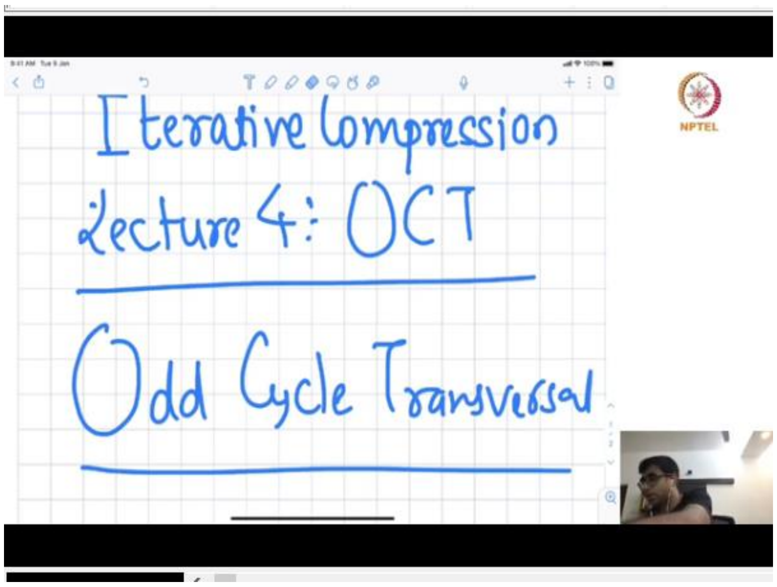
Parameterized Algorithms

Prof. Neeldhara Misra, Saket Saurabh
Department of Computer Science and Engineering
Indian Institute of Technology-Gandhinagar

Lecture-15
Iterative Compression IV: Odd Cycle Transversal

Hello, welcome to the last lecture on iterative compression.

(Refer Slide Time: 00:21)

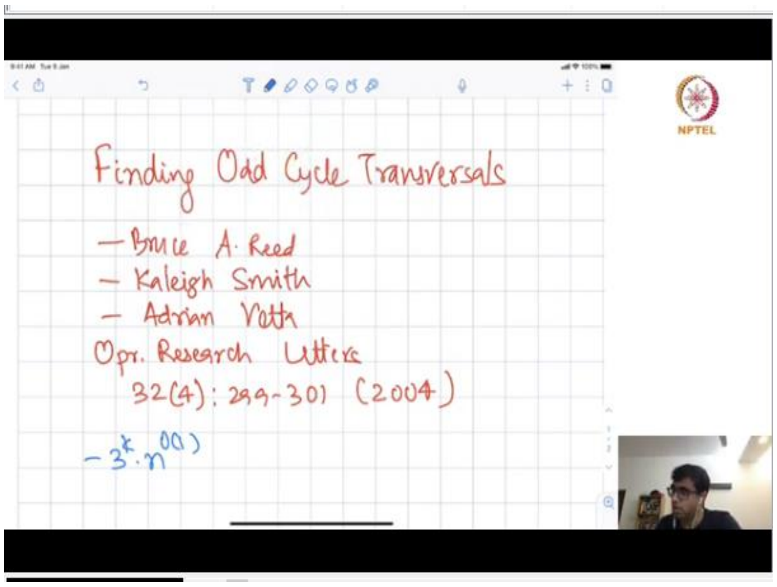


Iterative Compression
Lecture 4: OCT

Odd Cycle Transversal

This lecture we will do an algorithm for odd cycle transducer which I will define shortly, it is basically deletion to get a bipartite graph.

(Refer Slide Time: 00:29)



Finding Odd Cycle Transversals

— Bruce A. Reed
— Kaleigh Smith
— Adrian Vetta
Opr. Research Letters
32(4): 299-301 (2004)

$- 3^k \cdot n^{O(1)}$

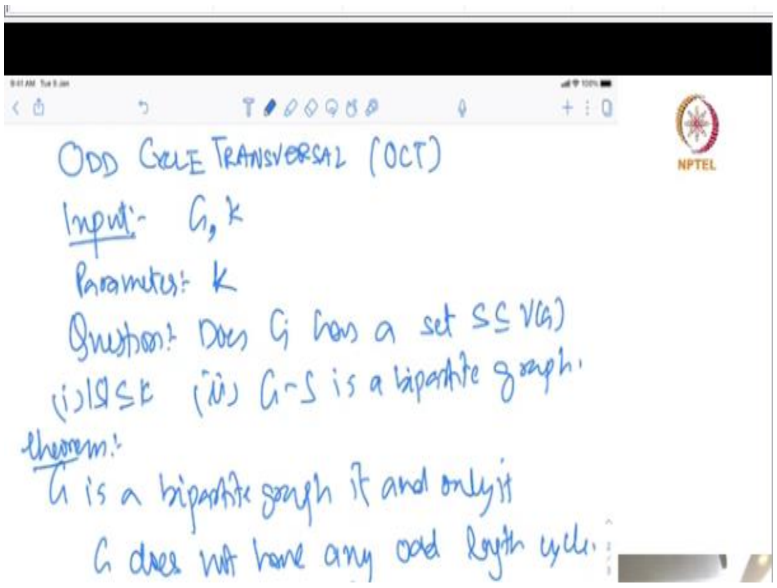
And actually, this is the paper which for the first time used the algorithm for which use the technique of iterative compression and this paper was called finding odd cycle transversal by

Bruce Reed, Kaleigh Smith, Adraian Vetta and appeared in journal operation research letters in 2004; though the preprint was available already starting from 2003 and so, it says a short 3 base paper, but it revolutionized or it gave a new technique to the field of parameterized complexity.

And so this algorithm ran in time 3 power k n to the power of O of 1. And this is an algorithm which we are going to talk today. And it is important to point out that whether an odd cycle transversal is fixed parameter tractable or not was an open problem for some time before this problem, this paper with all this in affirmative as well as for the first time as far as we are aware, use the iterative compression the way it is used.

Now, or the way we have been seeing in last 3 lectures. So, without any delay, let us just get started. So, the problem what we are going to deal with is the following.

(Refer Slide Time: 01:53)



ODD CYCLE TRANSVERSAL (OCT)

Input:- G, k

Parameter: k

Question: Does G has a set $S \subseteq V(G)$

(i) $|S| \leq k$ (ii) $G - S$ is a bipartite graph.

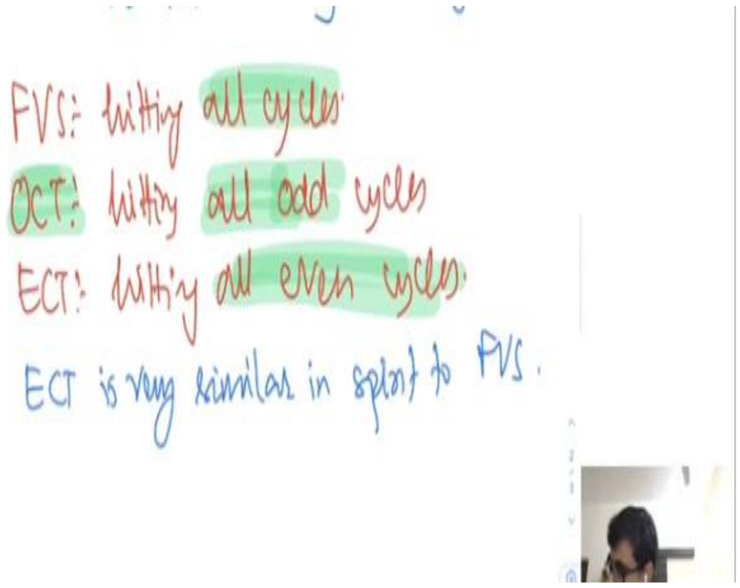
theorem:

G is a bipartite graph if and only if

G does not have any odd length cycle.

