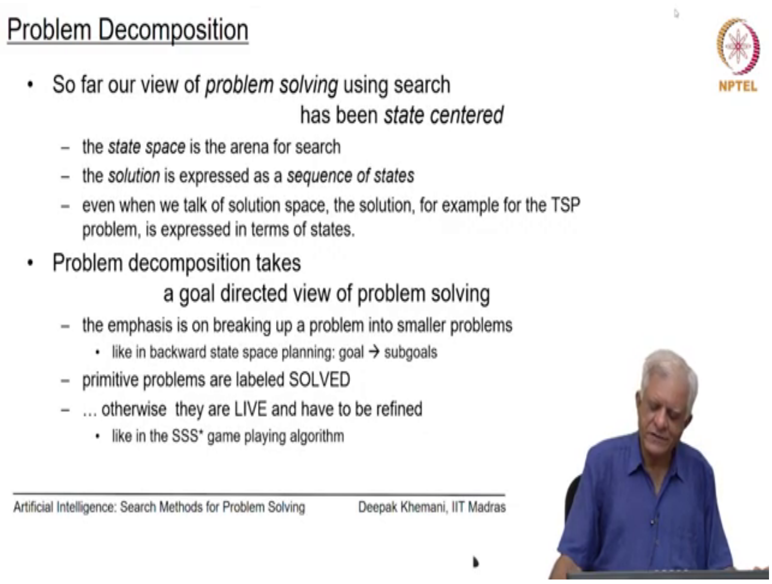
Artificial Intelligence: Search Methods for Problem Solving
Prof. Deepak Khemani
Department of Computer Science and Engineering
Indian Institute of Technology, Madras

Chapter – 06
A First Course in Artificial Intelligence
Lecture – 69
Problem Decomposition

So, welcome back, today we are ready to start slightly new topic. So, far in our course we have investigated methods to construct solutions, we are essentially we begin with a start state and look for a goal state; but very often we think of problems differently. We think of a problem as made up of sub problems and we break down the problem into sub problems and try to solve them and there is an approach which takes this strategy and this is what we want to look at today.

(Refer Slide Time: 00:56)



So, this area is called Problem Decomposition because we are breaking down a problem into smaller problems. So, a quick recap of what we did so far our view of problem solving was based on state space search it was centered around the state, the state space was the arena over which we searched. And the solution is expressed as a sequence of state.

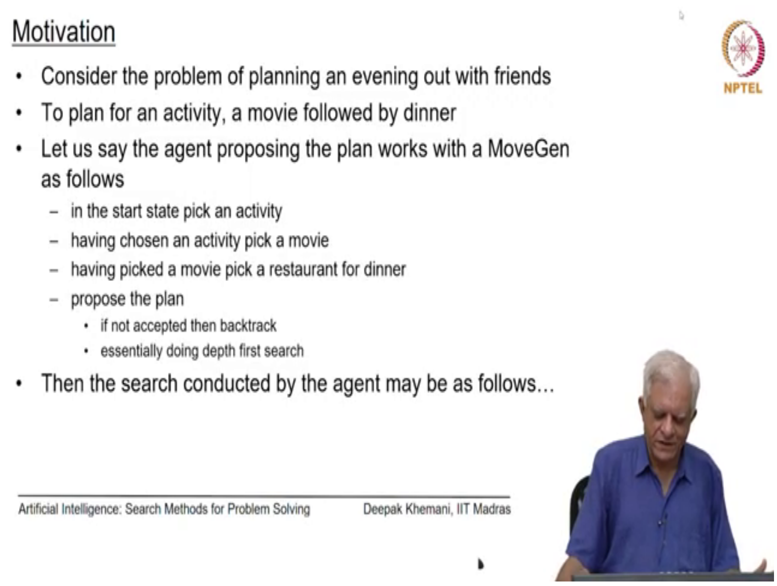
And we had seen this that even when we move to solution space; the solution was still expressed as a in terms of states. Now, we are moving on to something which is a little bit different which takes a goal directed view of problem solving. The emphasis is on breaking down a problem into smaller parts a little bit like what we did when we looked at backward state space planning where we regressed from a goal to sub goals and did that.

This approach has a similar flavor that we are thinking in terms of the goal we want to achieve and trying to also exploit the fact that problems have sub problems which can be solved independently.

So, in this space, in which we look at a goal directed approach to problem solving we have two kinds of problems one which are problems solved. These are primitive problems for which you do not need to do anything further and the other kind of nodes in our space search space would be live nodes and they would need refinement.

And this is a little bit similar to what we did when we looked at the SSS star algorithm for game playing; in which we had partial solutions which needed to be defined further and so on. In fact, as we study this approach you should try to compare it with the algorithm SSS star that we developed earlier.

