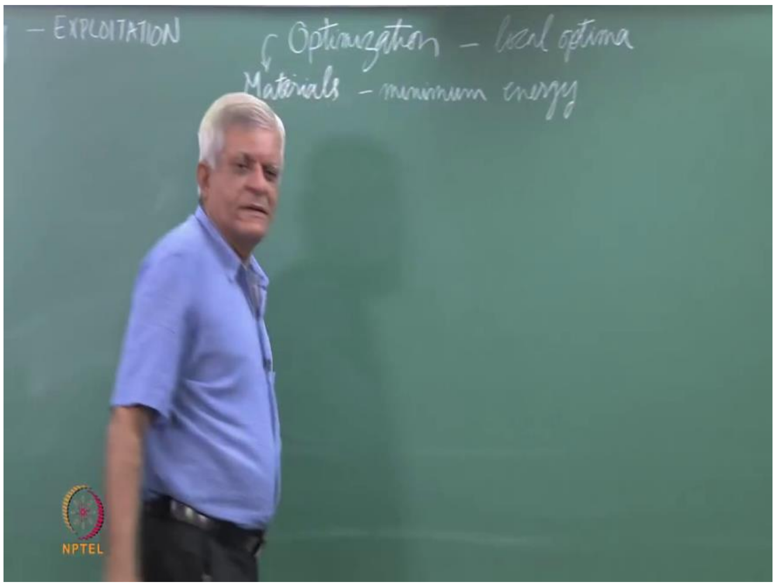
Artificial Intelligence

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

Lecture - 14
Optimization 1 (Simulated Annealing)

Let us get back to our local search. And the key words, that you must keep in your mind is Optimization.

(Refer Slide Time: 00:21)



Remember, that we converted the states space search problem to on optimization problem, by saying that we want the best value for the heuristic function. And essentially, we are developing these local search algorithms, to explore the surface define by the heuristic function, which we are now calling evaluation function. Because, that is for the optimizing community call set. And looking for minimum and maximum value, as a case maybe…

And because we are exploring local search algorithms, which means that, the algorithm will not necessarily search the entire space. But, we will terminate, using some criteria before that especially. We have the problem of local optimum. And essentially, we are exploring these different algorithms to try and get around this problem of local optima. Because, the simplest algorithm that we saw hill climbing is very efficient, it requires very little space.

In fact, a constant space or being search for that matter, which also requires constant space. And it will just go up the slop and stop. It will not take an exponential amount of time. But, the travel is that. That is not, what we want? We want an global optima, which will not be the maxima, that the hill climbing will find. So, we have looking at ways of getting around or getting beyond those local optima's. So, therefore one method that we saw in the last class was ((Refer Time: 01:55)) essentially.

And we have said that, hill climbing we will associate with exploitation. And by this, we mean exploitation of the heuristic function or exploitation of the local knowledge or the gradient, that you can measure the gradient around you. And you will always follow the steepest gradient path. And this, we will term as the exploitation of the gradient essentially. Now, optimization does not happen only in the computational world, essentially I mean.

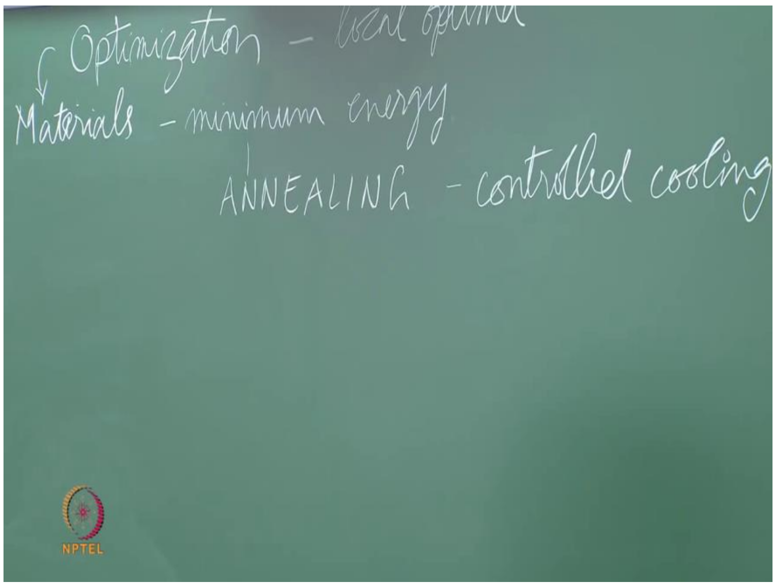
There is optimization happening in the physical world as well and in nature as well. We will see some examples from nature and from physical world. So, in physical world for example, when we talk of materials, you want to produce materials of certain quality. And very often, this quality is dictated by the way, it is atoms are arranged. So, for example, if atoms are arranged in the nice array like structure, then you get good crystalline materials or if atoms are arranged to the close to each other in a metal, then or in a ceramic. Then, you get material, which has good properties like good strength of material and things like that. Now, the way to arrive at materials with good properties, very often involves creating a liquid version of that material and then solidifying, it is essentially. So, casting for example, you make a mould. And then you melt the material from which we want to make the cast.

