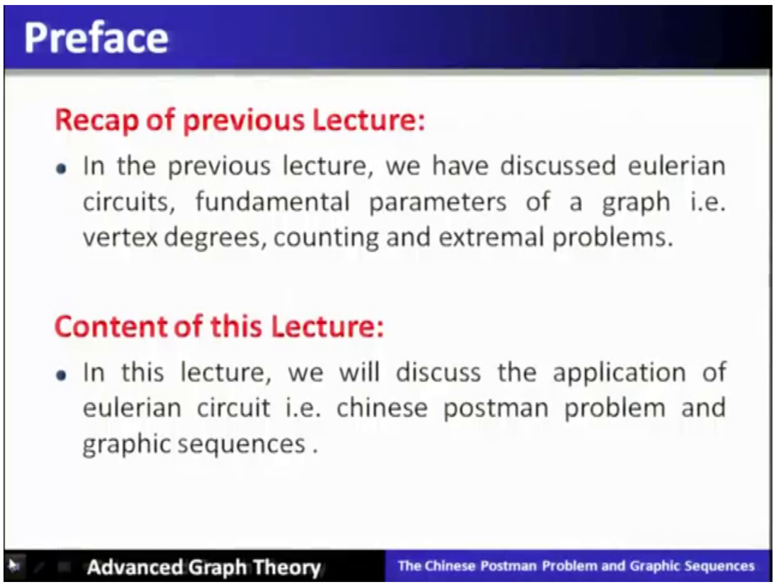
Advanced Graph Theory

Dr. Rajiv Misra
Department of Computer Science and Engineering
Indian Institute of Technology, Patna

Lecture – 04
The Chinese Postman Problem and Graphic Sequences

(Refer Slide Time: 00:21)



Preface

Recap of previous Lecture:

- In the previous lecture, we have discussed eulerian circuits, fundamental parameters of a graph i.e. vertex degrees, counting and extremal problems.

Content of this Lecture:

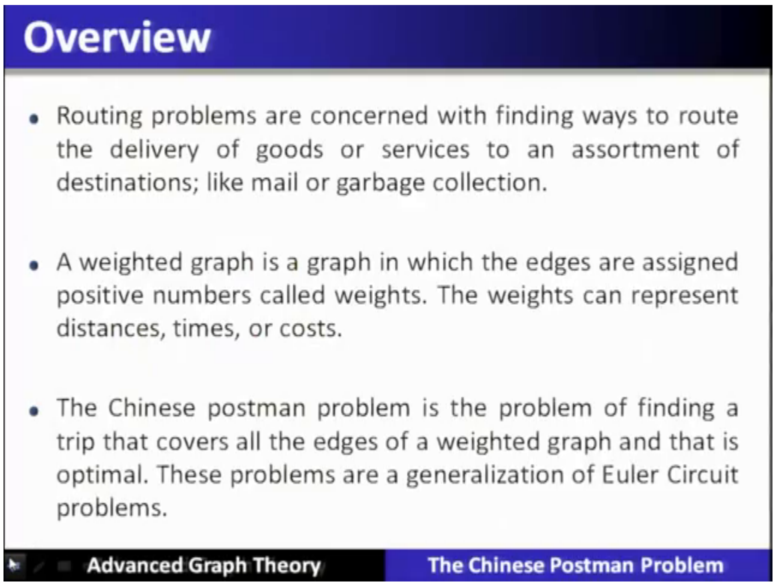
- In this lecture, we will discuss the application of eulerian circuit i.e. chinese postman problem and graphic sequences .

Advanced Graph Theory
The Chinese Postman Problem and Graphic Sequences

Lecture 4, the Chinese postman problem and graphic sequences recap of previous lecture.

In previous lecture we have discussed Eulerian circuits fundamental parameters of the graph that is vertex degrees counting extremal problems and so on. Content of this lecture; in this lecture, we will discuss one application of Eulerian circuit that is the Chinese postman problem and then we will go and discuss graphic sequences of a graph.

(Refer Slide Time: 00:46)



Overview

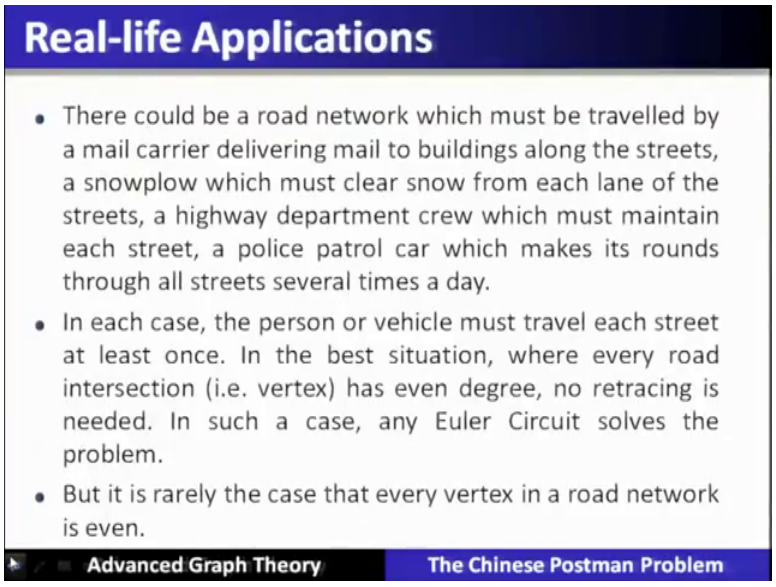
- Routing problems are concerned with finding ways to route the delivery of goods or services to an assortment of destinations; like mail or garbage collection.
- A weighted graph is a graph in which the edges are assigned positive numbers called weights. The weights can represent distances, times, or costs.
- The Chinese postman problem is the problem of finding a trip that covers all the edges of a weighted graph and that is optimal. These problems are a generalization of Euler Circuit problems.

Advanced Graph Theory | The Chinese Postman Problem

The Chinese postman problem deals with the routing problem. So, the routing problem are concerned with finding the ways to rout the delivery of the good or the services to an assortment, of the destinations like mail or garbage collection.

In this particular problem setting we require a graph which is called a weighted graph; the edges are assigned the positive numbers called weights. These weights can represent the entities like distance time and cost. The Chinese postman problem is the problem of finding are tour that covers all the edges of the weighted graph, and that is the optimal that is of the minimal cost. These problems are the generalization of an Euler circuit problem that we have seen in the last lecture, the real life applications.

(Refer Slide Time: 01:43)



Real-life Applications

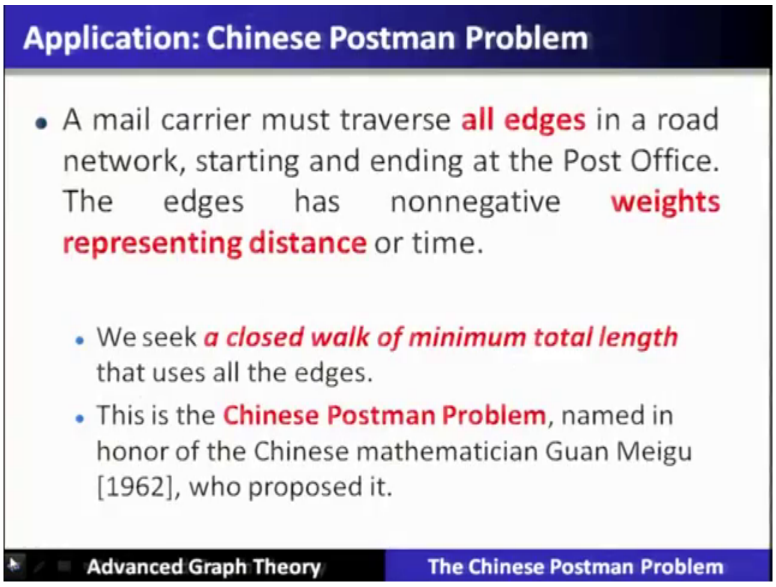
- There could be a road network which must be travelled by a mail carrier delivering mail to buildings along the streets, a snowplow which must clear snow from each lane of the streets, a highway department crew which must maintain each street, a police patrol car which makes its rounds through all streets several times a day.
- In each case, the person or vehicle must travel each street at least once. In the best situation, where every road intersection (i.e. vertex) has even degree, no retracing is needed. In such a case, any Euler Circuit solves the problem.
- But it is rarely the case that every vertex in a road network is even.

Advanced Graph Theory | The Chinese Postman Problem

There could be a road network which must be travelled by a mail carrier delivery mails to the building along the street, at snowplow which must clear the snow from each lane of the streets. A highway department crew must maintain each street; the police patrol car which makes it is round through all thy streets several times a day. So, the optimization of covering all these streets with a minimum cost is an important task for all such (Refer Time: 02:22) life applications. So, in each case a person or a vehicle must travel each street at least once. So, in the best situation where every road intersection that is the vertex has the even degree node increasing if needed, in such case any Euler circuit solves the problem very effectively.

But it is rarely the case that every vertex in the road network is even. So, how are we going to solve the Chinese postman problem in that problem setting?

(Refer Slide Time: 02:53)



Application: Chinese Postman Problem

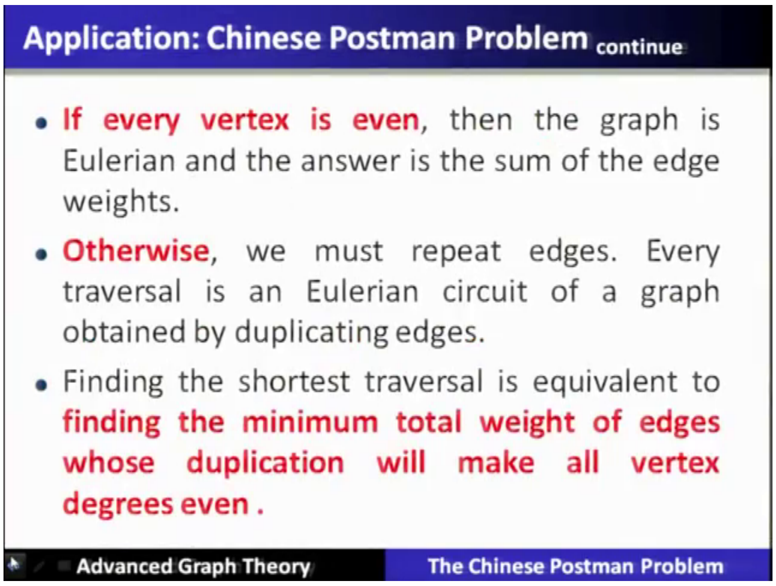
- A mail carrier must traverse **all edges** in a road network, starting and ending at the Post Office. The edges has nonnegative **weights representing distance** or time.
  - We seek ***a closed walk of minimum total length*** that uses all the edges.
  - This is the **Chinese Postman Problem**, named in honor of the Chinese mathematician Guan Meigu [1962], who proposed it.

Advanced Graph Theory | The Chinese Postman Problem

So, a mail carrier must traverse all the edges in the road network is starting and ending at a post office. The edges has non negative weights representing the distance or a time. So, we seek a closed walk of minimum total length that uses all the edges, hence this is an generalization of the Euler's circuit. So, Euler circuit is a closed walk, which does not have the repeated edges and that is why it is a closed trail containing all the edges called Euler circuit. We require that circuit which is having the minimum total cost, and this is the Chinese postman problem.

So, this is the Chinese postman problem named in the honor of Chinese mathematician Guan Meigu who proposed it in 1962.

