Artificial Intelligence: Search Methods for Problem Solving

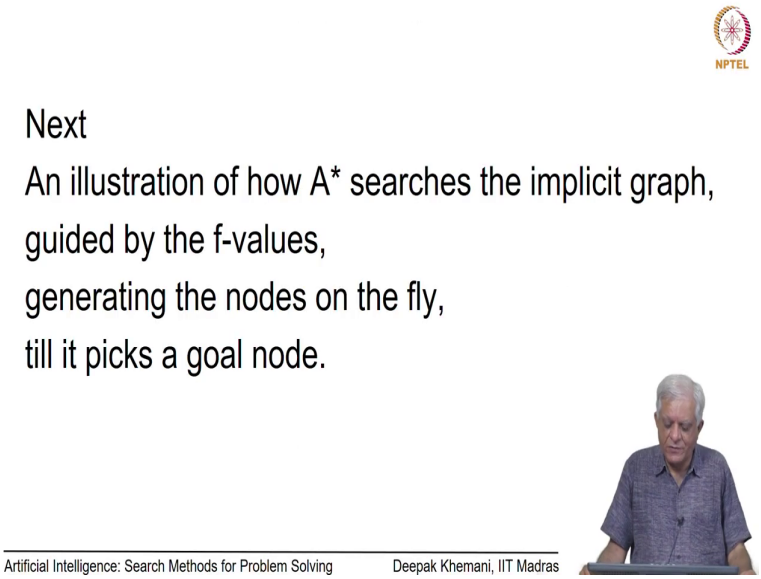
Prof. Deepak Khemani
Department of Computer Science and Engineering
Indian Institute of Technology, Madras

Chapter – 05
A First Course in Artificial Intelligence

Lecture – 37

Finding Optimal Paths: A*: An illustrated example

(Refer Slide Time: 00:14)

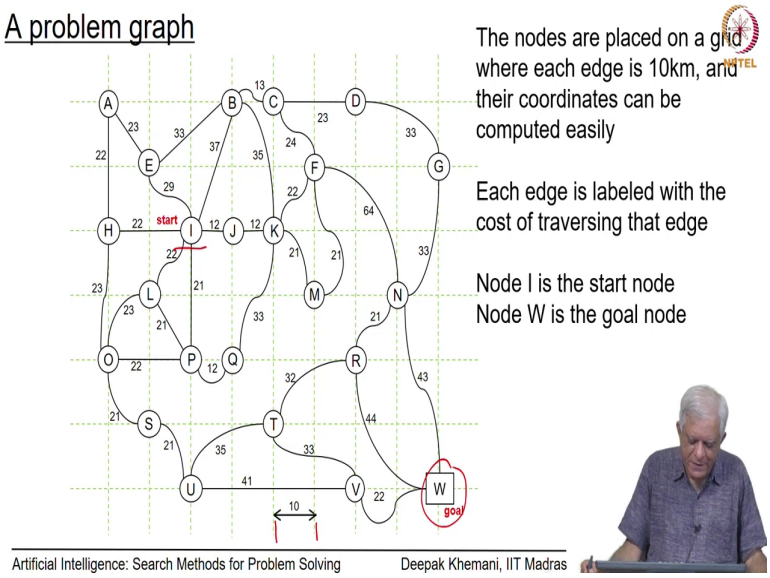


So, welcome back. We have just finished looking at the A star algorithm and let us look at a detailed example to see how the algorithm behaves and in particular we will compare its behavior to best first search because that is another heuristic algorithm.

And I will leave it as a small exercise for you to compare its behavior to branch and bound or Dijkstras algorithm and see in terms of both performance as to how quickly you find the goal node and quality as to what is the nature of the path you found.

How this algorithm A star differs from branch and bound from as well as from best first search. Best first search we will do in this session branch and bound hopefully you will do it offline.

(Refer Slide Time: 01:09)



So, let us look at this small example, its a problem graph as you can see the nodes are labeled with letters. So, A B C D E F G H and so on. The nodes have been drawn on a grid which is let us say 10 kilometers by 10 kilometers or 10 miles for 10 miles if you live in a country where you measure distances in miles.

So, the distance between two grid lines is 10 kilometers and the reason why we do this is because we are going to be using a heuristic function to compute estimate distances and because these points are on the grid and they have integer coordinates which are multiples of 10, its easy to compute at least in the example the heuristic distance.

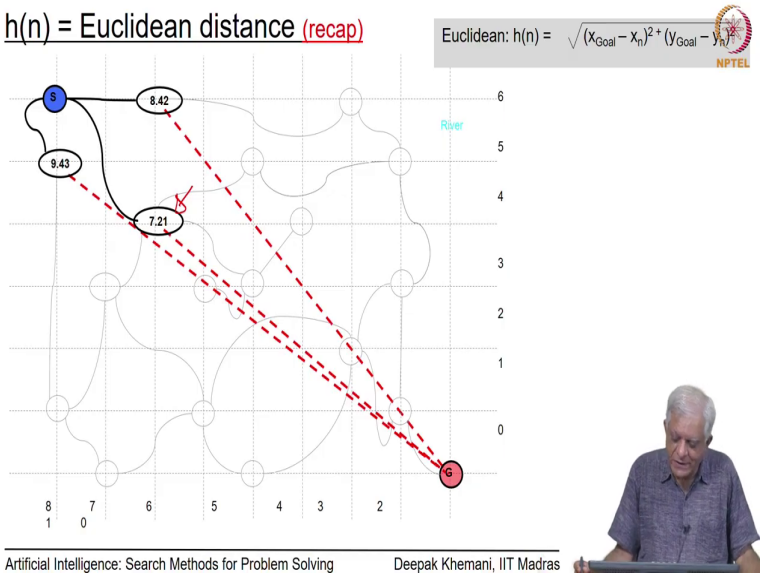
Of course, if you are working with a real map which is used for root finding and you are implementing an algorithm, the algorithm will have no difficulty computing fractional coordinates because that is one of the things machines are better than us is to do this kind of arithmetic very quickly and accurately.

Each edge in this graph has been labeled with a cost, this cost is the actual cost if you were to traverse that edge essentially. So, this is the cost that will add up to the final cost of the solution, the heuristic values are only estimates which are used for guiding search.

Whereas, the H cost are the actual cost which will be incurred by whatever is the application that is using this A star and like we did in the case of branch and bound and Dijkstra algorithm we will see that we will keep adding the edge cost as for every node that we are going from the start node to the goal node.

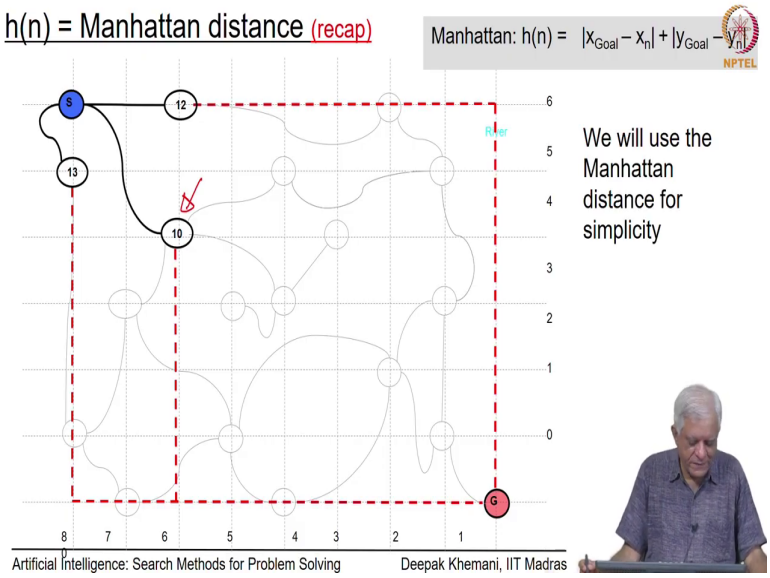
In our example here we will have we have chosen the start node I which is in the middle of the graph somewhere as a start node and this node W which is at the corner as a goal node essentially. So, we want to see how our algorithms they find paths from start node to the goal node.

