Artificial Intelligence: Search Methods for Problem Solving

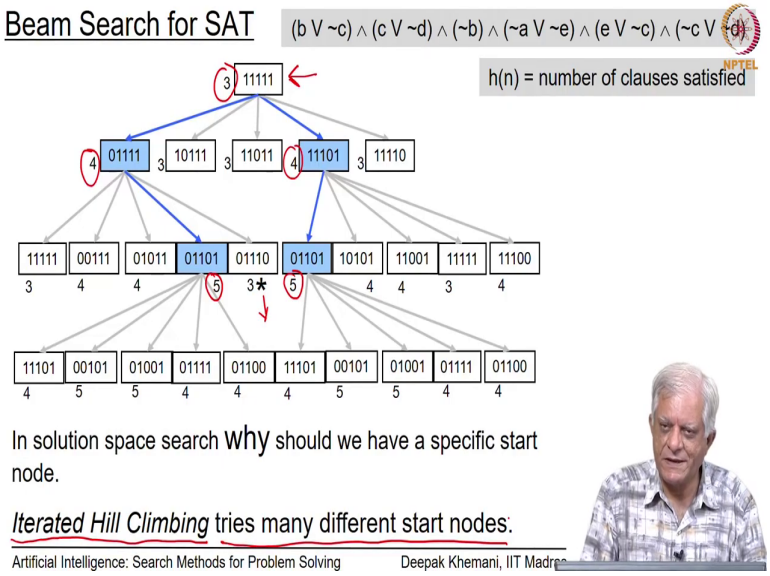
Prof. Deepak Khemani
Department of Computer Science & Engineering
Indian Institute of Technology, Madras

Chapter - 04
A First Course in Artificial Intelligence
Lecture – 28
Stochastic Local Search: Randomized Algorithms

So let us continue on our journey of Algorithms which help us move of local optima, local maxima or local minima entire and head towards a global optima. So, so far we have seen wholes of algorithms which were essentially deterministic in nature which allowed us to move of local maxima in particular we saw this algorithm called tabu search which allowed us to do that.

Now, let us move on to the Stochastic Algorithms or Randomized Algorithms which make moves which are probabilistic in nature which means that they may or may not make a particular move essentially.

(Refer Slide Time: 01:02)



So, let us look at some variations here algorithms like hill climbing or beam search for that matter. So, we saw this example of beam search where beam is first too and we were trying to solve this sat problem and we saw that we started off with a value of 3 and then we move to two neighbours whose value was better which was 4 and from there we move to two neighbours whose value was 5.

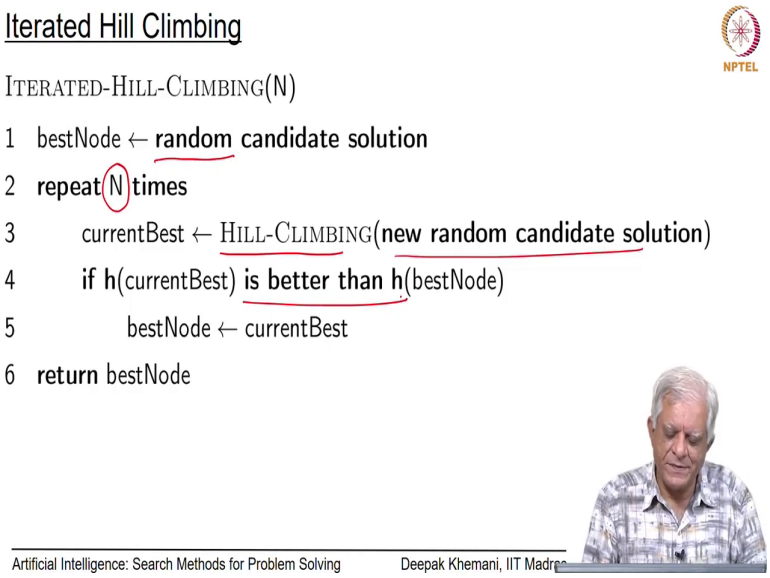
So, as you can see as you go down the value becomes better and better, but for these two nodes with value 5 all the neighbours had value 5 or less and so, therefore, the algorithm got stuck at there. And we had made the observation at that point that this particular node could actually have led to the goal node, but because of the heuristic nature of our algorithm this was ignored and the algorithm got stuck on the local maxima.

So, one question that one might ask is that if you are working in the solution space what is the significance or what is the import of having a specific start state essentially, why should search always be given a particular start space? In this case the start space is those 5 bits where each of the bits is 1 essentially I mean that that has no significance.