(Refer Slide Time: 03:56)



Application: Chinese Postman Problem continue

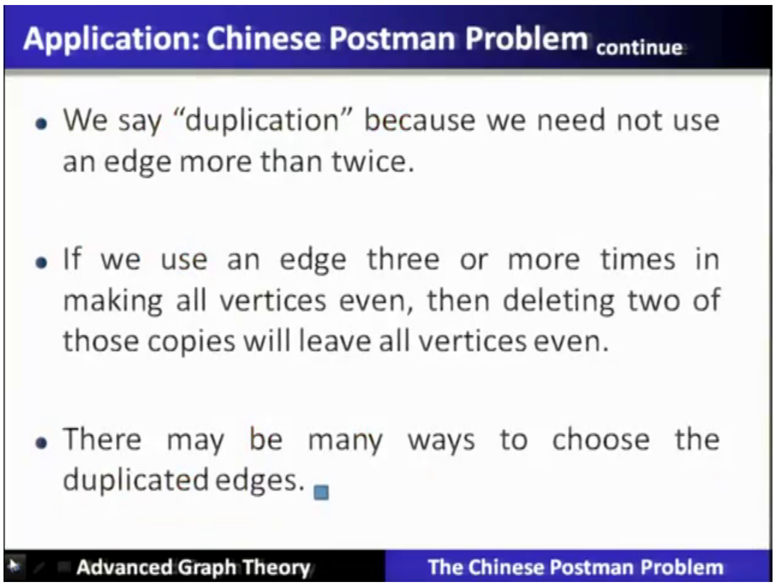
- If every vertex is even, then the graph is Eulerian and the answer is the sum of the edge weights.
- Otherwise, we must repeat edges. Every traversal is an Eulerian circuit of a graph obtained by duplicating edges.
- Finding the shortest traversal is equivalent to finding the minimum total weight of edges whose duplication will make all vertex degrees even.

Advanced Graph Theory The Chinese Postman Problem

Application of a Chinese postman problem, we are going to cover here in this part of this particular lecture.

Now, if every vertex is even then the graph is Eulerian and the answer is to the post Chinese postman problem is the sum of edge weights in the graph. Now if it is not then we must repeat the edges. So, every traversal is Eulerian circuit of a graph obtained by the duplicating the edges. So, finding the shortest traversal is equivalent to finding the minimum total weight of the edges, whose duplication will make all the degrees even.

(Refer Slide Time: 04:45)



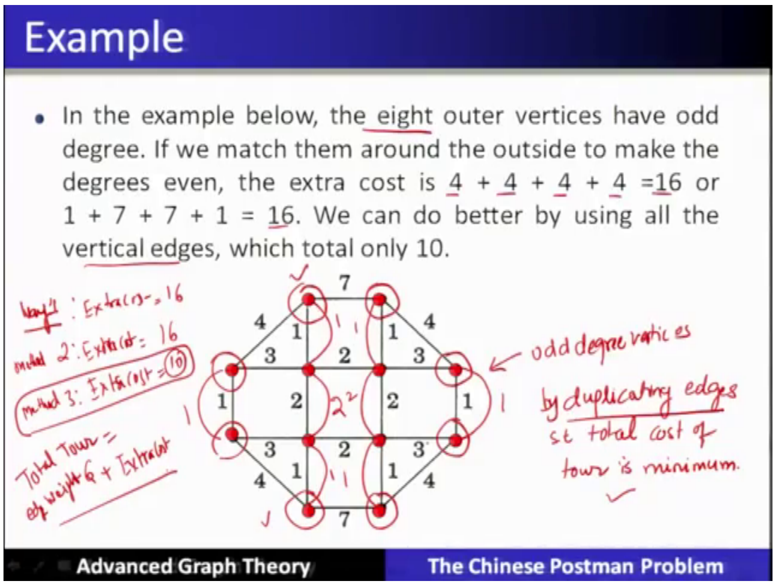
Application: Chinese Postman Problem continue

- We say "duplication" because we need not use an edge more than twice.
- If we use an edge three or more times in making all vertices even, then deleting two of those copies will leave all vertices even.
- There may be many ways to choose the duplicated edges.

Advanced Graph Theory The Chinese Postman Problem

Now, when we say duplication, then if we use a edge 3 or more times in making all the what is it even the deleting two of these copies we leave the all vertices let even. So, there are many ways to choose the duplicate edges, and that becomes an optimization problem.

(Refer Slide Time: 05:08)



Example

- In the example below, the eight outer vertices have odd degree. If we match them around the outside to make the degrees even, the extra cost is 4 + 4 + 4 + 4 =16 or 1 + 7 + 7 + 1 = 16. We can do better by using all the vertical edges, which total only 10.

Way 1 : Extra cost = 16
method 2: Extra cost = 16
method 3: Extra cost = 10
Total Tour = edge weight + Extra cost
odd degree vertices
by duplicating edges st total cost of tour is minimum.
Advanced Graph Theory
The Chinese Postman Problem

So, Chinese postman becomes an optimization problem. Let us see through an example this Chinese postman problem and the optimization issues involved in this particular Chinese postman problem.

So, in the example below the 8 outer vertices have odd degrees. So, they are basically this is an odd degree vertex, this is an odd degree vertex, this is an odd degree vertex, this is also an odd degree odd degree vertex. So, there are total eight such odd degree vertices present in this particular weighted graph. Now you know that the Euler circuit this is not an Eulerian graph why because all the vertices has to be of even degree here which is not.

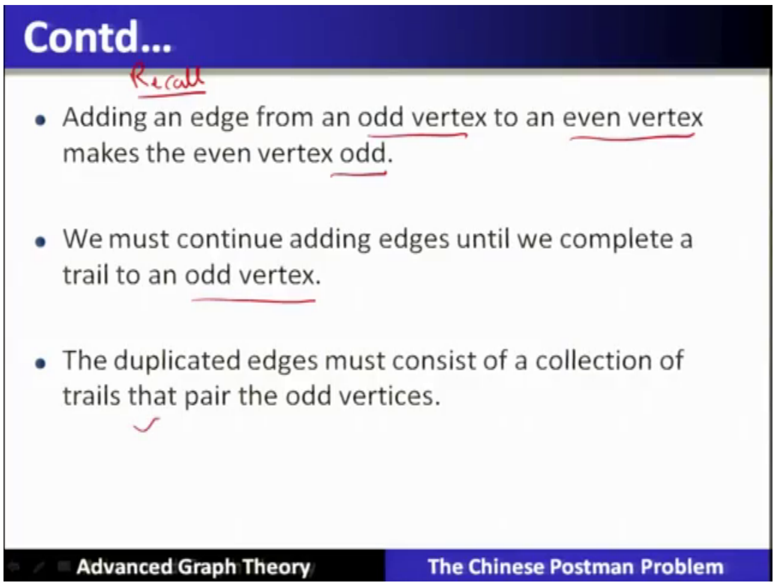
So, if we match them around outside around the outside to make the degrees even. So, we require we have to pay an extra cost. So, how to make this particular graph; which is having an odd degree vertices has an Eulerian graph by duplicating such that the total cost of the tour is minimum. So, here if you see the duplicate how to make the edges duplicate. So, that the graph this graph can converted into a Eulerian graph. So, the first way; is states that we have to match the vertices around outside to make the degrees even. So, this particular these two vertices for example, they are odd degree, if we match this particular edge of length 4, then you see that these degrees will become even then in that case similarly if you match this, if you match this, if you match this.

So, in this particular graph we have added 4 different edges, and all these edges are of cost 4. So, the total weight extra cost comes out to be 16 here in this particular method. Another method; if you take these two vertices and at an edge of cost one, then they will become even similarly here also we can add an extra edge now here also we have to add to make these two even and these two even of. So, these duplicacy will make the second method incurring an extra cost as 1 plus 7 plus 1 that comes out to be 16. Now there is another method which we can do better of these two methods, and which says that all the vertical edges if we include then the cost will be reduced.

So, vertical edges will include for example, this particular vertical edge, if we add this will add a cost one. So, this particular vertex will become even, but this particular vertex will become odd in this case because it was earlier given, adding one edge will become odd. So, we keep on adding an another edge to make it an even now this will become odd. So, again we have to add one more edge. So, this particular vertex will become even. So, this particular vertex also will become even.

Similarly, to make this particular vertex as even, we have to add these extra edges of length one. What about these two vertices? We have to add an edge over here we have to add an edge of cos 1. So, if you sum them 1, 2 then 3 4 4 4 8 9 10 that method 3 will give an extra cost will give an extra cost as 10. So, this becomes a this method 3 basically will require a minimum cost to make this particular graph as an Eulerian graph. So, the earlier total weights total cost of a tour will become the weights edge weights of the graph g plus this extra cost and hence having converted this particular Chinese postman problem can be solved this optimization problem we can solve here in this particular problem setting.

(Refer Slide Time: 11:23)



Contd...

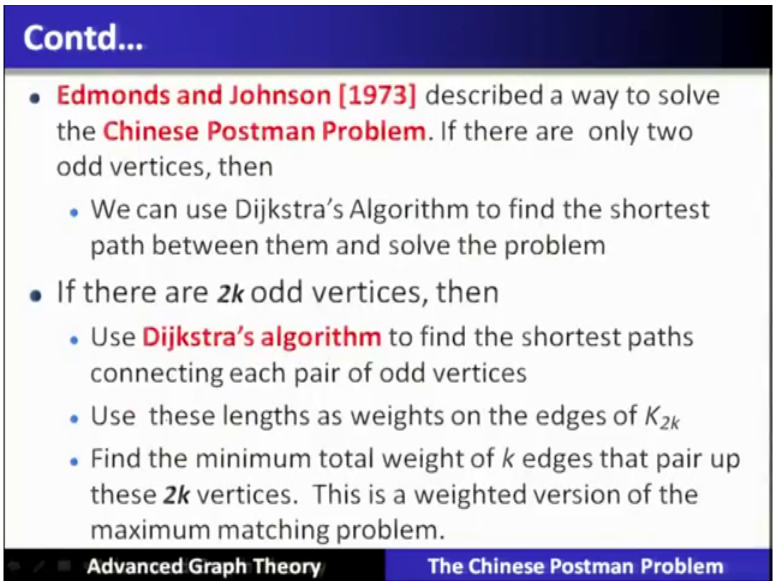
Recall

- Adding an edge from an odd vertex to an even vertex makes the even vertex odd.
- We must continue adding edges until we complete a trail to an odd vertex.
- The duplicated edges must consist of a collection of trails that pair the odd vertices.

Advanced Graph Theory The Chinese Postman Problem

So, let us recall, what we have so, far done in the previous example. So, adding an edge from an odd vertex to an even vertex makes the even vertex odd. So, we have to keep on doing it. So, we must continue adding the edges until we complete the trail to an odd vertex, and that particular process when it completes all the vertices will become even in that case. The duplicate edges must consist of the collection of the trail that pairs the odd vertices.

(Refer Slide Time: 12:01)



Contd...

- **Edmonds and Johnson [1973]** described a way to solve the **Chinese Postman Problem**. If there are only two odd vertices, then
  - We can use Dijkstra's Algorithm to find the shortest path between them and solve the problem
- If there are ***2k*** odd vertices, then
  - Use **Dijkstra's algorithm** to find the shortest paths connecting each pair of odd vertices
  - Use these lengths as weights on the edges of K_{2k}
  - Find the minimum total weight of k edges that pair up these ***2k*** vertices. This is a weighted version of the maximum matching problem.

