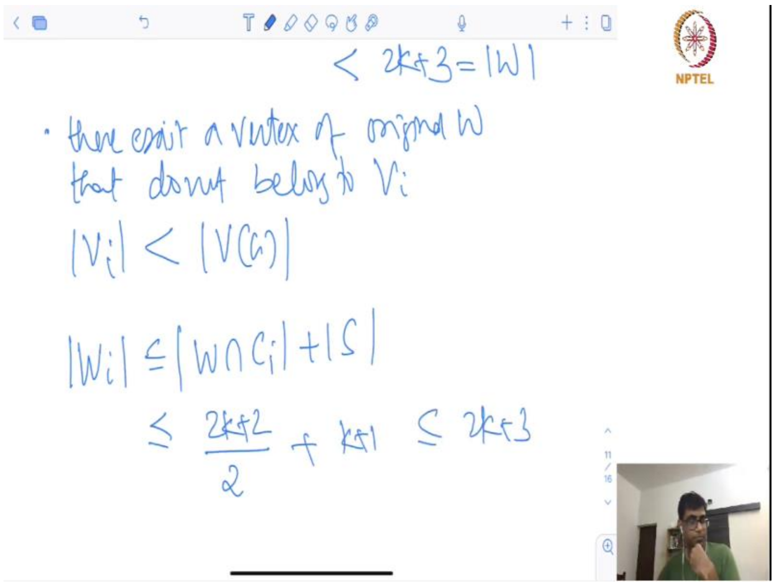
Parameterized Algorithms
Prof. Neeldhara Misra
Prof. Saket Saurabh
The Institute of Mathematical Science
Indian Institute of Technology, Gandhinagar

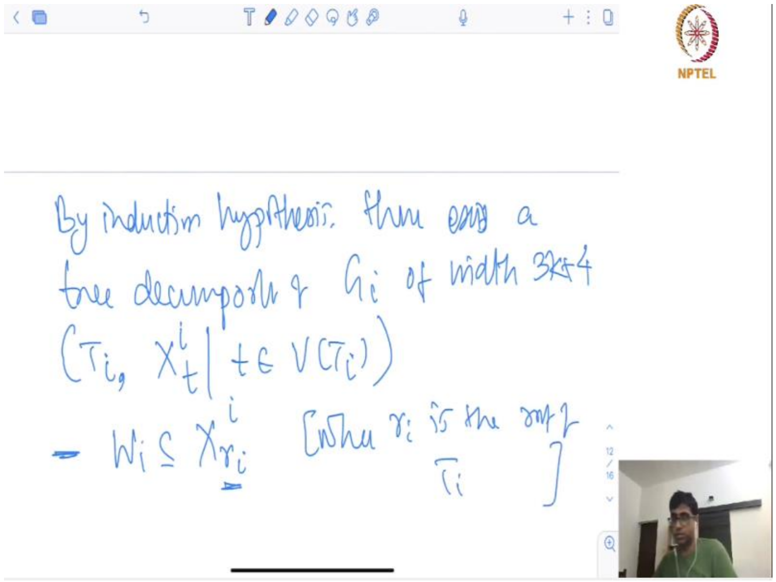
Lecture - 28
FPT Approximation Algorithm for Computing Tree Decomposition-Part 02

(Refer Slide Time: 00:16)



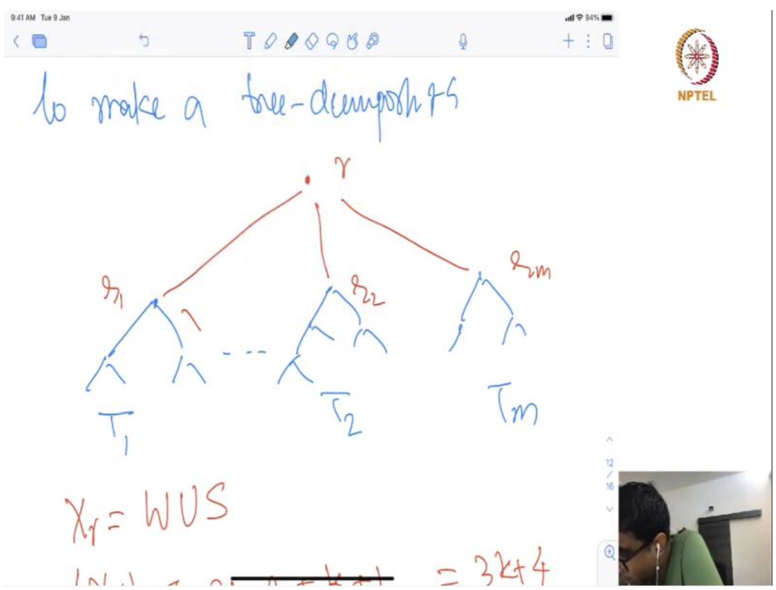
Now what about W i, let us look at W i, which we have assigned, its W intersection C i + S which is we just did the maths no so it is like 2k + 2 divided by 2 + k + 1 and we have shown that this is less than equal to k + 3. So, it is indeed true that when we recursively call right like by induction hypothesis what we know that if I will give up so what does this imply by induction hypothesis.

(Refer Slide Time: 00:56)



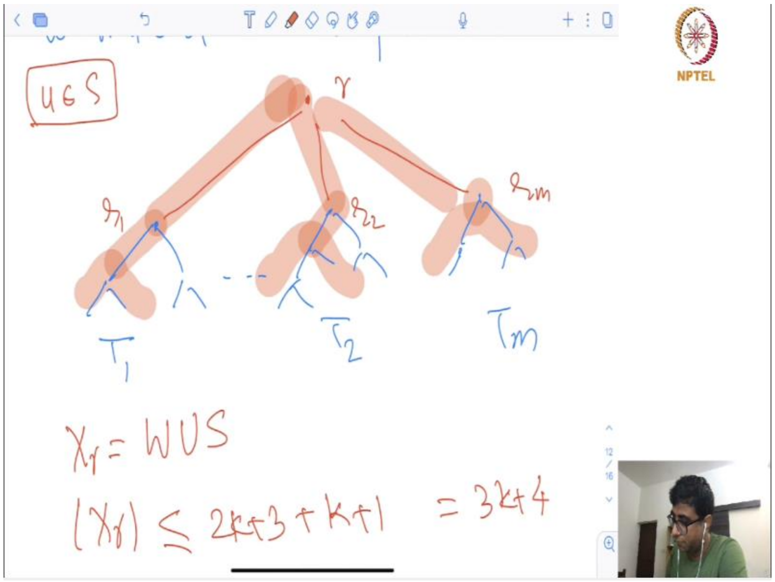
Now we can apply induction high induction by induction hypothesis. There exists right there exists a tree decomposition of G i graph G i that we created of G i of width 3k + 4 of width 3k + 4 and what is this say T i and let us say X i and t invert X set of t and so of with this. And what is the property of this, is that W i is contained inside let us say is contained inside say X r i this is a root X r i i where r i is the root of T i so this is given to us.

(Refer Slide Time: 02:14)



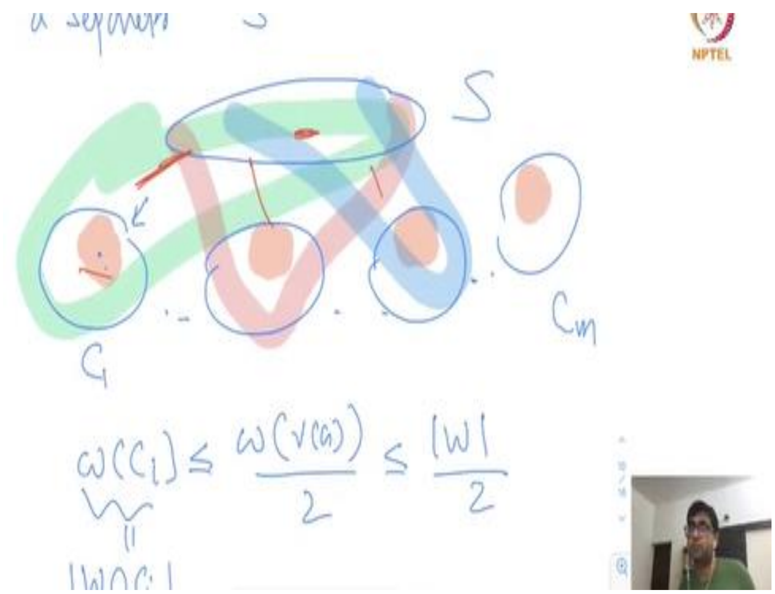
So, now how can we so now what will I do? To make a tree decomposition of G, what will I do? So, here is my; these are my T 1 T 2 T m these are the trees and their assignments. We now what I am going to just do is going to add and now these all are rooted at r 1 r 2 r m so these are like rooted at their root. I am going to make so for the tree structure for G I am just going to make one root call it r and make him adjacent to so this is my tree structure. And what I am going to assign X r to be I am just going to assign X r to be equal to W union X that is it.