In state space search of course, we have this notion of a start space and a goal state and we wanted to find a path from the goal start state to the goal state, but in solution space search we and in particular in configuration problems like SAT always (Refer Time: 02:44) is in a solution state or a solution node and if you are searching in the solution space, then there is no importance of being in a particular start state.

So, we next look at an algorithm called Iterated hill climbing and what these starts is that it tries many different start nodes and if one of them works then your home essentially.

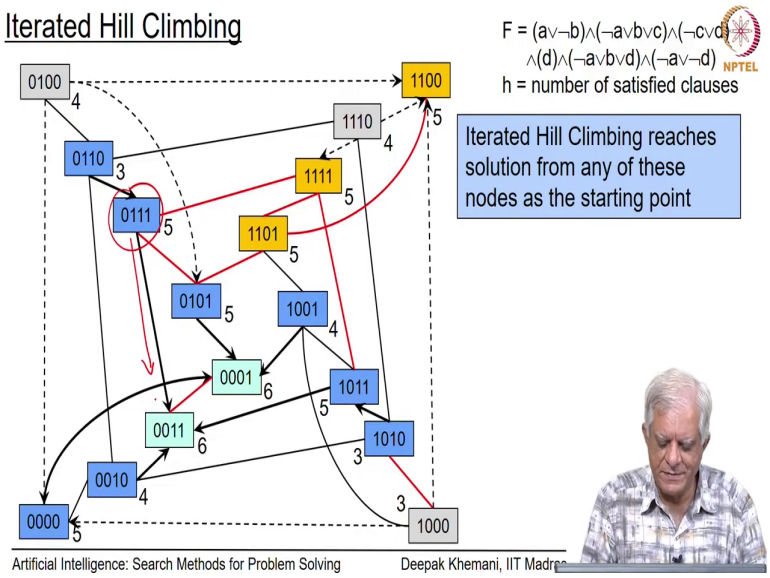
(Refer Slide Time: 03:11)



So, the algorithm is fairly state forward you are you choose some parameter N which is a number of tries that you are going to make and you first choose some random candidate solution and you do hill climbing with the random candidate solution and you pick the best node that hill climbing gives you essentially.

Now, you are repeating this whole process N times; N times you are starting with some random starting point and keeping the best node. If at any one of those N iterations you get a better node then you move to the better node. So, it is basically like doing N hill climbing algorithms in parallel or N runs of hill climbing algorithms in parallel starting from some N different random points.

(Refer Slide Time: 04:21)

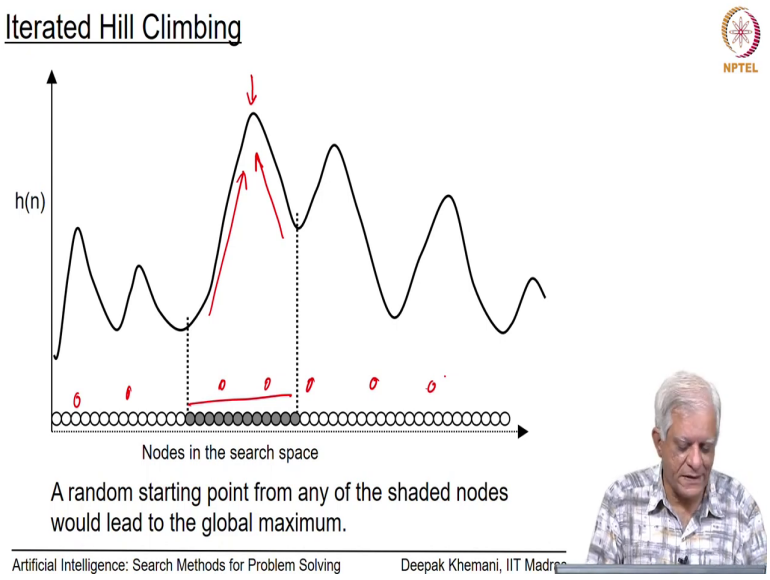


And let us look at the small SAT problem that we had seen earlier what we have showing here. So, remember that the average nodes for the local maxima that were existing in this particulars SAT space we had this formula with 6 clauses and they were two global maxima whose value was 6. And we had this edges which were for example, this one which said that if you were at this node then you would move along there is only one choice that had the hill climbing had and that choice would have taken into the global maxima.

So, if you look at all these nodes which are shaded in blue here, then any one of them would have if you are started from any one of them you would have ended up in a global maxima. So, if you had an iterated hill climbing and starting from any one of these 16 nodes let us say 4 times or 5 times and there is a good probability that you would heteroph one of those blue

nodes to start with and then you would find the solution. So, we can see that this is the reasonably good algorithm which can help you find the global maxima.

