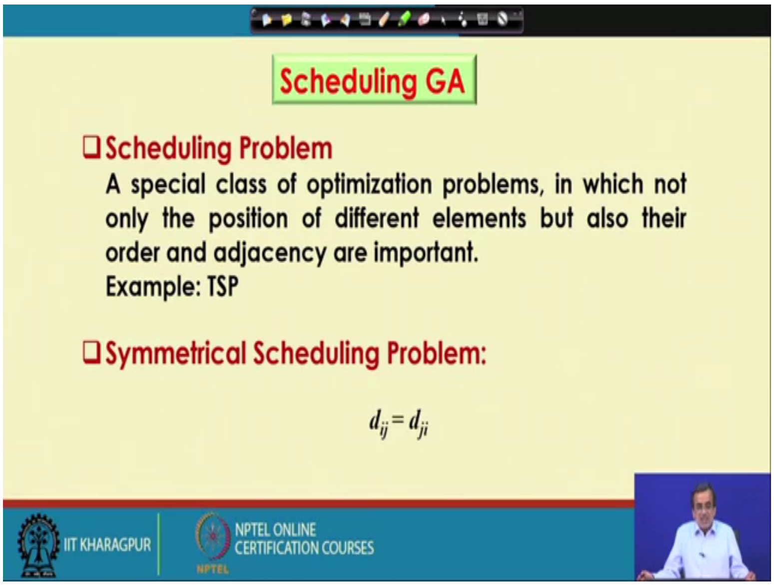
Traditional and Non-Traditional Optimization Tools
Prof. D. K. Pratihar
Department of Mechanical Engineering
Indian Institute of Technology, Kharagpur

Lecture - 17
Scheduling GA

Start.

Now, I am going to start with another topic and that is on scheduling GA now I am just going to discuss in details how to solve the scheduling problem with the help of a genetic algorithm. Now to start with let me tell you that the scheduling optimization problem is slightly different from the ordinary optimization problem like the optimal design of gearbox, optimal design of spring and so on. Now this scheduling problem is slightly different and more complex now let me try to find out the reason behind this complexity of the scheduling problem.

(Refer Slide Time: 01:03)



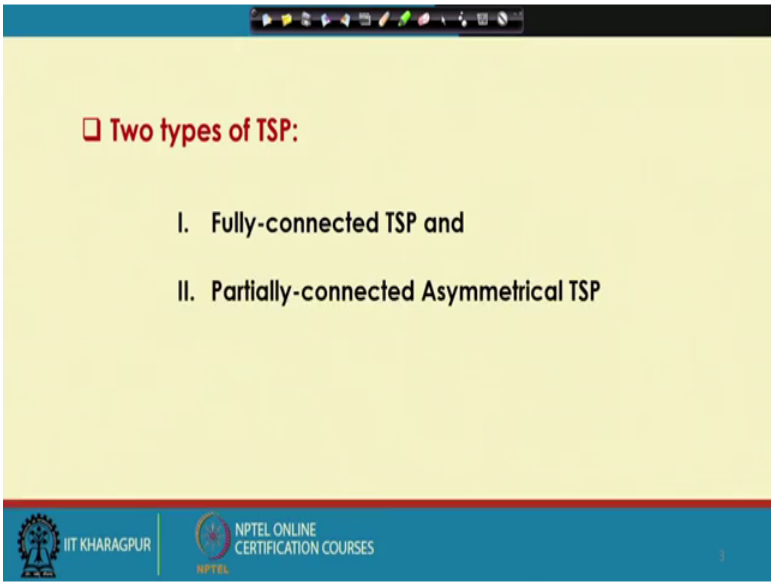
Now, what you do in scheduling problem, we try to find out the position and at the same time we try to find out the order and adjacency of the different elements in scheduling problem. Now let me take a very simple example, supposing that there is a travelling sales person and there are n cities, now the sales person will have to touch all n cities once by covering the minimum distance. So, this is a typical example of a scheduling problem.

Now, another example I can take supposing that. So, we will have to find out the class schedule of an institute that is nothing, but a scheduling problem or say jobs of scheduling problem is another example. Now this type of optimization problem are slightly more difficult compared to the ordinary optimization problem like the example I took like the optimal design of gearbox and so on.

