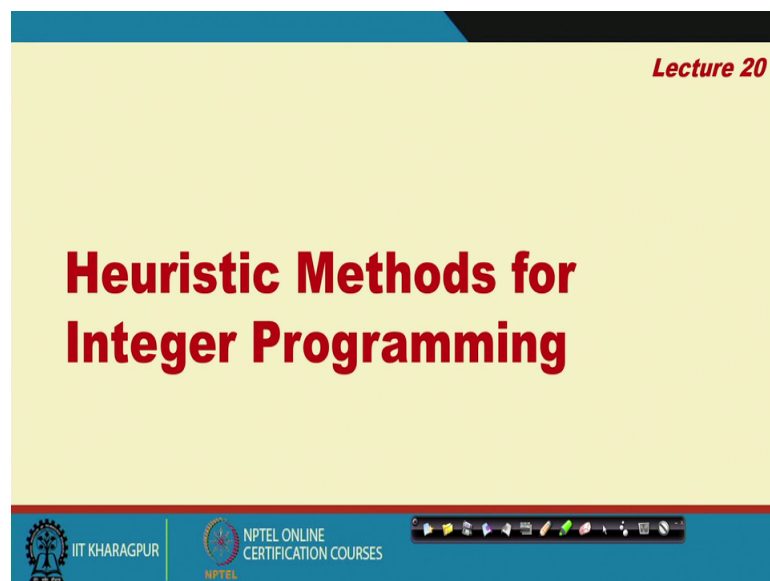


Selected Topics in Decision Modeling
Prof. Biswajit Mahanty
Department of Industrial and Systems Engineering
Indian Institute of Technology, Kharagpur

Lecture - 20
Heuristic Methods for Integer Programming

So, in our course Selected Topics in Decision Modeling, today we are in 20th lecture that is heuristic methods for integer programming right.

(Refer Slide Time: 00:27)



So, let us ahead the Heuristic Methods for Integer Programming. We have seen different other ways of solving travelling salesman problem, but you know there are certain methods which could be called heuristics.

(Refer Slide Time: 00:38)

Heuristic Methods for TS Problem

- Nearest Neighbour Method
- Sub-tour Reversal Method
- Greedy Heuristics Method
- Genetic Algorithm
- Other Evolutionary Algorithms

B&B	Heuristics
Optimal sol ⁿ	Optimal sol ⁿ is <u>not</u> guaranteed

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

Now, see TS problem is a very typical integer programming problem, there are other integer programming problems also there could be many other heuristics for solving them, but specifically we are going to discuss the heuristics methods for travelling salesman problem in this lecture.

So, here are some of the methods like the nearest neighbour method, the sub tour reversal method, greedy heuristics methods, genetic algorithms and other evolutionary algorithms such as stimulated annealing and so on. Certain things you must note about the heuristic methods in the very beginning. You see if I compare the branch and bound algorithm verses heuristic methods, one thing should be very clear to you that, this gives us an optimal solution, but here the optimal solution is not guaranteed right.

So, while an optimal solution is guaranteed in the branch and bound algorithm, in heuristic method the optimal solution is not guaranteed. But then you may ask then when we have algorithmic for getting optimal solutions why are we going to do heuristic method.

The reason is very simple that these are n p hard problems, the travelling salesman problem is an n p hard problem it means the complicity of the problem rises exponentially as the problem size increases. So, when we have a very large travelling salesman problem with number of cities very large, then it may so happen that the branch

and bound or other exact logarithms becomes extremely difficult to employ. In those situations we have the heuristics as the only option is it alright.

So, that is the essential idea that why we should still have the heuristics methods, which does not guarantee optimal solution is it alright. But it also there for the heuristic are methods, which can quickly give us a solution right, but it does not guarantee an optimal solution, so these point has to be remembered right.

Another note that should also we kept in mind is that, the genetic algorithms and other heuristic algorithms, they are quite involved right, that really require certain mode knowledge that really do not have at this point in our lecture. What I have really planned is that, a set of lectures are separately kept aside for the meta heuristics is it alright.