And pole the material in to the mould. And then it solidified and takes the shape of the cast essentially. This is how, many of the ((Refer Time: 04:17)) statue that you see, are made essentially. And there are many other things, which are made by casting. Now, as I said, the properties of these materials is dictated by the way, that these atoms are arranged in the… And it is desirable to arrange the atoms in a systematic fashion, which in other words corresponds to low energy levels, essentially.

So, you want the final form to have low energy. And in that sense, you have a minimization problem. So, materials you want to be in minimum energy state. So, you can imagine that, if some material is in gaseous form, it is got some maximum amount of random movements happening and more energy. And as it becomes liquid, the energy levels come down. And as it become solid, the energy levels come down.

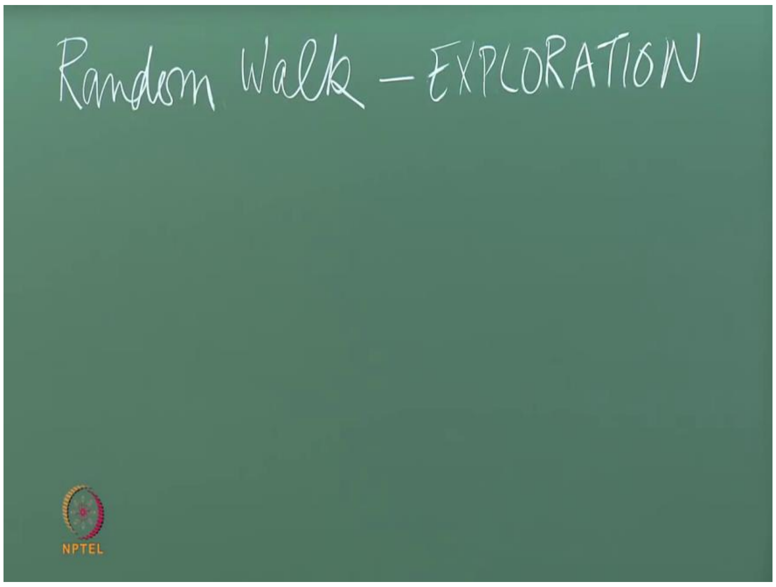
But, even in solids if we can arrange the matter in to minimum energy levels, then we get good properties of materials.

(Refer Slide Time: 05:43)


Optimization - local optima
Materials - minimum energy
ANNEALING - controlled cooling

And typically, the process that is used for minimum energy is call, annealing. And by annealing, we basically mean controlled cooling. That you do not allow to cool the material, at it is own space or in some sense. But, you make it cool slowly at a control rate. So, that the atoms settle down in to the minimum energy state. So, annealing is a physical process of minimization, which is a kind of an optimization problem, that we are also talking about. And we will take some inspiration from annealing, in a few moments.

(Refer Slide Time: 06:24)



Random Walk – EXPLORATION

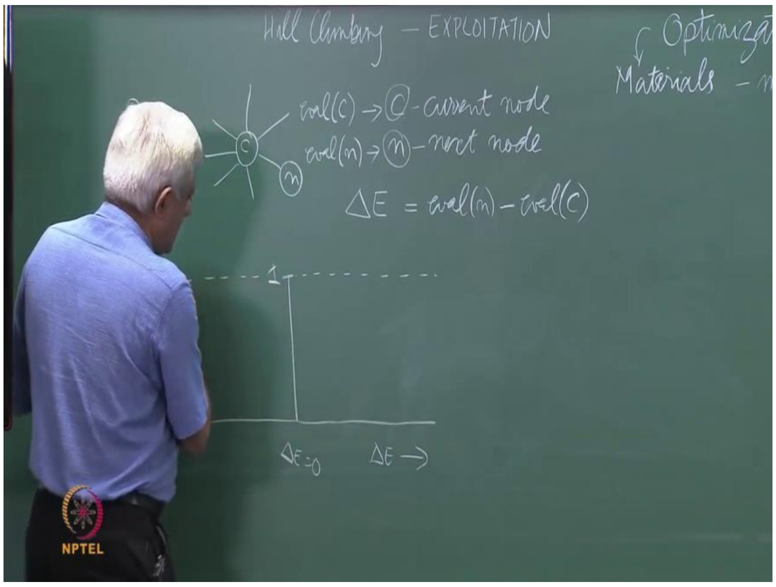
You know some ethic called as random walk, which we associate with exploration. So, random walk as a name suggest is that, you take a step in any random direction or make a move to any random neighbor essentially. And there is no other no criteria, which will tell you which neighbor to move to essentially. Now, hill climbing you generate the neighbors of a given candidate. Inspect the evaluation heuristic values or evaluation function values, objective function values.

And choose a best amongst them and move to that essentially. Random walk is simply just taking a step in any random direction essentially. Now, obviously you can see that, random work has it is property of exploring the states space or the search space, whichever the search space. It could be the solution space. But, of course it will not be guided. It will not have any desire to reach our state, still at maximize essentially. Just keep going one and one.

So, when deterministic way that we saw of going beyond maxima in hill climbing, we saw was Tabu search. But, today we want to look at a non deterministic or the stochastic or randomized approach to exploring the space. And our goal is to have a judicious mix of exploration and exploitation. Exploration, because it is exploration which will stop us from getting stock at local optima, and exploitation, because it is exploitation which takes us to optima in the first place essentially.

