Artificial Intelligence

Prof. Mausam
Department of Computer Science and Engineering,
Indian Institute of Technology, Delhi

Lecture - 20
Information Search: Admissible Heuristics and Domain Relaxation - Part 6

We should get started. So, in the last class, we looked that affair number of algorithms, which were in the category of informed search algorithms. So, the idea of informed search algorithm says that in addition to the search space, which is given through this expansion function and goal test and so on. The domain designer additionally gave us a heuristic function. This heuristic function h is defined over every node.

And it estimates what is the cost of the nearest goal closest goal from this node end, and then we studied how would we use such a heuristic function if it were given to us and we discussed the A star algorithm or the general idea of best first search algorithm which generalizes even best first` search and uniform cost search. But we extended it to be the best first search which uses the heuristic function A star algorithm which also uses the heuristic function

And also has nicer properties in particular, we can guarantee that A star algorithm is optimal in the tree search version is the heuristic is admissible, and then we went beyond and we said the memory problem of the algorithm remains the same. And so can we come up with versions of algorithms which use the heuristic function, which do not use that much memory and we talked about idea of A star and depth first search branch and bound.

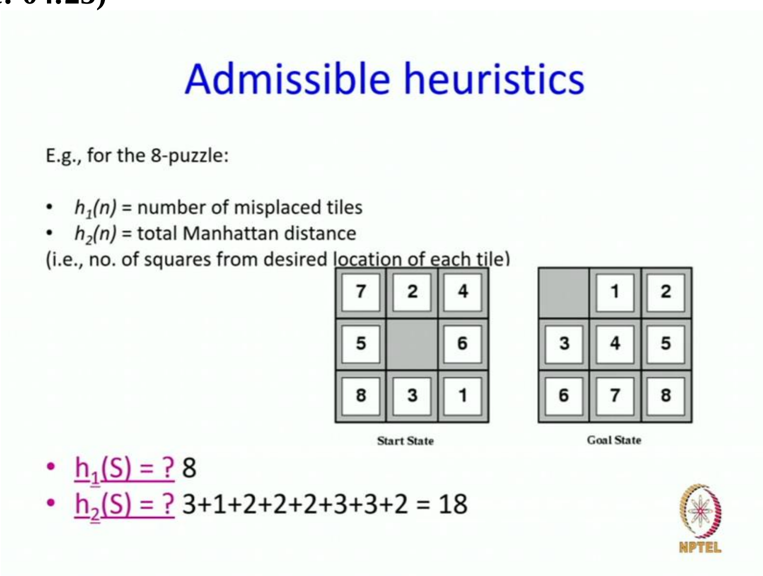
Now, last but not the least we said optimality is a luxury. The god knows the optimal solution. But these are hard problems; we may not get to the optimal. So can we come up with some suboptimal algorithm, but which is good enough, at least has an optimality bound. And we talked about (())(02:19). So this is what we were, we did in the last segment. And to complete the story, one important question remains.

How does the domain designer come up with a heuristic function? In particular, we are interested in admissible heuristics by the way, there is also a lot of work on coming up with inadmissible heuristics. Now when we use an inadmissible heuristic, all the bets are off with respect to optimality. It is okay optimality as I said is a luxury. So, all bets are off with respect to optimality these are sub optimal algorithms again.

But sometimes in practice in inadmissible heuristic, which is more informed is much better than an admissible heuristic which is less informed. And when I say less informed, I mean that when it estimates the distance to the goal, it is sort of making it sort of far away from it. And we also discussed, what is the most admissible heuristic possible, most uninformed heuristic possible and that is what we discussed this yesterday.

What is the most uninformative admissible heuristic? 0, and if my most uninformative admissible heuristic is used, so if 0 is used, what does that my algorithm become? A star becomes uniform cost search because A star uses fn = gn + hn if hn is 0 A star uses fn = gn which basically means it is uniform cost search. So these algorithms are in sort of a composite hold and services they are all extensions of each other etcetera in some ways. So the goal for today is to think about how can we come up with admissible heuristics given a new problem?

