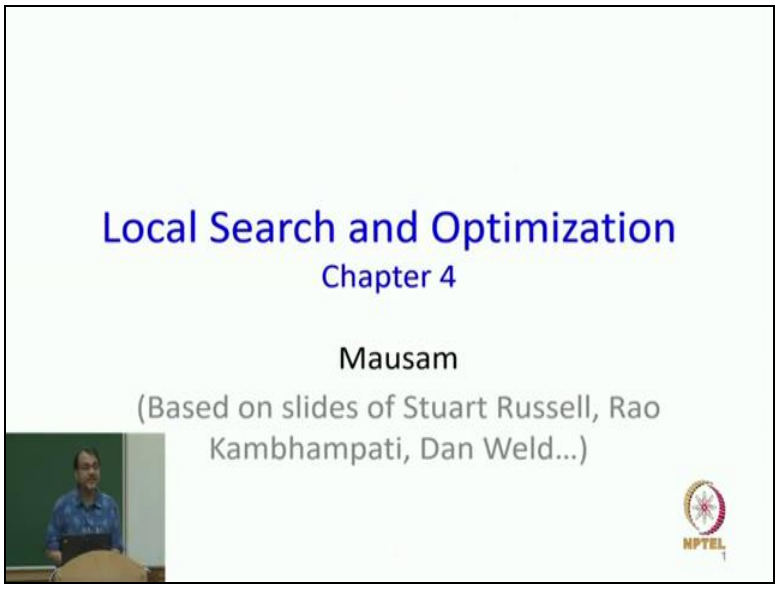
Artificial Intelligence
Prof. Masum
Department of Computer Science and Engineering
Indian Institute Technology Delhi

Lecture-22
Satisfaction Vs Optimization Part – 1

Hello everyone, today we talk about Local Search Algorithm. And I do not think I need to read the importance of search we are we doing this for the past many classes and we have learnt that search is extremely important. It is a fundamental way to think about solving a new problem and to solve a new problem; the first model it as a search problem and then apply as one of our favourite search algorithm.

And we have done many, many search algorithm. We have done at least 4 or 5 uninformed search algorithms at least for the 5 informed search algorithms. We have done a lot of algorithms already right and those are good algorithms. But today I am going to talk about local search algorithms, is the different way to think about search algorithm.

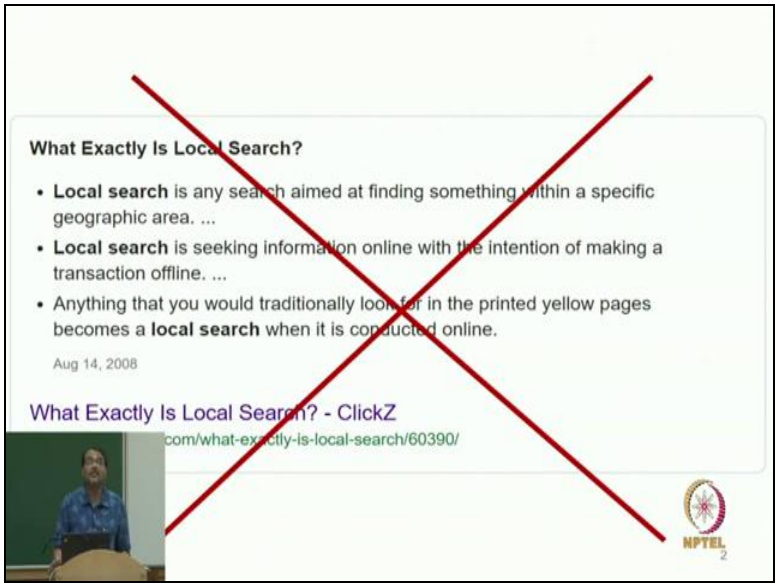
(Refer Slide Time: 01:19)



And before I get started as I always mention I would mention that look we stand on the shoulders of Giants and many of these slides are the material that we have learnt has been due to some of the forwards in the field in this particular case, you know that the slide that I have used, have been developed by Stuart Russell and used in Berkley and Partly in Arizona state and Dan Weld

and many other folks at University of Washington. So, what is exactly local search and why are we going to study this and what is the special about it right this is the goal for today. So, when you think of the word local search in the modern day age, we think about something like this, right.

