Lecture 4 Day 2

QUANTUM ALGORITHMS
QUANTUM ALGORITHMS
QUANTUM ALGORITHMS
QUANTUM ALGORITHMS

Overview :

① Deutsch - Jozsa Algorithm : oracles, DJ Theory, implementation with qiskit

② Grover's Algorithm : Grover Theory, amplitude, amplification, implementation with qiskit.

For all the studies we perform regarding quantum algorithms, we assume the quantum oracle correction is satisfied.

ORACLES : for lot of quantum algorithms, we assume we have access to some oracle. Example: oracle can be a physical device, we can think it as a black box. So we cannot look inside oracle! But it performs what we want, we pass some queries to it, and it returns some answers.

Goal: given such an oracle, we want to determine property of the oracle and ideally what we try is to minimize the number of queries that we have to the oracle. Example of an oracle can be phonebook : You can think of the oracle as a phone book where you raise query with a phone number rather than with names. when you ask, hey oracle! ., is this number exist with this name in the phonebook ? Then oracle will revert with yes or no ! The example of a phonebook is used a lot to understand how oracles work !

one could use oracle as if oracle solves some satisfiability problem, or in general if we have some constraints that we give to the oracle and we ask it, can this, this and this constraint, be all satisfied at the same time.? And the oracle can say, can, it should be some constraints that the oracle can more or less easily check, Yes its true or No, it cannot satisfy all of it same time. Example can be Sudoku

Sudoku :

we can run some algorithms, and give oracle an input of different combinations of numbers and ask it, "okay does it satisfy the fact that, we have at most have each number exactly 1's in every row and in every column.? and in every box.
If it says yes we have a solution then go ahead, if it says no, then its wrong solution.
But example of sudoku can be feasible.

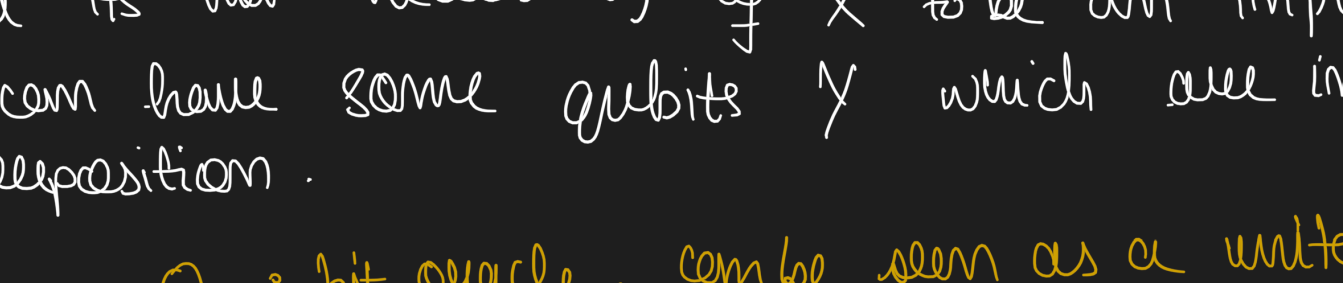
More on sudoku Example :- Not found except 13yo video of D-wave, thats also criticized by viewers !

On a classical computer, such an oracle is given by a function, $f:\{0,1\}^n$ or $f=(0+1)^*$
or
$$\boxed{f:\{0,1\}^n \rightarrow \{0,1\}^m}$$
Giving input n bits, taking m bits as output