(Refer Slide Time: 03:18)



Now X r = W union X which implies that the size of the bag X r is, what is W? W was like 2k + 3, what was X? This not X W union S k + 1 which is 3k + 4. Now why is this a valid tree decomposition? Let us ask ourselves why is this a valid tree decomposition look at any edge of look at any vertex definitely either like definitely every vertex appears in some of the bags, because and what about any edge. So, where is my edge?

(Refer Slide Time: 03:57)



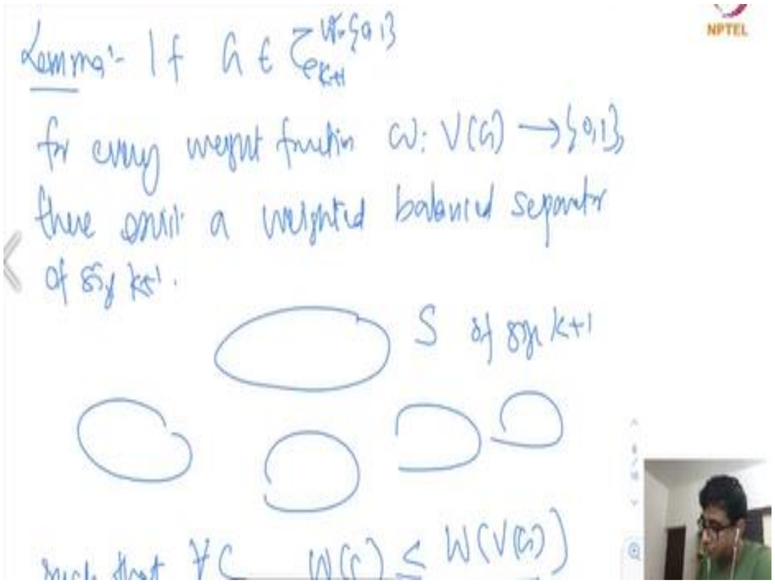
My edge is either completely contained inside here or it is completely contained inside here, here or here. Now all these edges are being taken care when we talk about C i union S. So, every edge is perfectly fine but what about what about connectivity. So, now if a vertex only appears in C 1 its connectivity is given by induction hypothesis because of the T i's. Now vertex of W could appear at several places W because it could belong to S.

So, basically what happens is that for those vertices in S. Now if those like so you could have a subtree fix a vertex here. So, the only vertex who could appear in several pieces is a vertex of S. So, now look at a vertex of S. Now so look at a fix a vertex of S let us say we fix a vertex U in f now you could appear several, several places. So, now look at its connected pieces like maybe let us use slightly different colour.

So, maybe it appears here in some connected pieces appears here maybe appears here. Now notice this is a vertex of S it means it is assigned look every vertex of S is part of W i. And I know that W i appears in the root which implies that actually this tree must contain this tree must look like this must look like this. And now what is the property the vertex of U is also part of this. So, I know that there is also a node like this.

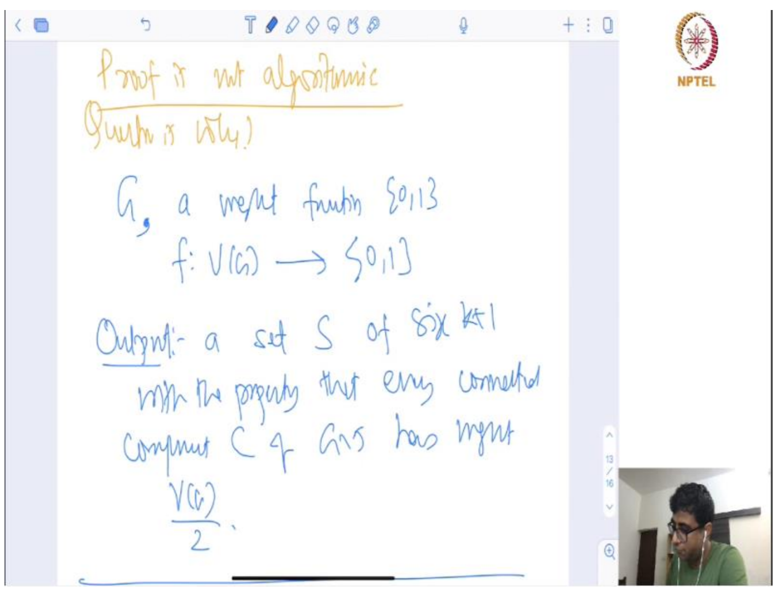
It means this gets connected. So, that shows the connectivity part of this which means that indeed what we have obtained is a valid tree decomposition of width 3k + 4. So, we have shown to you that so what we have shown to what we have shown?

(Refer Slide Time: 06:28)



Now that indeed if a graph has this weighted balance separator with respect to 0, 1 weight function, for every weight function then its tree width is indeed upper bounded by 3k + 4.

(Refer Slide Time: 06:46)

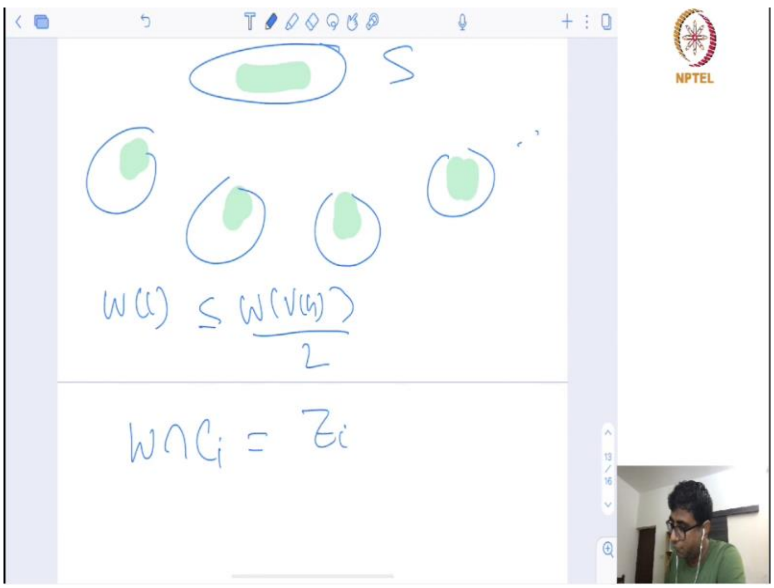


But in this proof is not algorithmic and the question is why? So, the only reason this proof is not algorithmic is that when we apply induction hypothesis this, we can we can make a designer we can replace induction with recursion. But what is not clear to us that what we used is right. I mean so, the computation of S we said well when you were trying to do you are given W, we assigned a function which assigned every vertex of W 1 and 0 and say because this graph belongs to this.

It has such a balance separator by definition and you apply that algorithm. So, basically it is a part what is missing is that so the part which is missing is the following. So, you are given a graph G and a weight function 0, 1 weight function from say f from vertex set of C to 0, 1 and what is what are what we want output a set S of size k + 1 with the property that with a property that every connected component C of G - S has weight vertex set of C divided by 2.

