

Advanced Operations Research
Prof. G. Srinivasan
Dept of Management Studies
Indian Institute of Technology, Madras

Lecture - 27
Heuristics for TSP (Continued)

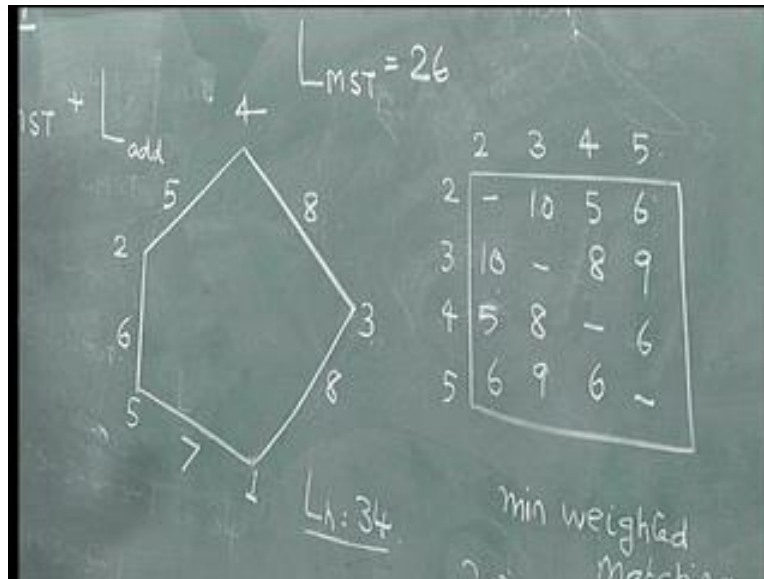
In this lecture, we continue our discussion on heuristics to solve the Traveling Salesmen Problem. In the previous lecture we were looking at the heuristics based on matching.

(Refer Slide Time: 00:41)



We start with the minimum spanning tree for the data that is given in the problem. We consider the same data here and the minimum spanning tree which we have already drawn like this.

(Refer Slide Time: 00:44)



The minimum spanning tree has these four edges 2 to 4, 2 to 5, 3 to 4 and 1 to 5 with length equal to 26, 8 plus 5 equal to 13 plus 6 equal to 19 plus 7 equal to 26. We also mentioned in the previous lecture that in the minimum spanning tree, the number of vertices that have odd degree will be even. This is a vertex that has odd degree. These three vertices have even degree equal to 2. There are two edges, here, this is a vertex that has odd degree equal to 1.

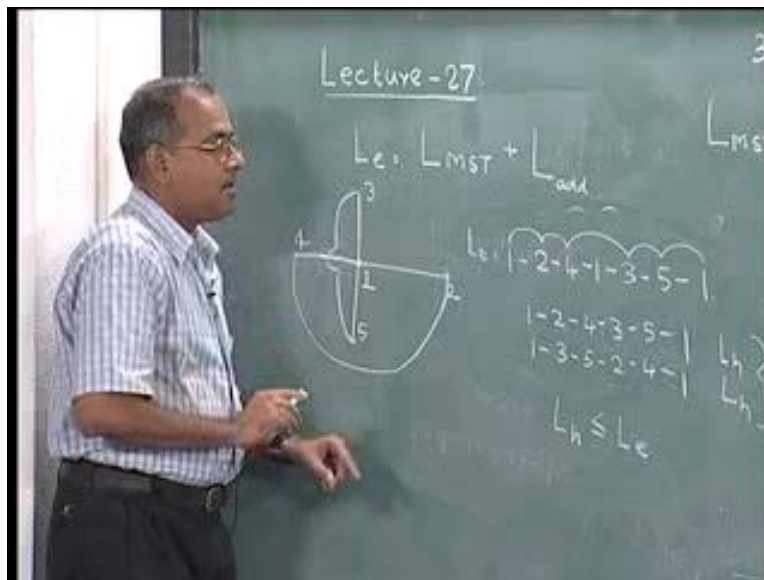
The number of vertices with odd degree will have to be even. We even mentioned in the previous lecture that this is an important result. It is a very easy result to prove because each edge is incident to two vertices. Therefore the sum of the degrees of every vertex when added is an even number, because each edge contributes to the degree of two vertices. So, sum of the degrees of the vertices is an even number and because the sum of the degrees of the vertices is an even number, the number of vertices that have odd degree is even. In this case there are two vertices that have odd degree. We could have a case where four vertices have odd degree because we are considering a 5 by 5 problem.

If we have only two vertices that have odd degree all that we do is, go back to that matrix and add that vertex. For example, 1 and 3 are the vertices that have odd degree, so add the edge 1 3 into the minimum spanning tree to get a Hamiltonian circuit or to get a feasible solution to the Traveling Salesmen Problem. If we do that, now, 1 3 has weight equal to 8. There is only 1

feasible solution here, which is a tour feasible to the Traveling Salesmen Problem. This will have a length equal to 34, so this method for this particular problem gives a feasible solution with length equal to 34.

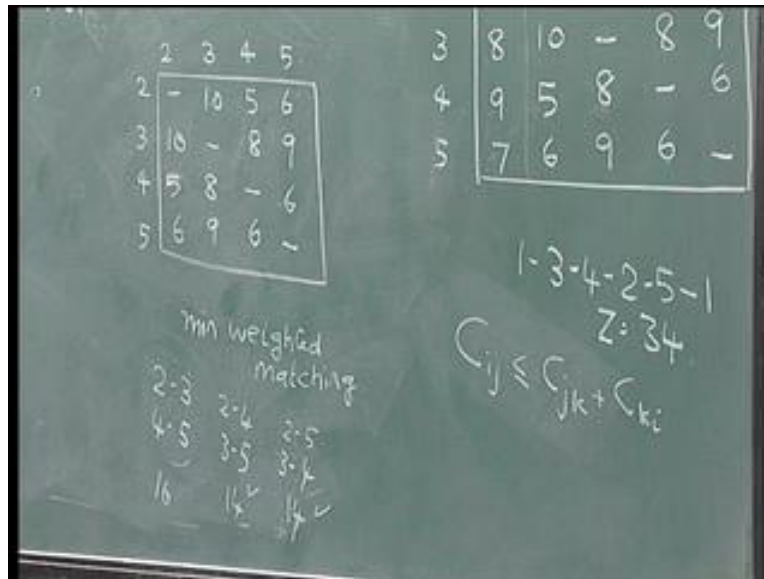
If the number of vertices that have odd degree is 2, then we add only one edge into the minimum spanning tree to complete it. Therefore we have only one solution that is given to this problem. For this instance, that solution gives a length equal to 34. We may have the case where there are more than two vertices and even number. It would be four vertices that have odd degree or for a larger size problem, we could have six vertices or eight vertices that have odd degree. Let us see what happens in such a case.

