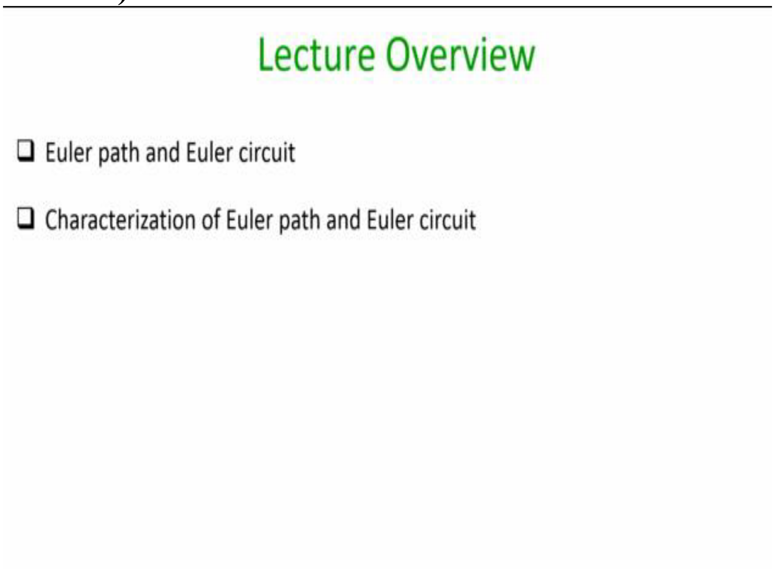
Discrete Mathematics
Prof. Ashish Choudhury
International Institute of Information Technology - Bangalore

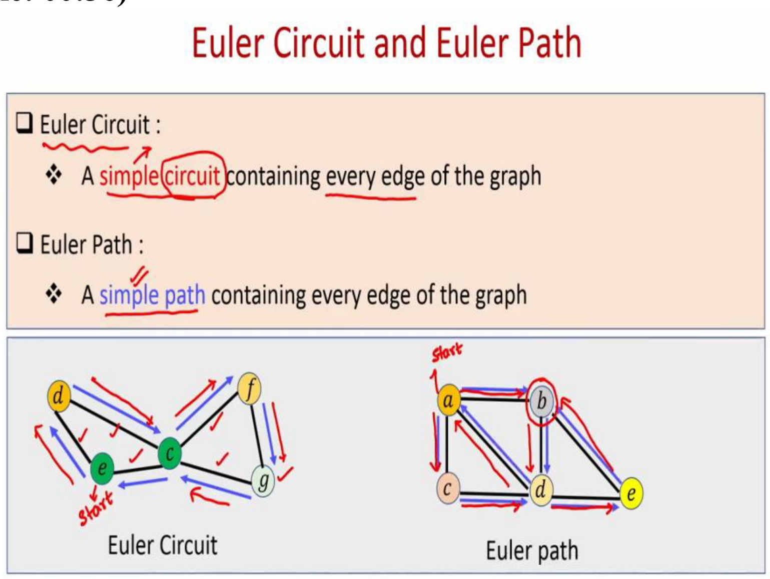
Lecture - 38
Euler Path and Euler Circuit

(Refer Slide Time: 00:25)



Hello everyone welcome to this lecture the plan for this lecture is as follows. In this lecture we will discuss about Euler path and Euler circuit and we will see the characterization for the existence of Euler path and Euler circuits in a graph.

(Refer Slide Time: 00:36)



So, let us start with the definition of Euler circuit and Euler path. So, imagine that you are given a graph then an Euler circuit is a simple circuit which contains every edge of the graph. So, since it is a circuit that means the starting point and the end point of the trail or the tour will be the same that means you have to start at the same vertex and you have to end at the same vertex in the tour.

And it is simple in the sense that during the tour the edges are not allowed to be repeated. So, it is a special type of simple circuit in the sense that it contains every edge of the graph; no edge of the graph will be absent in this simple circuit if you have the existence of such a simple circuit and the circuit will be called as an Euler circuit. And if you have an Euler circuit in your graph then the graph will be called as an Eulerian graph.

Whereas an Euler path is a simple path which contains every edge of the graph so the difference between Euler path and Euler circuit is that in the case of Euler path your starting point and end point are not same because it is just a path. However it is still simple and hence the edges are not allowed to be repeated. Whereas in the case of Euler circuit edges are not allowed to be repeated but you also need the fact that the starting point and end point should be the same.

So, let us see some examples of both these concepts. So, imagine this is a graph given to you then this graph has an Euler circuit and hence this graph will be an Eulerian graph. So, if you follow the tour along the blue edges or the blue arrows that gives you an Euler circuit. So, for instance suppose I start at e in fact you can start at any vertex and I first go from e to d that takes care of this edge then I go from d to c that takes care of this edge between d and c.

