T2 - Entanglement - Teleportation - ParsaVARES

November 1, 2024

1 Entanglement in Action

2 Teleportation

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

```
Collecting qiskit[visualization]
  Using cached
qiskit-1.2.4-cp38-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])
  Using cached stevedore-5.3.0-py3-none-any.whl.metadata (2.3 kB)
Requirement already satisfied: typing-extensions in
/opt/conda/lib/python3.11/site-packages (from qiskit[visualization]) (4.11.0)
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]) (3.9.2)
Collecting pydot (from qiskit[visualization])
  Using cached pydot-3.0.2-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]) (1.3.0)
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]) (4.54.1)
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)
Using cached stevedore-5.3.0-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)
Using cached pydot-3.0.2-py3-none-any.whl (35 kB)
```

```
Using cached
qiskit-1.2.4-cp38-abi3-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (4.8 MB)
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-3.0.2
pylatexenc-2.10 qiskit-1.2.4 rustworkx-0.15.1 scipy-1.14.1 seaborn-0.13.2
stevedore-5.3.0 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) (1.2.4)
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) (5.3.0)
Requirement already satisfied: typing-extensions in
/opt/conda/lib/python3.11/site-packages (from qiskit>=1.1.0->qiskit_aer)
(4.11.0)
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]:

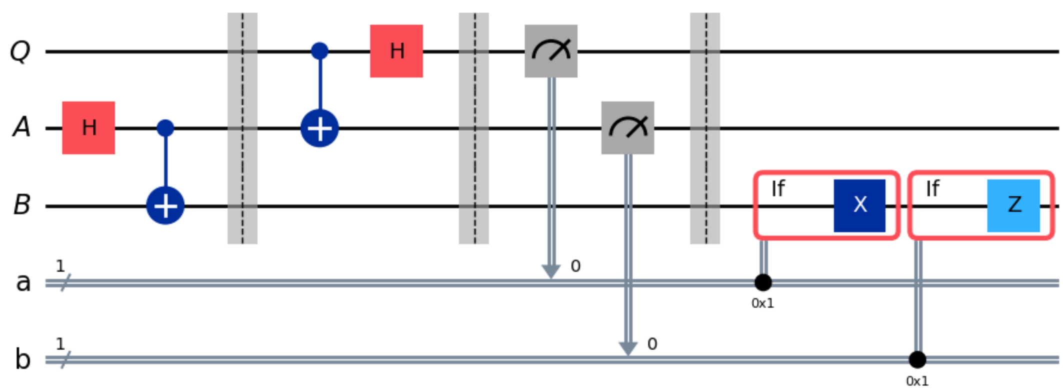
```python
# Required imports
from qiskit import QuantumCircuit, QuantumRegister, ClassicalRegister
from qiskit_aer import AerSimulator
from qiskit.visualization import plot_histogram
from qiskit.result import marginal_distribution
from qiskit.circuit.library import UGate
from numpy import pi, random
```

2.1 Create the Protocol

[4]:

```python
qubit = QuantumRegister(1, "Q")
ebit0 = QuantumRegister(1, "A")
ebit1 = QuantumRegister(1, "B")
a = ClassicalRegister(1, "a")
b = ClassicalRegister(1, "b")

protocol = QuantumCircuit(qubit, ebit0, ebit1, a, b)

# Prepare ebit used for teleportation
# Replace ?
protocol.h(ebit0)            # Apply Hadamard gate to Alice's qubit
protocol.cx(ebit0, ebit1)    # Apply CNOT gate from Alice's to Bob's qubit
protocol.barrier()

# Alice's operations
# Replace ?
protocol.cx(qubit, ebit0)    # Alice entangles her qubit with the ebit
protocol.h(qubit)            # Apply Hadamard gate on Alice's qubit
protocol.barrier()

# Alice measures and sends classical bits to Bob
# Replace?
protocol.measure(qubit, a)   # Measure Alice's qubit and store result in
 ↪classical register a
```

```
protocol.measure(ebit0, b)    # Measure the entangled qubit and store result in␣
  ↪classical register b
protocol.barrier()

