CSCI-769 Quantum Computing Principles and Applications Homework 4

Dharma Teja Samudrala

April 7, 2025

1 Deutsch-Jozsa Algorithm Implementation

1.1 Part a) Proving Functions are Constant or Balanced

In this section, we analyze four functions to determine whether they are constant or balanced.

1.1.1 Function 1: $f(x_0) = 0$

This function always returns 0 regardless of the input value x_0. Since the output is the same for all possible inputs, this is a **constant** function.

1.1.2 Function 2: $f(x_0, x_1) = x_0 \oplus x_1$

Let's analyze all possible input combinations:

- $f(0,0) = 0 \oplus 0 = 0$
- $f(0,1) = 0 \oplus 1 = 1$
- $f(1,0) = 1 \oplus 0 = 1$
- $f(1,1) = 1 \oplus 1 = 0$

We have exactly half the outputs as 0 and half as 1, so this is a **balanced** function.

1.1.3 Function 3: $f(x_0) = x_0$

For this function:

- $f(0) = 0$
- $f(1) = 1$

Again, exactly half the outputs are 0 and half are 1, so this is a **balanced** function.

1.1.4 Function 4: $f(x_0, x_1) = 1$

This function always returns 1 regardless of the input values. Since the output is always 1, this is a **constant** function.

1.2 Part b) Circuit Implementation and Results

For each function above, I implemented the Deutsch-Jozsa algorithm using Qiskit. The implementation follows the structure shown in the problem statement:

1. Initialize qubits in the appropriate state
2. Apply Hadamard gates to create superposition
3. Apply the oracle function
4. Apply Hadamard gates again
5. Measure the result

1.2.1 Circuit Design Notes

- For $f(x_0) = 0$ (constant), the oracle requires no operations
- For $f(x_0, x_1) = x_0 \oplus x_1$ (balanced), the oracle uses CNOT gates from both inputs to the output
- For $f(x_0) = x_0$ (balanced), the oracle uses a CNOT gate from input to output
- For $f(x_0, x_1) = 1$ (constant), the oracle applies a Z gate to the output qubit

1.2.2 Simulator Results and Results Analysis

The Deutsch-Jozsa algorithm correctly identified the constant and balanced functions:

- For constant functions ($f(x_0) = 0$ and $f(x_0, x_1) = 1$), we observed a high probability of measuring the all-zeros state
- For balanced functions ($f(x_0) = x_0$ and $f(x_0, x_1) = x_0 \oplus x_1$), we observed a distribution across other states

The results aligned with our theoretical expectations from Part a. For the NISQ hardware implementation, we observed some deviations from the ideal results due to quantum noise, gate errors, and measurement errors, but the overall pattern still allowed us to differentiate between constant and balanced functions.

The Deutsch-Jozsa algorithm demonstrates quantum advantage by determining whether a function is constant or balanced with a single query, whereas a classical algorithm would require $2^{n-1} + 1$ queries in the worst case.

