Artificial Intelligence: Search Methods for Problem Solving
Prof. Deepak Khemani
Department of Computer Science and Engineering
Indian Institute of Technology, Madras

Chapter – 05
A First Course in Intelligence
Lecture – 34
Finding Optimal Paths

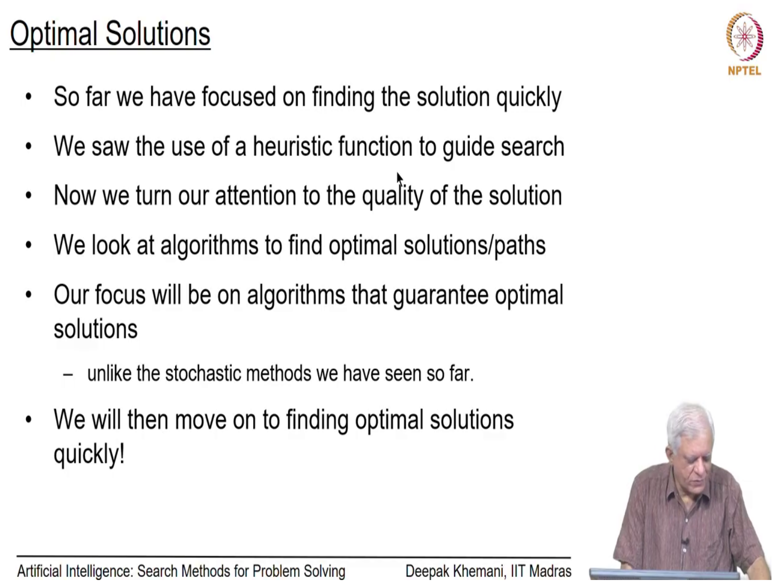
So welcome back, so far in our course we have focused on finding solutions and we found that in most search spaces the problem is of combinatorial explosion and exponential growth. So, we have focused on using heuristic functions to find solutions as quickly as possible and then we looked at this algorithm best first search.

We found that even that was not good enough in many problem domains. So, we investigated local search, then we looked at stochastic local search, then we looked at population based methods. Now, we come back to the problem of finding optimal solutions.

So, far we have not focused on finding optimal solutions, we have focused on finding solutions as quickly as possible using the heuristic function. Now, we want to look at methods which will give us optimal solutions, because in many problems you want to generate optimal solutions where for example, if you are planning to let us say colonize Moon or Mars, then you know each journey time is very expensive and it does not matter how much time you spend looking for an optimal solution; but an optimal solution would be lot of savings.

So, there are many domains where optimal solution is of utmost importance and there we would need to use some of the algorithms that we are going to study now onwards, ok. So, this is from my textbook, the material can be found in this chapter 5 on Optimal Paths.

(Refer Slide Time: 02:00)



Optimal Solutions

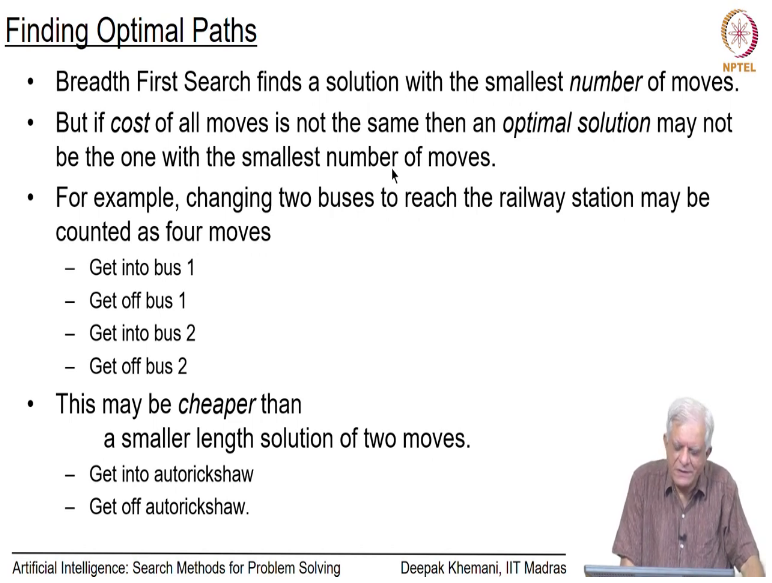
- So far we have focused on finding the solution quickly
- We saw the use of a heuristic function to guide search
- Now we turn our attention to the quality of the solution
- We look at algorithms to find optimal solutions/paths
- Our focus will be on algorithms that guarantee optimal solutions
  - unlike the stochastic methods we have seen so far.
- We will then move on to finding optimal solutions quickly!

Artificial Intelligence: Search Methods for Problem Solving   Deepak Khemani, IIT Madras

So, far as we have said I just said we have focused on finding the solution quickly and we saw the user heuristic function to guide search and now we turn our attention to the quality of the solution that we will find. We look at algorithms to find optimal solutions or paths and our focus will be on algorithms that guarantee optimal solutions.

Now means, we said that things like genetic algorithms and colony optimization can often give you optimal solutions, but it was not a guaranteed method. Here we are looking for solutions or algorithms which are guaranteed to give you optimal solutions. And then we will move on to finding those optimal solutions quickly something like using a heuristic function as we did earlier.

(Refer Slide Time: 02:50).


Finding Optimal Paths

- Breadth First Search finds a solution with the smallest *number* of moves.
- But if *cost* of all moves is not the same then an *optimal solution* may not be the one with the smallest number of moves.
- For example, changing two buses to reach the railway station may be counted as four moves
  - Get into bus 1
  - Get off bus 1
  - Get into bus 2
  - Get off bus 2
- This may be *cheaper* than a smaller length solution of two moves.
  - Get into autorickshaw
  - Get off autorickshaw.

Artificial Intelligence: Search Methods for Problem Solving Deepak Khemani, IIT Madras

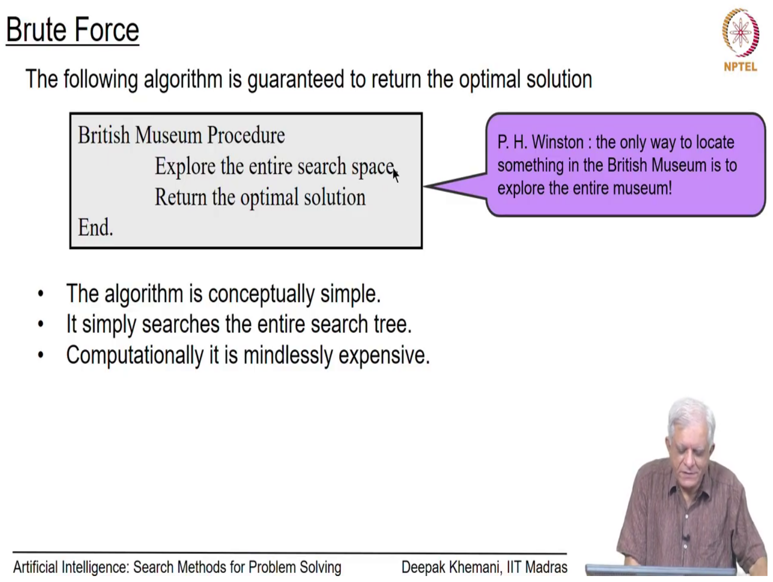
So, let us look at this problem of finding optimal paths. The only place where we have spoken about optimal solutions is breadth first search and in breadth first search in some sense you assume that every edge has the same cost or unit cost. And therefore the optimal solution is the one which has the smallest number of moves and we know that if that is the case then breadth first search thus guarantee you an optimal solution.

But if the cost of moves is not the same, so different moves may have different costs associated then the optimal solution may not be just a number of moves, but the total sum of the cost of each individual actions. For example, if you want to go to the railway station from wherever you are staying there may be a solution which says that you take 2 buses to go to the station.