So, it is just at what we want to reach the global optimize, as far as possible. So, we do not want to have pure exploitation. Because, we could have started in the wrong place and you would end up in a local optimize. So, as I mentioned in the end of the last class, we would want to make a move with a certain probability, which is control in such a fashion. That if the move is a good move, then the probability is high. And if the move is not a good move, then the probability is low.

(Refer Slide Time: 08:56)



So, we will use the notation c as current, as we did in the last class, current node and n as a next node. In hill climbing, n is the best amongst the neighbors. In random walk, n is any random neighbor of c. So, there is a relation between c. So, this is c and then there are many possible neighbors. And one of them is n. Associated with this c and this n, we will have their heuristic values or evaluation values, which you can call as a eval c for this node c.

Therefore, node n we will have eval n. And if we have maximizing, then we want eval n to be greater than eval c. If we want minimizing, then we wanted to be smaller essentially. So, which means that hill climbing would have chosen the one with the maximum eval n and value around here. But, now we are given up this strategy of inspecting all of them, and then choosing the best amongst them.

We are inside adopting a strategy, in which you will just take of random neighbor and either move to it or not move to it, essentially. So, this is the choice, we will make that.

We will either move from c to n or we will remain at c, which means that at, we will generate another random neighbor at that point. And again, either move to that or not move to that. It is a little bit difference from, what we have been doing so far, essentially.

In random walk, we always move with 100 percent surety to the next neighbor. But, we do not want to do that. We want to move with a certain probability. And the probability should be such that, for good moves it should be high and good move. So, we will associate that term delta E as eval n minus eval c. And the equation, that I am going to write is going to be for maximization. And you can just flip the sign for minimization essentially.

So, this delta is something that, we want our probability to be influence by. So, should we make this move from c to n? It depends upon, what is the delta E is? If delta E is positive, then I would more likely want to make a move with the high probability. If delta E is negative, which means that n is worst than c. I would still want to allow that move to be made, because that is our goal to get of maximum essentially. We do not want to stuck at maximum.

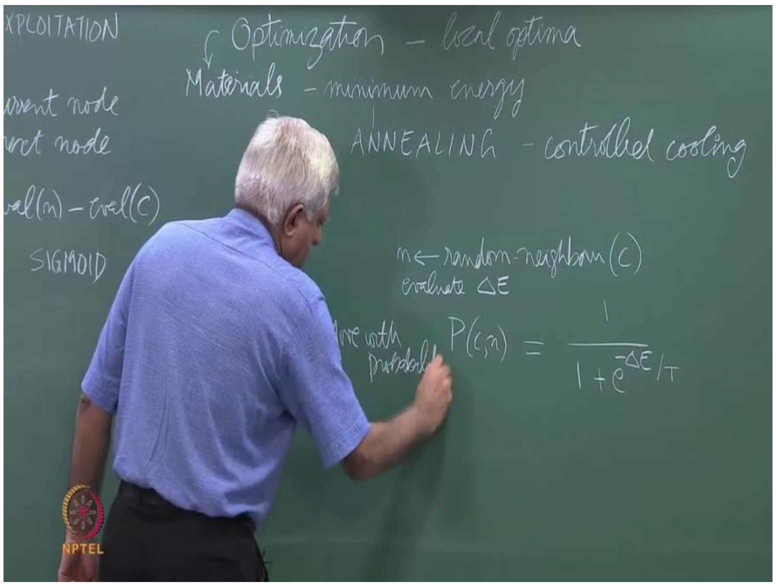
So, we will allow even bad moves, which means with negative delta E to be made but, with a lower probability essentially. So, the function that we will use for computing this probability will be a function of delta E. And one more parameter, which will be you will used to control, how this delta E is influences the probabilities essentially? So, we want two things. The delta E should influence the probability. And secondly, I want to control how delta E influences probably?

Did any one give a thought to this function? Or you might have write about it, some way. So, the function that I will use… So, what is the nature of this function? We want the function of delta E. The function should be range should be limited to one. It should be between 0 and 1, essentially. Because, it is a probability, we want to measure. We want to get a probability measure. It is domain should be infinite, practically infinite.

Because, I am not going to put any constraint on, how much is the difference between the evaluation function of two nodes? It could be a, to any degree essentially. So, we want a function, which will stay within this. So, let me draw that function here. I mean, let me draw the domain at least. So, this is what my destiny is... So, this is going to be 1, this is going to be 0. And here, let us say delta E is equal to 0. And here, delta E is increasing. And here, in the opposite direction it is decreasing essentially.

Delta E is equal to 0 means, the next node is as the same value as the current node essentially. So, obviously you can imagine that, you do not want to, you do not care. You can move to it or you do not move to it. You are getting the same evaluation function essentially. So, we want the function which is going to be a monotonically increasing, as you go from left to right but, stay within these nodes. So, you can imagine. They could be. There is more than one function, that you can choose to do that.

(Refer Slide Time: 14:21)


XPLOITATION
Optimization – local optima
Materials – minimum energy
ANNEALING – controlled cooling
urrent node
ext node
val(n) – eval(c)
SIGMOID
n ← random-neighbour(c)
evaluate ΔE
ove with probability
$P(c,n) = \frac{1}{1+e^{-\Delta E/T}}$
NPTEL

We will choose one. And that function is called the sigmoid function. So, we will write this as the following that P. It is the probability of making the move. So, let me just say P c n. If we interpreted, as saying that P c n is a probability of making a move from c to n is 1 over 1 plus. This is basically the sigma at function. And like as I said, we will have a second parameter, which will allow us to control, how much delta E influences the probability?