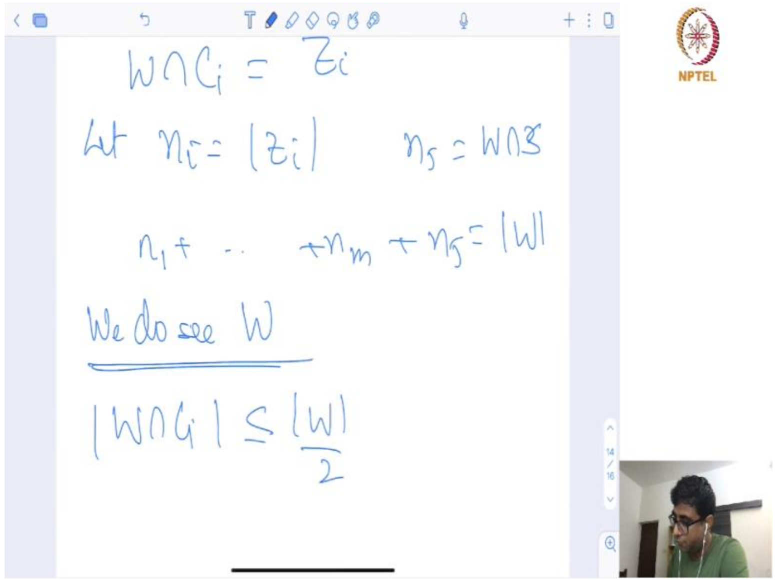
So, can we do this? If we can do this then we can make our algorithm constructive. So, next our idea is to make this separation algorithm and then induct that separation algorithm into the piece. So, there are, so now so a set S of size k + 1 with the property that every component of C of G – f heavy. Now first of all, we know that.

(Refer Slide Time: 09:50)



Suppose you have a set S such that weight of every component is weight of at most this. Then can I compute the set S effectively. So, suppose this is my piece this is my piece, this is my piece, this is my piece, and maybe let us write down the W. Now let us look at this W. So, let us call this W intersection C i and let us call them let us say some S or Z i.

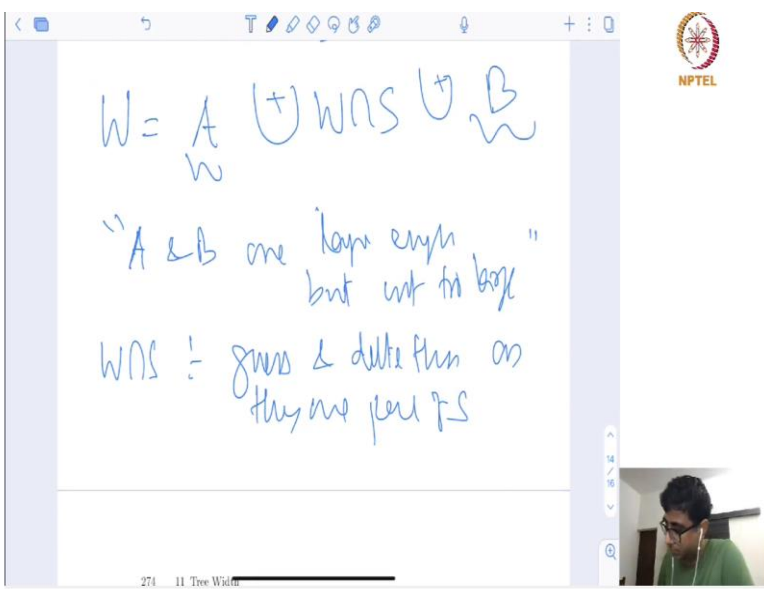
(Refer Slide Time: 10:48)



And let n i equal to cardinality of z, and let us call n s = W intersection S. Now what is the property of this? n 1 dot, dot, dot n m + n s equal to cardinality of W which is like whatever. So, we will be given some 2k + whatever the size may be for now. So, but notice we do see W. Now I want to ensure that w gets the, in the proof what we use either the only time we use that will look at the w intersection C i cardinality of this is upper bounded by carnality of W by 2.

How can I ensure that this happens? So, to ensure that what we will do we will say look let us guess how W is distributed. So, let us guess how W is distributed here and.

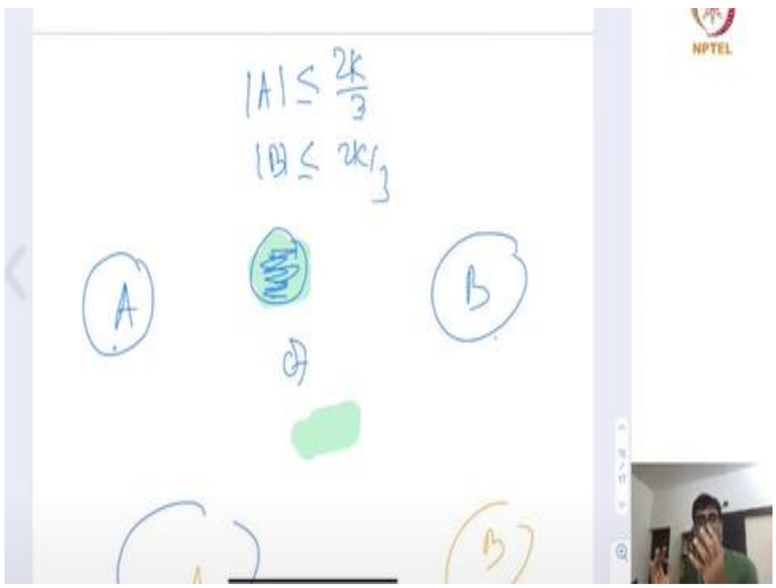
(Refer Slide Time: 12:14)



So, what I will say that look at W we can partition W into a, some piece disjoint union W intersection S and B. Can I group W into these pieces? Such that A and B are large enough but not too large we will quantify what this means. Understanding W intersection is easy. Guess and delete them as they are part of S. Guess and delete them as they are part of S that is great but what about how do I guess W 1 and W 2.

Now; that is also easy actually in some sense because still it is a function of k. So, I can guess some A and B which is basically will be collection of some green spots and B being some collections of green spot such that I mean you need to delete some vertices of S this property happens. But notice that I want that; W are separated nicely. It is not that one piece contains large fraction of W in the sense that it contains all but say 2k - 10 and 10 vertices here.

(Refer Slide Time: 14:20)

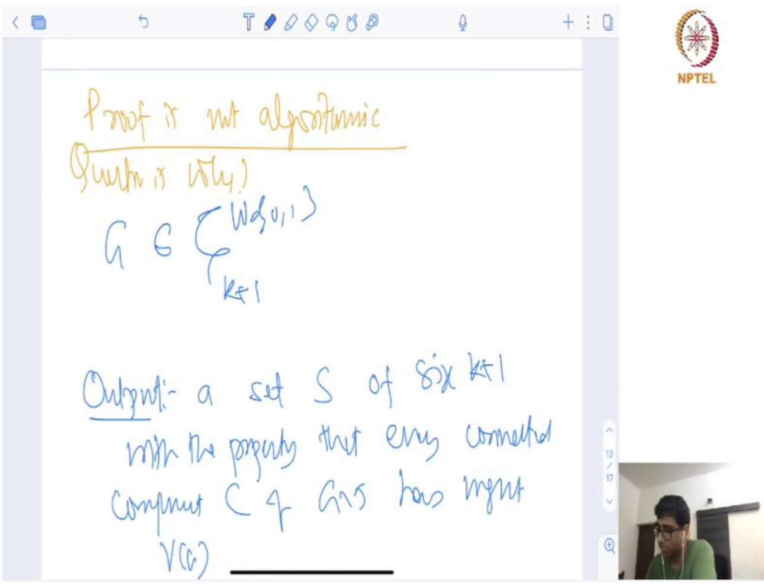


We do not want that to happen so to that to do that. I need to show to you imagine yourself that the property of A is that and B is that they are at most 2k by 3. Suppose I am able to prove that cardinality of n conductive B is at most 2k over 3. Then what happens that I guess this partition A and B of W, I guess this partition of A and B of W and then I will say find me the minimum balance separator which separates A and B.

Let us delete these vertices. Now what is the property? That when I delete this there is no component that contain both vertices in A and vertical B then what happens if you look at any connector component after you have deleted this. It can only contain either a vertex of a either A vertex of B but it cannot contain both vertex A and B because we have separated A from B of the minimum size.