In those lectures we are going to discuss genetic algorithm and other evolutionary algorithms separately is it alright. So, in these particular lecture we are going to discuss these methods, that is the nearest neighbour method, the sub tour reversal method and the greedy heuristics method.

(Refer Slide Time: 04:49)

Solving TSP by Nearest Neighbour Heuristics

Solve the 5-city TSP for minimizing costs as given in the matrix below:

	A	B	C	D	E
A	M	10	3	6	9
B	5	M	5	4	2
C	4	9	M	7	8
D	7	1	3	M	4
E	3	2	6	5	M

Nearest Neighbour Heuristics

- 1) Select a random city.
- 2) Find the nearest unvisited city and go there.
- 3) Are there any un-visited cities left? If yes,
- 4) repeat step 2.
- 5) Return to the first city.

So, let us start with the first one that is the nearest neighbour method. So, let us take a 5 city travelling salesman problem for minimizing cost as given in the matrix below. So, how the nearest neighbour heuristic works, first of all select a random city that means, we can start with any city, then find the nearest unvisited city and go there is alright and

then check whether there are any unvisited cities left, if yes repeat step 2 is it alright. If some more cities are available so suppose from A we have come to B, then from B again try to find out the nearest neighbour is it alright. So, that is how we should do and then repeat step 2 and then return to the first city is it alright.

Now, there is certain things to note here is that, while you apply the nearest neighbour nearest heuristics again you know there are certain things to note that you know A to B and B to A distance is may or may not be same is it alright. So, there are problems where you know suppose I have a network let us let us look at a particular network, suppose I have a network like these. Suppose we have 4 cities A B C D and we have a network like these and this network, we have certain distances let us call them 4 4 4 3 and 2 so those are the distances.

So, you will see A to B and B to A have the same distance is it alright in this network, but in our matrix A to B is 10 whereas, B to A is 5 this creates a certain difficulty you know. So, what is the nearest neighbour network of A, you know where is what is the nearest neighbour of A. So, you know we have to think whether we have to go row wise or column wise. So, usually it does not matter if you have a system like this; that means, if you have a symmetric matrix where the A to B and B to A are same you know in these like in these example that I have given, A to B is 4 and B to A is also 4. So, there is no confusion.

So when it is like this, whether you go row wise or column wise it does not matter, but in the kind of situation that we have had, that you know A which is the nearest B C D or E you know it really creates a problem whether we go row wise or column wise. So, arbitrarily I will show examples which we are selecting row wise, but it is not really necessary that you do row wise you can also do column wise is it alright.

It also creates a problem that when you go to the last city obviously, you have to choose a distance which is beyond you and since you know row wise and column wise distances are could be very different, your answer could be unexpected also suddenly. Because suppose A to D 6, but D to a is 7 right. So, look here like A to E just come to here A to E is 9, but E to A is only 3. So, you see while A to E is so high, but if you go back from E to A you really gain right.

So, these enumerations will be there if we have different distances to and from right. So, the problem that we have taken we have different distances, so you know there will be some enumerations of that type. But if you have a perfect problem then probably the algorithm would work a little better, but anyhow let us see the method right.

(Refer Slide Time: 09:21)

TSP by Nearest Neighbour Heuristics

	A	B	C	D	E
A	M	10	3	6	9
B	5	M	5	4	2
C	4	9	M	7	8
D	7	1	3	M	4
E	3	2	6	5	M

- Arbitrarily start at A
- Nearest neighbour C
Choose C

	A	B	C	D	E
A	M	10	3	6	9
B	5	M	5	4	2
C	M	9	M	7	8
D	7	1	3	M	4
E	3	2	6	5	M

- Now check from C
- C to A block by inserting M
- Nearest neighbour D
Choose D

A-C-A
Subtour

Nearest Neighbour Heuristics

- 1) Select a random city.
- 2) Find the nearest unvisited city and go there.
- 3) Are there any unvisited cities left? If yes,
- 4) repeat step 2.
- 5) Return to the first city.