You take get into bus 1 get off bus 1 at some point get into bus 2 and get off bus 2 you can think of this as 4 moves essentially. Now, this is likely to be cheaper than a shorter solution which is of only 2 moves in which says that you get into an auto rickshaw and then it takes you to the station and you get off at the auto rickshaw.

Especially if you live in a city like Chennai you will find that buses are much more cheaper than autos. So, our goal is to now find solutions whose total cost is the minimum not the number of moves essentially.

(Refer Slide Time: 04:26)



Brute Force

The following algorithm is guaranteed to return the optimal solution

```
British Museum Procedure
        Explore the entire search space
        Return the optimal solution
End.
```

P. H. Winston : the only way to locate something in the British Museum is to explore the entire museum!

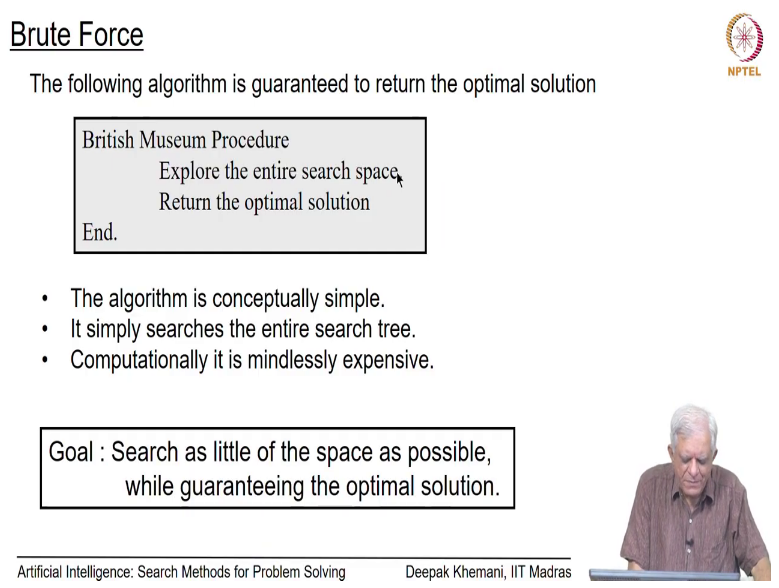
- The algorithm is conceptually simple.
- It simply searches the entire search tree.
- Computationally it is mindlessly expensive.

Artificial Intelligence: Search Methods for Problem Solving Deepak Khemani, IIT Madras

Now, to find an optimal solution one thing you could do is to use brute force and the following algorithm is called the British Museum procedure. It simply says that explore the entire search space and then return the optimal solution found. Obviously, this will give us optimal solution because it is a brute force algorithm. Conceptually, it is very simple it simply searches the entire space, but computationally it is mindlessly expensive.

Now, Patrick Henry Winston who was one of the pioneers of AI and who wrote some of the earliest books in AI he mentioned that the only way to locate something in the British Museum is to explore the entire museum and that is why he called it the British Museum procedure.

(Refer Slide Time: 05:16)



Brute Force

The following algorithm is guaranteed to return the optimal solution

```
British Museum Procedure
        Explore the entire search space
        Return the optimal solution
End.
```

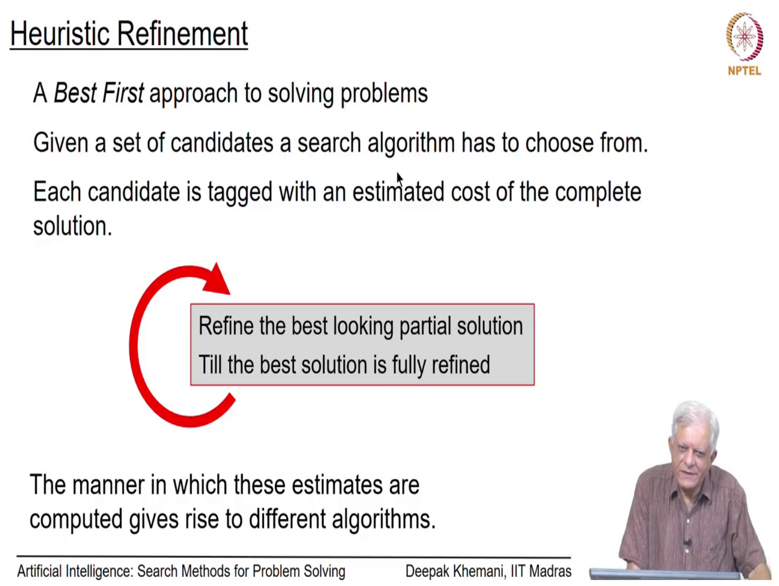
- The algorithm is conceptually simple.
- It simply searches the entire search tree.
- Computationally it is mindlessly expensive.

Goal : Search as little of the space as possible,
while guaranteeing the optimal solution.

Artificial Intelligence: Search Methods for Problem Solving   Deepak Khemani, IIT Madras

But we are going to be interested in searching as little as possible while guaranteeing that you get in get an optimal solution. So, but first we will focus on finding the optimal solution and then we will focus on finding them quickly essentially.

(Refer Slide Time: 05:35)



Heuristic Refinement

A *Best First* approach to solving problems

Given a set of candidates a search algorithm has to choose from.

Each candidate is tagged with an estimated cost of the complete solution.

Refine the best looking partial solution
Till the best solution is fully refined

The manner in which these estimates are computed gives rise to different algorithms.

Artificial Intelligence: Search Methods for Problem Solving Deepak Khemani, IIT Madras

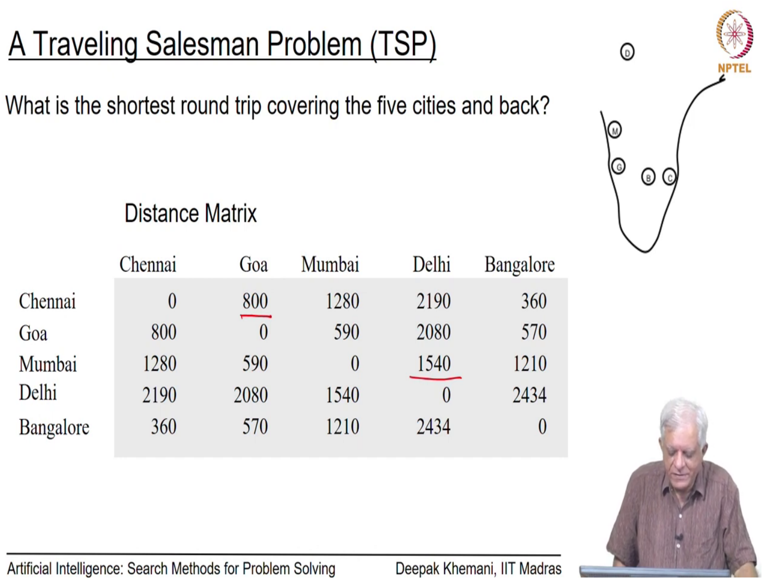
So, our general algorithm for search is now hopefully becoming clearer, in some sense we can say that our search algorithm is the best first approach to solving problems and we have a set of candidates that algorithms has to choose from we have called these candidates as the set open.

And now we will say that each candidate is associated with an estimated cost of the complete solution the complete solution. And somehow you estimate that how much will that candidate if you refine it completely, how much will that solution cost and that is the estimated cost of a solution. And what heuristic search does in general is to refine the best looking partial solution till the best solution is fully refined and we will expand upon this process.

Now with the restriction or with the definition that best means the least cost partial solution that you have found so far essentially. The manner in which these estimates of these solutions are generated varies and they give rise to different algorithms. In fact, we can see breadth first search depth first search best first search all as variations of this, each of them thinks of the estimated costs in a slightly different way and therefore the algorithm performs differently.

