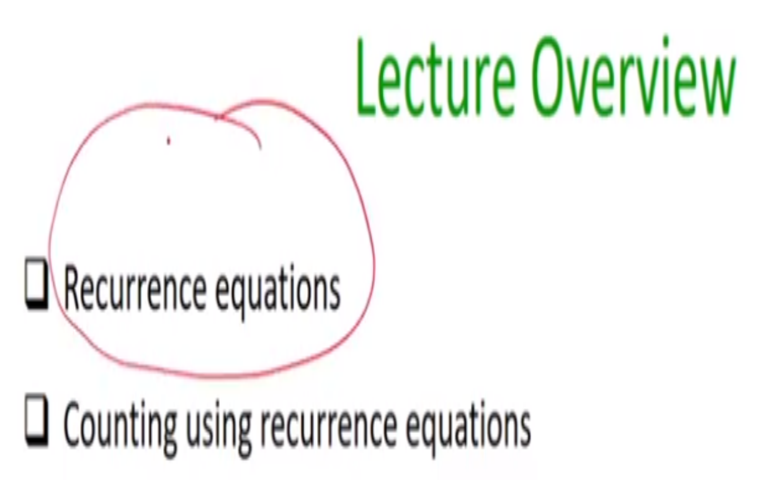
Discrete Mathematics
Prof. Ashish Choudury
Indian Institute of Technology, Bangalore

Module No # 07
Lecture No # 34
Counting Using Recurrence Equations

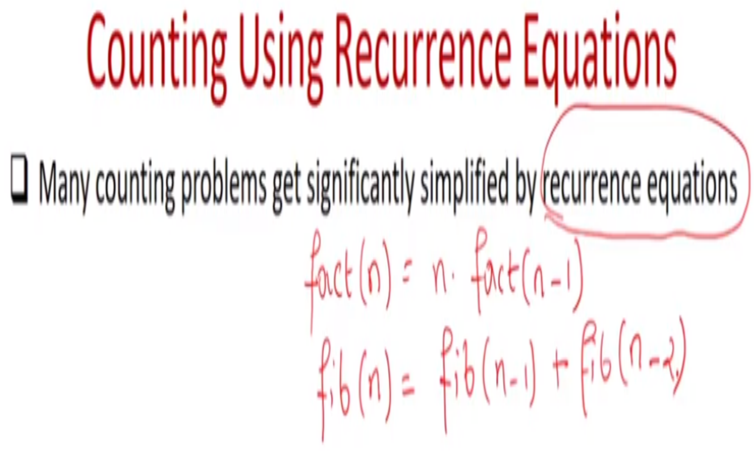
Hello everyone, welcome to this lecture on counting using recurrence equations.

(Refer Slide Time: 00:27)



Just to recap, in the last lecture we discussed about the rules of permutations and combinations used for counting. We also discussed about permutations with reputations and combinations with repetitions. And we also discussed about combinatorial proofs. So in this lecture we will introduce a new counting technique which is extensively used in discrete mathematics and in computer science and this is basically counting using recurrence equations.

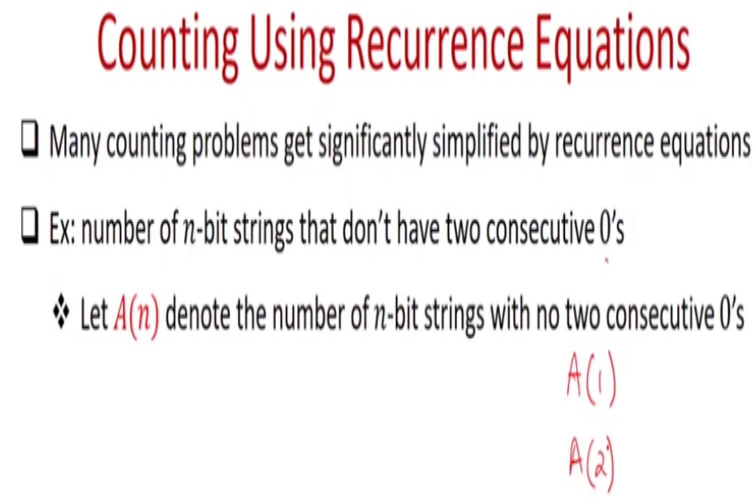
(Refer Slide Time: 00:58)



So it turns out that there are plenty of instances or counting problems which gets significantly simplified by recurrence equations. And to recap, what exactly is a recurrence equation? So you must have encountered different equations of the following form: $fact(n)! = n * fact(n-1)$. So this is a recursive function in the sense that the value of the factorial function on input n is expressed in terms of the value of the factorial function on smaller inputs.

Similarly we are familiar with the famous Fibonacci function where we know that $fib(n) = fib(n-1) + fib(n-2)$. So again this is an example of recurrence equation. So, our idea here will be that now we would like to count the number of things by formulating recurrence equations. And later we will discuss how to solve those recurrence equations.

