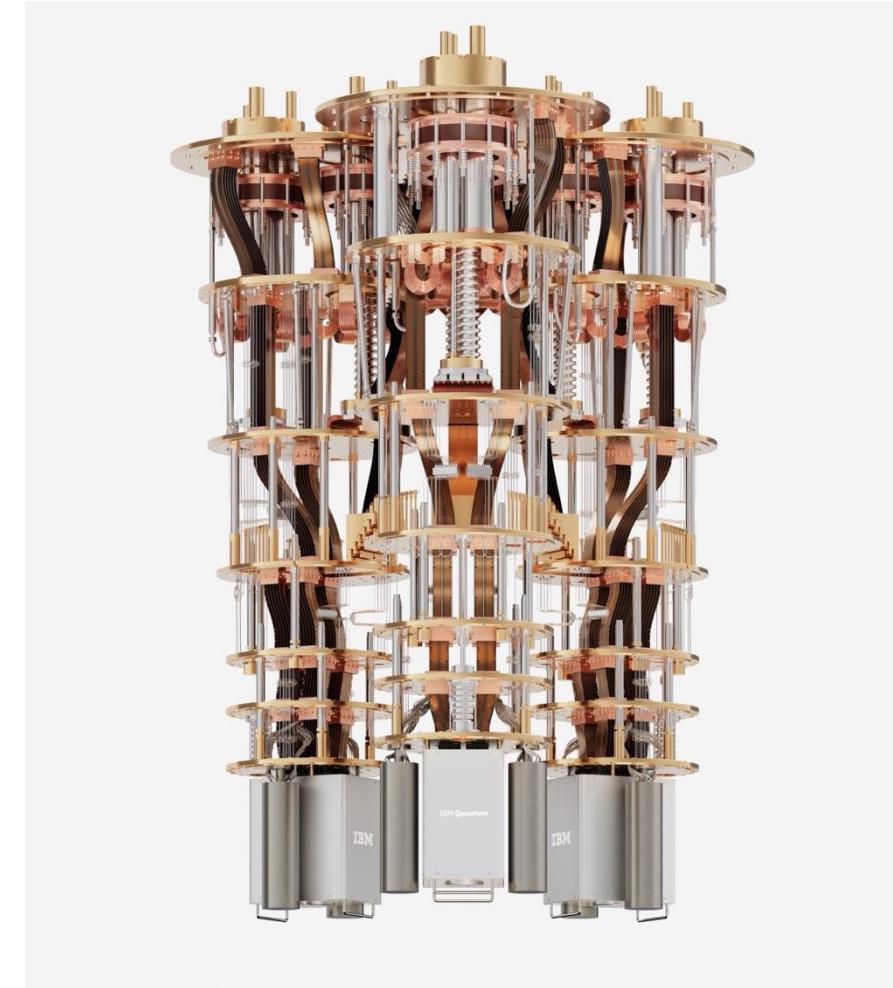


量子コンピューティング 入門

Jan 20, 2025

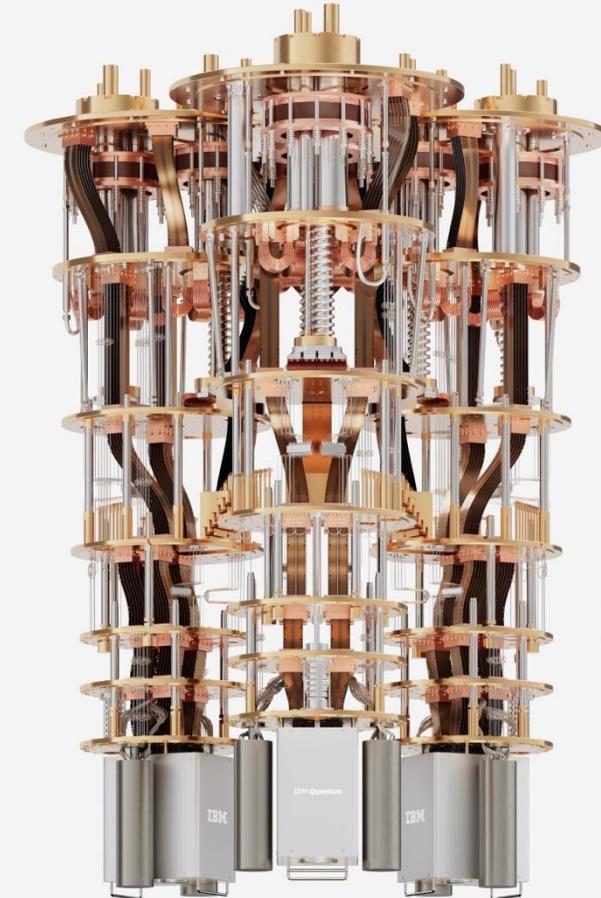
沼田祈史
Kifumi Numata
IBM Quantum



本日のアジェンダ

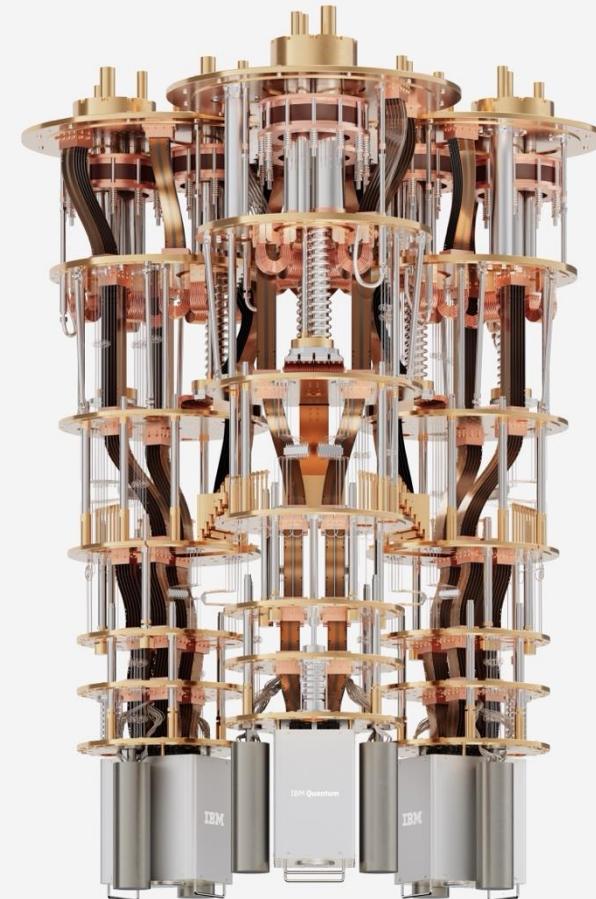
- 13:00-13:05 ご挨拶
- 13:05-14:00 **量子コンピューター入門**
- 14:00-14:45 **Qiskit入門**
- 14:45-15:00 休憩、および、シャンデリア見学
- 15:00-16:00 **量子機械学習入門**
- 16:00-16:30 ThinkLab 半導体/AI ツアー

量子コンピューター 入門



IBM

量子とは？

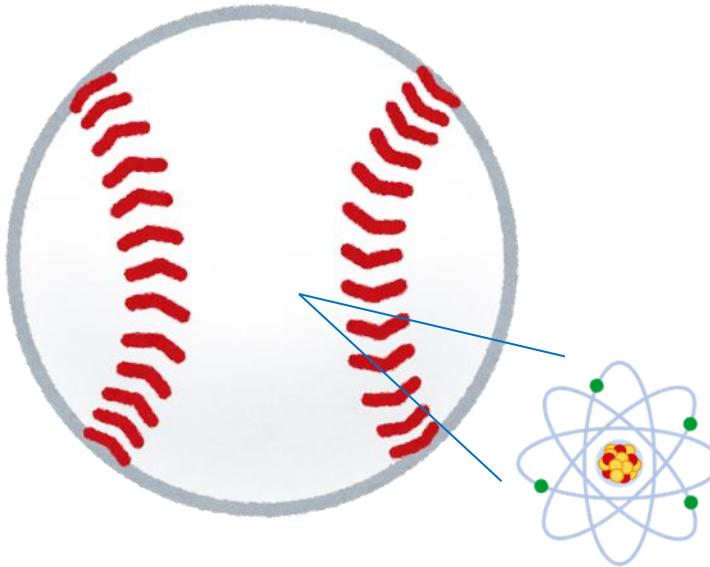


IBM

量子とは？



あらゆるものは、「原子」からできています。

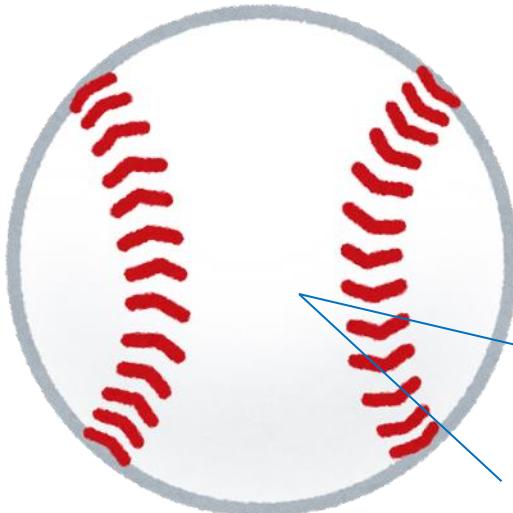


原子

量子とは？

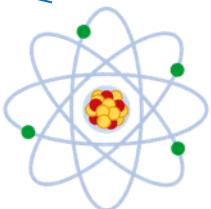


あらゆるものは、「原子」からできています。



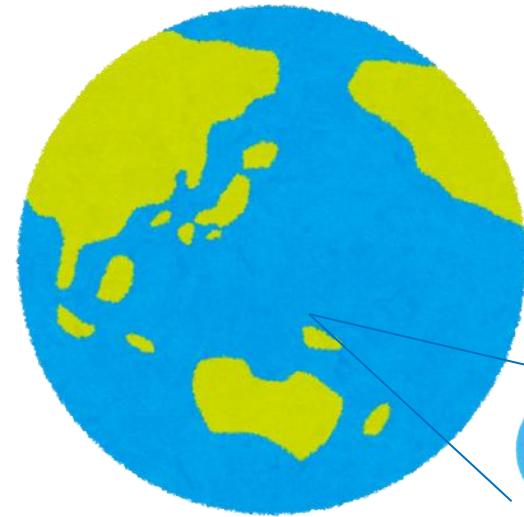
直径7cm

ほぼ
同じ比率



原子

直径0.1nm

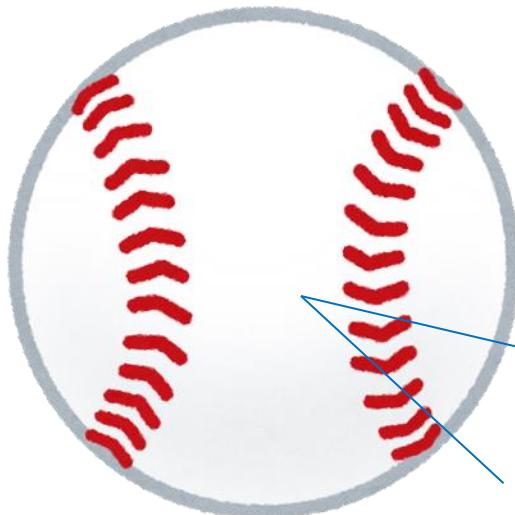


直径1万3000km

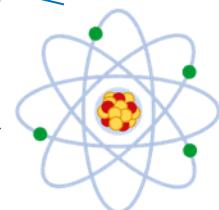
ビー玉
直径1cm

量子とは？

あらゆるものは、「原子」からできています。



直径7cm

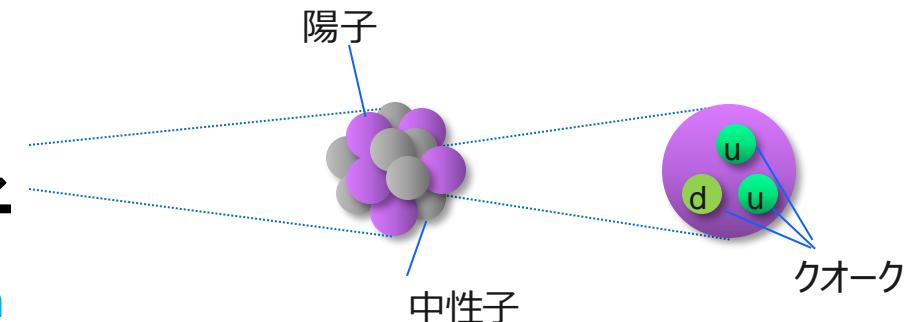


原子

直径0.1nm

ミクロな世界

原子や原子より小さい物質を量子と呼びます。
このミクロな物質は、私たちの常識では説明できない、
不思議なふるまいをします。



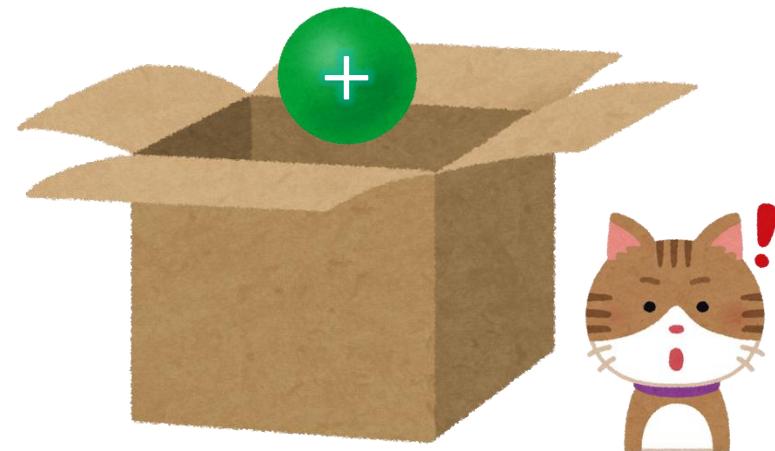
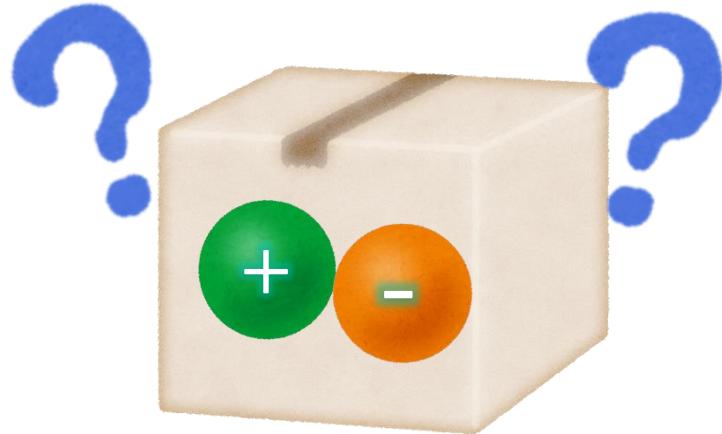
量子の不思議な現象：量子重ね合わせ