Now, if I consider that type of optimization problem where only we consider the position or where we consider only the numerical values, but their order is not important. So, that type of optimization problem if I call, that this is a scalar optimization problem then scheduling optimization problem is nothing, but a vector optimization problem. And as I told that scheduling problem is much more difficult compared to the ordinary optimization problem, and that is why we will have to take some care to solve. So, this type of optimization problem which is much more difficult using a genetic algorithm, and I am just going to discuss that to solve this type of optimization problem. So, we will have to use some special type of operators those things I am going to discuss in details.

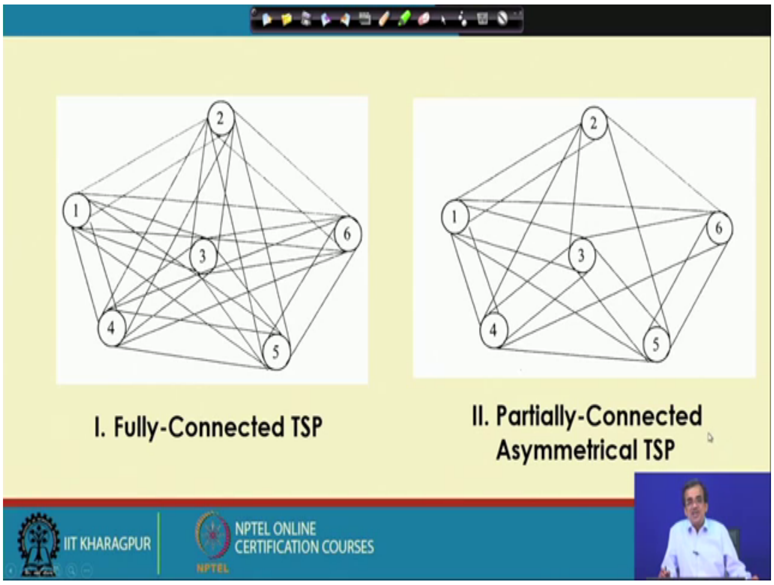
Now, if you see the scheduling problem, the scheduling problem could be either symmetrical or it could be asymmetrical now let me take one example.

(Refer Slide Time: 03:39)



Now this is one scheduling problem where a particular city is connected to all other cities in both the directions.

(Refer Slide Time: 03:41)

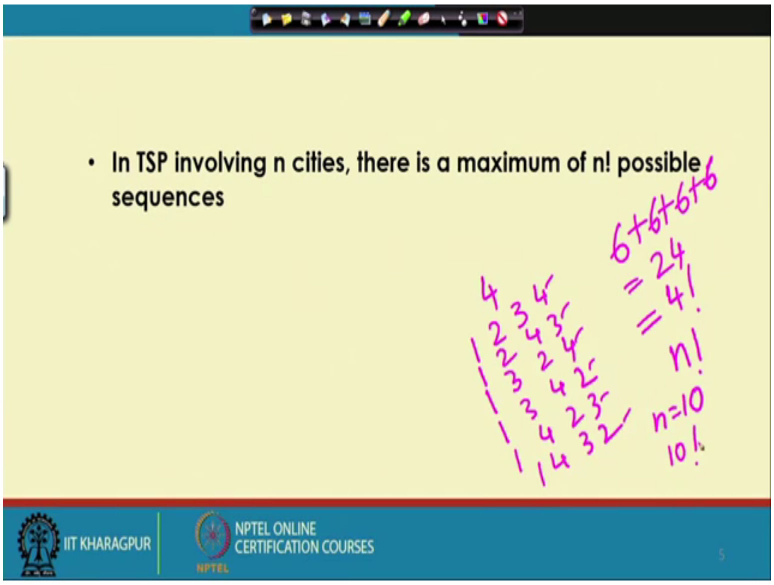


So, this a fully connected TSP and this is also a symmetrical TSP provided the distance between 1 and 3 becomes equal to the distance between 3 and 1. So, let me repeat. So, here this is a fully connected TSP and this can be called symmetric, if and only if the distance between 3 and 4 is equal to the distance between 4 and 3.

Now, on the other hand we have got the partially connected TSP, where some of the connectivity is missing. For example, here there is a connectivity between 2 and 6, but there is no connectivity between 6 and 2. So, the connectivity is there only in one direction. Now this type of TSP is known as the partially connected TSP and as it is partially connected it has to be asymmetrical TSP. On the other hand the fully connected network it could be either symmetric or asymmetric, but this partially connected TSP has to be asymmetrical TSP. So, this is the way actually we classify the scheduling problem or the travelling salespersons problem.