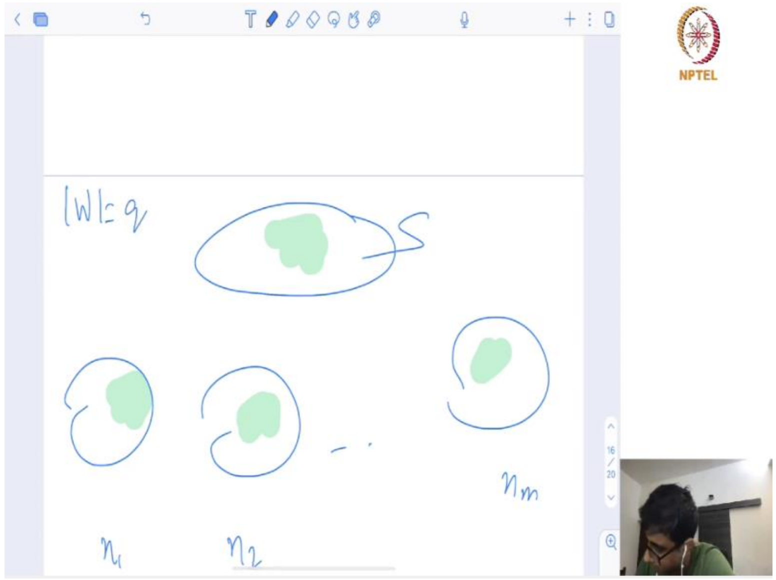
Then what happens then any connected component can have vertices at most either A or at most B which is at most 2k over 3 and then we I will show to you how we can make use of this also to make progress. So, that is the whole idea of our algorithm is that I want to say to you that look that exists a partition of my W into A and B such that no A and B are large and then I want to find a balance separator of that piece. And then I want to find the balance separator of this A and B such that these properties hold.

(Refer Slide Time: 16:35)



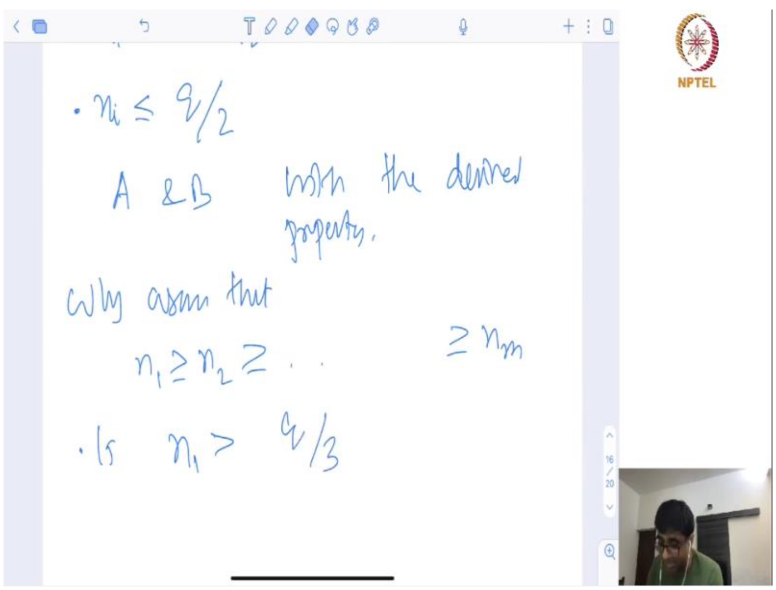
So, we know that the property that with respect to this weight function f the so basically, I should not write rather I should write this is not right way to say. Suppose G belongs to z weighted 0 ones like it has a property with this that of k + 1. Then what happens? So, you know such an object exists because G belongs to this look, we do not know whether G belongs to this order but we will be able to check. Suppose if it does not belong to this, we will be able to show that something bad happens.

(Refer Slide Time: 17:28)



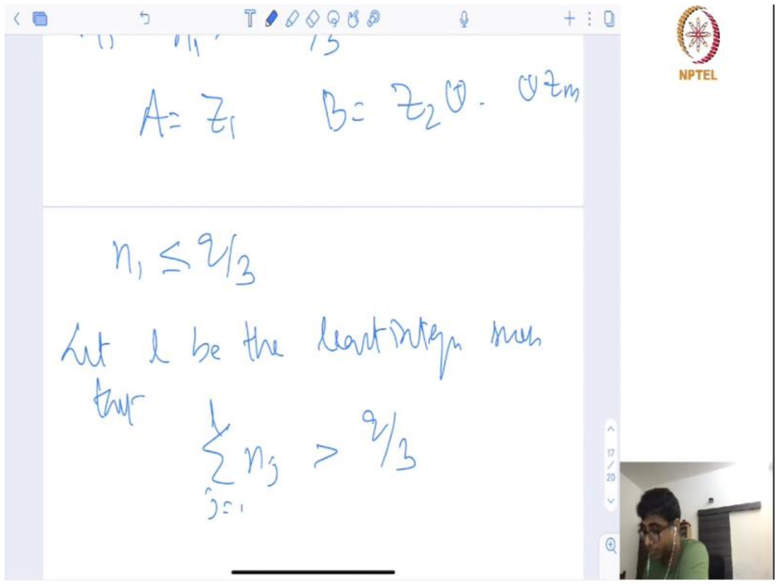
So, now what will they do as an algorithm. So, first let us try to show something interesting here. So, here is my set S, and here are my components. And what is the property of these? Components like and this is my W gets distributed here W gets distributed. So, suppose this is like, what is the property of each of these n is? The property is that each n i is at most let us say cardinality of W is same q.

(Refer Slide Time: 18:20)



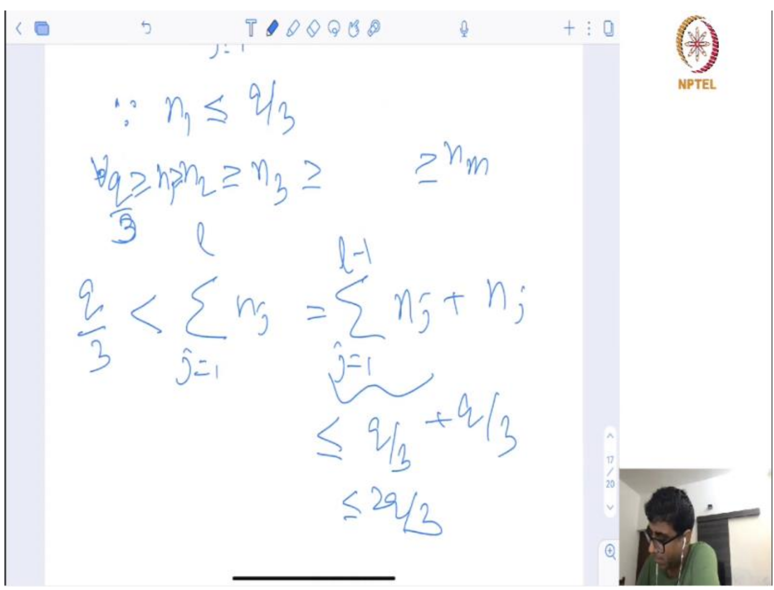
Say its q but then I am telling you that there exists a partition. Then I am going to show to you there exists partitions A and B with the desired property. We have seen this before but let us repeat it once again. Without loss of generality assume that n 1 is larger than n 2 is larger than dot, dot in m. I first checked myself is n 1 greater than equal to q over 3. If it is then we are very happy, because then I am going to take a as then what I am going to take.

(Refer Slide Time: 19:12)



Look and I am going to is I am going to take a as W 1 so suppose z 1 and remember that this is like the set was called z 1 z 2 and z m and this set, we called it z s. So, what will I take a as z 1 and my B as the 2 disjoint union z m. So, this is how we will partition?