(Refer Slide Time: 03:52)



Let us assume for a moment that the minimum spanning tree is like this. In our example, it is not so, but let us assume for a moment the minimum spanning tree is like this. This is a case where there are four vertices that have an odd degree. When there are four vertices that have an odd degree what we do is, we go back to the original matrix and then pull out the corresponding 4 by 4 from the original matrix. The original matrix has vertices 1 2 3 4 and 5 and 1 2 3 4 and 5. We pull out the corresponding 4 by 4 matrix. If this were the minimum spanning tree then the four vertices that we are interested are 2 3 4 and 5 that have odd degree.

(Refer Slide Time: 04:47)



We pull out a corresponding 4 by 4 matrix out of this. What we do is we try to find out what is called the minimum weighted matching. This number is always an even number. It is either 4 or 6 or whatever depending on the size of the matrix. In this case it is a 4 by 4. When we have a 4 by 4 matrix, what I mean is by a matching like this. Let us look at the edge 2 3 and the edge 4 5. There are four vertices, so what we try to do is with two edges, such that these two edges do not share a common vertex. We try to exhaustively look at all the vertices at a typical example of a matching could be 2 3 and 4 5.

There are two edges, the four vertices divided by two edges and all the vertices are covered. Another matching could be 2 4 3 5 and the third matching could be 2 5 3 4. At this case, there are only three possibilities. If it were a 6 by 6, then we will have more possibilities. Every matching will have three edges that will cover all the six vertices. Here, we have three possibilities. Now we find out the weight associated with each of them. The weight associated with this is 2 3 plus 4 5 which is 10 plus 6 which is 16.

The weight associated with this is 2 4 and 3 5, which is 8 plus 6 equal to 14. The weight associated with this is 2 5 and 3 4, which is again 8 plus 6 which is 14. Out of these three, we find out the best one which can be either this 14 or the other 14. Let us say we take this 14 and then we superimpose this matching on to the minimum spanning tree. We would get something like

this 2 to 4 and 3 to 5 on this. Once we superimpose this matching on to the minimum spanning tree, we can try and get an Eulerian circuit which means, starting from any vertex we will be able to go to every edge once and only once and come back to the starting vertex.

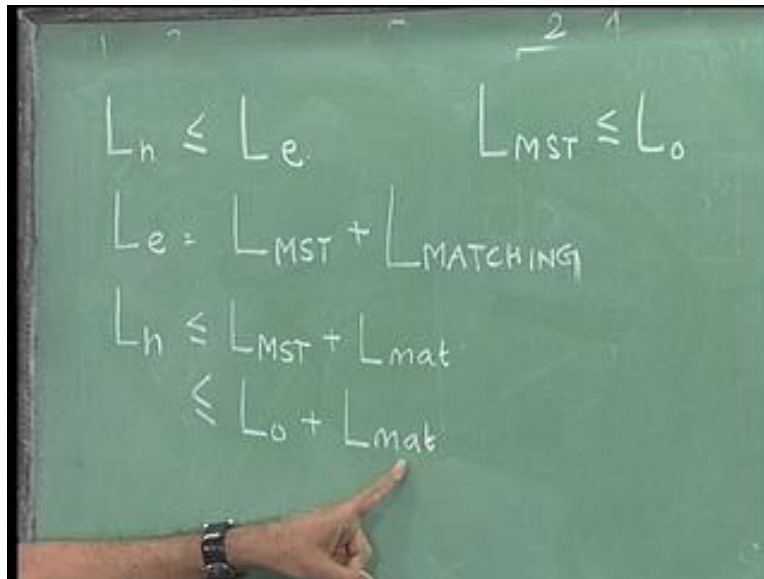
Such an Eulerian will look like this. For example, it will look like 1 to 2, 2 to 4, 4 to 1, 1 to 3, 3 to 5 and 5 to 1. We will have an Eulerian circuit like this and as we did in the earlier case, we can go back and get feasible solutions to the Traveling Salesmen Problem. They could be 1 2 4 3 5 1; 1 to 2, 2 to 4, 4 to 3, 3 to 5, 5 to 1 or it could be 1 3 5 2 4 1; 1 to 3, 3 to 5, 1 to 3, 3 to 5. Come back so we have 5 to 2, 2 to 4, 4 to 1 or it could be 1 4 3 5 2 1. We have only two cases, 1 to 2, 2 to 4, 4 to 3, and 3 to 5, 5 to 1.

We have the case starting with this one, 1 to 3, 3 5, 5 to 2, 2 to 4, 4 to 1. These are the possibilities that we have and we can find out the L_h associated with each one of them and choose the best. When we had in our spanning tree only two vertices that have odd degree, we simply add that edge and proceed. When we have more than two vertices that have odd degree, then we pull out the corresponding matrix from this matrix and then we find out what is called the minimum weighted matching. Superimpose the minimum weighted matching on to the spanning tree, obtain an Eulerian circuit, from which obtain feasible solutions to the Traveling Salesmen Problem. Then find out the best out of these and declare them or give them as the heuristic solution to the problem.

How good is this algorithm? Can we derive an expression like what we derived for the twice around the tree heuristic?

It is possible to have a derivation for this which we will look at right now. Let us try and derive that expression for this here. Let us call the best out of these as L_h and if triangle inequality holds, like we discussed in the earlier case, if triangle inequality holds we can show that, for example, this has been brought out of this, this is 1 2, 2 4, 4 3 3 5 5 1. If triangle inequality holds, 4 1 plus 1 3 is greater than or equal to 4 3. Therefore this length is greater than or equal to this length. If we call this length as L_e (length of the Eulerian), then L_e is greater than or equal to every one of these L_h . Therefore L_e is greater than or equal to the best out of these, which we are going to call as L_h . We have this simple relation, L_h is less than or equal to L_e which we can show. If triangle inequality holds where L_h is the best among these that we have taken.