量子は1つしかなくとも、複数の状態を同時にとることができます。



量子の不思議な現象：量子重ね合わせ

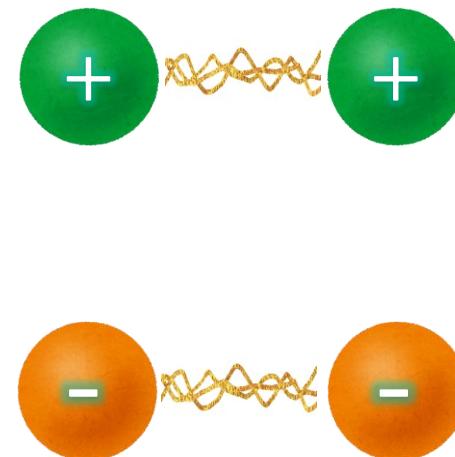
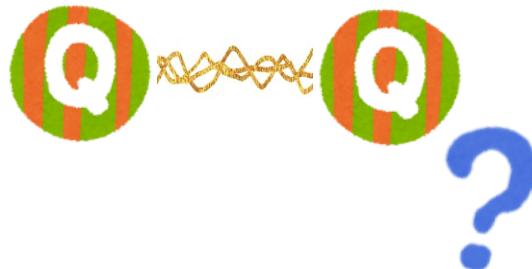
量子は1つしかなくとも、複数の状態を同時にとることができます。



例えば、+と-の2つの状態が共存していて、観測するとどちらかの状態に決まります。（分らなくて大丈夫です！量子とは直感では理解できないものです！）

量子の不思議な現象：量子もつれ

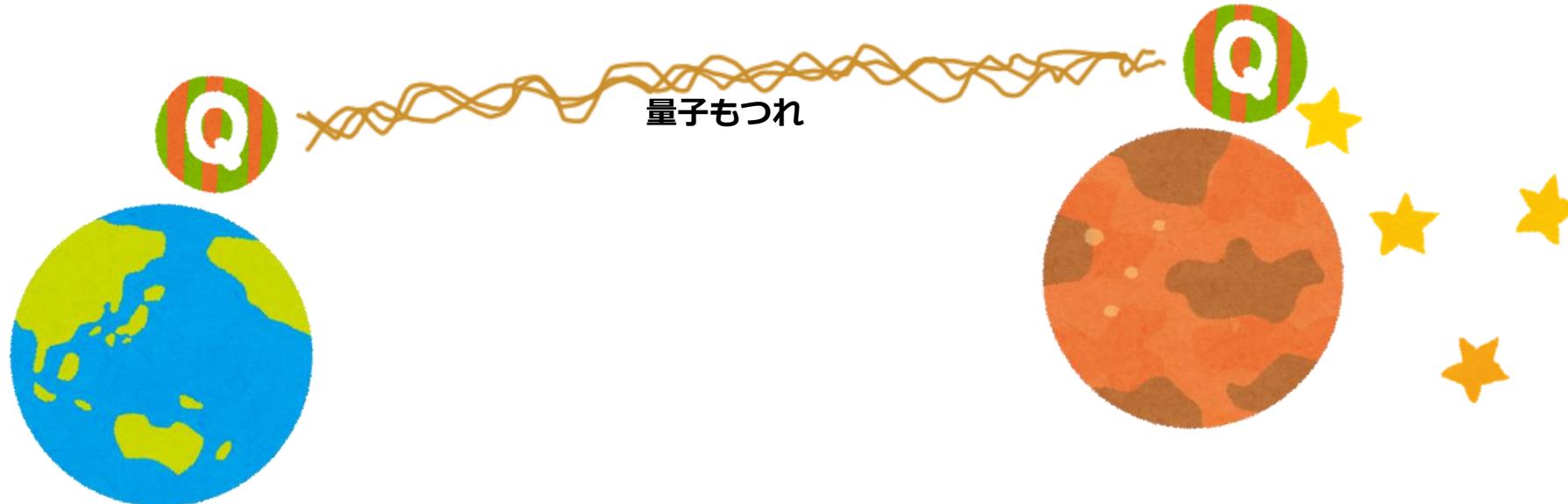
「量子もつれ」という特別な関係のふたごの量子は、
片方の状態を測定すると、もう片方の状態が測定しなくても分かります。
(片方しか測定していないのに！)



量子の不思議な現象：量子もつれ

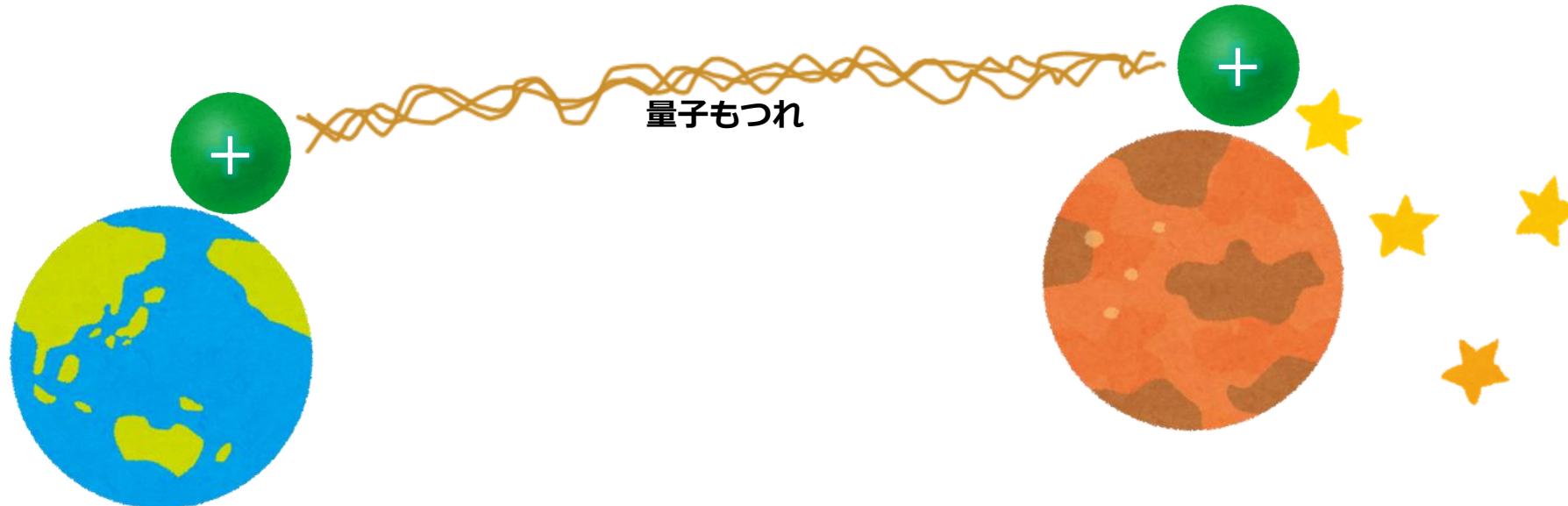
「量子もつれ」状態のふたごの量子を離れ離れにします。

地球の量子を測定して「+」と判明したら、火星の量子の状態はどうなるでしょうか？



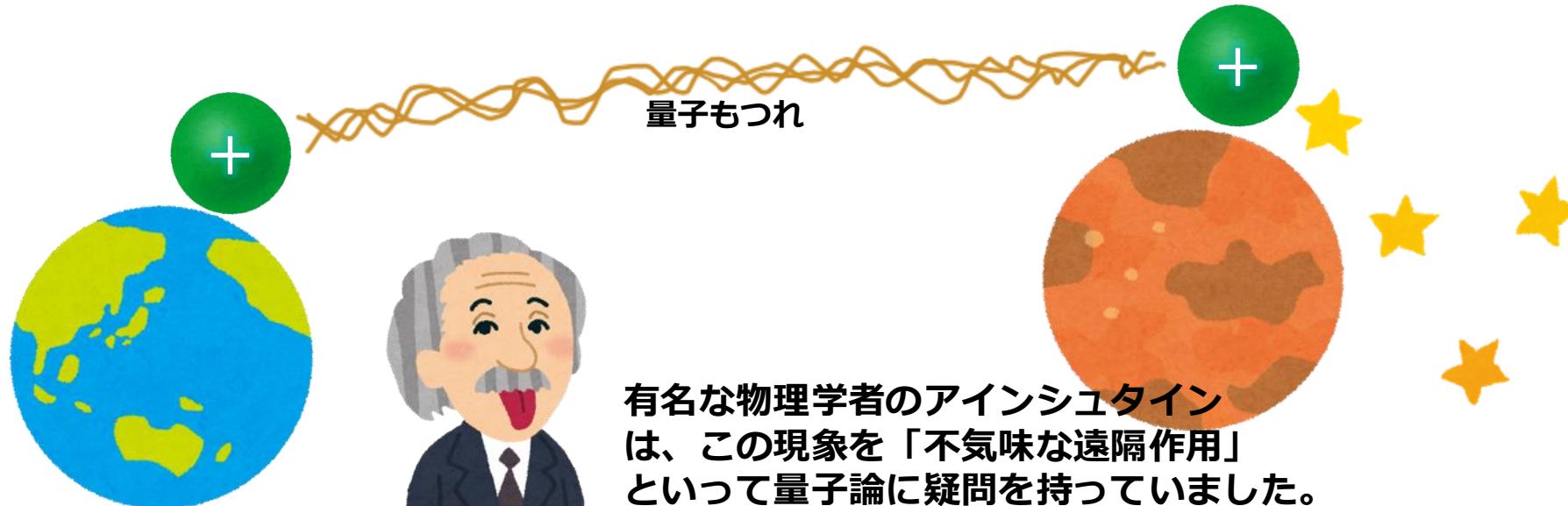
量子の不思議な現象：量子もつれ

「量子もつれ」状態のふたごの量子を離れ離れにしても
片方を測定するだけで、遠く離れたもう片方の量子の状態は測定しなくとも分かります！



量子の不思議な現象：量子もつれ

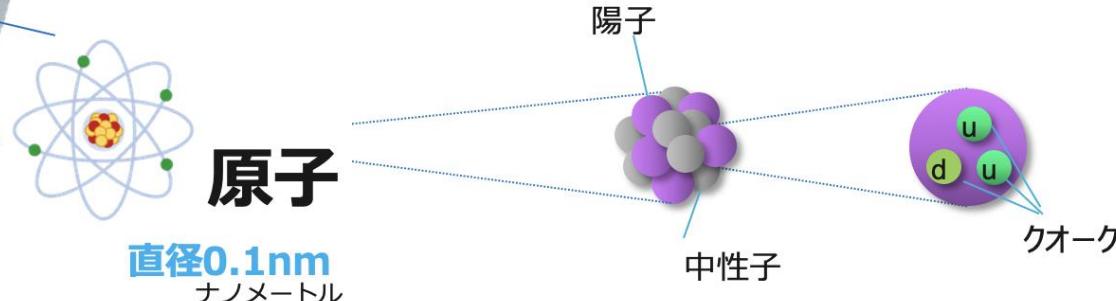
「量子もつれ」状態のふたごの量子を離れ離れにしても
片方を測定するだけで、遠く離れたもう片方の量子の状態は測定しなくとも分かります！