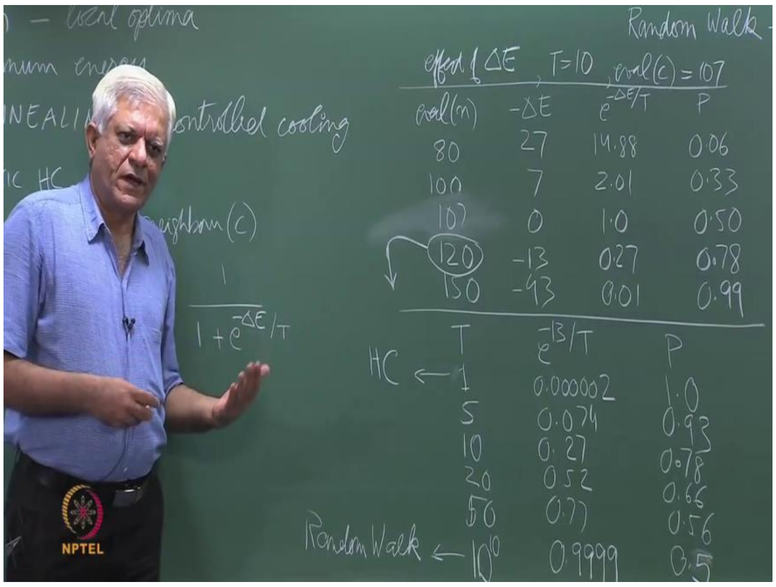
Because, we should be able to work at any in some sense, place in this spectrum. And the spectrum starts with hill climbing, which means that the moment delta E is positive. You will accept it, with probability one. The moment that is negative. You will reject it, with probability one or accept it with probability zero. And on the other hand is a random work at the other extreme, which means that it does not care, whether delta E is positive or negative.

You will always accept it with probability 0.5. But, we want to be able to operate, anywhere in this range essentially. So, we need a second parameter. And the second parameter, traditionally is call T essentially. So, this is a simple algorithm, generate. So, n is a random neighbor of c. So, let us assume. You have a function called random neighbor of c. And then we say eval delta E.

According to this formula, eval n minus eval c and move with probability. So, it is a little bit like the random work. Accepts, if I may use the analogy of somebody being inebriated. This person is little bit too inebriated. That sometimes, he does not even make a move ((Refer Time: 17:14)). So, sometimes he makes a move, sometimes does not make a move. So, sometimes he goes some c to n and sometimes, he just stays there and after while, he makes another moves and some other moves essentially.

This algorithm is call stochastic hill climbing. So, I will just write HC for hill climbing. It is not purely hill climbing. Hill climbing would only go in the direction of the steepest radiant. But, it has a tendency to go in the direction of better values. So, we have still talking about maximization. So, it still has a tendency to go up. So, let us inspect some values, which I have got from this book. So, how to solve it by ((Refer Time: 18:01))? So, there given some examples. So, I am just taking their example.

(Refer Slide Time: 18:15)



effect of ΔE, $T=10$, eval(c) = 107

eval(n)	$-\Delta E$	$e^{-\Delta E/T}$	P
80	27	14.88	0.06
100	7	2.01	0.33
107	0	1.0	0.50
120	-13	0.27	0.78
150	-43	0.01	0.99

	T	$e^{-13/T}$	P
HC ←	1	0.000002	1.0
	5	0.074	0.93
	10	0.27	0.78
	20	0.52	0.66
	50	0.77	0.56
Random Walk ←	10^{10}	0.9999	0.5

So, what is the effect of delta E. Let us assume that, T equal to 10, some value. And we are using that, value 10 here to compute this delta E essentially. And let us assume that, eval c is equal to 107, some value. And values better than 107 are good, values less than 107 are bad for this essentially. So, let construct a small table of values and say, how it effects? So, this is a value for eval n. So, if this is 80. That means, it is a not a good value.

It is less than 107. So, delta E so I write minus delta E here. Minus delta E is 27. Then, e raise to minus delta E by T, e's value is 14.88. And this is the probability. This is 0.06. So, if according to this formula, and this function is called sigmoid function. If my eval n equal to 80, then I will move to it to the very small probability, as you can see 0.06 essentially. So, 6 times out of 100, I will move to that node. Otherwise, I will not move to that.

And that is, what we want essentially. You want the search to focus more towards better moves and less towards, you see. Let us look at some more examples. If this is 100, this is little bit better. This becomes 7, this becomes 2.01 and this becomes 0.33. So, we are studying the effect of, how this function sigmoid function response to changing delta E essentially? So, this is still a bad value because, we are starting with eval c equal to 107.

We are still going 7 points lower. And it will moving to that with one third probabilities, essentially. Then, just for the sake of completion, you take this. This happens to be 1.0 and 0.50, which is nice. Because, what this function is telling us? Is that, if eval n equal to eval c you may or may not move to that, which means you move to that probability 0.5. And you can stay back at with probability 0.5, which is what we would expect?