(Refer Slide Time: 03:36)



Remember that we had looked at some heuristic functions already in this domain which is let us say the root finding domain, one estimate of distance is the Euclidean distance which is the straight line distance of the distance as the crow flies and you can see that for one particular blue node and its three successors which are drawn in bold here, the heuristic distances are 7.21 8.42 and 9.43 and best first search algorithm would presumably have this node to expand next.

(Refer Slide Time: 04:18)

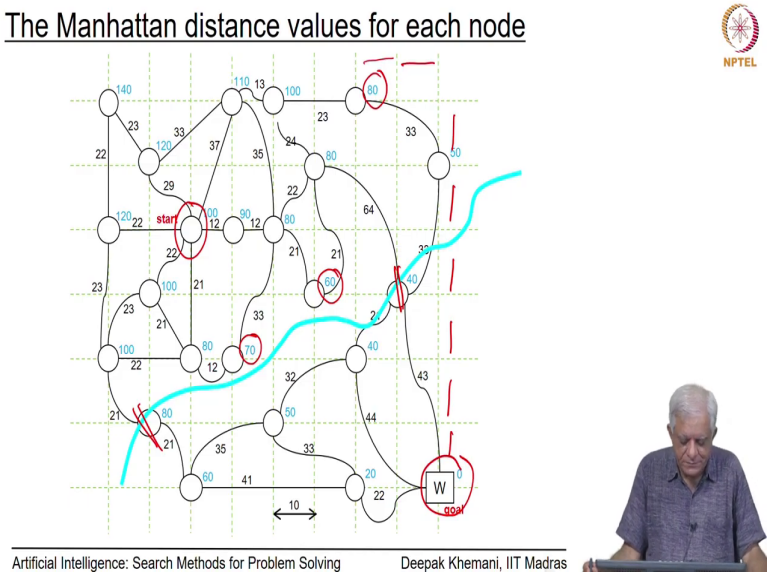


The alternative that we considered was Manhattan distance which is also known as the city block distance which assumes that we are living in some city which is brought parallel loads both in the horizontal and the vertical direction and the distance that you would have to cover is along those city blocks as they call them.

Now, remember that this is only an estimate the Manhattan distance is only an estimate the actual parts that our algorithms are going to be traversing are along the edges of the graph, but the heuristic distance is independent of that. It simply says how far are we how far do we appear to be from the goal the Euclidean distance and the Manhattan distance both give us estimates in some manner without looking at the graph only looking at the coordinates of the two nodes.

And you can see that even in the Manhattan distance case this node same node whose now Manhattan distance is 10 appears to be closest to the goal node. In this graph that I have drawn the distances are in units of 1 unlike in the example that we are looking at where there are units of 10 anyway that does not matter its just a matter of scaling this thing.

(Refer Slide Time: 05:38)



We will use a Manhattan distance for the sake of simplicity and further problem that we are trying to address here are the Manhattan distances shown in blue with every node. So, for example, this value has a value of 80. So, you can see that there are 2 segments here and 6 segments here and so, and since each segment is 10 units ok. So, this one is not there. So, these are the 6 segments. So, 6 plus 2, 8 and so, we have a distance of 80 essentially.

Now, the graph does not depict the reality of the problem and its possible that there is a river running as shown here in this and there are only two places where there is a bridge to cross the river.

And so, clearly any solution which goes from the start node which is here to the goal node which is here would have to go through one of these two bridges and we will see that all the algorithms that we are going to look at in fact, the two algorithms to start with which is best first and A star both will find paths through these two bridges.

But you will also observe that there are certain nodes for example, this node with a value 70 and this node with a value 60 and so, on they appear to be much closer to the goal node.

So obviously, both algorithms that we are looking at because they are going to be using heuristic functions they will head towards these nodes because they appear to be closest to the goal, but from there if there are no neighbors which are improving their prospects, they would have to in some sense retract and go back and try some other paths.

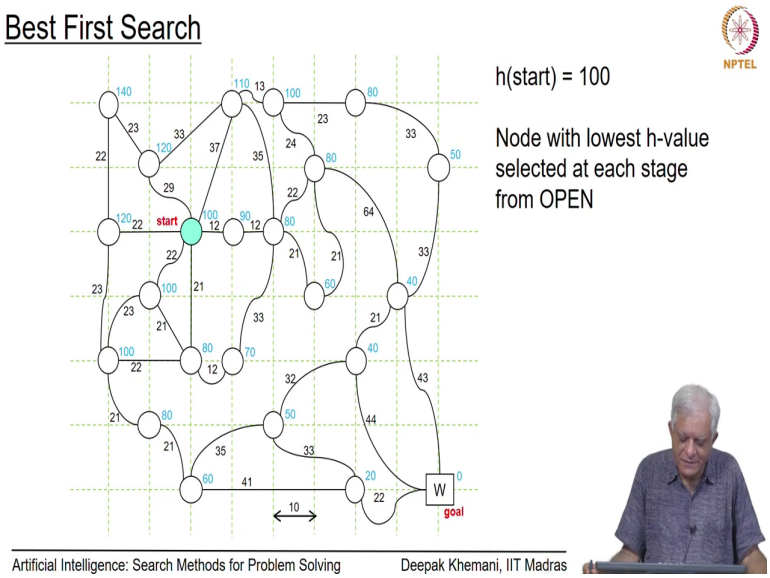
So, in that sense both best first and A star are complete algorithms unlike hill climbing which would have got stuck in one of these nodes and both best first and A star will find solutions or paths to the goal. Best first is expected to perform faster because it is only guided by the heuristic distance and as we will see in the example today that even though it terminates faster it terminates with a solution which is worse than the solution found by A star.

So, A star does no more exploration, but it finds a better solution essentially and later on in the next session when we meet we will prove that under certain conditions A star will always find the optimal solution.

The conditions will be as you can expect on the heuristic function, you cannot choose any arbitrary heuristic function and get estimates which are useful in guiding search. The heuristic

function must satisfy certain properties and these are properties that I would urge you to think about and also we will discuss them in the next session more formally.