Advanced Graph Theory The Chinese Postman Problem

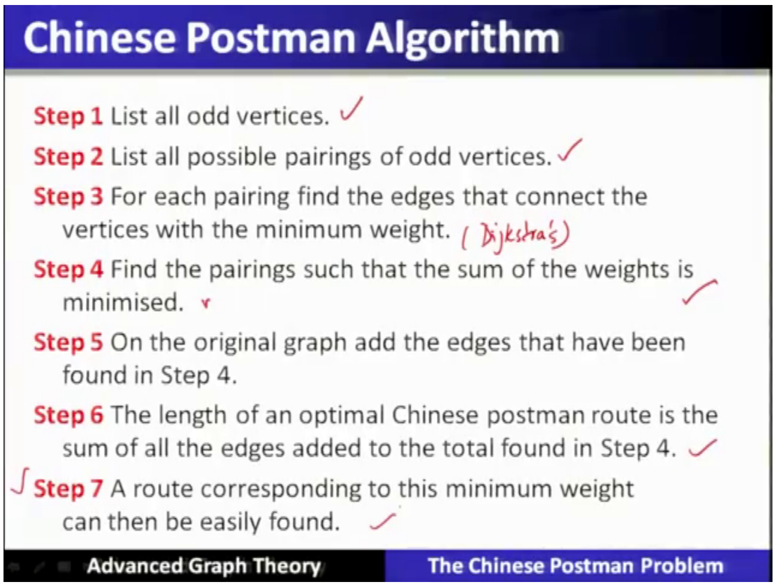
Edmunds and Johnson in 1973 described way to solve the Chinese postman problem. If there are two odd vertices then we can apply Dijkstra's algorithm to find the shortest path between them and solve the problem.

Now, if there are two k odd vertices then use of Dijkstra's law algorithm to find shortest paths connecting each pair of odd vertices use these lengths as weights on the edges of these 2 k odd vertices and find the minimum cost weight of k edges that pair of these two k vertices this is the weighted version of the maximum matching problem.

So, this way he uses the instead of direct edge indirect edges are implied why because when we implied why because when we implied a Dijkstras laws routing algorithm shortest path algorithm, it will find out a shortest path between two odd vertices and that particular shortest path will be added up, whether it is a direct edge or it will be a path connecting the two vertices.

So, this way this Chinese postman problem can be solved with the help of if a particular shortest path algorithm is available, then this particular one graph given graph can be converted into an Eulerian graph and these extra duplicacy of the edges which can be solved with the help of Dijkstras algorithm can be utilized in this particular process.

(Refer Slide Time: 13:40)



Chinese Postman Algorithm

Step 1 List all odd vertices. ✓

Step 2 List all possible pairings of odd vertices. ✓

Step 3 For each pairing find the edges that connect the vertices with the minimum weight. (Dijkstra's)

Step 4 Find the pairings such that the sum of the weights is minimised. ✓

Step 5 On the original graph add the edges that have been found in Step 4.

Step 6 The length of an optimal Chinese postman route is the sum of all the edges added to the total found in Step 4. ✓

✓ **Step 7** A route corresponding to this minimum weight can then be easily found. ✓

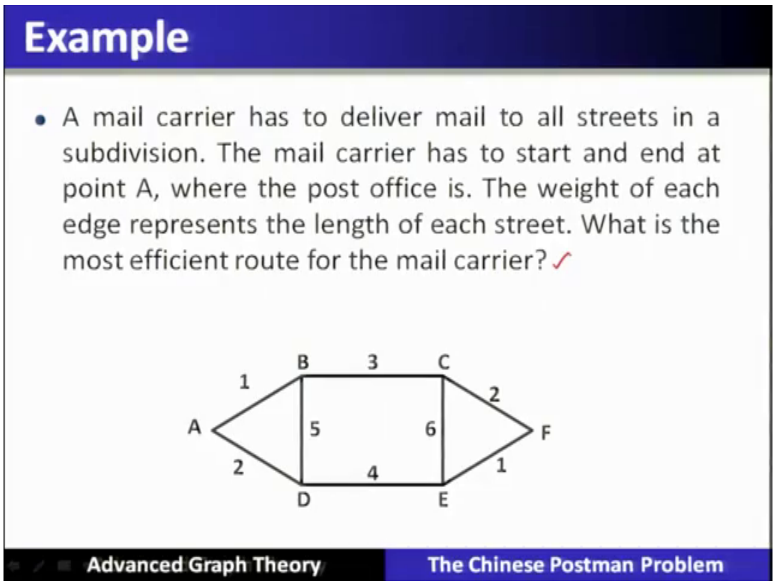
Advanced Graph Theory | The Chinese Postman Problem

So, let us see the Chinese postman algorithm. So, step one is to list all the odd vertices that we have shown you in the previous example, then after finding out all the odd vertices second is that would be to list possible pairing of all vertices.

The third is to would be that each pairing will find the edges that connect, the vertices with a minimum weight either Dijkstra's algorithm, will solve it why because if the graph is of big size manually cannot inspect and find out these minimum weights we have to apply the Dijkstra shortest path algorithm.

The step number 4 would be to find the pairings, such that the sum of the weights is minimized. So, in the original graph we have to add those edges, that have been found in the previous step; that means, with a edges with a minimum weights. Now the length of an optimal Chinese postman route is the sum of all the edges added to the total which is found in step number 4. So, this way up to step number 6 we could find out what is the total cost of an optimal Chinese postman route. Now we have to find out in the step number 7 the actual route means how the postman basically should traverse along the edges. So, that it will incur the minimum cost in that particular route. So, the route corresponding to this minimum cost can be then easily found out by tracing those edges starting from at one point.

(Refer Slide Time: 15:28)



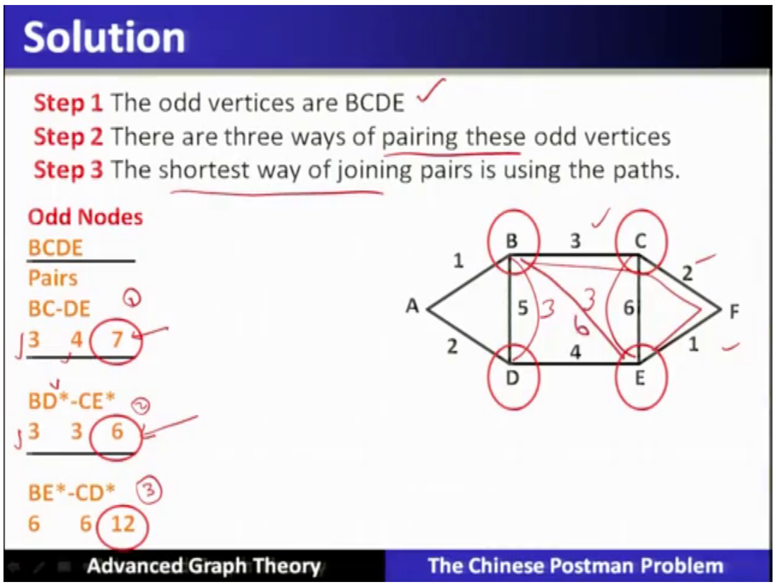
Example

- A mail carrier has to deliver mail to all streets in a subdivision. The mail carrier has to start and end at point A, where the post office is. The weight of each edge represents the length of each street. What is the most efficient route for the mail carrier? ✓

Advanced Graph Theory | The Chinese Postman Problem

Let us see this particular algorithm working through an example. So, here we have a mail carrier has to deliver the mail to all those feeds in the subdivision, the mail carrier has to start and end point at A, where the post office is located the weight of each edge is represented the represents the length of each street, which is most efficient route for the mail carrier in this particular problem setting, and we have to basically show with the help of that algorithm, which we have stated that Chinese postman algorithm. So, in this particular problem setting the first step is to find out the odd vertices.

(Refer Slide Time: 16:07)



Solution

Step 1 The odd vertices are BCDE ✓

Step 2 There are three ways of pairing these odd vertices

Step 3 The shortest way of joining pairs is using the paths.

Odd Nodes

BCDE

Pairs

BC-DE ①

3 4 7

BD*-CE* ②

3 3 6

BE*-CD* ③

6 6 12

Advanced Graph Theory | The Chinese Postman Problem

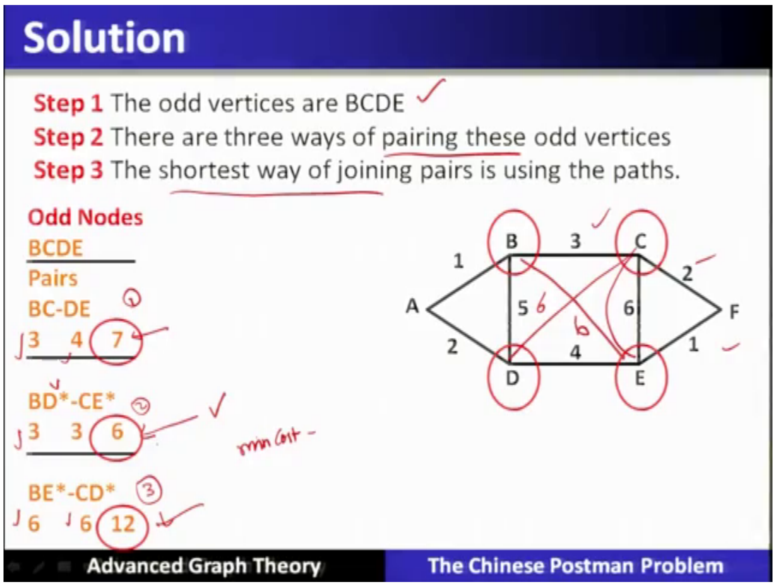
So, BCDE which is encircled here they are the odd vertices. So, BCDE are the odd vertices second step would be to pair them these pairing of these odd vertices.

So, how many possible pairings of BCDE are there, one pair is BC, the other pair is DE this is first pair, second pair is BD and CE this is second pair, third pair would be BE and CD this is the third pair. So, there are 3 different pairs possible now we have to find out the shortest way of joining the pairs using the path. So, BCD and C they can be reached with a direct edge of cos 3, but if we go by some other way it will become more costly so; obviously, B and C they can be connected with a minimum cost of 3; similarly D also can be connected with a minimum cost. So, total cost of a due these edges would be 7 here in this particular case. That means, if we add an edge of comma 3 and if we add an edge of cost 4. So, this particular total cost will be of this particular graph and all the nodes will become even Eulerian we require an extra cost of 7.

Now, another pairing which we have to see is BD; B and D. So, B and D have direct edge of cost 5, but if we come along A and then from A to D, then the cost will be 3. So, we have to basically add an edge of a cost 3 and 3 start means it is not a direct path, but indirectly we have to come in this way. So, basically an edge of length 3 is a rate over here. Similarly CE there is a direct edge of length 6, but if we come along F come through F then basically the cost will be 3. So, let us add an edge of cost three. So, this will make the graph Eulerian and will incur only the cost of 6 units. Similarly we have another pairing that is BE and. So, BE. So, BE do not have a direct edge. So, it has to come through via A. So, when it will come via A it will incur 7, but when it will come through C B C F and then basically it will cost. So, it will become a cost of 6. So, you can see.

So, 3 then 3 6; o, it will. So, if you add an edge BE of a cost 6 and which will basically obtain via F. So, that is why 6 and E star is given. Similarly CD there is no direct connection, but if you come via BAD that incur a cost of 6. So, total cost here 6 plus 6 that becomes 12.