So, the problem today we will be dealing with in the whole lecture is one problem called odd cycle transversal and also called as OCT in short. So, input is going to be G, k, parameter is going to be k and the question is does G has a set S subset of VG mod S less than equal to k, second G - S is a bipartite graph . So, what is a bipartite graph? It is well known fact that this is a well known theorem in the literature that in a graph theory G is a bipartite graph if and only if G does not have any odd length cycle. Now, so, if you recall feedback vertex set is a problem.

(Refer Slide Time: 03:21)



FVS: hitting all cycles

OCT: hitting all odd cycles

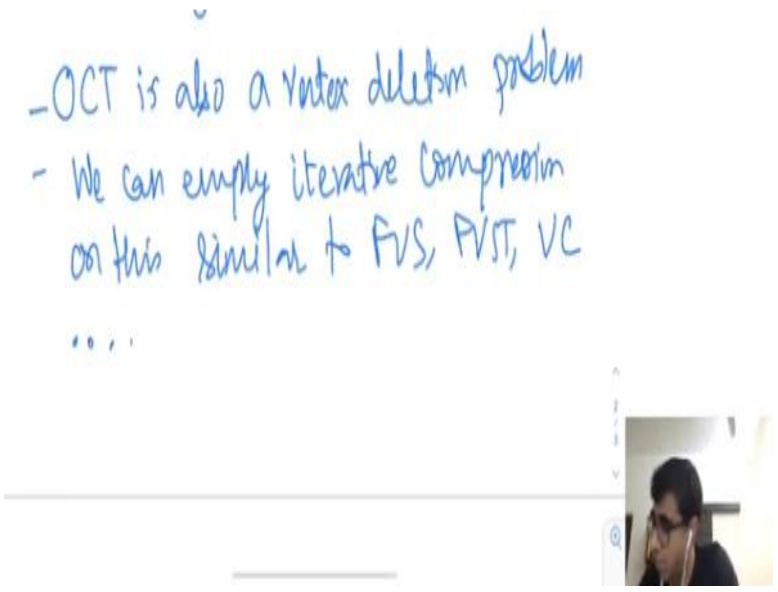
ECT: hitting all even cycles.

ECT is very similar in spirit to FVS.

So, feedback vertex set is a problem of vertical hitting all cycles. Because then when you can get a forest and OCT is about hitting all odd cycles. So, actually they are also probably called ECT which is hitting all even cycles, but so, OCT is about hitting all odds cycles, fvs is about hitting all cycles. ECT is about hitting all even cycles. And ECT is actually more like a feedback vertex set, but at least we will not go into in this course.

ECT is very similar in spirit to feedback vertex set. So, this is why the name came odd cycle transversal, transversal means something which hits something. So, you are looking for a set which hits all odd cycle transversal and hence the name odd cycle transversal or OCT in short term.

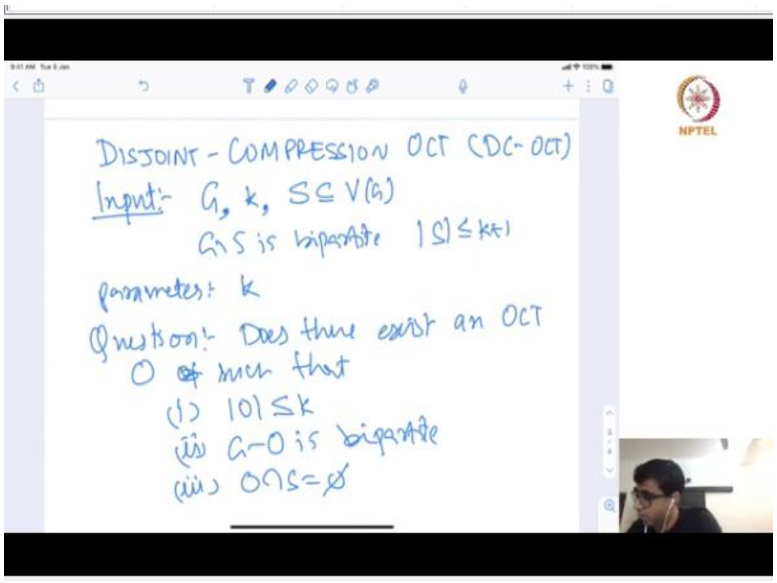
(Refer Slide Time: 04:45)



- OCT is also a vertex deletion problem
- We can employ iterative compression on this similar to FVS, FVST, VC ....

So, now, OCT is also a vertex deletion problem, we can imply iterated compression on this on this similar to a FVS feedback vertex set into num and vertex covered so on and so forth.

(Refer Slide Time: 05:27)



DISJOINT - COMPRESSION OCT (DC-OCT)

Input:- G, k, $S \subseteq V(G)$

$G \setminus S$ is bipartite $|S| \leq k+1$

parameter: k

Question:- Does there exist an OCT O ~~of~~ such that

(i) $|O| \leq k$

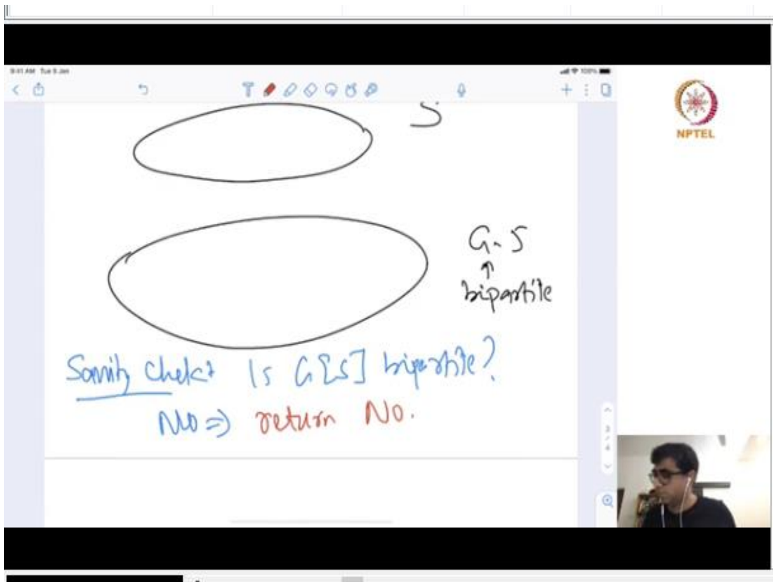
(ii) $G-O$ is bipartite

(iii) $O \cap S = \emptyset$

So, the problem which we will be interested in is straight away we will go to solving a disjoint compression, odd cycle transversal DC, OCT. So, what are we going to be input? Input is going to be graph G an integer k and S subset of vertex set of G, G - S is bipartite and mod S is less than equal to k + 1, parameter is k.