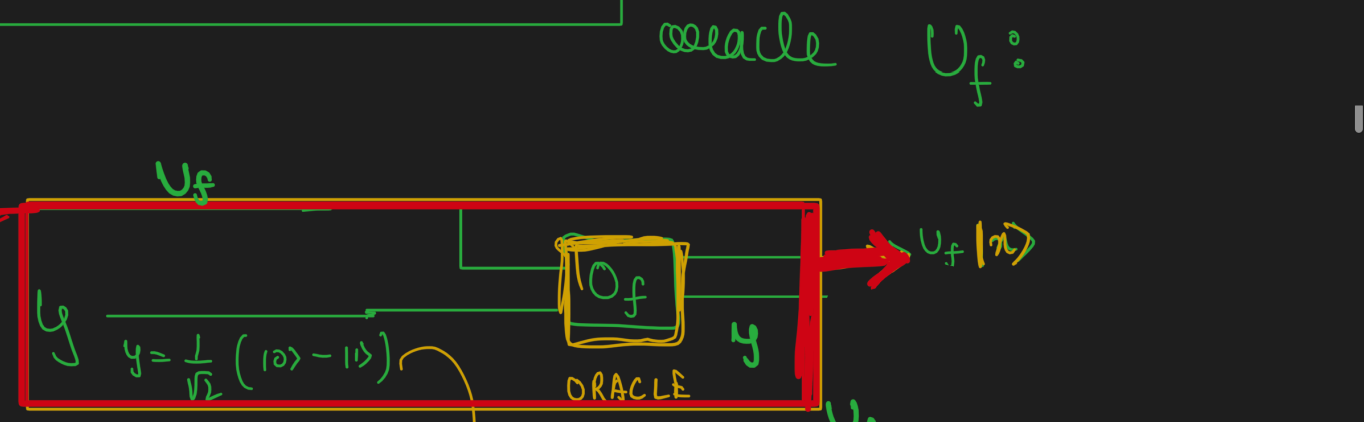
However,
On a quantum computer, that everything has to be unitary and reversible, the oracle must to be reversible.
So if we give x as input, we denote oracle as $\boxed{O_f}$

n qubits
X
O_f
X
f(x)

In order our oracle to be reversible we have to show input as O/P as well :-



In other case :-
so if we now have m qubits as input, or if we have n qubits as output, so, here on the bottom qubits, then we also need to have m qubits as additional inputs along with n-qubits, which we call select qubits, and as o/p we will get $Y \oplus f(n)$
so, easiest case we just put in zero so, we get put an X and zeroes, and then we get out X and f(x)



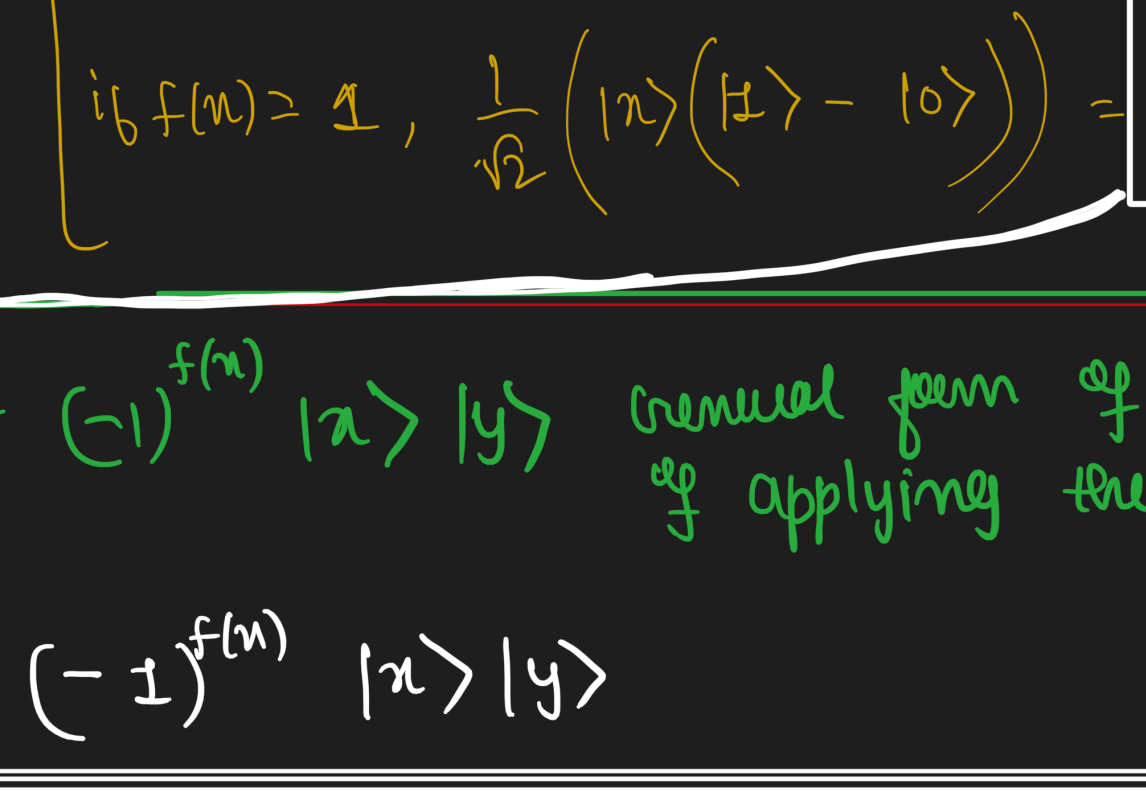
Now its not necessary of X to be an input we can have some qubits Y which are in superposition.