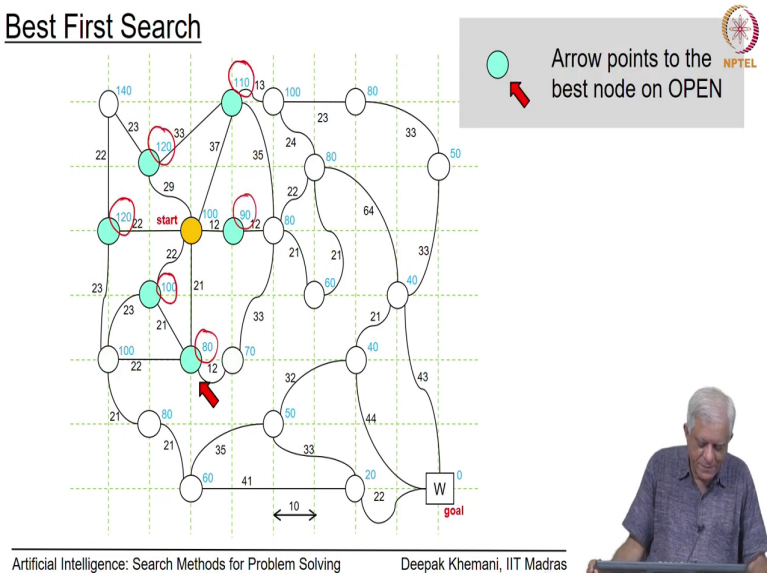
(Refer Slide Time: 08:41)



So, this is a graph that we are going to be working with and I remove the labels nodes from the nodes and instead we will use the f values which is what the algorithm uses for picking a candidate from the set of candidates or this of the open list. But first we start with best first search and best first search uses the h values and the h values have already been labeled here.

So, we will just show the open nodes in cyan and close nodes in orange and see how the algorithm proceeds. So, as any algorithm you start from the start node here and then start exploring the graph looking for a path towards the goal node.

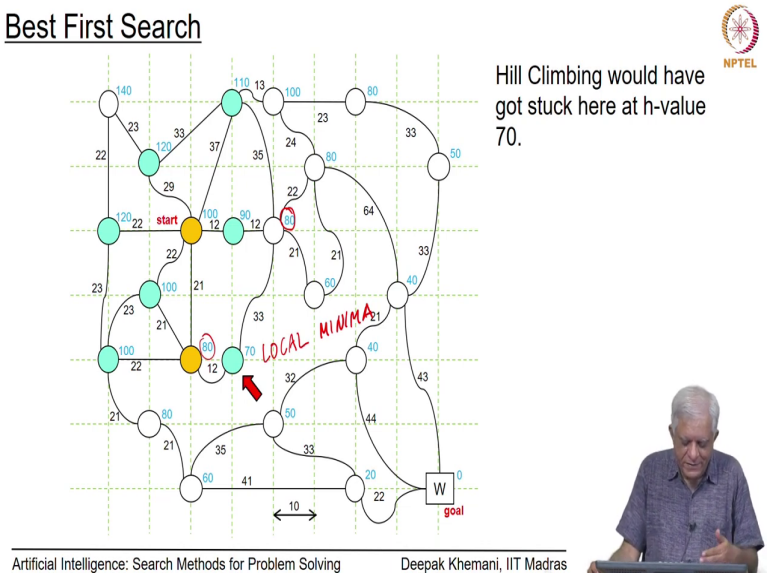
(Refer Slide Time: 09:30)



So, we put the start node in closed and as I said its in orange or brown or whatever it looks on your screen and once we generate one we expand that we generate its neighbors and you can see there are 5 neighbors of the start node and this arrow red arrow will tell us as to which is the best neighbor at any given point.

So, this nodes in cyan will be the set of open nodes and the arrow will tell us which is the best. So, you can see here that the nodes in open one has a value of 100, another has a value of 120, this has a value of 120, this as the value of 110, this as the value of 90 and this as the value of 80. So, since 80 is the smallest that is what best first is going to pick next and this red arrow will tell us at each stage as to what is a candidate that best first is going to pick.

(Refer Slide Time: 10:26)



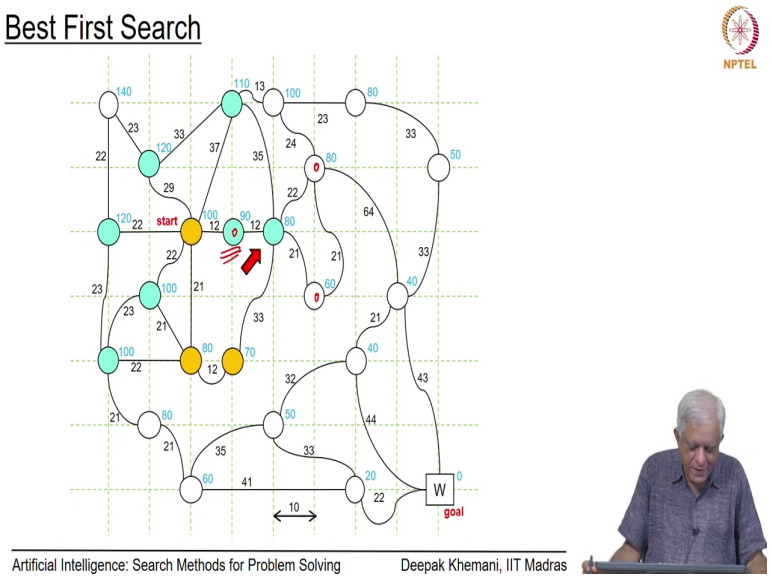
So, the next once it expands this node that it has just picked, its two neighbors you can see the one and the one here they will get added to the graph and now they have gone into the open list. So, the open lists are expanded we have more open nodes and the best looking open node is the one with the heuristic value 70.

Now, if you were to looking at the hill climbing algorithm. So, remember that hill climbing is a local search algorithm, it only looks at the local neighborhood of any given node and if this is a node with a value 70 that you are going to expand then it has only two neighbors because there are two edges going out of this, one has a value of 80 and the other also has a value of 80.

So, for hill climbing this is a local maxima or local minima and hill climbing would have got stuck at this point and terminated by saying that you cannot find a path to the goal node, but

as we have been saying best first search is a complete algorithm its a global search algorithm it does not throw away all these nodes in open that it has maintained and so, when it expands this node it generates new neighbors in particular the neighbor which appears to be the best which has a value of 80 this thing.

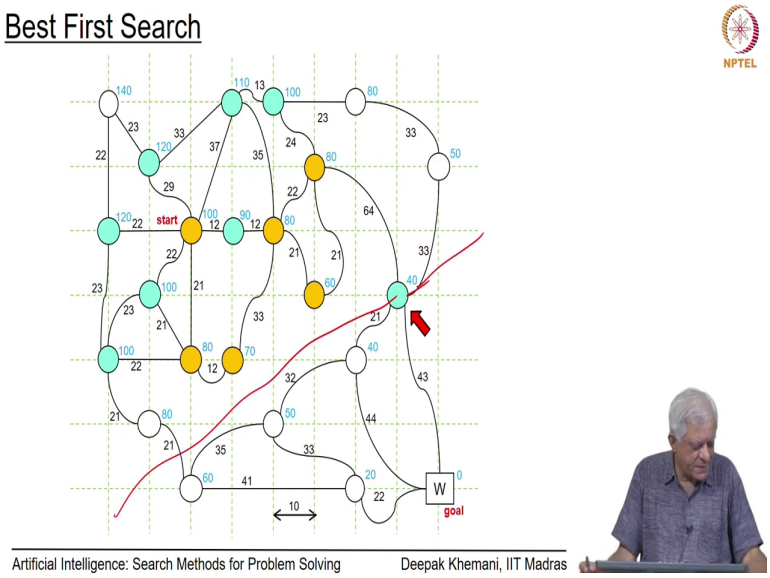
(Refer Slide Time: 11:41)



So, keep in mind that the nodes in orange or brown are the ones which have been inspected, the nodes in cyan are the ones which are the set of candidates or the open list and the best one is the one that is going to be expanded at each stage essentially.