Now, let me take one example let us try to see the complexity of this type of problem, and let me repeat the statement of the problem that there are n cities and a particular travelling sales person will have to touch all n cities ones by covering the minimum distance, how to find out the optimal schedule so that he or she can reach all the cities once by covering the minimum distance. Now to understand the complexity let me take a very simple example, supposing that I have got only 4 cities.

(Refer Slide Time: 05:58)



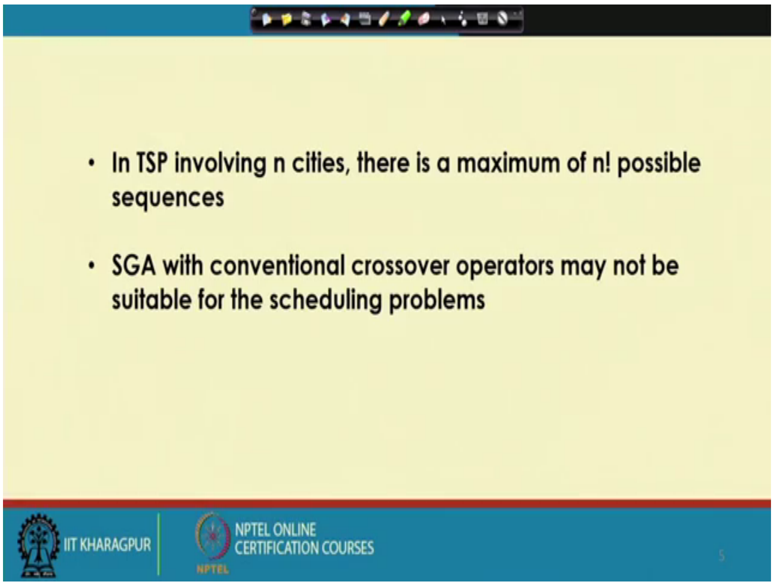
So, there are 4 cities 1 2 3 and 4, now out of these 4 cities I have decided that I am going to start with city 1.

Now, if I start with city 1 next it could be city 2, city 3, city 4 another possibility could be I will start with one then 2 4 3 another possibility I will start with 1 then 3 2 4 anther is 1 3 4 2 another is one 4 2 3, 1 4 3 2. So, if I start with city one. So, how many such possible sequences are there? 12 3 4 5 6. So, there are 6 possible sequence. Now if I start with city 1 there are 6 sequences similarly if I start with city 2, I will have 6 more similarly if I start with city 3, 6 more if I start with city 4 there will be 6 more, total we have got 24 possible sequences.

Now, the optimizer will have to find out the best sequence. Now what we do is, this 4 is nothing, but this 24 is nothing, but 4 factorial now. So, if we have got n such cities. So, I have got n factorial possible sequences, now GA will try to find out, out of these n factorial sequences the best one now let me assign some numerical value supposing that we have got n equals to 10 there are 10 cities. So, if there are 10 cities. So, there will be 10 factorial possible sequences and it is a huge number and GA will have to find out the best one out of all the possibilities and it will be a difficult problem for the GA.

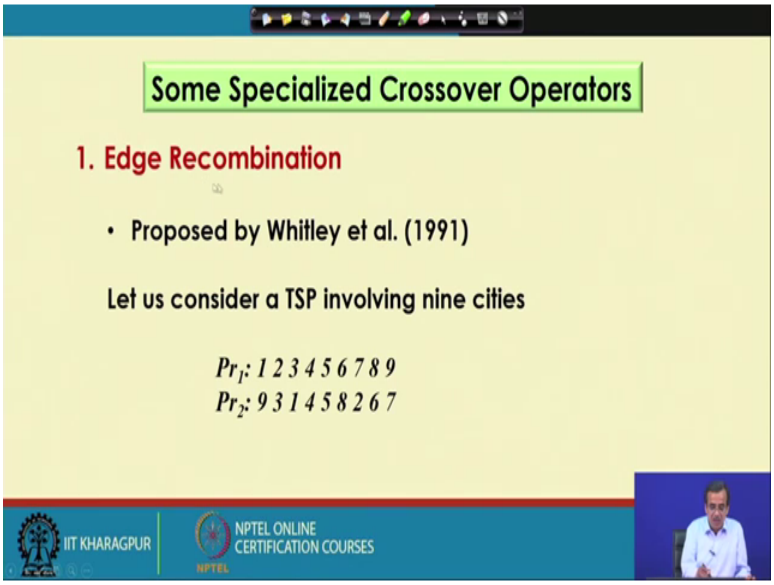
And moreover there is another problem in this type of optimization the problem, that problem is your like there is a possibility that if we use the conventional operators like the single point cross over, 2 point cross over, multipoint cross over or uniform crossover we may get some children solution which could be invisible; might be the same city will be ah copied more than once or there could be some missing city. So, that type of children solution. So, will be getting and that is why to solve this type of optimization problem. So, will have to use some special type of crossover operator and all such things I am going to discuss in much more details.