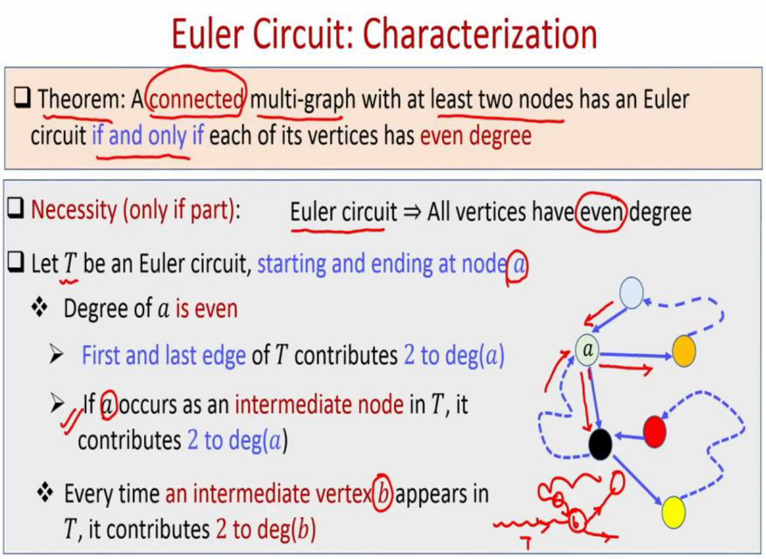
Then I go from c to f that takes care of this edge then from f I go to g that takes care of the edge between the node f and g then I go from g to c that takes care of this edge. And then finally I stop my tour by traversing the edge between c and e. So, you can see I started at e and ended my trip at e and in my tour all the edges of the graph are covered and no edge is repeated. Hence this is an example of an Euler circuit.

Whereas if you see this graph then it is easy to verify here that this graph does not have any Euler circuit you start at any vertex it is impossible to make a tour starting at the same vertex and ending at the same vertex and traversing every edge of the graph exactly once and without repeating any edge that is not possible. So, for instance let us try to make a tour starting from a so if I traverse from a to c and then if I go from c to d.

And then if I go from d to e and then from e to b and then if I go from b to d and then if I go from d to a. And then if I go from a to b by the time I reach b I have traversed all the edges but now you see that my current point is b and I started my tour at a. So, the requirement of Euler circuit is that tour should start and end at the same vertex. So, currently I am at b if at all I want to end my tour at a I will be repeating the edge between b to a.

And hence this tour will no longer be a simple circuit. However this graph has an Euler path because if you follow the tour along the edges highlighted in the red color then I have started the tour from a ended my tour at b and I have traversed all the edges of the graph exactly once. So, hence this graph is not an Eulerian graph because it does not have an Euler circuit but it does have an Euler path.

(Refer Slide Time: 05:33)



So, Euler gave a very simple necessary and sufficient condition according to which you can verify easily whether a given arbitrary graph as an Euler circuit or not. So, the theorem statement is the following. Imagine you are given a connected graph that is important and your graph need not be a simple graph it can be a multi graph by multi graph I mean that between the same pair of vertices you might have multiple edges.

So, the graph need not be a simple graph but still I can define the notion of Euler circuit and Euler path even for multi graph. So, imagine you are given a connected undirected graph which is a multi graph and the graph has at least 2 nodes because if my graph is just a single node then again the notion of Euler circuit does not make much sense there. So, imagine you are given a multi graph which is connected and it has at least 2 nodes.

Then what Euler proved is that the graph will have an Euler circuit if and only if each of the vertices in the graph has even degree. And this condition is both necessary condition as well as a sufficient condition because this is an if and only if statement. So, we will prove both the necessity condition as well as the sufficient condition. So, let us first prove the necessity condition namely the only if part.

And for that we have to show the following implication we have to prove that if your graph has an Euler circuit then it implies that each vertex of the graph has even degree you cannot have any vertex in the graph which has an odd degree. So, and it is very simple to prove this: so imagine, your graph. you are given an arbitrary graph which may not be a simple graph and imagine that a graph indeed has an Euler circuit.

So, I am calling an Euler circuit which is there in your graph by T. I am denoting it by T. So since it is an Euler circuit the tour T will start and end at the same vertex. So, I am denoting the starting point and the ending point of the tour by the node a. So, the first thing to observe here is that the degree of the vertex a in your graph will be even why so? Because the first edge of the tour will be incident with a, namely it will be an edge coming out or incident with a that means because your tour is starting from the node a.

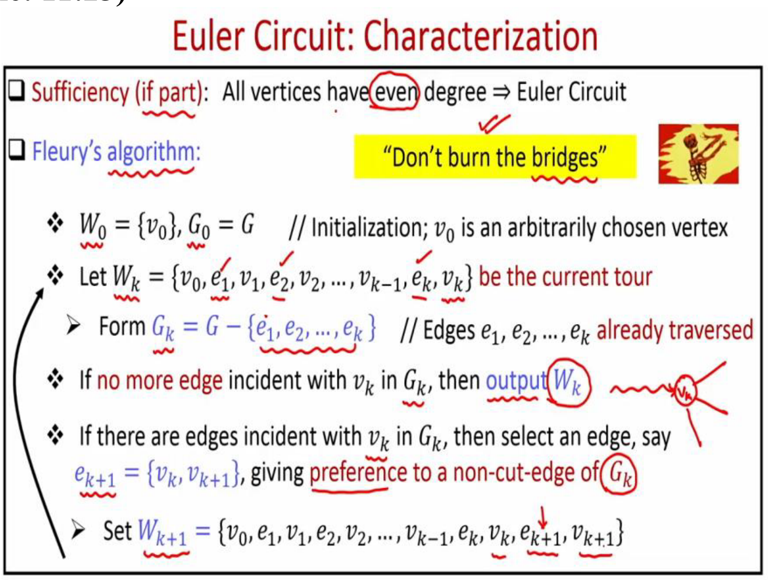
So, the the first edge in the tour which is incident with the node a will contribute 1 to the degree of a. And since your tour also ends with the node a that means last edge in the tour is also incident on the incident with the node a. So, that implies that definitely the degree of a is at least 2. And if your node a occurs as an intermediate node in the tour T then again it contributes 2 to the degree of a.

