A Brief Tutorial on Qiskit

Department of Physics
National Taiwan University

Installation

- Step 1. Install Anaconda
- Step 2. Open a **terminal** window in the directory where you want to work.
- Step 3. Create a new environment for running QISKit

```
conda create -n env_name python=3.7 jupyter
```

```
conda activate env_name
```

https://qiskit.org/documentation/install.html

Installation

- Step 4. Install QISKit, matplotlib

```
pip install qiskit
pip install matplotlib
```

- Step 5. Open Jupyter notebook

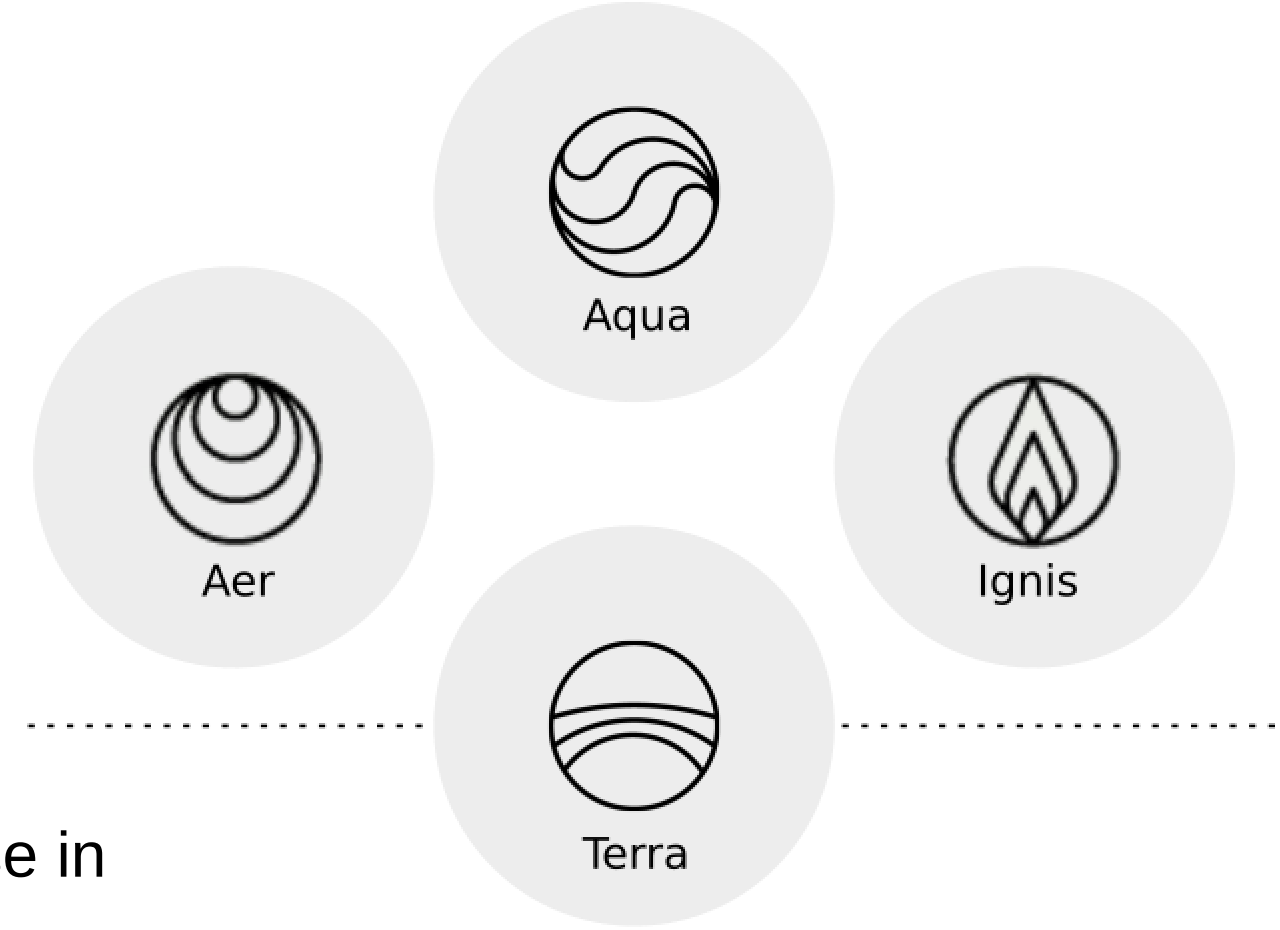
```
jupyter notebook
```

- Step 6. Get started!

