Module 4 : Quantum Algorithms

Contents

4.1 What is an Algorithm?
4.2 Time Complexity & Example
4.3 Deutsch-Jozsa Algorithm
4.4 Bernstein–Vazirani Algorithm
4.5 Simon’s Algorithm
4.6 Grover’s Algorithm
4.7 Shor’s Algorithm
4.8 Quantum Algorithms Table

4.1 What is an Algorithm ?

An algorithm is a step-by-step procedure or set of rules designed to solve a specific problem or perform a task.

In classical computing, algorithms are executed on traditional computers. In quantum computing, algorithms leverage quantum mechanical principles like superposition, entanglement, and interference to solve problems, often faster than classical algorithms for specific tasks.

Example:

Algorithm for sum of 2 numbers

```
Input a, b
c = a + b
Print c
```

Algorithm to print first 10 numbers

```
For a = 1 to 10:
    Print a
```

4.2 What is Time Complexity?

Time complexity measures the amount of computational time an algorithm takes to run as a function of the input size, typically denoted as n. It’s expressed using Big-O notation, which describes the upper bound of the algorithm’s running time in the worst-case scenario, ignoring constants and lower-order terms.

What is Time Complexity?

Example of Time Complexity:

- **Linear Search:** To find an element in an unsorted list of n elements, you may need to check each element one by one. In the worst case, you check all n elements.
    - **Time Complexity:** *O(n)*, meaning the time grows linearly with the input size.

- **Binary Search:** For a sorted list, you can halve the search space with each step by comparing the target to the middle element.
    - **Time Complexity:** *O(log n)*, as the number of steps grows logarithmically.

Why Time Complexity important?

- It helps compare algorithms' efficiency, especially for large inputs.
- we do not have infinite memory and time.
- Standard algorithms may take long time to execute.
- Quantum algorithms often aim to reduce time complexity compared to classical counterparts.

Ex: Try to find if $\mathbf{2^{64} + 5}$ is prime?

4.3 Deutsch-Jozsa Algorithm

- Problem Statement
- Classical solution
- Quantum Solution
- What is Quantum Oracle?
- Phase Oracle

Deutsch-Jozsa Algorithm

Overview

The Deutsch-Jozsa algorithm is a quantum algorithm that solves a specific problem exponentially faster than any classical algorithm. It demonstrates the power of quantum computing by exploiting superposition and interference.

Problem Statement

Given a function $f:\{0,1\}^n \to \{0,1\}$ which is promised to be either:

Constant:

$f(x)$ returns the same value (0 or 1) for all inputs x.

Balanced:

$f(x)=0$ for exactly half of the 2^n inputs, and $f(x)=1$ for the other half.

The Goal is to Determine whether f is constant or balanced with the fewest possible queries to the function (oracle).

Deutsch-Jozsa Algorithm

Example

Consider a function $F : \{0,1\}^n \mapsto \{0,1\}$

$F(x_0,x_1)=x_0 \oplus x_1$

$F(0,0)=0$

$F(0,1)=1$, So the given function is balanced.

$F(1,0)=1$

$F(1,1)=0$

Classical Solution

- Classically, for 2 inputs you have to check both the inputs to determine its global property.
- With 2^n inputs, you will have to check $2^{(n-1)}+1$ inputs in the worst case to check whether the function is balanced or constant.
- So classical computers will take exponential runtime to execute the algorithm.

Quantum Solution

The quantum algorithm uses a quantum oracle that evaluates f in superposition.

It requires only one query to determine whether f is constant or balanced.

Steps:

1. Initialize Qubits:

Start with n qubits in the state $|0\rangle^{\otimes n}$ and an auxiliary qubit in the state $|1\rangle$.

2. Apply Hadamard Gates:

Apply Hadamard gates to all n+1 qubits to create a superposition of all possible inputs.

3. Oracle Query:

Query the oracle, which applies the function f to the quantum state, encoding the function's behavior into the phase of the state.