(Refer Slide Time: 19:46)


Solution

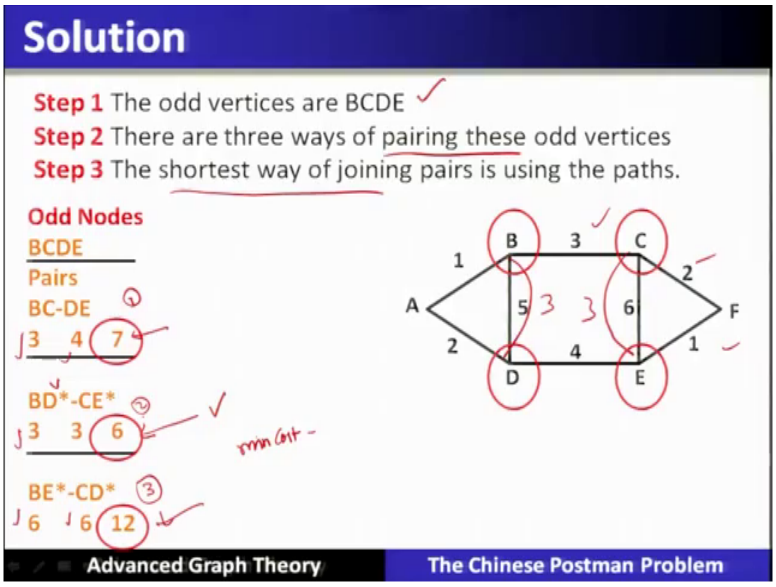
Step 1 The odd vertices are BCDE ✓
Step 2 There are three ways of pairing these odd vertices
Step 3 The shortest way of joining pairs is using the paths.

Odd Nodes
BCDE
Pairs
BC-DE ①
3 4 7
BD*-CE* ②
3 3 6 ✓
min cost -
BE*-CD* ③
6 6 12

Advanced Graph Theory | The Chinese Postman Problem

So, out of these 3 combinations we see that this is having the lowest cost or a minimum cost to convert this graph into an Eulerian graph, hence we will choose this particular that is B D of an extra cost 3 and C D C E of an extra cost D to solve this particular problem.

(Refer Slide Time: 20:15)



Solution

Step 1 The odd vertices are BCDE ✓
Step 2 There are three ways of pairing these odd vertices
Step 3 The shortest way of joining pairs is using the paths.

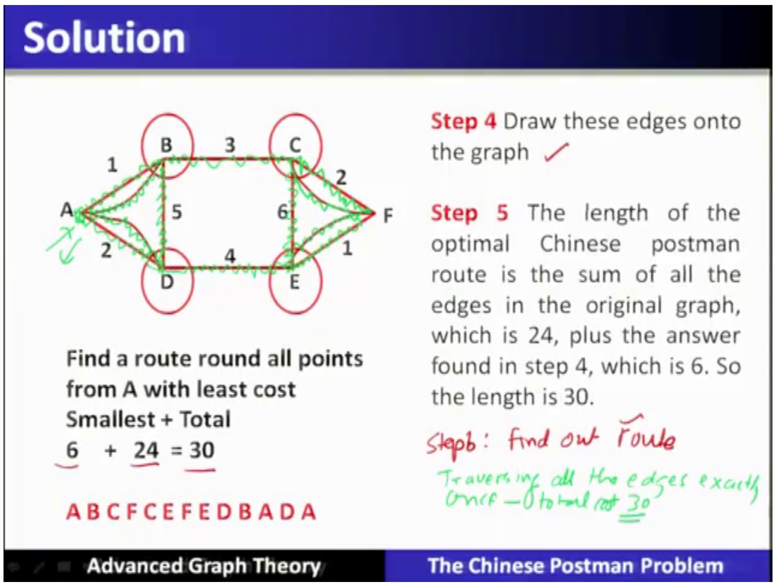
Odd Nodes
BCDE
Pairs
BC-DE ①
3 4 7
BD*-CE* ②
3 3 6 ✓
min cost -
BE*-CD* ③
6 6 12

A
B
C
D
E
F
1
3
2
5
3
3
6
2
4
1

Advanced Graph Theory | The Chinese Postman Problem

Now, the extra cost is 6 here in this particular case. So, step number 4 would be to draw the edges onto this particular graph.

(Refer Slide Time: 20:27)



Solution

Step 4 Draw these edges onto the graph ✓

Step 5 The length of the optimal Chinese postman route is the sum of all the edges in the original graph, which is 24, plus the answer found in step 4, which is 6. So the length is 30.

Step6 : find out route
Traversing all the edges exactly once — total cost 30

Find a route round all points from A with least cost
Smallest + Total
6 + 24 = 30

A B C F C E F E D B A D A

Advanced Graph Theory | The Chinese Postman Problem

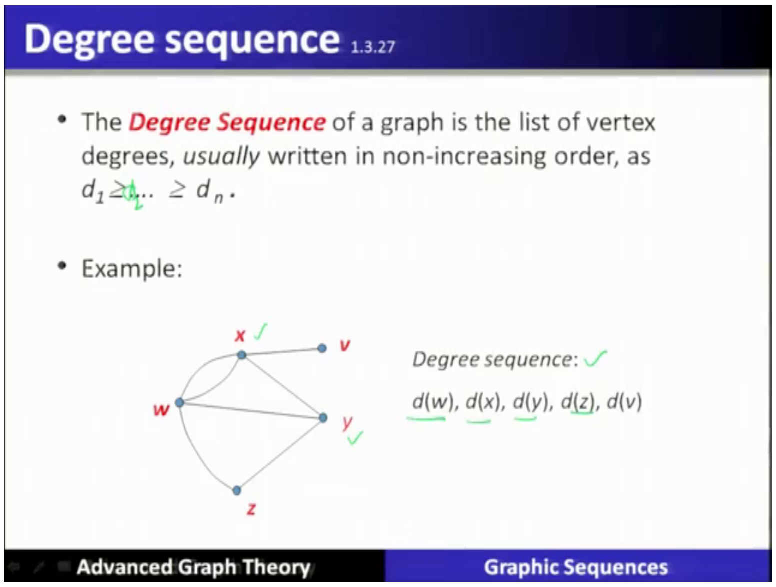
So, we have basically traverse through these edges. So, let us add the extra edges to connect this BE and C D. So, having added these edges we have completed the step number 4 of the algorithm.

Now, step 5 says that the length of the Chinese optimal Chinese postman route is the sum of all the edges in the original graph, which becomes 24 and extra edge cost which basically which we have found out in the step number 4 is 6. So, the total cost is 30. And the next step would be to find out that particular route. So, let us see how to find out that particular route.

Here in this case we start from A then we will go to B, then from B we are going to C, and from C we are going to F, from F we have to go back again to C and from C we have to go to E, and from E we are going to F and from F we have to come back to E again and from E we are going to D, and from D we are going to B and from B we are going to A, and from A we are going to D and from D we are coming back to A again.

So, we started from A and we have reached A traversing all the edges exactly once with the total cost 30. So, this is the solution of a Chinese postman problem. Graphic sequences.

(Refer Slide Time: 22:50)



Degree sequence 1.3.27

- The ***Degree Sequence*** of a graph is the list of vertex degrees, *usually* written in non-increasing order, as $d_1 \geq d_2 \ldots \geq d_n$.
- Example:

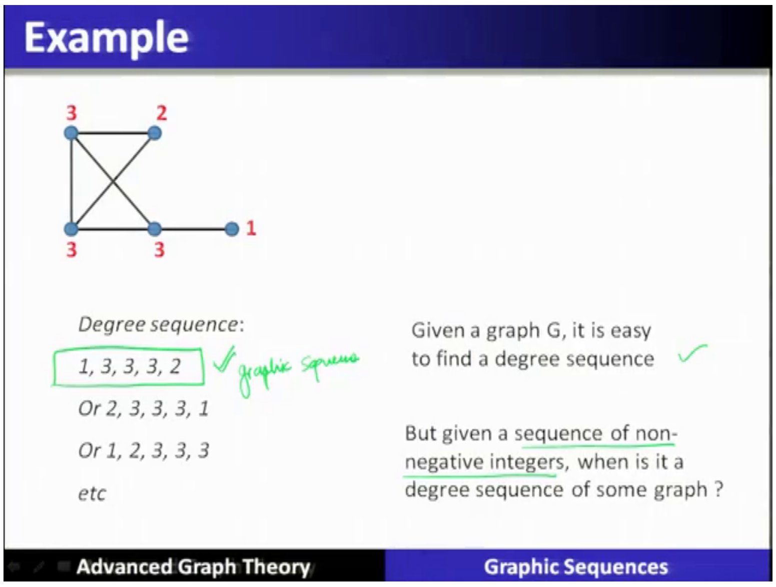
Degree sequence:

$d(w), d(x), d(y), d(z), d(v)$

Advanced Graph Theory | Graphic Sequences

So, a degree sequence, a degree sequence of a graph is the list of vertex degrees usually written in the non increasing order as d 1 is greater than or equal to d 2 greater than or equal to d 3 and form up to d n. So, this particular way we can state here, the degree sequence of a graph. So, degree of vertex the highest degree is w then followed by w it will be x then followed by x it will be y then followed by y it is z and followed by that it is v. So, this is the degree sequence of a graph.

(Refer Slide Time: 23:35)



Example

Degree sequence:

$\boxed{1, 3, 3, 3, 2}$ ✓ graphic sequence

Or 2, 3, 3, 3, 1

Or 1, 2, 3, 3, 3

etc

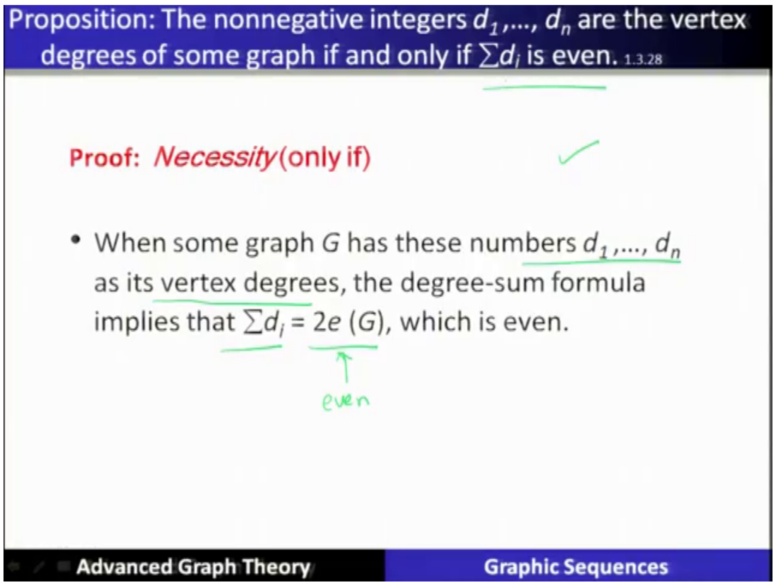
Given a graph G, it is easy to find a degree sequence ✓

But given a sequence of non-negative integers, when is it a degree sequence of some graph ?

Advanced Graph Theory Graphic Sequences

Now, another question is if we are given a sequence like this, now the question is whether this is a degree sequence of a graph or not. So, given a graph g it is easy to find out it is degree sequence, but given a sequence of a non negative integers like this when is it a degree sequence of some graph. If this sequence of non negative integers represents a degree sequence of a graph then it is called a graphic sequence. So, how do we find out this particular case whether this sequence of non negative integers is a degree sequence of a graph or not? That is called the degree sequence.