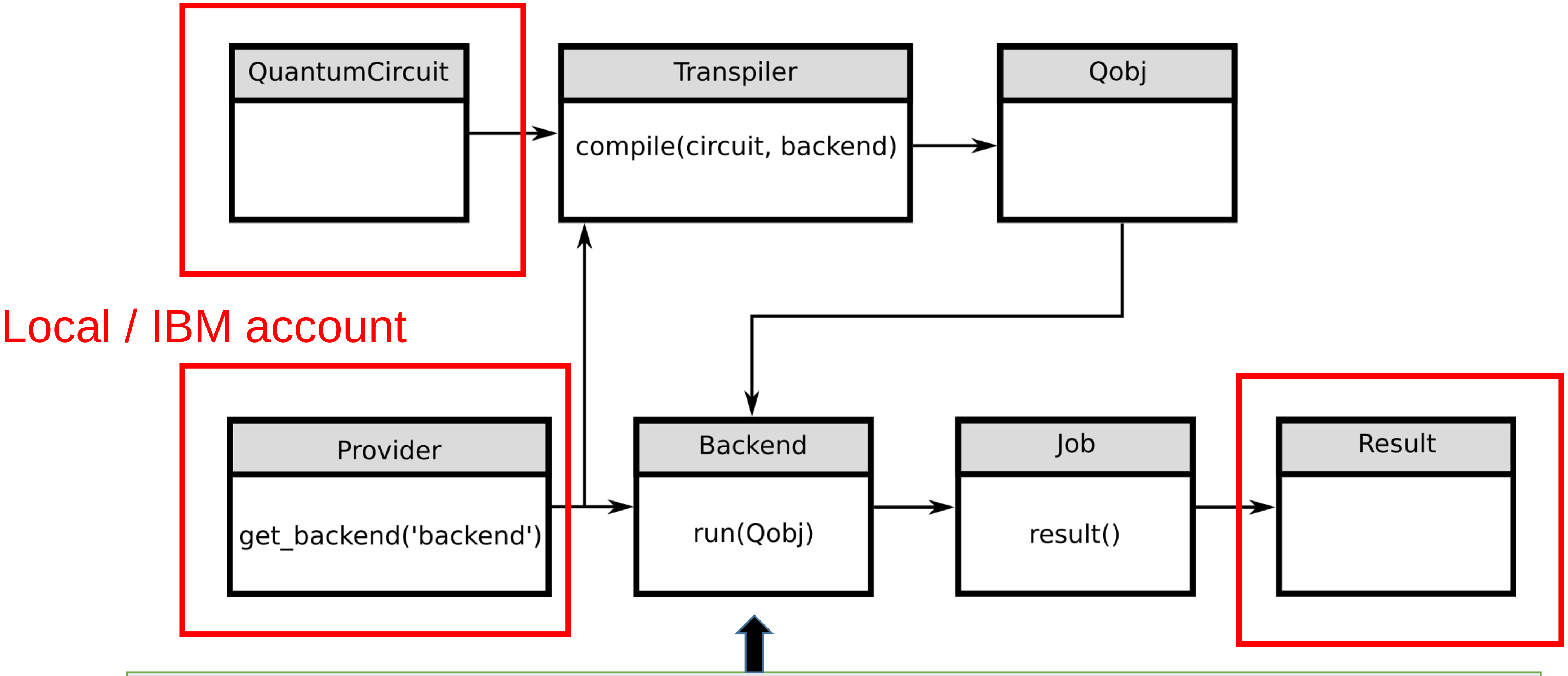
For more details, please refer to:
https://qiskit.org/documentation/install.html

Installation

- **Qiskit Terra**
  A solid foundation for quantum computing
- **Qiskit Aqua**
  Algorithms for near-term quantum applications
- **Qiskit Aer**
  A high performance simulator framework for quantum circuits
- **Qiskit Ignis**
  Understanding and mitigating noise in quantum devices



Executing Quantum Programs



A **backend** represents either a **simulator** or a **real quantum computer**.

Executing Quantum Programs

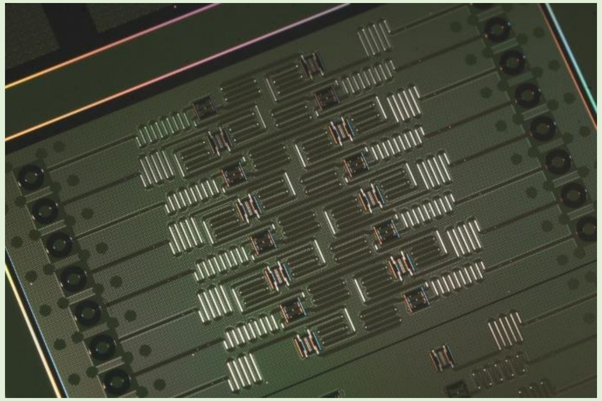
A **backend** represents either a **simulator** or a **real quantum computer.**

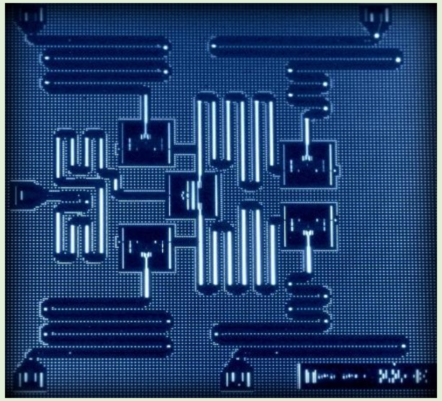
1. Build your circuit

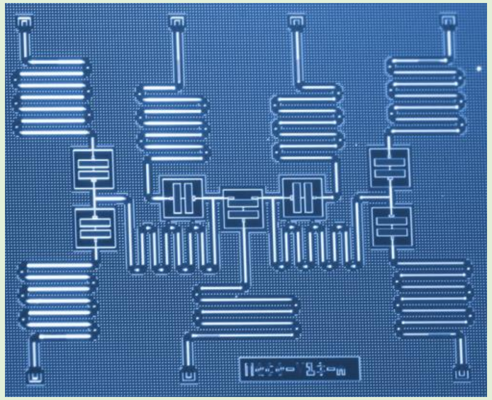
2. Choose your **backend**

3. Execute your circuit on your backend, returning a **job** object

4. Access the **result** from the job object via `job.result()`







Example

- Step 1 : Import Packages

```
import numpy as np
from qiskit import( QuantumRegister,
ClassicalRegister, QuantumCircuit, execute, Aer)
from qiskit.visualization import plot_histogram
```

https://qiskit.org/documentation/getting_started.html#workflow-step-by-step

Example

- Step 2 : Initialize Variables

```
q = QuantumRegister(5, 'q')
c = ClassicalRegister(2, 'c')
circ = QuantumCircuit(q, c)
```