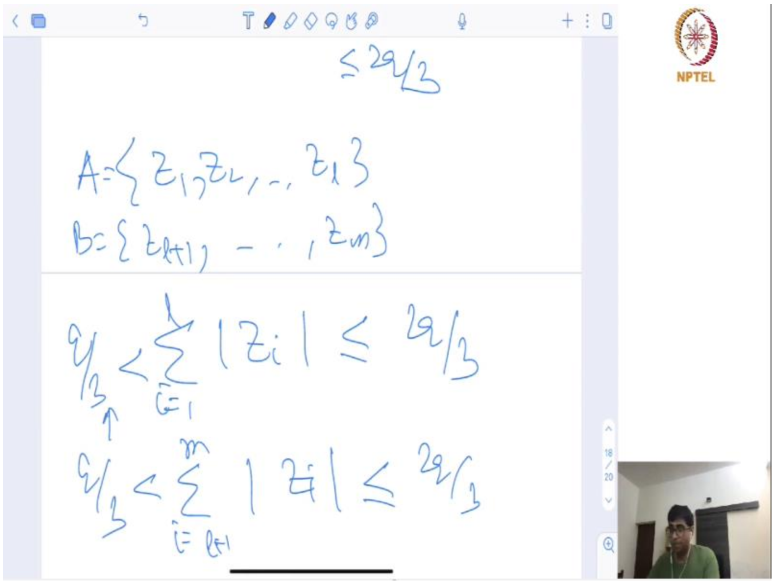
(Refer Slide Time: 19:48))



Now let us assume that n1 is less than equal to q by 3 then let l be the least integer. Such that l 1 with l be the least integer such that if I do summation n j equal to 1 to l n j it is becomes greater than q over 3. It this is the first time it becomes greater than cube or 3. So, then let us ask ourselves first of all since n 1 is less than q over 3. What do we know? All n 2 which is less than n 3 dot, dot n m every 1 is.

So, every number is at most q over 3. So, what is summation j = 1 to l n j is less than equal to summation j = 1 to l - 1 n j + n j which is less than equal to because this is at most q by 3 and what is n j is another at most q by 3 because which implies that this is at most 2q by 3.

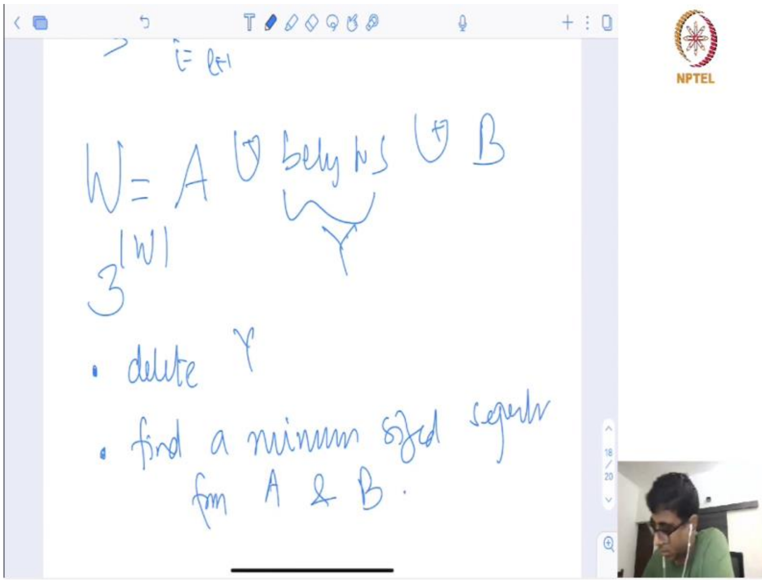
(Refer Slide Time: 21:34)



Great it is at most 2q by 3. But we also know that this quantity is strictly greater than q over 3 by our choice of l which implies that, what is my A then? My A is going to be z 1 z 2 dot dot dot z l and my B is going to be z l + 1 dot, dot z m which implies that if you sum this number what is the so I know that if I cardinality of z i, I going from 1 to l is at m is at least q over 3 at most 2 q 3. Now summation I going from l + 1 to m z i.

What is this? This quantity is at least q over 3 so this is at most 2k over 3 and since it is at most 2q over 3 this is at least q 3. So, in fact there does exist if we have a separator S with the property that it can like such that if I delete it every piece contains at most half the vertices of W then. Then actually there exists a partition of vertices that is not contained inside S into two parts each part has size at most q over at most 2q over 3.

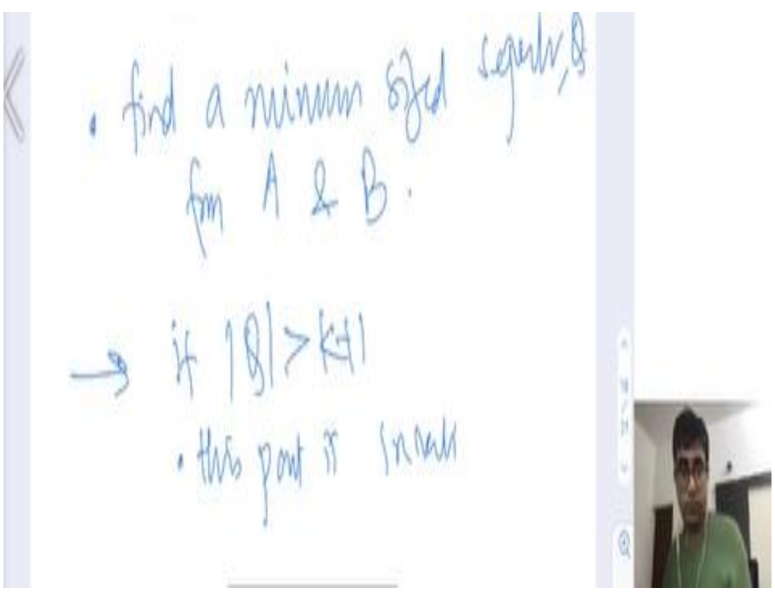
(Refer Slide Time: 23:16)



So, now what will I do to find? So, now you gave me w i will partition this into A disjoint union S like A belongs to S and belongs to B, how many such partitions are there? So, this is basically partition of w into three pieces so this is upper bounded by three times cardinality of W three times cardinality of W. So, it like right so once we have done this so what we what will this tell us? That it tells us that look you.

You wanted me my w if what will this tells us that look, I do this. Suppose there exists A set S with the property that this holds then I will do what head my algorithm delete G - f delete let us call this let us call this capital Y delete Y find a minimum sized separator from A and B.

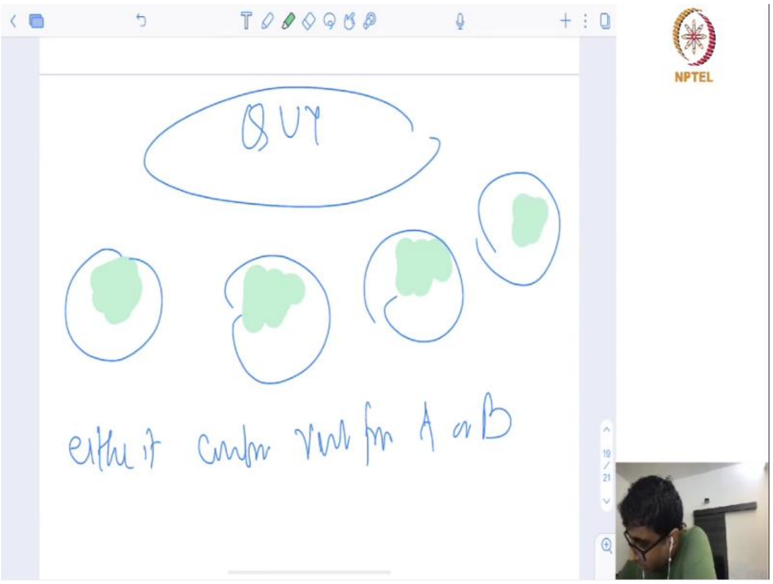
(Refer Slide Time: 24:59)



So, now what we know if minimum psi separator let us call it q if cardinality of q is more than k + 1 this part is invalid. Now so because you know that there why because you are guaranteed that if you delete for at least some if for some choice of S - Y like if you delete S - Y you know that A and B get separated you know they are guaranteed. So, and its size is upper bounded by k + 1. So, if the current of q is more than in fact you should not say carnality of k +1 if carnality of is k+1-y.