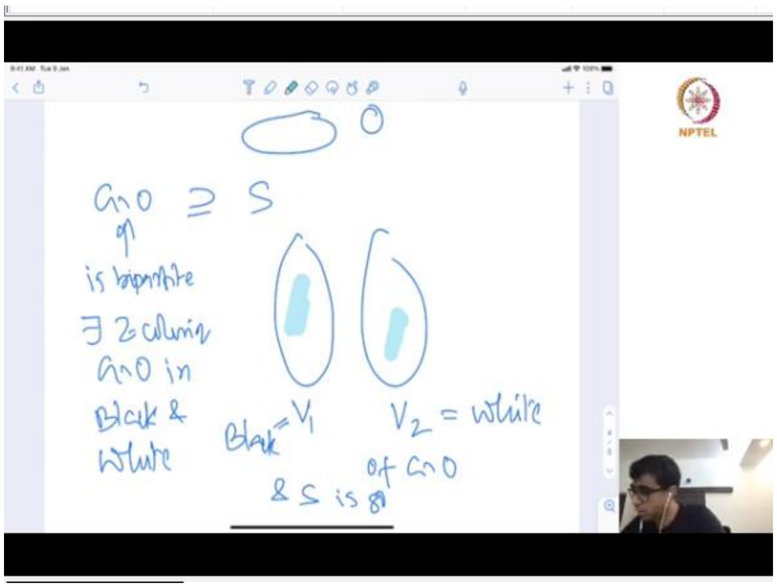
And the question we are asking does there exist on all cycle transversal O, such that first size of O that post k, second G - O is bipartite and third O intersection S is empty. So, this is why it is disjoint. So, I have been given a solution of size at most k + 1. And what I would like to do is to find a solution which is disjoint from this. So, if I look at picture wise so what are we given?

(Refer Slide Time: 07:18)



We are given here in S and here is G - S, which is bipartite and we need to find a solution from G - S. So, as always a sanity check, what I will do first let us do a sanity check, each graph induce an S bipartite, if not, then you need to at least any solution O that we are interested in must has to pick at least 1 vertex from S because there is a odd cycle that is completely contained inside the set S. So, no implies, you also say return. So, sanity check done.

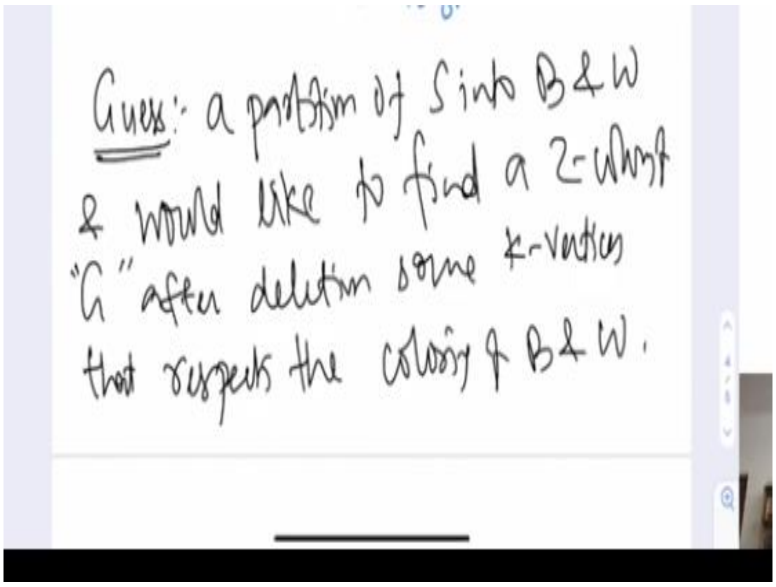
(Refer Slide Time: 08:21)



So, now what we know is a G of S is bipartite. Now, I know that S is not going to be part of the solution. So, what is going to happen is that this is my O and look at G – O, S is completely contained inside G - O. If S is completely contained inside G – O, then since G - O is bipartite. They are just a bipartisan V 1, V 2 of G – O and S is say bipartisan, say this is rather colouring is bipartite.

There exist 2 colouring of G - O into black and white. That said this is so V 1 is black, V 2 is white and S is some portion here, some portion here. S is divided into some portion of S becomes black and some portion becomes white. So, what are we going to do?

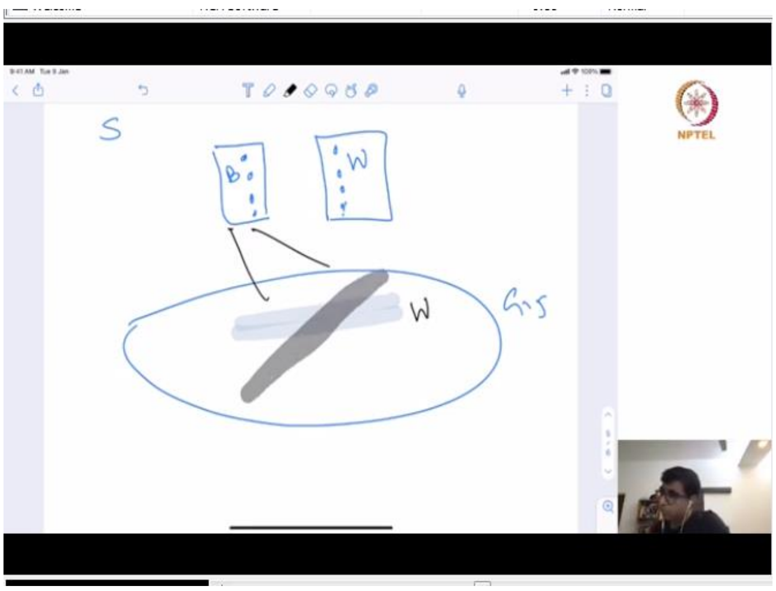
(Refer Slide Time: 09:53)



Guess: a partition of S into B & W & would like to find a 2-colouring "G" after deletion some k-vertices that respects the coloring of B & W.

We are going to guess a partition of S into B and W. This is we are going to guess. So, we are going to guess a partition of S into black and white and would like to find a 2 colouring of G what is my goal and my goal is to find a 2 colouring of G after deleting some k vertices that respects the colouring of B and W. So, now my world looks like. So, now let us look at how the world looks like.

