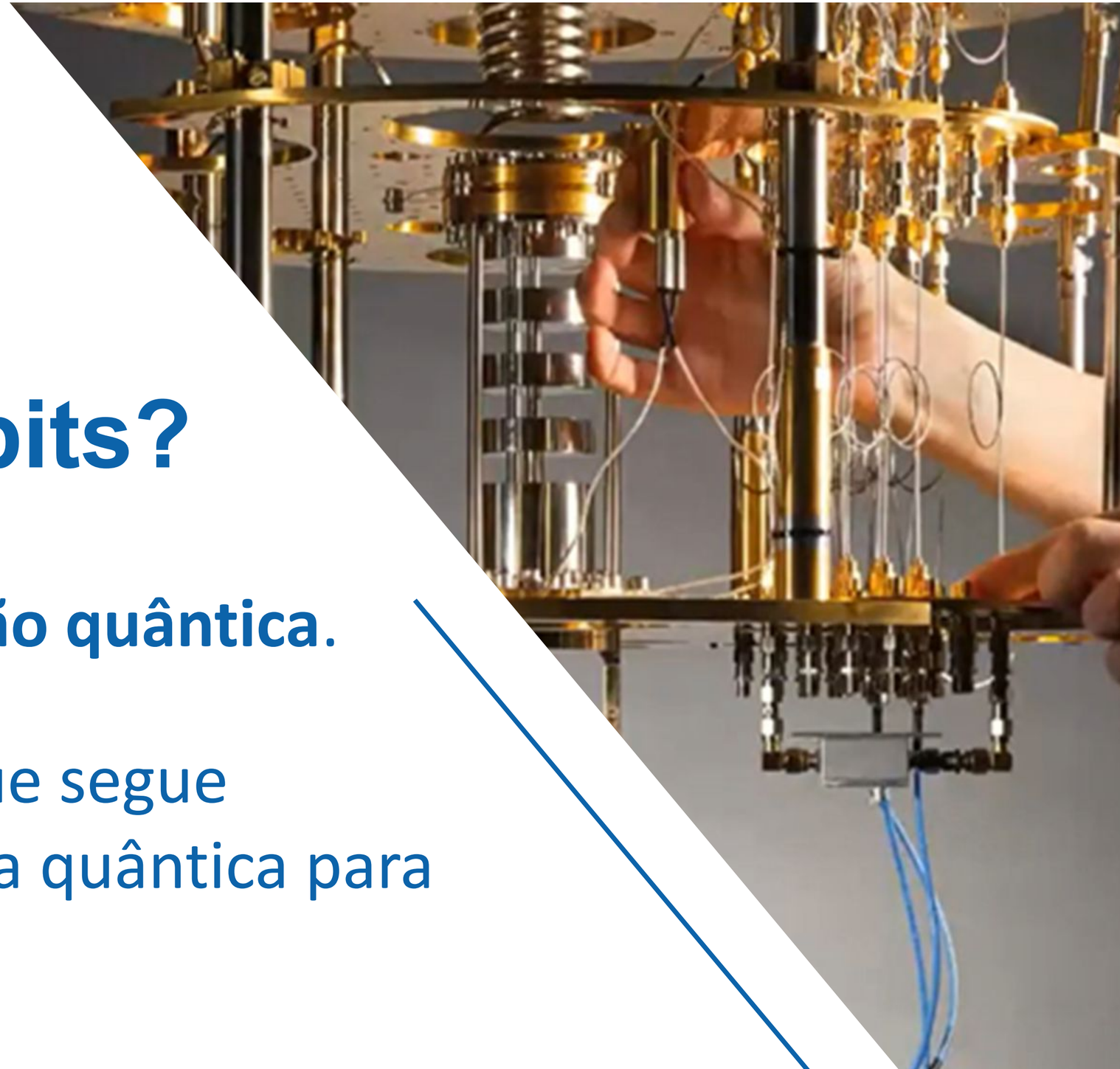
Temos problemas que são impossíveis de serem resolvidos em computadores clássicos.



O que são qubits?

Unidade de **informação quântica**.

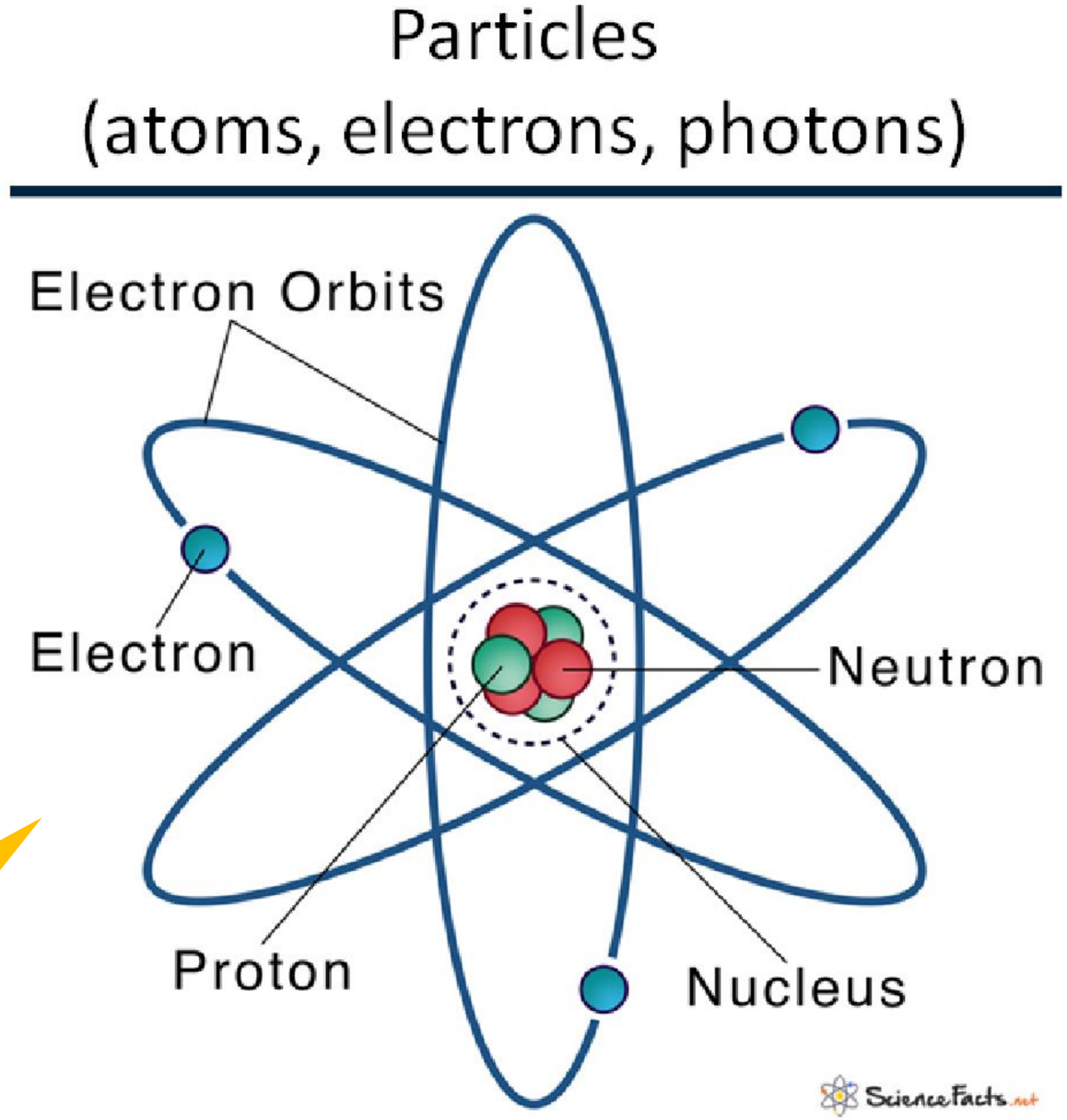
Precisamos de algo que segue princípios de mecânica quântica para realizar um qubit!



O que é um qubit ou quantum bit?

Preciso de alguma coisa que siga princípios de mecânica quântica!

A Natureza é quântica!

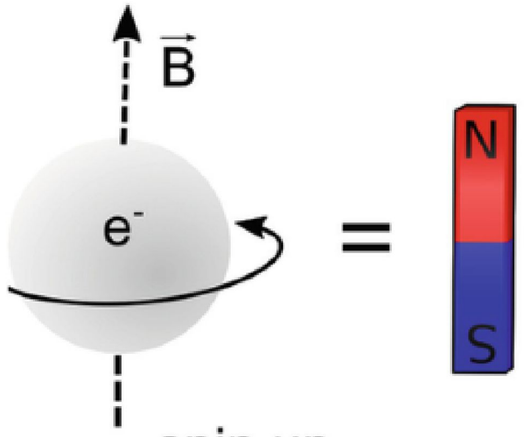


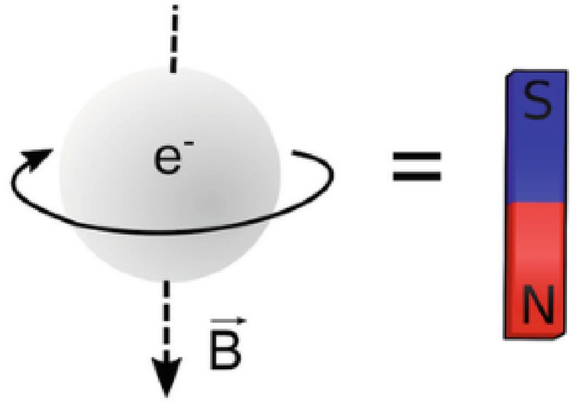
Exemplo: **SPIN de um elétron**

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

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

Representação de Dirac



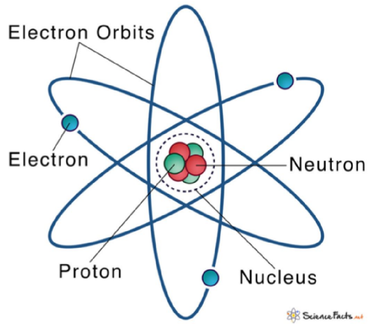


|0>

|1>

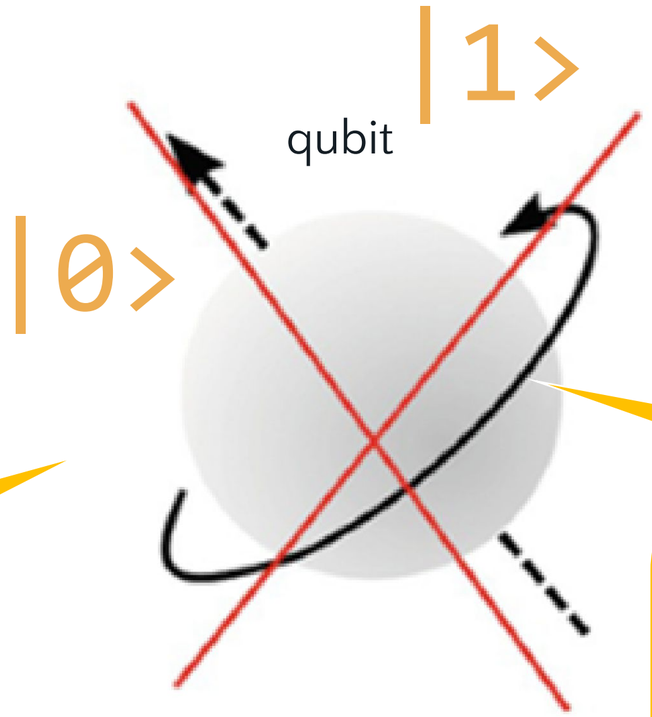
Quantum State |0 >

Quantum State |1 >

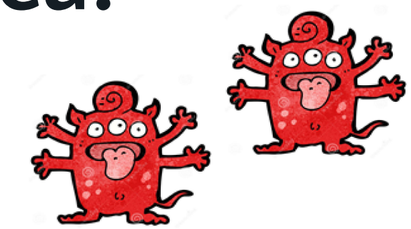


Em mecânica quântica, fenômenos acontecem...

Estado de Superposição



A Natureza é Quântica!

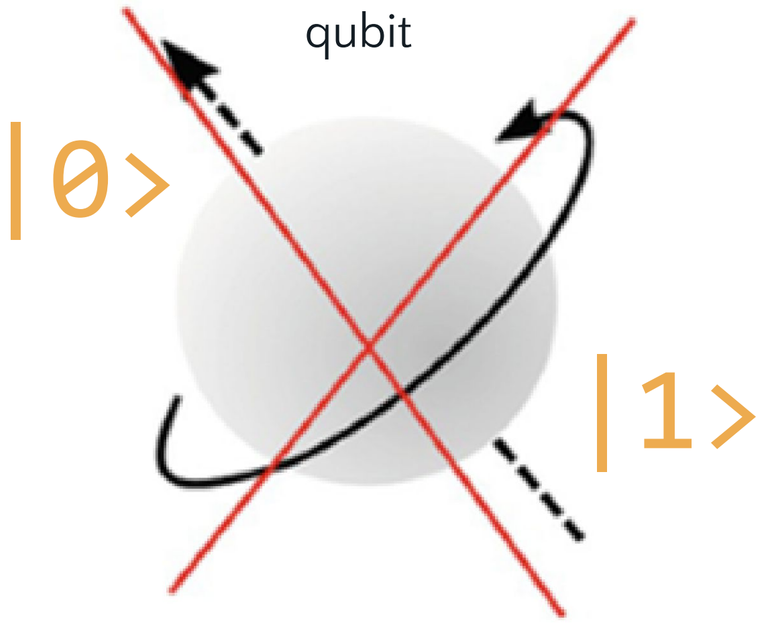


Em superposição, temos uma combinação de todos os estados ao mesmo tempo

Quando em superposição, meu qubit pode lidar com mais informações ao mesmo tempo, criando vantagem exponencial!

Como formalizar o estado de superposição!?

Superposition State



$$|\alpha|^2 + |\beta|^2 = 1$$

α e β **probabilidades** de estar em cada estado!

100% de estar em todos os estados possíveis

$$|\text{Qubit State}\rangle = \alpha|\text{spin up}\rangle + \beta|\text{spin down}\rangle$$

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

Comparando Bit X Qubit

Vamos ver como é a utilização do Qubit para representação da informação

Computação Clássica

1 bit = 0 ou 1
2 bits = 00, 01, 10 ou 11
4 bits = 0000, 0001, 0010, 0011, 0100, 0101, 0110, 0111, 1000, 1100, 1110, 1010, 1001, 1011, 1101 ou 1111

Computação Quântica

1 qubit = 0 e 1
2 qubits = 00, 01, 10 e 11
4 qubits = 0000, 0001, 0010, 0011, 0100, 0101, 0110, 0111, 1000, 1100, 1110, 1010, 1001, 1011, 1101 e 1111

Comparando Bit X Qubit

Desta forma podemos ver que a computação quântica apresenta um crescimento exponencial da representação simultânea da informação

Computação Quântica

2 qubits = 2^2
4 qubits = 2^4
8 qubits = 2^8

E assim sucessivamente

Comparando Bit X Qubit

Então, exponencialmente, temos:

Computação Quântica

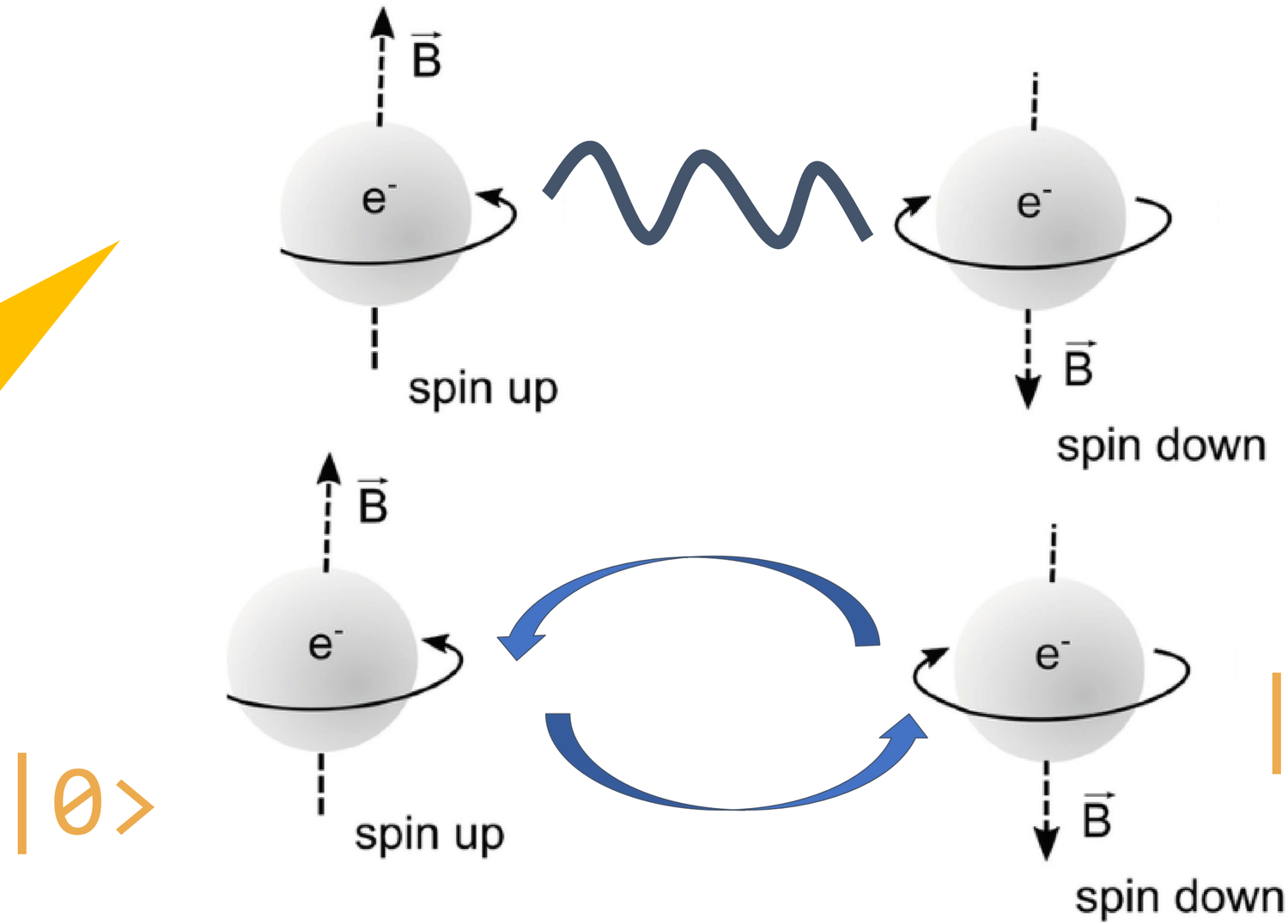
10 qubits = 2^{10} ~1kB
20 qubits = 2^{20} ~1MB
30 qubits = 2^{30} ~1GB
50 qubits = 2^{50} ~ 1 PetaByte
70 qubits = 2^{70} ~1 ZetaByte

1 ZetaByte representa aproximadamente a quantidade de informação digital existente na terra hoje!!

Em mecânica quântica, fenômenos acontecem...