But at all points you can say that we have seen that we maintain an open list of some sort and we always remove the node from the head of the open list. In the case of best first search open was maintained as a priority queue. So, that always a least heuristic function value node was picked up.

(Refer Slide Time: 07:18)



A Traveling Salesman Problem (TSP)

What is the shortest round trip covering the five cities and back?

Distance Matrix

	Chennai	Goa	Mumbai	Delhi	Bangalore
Chennai	0	800	1280	2190	360
Goa	800	0	590	2080	570
Mumbai	1280	590	0	1540	1210
Delhi	2190	2080	1540	0	2434
Bangalore	360	570	1210	2434	0

Artificial Intelligence: Search Methods for Problem Solving   Deepak Khemani, IIT Madras

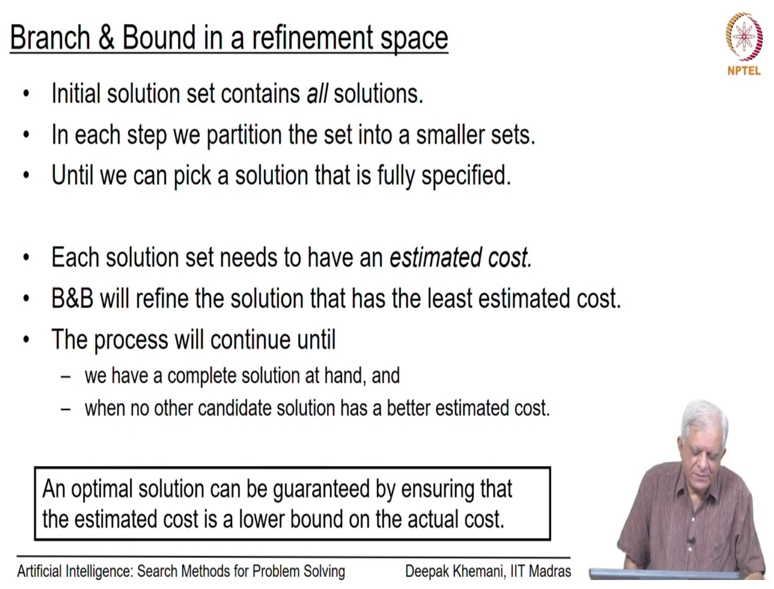
Let us start with looking at the Traveling Salesman Problem which is a problem that we have been studying recently using the stochastic methods and we looked at genetic algorithms we looked at ant colony optimization. But now we want to look at a method which will guarantee you the optimal solution.

And so let us look at this algorithm in the context of a small problem in which there are these 5 cities Chennai, Goa, Mumbai, Delhi and Bangalore and you want to visit all these cities exactly once and come back to the origin city whichever it may be.

So, a tour will basically be a tour over all these 5 cities it will have 5 segments of travel and each will terminate in 1 city and these are the costs that I have assumed here. These are rough estimates and we will just use this as a working example to show how the algorithm that we are looking at will work essentially. So, this matrix shows the cost for example, the cost between Chennai and Goa is 800, the cost between Mumbai and Delhi is 1540 and so on essentially.

So, we want to find a tour which is of the least cost solution possible. Of course, this is a small problem and it may seem that it is not so hard to find the tour. But for larger problems we know that the TSP problem is as complexity which is factorial which is in fact more than the exponential growth problems that we normally worry about.

(Refer Slide Time: 09:01)



Branch & Bound in a refinement space

- Initial solution set contains *all* solutions.
- In each step we partition the set into a smaller sets.
- Until we can pick a solution that is fully specified.

- Each solution set needs to have an *estimated cost.*
- B&B will refine the solution that has the least estimated cost.
- The process will continue until
  - we have a complete solution at hand, and
  - when no other candidate solution has a better estimated cost.

An optimal solution can be guaranteed by ensuring that the estimated cost is a lower bound on the actual cost.

Artificial Intelligence: Search Methods for Problem Solving Deepak Khemani, IIT Madras

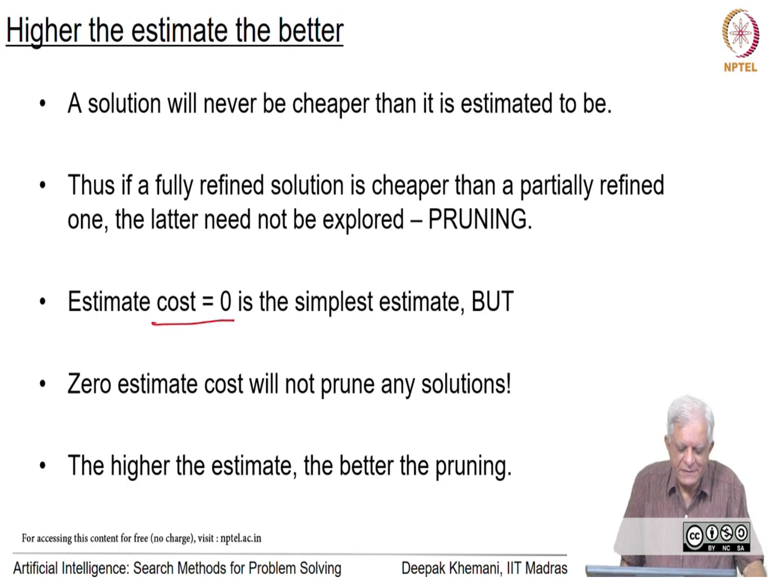
So, the algorithm that we are going to talk about is called branch and bound and we are looking at in the refinement space, because we are looking at this solving the TSP problem in the land space or the solution space. The way it works is that we initially construct a set which contains all possible solutions and in each possible each step of the algorithm we partition the set into smaller sets essentially, until we pick up a solution that is fully specified or which is complete essentially.

Now, each solution in our set or open list if you want to call it or open set has to have an estimated cost and the algorithm branch and bound B and B will refine the solution that has the least estimated cost. The process will continue until we have a complete solution at hand and when no other candidate solution has a better estimated cost.

So, we will clarify on this notion of better and estimated and we will also discuss in detail how do we guarantee even when we are working with estimated cost. We will see that an optimal solution can be guaranteed by ensuring that the estimated cost is lower than the actual cost.

So, if I have some partial solution and I want to complete it to a full solution, then the estimated cost of the partial solution will be actually lower than the actual cost. So, we are our estimates are always going to be lower and we will see and later on when we look at an algorithm called a star, we will also formally prove that when this condition is satisfied you always find the optimal solution.

(Refer Slide Time: 10:49)



Higher the estimate the better

- A solution will never be cheaper than it is estimated to be.
- Thus if a fully refined solution is cheaper than a partially refined one, the latter need not be explored – PRUNING.
- Estimate cost = 0 is the simplest estimate, BUT
- Zero estimate cost will not prune any solutions!
- The higher the estimate, the better the pruning.

For accessing this content for free (no charge), visit : nptel.ac.in

Artificial Intelligence: Search Methods for Problem Solving   Deepak Khemani, IIT Madras

Another thing we will study is that the higher the estimate the better essentially and this would be in the interest of finding the solution quickly. We will assume for the time being, you should take my word for it that if the estimate it cost is lower than the actual cost, which means that the estimate is an under estimate then you will guarantee to find you are guaranteed to find a solution.

