T3 - Exercise Simon

December 11, 2024

1 Simon's Algorithm

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

```
Collecting qiskit[visualization]
  Downloading
qiskit-1.3.0-cp39-abi3-manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata
(12 kB)
Collecting rustworkx>=0.15.0 (from qiskit[visualization])
  Using cached rustworkx-0.15.1-cp38-abi3-
manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (9.9 kB)
Requirement already satisfied: numpy<3,>=1.17 in /opt/conda/lib/python3.11/site-
packages (from qiskit[visualization]) (1.26.4)
Collecting scipy>=1.5 (from qiskit[visualization])
  Using cached
scipy-1.14.1-cp311-cp311-manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata
(60 kB)
Collecting sympy>=1.3 (from qiskit[visualization])
  Using cached sympy-1.13.3-py3-none-any.whl.metadata (12 kB)
Collecting dill>=0.3 (from qiskit[visualization])
  Using cached dill-0.3.9-py3-none-any.whl.metadata (10 kB)
Requirement already satisfied: python-dateutil>=2.8.0 in
/opt/conda/lib/python3.11/site-packages (from qiskit[visualization]) (2.9.0)
Collecting stevedore>=3.0.0 (from qiskit[visualization])
  Downloading stevedore-5.4.0-py3-none-any.whl.metadata (2.3 kB)
Requirement already satisfied: typing-extensions in
/opt/conda/lib/python3.11/site-packages (from qiskit[visualization]) (4.12.2)
Collecting symengine<0.14,>=0.11 (from qiskit[visualization])
  Using cached symengine-0.13.0-cp311-cp311-
manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (1.2 kB)
Requirement already satisfied: matplotlib>=3.3 in
/opt/conda/lib/python3.11/site-packages (from qiskit[visualization]) (3.9.3)
Collecting pydot (from qiskit[visualization])
  Downloading pydot-3.0.3-py3-none-any.whl.metadata (10 kB)
Requirement already satisfied: Pillow>=4.2.1 in /opt/conda/lib/python3.11/site-
packages (from qiskit[visualization]) (11.0.0)
Collecting pylatexenc>=1.4 (from qiskit[visualization])
  Using cached pylatexenc-2.10-py3-none-any.whl
```

```
Collecting seaborn>=0.9.0 (from qiskit[visualization])
  Using cached seaborn-0.13.2-py3-none-any.whl.metadata (5.4 kB)
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)
Collecting pbr>=2.0.0 (from stevedore>=3.0.0->qiskit[visualization])
  Using cached pbr-6.1.0-py2.py3-none-any.whl.metadata (3.4 kB)
Collecting mpmath<1.4,>=1.1.0 (from sympy>=1.3->qiskit[visualization])
  Using cached mpmath-1.3.0-py3-none-any.whl.metadata (8.6 kB)
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)
Using cached dill-0.3.9-py3-none-any.whl (119 kB)
Using cached
rustworkx-0.15.1-cp38-abi3-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (2.0
MB)
Using cached
scipy-1.14.1-cp311-cp311-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (41.2
MB)
Using cached seaborn-0.13.2-py3-none-any.whl (294 kB)
Downloading stevedore-5.4.0-py3-none-any.whl (49 kB)
Using cached
symengine-0.13.0-cp311-cp311-manylinux_2_17_x86_64.manylinux2014_x86_64.whl
(49.7 MB)
Using cached sympy-1.13.3-py3-none-any.whl (6.2 MB)
Downloading pydot-3.0.3-py3-none-any.whl (35 kB)
Downloading
qiskit-1.3.0-cp39-abi3-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (6.7 MB)
                         6.7/6.7 MB
```

```
62.7 MB/s eta 0:00:00
Using cached mpmath-1.3.0-py3-none-any.whl (536 kB)
Using cached pbr-6.1.0-py2.py3-none-any.whl (108 kB)
Installing collected packages: pylatexenc, mpmath, sympy, symengine, scipy,
rustworkx, pydot, pbr, dill, stevedore, seaborn, qiskit
Successfully installed dill-0.3.9 mpmath-1.3.0 pbr-6.1.0 pydot-3.0.3
pylatexenc-2.10 qiskit-1.3.0 rustworkx-0.15.1 scipy-1.14.1 seaborn-0.13.2
stevedore-5.4.0 symengine-0.13.0 sympy-1.13.3
Note: you may need to restart the kernel to use updated packages.
```

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

```
Collecting qiskit_aer
  Using cached qiskit_aer-0.15.1-cp311-cp311-
manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (8.0 kB)
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)
Using cached
qiskit_aer-0.15.1-cp311-cp311-manylinux_2_17_x86_64.manylinux2014_x86_64.whl
(12.3 MB)
Installing collected packages: qiskit_aer
Successfully installed qiskit_aer-0.15.1
Note: you may need to restart the kernel to use updated packages.
```

[3]:
```
from qiskit_aer import AerSimulator
```

To implement Simon's algorithm in Qiskit, we'll use the fact that we can convert unitary matrices into gates in Qiskit using the .unitary method. Specifically, we'll use this methodology to define a query gate for a randomly chosen function satisfying Simon's problem for a given string s.