So, what we do in the nearest neighbour algorithm, the first of all we arbitrarily start at a city A; so it is highlighted we have started at a city A and then we have all those distances. So, which is the nearest neighbour you know just see very clearly nearest neighbour is C, so we choose C so that means we have chosen C.

So, now C is going to be our the (Refer Time: 09:47) so now you check from C right, what we have done we look here we have blocked A right we have blocked A and while you check C you see C to A is 4, but since you have already come from a to C you know C is already taken right.

So, you cannot go back to C if you go back to C what will happen? Suppose you have A to C and then again go back to A you just cannot do that right, you just cannot do that if you do that there will be a problem, what is that problem you will get a sub tour. So, moment you get a sub tour then there is no point on carrying on with the heuristics and there will be a difficulty.

So, we cannot do that so what we do really we look at the C and we blocked C to A 2 time M there. So, you see we have blocked C to A is blocked by inserting M, so we have inserted M and we have blocked C to A. Now look at the remaining distances, which is the minimum it is C to D that is 7. So, you are clear that now we should choose D right, so that is our second step. So, we have A gone to C and then gone to D right.

(Refer Slide Time: 11:30)

TSP by Nearest Neighbour Heuristics

	A	B	C	D	E
A	M	10	3	6	9
B	5	M	5	4	2
C	M	9	M	7	8
D	M	1	M	M	4
E	3	2	6	5	M

- Now Check from D
- Block distances to A and C
- Nearest neighbour B

Choose B

A-C-D

Nearest Neighbour Heuristics

- 1) Select a random city.
- 2) Find the nearest unvisited city and go there.
- 3) Are there any unvisited cities left? If yes,
- 4) repeat step 2.
- 5) Return to the first city.

	A	B	C	D	E
A	M	10	3	6	9
B	M	M	M	M	2
C	M	9	M	7	8
D	7	1	3	M	4
E	3	2	6	5	M

- Now check from B
- Block distances from A, C, D
- Nearest neighbour E

Choose E

A-C-D-B

So, look here, now we have blocked A and C and we have come to D. So, what we have had? We have had A to C and then to D this is what we have. Now, from A to C to D see look A and C should be blocked and D is taken next, so as A and C is blocked again in D block distance is to A and C. So again you put this as M. Now, remaining distance is the minimum again is comes out be is 1 that is D to B right. So, nearest neighbour is now B from D and therefore we choose B is it alright.

So, we choose B so what will get now we have got A to C to D to B this is what we have got, again now check this time we have blocked A C D all are checked blocked and B and again we have put all this A C D as M right and this time only one distance is left that is E, so that E has to be chosen so E has chosen.

(Refer Slide Time: 12:50)

TSP by Nearest Neighbour Heuristics

	A	B	C	D	E
A	M	10	3	6	9
B	5	M	5	4	2
C	M	9	M	7	8
D	M	1	M	M	4
E	3	M	M	M	M


Hence, We have chosen A-C-D-B-E-A
This is a Complete Route – Feasible
TC* = 16
Compare this with Branch and Bound Solution A-C-D-B-E-A
Which was the same solution!
But, this is incidental!

- Now Check from E
- This is last node
- Block distances to all except A
- Nearest neighbour A
- Choose A
- Nothing more to choose!

	A	B	C	D	E
A	M	10	3	6	9
B	5	M	5	4	2
C	4	9	M	7	8
D	7	1	3	M	4
E	3	2	6	5	M

Nearest Neighbour Heuristics

- 1) Select a random city.
- 2) Find the nearest unvisited city and go there.
- 3) Are there any unvisited cities left? If yes,
- 4) repeat step 2.
- 5) Return to the first city.



So, right so we have got A C D and B right we have chosen like these that is A to C to D to B. So, now obviously when you check from E and this is the last note, now all other distances should be blocked. So, nearest neighbour will be nothing but A so we have to go back to A that means, A to C to D to B to E to A. So, look here we have got a complete route and it has to be fusible and what is D C start is 16 right so we have got a solution.

Now, if you compare with the branch and bound solution incidentally, that solution is also A to C to D to B to E to A. So, each one was also the same solution. But do you know we got the optimal solution here why optimal because, you knew it in branch and bound we got the same solution, but then there is no such guarantee that we are going to get these kind of optimal solution all the time right.