(Refer Slide Time: 02:21)

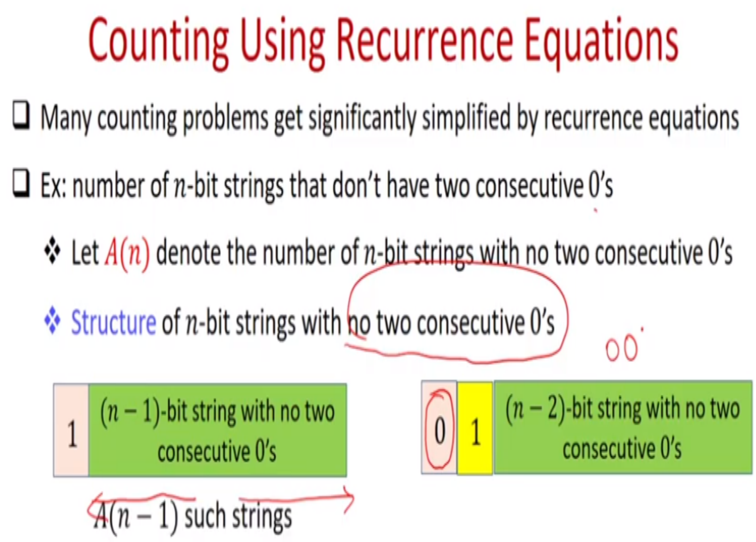


There are many counting problems which gets significantly simplified by formulating the recurrence equations. So let us see an example here to clarify my point. So imagine I want to find out the number of n bit strings that do not have an occurrence of 2 consecutive 0's. That means the substrings 00 is not allowed to appear in such a string and we want to find out how many such strings can be there.

And we want the general answer namely we want to find the number of such strings for any n. So the way we are going to count this is as follows. We say that let $A(n)$ be a function which on input n gives you the number of n bit strings which do not have two consecutive 0's. That is the definition of my $A(n)$ function. So if I say $A(1)$ that will give me the number of bit strings of length 1 that will not have the occurrence of 2 consecutive 0's.

$A(2)$ will give me the value of number of strings of length 2 that do not have occurrences of 2 consecutive 0's and so on. That is a definition of my function $A(n)$. Now we want to find out what exactly will the $A(n)$ function look like or what will be the output of the function A on input n. And I want to set up a recurrence equation for that. That means I would like to express the value $A(n)$ in terms of the output of A function on small size inputs.

(Refer Slide Time: 04:00)



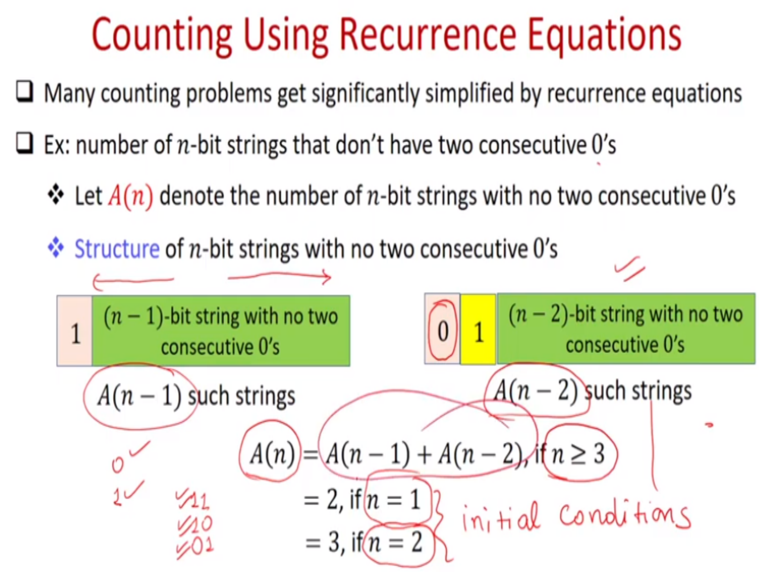
So for that what we are going to see here is the following. If we consider any n bit string which do not have 2 consecutive 0's then there are only 2 possibilities for the starting bit of such a string. The starting bit of such a string could be 1 in that case the remaining $n-1$ length substring should not have any occurrence of 2 consecutive 0's. Because if 2 consecutive 0's occur anywhere in the remaining substring of length $n-1$ then the overall string cannot be a valid string of length n bits. And without having any occurrence of 2 consecutive 0's that is not going to happen. So that is one possibility.

The second possibility could be that the string of length n starts with 0. If the string starts with 0 and if we want that overall string should not have any occurrence of 2 consecutive 0's. Then definitely the second position of that string should be 1. Because if the second position of such a string is also 0 and anyhow the first position is 0 then that is a violation. Violation of the property that the string has no occurrence of 2 consecutive 0's so definitely the second bit position has to be one.

