Oracles

An **Oracle** implements a specific function using quantum gates, it is essentially a "Quantum compatible" query.

By thinking or problems in terms of Oracles, it gives us a way to compare quantum and classical algorithms with each other. In the same way that a classical function may query an input: *Is x == 1* , a quantum Oracle will query a state : *is f(|00>) == True* .

Quantum Oracle

- Takes a quantum state and performs a query over each element of the state vector.
- The oracle queries the entire state vector at once.
- The Oracle then alters its output depending on whether it has found a state that satisfies it.
- Is **reversible**

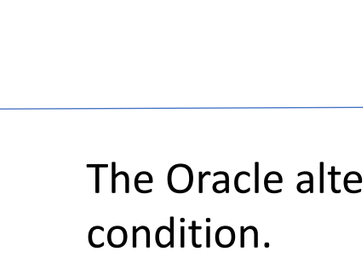
Classical Query

- Takes only one input at a time
- Each input must be queried individually
- The results of each query must be stored for comparison at the end.
- Is **not reversible** in general

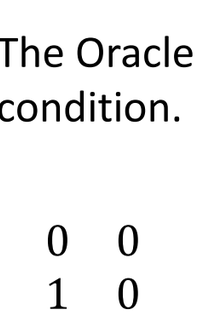
Deutsch-Jozsa

Complexity: O(1)

Complexity: $O(2^n)$



In the Deutsch-Jozsa case, our oracle takes an unknown function *f(x)* and wraps it in a unitary gate so that it is now "quantum compatible".



For the Oracle to work, the *f(x)* must be either a constant or balanced function.

This matrix is an example of an Oracle for a constant function.

$$\begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

The DJ algorithm only needs to make a **single** oracle query to arrive at a solution.

Takes *x* – as a single input of *f(x)*.

Records the total number of 1 or 0 outputs from *f(x)*

Returns 0 if the number of output = 1 is equal to the number of output = 0 (balanced), returns 0 otherwise (constant).

Needs approximately 2^n queries if *f(x)* has *n-bit* inputs

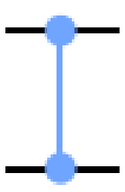
Grover's

Complexity: $O(\sqrt{n})$

Complexity: O(n)

The Oracle alters the state if the state meets the desired condition.

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix}$$



In the Grover's case, the Oracle is constructed using a condition that is unique to the element we are trying to find.
Grover's algorithm only needs to be run $\sqrt{n}$ times to find the desired element in a list of size *n*.

This matrix and gate represents the oracle for when the desired state is in position 4 out of 4 choices. Note that it can be constructed for any position without knowing the position first.

Linear search must go through each element one by one until it finds the element satisfies the condition.

The classical query can only operate over one input at a time.

Linear search will then return the position of that element in the database, it needs to be run *n* times in the worst case for a list of size *n*.