(Refer Slide Time: 14:09)

6-City TSP by Nearest Neighbour Heuristics

Solve the 6-city TSP for minimizing costs as given in the matrix below:

	A	B	C	D	E	F
A	M	25	18	35	50	39
B	21	M	28	16	30	13
C	22	28	M	14	16	20
D	35	12	14	M	12	12
E	50	30	16	12	M	8
F	39	15	20	12	7	M

Nearest Neighbour Heuristics

- 1) Select a random city.
- 2) Find the nearest unvisited city and go there.
- 3) Are there any un-visited cities left? If yes,
- 4) repeat step 2.
- 5) Return to the first city.

So, let us look at one more example that is the 6 city example.

(Refer Slide Time: 14:17)

TSP by Nearest Neighbour Heuristics

	A	B	C	D	E	F
A	M	25	18	35	50	39
B	21	M	28	16	30	13
C	22	28	M	14	16	20
D	35	12	14	M	12	12
E	50	30	16	12	M	8
F	39	15	20	12	7	M

- Arbitrarily start at A
- Nearest neighbour C
- Choose C ✓

A-C

	A	B	C	D	E	F
A	M	25	18	35	50	39
B	21	M	28	16	30	13
C	M	28	M	14	16	20
D	M	12	M	M	12	12
E	50	30	16	12	M	8
F	39	15	20	12	7	M

- Now check from D
- Block D to A, C
- Nearest neighbour B
- Choose B ✓

A-C-D-B

	A	B	C	D	E	F
A	M	25	18	35	50	39
B	21	M	28	16	30	13
C	M	28	M	14	16	20
D	35	12	14	M	12	12
E	50	30	16	12	M	8
F	39	15	20	12	7	M

- Now check from C
- Block C to A
- Nearest neighbour D
- Choose D ✓

A-C-D

	A	B	C	D	E	F
A	M	25	18	35	50	39
B	M	M	M	M	30	13
C	M	28	M	14	16	20
D	M	12	M	M	12	12
E	50	30	16	12	M	8
F	39	15	20	12	7	M

- Now check from B
- Block B to A, C, D
- Nearest neighbour F
- Choose F ✓

A-C-D-B-F

So, look at the 6 city example that we have taken earlier and see what happens there. So, if you take the 6 city example, so you see arbitrarily start at A so you see the nearest is C, so nearest neighbour C. So, we choose C right so we have come A to C. Now, again from C we block A. So, put an M out there C to A and which is the nearest neighbor? Nearest neighbour is D. So, choose D then we got A to C to D, then we block A and C look at D and put all those distance is to M. Now obviously B E F all are equal distance but

arbitrarily choose let us say B, so B is chosen. So, we have now got A to C to D to B right again A C D are all blocked and look from B put all these A C D to M out of the 2 distances the minimum is F so F is chosen. So, what we got? A to C to D to B to F right; So, all this things are obtained and now go back go to the next slide.

(Refer Slide Time: 15:34)

TSP by Nearest Neighbour Heuristics

	A	B	C	D	E	F
A	M	25	18	35	50	39
B	M	M	M	M	30	13
C	M	28	M	14	16	20
D	M	12	M	M	12	12
E	50	30	16	12	M	8
F	M	M	M	M	7	M

- Now check from F
- Block F to A, C, D, B
- Nearest neighbour E
- Choose E

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

	A	B	C	D	E	F
A	M	25	18	35	50	39
B	21	M	28	16	30	13
C	22	28	M	14	16	20
D	35	12	14	M	12	12
E	50	30	16	12	M	8
F	39	15	20	12	7	M

Hence, We have chosen A-C-D-B-F-E-A
This is a feasible and Complete Route
TC* = 114
Compare this with Branch and Bound
Solution A-C-E-F-D-B-A (TC* = 87)
Which was a better solution!
The heuristic has not work

	A	B	C	D	E	F
A	M	25	18	35	50	39
B	M	M	M	M	30	13
C	M	28	M	14	16	20
D	M	12	M	M	12	12
E	50	M	M	M	M	M
F	M	M	M	M	7	M

- Now check from E
- This is last node
- Block all except A
- Nearest neighbour A
- Choose A
- Nothing more to choose

Then we find that when we come to E right, so check from F block F from A C D B and from F we see we have put all M, now only one distance is left that is E right. So, what we get now? We get A to C noise] to D to B to F to E right. Then we have no other option, but to choose the E to A because, we have everything else is blocked and this is a last note and we have to come back to A.

