Artificial Intelligence for Economics

Prof. Adway Mitra

Artificial Intelligence

Indian Institute of Technology Kharagpur

Week – 02

Lecture - 08

Lecture 08 : Heuristic Search Techniques

Hello everyone. Welcome to this course on Artificial Intelligence for Economics. I am Adyay Mitra, an Assistant Professor at Indian Institute of Technology, Kharagpur. And today is our lecture number 8 in this course. In the past two lectures, we have been discussing about optimization techniques, constrained and unconstrained optimization. Now, today we will slightly change tracks and we will talk about different class of problems which also frequently arises in this domain.

we will discuss these problems which are related to heuristic search. So, like as we mentioned several times in the last two lectures the task of optimization often arises in economics when we are especially considering the resource allocation problem. That is to say we have like we have some resources which we want to distribute or allocate to different sectors with the hope of optimizing some possible some outcome. However, we may have different kinds of restrictions on which kind or what kinds of allocations are possible.

Now, here also we are discussing something which is not very unrelated to that, but we like here you can consider it like in a sequential problem. Let us say that like at any given point we have a system let us say like we are at the system is described by state of the system. Now, what is the state of the system it is something which describes the system. So, let us say that we have a number of sectors and to each of the sectors we have allocated some amount of money. So, that is the current state of the system I mean how much money has been invested in which sector and what is the like how much output I am getting that is the current state of the system.

Now, like let us say I am not satisfied with the current state of the system I want to reach some goal I that is there are some states of the system which I consider as very ideal that is let us say that both of the like let us say all the different sectors have been able to generate some minimum utility value may be that is something which I consider as a goal state. So, I want to reach a goal state, but right now I am not in the goal state. So, just like

in the numerical approaches to optimization I start with any initial point, but we generally step by step I we try to go towards the optimal point. Here also I am doing the same thing I am starting with any with some initial condition initial state of the system. Now I will try to gradually perturb the system by changing the system one step at a time so that I gradually move towards a desired or goal state of the system.

So, this is represented this kind of like the kind of algorithms which I am going to describe for this approach is known like is basically centered on the concept of graph. Now, what is graph you I am sure most of you are familiar with graph anyway it is a collection of vertices some of these pairs of some pairs of vertices may be connected by edges. For example, if you see here we have 6 vertices these blue bubbles they are the vertices. Now, some pairs of them like this and this they are connected by edges, but not all pairs need to be connected by edges for example, these two they are not connected by an edge ok. Now, these edges they can be directed or undirected as the as you can see in the diagrams.

Now, in the like in the example which we are just discussing each of these vertices they may indicate one state of the system and neighbor I will say that one let us say let us consider this vertex. So, it represents a state of the system and then it has these edges it means that from the current state of the system it can be perturbed to a different state. And, that state might be represented by this variable by this vertex or by this vertex, but it cannot be spurt up to a different state like say this one or this one. I mean these are also valid states, but from this particular state you cannot reach these in one shot. Even if you have to reach you will have to do it in multiple steps like this.

Say for example, like in these graphs let us say the I am currently here my this is my state and I want to take my state to here. Then now since these two do not have a direct edge between them. So, like it is not possible for me to change the system from here directly to here. So, what I can do is I can move in a step by step way from here I go here and then from here I go here or from here I go here and here I go here. So, it is like you can say that if a direct journey is not possible then we do break journey.

So, that is like a break journey in this case is nothing, but a connection like a sequence of these edges ok. So, this sequence of such edges this is known as a path in case of a graph. Now, in if it is a directed graph then a path is like a sequence of edges I mean directed edges such that only these kinds of. like tail to head connections are possible that is once I go from A to this node then from the from A to. So, this is I let us say I am currently here there is an arrow from here to here.