(Refer Slide Time: 02:58)



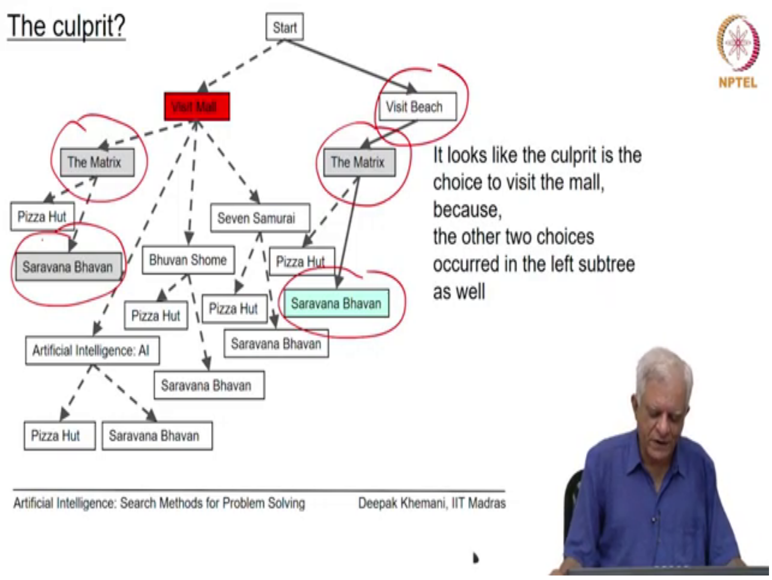
So, let us start off with some motivation first and as a motivation let us look at a problem which has typically three parts. And let us say this is a problem of planning an evening out with friends and let us say you are the one who's proposing the plan and the basic idea is that you must do some activity followed by a movie and followed by dinner essentially.

And the plan you are working with something which is like a MoveGen function which was in the state space, it would be as follows that; in the start state you first pick an activity, then having chosen an activity you pick a movie to go to and having picked a movie you pick a restaurant to dinner.

And you propose this as a plan to your friends, if they accepted it is good we you can go ahead and execute it, if they do not accept it you backtrack essentially. And the algorithm that we are just looking at now as a precursor to what we want to look at is essentially depth first search. And this is how the search space would be explode essentially.

Remember that this is basically depth first search and remember that your problems solution is in three stages pick an activity, pick a movie and then pick a restaurant for dinner. And only when you have picked all three will you ask your friends is that essentially.

(Refer Slide Time: 04:32)



So, this is how depth first search would start you pick some activity let us say the most common activity nowadays you Visit a Mall and then you say you will go and see the film Matrix and then you would have dinner at Pizza Hut. And you present this to your friends and they say no somebody objects to it and then you do this depth first search over these possibilities and this is how you explore the rest of the search space.

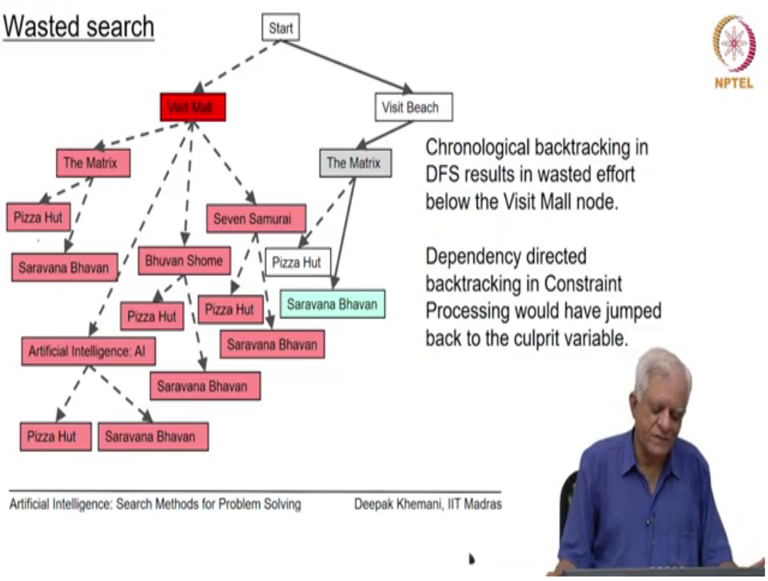
So, instead of Pizza Hut you propose Saravana Bhavan then you backtrack and you say maybe instead of Matrix you would see AI the film. And again you propose Pizza Huts and Saravana Bhavan it does not work, then he proposed Bhuvan Shome the film and again Pizza Hut and Saravana Bhavan and so on.