(Refer Slide Time: 01:58)



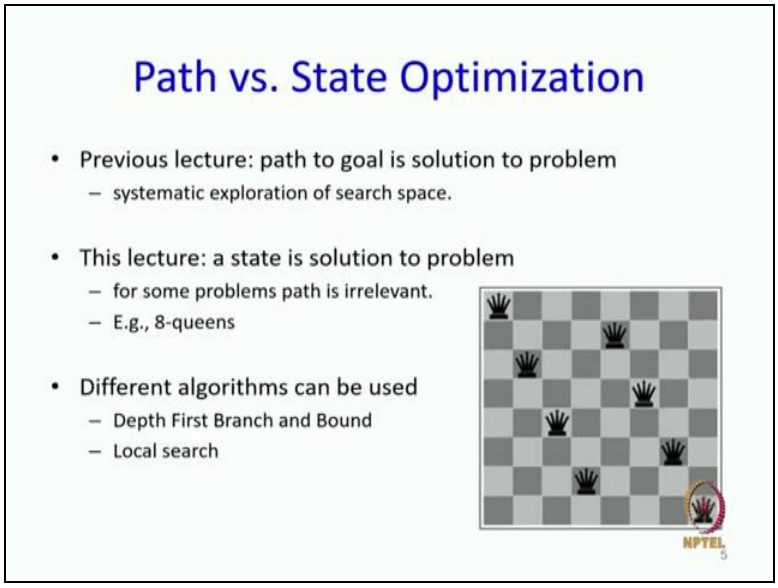
Tell me all restaurants near it right or as this says what exactly is local search when in doubt consult Google first, this is the, this is a rule in life these days. And local search as it says is any search aimed at finding something within a specific geographical area. Notice that this notion of local search is relatively model. Like this, happened in the last 10, 15, 20 years Google itself came into being in 1998 probably not more than that.

I do not know this notion of local search was present in this form, in the previous generation, non internet enabled system. But we are talking about local site that has been around for the last 50 years at least. But definitely we are not talking about I am not going to teach you how to do local search in Google, I am not going to teach you how to get the right restaurants or the right attractions near a certain location. This is not what we are going to talk about.

We are going to talk about general search algorithms, any model can be, any problem can be formulated as a research problem and is solved using one algorithm, one search algorithm called the Local search algorithm. It is not just one algorithms, it is a series, a set of algorithms. It is a

family of algorithms. We will talk about this in this lecture series we will end up talking about 20 algorithms and is ok even we will understand them with their all sorts of brothers and sisters of each other. So, up until now there are some differences in the way we have formulated the problem earlier and we are going to formulate it now.

(Refer Slide Time: 03:39)



In the previous set of lectures, the solution to our problem was a path to our goal. You have a start state and goal state, of many goal states and your solution was to find what sequence of actions can I apply so that I can go from the start State to a goal state. A state was some partial information about the search we have done so far. That was not about the state was representing at least a node was representing state maybe more than that but it is of node of that.

And we would do a relatively systematic exposition of the search. Of course, we also talked about the algorithms which are not systematic like idealistic during search. But we also realise that while they are important but not systematic they are mimicking a systematic algorithm in a memory efficient. So, internally principally what is going on is some kind of exploration so that once I have explored a subtree, I can go to a different subtree and so on.

We are going to turn things around completely in local search. It is the fundamental difference. We will not be in a setting where not the path but the state itself is the solution. Another way of saying this is we are going to search in a solution space. Each search node would be a solution a

good solution, a bad solution, but it will be a solution. In the worst case you can return it and get some correct one. And why is this useful? It is useful because in many situations parts are actually irrelevant.