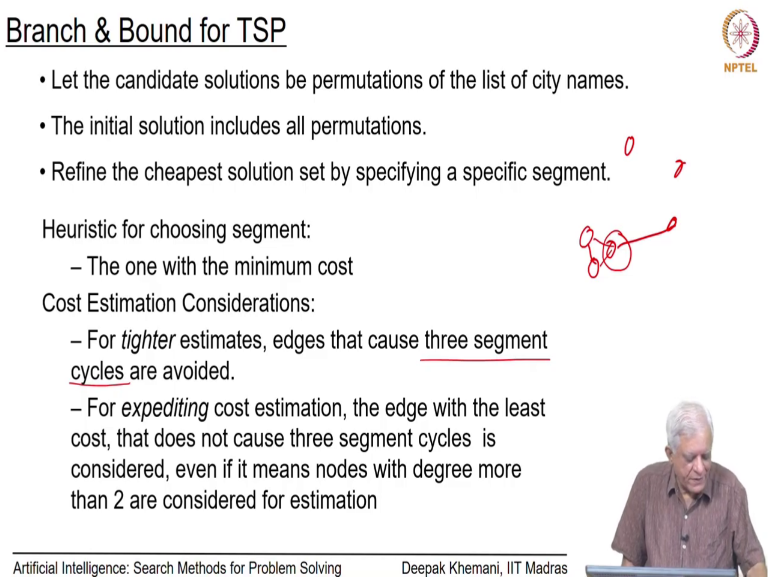
But the other feature that we again we will prove this formally, but for the moment you just have to assume that this is true that the higher the estimate the better in the sense of finding the solution faster essentially.

Now, we have said that our estimates are lower estimate. So, a solution will never be cheaper than it is estimated to be, thus if we have a fully refined solution and it is cheaper than all the other partial solutions in terms of their estimated cost. We know that the estimate those partial solutions will be more expensive than this fully refined solution, because it is already cheaper and those are lower bounds for this thing.

So, we can prune all those partial solutions we do not have to explore them and we can be sure that the fully refined solution that we have is cheaper. Now obviously, the simplest under estimate is to say that cost equal to 0 and in some sense this is what breadth first search does is that it simply assumes that you know the future work that you have to do is 0 essentially.

But the problem with that is that it will not prune any solutions and that is why we know that breadth first search will do very exhaustive kind of search before terminating with the solution. So, our goal is also to see that the higher the estimate and the better will be the pruning essentially.

(Refer Slide Time: 12:48)



Branch & Bound for TSP

- Let the candidate solutions be permutations of the list of city names.
- The initial solution includes all permutations.
- Refine the cheapest solution set by specifying a specific segment.

Heuristic for choosing segment:
- The one with the minimum cost

Cost Estimation Considerations:
- For *tighter* estimates, edges that cause three segment cycles are avoided.
- For *expediting* cost estimation, the edge with the least cost, that does not cause three segment cycles is considered, even if it means nodes with degree more than 2 are considered for estimation

Artificial Intelligence: Search Methods for Problem Solving Deepak Khemani, IIT Madras

So, let us explore this algorithm branch and bound for the Traveling Salesman Problem. Let the candidate solutions be permutations of all lists of city names we can represent in the path representation, we have seen that candidate is simply a permutation of all possible cities.

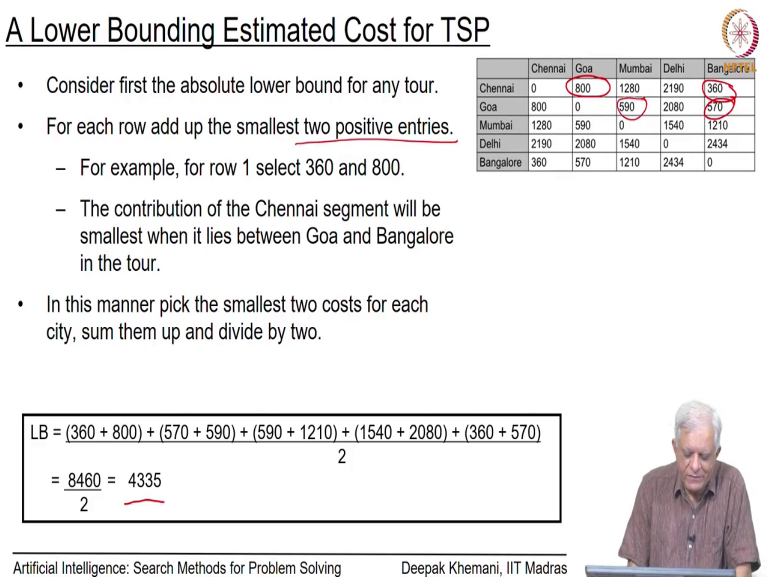
The initial solution will contain initial set of solutions will contain all possible permutations and we will refine the cheapest set from our open set the cheapest solution by specifying one segment and this is what we mean by refinements here that we will keep adding one more segment to our solution and then look at the estimated cost and see which way we should be going.

So, what should be the heuristic for choosing a segment? Obviously like in many algorithms like the greedy algorithm that we saw you choose the one with the minimum cost. So, keep adding edges which have low cost and as far as estimation is concerned we want tighter estimates by tighter we mean higher estimates. So, we should ignore costs where there are three segments in a cycle.

So, if there are three cities which are very close to each other, then obviously those three edges would be of very low cost. But we know that if there are other cities in the play then these 3 edges cannot be part of the solution, because you know there is a smaller cycle. So, we should avoid estimating such edges where 3 edges form a triangle.

So we will avoid those edges, but some but for the sake of doing this faster we will allow for the moment situations where more than 3 edges are forming part of the estimate essentially. And this is simply for the objective of doing it faster, if you have an efficient reasoning algorithm you could prune away those estimates also.

(Refer Slide Time: 15:05)



A Lower Bounding Estimated Cost for TSP

- Consider first the absolute lower bound for any tour.
- For each row add up the smallest two positive entries.
  - For example, for row 1 select 360 and 800.
  - The contribution of the Chennai segment will be smallest when it lies between Goa and Bangalore in the tour.
- In this manner pick the smallest two costs for each city, sum them up and divide by two.

	Chennai	Goa	Mumbai	Delhi	Bangalore
Chennai	0	800	1280	2190	360
Goa	800	0	590	2080	570
Mumbai	1280	590	0	1540	1210
Delhi	2190	2080	1540	0	2434
Bangalore	360	570	1210	2434	0

$$LB = \frac{(360 + 800) + (570 + 590) + (590 + 1210) + (1540 + 2080) + (360 + 570)}{2}$$

$$= \frac{8460}{2} = 4335$$

Artificial Intelligence: Search Methods for Problem Solving　　Deepak Khemani, IIT Madras

So, let us look at this process and so this is our matrix which gives us the distances between different cities. And the first thing we want to do is to find an estimate of the solution of all possible solutions, what is the lowest possible cost that we can think of which will be a lower bound which will be or what is the lower bounding estimate of this all the solutions so that the estimate will be lower than all possible solutions?

Later on we will see that we want this estimate to be as high as possible, but it should be lower than all possible solutions. So, because our initial set in refinement will contains the set of all possible solutions.

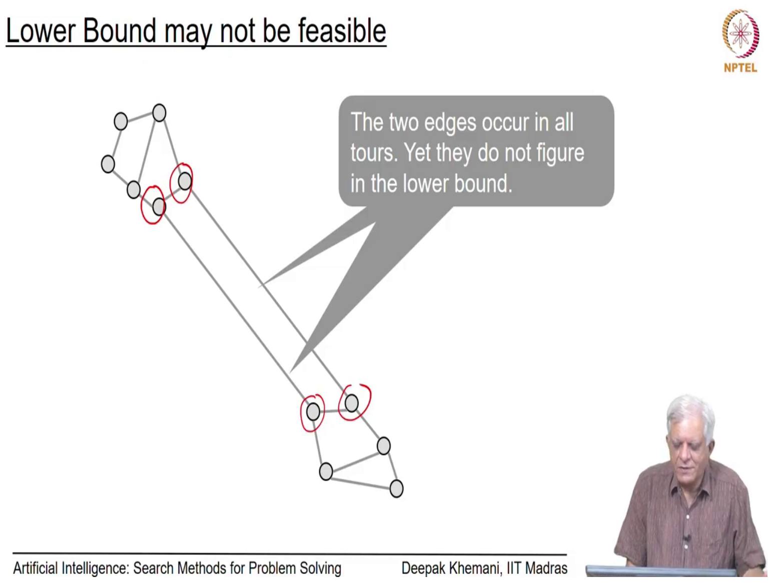
So, let us see how we can compute that. This is a array that we have and the algorithm for doing that is as follows that for each row in this matrix you add up the 2 smallest positive entries essentially, why is this? Because in the best case for every city you will comes come in through the smallest edge and go out through the next smallest edge.