(Refer Slide Time: 09:16)



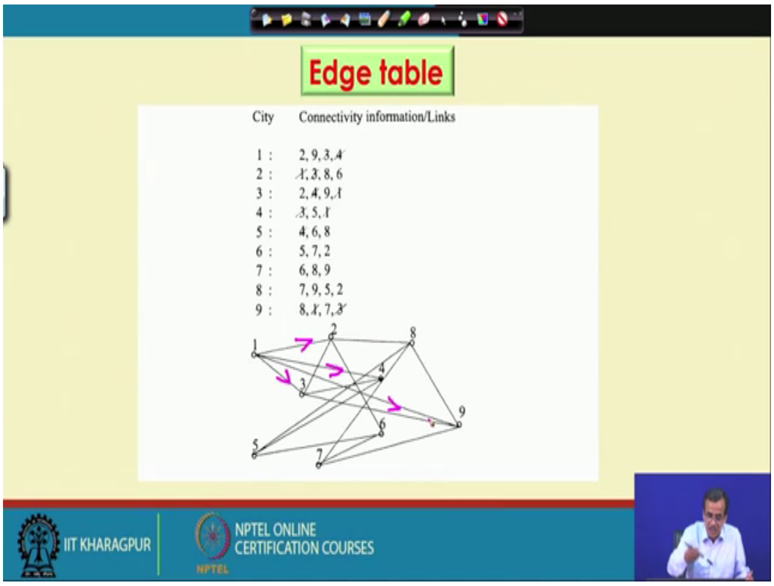
So, as I told that SGA that is simple genetic algorithm with the conventional crossover operator is not going to sub the purpose for solving the scheduling problem and that is why. So, we will have to go for some specialized crossover operators.

(Refer Slide Time: 09:30)



Now let me start with the first operator that is edge recombination. So, that was proposed in the year 1991 by Whitley and others now supposing that we have got 2 parents like parent one and parent 2 and I am just going to consider a TSP involving 9 cities. So, there are 9 factorial possibilities and GA will have to find out the best one. Now supposing that parent one is 1 2 3 4 5 6 7 8 9 and parent 2 is 9 3 1 4 5 8 2 6 7 now using this particular the 2 parents. So, how to find out the children solution, using the principle of edge recombination that I am going to discuss.

(Refer Slide Time: 10:33)

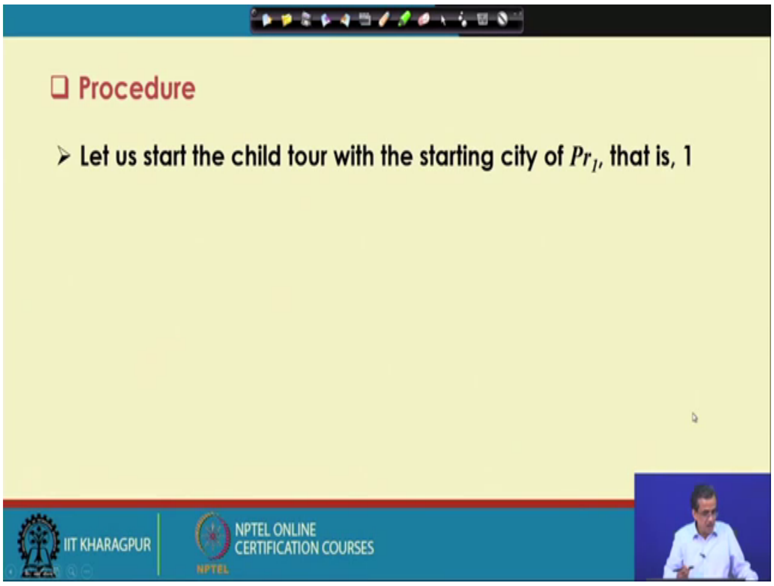


Now, here we will have to depend on the edge table.

Now, edge table provides information of each city and its connectivity. For example, say city 1 is connected to city 2 9 3 and 4, city 2 is connected to 1 3 8 and 6 3 is connected to 2 4 9 1 and so on and these shows actually the connectivity graph. For example, say one is connected to 2. So, one is connected to 2, one is connected to 2, then one is connected to 9. So, one is directly connected to 9 then one is connected to 3. So, it is connected to 3 and one is connected to 4. So, one is connected to 4, similarly for the other cities I can draw the connectivity and I will be getting this connectivity graph and truly speaking this is actually partially connected asymmetrical TSP sort of thing.