So, from A I can come to this node, but I cannot go here because there is no arrow from here to here. There is an arrow from here to here, but that is not the same as an arrow

from here to here. So, I cannot go to C. So, from this node I can only go to P ok. So, like in case of our under so, this is like the directions of the edges we have to consider in case of directed graphs, in case of undirected graph of course, there is no such problems.

So, here like you can understand that there is no like in the directed graph the there is a path from a to b that is a to this node and then this node to b. But if you consider the this ac path then you cannot find a path from a to c that is from a you come to this node, but from this node you are not able to reach c because there is no edge like starting here and ending here yeah the reverse is there happens So, like in the undirected graphs however you can see that both a b path like this as well as the a c path like this both of them exist. So, that is the difference between a directed graph and an undirected graph. Now in any graph If there is a path from between every pair of vertices, then we say that the graph is a connected graph. So, what is a path? Just to recap a path is a sequence of edges like this.

If it is directed edges, then only these tail to head like movements like this are possible. like from if I am calling this middle central node as let us say E then from A to E I have a edge, but I do not have an edge from E to C. So, I cannot say that A and C are connected by a path, but in the same thing I can say in case of the undirected graph also. So, now what now we have a graph like this what do we do on this graph. So, we may start at a particular vertex which we can call as the root node and our aim is to discover the other vertices by following the edges.

So, that is the graph search problem. It can also be that instead of just exploring the graph I may be looking for some specific vertices also. So, in the in graph theory there are multiple algorithms which are provably correct and provably converts to the correct solution like. So, some of these are known as the breadth first search algorithms that is to say whenever we reach a vertex we note down its unexplored neighbors and then visit them in that order. So, like when I come to let us say 2 if I reach this vertex called 2 here then it is like from there we have some like some other vertices I mean these and these they are all neighbors to this vertex where 2 is written on it.

So, like I can like consider that from 2 I just put like these 3 vertices I keep in a list and then I just visit one of them at a time. So, like when I come here then maybe I will come across some other new vertices which are linked to 2. So, those new vertices also I will add to the list and then again from the list I will find one more vertex and I will join there and so on and so forth and the whole this process will just go on and gone. So this is the graph search approach. Now another question that frequently arises is shortest paths in the weighted graph.

So what is the weighted graph? A weighted graph is slightly different from a graph like

this because in this case not only we have edges but we also each of the edge has a weight on top of it. And, the weight of a path a path remember is a sequence of edges. So, the weight of a path also we can define as simply the sum of the edge the weights of all the edges which form the part of the path ok. So, this way we can get like define the weight of the path. So, shortest path algorithm is like in in a given vertex.

sorry in a given graph we are provided with like a source vertex and a target vertex. Our aim is to find the sequence of moves which I must make in order to reach the that target or goal vertex. So, that is the shortest path algorithm. Now, already there are several algorithms in graph theory once again for calculating the shortest path between any pair of variables or let us say from the from a fixed origin to all other vertices. So, these kinds of algorithms are known as the Dijkstra's algorithm, the Floyd Warshall's algorithm and so on.

So, that is about the shortest path in a standard weighted graph that now suppose the thing is the algorithms which I just mentioned they are very good algorithms they are very solid and efficient algorithm plus they are provably correct. That means to say like if the whole graph is known then they will always give us the correct answer. They will always converge to the like the exact they will converge to the exact like exact solution. It will be able to find the path which truly minimizes the cost ok. But now as I said here we have made an assumption.

The assumption being that the entire graph along with all the vertices and edges and their weights are known beforehand. But, often it happens that this is these are not known beforehand the entire graph we do not have access to, but as we reach a vertex then we can access only the neighbors of that vertex. So, that that is a restrictive criteria. So, now like for example, consider the game of chess. So, let us say each of these vertices they represent the state of the game.

Now, I can from one state I can make one of the moves which are allowed according to the rules of the game and I can move to another like another state of the game. And in case of economics also like let us say that we some allocation is there from I make some changes that is I transfer some money from one sector to another. So, I move to a different state. So, but now it might happen that I do not have an idea of total how many of these states are there from one state which all states I can go and so on and so forth. But when we come to one particular state, then we may want to find like what are the possible next steps that are available.