Estado de Emaranhamento

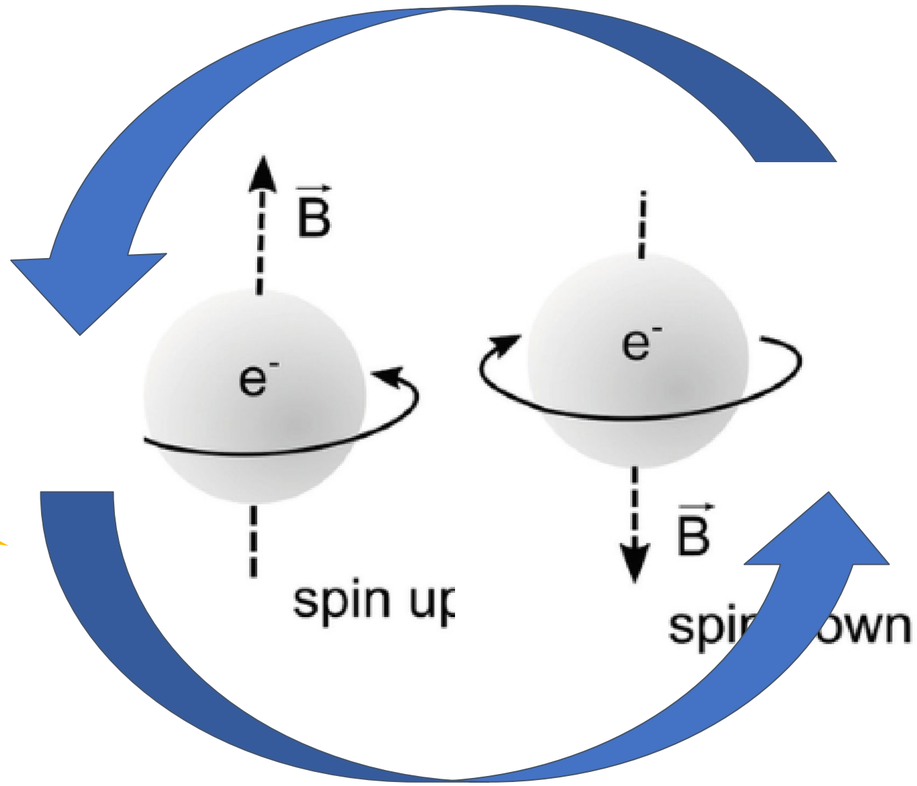
No emaranhamento, ou entrelaçamento, dois qubits interagem com polarização opostas



Em mecânica quântica, fenômenos acontecem...

Interferência

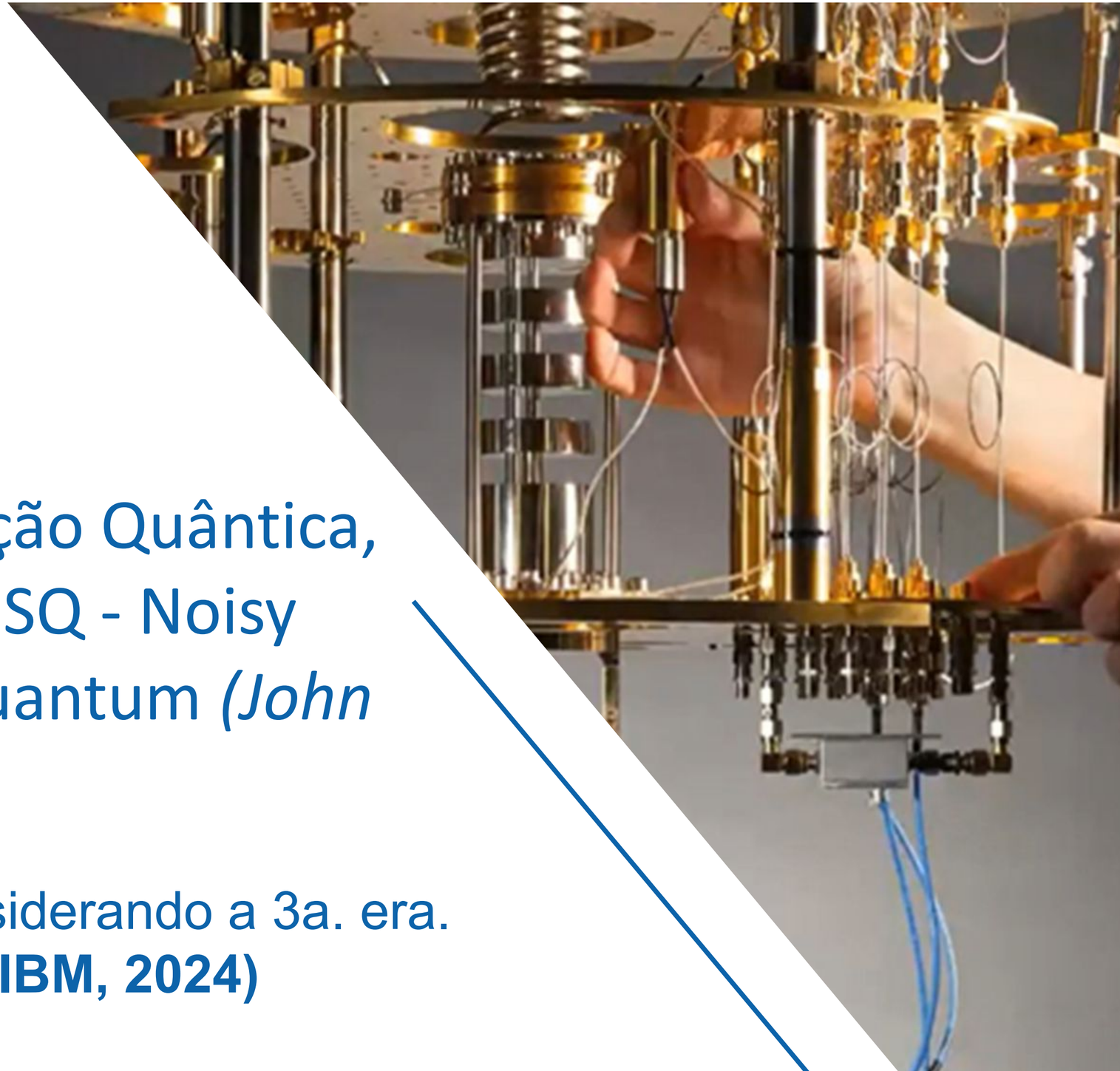
Qubits são sensíveis a Interferência, e colapsam quando ocorre algum distúrbio.



Impossibilidade de cópia

Decoerência Quântica => decaimento quântico, geração de ERRO

A Revolução Quântica



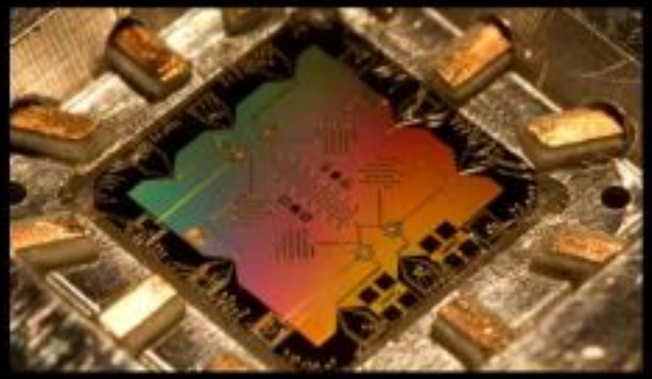
Estamos na 2º Revolução Quântica, com computadores NISQ - Noisy Intermediate-Scale Quantum *(John Preskill, 2017)*

A partir de 2024 estão considerando a 3a. era.
The Quantum Utility Era (IBM, 2024)

The current options

There is no obvious winner, the market yet might split multiple ways

Superconducting



Google

rigetti IBM D-Wave

✓ Quantum supremacy
✗ Cross-talk

Trapped Ions

Honeywell

IONQ

✓ High quantum volume
✗ Slow

Photonics

ORCA Computing

XANADU

PsiQuantum

✓ Fast
✗ Many components

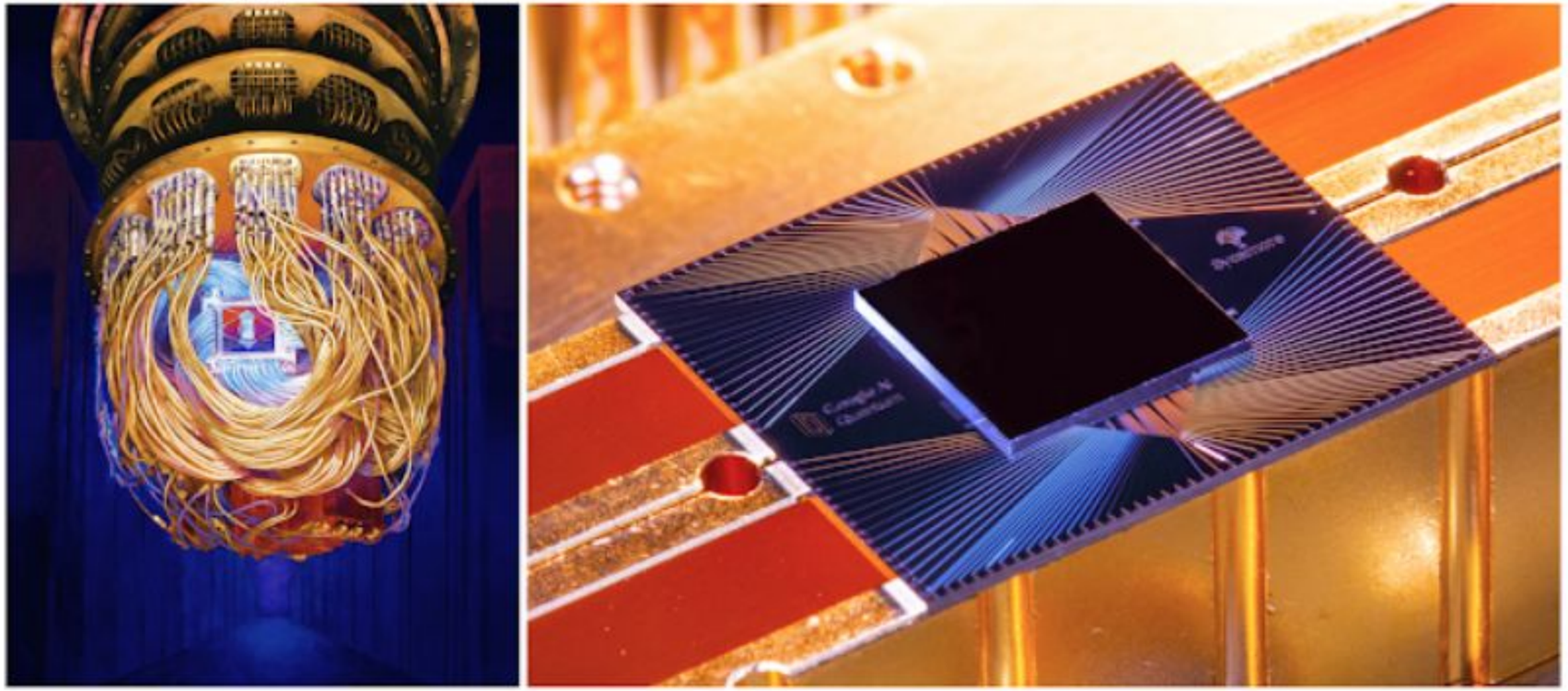
ORCA Computing, March 2022, IEEE Quantum, https://quantum.ieee.org/

Quantum Supremacy | Outubro de 2019

https://ai.googleblog.com/2019/10/quantum-supremacy-using-programmable.html

"We developed a new 54-qubit processor, named "Sycamore", that is comprised of fast, high-fidelity quantum logic gates, in order to perform the benchmark testing. Our machine performed the target computation in ***200 seconds,*** *and from measurements in our experiment we determined that it would take the world's fastest supercomputer* ***10,000 years*** *to produce a similar output."*

Google AI Blog



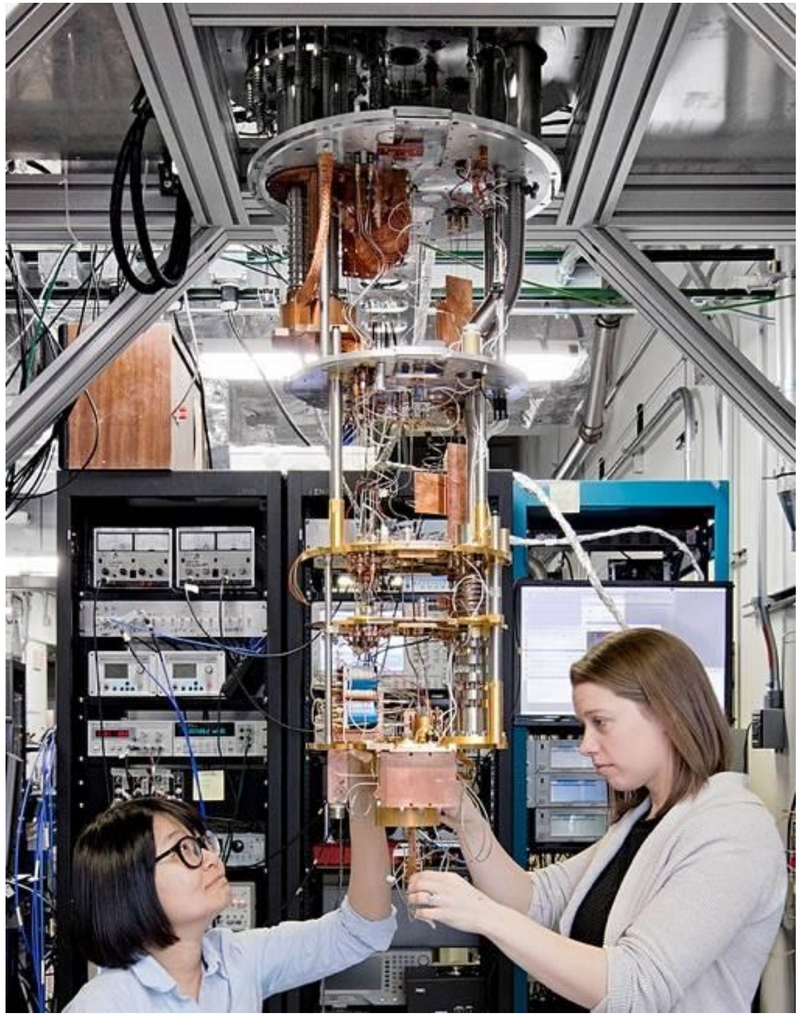
Left: Artist's rendition of the Sycamore processor mounted in the cryostat. (Full Res Version; Forest Stearns, Google AI Quantum Artist in Residence) **Right:** Photograph of the Sycamore processor. (Full Res Version; Erik Lucero, Research Scientist and Lead Production Quantum Hardware)

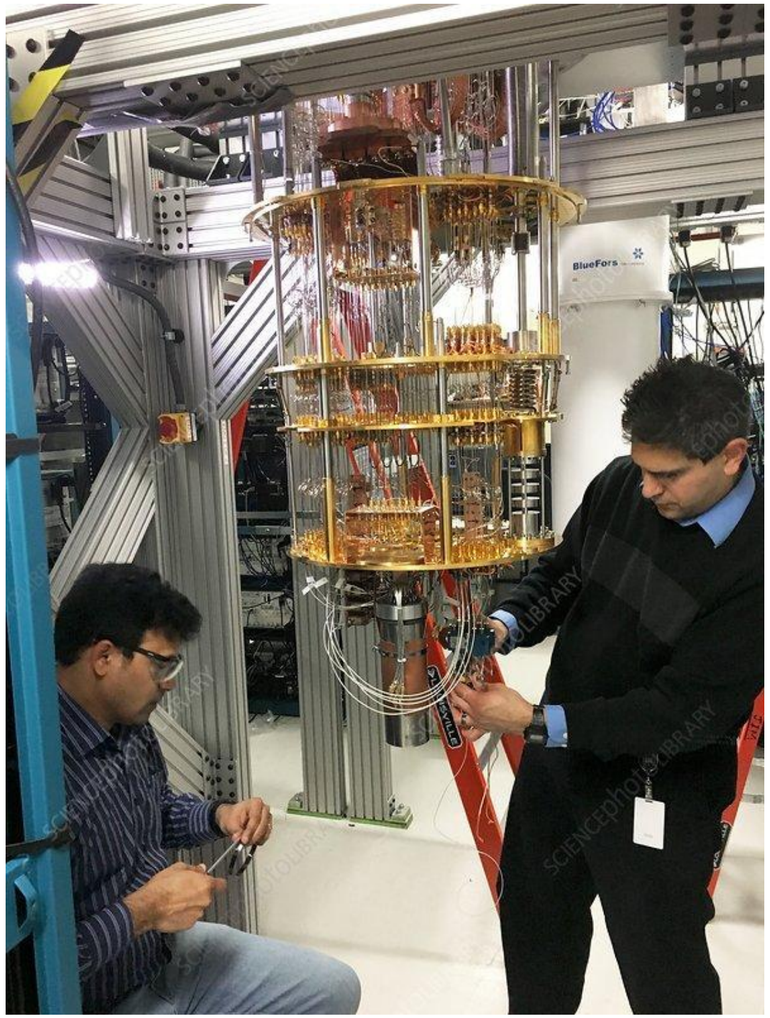
https://www.nature.com/articles/s41586-019-1666-5

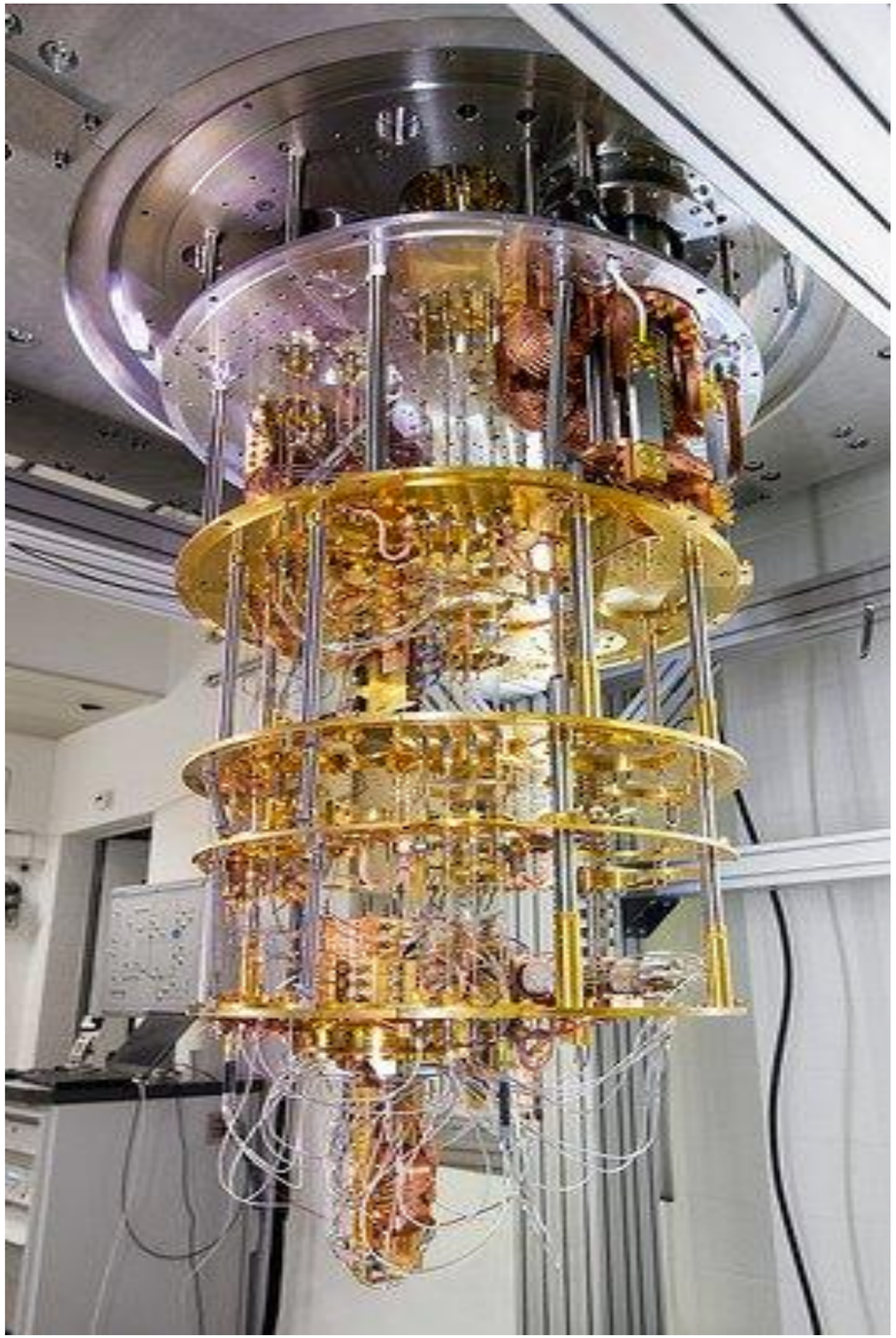
Vantagem Quântica !!