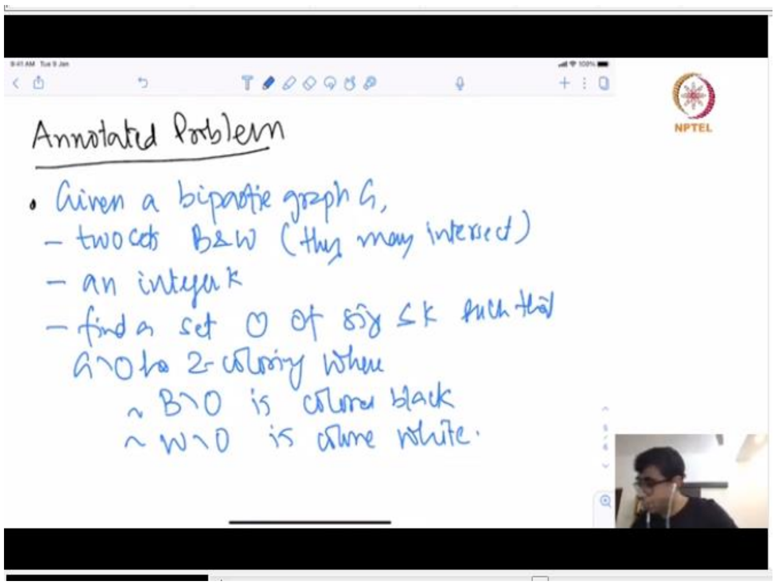
(Refer Slide Time: 11:04)



So, your S is here B some portion W and here is my bipartite graph G – S. Now that you have decided which vertices in S is going to be black look at this black and white. Now, what do you know about this? Look at the neighbours of B if we would like to propagate then, look at the neighbours of B black, here are neighbours of these guys, what should they get a coloured?

They should get a colour white. What should the neighbourhood have W get a colour? neighbourhood of W is let us say this is does not look like, let us say somehow white they should get white and look at the neighbour of white in the G – S, they should be getting a coloured black. So, actually we have reduced the following problem. So, what is the problem we have reduced? So, we have reduced to the following problem on bipartite graph. So, what we can do, let us see.

(Refer Slide Time: 13:10)



Annotated Problem

- Given a bipartite graph G,
  - two sets B & W (they may intersect)
  - an integer k
  - find a set O of size ≤ k such that G \ O has 2-coloring where
    - ~ B \ O is colored black
    - ~ W \ O is colored white.

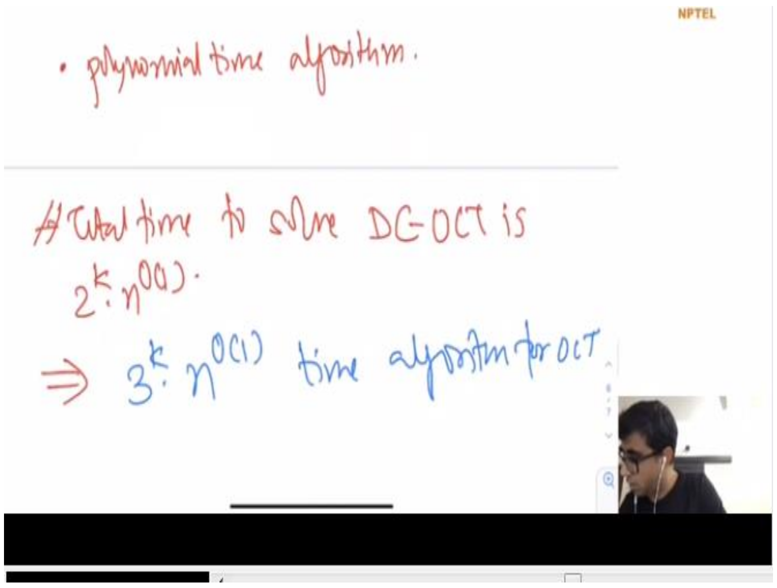
So, following annotated problem is what we have reduced our problem too, what is our given problem? Our given problem is we are given a bipartite graph G, we are given 2 sets B and W; they may intersect. I am not saying that they do not intersect. An integer k, that is what we are given, we have to find a set O of size at most k such that, says that G - O has 2 colouring, G – S has a 2 colouring what is the property?

Where B - O is coloured black and W - O is coloured white. So, what I am saying**,** fine. So, a moment to fix black and white of these vertices, which is fixed, then you know that look at the neighbours of the black vertices. They should either be deleted or they should be coloured white, so fine. So, that is what I have put my constraint on this.

Similarly, look at the neighbours of W in G - S. Since these guys are forced to be coloured white, their neighbours must be coloured black. Because this, so now that is what is so you are given a bipartite, graph G, you are given 2 sets of colours black and white and they may intersect, integer k value, your goal is fine, delete me some set of k vertices, that O, so, now, if I look at the 2 colouring of this G - O, then the vertices of B - O is really coloured black W - O is coloured white.

Because once this is done, then you can extend the colouring given by B and W 2 colouring of everything. So, we have reduced other problem on a general graph by guessing this B and W to an annotated problem and bipartite graph.

(Refer Slide Time: 15:59)



• polynomial time algorithm.

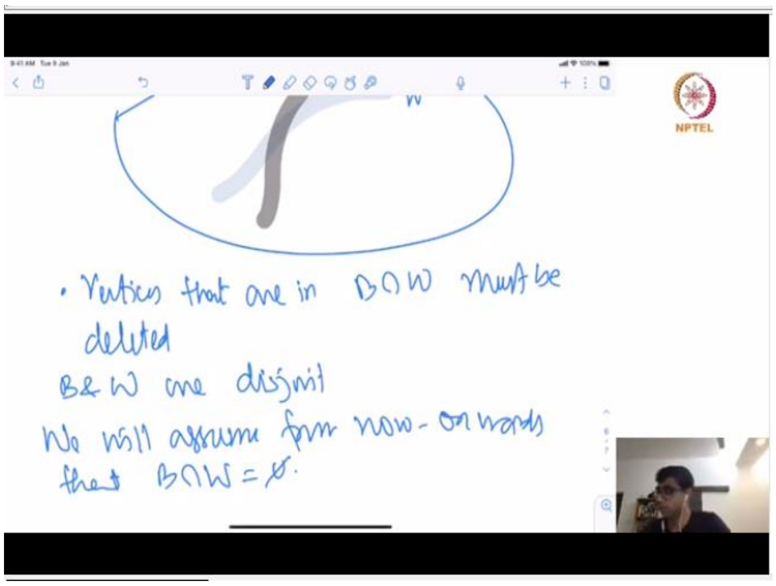
Total time to solve DC-OCT is $2^k \cdot n^{O(1)}$.

$\Rightarrow$ $3^k \cdot n^{O(1)}$ time algorithm for OCT

And for this, we are going to design a polynomial time, algorithm in a minute. Now notice, how many partitions are there? Number of partitions of S into B and W is 2 to the power k + 1. So, once I have guessed this, I am going to solve this problem in polynomial time, then the total time to solve disjoint compression OCT is 2 power k n to the power O of 1 which will imply 3 power k n to the power O of 1 time.

I will go to them for OCT, which implies that all we need is to be able to give a polynomial time algorithm for this annotated problem. Now let us look at how we are going to solve this annotated problem. So, let us focus on this annotated problem. So, let us just record this information in the annotated colour problem.