[4]:
```
# import random
import qiskit.quantum_info as qi
from qiskit import QuantumCircuit
import numpy as np

def simon_function(s: str):
    """
    Create a QuantumCircuit implementing a query gate for Simon problem obeying␣
 ↪the promise for the hidden string `s`
    """
    # Our quantum circuit has 2n qubits for n = len(s)
    n = len(s)
    qc = QuantumCircuit(2 * n)

    # Define a random permutation of all n bit strings. This permutation will␣
 ↪effectively hide the string s.
    pi = np.random.permutation(2**n)

    # Now we'll define a query gate explicitly. The idea is to first define a␣
 ↪function g(x) = min{x,x ^ s}, which
    # is a simple function that satisfies the promise, and then we take f to be␣
 ↪the composition of g and the random
    # permutation pi. This gives us a random function satisfying the promise␣
 ↪for s.

    query_gate = np.zeros((4**n, 4**n))
    for x in range(2**n):
        for y in range(2**n):
            z = y ^ pi[min(x, x ^ int(s, 2))]
```

```python
            query_gate[x + 2**n * z, x + 2**n * y] = 1

    # Our circuit has just this one query gate
    qc.unitary(query_gate, range(2 * n))
    return qc
```

Next we'll define a function that runs the circuit in Simon's problem k times and reports the results.

[5]:
```python
# Replace
from qiskit_aer import AerSimulator
from qiskit import ClassicalRegister

def simon_measurements(problem: QuantumCircuit, k: int):
    """
    Quantum part of Simon's algorithm. Given a `QuantumCircuit` that
    implements f, get `k` measurements to be post-processed later.
    """
    n = problem.num_qubits // 2

    qc = QuantumCircuit(2 * n, n)
    qc.h(range(n))
    qc.compose(problem, inplace=True)
    qc.h(range(n))
    qc.measure(range(n), range(n))

    result = AerSimulator().run(qc, shots=k, memory=True).result()
    return result.get_memory()
```

The following code cell illustrates how the function works when we plug in the query gate constructed above. Feel free to try different arguments, but keep in mind that the cost of the simulation we've built is exponential in the number of qubits we require — so don't make the string s too long if you don't want to wait!

[6]:
```python
display(simon_measurements(simon_function("11011"),k=12))
```

```
['00111',
 '11000',
 '10110',
 '01110',
 '10110',
 '01001',
 '00011',
 '10001',
 '11111',
 '10110',
 '01101',
 '01110']
```

To do the post-processing, we can make use of the galois package, which has a built-in function for computing the null space modulo 2.

[7]:
```
%pip install galois
```

```
Collecting galois
  Downloading galois-0.4.3-py3-none-any.whl.metadata (14 kB)
Requirement already satisfied: numpy>=1.21.0 in /opt/conda/lib/python3.11/site-
packages (from galois) (1.26.4)
Collecting numba<0.61,>=0.55 (from galois)
  Downloading
numba-0.60.0-cp311-cp311-manylinux2014_x86_64.manylinux_2_17_x86_64.whl.metadata
(2.7 kB)
Requirement already satisfied: typing-extensions>=4.0.0 in
/opt/conda/lib/python3.11/site-packages (from galois) (4.12.2)
Collecting llvmlite<0.44,>=0.43.0dev0 (from numba<0.61,>=0.55->galois)
  Downloading llvmlite-0.43.0-cp311-cp311-
manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (4.8 kB)
Downloading galois-0.4.3-py3-none-any.whl (4.2 MB)
                         4.2/4.2 MB
43.8 MB/s eta 0:00:00
Downloading
numba-0.60.0-cp311-cp311-manylinux2014_x86_64.manylinux_2_17_x86_64.whl (3.7 MB)
                         3.7/3.7 MB
68.4 MB/s eta 0:00:00
Downloading
llvmlite-0.43.0-cp311-cp311-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (43.9
MB)
                         43.9/43.9 MB
141.8 MB/s eta 0:00:0000:01
Installing collected packages: llvmlite, numba, galois
Successfully installed galois-0.4.3 llvmlite-0.43.0 numba-0.60.0
Note: you may need to restart the kernel to use updated packages.
```

[8]:
```
# Replace
import numpy as np
import galois

def simon_algorithm(problem: QuantumCircuit):
    """
    Given a `QuantumCircuit` that implements a query gate for Simon problem,␣
 ↪return the hidden string `s`.
    """

    # Quantum part: run the circuit defined previously k times and gather the␣
 ↪measurement results.
```

```
    # Replace +10 by +r for any nonnegative integer r depending on desired␣
↪confidence.

    measurements = simon_measurements(problem, k=problem.num_qubits // 2 + 10)
    print("Measurement results:")
    display(measurements)

    # Classical post-processing:

    # 1. Convert measurements of form '11101' to 2D-array of integers
    matrix = np.array([list(bitstring) for bitstring in measurements]).
↪astype(int)

    # 2. Interpret matrix as using arithmetic mod 2, and find null space
    null_space = galois.GF(2)(matrix).null_space()
    print("Null space:")
    display(null_space)

    # 3. Convert back to a string
    print("Guess for hidden string s:")
    if len(null_space) == 0:
        # No non-trivial solution; `s` is all-zeros
        return "0" * len(measurements[0])
    return "".join(np.array(null_space[0]).astype(str))
```

And finally we can try it out.

[9]:
```
display(simon_algorithm(simon_function("10011")))
```

```
Measurement results:

['00100',
 '10001',
 '10010',
 '10010',
 '00100',
 '01111',
 '11110',
 '10010',
 '11110',
 '10001',
 '01111',
 '01111',
 '10110',
 '11010',
 '11101']

Null space:
```

```
GF([[1, 0, 0, 1, 1]], order=2)
Guess for hidden string s:
'10011'
```

2 End of Notebook