And now what can I say about the remaining $n-2$ bits in this string. I can definitely say that the remaining $n-2$ bits in the string should not have any occurrence of 2 consecutive 0's. Because if 00 occurs or appears anywhere in the remaining substring of length $n-2$ then the overall string cannot be considered as a valid string of length n.

And now you can see that these are the only 2 categories of string of length n: either the string can start with 1 or the string can start with 0. So now let us try to count the number of strings in both the categories. If I consider category 1 then the number of strains of this category is $A(n-1)$. Why so?

(Refer Slide Time: 06:30)



Because I am interested to count the number of strings of length $n-1$ which do not have an occurrence of two consecutives 0's and as per the definition of my A function the number of such string is nothing but $A(n-1)$. You take any string of length n – 1 which do not have an occurrence of two consecutive 0's and you put a 1 at the beginning of such a string that will give a valid string of length n that do not have an occurrence of 2 consecutive 0's.

So that is why I will have $A(n-1)$ number of strings in this category. And how many strings in the second category? My claim is that the number of strings in this category is nothing but $A(n-2)$. Because as, per the definition of my A function the number of strings of length $n-2$ which do not have an occurrence of two consecutive 0's is $A(n-2)$. And you take any string of length $n-2$ bits which do not have an occurrence of two consecutive 0's and you put 0 and 1 at the beginning of such a string that will give you a valid string of length n bits which do not have an occurrence of two consecutive 0's. And as I said these are the only 2 categories of strings of length n which do not have an occurrence of two consecutive 0's in terms of that the number of strings of length n will be the output of A function on input $n-1$ and the output of A function on the input $n-2$ i.e., $A(n) = A(n-1) + A(n-2)$.

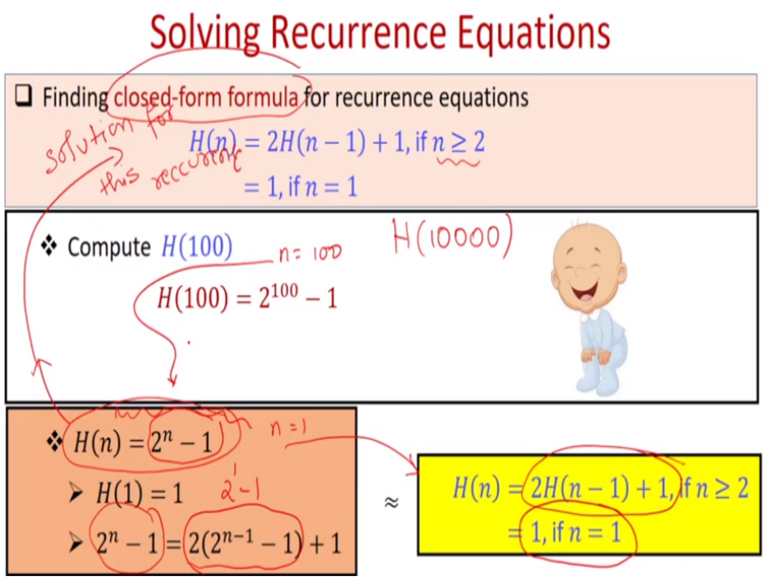
However the argument or the discussion that we had holds only if $n \geq 3$. Because if I take $n = 2$, then this argument would not hold because we $A(0)$ upon substitution. A(0) does not make any sense since, as per the definition of my A function, it is the number of strings of length 0 which do not have occurrence of two consecutive 0's.

So that is why this definition of this recursive function holds for all $n \geq 3$. So now you might be wondering what about the output of the A function on inputs which are less than 3. So we will be giving some initial conditions. They are called initial conditions because the recursive functions are not applicable for the case when $n = 1$ and $n = 2$.

Because if $n = 1$ then the number of bit strings of length 1 which do not have an occurrence of 2 consecutive 0 is 2. Because both the strings, 0 as well as the string 1, will be considered as valid string. There is no occurrence of consecutive 0's in this string, there is no occurrence of consecutive 0's in this string. Similarly, if I consider $n = 2$ then the possible strings of length 2 which do not have occurrences of 2 consecutive 0's are 11, 10 and 01.

These are the valid string and how many strings you have? 3. That is why $A(2) = 3$. But for any $n \geq 3$ I can find out the value of A of that particular n by using this recurrence equation.

(Refer Slide Time: 10:07)



So now we know how to count using recurrence equations. The next thing that we would like to do now is how to solve those recurrence equations? And when I say I want to solve a recurrence equation I mean finding a closed-form formula for that recurrence equation. So what exactly that means? So, suppose someone gives me this recurrence equation. The recurrence equation is $H(n) = 2H(n-1) + 1$ for all $n \geq 2$ and initial condition is that $H(1) = 1$.

