NPTEL

NATIONAL PROGRAMME ON TECHNOLOGY ENHANCED LEARNING

IIT BOMBAY

CDEEP
IIT BOMBAY

Quantum Infromation and Computing

Prof. D.K.Ghosh
Department of Physics IIT Bombay

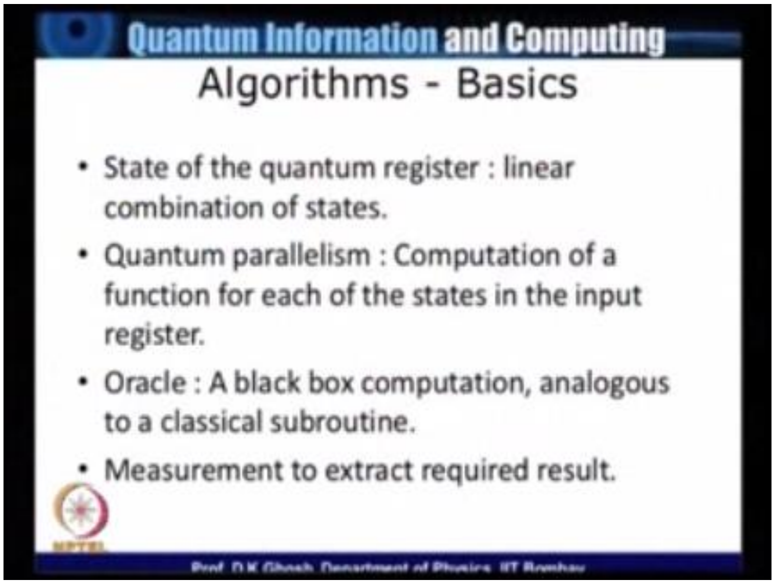
Modul No.04

Lecture No.16

Simple Algorithms- Deutsch Algorithm

In the lectures given so far we have talked about the basic principles of quantum mechanics and also brought up a few ideas about linear algebra which will essentially be used in our next part of the course. What you want to do today is to begin with a few simple algorithms they are simple because they do not bring out any particularly important point, but they establish the fact that quantum computing has certain advantages over the classical computing and for that let us look at what are the very basics.

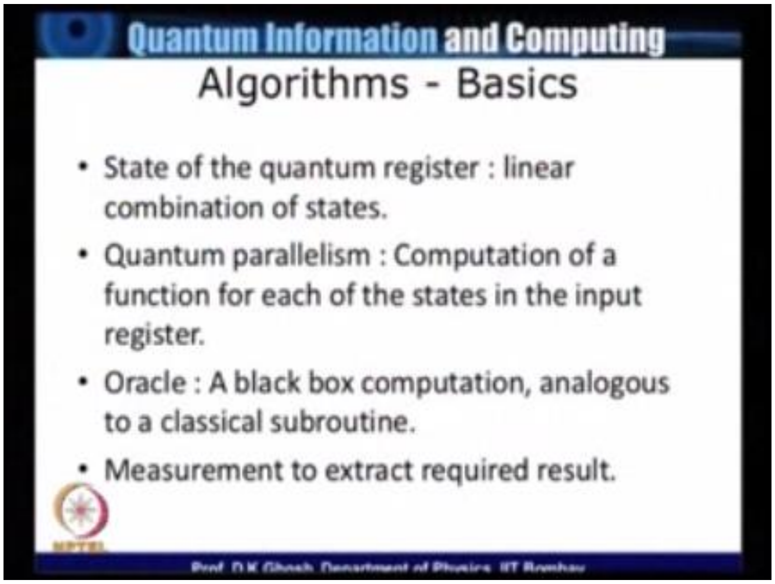
(Refer Slide Time: 01:01)



Of the algorithms that we are going to talk about the first question is this at that like any computer our quantum computer model has quantum registers corresponding to input and those corresponding to output. The difference as we have pointed out several times is that while the classical register at a given time can only contain a particular state, the quantum register can contain a linear computation of states.

And therein lies the great advantage of quantum parallelism, because corresponding to this linear computation of inputs our computer can compute functions, hope corresponding to each one of these linear computation components and the output will be generated by what we have been calling as a quantum oracle.

(Refer Slide Time: 02:07)

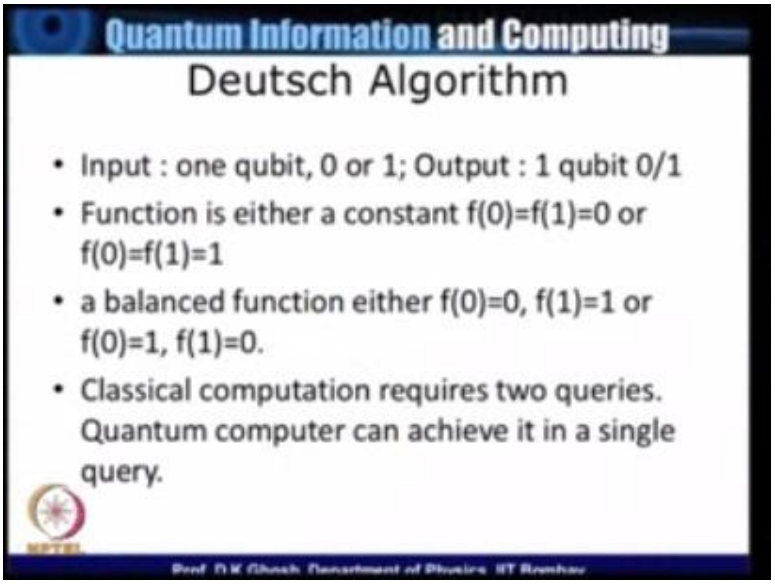


The oracle as we know is like a black box computation very similar to a classical subroutine which we call but do not necessarily detail out in our program. So therefore, a typical model of quantum computation consists of input registers a target register where the output will ultimately be stoled a black box or a quantum oracle which will compute the type of functions that we want so this is basically the constitution of the essential computation.