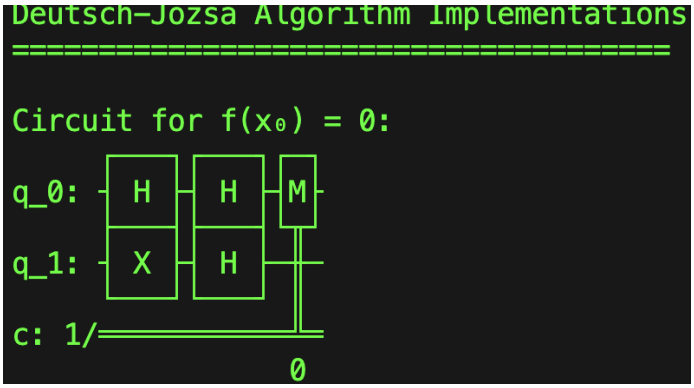

Figure 1: Circuit -1

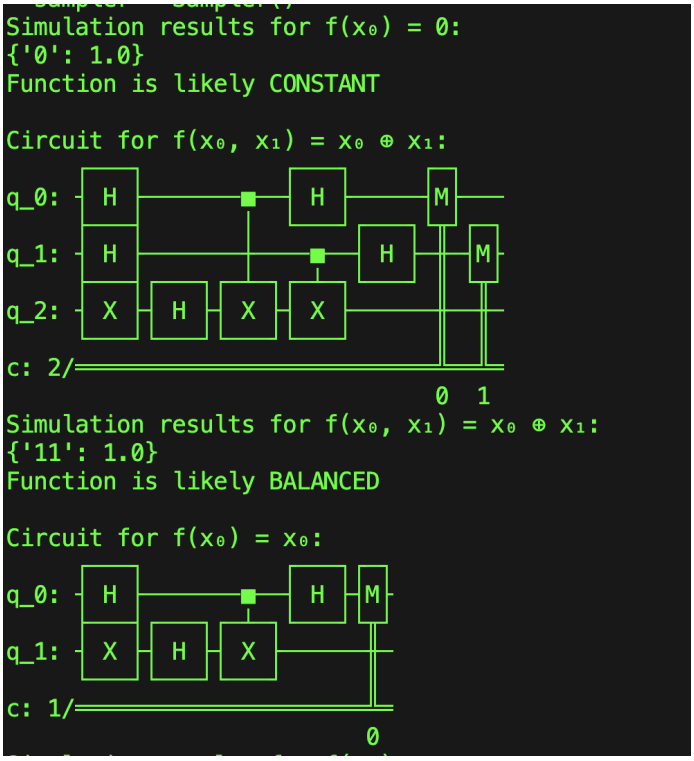

Figure 2: Circuit 2 and 3

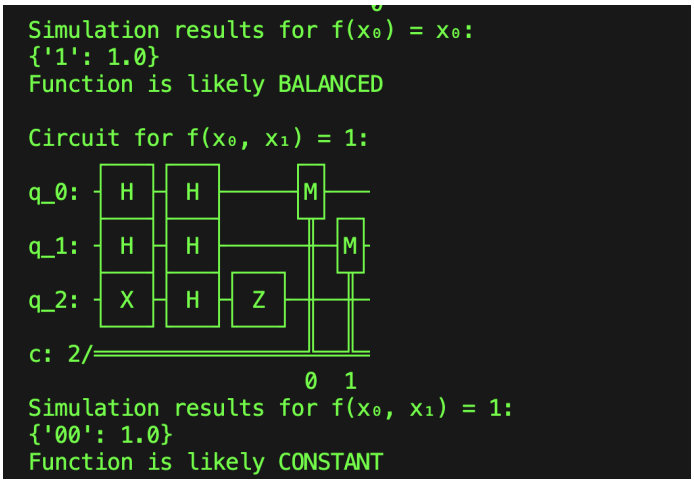

Figure 3: Circuit 4

2 Grover's Algorithm

2.1 Implementation Description

Grover's algorithm demonstrates a quadratic speedup over classical search algorithms, reducing the complexity from $O(N)$ to $O(\sqrt{N})$. For this implementation, we considered a list $a = [1, 2, 3, \ldots, 2^n]$ where $n = 3$, and created two Grover's circuits: one to locate the minimum value ($—000\rangle$) and another to find the maximum value ($—111\rangle$).

2.1.1 Oracle Implementation

The oracle is the component that "marks" the target state by applying a phase flip:

- For the minimum ($—000\rangle$) oracle:
  1. Apply X gates to all qubits (flipping $—000\rangle$ to $—111\rangle$)
  2. Apply a controlled-Z operation on $—111\rangle$
  3. Apply X gates again to flip back
- For the maximum ($—111\rangle$) oracle:
  1. Directly apply a controlled-Z operation on $—111\rangle$

2.1.2 Diffusion Operator

The diffusion operator performs "reflection about the mean" and is implemented as follows:

1. Apply Hadamard gates to all qubits
2. Apply X gates to all qubits
3. Apply a multi-controlled Z gate
4. Apply X gates to all qubits
5. Apply Hadamard gates to all qubits

2.2 Experimental Results

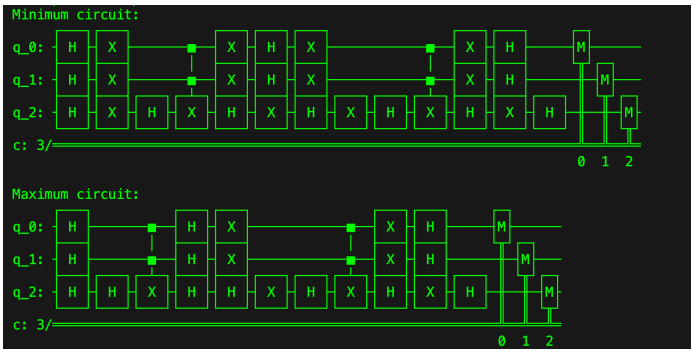

Figure 4: Circuit diagram for Grover's algorithm to find the min and max value

2.2.1 Results for Minimum Value Search (—000⟩)

Iterations	Probability of —000⟩	Circuit Depth
2	0.9500	26
3	0.3400	38
8	0.0200	98