Now, if suppose someone asked me can you find the value of this H function on the input 100. I can say well that is not difficult. As per my relation $H(100)$ will be $2H(99) + 1$ and $H(99)$ will be $2H(98) + 1$ and so on. But then this is a time consuming affair. I would not be able to do this or solve this very quickly using paper-pen. So that is a difficult thing. So in fact $H(100)$ you still might be able to do but what if I ask to compute say $H(10000)$.

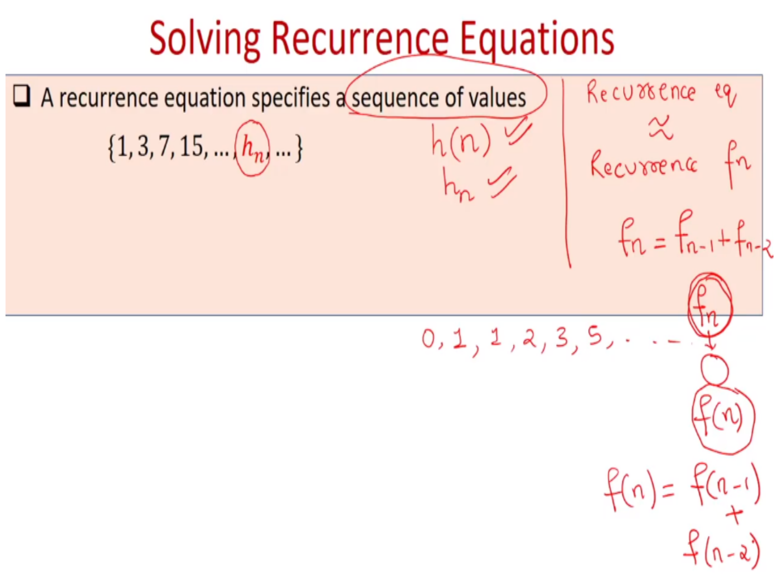
Then you will be stuck; you cannot find out the value of $H(10000)$ very quickly. Now suppose I tell you that $H(n) = 2^n - 1$. We will verify if this is indeed the case or not. Now, as per this formula $H(1) = 2^1 - 1 = 1$ and it is also easy to verify that $2^n - 1 = 2(2^{n-1} - 1) + 1$. That means I can say that this H(n) function satisfies this recurrence condition. Satisfies in the sense it has all the properties or characteristic that are specified by the recurrence equation.

If I substitute $n = 1$ here I get the same initial condition as given in the recurrence function and indeed this $2^n - 1$ is a function where if I substitute n being $n - 1$ then I get this recurrence condition. That means this $H(n)$ function will be considered as a solution for this recurrence equation. And why it is called as solution for this recurrence? Because it is a closed-form formula. Closed-form formula means it is just a function of n. Here $H(n)$ does not depend on the value of H function on previous inputs.

It is a function of variable n. You just substitute the value of n you will get the answer. That means, now, if I ask you what will be the value of $H(100)$ you do not have to follow the complicated process. That means for computing $H(100)$ you need not have to compute $H(99)$, you need not have to compute $H(98)$, you need not have to compute $H(97)$ and so on. You will just go and substitute $n = 100$ in this function and get the answer and that is all.

That is what we mean by solving a recurrence equation. So when I say I want to solve a recurrence equation by that I mean I am interested to find out a closed-form formula for that recurrence equation. Namely a function; in n which satisfies that the recurrence condition.

(Refer Slide Time: 14:11)



So now we will discuss about the general methods of solving recurrence equations. It depends upon the type of the recurrence equation. It is not the case that the same method of solving the recurrence equation will be applicable for every category of recurrence equation. So first of all you have to understand here that a recurrence equation basically specify a sequence of values.

So the terms *recurrence condition*, *recurrence equation, recurrence function* etc., will be used interchangeably. All these are equivalent terms. So whenever I am specifying a recurrence equation basically I am talking about an infinite sequence of values. And the recurrence equation basically specifies the characteristic of the n-th term of that sequence.

That n-th term you might either represent as $H(n)$ where H is a function or you might omit the parentheses and you can instead write down n in the subscript. Both these notations are equivalent. So, for instance we know that Fibonacci sequence is defined as follows: the first term is 0, the second term is 1, and then after that the next term is the summation of the previous 2 terms.