(Refer Slide Time: 17:28)



- Vertices that are in $B \cap W$ must be deleted

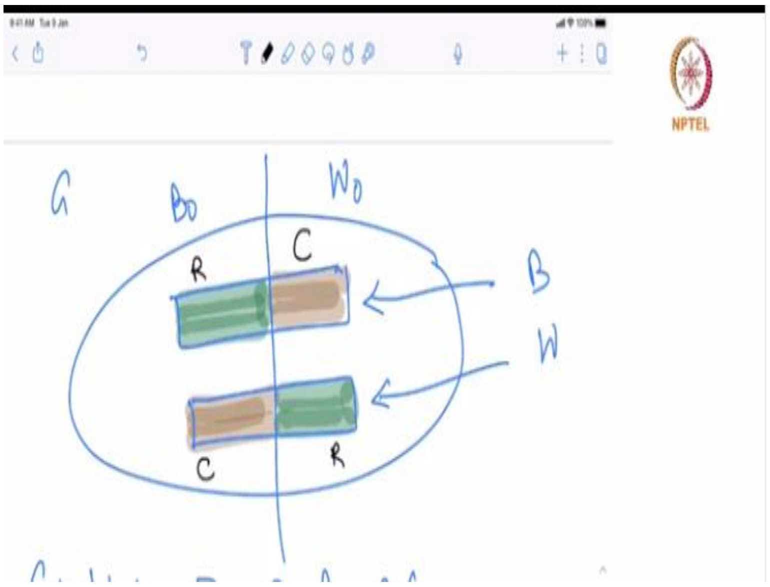
B & W are disjoint

We will assume from now-onwards that $B \cap W = \emptyset$.

So, now what we have is a set here, this is my white and this is my black, this is my black set and this is a white set. First of all, the vertices that are in B intersection W must be deleted, why because look if you do not delete them, then you have to keep them as black. As well as why that is not going to happen. So, you must delete them.

So, that implies that B and W are disjoint. So, we will assume from now onwards that B intersection W is empty. So, let us look at the picture again.

(Refer Slide Time: 18:49)



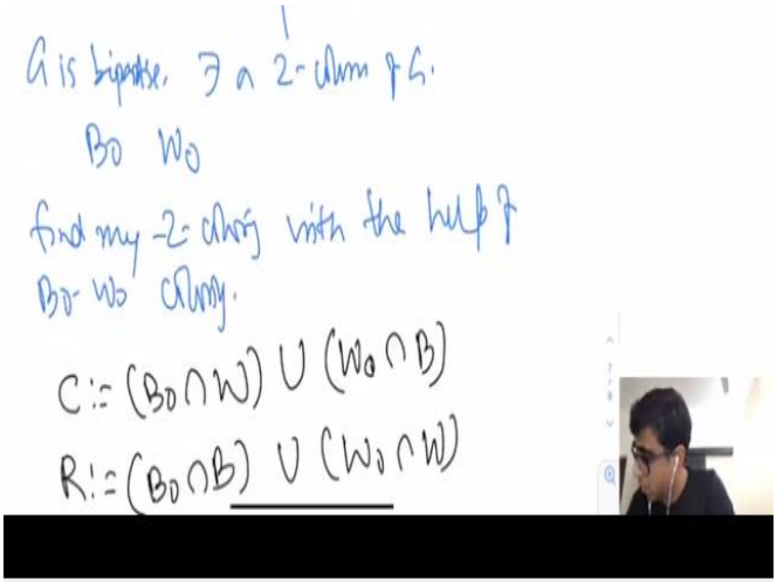
So, here is your bipartite graph. So, now we will come to that B and W and we have this set B and W. So, now we are going to treat them as a set B and W. Now, so this is my graph G, G is bipartite. What is the meaning of G is bipartite? There exist are 2 colouring of G. I am going to call that colour B 0 and W 0. Suppose this is my partition, B 0 and W 0. Now let us look at where the sets B are**.**

So, suppose this is my set B and this is my set W. Now, what you know, note this that, look my current colouring, so I am going to find my 2 colouring with the help of existing colouring, with the help of B 0 W 0 colouring. Now, what is B 0 W 0 colouring does? What it does is that for these vertices would like to be black but with a current colouring they are in some sense coloured white or W 0.

So, we know that these vertices would like to change their colours. Similarly, look at a W vertices, look at these vertices these has been coloured, these white vertices have been coloured potential v 0 or black vertices, but they would like to be white. So, they would like to find the 2 colouring where colours of these vertices should change, fine. Now look at the black vertices which has been potentially coloured black, but they are very happy.

So, look, keep us the way we are like green, where green good to go. And similarly, those vertices of W which has been good to go, they would like to be good to go. And I am going to call them as R vertices, like C represents for change, R represent for remaining. So, now what are the change vertices?

(Refer Slide Time: 21:58)



G is bipartite, $\exists$ a 2-colouring of G.

B_0 W_0

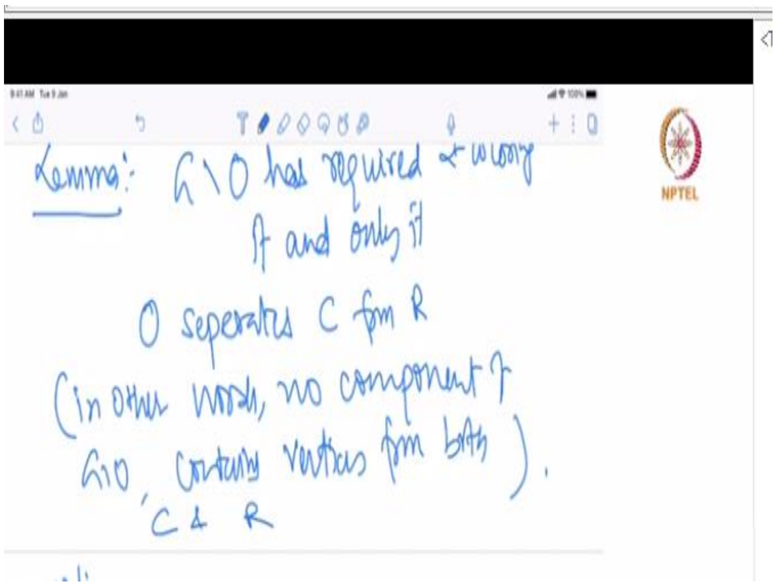
find my 2-colouring with the help of B_0-W_0 colouring.

$C := (B_0 \cap W) \cup (W_0 \cap B)$

$R := (B_0 \cap B) \cup (W_0 \cap W)$

Change vertices are B 0, black guys intersection W union W 0 intersection B and the remaining guys are B 0 intersection B union W 0 intersection W. So, those guys are of this nature. And the interesting thing which we are going to prove that to achieve such a colouring what are we going to show?