The search continues; and then eventually you backtrack all the way to start and say maybe we should go to the beach and then you propose the film Matrix and Pizza Hut and then Saravana Bhavan. And at this point let us say your friends accept and say and so what is a plan? The plan is visit the beach watch the film Matrix and then eat at Saravana Bhavan.

Now, if you look at this space that you have explored carefully you would see that probably the culprit choice was the Mall essentially. Because if you look at the solution that we have looked at which is to visit the Beach, to visit the Matrix and to see the film Matrix and go to Saravana Bhavan of these, these two which is a Matrix and Saravana Bhavan was there earlier also and the only thing that changed from this accepted plan to the rejected plan is it there was a mall visit essentially.

So, maybe that is a culprit, but searching the space using depth first search does not allow you to identify that.

(Refer Slide Time: 06:35)



In fact, this entire search tree that you have see on the left hand side covered in colored in reddish color is a waste of effort. Because having chosen the mall if the mall is a culprit choice then you work whatever you do subsequently is not going to lead to your solution essentially but depth first search that is chronological backtracking.

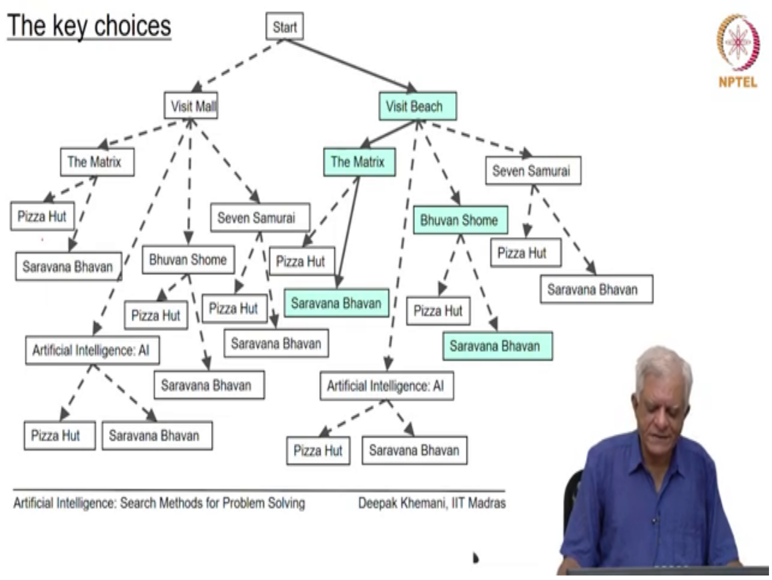
So, after having being rejected for a solution it first backtracks and tries a different film or a different restaurant; then if that does not work it tries go backs and tries a different film and so on. And evens only when it has completely explode the side where the first activity was mall that it backtracks and goes to the second activity.

Now in the community of constraint satisfaction or constraint processing, people have worked out with solutions where instead of chronological backtracking you can as they say jump back to a relevant node or a culprit node.

If you can somehow identify that the solution for example, the first solution that visit Mall, the Mmatrix, the Pizza Hut or even after the second one which is Saravana Bhavan; you realize that it is a Mall which is a culprit you go back to that variable or and then change the value of that variable.

That kind of an approach is called dependency directed backtracking and if you ever study constraint satisfaction problems you encounter various algorithms which take different approaches to jumping back to culprit nodes.

(Refer Slide Time: 08:10)

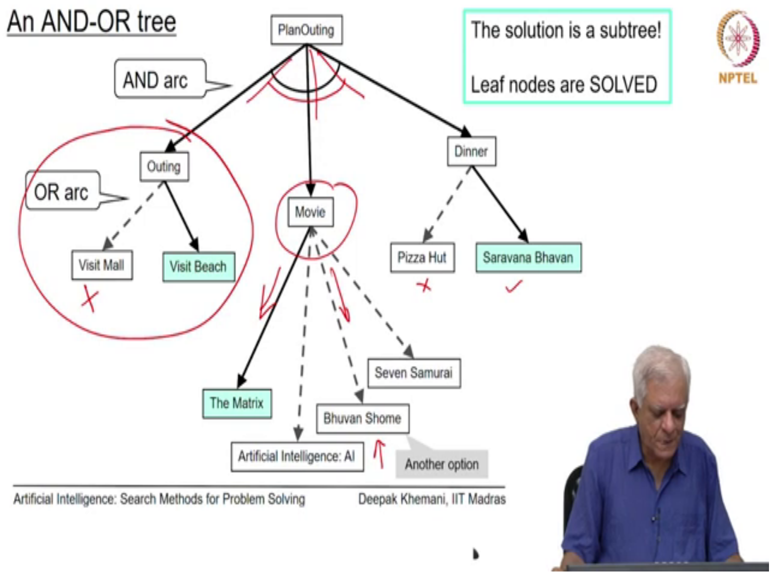


If you were to look at the complete search tree that of the options that were available to you the options that in this mall example we have are either go to the Mall. Or go to the Beach and either eat in Pizza Hut or eat in Saravana Bhavan or one of four films the Matrix, Bhuvan Shome, Seven Samurai and on three films Matrix, Bhuvan Shome and Seven Samurai.