(Refer Slide Time: 04:25)



What is it is there some theory behind is there some mechanism is there some thought process that allows us to construct admissible heuristics. And we will start by discuss first reminding you what is an admissible heuristic. So, an admissible heuristic is an optimistic heuristic. So, what is an optimistic heuristic and a heuristic which feels that goal is closer than what it looks like all I will make more money than I will actually end up going to make.

So in other words, if I am in a minimization problem and admissible heuristic is always a lower bound. And if I am in a maximization problem, my admissible heuristic is always an upper bound. But we will let us discuss there is the principle stay. But let us discuss only the minimization problems so we are only in the minimization world. So, we go back to our favorite 8 puzzle problem and we asked the question.

We have to go from start state to a goal state or the goal state in this case, how can we or can we estimate a heuristic for a node, so, the heuristics job is to estimate how many steps I am going to take to reach the goal. Now, I would need suggestions from your please do not shout raise your hands. But can you think about a think of heuristic, which is admissible, that is, it is a lower bound.

It underestimates the costs that I am going to need to go from, let us say the start state to the goal state any suggestions? Yes. What is your name? Symon, Yes. Number of misplaced tiles very good. So Symon one suggests that look, lots of tiles are not in the right position in particular, how many 2 4 5 6 8 3 1. All of them actually, all the 8 tiles are in the wrong position. And Symon says for this, I will at least need 8 steps to go from start to go.

Is he Is somebody from we already know this is not an admissible heuristic, this is not an underestimate. He is missing something. Nobody, why is he can you prove that you would at least take these many steps? How would you prove that each of the pieces that have to be solved with the gap. Now, if all the pieces or whatever the pieces are in misplaced positions, they have to come to the correct position.

It will at least take one step to come there independently because we are always swapping with the gap. So it is not possible that in 1 step 2 tiles get to the current position it is just not possible in this game. We are all always moving 1 tile at a time. So if we are only moving one tile at a time. And I have so many misplace tiles that I would at least need so many steps. So is it an admissible heuristic? Yes. Good, not very difficult.

So you can say, come on, no, we do not need a theory behind it. These are easy can you come up with a better admissible heuristic? Something that is better than side 1 heuristic better means that it is still admissible. But the cost that you get the heuristic function that you get is typically bigger. And therefore it is closer to the more or the actual optimal solution. Yes. What is your name? I am Anketh, yes Anketh.

The Manhattan distance between what? What do you mean the Manhattan distance between the final position and the initial position into puzzle problem some of them for each of the tiles so he bases some Anketh says, you look at each tile, you look at the Manhattan distance between the 2 positions. And you sum them over each tile and that will be an admissible heuristic. Now, before we understand this further, let us make sure that we all understand.

What is Manhattan heuristic or Manhattan distance Manhattan is, of course, a great, very famous place in New York. It is said that if you just stand there in Manhattan, you will hear all languages in the world. Of course, that is not true. There are many, many languages which are not extinct. So nobody speaks them. By the way, I do not know if you know, we are losing. Like a few languages every month, the world had 7000 languages I do not know how many we have now.

So we are losing a few languages every month, may be 2 or something, I have to check the actual numbers. But, that is a digression. So Manhattan has been engineered or has been designed, very sort of rectangular. So you can go if you have to go from one place to the other place, you will basically go in 1 direction and then go into the perpendicular direction going. Parallel direction, perpendicular direction.

