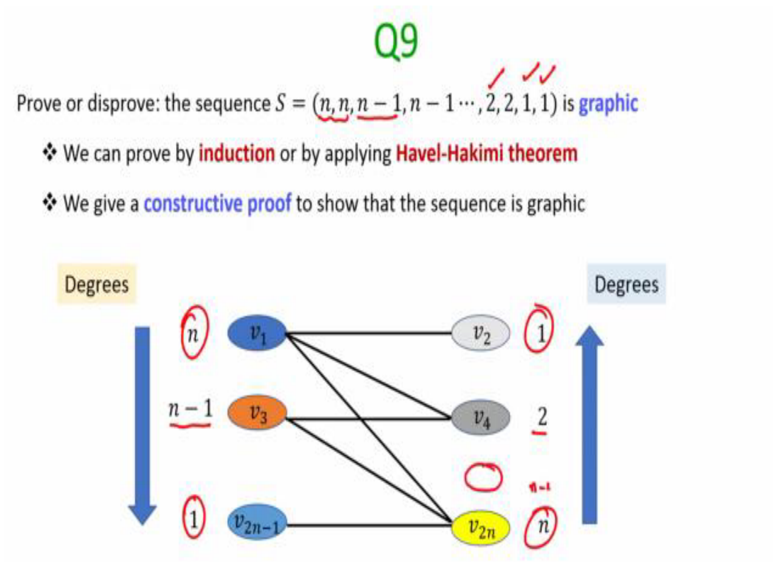


Q9

Prove or disprove: the sequence $S = (n, n, n-1, n-1 \cdots, 2, 2, 1, 1)$ is graphic

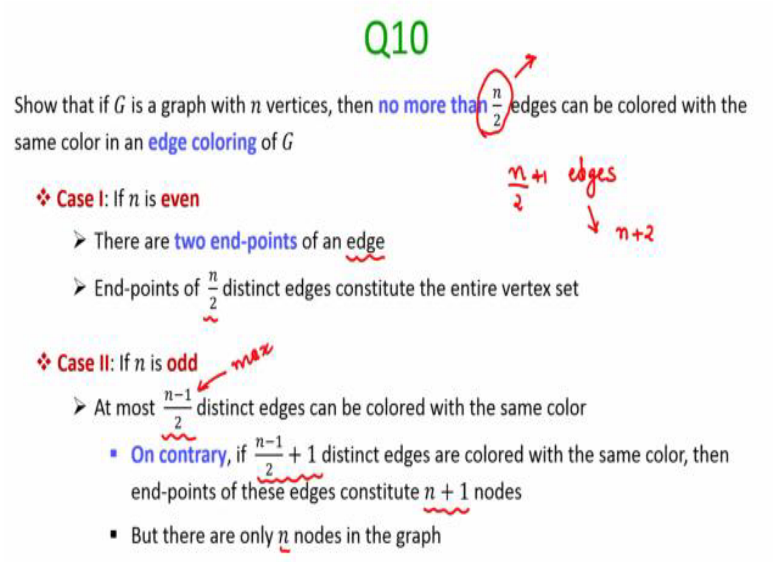
- We can prove by induction or by applying Havel-Hakimi theorem
- We give a constructive proof to show that the sequence is graphic

In question number 9, we want to either prove or disprove whether the sequence is a graphic sequence. So, there are 2 options: we can use either Havel-Hakimi theorem or we can use a proof by induction to prove that this sequence is a graphic sequence, but we will give a constructive proof to show that the sequence is graphic by showing a graph, a simple graph with 2n nodes whose degree sequence is same as S.

So here are the vertices: 2n vertices and what I do is the following. I take the vertex v_1 and add the edge with all vertices with even indexes. I take the vertex v_3 and I add an edge with all even index vertices except the vertex v_2 and I keep on doing this process and for the last vertex with odd index, I will give only 1 edge namely an edge with the last vertex with even index.