So, we come back to A so we get now A to C to D to B to F to E to A. But how does this solution compares with the best possible solution is alright. Look here when we compare this with the branch and bound solution, that is A to C to E to F to D to B to A we find the total TC star was 87 right.

So, if you really look that what is the solution that we have had in our branch and bound, we found that the optimal solution there was having A TC star 87, but here our TC star is 114, which is not good; obviously, when we have 87 getting 114 through the nearest neighbour method is not that good at all.

Obviously one of the reasons why it happen is if you compare A to F and you know sorry A to E ah; obviously, A to E and E to A was 50-50, but then not in every other case right. There are cases where the distances are very different. Because the way we have optimized everything, but the last leg that is E to A we had no choice we had to chose it right.

So, that really created imbalance and obviously we did not get very good solution. So, a good solution is really not guaranteed by the nearest neighbour method. Sometimes the heuristic may return not so good solution right. So, we may have to leave it that, fine let us see the other algorithms. Let us look at the next one this is called the sub rout reversal heuristics.

(Refer Slide Time: 18:37)

TSP by Sub-Route Reversal Heuristics

	1	2	3	4	5	6	7
1	M	12	10	M	M	M	12
2	12	M	8	12	M	M	M
3	10	8	M	11	3	M	9
4	M	12	11	M	11	10	M
5	M	M	3	11	M	6	7
6	M	M	M	10	6	M	9
7	12	M	9	M	7	9	M

Sub-Route Reversal Heuristics

Initialization
Start with any feasible initial trial solution

Iteration
For the current trial solution, consider all possible ways to performing a sub-tour reversal (not entire tour)
Select the one that gives the largest decrease

Stopping rule
Stop when no further reversal improve the current trial solution

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

So, here is a problem that is the 7 city problem, and you know we start with see look here the 7 city problem, just imagine if we have to even solve by branch and bound by hand it is not going to be easy, we may have to branch quite a bit right may have to branch quite a bit.

Whereas the sub tour reversal can give us a quick solution even minimum, the you know the neighborhood the nearest neighbourhood algorithm can also give us a quick solution no problem at all.

So, here start with any feasible initial trial solution and then for a current trial solution consider all possible ways to performing a sub tour reversal not entire tour. So, what is a sub tour reversal I will explain. Then select the one that gives the largest decrease right.

So, basically start with a feasible solution trial solution then keep doing sub tour reversals; obviously, not a complete tour reversal that you should avoid and then see what is the minimum that you get and then stop when no for the reversal improve the current trial solution right that is very interesting let us see how it works.

(Refer Slide Time: 19:52)

TSP by Sub-Route Reversal Heuristics

	1	2	3	4	5	6	7
1	M	12	10	M	M	M	12
2	12	M	8	12	M	M	M
3	10	8	M	11	3	M	9
4	M	12	11	M	11	10	M
5	M	M	3	11	M	6	7
6	M	M	M	10	6	M	9
7	12	M	9	M	7	9	M

Arbitrarily start with say **1-2-3-4-5-6-7-1** Feasible $TD^* = 69$

Suppose we reverse 2-3

Then out of 1-2, 2-3, 3-4, 4-5, 5-6, 6-7, 7-1

1-2, 2-3, 3-4 will be affected!

and will be replaced by 1-3, 3-2, 2-4

Then we get **1-3-2-4-5-6-7-1** Feasible $TD^* = 68$

Handwritten notes:

$12+8+11+10+8+12+3+9=69$

$12+8+11+10+6+9+7=68$


Sub-Route Reversal Heuristics

Start with any feasible initial trial solution

For the current trial solution, consider all possible ways to performing a sub-tour reversal (not entire tour)

Select the one that gives the largest decrease

Stop when no further reversal improve the current trial solution



So, arbitrarily suppose a you choose 1 to 2 see all those highlighted that is 1 to 2, 2 to 3, 3 to 4, 4 to 5, 5 to 6, 6 to 7 and 7 to 1, now why did we choose it? Because it is so easy to choose that is the reason no other reason that it is so, easy to choose that is 1 to 2, 2 to 3 to 4 to 5 to 6 to 7 to 1, so that is how we started.