IBM-QE | Superconducting

https://www.ibm.com/quantum-computing/

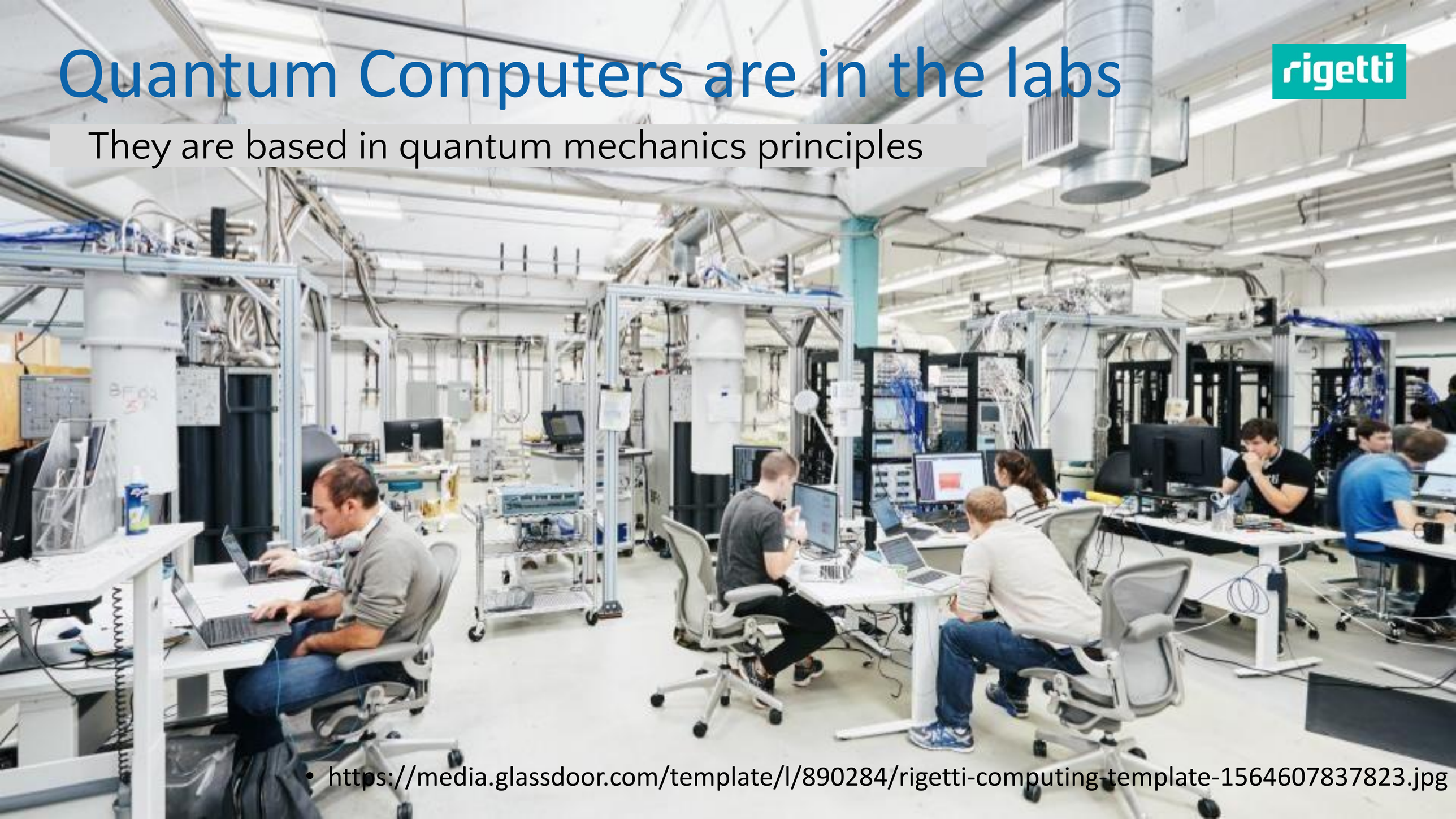






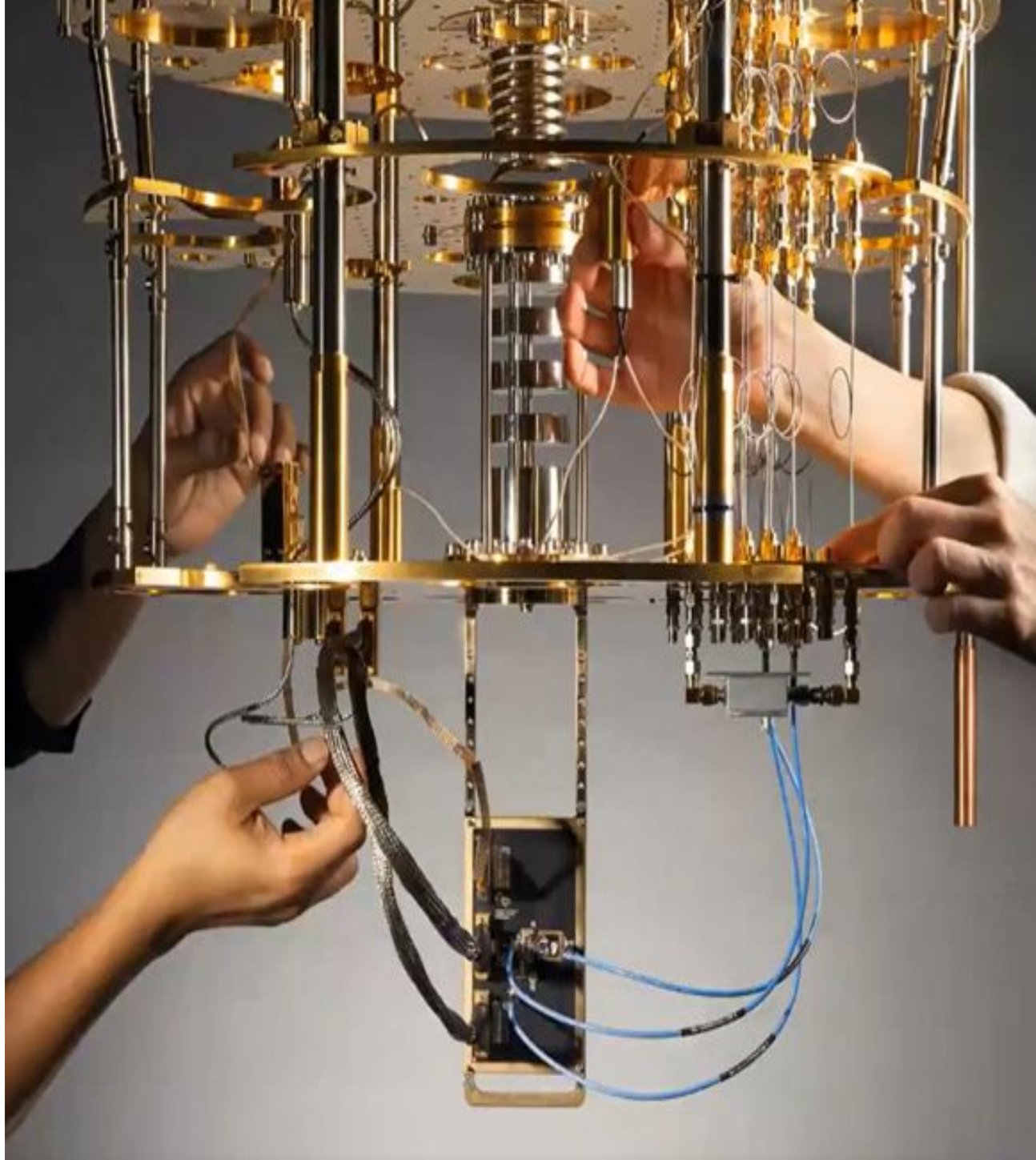
Quantum Computers are in the labs

They are based in quantum mechanics principles



- https://media.glassdoor.com/template/l/890284/rigetti-computing-template-1564607837823.jpg

O que podemos fazer com CQ atualmente?



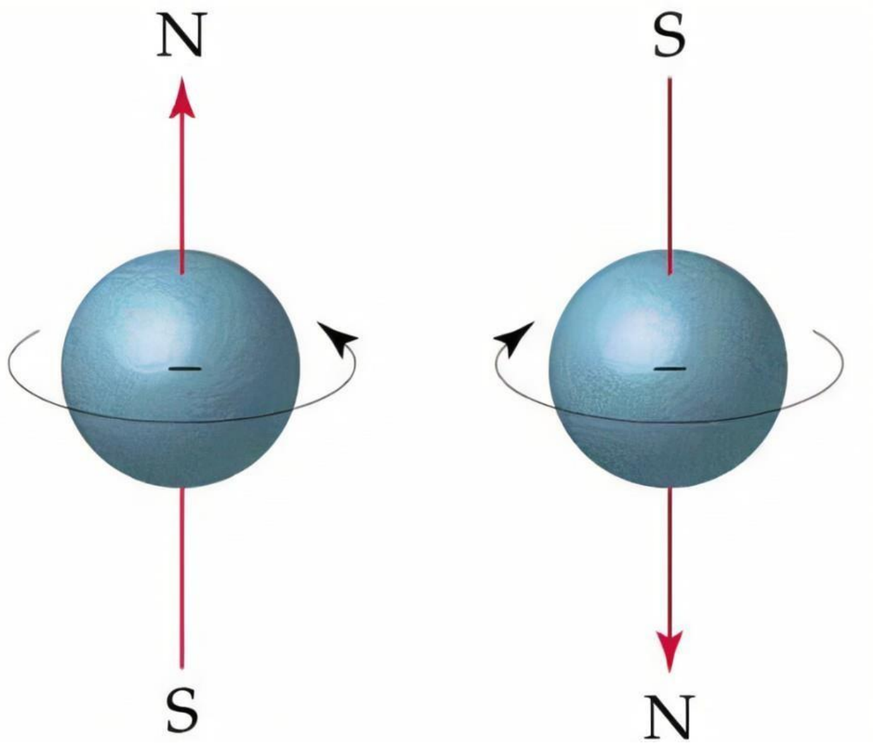
Potential applications of quantum computing

https://www.gartner.com/smarterwithgartner/the-cios-guide-to-quantum-computing/

- **Machine learning:** Improved ML through faster structured prediction. Examples include Boltzmann machines, quantum Boltzmann machines, semisupervised learning, unsupervised learning and deep learning.
- **Artificial intelligence:** Faster calculations could improve perception, comprehension, and circuit fault diagnosis/binary classifiers.
- **Chemistry:** New fertilizers, catalysts, battery chemistries will all drive improvements in resource utilization
- **Biochemistry:** New drugs, tailored drugs, and maybe even hair restorer.
- **Finance:** Quantum computing could enable faster, more complex Monte Carlo simulations; for example, trading, trajectory optimization, market instability, price optimization and hedging strategies.
- **Healthcare:** DNA gene sequencing, such as radiotherapy treatment optimization/brain tumor detection, could be performed in seconds instead of hours or weeks.
- **Materials:** super strong materials; corrosion proof paints; lubricants; semiconductors
- **Computer science:** Faster multidimensional search functions; for example, query optimization, mathematics and simulations.

Aspectos da Mecânica Quântica

Representando um Qubit | usando spin do elétron



|0> |1>

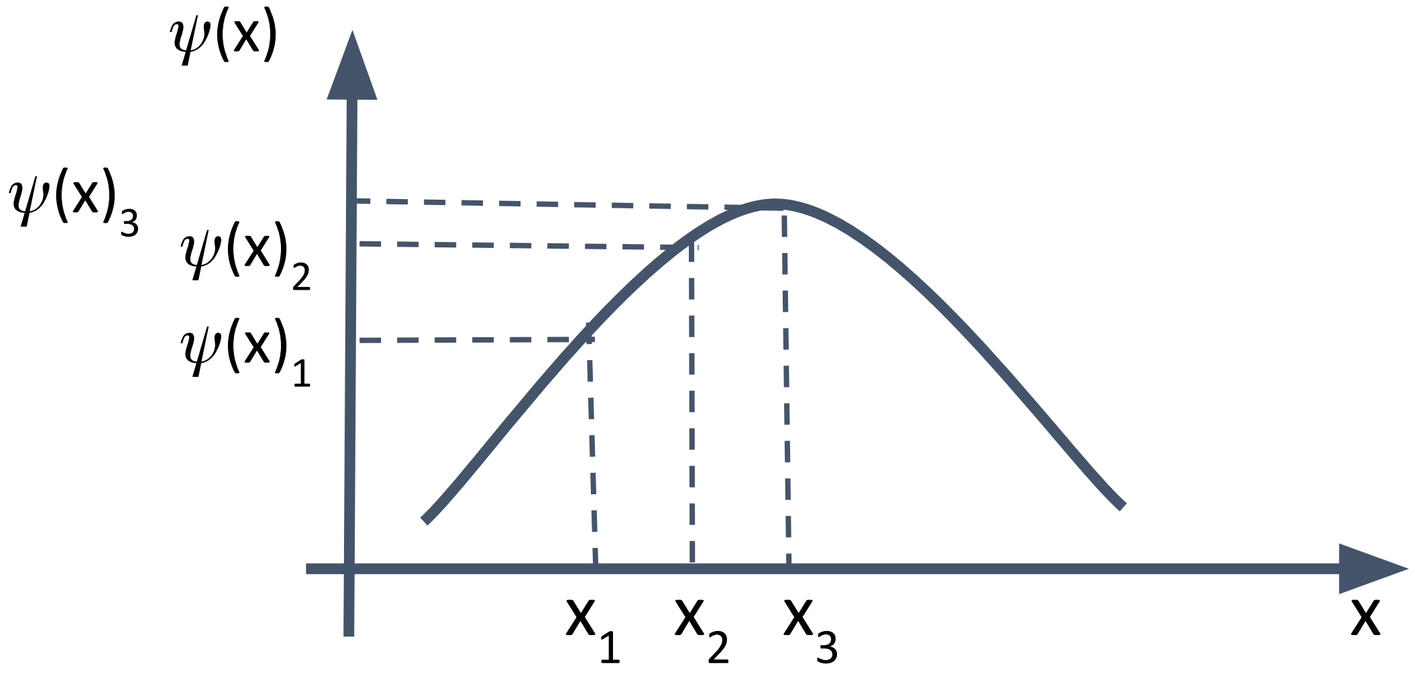
SUPERPOSIÇÃO

$|\Psi\rangle = \alpha|\text{spin up}\rangle + \beta|\text{spin down}\rangle$

by @gbressan

Representação Quântica

Supondo uma função de onda, de uma dimensão, que represente esse elétron usando a mecânica quântica



$$\psi(x) = \begin{pmatrix} \psi(x_1) \\ \psi(x_2) \\ \psi(x_3) \\ . \\ . \\ . \end{pmatrix}$$

Representação do estado de um qubit

Um qubit pode ser representado como um espaço vetorial de Hilbert complexo, de duas dimensões, ou seja, C^2

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

ESPAÇO DE HILBERT: Generalização do espaço Euclidiano, para espaço infinito e que inclua números complexos, além dos números reais

Estado do qubit => representado por um vetor

Operadores => representados por uma matriz

Produto Interno

$$\langle A|B\rangle = (a_1^* \; a_2^* \; a_3^* \ldots . a_n^*) \begin{pmatrix} B_1 \\ B_2 \\ B_3 \\ . \\ . \\ B_n \end{pmatrix}$$

$$= \Sigma \, a_k^* \, b_k$$

de k = 0 , a (n-1)

Se θ_{AB} = 90º => $\langle A|B\rangle = 0$

Se θ_{AB} = 0º => $\langle A|B\rangle = 1$

O produto interno entre os dois vetores A e B, nos permite representar a posição relativa entre eles, e assim obter o estado do qubit no espaço de Hilbert.

Note que: os vetores devem estar no mesmo espaços vetoriais.

Obs.: $\langle A|$ é o transposto conjugado de $|A\rangle$, representado por $(|A\rangle)^\dagger$

Produto Tensorial

$|A\rangle, |B\rangle$

$A \in \mathbb{C}^n$

$B \in \mathbb{C}^m$

Exemplo:

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

O produto tensorial é uma forma de combinar espaços vetoriais. Suponha que tenhamos os vetores $|A\rangle$ e $|B\rangle$, o produto tensorial será a multiplicação dos dois vetores.

Note que: os vetores podem estar em diferentes espaços vetoriais.

Produto Externo entre dois vetores

$$|0\rangle\langle 0| = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \begin{pmatrix} 1 & 0 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}$$

$$|1\rangle\langle 0| = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \end{pmatrix} = \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix}$$

$$|0\rangle\langle 1| = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \begin{pmatrix} 0 & 1 \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}$$

$$|1\rangle\langle 1| = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \begin{pmatrix} 0 & 1 \end{pmatrix} = \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix}$$

O produto externo entre dois vetores é uma operação que resulta em um novo vetor perpendicular aos dois vetores originais, que representa o momento angular entre eles.