量子コンピューターは 量子の現象を仕組みとして使ったコンピューター

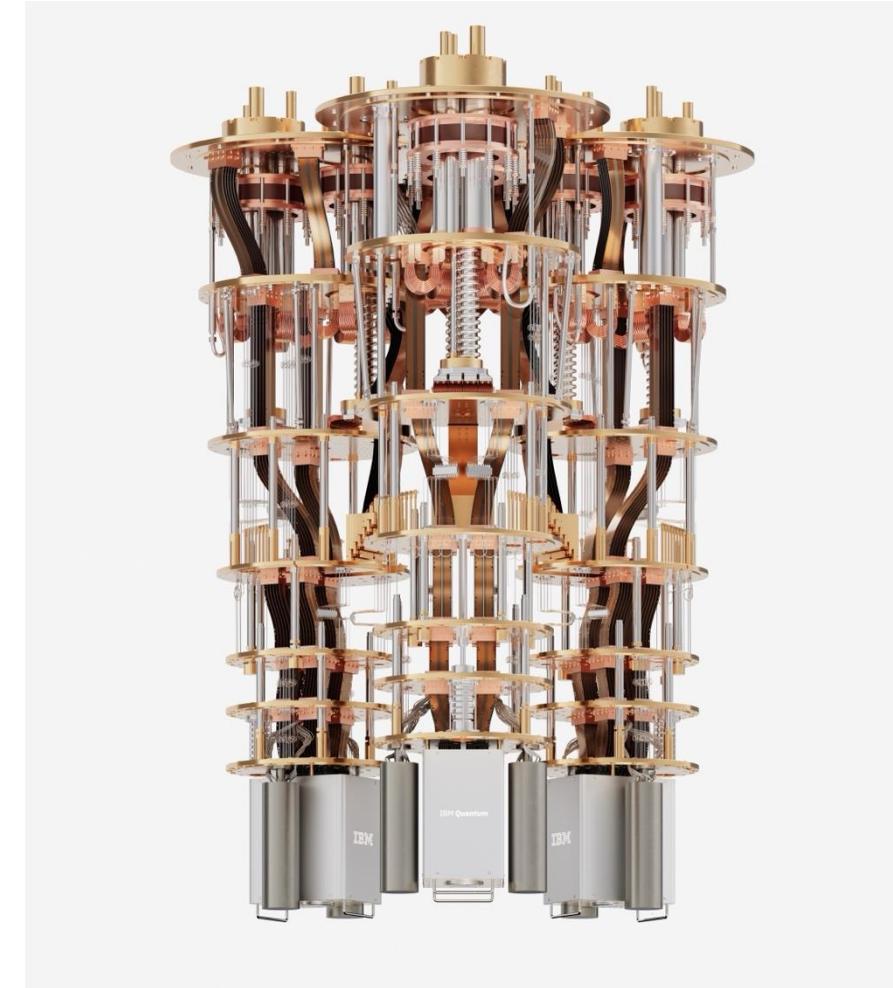


量子力学は、ミクロの世界を表現する物理法則
「重ね合わせ」「量子もつれ（エンタングルメント）」
などの量子現象を表現する手法。

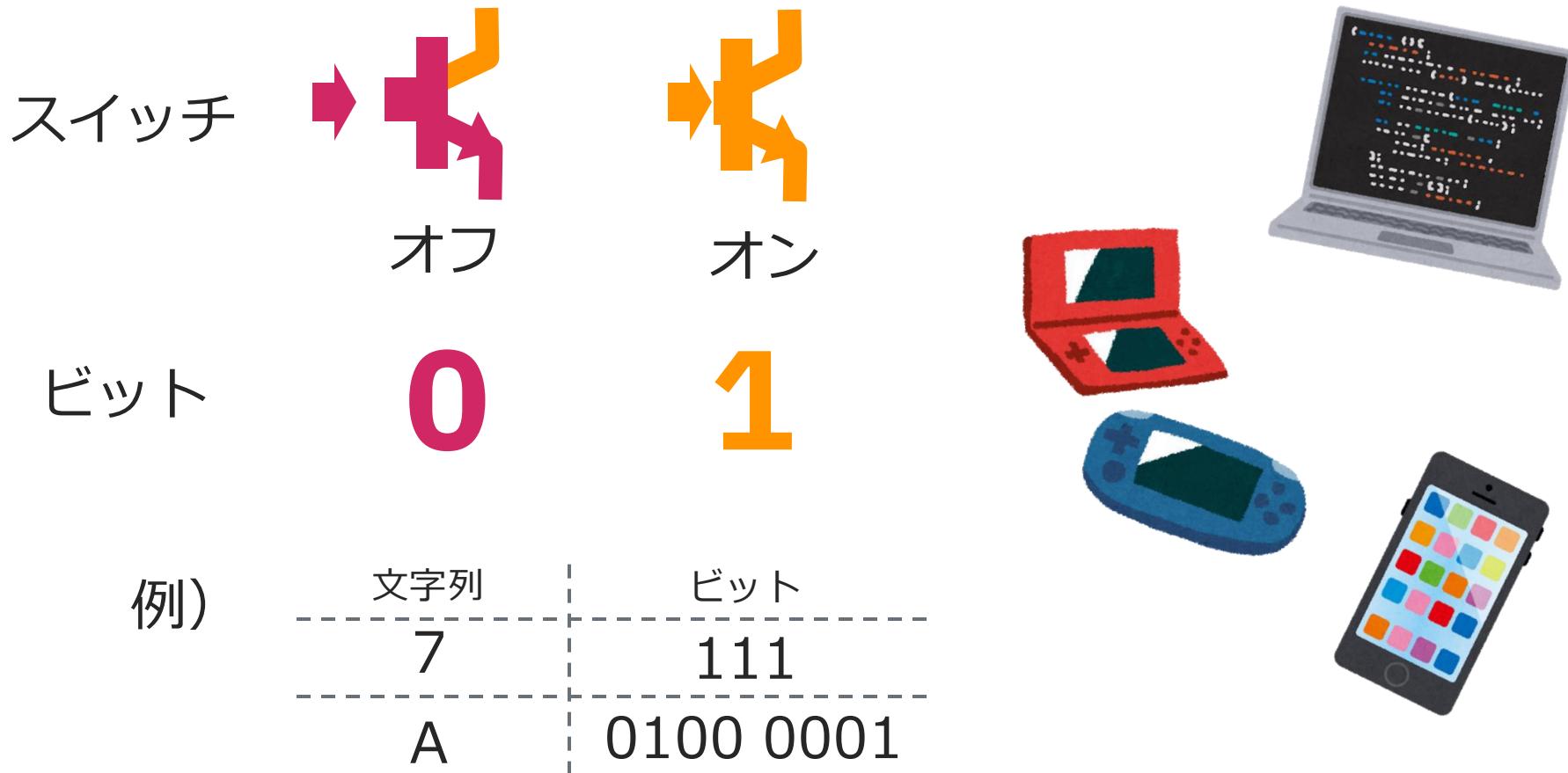


量子コンピューターは、この自然界に存在する**量子力学**の法則を利用したもの。
これまでのコンピューターの計算処理とは根本的に異なる手法を使っています。

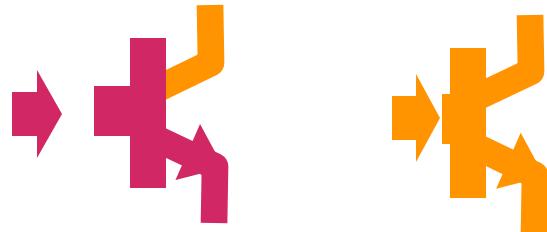
量子計算入門



コンピューターの中は、ビットで計算



いつも使っている コンピューターのビット



0 または 1

どちらか

いつも使っている
コンピューターのビット

0 または **1**

どちらか

量子コンピューターの
量子ビット

0 と **1**

両方

「重ね合わせ」

いつも使っている
コンピューターのビット

0 または **1**

どちらか



量子コンピューターの
量子ビット

0 と **1**

両方

「重ね合わせ」

いつも使っている コンピューターのビット

0 または **1**

どちらか

コイン

表

おもて



コイン

裏

うら

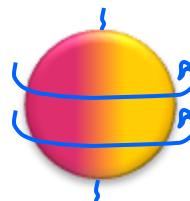
量子コンピューターの 量子ビット

0 と **1**

両方

「重ね合わせ」

くるくる回っているコイン（イメージ）



測定すると表か裏にバシッと決まる

いつも使っている
コンピューターのビット

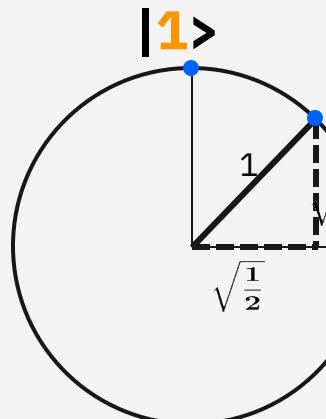
0 または 1

どちらか

量子コンピューターの
量子ビット

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

0と1の「重ね合わせ」

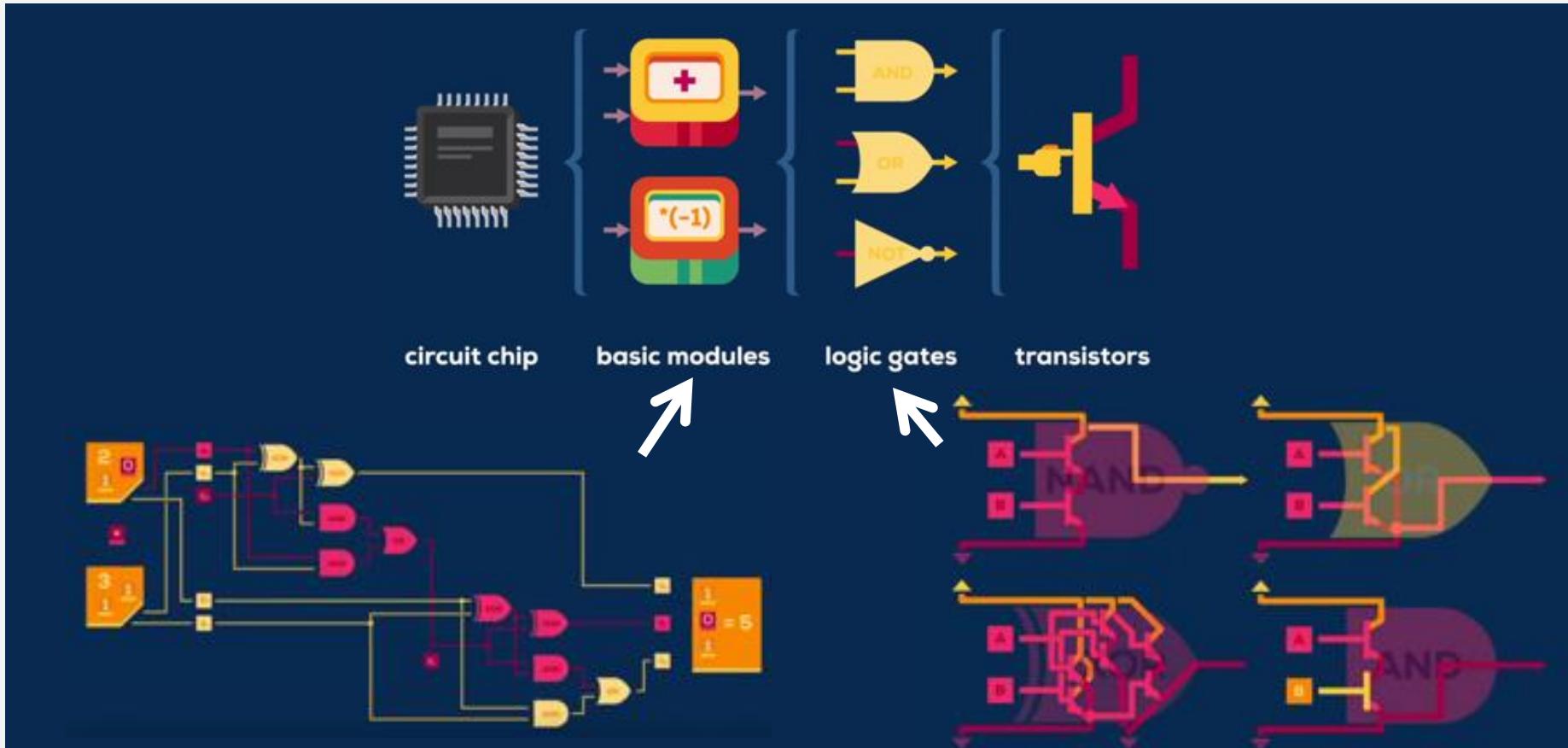


$$|0\rangle = 1 \times |0\rangle + 0 \times |1\rangle$$

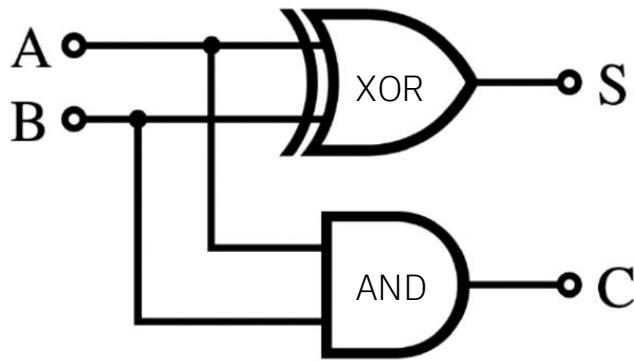
$$|1\rangle = 0 \times |0\rangle + 1 \times |1\rangle$$