4. Apply Hadamard Gates Again:

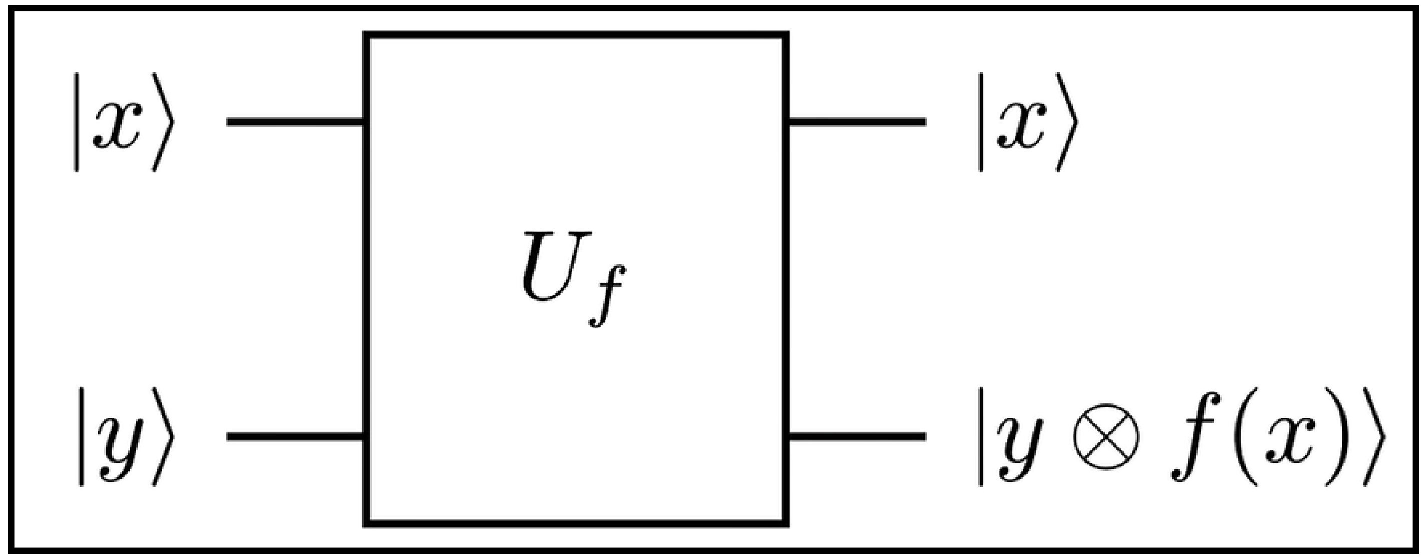
Apply Hadamard gates again to the first n qubits (ignore the auxiliary qubit).

5. Measure:

Measure the first n qubits. If the result is $|0\rangle^{\otimes n}$, the function is constant. Otherwise, the function is balanced.

What is Quantum Oracle?

- When classical gates or functions are fed input some output is given this gate may or may not reversible (or its inverse may or may not exist). Eg: AND gate.
- But in quantum realm, Reversibility of quantum gates must satisfy .i.e they must be a unitary operator hence we use a ancillary qubit and the oracle function gets xor with the input to give output.



$$O(|x\rangle \otimes |y\rangle) = |x\rangle \otimes |y \oplus f(x)\rangle$$

mathematical definition of a quantum oracle

Phase Oracle

Alternatively, you can encode f into an oracle O by applying a phase based on the input to O.
For example,
you might define O such that,

$$O|x\rangle = (-1)^{f(x)}|x\rangle.$$

If a phase oracle acts on a register initially in a computational basis state $|x\rangle$ then this phase is a global phase and hence not observable. But such an oracle can be a powerful resource if applied to a superposition or as a controlled operation.

4.4 Bernstein–Vazirani Algorithm

- Problem Statement
- Classical solution
- Quantum Solution

Bernstein–Vazirani Algorithm

Overview

The Bernstein–Vazirani algorithm is a simple quantum algorithm that solves a specific type of problem called the Bernstein–Vazirani problem. It efficiently finds a hidden binary string by utilizing quantum parallelism, requiring only one query to the black box function.