Now, what I can say about the degrees of the respective vertices here, so it is easy to see that these 2 vertices will have degree n so indeed, I need 2 vertices of degree n. I will have this vertex of degree n - 1 and this vertex of degree 2, so I got 1 vertex of degree n - 1 and 1 vertex of degree 2. And if I continue, I will find that I will get 2 vertices of degree 1 and then eventually I will obtain the second vertex of degree n - 1 and so on. So the vertex here will be of degree n - 1 and so on, that is a very simple construction to show that the sequence is a graphic sequence.

(Refer Slide Time: 30:07)



Q10

Show that if G is a graph with n vertices, then no more than $\frac{n}{2}$ edges can be colored with the same color in an edge coloring of G

$\frac{n}{2}+1$ edges

$\downarrow$ $n+2$

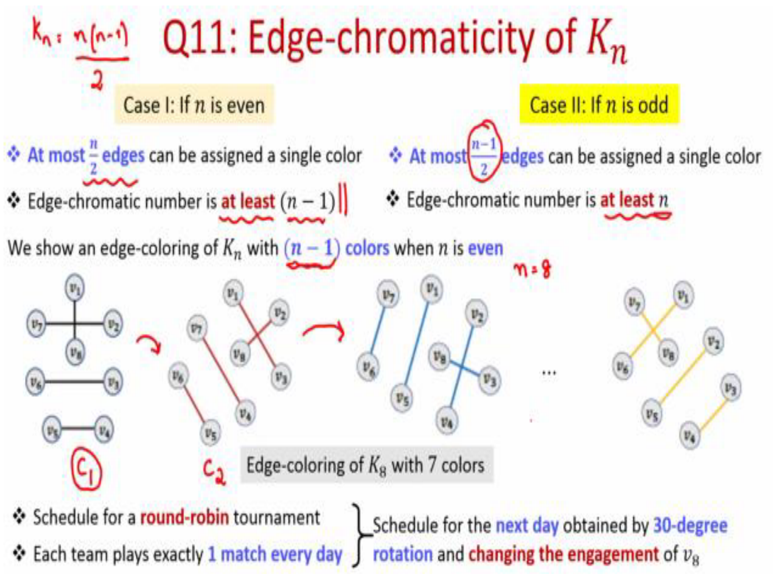
- ❖ Case I: If n is even
  - ➢ There are two end-points of an edge
  - ➢ End-points of $\frac{n}{2}$ distinct edges constitute the entire vertex set
- ❖ Case II: If n is odd
  - max
  - ➢ At most $\frac{n-1}{2}$ distinct edges can be colored with the same color
    - On contrary, if $\frac{n-1}{2}+1$ distinct edges are colored with the same color, then end-points of these edges constitute $n+1$ nodes
    - But there are only n nodes in the graph

Now let us come to question number 10. Here, we want to prove that if you are given a graph with n vertices, and if you are doing edge colouring, then you cannot use a single colour to colour more than $\frac{n}{2}$ edges. And it is a very simple fact, depending upon whether your n is odd or even, we can prove this very easily. So let us take the case where n is even. So remember, each edge has 2 endpoints.

That means if I consider $\frac{n}{2}$ distinct edges of the graph, and if I focus on their endpoints, that will constitute the entire vertex set. So that means I cannot colour $\frac{n}{2}+1$ edges with the same colour, because if I do that, then their endpoints will give me n + 2 vertices, but my graph at the first place has only n vertices. So at most I can colour $\frac{n}{2}$ edges, distinct edges with the same colour, I cannot colour more than those many edges.

Whereas if n is odd, then this quantity $\frac{n}{2}$ is not well defined, it will not be an integer value. So $\frac{n}{2}$ in that context of an odd value of n will be $\frac{n-1}{2}$. And indeed, it is easy to see that I cannot use a single colour to colour more than $\frac{n-1}{2}$ number of distinct edges, because if I try to do that, say for instance, I tried to colour $\frac{n-1}{2}+1$ number of distinct edges, then their endpoints will give or constitute will n + 1 nodes, but my given graph has only n nodes. So that is a maximum number of distinct edges, which can be coloured with a single colour.