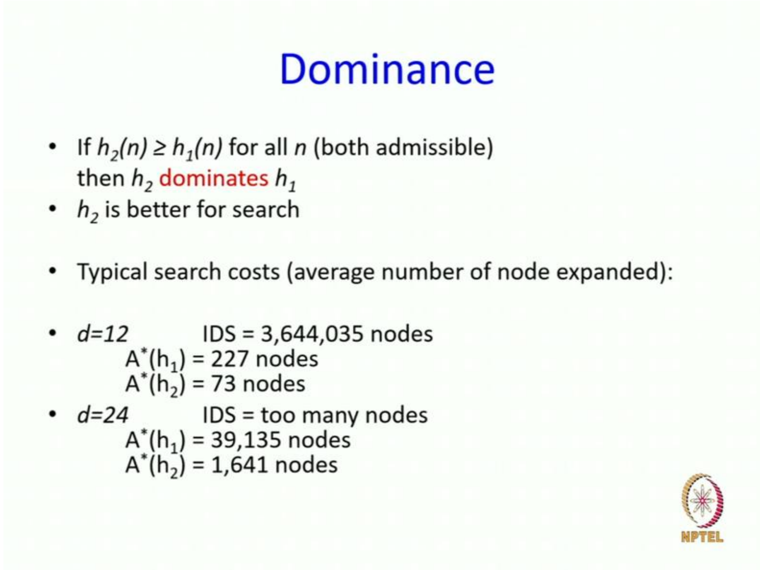
You can keep doing this until you get the final destination and you can do all parallel first and then perpendicular later all perpendicular first and then parallel later, or in any which way and you will get to almost the same distance. Because these are all equal equidistant. These are it is a very well planned part of the neighborhood. It is the downtown of New York. By the way, in India, also we have some cities, which are very well planned.

One of them is Chandigarh, so we could call it Chandigarh distance, but that would not have the same bus as Manhattan distance. So we will stick with Manhattan. So in the case of 7, what is the Manhattan distance 3, because I need to go 1 right and 2 down, I can go 1 down 1 right 1 down 2, down 1, right 1 right 2 down, but we will always end up with that distance 3 so that is the Manhattan distance of 7. So if we have to compute the heuristic, I will add 3 for 7, I will add 1 for 2, I will had 2 for 4 and I will keep doing this for all the 8 tiles.

And I will get a heuristic function. So, these are the 2 definitions of my heuristics h 1 and h 2, h 1 is a number of misplace tiles h 2 is the total Manhattan distance in this case h 1 is 8 and if you count h 2 will come out to be 18 now, some questions about h 2. Is this an admissible heuristic? Is this an admissible heuristic? Is this an admissible heuristic? Why? Because, in this Manhattan while you are all only allowed to make right and bottom or top and left moves.

And the shortest part would be you know, going using the Manhattan distance for that particular nodes that many steps it will have to take it is possible, it will end up taking more because there is a tile there and I have to clear that gap. And that increases the line. But you cannot do less than that. And you can prove it formally. Now, which of these 2 heuristics 2 think is a better heuristic h 1or h 2 why? Realistic is a good word, but let us be each term in each 1 will be less than equal to h 2. And both of them are admissible.

(Refer Slide Time: 12:37)



Therefore, overall, we can say that the h 2 will always be greater than equal to h 1. And h 2 is always as an equal to h 1. That means and if they are both admissible, that means h 2 is quote unquote, more informative than h 1. And the technical term for this is h 2 dominate h 1. If I had h 2, I do not need h 1. And now here is a quick example of what happens when I use a heuristic function in comparison with let us say, iterative deepening search.

So let us say I am in the example where the depth of the goal is 12. They are additive deepening search does solve the problem, but it expands 3 million nodes. That is a lot of nodes to expand. If In comparison, you see that A star h 1 requires only 227 months, and A star h 2 request only 73 nodes. Think about we cannot prove anything formally per say in the worst case things will be explanation.

But in practice A star algorithms are superior by a magnitude like a large magnitude for most problems, if you have a good heuristic, and here h 1 is not even that good a heuristic as you can see, if the goal is death 24 the goal is that depth 24 identity deepening search never get set. A star on h 1 does get it after 39000 expansions whereas A star h 2 gets it only after 1600 expansions. So a more informative heuristic more often than not leads to faster running time.

