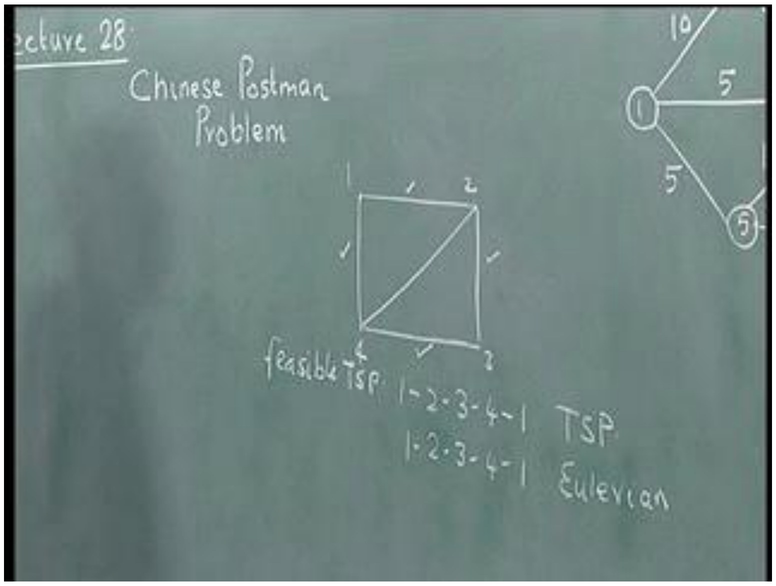
Advanced Operations Research

Prof. G. Srinivasan
Department of Management Studies
Indian Institute of Technology, Madras

Lecture - 28
Chinese Postman Problem

In this lecture we study the Chinese postman problem.

(Refer Slide Time: 00:20 min)



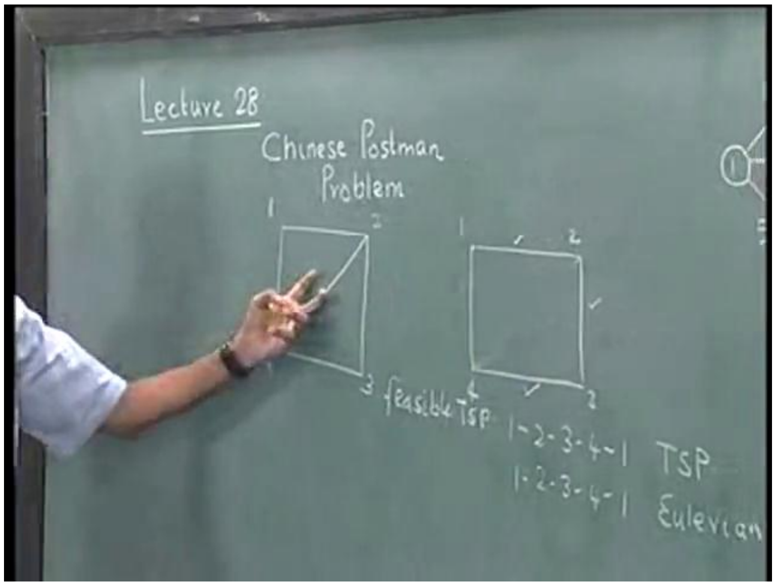
The Chinese postman problem is quite close to the Travelling Salesman Problem, but it has some differences when compared to the TSP. So, what is the Chinese postman problem? Before we formally introduce the Chinese postman problem, let us look at a simple graph and try to understand what this problem is. So we look at a simple graph, which has four vertices 1 2 3 4 and four edges 1to 2, 2 to 3, 3 to 4 and 4 to 1. Now, we know that this graph is Hamiltonian in the sense, that we can have a feasible solution to the Travelling Salesman Problem, which is 1 2 3 4 and 1.Starting at vertex one, we are able to visit every other vertex once and only once and come back to the starting point so this has a feasible solution to the TSP.

Now, can we consider another problem where we start at a vertex, we visit every edge once and only once and come back to the starting point. So we start at vertex one let us say we visit this edge first, then we visit this edge first, then we do another edge here and then this edge once and only once and we come back to the starting point. So again the feasible solution to the problem where we start at a vertex visit every edge of the graph once and only once and come back to the starting point turns out to be the same solution as that of the TSP, but then we call this circuit as an Eulerian circuit. Now, let us make a small change in this graph and say that we add another edge here between 2 and 4.

Now, if we want to solve the TSP or in this case to find out Hamiltonian circuit, which means you start at a vertex, visit every other vertex once and only once and come back to the starting point. Now, 1 to 22 to 33 to 44 to 1 is again feasible to the TSP and it is a Hamiltonian circuit. But if we try and solve the second problem for the same graph, which means we start at a particular vertex visit all the edges once and only once and come back to the starting point. Now we can go from 1 to 2 we can do 2 to 3 we can do 3 to 4 comeback 4 to 1, which means we have not come or covered this edge.

If we choose to come from 3 to 4 and go back from 4 to 2 and then come back to 2 to 1, it means we are duplicating this twice and we have not moved into this. So, for this particular graph we do not actually have a solution a feasible solution where starting from a given vertex we visit all the edges once and only once and come back to the starting point. So, the problem where if we are able to find in a graph, where starting from a vertex we visit every edge once and only once and we come back to the starting point, then such a graph is called Eulerian and this circuit is called the Eulerian circuit. Now, this graph along with this edge before is not Eulerian; whereas, the original version of this graph, which did not have this edge, was Eulerian.

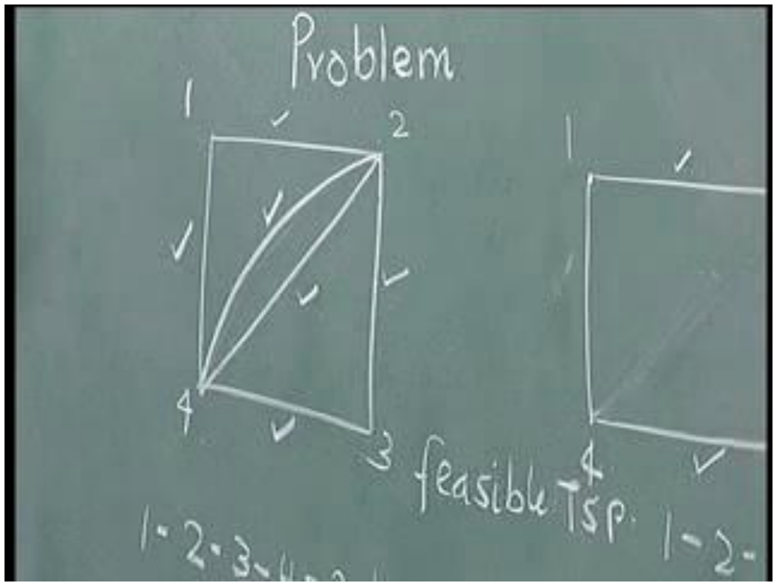
(Refer Slide Time: 04:37 min)



So let us keep this and let us again draw the second part of this graph where we say 1 2 3 and 4. This graph is Eulerian as well as Hamiltonian; whereas, this graph is Hamiltonian or it has a Hamiltonian circuit, but this is not Eulerian. So we observe a couple of things one this graph is Eulerian. One of the interesting things that we have seen earlier already is that is the degree of every vertex of the graph is an even number, then that graph is Eulerian. In this case we realize or observe that the degree is 2 degree is 2, but the degree is 3 and the degree is 3, so you have some vertices that have odd degree and we already know that the number of vertices that will have odd degree is an even number. So these two have odd degree or we can generalize and say, that if a graph has vertices with odd degree, then that graph is not Eulerian.

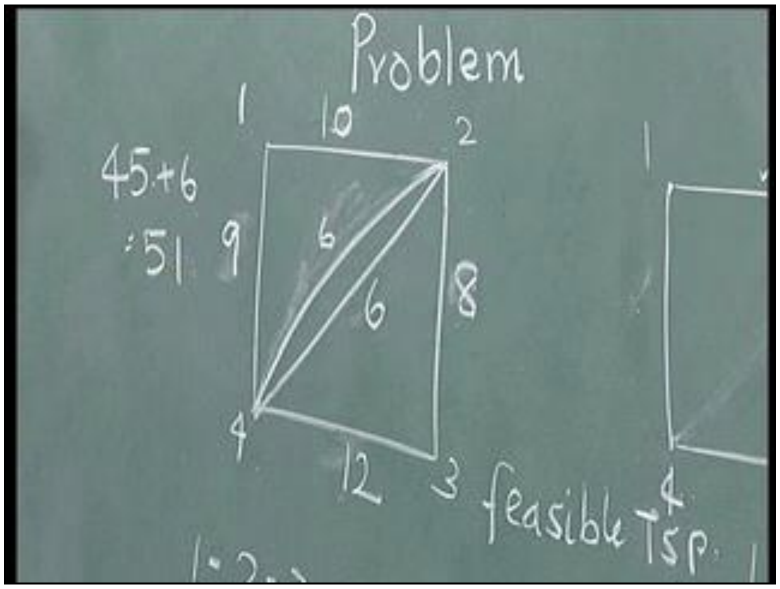
So, first and foremost given a graph like this, then we know that this graph is not Eulerian. Now we can do one more thing if we are still interested in getting an Eulerian circuit, then we may have to add some edges into this graph.