(Refer Slide Time: 32:04)



$K_n : \frac{n(n-1)}{2}$

Q11: Edge-chromaticity of K_n

Case I: If n is even

❖ At most $\frac{n}{2}$ edges can be assigned a single color

❖ Edge-chromatic number is at least $(n-1)$

Case II: If n is odd

❖ At most $\frac{n-1}{2}$ edges can be assigned a single color

❖ Edge-chromatic number is at least n

We show an edge-coloring of K_n with $(n-1)$ colors when n is even

$n=8$

C_1 C_2 Edge-coloring of K_8 with 7 colors

❖ Schedule for a round-robin tournament

❖ Each team plays exactly 1 match every day

Schedule for the next day obtained by 30-degree rotation and changing the engagement of v_8

Now, based on these information, I will try to solve question 11 where I want to find the edge chromatic number of a complete graph with n nodes and my solution will be divided into 2 cases depending upon whether my n is odd or even. So, remember, from the previous question, I know that if your n is even then you can use 1 colour and colour at most $\frac{n}{2}$ edges, whether indeed you will be able to colour $\frac{n}{2}$ edges or not that depends upon the structure of your graph, but at max you can colour $\frac{n}{2}$ edges using a single colour.

Now in a complete graph, k_n I have $\frac{n(n-1)}{2}$ number of edges. So, with colour number 1, I can take care of at most $\frac{n}{2}$ edges. With colour number 2, I can take care of another set of $\frac{n}{2}$ edges. So, like that, how many colours I will require at least? So, I will require at least n - 1 number of colours, because to the first colour, I can take care of $\frac{n}{2}$ edges, the next colour I can take care of another bunch of $\frac{n}{2}$ edges and I have to take care of n - 1 such bunches of $\frac{n}{2}$ edges.

So, that is why the minimum number of colours that will be required will be n - 1. Whereas if I take the case when n is odd, then from my analysis of question 10, I know that through 1 colour I can take care of at most $\frac{n-1}{2}$ number of edges. And I have to take care of n bunches of $\frac{n-1}{2}$ number of edges. So, that means I will require at least n colours if my n is odd.

Now, what I will show is I will show that these bounds on the edge chromatic numbers, which were the lower bounds because they were the least number of colours which are required, they are actually tight in the sense I will give you a constructive colouring, a concrete colouring for colouring the edges of a complete graph with n nodes where n is even. And where the number of colours required is exactly n - 1.

And you cannot beat this bound because the lower bound says you will need at least n - 1 colours. So that is why I am giving you an optimal colouring. So let me demonstrate the colouring assuming the value of n = 8. So remember, edge colouring here corresponds to scheduling of a round robin tournament. So we have 8 teams, and we have to schedule matches among the teams.

