NPTEL
NPTEL ONLINE CERTIFICATION COURSE

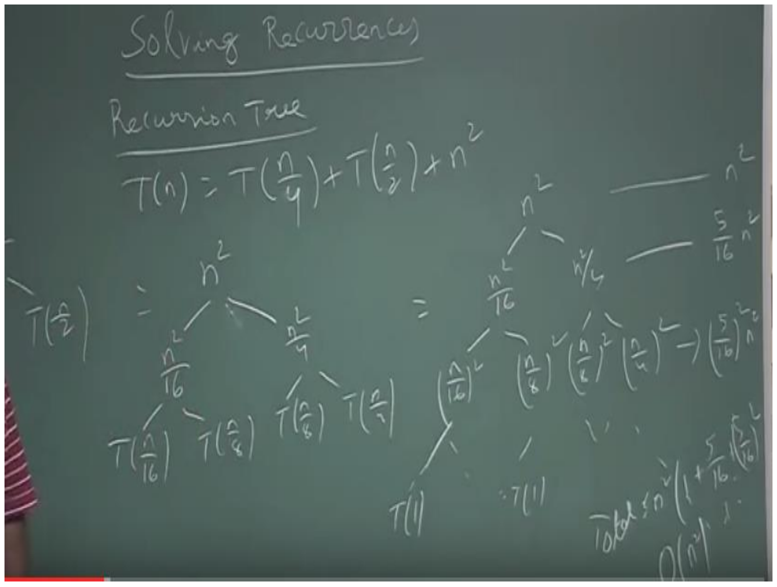
Course Name
Fundamental Algorithms:
Design and Analysis

by
Prof. Sourav Mukhopadhyay
Department of Mathematics
IIT Kharagpur

Lecture 02: Solving Recurrences

Okay, so we will see how we can solve the recurrence, the recurrence is we have seen the recurrence we are getting from our shot. So one way we can see the solution by the help of what is called, we have seen the recursion tree.

(Refer Slide Time: 00:35)


Solving Recurrences
Recursion Tree
$T(n) = T(\frac{n}{4}) + T(\frac{n}{2}) + n^2$
n^2
$\frac{5}{16}n^2$
$(\frac{5}{16})^2 n^2$
$T(1)$
Total $\leq n^2(1 + \frac{5}{16} + (\frac{5}{16})^2 ...)$
$O(n^2)$

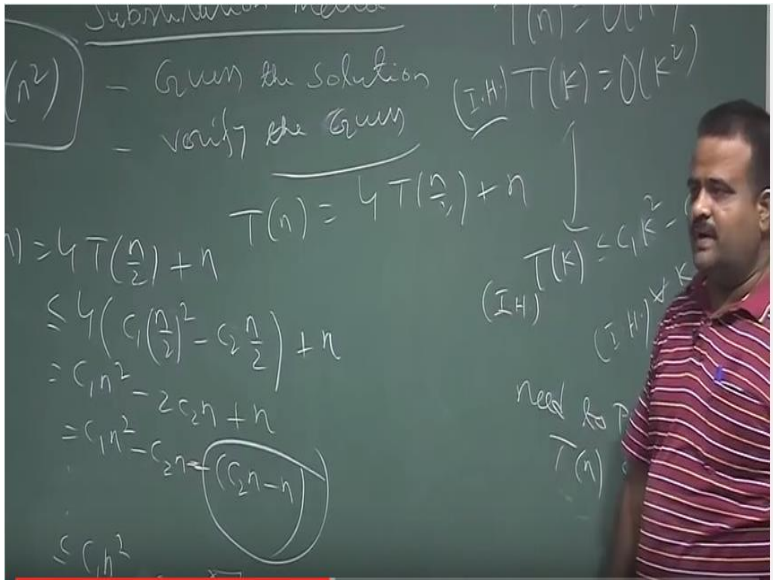
Okay, suppose we take this example of the recurrence, suppose our recurrence is T(n)=T(n/4) + T(n/2) + n^2, now suppose we want to solve this recurrence, so we want to see what is the recurrence tree corresponding to this so T(n) is basically n^2 and then we two recurrence T(n)/4 then T(n/2) and then again we assume for the subsequent stiff we have the same recurrence.