Because each time you will be entering the node a via some edge and you will be coming out of the node a in the tour. So, the edge through which you enter the node a in the tour that contributes 1 to the degree of a and edge through which you are coming out of the node a in the tour contributes again to 1 to the degree of a and it can happen multiple times. So, if your node a is appearing p number of times as an intermediate node in that tour, then the overall degree of a will be 2 times p and this is apart from the degree 2 which is contributed because the starting edge of the tour was incident with a and the ending edge of the tour is also incident with a. So, every time you enter the node a or a occurs as an intermediate node you are actually counting 2 to the degree of a and since my circuit T is an Euler circuit that means all the edges incident with a in my original graph will be covered; will be appearing somewhere in my tour T.

And as we have argued here each time the node a occurs in the tour we are actually counting 2 to the degree of a. So that shows that the degree of the node a will be even. In the same way I can argue that if you take any intermediate node b which is appearing in the tour it will contribute 2 to the overall degree of b because if your tour is, if this is a part of the tour where you enter the node b, and then by following some edge incident with b you go to some another node and then again suppose you come back to the node b through some edge and again you leave the node b. So, every time you enter the node b and you come out of the node b you are counting 2 for the degree of b. So, again following the same logic as we have used to argue that the degree of node a is even we can conclude that the degree of the node b also will be even.

And again since all the edges incident with the node b in your original graph will be appearing somewhere in the tour T that shows that the degree of the node b is even. So, necessity condition is very simple.

(Refer Slide Time: 11:23)



Now we will prove the sufficiency condition if part and for that we have to show the following: I have to show that if you are given a connected multi graph where all the vertices have even degree then there exists at least 1 Euler circuit in your graph there might be multiple Euler circuits also possible in your graph but at least 1 Euler circuit is definitely there. And for proving the sufficiency condition I will discuss here an algorithm called as Fleury's algorithm.

By running this algorithm on a graph we are the degree of every vertex is even we are guaranteed to obtain an Euler circuit and algorithm is very simple. And overall the main principle followed in this algorithm is that when you are trying to make a tour in the graph by using the Fleury's algorithm try to not burn the bridges; by bridges I mean the cut edges try to avoid traversing the cut edges until and unless it is not possible to avoid traversing the cut edges or the bridges.

So, the algorithm is an iterative algorithm because in each iteration we will be advancing our tour and after a certain number of iterations our tour will end and we will end up covering all the edges of the graph. So, we can start the tour from any vertex there is no restriction that you should start the tour only from a specific vertex you can pick any vertex to start your tour. So, the vertex from where I am starting my tour I am denoting it as v_0.

And I am defining a set W_0 which is initialized to the set v_0 my starting point of the tour and in each iteration I will be picking an edge which I will be traversing next in my tour and once I traverse that edge since I require overall; since my final output should be a simple circuit where the edges are not allowed to be repeated once I have traversed an edge in the graph I should not consider it in the future iteration.

So, I will keep on updating my graph once an edge is traversed I should remove it for further consideration. So, as a result my graph also will keep on getting updated. So my initial graph G_0 will be the input graph G itself because as of now I have not traversed any edge I have just decided the starting point of my tour. Now as I said the algorithm is iterative so imagine you have finished the k iterations.

So, right now k = 0 but imagine that you have already obtained a partial tour and your partial tour has already traversed k number of edges edge e_1, edge e_2 and edge e_k where the edge e_1 is incident with the node v_0 and v_1 the edge e_2 is incident with v_1 and v_2 and like that the edge e_k is incident with v_k - 1 and v_k. So, imagine that this tour has been this is the partial tour which you have already done through the Fleury's algorithm.

Now you have to decide what should you do in the next iteration. So, as I said your graph G also keep on getting updated because as you keep on traversing more and more edges those edges are removed from further consideration. So, since my edges e_1, e_2, e_k have been already traversed and covered in my tour I will be removing those k edges from my graph and updated graph is G_k. Remember by removing the edges we are not removing the vertices we are just removing the edges which we have traversed vertices remain as it is.

Now for the next iteration we will do the following. So, since my current tour has stopped at the node v_k I will check whether there are more edges to be traversed incident with the node v_k in my graph G_k that means I will just check whether there are any more edges left for traversing or not and that edge is incident with v_k or not. If there are no more edges left incident with the node v_k then I stop the algorithm and the tour W_k is my output tour.

We will argue later that indeed this tour is an Euler circuit. But suppose if there are still some more edges which are left which are not yet traversed and those edges are incident with v_k then I have to select the next edge incident with v_k for traversing. Now there might be 2 possibilities here if you have only 1 edge left in the graph which is incident with v_k you have no choice you have to traverse that edge because you have to ensure that that edge is covered as part of the tour.