- Step 3 : Add Gates

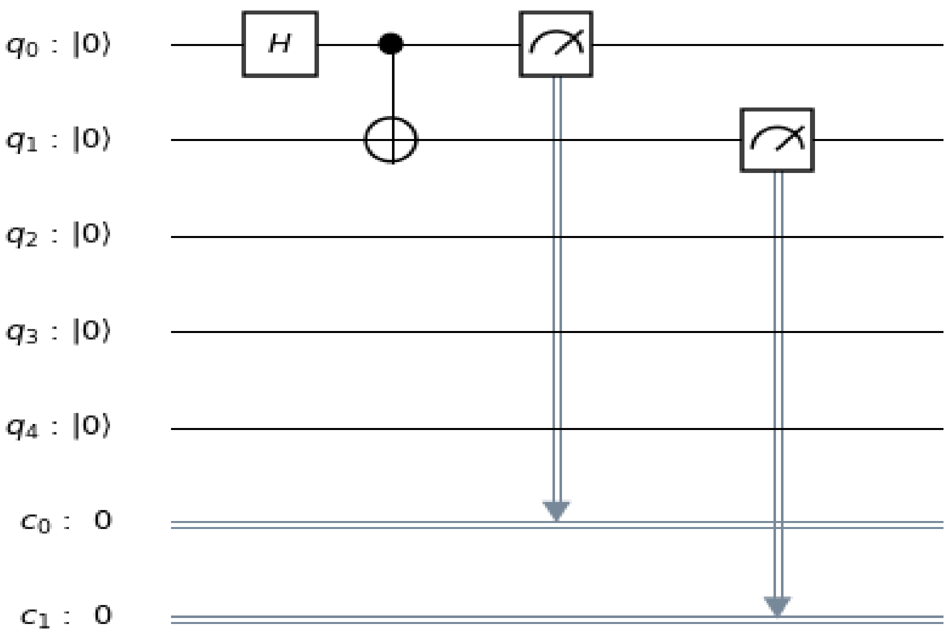
```
circ.h(q[0])
circ.cx(q[0],q[1])
circ.measure(q[0], c[0])
circ.measure(q[1], c[1])
```

https://qiskit.org/documentation/getting_started.html#workflow-step-by-step

Example

- Step 4 : Visualize the Circuit

```
circ.draw(output='mpl')
```



- Step 5 : Simulate the Experiment

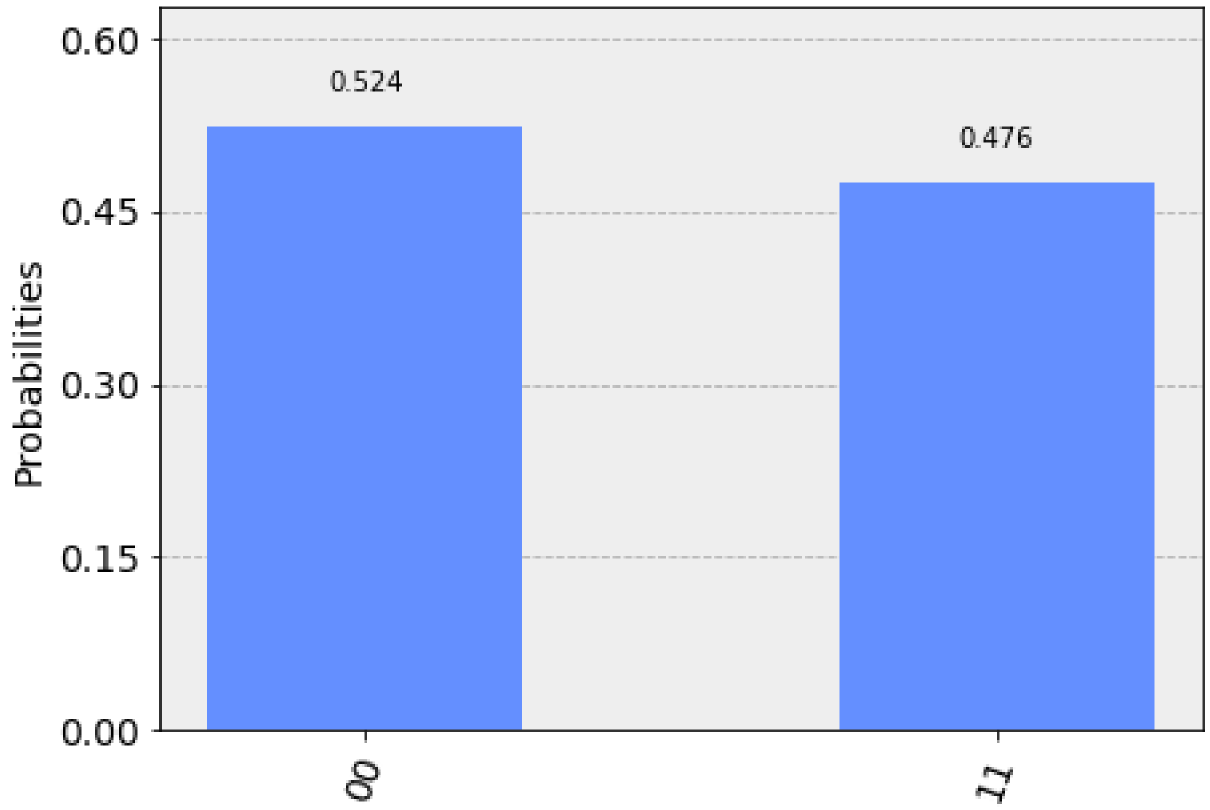
```
simulator = Aer.get_backend('qasm_simulator')
job = execute(circ, simulator, shots=1000)
result = job.result()
counts = result.get_counts(circ)
print("\nTotal count for 0 and 11 are:"
counts)
```

Total count for 0 and 11 are: {'11': 476, '00': 524}

Example

- Step 6 : Visualize the Results

```
plot_histogram(counts)
```



https://qiskit.org/documentation/getting_started.html#workflow-step-by-step

Elementary Operations

https://qiskit.org/documentation/terra/summary_of_quantum_operations.html

- Single-Qubit Gates
  - u3 gates
  - Identity gate
  - Pauli gates
    - X
    - Y
    - Z
  - Clifford gates
    - H
    - S
    - T
  - Standard Rotations
- Two-qubit gates
  - Controlled Pauli Gates
  - Controlled Hadamard gate
  - Controlled rotation gates
  - Controlled phase rotation
  - Controlled u3 rotation
  - SWAP gate
- Three-qubit gates
  - Toffoli gate
  - Fredkin Gate
- Measurement
- Conditional operations

Simulating Circuits with Qiskit Aer

- Statevector Simulator

  This backend returns the quantum state which is the output of the quantum circuit.