So this now n=n/4 if we put then this will be basically $n^2/16$ and then it will be again T(n/16) and T(n/8) and this will be basically n/2, n/2 we put here so $n^2/ 4$ and then n/8 t(n)/ 4, then again we, again for this, this size we will assume that we will be having the same recurrence, this recurrence we got some algorithm where we divide the problem into two sub problems, size is n/4, n/2 and this is the cost for combining, I mean the merging or combining, okay. So this will be n^2 and this is $n^2/16$, this is $n^2/4$ again we further divide this. So this will be going on so $(n/16)^2$, $(n/8)^2$ then this is $(n/8)^2$ and this is $(n/4)^2$, like this.

So we will stop at T(1) each of this branch, so this branch we stop at T(1) like this, so now the sum, how we can get the sum of this tree? All note we can do the level size sum, so this is basically $n^{2,}$ you could do the sum over here, it is $5/16n^2$ this will be $(5/16)^2n^2$ like this. So this we will continue like this. So the total is basically n^2, so $1+5/16+5/16^2$ like this up to finite term but if we put a less than it can be taken as up to infinity term.

So this is basically $O(n^2)$ so this $1+r+r^2$ so this is basically up to infinity it is 1/(1-r) so r is less than 1, r is basically 5/16. So this is $0(n^2)$. So but the question is, we are taking this is this like this so what is the, also the proof is missing that at the i th level it will be 5/16 to the power $i(n)^{2,}$ that proof is missing. so for that this method is not realistic method in the sense it is lack of some mathematical proof, we can try to prove this by induction that the i^{th} it will be like this but these can give us the rough idea of the solution so, so but there is another method which is called substitution method it is a, it is, we prove that result is true by induction okay.

(Refer Slide Time: 04:44)



- Guess the Solution
- verify the Guess
$T(n) = 4T(\frac{n}{2}) + n$
(I.H.) $T(K) = O(K^2)$
(I.H.) $T(K) \le c_1K^2 -$
$n) = 4T(\frac{n}{2}) + n$
$\le 4(c_1(\frac{n}{2})^2 - c_2\frac{n}{2}) + n$
$= c_1n^2 - 2c_2n + n$
$= c_1n^2 - c_2n - (c_2n - n)$
$\le c_1n^2$
need to P

So substitution method okay, so it has basically two steps, first one is we have given a recurrence, now we guess the solution of the recurrence we guess the solution and now we verify our guess by the method of induction, verify the, verify the guess by the method of induction and we say whether this is true or not. So let us take an example, suppose we have recurrence like this okay, and we want to guess the solution, suppose we guess the solution is, this is our guess, T(n)0(n3) so we need to verify this so we will use the some induction hypotheses, our induction hypotheses is.

Suppose this result is true for, result is true for all k up to n so this means and we need to prove that result is true for k= n, this is our induction hypotheses and we need to prove that this is true for k is equal to n, so this is need to prove and if we can prove this then we can say the result is true for all n, so from here this is we can take the some SK^3 for all k less than n and what we need to show, we need to show that $T(n) \le Cn^3$ then we are done, then we can establish the result is true for all n.

Okay so this is the recurrence and this is our induction hypothesis so we will use this, so our T(n) is basically 4 T(n)/ 2 + n. Now this is, this case $\le$ n so we can use the induction hypotheses so this is $\leq$ than 4 C k is n/2 here + n so this is basically $Cn^3/2 + n$, so this we want to be less than Cn^3 so for that we will take this to the Cn^3 – this part if this part is > 0 then we can establish this, now I need to see what is this part, this part is basically $Cn^3/2 - n$.

Okay now C is a constant which we can play with so if we choose C is > than 2 then this is n^3 so this is very much positive so this is can be established, so that means this is basically $O(n^3)$ this is done. Now we want to see whether this is $O(n^2)$ or not so that we want to see, so now we want to give the solution as this so that means we want to guess T(k) is order of k^2 for all k n and we want to show, this is our induction hypotheses assumption.

And we want to show that T(n) is $O(n^2)$ so from here what we can say this is basically $\leq Ck^2$ and we want to achieve this T(n) is $\leq Cn^{2,}$ so we will try to do it here so this our recurrence now this is < n/2 so you can very well use this induction hypotheses, now this is < 4 C $(n/2)^2$ + n, this is $Cn^2 + n$ now we want this to be – Cn^2 so if you write this Cn^2- (-1) so this cannot be a, we want this continue to be positive in order to get this.