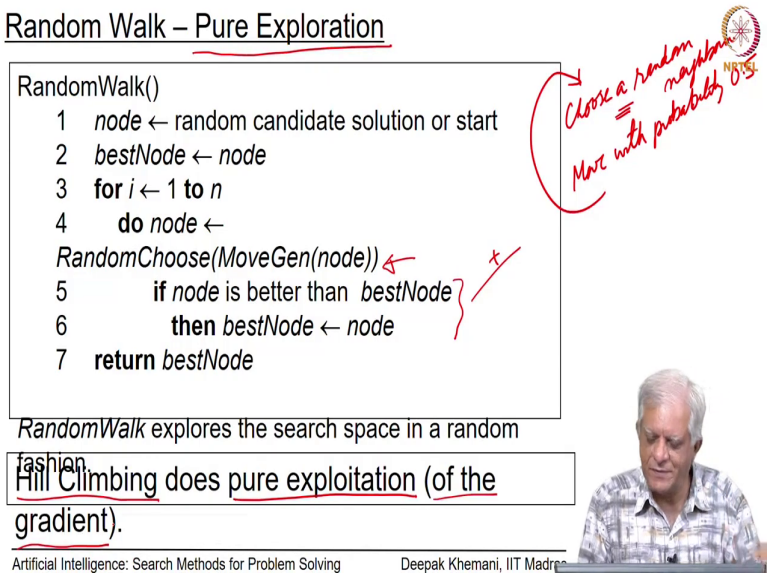
(Refer Slide Time: 05:35)



If you were to try to visualize what is happening in hill climbing is that if you have a heuristic surface which is plotted in some search space, then there are this shaded starting points from where the path through the global maxima which is this one is a monotonic path essentially so, either this way or this way.

So, if we were to start with any one of these nodes then you would end up in a global maxima and what iterated hill climbing is saying is that you start with some random choices in different parts and so, one of these or more than one of these would take you to the goal node essentially. So, it will; obviously, work in many cases where the surface has a reasonably continuous shape essentially.

(Refer Slide Time: 06:25)



Now, let us look at the other extreme of search essentially, the algorithm that we have seen so far they follow the heuristic function. They look at the neighbours they compare the difference in the heuristic value which gives you the gradient and then they choose the node with the maximum gradient. So, that is why they were also call as steepest gradient approaches essentially.

At the very other extreme is this process of pure exploration just this algorithm which is a random walk algorithm simply ignores the heuristic value, it does not look at whether you are moving to a better node or not to a better node and it simply says that move to a random neighbour essentially that is done in this line here it says that from your neighbours move to a random one.

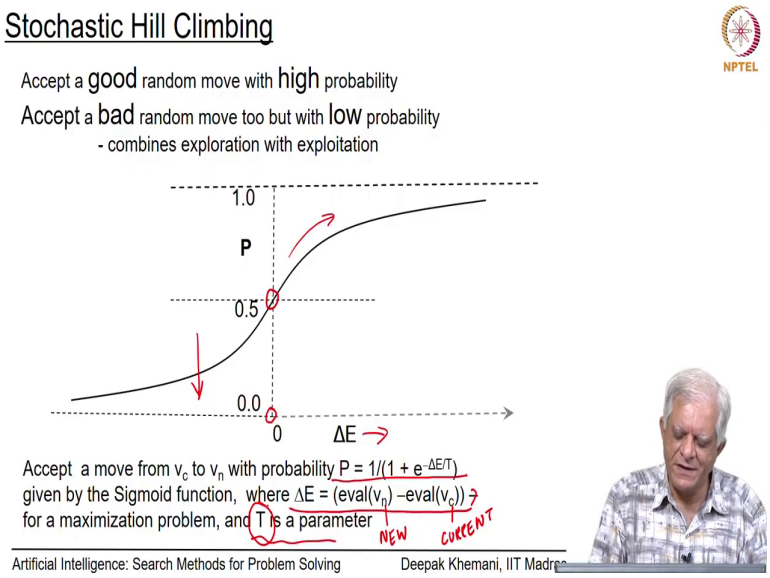
In fact, the ways that we implement this algorithm is often a little bit simpler, it simply says may choose a random neighbour and move to it with the probability half. So, if I write this algorithm here choose a random neighbour and move with probability 0.5 or half essentially. And you just keep doing in a cycle and of course, you keep this part which is that if you have seen a better node remember it essentially.