But after that comes the most important point of the computation namely in any computing we need a read out or a print whatever you feel like, so we have to extract the required information from the output that we got. Now that we have pointed is a very tall order because even though our output is a linear combination of the results corresponding to the linear combination inputs that we have given.

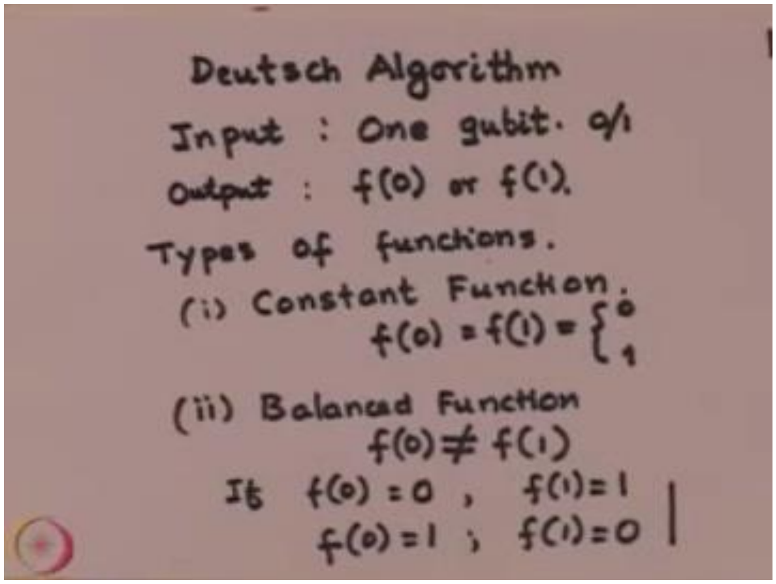
If we want to measure the output there will not be an information about every output corresponding to every input, but one of the random results will come up. As a question all this is that how do you use these random results to your advantage.

(Refer Slide Time: 03:41)



The first algorithm that we talk about is a simple algorithm which is known as the Deutsch algorithm where the input is a one qubit state which is either 0 or1 and the output is also a 1 qubit state which is also either 0 or 1 so as we have said if Deutsch algorithm.

(Refer Slide Time: 04:03)



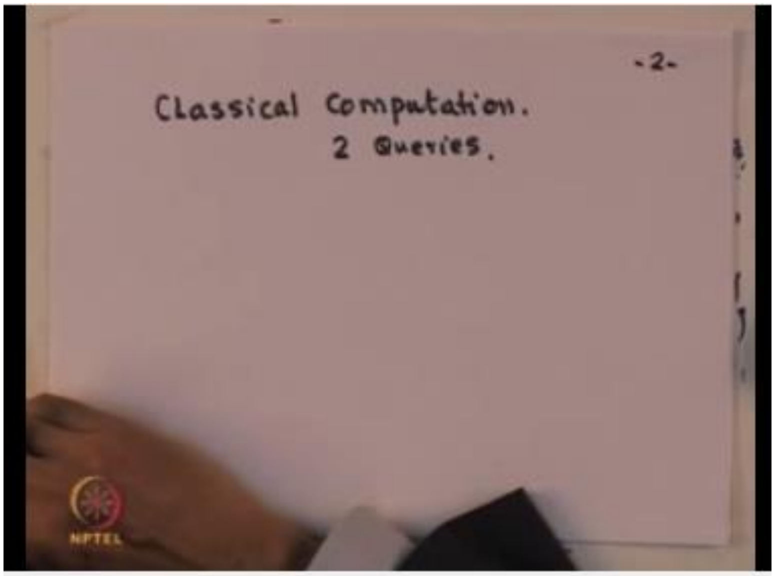
Input is a one qubit which as we have said it could be a 0 or 1 or a linier combination there, and the output which is actually a calculation of a function. So we will calculate f(0) or f(1). Now there are two types of functions that we will be talking about of the types of functions that Deutsch algorithm calculates one will call as a constant function, constant function as we know is a function who is value is independent of the input that we put in, so therefore let us say f(0) equal to f(1) .

And this could be either 0 or 1, so this will be called a constant function. A second type of function will be called a balanced function where f(0) is not equal to f(1), but since our values can be either 0 or 1 it implies if f(0) is equal to 0 then f(1) will be equal to 1 and if f(0) is equal to 1 then f(1) will be equal to 0. Now supposing there is a very trivial programming task supposing I where to do it by a classical algorithm.

How would I determine whether the function that I am calculating which remember I have said is in a black box whether the function that we are calculating is a constant function or a balance function. Now if you are doing it classical the only way you can do it is to first find out what is f(0) get a value which is either 0 or 1 and the find out what is f(1) and check whether the f(1) value that you get now is the same as the value of f(0) that is you got in your pervious step.