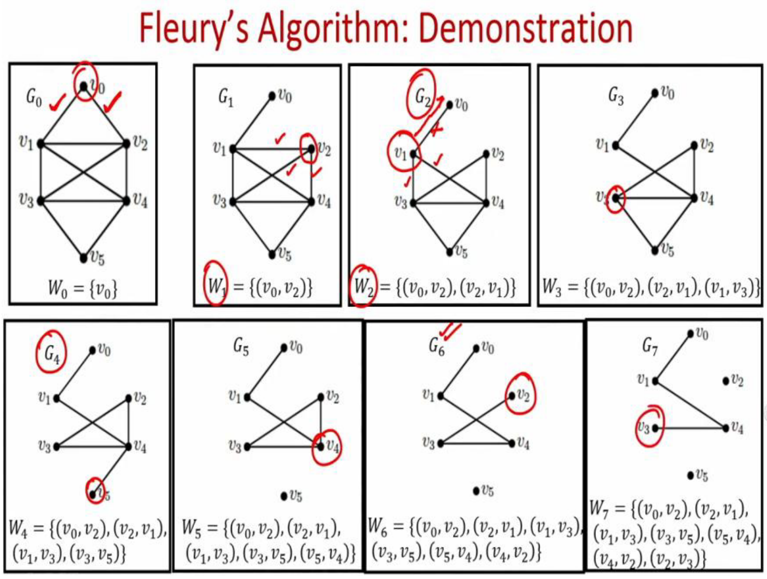
So, in that case you have no other choice; no other option. But imagine you are in a scenario where there are multiple edges which are still not traversed and incident with v_k then among all those edges which are still incident with v_k you should select your next edge for traversing. And you should give preference to a non cut edge of the graph G_k and that is what I mean by do not burn the bridges that means say you have reached the node v_k.

And you have multiple edges still left in the graph incident with the node v_k. So, what the algorithm says is you cannot arbitrarily choose any of those edges. Among all the edges which are still incident with v_k and not yet traversed, check which of the edges are non cut edges for the graph G_k I stress for the graph G_k because now your graph is G_k not the original graph because in the original graph you have already removed the edges e_1 to e_k.

So, if you have non cut edges still left in the graph G_k give preference to non cut edge but if you have no non cut edge available incident with v_k then feel free to use or traverse any of the cut edge incident with the vertex v_k that is what is Fleury's algorithms. So, once you have decided that edge e_{k+1} has to be traversed next by following the preference rule dictated by the Fleury's algorithm you will update your tour to W_{k+1}.

And that updated tour will now have this new edge e_{k+1} included incident with the nodes v_k and v_{k+1}. And then you will again go to the next iteration, that is the algorithm; a very simple algorithm.

(Refer Slide Time: 18:44)



So, let me demonstrate this algorithm so imagine this is a graph given to you. And you can easily check here that this graph indeed satisfies the sufficiency condition as dictated by the Euler's theorem because indeed every vertex in this graph has an even degree. So that means if I run the Fleury's is algorithm I should definitely get an Euler circuit. So, let us check whether we get an Euler circuit here or not.

So, I start with the node v_0 and there are 2 edges left incident with the vertex v_0 my graph G_0 is my original graph and both the edges are non cut edges for the graph G_0 so you are free to traverse any of them. So, suppose I traverse the edge between the node v_0 and v_2 and hence I remove that edge and I updated my tour. Now this is my graph G_1. Right now I am at the node v_2 and I have 3 edges which are not yet traversed then incident with the node v_2 and none of them is a cut edge.

So, I am free to traverse any of them, so suppose I decide to traverse the edge between v_2 and v_1. And hence I remove it from the graph I updated my tour and my graph gets updated to G_2.

Now I am at the node v_1 now you can see that I have 3 edges incident with v_1 the edge between v_1 and v_3 the edge between v_1 and v_4 and the edge between v_1 and v_0. And you can see that the edges between v_1 and v_3 is a non cut edge.

And also the edge between v_1 and v_4 is also a non cut edge but the edge between v_1 and v_0 it is a cut edge because, indeed if you remove the edge between v_1 and v_0 the vertex v_0 gets disconnected from the rest of the graph in G_2. So that is why as per the Fleury's algorithm when you have the choice here between selecting the cut edges and non cut edges you should give preference to the non cut edges that means you should either traverse edge v_1, v_3 or you should traverse the edge v_1, v_4.

And you can check why that is the case because if you do not follow the Fleury's algorithm and you decide to make or include this edge namely the edge from between v_1 and v_0 and you advance your tour and you reach v_0 then you are stuck you still have lots of edges to cover. And you have now reached a point where from that point if you want to come back to the graph you have to repeat the same edge between v_0 and v_1.

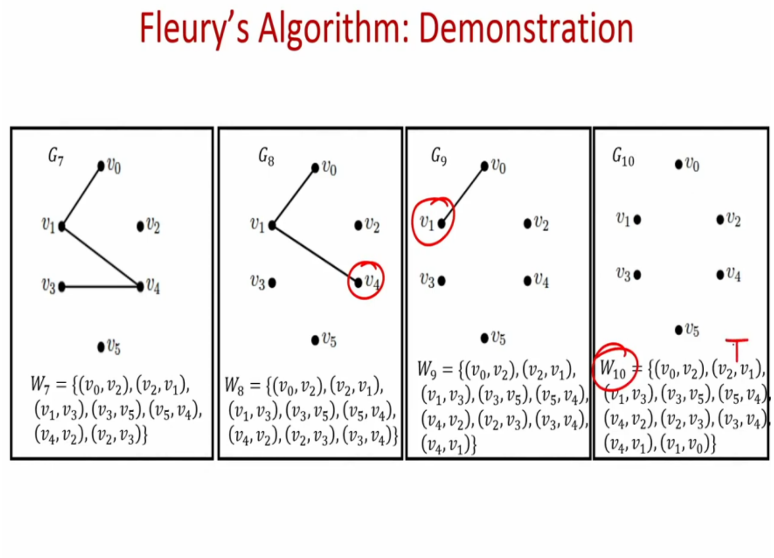
And hence you will not obtain a simple circuit. So that is why Fleury's algorithm says when you have an option between cut edge and non cut edge you should give preference to non cut edge. So, we will give preference to either the edge between v_1 and v_3 or the edge between v_1 and v_4. So, I followed the edge between v_1 and v_3 and now I am at v_3. I have 3 edges incident with v_3 none of them is a cut edge so I can select any of them.