(Refer Slide Time: 24:31)



Proposition: The nonnegative integers $d_1,\dots, d_n$ are the vertex degrees of some graph if and only if $\sum d_i$ is even. 1.3.28

Proof: *Necessity* (only if) ✓

- When some graph G has these numbers $d_1,\dots, d_n$ as its vertex degrees, the degree-sum formula implies that $\sum d_i = \underset{\substack{\uparrow \\ \text{even}}}{2e(G)}$, which is even.

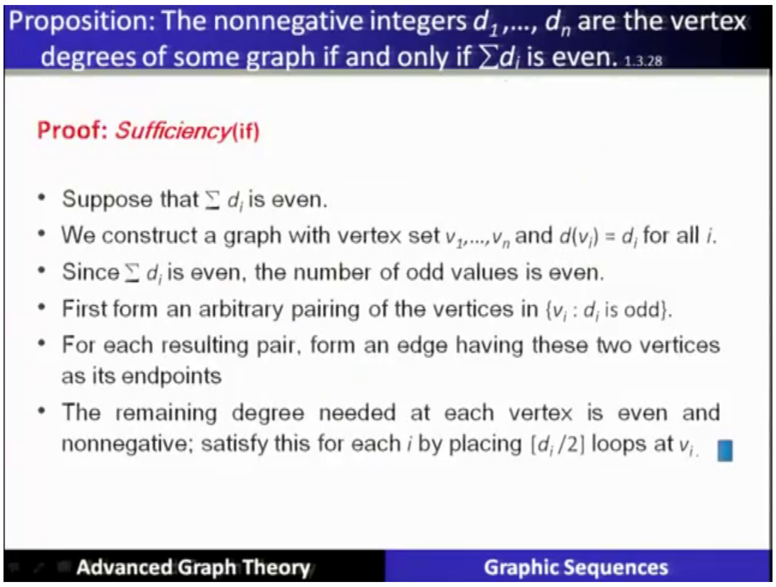
Advanced Graph Theory Graphic Sequences

So, we have to see a preposition. So, basically come out with a particular strategy to basically test whether the given sequence is a graphic sequence or not.

Proposition the non negative integers d 1 d 2 and so on d n or the vertex degrees of some graph if and only if the sum of all degrees is even. Let us prove this first we have to see the necessity condition. So, in the necessity condition let us assume that some graph G has these numbers d 1 d 2 and so on up to d n as it is vertex degrees. So, if we apply the degree sum formula, then this sum of all degrees will be the twice the number of edges in the graph and this particular value the parity of this particular value will become even; why because it is multiplied by even number hence the sum of the degrees is even and one side of this particular theorem or a preposition is proved let us see the other side of a theorem that is the sufficiency condition.

Here in the sufficiency condition we will assume that the sum of degrees is even and so, we have to find out.

(Refer Slide Time: 26:10)



Proposition: The nonnegative integers $d_1, \ldots, d_n$ are the vertex degrees of some graph if and only if $\sum d_i$ is even. 1.3.28

Proof: ***Sufficiency*(if)**

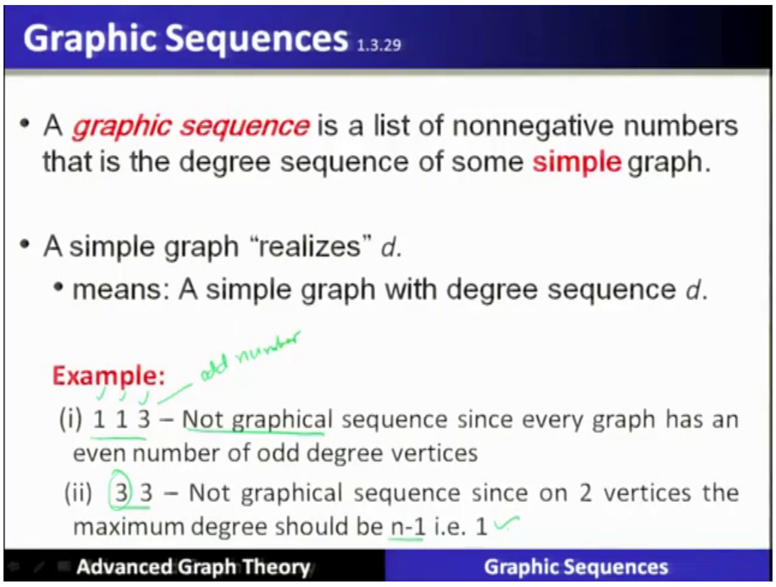
- Suppose that $\sum d_i$ is even.
- We construct a graph with vertex set $v_1, \ldots, v_n$ and $d(v_i) = d_i$ for all i.
- Since $\sum d_i$ is even, the number of odd values is even.
- First form an arbitrary pairing of the vertices in $\{v_i : d_i \text{ is odd}\}$.
- For each resulting pair, form an edge having these two vertices as its endpoints
- The remaining degree needed at each vertex is even and nonnegative; satisfy this for each i by placing $\lfloor d_i/2 \rfloor$ loops at v_i.

Advanced Graph Theory | Graphic Sequences

So, sufficiency condition let us assume that the sum of degrees is even. So, we have to basically find out thy statement that is d 1 d 2 and so on are the vertices of some graph. So, we have to construct a graph with the vertex x v 1 v 2 and so on up to v n and d v i is equal to d i for all the vertices i.

Now, since this sum of degrees of these vertices is even, the number of odd values is even then in that case in the previous slides this particular result we have already seen. So, first we form an arbitrary pairing of the vertices in set of or degrees of vertices, and for each resulting pairing form an edge having these two vertices as it is end points the remaining degrees needed at each vertex is even and non negative, satisfy this by each eye placing a d i by 2 loops at these vertices, and hence we can produce a graph out of it shown that the given set of integers d 1 and so on up to d n are the vertices of some graph.

(Refer Slide Time: 27:47)



Graphic Sequences 1.3.29

- A *graphic sequence* is a list of nonnegative numbers that is the degree sequence of some **simple** graph.
- A simple graph "realizes" *d*.
  - means: A simple graph with degree sequence *d*.

Example:

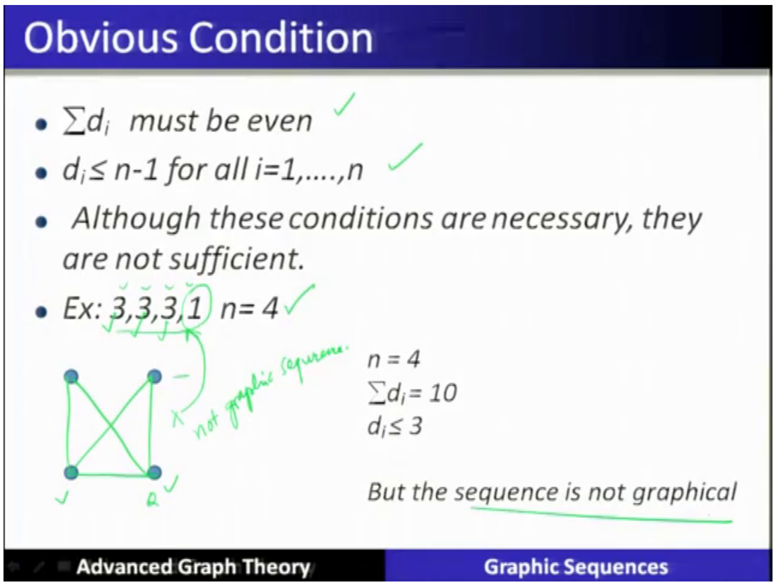
(i) 1 1 3 – Not graphical sequence since every graph has an even number of odd degree vertices

(ii) 3 3 – Not graphical sequence since on 2 vertices the maximum degree should be n-1 i.e. 1

Graphic sequence; graphic sequence is a list of non negative numbers that is the degree sequence of some simple graph. A simple graph realizes D means a simple graph with a degree sequence d. For example, here 1 1 3 these sequence is given, then we can instruct quickly and tell that this is not a graphical sequence why because here we can say that 1 2 3 the odd vertices the degree of odd vertices here, it is a odd number. For any given graph the number of odd vertices is E always even hence this is not a graphical sequence or this particular sequence does not represents the degree of any graph.

Let us take another example 3 and 3 3 and 3 they are not graphical sequence although this particular odd vertices are two in number; that means, they are even in number even then it is not a graphical sequence, because there is another condition which says that the maximum degree here which is called a 3, it should be n minus one at most and that is n minus 1 is only two vertices are there. So, n minus 1 is equal to 1. So, the maximum degree should be 1, but it is 3 so obviously, it is not the degree sequence of any graph.

(Refer Slide Time: 29:29)



Obvious Condition

- $\sum d_i$ must be even ✓
- $d_i \leq n-1$ for all $i=1,\ldots,n$ ✓
- Although these conditions are necessary, they are not sufficient.
- Ex: 3,3,3,1 n= 4 ✓

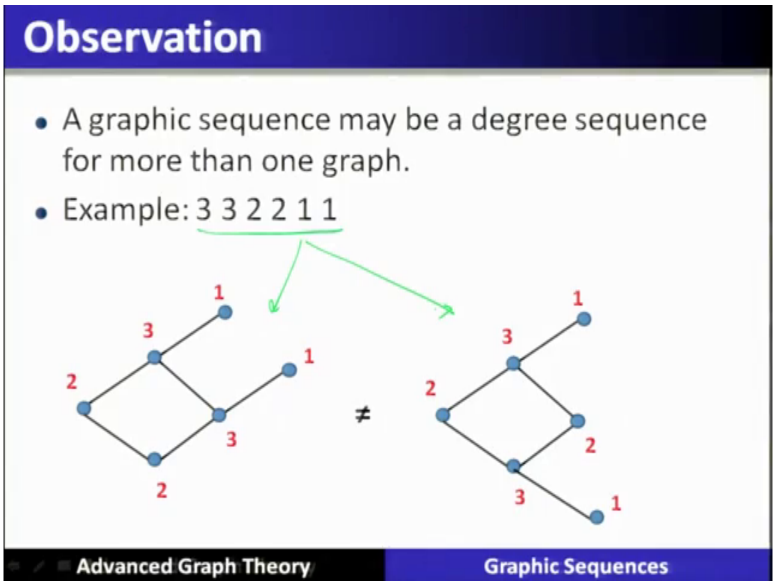
$n = 4$
$\sum d_i = 10$
$d_i \leq 3$

But the sequence is not graphical

So, out of this particular examples we have collected up some of the obvious conditions that degree sum must be even and also the degrees must be at most n minus 1 for all eyes. Although these conditions are necessary, but they are not sufficient for example, we can see a 4 vertex node having the degree sequence 3331 whether. So, here you can see that. So, n is equal to 4. So, the maximum degree is 3. So, 3 is basically allowed and the degree sum is 10, that is also an even number. So, although this is obvious conditions are that is the necessary conditions are satisfied, but it is also not sufficient that these sequence is the graphical sequence let us see how. So, basically let us see this particular vertex. So, we can place an edge like this. So, it has consumed all 3 degrees. So, what about these other two vertices? Let us say this vertex whose degree is now 2 can connect let us say 2 other vertices then only it is degree will become 3 and it will be exhausted.

These particular node has the degree two still it has one degree remaining, what about this node has basically has exceeded it is degree two hence this is not possible to make a graph out of these particular degree sequence hence it is not a graphic sequence and the sequence is not graphic. So, we have seen that although is the necessary conditions are satisfied, but it is also require to be sufficient condition here the sufficient condition is failed hence this particular sequence is not basically graphical.