|>:量子ビットを表す記号

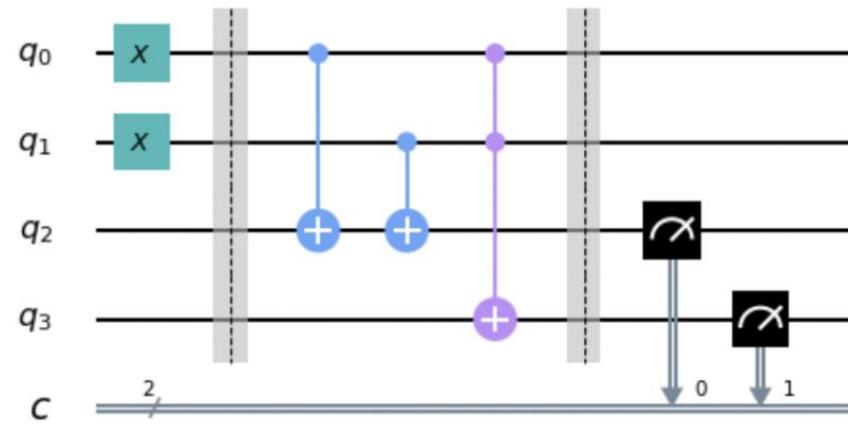
現在のコンピューター



ゲートと回路図を使った計算

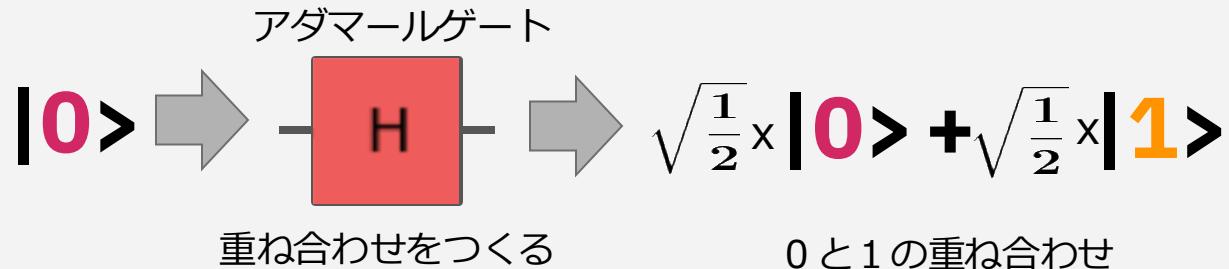


古典回路

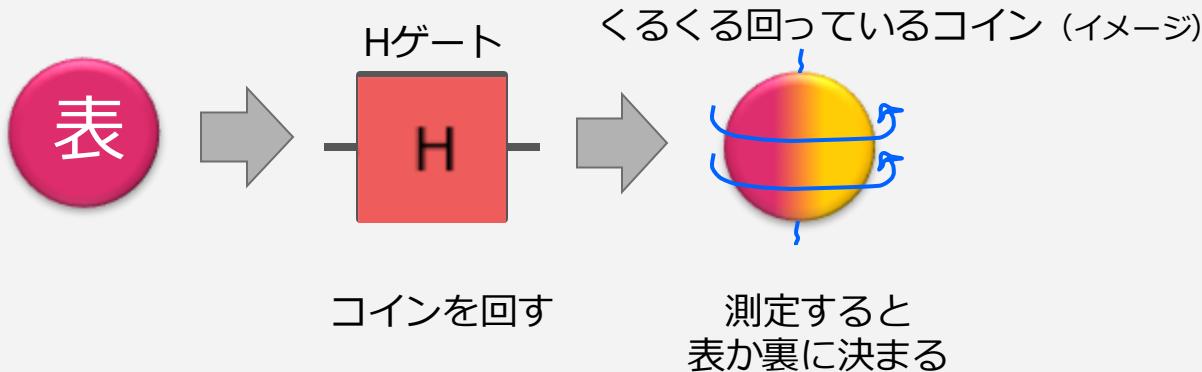
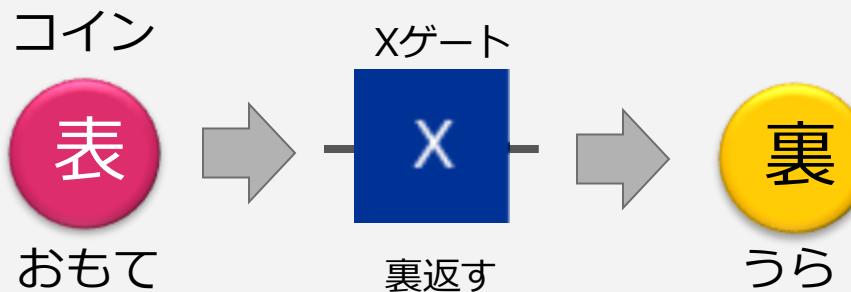


量子回路

量子コンピューターのゲート計算

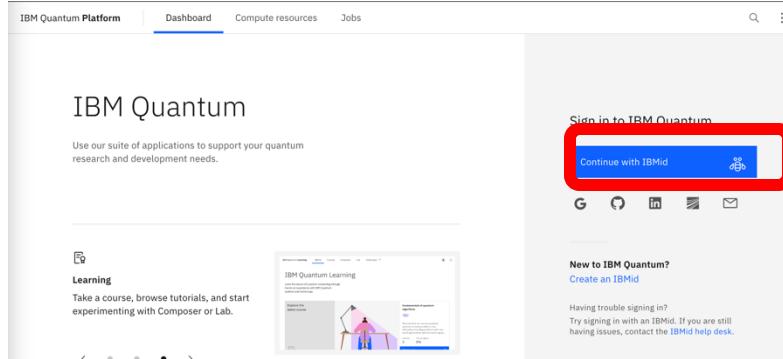


量子コンピューターのゲート計算

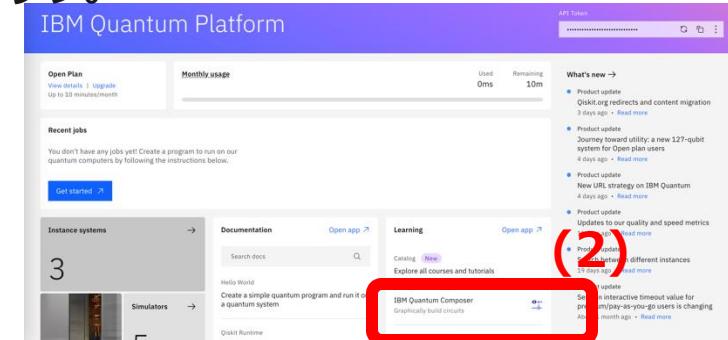


ハンズオン: IBM Quantum Composer

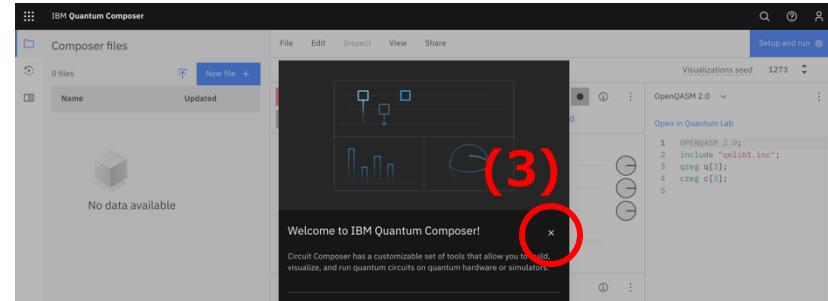
(1) IBM Quantum にログインします。URL:
<https://quantum.ibm.com/>



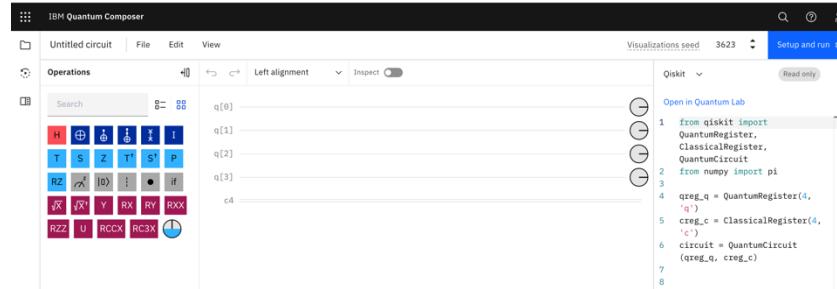
(2) 中央下の方の「IBM Quantum Composer」をクリック。



(3) ポップアップウィンドウは「x」をクリックして、閉じます。



(4) この画面になつたら準備完了です。



1量子ビット回路

The screenshot shows the IBM Quantum Composer interface. At the top, it says "IBM Quantum Composer". Below that is a menu bar with "Untitled circuit", "File", "Edit", "View", and "Visual". On the left, there's a sidebar with icons for operations, search, and Qiskit. The main area is titled "Operations" and shows a grid of quantum gates. A red arrow points from the text below to the trash bin icon next to the q[1] register. The circuit itself has four qubits labeled q[0], q[1], q[3], and c4. There are several gates applied to q[0]: H, CNOT, T, S, Z, T†, S†, P, and a multi-controlled NOT gate. The q[1] register is empty. The q[3] register has a CNOT gate followed by a multi-controlled NOT gate. The c4 register is also empty.

マウスでq[1]をクリックするとゴミ箱マークが出てくるので、
クリックして消します。
q[0]だけにして、1量子ビット回路の準備をします。

```
1 from qiskit import  
2 QuantumCircuit  
3 from numpy import  
4  
5 qreg_q = QuantumRe  
6 creg_c = Classical  
7 circuit = QuantumC
```

1量子ビット回路

The screenshot shows the IBM Quantum Composer interface. On the left, there's a sidebar titled "量子ゲート" (Quantum Gates) with a search bar and a grid of gate icons. A red circle highlights the CNOT gate icon. In the center, there's a workspace titled "量子回路" (Quantum Circuit) with two horizontal lines representing qubits and cubits. A red arrow points from the CNOT icon in the sidebar to the CNOT gate icon on the workspace. On the right, there's a panel titled "Qiskit" which displays the generated Qiskit code:

```
from qiskit import QuantumRegister, ClassicalRegister, QuantumCircuit
from numpy import pi
qreg_q = QuantumRegister(1, 'q')
creg_c = ClassicalRegister(4, 'c')
circuit = QuantumCircuit(qreg_q, creg_c)
```

量子ゲートをマウスでドラッグ&ドロップして、
量子回路を作ります。
右側には、Qiskitのコードが自動生成されます。

1. Xゲート(NOTゲート)

図の回路を作ってみてください。下に表示される棒グラフの変化を確認しましょう。

1-1) $q[0]$ 



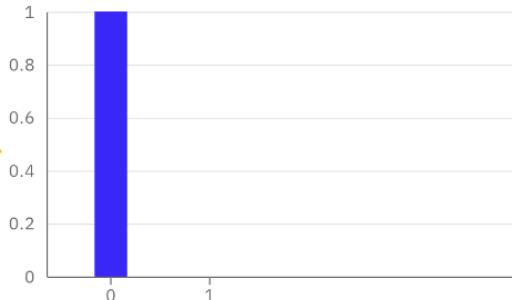
1-2) $q[0]$ 

左下の棒グラフが「Probabilities」になっている場合は、クリックして「Statevector」に変更してください。

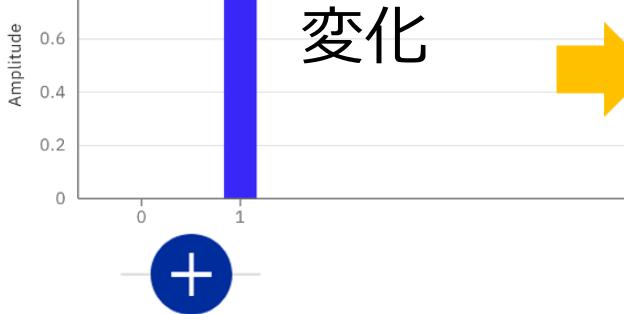


1-3) $q[0]$ 

初期状態は $|0\rangle$



$|1\rangle$ に
変化

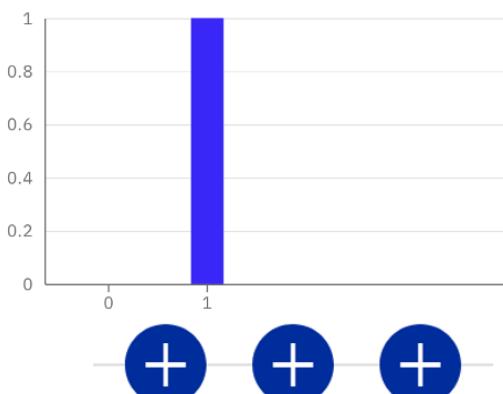
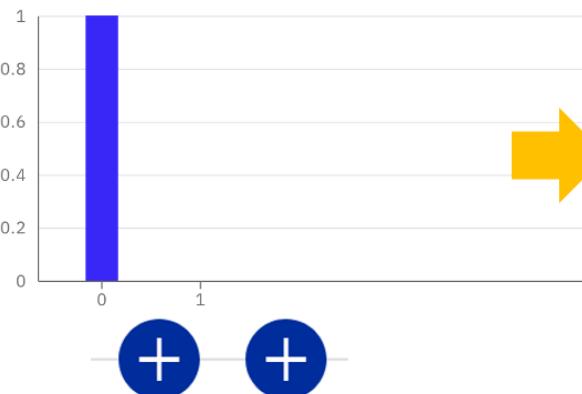


棒グラフ (Statevector 表示) は
量子ビットの状態

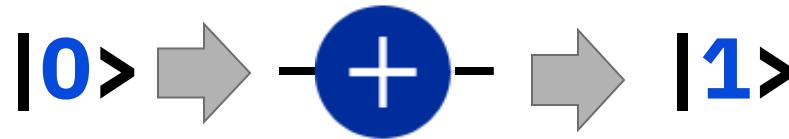
$\alpha \times |0\rangle + \beta \times |1\rangle$
の α, β (確率振幅) です。

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

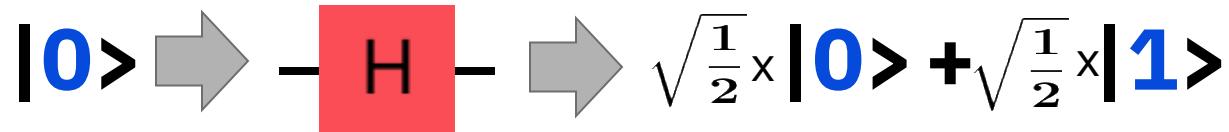
$$|1\rangle = 0 \times |0\rangle + 1 \times |1\rangle$$