So, when it picks this node 80 and expands it, it add it will add its new neighbors this one and this one as well as this node it will find again, but since it is guided only by heuristic values, we will see that it will ignore this node completely and as a result it misses out on the shorter path which is coming through that node to this particular node.

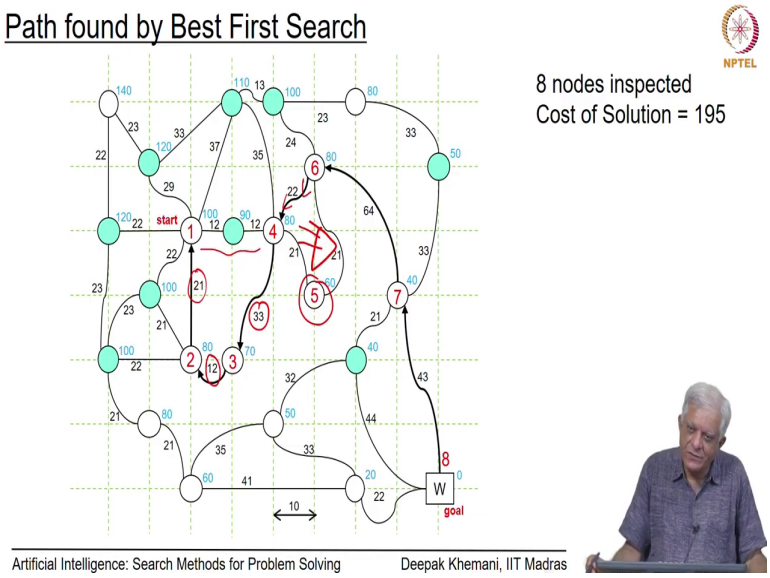
(Refer Slide Time: 12:33)



So, when it expands this node you see that this neighbor with a value 60 is appears to be closest to the goal and it does indeed appear to be closest to the goal except for the fact that there is no edge going from there towards the goal and that we observed was could be because for example, there was a river running across that part of the thing. So, it does expand that node, but it does not find any new neighbors. So, it shifts its attention to this node with a value 80.

And once it expands that you can see that in some sense its on the bridge if you remember the river, the river was running somewhere like this and its about to cross the bridge and it will rapidly find a path to the goal node and so, here it is the goal node is on its open list and it clearly the best node because it has a heuristic value of 0.

(Refer Slide Time: 13:35)



So, it picks that node and finds a path for us. You can see that the path it finds is does not look the best we have already observed that this could have been better, this would have been 12 plus 12 instead of this 21 plus 12 plus 33 which is a long path to this fourth node.

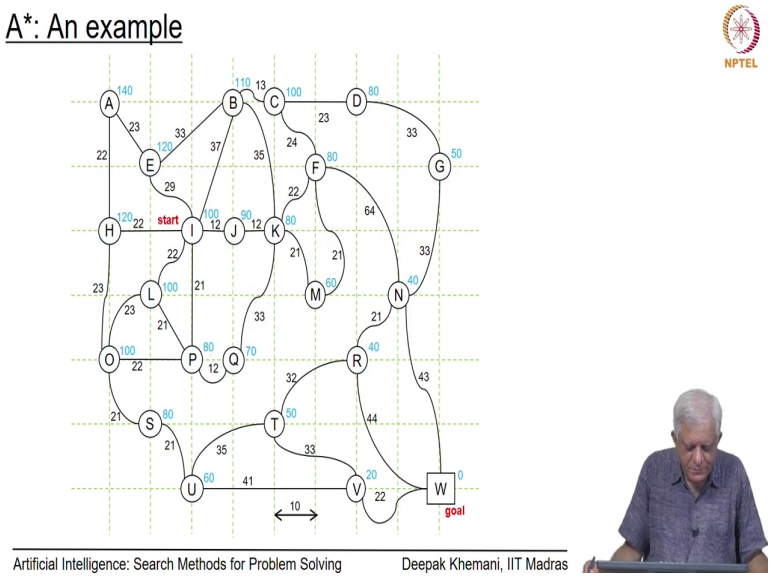
And the reason why that is that because it is guided by heuristic search, it is only looking forward from this direction it is not looking backward to see if there is a could have been found a better path it ignores the shortest path to that node.

Also observe that the path that it did explore this node number 5, but the path to node number 6 is actually through path number 4 because 6 would have been generated as a child of 4 first essentially.

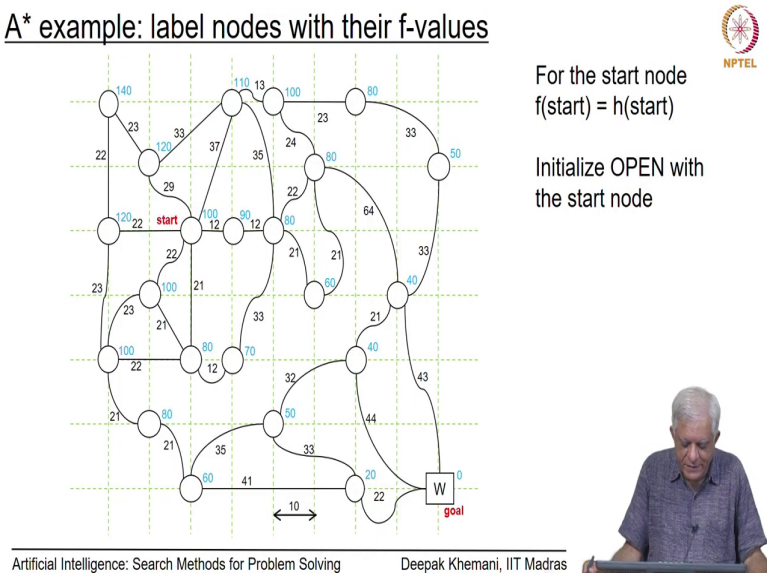
So, in that sense it keeps track of who was the parent and therefore, the path reconstruction algorithm will trace back to parents and therefore, this is the path it is found it has inspected eight nodes numbered here from 1 2 3 4 in the order in which it inspected them eighth node was the goal node and the cost of the solution it is found is 195 let us say kilometers.

Let us see now what A star does with this same problem. Now remember that best first search just ignored the edge cost it was not concerned with the h cost at all it was only looking at heuristic distances to the goal which are these values in blue that have been labeled next to that and therefore, it simply rushed towards the goal hoping to finish its job faster.

(Refer Slide Time: 15:23)

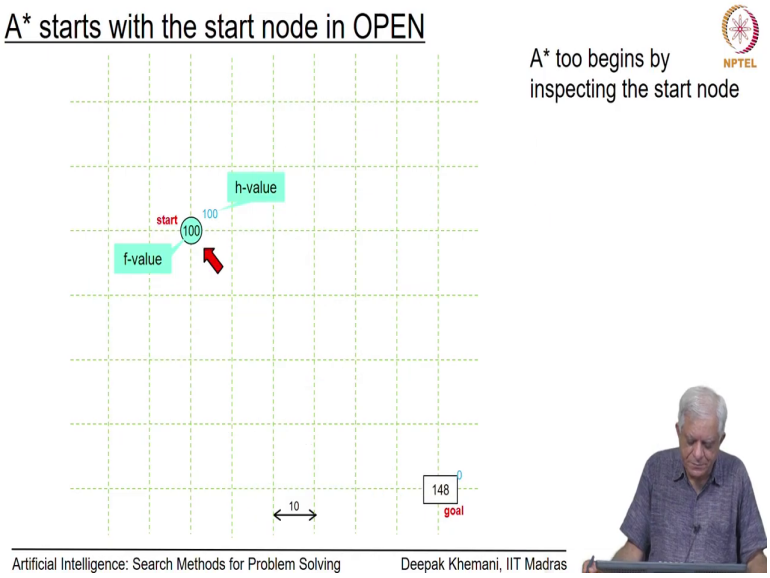


(Refer Slide Time: 15:30)



Now, we come back to A star with this same problem and what we will do is we will remove the node labels and instead we will label the nodes with f values. Remember that f values are the sum of g values and the h values the h values are the heuristic distances which have been labeled here for the entire graph and g values will be completed as and when we computed as and when we explore more and more of the graph.