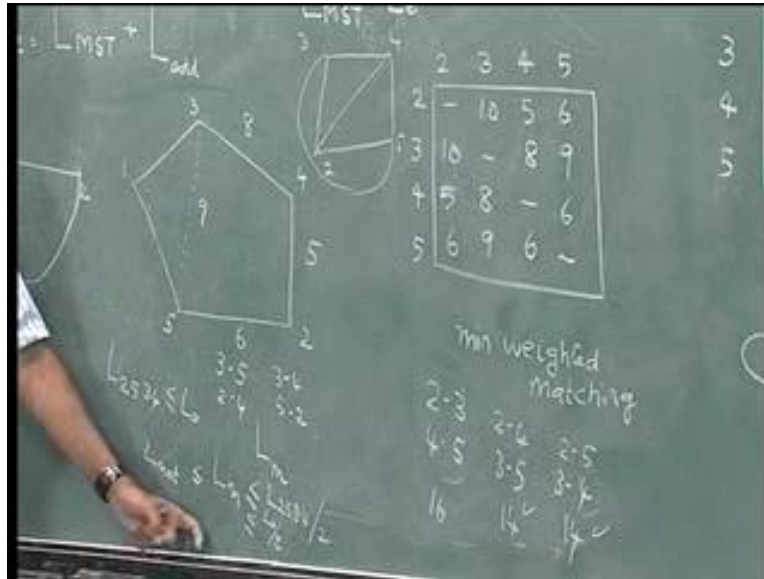
(Refer Slide Time: 11:14)


$$L_h \leq L_e \quad L_{MST} \leq L_0$$
$$L_e = L_{MST} + L_{MATCHING}$$
$$L_h \leq L_{MST} + L_{mat}$$
$$\leq L_0 + L_{mat}$$

We write the first result, that L_h is less than or equal to L_e . Now, what is this L_e ? This L_e is nothing but the length of the minimum spanning tree plus the length of the matching, that we have added on to this, so L_h is less than or equal to L_e . We have L_e equal to L_{MST} , length of the minimum spanning tree plus length of the matching that we have added to the minimum spanning tree. Therefore L_h is less than or equal to L_{MST} plus $L_{MATCHING}$.

We already know that L_{MST} is less than or equal to L_0 . So L_h is less than or equal to L_0 plus $L_{MATCHING}$. We have already written L_{MST} with reference to L_0 . We need to write this $L_{MATCHING}$ with reference to L_0 that we will be able to derive L_h by L_0 . What is the relationship between this $L_{MATCHING}$ and L_0 ?

(Refer Slide Time: 12:43)



Let us look at the case where we have four vertices having odd degree. Let us say that we have some unknown L_0 . For this problem, there is an unknown L_0 which have five edges here. We do not know exactly what L_0 is but one thing that we know is that all the five vertices are actually present in L_0 . For the moment let us assume that the L_0 since we know the L_0 . Let us assume that L_0 is 1 3 4 2 5. We can actually derive this by assuming that we do not know L_0 just to make the derivation simpler, I am assuming that we know the L_0 and I am going to show this to you. This is 2 3 4 5 that are here.

What I do is I simply take 2 3 4 5. What I am trying to say, it is always possible from L_0 to remove edges and bring it such that we have a smaller thing corresponding to these vertices that we have here. Now, again based on triangle inequality, this plus this is greater than this. Therefore the L , 3-4 plus 4-2 plus 2-5 plus 3-5 is now less than or equal to L_0 .

Now, we have this, now for these four vertices, we have the matrix here and we have found out what is called the minimum weighted matching associated with this, which means we have looked at 2 3 4 5 which is not here. If we take a matching out of this, if we take a matching out of these four, there are only two possibilities; 3 5 plus 2 4 is one matching and 3 4 5 2 is another matching. If we take a matching out of this, we say 3 5 2 4 is one matching and 3 4 5 2 is another matching. The minimum of these two will have to be less than or equal to half of this.

Let us call this length as L_{2345} . Let us call this as L_{2534} , as this length, by triangle inequality L_{2534} is less than or equal to L_0 , because $1 \cdot 3$ plus $1 \cdot 5$ is greater than or equal to $3 \cdot 5$.

So L_{2534} is less than or equal to L_0 . From this $2 \cdot 5 \cdot 3 \cdot 4$, we may get two matches, which are either $3 \cdot 5 \cdot 2 \cdot 4$ or it can be $3 \cdot 4 \cdot 5 \cdot 2$. The minimum of these two will have to be less than or equal to, let us call this, as sum L_m small matching.

This L_m which is the minimum of these two should be less than or equal to L_{2534} by 2. This is like saying if we add these four lengths, for example, $2 \cdot 5$ is 6, this is 6, $2 \cdot 4$ is 5, $3 \cdot 4$ is 8 and $3 \cdot 5$ is 9. We can look at the two matchings. The total is 9 plus 8 equal to 17 plus 5 equal to 22 plus 6 equal to 28. One matching is 14, $3 \cdot 5$ and $2 \cdot 4$. This is 9 plus 5 is 14, the other is 8 plus 6 equal to 14. You see that this is the minimum of them, has to be less than or equal to 28 by 2. It is just like saying I have a number, I divided into two numbers and if I divided into two numbers, such that the sum of these two, is the given number, the minimum of the 2 should be less than or equal to half of it. It is a very simple result, from which we can say that this L_m is less than or equal to L_{2534} by 2. Therefore this L_m is less than or equal to L_0 by 2. Now, these are matchings that are got from this and what are these?

These are matchings that are got from the complete graph associated with this. The complete graph is here. These matchings are also part of this. For example, $3 \cdot 5 \cdot 2 \cdot 4$ is here and $3 \cdot 4 \cdot 5 \cdot 2$ is also here. When this is part of this, when we identified this, we looked at $3 \cdot 4 \cdot 5 \cdot 2$ and we looked at the complete graph which is a 4 by 4 and we identified the minimum matching out of this complete graph. This is part of the complete graph and then we identified the minimum matching associated with this part.