(Refer Slide Time: 06:00 min)



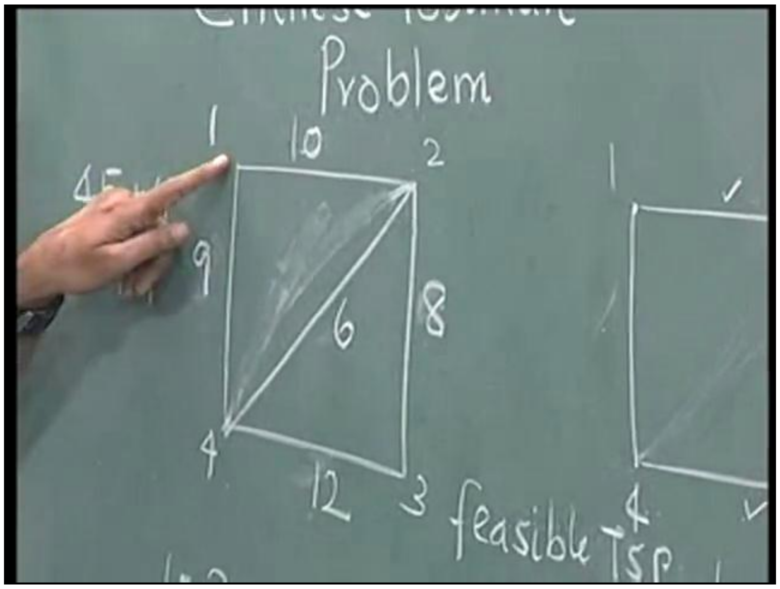
For example if we add one more edge here so that we say we go from 1 to 2, 2 to 3, 3 to 4 again we go from 4 to 2 come back from 2 to 4 and then do 4 to 1, then we have got an Eulerian circuit. So the Eulerian circuit can be written as 1 2 3 4 2 4 1. Now, if the graph is Eulerian, then this graph has one Eulerian circuit or a unique circuit in the sense that the distance that it has is the same and such a distance are length or such an Eulerian circuit will be equal to sum of the weights of all of them. Now in this example, this particular edge was not there in the graph and the graph was not Eulerian. So to make this Eulerian we have to add some edges or essentially duplicate some edges. So a graph that is non Eulerian can be made Eulerian by duplication of edges so this is the background that is required to understand the Chinese postman problem. Now what is the Chinese postman problem, let us go back to our discussion on this graph again and let us say we also provide some distances, which are associated with this as arc weights.

(Refer Slide Time 07:42 min)



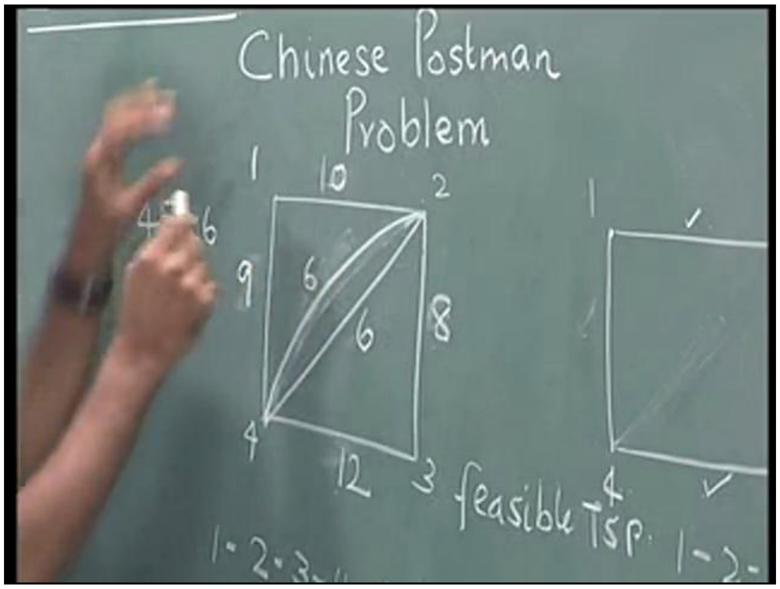
Suppose, we say that this distance is 10, this is 8, this is 12, this is 6 and this is 9, now the total if this graph is not Eulerian so sum of the distances 10 plus 8 equal to 18 plus 12 equal to 30. 36 plus 9 equal to 45 and we know that this is not Eulerian, because this has an odd degree of 3 this also has an odd degree of 3, so this graph is not Eulerian. Now, in order to make this graph Eulerian, we just now said that if we can duplicate this and add another 6 more then it becomes an Eulerian, so 45 plus 6 equal to 51. So by adding some edges into a graph that is non Eulerian, then we will be able to bring it or make it Eulerian and get a solution to this. So, automatically the question comes that if the graph is not Eulerian and we have to either add edges or duplicate edges what is the minimum weight or minimum extra length that we add, such that we are able to get an Eulerian out of this.

(Refer Slide Time 09:00 min)



The physical meaning or significance of this problem, which is called the Chinese postman problem, comes like this. We can assume that this one is a post office and 1 to 2, 2 to 3, 3 to 4, 4 to 1 and 2 to 4 are streets that a postman starting from the post office has to deliver letters. So, what the postman would ideally like to do is to start from the post office visit each street once and only once and is able to come back to the post office. Now that means, if this graph is Eulerian, then the corresponding Eulerian circuit will be the tour, that the postman takes from the post office visit each street once and only once and he comes back to this.

(Refer Slide Time 09:55 min)



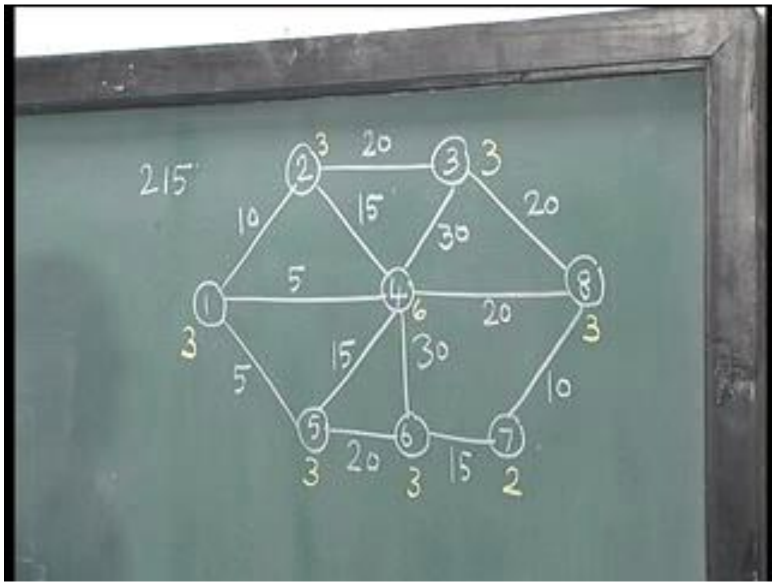
Now, this is not Eulerian and therefore we said that by adding this we make it Eulerian. Now, if we look at this carefully the postman will now deliver in this street 1 to 2, deliver in this street 2 to 3, deliver in this street 3 to 4, deliver in this street 4 to 2 come back and do a dummy, which means where this particular thing the postman does not deliver anything he comes back and then delivers in 4to 1. So, this is like the extra distance that the postman has to walk, so that the post man is able to get an Eulerian circuit out of it. So, if the given graph is not Eulerian the problem is, what is the minimum extra length that we add? And from the point of view of the post man what is the extra length that the postman walks, without delivering any letter such that the postman gets an Eulerian circuit out of it and comes back to the starting point. Now, this is called the Chinese postman problem, because it originated somewhere in a Chinese journal many years ago.

Now there are some interesting applications of the Chinese postman problem, even though the problem is called the Chinese postman problem. And the application to postal delivery is fairly obvious, there are other applications of this problem, where these ideas can be used particularly in collecting waste or garbage in streets in a community, which is quite similar to the postman going and delivering the letters, or particularly in cleaning the streets, when the street is full of snow in particularly countries, where its snows during certain parts of the year.

So the Chinese postman problem has such kind of applications, but largely is the problem where number one is to verify, whether the graph is Eulerian and this graph is not Eulerian what is the minimum extra weight that that the person has to walk extra or the minimum extra weight or length, that has to be added to the network so that the resultant network becomes Eulerian, so that is called the Chinese postman problem.

Now we will look at a straightly bigger network and then we will try and explain two algorithms, which will help us to solve the Chinese postman problem, which means two algorithms that would help us to identify arcs with minimum weight, that have to be added to the network so that an Eulerian circuit is obtained.