So, one difference between this and hill - climbing is or this version that we have written on the left is that you do not generate all the neighbours you just make one random choice and either you have move to it or you do not move to it with the probability half essentially. And this algorithm is called random walk algorithm it is pure exploration it just ignores the heuristic value totally and explores the space in a very free fashion.

This is as opposed to hill climbing is a little bit of overlap here, but let us hope we can read through this. So, hill climbing does pure exploitation, what do you mean by exploitation. Exploitation of the gradient essentially, hill climbing simply exploits a gradient and if the gradient says stop it stops otherwise it you know follows the steepest gradient approach.

And we want to look at methods which are some more in between exploration and exploitation, exploitation is good because it takes you towards the optimum as a case may be maximum or minimum. Exploration is good because it takes in your areas in the search space essentially and if you had a judicious mix of exploration and exploitation maybe our algorithms will perform better.

(Refer Slide Time: 09:44)



So, we look at in a algorithm call stochastic hill - climbing and what it says is you make a choose a random neighbour and if it is good in terms of the gradient or in terms of the difference in heuristic values or evaluation values, accept a good neighbour with a high probability, but at the same time accept a bad neighbour or bad random move, but with a lower probability.

So, in general you look at a random neighbour if it is better you accepted with a higher probability and if it is worse you still accept it, but with a lower probability. So, the tendency of this algorithm would be to move towards better neighbours because you know the chance of moving to a better neighbour would be higher, but it would also allow you to move to a worse neighbour because that is what you need if you want to get of a local maxima and head towards the global one essentially.

So, you can see that this algorithm combines both exploration and exploitation, exploitation because the probability of moving to better moves is higher and exploration because it allows you to even move to worse neighbours. How do we define this or how do we what is the probability values that we choose.

So, if we were to choose a function which is like this which is many of you will recognize as a sigmoid function. Then the x axis shows the difference delta E between the values of the current the new node and the current node. So, new node minus the current node and the y axis shows the probability of accepting that move essentially.

So, you can see that at the place where the delta E is equal to 0 we have chosen the probability to be 0.5 which is like what we would do in a random walk essentially, but as delta E increases the probability goes higher than 5 and tends to one as delta E becomes larger and larger.

As delta E decreases it becomes less than 0 the probability comes down and it becomes lesser than half, but it is still a positive value which means that you could still make a bad move essentially. So, this stochastic hill climbing algorithm makes both good moves and bad moves, but it makes good moves with high probability and bad moves with low probability. So, it has a tendency towards making good moves.

So, the sigmoid function is basically given by this expression 1 over 1 plus e raise to minus delta E divided by T and where T is a parameter that we will discuss shortly essentially that that allows you to control the shape of this curve as we will see essentially. This definition of delta E where n is a new node and c is a current node is for a maximization problem, for a minimization problem it would be the negative of this essentially ok.

So, what we want to do is to how to essentially a now our choice boils down to saying that you want to do this stochastic hill climbing, but you want to choose this parameter T in an

appropriate fashion because that as we will see shortly determines the shape of the sigmoid function.

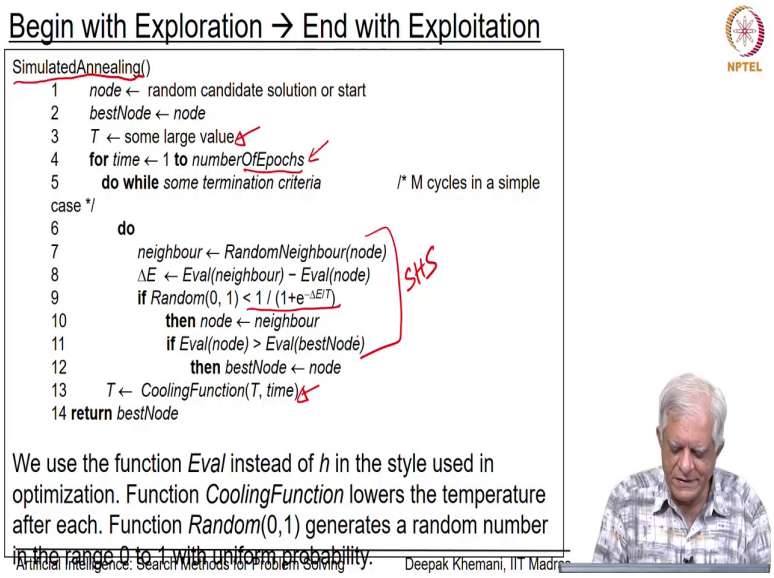
Now, the algorithm that we are going to look at gets inspiration from material science and in material science very often people are involved or concerned with producing materials which have some good properties and very often this good properties are associated with systematic and smooth or nice, structured arrangement of atoms and typically such nice arrangement of atoms are associated with minimum energy states essentially.