(Refer Slide Time: 31:43)



Observation

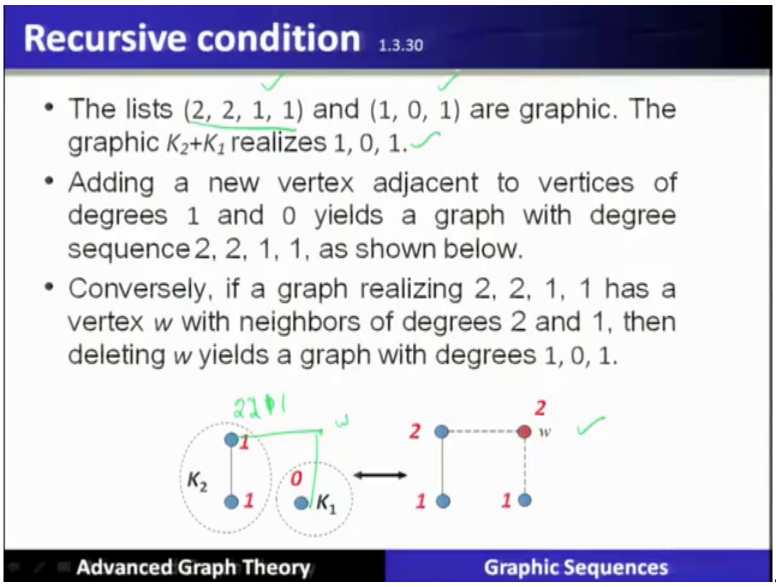
- A graphic sequence may be a degree sequence for more than one graph.
- Example: 3 3 2 2 1 1

1
3
1
2
3
2
≠
1
3
2
2
3
1

Advanced Graph Theory | Graphic Sequences

Another observation is that the graphic sequence for example, this particular sequence represents a particular graph more than one graphs. So, it does not matters so that means, at least one graph it should presents to satisfy that it is a graphic sequence.

(Refer Slide Time: 32:03)



Recursive condition 1.3.30

- The lists (2, 2, 1, 1) and (1, 0, 1) are graphic. The graphic K_2+K_1 realizes 1, 0, 1.
- Adding a new vertex adjacent to vertices of degrees 1 and 0 yields a graph with degree sequence 2, 2, 1, 1, as shown below.
- Conversely, if a graph realizing 2, 2, 1, 1 has a vertex w with neighbors of degrees 2 and 1, then deleting w yields a graph with degrees 1, 0, 1.

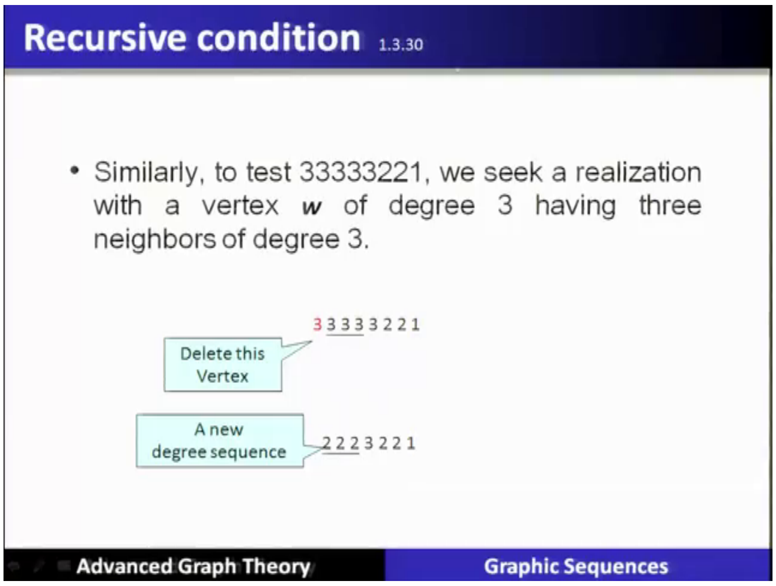
2211
w
1
K_2
1
0
K_1
2
2
w
1
1

Advanced Graph Theory | Graphic Sequences

Now, this particular graphic sequence they are necessary and sufficient condition both are required to be satisfied. So, we will see how this particular both the conditions are to be evolved and how they are going to be satisfied and what that we are going to see a theorem, but before that let us see some of the background why the theorem is required where necessary and sufficient condition both are required to be satisfied.

So, the let us see this particular list 2, 2, 1, 1 and 1,0,1 they are graphic. So, if this is graphic 1, 0, 1. So, 1, 0, 1 means that it is a K 2 plus K 1 this is K 2 plus K 1 this represents 2 2 1 1 graphic sequence. Now this particular graphic sequence if we add a vertex to it w like this and place a edge to this particular node. So, what will happen is another vertex w whose degree will be basically two will come. So, another node w will be added it is degree is two and this degree will be increased to two and the zero degree will become one. So, it is nothing, but the same graphic sequence which we have obtained. So, this we have seen that there is by recursive conditions if one graphic sequence is basically given to us, and we obtain another sequence through that particular sequence by adding a particular vertex and the edges, then if we obtain another graphic sequence this is a recursive way by which we have to prove that this a particular sequence is a graphic sequence or not.

(Refer Slide Time: 34:02)

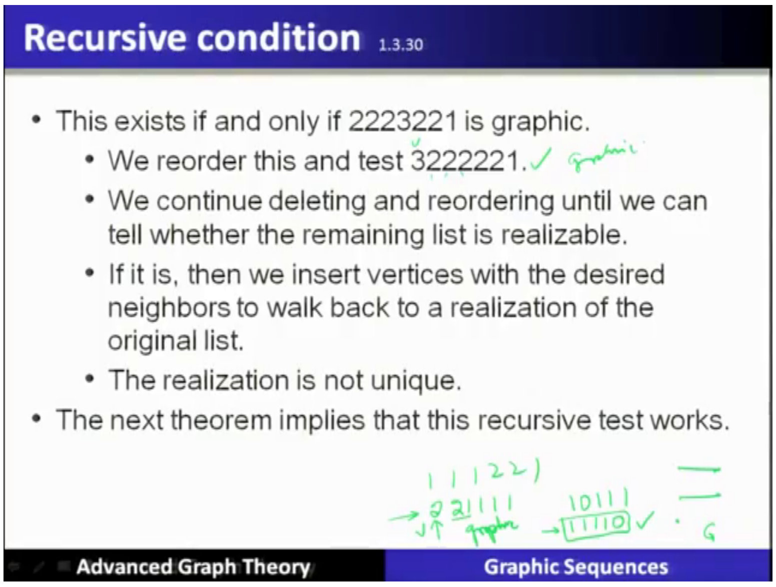


Recursive condition 1.3.30

- Similarly, to test 33333221, we seek a realization with a vertex ***w*** of degree 3 having three neighbors of degree 3.

Similarly, to test whether this is a sequence 333221 is a graphic sequence or not. So, we seek realization with a vertex at w. Let us take a vertex with a highest degree and it is that many number of neighbors for example, degree is three. So, 3 neighbors 1 2 3 and then if we delete this particular vertex and remove the degree and reduce the degree by one we will obtain another sequence.

(Refer Slide Time: 34:40)



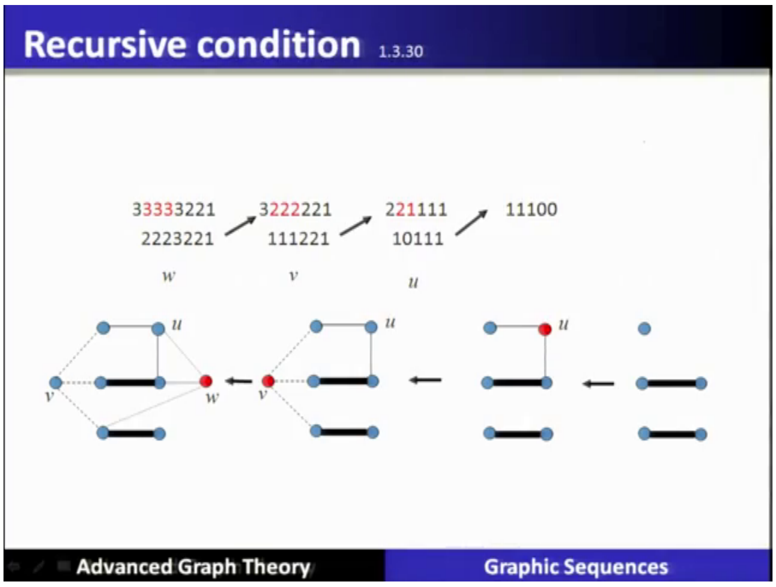
Recursive condition 1.3.30

- This exists if and only if 2223221 is graphic.
  - We reorder this and test 3222221. ✓ graphic
  - We continue deleting and reordering until we can tell whether the remaining list is realizable.
  - If it is, then we insert vertices with the desired neighbors to walk back to a realization of the original list.
  - The realization is not unique.
- The next theorem implies that this recursive test works.

Let us see whether this sequence is graphic or not. So, we will order it and then we will continue doing it. So, 3 is the highest degree and 3 operates neighbor we choose and once we delete it we will obtain 1 1 1 2 2 1. So, if we reorder them it will become 2 2 this becomes the sequence.

Now, we have to check whether this is the graphic or not. So, we have to choose this particular vertex to it is two neighbors we have to choose and if we remove it. So, it will become 1 it will become 0 then 1 1 1. So, if we reorder it 1 1 1 1 0. So, we have to check whether this is graphic or not. So, 1 1 1 0 we can see whether this graph can be drawn out of it or not. So, 1 2 3 4, 1 2 3 4 and 1. So, this will represent a particular graph. So, hence this is a graphic sequence, if this is the graphic sequence we have we can add an edge to it and place the vertices like this. So, this also will become graph 8 sequence and this particular on this particular sequence if we add an edge we will obtain this particular sequence. So, this sequence has also become graphic hence this is a recursive construction by which we can prove that this whether the given sequence is a graphic sequence or not.

(Refer Slide Time: 36:18)

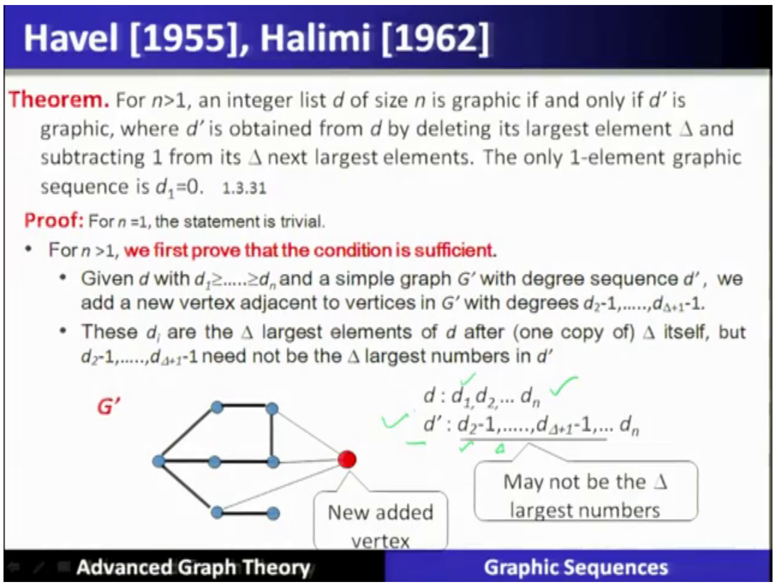


Recursive condition 1.3.30

Advanced Graph Theory Graphic Sequences

The entire thing is illustrated here in this particular picture that I have already explained you. Now with this particular background, a theorem is given on this particular graphic sequence and this theorem is given by Havel and Halimi.