So, I select the edge between 3 and 5 now I am at v_5. And now you can see that there is only one edge incident with the node v_5 which is not yet traversed namely the edge between v_5 and v_4. And indeed that is a cut edge in the graph G_4. But I do not have any choice I have to traverse that edge because there is no other edge left incident with v_5 other than the edge between 5 and 4. So, I have to traverse that edge but that would not cause any issue.

Because if I traverse the edge between v_5 and v_4 that means I have now taken care of all the edges incident with v_5 in my original graph. And I do not need to come back to the node v_5 in my future iterations of the tour. So, now I am at the node v_4, multiple edges are incident with the node v_4 none of them is a cut edge so we can choose any of them I choose to traverse the edge between 4 and 2.

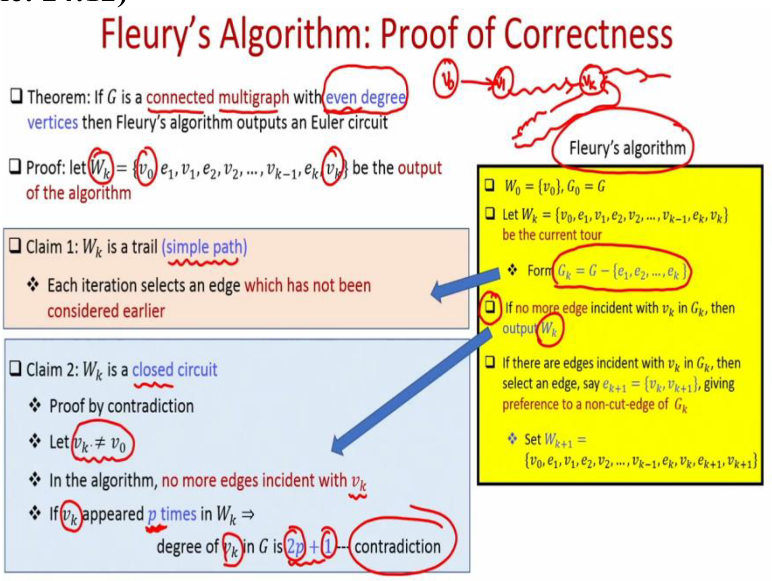
Now at vertex v_2 there is only 1 edge incident namely between v_2 and v_3 which is indeed a cut edge for the graph G_6 but I have no other option. So, I have to traverse that edge but that would not cause any issue. Now I am at v_3 there is only 1 edge incident with v_3 which is a cut edge but again not an issue and I have to follow that edge because there is no other option.

(Refer Slide Time: 23:36)



So, I go to the node v_4. Again there is only 1 edge left incident with v_4 which is a cut edge. So, I have no choice I have to traverse that edge I am now at v_1 and there is only 1 edge left in the graph, traverse that edge and now you have your tour ending. So, W_{10} will be the tour T which was output tour of your Fleury's algorithm and it is easy to see thatwe have indeed obtained an Euler circuit. So that is a demonstration of Fleury's algorithm.

(Refer Slide Time: 24:12)



So, now we want to prove that indeed the output of Fleury's algorithm is an Euler circuit and the proof is slightly involved. But I will try to give you a high level overview of the proof of correctness. So, what we want to prove here is that if your graph G is a connected graph and multi graph; remember for Euler circuit I do not need my graph necessarily to be a simple graph. As long as all the vertices of the graph has even degree that is fine.

I will end up getting a circuit which covers each and every edge of the graph even if there are multiple edges between the same pair of nodes. So, the theorem statement that I want to prove here is that if your graph is connected and every vertex has even degree then by running this Fleury's algorithm a very simple algorithm you can see the algorithm is very simple. So, the claim is that by running this simple algorithm the output tour that we obtain is indeed an Euler circuit.

So, there are multiple things which we have to prove regarding the output that we obtain as part of the Fleury's algorithm. So, let the output be W_k that means I have run for k iterations I have to first show that indeed the output is a simple path that means no edge is repeated. That means all the edges e_1, e_2, e_k which I obtain in the tour W_k are distinct edges but that is very simple to prove because what we are doing in the Fleury's algorithm if you check this step.

Once I have decided the next edge to traverse I am not going to consider it in the future iterations I am simply removing it from my original graph G and I am updating my graph. So that ensures that in each iteration I am selecting distinct edges and hence my output will be a simple path. The next thing I have to prove is that not only the output is a simple path it is a closed circuit that means the starting point and the end point they are the same.

And there are multiple ways to prove this: a very simple proof will be proof by contradiction. So, we want to prove that v_0 and v_k are same that is what we want to prove but on contrary assume that v_k and v_0 are different. So, assuming this contrary statement I have to arrive at some false conclusion or false statement. So let us see what is the false conclusion we can arrive at. So, since I have terminated my tour with the node v_k, that means this particular step which determines the termination condition of your algorithm guarantees or implies that there are no more edges incident with the vertex v_k in your graph G_k . There are no more edges left that means what I can say is the following if the node v_k which is my endpoint of the tour has appeared p number of times in there tour. So, remember the vertices are allowed to be repeated in your Euler circuit.