So, the cost of traversing those cities can never be lower than that. So, for example, if you want to you are considering the city Chennai, then you can see that the two smallest edges are 360 and 800, possibly in a optimal solution this will be part of the solution. So, we put that as part of the estimate and observe that it estimates cannot be lower than that.

We do this for every city, for every city we pick the smallest 2 segments. So for example, this 2 here and so on and then we sum them up. So, we have 10 segments because for each city we are coming in from 1 segment and going out through another segment and so essentially we should divide this by 2 because our solution is going to have 5 segments.

So, you can see that given this cost matrix that we have the estimated the lower bounding estimate of all possible tours is 4335 kilometres this is in kilometres I have not mentioned it here, but no tour can be smaller than 4335.

(Refer Slide Time: 17:32)



Lower Bound may not be feasible

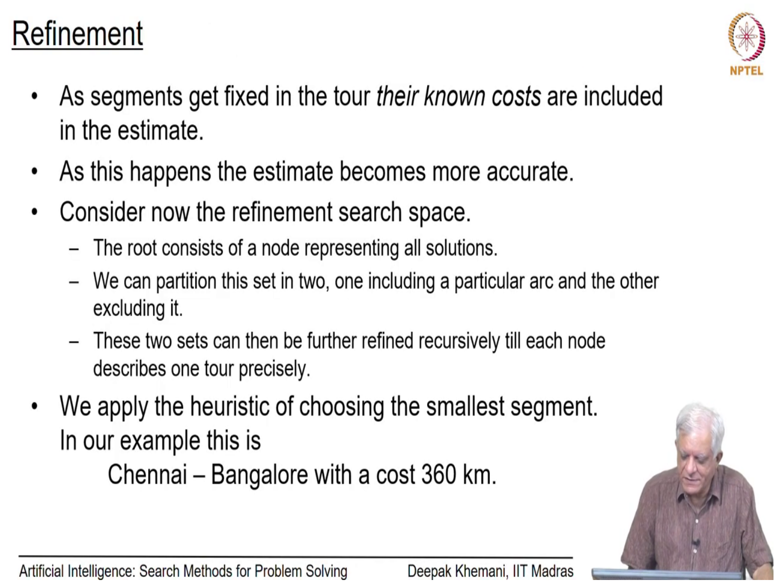
The two edges occur in all tours. Yet they do not figure in the lower bound.

Artificial Intelligence: Search Methods for Problem Solving Deepak Khemani, IIT Madras

Now let us look at how to do branch and bound on this. Now, notice that these lower bound estimates may not be feasible in fact. If you see this diagram here there are two very long edges in this map and these two long edges will not figure in any lower bound estimate.

Because for each of the cities that they touch each of the 4 cities that they touch this one, this one, this one and this one there are 2 other edges which are of lower cost and so those 2 other edges will figure in those lower bound estimates. So obviously our lower bound may not be feasible, but it is definitely a lower bound.

(Refer Slide Time: 18:10)



Refinement

- As segments get fixed in the tour *their known costs* are included in the estimate.
- As this happens the estimate becomes more accurate.
- Consider now the refinement search space.
  - The root consists of a node representing all solutions.
  - We can partition this set in two, one including a particular arc and the other excluding it.
  - These two sets can then be further refined recursively till each node describes one tour precisely.
- We apply the heuristic of choosing the smallest segment. In our example this is
  Chennai – Bangalore with a cost 360 km.

Artificial Intelligence: Search Methods for Problem Solving Deepak Khemani, IIT Madras

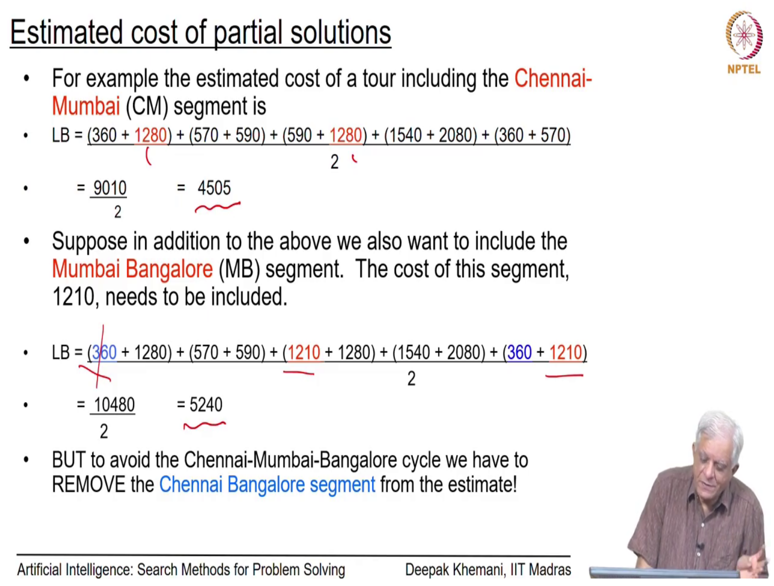
Now, we look at the process of refinement essentially. So, we will work with this example and look at this as we fix segments. So, we said that we will specify the refinement process will continue by saying that add a particular segment to a tour. So, whenever we add a particular segments.

So, let us say we start with the Chennai Bangalore segment we have partitioned the set of possible tours into two sets; one in which the Chennai Bangalore segment is present and the other in which the Chennai Bangalore segment is absent.

So, we are partitioned that the original set of all possible tours into two sets; one which has this segment and the other which does not have this segment. Now we need to refine our estimates, because we know that if you have added some segment that will definitely be part of the estimate and if you have excluded certain segment then that will definitely not be part of the estimate.

So, we can look at this process and refine our estimates for each of the candidates that we generate. What are the candidates? The candidates here are initially the two sets; one containing Chennai Bangalore and one other not containing Chennai Bangalore.

(Refer Slide Time: 19:24)



Estimated cost of partial solutions

- For example the estimated cost of a tour including the Chennai-Mumbai (CM) segment is
- $LB = \frac{(360 + 1280) + (570 + 590) + (590 + 1280) + (1540 + 2080) + (360 + 570)}{2}$
- $= \frac{9010}{2} = 4505$
- Suppose in addition to the above we also want to include the Mumbai Bangalore (MB) segment. The cost of this segment, 1210, needs to be included.
- $LB = \frac{(\cancel{360} + 1280) + (570 + 590) + (1210 + 1280) + (1540 + 2080) + (360 + 1210)}{2}$
- $= \frac{10480}{2} = 5240$
- BUT to avoid the Chennai-Mumbai-Bangalore cycle we have to REMOVE the Chennai Bangalore segment from the estimate!

Artificial Intelligence: Search Methods for Problem Solving Deepak Khemani, IIT Madras

So, let us see what is the estimated cost of those partial solutions. So, supposing that we want to estimate a cost of a tour which includes the Chennai Mumbai segment. Now if you look at our costs Chennai Mumbai is 1280 and we want to look at all those tours which contain 1280 and all those tours which do not contain 1280. So, you have seen that in two rows, one for the Chennai row and one for the Mumbai row we have added this 1280 to the estimated cost.

