Lab_7

January 9, 2025

1 Introduction to Quantum Information and Quantum ML

Instructor: Dr Sci. Eng. Przemysław Głowacki

Kacper Dobek 148247

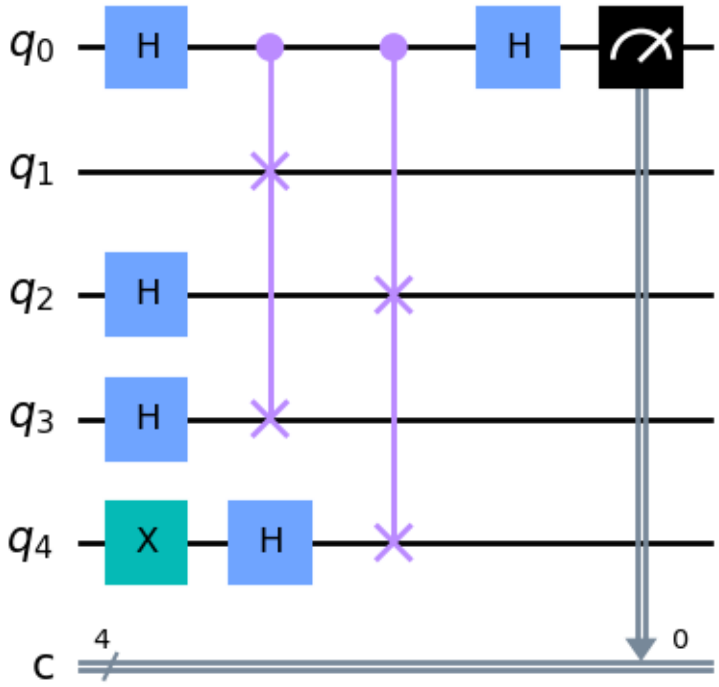
[1]:
```
from qiskit import QuantumCircuit

from math import sqrt
from typing import List
import pandas as pd
```

[2]:
```
from qiskit import Aer, execute

simulator = Aer.get_backend("qasm_simulator")
```

[3]:
```
def construct_circuit(qs: List[str]) -> QuantumCircuit:
    n = len(qs) + 1
    circuit = QuantumCircuit(n, 4)
    circuit.h(0)
    for i in range(1, n):
        if qs[i - 1] == "1":
            circuit.x(i)
        if qs[i - 1] == "+":
            circuit.h(i)
        if qs[i - 1] == "-":
            circuit.x(i)
            circuit.h(i)
    # Include swaps
    n_swaps = len(qs) // 2
    for i in range(1, n_swaps + 1):
        circuit.cswap(0, i, i + n_swaps)
    circuit.h(0)
    circuit.measure(0, 0)
    return circuit


# Sample circuit
```

```
circuit = construct_circuit(["0", "+", "+", "-"])
circuit.draw("mpl")
```

[3]:



[4]:
```
# Set the reference pattern
selected_pattern = ["-", "0"]

# Generate all possible patterns of length 2
charset = ["0", "1", "+", "-"]
patterns = []
for c1 in charset:
    for c2 in charset:
        patterns.append(selected_pattern + [c1, c2])

print("Number of patterns:", len(patterns))
```

```
Number of patterns: 16
```

[5]:
```
def calculate_fidelity(counts: dict, n: int) -> float:
    p0 = counts["0" * n] / sum(counts.values())
    fidelity = abs(2 * p0 - 1.0)
    return fidelity
```

```
def calculate_overlap(fidelity: float) -> float:
    return sqrt(fidelity)
```

[6]:
```
column_names = ["q1", "q2", "q3", "q4", "fidelity", "overlap"]
results = []

# Run simulation of circuits
for pattern in patterns:
    circuit = construct_circuit(pattern)
    job = execute(circuit, simulator, shots=1024)
    result = job.result()
    fidelity = calculate_fidelity(result.get_counts(), len(pattern))
    overlap = calculate_overlap(fidelity)
    row_result = pattern + [fidelity, overlap]
    results.append(row_result)
```

[7]:
```
# Create a pandas DataFrame with the results
df = pd.DataFrame(results, columns=column_names)
df = df.sort_values(by="fidelity", ascending=False).reset_index(drop=True)

display(df)
```

	q1	q2	q3	q4	fidelity	overlap
0	-	0	-	0	1.000000	1.000000
1	-	0	0	0	0.541016	0.735538
2	-	0	-	-	0.517578	0.719429
3	-	0	1	0	0.507812	0.712610
4	-	0	-	+	0.507812	0.712610
5	-	0	1	-	0.281250	0.530330
6	-	0	0	-	0.248047	0.498043
7	-	0	0	+	0.230469	0.480072
8	-	0	1	+	0.228516	0.478033
9	-	0	0	1	0.050781	0.225347
10	-	0	+	0	0.037109	0.192638
11	-	0	-	1	0.035156	0.187500
12	-	0	+	+	0.033203	0.182217
13	-	0	1	1	0.027344	0.165359
14	-	0	+	1	0.005859	0.076547
15	-	0	+	-	0.001953	0.044194