(Refer Slide Time: 22:38)

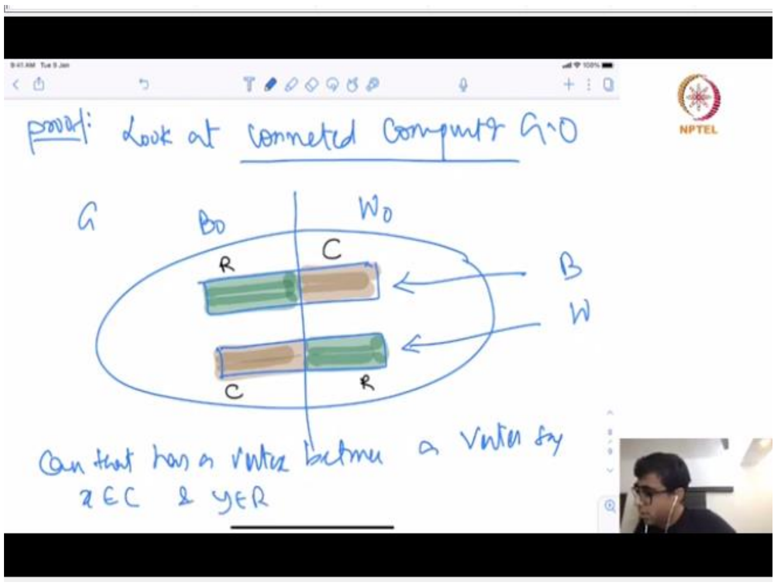


Lemma: G \ O has required 2-colouring if and only if O separates C from R (in other words, no component of G \ O containing vertices from both C & R).

We are going to show the following lemma which is a crux of the proof. And which is not very hard to show is that G – O has required. So, O is my solution. So, G – O has a required 2 colouring if and only if no components of G - O. So, if and only I would say O separates C or in other words, no component of G - O contains vertices from both C and that this is what the meaning of that.

So, let us let us try to prove this lemma. And then we will be done. So, I am saying look at any set O which does your job, then that is a property that G - O has required 2 colouring. If and only if all separate C from R. So, let us prove this. Because then it is all about finding separation from C to R, which we can do. So, now look at G - O has required 2 colouring. So, now G - O has the required 2 colouring. Now then have to show that O separate C from R.

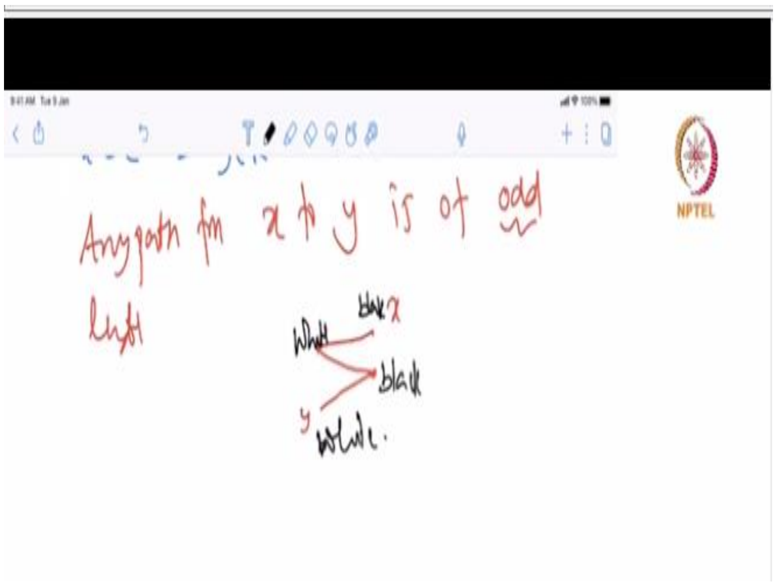
(Refer Slide Time: 25:00)


proof: Look at connected components G - O
G
B0
W0
R
C
B
W
C
R
Can that has a vertex between a vertex say
x ∈ C & y ∈ R
NPTEL

So, look at a connected component of G - O. So, let us take this picture, it will be very helpful. Look at a component of G - O. Now what is the property? Now let us ask ourself can that contain it is a connected component. Let us look at a connected component of G – O. Now let us ask ourselves can that has vertex between a vertex say x in C and y in R? So, let us just try to understand what happens.

So, suppose x is here and y is here. Now, what do you know? X wanted to x is there. Now, let us see what happens. Now, look at the vertex y in this part of the R.

(Refer Slide Time: 26:27)

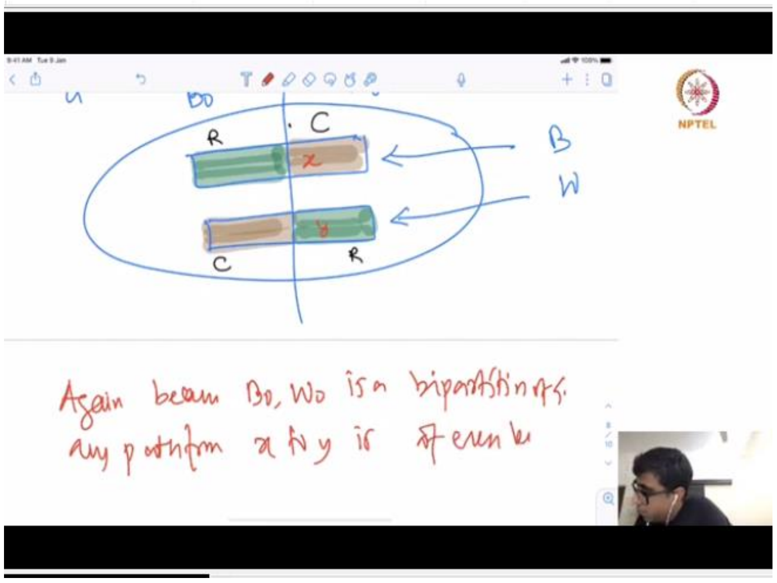


Now, since B 0 W 0 is a bipartisan any path from x to y is of odd length and by odd I mean number of edges. Because why because x has to go to the other side, either this is y or he has to come back again because there are no edges inside graph induced and digital has no edges.

Graph induced and W 0 is no edge, every edge is going across. So, now what is the point then?

Look below that it is a value to colouring. So, I started from x here, so he needed to change its colour. So now this coloured this guy has become black. So, in that component this is white, then this is black and this is white. But what did y wanted. Y wanted to be a black. So, this kind of path cannot exist.

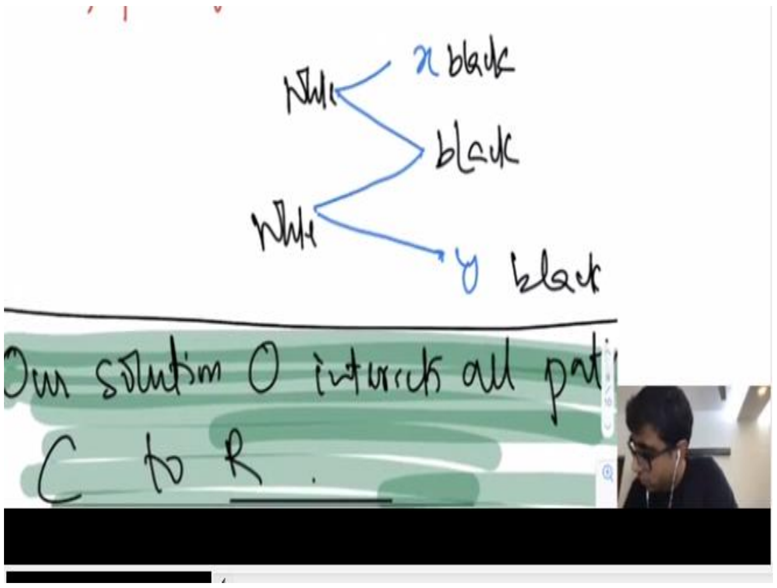
(Refer Slide Time: 27:39)



Again because B0, W0 is a bipartition of G. any path from x to y is of even length

So, let us look at here, look at the other case, what about x here and y here, other cases are just symmetric. Now, notice again because B 0 W 0 is a bipartition of G, look at the path, any path from x to y is of even length, even again number of edges.

