T3 - Exercise Deutsch-Jozsa

December 11, 2024

1 Deutsch-Josza Algorithm

[1]:
```
%pip install qiskit[visualization]
```

```
Requirement already satisfied: qiskit[visualization] in
/opt/.qbraid/environments/qbraid_000000/pyenv/lib/python3.11/site-packages
(1.3.0)
Requirement already satisfied: rustworkx>=0.15.0 in
/opt/.qbraid/environments/qbraid_000000/pyenv/lib/python3.11/site-packages (from
qiskit[visualization]) (0.15.1)
Requirement already satisfied: numpy<3,>=1.17 in /opt/conda/lib/python3.11/site-
packages (from qiskit[visualization]) (1.26.4)
Requirement already satisfied: scipy>=1.5 in
/opt/.qbraid/environments/qbraid_000000/pyenv/lib/python3.11/site-packages (from
qiskit[visualization]) (1.14.1)
Requirement already satisfied: sympy>=1.3 in
/opt/.qbraid/environments/qbraid_000000/pyenv/lib/python3.11/site-packages (from
qiskit[visualization]) (1.13.3)
Requirement already satisfied: dill>=0.3 in
/opt/.qbraid/environments/qbraid_000000/pyenv/lib/python3.11/site-packages (from
qiskit[visualization]) (0.3.9)
Requirement already satisfied: python-dateutil>=2.8.0 in
/opt/conda/lib/python3.11/site-packages (from qiskit[visualization]) (2.9.0)
Requirement already satisfied: stevedore>=3.0.0 in
/opt/.qbraid/environments/qbraid_000000/pyenv/lib/python3.11/site-packages (from
qiskit[visualization]) (5.4.0)
Requirement already satisfied: typing-extensions in
/opt/conda/lib/python3.11/site-packages (from qiskit[visualization]) (4.12.2)
Requirement already satisfied: symengine<0.14,>=0.11 in
/opt/.qbraid/environments/qbraid_000000/pyenv/lib/python3.11/site-packages (from
qiskit[visualization]) (0.13.0)
Requirement already satisfied: matplotlib>=3.3 in
/opt/conda/lib/python3.11/site-packages (from qiskit[visualization]) (3.9.3)
Requirement already satisfied: pydot in
/opt/.qbraid/environments/qbraid_000000/pyenv/lib/python3.11/site-packages (from
qiskit[visualization]) (3.0.3)
Requirement already satisfied: Pillow>=4.2.1 in /opt/conda/lib/python3.11/site-
packages (from qiskit[visualization]) (11.0.0)
```

```
Requirement already satisfied: pylatexenc>=1.4 in
/opt/.qbraid/environments/qbraid_000000/pyenv/lib/python3.11/site-packages (from
qiskit[visualization]) (2.10)
Requirement already satisfied: seaborn>=0.9.0 in
/opt/.qbraid/environments/qbraid_000000/pyenv/lib/python3.11/site-packages (from
qiskit[visualization]) (0.13.2)
Requirement already satisfied: contourpy>=1.0.1 in
/opt/conda/lib/python3.11/site-packages (from
matplotlib>=3.3->qiskit[visualization]) (1.3.1)
Requirement already satisfied: cycler>=0.10 in /opt/conda/lib/python3.11/site-
packages (from matplotlib>=3.3->qiskit[visualization]) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in
/opt/conda/lib/python3.11/site-packages (from
matplotlib>=3.3->qiskit[visualization]) (4.55.2)
Requirement already satisfied: kiwisolver>=1.3.1 in
/opt/conda/lib/python3.11/site-packages (from
matplotlib>=3.3->qiskit[visualization]) (1.4.7)
Requirement already satisfied: packaging>=20.0 in
/opt/conda/lib/python3.11/site-packages (from
matplotlib>=3.3->qiskit[visualization]) (24.0)
Requirement already satisfied: pyparsing>=2.3.1 in
/opt/conda/lib/python3.11/site-packages (from
matplotlib>=3.3->qiskit[visualization]) (3.2.0)
Requirement already satisfied: six>=1.5 in /opt/conda/lib/python3.11/site-
packages (from python-dateutil>=2.8.0->qiskit[visualization]) (1.16.0)
Requirement already satisfied: pandas>=1.2 in /opt/conda/lib/python3.11/site-
packages (from seaborn>=0.9.0->qiskit[visualization]) (2.2.3)
Requirement already satisfied: pbr>=2.0.0 in
/opt/.qbraid/environments/qbraid_000000/pyenv/lib/python3.11/site-packages (from
stevedore>=3.0.0->qiskit[visualization]) (6.1.0)
Requirement already satisfied: mpmath<1.4,>=1.1.0 in
/opt/.qbraid/environments/qbraid_000000/pyenv/lib/python3.11/site-packages (from
sympy>=1.3->qiskit[visualization]) (1.3.0)
Requirement already satisfied: pytz>=2020.1 in /opt/conda/lib/python3.11/site-
packages (from pandas>=1.2->seaborn>=0.9.0->qiskit[visualization]) (2024.1)
Requirement already satisfied: tzdata>=2022.7 in /opt/conda/lib/python3.11/site-
packages (from pandas>=1.2->seaborn>=0.9.0->qiskit[visualization]) (2024.2)
Note: you may need to restart the kernel to use updated packages.
```