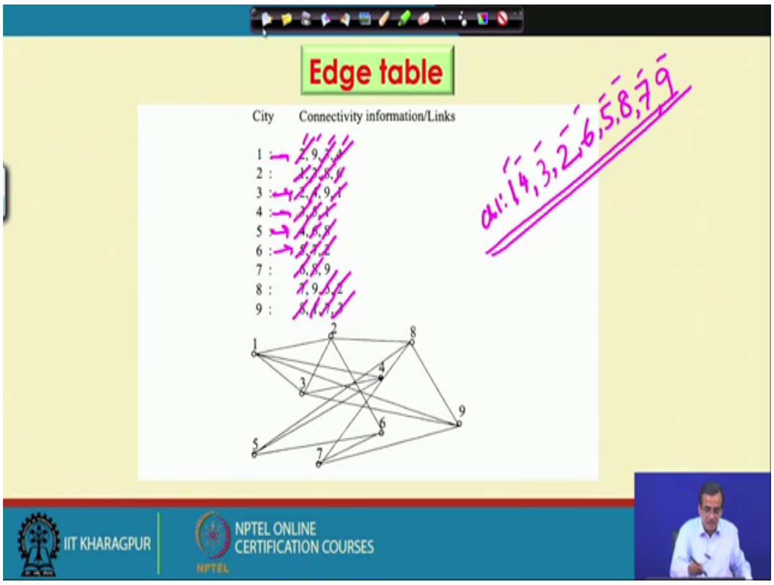
Now, if this is the situation and corresponding to the 2 parents, which have already ah mentioned like how to find out these the child solution.

(Refer Slide Time: 11:58)



Now let us see the parent one once again. Now parent one start with city one. So, I am just going to start with city 1 if I want to find out the child one. Now let us see how to get child one starting from city 1 and that is the starting city of parent one also. So, let us try to find out what should be the child one and child 2. Now let me just find out the child one and child 2 like this.

(Refer Slide Time: 12:37)



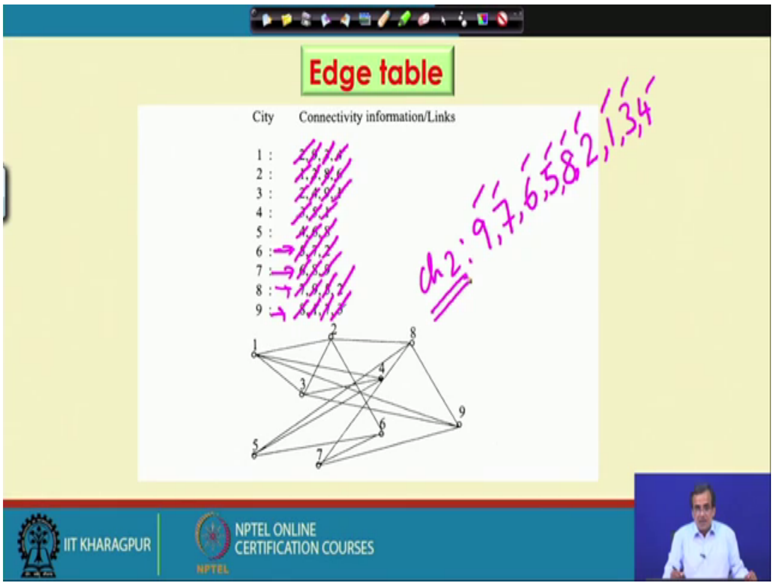
So, child one as I told I will going to I am going to start with city one, and according to this particular the edge recombination principal, once I have selected city 1 the all the entries of city 1 should be removed from the right hand side of the edge table. So, one I have selected. So, let me remove one here, let me remove one here, one here and one here. Now city 1 has been selected now one is connected to 2 9 3 and 4 and let us see the connectivity of 2 9 3 and 4.