(Refer Slide Time: 36:26)



Havel [1955], Halimi [1962]

Theorem. For $n>1$, an integer list d of size n is graphic if and only if d' is graphic, where d' is obtained from d by deleting its largest element Δ and subtracting 1 from its Δ next largest elements. The only 1-element graphic sequence is $d_1=0$. 1.3.31

Proof: For $n=1$, the statement is trivial.

- For $n>1$, we first prove that the condition is sufficient.
  - Given d with $d_1 \geq \ldots \geq d_n$ and a simple graph G' with degree sequence d', we add a new vertex adjacent to vertices in G' with degrees $d_2-1, \ldots, d_{\Delta+1}-1$.
  - These d_i are the Δ largest elements of d after (one copy of) Δ itself, but $d_2-1, \ldots, d_{\Delta+1}-1$ need not be the Δ largest numbers in d'

G'
$d: d_1, d_2, \ldots d_n$
$d': d_2-1, \ldots, d_{\Delta+1}-1, \ldots d_n$
Δ
May not be the Δ largest numbers
New added vertex

Advanced Graph Theory Graphic Sequences

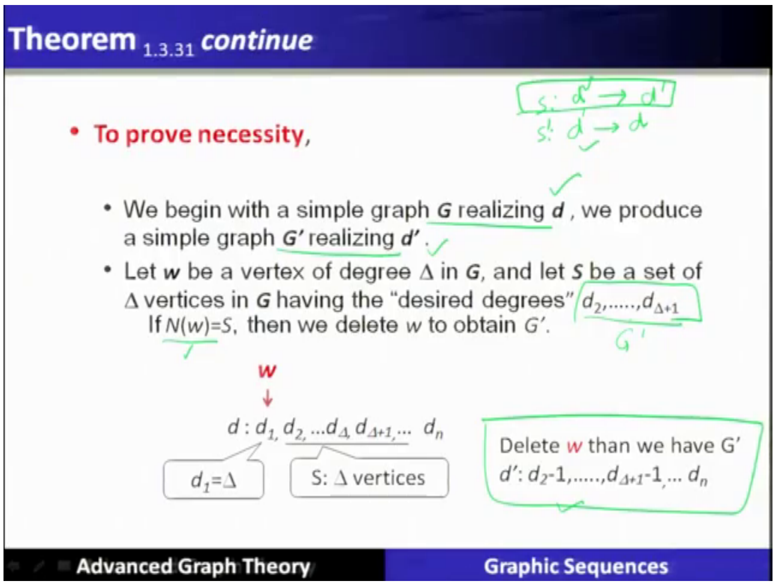
So, the theorem states that for n is greater than 1 and integer list d of size n is graphic. If and only if d prime is graphic where d prime is obtained from if I deleting it is largest element delta and subtracting one from it is delta next largest element, the only one element graphic sequence is d 1 is equal to 0.

So, let us see the proof. So, for n is equal to one thy statement is trivial. So, for n is greater than 1 we have to prove the condition is sufficient. To prove the condition is sufficient let us assume that we have given a d with d 1 greater than or equal to d 2 and so on up to d n and a simple graph G prime with a degree sequence D prime we add a u vertex adjacent to the vertices in g prime with the degrees with the degrees d 2 minus one and so on D delta plus 1 minus 1.

So, these d i s are the delta largest elements of d after one copy of that delta itself, but d 2 minus one and so on up to D delta plus one minus one need not be delta largest element in d dash. So, we can see here that d is given d 1 d 2 and so on up to d n and D prime which we obtain is basically nothing, but the d 2 minus 1 that is a if we remove d 1 and that delta plus 1 minus 1 that particular delta number of elements we have to reduce it is degree 1 and after delta plus 1 all the up to d n all this particular degrees are not basically touched upon. So, that is nothing, but D prime

Now, to prove the necessity so that was the sufficiency condition.

(Refer Slide Time: 39:01)



Theorem 1.3.31 *continue*

- To prove necessity,
  - We begin with a simple graph **G** realizing ***d***, we produce a simple graph ***G'*** realizing ***d'***.
  - Let ***w*** be a vertex of degree Δ in ***G***, and let ***S*** be a set of Δ vertices in ***G*** having the "desired degrees" $d_2,\ldots\ldots,d_{\Delta+1}$ If $N(w)=S$, then we delete w to obtain G'.

$$d : \overset{w}{\overset{\downarrow}{d_1}}, \underbrace{d_2, \ldots d_\Delta, d_{\Delta+1}}_{S:\ \Delta \text{ vertices}}, \ldots\ d_n \qquad d_1=\Delta$$

$$\boxed{S: d' \to d'}$$
$$S: d' \to d$$
G'

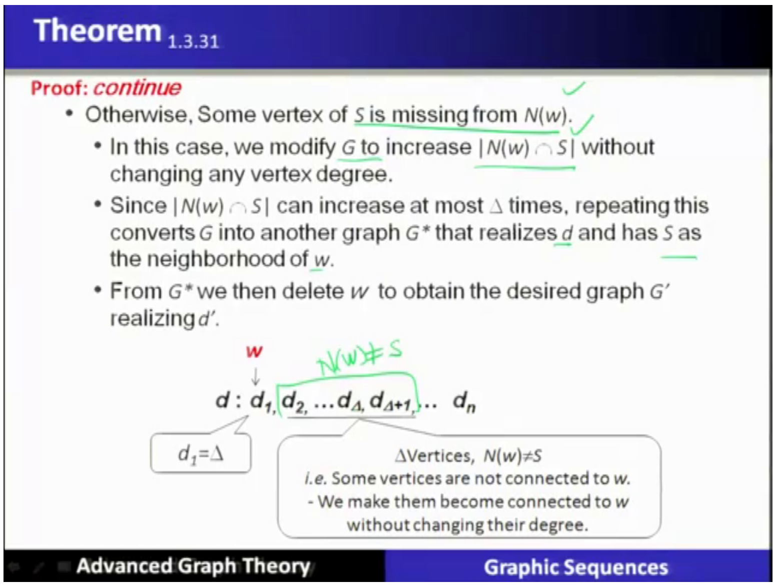
Delete *w* than we have G'
$d': d_2-1,\ldots\ldots,d_{\Delta+1}-1,\ldots\ d_n$

Advanced Graph Theory — Graphic Sequences

Now, we have to prove the necessity condition, in this necessity condition we have to begin with the simple graph G realizing d v produce a simple graph g prime which is realizing D prime. So, in the sense what we are proving is if we are given a sequence s of d. So, if we can obtain another graphic sequence D prime from it then this is the graphic sequence or if we have a graphic sequence then if we place a vertex, and if we can obtain d then basically it is also a graphic sequence; so, here s prime is given and if we place an edge. So, both ways we are going to prove.

Now, here we are going here we are given a graph G which is realizing d so; that means, a graph G is given and we can sequence it is vertices according to their non increasing degrees. So, we produce after that using that another simple graph d prime realizing d prime by removing that particular edge. So, let w be the vertex which is having highest degree in delta and let s B the set of delta vertices in g having the degrees d 2 and so on up to d delta plus 1. Now if this N w that is neighbors of w is S, these set of vertices then we delete that particular w and we obtain g prime in this particular manner this is pretty straight forward. However, if they are not basically if neighbors of w is not these particular set of degrees, but some other degrees are there.

(Refer Slide Time: 41:10)



Theorem 1.3.31

Proof: *continue*

- Otherwise, Some vertex of S is missing from $N(w)$. ✓
  - In this case, we modify G to increase $|N(w) \cap S|$ without changing any vertex degree.
  - Since $|N(w) \cap S|$ can increase at most Δ times, repeating this converts G into another graph G^* that realizes d and has S as the neighborhood of w.
  - From G^* we then delete w to obtain the desired graph G' realizing d'.

$$d : \overset{\overset{w}{\downarrow}}{d_1}, \overbrace{d_2, \ldots d_\Delta, d_{\Delta+1}}^{N(w) \neq S}, \ldots\ d_n$$

$d_1 = \Delta$

ΔVertices, $N(w) \neq S$
i.e. Some vertices are not connected to w.
- We make them become connected to w without changing their degree.

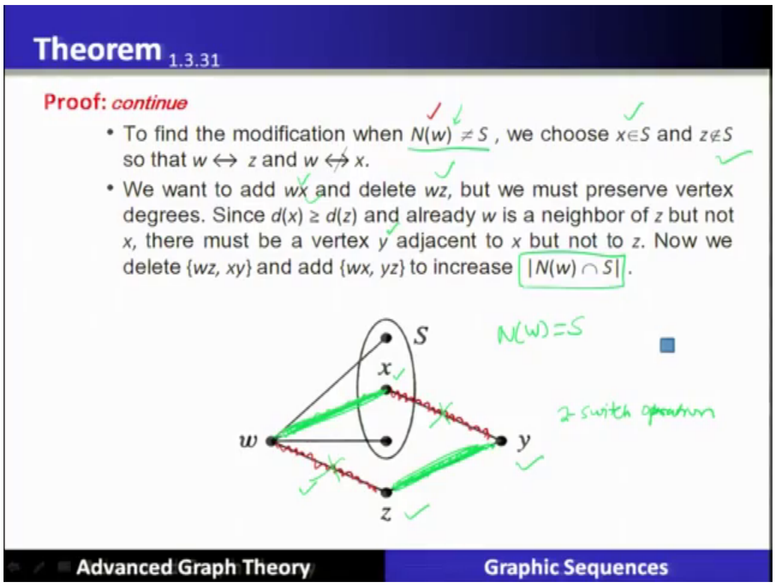
Advanced Graph Theory | Graphic Sequences

Where some vertex of s is missing from N of w, then in that case we have to modify this particular G and we will we will increase this neighbor of w intersection of S without changing any vertex degrees.

So, this neighborhood of w intersection S can increase at most delta times repeating this converts G into another graph G star that realized d and has S as the neighborhood of w. Now from G star we so; that means, if the maximum degree node does not have it is neighborhood and w provided in s then we can basically swap or a two switch operation we have to do one node we have to remove out of s and another node which is basically there should be there will be entered it and make it the case like the previous case here and then we can go ahead for the delusion let us see how we are going to do this.

So, let us see that if w is the vertex d 1 and delta vertices which we are basically considering, but here these are not the neighbors of S; that means, there exist some vertex which is not according to the order, it should be here it should be on the other side.

(Refer Slide Time: 42:48)



Theorem 1.3.31

Proof: *continue*

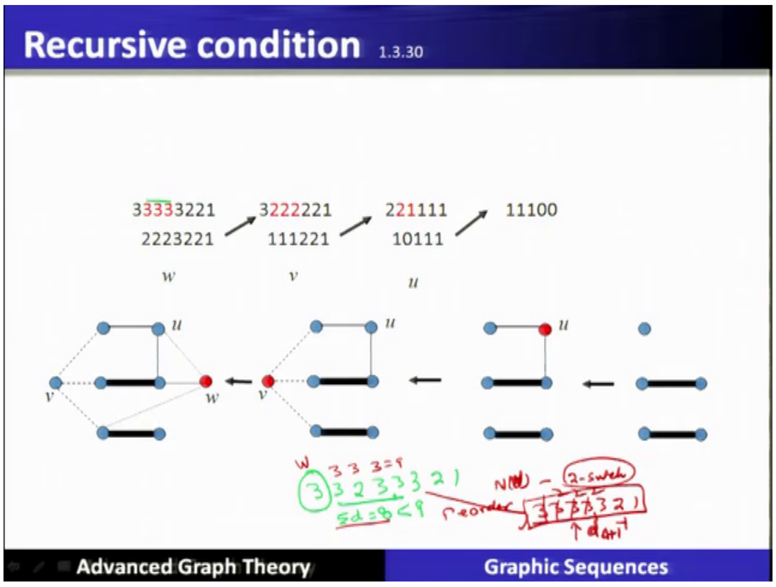
- To find the modification when $N(w) \neq S$, we choose $x \in S$ and $z \notin S$ so that $w \leftrightarrow z$ and $w \not\leftrightarrow x$.
- We want to add wx and delete wz, but we must preserve vertex degrees. Since $d(x) \geq d(z)$ and already w is a neighbor of z but not x, there must be a vertex y adjacent to x but not to z. Now we delete $\{wz, xy\}$ and add $\{wx, yz\}$ to increase $\boxed{|N(w) \cap S|}$.