However, there is also a theory that more than heuristic can sometimes hurt, but we are not going to, again go into that territory. For our practical purposes more often than not, if you have a better more informed heuristic, you will probably get to a better solution faster same solution fast. So now these are some examples of admissible heuristics in the context of the 8 puzzle problem.

But that does not get us to the general principle. We need to somehow get to some general principles that given a new problem allow us to come up with an admissible heuristic quickly. And that general principle is the notion of a relaxed problem. Let me first describe what a relaxed problem is. And then we will think more about what it means in the context of our 2 heuristics. So, a problem with fewer restrictions is called a relaxed problem.

So, think about it, there are some rules of the game. I have to satisfy all the rules of the game to solve my problem, but let us say I take away one rule for you. Any rule I take away one of those rules. Now the problem that I am left with is called a relaxed problem. It says relaxation of the original problem. Have any of you heard the word LP relaxation, some of you may have you can connected with that if you want to solve an ILP.

An integer linear programme that means every solution has to be integer and then you do an LP relaxation which means that I am not solving an integer linear programme only I am not solving a linear programme that means, I have removed the restriction that my variables are integer am I removing some of the rules and removing the rules that all my values should be integer. Such a problem would be called relaxed problem.

And because it is all about using LP It is called LP relaxation. This concept of relaxation is actually a very important idea whenever you are given a new problem, because a new problem if it is a challenging enough problem will have lots of issues, lots of little things that you need to take care of and initially not having intuition. Even just to do intuition building you say let me get over get rid of some these rules, let me solve a seemingly simpler problem.

Let me at least solve that and get an intuition about what is happening in this problem, then I will think about you know, how to solve additional problem that is just a strategy that you will use as an intuition, but actually there is an operation of that in the context of search algorithms is relaxed problems are actually critically important in computing admissible heuristics can somebody guess why?.

So, the solutions of the relaxed problem, any solution, optimal solution the optimal solution of the relaxed problem wait. Can you say something about that, with respect to the original problem, it will always be either executable in the original problem or not executable. But if it is not executable, will it cost be less or more why less? Every solution very good wishes you every solution of the original problem is a solution of the relax problem.

Because I have already removed some rules have not added any new rules. So any solution of the original problem is a solution in the relaxed problem, if I am solving optimally the relaxed problem, that means have considered all the solutions also and all the new solutions which are not even executable. So the policy that the solution that I am going to get would be better than the optimal solution of the original problem in the relaxed world.

Therefore, its cost will always be less or less than equal to the original. In other words, the general principle here is the cost of an optimal solution. In the relaxed problem is always by definition man not by definition, but by our analysis is going to be an admissible heuristic for the original problem at. Now, let us intuitively think about this. What are we relaxing when we say h 1? What role are we relaxing with? Let others think.

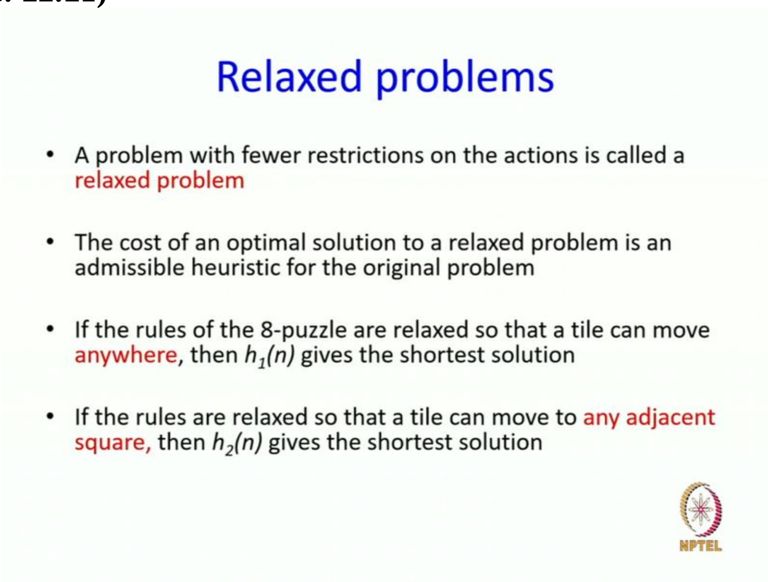
What are the rules in 8 puzzles? We can only swap with an empty space very good. What is the name? Shivan says we can only swap with an empty space. What is another rule? That is what all you can only swap with the neighbors. Ansh so much basically Ansh and Shivam have given us 2 constraints of the problem you can only swap with. And you can only swap the gap with a neighbor.