量子コンピューターの計算方法

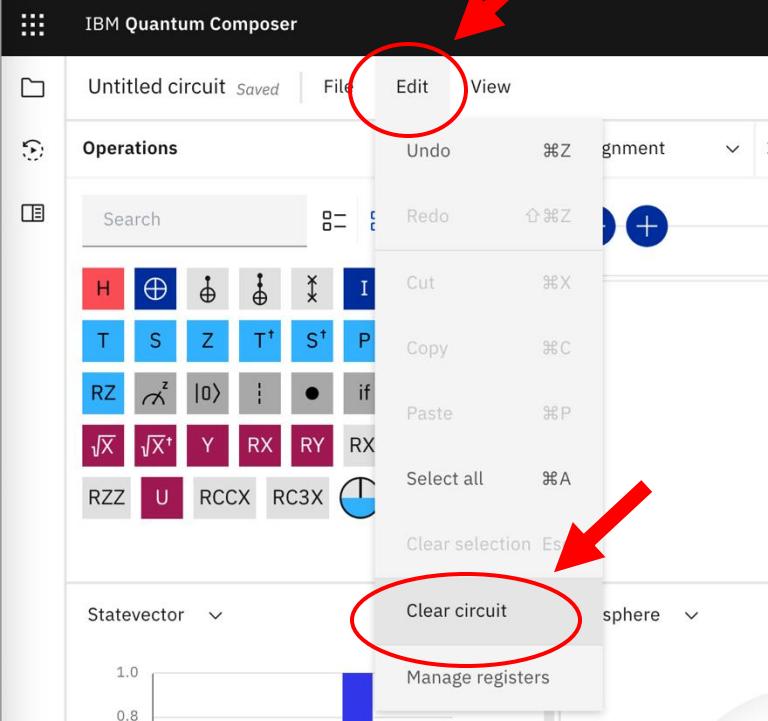


ノット（反転）ゲート

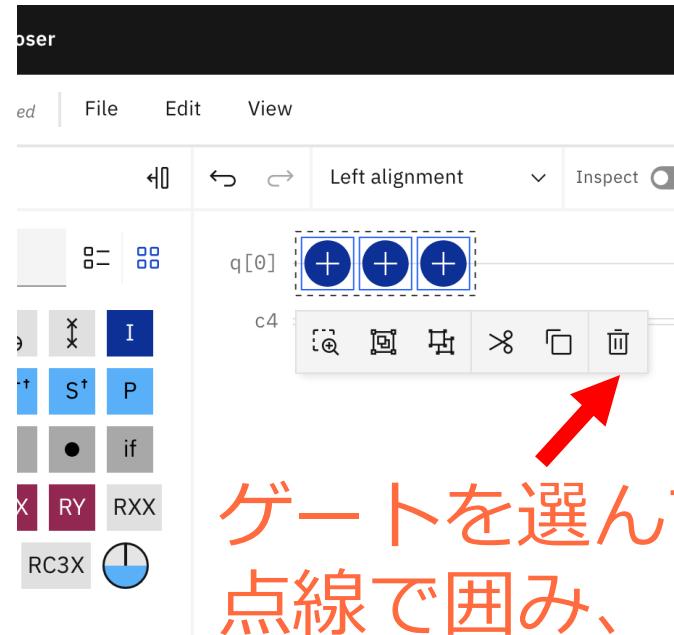


重ね合わせをつくる

置いたゲートを取り除く



または

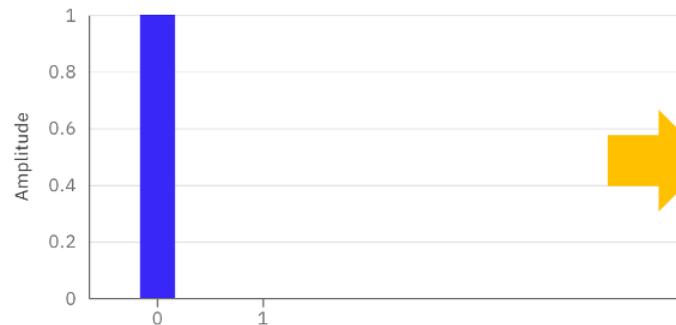
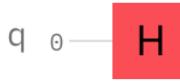


ゲートを選んで
点線で囲み、
ゴミ箱マークを
クリック

2. Hゲート

図の回路を作ってみてください。下に表示される棒グラフの変化を確認しましょう。

2-1)



重ね合わせ



例えば1000回同じ状態を作って、測定すると約500回は0が観測され、約500回は1が観測される状態。

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

$$\begin{aligned} & 0.707 \times |0\rangle + 0.707 \times |1\rangle \\ &= \frac{1}{\sqrt{2}} \times |0\rangle + \frac{1}{\sqrt{2}} \times |1\rangle \end{aligned}$$

2. Hゲート

図の回路を作ってみてください。下に表示される棒グラフの変化を確認しましょう。

2-1)



2-2)



2-3)



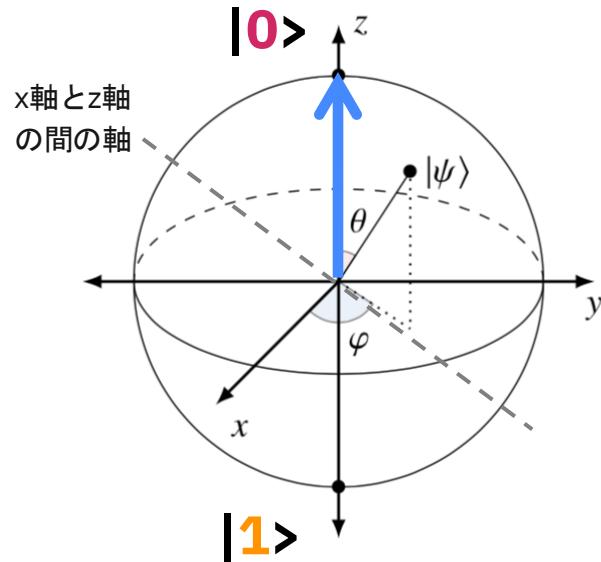
量子コンピューターの計算方法

$$|0\rangle \xleftarrow{H} \sqrt{\frac{1}{2}}|0\rangle + \sqrt{\frac{1}{2}}|1\rangle$$

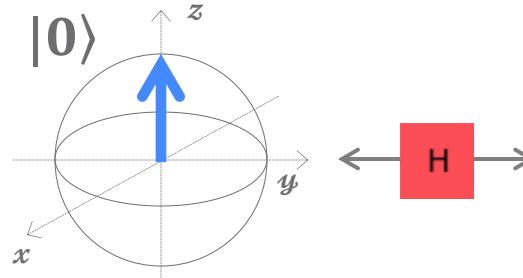


$$|1\rangle \xleftarrow{H} \sqrt{\frac{1}{2}}|0\rangle - \sqrt{\frac{1}{2}}|1\rangle$$

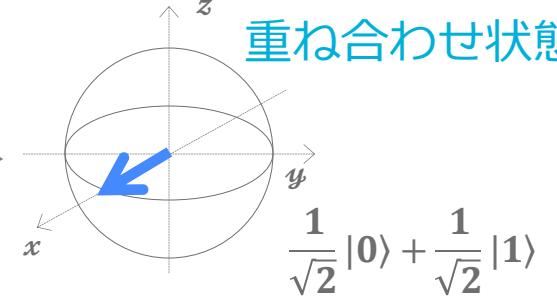
ブロツホ球



北極 : 0の確率が100%

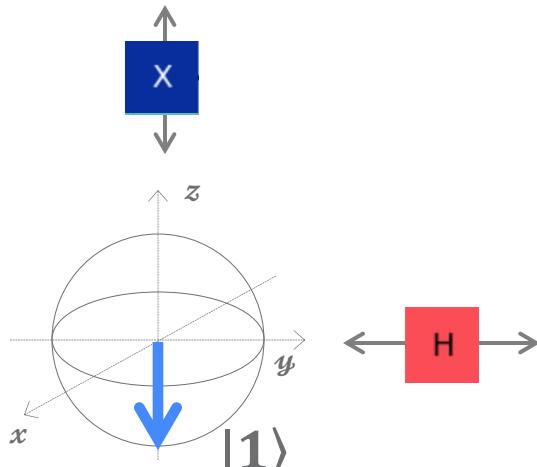


赤道 : 0と1が50%ずつの重ね合わせ状態



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

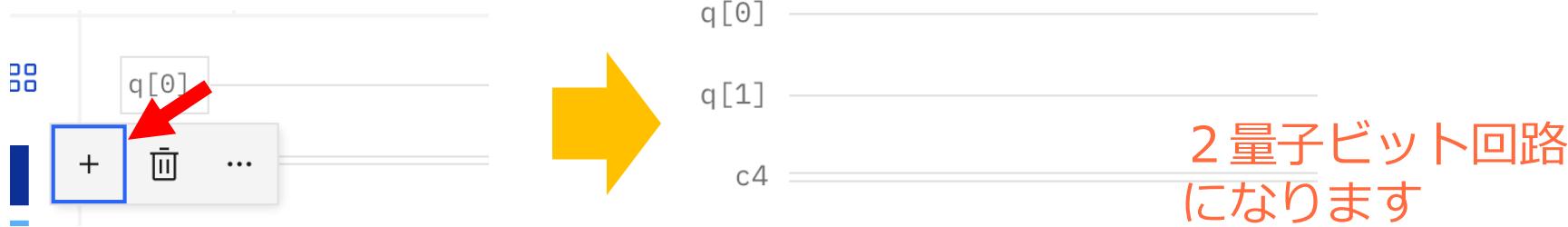
南極 : 1の確率が100%



$$\frac{1}{\sqrt{2}}|0\rangle - \frac{1}{\sqrt{2}}|1\rangle$$

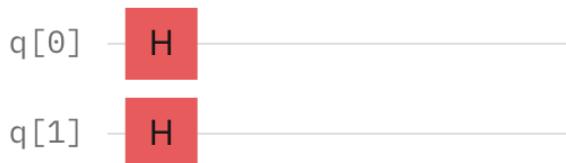
3. 量子重ね合わせ

q[0]をクリックして、さらに「+」マークをクリックして、2量子ビットの回路を準備します。



図の回路を作成してください。下に表示される棒グラフの変化を確認しましょう。

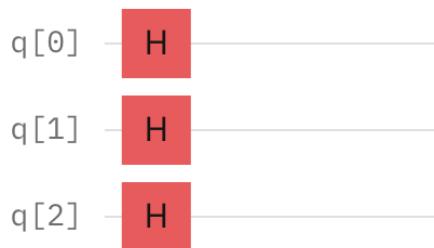
3-1)



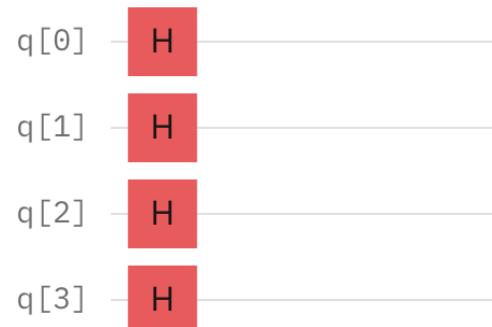
さらにq[1]をクリックして、さらに「+」マークをクリックして、3量子ビット、4量子ビット、5量子ビットの時の重ね合わせ状態を確認します。



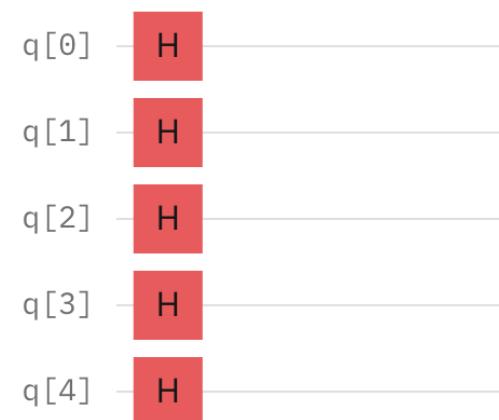
3-2)



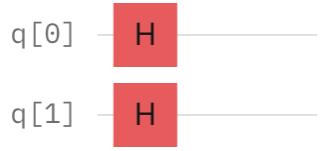
3-3)



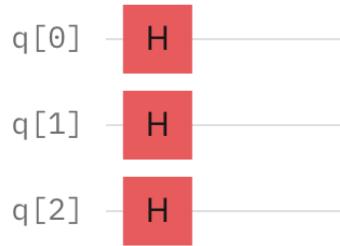
3-4)



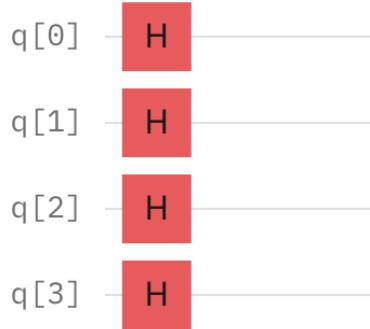
3-1)



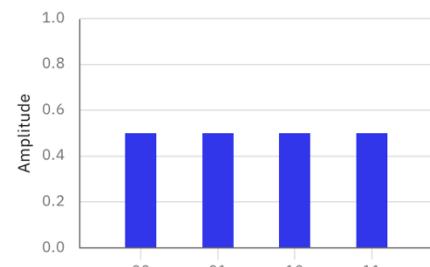
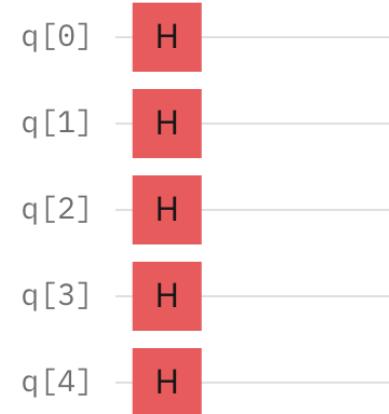
3-2)