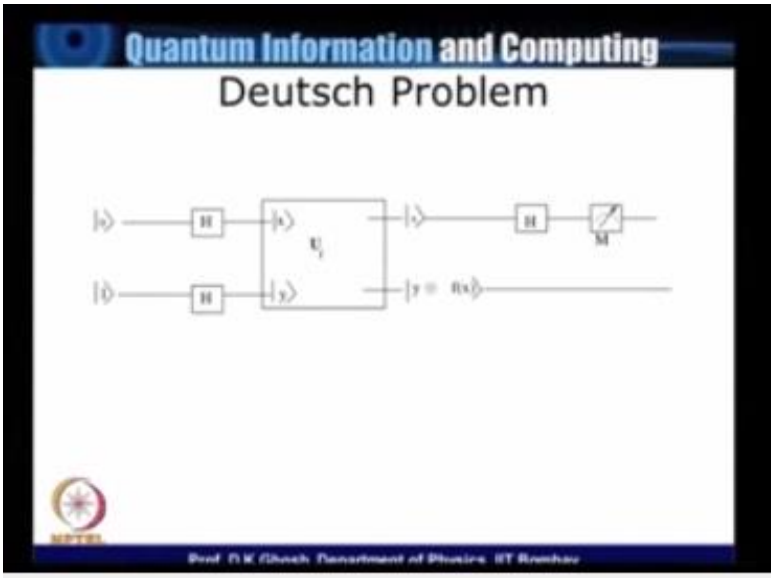
So in other words a classical computation to determine whether the function is a constant function or a balance function requires two queries.

(Refer Slide Time: 07:04)



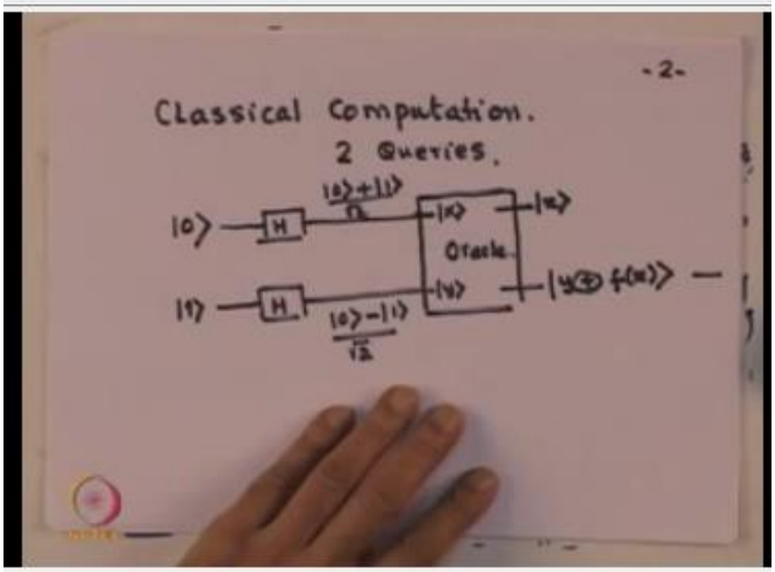
Not a very involved computation so but we require two queries or two steps, so what does the contempt computing do? Is already a trivial problem but there is a simple point we are trying to assume and the point is that in quantum computing I can determine a function is constant or balanced by a single query it is not great deal of advantage because two queries are not because they don't take normally a lot of time. But is the principle of the thing which is important here.

(Refer Slide Time: 07:52)



So let me show you this picture and in this figure you see that there is an input the input state is 0 I will refer back to this figure time and again, so let us look at that.

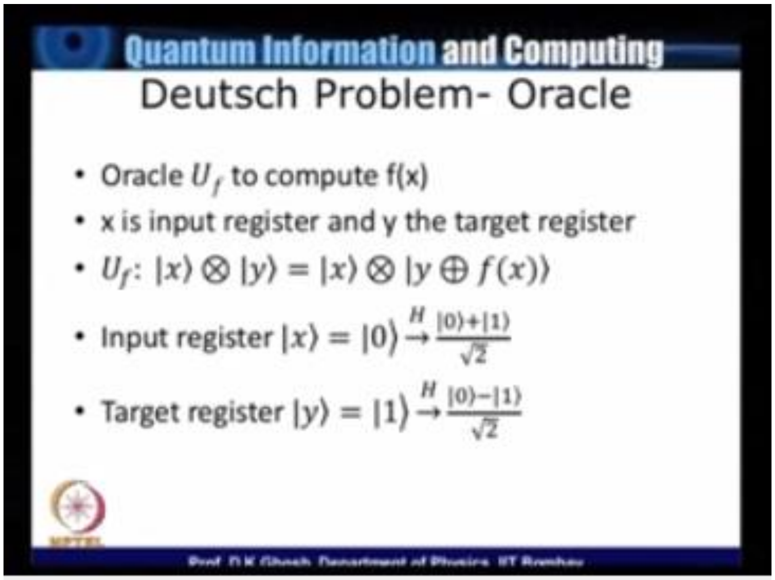
(Refer Slide Time: 08:07)



So if I have an input 0 and I pass it through a Hadamard gate you know that this gives me input 0 pass through a Hadamard gate will give me |0> + 1/√2 this we have talked about several times. Now this is my input to an oracle so this is my oracle, so this is what we have been calling in |x> and as we know that this input after passing through the oracle is not disturbed, it remains the same.