O_f : bit oracle ., can be seen as a unitary gate, which performs the map
$$O_f|x\rangle|y\rangle = |x\rangle|y\oplus f(x)\rangle$$

In practice we actually does not consider more than one output

$$\boxed{\begin{array}{l} \text{for } f:\{0,1\}^n \rightarrow \{0,1\} \\ f:(0+1)^* \rightarrow (0+1) \end{array}}$$

we can not only construct bit oracle but also, we can use bit-oracle and construct phase oracle U_f :



$$O_f|x\rangle|y\rangle = \frac{1}{\sqrt{2}}\left(O_f|x\rangle|0\rangle - O_f|x\rangle|1\rangle\right)$$

$$\because y = \frac{1}{\sqrt{2}}\left[|0\rangle-|1\rangle\right]$$

$$= \frac{1}{\sqrt{2}}\left(|x\rangle|0\oplus f(x)\rangle - |x\rangle|1\oplus f(x)\rangle\right)$$

$$= \begin{cases} \text{if } f(x)=0, & \frac{1}{\sqrt{2}}\left(|x\rangle\left(|0\rangle-|1\rangle\right)\right) = |x\rangle|y\rangle \\ \text{if } f(x)=1, & \frac{1}{\sqrt{2}}\left(|x\rangle\left(|1\rangle-|0\rangle\right)\right) = -|x\rangle|y\rangle \end{cases}$$

$$= (-1)^{f(x)}|x\rangle|y\rangle$$ general form of application of applying the oracle

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

This expression, $(-1)^{f(x)}|x\rangle|y\rangle$ is independent of $|y\rangle$ =>, we put in y and we get out Y, we dont really need Y, we need it to construct its own experiment, when we actually want to implement this function we do need Y, however, if we just mathematically now wanna say, we are applying oracle, we do not need to write down Y everytime, but we just write U_f : ancilla qubit and if we write U_f then in theory always has this like qubit.
U_f : phase oracle, we will get phases which performs the map :
$$\boxed{U_f|x\rangle = (-1)^{f(x)}|x\rangle}$$