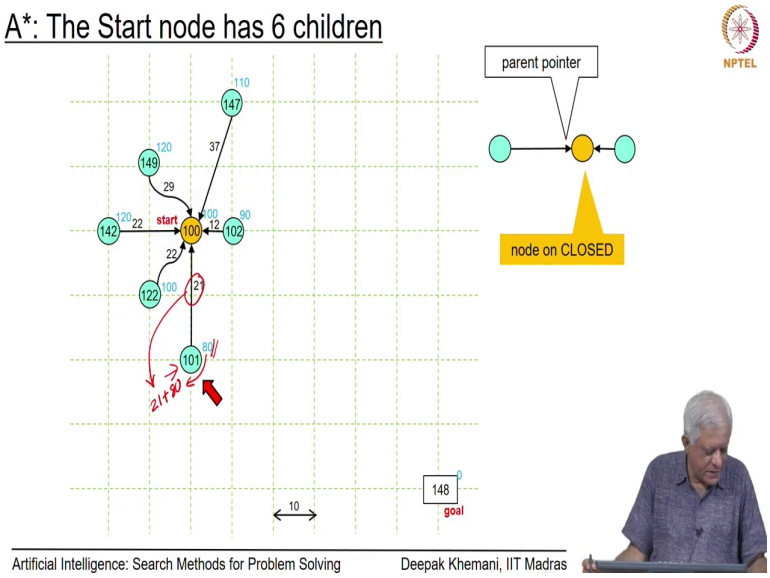
(Refer Slide Time: 16:02)



We start off with the start node as before and you can see that the label inside the node is 100 which is its f value. Remember its f value, but f value is equal to h value for the start node because g value is 0 because you are already at the you are already at the start node and this is the only node which is in contention. So, as all other algorithms algorithm A star also begins by picking this node as the first node for expansion.

For inspection checking whether its a goal and if it is not the goal then expanding it and generating its neighbors and you can see here that it has gone into closed list it has generated the same 5 neighbors that best first also generated.

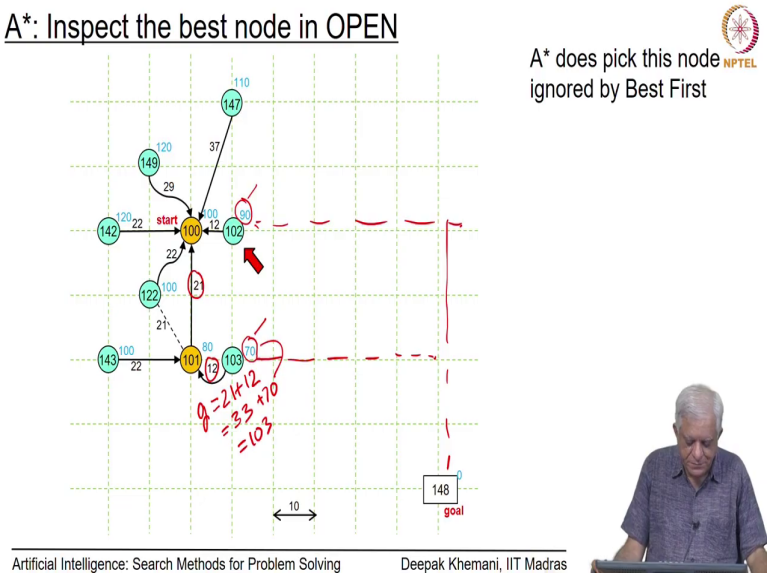
(Refer Slide Time: 16:41)



Except that now the neighbors are listed labeled with their f values and it will choose the node which is lowest in the f value in this case it happens to be the same node as picked by best first, but its value is not a which is what best first thought its value is 101 and this 101 is the sum of 21 plus 80.

So, this 80 comes from the h value and this 21comes from the g value and so, it looks at the sum of g plus h values and its value is 101. So, remember this 101 is the estimated cost of finding a path from the start node to the goal node which passes through this particular node and likewise for all nodes on open, it is an estimated cost of a path from start to goal which passes through that node. And A star will always pick the one which appears to be the cheapest estimated complete solution in this case it is the value 101 as we can see here.

(Refer Slide Time: 18:03)



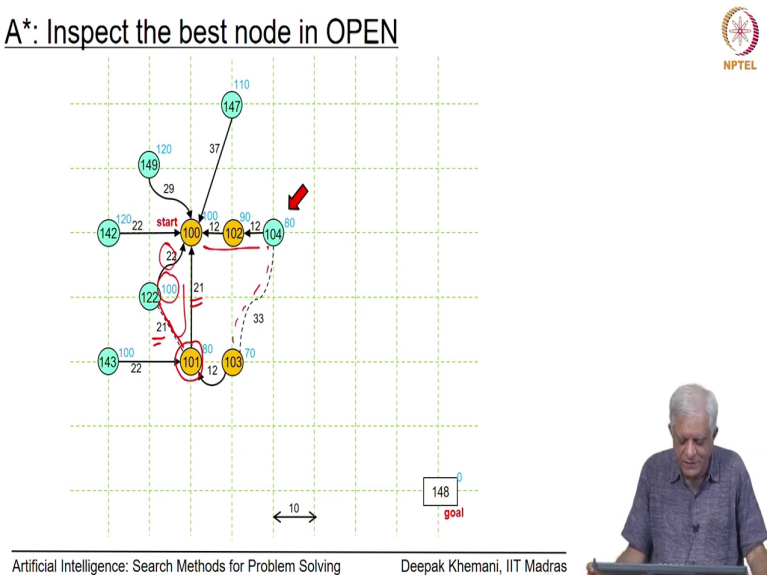
And like best first it generates its children and the same two children are generated, but their f values are computed here. But you can see that this node with a f value of 102 was ignored by best first because best first would have preferred this node with a value 70 as opposed to this value of 90 and after that it never found a chance to go back to 90 because its heuristic value was always less than 90.

But A star has been able to recognize the fact that even though the cost of estimated distance of this node is 90 to the goal node that is the estimated distance to the goal node. In fact, it is a Manhattan distance. So, you should draw it like this and not like this and the estimated distance of this is 70, the actual cost of reaching this g value is 21 plus 12 which are these two h cost is equal to 33.

So, 33 plus 70 is equal to 103. So, the estimated cost of going through this node is 173 even though its heuristic value is less than the 1 that A star is preferring whose heuristic distance is 90, but it knows that its g value is only 12 and therefore, the total estimated cost is 90 plus 12 is 102.