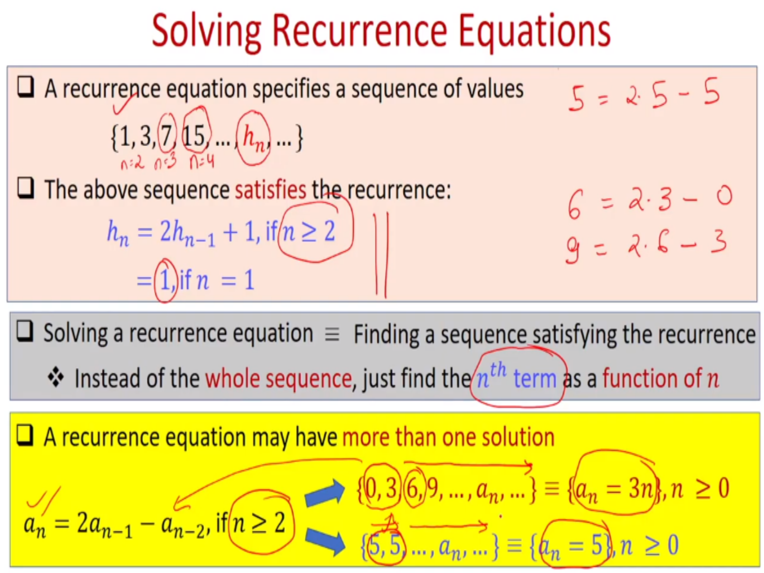
If I somehow want to specify the n-th term of this sequence, I can use 2 notations. I can either use the notation f_n to denote the n-th term of this infinite sequence. So remember this Fibonacci sequence is an infinite sequence. Or I can use the function $f(n)$; both means the same thing. If I am using this notation f_n then I will use this notation: $f_n = f_{n-1} + f_{n-2}$ to denote that the n-th term of the sequence is related to the previous 2 terms of the sequence.

Whereas if I am using the function notation for denoting the n-th term of this infinite sequence then I will characterize this property by saying that $f(n) = f(n-1) + f(n-2)$. Both these interpretations are equivalent.

And that is why for the rest of our discussion on recurrence equation I will be interchangeably using the terms recurrence function or the n-th term of a sequence and so on. So whenever we are talking about a recurrence function we are talking about an infinite sequence and we are interested to specify the property of the characteristic of the n-th term of that sequence by saying how exactly it is related to the previous terms of the same sequence.

So somehow someone gives me this infinite sequence; now I want to characterize the property of this sequence.

(Refer Slide Time: 17:48)



That means what exactly is the characteristic of the n-th term of this sequence. So it is easy to see that the n-th term of this sequence satisfies the following characteristics. The first term of the sequence is 1 and if you take the n-th term for any $n \geq 2$, it is twice the $(n-1)$-th term, that means twice the previous term plus 1. So for instance if you take 7 where $n = 3, 7 = 2 * 3 + 1$ i.e., it is one more than twice the previous term. Similarly, the fourth term $15 = 2 * 7 + 1$ i.e., it is one more than twice the previous term.

So that is what is the interpretations of a recurrence equation. If someone gives me a recurrence equation and does not give me the infinite sequence; solving a recurrence equation basically means finding the sequence which satisfies that recurrence condition. So, someone tells me "solve this recurrence equation" without telling me the infinite sequence then my goal will be to find out that infinite sequence.

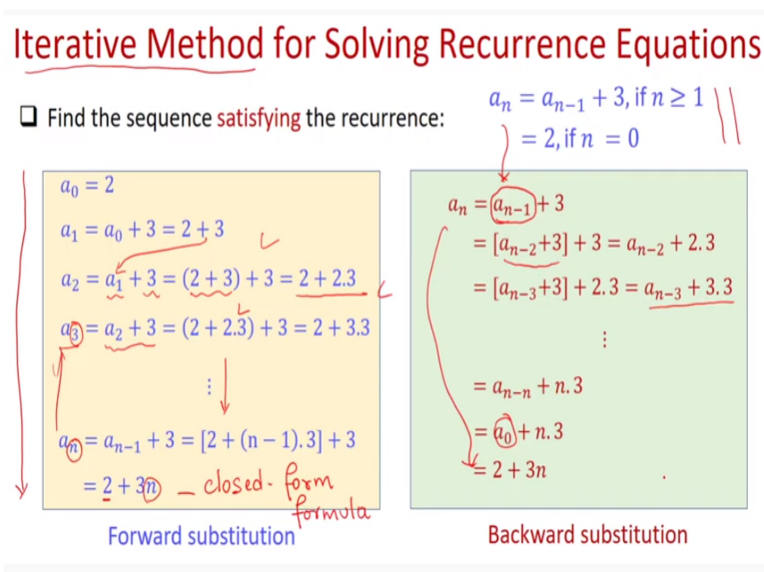
But I would not be finding the whole sequence because it is infinite, instead I will just find out the closed-form formula for the n-th term of that sequence as a function of n. That is what I mean by solving a recurrence equation. It turns out that a recurrence equation may have more than 1 solution. So for instance if I take this recurrence equation namely I am talking about an infinite sequence where the n-th is the difference of twice the previous term and the previous-to-previous term.

For all $n \geq 2$, this sequence has this recurrence condition being satisfied. Now my goal is to find out a possible infinite sequence whose n-th term has this characteristic for every $n \geq 2$. So I will now show you 2 possible sequences. So consider this infinite sequence whose n-th term is $a_n = 3n$. You can easily verify that this infinite sequence satisfies this recurrence condition. You take any term, say 6, it is indeed twice the previous term minus the previous-to-previous term.