Now, let us took at better values 120, which becomes minus 13, 0.27 and 0.78. So, if you find the better value, then we will move to with probability 0.78. How do you move it to the probability 0.78? We discuss at the last class. You generate ((Refer Time: 21:37)) number in the range 0 to 1. And if that number is less than 0.78, you move. Otherwise, you do not move essentially. Less, you know equal to 0.78. One more value 150, which is much better.

So, this table basically which I have taken from this book. Illustrates, how stochastic hill climbing response to different values of delta E. As you can see, if delta E is high and what you have written here is negation of delta E. If delta E is high, if we get a good improvement, it makes some move with greater probability. Otherwise, it makes a move with the lesser probability. All this is for, this value of T equal to 10 essentially.

How do you choose this value? It is a next question, because we will see in a moment. What is a influence of T essentially? I will, let us first finish that and then we will do the discussion. So, now let us assume that we have looking at this case. So, not this case. We are looking at this case, which is the better case. And let us see that if eval n equal to 120 and of course, eval c is 107, how does temperature affects the probability?

So, let us look at different values of T. Then, we have e is to minus 13 by T. And then we have probability P. So, let us take a low value and we will assume, 1 is a low value. This comes to and you can do this yourself actually. So, let me first write out this values. With 5, it is... So, this is how temperature of it is, this. So, I have used our term temperature. And in fact, that there is a connection with annealing, that we will shortly see.

So, we will call this parameter, temperature. And delta E is also stand for energy in some sense. Of course, in this case you want to maximize. Otherwise, you would have a formula in which, this goes to negation of… Instead of minus delta E, you would have plus delta E, if you wanted to minimize. So, how does this behave? Let us see. At very low temperature or very low value of T, the probability is 1. So, you can see that, this is like hill climbing.

If you keep this parameter low, then it tends to behave like hill climbing. Well, in the sense that if you see the better more, it will make it essentially. Then, as temperature increases and this is where the analogy with real world materials comes in to play. The energy levels increase. The entropy keeps increasing and so on so forth. Or randomness keeps in improving. And at very, very high temperatures, this is 10 is to 10.

The probability is 0.5. And you can see that, there is a bit like random walk. So, we can control the behavior of our probability function by controlling temperature. If you want it to be more random, we will keep high temperature. So, as you are above 0.5, it is approaching a random walk. It is more random essentially. Irrespective of, what value of delta E is? It will be 0.5 essentially. So, it will be randomly make a move essentially.

If you wanted to make it, so if you want to explore more, you keep the temperature high, making to behave like a random walk. If you wanted to follow the radiant, then keep the temperature low. Making it, behave like a little bit like hill climbing, essentially. If it sees that, well hill climbing in the sense, hill climbing always moves to a better spot. The actual algorithm moves to the best amongst them.

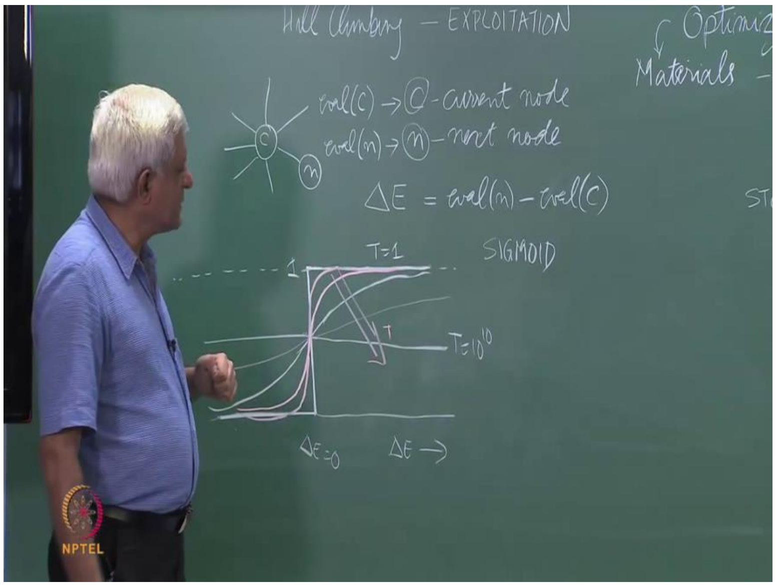
But, since we are generating only one, we will take this as an approximation to hill climbing. So, it would move only, if the node n was better than node c, if temperature was low. If temperature was high, it would less move, irrespective of whether. So, as an exercise I would ask you to look at, either of these values and construct the table for T.

And you will see that, there would be similar thing. That they would, at high temperature they will, it will still converts to 0.5 and at low temperatures, it will converts to zero, towards zero. Then, it will not move it all, essentially. If we are getting a worst, these two values are worst in this and these two values are better than this, essentially. So, we have noted. We have seen the example of a better value.

So, how does this sigmoid function look? If you are to plot it, it depends on the parameter temperature ((Refer Time: 28:51)). So, let us say, this is the value of 0.5 through this. So, this is all curves was passed to this point, essentially. Why because? We have said here that when delta E is equal to 0, it gives us a value of 0.5. When, you can see that, this is going to be irrespective of temperature, because this term has become zero ((Refer Time: 29:13)).

So, irrespective of temperature, this value is going to be 0.5. So, all the curves for temperatures will pass through ((Refer Time: 29:20)). When temperature is low, this curve looks like this.