① Deutsch - Jozsa Algorithm : oracles, DJ theory, implementation with qiskit

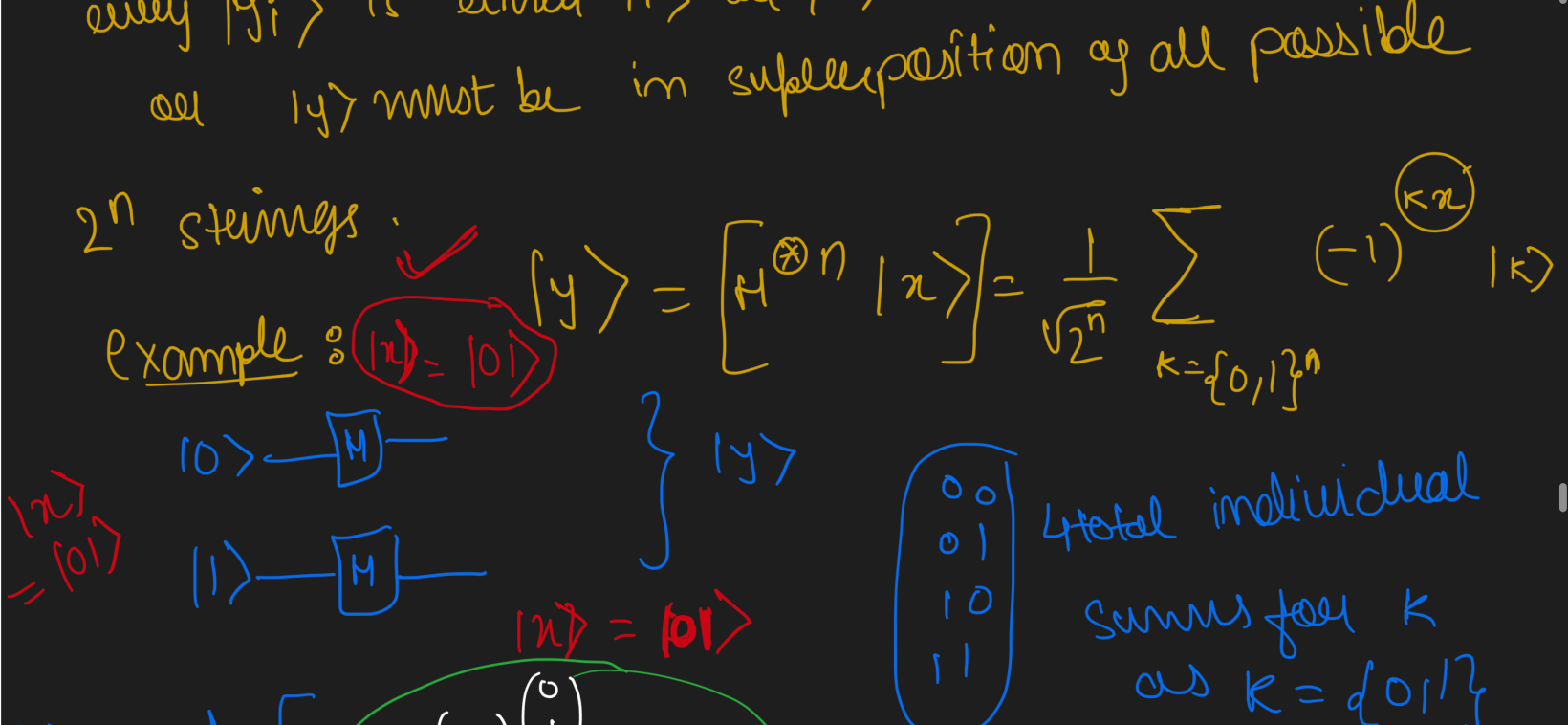
① Deutsch - Jozsa Algorithm :-

hadamard gate on n-qubits : recall that
$$H|0\rangle = |+\rangle = \frac{1}{\sqrt{2}}\left[|0\rangle+|1\rangle\right]$$
$$H|1\rangle = |-\rangle = \frac{1}{\sqrt{2}}\left(|0\rangle-|1\rangle\right)$$
for $x\in\{0,1\}$

for $x\in\{0,1\}$: $|x\rangle$ —H— $|y\rangle$
$$|x\rangle \text{—H—} |y\rangle = \frac{1}{\sqrt{2}}\left(|0\rangle + (-1)^{x}|1\rangle\right)$$
$$= \frac{1}{\sqrt{2}}(-1)^{0\cdot x}|0\rangle + (-1)^{1\cdot x}|1\rangle$$
$$= \frac{1}{\sqrt{2}}\sum_{k\in\{0,1\}}(-1)^{k\cdot x}|k\rangle$$
fancy way to rewrite the action of hadamard gate on the states $|0\rangle$ and $|1\rangle$

for n-qubits,
for $x\in\{0,1\}^n$:

(⊗ = tensor product)



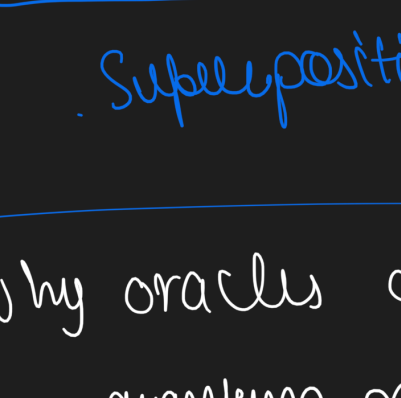
we put a tensor product b/w them, And if they are met upon, thats why one hadamard gate on each different qubit. Then we write it H to the tensor product power n. $H^{\otimes n}$

$$|y\rangle = \left[H^{\otimes n}|x\rangle\right] = \frac{1}{\sqrt{2^n}}\sum_{k=\{0,1\}^n}(-1)^{k\cdot x}|k\rangle$$
where $K\cdot X$ = inner product

every $|y_i\rangle$ is either $|+\rangle$ or $|-\rangle$ or $|y\rangle$ must be in superposition of all possible 2^n strings.

$$|y\rangle = \left[H^{\otimes n}|x\rangle\right] = \frac{1}{\sqrt{2^n}}\sum_{k=\{0,1\}^n}(-1)^{(k\cdot x)}|k\rangle$$