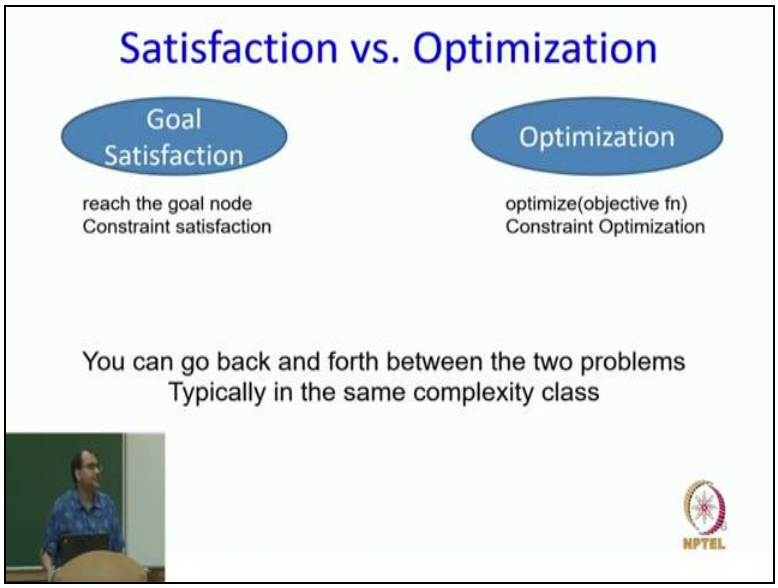
If you think about the n Queens Problem just to remind you what is the n Queens Problem I have to put n Queens on the board that no pair of queens is attacking. Now in this situation, how did we model it as a search problem in the previous lecture slides? Does anybody remember what was my start date? MP board, very good. And what was each action? Add 1 queen, very good. And what was my goal test? All the Queen have been placed and note ((()) 6:17).

Everybody with me; this is how we model it and there is nothing wrong in it. But notice what this was trying to do? It was trying to find the shortest path to a goal. Of course, in this case all the goals reside at Length and I mean at depth and it does not matter as soon as we find the first we will stop etcetera. But really it really did not matter whether I place the queen number 1 first in the first column, of queen in the second column, first.

In fact we can even do a better job of search by imposing and ordering in which we place the queen if we want to. And will get to that in the next set of search. But for now we recognized that we are not really interested in a part. We are only interested in the final state which satisfies all our constraints where should the Queens be placed in that final state, right? And we do not know that final state and that is what we are looking for.

These kinds of problem are really well modelled as local search problem. If you can use depth first branch and bound Idea Star anything like that, but you can also use local search method for these problem and they will open end up being faster in practice as I will show you, ok. So, so let us there is one other thing I want to talk about.

(Refer Slide Time: 07:40)



Up until now, we had been looking at satisfaction problem. Satisfaction problem says I want to find the path to the goal and then once I find I want to find the best path to the goal. So, there is some optimization going on here also, but suppose I am only interested in find me a path to the goal. That is what I am going to call a satisfaction problem, right. I am just interested in making sure that all my constraints are satisfied. It does not matter whether it is the most optimal way to satisfy them are not.

On the other hand, optimization problems, like find the optimal path to the goal or optimize the objective function in general. And notice I am doing this sort of at the back of your mind is some people know gradient Descent they can start thinking about gradient descent some people know back propagation they can start thinking about back propagation if some people know any other Optimization Framework you can start thinking about this.

Now, what is the harder problem satisfaction or optimization? Optimization so actually if you going to the theory of Turing machines and so and so, if you are complexity course, you will find that by and large all satisfaction Optimization problem set in the same complexity class. Let us see if we can convert a satisfaction problem into the Optimization problem. I have some constraints. I want to satisfy all search constraints.

Can I cast into an Optimization framework? How? Let us say given a state of constraint fn And my goal was to find a state which satisfies all of them. On the other hand now, I am interested in Optimization problem. Can I cast this original problem into an Optimization Framework for Optimization have to minimise something what will I minimise? Cost, what will be the cost here? Oh, Total cost for an operation, I do not understand.

Number of constraints satisfied, very good. Right, Simple. Just minimise the number of constraints satisfied. Maximize the number of constraints satisfaction and minimise the number of constraints which are not satisfied. And you got to zero, in this minimization problem, then, you have found a solution to the satisfaction problem. If you optimized it to that level and if you still cannot find that means the constraints cannot be satisfied.

Really, any constraint satisfaction problem can be written as an Optimization problem by just by saying that minimise the total number of constraints which are not satisfied. Now, going back in their direction is slightly tricky but not that much more tricky. For example, if given Optimization problem when I say minimise and Optimization function subject to some constraints, if I want to send it back to the satisfactions world. What would I have to say?

So, in the satisfaction world we have to find a solution that satisfies all constraints, but now we are objective function and want to minimize or maximize. That is a maximize. No, you have to write it down. Now, I do not want an algorithm. This is one major learning in that I am hoping you would get, I have been trying to install it again and again. Please, when I say formulate this problem as the other do not think of an algorithm. Do not think of an algorithm.

Please think of the modeling, does that make sense. Model of a problem has the other means, whatever inputs which algorithm needs you change the inputs of the first problem in, in the format that the other problem is able to take it the other algorithm is able to take it, Ok. So, there is another formulation it is able to take it. I am interested in formulating and Optimization problem as a satisfaction problem.