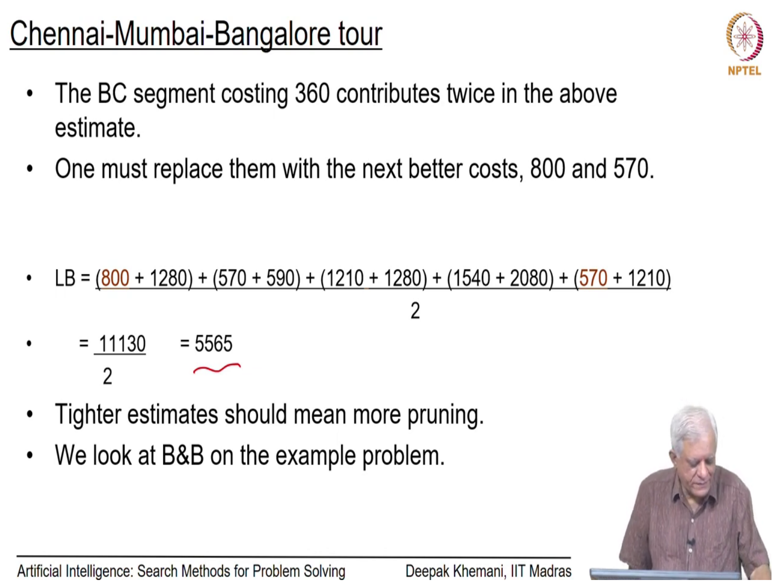
Even though originally we did not add this to the Chennai tour because this is not the shortest of the 2 edges, but if we know that Chennai Mumbai is going to be part of that tour then we can revise our estimate. And from our lower estimate about 3500 something now we have come to a slightly higher estimate of 4500 we know that this is going to be lower bound.

So, just make sure that you have understood that fact that our all our estimates are going to be lower bound because this is going to be a lower bound. Because, we have said that all the tours which contain the Chennai Mumbai segment what is going to be the lower bound and that is going to be 4500. Now suppose in addition to that we say that we also want to include Mumbai Bangalore segment.

So, the Mumbai Bangalore segment contains this cost of 1210 and so if you want to include that then you can see that because we said Chennai Mumbai Bangalore will form a triangle. We remove this edge from our estimated cost and we will get a better estimate and we see that the estimate is for 5240 here.

So, in this manner as we keep adding edges to our partial solutions, we can keep revising our estimates and the estimates will definitely go higher and higher from the original one which was the absolute lower bound on all possible tours.

(Refer Slide Time: 21:30)



Chennai-Mumbai-Bangalore tour

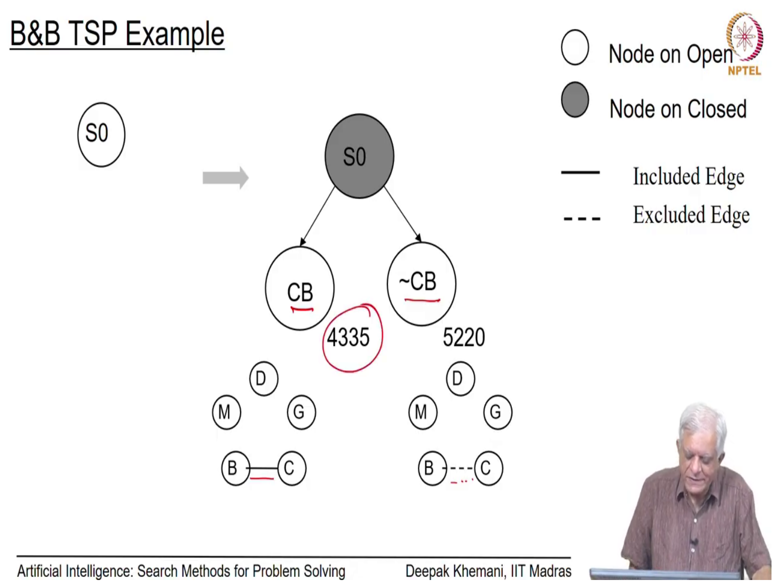
- The BC segment costing 360 contributes twice in the above estimate.
- One must replace them with the next better costs, 800 and 570.
- $LB = \frac{(800 + 1280) + (570 + 590) + (1210 + 1280) + (1540 + 2080) + (570 + 1210)}{2}$
- $= \frac{11130}{2} = 5565$
- Tighter estimates should mean more pruning.
- We look at B&B on the example problem.

Artificial Intelligence: Search Methods for Problem Solving Deepak Khemani, IIT Madras

So, let us see how that works. The Bangalore Chennai segment costing 360 contributes twice in the above estimate one must replace them with the next better costs which are 800 and 570. And after having removed that 360 as we just saw the revised estimate is now going to be even higher which is 5565. And we have already said with the higher the estimates the better for us as long as they are lower bounds essentially.

So, tighter estimate should mean more pruning and we will prove this more formally when we look at this algorithm called A star. For the moment we just take it for granted that the higher the estimate it is better for us and we now look at how branch and bound would solve this small example problem of 5 cities.

(Refer Slide Time: 22:22)

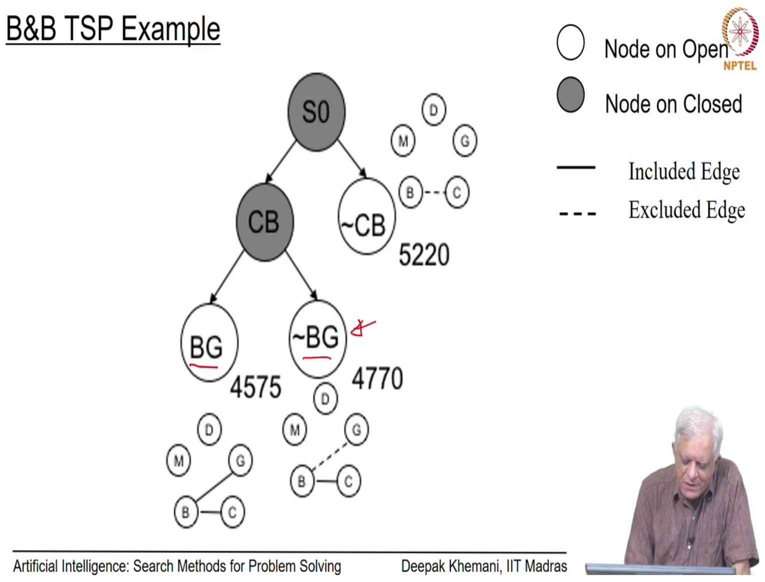


So, initially we start with this set S0, this set S0 is a set of all possible tours and we have already estimated it cost to be about 3500 something and then we refine this by saying that we are going to add the Chennai, Bangalore segment.

So, this CB that you see here stand for the fact that this is all those tours which contain this Chennai, Bangalore segment and this not CB this logical notation we are using represents the fact that all those tours who do not contain the Chennai, Bangalore segment essentially.

Now, the fact that this tour is this segment is included is shown in this line here and it is excluded is showing in this dotted line here essentially. And then we have these two estimates for the two candidates in open now one containing the Chennai, Bangalore segment and the other not containing the Chennai, Bangalore segment. Now clearly this is the lower of the 2.