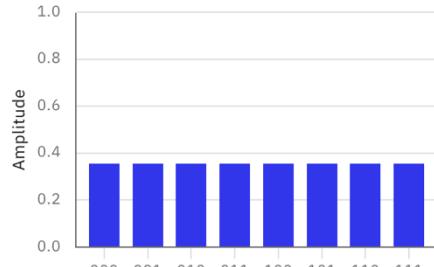
3-3)



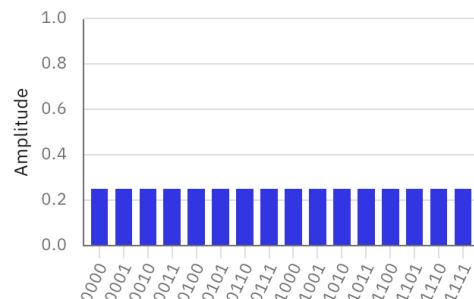
3-4)



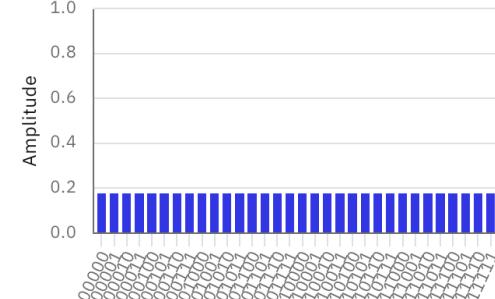
4個



8個



16個

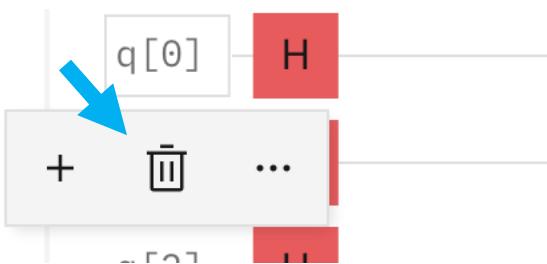


32個

量子ビット数(n)が増えるにつれて、量子状態が倍々に(2^n 個に)増えていくことがわかります。

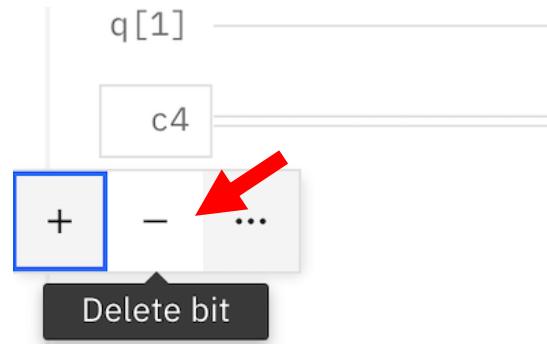
2量子ビット・2古典ビットの状態を作る

$q[0]$ をクリックして、さらに「ゴミ箱」マークをクリック、を繰り返して、2量子ビットの回路を準備します。



2量子ビット回路
にします

次に、 $c4$ をクリックして、「-」マークをクリックするを2回繰り返して、2古典ビットにします。



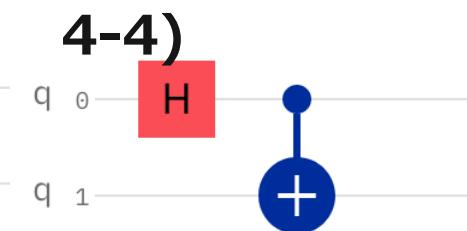
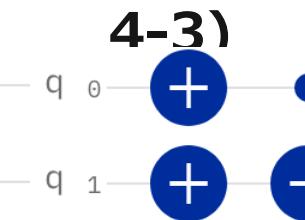
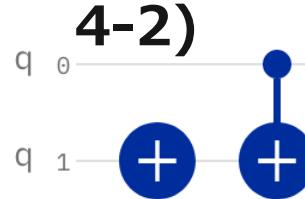
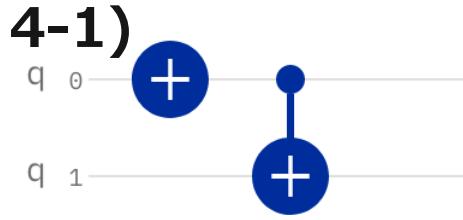
2古典ビットになります

4. CNOTゲート(制御Xゲート)

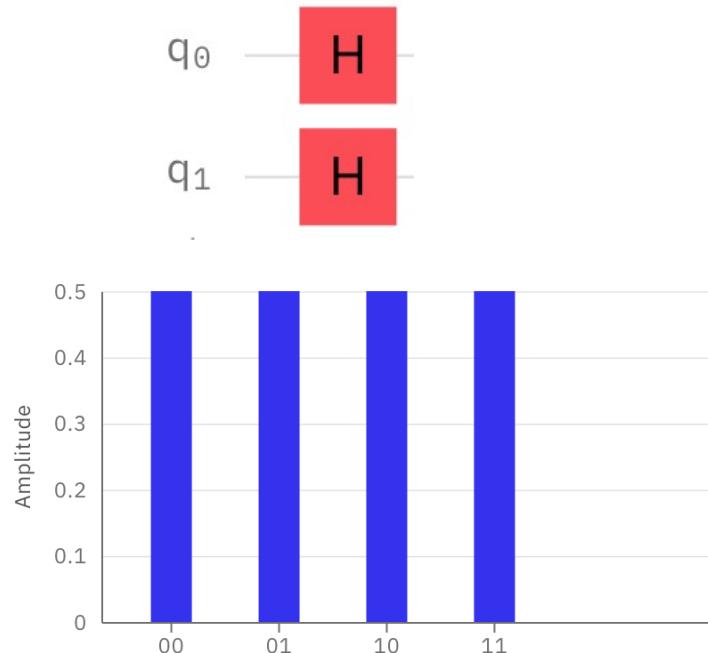
制御ビットが $|1\rangle$ のときのみ、目標ビットを反転（NOT）するゲートです。



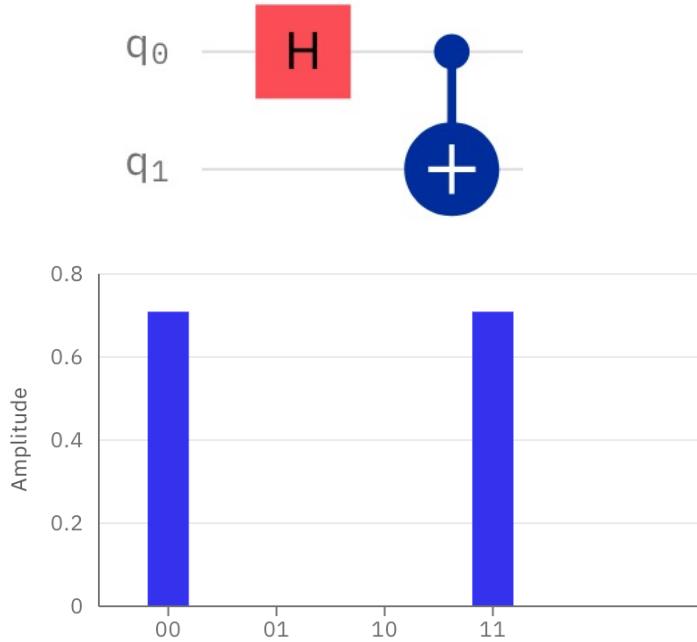
入力		出力	
目標 ビット	制御 ビット	目標 ビット	制御 ビット
0	0	0	0
1	0	1	0
0	1	1	1
1	1	0	1



量子重ね合わせ

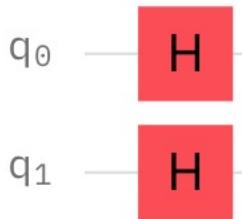


量子もつれ (エンタングルメント)



CNOTゲートは、
エンタングルメントを作ります。

量子重ね合わせ



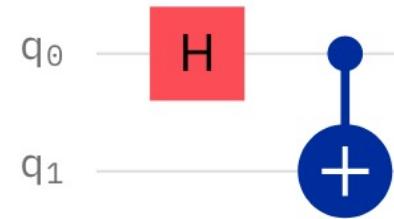
0 0 ... 25%

0 1 ... 25%

1 0 ... 25%

1 1 ... 25%

量子もつれ (エンタングルメント)



0 0 ... 50%

0 1 ... 0%

1 0 ... 0%

1 1 ... 50%

1量子ビット目が0だと分かったら
2量子ビット目も0

5.量子コンピューターで実験

ファイル名を変更

(2) (Entanglementなど)

The screenshot shows the IBM Quantum Compose interface. On the left, the 'Operations' panel displays a grid of quantum gates. A red box highlights the 'Measurement' gate icon at the bottom right of the grid. In the center, a quantum circuit is shown with two qubits (q[0] and q[1]) and one classical register (c[2]). A red box highlights a measurement gate (indicated by a circle with a plus sign) on q[1]. Another red box highlights a Hadamard gate (H) on q[0]. An arrow labeled '(1)' points from the text '測定ゲートを追加' to the measurement gate on q[1]. On the right, the circuit's Python code is displayed in a code editor:

```
1  from qiskit import QuantumRegister, ClassicalRegister, QuantumCircuit
2  from numpy import pi
3
4  qreg_q = QuantumRegister(2, 'q')
5  creg_c = ClassicalRegister(2, 'c')
6  circuit = QuantumCircuit(qreg_q, creg_c)
    .h(qreg_q[0])
    .cx(qreg_q[0], qreg_q[1])
    .measure(qreg_q[0], creg_c[0])
    .measure(qreg_q[1], creg_c[1])
```

A red box highlights the 'Setup and run' button at the top right of the interface. An arrow labeled '(2)' points from the text 'ファイル名を変更 (Entanglementなど)' to this button. A red box labeled '(3)' points to the text 'クリック' (Click) located next to the 'Setup and run' button.

Set up and run your circuit

x

Step 1
Choose a QPU

Search by QPU name ↑

ibm_sherbrooke

QPU status Online

Total pending jobs 630

127 Qubits 2.9% EPLG 5K CLOPS

See details

ibm_brisbane

QPU status Online

Total pending jobs 862

127 Qubits 4.5% EPLG 5K CLOPS

See details

ibm_osaka

QPU status Online

Total pending jobs 918

See details

(4)

Step 2
Choose your settings

Instance
ibm-q/open/main

Shots *
1024

Job limit: 3

Tags (optional)
Add tags

(5) デバイスを選択します。

- Total pending jobsが少ない
 - EPLG(エラー)が小さい
- などから選んでみましょう。

Close

Run on ibm_sherbrooke

(6)

IBM Quantum Composer

Untitled circuit Saved File Edit View Visualizations seed 3623 Setup and run

(7) Jobの確認

Composer jobs

Search

H \oplus \ominus \otimes \otimes^{\dagger} X I
T S Z T^{\dagger} S^{\dagger} P

q[0] H \otimes^z
q[1] + \otimes^z
c2 0 1

Qiskit

Open in Quantum Lab

```
1 from qiskit import QuantumRegister, ClassicalRegister, QuantumCircuit
2 from numpy import pi
3
4 qreg_q = QuantumRegister(2, 'q')
```

View all jobs →

Untitled circuit Saved File Edit View Visualizations seed 3623 Setup and run

Operations

Search

H \oplus \ominus \otimes \otimes^{\dagger} X I
T S Z T^{\dagger} S^{\dagger} P
RZ \otimes^z $|0\rangle$ | \bullet if
 \otimes^x \otimes^y \otimes^{RY} \otimes^{RY} \otimes^{RY}

Composer jobs

Showing jobs in ibm-q/open/main

View all jobs →

Search jobs

from this file ×

Queued: Jul 28, 2024 5:59 PM ID: ctk0gpa6g3rg0086x2ng | ibm_sherbrooke

(8)

Jobs /

ctk0gpa6g3rg0086x2ng

待ち時間がある場合

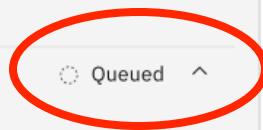
See more details

QPU name:

ibm_sherbrooke

Status timeline

Queued

 Created: Jul 28, 2024 5:59 PM

In queue

Estimated wait time: in about 2 hours, project pos: 2, QPU pos: 3

Running

Estimated usage: 40.7s

Completed

Details

Untitled circuit Saved

File

Edit

View

Visualizations seed

3623

Setup and run

Operations

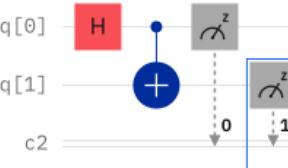
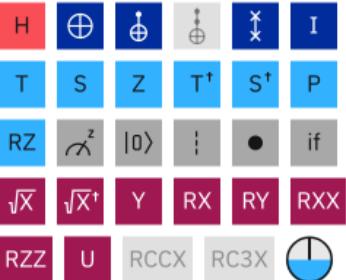
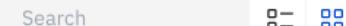


Left alignment



Inspect

Search



Statevector



Q-sphere



待ち時間がかかる場合は、少し経つてから確認します。

[See more details](#)