[2]:
```
%pip install qiskit_aer
```

```
Requirement already satisfied: qiskit_aer in
/opt/.qbraid/environments/qbraid_000000/pyenv/lib/python3.11/site-packages
(0.15.1)
Requirement already satisfied: qiskit>=1.1.0 in
/opt/.qbraid/environments/qbraid_000000/pyenv/lib/python3.11/site-packages (from
qiskit_aer) (1.3.0)
Requirement already satisfied: numpy>=1.16.3 in /opt/conda/lib/python3.11/site-
```

```
packages (from qiskit_aer) (1.26.4)
Requirement already satisfied: scipy>=1.0 in
/opt/.qbraid/environments/qbraid_000000/pyenv/lib/python3.11/site-packages (from
qiskit_aer) (1.14.1)
Requirement already satisfied: psutil>=5 in /opt/conda/lib/python3.11/site-
packages (from qiskit_aer) (5.9.8)
Requirement already satisfied: rustworkx>=0.15.0 in
/opt/.qbraid/environments/qbraid_000000/pyenv/lib/python3.11/site-packages (from
qiskit>=1.1.0->qiskit_aer) (0.15.1)
Requirement already satisfied: sympy>=1.3 in
/opt/.qbraid/environments/qbraid_000000/pyenv/lib/python3.11/site-packages (from
qiskit>=1.1.0->qiskit_aer) (1.13.3)
Requirement already satisfied: dill>=0.3 in
/opt/.qbraid/environments/qbraid_000000/pyenv/lib/python3.11/site-packages (from
qiskit>=1.1.0->qiskit_aer) (0.3.9)
Requirement already satisfied: python-dateutil>=2.8.0 in
/opt/conda/lib/python3.11/site-packages (from qiskit>=1.1.0->qiskit_aer) (2.9.0)
Requirement already satisfied: stevedore>=3.0.0 in
/opt/.qbraid/environments/qbraid_000000/pyenv/lib/python3.11/site-packages (from
qiskit>=1.1.0->qiskit_aer) (5.4.0)
Requirement already satisfied: typing-extensions in
/opt/conda/lib/python3.11/site-packages (from qiskit>=1.1.0->qiskit_aer)
(4.12.2)
Requirement already satisfied: symengine<0.14,>=0.11 in
/opt/.qbraid/environments/qbraid_000000/pyenv/lib/python3.11/site-packages (from
qiskit>=1.1.0->qiskit_aer) (0.13.0)
Requirement already satisfied: six>=1.5 in /opt/conda/lib/python3.11/site-
packages (from python-dateutil>=2.8.0->qiskit>=1.1.0->qiskit_aer) (1.16.0)
Requirement already satisfied: pbr>=2.0.0 in
/opt/.qbraid/environments/qbraid_000000/pyenv/lib/python3.11/site-packages (from
stevedore>=3.0.0->qiskit>=1.1.0->qiskit_aer) (6.1.0)
Requirement already satisfied: mpmath<1.4,>=1.1.0 in
/opt/.qbraid/environments/qbraid_000000/pyenv/lib/python3.11/site-packages (from
sympy>=1.3->qiskit>=1.1.0->qiskit_aer) (1.3.0)
Note: you may need to restart the kernel to use updated packages.
```

To implement the Deutsch-Jozsa algorithm in Qiskit, we'll start by generating a quantum circuit that implements a query operation for a randomly selected function that satisfies the promise: with 50% chance the function is constant, and with 50% change the function is balanced. For each possibility, the function is selected uniformly from the possibilities.