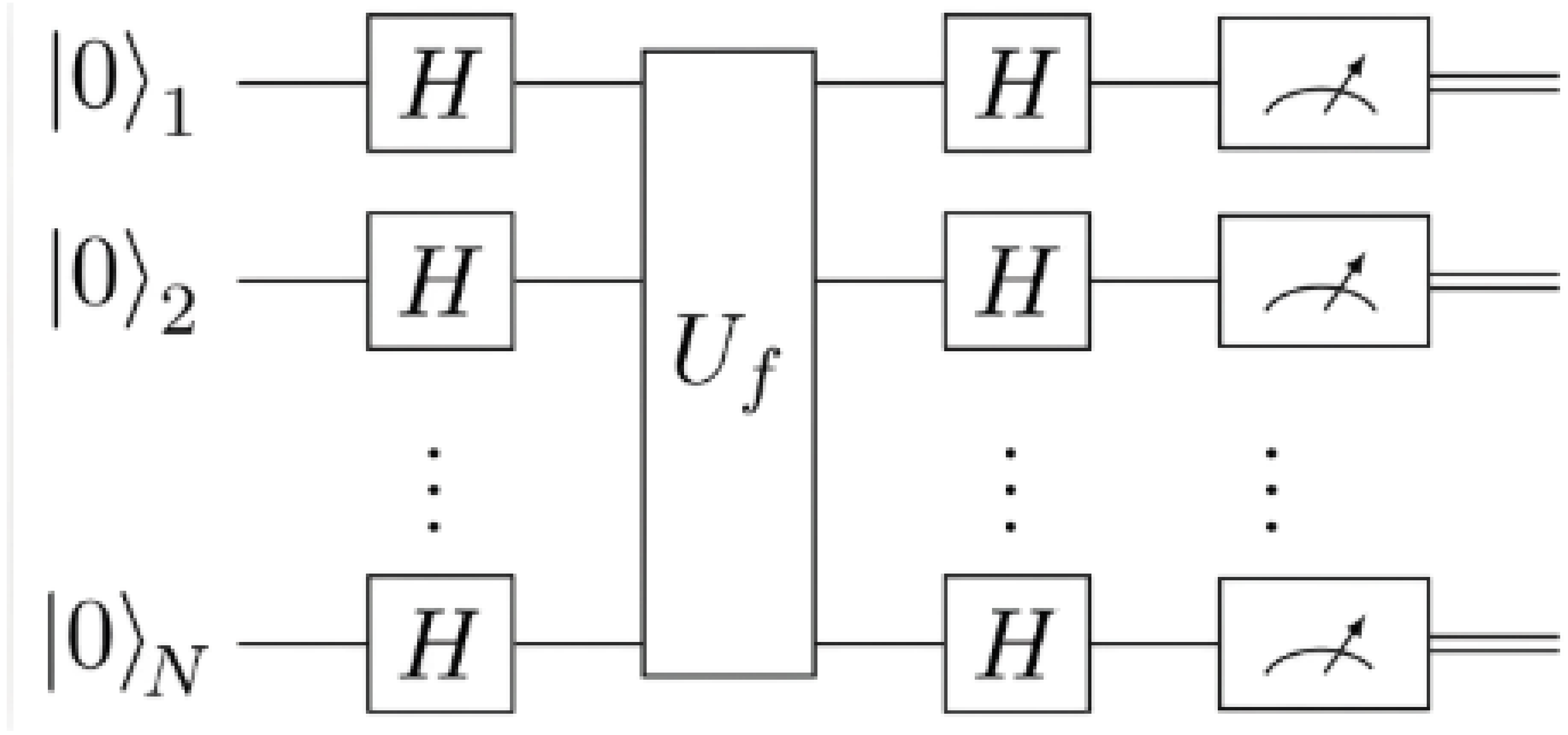
Problem Statement

We have a function, $f{:}\{0,1\}^n \rightarrow \{0,1\}$,
which basically takes an input x, and returns the value of the inner product of x with s modulo 2, where s is a number known by the function.
$f(x)=(x \cdot s) \mathrm{mod} 2=((x1 \cdot s1)+x2 \cdot s2+\ldots+(xn \cdot sn)) \mathrm{mod} 2=$
$((x1 \cdot s1) \oplus (x2 \cdot s2) \oplus \ldots \oplus (xn \cdot sn))$
The Goal is to Determine s.

Quantum Solution

The quantum algorithm uses n input qubits and one output qubit..

Steps:



Steps:

3. Oracle Application:

The oracle encodes $f(x)=(x \cdot s) \bmod 2$

$$|x\rangle|y\rangle \to |x\rangle|y \oplus f(x)\rangle$$

So the state becomes:

$$\frac{1}{\sqrt{2^n}} \sum_x (-1)^{s \cdot x} |x\rangle \otimes \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$$,Since y is now decoupled, we ignore it.

4. Apply Hadamard Gate Again:

Apply Hadamard on |x⟩ to extract s.

$$\frac{1}{\sqrt{2^n}} \sum_x (-1)^{s \cdot x} H^{\otimes n} |x\rangle = |s\rangle$$

5. Measure:

The result of measurement will be the binary string s with 100% probability.