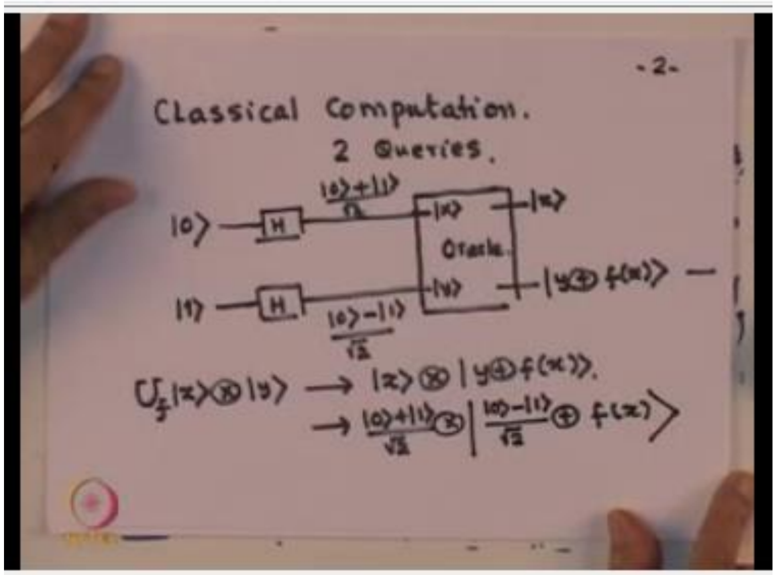
Now the target bit I start with 1 passing through a Hadamard gate again and we had seen earlier that this becomes |0> - 1/√2 this is what we have been calling at the y and the result that you get here in the target register is the addition modulo 2 of y with f(x) which is the oracle calculus, so let5 us look at how does this help.

(Refer Slide Time: 09:29)



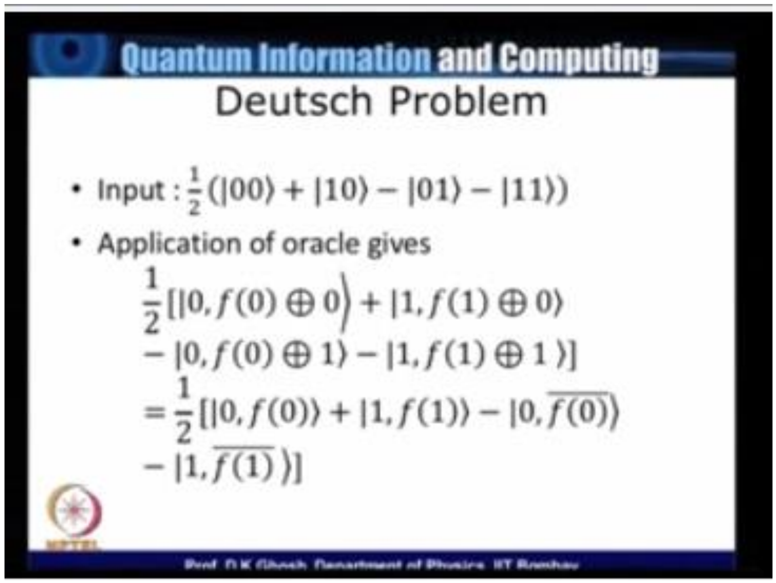
We come back to the question of measurement in a second.

(Refer Slide Time: 09:36)



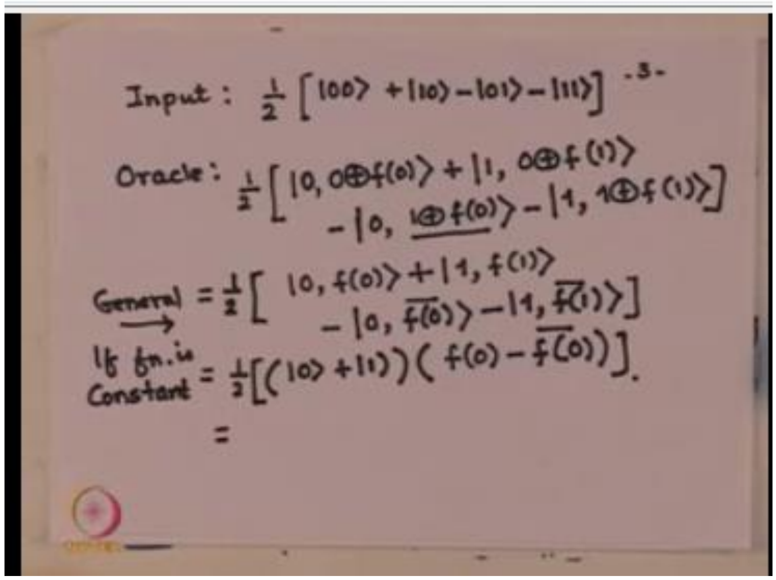
So we have see the input is this, so my input let me write it as |x>, |y> and the what the oracle is doing is to apply a unitary transformation on it which will give me the state x undisturbed and y becomes y + f(x). Now since my input register contains x, my x is going to be |0> + 1/√2 this my input, now let us look at what do I have in the output register this y is |0> - 1/√2 addition modulo 2 with f(x). So let us look at analyze this little more and find out what do I get as a result.

(Refer Slide Time: 10:47)



The slide tells you the situation but I will work it out for our convenience. So if you look at it supposing I expand my input.

(Refer Slide Time: 11:01)

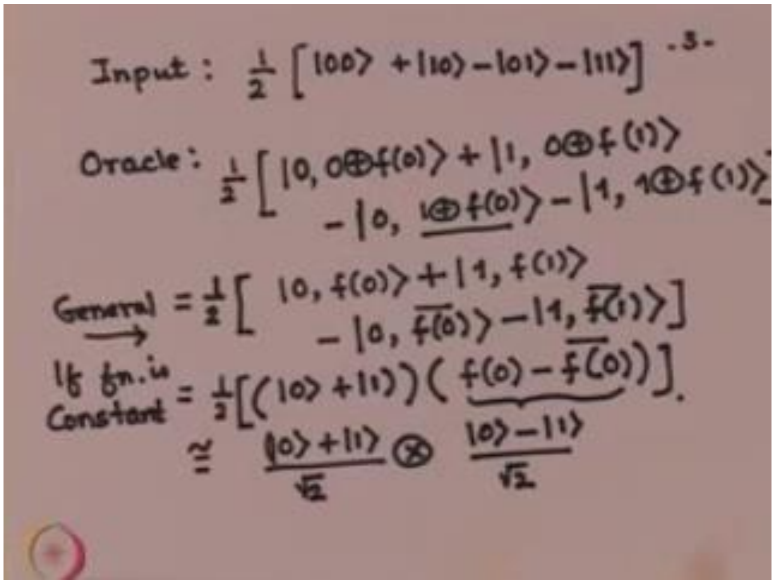


In 2 qubit language then I have (1/√2) x (0+1) x (1/√2) x (0-1) the last one was the y so I get 1/√2 [|00 the first one came from x the second one from y plus |10> - |01> - |11> this is what my input part, now what does the oracle gives? So oracle will give you half remember what the oracle does, it keeps the x bit unchanged the y bit becomes 0+f(x) so which is f(0), so there will be four terms here.