So, in that sense this is what I have been trying to emphasize is that the A star algorithm looks both behind as to how long did it how much did it cost to reach that node plus how much will it cost to go from that node to the goal node and therefore, its behavior is a little bit different from that of best first search. Best first search would have picked this node with an estimated cost of 70 because it does not look behind whereas, A star is looking at a combination of f and of g and h values and picks this node 102.

(Refer Slide Time: 20:31)

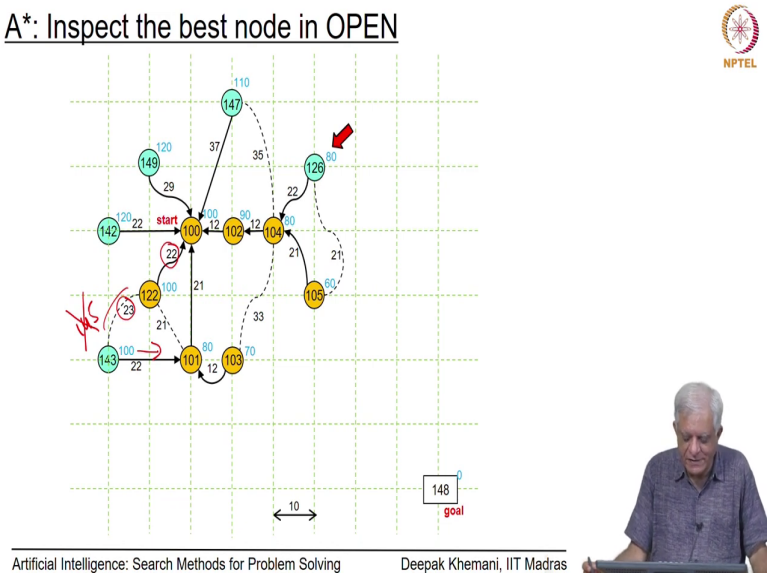


So, once it picks this node it adds its neighbors and its neighbor has an estimated cost of 104 the new neighbor it has generated and the node 103 which is what best first search inspected comes back into contention and it becomes the best node.

So, again it expands that now notice that when it expanded this node 101 it found a new path to the node whose value was 122, but it did not prefer that path because 122 was found was generated by adding up 22 and 100 and that is 122 whereas, if we had found if we had chosen this path then you would have add to add 21 and 21 which is 42 plus 100. So, it would have been 142.

So, A star keeps track of only the best path and it ignores this new path likewise for this node it has found a new path, but it is going to ignore it because clearly this other path that we spoke about earlier also is the shortest path. So, it keeps the parent pointer like Dijkstras algorithm pointing to the best way of reaching that particular node essentially.

(Refer Slide Time: 21:49)



So, now it is going to expand this node with a f value of 104 and h value of 80 as before it generates that node which appears to be close to the goal node as before it picks that node and find that it does not make any headway in that direction and it shifts its attention back to a node which on the surface it looks to be in the wrong direction heuristic.

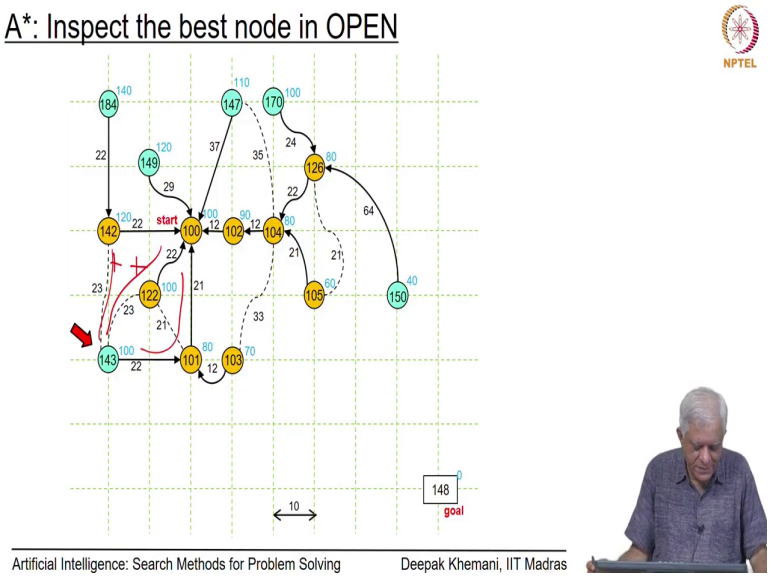
The heuristic function says that its value is 100 estimated value is 100 and that is also reflected in the placement in this graph whereas, there are nodes with for example, value 80 which seem to be better, but A star is picking that node with A value 100 with the h value of 100 and that again is because its f value is smaller if s value f value is 122 whereas, the f value of this is 126.

So, A star prefers to explore that node and in that sense it has this inherited behavior from Dijkstras algorithm and branch and bound that it will extend the cheapest estimated paths

first. So, that is what it does here and it has found another path to this node with value 143 which had found on earlier, but this new path would have been 22 plus 23 is 45 plus 100 would have been 145.

So, clearly it is not interested in that path of 145. So, it maintains this back pointer which is the shortest path of reaching that node, but it shifts its attention back to this node with heuristic value of 80 and f value of 126 generate its neighbors and as you can see here it has shifted this attention back to another node whose value is 142 f value because the other nodes are more expensive.

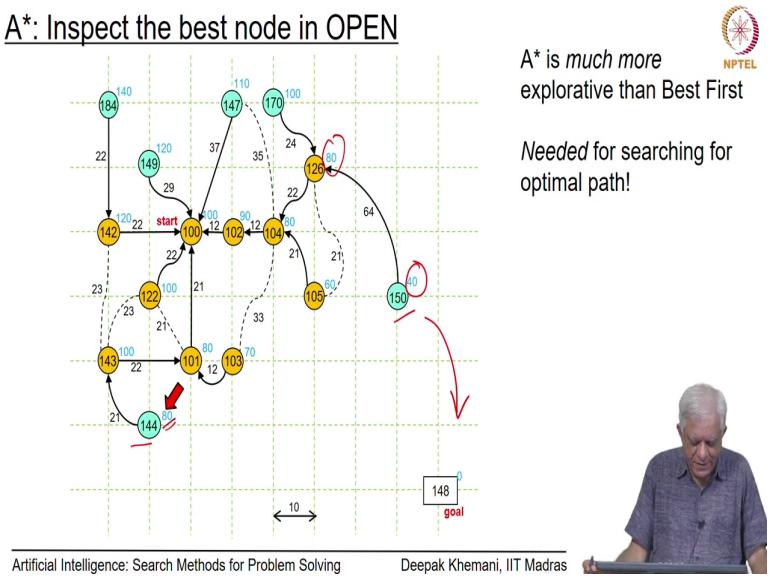
(Refer Slide Time: 23:38)



So, it expands that it finds yet another path to this node the third path the first path was the one that we found earlier the second path was this which was ignored and the third path is also ignored, but at this point this node has become the cheapest node and it explores this

next and then it turns out that the example that I have constructed, it is heading towards a different direction from what best first did.