So, this is how the I mean the graph the entire graph is not shown to me right away, but only but as I keep traversing through the graph from jumping from one vertex to another new vertices are progressively revealed to me. So, in such situations these algorithms like

Dijkstra or Bellman Ford they will they may not be able to work. So, then like what we need is what is known as a heuristic search technique. So, now what is a heuristic search technique? The heuristic search technique is basically the following. So, we at any given state that is at any given vertex we have an estimate how far that state or that goal vertex is.

from that state or that corresponding vertex is to any of the goal vertices that is let us say I am currently here. So, that this node from here to this goal state the path length is obviously 6 from here to here if you want to go then the goal I mean then the path length is equal to 8 and like and from here if you want to go like this. Then the Basland is equal to 9. So, then what is our task our like or what happens in heuristic search the heuristics in case of heuristic search at a whenever I am at any of the vertices I make an estimate of how far the goal state is from it that the estimate which I am making it may not be the right estimate or it may not be the right value it is just a guess which I have made. So, in this case for example, the numbers which I have written down in any of these vertices these are numbers are the estimate.

So, from here as I already said the length of the path from here to the goal node is 6, but instead I have made I do not know that because I do not have know the graph structure, but I have somehow made an estimation that it is equal to 5. in this case also I have made an estimate that is equal to 4 and so on and so forth ok. So, these values these are known as the heuristic function. So, what is a heuristic function? Heuristic function is basically you are at this like any particular node let us say x. So, what is the projected cost from of reaching the or any goal state for from that variable from that node x.

Now, so if we are going to consider path from the vertex from the start vertex to a goal vertex via the your current state which is x, then the total cost you will incur for that path is of course, is let us call that as $f(x)$. So, that cost has two components one is the cost of going from the origin or from the starting point to the current node x and which we call as the $g(x)$ and the second part is called as the $h(x)$ which is basically the cost of going from the current vertex to one of the goal nodes. So, now we come to well known algorithm for such heuristic search which is known as the A^* algorithm. So, I will explain the A^* algorithm like as follows, but that is it can be written down in the form of an algorithm, but I think it is easier if we discuss it through the example below. So, what happens in this example? So, we have this graph like this, this green nodes they indicate that they are the goals that is if I reach any of these it is enough and what is A? A is the starting point of the whole thing that is I am it is the state at which I currently am from here I will go to one of these variables and I mean I will go to one of these next states and so on and so forth.

So, the value of the heuristic functions like $h(x)$ for all of these vertices has been provided to us in case of the goal nodes g and h the heuristic function is not needed because they are already the goal. So, the heuristic function value for them is going to be 0 that is since we are already at the goal then there is no cost of reaching a goal right. So, now for the all the remaining points a b c d e f we have the values of the heuristic functions at each of them. So, now what we do our aim as I already said is to find the shortest path from the start node a to any of the goal node goal nodes either g or h. So, now how do we proceed the a now what does the a A^* algorithm say the A^* algorithm says from the from your current state you like if like if the current state is already a goal then you just stop.

But if that is not the case which is not the case in this case obviously list down your neighbors and add them to a list called close. So, who are the neighbors of a obviously c and d. So, c and d I now add to like I mean they are already in the table, but they are that is like earlier i was considering the $g(x)$ for c and d as very high because i did not like like basically i did not know the their values so i just i mean so so what is meant by gx gx is the cost of reaching these variables i mean these these nodes from the starting point so since i did not know earlier that there exists a path from a to c or a to d I just assume that I just said the high value of 100 here assuming the worst case scenario. But now I have seen now that I have seen that C and D are the neighbors of A and AC length is 2 and AD length is 5, then we can actually make some changes. now i know that like this the g values of c and d need not be a very high that is one can easily reach c and d both from a with cost like 2 and 5 which is shown in the graph So, the like once I start out from A and I find its neighbors are C and D.