(Refer Slide Time: 23:27)

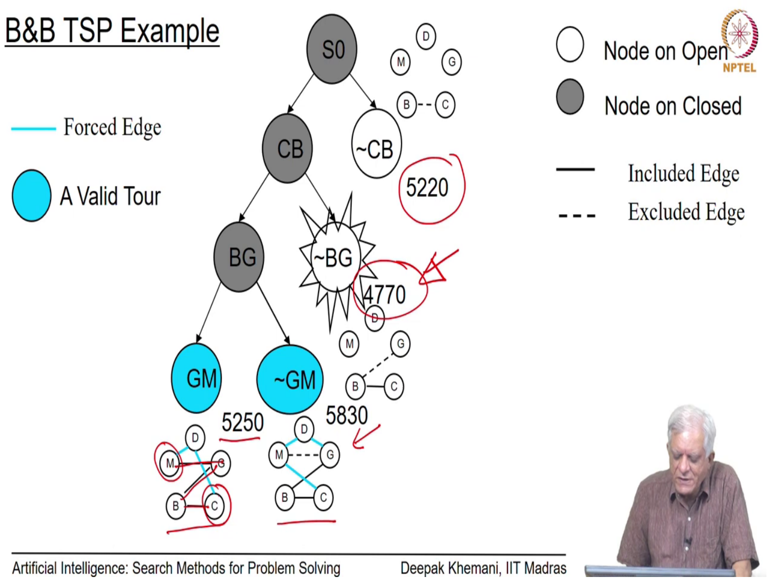


B&B TSP Example

Artificial Intelligence: Search Methods for Problem Solving Deepak Khemani, IIT Madras

So, we refine this one next and what we get is another refinement in which we add the Bangalore, Goa segment and here we exclude the Bangalore, Goa segment. So, what does this node represent this node represents all those tours which contain the Chennai, Bangalore section, but exclude the Bangalore, Goa section.

(Refer Slide Time: 23:53)

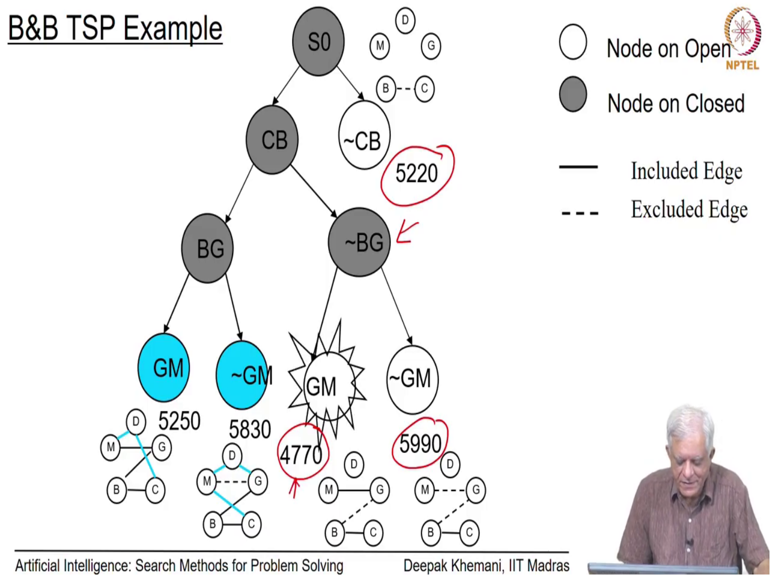


In the diagrams below that the rest of the edges become forced essentially. Now, here because you have added the Bangalore, Chennai section you have added the Bangalore, Goa section and you have added the Goa, Mumbai section. The only way you can go to Delhi is from one of those 2 cities which you have visited once one of them is Mumbai and the other is Chennai and therefore you have to add those two edges.

So, this is completely refined we have shown this in a blue colour node and we know it is exact cost which is 5250. And likewise this other tour is also completely refined, because the other edges are forced you can just think about this and see that if you include the Chennai, Bangalore section and include the Bangalore, Goa section and exclude the Goa, Mumbai section then the rest are forced to be as they are shown in those blue edges.

So, we have found actually 2 solutions at this moment shown in this blue nodes and we have their costs one is 5250 another is 5830. But there are other candidates whose costs are lower and branch and bound says that only when you pick a node which is fully refined then you should terminate. So, our algorithm keeps always picking the node with the lowest estimate in this case the lowest estimate node is this and so we refine that next.

(Refer Slide Time: 25:21)

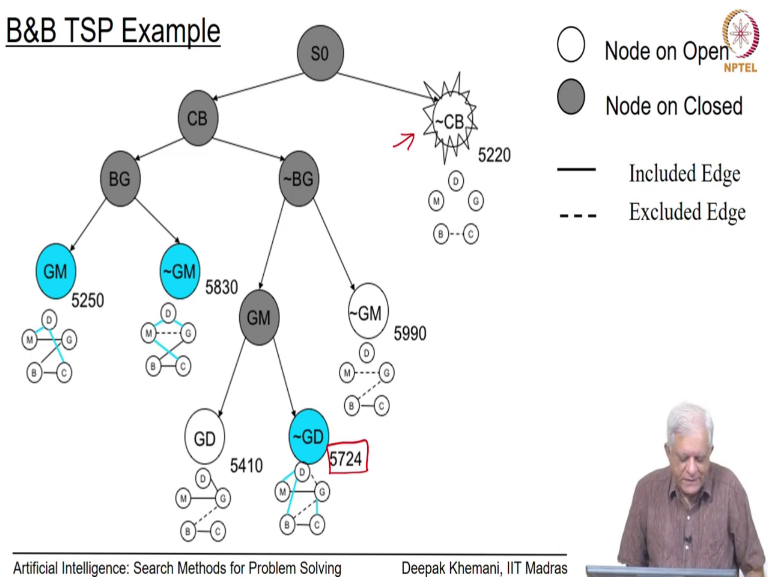


B&B TSP Example

Artificial Intelligence: Search Methods for Problem Solving Deepak Khemani, IIT Madras

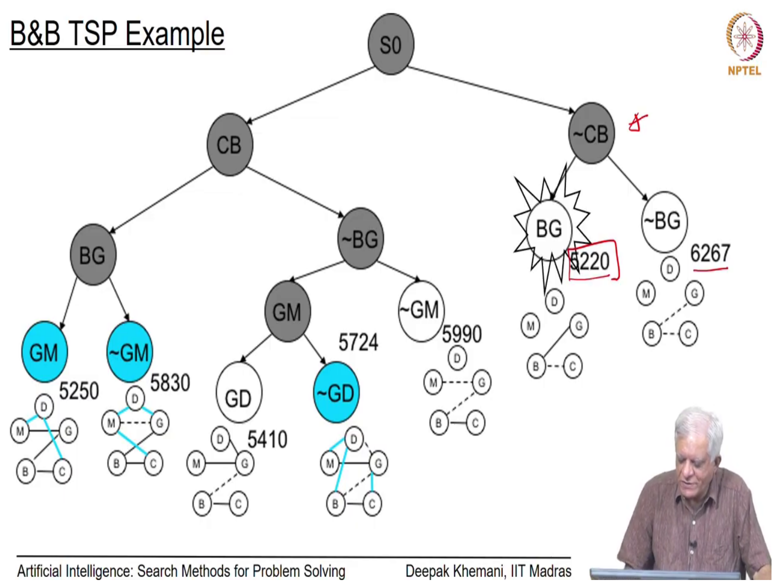
So, that is a node that we just refined and we added 2 sets. So, in one case we added the Goa, Mumbai section and in another case we added the excluded the Goa, Mumbai section. So now we have 3 nodes left and open one is this one; one is this one and the third is this one. So obviously, we pick the one with the lowest cost and refine that next and then we get another tree.