1, 0 + f(1) + – [0, 1 + f(0) – [1, 1+ f(1)] so this will be equal to 1, now you know that when I exude state with 0 I get the same value back, so therefore this is 0, f(0) now so this will be 1, f(1) – here I have exude f(0) as 1 which will give me the compliment of f(0) so let us write that as [f(0) bar – 1, f(1)] so this is what is here. Now this output is if you look at the factors it tells me that I can write it as ½ [|0> + |1>] take common.

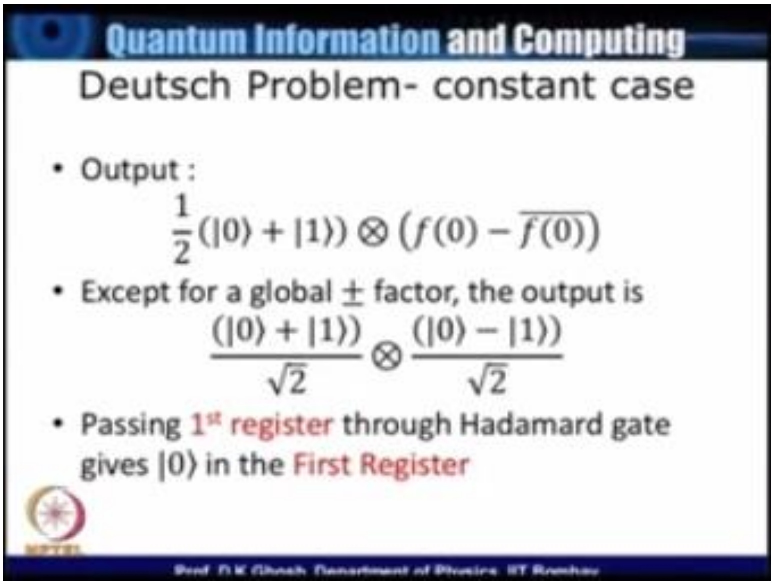
And I am left with then f(0)-f(0) bar, okay. So what I have done here okay this is general case, now what I have done here is I am writing down what happens from the function is constant so if function is constant I have f(0) = f(1). So I have replaced f(1) with f(0) so I get 0+1 times f(0), f(1) bar is same as f(0) bar, so this is what I would get. So look at this now, that since f(0) or f(1) can take values either 0 or 1, f(0)-f bar of 0.

(Refer Slide Time: 14:28)

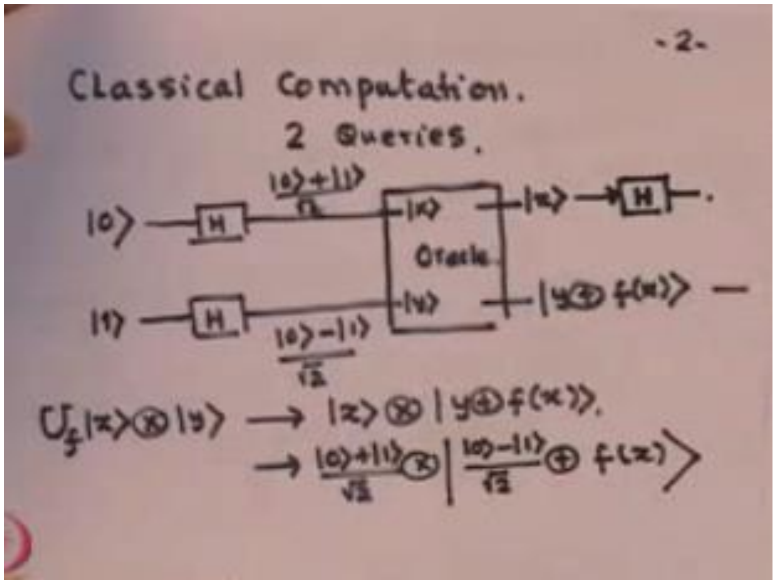


Is either 0-1 or 1-0, so what it means is that this term but for a global phase factor so I will indicate with this is given by |0>+|1>/√2 and |0>-|1>/√2 that is said that this term could be |0>-|1>/√2 or |1>-|0>/√2 but in any case it is a matter of an overall global phase one. So therefore now, the if I were to pass this through a Hadamard gate.