- Unitary Simulator

  This backend calculates the matrix representing of the gates in the quantum circuit.

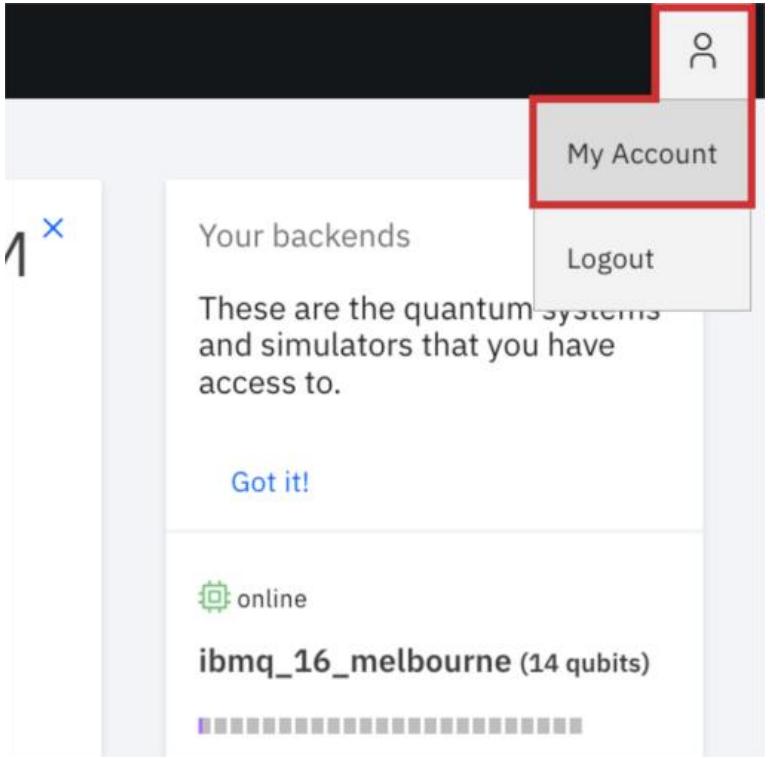
- OpenQASM Simulator

  This backend simulates the quantum circuit with measurements.