(Refer Slide Time: 28:24)



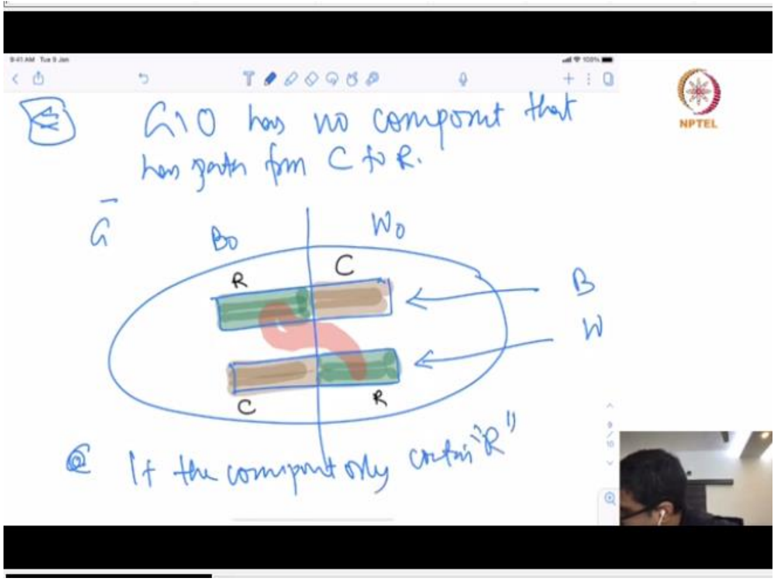
Our solution O intersects all path C to R.

So, now look at x go here that because then either this is a y or I go again to the other side and come again and suppose I get y. Now notice what colour is x? x wanted to change its colour. So, x has become black. What is this colour? White. What is this color? Black. What is this colour? White. What is this color? Black. So, every vertices, which can be reached by odd path gets a white colour and this black.

But what is the property of this particular y? This guy is in, remain that is it wanted to be white, but now you have made him black. So, these kinds of parts are not possible if we have valid 2 colouring. So, what is the meaning of that which means that our solution O vertical intersects or intersects all paths from change to remain? So, this is one direction of our proof. What is the other direction tells us? Other direction tells us suppose O separate C from R then I am saying that look if you find a smaller set of vertices which separates C and R.

Then that is good enough for you because then I can find this 2 colouring. So, this proof tells us that look every set which does this job is a separator from C to R and every like your solution is a separator from C to R and in fact every separator gives you a solution. So, in this case, just find a minimum size separator and you should be done. Now, let us look at.

(Refer Slide Time: 30:32)

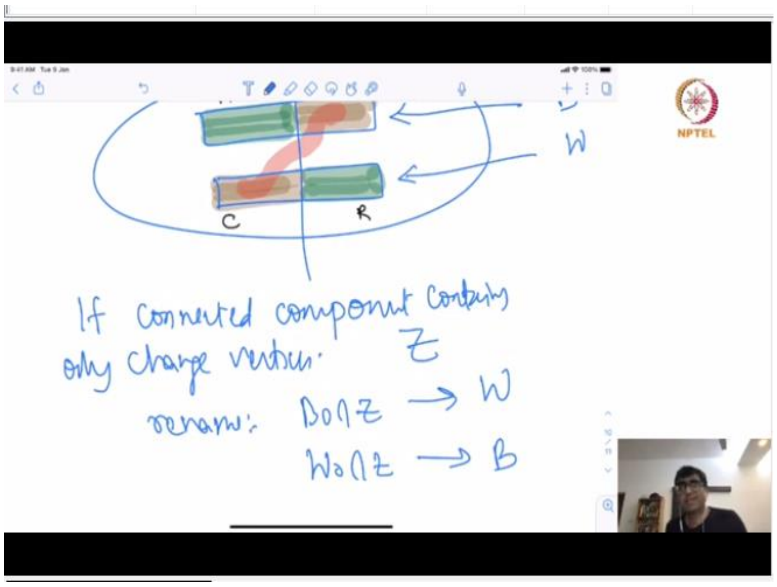


$G \setminus O$ has no compont that has path from C to R.

@ If the compont only contain "R"

Now what I know that in we have a G - O has no component that has path from change to remain. Now how the connected component here? Let us look at the connected component here. Now, any connected component here is going to contain let us see some vertex from here and some vertex from here or rather let us say or rather it is going to contain some vertex from the remaining maybe some vertex from here and some vertex from the lay. But look at this component, if this component say if the component one only contains R vertices remain vertices leave the colouring, do not change the colouring.

(Refer Slide Time: 31:51)



If connected component contains only change vertices. Z

rename: $B_0 \cap Z \rightarrow W$

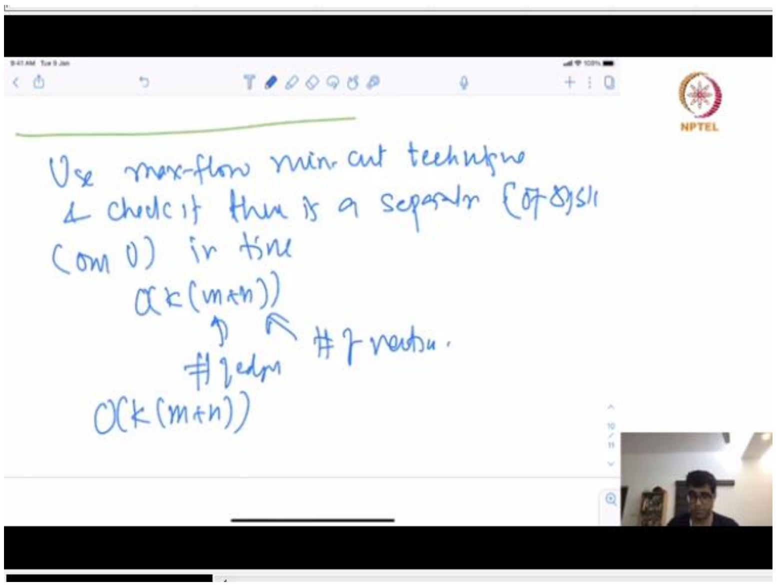
$W_0 \cap Z \rightarrow B$

But what could happen is that my component only contains change. So, it contains somebody says from here and somebody says from here, but now, this is how the by partition look like, I mean look at the induced by partition, it only contains change vertices. So, what you will do? But they are different connected components, if connected component contains only change vertices what will you do or rename B 0 to W and W like suppose this component is Z, then what will I do?

So, we are going to rename B 0 intersection Z to W and B 0 and W 0 intersects Z to black. That is it and then you have got a colouring for everything. So, look at a connected component if it is a remained do not change the colour. If it only contains vertices from the change time then just change the colours. Now, it is consistent and what is the property you have achieved that after you have deleted these vertices, you have a valid 2 colouring of my bipartite graph.