It is not you have to start like that you can you can randomly choose anything and you can start with that. So when we started with that if you add them all you get 20, 31, 42, 48, 57, 69 so total distance 69 this is definitely feasible solution so that is the beginning solution for us.

Now, suppose we reverse 2 3 so you see what happens? You know this is suppose this is node 1 this is node 2 then node 3 , so we are taking in this way 3 to 4 etcetera right. So, if we reverse then what you do is you see we then connect 1 to 3 and 3 to 2 and 2 to 4 see

the difference. Earlier it was 1 to 2, 2 to 3, 3 to 4 right now it has become 1 to 3. So, 2 to 3 becomes now 3 to 2, so 1 to 3, 3 to 2, 2 to 4 1 to 3, 3 to 2, 2 to 4 that is exactly what is shown here see 1 to 3, 3 to 2 and 2 to 4 right.

So, earlier it was 12 plus 8 plus 11 plus dot dot dot. So, 12 plus 8 plus 11 plus dot dot dot replace by 10 plus 8 plus 12 plus dot dot dot, so that means earlier it was 31 plus dot dot dot now it is 30 plus dot dot dot.

So, what is the difference? Your guess is right now the new route is one lays is alright. So, you see 1 3 2 4 5 6 7 1 is a feasible route and this route has a TD star equals to 68 which is one lays. So, we see by sub tour reversal we have got one more route that is better than the original. The question is will it always be like that. Answer is no no by sub tour reversal there is no such guarantee that you get a better route.

In fact, you may get worst rout also; so is there is no such guarantee that you do sub tour reversal and like magic you get something better right. But if you keep on doing those sub tour reversal see how many options you have you can reverse 2 3, you can reverse 3 4, you can reverse 4 5, you can reverse 5 6, you can reverse 6 7 right. So, you can do and then later on you can reverse 2 3 4, 3 4 5, 4 5 6, 5 6 7 etcetera etcetera.

So, you can keep on doing like these, you can do different sub tour reversals and then you can get something, which is better than what you started with right. So this is the mechanism etcetera and how many different options you may have to go through the number is going to increase with the size of the problem, but still this is not a big calculation really; so obviously is not going to be that difficult.

(Refer Slide Time: 24:15)

TSP by Sub-Route Reversal Heuristics

	1	2	3	4	5	6	7
1	M	12	10	M	M	M	12
2	12	M	8	12	M	M	M
3	10	8	M	11	3	M	9
4	M	12	11	M	11	10	M
5	M	M	3	11	M	6	7
6	M	M	M	10	6	M	9
7	12	M	9	M	7	9	M

Iteration 1		
Reversal	Route	TD*
Original	1-2-3-4-5-6-7-1	69 ✓
Reverse 2-3	1-3-2-4-5-6-7-1	68
Reverse 3-4	1-2-4-3-5-6-7-1	65
Reverse 4-5	1-2-3-5-4-6-7-1	65
Reverse 5-6	1-2-3-4-6-5-7-1	66

Choose 1-2-4-3-5-6-7-1, TD* = 65

33

So, you see chart has been prepared as a iteration 1, the original was 1 2 3 4 5 6 7 1 the TD star was 69 that was a starting with. If you reverse 2 3 you get 1 2 3 4 5 6 7 1 you get TD star 68,. If you reverse 3 4 you will get 1 2 4 3 5 6 7 1 look here this 3 4 look 3 4 has been reversed to 4 3 alright, then you get TD star 65 alright.