Representação Computacional

Esfera de Bloch

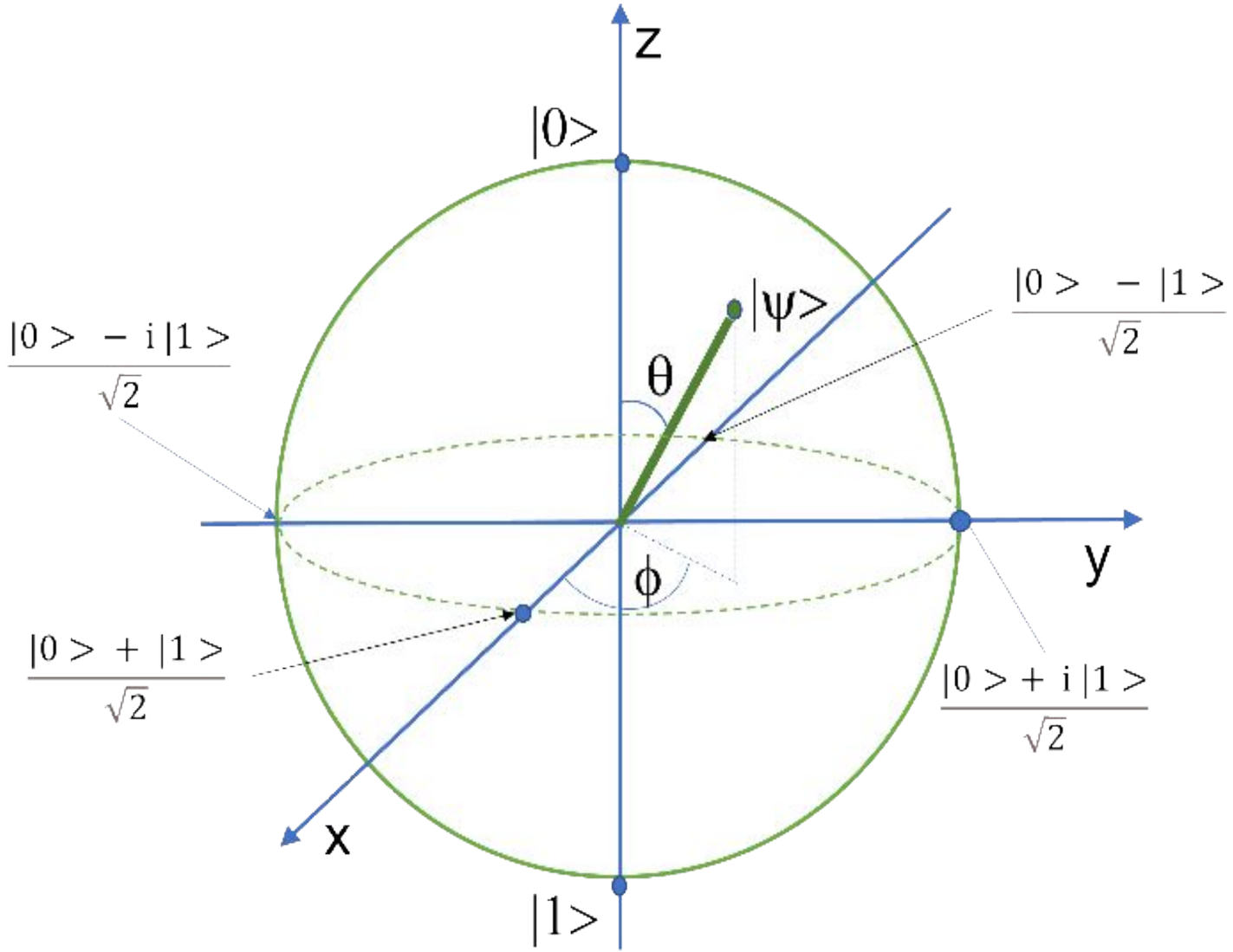
Utilizada para representar o estado de um qubit. A superposição de estados é representada por:

$|\Psi> = \alpha|0> + \beta|1>$

Regra de Born:

$$|\alpha|^2 + |\beta|^2 = 1$$

A probabilidade do qubit estar no estado |0> é $|\alpha|^2$ e de estar no estado |1> é $|\beta|^2$.



https://javafxpert.github.io/grok-bloch/

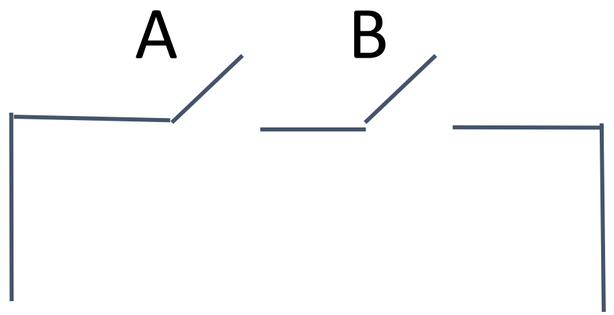
by @gbressan

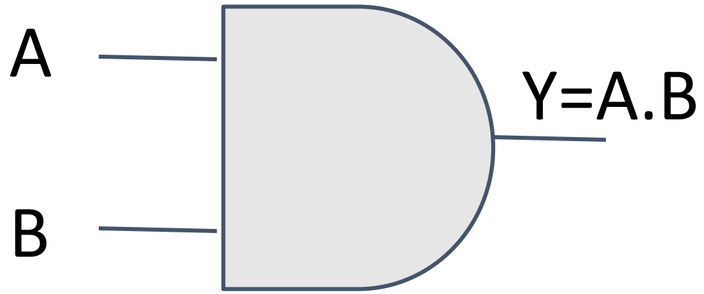
Portas Quânticas - Analogia

As portas quânticas são o equivalente às portas lógicas, que fazem as operações lógicas sobre as unidades básicas de informação clássica, os bits.

Os bits clássicos, representados por 0 e 1, são manipulados para fazer cálculos por portas como: AND, OR, NOT, NOR, entre outras.

Exemplo: Porta AND



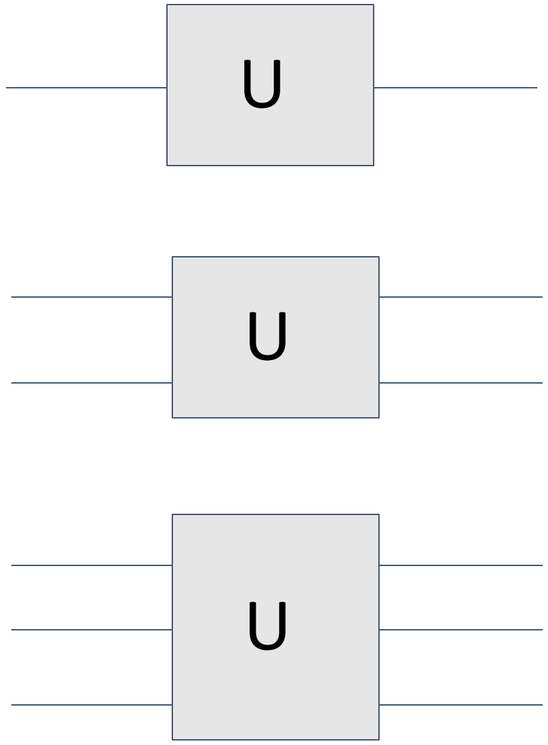


X	Y	f(x, y) = x.y
0	0	0
0	1	0
1	0	0
1	1	1

Portas Quânticas

As portas quânticas são representadas por uma matriz, e é o operador que irá alterar o estado do qubit.

- Portas de 1-qubit – unárias
- Portas de 2-qubits – binárias
- Portas de 3 ou mais qubits – ternárias

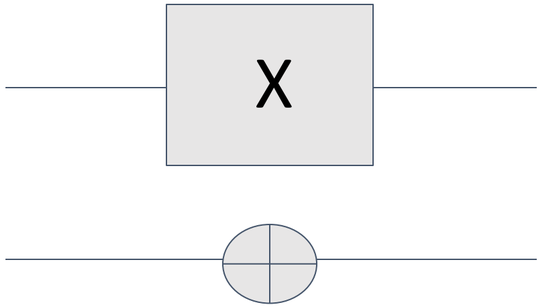


Portas Quânticas: Matriz de Pauli

A porta X, ou operador NOT, também conhecido como *bit flip*.

$$X := \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$

Representado por:



X aplicado a $|0\rangle$, temos:

$$\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0+0 \\ 1+0 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix} = |1\rangle$$

Portas Quânticas: Matriz de Pauli

A porta Y, que rotaciona o estado do vetor no eixo y.

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

Representado por:

Y aplicado a $|1\rangle$, temos:

$$\begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 + (-i) \\ 0 + 0 \end{pmatrix} = \begin{pmatrix} -i \\ 0 \end{pmatrix} = -i|0\rangle$$

Portas Quânticas: Matriz de Pauli

A porta Z, que rotaciona o estado do vetor no eixo z, também conhecido por *phase flip*.

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

Representado por:

Z aplicado a $|0\rangle$, temos:

$$\begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 1+0 \\ 0+0 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix} = |0\rangle$$

Z aplicado a $|1\rangle$, temos:

$$\begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0+0 \\ 0-1 \end{pmatrix} = \begin{pmatrix} 0 \\ -1 \end{pmatrix} = -|1\rangle$$

Matriz de Pauli: Representação matemática

Para o eixo Z:

$$|0\rangle = 1|0\rangle + 0|1\rangle$$
$$|1\rangle = 0|0\rangle + 1|1\rangle$$

Para o eixo X:

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

Para o eixo Y:

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

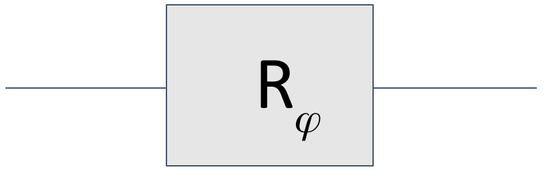
by @wcambiucci

Portas Quânticas: Operador *phase shift*

A porta R_φ, deixa o estado $|0\rangle$ imutável e rotaciona o estado $|1\rangle$ pelo ângulo φ

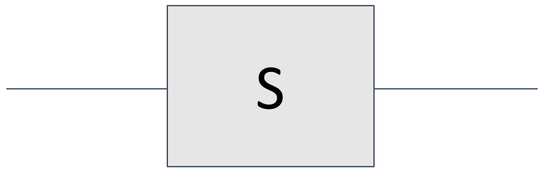
$$R_\varphi := \begin{pmatrix} 1 & 0 \\ 0 & e^{i\varphi} \end{pmatrix}$$

Representado por:



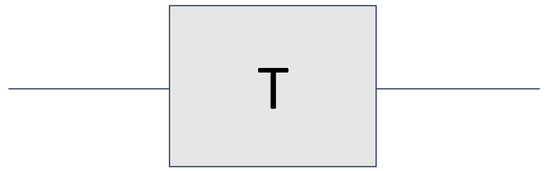
Se $\varphi = \pi/2$, temos:

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



Se $\varphi = \pi/4$, temos:

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

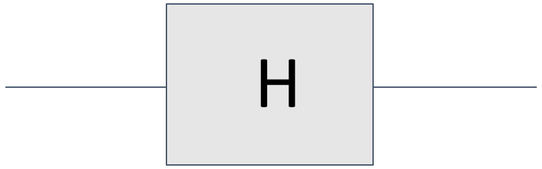


Portas Quânticas: Operador Hadamard

O operador Hadamard é crucial para a computação quântica, porque ele coloca o qubit no estado de **sobreposição** de dois estados

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

Representado por:



H aplicado a $|0\rangle$, temos:

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

H aplicado a $|1\rangle$, temos:

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

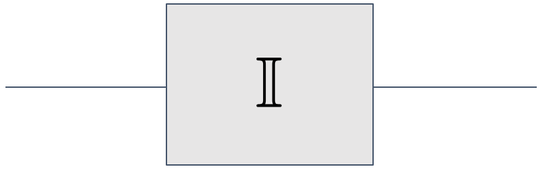
Lembrando que

$|\alpha|^2 + |\beta|^2 = 1$

Portas Quânticas: Matriz Identidade

A matriz Identidade irá manter o estado corrente do qubit

$$\mathbb{I} := \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

Representado por: 

Podemos ver que com estes operadores temos que:

$$HXH = Z$$

$$HZH - X$$

$$HYH = -Y$$

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

$$H^2 = \mathbb{I}$$

Representação de 2 qubits

Dois qubits são indicados como | x y>. Exemplos:

$$|00\rangle = \binom{1}{0} \otimes \binom{1}{0} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}.$$

$$|01\rangle = \binom{1}{0} \otimes \binom{0}{1} = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix}$$

$$|10\rangle = \binom{0}{1} \otimes \binom{1}{0} = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix}$$

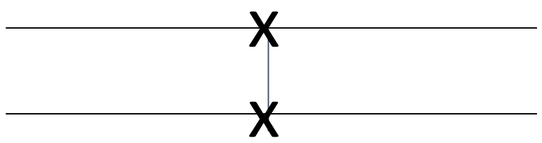
$$|11\rangle = \binom{0}{1} \otimes \binom{0}{1} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}.$$

Portas Quânticas para 2 qubits: SWAP

Esse operador transforma o estado $|01\rangle$ no estado $|10\rangle$, e vice-versa

$$\text{SWAP} := \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Representado por:



Aplicando SWAP no vetor de estado $|01\rangle$, teremos:

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 0+0+0+0 \\ 0+0+0+0 \\ 0+1+0+0 \\ 0+0+0+0 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} = |10\rangle$$

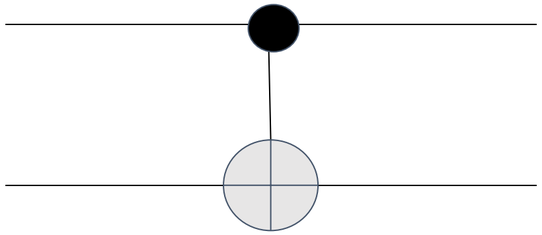
Portas Quânticas para 2 qubits: CNOT

Neste operador, o primeiro qubit é o controle e o segundo o alvo. Se o primeiro é $|0\rangle$, nada é feito, se é $|1\rangle$ é feito o bit flip no segundo.

Este operador permite fazer o emaranhamento entre 2 qubits.

$$\text{CNOT} := \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

Representado por:



Aplicando CNOT em $|10\rangle$, temos:

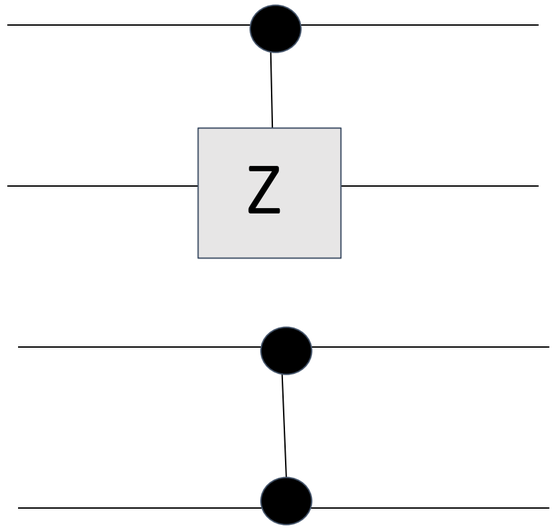
$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0+0+0+0 \\ 0+0+0+0 \\ 0+0+0+0 \\ 0+0+1+0 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} = |11\rangle$$

Portas Quânticas para 2 qubits: CZ

Neste operador, o primeiro qubit é o controle e o segundo o alvo. Se o primeiro é $|0\rangle$, nada é feito, se é $|1\rangle$ é aplicado CZ no segundo.

$$CZ := \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix}$$

Representado por:



Aplicando CZ em $|11\rangle$, temos:

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0+0+0+0 \\ 0+0+0+0 \\ 0+0+0+0 \\ 0+0+0-1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ -1 \end{pmatrix} = -|11\rangle$$

Representação de 3 qubits

Três qubits são indicados como | x y z>. Exemplos:

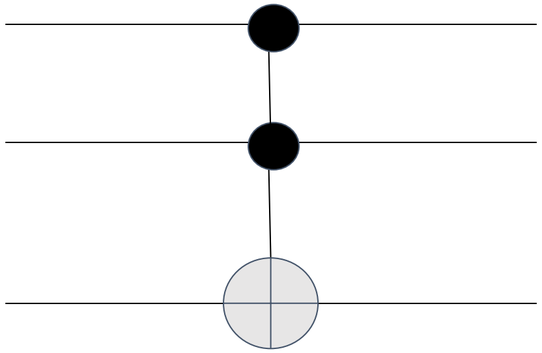
$$|000> = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} \quad |001> = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} \quad |010> = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} \quad |011> = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

$$|100> = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} \quad |101> = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{pmatrix} \quad |110> = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} \quad |111> = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}$$

Portas Quânticas para 3 qubits: TOFFOLI

O operador TOFFOLI é uma porta CNOT com dois bits de controle, chamada de CCNOT.

Representado por:



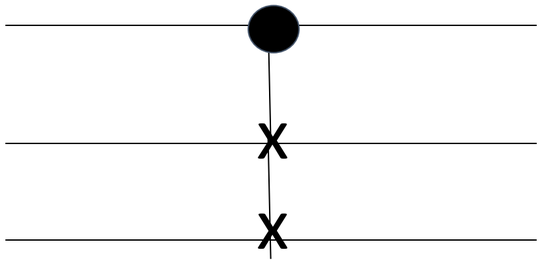
A tabela verdade aplicado a z:

Entrada	Saída
$\|x\rangle\|y\rangle\|z\rangle$	$\|x\rangle\|y\rangle\|z\rangle$
$\|0\rangle\|0\rangle\|0\rangle$	$\|0\rangle\|0\rangle\|0\rangle$
$\|0\rangle\|0\rangle\|1\rangle$	$\|0\rangle\|0\rangle\|1\rangle$
$\|0\rangle\|1\rangle\|0\rangle$	$\|0\rangle\|1\rangle\|0\rangle$
$\|0\rangle\|1\rangle\|1\rangle$	$\|0\rangle\|1\rangle\|1\rangle$
$\|1\rangle\|0\rangle\|0\rangle$	$\|1\rangle\|0\rangle\|0\rangle$
$\|1\rangle\|0\rangle\|1\rangle$	$\|1\rangle\|0\rangle\|1\rangle$
$\|1\rangle\|1\rangle\|0\rangle$	$\|1\rangle\|1\rangle\|1\rangle$
$\|1\rangle\|1\rangle\|1\rangle$	$\|1\rangle\|1\rangle\|0\rangle$

Portas Quânticas para 3 qubits: CSWAP

O operador CSWAP é uma porta onde o primeiro bit é o controle, e os dois outros os alvos.