Good now let us think about this in h 1 what are we relaxing we can only swap with a neighbor. So if I have to, if I allow my tiles to fly, but I was still only swapping with gaps what will be the optimal solution here? The first thing I will do is move for I will put 4 up I will have it fly and go into the gap. Now 4 is in its correct position, it took me 1 step. And now the 4 original position is open.

Where I can move 2 so again 2 steps I have been 4 in the correct place. I have position open so I can fly 1 and so on so 4th. Now, by relaxing the assumption that I can only move swap with the neighbors, I have created a relaxed and in the relaxed problem the optimal cost is going to be and that is my h 1 heuristic, therefore, it is admissible in the very same way. Which constraint am I relaxing when I get to h 2 you can only swap with a space.

So, that means I can or I would say not swap, but you can only go into a space. So swapping means that if I am swapping 7 and 2 then 2 comes left and 7 comes that is trickier, but we can say you can also say that one of my constraints is that there is only one tile at a location. I will add to the last both of those so I can say, I will move 4 is first 4 will go to 6, 6 is also there. That is who cares 4 and 6 are both sitting and then 4 comes left and it gets into its correct position. So, I can move over tiles. So, these are the relaxed versions.

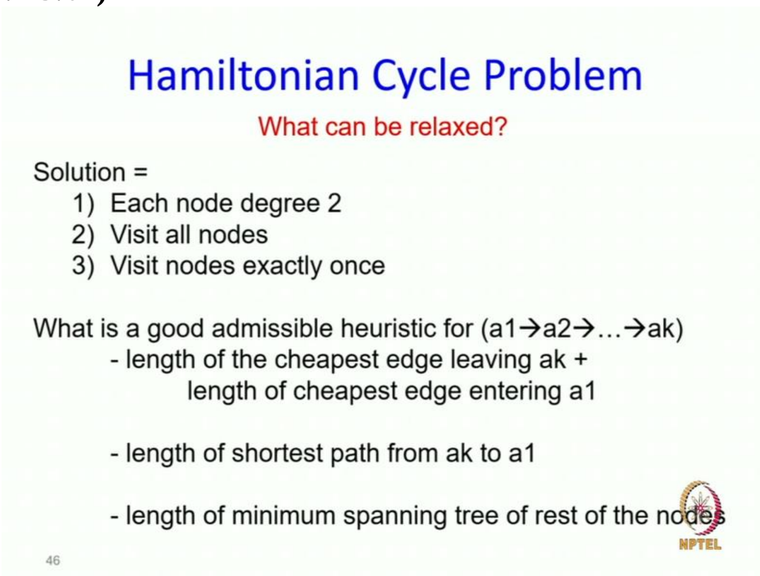
(Refer Slide Time: 22:21)



So, as it says in this slide, if the rules of the 8 puzzle are relaxed so that a tile can move anywhere then h 1 n and gives the shorter solution. if the rules are relaxed so that a tile can move to any adjacent square in spite of whether there is a tile there or not? Then 8 puzzle gives me this shorter solution. So therefore, you can think about so whenever you are given a new problem, you have to analyze what constraints are going on.