So $6 = 2 * 3 - 0$. The term $9 = 2 * 6 - 3$ and so on. So, this is one of the infinite sequences which satisfies this recurrence condition. Now you take another infinite sequence namely the sequence where all the terms are 5 i.e., $a_n = 5$. This is also a sequence where the values in the sequence are completely different from the values in the upper sequence.

But the second sequence also satisfies the recurrence condition. You take any term of the sequence it will be 5. And 5 is indeed twice the previous term which is also 5 minus previous-to-previous term which is also 5. This shows that if someone gives me a recurrence condition then there can be multiple solutions or multiple sequences satisfying that recurrence condition.

(Refer Slide Time: 21:27)



So we will kick start our discussion with some simple methods of solving recurrence equations. We will discuss advanced methods in our later lectures. So we will discuss the simplest method called as the iterative method. What exactly we do in the iterative method? So consider this recurrence equation. I am supposed to find out an infinite sequence whose zeroth term or the starting term is 2 and from the next term onwards the term is the pervious term plus 3.

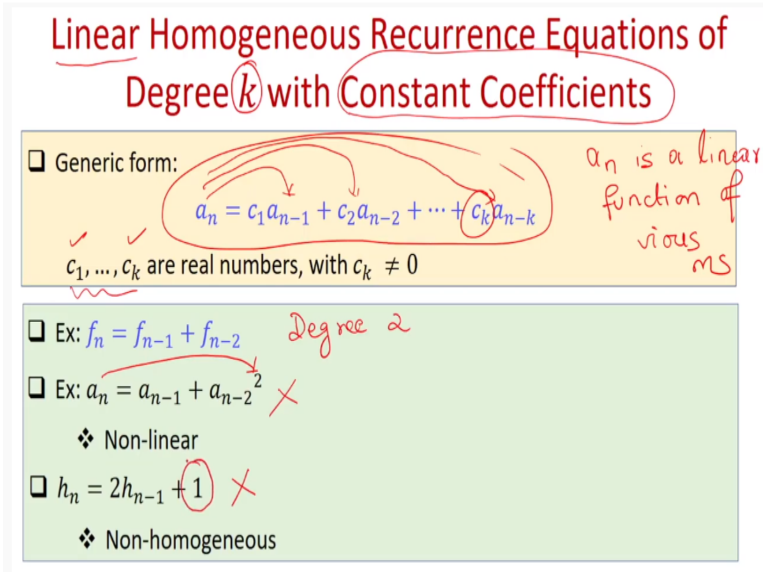
So I have to find one such sequence satisfying this condition. And when I say I want to find out the sequence as I said I would not find the entire sequence but rather I will find a closed-form formula characterizing the n-th term of that infinite sequence. So this is how we can do that using iterative method. So we starting with a_0 which is the starting term. It is given to be 2. Then we will say that as per our recurrence condition the second term of the sequence will be the first term plus 3. And now we have already obtained the value of $a_1 = 2 + 3$, so I am substituting that. And if I substitute overall then I get $a_2 = 2 + 2 * 3$. And now if I want to find out a_3, again I will apply the recurrence condition. a_3 will be summation of a_2 and 3. a_2 I have already obtained so I can substitute here and now you see that I start getting a pattern here that $a_n = 2 + 3n$. That is the general pattern that I am getting here and that is why if I solve and I keep on solving then I will get the fact that $a_n = 2 + 3n$. And this is now a closed-form formula. Why it is a closed-form formula? Because, here the value of $a_n$is just a function of n.

You substitute $n = 2$, $n = 3$; you get whatever we have derived till now. So this will be the solution of this recurrence equation. I can get the same closed-form formula as follows. So here what I did I start with the initial condition and derived the n-th condition. I can do it in the reverse direction as well. I will say that as per the recurrence condition a_n is this, then I apply the recurrence can be condition again on a_{n-1}.

I will say that as per the recurrence condition $a_{n-1} = a_{n-2} + 3 + 3$ and $a_n = a_{n-2} + 2 * 3$. Now I will again solve a_{n-2} by applying the recurrence condition and I will get the value of a_n in terms of a_{n-3}. And now if I continue like this I'll obtain the initial condition which is a_0 and if I stop at a_0 then I will get a closed-form formula for a_n.

So now you can see that it does not matter whether I go from a_0 to a_n or whether I go from a_n to a_0. In both the cases I obtain the same closed-form formula. The first method here is called the *forward substitution* iterative method. The second process here is called as the *backward substitution* iterative method. So this is one of the simplest methods of solving recurrence equations.

(Refer Slide Time: 25:37)



Now the recurrence equation that we considered here can be categorized as a *linear homogenous recurrence equation of degree* k *with constant coefficient*. And why we are interested in this class of recurrence equations is that we will next see a generic method of solving this category of recurrence equations. So what exactly is this category of linear homogenous recurrence equations of degree k with constants coefficients.