(Refer Slide Time: 15:21)

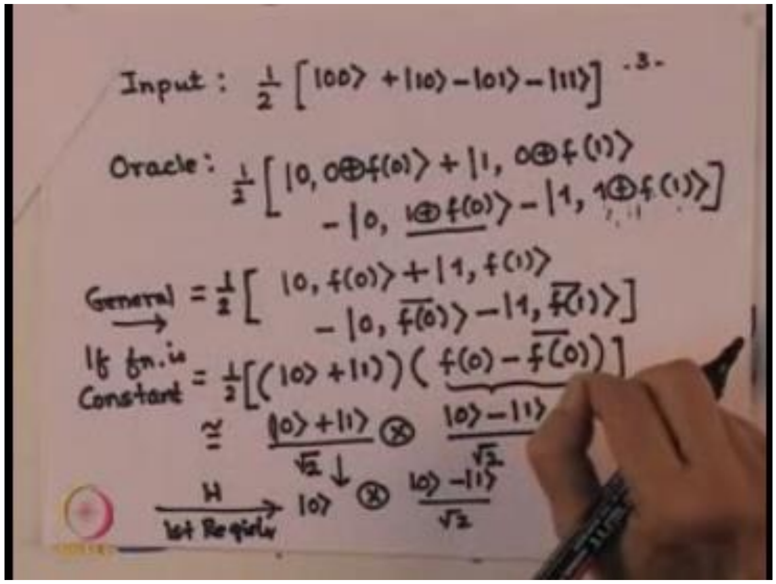


(Refer Slide Time: 15:16)



So let me bring back my picture, so if this is pass through a Hadamard gate the first register is being passed through a Hadamard gate.

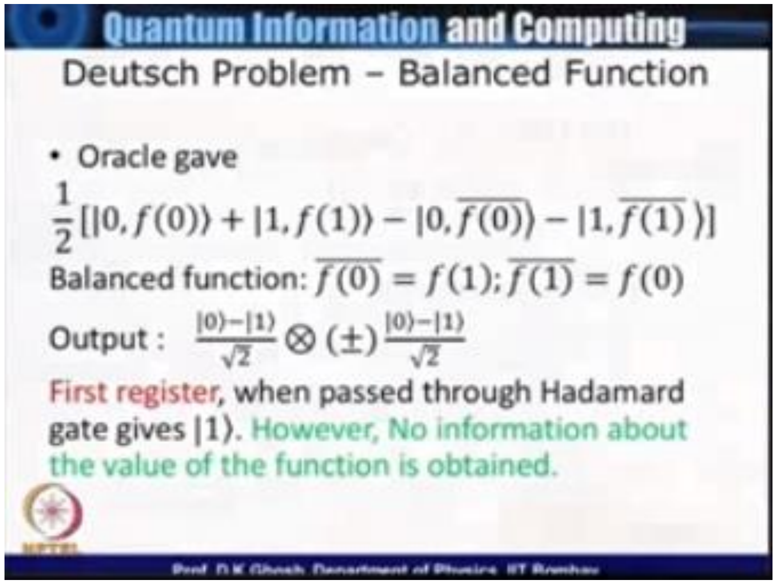
(Refer Slide Time: 15:30)



What will happen if the first register is passed through a Hadamard gate I will get this is |0> and of course this will be |0>-|1>/√2, so if my function where constant and I pass the first register that we Hadamard gate and if I now make a measurement this will give me the first register will be found to be in state |0>. Now us see what happens if my function is balanced.

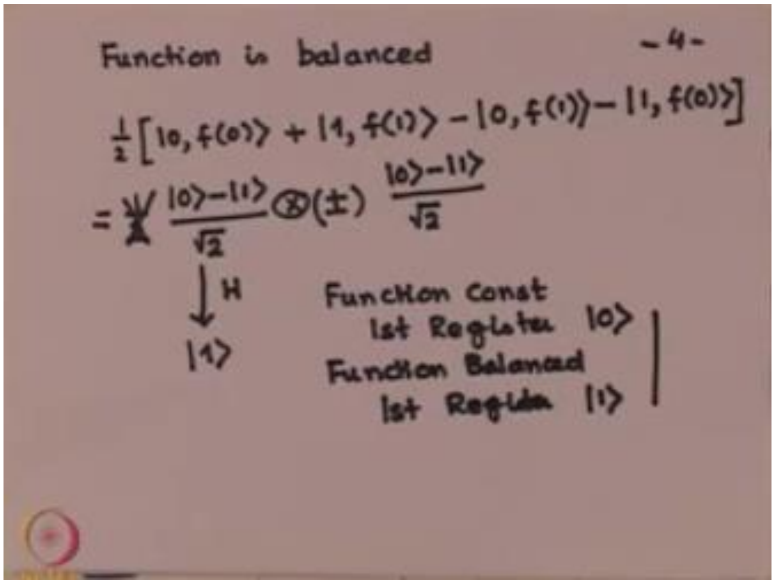
(Refer Slide Time: 16:17)

(Refer Slide Time: 16:29)



So look at what the function gives us if it is balanced just referring back to my general expression.

(Refer Slide Time: 16:36)



I had this expression here, now if it is the balance thing then f(1) is same as f(0) bar and vice versa so therefore, I can write this as 1/2 [|0,f(0)>+|1,f(1)> the first two terms are not changed there the seen as what is written here. Now since if the balance function my f(0) bar is same as f(1) and f(1) bar is same as f(0) so therefore, I write this term as [|0,f(1)>-|1,f(0)>. Notice again that if you look at what this function is, the I take out for example, you take f(0), f(0) you wrote as this is 0,-1.