So, I like explore the neighbors. So, now you see that the G value of these neighbors which were earlier 100 before they were discovered now that has gone down. Why because I know that there is a path from A to C as well as A to D and we know the cost of those paths also. So, now if I want to estimate the value of f at a c and d. So, what is meant by the value of f at c it means the what is the least cost of a path that goes from the start node a to one of the goal nodes g or h via c. So, it and as we have already discussed $f(x)=g(x)+h(x)$.

Now like h x we already have calculated $g(x)$ is like also now known to us I mean $g(x)$ is like basically we are considering the cost of coming to a and then to that I am adding the cost of coming to c. So, the $g(x)$ is now equal to 2 because a is the start note anyway. So, there is no cost of coming to a, but from for going from a to c like this a to c I have I incur a cost of 2 and similarly for going from a to d I incur a cost of 5 which is also noted here. Now, in addition I am since I am interested not in the length of the subpart from a to c or d, but I am interested in a path from a to either g or h. Now, I have I am considering

only those paths which pass through c and d and for each of those paths I am estimating the possible value of f.

So, applying the formula that is $f(x)=g(x)+h(x)$ we find that the f score for both c and d are 7 and 6 respectively. So, what does that mean? That means that like my heuristic function note the again reminding that heuristic function is a guess of how far the solution is. Now, if we but if we trust that heuristic solution it is suggesting to me that there exists a path from A to a goal node via D whose length is 6 and there also exists another path from A to a goal node via the vertex C with the $f(x)$ value of 7 ok. So, and like in both cases I note down that if for C and D the its ancestor is A because it is we started from A and we came to C and D. So, this is what the table looks like at this point.

Now, what we do since we have since we our estimated path cost from A to a goal node via D is better than that via C. Note that in case of D the our estimated total path length is C 6, but while the other two that is C and E this quantity has gone up to 7. So, now what I so this A^* algorithm this is known as the best first search. So, which means that whatever option appears best to me right now I take that option. So, like I move so since it seems that a path via d is cheaper then I just move to d.

So, that is the thing which I do. Now, once I have reached d from d I can like find some other points. So, what are these other points? So, note that from like from A I found C and D. Now, I have decided to go to D. Now, once I go to D, I will find its neighbors. Now, D actually as you can see has only one neighbor which is equal to E.

So, I have now found a path from A to E. So, I will now examine this path further that is first we already know that the heuristic function at E at node E that is equal to 1. That means I have a reason to believe that from node E I can reach a goal node in just one step. So, but on the other hand what is $g(x)$? $g(x)$ is like the cost of reaching the node d sorry the cost of reaching the node e via d. So, like obviously you can understand that I have come from a to e in that process I have incurred a loss of 6 which is which I store here. Now using this $h(x)$ and $g(x)$ I calculate $f(x)$ according to the formula $f(x)=g(x)+h(x)$.

Once again reminding what is $f(x)$? $f(x)$ is the estimated path length from a to a goal node via e. What is $g(x)$? $g(x)$ is the path cost of the path already traversed that is from a to e and what is $h(x)$ is the projected cost of the of an optimization path starting from the current node e and reaching a goal node g and h ok. So, this is where we stand right now and we continue this process. So, I reach c sorry I reach e and I find like its one of its neighbor is goal and I also find that one of its another of its neighbor is f ok. So, for f also I like you can say that now that I have that is I now that I have come to e I find that

the possible solutions that is I that is I have now added this equation this sorry I have added this variable or this node called E into my into an open list that means it is a vertex which I am considering to visit.

So, now from the list of my open variables note that my earlier these two variables were open C and E. So, from these two I like I will now choose any one of them for C I had already calculated the $f(x)$ was equal to 7 for e I just calculated that $f(x)=7$. Once again $h(x)$ from e is 1 even though it is written as 2 here. So, again once again there is a typo and 6, 6 is like the path length from a to e via like via d this a to d is 5 and d to e is 1.