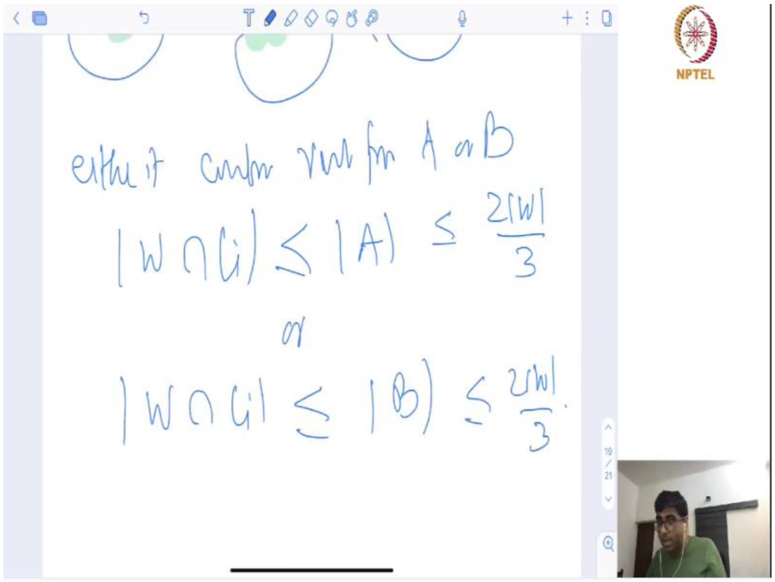
Then you should say no or then maybe you found this but now if you found this. Then what is the property? If you found this then you are done.

(Refer Slide Time: 26:24)



Because, because then what is the property of Q union Y the property of Q union Y is that if you delete Q union Y look at any connected component any connected component has a property that either it contains vertices from A or B. So, if I look at the intersection of w now to this component.

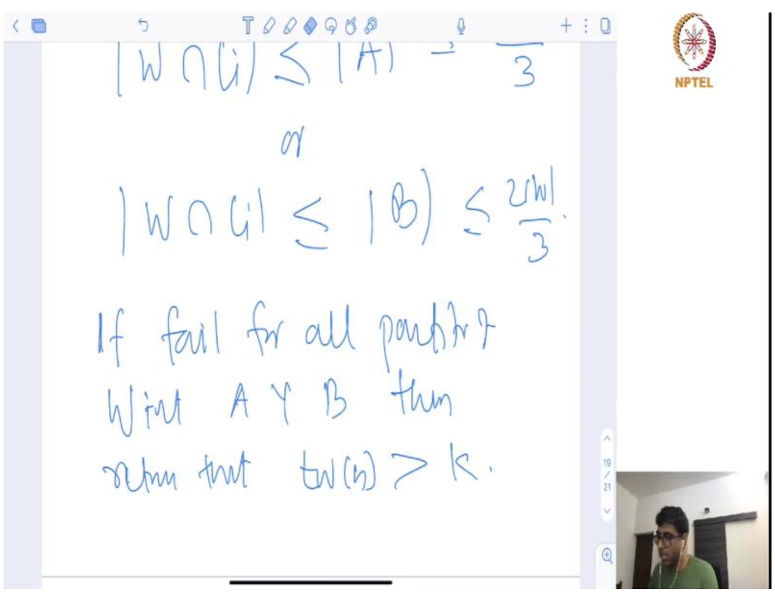
(Refer Slide Time: 26:51)



This so w intersection c i is upper bounded by either A or w intersection c i is upper bounded by B, which is upper bounded by whatever 2 times w divided by 3 which is at most 2 times w divided by 3. So, there is a balance separator but what happens but so this happens. Now what happens? That for every A and B if every part you fail this it just implies that this part is like for every for every partition of A like A y B.

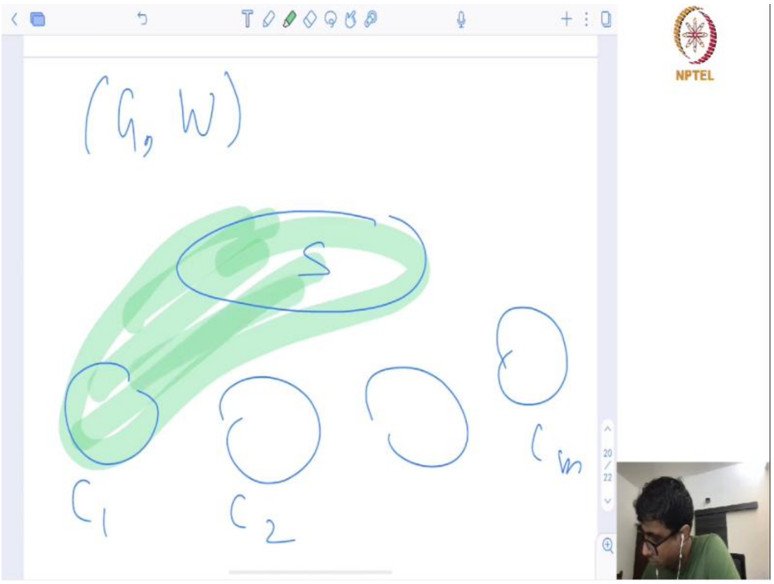
If you fail it just means that the graph G does not have a desired S and then you said say tree width of G is definitely larger than this. So, this is a time when you will return no.

(Refer Slide Time: 27:59)



If fail for all partition of W into A Y B then return that tree width of G is strictly greater than because if the tree width of G was at most k, then we will definitely find such the x partition for which we will find the desired S with the properties. Now notice that in our existential algorithm we will be we needed half to half to do all this but since we do not have half any more, we will do something more.

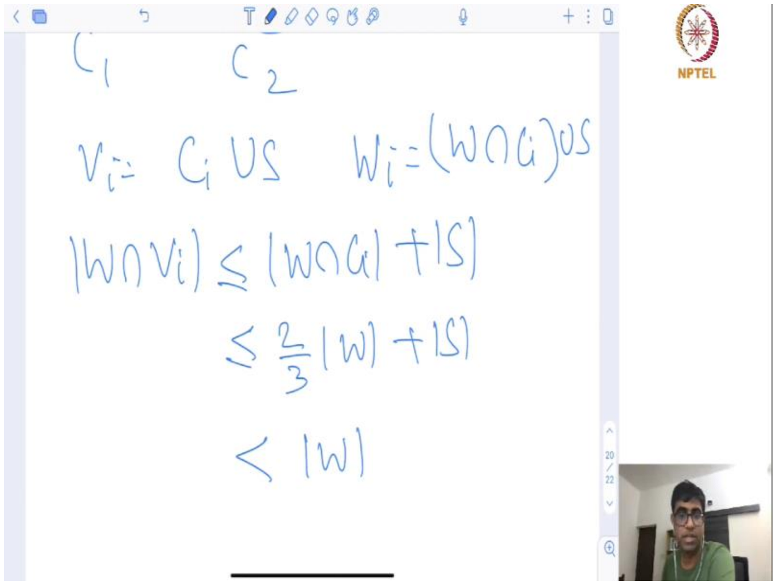
(Refer Slide Time: 29:10)



So, now so for the algorithm what we are going to do inductively we are going to have W you will see W so remember so at any point of time in our algorithm we were constructing we had a graph G and a W and of course. So, now the cardinality of W that we are going to fix if let us see how much we are able to fix this time. So, where did we use the fact so if you recall correctly so the what we did? So, we given this W we found a separator S.

And we had this connected components C 1 C 2 C m and we actually say that this is the part where you recursively go compute.

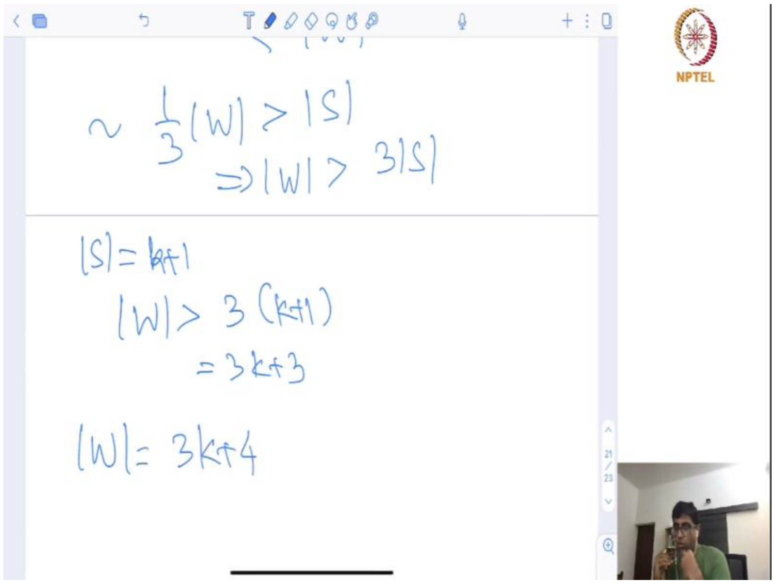
(Refer Slide Time: 30:32)



Now we the we wanted to show that look at this say look at this V i which was nothing but C i union S. So, we needed to and what did we set our W i the set W i was basically W intersection C i union S and now both needed to have the sum property. Now what was V i so to upper bound V i what we said, so we said let us I wanted to upper bound V i. So, let us say let us try to upper bound W intersection V i and we said that look at this W intersection V i.