4.5 Simon’s Algorithm

- Problem Statement
- Classical solution
- Quantum Solution

Simon’s Algorithm

Overview

Simon’s Algorithm is a quantum algorithm that efficiently finds a hidden string s in a function with a promised periodicity, offering exponential speedup over classical methods. It was a key precursor to Shor’s algorithm, highlighting the power of quantum computing.

Problem Statement

Given a function
f:$\{0,1\}^n \rightarrow \{0,1\}^n$ with the promise that there exists a secret string $s \in \{0,1\}n$ such that:
$f(x)=f(y)$ if and only if $y= x \oplus s$,
If $s=0^n$, then f is one-to-one,
If $s \neq 0^n$, then f is two-to-one.

Goal: Determine the hidden string s using as few queries to the oracle as possible.

Simon’s Algorithm

Classical Solution

- Query the oracle with different inputs to find a collision, i.e., two inputs x and y such that $f(x)=f(y)$.
- If $f(x)=f(y)$, then $y=x\oplus s$, so $s=x\oplus y$
- Due to the birthday paradox, finding a collision in a 2-to-1 function over
- 2^n inputs requires $O(\sqrt{2^n})=O(2^{n/2})$ queries with high probability.
- After finding a collision, compute $s=x\oplus y$
- If no non-trivial s is found after sufficient queries, conclude $s=0^n$

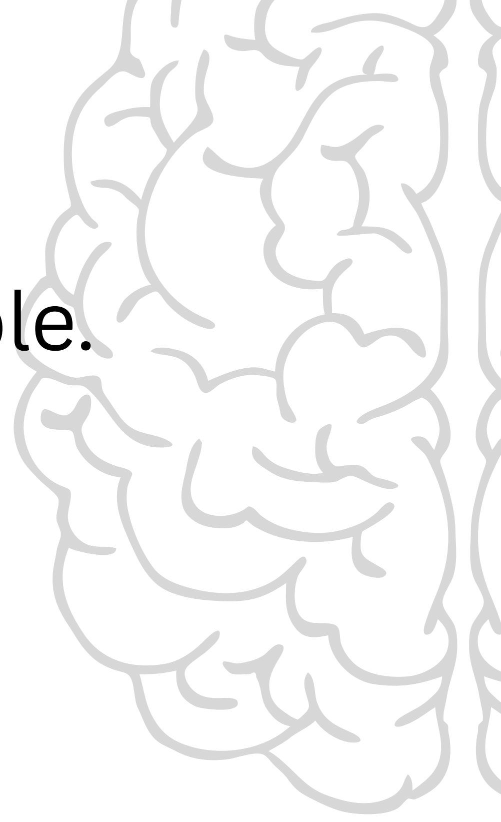
Simon’s Algorithm

Example:

- Example (n = 2, s = (1, 0)):
- Query random inputs, e.g.:
- $x1=00, f(00)=00$
- $x2=01, f(01)=01$
- $x3=10, f(10)=00$
- Collision: $f(00)=f(10)=00$
- Compute: $s=00\oplus10=(1,0)$
- Expected queries: $O(\surd 22)=O(2)$

Quantum Solution

Determine the hidden string s using as few queries to the oracle as possible.



Steps:

1. Initialize the Qubits:

Start with input qubits and nnn output qubits in the state:

$$|0\rangle^{\otimes n} \otimes |0\rangle^{\otimes n}$$

2. Apply Hadamard Gates to Input Qubits:

This creates a superposition of all possible input values:

$$\frac{1}{\sqrt{2^n}} \sum_{x \in \{0,1\}^n} |x\rangle \otimes |0\rangle$$

3. Query the Oracle:

The oracle maps each $|x\rangle\, |0\rangle$ to $|x\rangle\, |f(x)\rangle$ giving:

$$\frac{1}{\sqrt{2^n}} \sum_{x} |x\rangle |f(x)\rangle$$

Steps:

4. Apply Hadamard Gates to Input Qubits

This creates a superposition of all possible input values:

$$\sum_{y\in\{0,1\}^n} c_y|y\rangle$$

where the amplitudes c_y are non-zero only for vectors y such that

$$y\cdot s=0 \mod 2$$

5. Measure the Input Register

You get a random vector $y\in\{0,1\}^n$ such that $y\cdot s=0$

Steps:

6. Repeat the Process

$O(n)$ Times Collect n linearly independent y vectors satisfying $y \cdot s$**=0**

7. Solve the Linear System

Solve the system of equations to determine the hidden string s.

$$y \cdot s = 0 \mod 2$$

8. Interpret the Result

- If the only solution is $s=0^n$, then f is 1-to-1.
- If a non-zero s satisfies the system, then f is 2-to-1, and you have found the hidden s.