In satisfaction, I have given just a number of constraints. In Optimization, I have given the number of constraint and an objective function. So, what do I do? Which cost? The optimization cost is maximum that is the constraint. So yah possibly, you could say something like this, you could add a constraint and objective function less than equal to C or greater than equal to C that does not matter.

If I get a solution Repeat with objective function greater than equal to C Prime which is greater than C and do some search in the constraint settings and try to solve several satisfaction problem, leading to the answer to the Optimization problems, arbitrary closer to it, you can do that. And in practice what people have found for not in fact that people found is that most Optimization problems when converted to satisfaction problem are in the same complexity class.

So, really we can assume that the satisfaction problems and Optimization problems are not very different from each other. They are certainly equally difficult or easy more of in complexity class. You can still see yah question is optimization cost is in the real domain. What do you do? You can get to arbitrary closest you can keep doing binary search in that real space and keep asking the satisfaction function if you have a solution or not.

More number of times you ask that question, the more number of times would be able to see that where the exact optimality is done. At some point will find the right solution. Complexity will not change very much because in practice you will not need to do too many satisfaction calls. Again, you can you can have strategies; you can double the optimization cost due to binary search in that space and some tricks. So they are the tricks of the trade which we can employ to model and Optimization problem for a major satisfaction. Ok

Nobody does that in practice. These are more of security collection but can be done. Ok. For example, a constraint satisfaction problem that you are going to study next, in the next set of lectures would be as goal satisfaction problem satisfaction problem, but they are equivalent Optimization problem recharge which have constraints in addition they also have an objective function.

And now if somebody knows neural networks what is the training procedure? Most people do not know that one. Same procedures of Optimization problem subject to the constraints of the network, what is the meaning of a parameter what function is thing is computing, optimise the objective function for me. The objective function could be a fit as closely training data as possible. All this deep learning business that people are crazy about, at the core of it, it is in Optimization function.

However, in the class that we are studying today, we are not going to talk about those kinds of Optimization problems very much because those kinds of Optimization problems are continuous optimization problem, where your variables are real valued because the parameters are real value. However in this class today, we are going to talk about discrete optimization problems. Optimization problems are where the variables are discrete and not continuous.

So, ILP, Integer linear program can be thought of as an Optimization problem that we can solve. But linear program we will not solve because that is a different set of techniques. Although at one Level abstraction they will be similar that is the beauty of it and we will touch on that towards the very end of the class. So, with that background where are we? So, we are now in a setting where my problem is an Optimization problem.

And the solution of the problem is a state. These are the two constraints that are needed to apply local search. And of course, if you think a little bit deeper than if you really want pass as a solution then what will you do as a state? Your state would be many, many parts? You can always convert a systematic problem into a local search formulation. We are we are saying state is a solution to let me search in the park space. If I can search in the path space, then I would be able to apply local search. So If I did not get that. It's ok not a big deal.

But the point I am trying to make is anything that you could have done earlier, you can do now in the local search version also. And we will be interested in the Optimization problems as soon as they look at a state; as soon as a look at a state, we should be able to say that this state has this much objective function. So, we have an objective function, black box given to us where we have put in a state a solution a good solution and a bad solution and it tells me how bad it is or

good it is and what is the value of the objective? This is the setting we are in. Ok. So, just to give you a heads up where we had a local search.

(Refer Slide Time: 17:35)

Is this mechanism, which only keep track of a single state does not worry about everything else how it reach there, etcetera etc. It just worries about the single state and gets moved to neighbouring States which is why it is called local search. You are searching in the local neighbourhood. And what is local neighbourhood we will talk about. And the advantages of this procedure would be reduced very little memory and moreover its scale is really well in practice.

No guarantees more often than not but it will be able to find a reasonable solution in large or even infinite state spaces. As we said, these are a pure Optimization problems all state seven objective function. The goal is to find the state with maximum or minimum objective functions. It does not really quite fit into the path cost, goal state formulation. All the path cost goal state can be converted into local search formulation and technically you can convert a local search formulation into a path as post state formulation, but may not be most effective.

And local search really does well on these problems. So, we can go back and forth but the formulation that we have is given a state, given a neighbourhood of a state, given an Objective function with evaluate the state find me the state which has the highest and the lowest value of the objective function will continue from here in the next lecture.