So, can you locate this one you see 1 to 2, 2 to 4, then 4 to 3 3 to 5, 5 to 6, 6 to 7 7 to 1. So, you add them 24 35 38 44 43 65, so you get 65 if you reverse 4 5 again you get 65. If you reverse 5 6 you get 66. So, the lowest one is your 3 4 reversal and then we have now 1 2 4 3 5 6 7 1 and we have a TD star of 65, so this is what we have got by iteration 1 right. So, basically started arbitrarily with some rout keep on doing reversals and we get a better solution the best one we choose, so that is a first iteration.

(Refer Slide Time: 26:05)

TSP by Sub-Route Reversal Heuristics

	1	2	3	4	5	6	7
1	M	12	10	M	M	M	12
2	12	M	8	12	M	M	M
3	10	8	M	11	3	M	9
4	M	12	11	M	11	10	M
5	M	M	3	11	M	6	7
6	M	M	M	10	6	M	9
7	12	M	9	M	7	9	M

Iteration 2		
Reversal	Route	TD*
Original	1-2-4-3-5-6-7-1	65
Reverse 3-5-6	1-2-4-6-5-3-7-1	64

No further improvement is possible

Hence, Best solution: 1-2-4-6-5-3-7-1; TD* = 64 *Near optimal*

But this is not optimal!

Optimal Solution is: 1-2-4-6-7-5-3-1; TD* = 63
 or 1-3-5-7-6-4-2-1; TD* = 63

34

Now, in the second iteration, now you see this is your original and that is what you got in the last slide, now in the second iteration if you reverse 3 5 6 right if you reverse 3 5 6, so look here this is our 3 5 6; if you reverse then you get 6 5 3 this time it is shown here 1 to 2 2 to 4 4 to 6 right 4 to 6 then 6 to 5 5 to 3 3 to 7 and 7 to 1. So, if you add them up you get 64 right.

So, this time the base solution will becomes 1 to 2 2 to 4 4 to 6 6 to 5 5 to 3 3 to 7 and 7 to 1 and that is 64, so this is our base solution obtained. So, far right but this is still not optimal see the optimal solution is really 1 to 2 to 4 to 6 to 7 to 5 to 3 to 1 with a TD star equals to 63 or there is one more, which is also giving TD star equals to 63. But these improvement, we see I have not shown further iterations. So, further improvements not possible in this particular problem.

So, what happens we have to be happy with a less near optimal, so this is not optimal this is really near optimal. So, all this heuristic method can really give near optimal say quick way they can reach the near optimal and we can work with a very good solution right, so that is the advantage of the heuristics good.

(Refer Slide Time: 28:00).

Solving TSP by Greedy Heuristics

Solve the 5-city TSP for minimizing costs as given in the matrix below:

	A	B	C	D	E
A	M	10	3	6	9
B	5	M	5	4	2
C	4	9	M	7	8
D	7	1	3	M	4
E	3	2	6	5	M

Greedy Heuristics

1. Sort all edges.
2. Select the shortest edge and add it to the tour.
3. It should not create a cycle of less than N edges or increase the degree of any node to more than 2.
4. Do we have N edges in our tour? If not, repeat step 2.

Handwritten diagrams show a graph with nodes A, B, C, D, E. Edges are drawn between nodes, with some labeled as 'Subtour' and others as 'A to B' or 'A to C'. A cycle A-B-C-A is shown, and a path A-B-C is also indicated.

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

So, let us look at one more that is called the greedy heuristics. So, what is the greedy heuristics really is a very simple method, you sort all the edges select the shortest edge and add it to the tour. It should not create a cycle of less than n edges or increasing the increase the degree of any node to more than 2 is alright.

So, do we have n edges in our tour if not repeat step 2. So, really speaking what it does in you sort all the tour and then choose the shortest but then add something in such a manner, so that you know you really have a complete tour sort of things say this given example supposing I have chosen let us say A to B so this is chosen.

Now, can you choose B to A no why because it gives us sub tour this is sub tour this is sub tour not possible right, we have chosen A to B can I choose A to C no because, you know then you see C to A is ok. But A to C is not is alright because, you see you cannot go simultaneously from A to B and A to C what it is basically increase the degree of node more than 2 right.