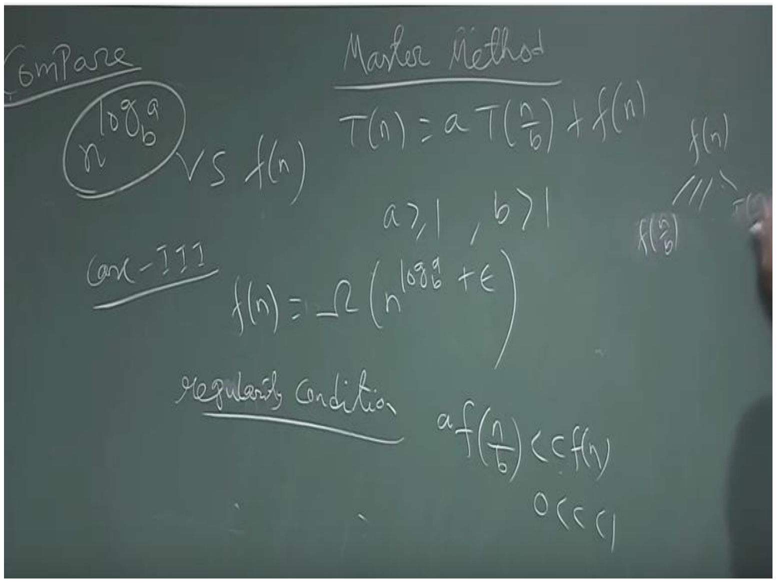
But this cannot be positive, this is n, n is a natural number so this cannot be positive so this is wrong, so in this way we cannot achieve that this $\leq Cn^{2,}$ so what is the problem, so can you conclude that this not order of $n^{2,}$ no, the thing is we need to tighten the bound so we know that T(k) is $O(k)^{2,}$ that means we need to tighten the bound, so that means T(k) can be written as $C_1n^2 - C_2n$, sorry $C_1k^2 - C_2k$ and this is true for all k < n, so this is our induction hypotheses, basically we have tightened the bound.

So this is also order k^2 okay so this is the indication hypotheses, I am what doing it to prove we need to show $T(n) \leq C_1n^2 - C_2n$ if we can show this then we are done, then it is basically $O(n)^2$ so let us try back. So now our induction hypotheses is that one so this is basically $\leq$ 4 so this is n/2 so $C_1(n/2)^2 - C_2(n/2) + n$, so this is basically $C_1n^2 - 2C_2n + n$ so this we want to be $\leq C_1n^2 - C_2$- C_2n, sorry – C_2n so this is basically $C_1n^2 - C_2n - (C_2n - n)$, so now this the received well, now these this is > than zero then we can very well say this, now we can choose C_2 is > than 1 then it will be > than 0 so this is true.

Now once we establish this we can say that now T(n) is O(n^2), okay. We go of n^2 so similar way we can try to see whether this is a [indiscernible][00:12:23] so we can easily verify that so in that case it will be greater that equal to instead of less than equal to. So this is the substitution method, we will guess the solution and that case quite from that case can come.

The recursive tree can help us to guess the solution and then once you have, we are guessing the solution we can justify our guess by using the method of induction and this is called substitution method, okay.

(Refer Slide Time: 13:10)



Next we will talk about master method to solve particular type of recurrence. Master method to solve particular type of recurrence and recurrence is of this form T(n) = a T(n/b) + f (n). Where a ≥1 and b > 1 and f(n) is positive asymptotically positive function, asymptotically positive, what do you mean by asymptotically positive? Because positive excepts may be some few points I mean f(n) > 0 for all n > k. So to some beginning points may be negative but if f(n) is always positive then it is also asymptotically positive. So, but we need for this master method f(n) is asymptotically positive.

That means it is throughout positive except up to some k and this can be treated as like this, suppose you have a problem of size a and we reduce the problem into a meaning sub problems each of size (n/b) and then we have to merge those sub problems with a cost of f(n) to get the solution of the whole problem, okay. So now this is the recurrence for the master method and the solution will come by comparing the value n to the power log a base b versus f(n).

We need to compare this values n to the power log a base b f(n), now this is the cost so we are recursively solving this, so we can say this is the cost for how much time we are spending here by recursively, by recursively solving this sub problems. And f(n) is the cost for margin those, once we have the solution then the cost for margin, so the question is who is dominating? And based on that we get the solution.