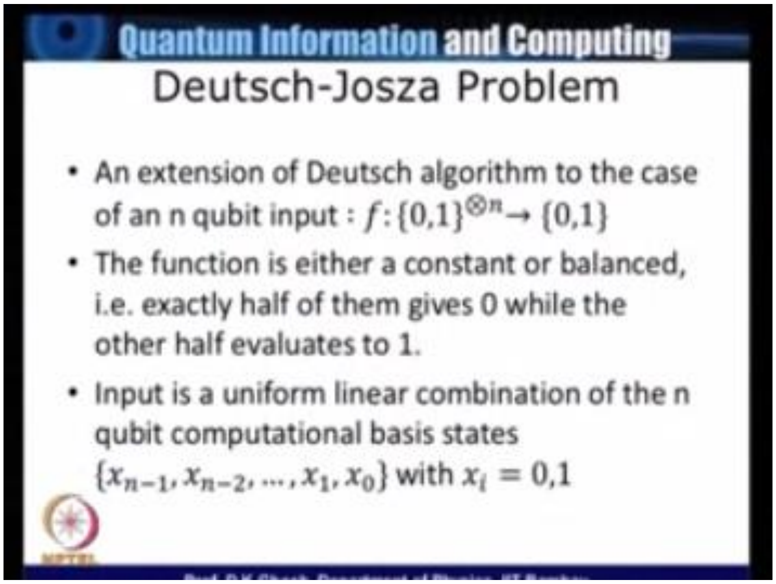
Similarly f(1), f(1) is again 1-0 so thus the output is [|0>-|1>/√2 let me bring that into + or – there is a global phase problem as if I have told you |0>-|1>/√2 global phase as are remain. Now look at what is happening if you pass this first register we Hadamard gate. Since this is |0>-|1>/√2 this is |1> and of course the second register retains whatever it is. So what it tells is this that if I now make a measurement of the first register and the function is balanced I would get the result to be equal to |1>.

So summarizing the result function constant first register gives me |0>, function balanced first register gives you |1>. So therefore, all that I need to do is to measure the first register and if I get a |0> I know that it is a constant function and if I get |1> it is a balanced function. Now that

that seems like 50% reduction in the computing resources that we have reached, there is a flip side to this in the classical computing case where I said that we will need two queries.

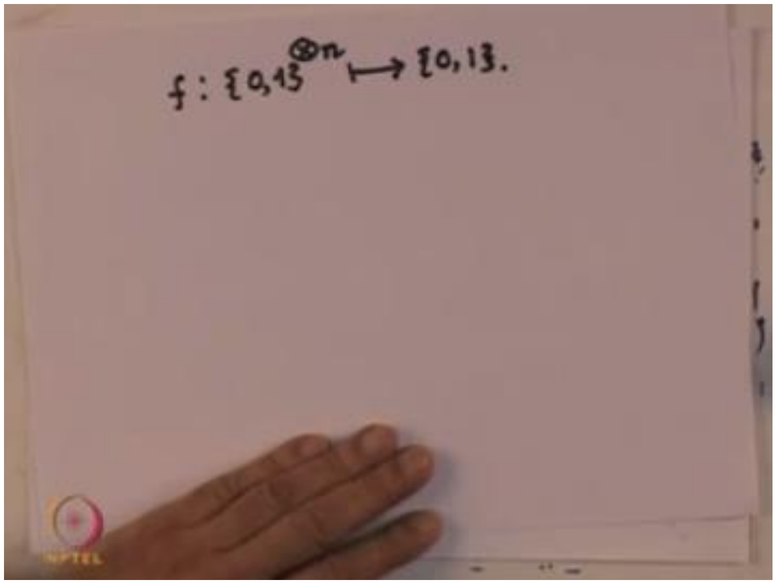
We actually could know what is the value of f(0) and what is the value of f(1), and it is only while comparing them that we could determine whether it was a constant or a balanced function. What happens in this case however is though a single quire is good enough to determine whether the function is balanced or it is a constant function we do not have the information about what the value of the function was, but then that was not our requirement if the requirement was simply to find out whether the function is constant or otherwise then a single quire of a quantum compute is good on. What I will now do is to extend this, is to extend this and introduce the situation there.

(Refer Slide Time: 20:47)



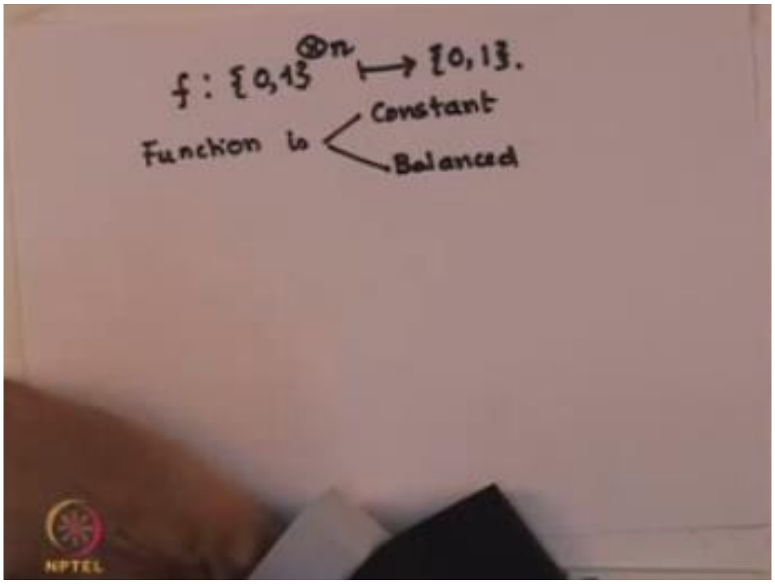
My input instead of the a single qubit input my input is a n qubit input. So therefore, I am interested in functions n qubit input which is represented like this, but the function a value is the single qubit.