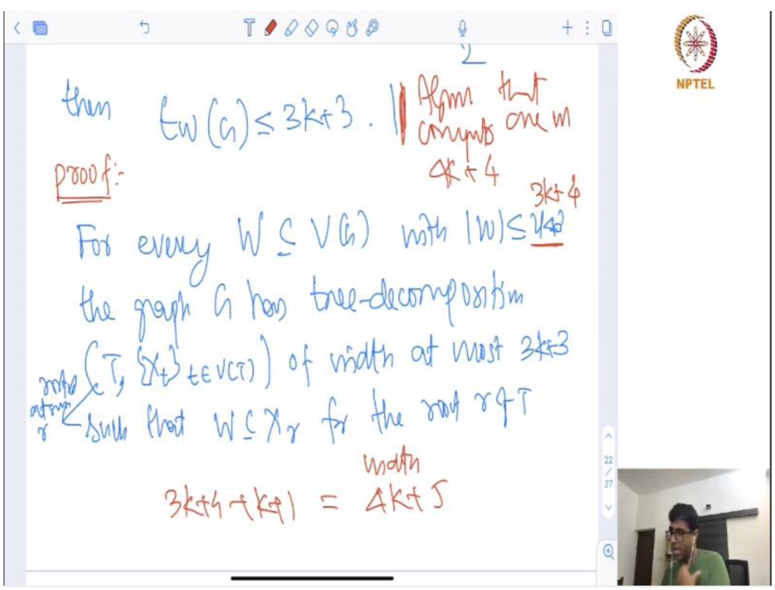
It is cardinality of it is what it is W intersection C i + mod S. Now what can I say now about W intersection C i only think we can say about W intersection C i is this is at most 2, 3 times W. We cannot say half anymore plus mod S, and I want this to be strictly less than cardinality of W because then I will say that there is a vertex of W which is not in this part and hence the size of this graph is upper bounded by something. So, now if I do this?

(Refer Slide Time: 32:00)



So, what do I get what I get from here is that one third of W should be greater than cardinality of S, one third of W should be because I just taken should be greater than equal to S which implies that W should be greater than equal to 3 times mod S. Now what is the meaning of this? I know that the mod S has size k + 1. So, W better be 3 times k + 1 which is 3k + 3. So, now I am going to set w = 3k + 4, that is it. So, I am going to run my algorithm. So, what is my algorithm? My algorithm is as before let us go back to our algorithm.

(Refer Slide Time: 33:09)



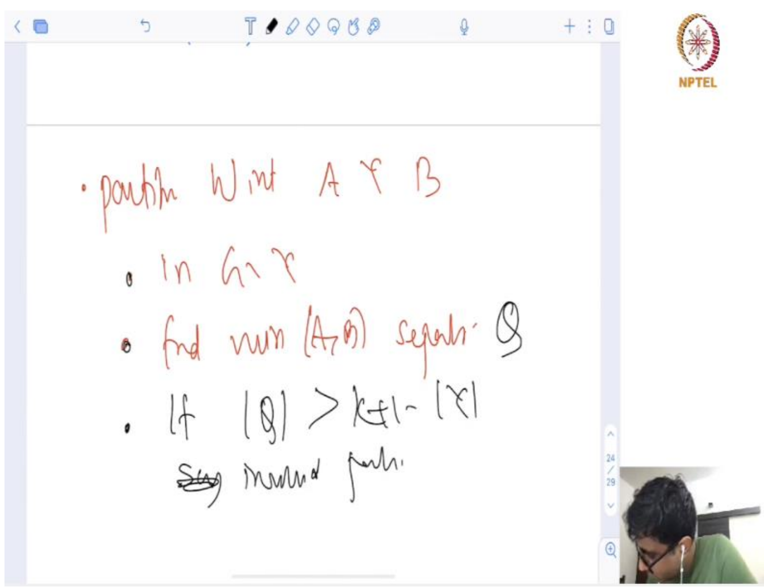
So, this is going to be inductive algorithm, but now let us copy paste this we did already copy, copy, copy, copy, copy. So, now I am going to this is the same thing wait here this then this is true but now I am going to give an algorithm that computes 1 with 4k + 4. So, now rather than having this 2k + 3 I am going to replace this with what did we get just now 3k + 4. I am going to replace this with 3k + 4 and we are going to so 3k + 4 + k + 1 and so that will may imply 4k + 5 which is nothing but 4 times.

So, we will give a tree decomposition with this 4k + 5 width and again the property of width but now the width has increased to 4k + 5. So, the w is x r for everything. So, now how will do? I think there is something missing here I think there is some piece missing here. So, again my step will be that, if the number of vertices is less than say 4k plus if the number of word this is like this is going to be 4k + 4.

So, if the number of vertices is going to be at so now, we are going to replace this item with if n is less than equal to 4k + 5. Then one node tree decomposition otherwise you assume that the number of vertices in the graph is more than say 4k + 5 and be such that again you will do the same thing. But now then you append it so that it becomes equal to 3k + 4 just for simplicity you again do this 3k + 4 you assign this weight function a separator S.

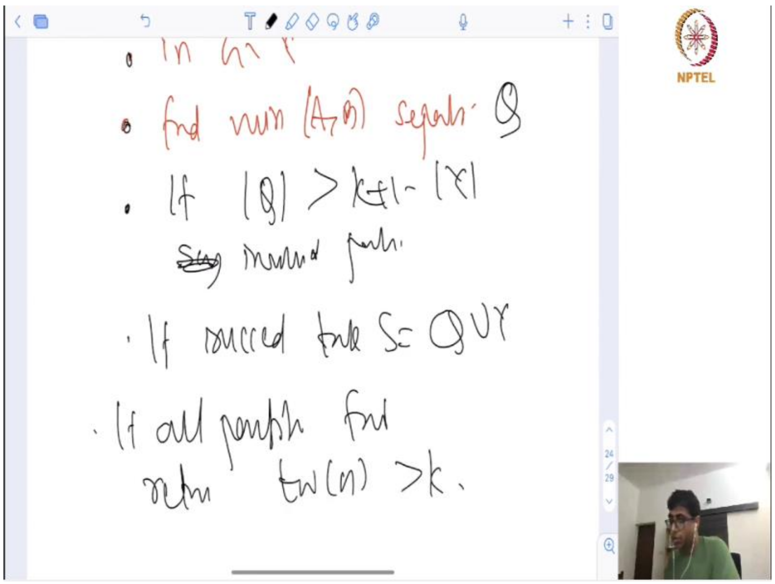
And this is where we will make all the changes; we needed to find this separator. So, at this place what will we do?

(Refer Slide Time: 38:28)



This is the place we will say partition W into A Y B and in G - Y find min balance min separator minimum separator min A, B separator and by the way you can find a case size separator in times k times the whole graph. So, this is polytechnic this is basic little separator so this can be actually done. You find min A, B separator but let us I will not worry about it suppose this you can do in poly time. Now if and that was we called Q if the Q has size more than k +1 + mod Y say invalid partition.