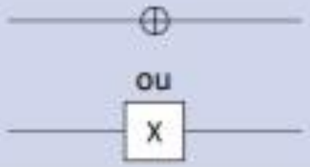
Representado por:



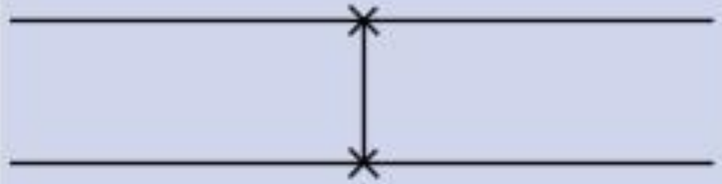
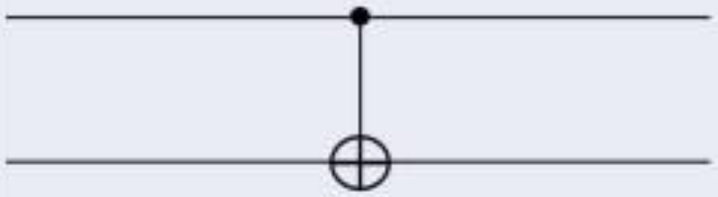
A tabela verdade aplicado a y e z:

Entrada	Saída
$\|x\rangle\|y\rangle\|z\rangle$	$\|x\rangle\|y\rangle\|z\rangle$
$\|0\rangle\|0\rangle\|0\rangle$	$\|0\rangle\|0\rangle\|0\rangle$
$\|0\rangle\|0\rangle\|1\rangle$	$\|0\rangle\|0\rangle\|1\rangle$
$\|0\rangle\|1\rangle\|0\rangle$	$\|0\rangle\|1\rangle\|0\rangle$
$\|0\rangle\|1\rangle\|1\rangle$	$\|0\rangle\|1\rangle\|1\rangle$
$\|1\rangle\|0\rangle\|0\rangle$	$\|1\rangle\|0\rangle\|0\rangle$
$\|1\rangle\|0\rangle\|1\rangle$	$\|1\rangle\|1\rangle\|0\rangle$
$\|1\rangle\|1\rangle\|0\rangle$	$\|1\rangle\|0\rangle\|1\rangle$
$\|1\rangle\|1\rangle\|1\rangle$	$\|1\rangle\|1\rangle\|1\rangle$

Resumindo: Operadores para 1 qubit

Operador	Matriz	O que faz	Diagrama de circuito	Exemplo
X	$\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$	Not $\|0\rangle = \|1\rangle$ e Not $\|1\rangle = \|0\rangle$		$X\|0\rangle = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}\begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \|1\rangle$
Y	$\begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}$	Rotaciona o vetor no eixo y		$Y\|1\rangle = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}\begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} -i \\ 0 \end{pmatrix} = -i\|0\rangle$
Z	$\begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$	Rotaciona o vetor no eixo z		$Z\|1\rangle = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}\begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ -1 \end{pmatrix} = -\|1\rangle$
H	$\frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$	Operador Hadamard que permite criar a sobreposição de estados		$H\|0\rangle = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}\begin{pmatrix} 1 \\ 0 \end{pmatrix} = \frac{\|0\rangle + \|1\rangle}{\sqrt{2}}$ $H\|1\rangle = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}\begin{pmatrix} 0 \\ 1 \end{pmatrix} = \frac{\|0\rangle - \|1\rangle}{\sqrt{2}}$

Resumindo: Operadores para 2 qubits

Operador	Matriz	O que faz	Diagrama de circuito	Exemplo
SWAP	$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$	Troca os valores de dois qubits		$SWAP\ \|01>\ =\ \|10>$ $SWAP\ \|10>\ =\ \|01>$
CNOT	$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$	Se o primeiro qubit é 1 aplica o operador X (not) no segundo qubit. O primeiro qubit é de controle e o segundo é o alvo (target). É a operação utilizada para gerar o entrelaçamento de dois qubits		$CNOT\ \|00>\ =\ \|00>$ $CNOT\ \|01>\ =\ \|01>$ $CNOT\ \|10>\ =\ \|11>$ $CNOT\ \|11>\ =\ \|10>$

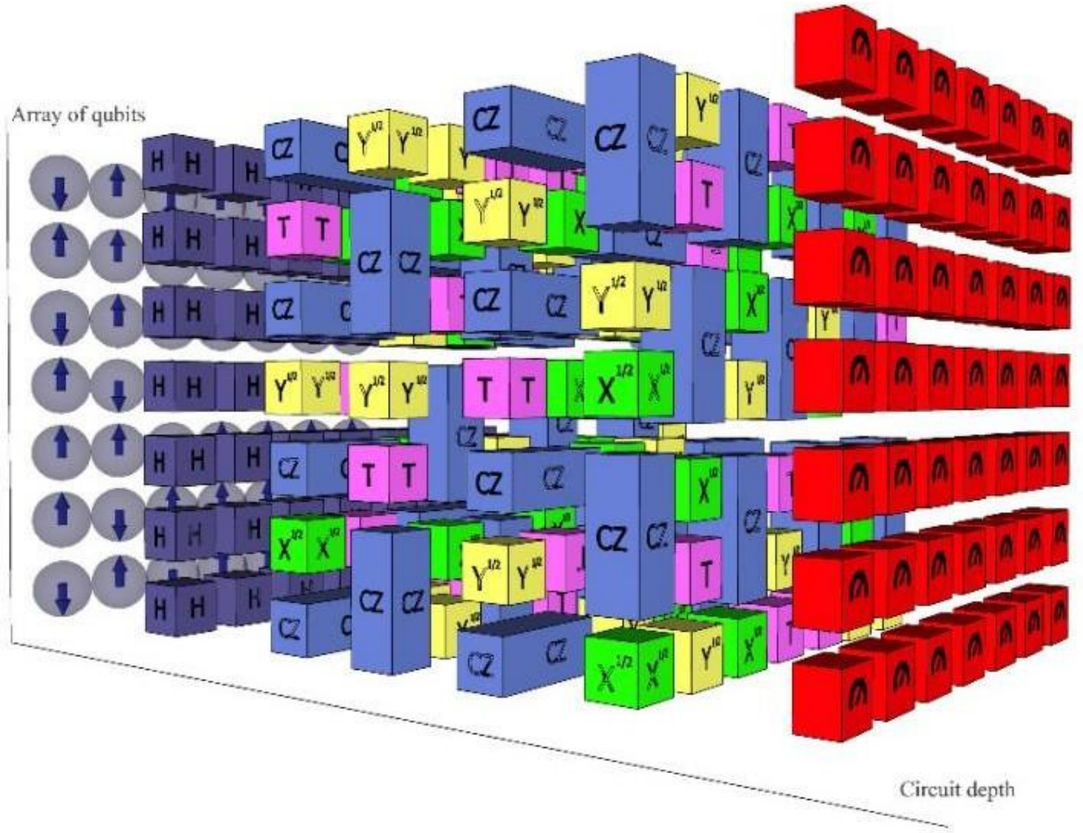
Resumindo: Operadores para 3 qubits

Operador	Matriz	O que faz	Diagrama de circuito	Exemplo
CCNOT	$\begin{pmatrix} 10000000 \\ 01000000 \\ 00100000 \\ 00010000 \\ 00001000 \\ 00000100 \\ 00000001 \\ 00000010 \end{pmatrix}$	Se o primeiro e o segundo qubits forem 1 aplica o operador X (not) no terceiro qubit. O primeiro e o segundo qubits são de controle e o terceiro é o alvo (target).		$CCNOT\ \|001> = \|001>$ $CCNOT\ \|011> = \|011>$ $CCNOT\ \|110> = \|111>$

Circuitos Quânticos

Circuitos quânticos

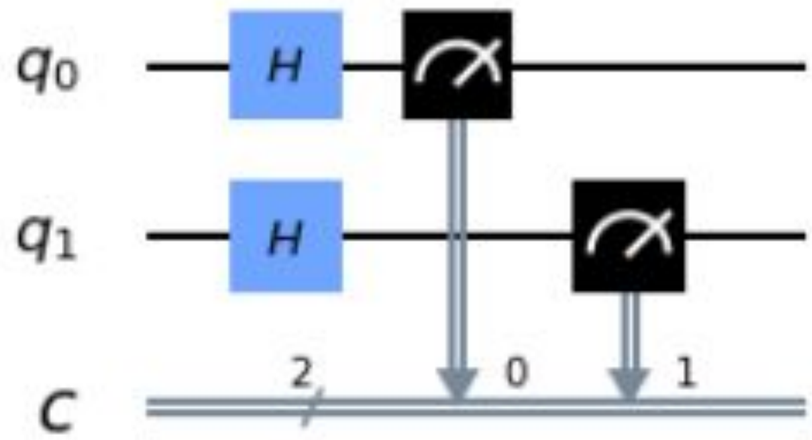
Usam o princípio de Computation-in-place, isto é, que não utilizam nenhuma estrutura auxiliar e as transformações são realizadas sobre a própria entrada.

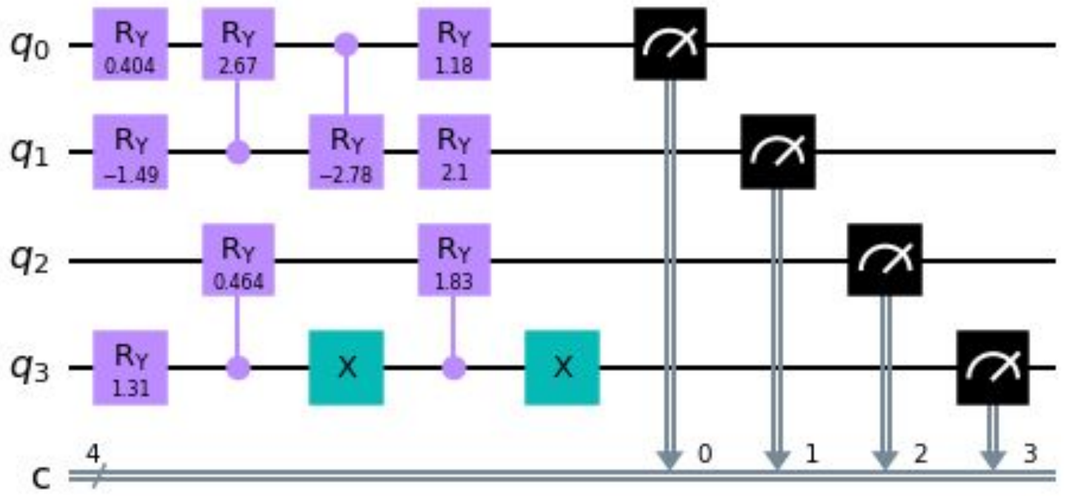


by @gbressan

Circuitos quânticos

Um circuito quântico permite representar o algoritmo, ou seja, os qubits de entrada e a aplicação de operadores a estes qubits, dando um resultado de saída.

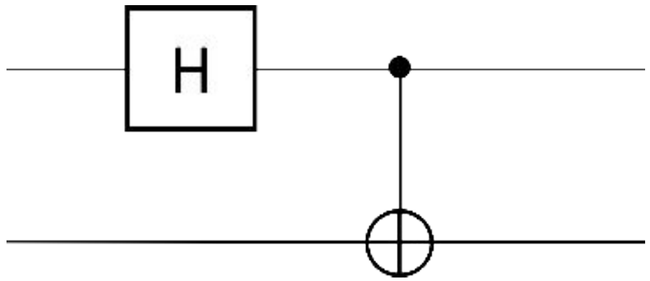




Gerador emaranhado composto por portas R Y , c R Y e X, com valores de parâmetros para {θ 1 , . . . , θ 9 } indicados ao lado das portas.

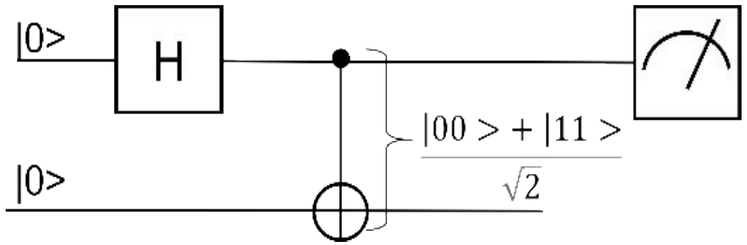
Circuito de Bell e estados de Bell

Cria estados entrelaçados chamados estados de Bell



Se medirmos cada uma das saídas do circuito obteremos, de acordo com a regra de Born, 0 com probabilidade de 50% e 1 com probabilidade 50% considerando o estado entrelaçado $\frac{|00>+|11>}{\sqrt{2}}$

Entrada	Após Hadamard no primeiro qubit	Saída após CNOT (estados de Bell)
$\|0\>\|0\>$	$\frac{\|0>+\|1>}{\sqrt{2}}\|0>=\frac{\|00>+\|10>}{\sqrt{2}}$	$\frac{\|00>+\|11>}{\sqrt{2}}$
$\|0\>\|1\>$	$\frac{\|0>+\|1>}{\sqrt{2}}\|1>=\frac{\|01>+\|11>}{\sqrt{2}}$	$\frac{\|01>+\|10>}{\sqrt{2}}$
$\|1\>\|0\>$	$\frac{\|0>-\|1>}{\sqrt{2}}\|0>=\frac{\|00>-\|10>}{\sqrt{2}}$	$\frac{\|00>-\|11>}{\sqrt{2}}$
$\|1\>\|1\>$	$\frac{\|0>-\|1>}{\sqrt{2}}\|1>=\frac{\|01>-\|11>}{\sqrt{2}}$	$\frac{\|01>-\|10>}{\sqrt{2}}$



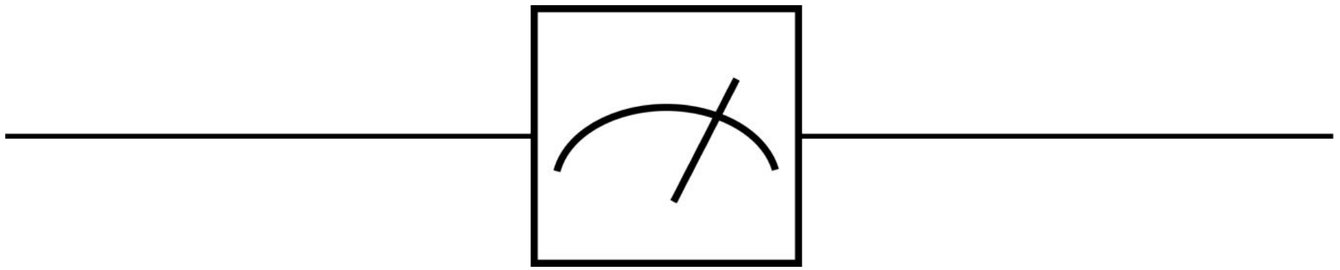
by @gbressan

Operador de medição

Quando queremos observar o estado de um qubit aplicamos o operador de medição e teremos um valor inteiro 0 ou 1 dependente das probabilidades dos estados de acordo com a regra de Born:

- Dado o estado quântico $|\Psi> = \alpha|0> + \beta|1>$ então será medido o valor 0 com probabilidade $|\alpha|^2$ e será medido 1 com probabilidade $|\beta|^2$.

Notar que teremos agora um valor em bit e a operação é irreversível, isto é não podemos voltar ao qubit $|\Psi>$.

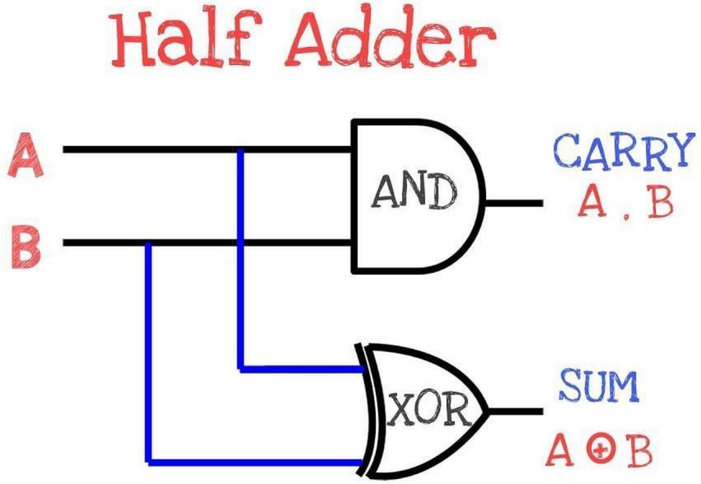


by @gbressan

Exemplo de Algoritmo Quântico

Exemplo 1: Half-Adder (Meio Somador)

- O Half-Adder é um circuito lógico utilizado em eletrônica digital para realizar a adição de números binários.
- Classicamente, o circuito para implementar um half-adder e a respectiva tabela verdade são mostrados abaixo:



Input		Output	
A	B	Sum	Carry
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

- Vamos construir o circuito quântico para implementar um half-adder.

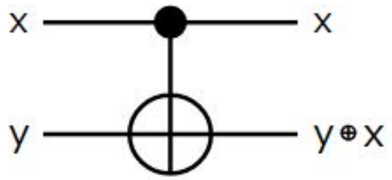
by @jrossetti

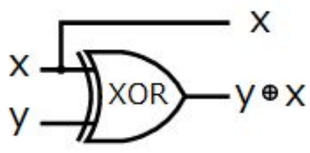
Exemplo 1: Half-Adder

- Para construir esse circuito, vamos precisar de 4 qubits: dois qubits para as entradas A e B e 2 qubits para os resultados Sum e Carry (reversibilidade).

- Precisamos do equivalente quântico de duas portas lógicas clássicas: XOR e AND.

- Como vimos, a porta CNOT comporta-se de forma similar a uma porta XOR:





input		output	
x	y	x	y+x
|0⟩	|0⟩	|0⟩	|0⟩
|0⟩	|1⟩	|0⟩	|1⟩
|1⟩	|0⟩	|1⟩	|1⟩
|1⟩	|1⟩	|1⟩	|0⟩

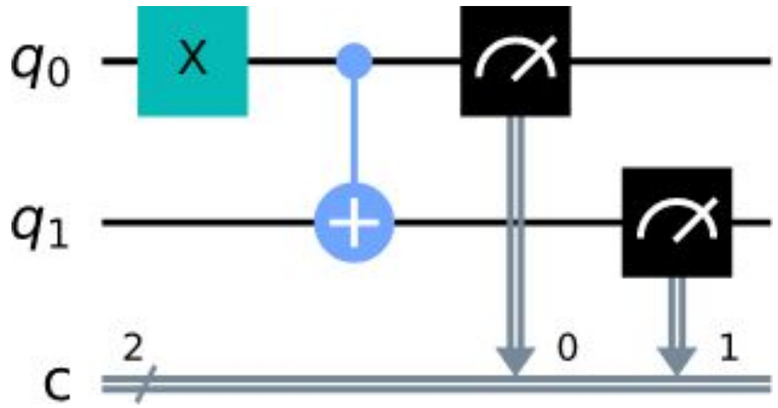
input		output	
x	y	x	y+x
0	0	0	0
0	1	0	1
1	0	1	1
1	1	1	0

by @jrossetti

Exemplo 1: Half-Adder

- Lembrando que assumimos os estados iniciais dos qubits sendo preparados em 0.

- O uso de portas X (bit-flip) nos permitirá codificar a entrada que desejarmos:

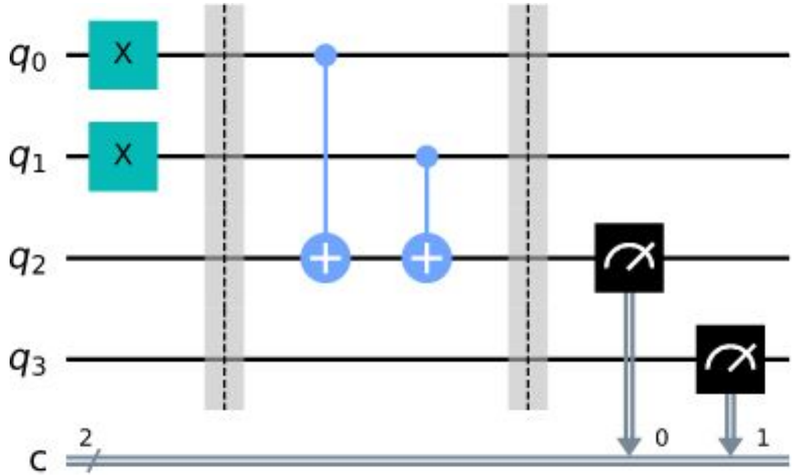


Input (q1 q0)	Output (q1 q0)
00	00
01	11
10	10
11	01

by @jrossetti

Exemplo 1: Half-Adder

- Construíndo o circuito mostrado anteriormente, nossa entrada registrada no qubit q_1 será perdida. Para evitar isso, utilizamos um terceiro qubit para guardar o resultado da operação CNOT, mantendo as duas entradas preservadas.



- Com isso, nossa operação Sum está implementada. Falta agora a operação Carry.

by @jrossetti

Exemplo 1: Half-Adder

- Olhando a tabela verdade do half-adder, verificamos que o bit de carry será 1 apenas quando as duas entradas forem 1.