# Bob uses the classical bits to conditionally apply gates
# Replace ?
with protocol.if_test((a, 1)):
    protocol.x(ebit1)         # Apply X gate if Alice's measurement result a is␣
  ↪1
with protocol.if_test((b, 1)):
    protocol.z(ebit1)         # Apply Z gate if Alice's measurement result b is␣
  ↪1

display(protocol.draw('mpl'))
```



The circuit makes use of a few features of Qiskit that require some explanations, including the barrier and if_test functions. The barrier function creates a visual separation making the circuit diagram more readable, and it also prevents Qiskit from performing various simplifications and optimizations across barriers during compilation when circuits are run on real hardware. The if_test function applies an operation conditionally depending on a classical bit or register.

The circuit first initializes (A,B) to be in a + state (which is not part of the protocol itself), followed by Alice's operations, then her measurements, and finally Bob's operations.

2.2 Test the Protocol

To test that the protocol works correctly, we'll apply a randomly generated single-qubit gate to the initialized 0 state of Q to obtain a random quantum state vector to be teleported. By applying the inverse (i.e., conjugate transpose) of that gate to B after the protocol is run, we can verify that the state was teleported by measuring to see that it has returned to the 0 state.

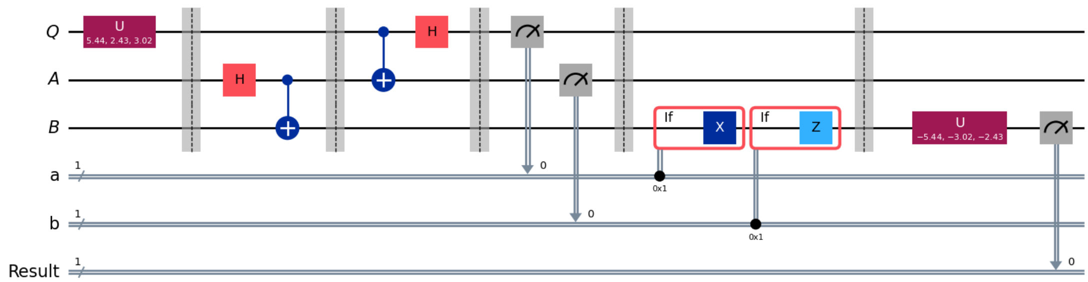
First we'll randomly choose a unitary qubit gate.

[5]:
```
random_gate = UGate(
    theta=random.random() * 2 * pi,
    phi=random.random() * 2 * pi,
    lam=random.random() * 2 * pi,
)

display(random_gate.to_matrix())
```

```
array([[-0.91172136+0.j        ,  0.4078565 -0.04916533j],
       [-0.31217036+0.26704648j, -0.61690014+0.67131964j]])
```

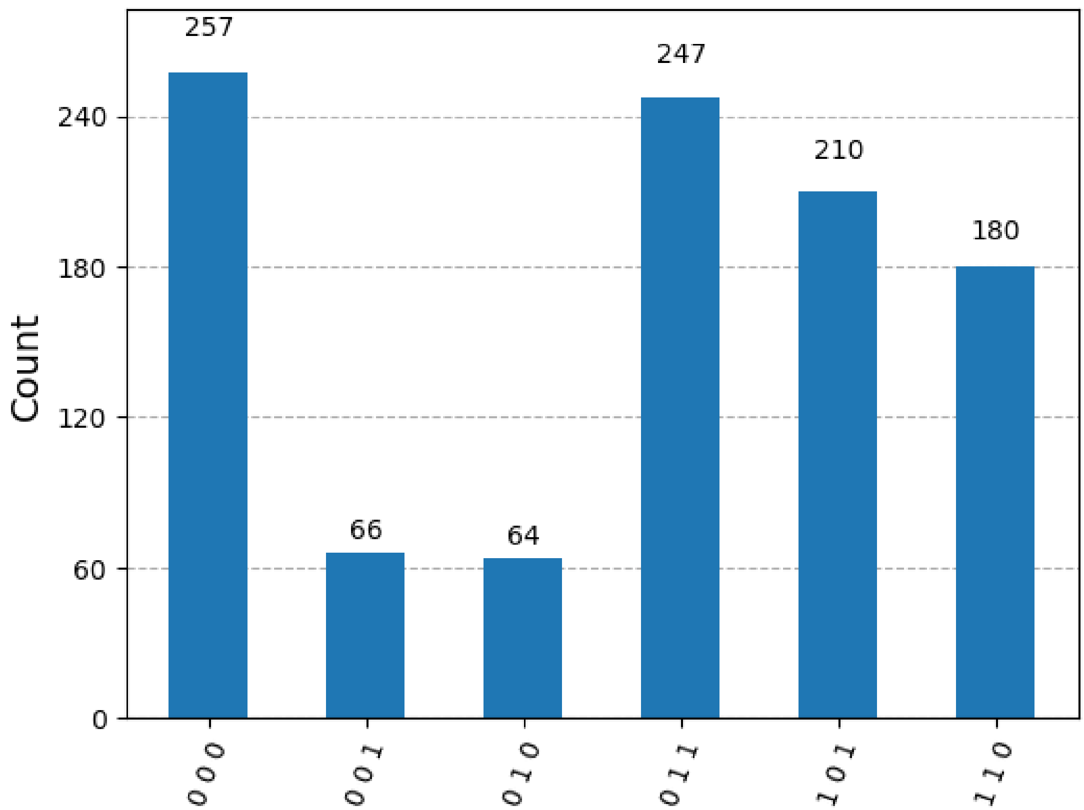
Now we'll create a new testing circuit that first applies our random gate to Q, then runs the teleportation circuit, and finally applies the inverse of our random gate to the qubit B and measures. The outcome should be 0 with certainty.

[7]:
```
# Create a new circuit including the same bits and qubits used in the
# teleportation protocol.