Now, city 2 is now connected to 3 cities, because one has been removed then 3 is connected to 3 cities then 4 is connected to 2 cities and 9 is connected to 3 cities; that means, I will have to select 4 because 4 is connected to less number of cities that is 2 compared to the others now if I compare the 2 3 4 and 9, which one to be selected next. So, out of these 4 I will have to select 4 because 4 is connected to only 2 cities that is 3 and 5. So, let me select 4 here and once I have selected 4 all the entries of 4, I will have to remove from this particular the edge table. So, 4 is removed, 4 is removed, 4 is removed and there is no other 4 here. And once I have selected 4 now you see 4 is connected to 3 and 5 and 3 is connected to 2 cities 5 is connected to 2 city. So, there is a tie; that means, out of this 3 and 5 anyone I can select. So, let me select 3 here and if I select 3 now all the entries of 3 will have to remove. So, 3 is removed here, 3 is removed here, and this particular 3 is removed here.

Now, 3 is connected to 2 cities that is 2 and 9, 2 and 9 now 2 is connected to 2 cities and 9 is also connected to 2 cities. So, any one you take let us take 2 now if I take now 2 has to be removed, 2 is removed, 2 is removed, 2 is removed, 2 is removed from here now 2 is connected to 2 cities 8 and 6, 6 is connected to 2, 8 is connected to 3. So, I will have to select 6 because 6 is connected to 2 cities only. So, let me select 6 and once I have selected 6 you remove 6 from here, remove 6 from here, remove 6 from here. So, all the entries of 6 have been removed. Now 6 is connected to 2 cities 5 and 7 5 is connected to one 7 is connected to 2. So, I will have to select 5. So, let us select 5 here. So, if 5 is selected this 5 has to be removed, this 5 has to be removed, this 5 has to be removed and once I have removed the 5 now you look into 5. So, 5 is connected to 8. So, there is only one option. So, you select 8. So, once 8 has been selected you remove 8, 8, 8, 8 and now 8 is connected to 7 and 9, 7 is connected to 9, 09 is connected to 7 only one. So, any one we can take. So, let me take 7 and once I have selected 7, you remove 7 from here, here and here and 7 is connected to9. So, 9 is selected.

So, this is child one now, you check whether all the cities have come or not. So, 1 2 3 4 5 6 7 8 9. So, all the cities have been consider once and there is no missing city. So, this particular child solution is a valid child solution. Now similarly I can find out the child 2 also let me try to find out the child 2 also.

(Refer Slide Time: 17:46)



Now, I am just going to find out child 2. So, child 2 and the starting city for the child 2 will be 9 that is the starting city for parent 2 as I discuss. Now once I have selected 9 all the entries of 9 will have to remove from here. So, 9 is removed 9 is removed 9 is removed, 9 is removed, now 9 is connected to 8 1 7 and 3. Now you see 8 is connected to 3, 7 is connected to 2, 1 is connected to 3 and 3 is connected to 3. So, 7 is connected to 2 only. So, 7 is connected to 2 that is 6 at 8. So, you select 7 here and once you have selected 7 all the entries of 7 have to be removed, 7 is removed, 7 is removed, 7 is removed there is no other 7.

Now, 7 is connected to 6 and 8, 6 is connected to 2, 8 is connected to 2. So, there is a tie any one you take let me take 6. Now if I take 6, this 6 I will have to remove this 6 is removed, removed. So, there is no other entries of 6 here. Now 6 is connected to 5 and 2, 5 is connected to 2 cities and 2 is connected to 3 city. So, I will have to select 5. So, let me select 5 here. So, once 5 is selected remove 5, remove 5, remove 5, now 5 is connected to 4 and 8, 4 is connected to 2 cities 8 is connected to only one city. So, you select 8 here now all the entries of 8 you remove, 8 is removed, 8 is removed, 8 is removed, 8 is removed now you concentrate on 8, 8 is connected to 2 only 1. So, you take 2 and you remove 2 here, 2 here, 2 here, 2 here and once I have selected 2. Now see 2 is connected to 1 and 3, one is connected to 2 cities and 3 is connected to 2 cities. So, there is a tie you take any one. So, let me consider opted to let me consider 1. So, 1 is removed from here one is removed, one is removed, and here also one is removed.

Now, one is connected to 3 and 4, 3 is connected to only, 4 is connected to only 1. So, any one I can select. So, let me select 3. So, 3 is removed from here is removed from here, 3 is removed from here, 3 is removed from here, now 3 is connected to 4. So, 3 is connected to 4. So, 4 is removed, removed, removed that actually completes these particular the search. Now we can find out this is 1 2 3 4 5 6 7 8 9. So, all the 9 cities have been considered once and no city has been repeated there is no missing city also. So, this is nothing, but a valid child. So, using this particular, the principal of edge recombination. So, we can find out 2 valid children corresponding to the 2 parents.

Thank you.