It is the edges only which are not allowed to be repeated. So, it is not necessary that v_k has appeared exactly once it can appear multiple times in fact it can appear multiple times. So, imagine it has appeared p number of times that means you started with your tour with v_0 you went to v_1 and you continued your tour and you stuck v_k somewhere and then again you came out of v_k.

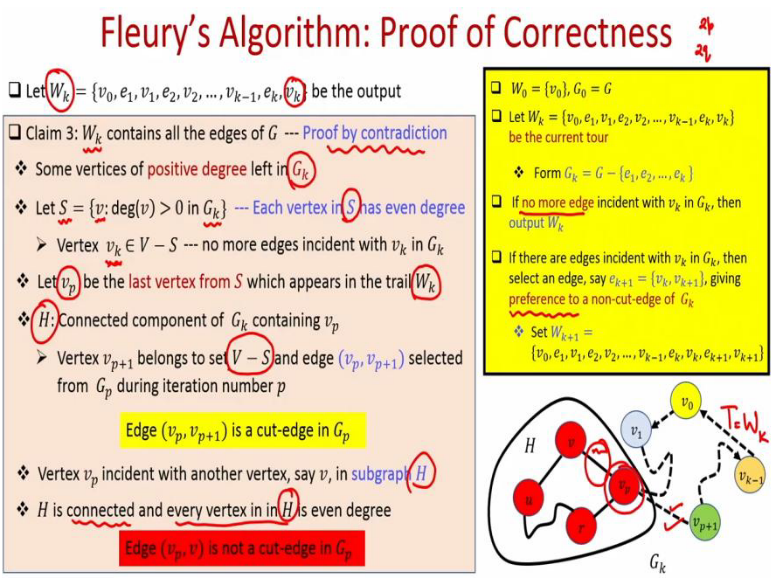
And then again suppose you entered v_k and then again you came out of v_k and so on. So like that assume that v_k has appeared p number of times in your tour. That means the degree of the vertex v_k in your original graph is 2 times p + 1 why 2 times p + 1? Because out of those p times where the vertex v_k is appearing, the last occurrence is actually the occurrence where you are actually terminating the tour.

So, you remember you are terminating your tour with the vertex v_k. So that means out of those p times definitely 1 time is the last occurrence. And the remaining p - 1 times you have entered you have come out you have entered you have come out you have entered you have come out. So, I am assuming here that the vertex v_k is occurring as an intermediate node p times apart from the final occurrence.

So, where a p number of occurrences of the vertex v_k as an intermediate node and there is a final occurrence of the vertex v_k as the endpoint of your tour. So that means that the overall degree of the vertex v_k is 2 times p + 1, 1 because of the final appearance of the vertex v_k in the tour and 2 times p because it is occurring as an intermediate node and each time it is occurring as an intermediate node we are counting 2 to the degree of the vertex v_k.

So that means the overall degree of the vertex v_k is 2 times p + 1 which is an odd quantity and this is a contradiction, contradiction to the fact that in my graph it is guaranteed that all the vertices are of even degree. So that means whatever I have assumed here is contrary; that means my starting point and end point are the same.

(Refer Slide Time: 29:57)



So that means whatever output I obtain it is indeed a simple circuit now left is the tricky part to prove. So, we have to prove that indeed the tour W_k, the simple circuit W_k, which we obtain here it has all the edges of the graph G, no edge is missing in this tour and the proof will be by contradiction. Again there could have been multiple ways but we will follow a proof by contradiction approach.

So, imagine that some vertices of positive degree are still left in my graph G_k. So, remember I have stopped my algorithm at the kth iteration because I am assuming that my tour consists of k edges. So, the instance of the graph left at that point is the graph G_k and since I am assuming that there are still some untraversed edges left in my graph that means there are some vertices which do have incident edges left in my graph G_k.

That means in other words there are still some vertices of positive degrees left in my graph G_k. So, I am denoting the set of all the vertices which still have some untraversed edges left in the graph G_k by the set S. So, namely it is the set of all the vertices which have degree greater than 0 in the leftover graph G_k. The first thing to observe here is that since you started with your original graph where all the vertices have even degree, it is guaranteed that even in the set S all the vertices still have even degree. So, if you have not at all traversed any edge incident with the vertex in a set S that means whatever was its original degree that is still left that means its original degree was even and that even quantity is retained as a degree in my graph G_k as well. Whereas if you have traversed some of the partial edges incident with node v in the set S that means if its original degree was some 2 times p, and if you have traversed some of the edges that may say the vertex v has occurred q number of times that means you have taken care of 2 q degree that means the leftover degree will be 2 p - 2 q which is still an even quantity. So, that ensures that each vertex in my graph S still have even degree; whereas for every vertex v_k, the vertex v_k with which I have stopped my tour it is not a member of S. Because since I have ended my tour with the vertex v_k and ending condition or the terminating condition was that there are no more edges left incident with the vertex v_k.

So, I can say that the vertex v_k is indeed a member of the set V - S. So, now the proof by contradiction here basically would like to derive the following : we would like to derive the fact that if at all we have not obtained Euler circuit by running Fleury's algorithm, that means at some point during some iteration in the algorithm we have not followed the Fluery's rule namely there must have been some intermediate iteration where we would have traversed cut edge rather than traversing a non cut edge - that is a contradiction we have to arrive at.