(Refer Slide Time: 25:52)



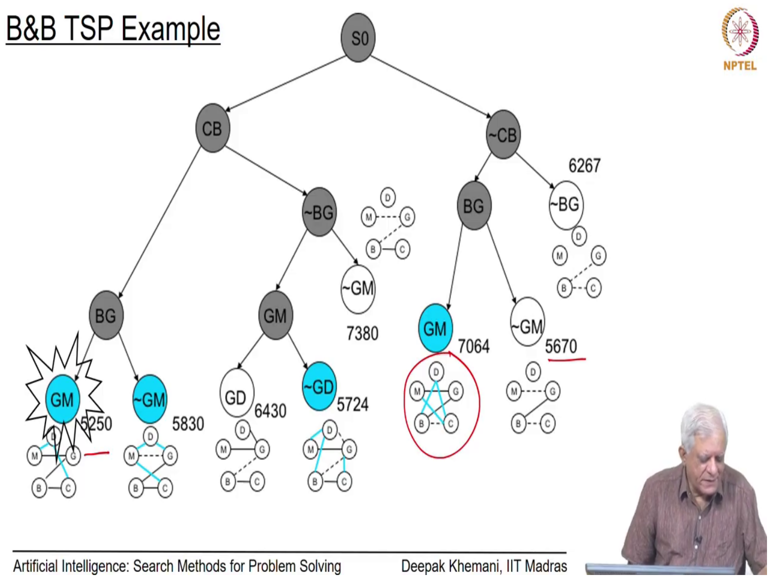
And in this 2 new nodes that we have added one of them which excludes a Goa, Delhi segment is fully refined, because the other edges are forced here. And a node which includes the Goa, Delhi segment is still not fully refined, because there is more than one possibility of doing that essentially. The lowest node however has shifted to this that is a value of 5220, so that is the one that we refine next.

(Refer Slide Time: 26:26)


B&B TSP Example
S0
CB
~CB
BG
~BG
BG
5220
~BG
6267
GM
~GM
5250
5830
GM
~GM
5724
5990
GD
~GD
5410
NPTEL
Artificial Intelligence: Search Methods for Problem Solving
Deepak Khemani, IIT Madras

So, when we refine that next and that was a node that we just refined we get 2 new nodes with a cost of 5220 and 6267. So, you notice that this has not changed because this Bangalore, Goa segment was anyway part of that estimate and having added it to the tour does not change the estimate essentially. So, we continue this process so this is a lowest node at this point and we refine that next.

(Refer Slide Time: 26:53)

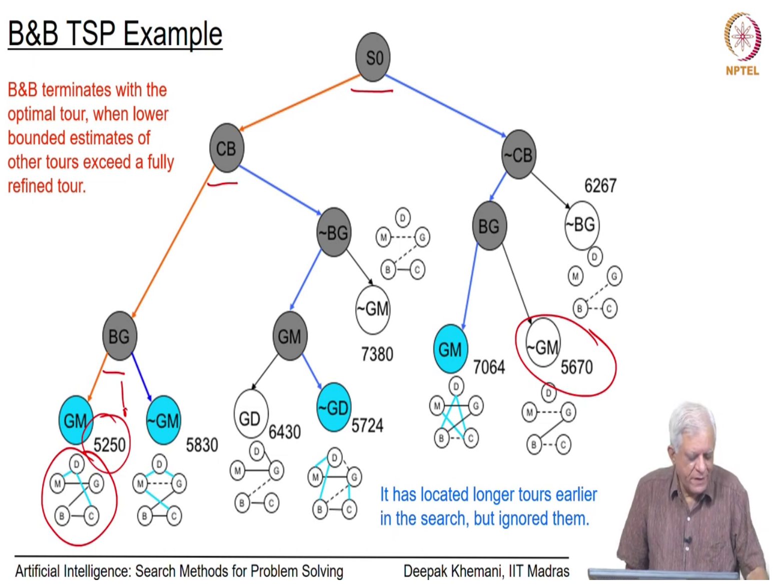


And we have got these two new tours one which includes Goa, Mumbai which is now fully refined and therefore it is shown in blue. And the one which excludes Goa Mumbai which is not fully refined but it is so it is still a partial solution, it may have more than one possible refinements now essentially.

But at this moment you see that one fully refined solution which is the leftmost solution with a cost of 5250 is become the lowest cost and the algorithm branch and bound will now pick this node.

And if you remember what the algorithm said the algorithm said if keep refining the best looking partial solutions till the best solution is fully refined. So, at this stage we have got a solution which is fully refined and we pick this solution and the algorithm should terminate.

(Refer Slide Time: 27:49)



B&B TSP Example

B&B terminates with the optimal tour, when lower bounded estimates of other tours exceed a fully refined tour.

It has located longer tours earlier in the search, but ignored them.

Artificial Intelligence: Search Methods for Problem Solving Deepak Khemani, IIT Madras

So, the algorithm terminates by finding this path which we have shown that and this path is in the search space the search space is a refinement space the space of all possible solutions.

So, here we said the in S0 we said that this we consider all possible tours. Then this node CB says that all those tours which contains the Chennai, Bangalore segment, then this node BG says that all those tours which contains the Chennai, Bangalore segment and the Bangalore, Goa segment.

And this last node that we have just picked as the solution says that all those tours and in fact there is only one such tour which contains the Chennai, Bangalore segment, the Bangalore, Goa segment and the Goa, Mumbai segment. And as we discussed a little while ago this

Delhi is connected to Mumbai and to Chennai and that gives us a complete tour and since this complete tour has a lowest cost we can terminate with this algorithm.

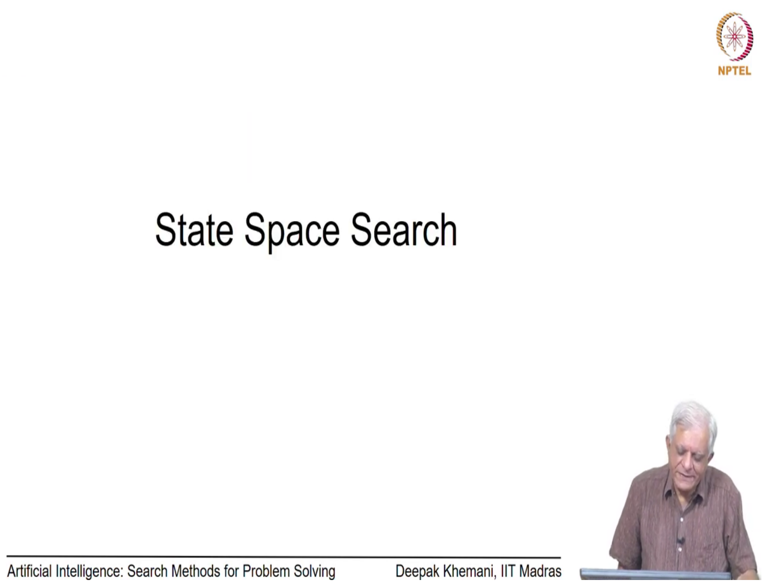
So, branch and bound terminates with the optimal tour when the lower bound estimates of other tours exceed a fully refined tour. So, our fully refined tour is 5250 and you can see that all the other nodes which are in open or which are leaf nodes, in this search tree all of them have got a cost which is higher than 5250.

So, we are safe to terminate at this stage because all those other solutions. For example this solution which contains the which excludes the Chennai, Bangalore section adds the Bombay, Goa section, but excludes the Goa, Mumbai section has a lower bound of 5670; 5670 is a lower bound it means that the actual cost of refining the solution would be higher than 5670.

And since we already have a solution with 5250 there is no need to explore that any further and the algorithm can safely terminate guaranteeing that the solution that we have found of 5250 is the lower bound tour.

Now observe that on the way it had found many solutions. So, there are four nodes coloured blue in this search space all 4 are complete solutions, but it terminated only when a complete solution was completely refined and that is how branch and bound guarantees that you terminate with an optimal solution. So, this was looking at branch and bound in the TSP problem which has been our favourite problem for the last few last couple of weeks.

(Refer Slide Time: 30:31)



State Space Search

Artificial Intelligence: Search Methods for Problem Solving Deepak Khemani, IIT Madras

We will not depart from the Traveling Salesman Problem and move back to state space search that is what we started with that if you are in some state space. And you are in a given state and you want to go to some desired space desired state what is a solution what is a path from the start state to the goal state.

Now, the question we will ask is what is an optimal solution, what is the least cost path from the start state to the goal state and we will take this up in the next session. In which we will first start by looking at branch and bound in the state space search and then move onto something which is more interesting. So, see you in the next class.