So, essentially they would like to so, the process is as follows that you would whatever is a material that you are working with you take it in a molten state or a liquid state and then you let it solidify essentially, but you want to make it solidify in a very controlled fashion. So, that the cooling of that material is done in a very controlled fashion. So, that essentially the atoms get time to settle into low energy state essentially and this process in material science is called as annealing essentially.

So, inspired by this way of driving materials towards a low energy state which you can see is a kind of an optimization problem that you want the material to have the lowest possible energy is this algorithm that we will study next which is called simulated annealing essentially. So, it follows that process of annealing and we will call this parameter T as temperature.

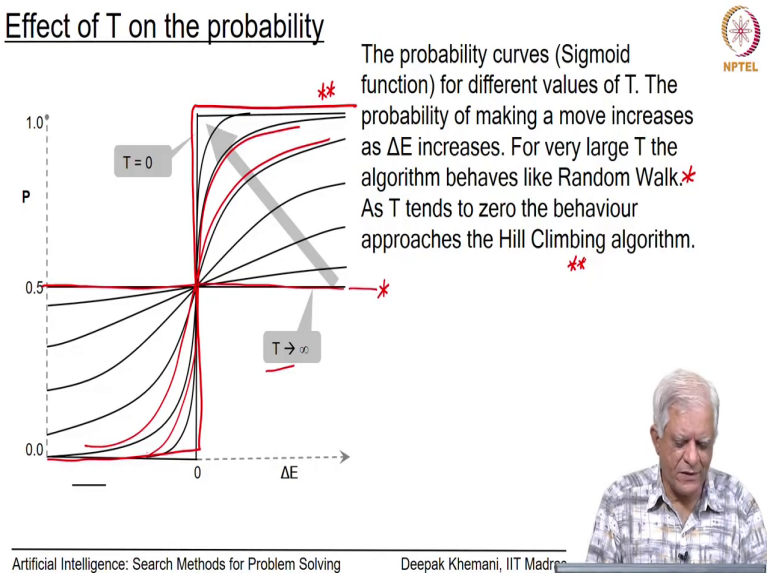
So, it is like bringing down the temperature in the environment gradually and so that this method settles down into the material settles down into a low energy state essentially. What is the impact of temperature on our algorithm of stochastic hill climbing that we will see shortly.

(Refer Slide Time: 16:04)



So, first let us look at the algorithm ah, initially we choose temperature to be a large value and we will see shortly that when the temperature is high the tendency of the algorithm is to be more explorative in nature. In fact, we can make that observation first and then we will come back to this algorithm.

(Refer Slide Time: 16:38)



And this is the impact of the temperature parameter T on the shape of the sigmoid function here.

So, you can see there are different curves lone here and the arrow depicts that what happens as you reduce the temperature, when temperature is very high which we have denoted by saying that it tends to infinity then the sigmoid is curve is simply horizontal line at the value of 0.5 which as you know is associated with the random walk. So, one version of random walk (Refer Time: 17:27) set that make a random move or choose a random neighbour and move to it with a probability 5 essentially.

Now, as we decrease this parameter T which we have also going to call temperature the shape of the sigmoid curve in a moves as shown in this arrow here. So, initially it becomes like this

and then it becomes more and more steep and when temperature is equal to 0 it is simply a step function.

What is the step function mean? It simply says that if delta is positive accept the move with a value probability 1 if delta is negative accept the move with probability 0 essentially and you can see that this step function is for would happen in hill climbing if you are generating only one neighbour, you generate a neighbour if it is better then move to that if it is worse do not move to that essentially.

So, there are these two extreme contrast in behaviour all control by the single parameter called T when temperature is very high the nature of the search is explorative in nature that it slightly to move randomly around in the space, as we decrease T this curve start shifting and becomes more and more deterministic and eventually when T becomes slow it becomes like hill climbing.

So, as you can see that the motivation behind this algorithm is to allow a lot of random movement in the initial part of the search this thing. So, that the algorithm does not get stuck in a local optima and then gradually bring the temperature down. So, that hopefully it will move to a region which is leading to the global optima and eventually in a hill - climbing like fashion it will move to the global optima.

Now; obviously, there are no guaranties here stochastic algorithms ah like this can never say that you will definitely reach to global optima, but we have also seen that in problems. For example, in the TSP problem or the traveling salesman problem which is an optimization problem in the sense that you have to find the tour with the lowest cost the search space is often. So, vast that you cannot search the whole space exhaustively which is what you would need if you want to give a guarantee that you are going to reach the optimal state.