The minimum matching that we get from here, will again have to be less than or equal to the minimum matching that we get from here. Therefore the L_{MATCHING} is less than or equal to L_0 by 2. This is less than or equal to L_0 by 2. L_{MATCHING} is less than or equal to L_m . Therefore L_{MATCHING} is less than or equal to L_0 by 2. This will be less than or equal to L_0 plus L_0 by 2. This is less than or equal to 3 by 2 L_0 . So, L_h by L_0 is less than or equal to 3 by 2 or 1.5.

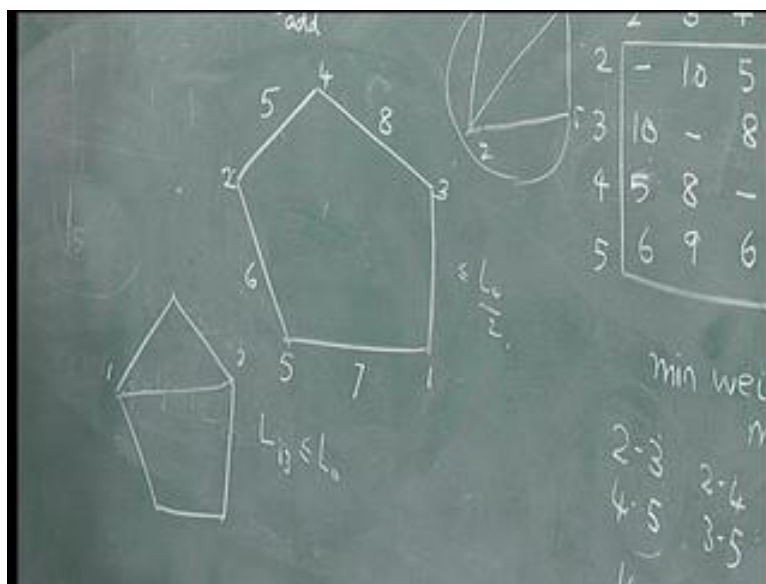
Again, just to explain again this L_0 by 2, let me quickly do that one more time. This is little more involved portion up to this it is fine. What we try to do is the following. If there is an unknown

L_0 , the moment we know that these are the vertices, it is always possible to do something like this, in this example, just you leave out 1. This 1 will have to be connected to either of the four out of these 2 4 3 and 5. It is like just leaving 1 out of the L_0 and joining the rest of them to get this figure L_{2534} . This L_{2534} is derived from L_0 , assuming that L_0 is not known, but the only thing we know is that, no matter what is the optimum solution, this L_{2534} will be less than or equal to L_0 based on triangle inequality. Because this 1 has been left out, so, L_{2534} , which are corresponding to the vertices that have odd degree.

Now that L_{2534} will be less than or equal to L_0 , then we go back and say that it is always possible to get a matching out of this. The minimum such matching will be less than or equal to L_0 by 2. L_m is less than or equal to L_0 by 2. Then we go back and explain that this is only a part of the complete graph from which we have got a minimum matching. This is the complete graph from which we get a minimum matching. Obviously the one that we get from the complete graph, that minimum matching should be less than or equal to every matching that is obtained out of this.

The $L_{MATCHING}$ will be less than or equal to this L_m which in turn is less than or equal to L_0 by 2. Therefore $L_{MATCHING}$ is less than or equal to L_0 by 2. L_h is less than or equal to L_0 plus L_0 by 2 which would give L_h equal to 3 by 2 L_0 . Whenever we have n is odd, if we have n equal to 5 then the number of vertices that we may have to look at is either 2 or 4.

(Refer Slide Time: 21:43)



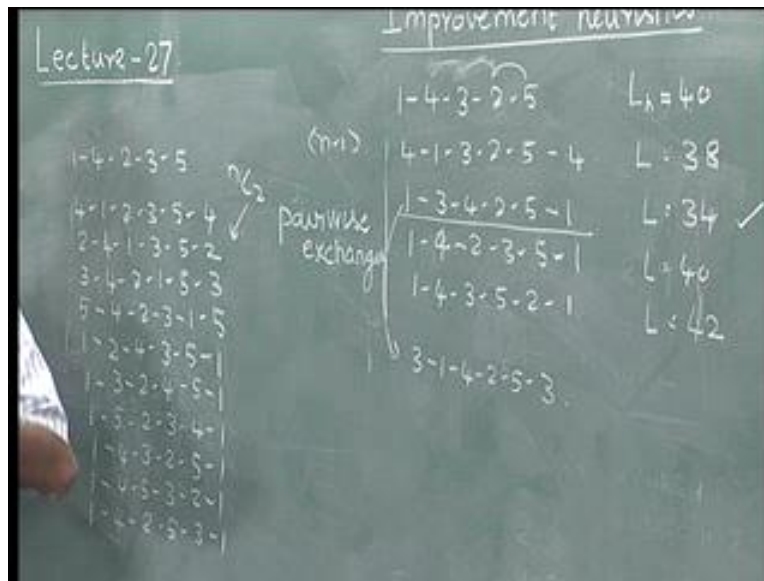
For 4 we went back and did this. For 2 what do we do, in the actual case where 1 3, where the only two vertices that had an odd degree. The spanning tree actually looks like this. The spanning tree had 5, 2 to 4 is 5. Then 2 to 5 is 6, 2 to 4 is 5 and 2 to 5 is 6, 5 to 1 is 7, 3 to 4 is 8 and the spanning tree looks like this. Then we actually do not know L_0 , but if we assume that somewhere the only 1 and 3. What we do is, we do not know but let us assume that 1 and 3 are here and we simply put this.

By a triangle inequality this L_{13} is less than or equal to L_0 , because L_{13} is less than or equal to this plus this. Therefore L_{13} is less than or equal to L_0 . Simply adding 1 3, again we are adding a quantity that is less than or equal to L_0 by 2. We can even prove that actually L_{13} is less than or equal to L_0 by 2 because from triangle inequality this portion, L_{13} is less than or equal to sum of these 2 and L_{13} is less than or equal to sum of this plus this plus this. Therefore L_{13} is less than or equal to L_0 by 2.