(Refer Slide Time: 29:28)

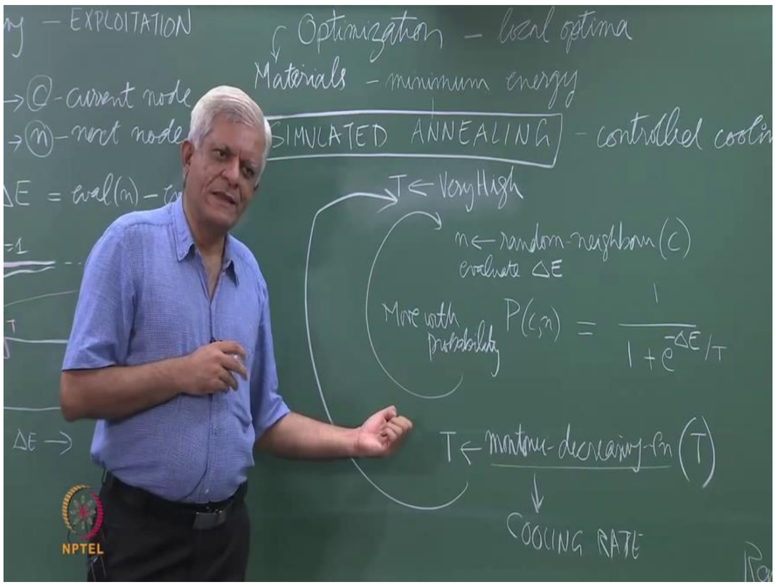


Basically the step function. This is T equal to 1. When temperature is high, this curve looks like a straight line, here. This is, what T equal to, this is 10 is to 10. It is always 0.5. Otherwise, it gives us a behavior, which looks like the following. So, a typical sigmoid curve will cross this at some place and then go down towards zero here. And go up towards one place. So, this is a typical shape of a sigmoid curve.

And it is nice for us, because we want a function, which will have the range of 0 to 1 and domain which is infinite. And as we change the values of temperature, the shape of this curve, changes. So, with a different temperature, the curve may look like this. So, as temperature goes down, this curve, this slope become sharper and sharper, like this. And as the temperature goes up, it becomes flatter and flatter essentially.

So, this is T raises to 10 is to 10. Somewhere, you know there will be a curve, which will be like this. So, there is a series of curves. So, this is a direction of increasing T. The curves become flatter as T. This is just a nature of this function. We are just looking at it, to get some insides and towards happening essentially. Now, instead of saying that, I will make a choice of the value T. We follow, what is done in the physical world, which is to say that, I will cool down the system gradually. And hope that, it will settle into an optimal state essentially.

(Refer Slide Time: 31:57)


EXPLOITATION
Optimization – local optima
Materials – minimum energy
SIMULATED ANNEALING – controlled cooling
ⓒ-current node
ⓝ-next node
T ← Very High
n ← random-neighbour (c)
evaluate ΔE
Move with probability
$P(c,n) = \frac{1}{1+e^{-\Delta E/T}}$
T ← monotone-decreasing-fn (T)
COOLING RATE

So, if we remove this and put in a clause, where we initialize T into some values, let us say very high. Whatever that very high value is… And very often, the algorithm has two loops. There is one loop in which, you do this loop a few times at a constant T. Some people, call it as e power something. But, that does not a matter. It is a inner loop, in which you do this whole process. What is this process?

Generating a random neighbor and either, moving to it and not moving to it, depending on how good that neighbor is given by this probability here, essentially. You keep doing, this as a certain number of types. And then you change T from monotonic decreasing function. So, something, some function. The simplest is, T is equal to T minus 1. But, that may not necessarily with the best function, you know. This is a very empirical process.

This function, that we are looking for determines. What this people call it as a cooling lead. And this is, at most of the times determined empirically essentially, which means that you take a domain, do some experiments and try to see, what works and what does not work and then try to keep that value. And then this is a outer loop. So, you do in an inner loop, you do this some number of steps. Then, you decrease the temperature and do some more number of steps essentially.

So, what is the intuition behind this? And simulated annealing. So, it is called simulated annealing. It is a algorithm, very well known algorithm call simulated annealing. It is very popular in the optimization community. It is used very often. And this is basically the algorithm. That you started at the high temperature, which means you start with high random movement. Allow the system to go to any candidate.

But, gradually bring down the temperature and make it behave more and more like, hill climbing. What would be the intuition? Why should it work? What is the idea behind this decreasing of temperature? How does it help? Decreasing the random means, therefore it does. But, what is the intuition behind? Why does it work? Why does it help us find, good values good.

Student: ((Refer Time: 34:55)) local meaning.

But, remember that towards the end, when T becomes low, it is going to behave more and more like hill climbing. How do you account for that fact? It is true that, initially we wanted to behave randomly. And then later on we wanted to follow the heuristic function or all essentially. What is changing over this period of time?