The time will be that, whether this cost how much time we are spending here that is the signal of this one and all this combine cost or merge cost, so that is basically will give a solution. So when we compare there are three cases of this master method, case- 1, case 1 is basically f(n), f(n) is polynomically f(n) is slower, f(n) is lesser, f(n) is f(n) grows polynomically slower than n to the power log a base b.

So that means this is the bigger term, polynomically slower means f(n) is basically O(n $^{\log}$) a base b minus ε, where ε > 0, that means this means what, this means some constant into n to the power log a base b by n to the power ε, so this means this is bigger even it is if we divide some polynomial factor with this then also this is bigger, then also f(n) is slower. So that means this is the dominating term.

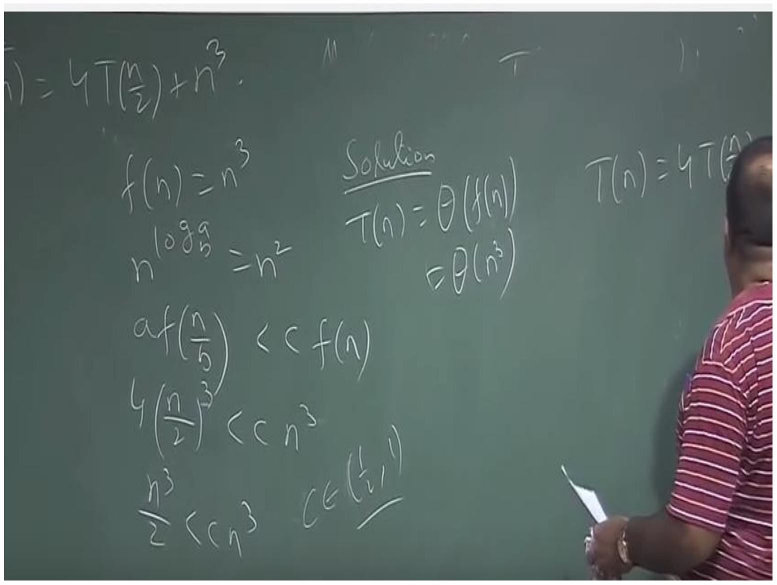
This cost is dominating then the solution will be what? Then the solution will be this one, then the solution is, this is in case 1 T(n) is n to the power log a base b because this is the cost which is dominating, because f(n) is lesser, even polynomically lesser, this is the case 1, Now case II, when their growth is same, when they are similar, so case II is basically f(n), this $n^{\log}$ a base b grow at similar rate. So this we can just write as both sided bounded by this term $n^{\log}$ a base b with a log factor, $(\log n_2)^k$ so there are differing so if this case k= ≥0. If k is 0, if k is 0 then this term is not there, then they are, they are both exactly similar. But if case having some I will say k is 1, then they are similar but with the log factor, log is very less, because if n is 2^{10} then log n is basically 10.

So their difference is a log factor multiplication, so in that case, so they are saying, so the solution will be, so $n^{\log}$ a base b into this due to this log factor, log n^{k+1}, this is the solution for case II. Now case III, case III is basically the f(n) is the domain, f(n) is bigger. Case III is basically, so f(n) is polynomically bigger than $n^{\log}$ a+b, so that means even if you multiply by polynomial factor n^{ε} then also f(n) is bigger.

And here we have another condition what is called regularity condition, this is basically telling us we have to check this af(n/b) must be $\leq$ cf(n) here c is lies between 0 and 1. So if this is the recursive, if you draw the recursive tree this is, this is basically a times T(n/b), now if you again further do this, so this is f(n/b), f(n/b) like this, so at, so this is the f(n) at the top level, so which is dominating? Now in the next level the sum is af(n/b), now this is the next level. Now we want f(n) is really bigger, so that means if you take a fraction of f(n), if you multiply by c, fraction f(n) then also next level term is lesser than this, this is the extra condition for the third case.

In that case what is the solution, solution is T(n)=f(n), because this is the dominating term. Now we take some example to see the solution for this master method.