So there are multiple terms here so let us, decode each of them one by one. The general form of recurrence equations in this category is the following. You will have a_n being expressed as the linear function of previous terms. So that is why the name linear. So you can see that here a_n here is a linear function of previous terms. What is the linear function?

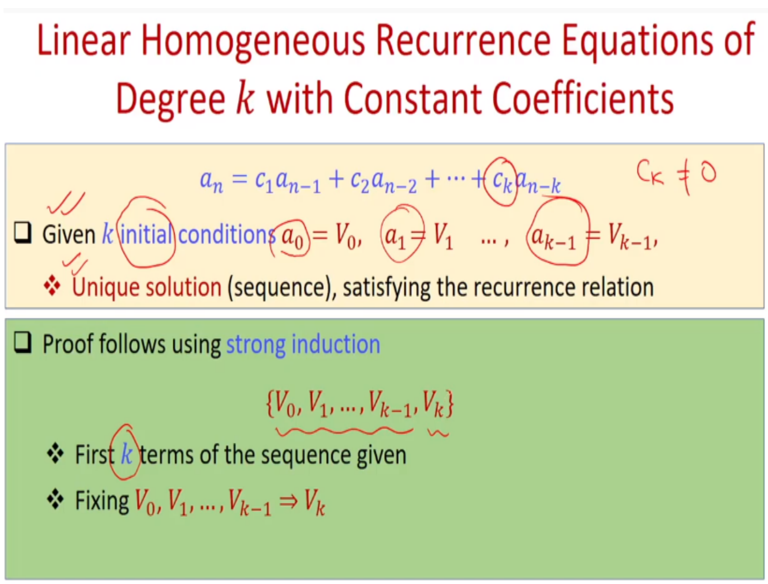
Because the combiners namely c_1, c_2, c_k are constants here and a_n depends linearly on previous term. Namely, it depends only the first power of a_{n-2}, only the first power of a_{n-2}, first power of a_{n-k} and so on. Why it is called homogeneous because the dependency of a_n is only on the previous terms and nothing else. We will see what exactly non-homogenous means very soon.

Why it is called degree k? Because c_k is not allowed to be 0 here. If $c_k = 0$ then a_n does not depend on a_{n-k}. In that case the degree will not be called k. $c_1, c_2 \ldots c_{k-1}$ are allowed to be 0. But c_k cannot be 0 if the degree is k. That is what the degree means here. It is similar to what we say in the context of polynomials. If I say I have a polynomial in x of degree t then the coefficient of x^t should be non-zero. The coefficients of other powers of x may be 0. But since the degree is t

the coefficient of x^t cannot be 0. In the same way when I say degree k this c_k is not allowed to be 0. And the constants coefficients basically say is that my combiners are constants here; c_1 to c_k.

So let us see some examples of this category of recurrence equations. Your well-known Fibonacci function or Fibonacci recurrence equation falls in this category. Here the degree is 2. Because $f(n)$ depends on the previous 2 terms. Now the equation $a_n = a_{n-1} + {a_{n-2}}^2$ is not falling in this category. The problem is that a_n does not depends linearly on a_{n-2}. It depends quadratically on a_{n-2}that is why it is a non-linear function. Whereas this third function or third equation is not falling in this category because we have a non-homogenous component here.

(Refer Slide Time: 29:35)



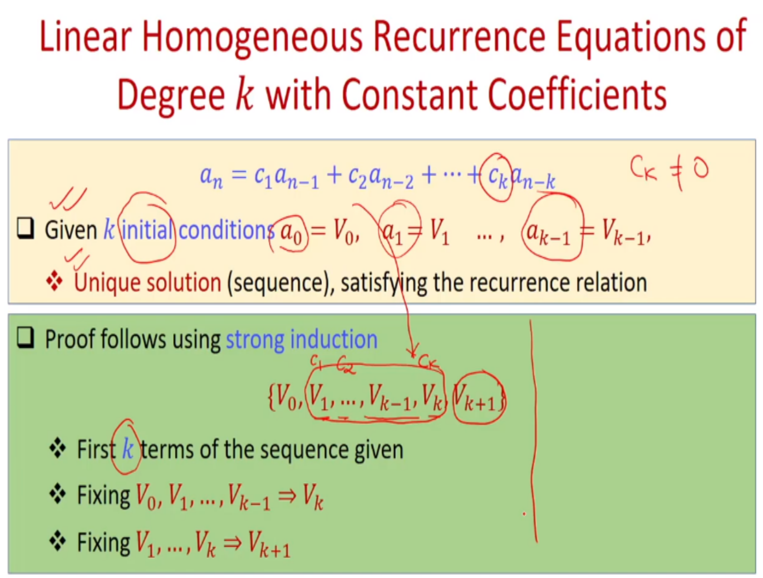
Now the next thing that we want to show here is that, I said few slides back that if you are given a recurrence equation then there might be multiple solutions for that recurrence equation and we saw an example as well. Now the question is, is it possible that a given recurrence equation has a unique solution. Well it is possible provided you are given initial conditions as well. More specifically, imagine you are given a linear homogenous equation of degree k. That means c_k is not 0.