So, let us go back to the algorithm that we were has started looking at, this algorithm as I said is called simulated annealing. So, it kind of mimics the process of annealing in materials, but in the case of our stochastic hill climbing algorithm it is and remember this stochastic hill climbing says that move to a better node with a high probability move to a worse node with a

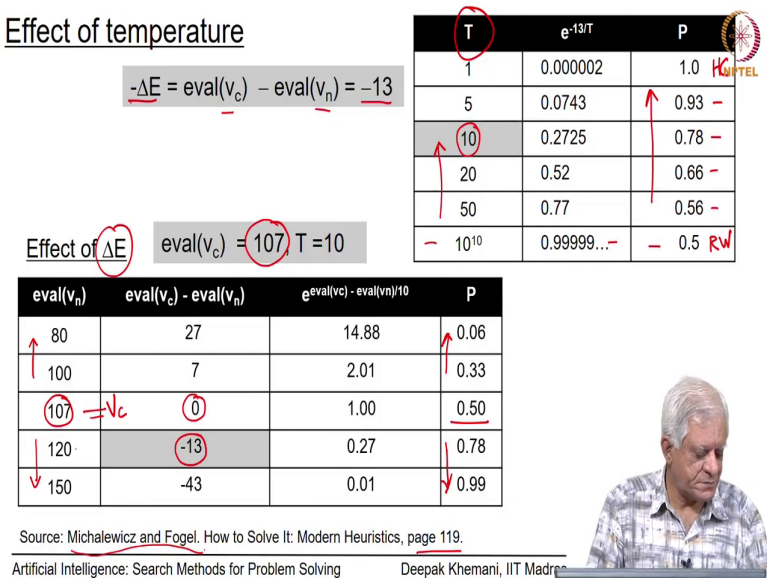
lower probability and the probability as you remember is given by the sigmoid function. So, this algorithm basically spells out some of the retails essentially or how to do that.

Now, typically you run this algorithm for a certain number of epochs and decide that this is how much computing power you have and very often this is decided experimentally and you keep doing this. In each epoch you reduce the temperature by a certain amount and then run it run the stochastic hill climbing algorithm for a certain period of time essentially.

So, you can see that this part is stochastic hill climbing where the temperature has been fixed and in the outer cycle temperature is gradually reduced which is like the process of annealing essentially. And so, essentially this is a this is a algorithm that you that you gradually decrease the temperature and keep doing stochastic hill climbing and the hope is that you will eventually reach the global optima.

So, let us look at this is the little bit and try to analyze this a little bit. So, this is again repeating what we just discussed that the behaviour of the algorithm is controlled by the parameter T, when T is high the algorithm is more like random walk or more explorative in nature and as gradually as T is lowered and becomes less and less the algorithm becomes more and more exploitative of the heuristic function remember and becomes more and more deterministic essentially, in the end at very low values it is it is just like hill - climbing essentially.

(Refer Slide Time: 22:45)



So, let us look at some effect of this parameter. So, in this example which I have taken from this book by Michalewicz and Fogel from page 19 it is a very nice illustrative example. We assume that delta so and this is the maximization problem and we assume that delta E is 13 which means, it is a good node the difference between V n and V c is 13. So, the negative of the difference which is what we are interested in not minus delta E is minus 13 essentially, but remember that this signifies that the new node V n is better than the current node which is V c essentially.

First we want to see, what is the impact of temperature. So, as we have already observed when the temperature is very high in this case 10 raise to 10 the value of this expression e raise to minus 13 divided by T tends to 1 and correspondingly 1 over 1 plus 1; 1 over 1 plus 1 would be half and so at very high temperatures you can see that it is like a random walk. That

if you move to this new node with only a probability 50 percent even though we can see that the new node is better than the current node.

As temperature decreases you can see that this values for e raise to minus 13 by T they also becomes smaller and smaller and smaller which means the denominator becomes smaller and smaller and smaller which means the probability increases as you can see here and at a very lower temperature of 1 the probability is close to 1 essentially.

So, now we can see that this is like hill climbing and at very higher temperature it was like random walk essentially. So, the effect of temperature is as we have discussed to gradually increase the probability, as we as we decrease the temperature the probability initially was 0.5 which means it was just in not looking at delta E at all. But now since this delta E is a good one as temperature decrease it became 0.56, and then it became 0.66, then it became 0.78, then 0.93 and so on then eventually it will settle in to the it will the probability of moving to the new node will tend to 1 as towards the later part of the algorithm.