test = QuantumCircuit(qubit, ebit0, ebit1, a, b)

# Start with the randomly selected gate on Q
# Replace ?

test.append(random_gate, qubit)
test.barrier()

# Append the entire teleportation protocol from above.
# Replace ?

test = test.compose(protocol)
test.barrier()

# Finally, apply the inverse of the random unitary to B and measure.
# Replace ?

test.append(random_gate.inverse(), ebit1)

result = ClassicalRegister(1, "Result")
test.add_register(result)
test.measure(ebit1, result)

display(test.draw('mpl'))
```



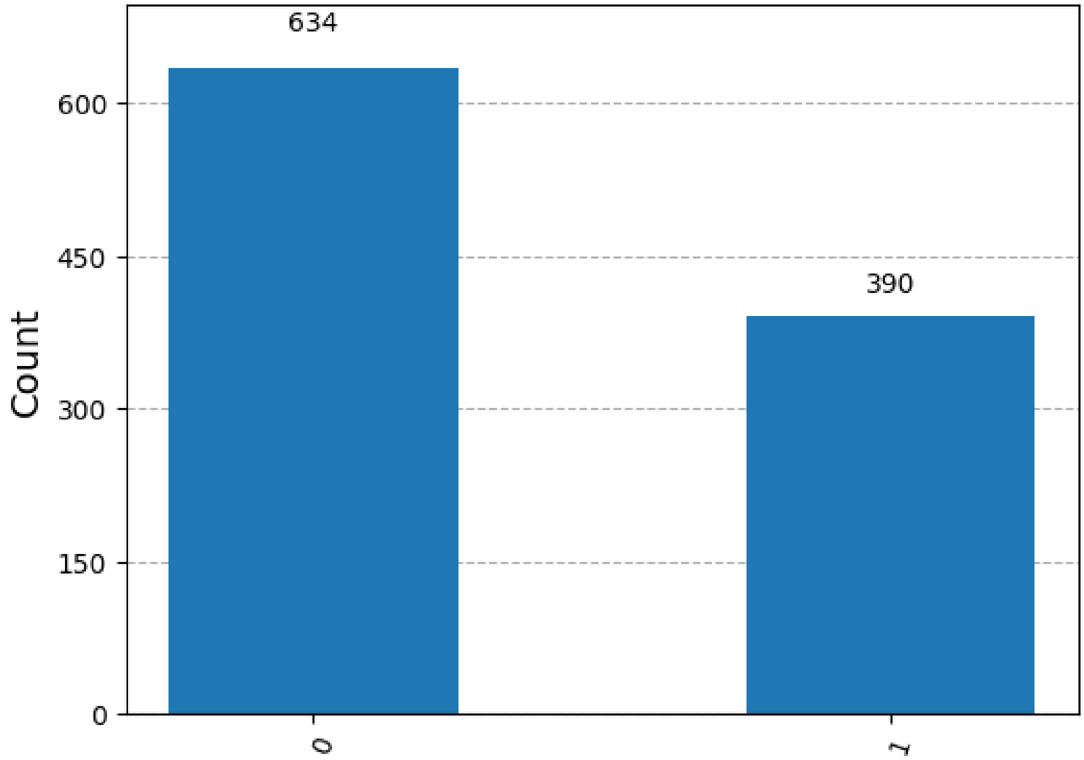
Finally let's run the Aer simulator on this circuit and plot a histogram of the outputs. We'll see the statistics for all three classical bits: the bottom/leftmost bit should always be 0, indicating that the qubit Q was successfully teleported into B, while the other two bits should be roughly uniform.

[8]:
```
# Replace ?
result = AerSimulator().run(test).result()
statistics = result.get_counts()
display(plot_histogram(statistics))
```



We can also filter the statistics to focus just on the test result qubit if we wish, like this:

[9]:
```
filtered_statistics = marginal_distribution(statistics, [2])
display(plot_histogram(filtered_statistics))
```



3 End of Notebook