Example : $|x\rangle = |01\rangle$



$|x\rangle = |01\rangle$

00
01
10
11

4 total individual sums for k as $k=\{0,1\}^2$

$$|y\rangle = \frac{1}{\sqrt{2^2}}\left[(-1)^{(00)\binom{0}{1}}|00\rangle + (-1)^{(01)\binom{0}{1}}|01\rangle + (-1)^{(10)\binom{0}{1}}|10\rangle + (-1)^{(11)\binom{0}{1}}|11\rangle\right]$$

$+|00\rangle$ $-|01\rangle$ $+|10\rangle$ $-|11\rangle$

$$(00)\binom{0}{1}$$
1×2 2×1

$$= \frac{1}{2}\left(|00\rangle - |01\rangle + |10\rangle - |11\rangle\right)$$

Superposition of all states !

why oracles are reversible ?:-
so^m = quantum mechanics is always unitary, This is related to unitarity of quantum mechanics, there's a postulate that every possible evolution that you can consider is always unitary. So if we have the oracle, and we dont have the oracle, if you look at the oracle, it also meant to be unitary and if its unitary, its immediately also reversible that implies that.

① Deutsch - Jozsa Algorithm :-

=> This is the first quantum algorithm which has proven the quantum advantage because of superposition and entanglement in quantum computing. Easy way to understand, less practical applications.

we are given $f:\{0,1\}^n \rightarrow \{0,1\}$ realized by an oracle, of which we know that its either constant, (constant means in this case that, or balanced all inputs maps to same output)

so, either all inputs map to zero, or all they map to $|1\rangle$ all time. But if the function is constant it means that not matter what we input we would always get same output.

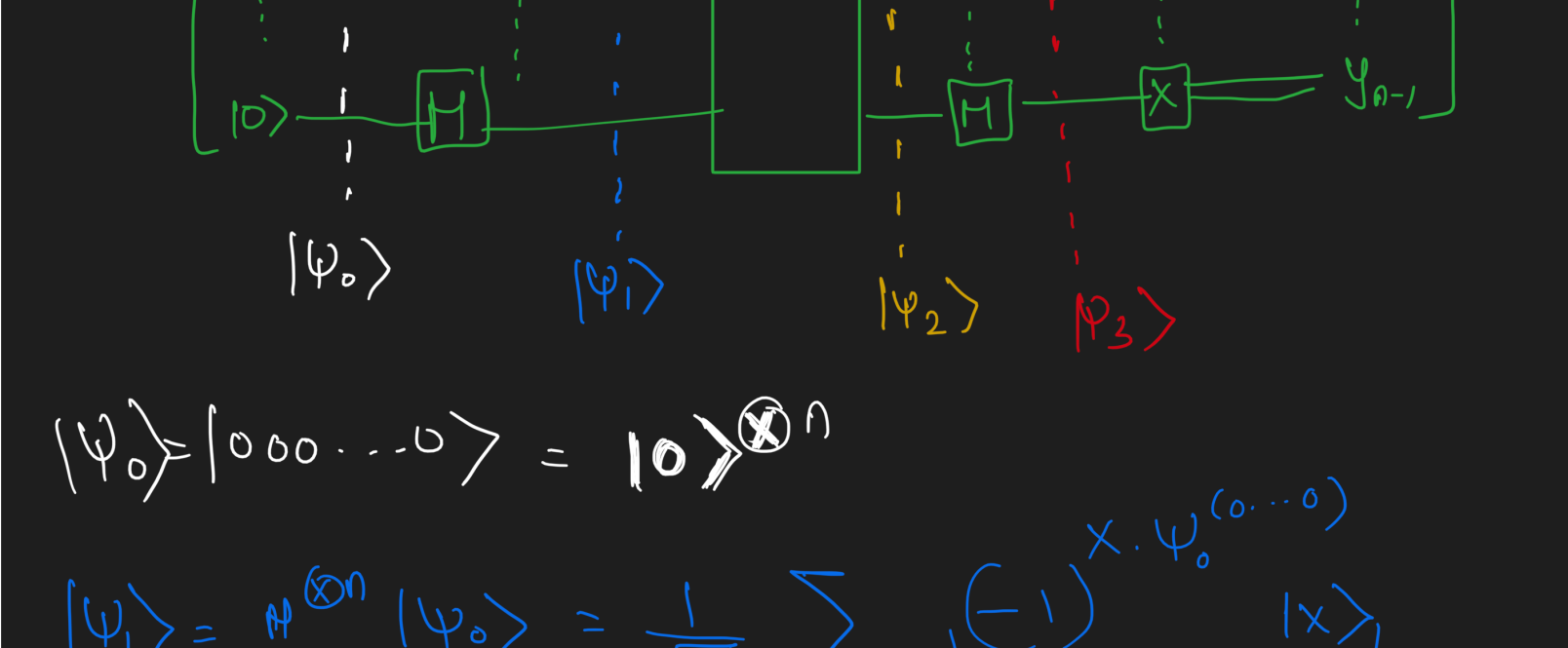
Goal : determine which case we are in ? determine whether f is constant or balanced ?