(Refer Slide Time: 21:13)



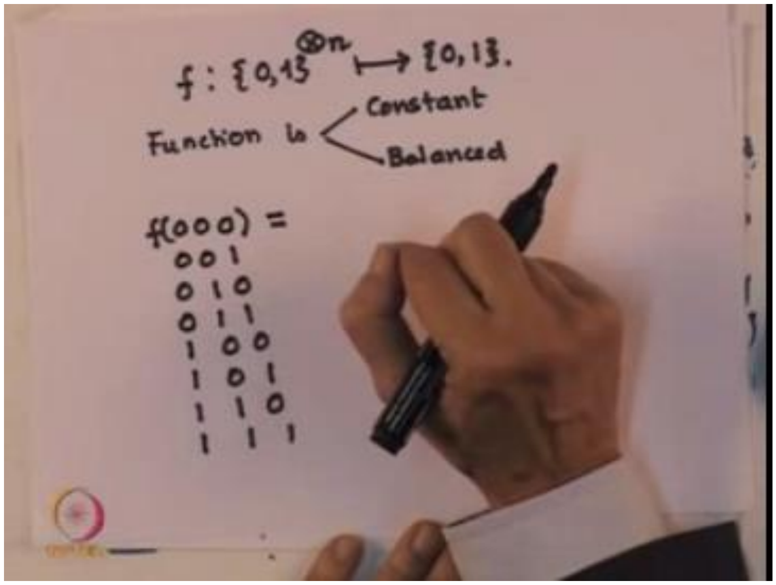
So once again like the Deutsch algorithm I have either a constant function or a balance function. So function is either constant or a balanced constant, the constant function is obviously clear that for all possible inputs the value is constant the same it could be either 0 or 1.

(Refer Slide Time: 21:44)



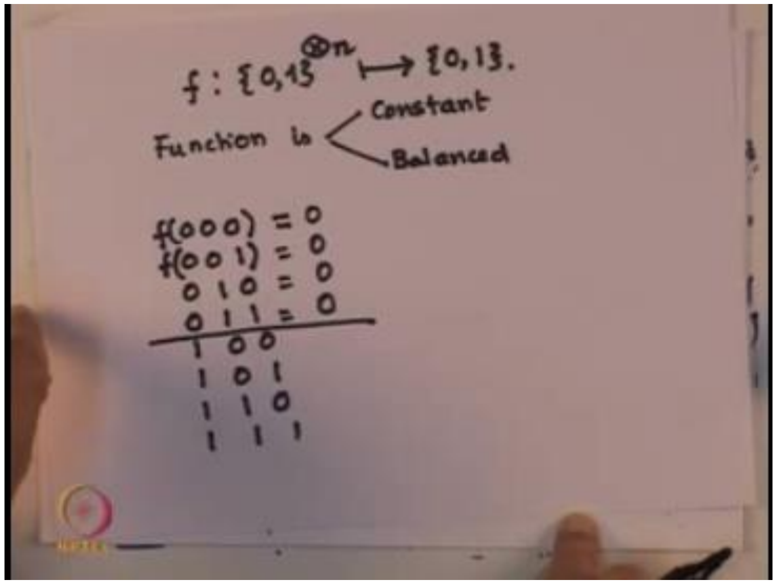
I do not care what actually the value of the function is, but for the balanced case half of the inputs will give you value 0 and the balance half of the input will give you value 1. Once again we do not question as to which input gives 0 which inputs gives one. Now if you look at a classical algorithm to different unit now let us look at what we can do, so what we could do is to start with some input supposing I am just to illustrate supposing I want to find out the situation corresponding to three qubits or three classical bits. Remember that three qubits essentially gives me eight possible inputs.

(Refer Slide Time: 22:44)



So I have 000, 001, 010, 011, 100, 101, 110, and 111. So what you can do is to find out what is f(000) and I will get some value either 0 or 1, then change the input find out what is f(001) having done that find out whether this value is same of the other value. Now if it is the same you still have not reach the computation, if it is not to the same you can stop your computation because we have said there are just two types of function.

(Refer Slide Time: 23:31)



So for example, if f0, 0, is equal to 0 and f 0, 0, f 0, 0, 1 happens to equal to 1 then you know immediately it was a balance function because two of them did not agree, but in principle I could be ongoing supposing this is 0, this is 0, this is 0 this also 0 I still cannot conclude what type of function it was till I step on to the fifth one. Now when I come to the fifth one if it is 0, I stop my computation for the simple reason that it will already proved that it is the constant function.

So therefore the possibilities in the worst case of classical computation you need if is the n qubit unit N/2 + 1 trials, that is the worst case I get in all likelihood you are not going to be that unlucky and find out the result much before N/2+1. So probabilistically it is quite a reasonable statement to make that a classical algorithm there in principle requires of the order of n steps in practice it is likely that will be less.

So therefore in principle it is possible for classical algorithm to require N/2+1 evaluations before we can conclude whether it is a constant function or a balanced function. In the next lecture we will see that this extension of n qubit to 1 qubit function a quantum computer can determine it is nature that is whether it is a balance function or a constant function in a single query.

NATIONAL PROGRAMME ON TECHNOLOGY ENHANCED LEARNING (NPTEL)

NPTEL
Principal Investigator
IIT Bombay

Prof. R.K. Shevgaonkar

Head CDEEP

Prof. V.M. Gadre

Producer

Arun Kalwankar

Online Editor
& Digital Video Editor

Tushar deshpande

Digital Video Cameraman
& Graphic Designer

Amin B Shaikh

Jc. Technical Assistant

Vijay Kedare

Teaching Assistants

Pratik Sathe
Bhargav Sri Venkatesh M.

Sr. Web Designer

Bharati Sakpal

Research Assistant

Riya Surange

Sr. Web Designer

Bharati M. Sarang

Web Designer

Nisha Thakur

Project Attendant

Ravi Paswan
Vinayak Raut

NATIONAL PROGRAMME ON TECHNOLOGY ENHANCED LEARNING (NPTEL)

Copyright NPTEL CDEEP IIT Bombay