(Refer Slide Time: 22:19)



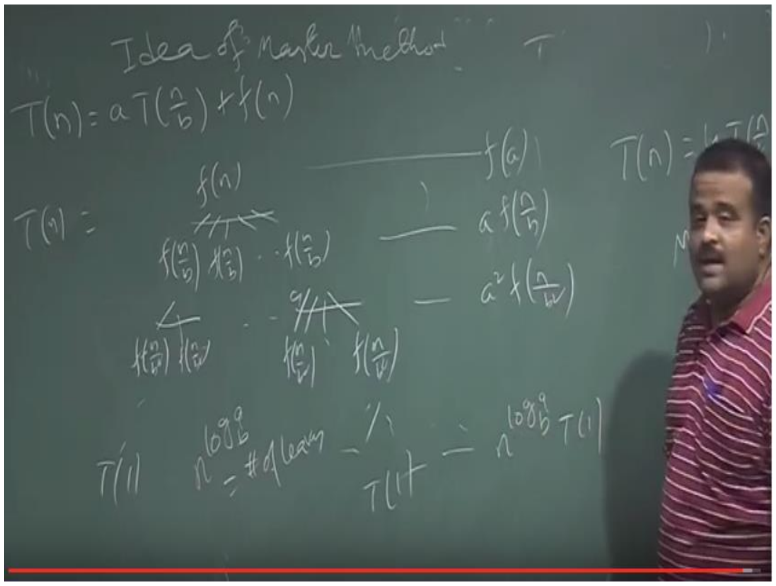
Okay, suppose our recurrence is T(n)=4T(n/2)+n, okay, so what is our a, our a is 4, b is 2. What is $n^{\log a}{}_{b}$ it is basically n^2 and what is f(n), f(n) is basically n, so we are in case II, I am sorry case 1. Because n^2 is dominating, here in case so what is the solution, so the solution is O into this is the dominating term this. Okay, so we are discussing some example here for master method.

Now second example is, suppose we take $4T(n/2)+n^2$ so a is 4, b is 2 $n^{\log a}{}_{b}$ n^2 now this is case II, because these and these are similar f(n) is n^2 with K =0, there is no log factor over here. So what is the solution, solution is basically $T(n)=nl^{og\ a}{}_{b}$ $(\log^{nk+1})$. Now K is 0 over here, k + 1 so this will be n^2 k is 0 $\text{login}^{2,}$ okay. So now the next example if we choose this same one but instead of this is n^3 so now f(n) is n^3, now $n^{\log a+b}$ is n^2 so this is polynomically faster than this, now we have to check, so this could be a third case but we have to check the regulatory condition so for that we have to see af(n/ b) whether this is less then = cf(n) or not, so a is basically what, a is basically 4 af (n/b: $n/^2$ so this is basically $n/2^3$ is less than cn^3 so this is basically $n^3/2$ less than cn^3.

Now if I choose c lies between a ½ to i then we are done so regulatory condition is also satisfied so the solution is T (n) = Ɵ fn which is basically $n^{3,}$ okay. So now if we take another example, suppose T(n) is 4T n/2 + n^2 /log $n^{2,}$ now you can see this is neither in any of this case this is another one between case two and case three so this cannot be solved by master method, so it is not that all this recurrence will have solution by the master method or there is a, here is the example where we cannot solve by the master method.

So there are some generalize of the master method like acromagaly, those are out of this, in the scope of this course but this is no master method cannot be apply, master method fails here.

(Refer Slide Time: 26:50)



Okay now we will give the rough idea of the proof of the master method, okay, so this is we will try to see by the help of recurrence tree, so this is the recurrence T (n/b) +f(n) so if you draw the recursive tree this will be like T(n) is f(n) so af(n/b), af(n/b), af(n/b) like this, so we continue like this.

So this is af(n/b^2) a times af(n/b^2) a times, af(n/b^2) like this and we will stop when we reach to a 1. Every branch we will stop when we reach to a 1, so now if we take the level wise some this is a, af(n/b) this is s^2 f(n/b^2) and this is basically the number of leaps is basically how much n$^{\log}$ a base b, this is the number of leaps so n$^{\log}$ a base b is the number of leaps, it is coming from the height of the tree so this basically $n^{\log}$ a+b a for 1, so this is the cost at the bottom, top level, this is the cost at the bottom level, now we want to see.

Whichever is the dominating, in the case one this is the dominating cost so this is asymptotical, this is first step, so this is the cost and in the case two all the level they are similar, so the height, so the that is why if it is exactly similar then these into height of the tree, height of the tree is the log n factor so that is why it is these into log n and in the case three this is dominating so the, and also you have a regulatory condition over here and this is much more better.

Much more faster than this so that is why the cost is dominating by f(n) so this is f(n) okay so this is the rough idea of the proof but in the [indiscernible][00:29:23] we have a details proof idea, so thank you.