Now, let us look at the impact of delta E on this algorithm. So, we will assume for the next table to that the value of T is frozen it 10 and we want to see what is the impact of delta E. So, the starting the current node has a value of 107 and we want to look at different possible values of the new node essentially. In the last table we saw a value of minus 13 which was a good value, but we will look at different value. So, 120 is about corresponded to the previous table and we look at values ranging from 80 to 150.

Remember that that the value of 107 is equal to value of V c. So, delta E is 0 here essentially and as one would expect the probability tends to 0.5 essentially, but as the node becomes better which means it has a higher and higher value delta E becomes higher and higher negation of delta E becomes lower and lower and the probability becomes higher and higher. So, as the neighbour becomes better the chances of moving to that neighbour increase essentially.

On the other hand if the new node becomes worse and worse so, 100 is 7 less than 107 and it is 27 less than 107. So, as a neighbour becomes worse delta E becomes negative negation of

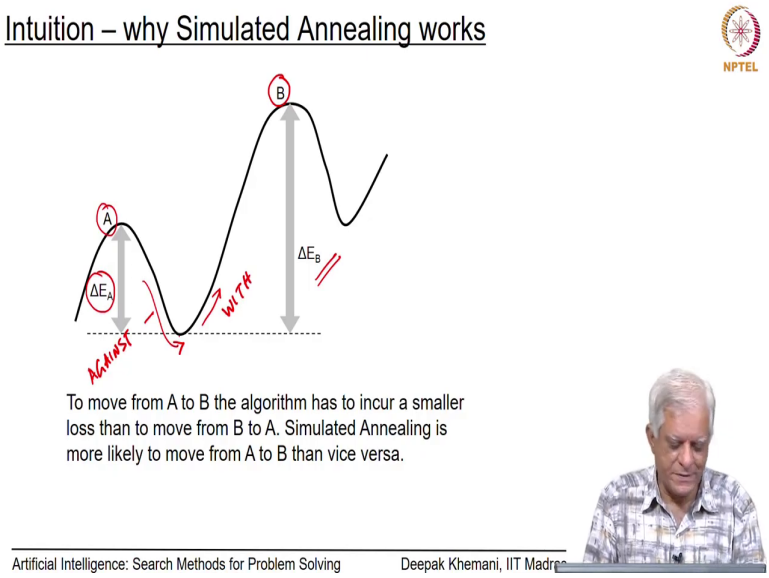
delta E becomes positive and the probability starts going lower and lower. So, the worse the worser the node is the lower is the probability of moving to that node, the better the node is the higher is the probability of moving to that node.

So, the second table has shown us the impact of delta E on probability as a node becomes better the probability increases. The first table showed as the impact of T on the probability that and we saw this table for a better node where the new node had a value of 120 and delta E was 13 and we saw that as temperature was reduced the probability became close to 1, as a small exercise one can encourage you to look at a worse node. So, for example, the node with value 100 or value with 80 and do this table for yourself and you would see that as temperature goes down then the probability will decrease from 0.5 to gradually 10 toward 0 essentially.

So, there are these two things which influence whether you move to a random neighbour or not one is temperature, if the temperature is high you move to that random neighbour whether it is it is good or bad it does not matter, but if the temperature is low then you move to a random neighbour if it is better.

For a given temperature if the neighbour the better the neighbour is the more is the probability of moving to that neighbour and the worse the neighbour is the less is a probability of moving to that neighbour, but the probability is still greater than 0 which means that the algorithm will still make moves which are against the heuristic function and in that process it still has a chance of moving away from local optima.

(Refer Slide Time: 29:47)



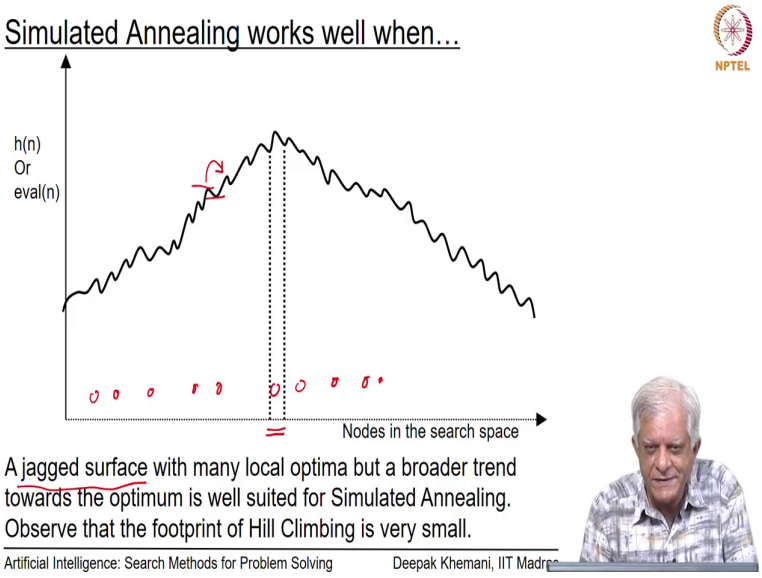
So, we can also try to understand why simulated annealing works and in fact, it works and empirically it has been found to work extremely well and it is a very popular optimization based method. And we can think of it like this that if you are at a local maxima which you will call A and if you want to move to another better maxima which you will call B, then you have to do a certain amount of work against the heuristic function and that amount of work is enough captured by the difference between the minima that you have to somehow cross.