Classical solution : classical solution, requires to ask the oracle once, Then we get one o/p but then we have no idea whether its constant or balanced. we again ask oracle twice, and we might need to repeatedly ask. Total, up to $\left[\left(\frac{N}{2}\right)+1\right]$ times. at most. $= 2^{n-1}+1$

N= number of realizable strings

n: number of input bits

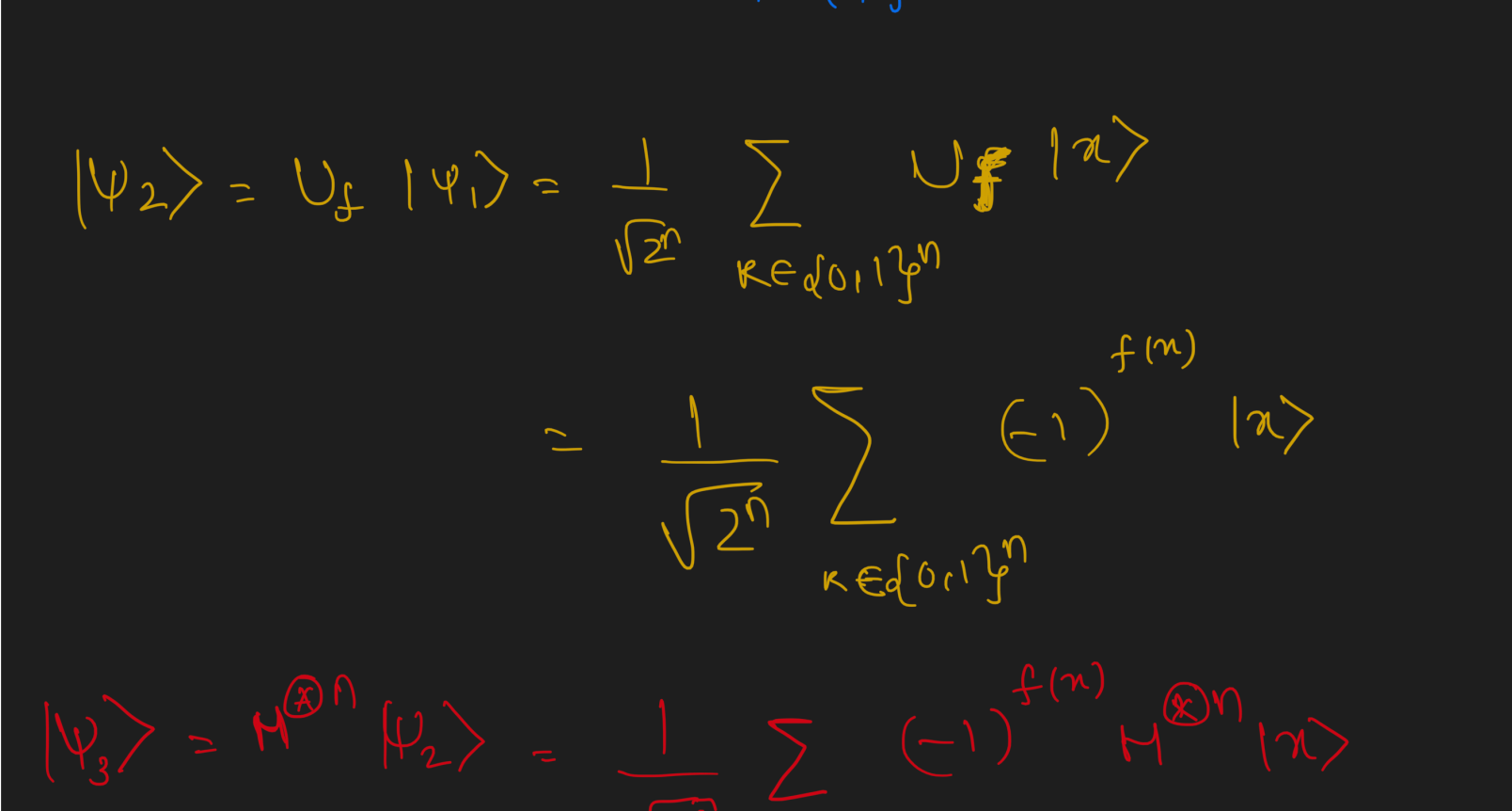
Quantum solution : in this case we only need one query ! we will realize using DJ algorithm circuit.