Even when we are adding only 1 edge into this, we are adding an edge whose weight is less than equal to L_0 by 2. Whenever the number of vertices with odd degree happens to be less than or equal to the actual number of vertices, we assume that we actually use triangle inequality to get this and prove that whatever is added into the minimum spanning tree has a weight, which is less than or equal to L_0 by 2. Therefore L_h by L_0 is always less than or equal to 3 by 2. This heuristic based on minimum matching, it has a worst case performance of 3 by 2, which means no matter what the size of the problem is, no matter what these numbers are, as long as the data satisfies triangle inequality, we can get a heuristics solution. This is utmost 50 percent away from the optimum.

These kinds of heuristics also give us an opportunity to understand how to derive or evaluate the worst case performance of these heuristics. It is possible to do this for certain heuristics. It is not possible to do for certain others. Those heuristics for which we could get this kind of analysis are called approximate algorithms. Once we define a heuristic as an approximate algorithm, then it means that there is an expression or a derivation for the worst case performance of the algorithm. We now move on to some improvement heuristics for the TSP. We will look at some improvement heuristics.

(Refer Slide Time: 25:29)



Improvement heuristics means, you already have a solution and you are trying to improve it. In a construction heuristics you do not have a solution; you build or construct a solution. All the 3 or 4 heuristics that we saw, the nearest neighbour, the one that was based on the branch and bound, twice around the tree heuristics and the heuristics based on the matching, they are all examples of construction heuristics. Let us look at some improvement heuristics. In order to get an improvement heuristic, you need a starting solution which we actually try and improve. Let us look at a starting solution. Let us look at the solution 1 4 3 2 5 1. This solution has length 1 to 4 is 9, 4 to 3 is 8, 17. 3 to 2 is 10, 27. 2 to 5 is 6, 33 and 5 to 1 is 7, 40.

Here L_h equal to 40. This has length equal to 40. Can we improve this further? A very simple thing to do is called pairwise interchange, which we can do. What are the easiest ways of doing pairwise? The pairwise can be done in more than one way.

In TSP, we generally leave this and we assume that we come back to this. The first thing we do is called as adjacent pairwise interchange. When we do an adjacent pairwise, I first swap just these two, I get 4, 1, 3, 2, 5, 1 which means I will come back to 4. This will give me a length. 4 to 1 is 9, 1 to 3 is 8, 17. 3 to 2 is 10, 27, 2 to 5 is 5, 32 plus 5 to 4 which is 6. So, this gives L equal to 38.

I can do another adjacent pair with this. So I try 1 3 4 2 5 and 1. So 1 3 is 8. 3 4 is 8, 16. 4 to 2 is 5, 21. 2 to 5 is 6, 27. 5 to 1 is 7. L equal to 34. We can try this adjacent pairwise to get 1 4 2 3 5 and 1. 1 4 is 9, 4 2 is 9 plus 5 equal to 14, 2 3 is 10, 14 plus 10 equal to 24. 3 5 is 9, 33. 5 1 is 7. This L equal to 40.

We can try this pairwise, so we get 1 4 3 5 2 1 with L equal to 1 4 is 9. 4 3 is 8, 17. 3 5 is 9, 26. 5 2 is 6, 32. 2 1 is 10, 42. We can do these pairwise exchanges. There are five numbers so we can do four pairwise exchanges. If there are n numbers we can always do (n minus 1) adjacent pairwise exchange. Each of these is called adjacent pairwise exchange. We can do n minus 1 adjacent pairwise exchange and take the best solution. In this case the best solution happens to be 34. At the moment we do not know that it is optimum.

The next thing that we can do is to go back to this and then look at some adjacent pairwise exchanges that are possible. If there is no improvement, then stop the algorithm. That is one way of doing this. Another way which is a slightly greedy way is, the moment we start with this 40 and moment we get this 38, we then look at this 38 and continue to do pairwise exchanges. That is also possible, but ordinarily you do one pass of the adjacent pairwise exchange and then leave it here. Because even if we try and make pairwise exchanges out of this, we would still get some solutions, some changes. This would give us the first exchange, would give us 3 1 4 2 5 3. 3 1 4 2 5 3 we have not evaluated this, so there is still a possibility that some changes can happen.

We can do one more pass of this algorithm to see if we can get an improvement. Of course, for this instant we will not get improvement because we already know that 34 is the optimum. Once we do not know the optimum, we can continue doing this, till there is no improvement and then stop with the best solution that we have. This is the adjacent pairwise exchange heuristics, which is an improvement heuristics.

Then next one is a general pairwise exchange heuristics, which means we can take out of these five and exchange all possible pairs. There are five nodes or there are five cities. A pairwise interchange can be done in $5C_2$ ways which is 10 ways. Let me just write down only those 10 and the first one would be one. The given one is 1 4 2 3 5 1. The first pairwise can be 4 1 2 3 5 4. The second one can be 1 and 3. I will have 2 4 1 3 5 2; then this exchange 3 4 2 1 5 3 and then this exchange 5 4 2 3 1 5. These are the four possible exchanges with 1. Then we look at this.

This is 1 2 4 3 5 1, that is I am exchanging this 4 and 2; I exchange this 4 and 3, I will get 1 3 2 4 5 1; then I exchange 4 and 5, to get 1 5 2 3 4 1, I will exhausted this. I come to 2 and 3, so 1 4 3 2 5 1; then I do 2 and 5, 1 4 5 3 2 1, I have exhausted this.

Then I come to this, I do 1 4 2 5 3 1. I start with this and I have 1 2 3 4 5 6 7 8 9 10 possible pairwise exchanges. This is superscript $n C_2$ number of exchanges. This is n minus 1 exchanges and so on.