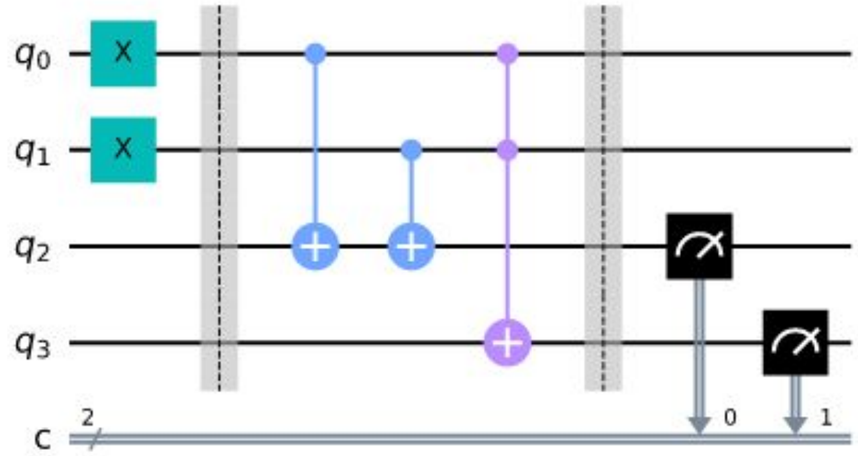
- Precisamos de uma porta que consiga identificar que dois qubits representam o estado $|1\rangle$ e indicar isso em sua saída. Nesse caso, levamos nosso segundo qubit de saída (q_3) para o estado $|1\rangle$.

- Para isso, precisamos que uma porta X seja aplicada ao qubit q_3 quando as entradas q_0 e q_1 forem ambas $|1\rangle$. Vimos anteriormente que a porta Toffoli exibe este comportamento, com suas duas entradas de controle se comportando como uma porta clássica AND, que é o que precisamos.

by @jrossetti

Exemplo 1: Half-Adder

- Aplicando uma porta Toffoli nos respectivos qubits, obtemos o circuito quântico final implementando um half-adder:

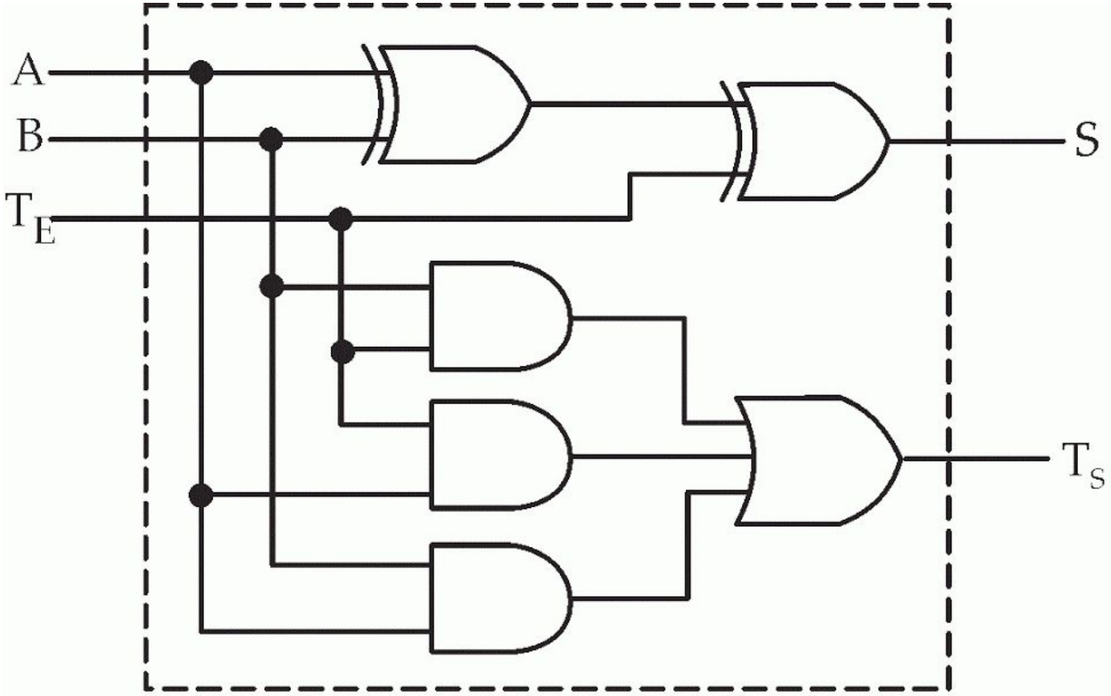


- Na figura acima, queremos somar os bits 1+1. Fazemos a codificação utilizando portas X, aplicamos as portas CNOT e Toffoli, e fazemos a medição, obtendo a saída q_3 = 1 e q_2 = 0.

by @jrossetti

Exemplo 2: Somador Quântico Completo (Full-Adder)

Esse circuito fornece a soma completa, de forma mais complexa do que um Half Adder. Ele é capaz de somar três bits de entrada, sendo dois operadores e um carry-in, e produzir um resultado de soma e um carry-out.

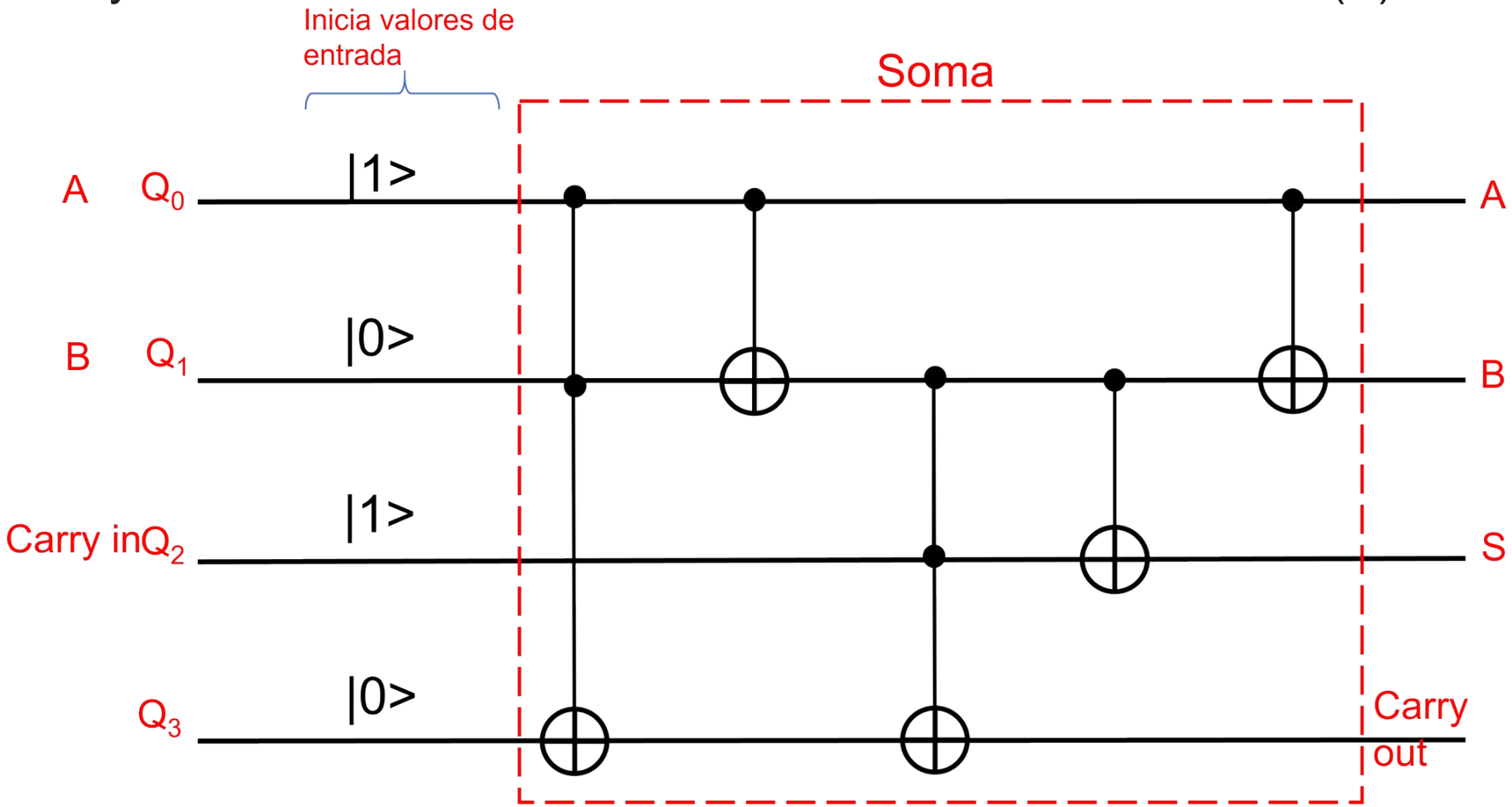


Circuito para o Somador Completo

Diagrama em bloco

Exemplo 2: Somador Quântico Completo (Full-Adder)

O circuito do Full Adder é composto por duas etapas principais: a soma dos bits de entrada e a geração do carry-out. A saída do Full Adder consiste em dois bits: a soma (S) e o carry-out (Cout).



by @gbressan

Algoritmos Quânticos Conhecidos

Algoritmo de Deutsch–Jozsa

Algoritmo de Bernstein–Vazirani

Algoritmo de Simon

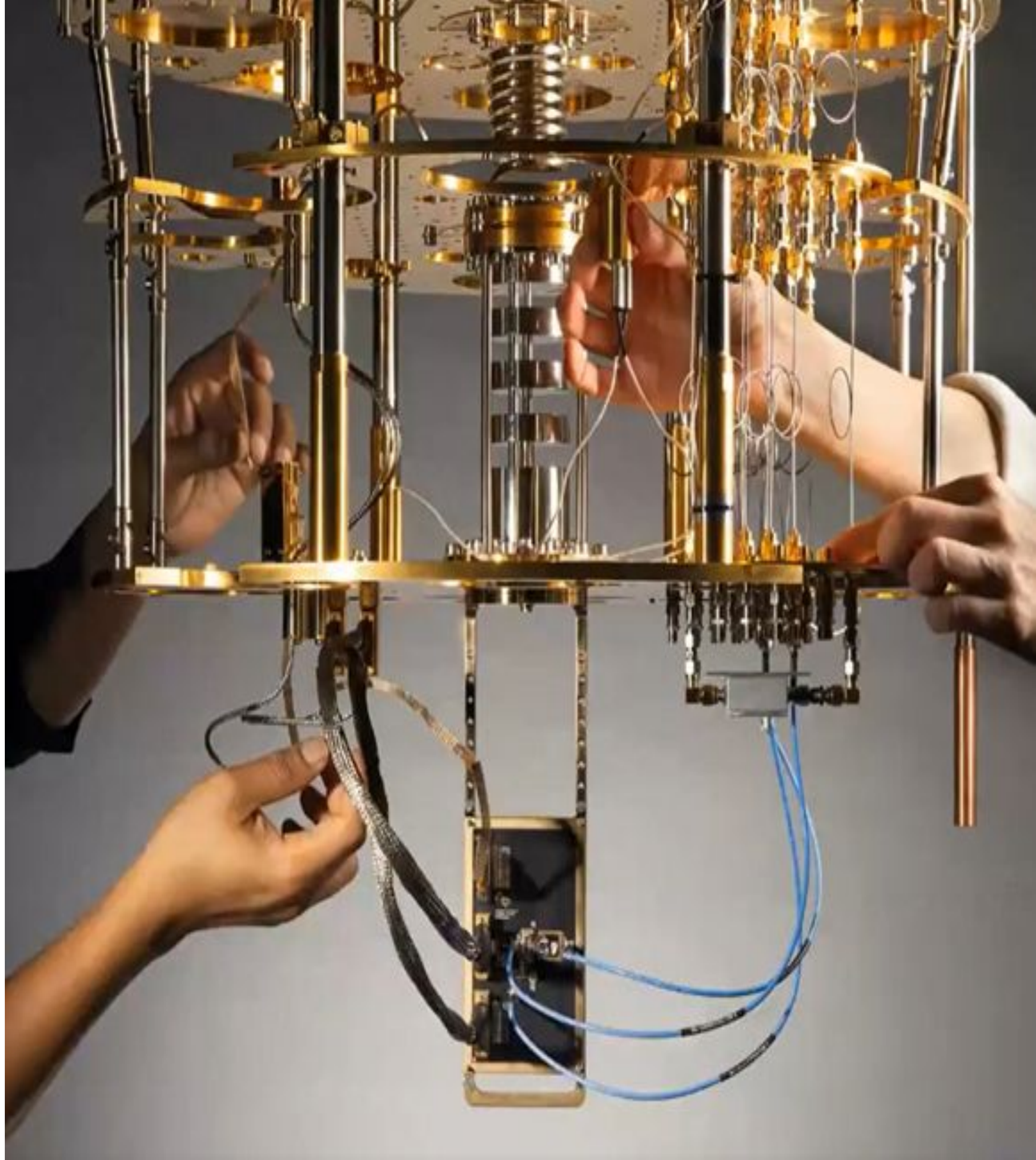
Algoritmo de Shor

Algoritmo de Grover

Transformada de Fourier Quântica

Ambientes de Programação

Simuladores e Ferramentas para Programação Quântica



IBM

Open Source - v 1.0. 2024
Apache-2.0
Bibliotecas no GitHub
Python + OpenQASM

GOOGLE

Open Source - v 1.4.1 - 2024
Apache-2.0
Bibliotecas no GitHub
Python

MICROSOFT

QDK
Quantum Developed Kit

Open Source - v 1.0 - 2024
MIT
Bibliotecas no GitHub
Q#

RIGETTI

Open Source - v 1.0 -. 2017
Apache-2.0
Bibliotecas no GitHub
Python + Quil -> pyQuil

Ambientes de Computação Quântica

Existem vários ambientes e bibliotecas para programação quântica de empresas que
desenvolvem computadores quânticos entre eles:

- Microsoft Azure Quantum: a linguagem utilizada é a Q# baseada na C# e o ambiente está disponível em Visão geral do Microsoft Quantum – Computadores quânticos | Microsoft Azure
- Cirq da Google: programação quântica em Python pode ser estudada e simulada a partir do site https://quantumai.google/cirq
- Ambiente utilizado pelo livro “Programming Quantum Computers” no site https://oreilly-qc.github.io/
- Quirk Quantum Simulator https://algassert.com/quirk
- IBM : https://quantum-computing.ibm.com/ programação em Python no ambiente Qiskit https://qiskit.org/

Criando um Circuito com Python

Programando em Qiskit

- Qiskit é o framework de desenvolvimento para computadores quânticos da IBM.

- A plataforma IBM Q Experience fornece acesso ao Qiskit, um ambiente de programação, e acesso a simuladores e protótipos quânticos.

- Acesso gratuito, basta criar uma conta.

- Utiliza Python como linguagem de programação.

Como funciona o QisKit

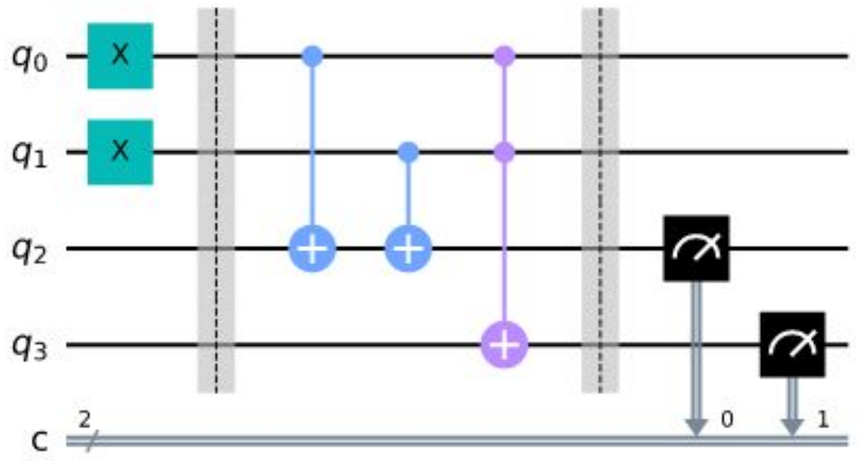
1. Importar pacotes
2. Inicializar variáveis
3. Adicionar portas
4. Visualizar o circuito
5. Simular o experimento
6. Visualizar os resultados

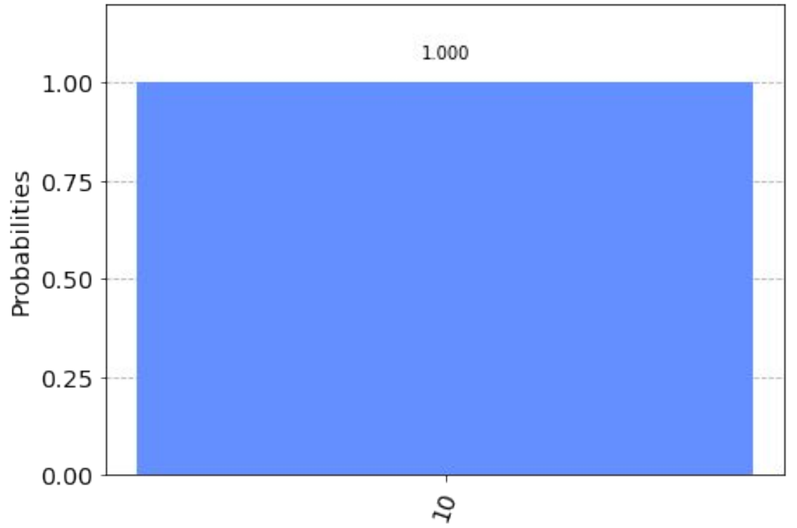
Codificando em Python

Chamada dos Operadores

```
from qiskit import QuantumCircuit, Aer
from qiskit.visualization import
plot_bloch_multivector, plot_histogram
from qiskit.quantum_info import Statevector
from math import pi

# unárias
qc = QuantumCircuit(1)
qc.h(0)
qc.x(0)
qc.y(0)
qc.z(0)
qc.p(pi/4, 0)
qc.s(0)
qc.t(0)
qc.i(0)

# binárias
qc = QuantumCircuit(2)
qc.x(0)
qc.swap(0, 1)
qc.cx(0, 1)
qc.cy(0, 1)
qc.cz(0, 1)

# ternários
qc = QuantumCircuit(3)
qc.ccx(0, 1, 2)
qc.cswap(0, 1, 2)

# desenha o circuito
display(qc.draw(initial_state=True))

# plota na esfera de Bloch
state = Statevector.from_instruction(qc)
display(plot_bloch_multivector(state))

# adiciona medição
qc.measure_all()

# executa no simulador
sim = Aer.get_backend('aer_simulator')
result = sim.run(qc).result()
counts = result.get_counts()
display(plot_histogram(counts))
```

```
from qiskit import QuantumCircuit, Aer, transpile, IBMQ
from qiskit.providers.ibmq import least_busy
from qiskit.visualization import plot_bloch_multivector, plot_histogram
from qiskit.quantum_info import Statevector
from math import pi

# cria o circuito
qc = QuantumCircuit(4, 2)

# codifica as entradas A e B nos qubits 0 e 1
qc.x(0) # Se A=1, usa a porta X
qc.x(1) # Se B=1, usa a porta X

# barreira para facilitar visualização
qc.barrier()

# operação Sum utilizando portas CNOT com saída no qubit 2
qc.cx(0,2)
qc.cx(1,2)

# operação Carry utilizando porta Toffoli com saída no qubit 3
qc.ccx(0,1,3)

# barreira para facilitar visualização
qc.barrier()

# cria os operadores de medição para extrair os resultados
qc.measure(2,0) # Sum
qc.measure(3,1) # Carry

# desenha o circuito
qc.draw(initial_state=True)
```