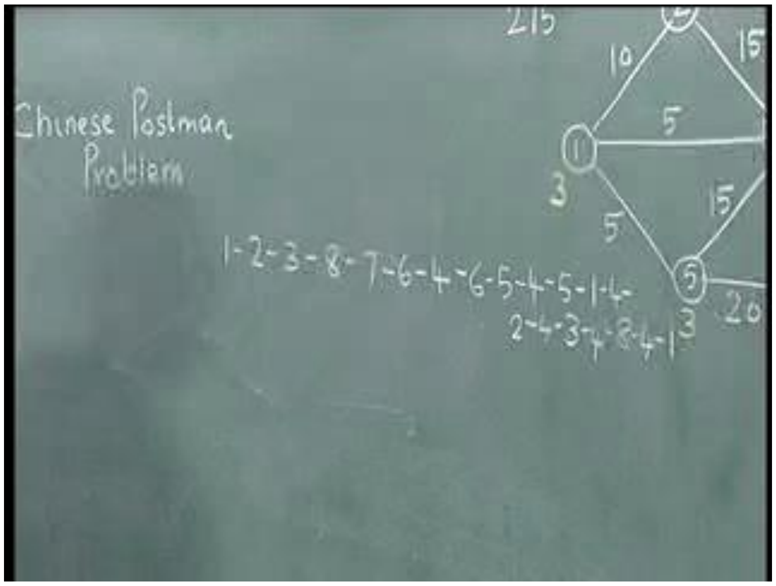
(Refer Slide Time: 12:30 min)



Now we look at a network like this, now this network has eight nodes and several arcs that are given here. The arc lengths are also given accordingly. So the first thing we need to observe is whether this is Eulerian. If this is Eulerian, then the question is to find the Eulerian circuit. We have already said that if the graph is Eulerian, there is one Eulerian circuit and the weight is equal to sum of the weights of all the arcs. And if the graph is not Eulerian, then in order to make it Eulerian we have to add some more, so obviously the sum of all these weights is certainly a lower bowel to the total distance travelled or is the lower bowel to the length of the Eulerian circuit that we wish to have. So sum of these lengths would now be 10 plus 5 is 15 plus 5 is 20, 20 plus 20 is 40 plus 15 equal to 55 plus 15 equal to 70, 70 plus 20 is 90 so 120 - 135 - 155 - 185- 205 plus 10 is 215 is the sum of the lengths of all these. So two 215 is a sum of all these lengths and 215 is an obvious lower bowel to the length of the Eulerian.

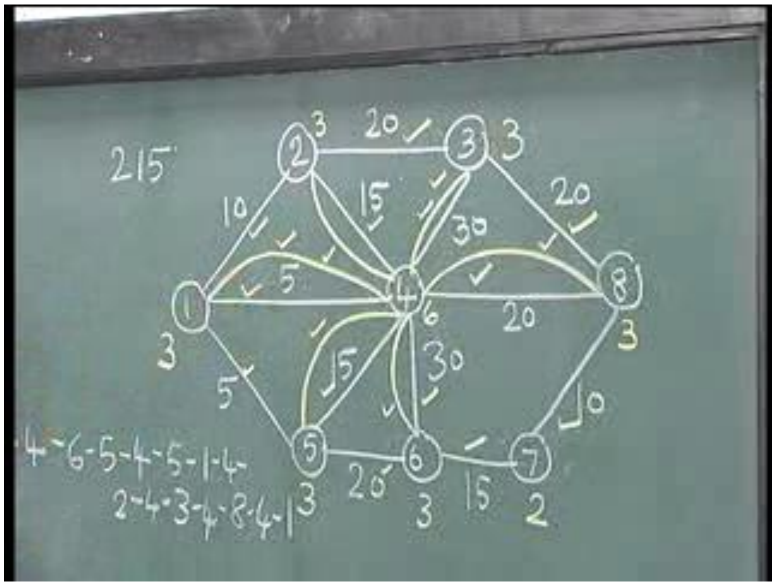
Now, we need to verify whether this graph is Eulerian in order to do that we try and find out the degree of every vertex. Now, the degree of this vertex is 3 so the moment we find a vertex with odd degree we can say that the graph is not Eulerian, nevertheless we find the degrees of all the vertices. So the degree of each of these vertex degree of this vertex is 3.There are three arcs degree again is 3 degree, again is 3 degree is 123 456 degree is 3, degree is once again 3 degree is 21 2 degree is again 3. So how do the eight vertices six of them have odd degree the number of vertices having odd degree is an even number. So six of these vertices have odd degree, while two vertices have even degree vertex 7 and vertex 4, they have even degree while the rest of them odd degree. So this graph is not Eulerian. So what are the things we can do to get an Eulerian is by arbitrarily duplicating edges in this, so that we may still get a resultant Eulerian. So let me just give an arbitrary solution or an arbitrary Eulerian by duplicating edges extra, so that the resultant graph we get as Eulerian.

(Refer Slide Time: 15:37 min)



So let us go back and do that lets look at this solution. So it is 1 - 2 - 3 - 8 - 7 - 6 - 4 - 6 - 5 - 4 - 5 - 1 - 4 - 2 - 4 - 3 - 4 - 8 - 4 - 1. Now this looks like a fairly lengthy Eulerian circuit. Let us try and map this circuit on this.

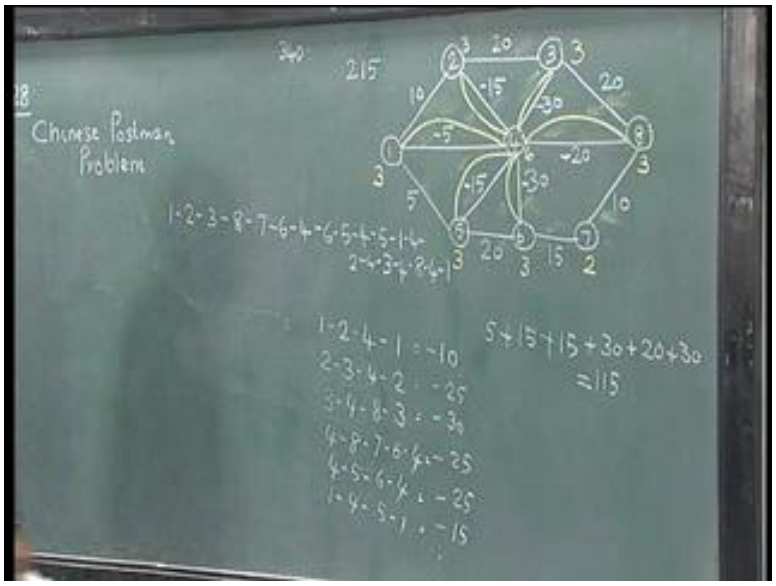
(Refer Slide Time: 16:19min)



Now, what we do, we first assume that the person goes from 1 to 2, which means we clear this edge, then the person goes from 2 to 3 so this edge, 3 to 8 this edge, 8 to 7 this edge, 7 to 6 here, 6 to 4 here, again 4 to 6 which means we are duplicating this edge. So here 6 to 5 this one, 5 to 4 again this one again duplicate this 4 to 5 this one 5 to 1 here, 1 to 4 we go back here and then 4 to 2 you come here, 2 to 4 we duplicate 4 to 3 and then we come back again 3 to 4 and then we go from 4 to 8 and then we again duplicate this to come from 8 to 4 and then we duplicate this again to come from 4 to 1. So this is a solution where the ones shown in yellow are the arcs that have been duplicated.

Now, if we duplicate this many arcs we are able to get an Eulerian as we have shown, because we have already shown with the tick that every edge that is there whether it was an original edge or whether it was a duplicated edge there been now covered in the Eulerian circuit. So when we consider this big Eulerian circuit, which means that, we have to duplicate all these edges, which are shown in yellow.

(Refer Slide Time: 18:27 min)



So the extra distance that the person has to travel, if the person is accepting this Eulerian is given by the sum of the weights of those duplicated edges. And those are there is duplication here. So there is a 5 there is a duplication here there is a 15, there is a duplication here there is a 15, so these three are covered, then we do this, this and this, so 5 plus 15 plus 15 plus 30 plus 20 plus 30. So this is 5 plus 15 plus 15 corresponding to these three and then we have a 30 that is duplicated, a 20 that is duplicated and a 30 that is duplicated. So the total extra distance that the person travels for this is given by 5 plus 1520 plus 15 is 3565 plus 20 85 plus 30 is 115 is the extra distance this person travels and the total distance will be 215 which is the sum of the weights plus another 115 which is equal to 340.

Now after the duplications we can observe that this had an even degree of 3 now this has 4 this has an even degree of 3 this has 4, this has an even degree of 3 now this has 4 this has an even degree of 3this has 4. Again this had an even degree of 3 it has 4 this had an even degree of 3 this has 4 this 2 remains as 2, this 4 which had 6 now has 24 6 810 12 again an even degree. So, all the vertices have even degree so we have got this Eulerian.

Now, for this feasible solution we have an extra distance of 115. So the real optimization problem there is we want to keep this 115 minimum minimize the extra distance such that after duplication the graph continues to be Eulerian. So what is the extra distance minimum extra distance that the person has to travel and which are the edges corresponding to those extra distances. Now this 115 has is to come out of a feasible solution is it optimal, can we bring down this 115 is the next question. If this 115 be infeasible actually gives us an upper bound value of 340 to the Eulerian circuit the load 215 was the lower bound this was the sum of the original weights of the original arcs the graph was not Eulerian, therefore 215 is a lower bound to the Chinese postman length or the Eulerian length.