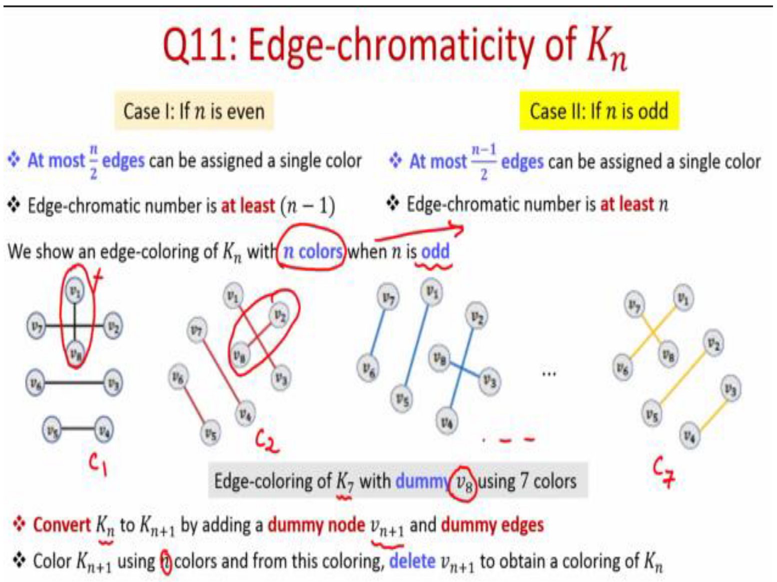
And the requirement is that each team has to play against every other team once but at the same time, we do not want to enforce a team to play more than a single match on any day. So the way I do the colouring here is as follows. So on the first day, I keep v_8 at centre and engage v_8 with v_1 and engage v_2 with 7, engage v_3 with 6, engage v_4 with 5. So, this is equivalent to saying that this colour c_1 is used to colour the edges between (v_4, v_5), (v_3, v_6), (v_2, v_7) and (v_1, v_8).

That means, I have coloured the maximum number of edges using colour number 1 and now, I have to use colour number 2 and using colour number 2, I will try to colour another set of 4 edges. So, which is equivalent to saying that I now want to find a schedule for the next day. So, the schedule for the next day is obtained by kind of rotating this diagram by 30 degrees and changing the assignment of or engagement of v_8.

So, earlier v_8 was engaged with v_1, but now v_8 will be engaged with the next team in the clockwise direction which is v_2. So, now, the assignment of the colours is the following. So, I use the colour number c_2 to colour these 4 edges, or equivalently I schedule these matches on day number 2, then again I shift it by 30 degree and change the engagement of v_8 . Now v_8 will be engaged with v_3 and so on.

So, now you can see that I have to do this rotation 7 number of times, and then I will be able to colour all the remaining edges of my complete graph with n nodes: 8 nodes.

(Refer Slide Time: 32:04)



Q11: Edge-chromaticity of K_n

Case I: If n is even

❖ At most $\frac{n}{2}$ edges can be assigned a single color

❖ Edge-chromatic number is at least $(n-1)$

Case II: If n is odd

❖ At most $\frac{n-1}{2}$ edges can be assigned a single color

❖ Edge-chromatic number is at least n

We show an edge-coloring of K_n with n colors when n is odd

Edge-coloring of K_7 with dummy v_8 using 7 colors

❖ Convert K_n to K_{n+1} by adding a dummy node v_{n+1} and dummy edges

❖ Color K_{n+1} using n colors and from this coloring, delete v_{n+1} to obtain a coloring of K_n

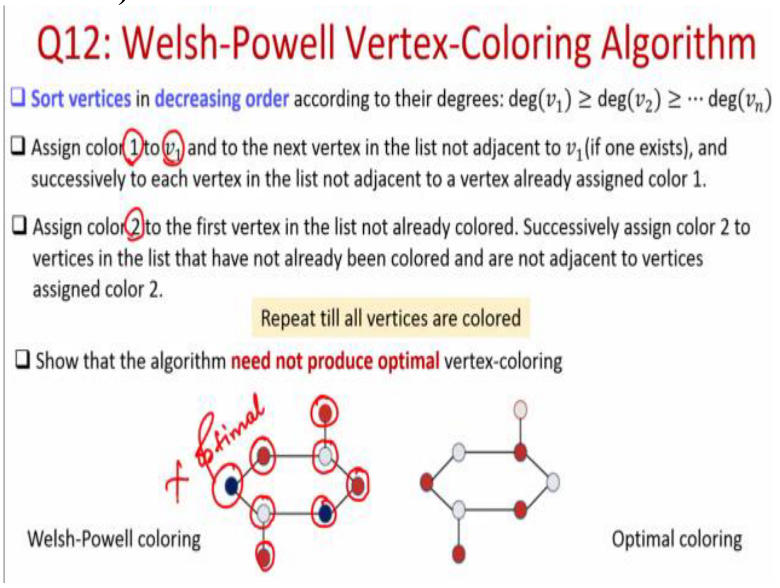
Now, let us see how exactly we can colour all the edges of a complete graph with n nodes where n is odd. And I will be using exactly n colours, which is optimal because the lower bound says that for n being odd, I need at least n colours. So, the idea here is I can convert the complete graph with n nodes to a complete graph with n + 1 node by adding a new vertex and the required dummy edges.