Such that whatever vertices that you wanted to be black, they are black, whatever vertices of W that remains or white and whatever if your black remains are black and that is what we wanted. So, now, because of this lemma, what we have been able to show to you is that if you are looking to find a required 2 colouring in this bipartite graph, it is enough for us to find separate C from R. And how can we achieve that to achieve that we are just going to apply.

(Refer Slide Time: 34:16)



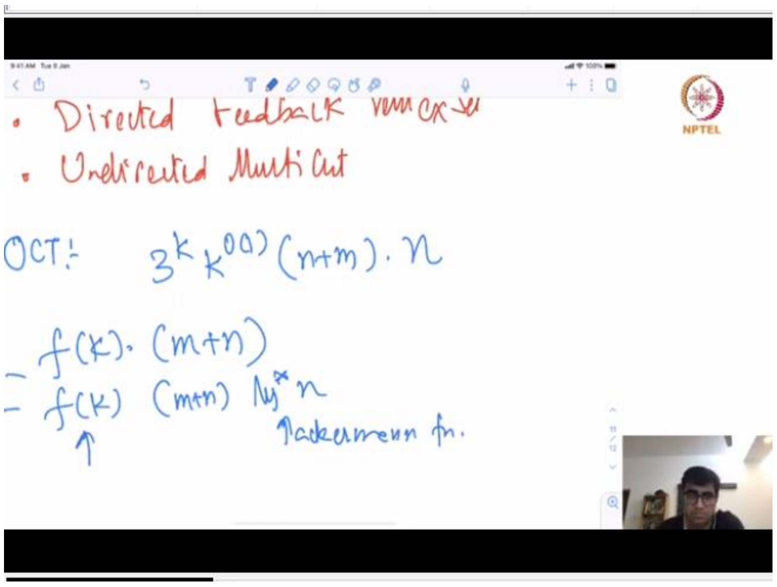
Use max-flow min cut technique & check if there is a separator of size $\leq k$ (on U) in time $O(k(m+n))$

$\uparrow$ # of edges $\uparrow$ # of vertices.

$O(k(m+n))$

So, now that we have proved this lemma, all we need to say is that now we can use max-flow min-cut and technique. And check if there is a separator phi at most k our O in k times order k times m + n with number of edges, number of vertices. So, in order k m + n time, we can check whether our annotated graph, bipartite graph problem is feasible or not I guess instance.

(Refer Slide Time: 35:17)



- Directed Feedback vertex set
- Undirected Multi Cut

OCT: $3^k k^{O(1)} (n+m) \cdot n$

– $f(k) \cdot (m+n)$

– $f(k)$ $(m+n)$ $\alpha(n)$

$\uparrow$ Ackermann fn.

Which immediately implies that we can solve odd cycle transversal in 3 to the power k times 3 to the power k n to the power of the O of 1. So, that was the last lecture on odd cycle transfer on the iterative compression and this method has been used to set parameterized complexity of several other problems like directed feedback vertex set, undirected multi cut and several other problems.

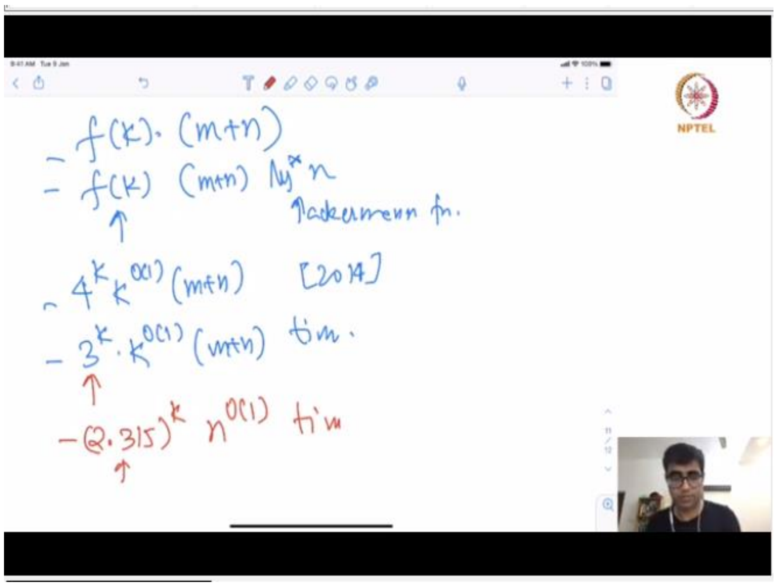
But, we will be talking about directed feedback vertex set in one of the lectures later, because not only it requires the technique, which we have seen of iterative compression, it also requires techniques what is called important separators and with that will only be introduced in later weeks. And when we will do that, we will be giving an algorithm for directed feedback vertex set.

And it was a very big open problem with the directed feedback vertex set does admit an 80 algorithm or not. And it was only settled in 2007 or 2008. So, before I go I like to make a remark is that if you notice that, OCT algorithm actually runs in time 3 to the power k sum k to the power of O of 1 and actually m + n times n for the iterative compression. And it was a very big open problem whether we could get an algorithm with running time F of k times linear dependence.

So, in the first paper of this nature, so, look so the reason why I am trying to tell you this is that there is also a lot of importance in on reducing the dependence on input size. So, classically, these are the kind of albums which were told linear time or quadratic time or of this nature. So, getting a linear time algorithm for every fixed k was an important open problem for OCT.

And first author's got f of k times m, say log star m**.** So m + n log star n, so it is like small inverse of like Ackermann function, but this f of k was very bad.

(Refer Slide Time: 37:56)



- $f(k) \cdot (m+n)$
- $f(k)\ (m+n) \log^* n$

$\uparrow$ $\quad$ Ackermann fn.

- $4^k k^{O(1)} (m+n)$ $\quad$ [2014]
- $3^k \cdot k^{O(1)} (m+n)$ tim.

$\uparrow$

- $(2.315)^k\ n^{O(1)}$ tim

$\uparrow$

Later a 2 group of people got an algorithm with running time this in 2014 and the current best algorithm runs in time 3 to the power k, k to the power of O of 1 m + n time. So, this is a history on this problem. And there has also been a lot of work to improve the base of the exponent and the best known base of the exponent algorithm is runs in time I think 2.3 something to the power k I do not remember very well.

And but this is not linear time, this is I would say some enter about different term, I think 2.3 or some 3 some numbers here I do not remember and but this is based on this algorithm is based on LP branching, the kind which you saw in the last lecture for vertex cover guarantee. And so it is basically it is essentially making that algorithm what we call for vertex covered a little bit more specialized and utilize the structure that comes from vertex cover.

And then, because the known reductions from odd cycle transversal to vertex cover guarantee problem, we immediately get an algorithm for odd cycle transversal running in time some 2.3 something, which I do not remember, maybe let me check. So it is 2.315 so for completion, so I think that is just ends the lectures on iterative compression. And in the next week, you will be seeing colour coding or the randomized methods in parameterized algorithm followed by decomposition based on them. So, thank you.