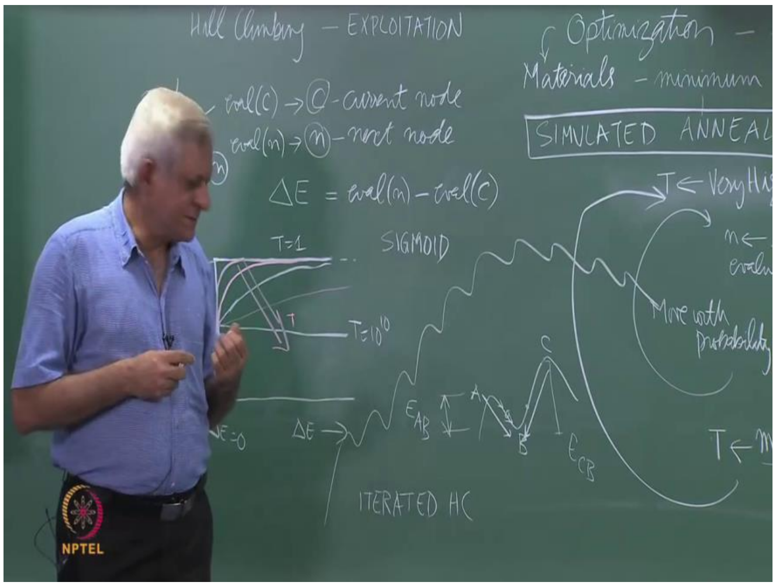
Student: ((Refer Time: 35:31))

If that is high initially, why do you want this high initially and what is the effect of this cooling? You know, I heard that somebody say that, this is like one of those toys, which has the concerting rings. We might have seen with some small balls of wall bearing like objects inside them. And you have to move them all towards the center. And this person who is a respected computer scientist said that, you know.

It is a bit like, you first move it randomly and then gradually. You let it try to control the movement. But, you think that object will little bit or something like that. So, let we give you an intuition behind this. So, what is the claim that we are making? The claim that we are making is that, at many problems this gives us very good solutions, which means it gives us close to optimal solutions essentially.

What are the kind of problems, where this works, were the surface is gadget essentially. Where hill climbing would have got stuck at first local maxima or minimize, it is found? But, this algorithm can get beyond many local maxima and move towards better maxima. So, what is an intuition to say that, it moves towards better maxima? So, this is a basic idea that, I would not make you to think about. Let us assume that, that is… So, there is this big gadgets surface.

(Refer Slide Time: 37:02)



So, let us say. This is how your one dimensional surface looks like, essentially. Its gadget in the sense that, hill climbing would, depending on where it is started, hill climbing would have got stuck in one of these local maxima. But, we are saying that something like simulated annealing, will go very close to the global maxima, essentially. So, the idea is this essentially. That, take a certain snap shot of this, which looks like this.

It is like that, one of these things here. Let us call this point A and next, call this point as B and next call this point C. And we want to maximize which means, we want this value to be high, higher. We want to go to C rather than to A. We want to go to the top of this essentially. Now, in this one dimensional world which means, you know these are the set of candidates and these are the heuristic functions and you can move either left or right.

You want to not get stuck at A but, you want to be at C. And then this I have given to apply all over this slope, essentially here. Now, you can see that, to move from A to B, it has to overcome an energy gap, which is this much, which means when I say overcome the energy gap, basically I am saying that, it is going against the heuristic function. You want to maximize but, you want to go down here.

You want to go down here, maybe two three steps depending on, what is the granularity? You want to take two three steps down there and end up in B. And of course, then after that, it will have a natural tendency to go up because, that is the natural tendency of this algorithm. If it sees a better move, as we have seen here ((Refer Time: 39:01)). The better the delta E, the better the probability of moving to that move, essentially.

So, it is a natural tendency of going here. So, to go from… So, let us call this E AB. And likewise, let us call this E CB. To go from A to C, it has to overcome an energy gap of E AB. To go from C to A, it has to overcome an energy gap of E CB. So, just imagine that, there is somebody with the helium balloon tied to that first. Let us try to visualize, trying to pull him up essentially. So, he has to walk down all this way. Pulling that helium balloon which is a greater distance, essentially.

So, a greater energy gap essentially. So, to go to from A to C, you have to overcome this energy gap. And then of course, naturally the algorithm will take you up, because it has this tendency of going up. To go from C to A, you have to emerge the greater energy gap essentially, which is this E CB essentially. As you works for this, you can certain comment. The idea, which says that, there is a greater chance of it, moving from A to C, essentially.

In general, not initially. Initially, when the temperature is high, going up and going down is all the same for this algorithm. It will just do any more randomly. But, as the temperature goes lower and lower, it is still not hill climbing. It is still some intermediate temperature. But, either intermediate temperature as you can see, ((Refer Time: 40:42)). The probabilities, unfortunately we have drawn the probabilities for negative moves.

But, maybe we should do that and see that. That, as the temperature goes down, the probability of making negative moves goes down essentially. As a temperature becomes one, the probability or negative moves tends to zero essentially. So, as a temperature goes down, the probability of moving some energy gap will depend on the energy gap essentially, And since this gap is larger than this. It is no likely, that algorithm will go for A to B and then to C.

And remember that, B to C is a kind of automatic. Like in the sense, there is a most likely behavior, because of this thing. And it is less slightly, it is come down from C to B. And then go to A essentially. And this is specially the case, as temperature is brought down essentially. So, you can see in some sense, you can see that temperature it is ability to go down in this slope essentially. As temperature is high, the algorithm has higher ability to go down or you can say, go down greater distance essentially.