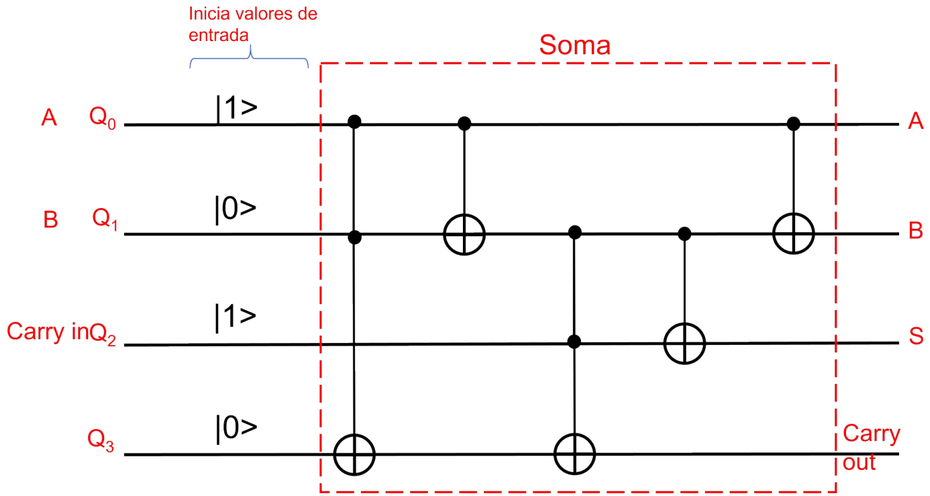
Enviando para o computador Quântico

```
# executa no protótipo quântico real
IBMQ.load_account()
provider = IBMQ.get_provider(hub='ibm-q')
backend = least_busy(provider.backends(filters=lambda x: x.configuration().n_qubits >= 4 and
                                    not x.configuration().simulator and x.status().operational==True))

# Transpila e executa
shots = 1024
transpiled_bv_circuit = transpile(qc, backend)
job = backend.run(transpiled_bv_circuit, shots=shots)

# obtém os resultados
results = job.result()
device_counts = results.get_counts()

# mostra os resultados
plot_histogram(device_counts)
```

Codificando em Python

```
from qiskit import QuantumCircuit, Aer
from qiskit.visualization import  plot_histogram
from qiskit.quantum_info import Statevector
from math import pi

#  qubit definitions
# q[0] --> A
# q[1] --> B
# q[2] --> CarryIn_SumOut
# q[3] --> CarryOut

# initialize inputs to some values
.init
  #initialize inputs A=1, B=0 and carry_in=1
  {x q[0] | x q[2]}

# perform addition
.add

  toffoli q[0],q[1],q[3]
  cnot    q[0],q[1]
  toffoli q[1],q[2],q[3]
  cnot    q[1],q[2]
  cnot    q[0],q[1]
```

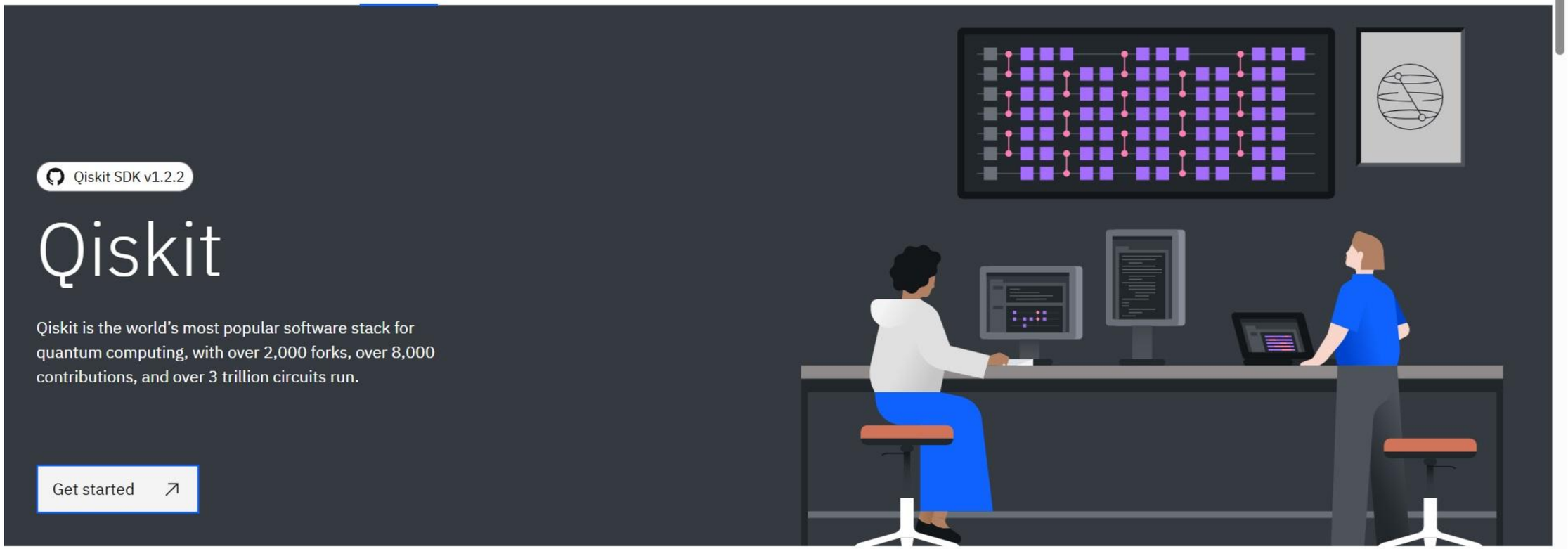
```
# desenha o circuito
display(qc.draw(initial_state=True))

# plota na esfera de Bloch
state = Statevector.from_instruction(qc)
display(plot_bloch_multivector(state))

# adiciona medição
qc.measure_all()

# executa no simulador
sim = Aer.get_backend('aer_simulator')
result = sim.run(qc).result()
counts = result.get_counts()
display(plot_histogram(counts))
```



QisKit da IBM

https://www.ibm.com/quantum/qiskit

IBM | Quantum | Technology | Qiskit | Research | Pricing | Blog | Community | Resources | Sign in to Platform

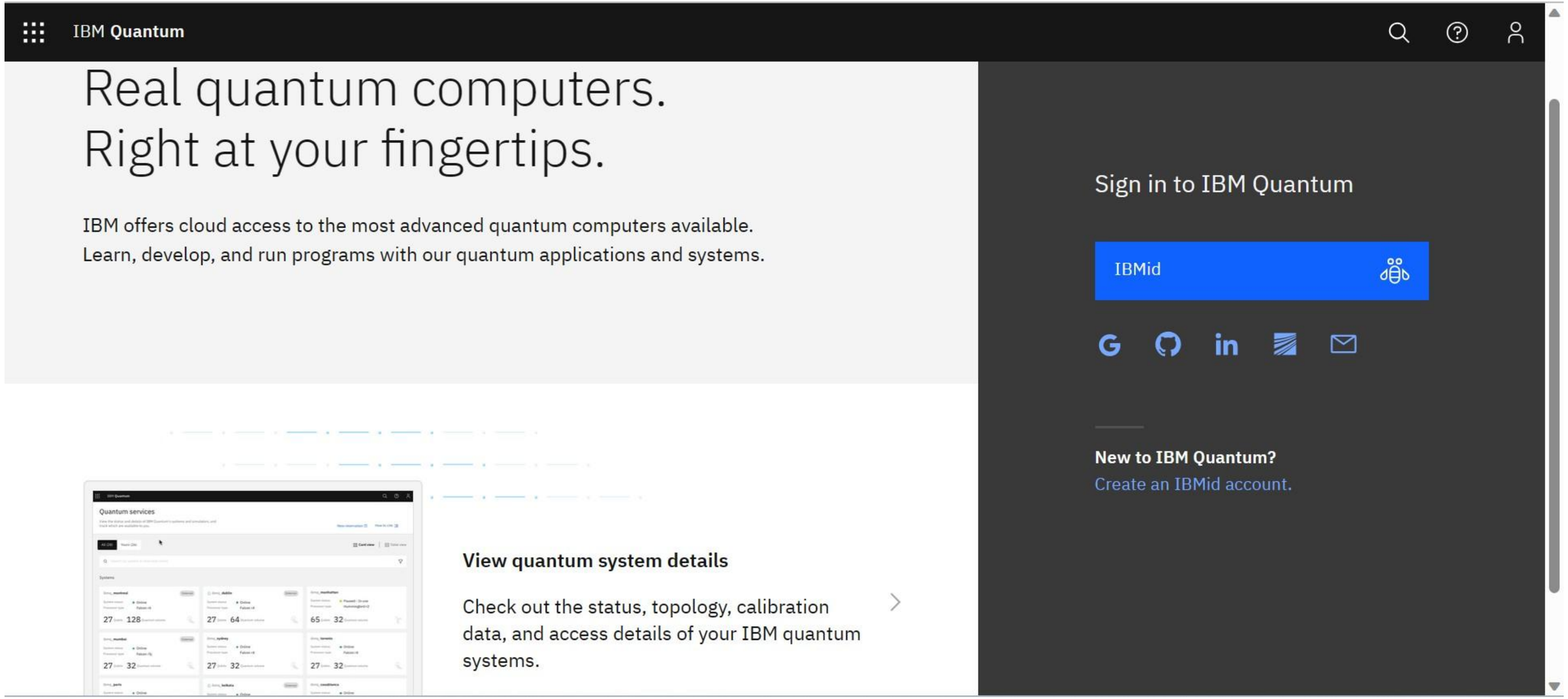


01 Performance | 02 Technologies | 03 Workflow | 04 Systems | 05 Tutorials | 06 Community | 07 Get started | 08 Qiskit blog

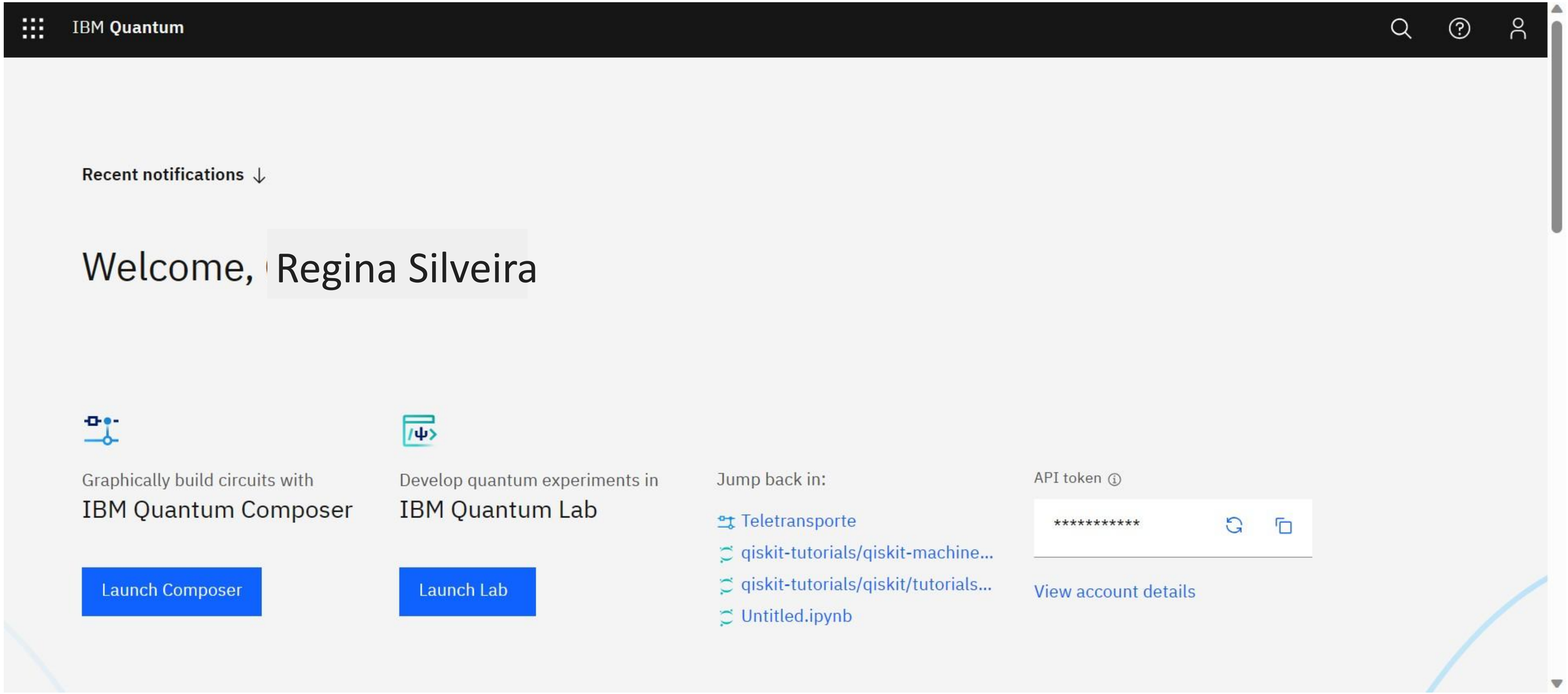
Qiskit is the highest performing quantum SDK

A new open-source suite of tests from leading universities, national labs, and IBM compares the relative speed and quality of quantum frameworks. Based on the results of these tests, Qiskit is the highest performing quantum software development kit for building and transpiling quantum circuits.

1. Criar a conta em IBM Quantum



2. Login na conta IBM



Regina Silveira

IBM Quantum Platform

API Token

Announcing Qiskit functions! Explore the catalog of IBM and third-party services to learn how they can accelerate your development workflow. Browse the catalog →

Open Plan
View details | Upgrade
Up to 10 minutes/month

Monthly usage

Used	Remaining
0ms	10m

Recent workloads

You don't have any workloads yet! Create a program to run on our quantum computers by following the instructions below.

Get started ↗

Instance QPUs →

3

All QPUs →

Documentation Open app ↗

Search docs

Hello World
Create a simple quantum program and run it on a quantum processing unit (QPU)

Qiskit Runtime

Learning Open app ↗

Featured course
Quantum Computing in Practice

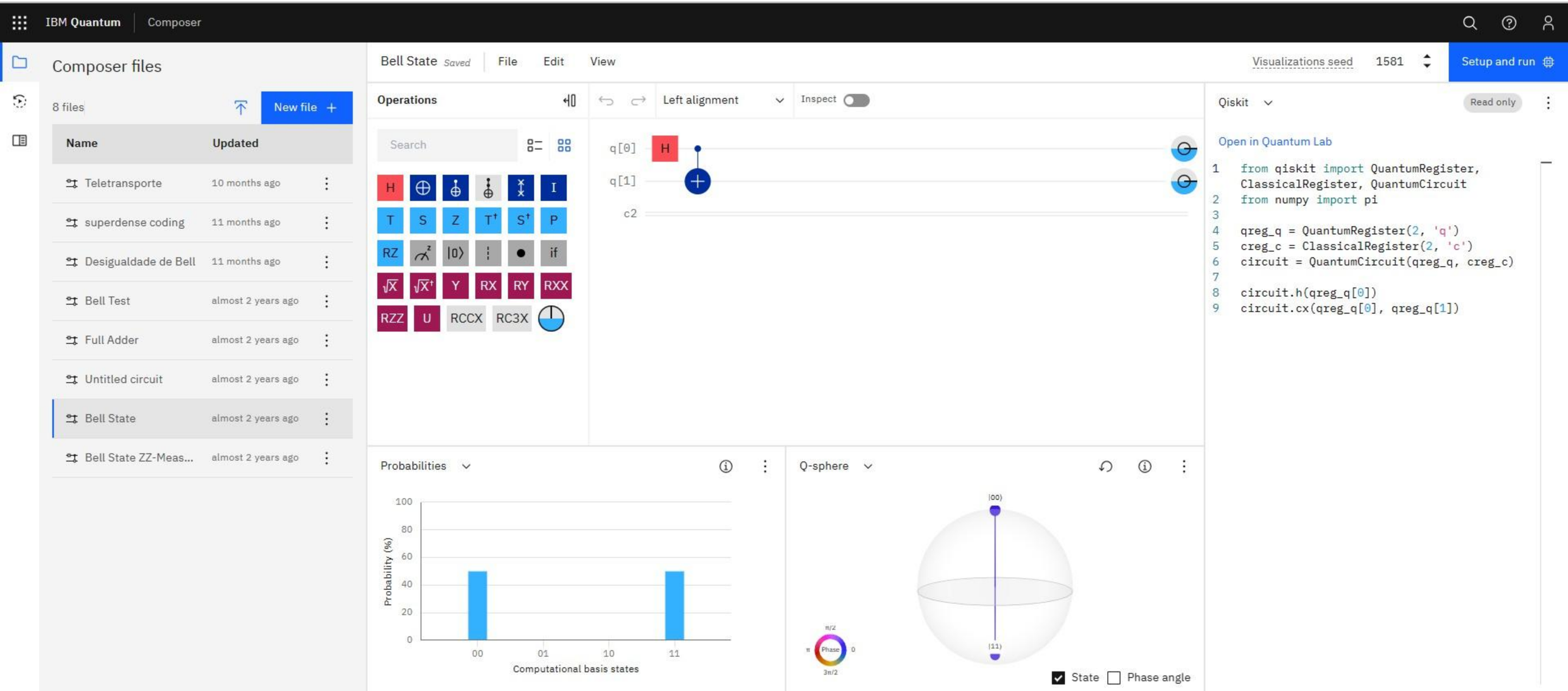
Catalog
Explore all courses and tutorials