So, A has to come from somewhere may B C. So, if again A to C then what will happen this you know degree becomes 3, so this is not allowed right also another thing is not allowed not written here we have A to B to C can you have A to C again answer is no right.

Because if you have A to C again this is the same problem because A has to come from somewhere may be D. So, look here this is difficult this is the degree increases more than 3 in other words it will do not get complete tour.

So that means, you sort the edges and take the shortest one, but take the next ones in such a manner that first of all no sub tours are there first then it should not disturb the tour by any other way, that is the way you should proceed, so that is the greedy heuristics.

(Refer Slide Time: 30:35)

Solving TSP by Greedy Heuristics

	A	B	C	D	E
A	M	10	3	6	9
B	5	M	5	4	2
C	4	9	M	7	8
D	7	1	3	M	4
E	3	2	6	5	M

Shortest edge: D-B length 1; Choose this
Chosen D-B

Next shortest edge: B-E or E-B length 2
Say, choose B-E
Chosen D-B-E

Next Shortest edge: E-B length 2
Not possible, creates sub-tour with D-B-E

Greedy Heuristics

- Sort all edges.
- Select the shortest edge and add it to the tour.
- It should not create a cycle of less than N edges or increase the degree of any node to more than 2
- Do we have N edges in our tour? If not, repeat step 2.

So, let us see what we do so shortest edge if you look here is D B. So, take D B right D B length 1. So, very clear that we should choose D B right. Next shortage edge is B E or E B right we can there is no problem let us choose E.

So, B E we get D to B to E next is E B can you choose E B we cannot in that case we make like this. So, you see sub tour is obtained not possible. So, we cannot take E B right E B not possible. So, we got D to B to E right so that is the first.

(Refer Slide Time: 31:23)

Solving TSP by Greedy Heuristics

	A	B	C	D	E
A	M	10	3	6	9
B	5	M	5	4	2
C	4	9	M	7	8
D	7	1	3	M	4
E	3	2	6	5	M

Chosen D-B-E

Next Shortest edge: A-C or D-C or E-A length 3

D-C not possible (D-B chosen) *D-B-E*

So, choose A-C as well as E-A *E-A*

Chosen D-B-E-A-C

	A	B	C	D	E
A	M	10	3	6	9
B	5	M	5	4	2
C	4	9	M	7	8
D	7	1	3	M	4
E	3	2	6	5	M

Now, we have to choose C-D length 7

Because that will make a complete tour




Hence, TSP Feasible solution: *D-B-E-A-C*

A-C-D-B-E-A, TD* = 16

This was our optimal solution in B&B method as well.

Greedy Heuristics

- Sort all edges.
- Select the shortest edge and add it to the tour.
- It should not create a cycle of less than N edges or increase the degree of any node to more than 2
- Do we have N edges in our tour? If not, repeat step 2.

Next again if you go further the what is the next shortage shortest? It is A to C or D to C or E to A right length 3 D C not possible because look here we have already chosen D to B to E can you choose D to C not possible, because then same problem as I have explained earlier not possible. So, we can choose A C obviously, yes we can also choose E A right.

So, we can choose both so when you choose we get already D to B to E to A to C right. Now obviously, we have to choose C to E C to D then there is you know no other option because, we have to make it a close tour. So, we choose that and when you do that you know if you return differently this is a C D B A C and what is the TD? TD will be 3 2 7 1 3 total distance is going to be 16, so this was our optimal solution in B and B method as well is alright.

So, by greedy heuristics also we can quickly find the TSP solution, but once again there is no guarantee that we are going to have the shortest possible path right. It may it could be a complete tour definitely it will be a feasible solution, but there is no guarantee that will be the optimal as well. So, these are the weakness of the heuristic methods and that is how we solve such problems.

So, in our integer programming we have seen the different formulations, different ways of solving integers programming problems, like cutting plane algorithms, branch and

bound algorithms heuristic methods and so on and so forth right. So, in our next lecture we begin new topic on non-linear programming so;

Thank you very much.