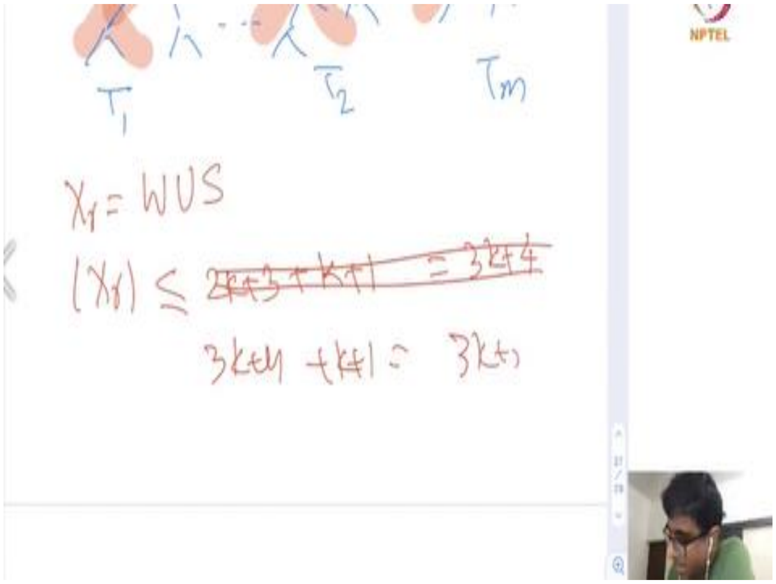
(Refer Slide Time: 39:30)



If succeed take f = Q union Y, if all partition fail return tree width of G is more than k. So, this is what we did so now we needed this balance separator. So, we will apply this algorithm and get this S, and what is the property of this? That I cannot say that weight of C is this but now I can only say that this is 2 times 3. We cannot say because of that we are only guaranteed that this is what it is and again we need to have invariant all this and that will happen.

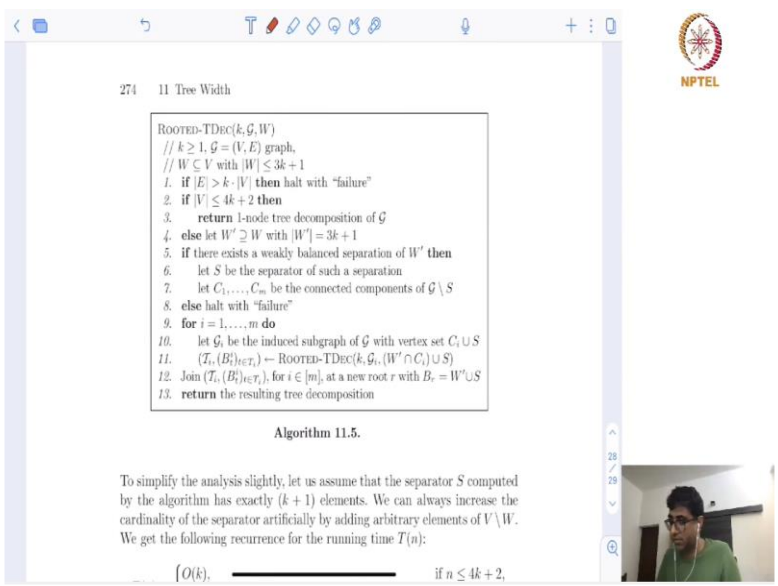
So, if you just do the maths again with not half but two third you will succeed and prove everything right so then by induction hypothesis you will get a tree decomposition like by on the smaller graph this is a recursive algorithm.

(Refer Slide Time: 40:50)



So, by the smaller graph you will get a tree decomposition you can combine the tree decomposition everything. But now it will because you are getting something like this you will get a tree decomposition of the eighth size. So, this is what the whole algorithm will be so though.

(Refer Slide Time: 41:02)



274 11 Tree Width

```
ROOTED-TDEC(k, 𝒢, W)
 // k ≥ 1, 𝒢 = (V, E) graph,
 // W ⊆ V with |W| ≤ 3k + 1
 1. if |E| > k · |V| then halt with "failure"
 2. if |V| ≤ 4k + 2 then
 3.     return 1-node tree decomposition of 𝒢
 4. else let W' ⊇ W with |W'| = 3k + 1
 5. if there exists a weakly balanced separation of W' then
 6.     let S be the separator of such a separation
 7.     let C_1, ..., C_m be the connected components of 𝒢 \ S
 8. else halt with "failure"
 9. for i = 1, ..., m do
10.     let 𝒢_i be the induced subgraph of 𝒢 with vertex set C_i ∪ S
11.     (T_i, (B^i_t)_{t∈T_i}) ← ROOTED-TDEC(k, 𝒢_i, (W' ∩ C_i) ∪ S)
12. Join (T_i, (B^i_t)_{t∈T_i}), for i ∈ [m], at a new root r with B_r = W' ∪ S
13. return the resulting tree decomposition
```

Algorithm 11.5.

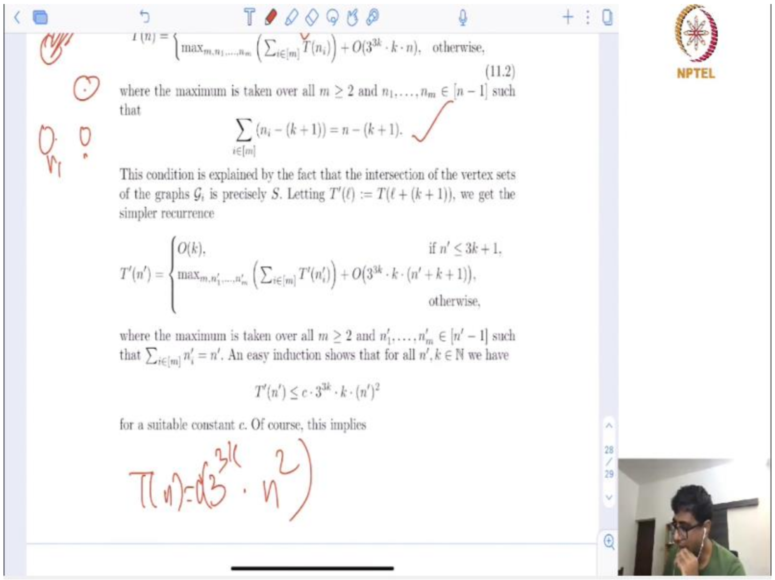
To simplify the analysis slightly, let us assume that the separator S computed by the algorithm has exactly $(k+1)$ elements. We can always increase the cardinality of the separator artificially by adding arbitrary elements of $V \setminus W$. We get the following recurrence for the running time $T(n)$:

$\{O(k)$, if $n \leq 4k+2$,

I have written down this whole algorithm so it is taken it is a picture from a book, but basically this is what the algorithm does. If the W is less than equal to 3, I mean basically this is exactly what this algorithm does is that like it. Actually, it is you can prove we do not need some 3k + 4 but with 3k + 1 itself things would hold and like you make it equal you find this what called weekly balance separator or W prime with two third.

Let S be such separator and this and this and you can achieve this. So, what is the running time in this algorithm? It is it is not precisely what we have learned but this should give you how these kind of routines are written these numbers could change here and there.

(Refer Slide Time: 42:00)



So, what is the running time of this algorithm? If the n is like at most some 4k + 3. We just returned one size tree decomposition one no tree decomposition. Otherwise, you know like otherwise we look at this graph induced on this union this. So, it is like summation of the running time to compute the tree decomposition of each of the G i. So, this is what is written here and now there was one running time to compute the particular W which is like 3 to the power 3k.

Because it is like it is like 3 to the power W and running time to compute the min balance separator but notice that whenever we had this, we actually took this so the size of the graph is basically n i is nothing but if I delete this this like n 1 so it is like n 1 + k n 2 + k so 1 and so forth. So, where this n 1 to m i are at most n - 1 and what is the property that if I delete n - i k + 1 then this equal to n - k + 1, because you have found k + 1 psi separator and so on.

So, you can simplify these recurrences using this and by induction you can actually prove that this is like T of n is like 3 to the power 3 big O of 3 to the power 3k n square. So, an existential algorithm all we needed to do in the existence algorithm to find a separator at some point of time. we do not know how to find that balance separator. So, we reduced that question of finding balance separator to like finding an S T separator still guaranteeing that the large fraction of W were not present.

So, everyone not contain we were not able to guarantee that they contain half of the fraction of W but we will say. But it does not contain everything it only contains two third fraction so because of that we were able to guarantee a constant factor approximation but slightly more like maybe like so existential of them says factor 3 approximation this told us factor 4 approximation. I hope the algorithm was clear you can do a running time analysis this this template was just given.

So, that you know how these things are but I took the liberty of like integers plus 1 plus 2 to make the lecture simpler and easier but I will still keep this page to just to help you how these things are how in general these things are how formally you can write the running time and how you can formally prove the running time using reduction hypothesis.