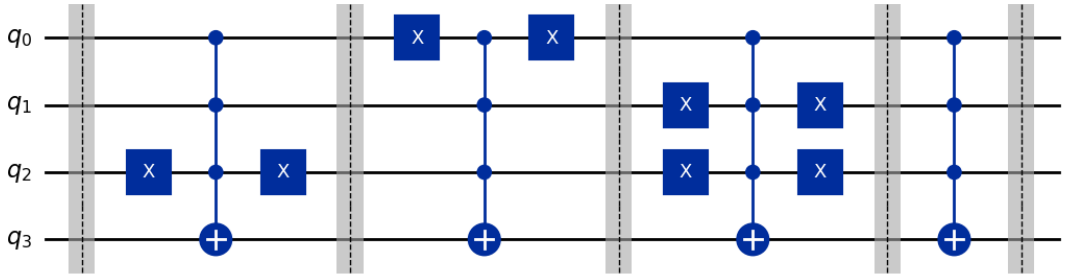
The argument to dj_function is the number of input bits of the function.

[3]:
```python
from qiskit import QuantumCircuit
import numpy as np

def dj_function(num_qubits):
    """
```

```
    Create a random Deutsch-Jozsa function.
    """

    qc = QuantumCircuit(num_qubits + 1)
    if np.random.randint(0, 2):
        # Flip output qubit with 50% chance
        qc.x(num_qubits)
    if np.random.randint(0, 2):
        # return constant circuit with 50% chance
        return qc

    # next, choose half the possible input states
    on_states = np.random.choice(
        range(2**num_qubits),  # numbers to sample from
        2**num_qubits // 2,  # number of samples
        replace=False,  # makes sure states are only sampled once
    )

    def add_cx(qc, bit_string):
        for qubit, bit in enumerate(reversed(bit_string)):
            if bit == "1":
                qc.x(qubit)
        return qc

    for state in on_states:
        qc.barrier()  # Barriers are added to help visualize how the functions␣
↪are created. They can safely be removed.
        qc = add_cx(qc, f"{state:0b}")
        qc.mcx(list(range(num_qubits)), num_qubits)
        qc = add_cx(qc, f"{state:0b}")

    qc.barrier()

    return qc
```

We can show the quantum circuit implementation of the query gate using the draw method as usual.

[3]:
```
display(dj_function(3).draw('mpl'))
```



Next we define a function that creates the Deutsch-Jozsa circuit, taking a quantum circuit implementation of a query gate as an argument.

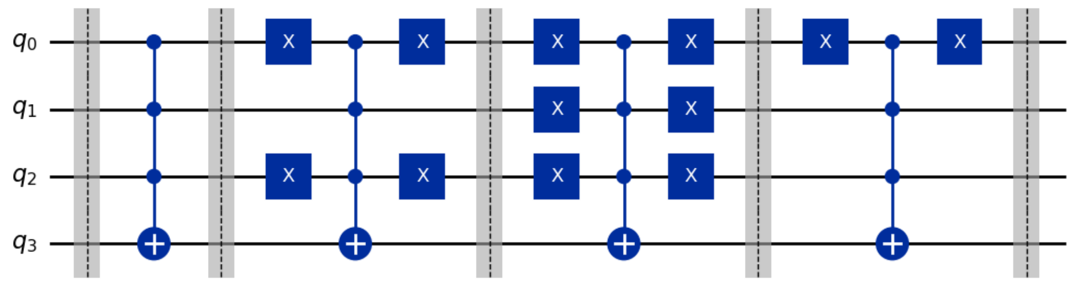
[4]:
```
# Replace the ?
def compile_circuit(function: QuantumCircuit):
    """
    Compiles a circuit for use in the Deutsch-Jozsa algorithm.
    """
    n = function.num_qubits - 1
    qc = QuantumCircuit(n + 1, n)
    qc.x(n)
    qc.h(range(n + 1))
    qc.compose(function, inplace=True)
    qc.h(range(n))
    qc.measure(range(n), range(n))
    return qc
```

Finally, a function that runs the Deutsch-Jozsa circuit once is defined.

[5]:
```
# Replace ?
from qiskit_aer import AerSimulator

def dj_algorithm(function: QuantumCircuit):
    """
    Determine if a Deutsch-Jozsa function is constant or balanced.
    """
    qc = compile_circuit(function)

    result = AerSimulator().run(qc, shots=1, memory=True).result()
    measurements = result.get_memory()
    if "1" in measurements[0]:
        return "balanced"
    return "constant"
```

We can test our implementation by choosing a function randomly, displaying the quantum circuit implementation of a query gate for this function, and then running the Deutsch-Jozsa algorithm on that function.

[6]:
```
# Replace ?
f = dj_function(3)
display(f.draw('mpl'))
display(dj_algorithm(f))
```



'balanced'

2 End of Notebook