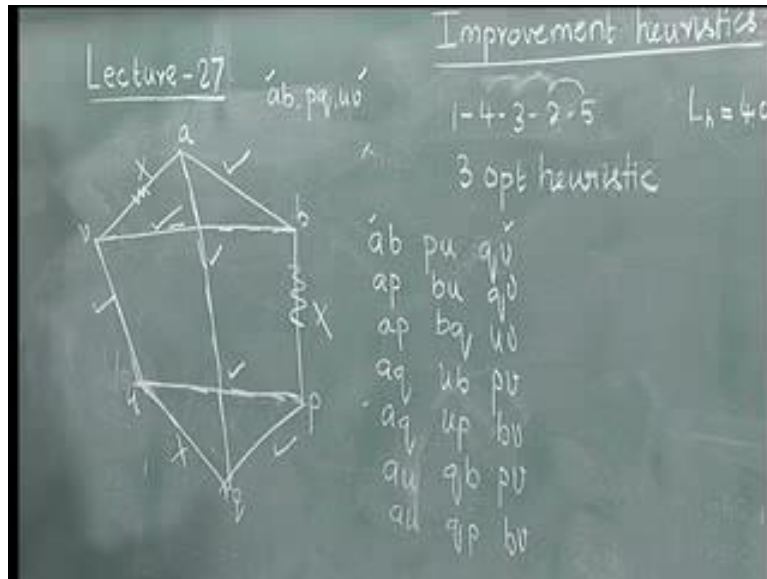
We will also realise that this is a subset of this 10. We can evaluate all these 10 solutions and then choose the best. That is the normal pairwise exchange heuristics or a pairwise exchange algorithm applied to the travelling salesmen problem.

Once again, in these instants we will get L equal to 34, because in the subset that we have evaluated, we have already got the solution with L equal to 34. We could do either an adjacent pairwise exchange, which means we evaluate n minus 1 new solution in one pass and then we do one pass of the total pairwise and then take the best out of this. We will have a small change in this and then we can do one more pass and proceed because we can take the best out of this and then again do one more pass of superscript $n C_2$ and see if there is a further improvement.

There will be some new solutions that will come in, while majority of the solutions will repeat because here we have evaluated superscript $n C_2$ out of the possible n minus 1 factorial. n minus 1 factorial, 24 solutions are possible. In this pass, we are evaluating only 10 out of the 24 solutions.

Once we get the best out of this we can again subjected to another pairwise interchange which means we will evaluate 10 more solutions and keep doing it till there is no improvement. We can either use an adjacent pairwise exchange or a complete or total pairwise exchange, where we evaluate $(n$ minus 1) solution in 1 pass or superscript $n C_2$ solutions in 1 pass and so on. Now we look at another type of improvement heuristics, which is very specific to the Traveling Salesmen Problem.

(Refer Slide Time: 35:52)



That is called the 3 opt heuristic or a 3 way exchange heuristic. Let us assume we look at a Traveling Salesmen Problem, let us say we consider a six TSP and let us call these six nodes as ab, pq and uv. What we can do is for these six nodes or 3 pairs of nodes, it is possible to create some more feasible solutions. For example, this is one feasible solution; we could create ab, pu, qv. What is that mean, ab, pu, and qv? It means these two edges go and they are replaced by these 2 edges. What we can do is, if we take three 3- edges ab, pq and uv, then we can show that 7 new solutions can be obtained. Including the given one, there are eight solutions, but seven new solutions can be obtained. One of which is ab, pu, qv. The others are ap bu qv, ap bq uv, aq ub pv, aq up bv, au qb pv and au qp bv. These are seven new solutions that are possible.

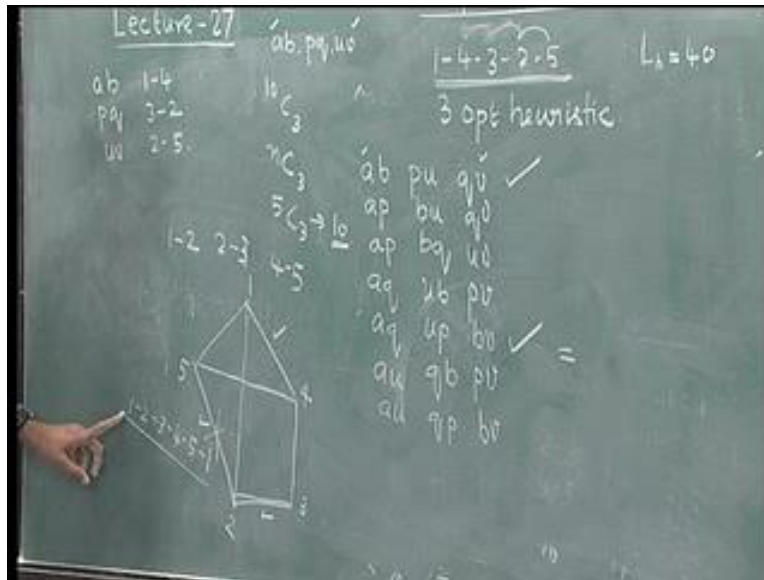
We also realise what are common in these including the given one is that the first one is always 'a' and the last one is always 'v'. The rest of them are shuffled inside, so since there are three edges, you can do that in 2 to the power 3 ways, So 2 to the power 3 minus 1 would give us seven new solutions one of which we showed as ab pu qv, they now have become ab pu qv.

Let us try and show this aq, up and bv. We show aq. Now the new solution will become aq up bv which means it comes here. aq qp pu, this goes, this goes and this goes. This means that this is not there, this is there aq qp. This one is there which comes here aq qp pu uv, so this one is there;

this one is there; this one is there; this one is left out; this is not there; this is not there and this is not there. The 3 edges are replaced by 3 other edges so we have seven such possibilities.

If the given problem has more than six nodes then it will have more than ab pq uv.

(Refer Slide Time: 40:28)



For example, if we look at 10 city problem in a 10 city problem the ab pq uv by itself can be chosen in $10C_3$ ways. In general it can be chosen in superscript $n C_3$ ways. If we have a 5 city problem like what we have, again these ab pq uv can be chosen in $5C_3$ ways which are ten ways except that there will be repetitions as we move along.