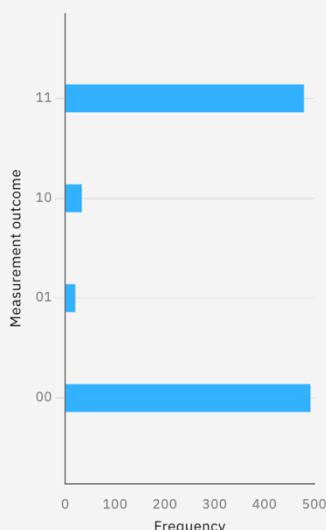
Completed
Jul 21, 2022 9:00 PM (in 38.2s)

Backend
ibmq_belem

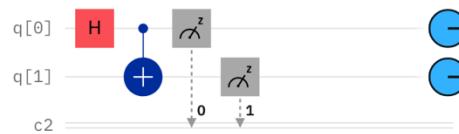
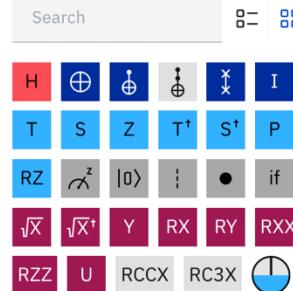
Status timeline Completed

Details

Result - histogram



Operations



Qiskit

[Open in Quantum Lab](#)

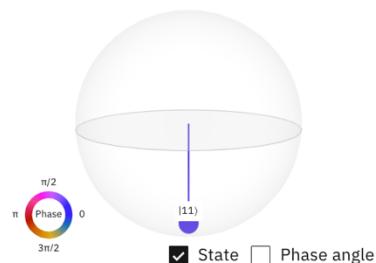
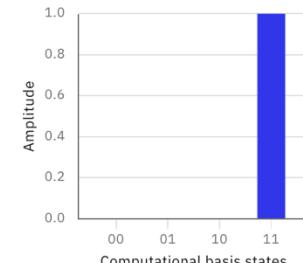
```
1 from qiskit import QuantumRegister, ClassicalRegister, QuantumCircuit
2 from numpy import pi
3
4 qreg_q = QuantumRegister(2, 'q')
5 creg_c = ClassicalRegister(2, 'c')
6 circuit = QuantumCircuit(qreg_q, creg_c)
7
8 circuit.h(qreg_q[0])
9 circuit.cx(qreg_q[0], qreg_q[1])
10 circuit.measure(qreg_q[0], creg_c[0])
11 circuit.measure(qreg_q[1], creg_c[1])
```

(9)

実機での結果

00と11以外の結果も含まれています。
ノイズによるエラーが原因です。

Statevector



まとめ

- 量子コンピューターは、量子力学の性質（重ね合わせ、もつれなど）を使った新しい計算技術です。
- IBM Quantum Composerを使って量子計算の基礎を学び、エンタングルメントの実験を実際の量子コンピューターで行いました。

次は、Qiskitを使って量子アルゴリズムの構築を行います。



ファイルをダウンロード

URL: ibm.biz/20240120

(1) 「20250120_2_qiskit.ipynb」
を選択

A screenshot of a GitHub repository page for 'kifumi / qiskit-hanson'. The repository has 1 star. It shows tabs for Code, Issues, Pull requests, Actions, and a main branch dropdown set to 'main'. Below the dropdown is a search bar with 'qiskit-hanson / 20250120 /'. A sidebar on the left lists 'kifumi rename files'. The main area shows a list of files: '20250120_2_qiskit.ipynb' (selected), '20250120_3_qml.ipynb', and '20250120_4_qiskit.ipynb'. A red arrow points to the '20250120_2_qiskit.ipynb' file.

(2) 右上の アイコンからファイル
をローカルにダウンロード

A screenshot of a GitHub file preview for '20250120_2_qiskit.ipynb'. The file was last updated on '9a45b1e · last week'. The preview shows the first few lines of the notebook. Below the preview is a toolbar with icons for Raw, Copy, Download (highlighted with a red arrow), and Edit. A red arrow also points to the download icon in the toolbar.

(3) ブラウザーで1つ前のページに戻って
Google ColabまたはqBraidの準備をする。

Google Colab または qBraid Lab を使ってQiskitを実行

(1) Google コラボにログイン

<https://colab.research.google.com/>

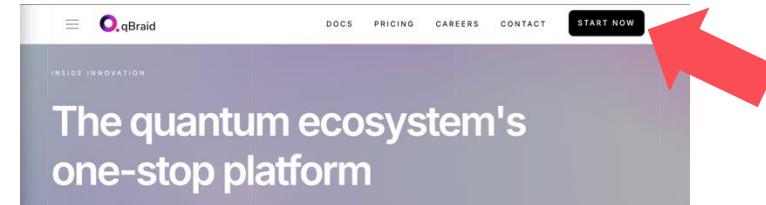


(2) 「ファイル」→「ノートブックをアップロード」



(1) qBraidにログイン

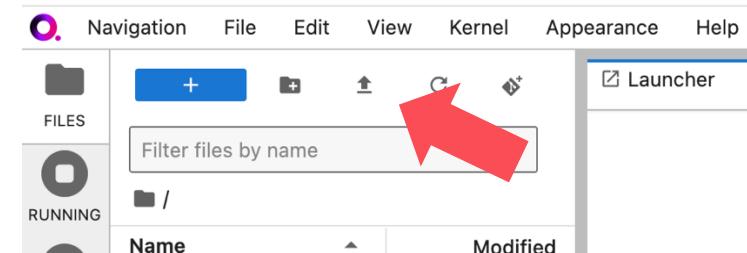
<https://www.qbraid.com/>



(2) 手順に従って環境作成

URL: ibm.biz/qbraidja

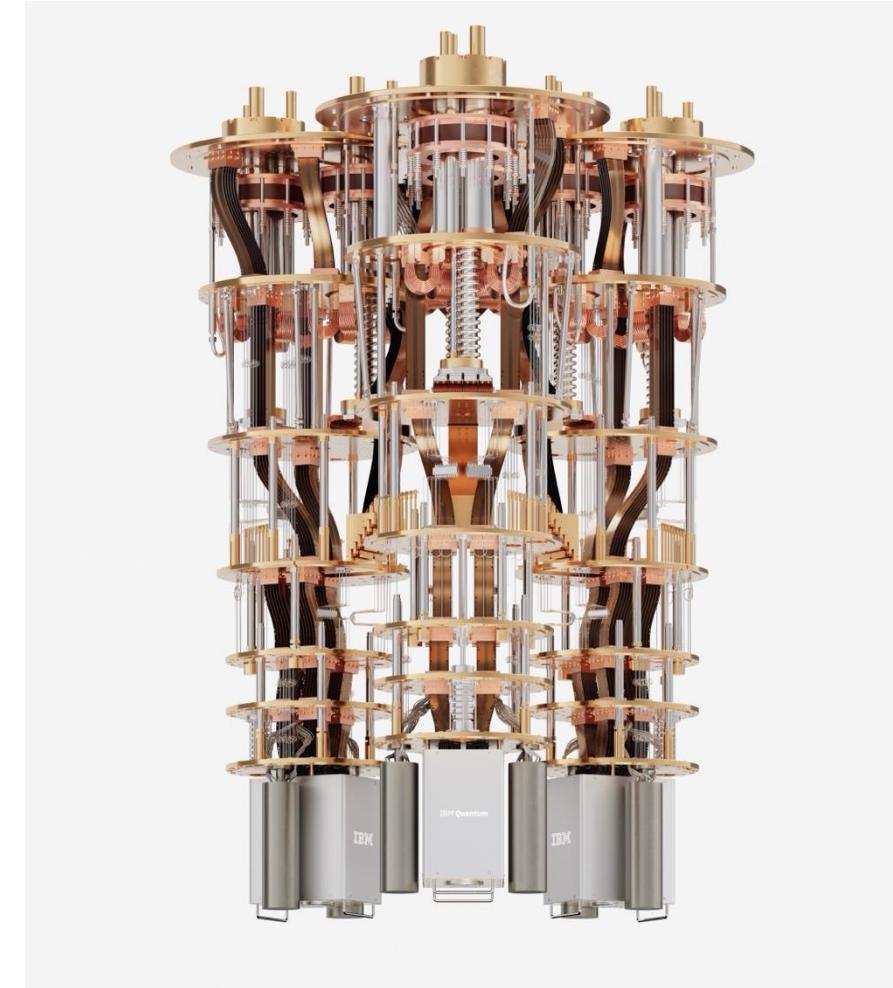
(3) 左上の マークからファイルをアップロード



Qiskit入門

Jan 20, 2025

沼田祈史
Kifumi Numata
IBM Quantum



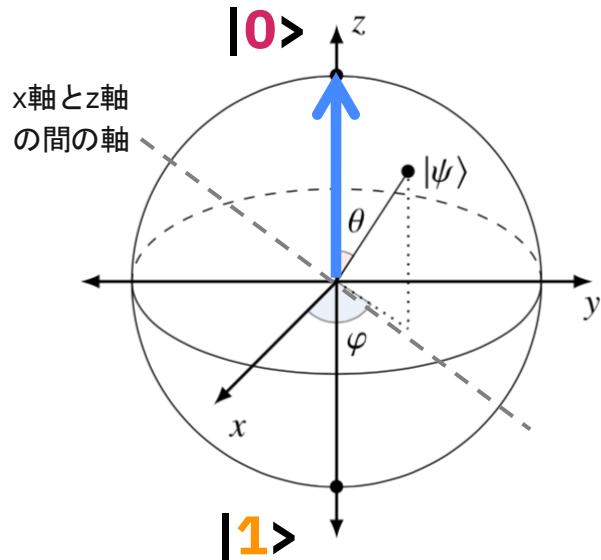
量子ビットと量子ゲート

	量子	ベクトル・行列表示
量子ビット	0と1の重ね合わせ $\alpha \times 0\rangle + \beta \times 1\rangle$	$ 0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, 1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$ $ \psi\rangle = \alpha 0\rangle + \beta 1\rangle = \begin{pmatrix} \alpha \\ \beta \end{pmatrix}$ $\alpha, \beta : \text{確率振幅}$ $ \alpha ^2 + \beta ^2 = 1$ 状態ベクトル
論理ゲート	X, H, CNOT など 	$\begin{pmatrix} u_{11} & u_{12} \\ u_{21} & u_{22} \end{pmatrix} \begin{pmatrix} \alpha \\ \beta \end{pmatrix} = \begin{pmatrix} \alpha' \\ \beta' \end{pmatrix}$ 量子ゲート 入力ビット 出力ビット

例) Xゲートの場合

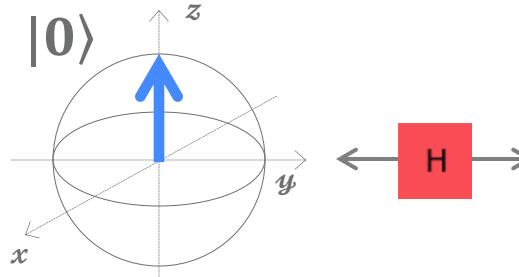
$$X|0\rangle = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \cdot 1 + 1 \cdot 0 \\ 1 \cdot 1 + 0 \cdot 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix} = |1\rangle$$

ブロツホ球

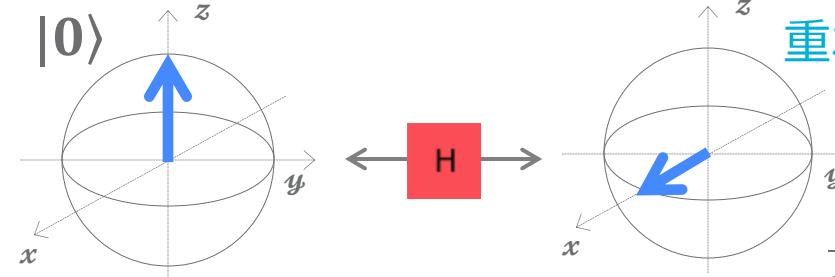


半径1の球面上の点をさす
ベクトルが量子ビット

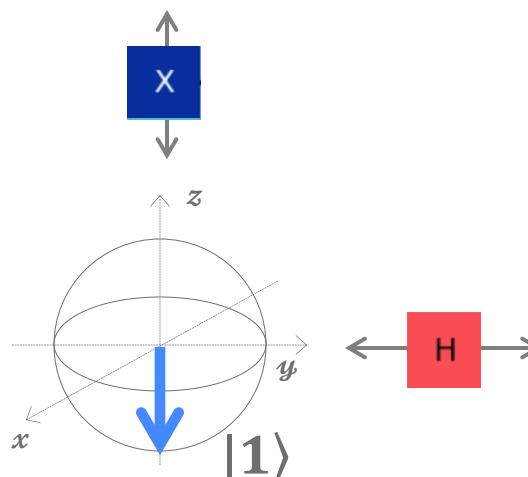
北極 : 0の確率が100%



赤道 : 0と1が50%ずつの
重ね合わせ状態



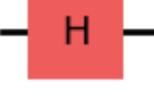
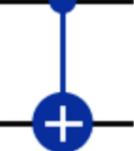
$$\frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle$$



$$\frac{1}{\sqrt{2}}|0\rangle - \frac{1}{\sqrt{2}}|1\rangle$$

南極 : 1の確率が100%

代表的な量子ゲート一覧

	行列表現	Qiskitのコード	回路表示	
Xゲート	$\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$	x		ビット反転
Yゲート	$\begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}$	y		位相・ビット反転
Zゲート	$\begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$	z		位相反転
Hゲート	$\frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$	h		重ね合わせを作る
CNOTゲート	$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$	cx		量子もつれを作る