As we gradually reduce the temperature, it is ability to go down in this. Remember, this is maximization problem. Decrease this, which means that it is easier for it to go from A to B, then it is come from C to B, which means that it is more likely to end up in C. Because, after the temperature has become really down. If it happens to be here, it will go to C. If it happens to be here, it will go to A essentially.

That is hill climbing part of it, essentially. So, this is a kind of intuition behind simulated annealing. That initially, you allow the algorithm to explore many different parts. But, gradually bring down the temperature. In the process, in some sense pushing it towards the global maxima, essentially. Pushing it towards higher peeks, in this example essentially. Why because? It is at any given temperature, which is not infinite.

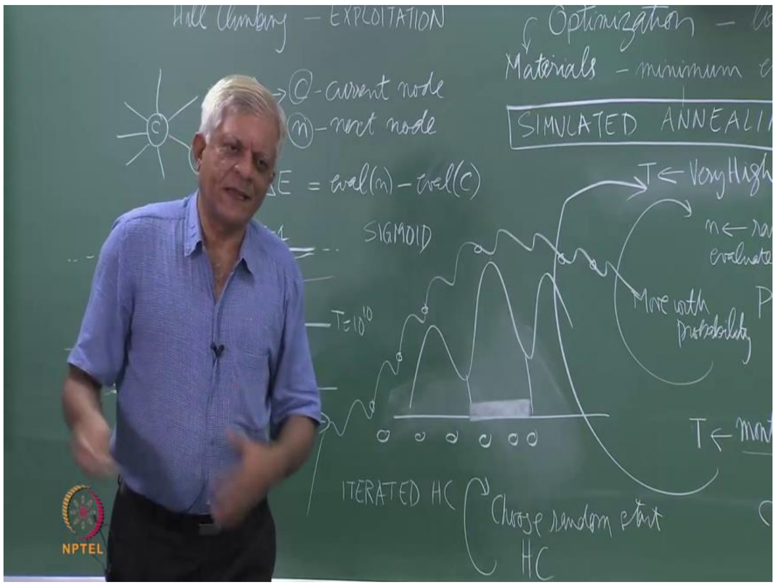
I mean, if you assume 10 is to 10 is infinite. It is more likely that, it will climb down form it is local peek, then it is from a higher peek, which means it is more likely, that it is a little climb up to the higher peek from a local peek, which means that this is very likely, that it will go from A to C essentially at any given temperature. And eventually, you bring the temperature down. So, that it is a kind of just goes to that nearest peek that it is that.

Actually, there is a simpler version of randomize algorithm, which I have not mention. But, this is probably a good time to mention it. That algorithm is called iterated hill climbing. And what it essentially does? And this is, this works in the solution space. So, something like, if you are solving sat or solving TSPS or something like that, where you are looking for a solution. You are not looking for a path from the start state to the goal state.

In fact, that there is no notion of go start state. So, ((Refer Time: 44:06)) for examples. There is no start. Unless of course, you think of the empty board as a start state. But, that is not really critical to the solving the problem. So, if you want to solve ((Refer Time: 44:16)). You have to find with the final solution, somehow with the other essentially. And so when we look at the solution space search and perturbation methods, you could start at any random location, essentially.

So, again consider SAT as a example. You can choose any supposing, you have n variables. You could choose any assignment for those n variables. And that is a starting point from which you apply those operators, that we discussed earlier, essentially.

(Refer Slide Time: 44:48)



So, what iterated hill climbing says? You essentially, that choose a random state of starting point random means. So, in this example, it would mean choose one here, choose one here, choose one here, choose one here and so on. So, put it in some kind of a loop. Choose random start. And simply do hill climbing. Do not do anything else. Just do hill climbing. So, you can imagine that, if I want to do this, then this starting point will take me to this maximum.

This starting point will take me to this maximum and so on and so forth. And if you have chosen a sufficiently large number of random starting points, then one of them is likely to hit global maxima, essentially. So, that is a hope. Of course, it depends on the nature of the surface. If it is very gadget, then you may need many, many starting points. But, if the surface is like this, then you can see that if we start anywhere between this range, means anywhere here and you will end up with this global maxima essentially.

So, iterated hill climbing is very simple algorithm. It says that, do many hill climbing searches. But, start at different randomly selected points and because, hill climbing is simple. Computationally, this algorithm is simple essentially. You can also think of this as a parallel search. That you start all these searches at the same time and all of them happens in the same time. So, after the end, you will just find what is the best candidate that you got? And you, that is an answer essentially.

So, there is something to be said with more than one candidate, searching with more than one candidates. So, we have already seen a couple of examples. Beam search, searches through. Always selects be best candidates in the next level and so on, if remember. This is little bit like beam search. In the sense that, except that you ,these things are not allowed to interact with each other.

So, this will have its own trajectory and this will have its own trajectory and so on. But, they are the be parallel searches taking place, essentially. Does it make sense for two candidates to interact in some way? If you are doing a search in which, there are many candidates like a parallel search, like this. Does it make sense for two candidates to interact? We will address that, in the next class which will be and we will stop here, with this. So, that is a new algorithm that we will look at, which is based on working with a population of candidates, which are allowed to interact with each other in some sense, essentially. We will stop here.