And since n was odd, n + 1 will be even. And I know a colouring mechanism to colour a complete graph with n + 1 nodes where n + 1 is even using n colours, namely the colouring that I had discussed just now. So, take that colouring and now you delete the dummy node and the corresponding edges. That will give you the colouring for the original complete graph with n nodes where n was odd.

So, for instance, what I am saying here is if you have only 7 teams, and you want to come up with a schedule, you imagine that you have included a dummy team, say the 8th team and now you want to come up with a round robin scheduled tournament for 8 teams with the same restrictions that you had earlier. So, this will be the schedule you will require 7 days. Now in the first day you can see that; on the first day you can see that v_8 is engaged with v_1.

So you can forget about that you can imagine that match is not going to be held; remaining other matches will be held as per the colouring assignment namely v_2 will play with v_7, v_3 will play with v_6 and v_4 will play with v_5 . On the second day v_8 is engaged with v_2. So, you can imagine that match will not be there actually and remaining 3 matches will be played and so on. So, this now gives you a colouring; edge colouring for complete graph with n nodes when n is odd.

(Refer Slide Time: 38:37)



Q12: Welsh-Powell Vertex-Coloring Algorithm

- Sort vertices in decreasing order according to their degrees: $\deg(v_1) \geq \deg(v_2) \geq \cdots \deg(v_n)$
- Assign color 1 to v_1 and to the next vertex in the list not adjacent to v_1 (if one exists), and successively to each vertex in the list not adjacent to a vertex already assigned color 1.
- Assign color 2 to the first vertex in the list not already colored. Successively assign color 2 to vertices in the list that have not already been colored and are not adjacent to vertices assigned color 2.

Repeat till all vertices are colored

- Show that the algorithm need not produce optimal vertex-coloring

Now, question 12 we are giving a greedy strategy for vertex colouring and we want to prove that this strategy need not give you the optimal vertex colouring. So, the colouring strategy is the following. We first sort the vertices according to their degrees and then use colour number 1 to colour the vertex which has the highest degree that you have arranged as per your degrees and then the next vertex in the list which is not adjacent to be v_1, if at all it exists and successively try to colour as many vertices as possible according to the colour number 1, keeping in mind that the next vertex which you are selecting is selected according to their degree. That means, you are following a greedy strategy and trying to occupy or colour as many vertices with that colour and now do the same process with the next colour and so on.

So now we have to give a counter example, namely a graph where Welch-Powell algorithm will end up utilising more colour than the optimal number of colours. So consider this graph and let us see how many colours we need. Actually we need 4 colours as per the Welsh-Powell algorithm because this vertex has the highest degrees so I will colour it and then I can assign the same colour to this vertex which has also the same degree.

And now I cannot use the same colour to colour any other vertex. Now, I will focus on the next set of vertices which has the highest degree. So let us use this vertex, this vertex and then the same colour I can assign to this vertex and this vertex. So that is the maximum number of vertices which I can colour with the second colour. Now among the remaining vertices I will pick the vertices which have according to their degrees.

So I can pick this vertex and the same colour can be assigned to this vertex. So we need total 3 colours; but optimal colouring is 2 this will require only 2 colours and 2 colours will be sufficient to colour all the vertices in this graph so that shows this is not optimal colouring, so with that I conclude this tutorial. Thank you.