4.6 Grover’s Algorithm

- Problem Statement
- Classical solution
- Quantum Solution

Grover’s Algorithm

Overview

Purpose: Grover’s Algorithm searches an unsorted database of $N=2^n$ items to find a unique item (the "marked" item) that satisfies a given condition, as identified by an oracle.

Problem Statement

Given an unsorted list of N items and an oracle function $f(x)$ that returns 1 if x is the target item and 0 otherwise, find the target item $x0$.

Grover’s Algorithm

Classical Solution

Process:

Sequentially check each item by querying the oracle until the target is found.

On average, requires checking half the items.

Time Complexity:

$O(N)=O(2n)$ queries.

Example:

List: $\{A,B,C,D\} N=4$, target is C.

Query:

$f(A)=0, f(B)=0, f(C)=1$

Stop at C.

Average queries:

$N/2=2$

Quantum Solution

n qubits to represent N=2^n items and an oracle that marks the target.

Steps:

1. Initialize the Qubits: Start with quantum state, Initialize qubits to $|0\rangle^{\otimes n}$.

2. Apply Hadamard Gates: This creates an equal superposition of all possible states.

$$\frac{1}{\sqrt{N}}\sum_{x=0}^{N-1}|x\rangle$$

3. Grover's Iteration(Repeated ≈ √N times) :

Oracle Operation: Apply a phase flip to the target state using the oracle function, If x is the correct item, its phase is flipped. Apply diffusion operator to amplify the amplitude of the target.

$$|x\rangle \rightarrow (-1)^{f(x)}|x\rangle$$

4. Measurement: After approximately √N iterations, measure the quantum state.

4.7 Shor’s Algorithm

- **Problem Statement**
- **Classical solution**
- **Quantum Solution**

Shor’s Algorithm

Overview

Purpose: Shor’s Algorithm factors a large composite number N into its prime factors, critical for breaking RSA encryption.

Problem Statement

Given an unsorted list of N items and an oracle function $f(x)$ that returns 1 if x is the target item and 0 otherwise, find the target item $x0$.

Shor’s Algorithm

Classical Solution

Process:

Use algorithms like trial division or the General Number Field Sieve (GNFS).

- Trial division: Test divisibility by numbers up to √N.
- **GNFS:** More efficient for large N, but still slow.

Example:

- **Trial division:** Check 2 (not a factor), 3 (divides: 15÷3=5)
- **Result:** Factors are 3 and 5.
- **Queries:** Up to √15≈4 but GNFS is faster for large N.

Quantum Solution

Reduces factoring to finding the period r of
$f(x)=a^x \mod N$
where a is coprime with N.

Steps:

1. Choose a random a that is coprime with N.

2. Initialize two quantum registers:

$$|0\rangle^{\otimes n} \otimes |0\rangle^{\otimes n}$$

3. Apply Hadamard Gates: Apply Hadamard gates to the first register

$$\frac{1}{\sqrt{2^n}} \sum_x |x\rangle|0\rangle$$

4. Compute the modular exponentiation function:
$f(x)=a^x \mod N$, resulting in

$$\frac{1}{\sqrt{2^n}} \sum_x |x\rangle|a^x \mod N\rangle$$

Steps:

5. Apply the Quantum Fourier Transform (QFT) to the first register to extract the period r.

6.
Measure and classically compute:
gcd($a^{r/2}$±1,N), to find the non-trivial factors of N.

4.8 Quantum Algorithms: Problems, Complexities & Speedups

Algorithm	Problem	Classical Complexity	Quantum Complexit y	SpeedUp
Bernstein-Vazirani	Find hidden string ‘*s*‘	$O(n)$	$O(1)$	Linear
Deutsch-Jozsa	Constant or balanced function	$O(2^n)$	$O(1)$	Exponential
Simon’s	Find hidden XOR mask s	$O(2^{n/2})$	$O(n)$	Exponential
Grover’s	Unstructured search	$O(2^n)$	$O(\sqrt{2^n})$	Quadratic
Shor’s	Integer factorization	$O(\exp((\log N)^{1/3} (\log\log N)^{2/3}))$	$O((\log N)^3)$	Exponential

Thank you