https://qiskit.org/documentation/terra/executing_quantum_programs.html

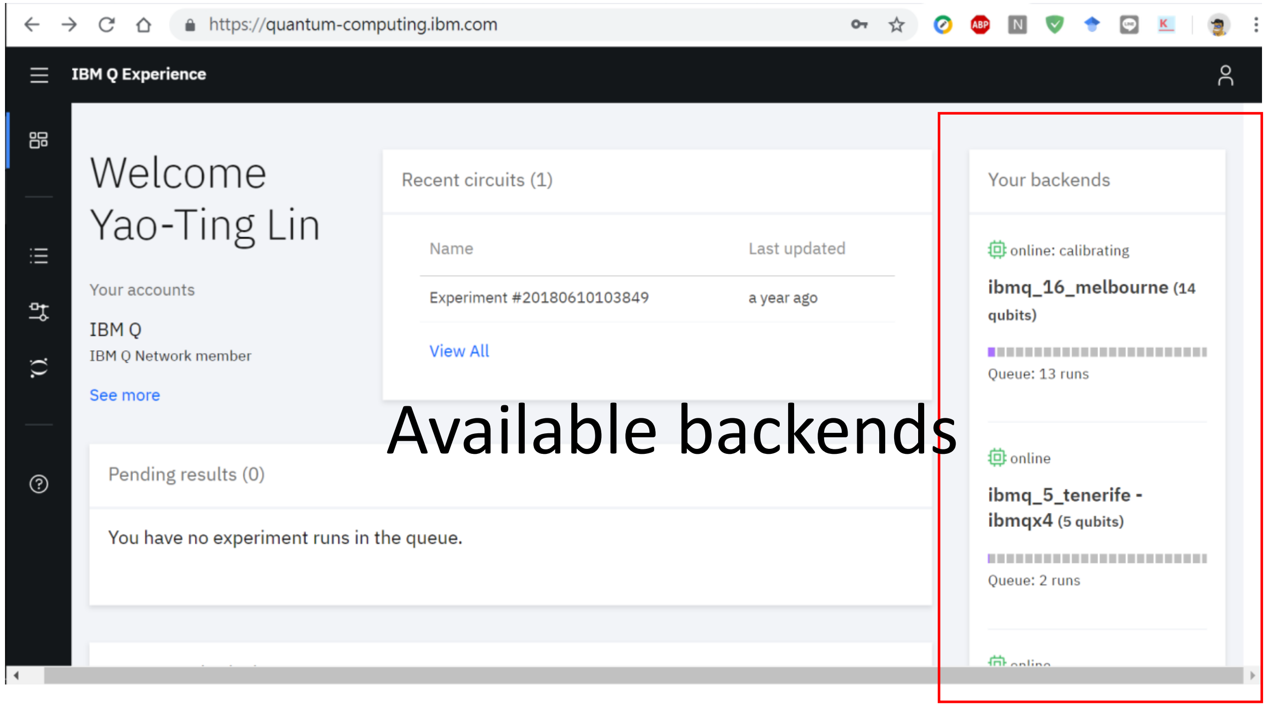
Running Circuits on IBM Q Devices

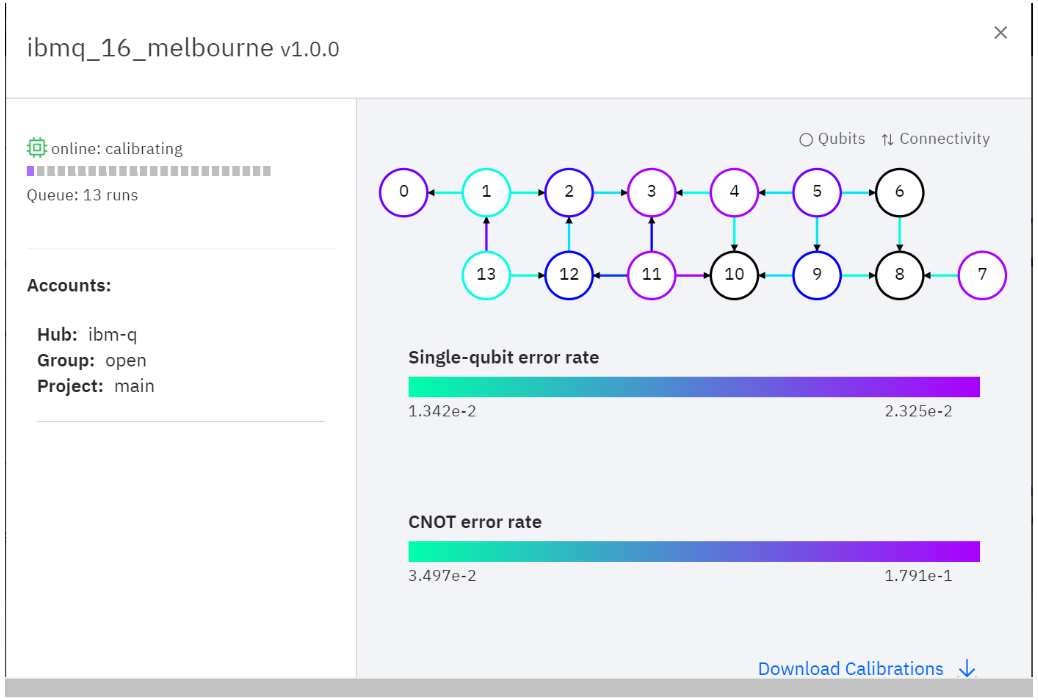
- Step 1. Create a free IBM Q Experience account.
- Step 2. Navigate to **My Account** to view your account settings.



Running Circuits on IBM Q Devices

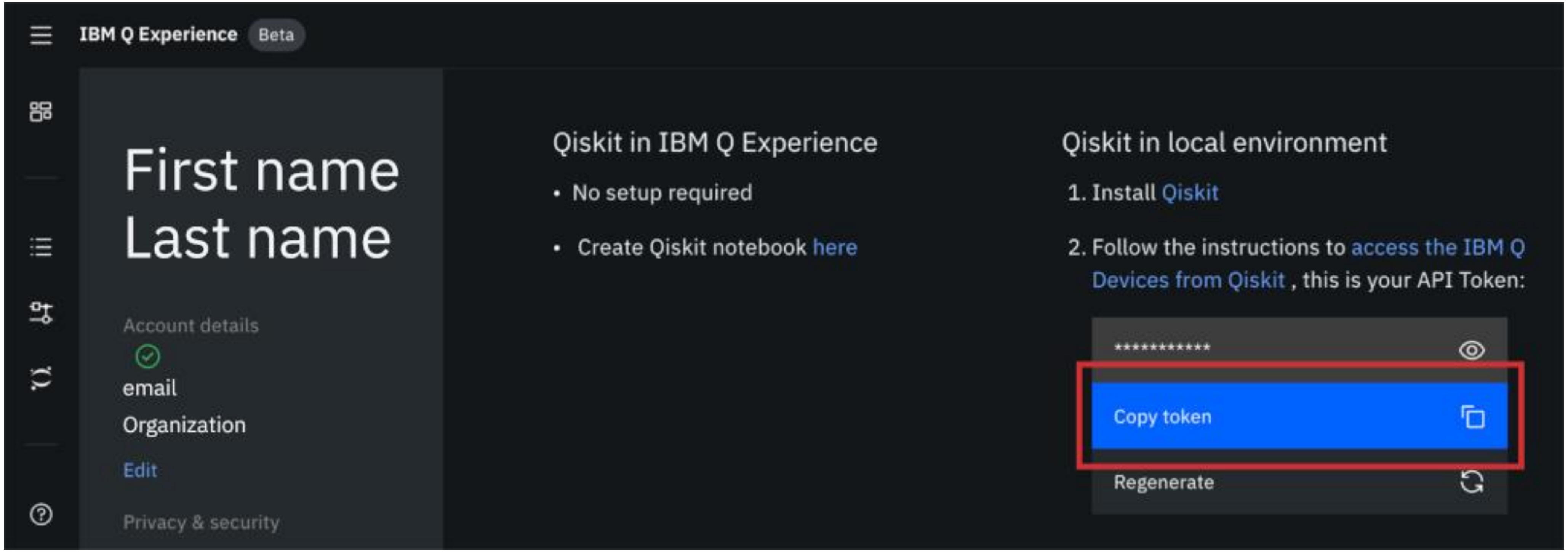
- Step 1. Create a free IBM Q Experience account.
- Step 2. Navigate to **My Account** to view your account settings.





Running Circuits on IBM Q Devices

- **Step 3:** Click on **Copy token** to copy the token to your clipboard. Temporarily paste this API token into your favorite text editor so you can use it later to create an account configuration file.