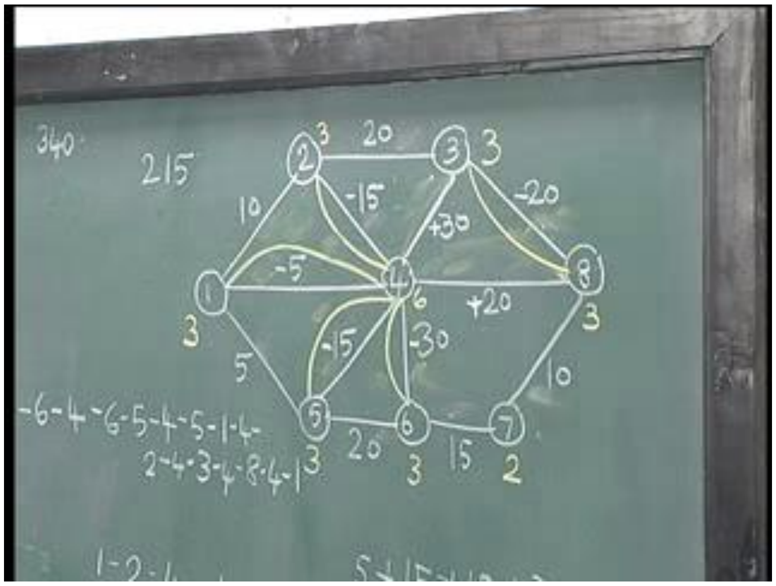
Now 340 is an upper bound, because by duplicating some edges arbitrarily we have now got an Eulerian circuit, which is a feasible solution. So, feasible solution to a minimization problem is an upper bound so we get solution equal to 340. So the real optimization now comes in where we can bring down this 115 it is not about here we are duplicated six edges. It is not about minimizing the number of edges that we wish to duplicate, it is about minimizing the total distance associated with the extra edges that have to be duplicated. So can we bring down this 115 is the optimization question now. One of the ways of handling this is now we look at two algorithms, which actually help us reduce this 115.

The first algorithm works as follows, right now let me leave out all this. What we do in the first algorithm is this, whichever arc has been duplicated the first thing we do is simply make the arc weight negative. So this 15 becomes - 15 this 5 becomes minus 5 this 15 becomes minus 15 this becomes minus 30 this becomes minus 20 and this becomes minus 30. So, whichever arc has been duplicated we simply make that corresponding arc weight as negative.

Then we go back and identify what are called negative length cycles from this. For example now the movement we put this we can now go back and see 1 - 2 - 4 - 1 that is 1 to 22 to 4 4 to 1 is a cycle. We start from 1 we go through some edges and we come back to 1 so 1 to 2, 2 to 4, 4 to 1 is a cycle. Now what is the total weight associated with this cycle 10 minus 15 minus 5 which is equal to minus 10 minus 20 plus 10 is minus 10, so we have identified a negative cycle. Now we can identify some other negative cycles 23 4 2 is a negative cycle 34 83 is a negative cycle 3 to 4, 4 to 8, 8 to 3 is minus 50 plus 20, 2 342 is a negative cycle minus 45 plus 20 is minus 25,4 8 764 is a negative cycle. Similarly, 456 4 is also a negative cycle in fact 1 451 is also a negative cycle, all these are negative cycles.

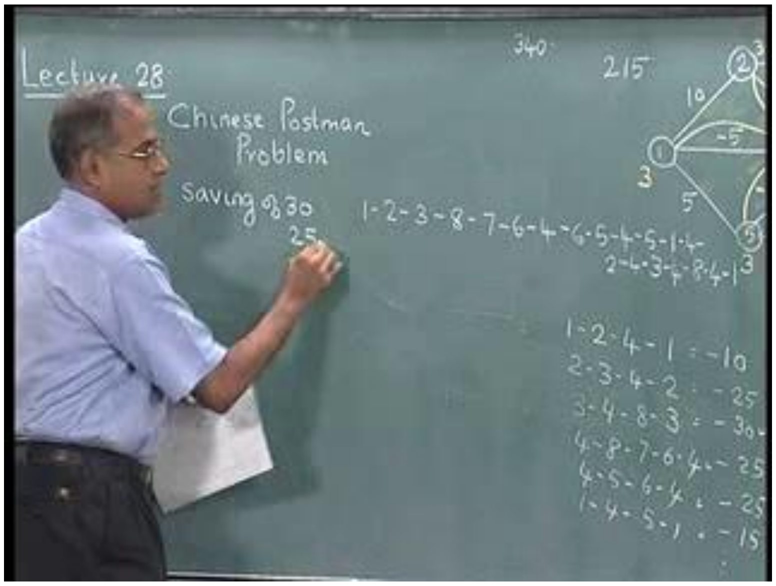
Let us find out the length of each of them, we can even identify more negative cycles than what we have shown here, but then let us look at we can take any one of the negative cycles. So we can right now evaluate some of them and take the one that is most negative. So if we do 234 2 we get minus 45 plus 20, which is minus 25. 3483 is minus 30, 4876 4 is minus 10 plus 5 minus 25, 4564 is minus 45 plus 20 minus 25, 1 451 is minus 15 and so on. Now we can take any one of them to begin with or we can take the most negative cycle, which is the one with minus 30 and then perform some operations. So what we do now, take the negative cycle 3 to 4, 4 to 8, 8 to 3 and simply change the duplications.

(Refer Slide Time26:53 min)



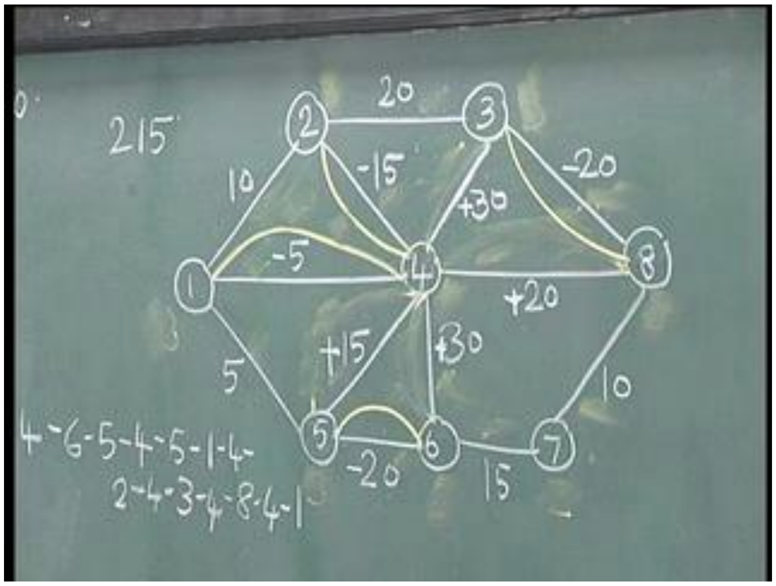
For example what we are going to do now is 3 to 4, 4 to 8, 8 to 3 here there is duplications, so bring it back remove the duplication. Here there is duplication so remove this duplication. Now here there is no duplication, now make duplication and then wherever there is duplication you put a negative, wherever there is no duplication you put a positive. So we considered this 3 - 4 - 8 - 3 and we have a saving of 30 by considering this.

(Refer Slide Time: 27:21 min)



Now go back and see whether we can get one more negative cycle out of this. We said 456 4 is a negative cycle 4 to 55 to 66 to 4 is a negative cycle minus 45 plus 20.So let us consider this to begin with minus 45 plus 20 so we get a saving of 25 out of this.

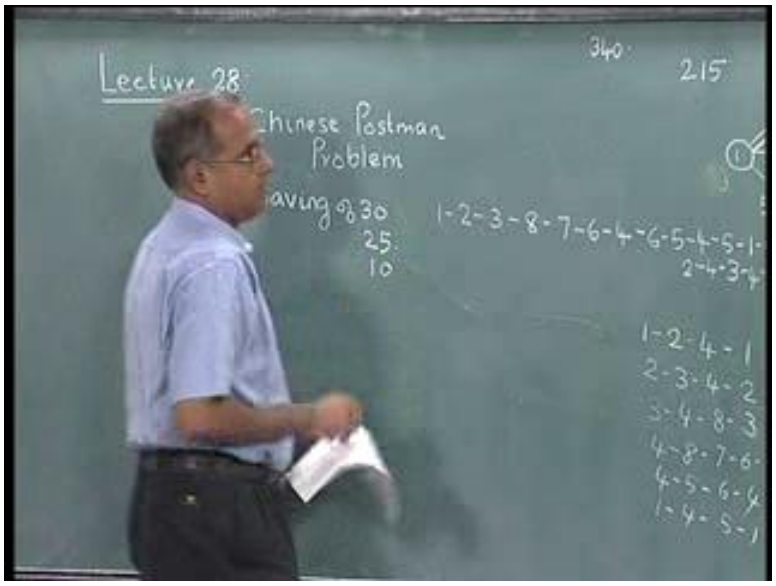
(Refer Slide Time: 28:06 min)



So wherever there is duplication, change it so, this minus 15 is going to become a plus 15.