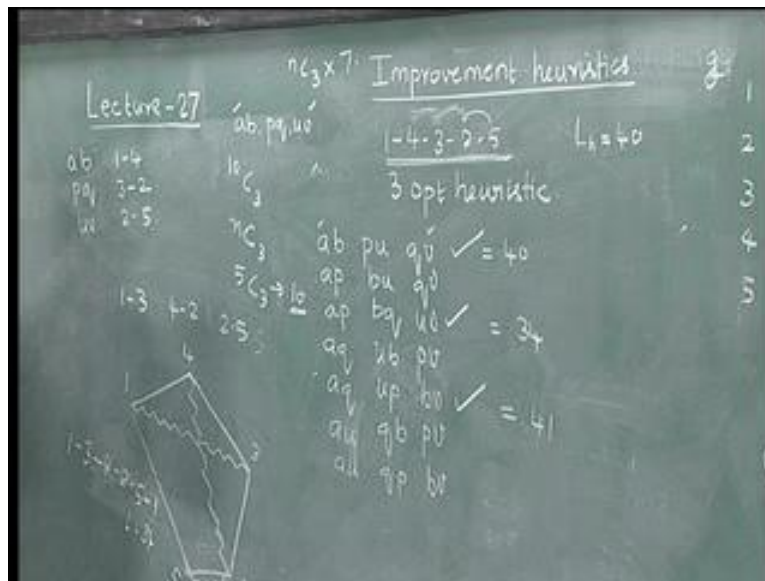
For example, let us start with this feasible solution with length equal to 40. If we chose ab as 1 4, pq as 3 2 and uv as 2 5 then the first one will be ab pq uv, ab pu. ab is 1 4. We include 1 4 ab, pu is 3 2, we are looking at this, ab is 1 4, pu is 3 2, qv is 2 5, which means we go back to this. We do 1 4, then we do 4 3, 3 2 to 5 1 which is the same as the given one then there is a repetition whereas if when we look at this one we get aq is 1 2, ub. is 2 1 which we cannot have, aq,

The up is 2 3 and bv is 4 5. So aq up bv, aq is 1 2, up is 2 3 and bv is aq, up and bv, bv is 4 5. Now we start with 1 2. What it means is, we have 1 4 3 2 5. Now ab pq, ab is 1 4, pq is 2 3 and uv is 2 5. These are the 3. We go to 1 2, we go to 1 2, then we use 2 3. We again use 2 3 and we use 4 5. The new solution will be 1 2 2 3 3 4 4 5 and 5 1. This goes, this remains in the solution.

This remains in the solution. So we have 1 2 2 3 3 4 4 5 5 1. This goes out of the solution. 2 5 goes out of the solution. We get a different solution out of this.

If we take $ab\ pq\ uv$ to be 1 4 3 2 2 5 and then we can substitute in these. For example, this substitution would give us a solution with 1 to 2 is 10, 2 to 3 is 10, 20. 3 to 4 is 8, 28. 4 to 5 is 6, 34 plus 7 equal to 41.

(Refer Slide Time: 45:06)



This would give us the same solution with 40. If we look at this, for example, $ap\ bq\ uv$, would give us ap is 1 3, bq is 4 2, uv is 2 5, so, we look at this. Then our original solution is 1 4 3 2 5. We are adding 1 3. Let me put it this way. We are adding 1 3. We are adding 4 2 and we have 2 5. I have one the new solution, is now 1 3 which I have added. 2 4, 1 3 4 2 5 1, because I have added 1 3 3 4 4 2. I have added 2 5 again and 5 1 with L equal to 34. This would give us L equal to 34.

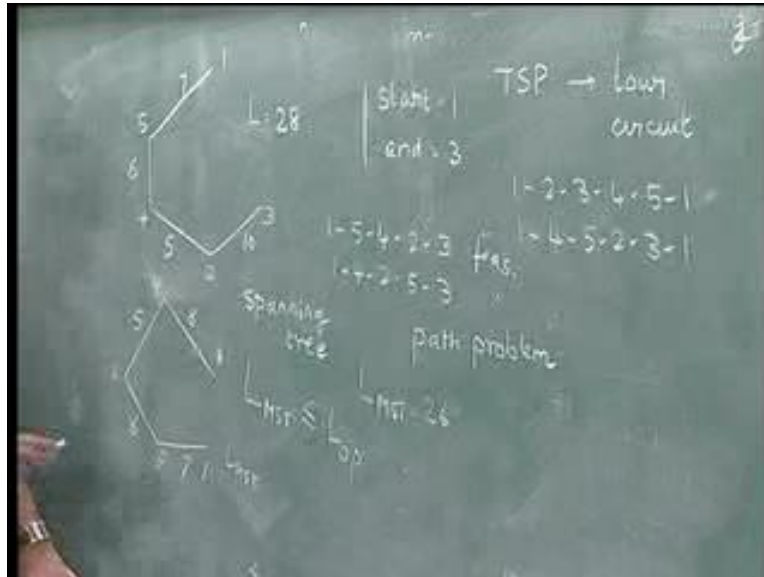
Like this, for every $ab\ pq\ uv$ we can get 7 solutions. But if $ab\ pq$ and uv are not distinct, as in our example, there are only 5 nodes, therefore we cannot get distinct $ab\ pq\ uv$. When we do not have distinct $ab\ pq\ uv$, you will not get 7 distinct solutions. You may have one or two solutions repeating as it happened here. The original solution came back again when we computed this. Nevertheless for every $ab\ pq\ uv$, which can be done in superscript $n\ C_3$ ways, we can get a

maximum of, or we evaluate superscript $n C_3$ into 7. Sum of which may repeat. It still does not matter.

One pass of the 3 opt will actually do superscript $n C_3$ into 7 solutions, sum of which may repeat and we may take the best out of the superscript $n C_3$ into 7 solutions. For our example, we have already got this 34 and since we know that this is the optimum, we cannot have any more improvement over the 34. One pass will evaluate a maximum of superscript $n C_3$ into 7 and then we can continue the passes till it does not show any improvement. Then the algorithm will terminate with the best available solutions.

In our case, it will terminate with this solution with Z equal to 34. The improvement heuristics, we have seen 3 of them. One is the adjacent pairwise exchange heuristics, the other is the 2 opt or the pairwise exchange and the other is called 3 opt. 3 opt is a very famous heuristic. This came in the year 1971 and it is proposed by Kernigan and Lin it is a very famous heuristic even today for solving the TSP, particularly large instances of the Traveling Salesmen Problem. But all the improvement heuristics, essentially work on the idea that we have a solution and then we try to improve the solution using these heuristics. So far in the Traveling Salesmen Problem, we define the TSP and we saw the formulations and then we saw three branch and bound algorithms. Then we saw some heuristics, some number of construction heuristics and some number of improvement heuristics. Now we look at one more aspect of the Traveling Salesmen Problem where we again use the same matrix and study this aspect.