IBM Quantum Composer

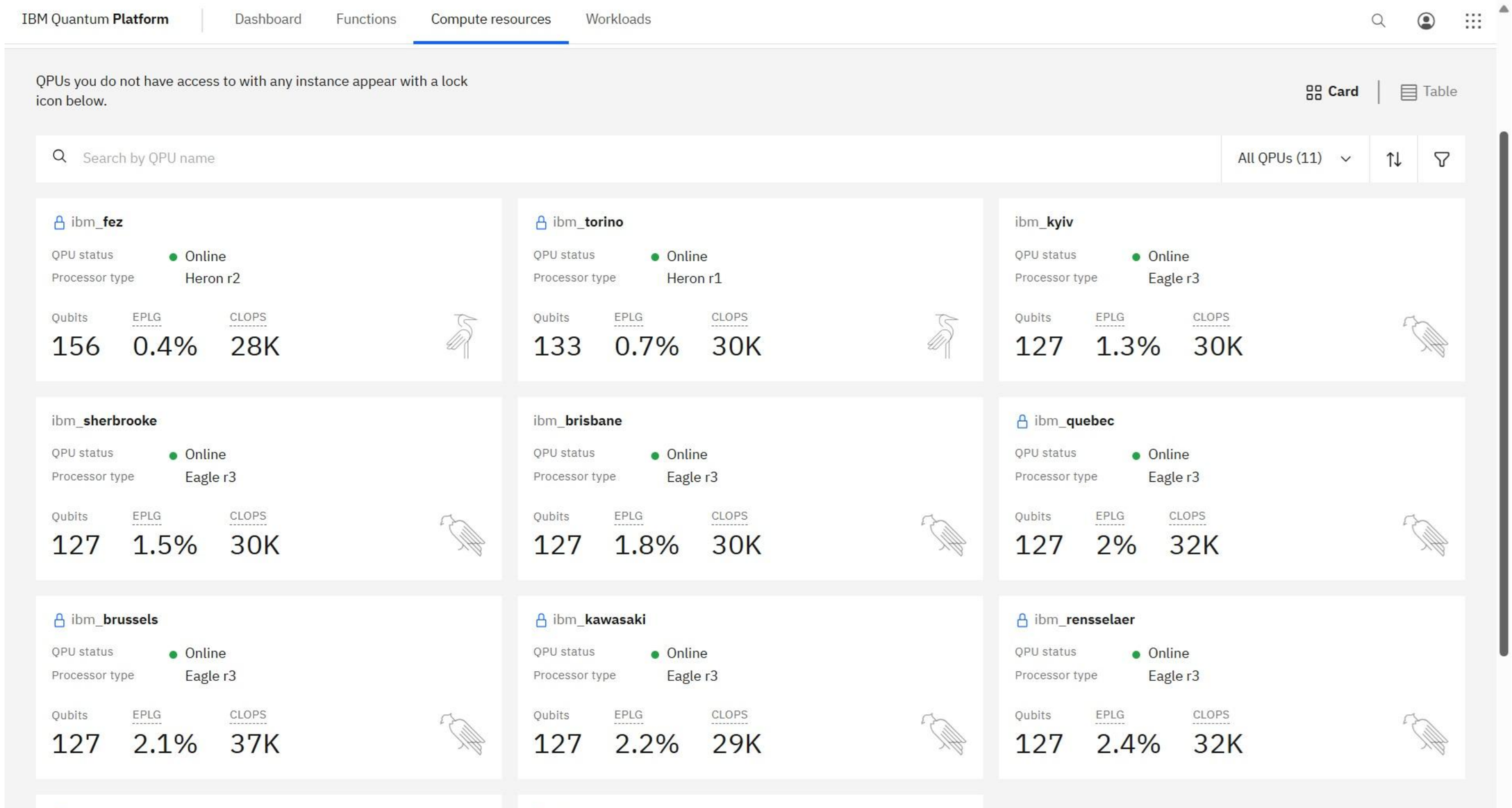
What's new →

- Product update
  Upcoming sunset of backend.run
  3 days ago • Read more
- Product update
  Qiskit is the most performant quantum SDK
  6 days ago • Read more
- Product update
  Qiskit Code Assistant preview release
  7 days ago • Read more
- Product update
  Introducing the Qiskit Functions Catalog
  7 days ago • Read more
- Product update
  What's new in the docs?
  17 days ago • Read more
- Blog
  Qiskit Serverless sets the stage for Qiskit Functions in the cloud
  18 days ago • Read more

3. IBM Composer



Algumas das QPUs (Quantum Processor Unit) IBM disponíveis

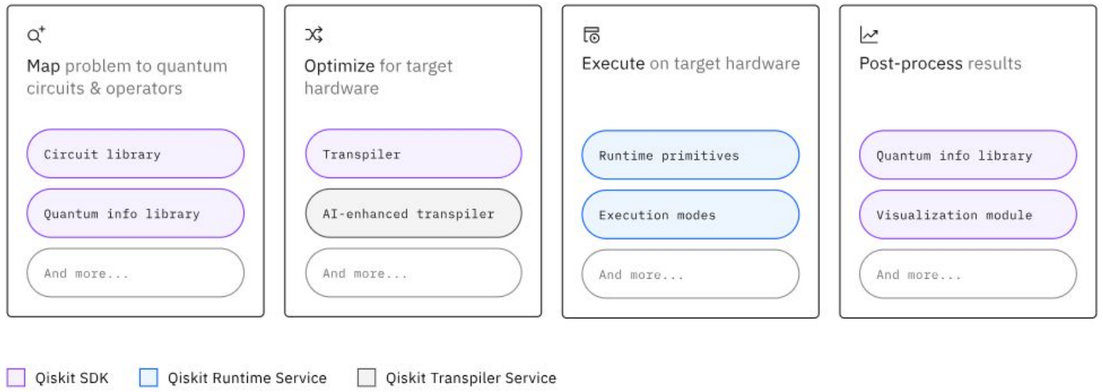


Instalação

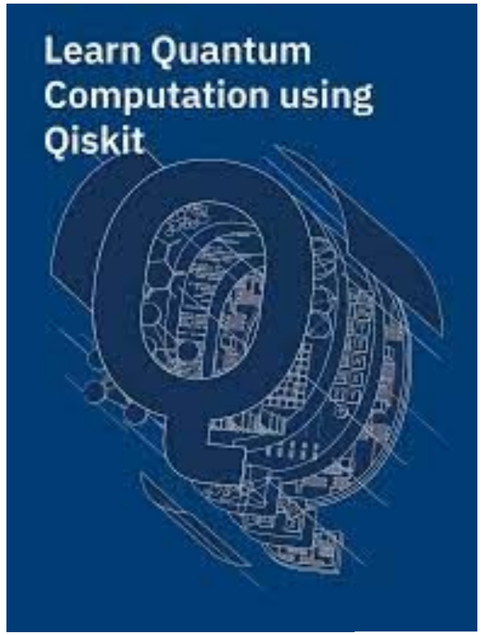
Introduction to Qiskit

The name "Qiskit" is a general term referring to a collection of software for executing programs on quantum computers. Most notably among these software tools is the open-source Qiskit SDK, and the runtime environment (accessed using Qiskit Runtime) through which you can execute workloads on IBM® quantum processing units (QPUs). As quantum technology evolves, so does Qiskit, with new capabilities released every year that expand this core collection of quantum software.

In addition, many open-source projects are part of the broader Qiskit ecosystem. These software tools are not part of Qiskit itself, but rather interface with Qiskit and can provide valuable additional functionality.



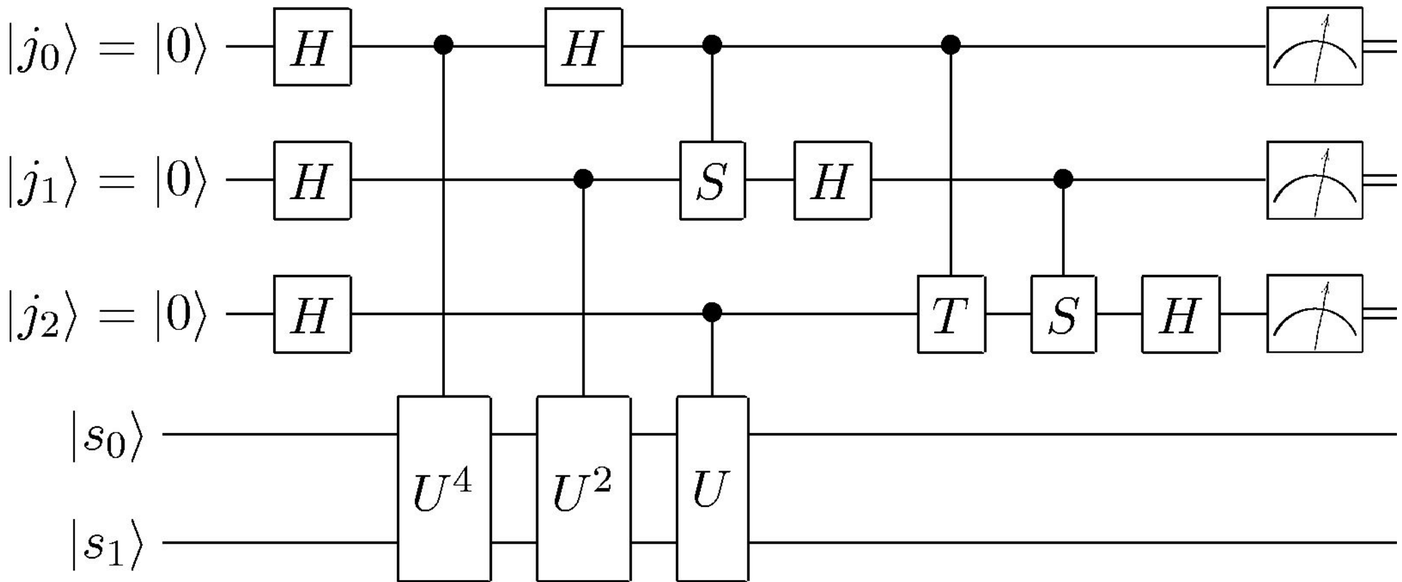
IBM is committed to the responsible development of quantum computing. Learn more and review our responsible quantum principles in the Responsible quantum computing topic.


Learn Quantum Computation using Qiskit


IBM
Associate Certified
Developer
Quantum Computation using Qiskit v0.2X


Quantum Computing in Practice with Qiskit® and IBM Quantum Experience®
Packt

python


$|j_0\rangle = |0\rangle$
$|j_1\rangle = |0\rangle$
$|j_2\rangle = |0\rangle$
$|s_0\rangle$
$|s_1\rangle$
H
S
T
U^4
U^2
U

Considerações finais

A evolução deve continuar

- Apesar da disponibilidade de alguns computadores quânticos, ainda existem muitas limitações, como:
- Taxa de erro de processamento e leitura, necessita métodos de correção de erros e tolerância a falhas eficientes
- Necessidade de maior capacidade de processamento
  - Aumento do número de qubits
  - Processamento distribuído
- Necessidade de armazenamento quântico

Atuação profissional - O FUTURO PRÓXIMO

- Necessidade de formar corpo técnico para atuação em várias frentes e em especial na área da programação
- O campo para atuação profissional nesta área deve crescer rapidamente
- Já há disponibilidade de vagas principalmente nos EUA
- Algumas empresas no Brasil já estão contratando

Referência para Desenvolvimento e Simulação

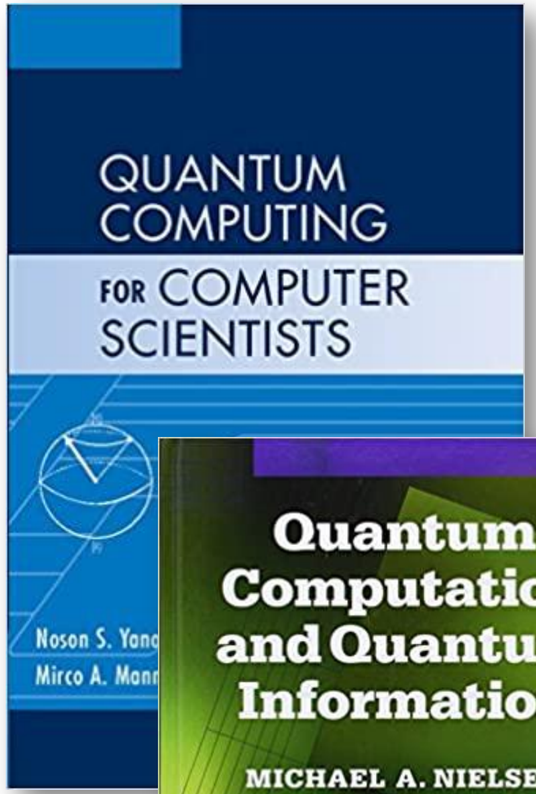
Google - CIRQ
https://github.com/quantumlib/Cirq

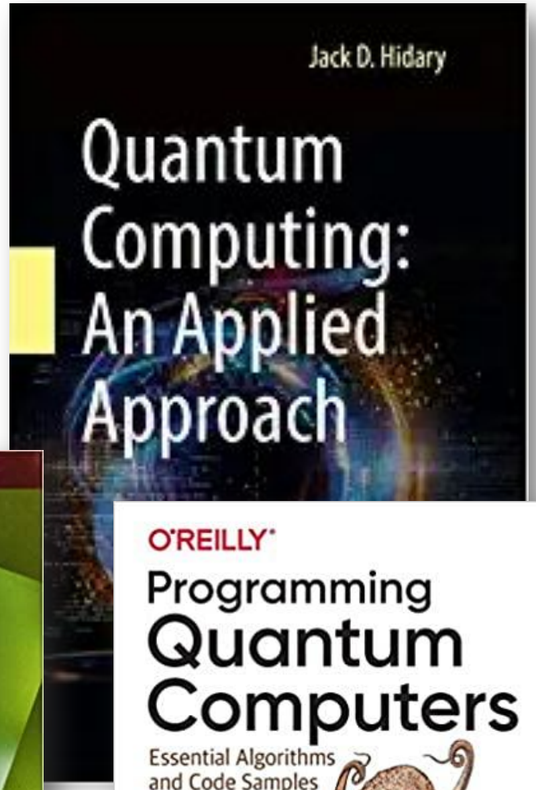
IBM - QisKit
https://qiskit.org/
https://github.com/Qiskit

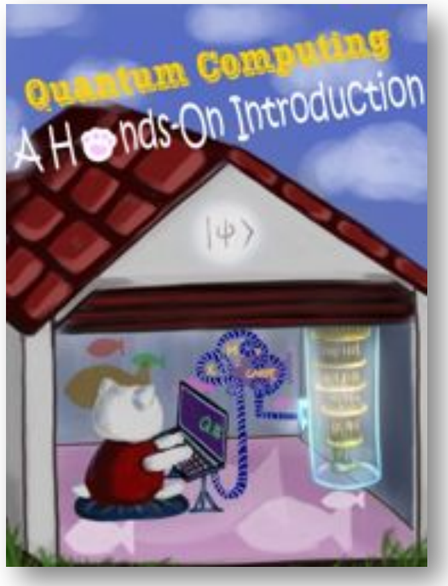
RIGETTI - Forest
https://www.rigetti.com/forest
https://github.com/rigetti/pyquil

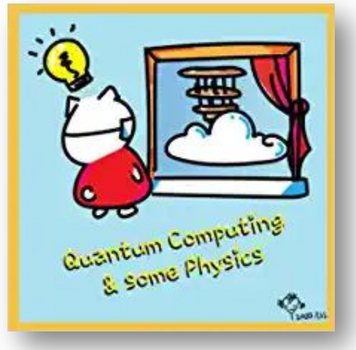
Microsoft - QDK
http://microsoft.com/en-us/quantum/development-kit
https://github.com/Microsoft/Quantum

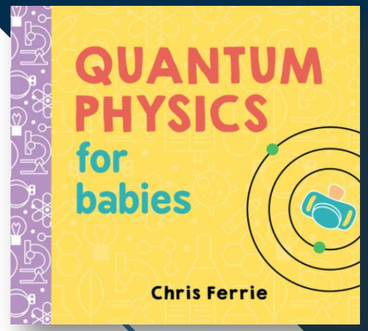
Referências

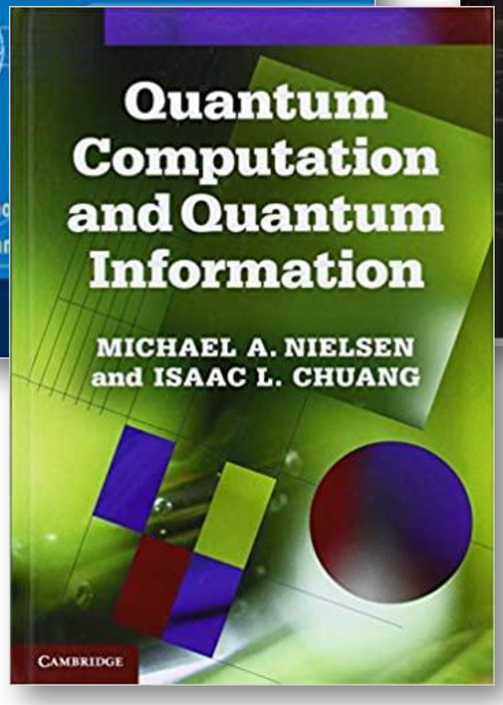


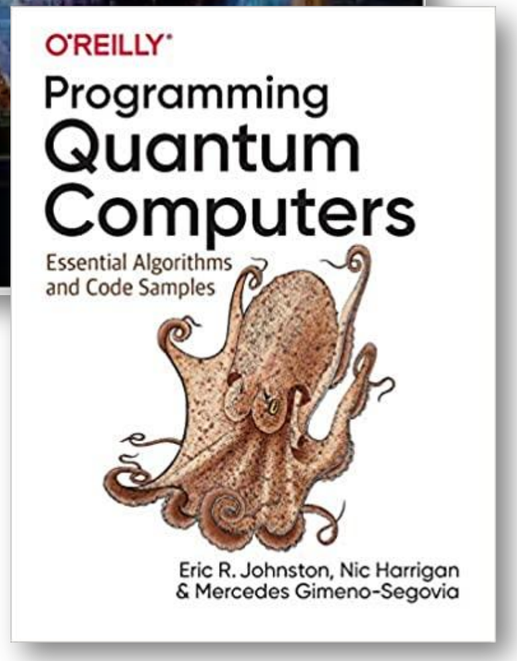


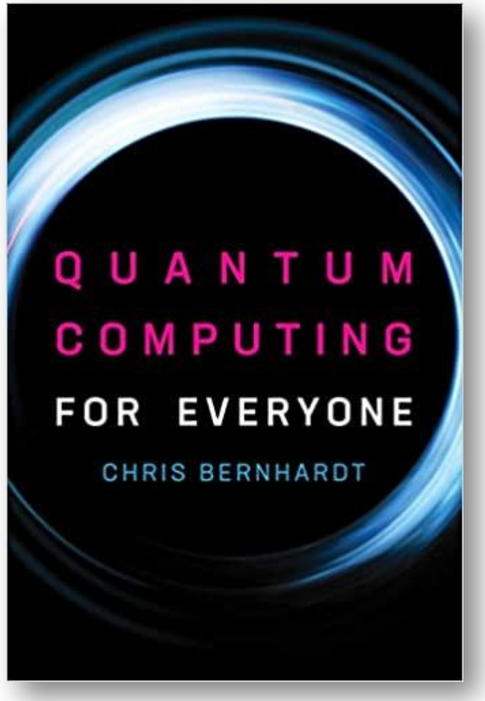


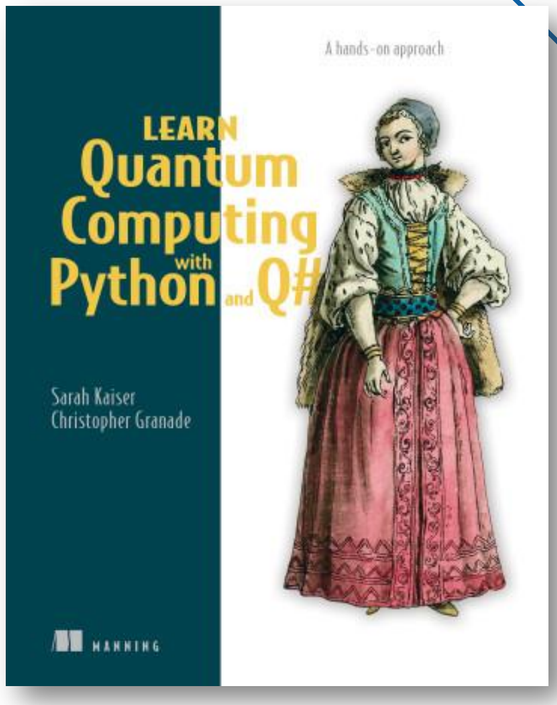












Obrigada!

regina.silveira@usp.br