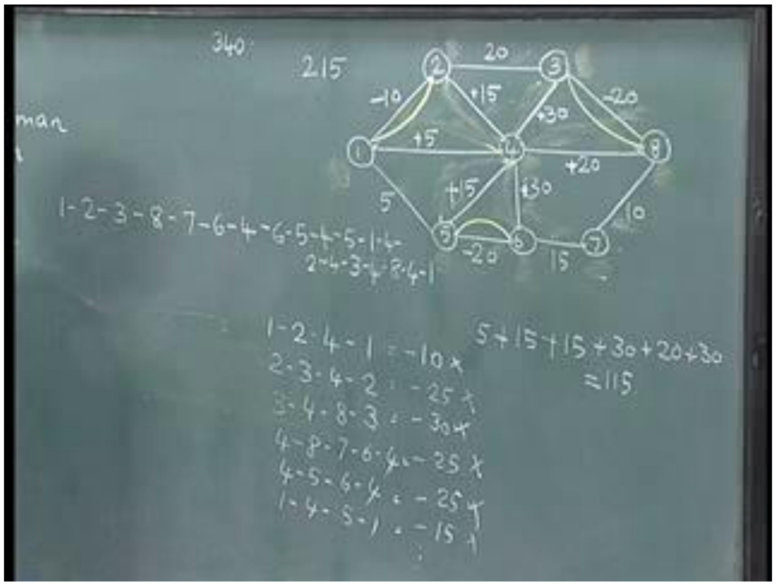
This minus 30 goes this will become a plus 30, but this 20 will now be duplicated and we will get a minus 20 here and we have made a saving of another 25.

(Refer Slide Time: 28:50 min)



Now, let us see if there is any other negative length cycle. And right now we do not need these, these have also changed we do not need these degrees now look at 1 241 is a negative cycle. So there is a saving of 10, now 2 to 4 is presently duplicated so a change this so this becomes positive.

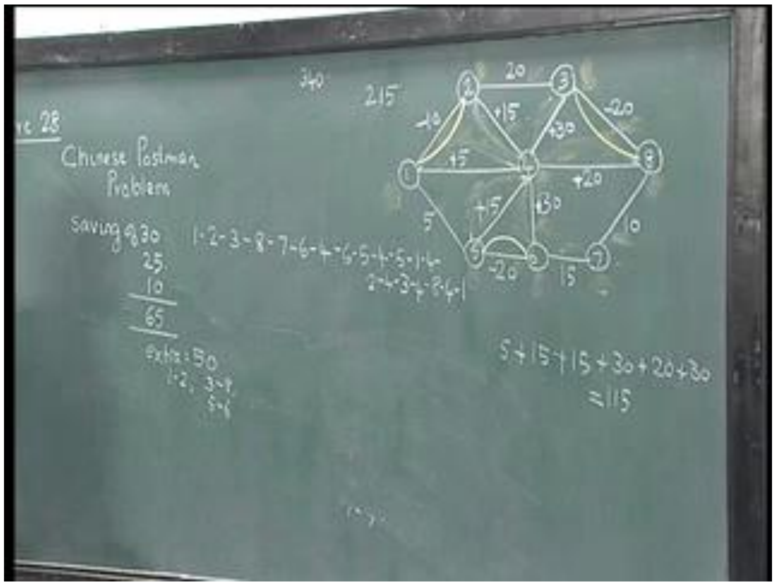
(Refer Slide Time: 29:00 min)



Now this changes, so this becomes positive while this arc gets duplicated and this becomes negative. Now we go back and see whether we have any more negative length cycles. For example 2 3 4 2,2 to 33 to 4 4 to 2 is now not a negative length circle, so we do not consider this 4 8764 is not a negative cycle so we do not consider this 1 4 51 is not a negative cycle so we do not consider this.

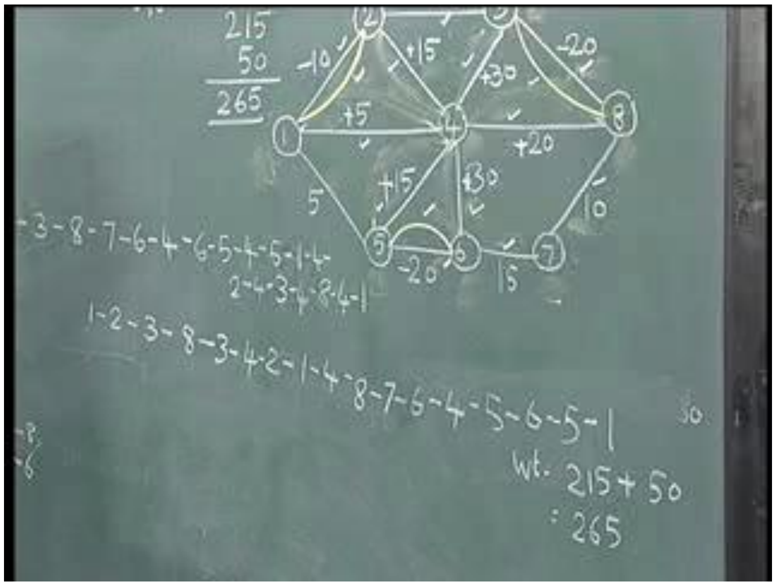
Now we need to check whether we can have any other negative cycle. We actually realize that we do not have any other negative cycle in this. Any other cycle that we evaluate is now strictly positive length so the algorithm actually stops, then we cannot find out any more negative length cycle so we need not find any more negative length cycle.

(Refer Slide Time: 30:18 min)



So the savings that we have had right now is 30 plus 25,55 plus 10 is 65 which means earlier we had an extra 115. Now, we have a saving of 65 which means we have an extra 50 equal and these extras are nothing, but the lengths of duplicated edges, which is 10 plus 20 plus 20. So this is the duplicated edges are 1 to 23 to 8 and 5 to 6.

(Refer Slide Time: 31:03 min)



So the Eulerian will now have a length of the original length of 215 plus the extra 50 now, which are 265. Now let us go back and redo the Eulerian. So we start with 1 to 2 2 to 3 so let us tick this we will start with 1 to 2 and then we do 2 to 3 and then we do 3 to 8 then we come back 8 to 33 to 4, then 4 to 2 and 2 to 1 and so we are back, then we do 1 to 4, which is here. Then we do 4 to 8, then we come back from 8 to 7 7 to 66 to 4will have4 to 5 back here 5 to 6 again and back 6 to 5, which is here and then 5 to 1. So, now this is the Eulerian with weight equal to 215, which is the original one plus 50 that is duplicated with weight equal to 265.

Since now again you observe here, that all those that have been duplicated are shown in yellow and wherever there is a duplication the weight is shown with a negative, so we are not able to find a negative length cycle and therefore the algorithm terminates here by saying for this network it is enough if we duplicate this this and this we can minimum duplicated weight of 50 and the Eulerian now has 215 plus 50 which is 265.

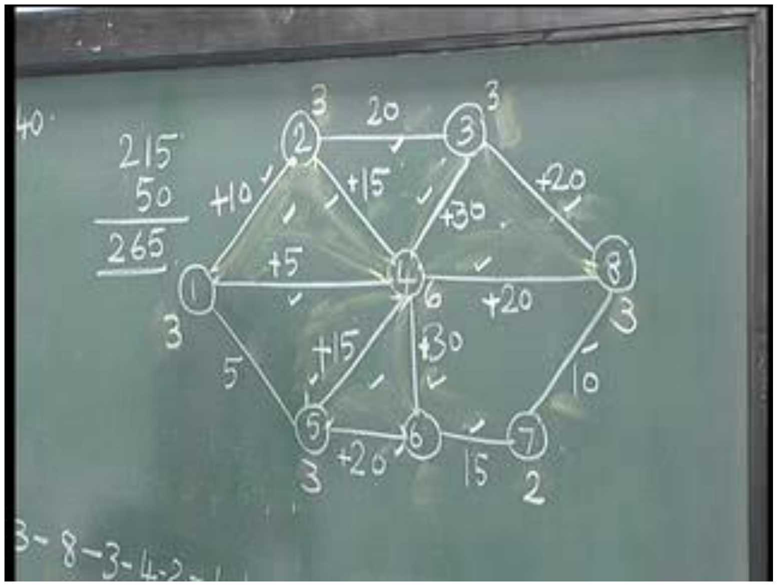
So this is the first algorithm that we have seen to solve the Chinese postman problem. This algorithm is very nice in efficiency except that in this algorithm we now have to find out the starting solution. Of course, we need to find out given a start, this is more like an improvement algorithm. We have a starting solution and for the starting solution we duplicated the edges and then we identified negative length cycles and then changed the duplicated edges so that the weight keeps reducing and therefore we eventually minimize the total weight that has been added to this later.

The two aspects which are little difficult is, one of which is to find out an initial feasible solution, which could be arbitrary. So we start with the initial solution. The second is we have to evaluate all the cycles to see whether we have a negative length cycle. Now these are the two slightly more involved procedures in the first algorithm.