(Refer Slide Time: 24:13)

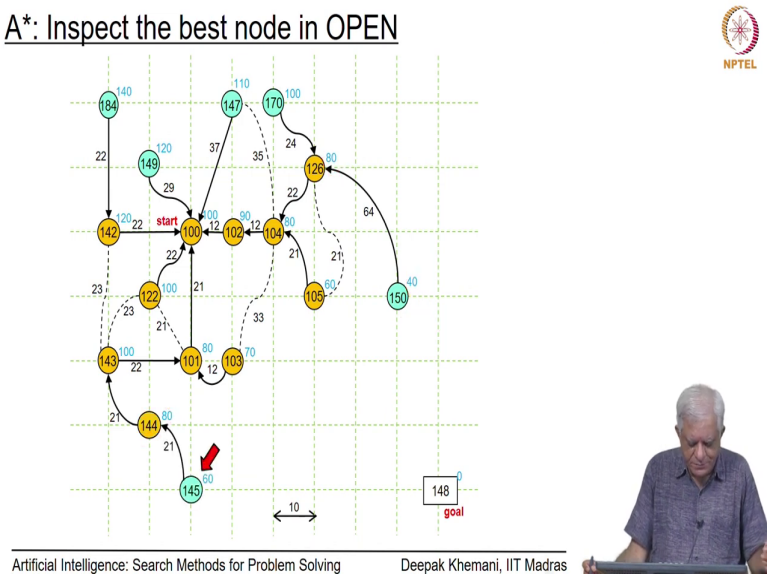


So, remember that that best first once it reached here then it found this 40 and then it headed directly to the goal and terminated there because it was only looking at distance to the goal.

A star is much more explorative than best first search and even at that point where it looks like we have a node which is only 40 kilometers away, it is still considering a node which is actually 80 kilometers away because it is also computing the actual cost found so, far and if you incorporate that this value is 150 and this value is 144 and therefore, clearly this has more promise of finding the optimal path essentially.

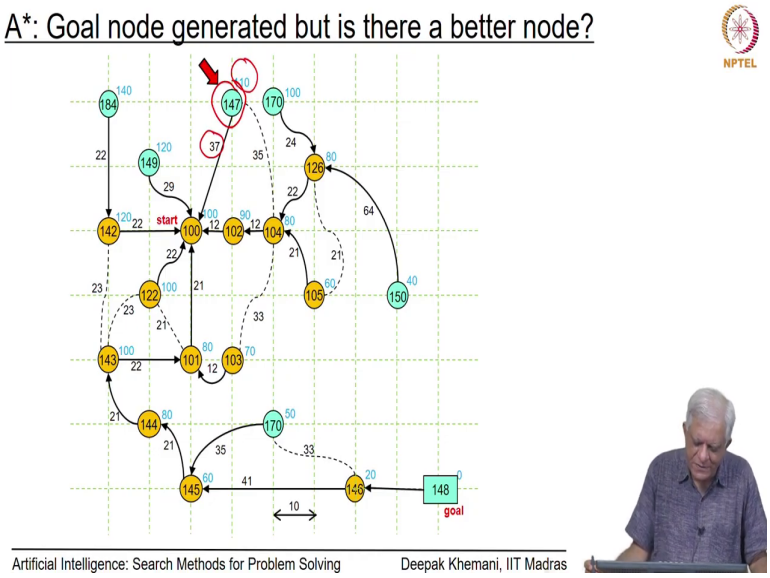
This extra explorative behavior is necessary if you want to guarantee that you are not going to miss out on optimal path and we will talk about optimality more formally in the next this thing.

(Refer Slide Time: 25:23)



So, it continues here and it so, happens in this graph that it rapidly now finds an different paths to the goal node and you can see that it is at a node which is only 20 kilometers away from goal or appears to be 20 kilometers away from goal based on the heuristic function and adds the goal node to the open list.

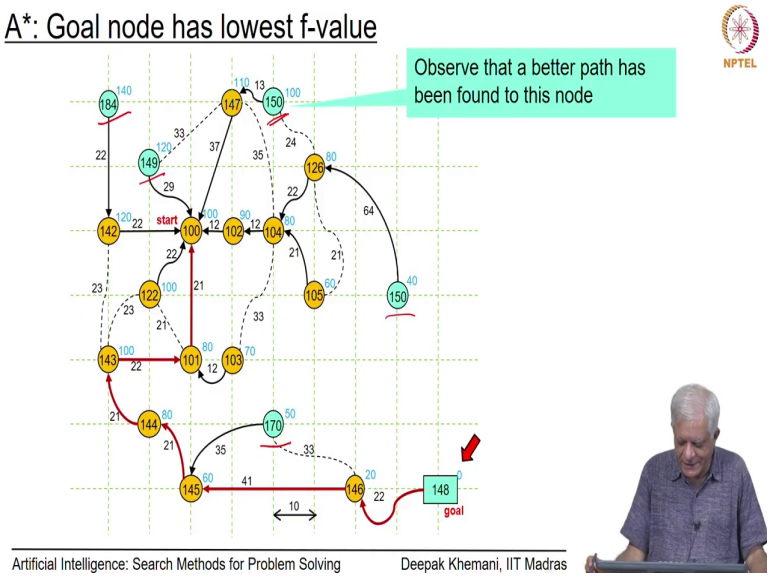
(Refer Slide Time: 25:47)



But it does not it does not pick up goal node next it picks up this node whose value is 147 which is less than the goal value of 148. Now, remember that the paths that we have found to the goal node, we know the exact cost which is 148.

The paths that we have found for this node to 147, its only an estimate its an estimate because its heuristic value its g value is 37 and its h value is 110 and the estimated cost is 147, but because this estimated cost appears to be better than the actual cost that we have found it like Dijkstras algorithm it will not pick the goal node till that has become the best.

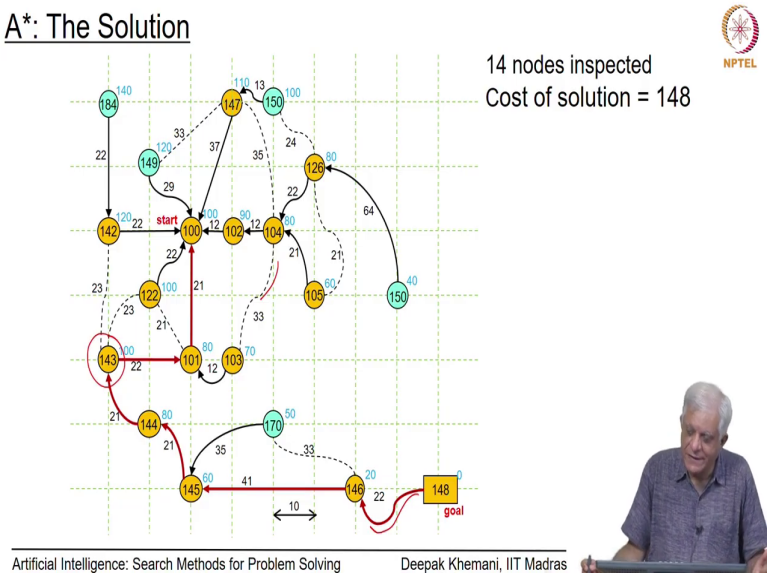
(Refer Slide Time: 26:37)



So, it picks this node with 147 and you can observe that at this point its found a better path to this particular node its value has come down it was 170 just before this node was expanded and now its become 150 its found a better path, but 150 is not good enough to compete with the paths.

That it has found to the goal which is of 148 and observe that all other open nodes are more expensive than 148 and therefore, A star is kind of justified in picking up the goal node and once it picks a goal node because it is completely refined it will terminate with a path of 148 essentially.

(Refer Slide Time: 27:17)



So, this is the graph at the end of this whole process of generation that A star does, it has inspected 14 nodes all shown in this orange brown kind of color much more than what best first search did. Best first search explored only 8 nodes, but best first search found a path of cost 195 if you remember and if you go back to the slides whereas, A star has found a cost of 148 which is of course, much better than that.