(Refer Slide Time: 49:10)



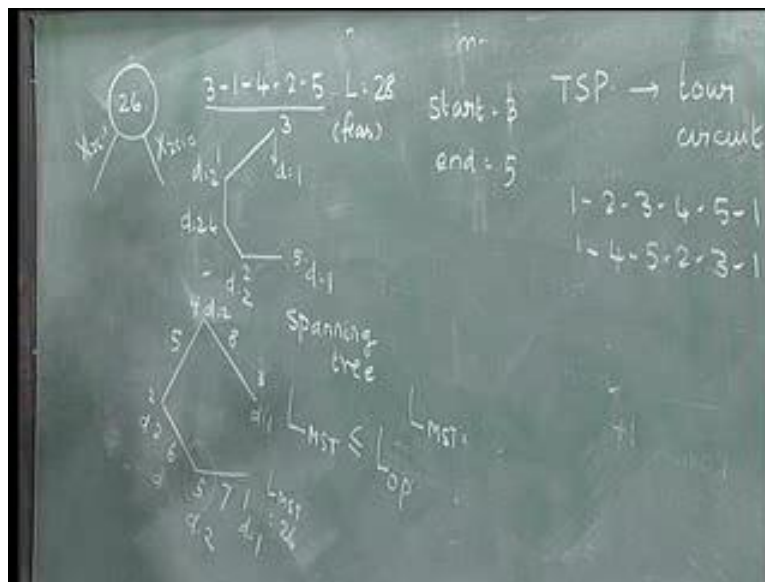
Usually in the Traveling Salesmen Problem, we said the person starts in any city, goes to every city once and only once and comes back to the starting point. Such a problem is called the TSP or the Traveling Salesmen Problem or the feasible solution is a tour or a circuit which means a solution 1 2 3 4 5 1 is feasible. A solution 1 4 5 2 3 1 is feasible for a 5 city problem. Can we define another kind of a TSP where I say that I want to start from 1. One is a starting node. Let us say my ending node is 3. But i want to go from 1 to 3, going through every other vertex once and only once. For example, 1 5 4 2 3 is a feasible solution to this problem. Because I start with 1, I end with 3, I can go through this. I may have 1 4 2 5 3 is also another feasible solution. Such a problem is called a traveling salesmen path problem where you want to find out a path from given i to j , such that, in this path you go to every other vertex once and only once. You know the starting node, you know the ending node. These are feasible solutions to the traveling salesmen path problem. There are times traveling salesmen path problem is also an important problem. There quite a bit of work that is been done on the traveling salesmen path problem.

If we try to plot this, if we go back and draw a graph associated with this. Let us say 1 5 4 2 3. This is a path. Now, one important observation is that the any feasible solution to the path problem is a spanning tree, by itself is a spanning tree. Let us look at the length of this. This is 1 to 5 is 7, 4 to 5 is 6, 2 to 4 is 5 and 2 to 3 is 10. This is 7 plus 6 equal to 13 plus 5 equal to 18 plus 10 equal to 28.

We also know for the same problem L_{MST} is 26 which we computed earlier. Therefore L_{MST} is always less than or equal to L_0 of the path problem. This is a feasible solution. If we call for example, L_{0p} as the L_0 for the path problem then L_0 for the path problem is less than or equal to 28. But L_0 of the path problem being a spanning tree L_{MST} has to be less than or equal to L_0 of this one. So L_{MST} is not only a lower bound to the Traveling Salesmen Problem, it is also a lower bound to the traveling salesmen path problem. If we draw our MST with 26, we would get 2 4 5, 2 to 5 is 6 and 4 to 3 is 8 and 1 to 5 was 7, so we got 5 here. 3 to 4 was 8. 1 to 5 was 7, 2 to 5 was 6, 7 plus 6 equal to 13 plus 5 equal to 18 plus 8 equal to 26. Now this is the L_{MST} .

In this path problem, we wanted to do with 1 and 3, so straight away we have L_{MST} is a feasible solution to this path problem. Therefore for this instance, L_{MST} is the optimum solution, so you could get the solution to this problem with MST itself satisfying it and giving us a solution with L equal to 26. Whereas if we look at a path problem not between 1 and 3. Suppose we say the starting node is as follows.

(Refer Slide Time: 54:10)



Hence 3 is the starting node and the ending node is 5. Let us say a feasible solution is 3 1 4 2 5 with L equal to 3 to 1 is 8, 1 to 4 is 9 17, 2 to 4 is 5 22 and 2 to 5 is 6 28. With L equal to 28 is a feasible solution to this problem. This L_{MST} is again a lower bound to this with L equal to 26, is a lower bound to this. We can actually say that the lower bound is 26 and then if we draw this 3 1

4 2 5, this is a spanning tree. This is a feasible solution, so this should have a property, that the degree of this is equal to 1; degree equal to 1; degree equal to 2; degree equal to 2; degree equal to 2.

Now, L_{0p} will satisfy the property that the starting and ending nodes will have degree equal to 1 while the rest of the nodes will have degree equal to 2. If we go to the MST, we have degree equal to 1 here; degree equal to one; degree equal to 2; degree equal to 2 and degree equal to 2. We know that for 5 and 3, the degree has to be equal to 1 while for the rest of the things the degree has to be equal to 2. There is a violation here. We could say that either this is in the solution or this in the solution. We could do a branch and bound algorithm from here, saying either d_{25} equal to 1 or X_{25} equal to 1 or X_{25} equal to 0.

Like this we can build a branch and bound algorithm to get the optimal solution to the traveling salesmen path problem also. In this lecture we are not going to elaborate on this but the purpose is to show that it is possible to have a branch and bound algorithm to get the optimal solution to this problem which uses the minimum spanning tree because for this problem L_0 should have some properties and the minimum spanning tree. If it violates that then it is always possible to get a spanning tree which satisfies that.

This branch and bound algorithm at every node will solve a minimum spanning tree problem and tries to see whether this property of degree equal to 1 for 3 and 5 and degree equal to 2 for the others. It satisfies this so it is possible to do a branch and bound to solve the traveling salesmen path problem also. The more important understanding is that the minimum spanning tree is lower bound to the path problem as well. With this we come to the end of our discussion on the traveling salesmen circuit problem and the path problem. In the next lecture we will look at the Chinese postman problem.