So, how do we arrive at that contradiction? So, let v_p be the last vertex from the set S which appears in your output tour W_k. So, pictorially imagine that this path that is indicated by this dotted arrow that is a tour T obtained by your Fluery's algorithm. And since we have terminated in the kth iteration this is my graph G_k. And what I am saying here is there must be some vertex v_p which is there in your set S. That means there are still some untraversed edges incident with the vertex v_p and vertex v_p is the last vertex; last vertex in the sense that there might be multiple vertices from the set S which could have occurred along your tour W_k. Among all those vertices from the set S which has occurred in your tour W_k I am focusing on the last vertex which has appeared I am then calling that vertex as v_p.

So, first of all you might be wondering that what is the guarantee that such a vertex v_p is there? Well if the vertex v_p is not there that means the unexplored part of the graph which is not yet covered is completely separate or not at all have any overlap with your tour W_k; that means your original graph is a disconnected graph. But I am assuming that my original graph is a connected graph that means there must be some overlap between the uncovered portion and the output tour which you have obtained as part of your Fluery's algorithm.

And I am focusing on the last overlap here; overlap in terms of the vertex. So, the last overlap I am calling it as the vertex v_p. And since there are still some edges incident with my vertex v_p in my graph G_k when I have terminated my algorithm, I denote by H the connected component in my graph G_k containing the vertex v_p; that means whatever is the unexplored portion left in my graph and G_k incident with the vertex v_p.

So, the vertex of v_{p+1} was selected during the p + 1th iteration and definitely the vertex v_{p+1} belongs to the set V - S that means in my final output when I am considering the graph G_k there would not be any more edges left which are not yet traversed and still left and incident with v_{p+1} because v_{p+1} is not a member of the set S it belongs to the set V - S because the last appearance of a node from the set S is v_p not v_{p+1} that means all the edges which are incident with v_{p+1} would have been traversed as part of the tour W_k.

This implies that the edge v_p and v_{p+1} which you have selected during the pth iteration is a cut edge in the graph instance G_p. Because if this edge v_p and v_{p+1} is not a cut edge that means there is still a way to go to edge and then come back to the vertex v_{p+1} then that violates the assumption that v_p is the last occurrence of a node from the set S which appears along your tour W_k.

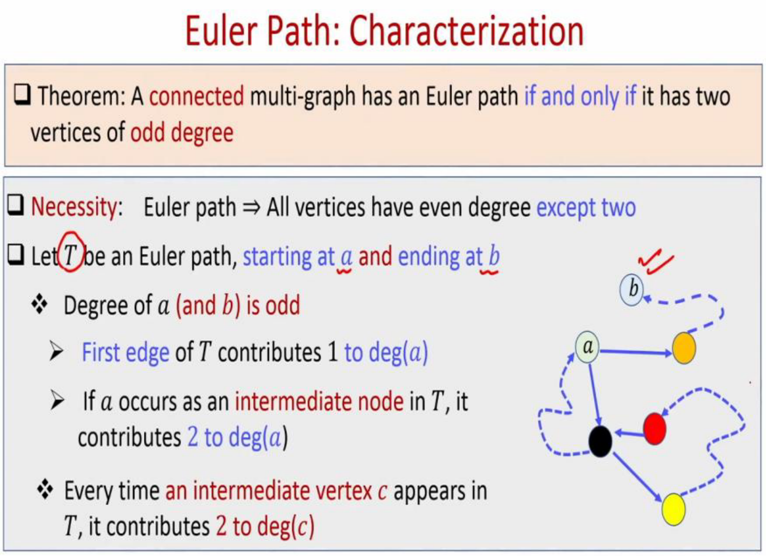
So that is why there is no way to go back to this unexplored portion H and come back which implies that this H between v_p and v_{p+1} is actually a cut edge in your graph G_p. Now since my vertex v_p is a member of the set S and it has some untraversed edges left over in the graph G_k that means it has at least one edge incident and that edge is a part of my sub graph H. So, I am calling that vertex as vertex V so there might be still multiple edges incident with the vertex v_p left I am calling one of the edges as the edge v, v_p.

And notice that H is connected because that is our definition of a connected component. And we have already argued that whatever vertices that are there which have untraversed edges left they still have even degree; that means every vertex in this connected component has an even degree. So that means now you have an untraversed portion in the graph which is connected and where every vertex has an even degree.

So, it is a very simple fact to prove which I am not proving here that if you have a connected graph or a connected sub graph where you have every vertex of even degree. Then it would not have any cut edge that means none of the cut edges in the graph H will be a cut edge. So that shows that during the pth iteration you have an option of selecting a non cut edge namely the edge between the vertex v and vertex v_p this edge.

That edge was still there to traverse which was a non cut edge during the pth iteration but you did not follow the Fleury's instruction but rather followed the edge between the vertex v_p and v_{p+1}. And due to which you have leftover portion of the graph namely the portion H which has not yet obtained as an output in your overall algorithm. So that means you have not followed the Fleury's algorithm that means you have given preference to a cut edge rather than a non cut edge which is the contradiction. So that means if we follow the Fleury's algorithm systematically there is no possibility of leaving out any edge in the graph.