And based on those constraints, you have to say, let me create a relaxed problem and based on that come up with an admissible heuristic, and do one other example just for fun and that is the Hamiltonian cycle problem.

(Refer Slide Time: 23:02)



So first of all for people who do not know what is Hamiltonian cycle is a Hamiltonian cycle you have to visit all nodes starting from some node exactly once and come back to the original node. So, for example, you can do it in the order a1 a2 a3 a4 a1 if therefore nodes in the graph all you can do a3 a1 a2 a4 a3 and that will also be a Hamiltonian cycle. So, you are looking for the cycle starting any node which comes back to the same node visiting each node exactly once everybody with me?

And as we know we can model any problem a search problem at least these kinds of deterministic single agent problems. So, what will be my node what do we my state here the first thing, the state space my state spaces any suggestion set of visited states so far. But set of visited is not necessarily the because we need to maintain what is the order because we have to maintain the costs somehow and also we have to know which is the endpoint.

So, we can say a partial part, a partial part could be my state space and what is my start state empty nothing and my edge would be expansion and the expansion would be what expand into. So suppose I currently have a part a1 a2 a3 up to ak in my state space in my state, then the next child would do what visit and unvisited not from ak. So, I will go to from a1, a2 a3, up to ak 2 a1 a2 a3 up to ak + 1 and ak + 1 can be one of the many.

So, therefore, I will have many children so this is my state space and now my goal is to figure out an admissible heuristic for the state let us a a1 up to ak. And in order to do this and for any such problem that comes along think about solution constraints. So, what are the constraints on the solution? Each visit each node exactly once what else visit all nodes so, not only visit an node 1 but visit all of them.

All you can say that the final solution that degree of the notes should be degree of each note should be 2 because one will be incoming when will be outgoing for every node. So this becomes now my constraints on the solution. And now my only job is to see if I can relax any one of them and based on that come up with a good heuristic. So any suggestions on heuristics? What if I get rid of visit all nodes? What can you create what heuristic can you create? Start from this reverse start from ak and end with a1.

And with any part with any part using which nodes remaining nodes so get rid of a1 to again. The ground a2 to ak - 1 in the graph and then try to compute the shortest part from ak to a1. And that is going to relax visit all nodes constraint. Can you relax any other constraints? The there is a nicer admissible heuristic for this problem. Let us see if you can figure this out. Let us say I allow you to relax. Each note has degree to constraint. But you still want to visit all nodes. You have to be louder, please good.

So the suggestion is that if degree is more than 2, then third concern also gets violated I agree. So let us violate all of them. Not all of them, both of them first and the third constraint can you think of something that you have studied which does not return something with degree to but has some notion of shortest optimality and something like that yes Sukuruthi minimum spanning tree. So let us think about what is minimum spanning tree minimum spanning tree on the rest of the nodes. So on the let us say ak to a1 nodes.

Not using a2 to ak - 1. This minimum spanning tree would it be an admissible heuristic of the original problem? What is it saying it is saying visit all nodes yes, but should all nodes have degree to no should all not be visited exactly once? There is no it is a tree so there is no notion of visiting for say, so not exactly. So, but will the original solution the part from ak to a1 will that also be a solution in the minimum spanning tree problem will that also be a spanning tree it is it is a spanning tree it is spanning every node, it is not the minimum spanning tree necessarily.

So, the minimum spanning tree cost will always be less than the cost of the part from ak to a1. So, that is the intuition of how you come up with admissible heuristics. It is a very general intuition it is a very valuable intuition. Yes question. Original solution is a cycle but for the node what is the remaining solution? The remaining solution is a part from ak to a1 not using any of the a2 to ak - 1 but using every other node exactly once.

So, we are now at Interested in estimating the cost of this node to the goal which is addition all the other edges. So, you have to think in terms of the partial state that you have defined and the heuristic on top of it, but yes, original solution is a cycle. Exactly. Good. So, this sort of ends our discussion about domain relaxations will stop you.