So, total 6 coming from a to e. So, the expected or the estimated length of a path from the start to a goal via e that seems to be 7. Now on the other hand for like that is I have already got d, but in the open list I already had the variable c and for that also the path length is also 7. So, I now have two options in front of me either I can explore E further or I can now explore C. Now for whatever reasons I choose to do the latter that is I now focus again on C and see what are its neighbors. From the neighbors of C among them A has already been visited its close which put in the close list.

e also has been visited and even although e however is not in the closed list its yellow color indicates that it is right now in a open list now from a from c however we can reach these two new variables f and b so note that f and b they were earlier blue now they are orange which means that they have now joined the open list. So, now, I must calculate what is the like $f(x)$ in case of both b and f I mean capital F. So, that is the value of this function. Now, you can it can it is not very difficult to calculate or understand that from b the distance to a goal node is quite high that is equal to 9. The same however, cannot be said about the point this like this the other points which we have in the open list.

that is say f and so on or e e f and so on these are the points currently the points which are in my open list are b f and e. So, for b the f value is 9 for F the value F value is also 9 and for E however, the F value is only 7. So, it suggests that if we go to E now that is if we are trying to reach a goal via E that may be cheaper than trying to reach a goal via either B or F. Now, so what do we do? We now shift to E.

So, note that E can be reached in two different ways. I can either reach it via D or I can reach it via C. So, like both of them have the same path cost of 6 each. So, had it these two path cost not been the same, then I would have chosen it to come to E. via the cost that cost I mean via the path that cost more, but since that I that is not the case I have come to E, but I am considering D as my predecessor. So, that is I could also could have considered C as my predecessor, but to break ties I have simply chosen D.

So, the path is currently A D E. Now, that E has been now that these E, F and B are in the list I now must explore their neighbors in turn. So, now between B, F and E the three the three vertices which are still open we see that E is the one which has the lowest value of F. So, accordingly we visit the node E. So, from A we went to D and now to E. Now, again we repeat the same thing for E again we consider which all neighbor neighbors it has the answer is of course, F and H apart from I mean sorry C F H and G.

Now, So, like all of these new variables I mean the f, g etcetera they like h for example, h was not part of the list earlier, but now that we have reached e and h is a neighbor of e. So, now like h also it changes color and it becomes yellow. So, now we have so many open vertices b f g and h. So, for each of them I have some estimate of the this cost function.

So, now G of course is the goal node. So, now you can see that in case of B the cost is 9, in case of F the cost is also 9, in case of G as you can see the cost is 8 and in case of H also the like we can like reach E at cost 6 and we can reach the H at cost 10. So, the like the cost for g is 8 and the cost for h is 10. So, obviously, following the algorithm as we have been following we will go to g.

So, g is our goal state. So, we stop here. So, we have now been able to find a path from A to G. So, as you can understand it is the path A D E G and its cost is 8. Note that if you consider the this path A C E G that is an alternative path that also has the same length of 8, but it was not considered. So, this is the A^* algorithm. Now, so what does this A^* algorithm give us? I mean the other graph search algorithms like breadth first search, depth first search, they are guaranteed to give us the correct solution or to like help us to search the vertex we are looking for.

Same goes for Dijkstra's algorithm or the these Floyd Warshall's algorithm of finding the shortest path. in this case also can we say that or can we prove that it will this algorithm will give me the shortest path from the start to one of the goal states the answer is yes as long as the choice of heuristic $h(x)$ that is admissible. Now what is admissible? It means that it should always underestimate the least cost to any goal state. Like for example, if you consider this node C, so we can see that from C we can reach a goal state at the cost of at most like that is the shortest path seems to be of length 6. that is basically as I said earlier also the heuristic function is the estimated path cost from C to a goal state and its value then should not exceed the actual cost which is 6.