Now the next is we can have a kind of a construction algorithm not an improvement algorithm. The first one is an improvement algorithm but it is a very intuitive algorithm to understand the principle because they are effectively trying to minimize the additional weight that goes in. So we have an arbitrary solution with an arbitrarily large weight as we shown here and then the idea that a negative length cycle gives us a way by which we can reduce the extra weight that goes into this network. thus, it is a very intuitive algorithm but the two aspects of it which are little cumbersome, which is to get a feasible solution and then to have to find out what are all these cycles and then evaluate every cycle to check whether it is a negative length cycle.

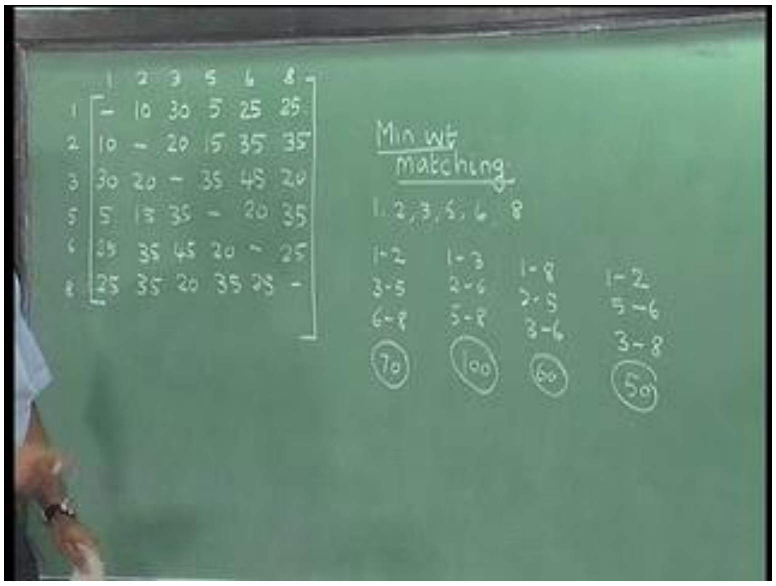
The next question is can we have a construction algorithm to solve this problem which is different from the one that we have right now seen. Now to do that, let us go back to the same network. Now let us for a moment leave out the duplicated edges here.

(Refer Slide Time: 35:26 min)



This is the original network with positive numbers here. We also saw that this is not Eulerian because the degree of all the vertices are not even. We said this has a degree of 2, this has a degree of 6, while the rest of them have degree of 3 and have odd degree. So this has a degree of 3, this has a degree of 3, 3, 3, 3 and so on. So the first thing that we do is we also know that the number of vertices that have odd degree is an even number so we first get a sub matrix out of this which are all the vertices that have odd degree so the vertices that have odd degree are 1 2 3 5 6 and 8 so the odd degree vertices are 1 2 3 5 6 and 8.

(Refer Slide Time: 36:17 min)



$$\begin{array}{c|cccccc} & 1 & 2 & 3 & 5 & 6 & 8 \\ \hline 1 & - & 10 & 30 & 5 & 25 & 25 \\ 2 & 10 & - & 20 & 15 & 35 & 35 \\ 3 & 30 & 20 & - & 35 & 45 & 20 \\ 5 & 5 & 15 & 35 & - & 20 & 35 \\ 6 & 25 & 35 & 45 & 20 & - & 25 \\ 8 & 25 & 35 & 20 & 35 & 25 & - \end{array}$$

Min wt matching

1, 2, 3, 5, 6, 8

1-2	1-3	1-8	1-2
3-5	2-6	2-5	5-6
6-8	5-8	3-6	3-8
$\boxed{70}$	$\boxed{100}$	$\boxed{60}$	$\boxed{50}$

We construct a matrix which has 1 2 3 5 6 and 8. Now we find out a distance matrix that connects these vertices with these vertices such that we find the shortest distance. For example, 1 to 1 is dash 1 to 2 is 101 to 3 is not directly connected but we can always use the Dijkstra's algorithm to find the shortest distance from 1 to 3. In the Chinese postman problem we can conveniently assume that all these arc weights are greater than or equal to zero because in this principle they represent distances that the postman has to travel. Since all these weights are greater than or equal to zero Dijkstra's algorithm can be used to get the shortest path or shortest distance from 1 to 3, otherwise the algorithm has to be modified if you have negative lengths coming in here.

Remember the original graph will have all distances or weights greater than or equal to 0, only when we are considering duplications we converted some of them in negative. The negatives that we used were only helpful to us in identifying negative length cycles and they are not the correct representations of the edge weights.

So all the edge weights are greater than or equal to zero we can use a Dijkstra's algorithm to get it, so 1 to 3 the shortest distance will be 30 which we keep here as 30. 1 to 5 is again is 5 from this graph,1 to 6 is not directly connected so 1 to 6 is 5 plus 20 = 25. hence, from the Dijkstra's algorithm we will drain as that is the smallest otherwise you could do 1 to 2, 2 to 4, 4 to 6 or 1 to

4, 4 to 6 and so on, but 5 plus 20 = 25 is the smallest and like this we can complete this graph. Again 1 to 8 we can find the shortest path, which is 5 plus 20 = 25 and so on. So wherever i to j is not directly connected you can solve the Dijkstra's algorithm to get the shortest distance from i to j. So when that is done, then the matrix will look like this however 20 15 35 35 35 45 20. Now this represents the shortest distance so obviously to construct this matrix it will take some time, but then we realize that the number of vertices with odd degree is even and it is utmost n, which is the number of vertices in this graph its utmost n. Therefore, this will have a size of n by n this is symmetric therefore it is enough to do superscript n C_2 computations. And each of this superscript n C_2 computations is a single pass of the Dijkstra's algorithm and therefore the effort take in to create this is still polynomial.

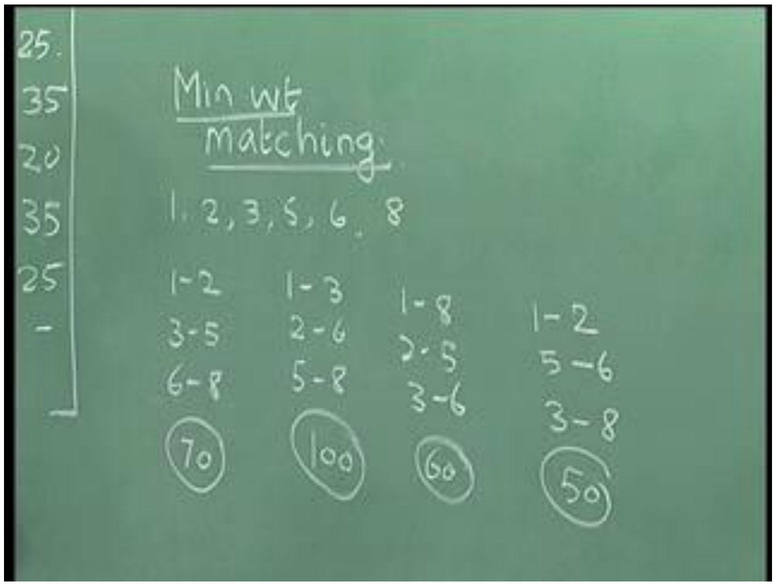
So it is little cumbersome if one does it by hand, but one can always use a software or computer program or solver to get the shortest path from this i to j. This can be constructed. Once this is constructed we now try to again find out a minimum weighted matching. Now we already seen this minimum weighted matching in the earlier lecture, when we looked at a heuristic solution to the TSP that involved minimum weighted matching. Now we know that this is always in even number because this represents the number of vertices in the graph which has an odd degree. Therefore this is always an even number so this is an even number. Here we have 6 - 1 2 3 5 6 and 8 we have six vertices so now from six vertices we get a matching of three edges. For example we are looking at 12 35 68. Now 1 to 2, 3 to 5, 6 to 8 is a matching, because it comprises of three edges 1 to 2, 3 to 5 and 6 to 8 it comprises of three edges which covers all the six vertices, which means these three edges do not share a common vertex, so this is the matching a matching that comes out of these vertices.

Another example of a matching could be 1-3, 2-6, 5-8 - this is another example of matrix. The third example could be 1-8, 2-5, 3-6 and so on. So we could have several matchings like this. Now we try to find out a weight associated with each of these matchings. Now this matching will have a weight 1 to 2, 3 to 5 -10 plus 35 = 45 plus 6 to 8 45 plus 5 = 50 plus 20 = 70. Now this matching will have 1 to 3 is 30 plus 35 = 65, 65 plus 35 is 100. This matching will have 25 plus 15 = 40 plus 2 to 6 we could think of another matching. For example, 1 to 25 to 6 and 3 to 8 as another matching that comes out that would give us 10, 1 to 2 is 10, 5 to 6 is 10 plus 35 is 45 to 6 is 10 plus 20 = 30, 30 plus 20 = 50 and so on.

Here I have not shown exhaustively the entire possible matches but there are several possible things, but then the problem is to find out the minimum weighted matching. Now there are six vertices, each of these is a matching there are many more that are not shown here. Now each one has described earlier is a matching because it has all the six vertices, three edges with no edge having any common vertex.

So each of these is the matching and there are several more so there are algorithms with which we can find out the minimum weighted matching given a network in this. This is somewhat similar to the assignment problem but it is not exactly the assignment problem because an important condition for a matching is that this has to be 21 to 23 to 56 to 8. For example, 1- 2, 3-5 6-8 cannot be something like a matching, so this is somewhat closer to the assignment problem.