and now the claim is

Claim : if the outcome y equals the bit string $\underbrace{00\ldots0}_{n}$ then f is constant, for any other output we know its balanced

Proof ! let us check the state after each step :



$$|\psi_0\rangle = |000\ldots0\rangle = |0\rangle^{\otimes n}$$

$$|\psi_1\rangle = H^{\otimes n}|\psi_0\rangle = \frac{1}{\sqrt{2^n}}\sum_{x\in\{0,1\}^n}\underbrace{(-1)^{x\cdot\psi_0^{(0\ldots0)}}}_{+1}|x\rangle$$
$$= \frac{1}{\sqrt{2^n}}\sum_{x\in\{0,1\}^n}|x\rangle$$

$$|\psi_2\rangle = U_f|\psi_1\rangle = \frac{1}{\sqrt{2^n}}\sum_{x\in\{0,1\}^n}U_f|x\rangle$$
$$= \frac{1}{\sqrt{2^n}}\sum_{x\in\{0,1\}^n}(-1)^{f(x)}|x\rangle$$

$$|\psi_3\rangle = H^{\otimes n}|\psi_2\rangle = \frac{1}{\sqrt{2^n}}\sum_{x\in\{0,1\}^n}(-1)^{f(x)}H^{\otimes n}|x\rangle$$
$$= \frac{1}{\sqrt{2^n}}\sum_{x\in\{0,1\}^n}(-1)^{f(x)}\sum_{k\in\{0,1\}^n}(-1)^{k\cdot x}|k\rangle$$
$$= \sum_{k\in\{0,1\}^n}\underbrace{\left[\frac{1}{2^n}\sum_{x\in\{0,1\}^n}(-1)^{f(x)+\vec{k}\cdot\vec{x}}\right]}_{=\text{ constant } C_k}|k\rangle$$
$$= \sum_{k\in\{0,1\}^n}C_k|k\rangle$$ ✓

Sum of all ~~states~~ possible states k with diff Prefactor C_k

whats our probability to measure the zero string ? claim was if we measure 0 string we had a constant function and otherwise balanced function.

=> probability to measure the zero-string $|00\ldots0\rangle$:
$$P\left[(y=000\ldots0)\right] = \left|\langle 000\ldots0|\psi_3\rangle\right|^2$$
inner product and absolute value.

$$= \left|\sum_{k=\{0,1\}^n}C_k\underbrace{\langle 000\ldots0|k\rangle}_{\text{orthogonal to each other}}\right|^2$$
1 for k = 000...

However we know, all the different bit strings will be orthogonal to each other.

$$= \left|\sum_k C_k\underline{\langle 000\ldots0|k\rangle}\right|^2 = \left|C_{000\ldots0}\right|^2$$
$$\begin{cases} 0 & \text{otherwise, else orthogonal} \\ 1 & \text{if } k = 0000\ldots0 \end{cases}$$

instead of summing all k, we just need to take k as 0 bit string and we get $|C_{000\ldots0}|^2$ probability.

$$= \left|\frac{1}{2^n}\underline{\sum_{x\in\{0,1\}^n}(-1)^{f(x)}}\right|^2 = \begin{cases} 1 & \text{if } f \text{ constant} \\ 0 & \text{if } f \text{ balanced} \end{cases}$$

$$\begin{cases} \text{if } f(x)\equiv 0, & +2^n \\ \text{if } f(x)\equiv 1, & -2^n \\ \text{if } f(x) \text{ balanced}, & 0 \end{cases}$$

End