Then my claim is that if you are also given k initial conditions, that means you are given the value of say a_0, a_1 and a_{k-1}, if these values are explicitly given to you which we call as initial conditions then there always exists a unique solution satisfying the recurrence condition as well as the initial conditions. See, the example that we considered, there the initial conditions were not given. You just were given the recurrence equation and that is why we found 2 different sequences satisfying the same recurrence conditions. But what I am saying here is that apart from the recurrence equation you are also given the initial conditions and my claim is that only when you are given those many initial conditions which are same as the degree of your recurrence equation then you always have a unique solution.

And this can be proved very easily using strong induction. So my claim is that there is only a unique solution satisfying the initial conditions as well as the recurrence condition. This is because since you are given the initial conditions namely the k initial conditions the first k terms are explicitly given to you. You cannot substitute the first k terms arbitrarily. Now since the first k term are given to you, I can say that by applying the recurrence condition on the first k terms I get the next term.

The next term cannot be an arbitrary term because my reference condition explicitly says that what will be the term V_k once I have frozen $V_0, V_1 \ldots V_{k-1}$.

(Refer Slide Time: 32:00)



In the same way, once I have frozen or decided that what are $V_1, V_2 \ldots V_{k-1}$ and V_k. That means these are the k terms which I am considering. As per the recurrence condition by applying the linear function or the recurrence function here, that automatically freezes the next term which is V_{k+1}. And now this process keep on going. You take the next k terms apply the linear homogenous function, that will give you the unique value of V_{k+2} and so on.

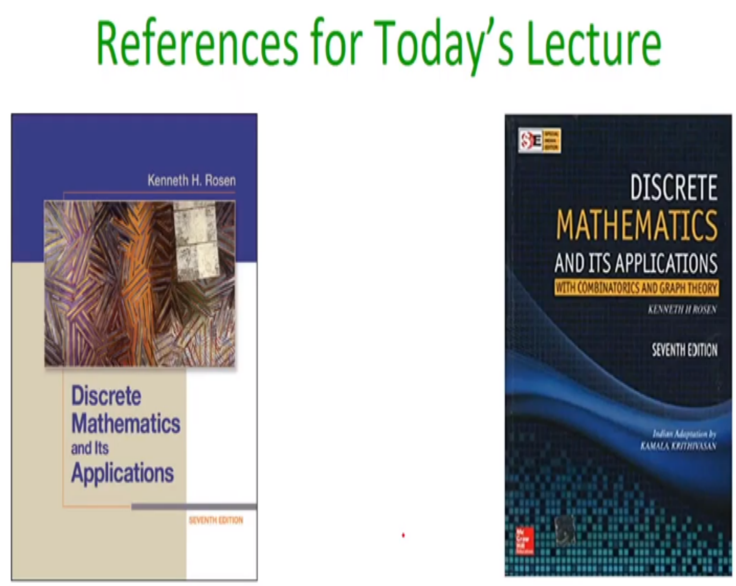
That means since my initial conditions are fixed and the number of initial conditions that are given to me are same as the degree of your equation, I will be getting a unique sequence satisfying the initial conditions as well as the recurrence condition. If you take the example where I showed you 2 sequences satisfying the same recurrence condition that was because you were not given any initial conditions.

And that is why I can fill any initial conditions or any initial terms in those sequences. So in the previous example you were just given the recurrence condition, you were not given the initial term. So that is why I was freezing the initial terms here in 2 different ways. In my first sequence my initial terms were 0 and 3.

And as soon as I froze the initial sequence or initial terms to be 0 and 3 that automatically froze my entire sequence. Whereas if I fit the first 2 initial terms with 5 and 5 and then apply the recurrence equation that will give me another sequence. So it is due to the absence of the initial conditions that I was getting 2 different solutions or sequences for the same recurrence condition.

But if you are given the initial conditions as well and the number of initial conditions if they are same as the degree then there will be a unique sequence or solution satisfy your recurrence condition.

(Refer Slide Time: 34:30)



So that brings me to the end of today’s lecture. These are the references used for today’s lecture. Just to summarize, in this lecture we introduced a new counting method called counting by recurrence equations. And we discussed a very simple method of solving linear equations (as well as non-linear equations depending upon the structure of the equation) called as the iterative method and we saw the forward substitution as well as the backward substitution method under the iterative method. We also discussed about an important category of recurrence equations namely linear homogenous recurrence equations of degree k with constant coefficients. Thank you.