(Refer Slide Time45:30 min)



25
35
20
35
25
-

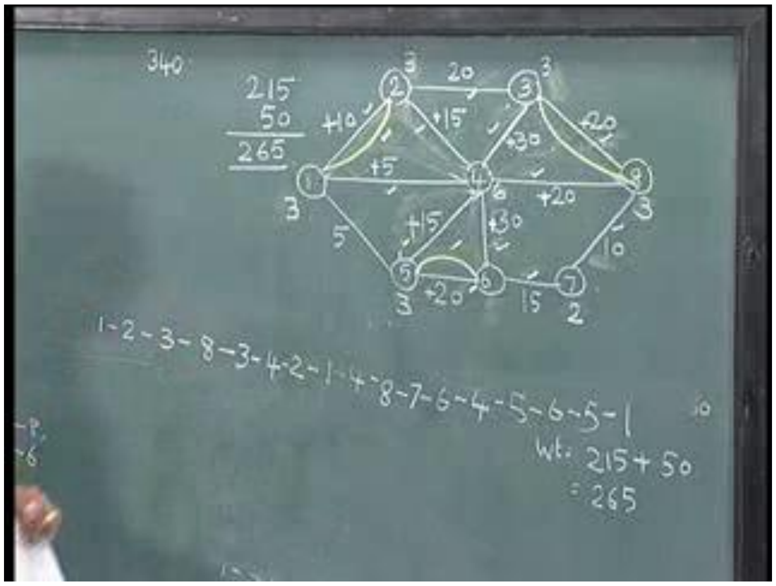
Min wt Matching

1, 2, 3, 5, 6, 8

1-2	1-3	1-8	1-2
3-5	2-6	2-5	5-6
6-8	5-8	3-6	3-8
70	100	60	50

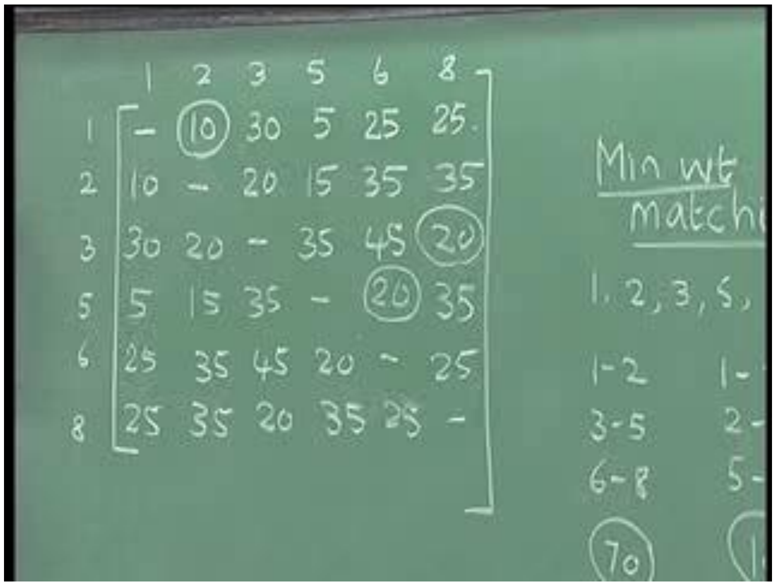
But it is actually different from that so some algorithms are available to find out the minimum weighted matching given a graph like this. And we can show that the best one is actually this with weight equal to 50, now when the best one has weight equal to 50 all we need to do is to duplicate that.

(Refer Slide Time45:52 min)



So we duplicate 1 to 23 to 8 and 5 to 6 and when we do that and we get the Eulerian. So once we get the Eulerian, this is the Eulerian already we have seen with the earlier algorithm. So this is the Eulerian so the extra weight that we have added is actually from here so we get 215 plus 50 which is 265.

(Refer Slide Time 46:32 min)



$$\begin{array}{c|cccccc} & 1 & 2 & 3 & 5 & 6 & 8 \\ \hline 1 & - & 10 & 30 & 5 & 25 & 25 \\ 2 & 10 & - & 20 & 15 & 35 & 35 \\ 3 & 30 & 20 & - & 35 & 45 & 20 \\ 5 & 5 & 15 & 35 & - & 20 & 35 \\ 6 & 25 & 35 & 45 & 20 & - & 25 \\ 8 & 25 & 35 & 20 & 35 & 25 & - \end{array}$$

Min wt matchi

1, 2, 3, 5,

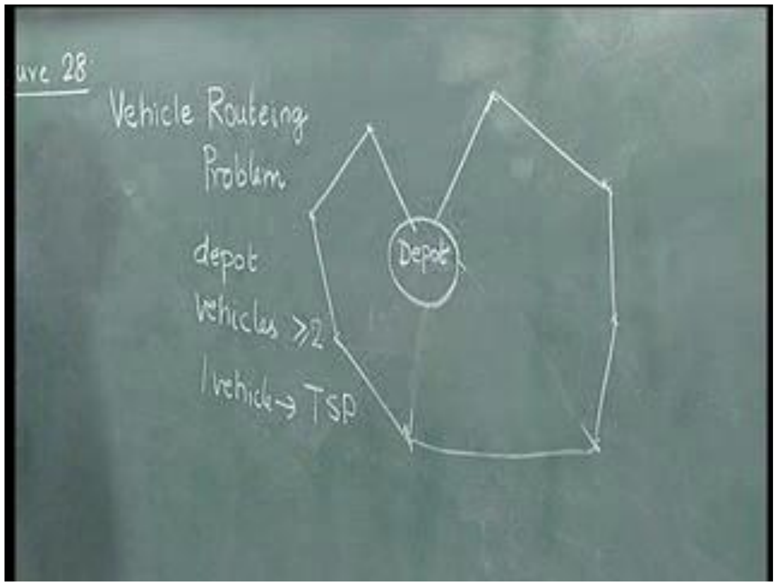
1-2	1-
3-5	2-
6-8	5-
70	

One should also keep in mind that in this example 1 to 2, 5 to 6, 3 to 8 happens to be direct edges that are available. Just in case something like 1to 3 happens to be in the solution then it means we add 1 to 2 and 2 to 3 because 1 to 3 is not at a shortest distance which means 1 to 2 and 2 to 3. So depending on what we get from this we have to go back and see if it is a direct edge we will have a direct duplication. If it is a path which has more than one edge then all those edges will have to be duplicated.

Now for this, once we solve the minimum weighted matching problem out of all the vertices that have odd degree and the superimpose and the minimum weight matching on to the Eulerian then we get the optimal solution to be Chinese postman problem. So what we have seen are two algorithms. The first one that we saw was an improvement algorithm. We started with an arbitrary Eulerian circuit by duplicating edges arbitrarily. And then wherever there was duplication we added a negative and then we tried to find out negative length arcs. And wherever we found the negative length arc we switched the positives became negatives the negatives became positive. And the net was a gain and then we did this such that there are no more negative arcs in this solution. Now that is a very intuitive algorithm which is based on the principle that we want to minimize the extra distance.

The other one is the construction algorithm where we identify those vertices that have hot degree create a matrix of distances obtained by solving a shortest path problem and from that we solve a minimum weighted matching problem and then get this matching and superimpose this matching on to the given Eulerian so that we now find out the extra length or extra distance this person has to travel. So this brings to the end of our discussion on the Chinese postman problem.

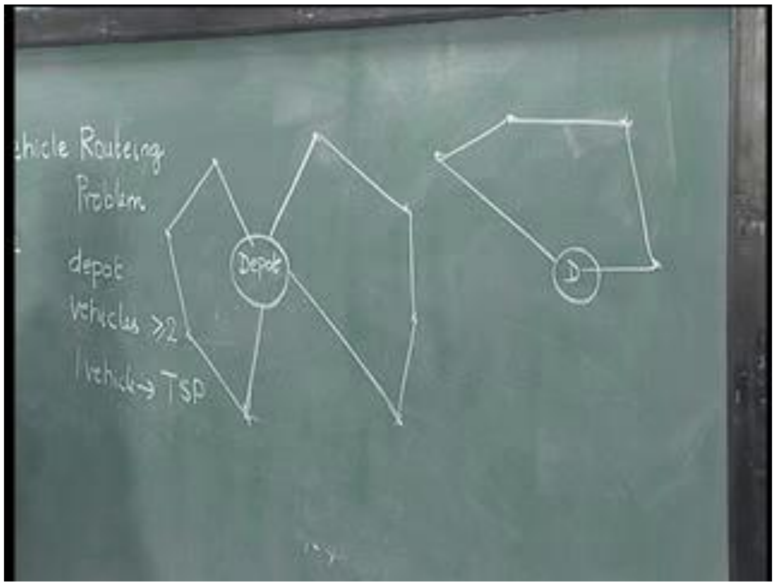
(Refer Slide Time48:50 min)



And then we look at yet another problem which is quite similar to the TSP and that is called the vehicle routing problem. So let us describe the vehicle routing problem first, now vehicle routing problems are very common and they are very well known that the vehicle routing problem happens like this. Now you may consider a central point, which could be a school where there are several school buses; these buses go and pick up people from different places. For example, there could be a bus that goes here and comes back, could be another bus that goes here picks a people and comes back. This could be a supermarket, where there are vehicles that deliver goods to people and so on. So wherever there is a pickup and delivery you have what is called a vehicle routing problem.