So, I can put H of C as 4 or H of C as 3 or even 2, but it should not be say 7 or 8 if that happens. that is if I am using the heuristic function if I choose in such a way that it always underestimates the total cost to the goal node then I am guaranteed to find the

correct solution. However, if I by any chance if I overestimate then it can it is not difficult to construct an example where we will see that we will not be able to find the correct solution. So, like I leave this as an exercise to you. So, in this case I have chosen some of the heuristic functions like in case of d for example, I have chosen the heuristic function as 10 even though from d I can easily reach a goal node at a cost of only 3 that is from d to e and e to g total cost of 3, but instead I have taken 10 as the heuristic function.

Similarly, I have also changed some of these the heuristic functions at some of the other variables. So, now if it if you run the A^* algorithm the way we defined it I am not showing the full run due to lack of time, but you will find that the path we come up with is this path a c f h and whose cost is 9. So, even though the if we had taken this path a d e g its length its length is 8. So, clearly we have made a mistake we have found we have missed the actual shortest path and we have found the longer path and I am thinking that the longer path is a shortest path which is of course, wrong. So, why did this mistake happen? The mistake happened is because the heuristic function at D was set to 10 that is when I found the vertex D I by looked at its heuristic and I thought that the goal node is very far away from here.

So, I did not consider it any further and by doing so I missed a trick there the solution actually there is actually a shortest path solution from a to g via d which cost only 8, but because of the of that the the problem with this heuristic function I never considered this d at all. So, that is why it is important to that the heuristic function should always underestimate the true cost. Now, in one simple choice of this heuristic function H is you simply take the minimum cost to any of its neighbors that is from D let us say its neighbors are A and E the path cost are 5 and 1. So, you simply take the minimum value of them as 1. So, because that is a safe choice that is always going to underestimate the goal I mean the cost to the goal.

So, because like the 1 is the minimum cost at which you can reach some neighbor from D. Now, from that neighbor if you have to reach the goal that will be something even more. So, if you just say 1 as the heuristic function obviously, that is admissible that is definitely underestimating the total cost to any of the goal nodes. So, this is often a good heuristic I mean a good choice for each. So, anyway so we have why are we discussing all the these things in the class of AI for economics because these heuristic search it the need for heuristic search it often arises in economics.

That is when we are navigating a sea of different possible decisions, but we cannot do not have knowledge of the consequences of all the decisions. When we are taking a decision we know that what next decisions I can take and then I can calculate the costs of taking those decisions. Like for example, during covid we like I at in like the we often

have to depend on guess work depending on the situation right now I can have to decide whether to impose a lock down or just let us see things carry on normally and in for both there is a cost, but I cannot predict beforehand that if I keep on imposing lockdown then what will be like the result after let us say 4 months or 5 months that I cannot say beforehand. I have to take a short steps at a time let us say the next few days. Similar situations also arise in case of supply chain optimization where the companies may need to optimize their supply chains to minimize the total cost or maximize the efficiency.

So, they have to go for these kinds of graph search algorithms and they often do not have the access to the entire graph due to various uncertainties. So, they have to use this kind of heuristic approach. Similarly, for in case of portfolio optimization also when I that is when I am I have a set of assets and I am trying to like invest some of those assets hoping to maximize the gain. And now I do not know like I do not why also I may want to minimize the risk, but I do not know which assets to access like I may in fact it like right now I may be investing on some assets.

later I may invest some other assets. So, the my decisions are changing over time. So, it is a sequential case and I like at a time I can only see what will happen in the immediate future not what will happen after a sequence of case. So, I do not have access to the full graph I am taking one step at a time. So, that is where I need the this concept of like heuristic based graph search. So, with that we have come to the end of this lecture in the which is lecture 8 in the coming lecture also we will discuss some concepts which are related to this type of heuristic search. So, we will see till then if all of you please take care and stay well see you soon bye.