量子テレポーテーション

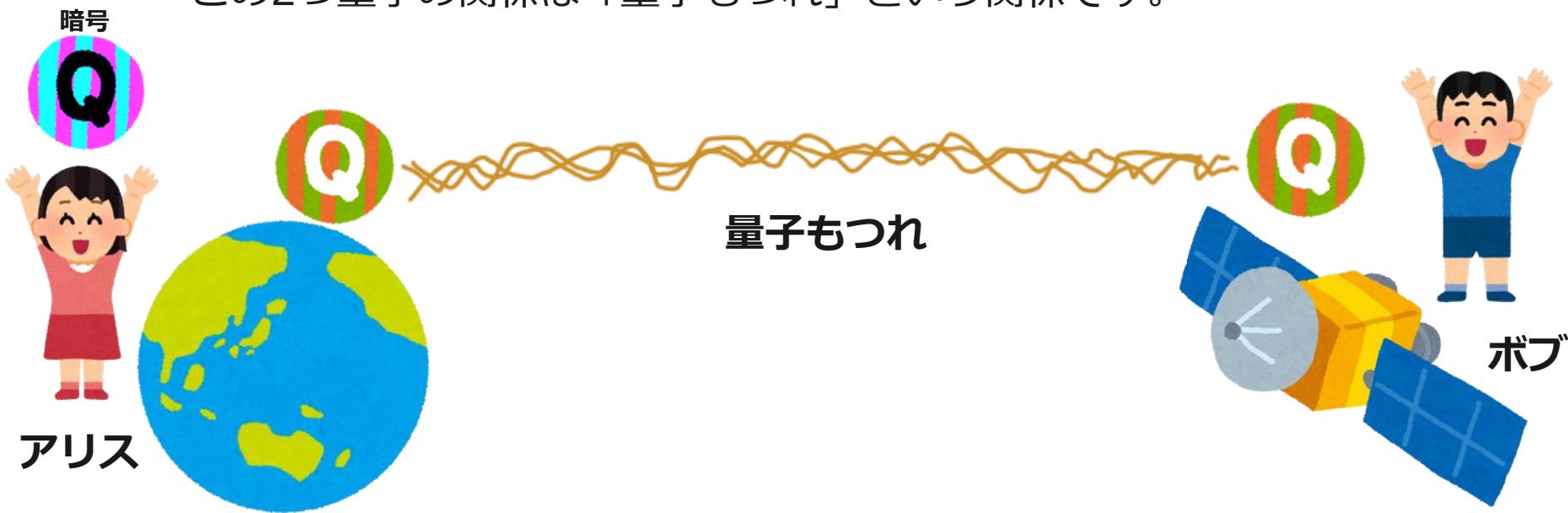
Aliceが**未知の量子状態** $|\psi\rangle$ を遠く離れたBobに送りたいけれど、通信手段が**古典的な手法**（メールや電話など）でしか連絡が取れない場合を考えます。



AliceとBobが事前に**エンタングルした量子ビット**のペアを持っていれば、古典情報を2ビットBobに送ることで、Bobは、完全な $|\psi\rangle$ を手に入れることができます。

量子テレポーテーションのプロトコル

- (1) 地球のアリスがある量子  (暗号) を持っています。
- (2) 特別な関係にあるふたごの量子  が地球と人工衛星の上にあります。この2つ量子の関係は「量子もつれ」という関係です。



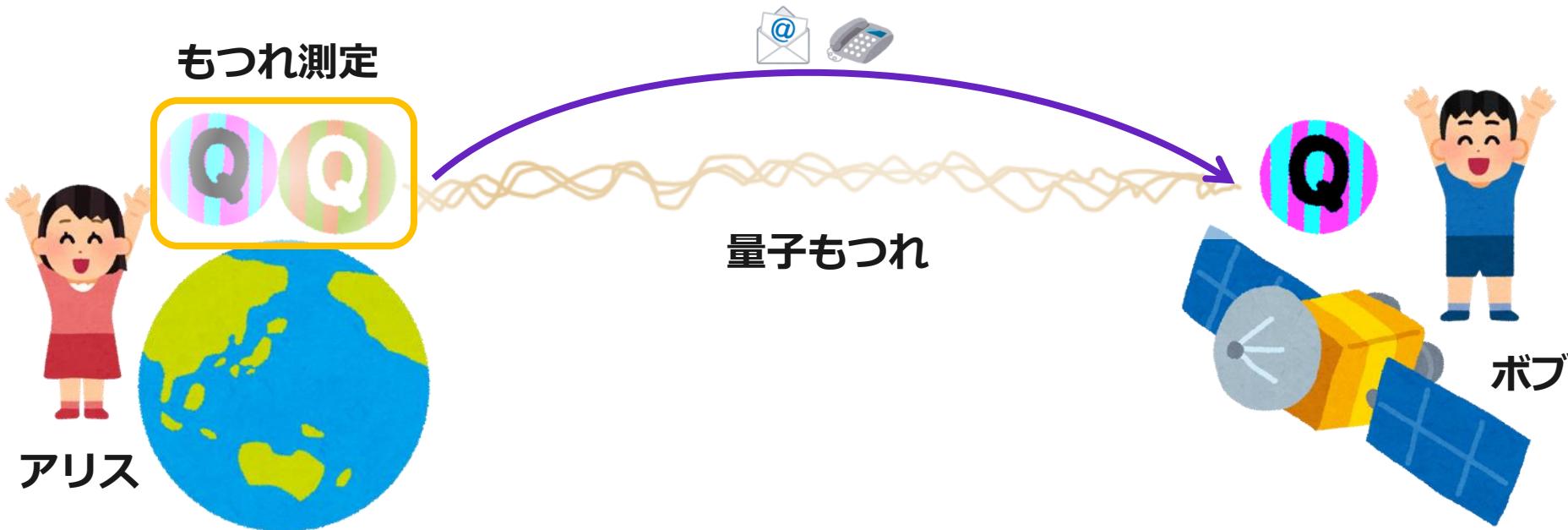
量子テレポーテーションのプロトコル

(3) 地球のアリスが地上の2つの量子に特殊な測定（もつれ測定）をします。
(量子もつれ状態にあるボブの量子の状態が瞬時に変わります。)



量子テレポーテーションのプロトコル

(4) アリスが測定結果をメールや電話でボブに送り、
ボブはもらった結果をもとに自分の量子を補正します。
ボブの量子がアリスの持っていた暗号に変化します！



(補足) 量子テレポーテーションアルゴリズムの詳細

Qiskitではビットの並びが $|q_2\ q_1\ q_0\rangle$ です

$$|\psi_0\rangle = |00\rangle \otimes (\alpha|0\rangle + \beta|1\rangle)$$

Aliceの持っている暗号

$$\begin{aligned} |\psi_1\rangle &= \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle) \otimes (\alpha|0\rangle + \beta|1\rangle) \\ &= \frac{1}{\sqrt{2}}(\alpha|000\rangle + \alpha|110\rangle + \beta|001\rangle + \beta|111\rangle) \end{aligned}$$

エンタングルメント

$$\begin{aligned} |\psi_2\rangle &= \frac{1}{\sqrt{2}}(\alpha|000\rangle + \alpha|110\rangle + \beta|011\rangle + \beta|101\rangle) \quad q_0が1の時のみq_1にXを操作 \\ &= \frac{1}{\sqrt{2}}(\alpha(|00\rangle + |11\rangle)|0\rangle + \beta(|01\rangle + |10\rangle)|1\rangle) \quad \alphaと\betaでまとめる \end{aligned}$$

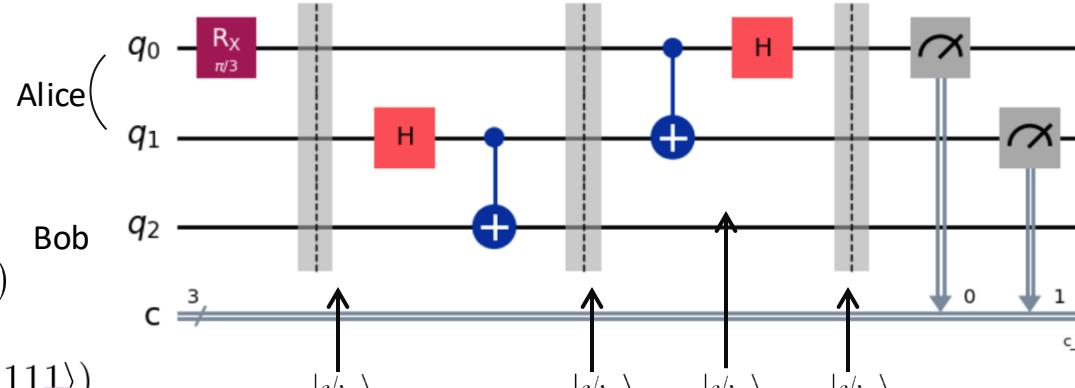
$$\begin{aligned} |\psi_3\rangle &= \frac{1}{2}(\alpha(|00\rangle + |11\rangle)(|0\rangle + |1\rangle) + \beta(|01\rangle + |10\rangle)(|0\rangle - |1\rangle)) \quad q_0にHを操作 \\ &= \frac{1}{2}((\alpha|0\rangle + \beta|1\rangle)|00\rangle + (\alpha|1\rangle + \beta|0\rangle)|10\rangle + (\alpha|0\rangle - \beta|1\rangle)|01\rangle + (\alpha|1\rangle - \beta|0\rangle)|11\rangle) \end{aligned}$$

q2にそのまま暗号が現れている

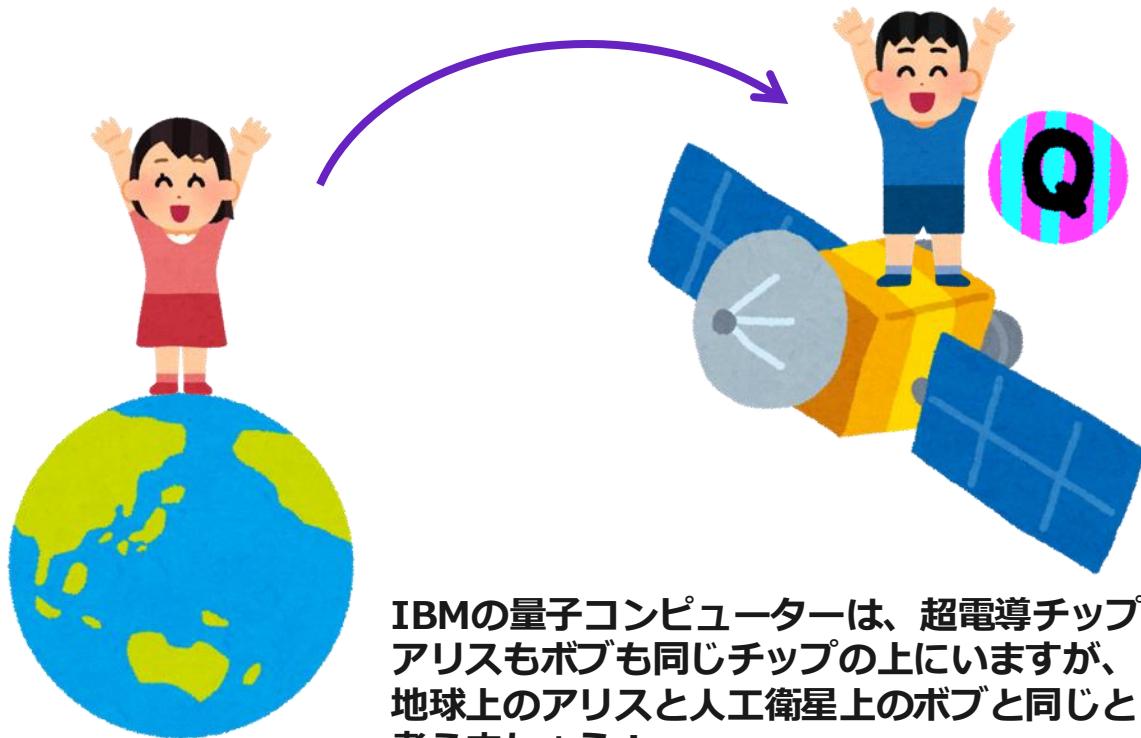
q1が1の時はq2にXゲートをかける

q0が1の時はq2にZゲートをかける

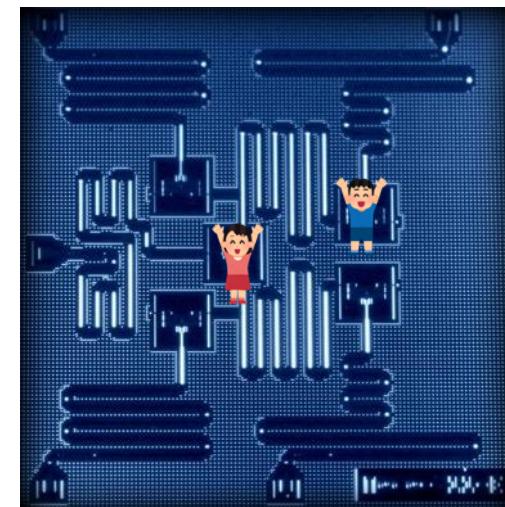
q0とq1が1の時はq2にXゲートとZゲートをかける



アリスからボブに暗号（量子状態）を送ります



IBMの量子コンピューターは、超電導チップなので、
アリスもボブも同じチップの上にいますが、
地球上のアリスと人工衛星上のボブと同じと
考えましょう！



まとめ

- Qiskitで、量子回路を作り、実機の量子コンピューターで計算を行う方法を学びました。
- 量子テレポーテーション回路をコーディングし、ダイナミックサークットを使って実行しました。

次は、量子機械学習です。