Running Circuits on IBM Q Devices

- **Step 4**: Run the following commands to store your API token locally for later use in a configuration file called qiskitrc. Replace MY_API_TOKEN with the API token value that you stored in your text editor.

```
from qiskit import IBMQ
IBMQ.save_account('MY_API_TOKEN')
```

For more details, please refer to:

https://qiskit.org/documentation/advanced_use_of_ibm_q_devices.html#advanced-use-of-ibm-q-devices-label

Demo: Construct entanglement

```
import qiskit
from qiskit import IBMQ
TOKEN =
'06d9f357cba32e165a02f96104e72a82b5281ee24d4adee817e18d82655980b08
551d229cbf613527275f5e444756b08a3dfbfb7fae72ce27838f444fd03'
IBMQ.save_account(TOKEN)
```

```
# Load account from disk
provider = IBMQ.load_account()
# List the account currently in the session
print(IBMQ.active_account())
# List all available providers
print(IBMQ.providers())
# List all available providers
print(provider.backends())
```

Demo: Construct entanglement

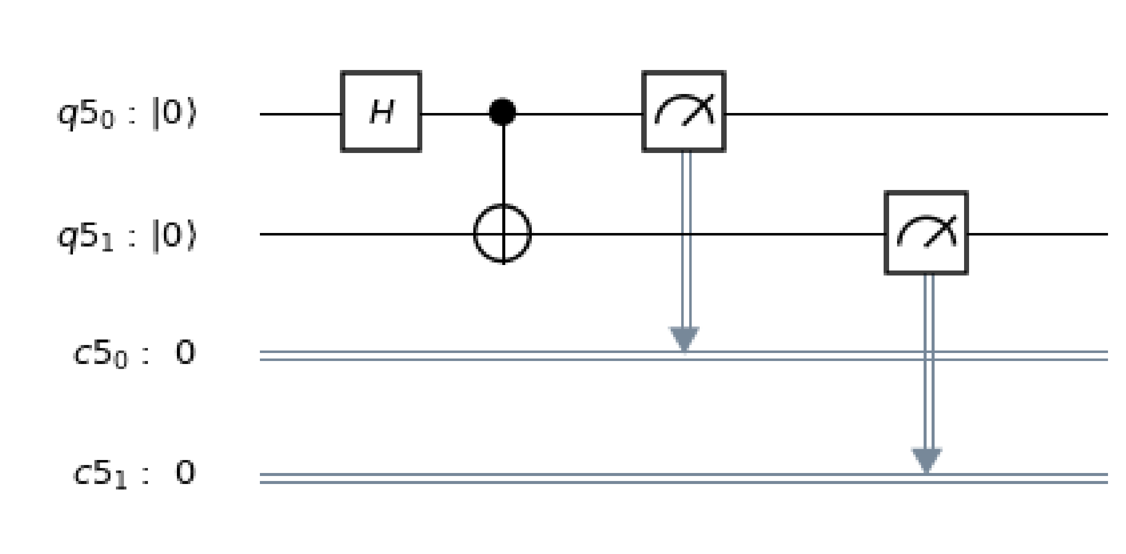
```python
from qiskit import QuantumRegister, ClassicalRegister
from qiskit import QuantumCircuit, Aer, execute

q = QuantumRegister(2)
c = ClassicalRegister(2)
qc = QuantumCircuit(q, c)

qc.h(q[0])
qc.cx(q[0], q[1])

qc.measure(q[0],c[0])
qc.measure(q[1],c[1])

qc.draw(output='mpl')
```



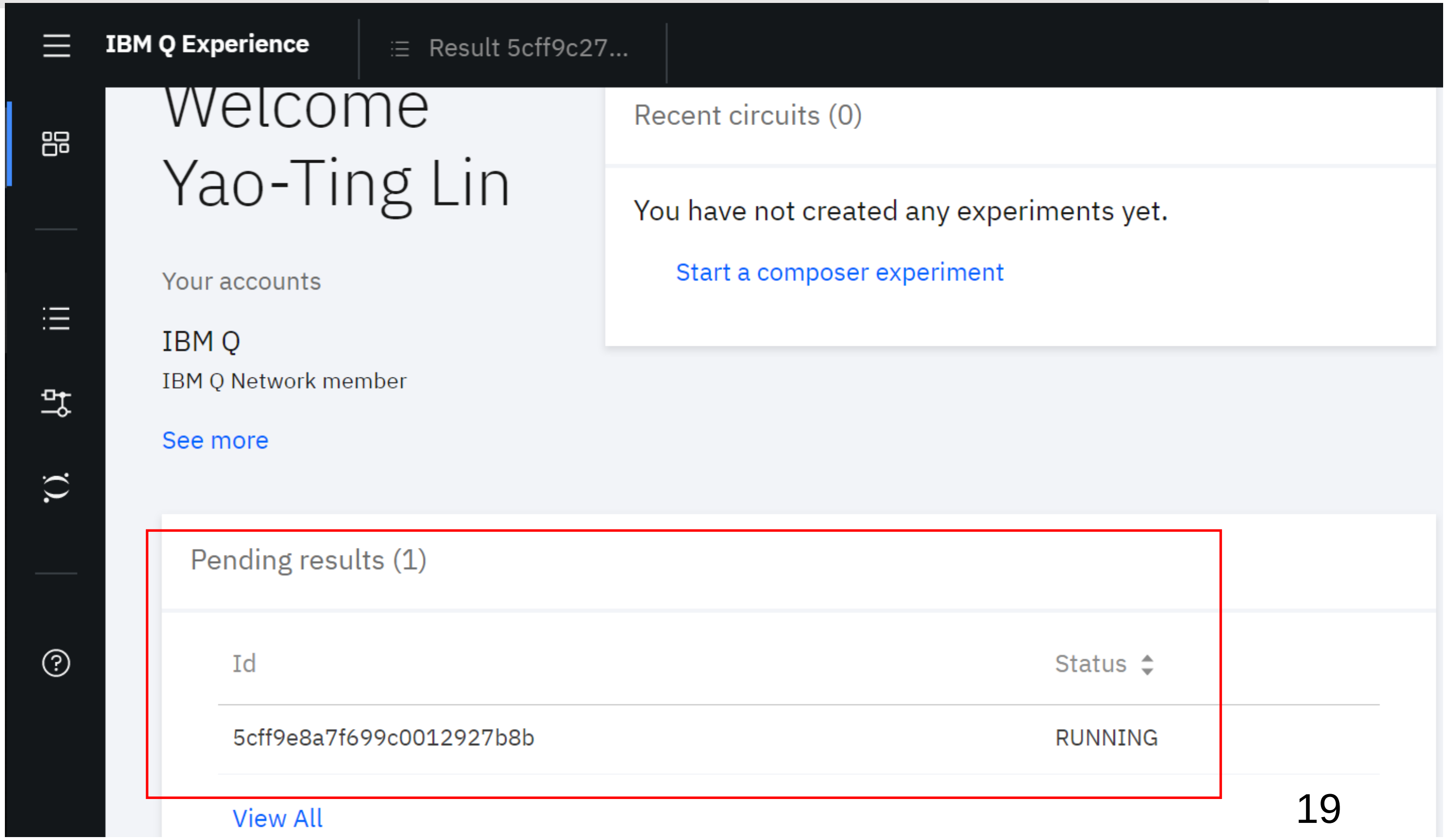
Bell state: $|\phi^+\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$