Now for the standard rotation the central place from which the vehicles go is called a depot. A vehicle routing problem ordinarily means that there is a depot; there are at least two vehicles now each vehicle will have its own specified route, so there is a depot and there are vehicles. Usually the vehicle is greater than or equal to two the reason been if there is only one vehicle, then this vehicle will come here it will go here and then it will come back, then it becomes a Travelling Salesman Problem. So when we have one vehicle the problem becomes a TSP, then problem becomes a vehicle routing problem which is called a vehicle routing problem, when there are more than one vehicle at least two or more vehicles.

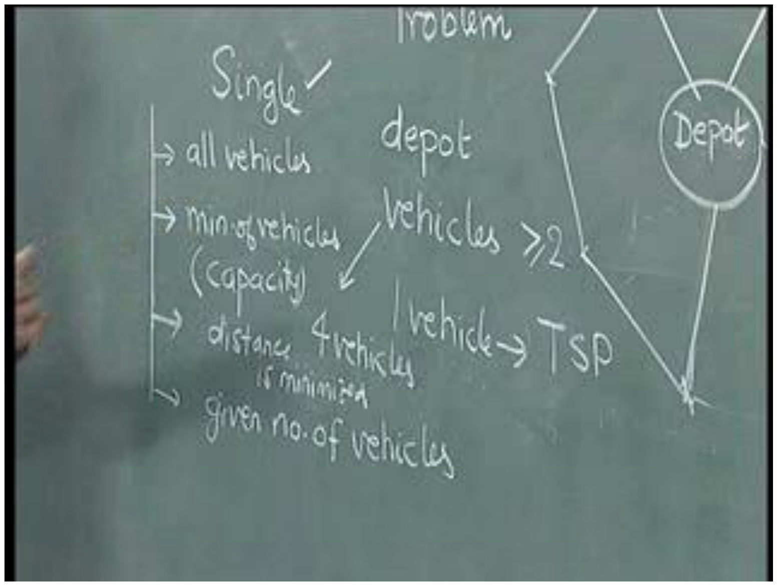
(Refer Slide Time51:38 min)



Now there are several things in a vehicle routing problem, there could be a single depot vehicle routing problem or there could be multiple depot vehicle routing problem. So you could have multiple depots. Now there could be another depot and there are customers like this whereas the vehicle could start from this depot meet the requirements of these customers and come back.

Now in this lecture series we will look only at the single depot vehicle routing problem and we are not going to look at the multiple depot vehicle routing problem. There are a couple of other issues. Now there is a depot, there are a set of vehicles and there are normally three problems that are associated with vehicle routing.

(Refer Slide Time52:14 min)



Now let us assume that we have four vehicles, now there could be a case where I have depot I have different sets of customers who say have to be picked up, then we would say that I wish to use all the vehicles. So there is a problem where you could say I want to use all the vehicles and send these vehicles to pick up these people. And then we want to find out the route in such a manner that the total distance traveled is minimized.

Now all vehicles problem is also important because particularly say in office pickup or school pickup kind of problems where say the office is going to begin at 9 o' clock so the more vehicles we send it becomes easier for people to come. So the time that is spent by the person who is farthest away from the office would be minimized if we have more and more vehicles. But at times we get into a problem where we have certain number of vehicles but we wish to use all the vehicles so there is a problem of this type.

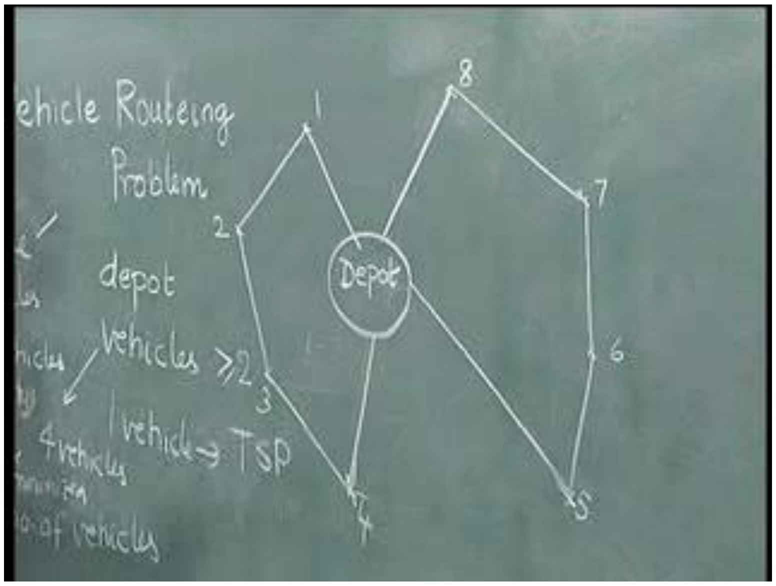
Then there is second type of problem where we want to use the minimum number of vehicles that are required. Now this comes in from the moving, we say minimum number of vehicles is required we could say two or we could say that there is a vehicle capacity and there is an individual requirement at each of these customers. So rather than describe the pickup problem we may assume that this problem is like a super market where these customers have ordered something from the supermarket, therefore things has to be dropped or things has to be delivered in the house of each of these customers. So each van or vehicle that goes from this depot has a certain capacity and there is a requirement. So based on the capacity of the vehicle and the requirement we can find out the minimum number of vehicles that are actually required and then we can use that. So we have problems of this type. Ordinarily when we look at this problem we assume capacity.

There is a third one where we would say use minimum number of vehicles such that the total distance is minimized in the uncapacitated case. We could even do this total distance is minimized. Now the third is used a given number of vehicles and we have four vehicles but today I would like to use only three vehicles so we may say now give a solution where only three vehicles are being used. Therefore, we have these kinds of categories there is a capacity so we would say we want to use all vehicles in one type of a problem.

Use minimum number of vehicles is another type of a problem. Use minimum number of vehicles such that total distance is minimized is another type of problem and use given number of vehicles is also another problem. There are vehicle routing problems where there is capacity of the vehicle and demand in each of these and there are problems where we do not explicitly consider the capacity.

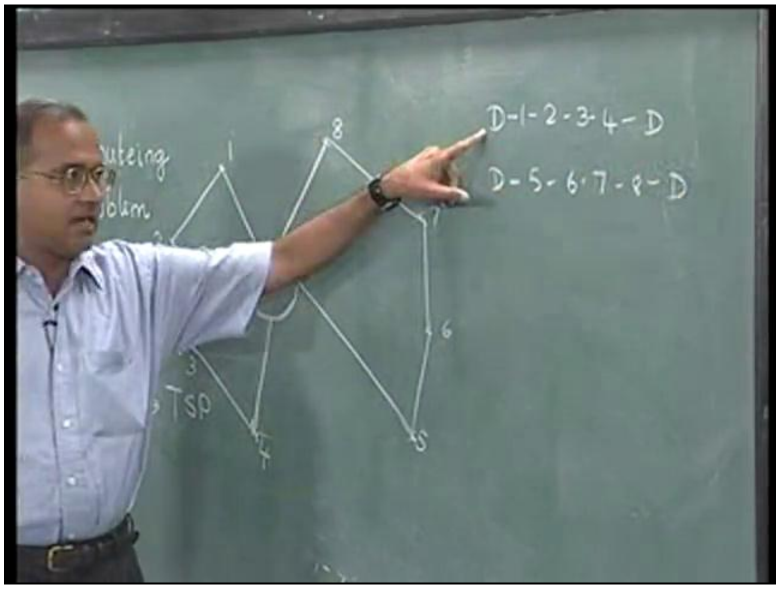
Now there are other cases in the vehicle routing. there are things like pick up and drop problem. for example a vehicle will go, it may drop something, it may pick up something and then it can move. typically mail motor problems sometimes do that. the mail motor van will go to the post office, it will drop a certain set of letters that have come into the post office and then simultaneously it will also pick up letters that come from the post office which have to be taken to the central office. Now we do not consider such problems in our discussion right now. So we are going to look at only these types of problems; a single depot vehicle routing problem and we would try to see as much of these as we can consider.

(Refer Slide Time: 56:42 min)



Now coming back again to the single depot vehicle routing problem if we assume a single depot and then say we call these people as customers 56 78 and if we do not consider capacities and if we consider two vehicles then we might say the first vehicle will go from the depot to 12 3 4 and back to the depot, the second vehicle will go from depot 56 78 and back to the depot if this is a feasible solution to the vehicle routing problem.

(Refer Slide Time 57:16 min)



This is usually described as D 1 234 D and D 56 78 D where the D simply means that the vehicles starts from a depot completes this round and comes back to this depot. Now there are two aspects to this feasible solution. Now if we look at this solution we now first say that this goes from this 123 4 this covers 5 67 8. Now that 1 2 34 is covered in the order 1 234 this is covered in the order 5678. So there is a grouping problem, there is routing problem and then we get the vehicle routing problem. We will see more aspects of the VRP in a next lecture