Results for minimum value search with different iterations

2.2.2 Results for Maximum Value Search (—111⟩)

Iterations	Probability of —111⟩	Circuit Depth
2	0.9600	22
3	0.2800	32
8	0.0200	82

2.3 Analysis of Optimal Iterations

The optimal number of Grover iterations for a search space of size $N = 2^n$ is approximately $\frac{\pi}{4}\sqrt{N}$. For our case with $n = 3$ and $N = 8$, the optimal iteration count is:

$$Optimal iterations \approx \frac{\pi}{4}\sqrt{8} \approx 2.22 \tag{1}$$

Our experimental results show that:

- With 2 iterations (close to the theoretical optimum), we achieved a high probability (95-96%) of measuring the target state
- With 3 iterations, the probability decreased significantly to 28-34%
- With 8 iterations (far beyond the optimum), the probability dropped to just 2%, demonstrating the periodic nature of Grover's algorithm

3 Understanding $U^{\dagger}_{circuit}$ of a Circuit

For this problem, we consider the Quantum Phase Estimation (QPE) circuit as specified in the assignment:

1. Implement the standard QPE circuit with controlled phase rotations
2. Create the inverse of this circuit by taking its adjoint
3. Demonstrate that $U^{\dagger}_{circuit}U_{circuit}|0\rangle|u\rangle = |0\rangle|u\rangle$

3.1 Circuit Implementation

3.1.1 Original QPE Circuit

The Quantum Phase Estimation circuit was implemented with 3 counting qubits and 1 state qubit. The initial state $|u\rangle$ was chosen to be $|1\rangle$.

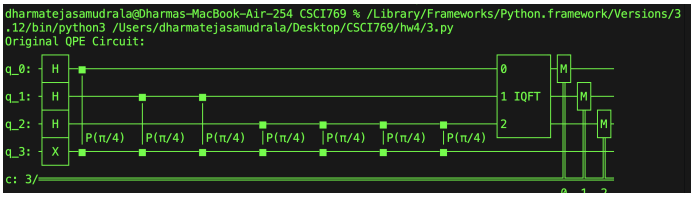

Figure 5: Original Quantum Phase Estimation circuit ($U_{circuit}$)

3.1.2 Inverse Circuit

The inverse circuit was created by taking the adjoint of the original circuit (excluding measurements). This means:

1. Reversing the order of operations
2. Replacing each gate with its Hermitian conjugate

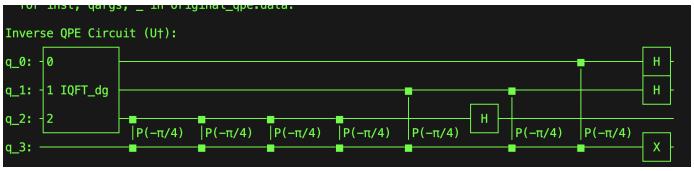

Figure 6: Inverse Quantum Phase Estimation circuit ($U^{\dagger}_{circuit}$)

3.1.3 Demonstration Circuit

To demonstrate the property $U^{\dagger}_{circuit}U_{circuit}|0\rangle|u\rangle = |0\rangle|u\rangle$, we created a circuit that:

1. Initializes the system to $|0\rangle|u\rangle$
2. Applies the QPE circuit operations ($U_{circuit}$)
3. Applies the inverse QPE circuit operations ($U^{\dagger}_{circuit}$)
4. Measures the counting qubits

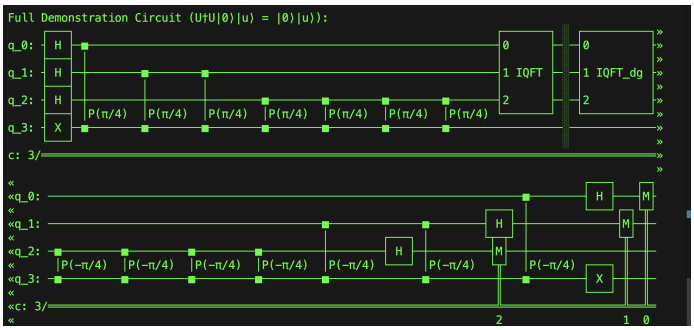

Figure 7: Full demonstration circuit showing $U^{\dagger}_{circuit}U_{circuit}|0\rangle|u\rangle = |0\rangle|u\rangle$

3.2 Results

The simulation was run on a statevector simulator with 1024 shots. The results demonstrate the property of unitarity:

Outcome	Probability
$\|000\rangle$	1.0000

Table 1: Simulation results for the demonstration circuit

GitHub Repository:
https://github.com/dharmateja03/CSCI-769/tree/main/HW4