So, to find the modification when N of w that is neighbor of w is not equal to s we choose x which is in S and element z which is not in S. So, that w and z which are adjacent, but w and x are not adjacent. So, we obtain in this case this particular edge, this particular edge which is given here in this particular case present, but the elements which is not in s that is called x is not present and this edge is also not present.

So, we will do a two switch operation, in this two switch operation what we will do that we want to add w x this is to be added up and will be introduced in basically N of w and we will delete w z. But we must preserve the vertex degrees since the degree of d of x is greater than degree of z and already w is the neighbor of z, but not of x there must be another vertex y. So, if x is entered and w and z is removed. So, another vertex has to be there and that is y which is adjacent to x, but not to z. Now we delete w z and x y is also to be deleted, and add w x this we have already added and y z is also added.

So, having done that we will basically increase n w and now the n w will become equal to S. In another way, we can understand that a particular degree.

(Refer Slide Time: 45:15)



Recursive condition 1.3.30

Advanced Graph Theory

Graphic Sequences

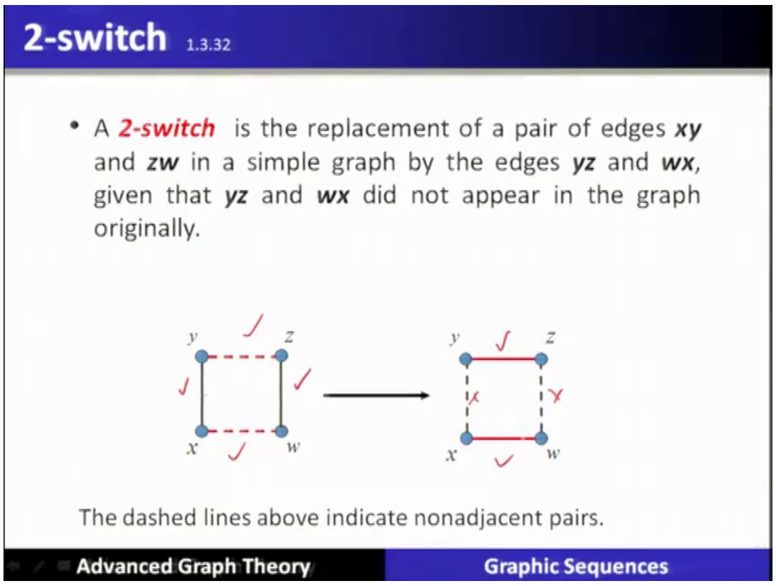
Let us say take the example 33331. So, in this particular example what we see is that 3 and it is 3 neighbors are having degree 3 3 3 if it is not for example, if we place this particular 3 here and 2 is over here, then this particular degree sum that is the sum of the degrees of 1 2 3 is basically this becomes 8 and which is less than. If you consider this as 3 degree that is 9 that is when it should have been in this particular condition, then this particular degree sum will become 9.

So, in any case if the degree sum is less than some other vertices, which is not present in this N of s sorry N of w this is w neighbor of w, then we have to apply the two switch operation and that is why we have to call it as reorder every time. So, 3 after this reordering it will become 3 3 3 3 2 1, then this particular 3 will have it is 3 neighbors in this basically non increasing order of the degree. So, here it is not, it is wall hitting the non increasing order degree. So, we have to reorder the operation and in this theorem we have seen the application of a two switch operation.

So, once it is done in this particular form, then we can apply; that means, we remove this w and basically we can reduce the number of vertices how many times? D delta plus one minus one we have to do this become 2, this become 2, this become 2 and so on then again we have to reorder it.

So, reordering is important operation and that we have not seen how that is being basically covered here in the sufficiency condition of the proof. So, necessary and sufficient condition both are required, we have already seen in this particular proof of graphic sequence.

(Refer Slide Time: 47:50)



2-switch 1.3.32

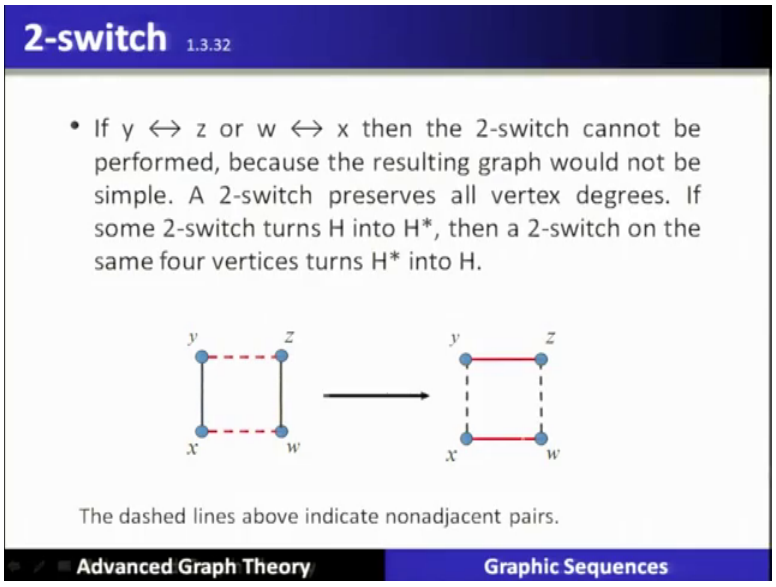
- A ***2-switch*** is the replacement of a pair of edges ***xy*** and ***zw*** in a simple graph by the edges ***yz*** and ***wx***, given that ***yz*** and ***wx*** did not appear in the graph originally.

The dashed lines above indicate nonadjacent pairs.

Advanced Graph Theory | Graphic Sequences

Now, two switch operation, we have used let us briefly explain it a two switch is the replacement of a pair of edges x y and z w in a simple graph by the edges y z and x w. Given that x y z and w x did not appear in the graph originally. So, we will switch it we will exclude this particular edges, and we will include these two edges and this is called a two switch operation, that we have already used in the previous theorem.

(Refer Slide Time: 48:33)



2-switch 1.3.32

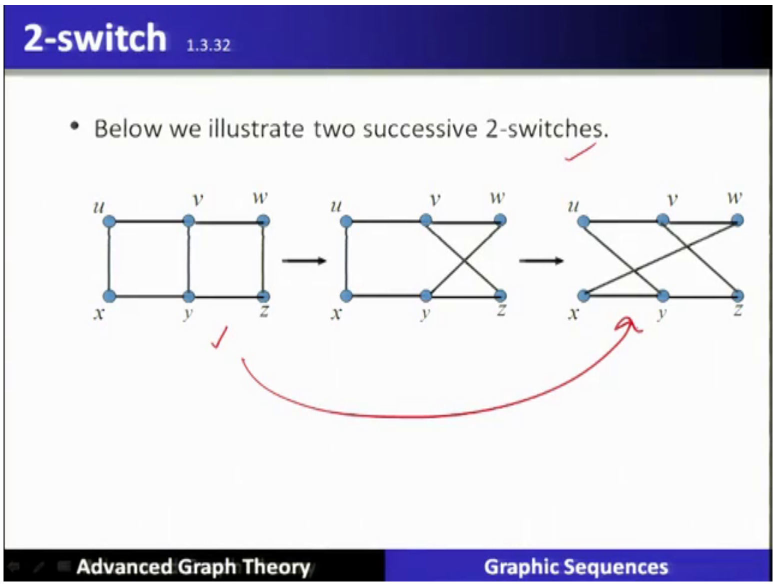
- If y ↔ z or w ↔ x then the 2-switch cannot be performed, because the resulting graph would not be simple. A 2-switch preserves all vertex degrees. If some 2-switch turns H into H*, then a 2-switch on the same four vertices turns H* into H.

The dashed lines above indicate nonadjacent pairs.

Advanced Graph Theory | Graphic Sequences

Now, if y and z they are not they are adjacent or w and x is adjacent, then two switch cannot be performed because the resulting graph would not be simple the 2-switch preserves all the vertex degrees if some 2 switch down H to H star then a 2 switch on the same four vertices downs H star into H. So, that is lines above indicate non adjacent pairs.

(Refer Slide Time: 49:06)



2-switch 1.3.32

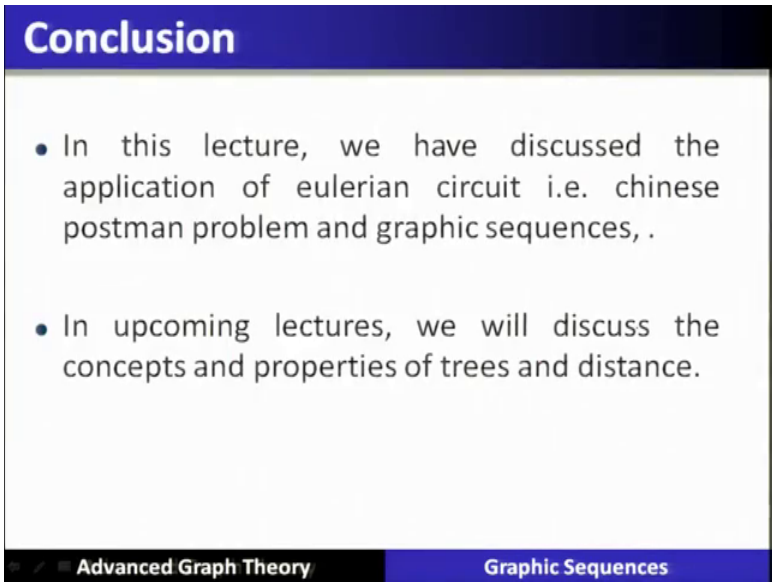
- Below we illustrate two successive 2-switches.

u
v
w
x
y
z
u
v
w
x
y
z
u
v
w
x
y
z

Advanced Graph Theory | Graphic Sequences

Now, we can see here in this particular illustration that two successive two switches operation has transformed this particular graph into this particular graph, and hence it will preserve the vertex degrees as we have already mentioned.

(Refer Slide Time: 49:23)



Conclusion

- In this lecture, we have discussed the application of eulerian circuit i.e. chinese postman problem and graphic sequences, .
- In upcoming lectures, we will discuss the concepts and properties of trees and distance.

Advanced Graph Theory | Graphic Sequences

So, conclusion in this lecture we have discussed the application of Eulerian circuit, that is a Chinese postman problem which is an optimization problem in many applications. And also we have seen the graphic sequence and a particular theorem and in this particular theorem we have seen the necessary and sufficient both condition are required to basically test whether a sequence is a graphic sequence or not. In the upcoming lectures we will discuss the concepts and properties of a trees and distance.

Thank you.