Demo: Construct entanglement

```
job = qiskit.execute(qc, provider.get_backend('ibmqx4'))
result = job.result()
```

```
print(job.job_id())
```

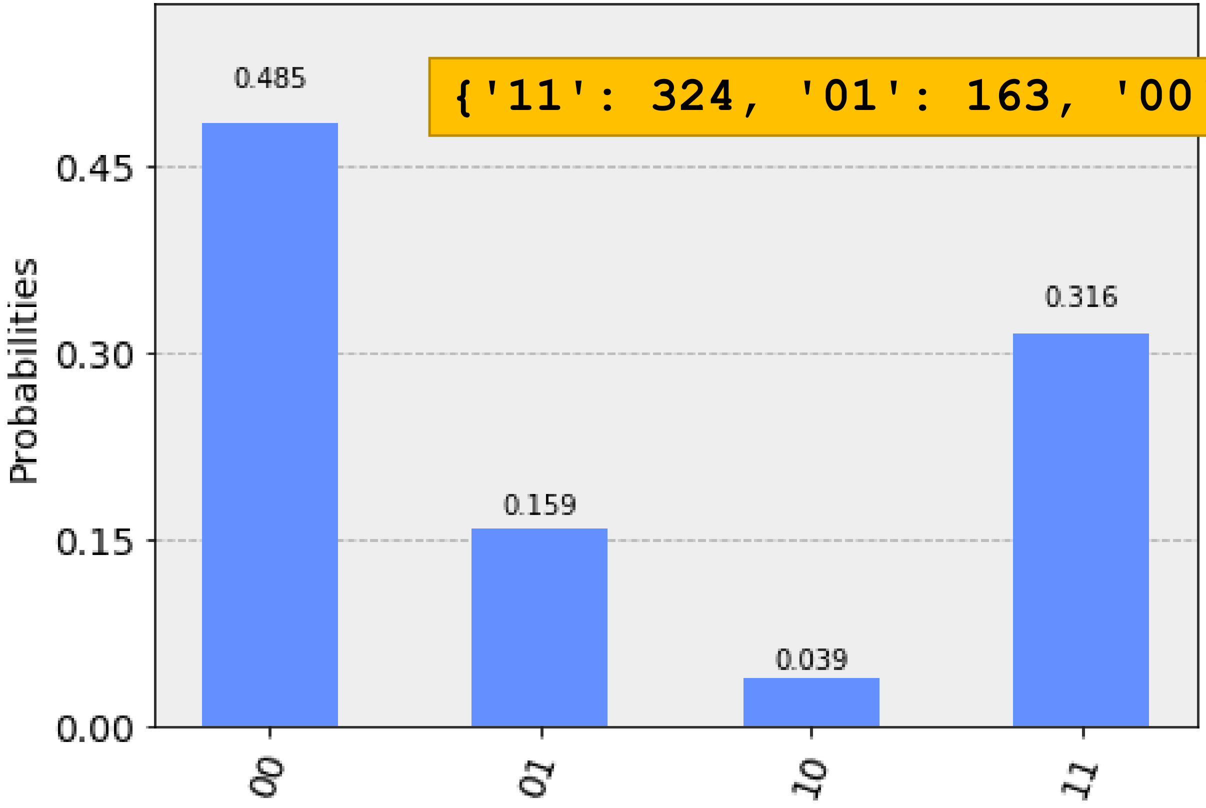
5cff9e8a7f699c0012927b8b

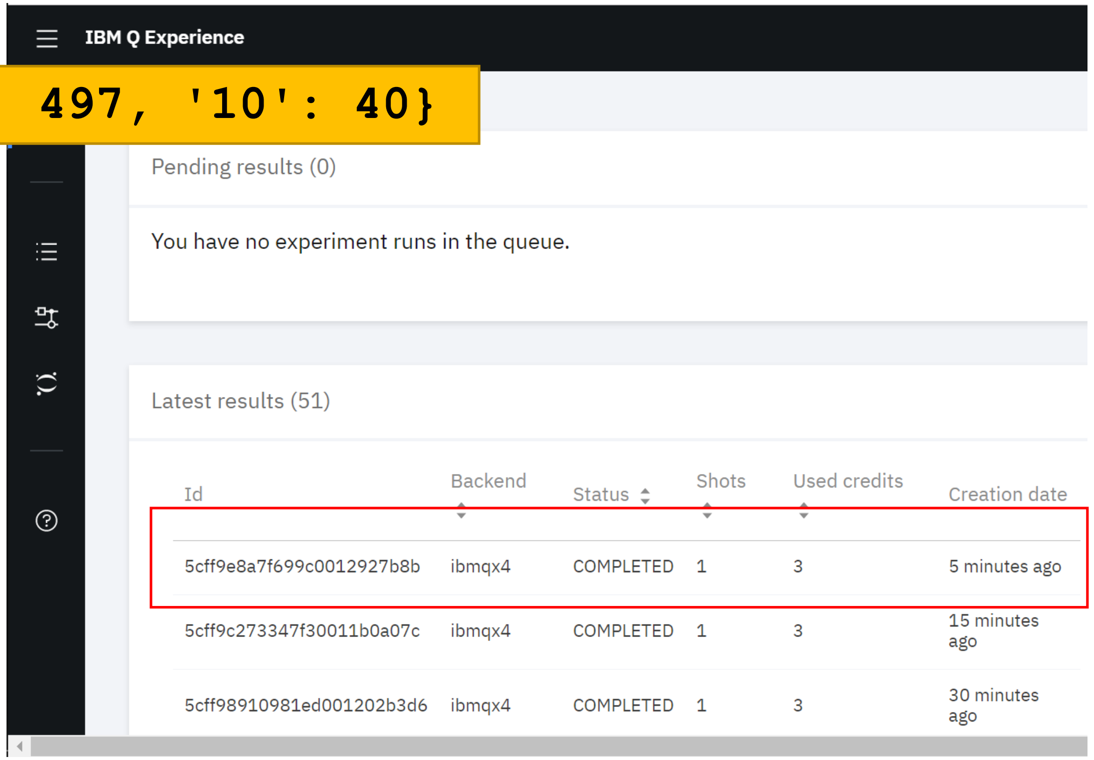


Demo

```python
from qiskit.visualization import plot_histogram
print(result.get_counts())
plot_histogram(result.get_counts())
```

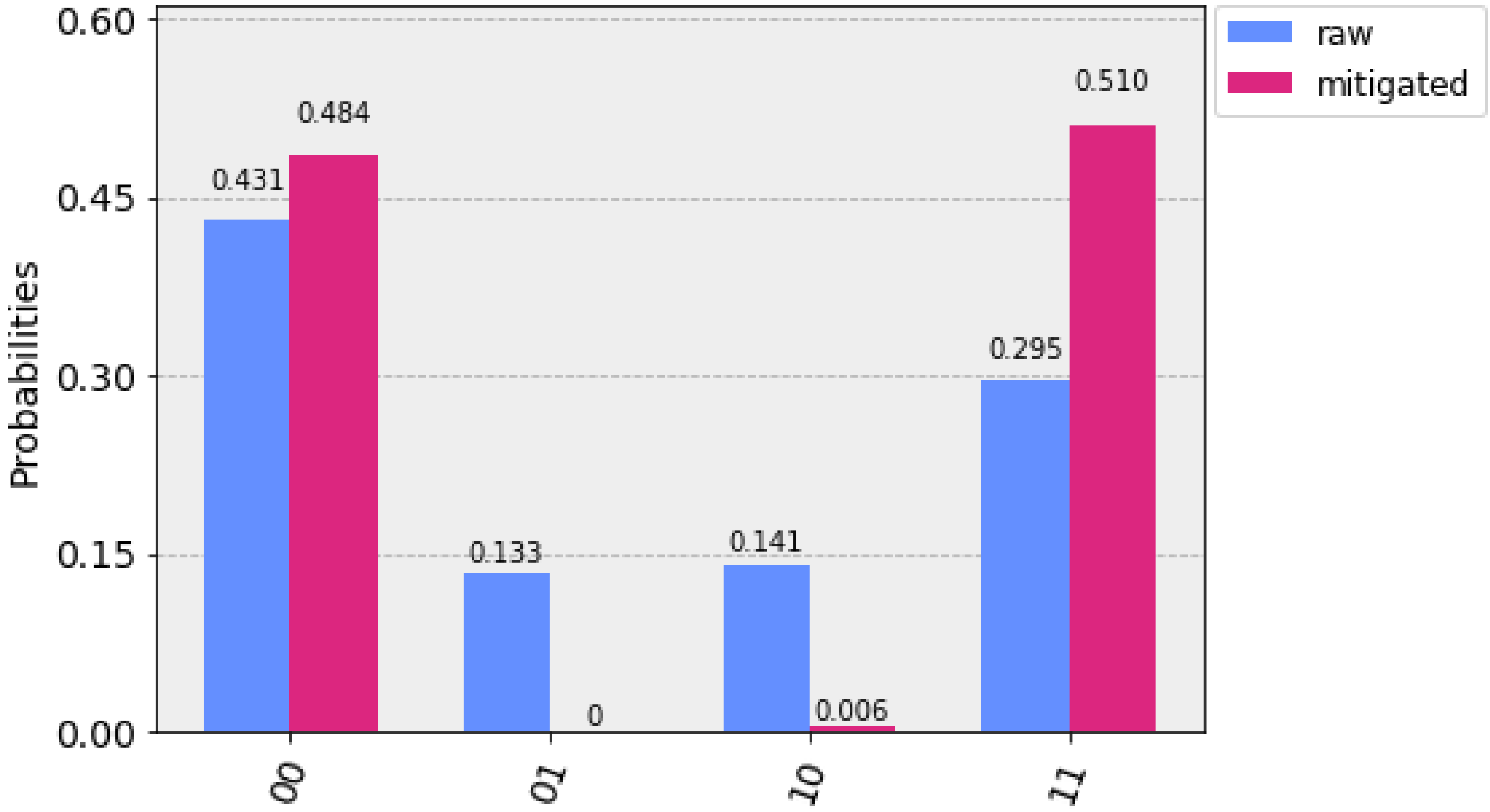
```
{'11': 324, '01': 163, '00': 497, '10': 40}
```





Demo

Error mitigation will be introduced in tomorrow’s lecture.



臺灣大學-IBM量子電腦中心

- NTU-IBM Q 開放申請　http://quantum.ntu.edu.tw/
- 2019年3月28日
  本中心即日起開放國內各大學及學術研究機構之研究人員申請使用IBM Q 20-qubit量子計算系統，

- 請詳閱「 臺灣大學 IBM Q System 使用須知及簽署 」，
- 並上傳簽署及 申請書 至本中心審核
  (Email： ntuq2018@gmail.com)，
- 審核通過者始得使用IBM Q System。

Thanks for your attention!