But remember that if you could somehow cross this point then the chances of moving up will become better and better because now that. So, here you are going against the heuristic function and here you are going with the heuristic function or you are exploiting the heuristic or you are exploiting the gradient.

So, when you move from A to B then you are the amount of work you are doing against the heuristic function is much lesser than if it was in the other direction. If you have to move from B to A then the amount of work you would do is delta E B which you can see is much higher.

So, since the stochastic hill climbing algorithm at any given temperature is has a greater tendency of going with the heuristic function you can see that what you want is that in the initial phase if you keep the temperature very high then you allow movement in either direction, but gradually you start reducing the temperature and you hope that your algorithm we will get stuck in a on a slope when temperature comes down, so that it will eventually go to the global maxima.

(Refer Slide Time: 32:00)

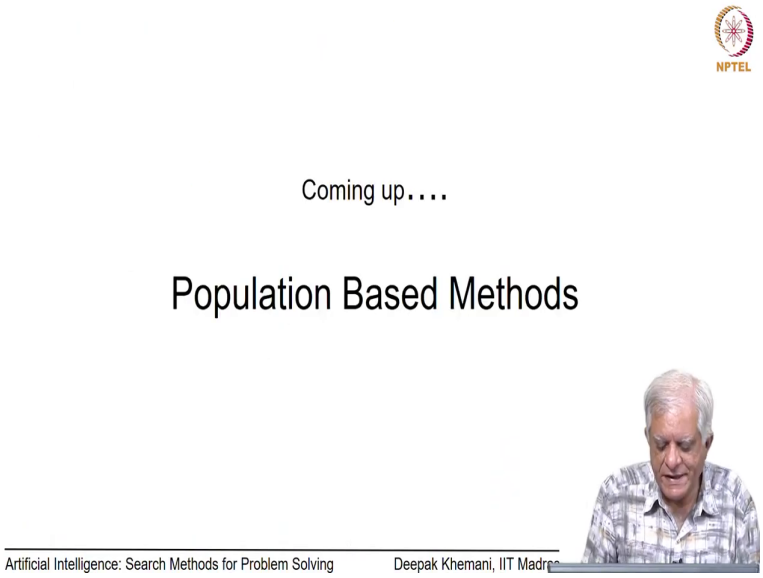


So, if we were to imagine when will it work well if you had a surface which is a little bit like this which is like, a jagged heuristic surface remember that the heuristic surface is what defines a terrain that the algorithm is performing over this.

So, if you are a jagged surface with you know very lower differences between conjugative local maxima or local maxima in this case, then the amount of work one has to do moving from one maxima to another is not too much essentially and hopefully the algorithm will be able to you know recommend this small differences and climb up to the global maxima. So, if the surface looks something like this then you can imagine simulated annealing would really work well essentially.

We can also see that the footprint of hill climbing and by footprint we mean the nodes from which hill climbing will take you to the global maxima is pretty small in this particular illustration. So, even if you are doing iterated hill climbing then you would have to strike at least ones within this gap for your algorithm to work essentially.

(Refer Slide Time: 33:13)



Next we will look at population based methods and by this we mean that we will work with a population of candidate solutions we will look at two algorithms, one is called genetic algorithms which kind of tries to take inspiration from the way nature does evolution and those algorithms are also called evolutionary algorithms.

And the second algorithm we will see is ant colony optimization which also some times is called as form optimization which also takes inspiration from nature has to how a colony of ants or a colony of bees for that matter happens to do things in a more efficient within a single ant could do essentially.

In away when we looked at iterated hill climbing that was also a population based method we could think of it is a population based method where you have a population of candidates and one of them takes into the solution. But if you are not exploiting the fact that there is a

population at work together you know how do they collaborate with each other, that is the key feature of both genetic algorithms and colonial optimization and we will see that in the coming few lectures ok. So, see you then.