(Refer Slide Time: 40:01)

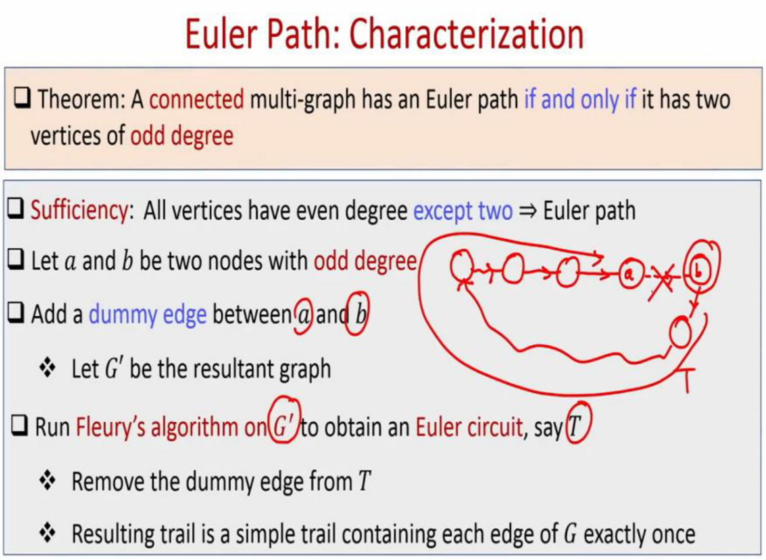


So that was the simple characterization of Euler circuit. Now let us quickly prove a characterization of Euler path. So, the characterization of Euler path is that in your graph there should be exactly 2 vertices of odd degree and remaining all vertices should have even degree. So, the necessity condition again can be proved along similar lines as we did for the characterization of Euler circuit. So, we want to prove that if at all you have an Euler path then there are exactly 2 vertices of odd degree.

So, imagine your Euler path that is there in a graph is T which starts at the vertex a and ends at the vertex b. So, it is easy to argue here that degree of and degree b will be both odd because the first edge of the tour will be incident with the node a which will contribute 1 to the degree of a and if a occurs as an intermediate node p number of times then that contributes 2 times p to the overall degree of a.

So, hence the overall degree a will be 2 times p + 1 and same we can argue for the node b as well. Whereas if you take any other intermediate node c different from a and b which is occurring say k number of times in the tour then the overall degree of node c will be 2 times k; so that proves the necessity the condition.

(Refer Slide Time: 41:26)



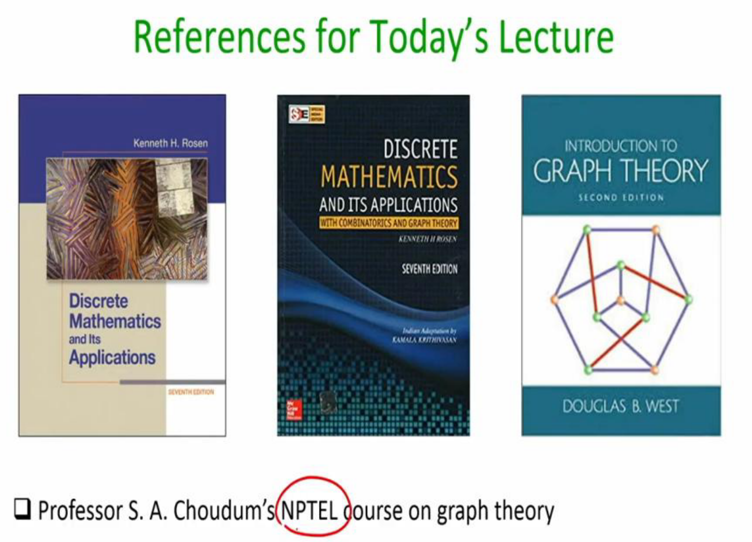
How do be prove the sufficiency condition? So, imagine that you have a connected multi-graph where you have exactly 2 vertices of odd degree remaining all vertices of even degree then I have to show that I can find out an Euler path; so imagine that 2 vertices which are having odd degrees are a and b. So, what we do is we add dummy edge in my graph between those special nodes a and b which have odd degrees.

And let G' be the resultant graph. Now it is easy to see that the graph G' all the nodes including the node a as well as node b have even degree and that satisfies the characterization of Euler circuit. So, I can run the Fleury's algorithm on the modified graph G' to obtain an Euler circuit; I call it T. And now in that Euler circuit just remove the dummy edge imagine or pretend thus edge is dummy edges are not there.

The resulting trail will be a simple trail which will have all edges of the original graph it would not have the dummy edge that you have added. So, for instance suppose tour T is like this you traversed and suppose you reached the vertex a and suppose as part of the tour that you have obtained as an output of Fleury's algorithm; this is a tour you started at a same vertex you ended at same vertex and this tour has all the edges of the dummy graph.

So, you can imagine or you can extract out an Euler path of this term is tour as the following you can imagine as your tour starts with b and then follow this tour and end at a and just ignore this dummy edge that is all so that will be continue that has an Euler path.

(Refer Slide Time: 43:29)



So that brings me to the end of this lecture. These are the reference used for today's lecture, I also follow some of the notes from Prof. Choudum's NPTEL lecture on graph theory specially for the proof of correctness of Fleury's algorithm. So, just to summarize in this lecture we saw the definition of Euler circuit, Euler path and we proved the necessary and sufficient condition for the existence of Euler circuit and Euler path in the graph, thank you.