And I will urge you to compare this with branch and bound and Dijkstras algorithm which are known to find guaranteed which are known to guarantee the optimal path, you will see that in this particular example at least A star has also found the optimal path.

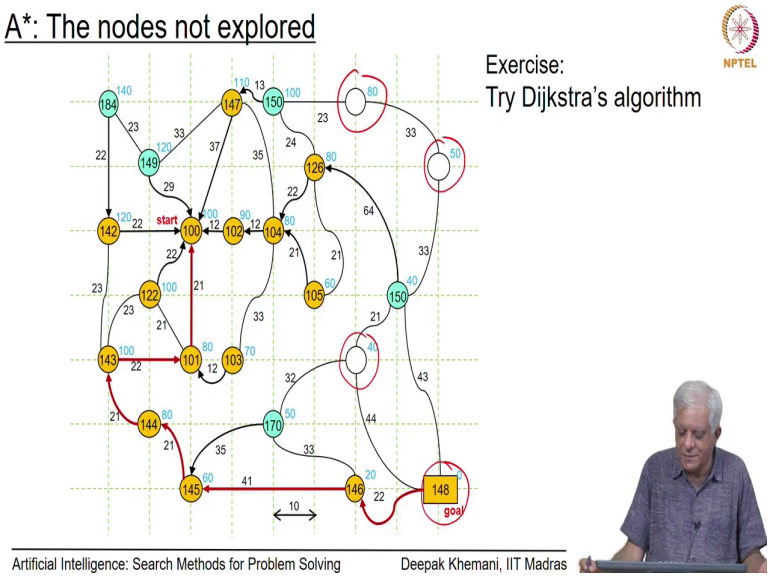
Why it has found the optimal path is on the conditions that I had mentioned we will put on the heuristic function, but we will see that I have chosen the heuristic function in such a way that A star does find the optimal path, we will discuss exactly what those conditions are under

which A star is admissible as we say or it finds optimal path, but we will do this formal proof in the next session.

Meanwhile, it this is a graph that it has explored these dashed edges are edges that it also inspected, but ignored because they were not path of the optimal path like thus Dijkstra algorithm for every node it has a back pointer showing to what is the best way of reaching the node from which parent and so on.

And the optimal path has been shown in this red thicker arrows and you can see that it is a totally different path from the path found by best first search this had to in some sense go to a node which appeared to be going in the opposite direction because that is how it found the path which is the shorter length essentially.

(Refer Slide Time: 29:26)



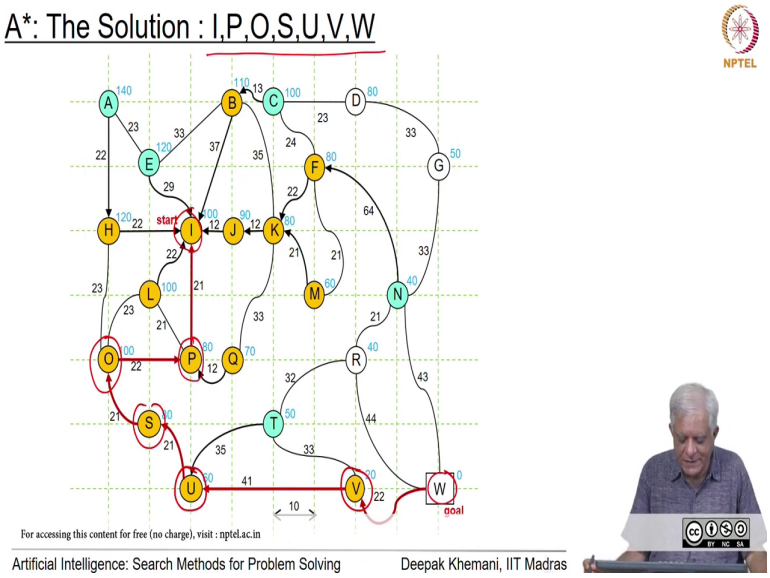
You can see that there were some nodes which this algorithm did not explore at all these are the nodes which have no filled in color and as an exercise please try the Dijkstra algorithm on the same problem and confirm that this cost of 148 that this algorithm has found is also the cost found by Dijkstra algorithm.

In fact, the same path will be found by Dijkstra's algorithm, but it would do more work than A star because it would explore some of those nodes which we have not explored using A star.

So, you can see that there is a gradation of behavior Dijkstra does the most amount of work in terms of the number of nodes that it will explore guarantees with the optimal path, A star does less work than Dijkstra's it explores fewer nodes 14 in this example, but under certain conditions which again I am we will study in the next session it will also guarantee an optimal path.

Best first search does the least amount of work, but it does not find the optimal path and we can try and see that are there behaviors between best first and A star which will do less work, but hopefully give you optimal paths or at least it would have a better chance of giving an optimal path even if they may not guarantee the optimal path we look at variations of A star they do that as well essentially.

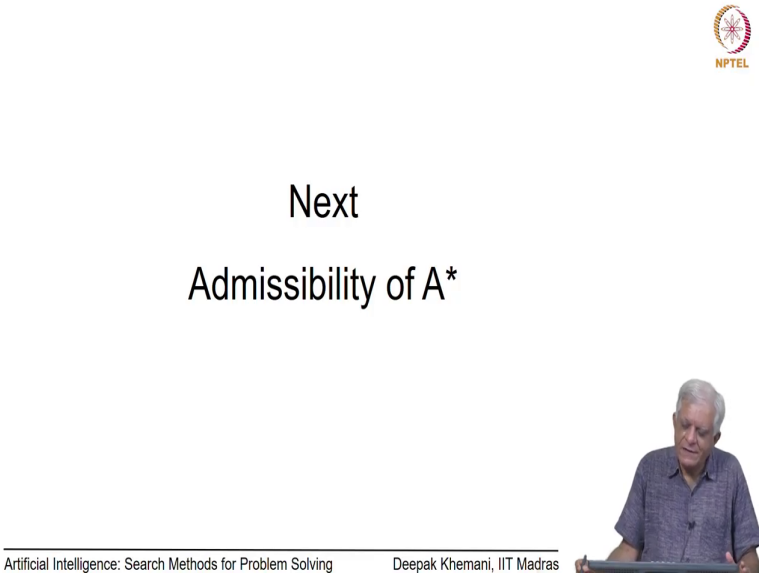
(Refer Slide Time: 31:03)



So, if you were to list the path that has been found you can see that the path is given here, it is the list of nodes starting with the node I the node P which is on the optimal path essentially. So, if you are asked to list the nodes then this is the list that you would be expected to give in for example, if you are writing an exam or doing something like that.

So, I hope this has given you some good insight into how these different search algorithms best first hill climbing Dijskstra's A star they behave on similar problems, we will look at some variations as we go along particularly we will look at variations where you can save on some space because we will as we will discuss A star also the space complexity depends upon how good your heuristic function is and heuristic functions are not always very good and therefore, maybe we have to adopt different methods to save one space, but we will come back to this later.

(Refer Slide Time: 33:20)



What we will do next is to see as to under what conditions A star guarantees an optimal solution we will give a proof of the fact that under those conditions A star will always find the optimal solution and we will also show that if you have a more informed heuristic function which is closer to the actual cost of a path then A star will do less work in terms of number of nodes that it will explore for finding the optimal solution.

So, we will come back and do some of this formal stuff in the next session. So, see you then.