So let us say what we have depicted here are the two solutions that would have been accepted by your friends the one on the left which is the Beach, the Matrix and Saravana Bhavan is the one that you encounter first when you are solving the problem using depth first search. And maybe there is a second solution that you could have possibly proposed, but in this case you did not have to which involved watching the film Bhuvan Shome.

Yeah, the first film is AI the film essentially. So, there are four films to activity than two restaurants. Now it makes much more sense to look at this problem; now this problem has clearly to three different components which is to organize the activity, think of a film and think of dinner.

(Refer Slide Time: 09:28)



So, the search space that we are interested in looks something like this essentially. So, at the top level this thing and this is called an AND - OR tree and we also have AND OR graphs as we will see in a some examples. So, there are two kinds of nodes here one are AND nodes. So, the top most node here is that you have to plan an activity or plan an outing. And those three edges coming from there connected by this arc you can think of that as a hyper edge or a hyper arc and it is an AND arc essentially.

It says that to plan an activity you must plan an outing, plan an outing you must plan an evening activity or outing you must decide on a film and you must decide on where to have dinner. So, these are three separate problems, but the fact that they are depicted in an AND arcs means that you have to solved all three of them essentially.

And then at the next level when you are talking about the evening activity which is the outing you have two choices, either you visit the Mall or you visit the Beach and this problem of evening activity is completely separated from the rest of the problem other two problems which is movie and dinner.

So, likewise movie and dinner have their own choices and these are or choices. So, this kind of a space that you are searching is called an AND - OR tree and if there are common activities it can be called and or graphs, you could have we will see an example where it is a graph and not a tree essentially.

Now, the important point to observe here is that the solution is a sub tree; unlike in the state space approaches that we have seen where the solution was a path from a start state to a goal state. Here the solution is a sub tree of course, you might remember that when we looked at algorithm SSS star for playing games, there the solution was a strategy which was also a sub tree of the game tree.

And you will find more similarities as we go along looking at this approach of solving problems using and or trees or AND OR graph essentially. The leaf nodes in this space are the SOLVED nodes which means you do not have to refine them any further essentially.

Of course if you are working with a different level of detail then for example, the visit beach node would have been further broken down into sub steps and so on. And the kind of sub steps that we talked about in it hierarchical planning or even when you looked at mean sense and analysis we will do a quick recap of means sense and analysis after we have done with this topic.

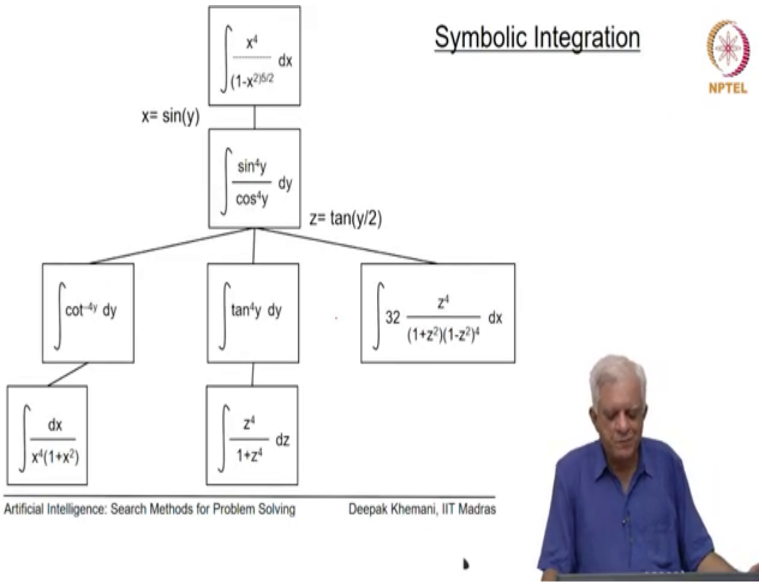
But it is a level of detail that may be different essentially, but in an AND OR tree the leaf node is a SOLVED node which means you do not need to refine any further; because in this case you are simply trying to make the choices where to go out where what movie to watch and where to have dinner we end the problem solving phase here essentially.

If you were to do the whole thing autonomously they may have been further refinement of each of those three components of the solution, which is visiting the beach and going to the matrix for example, you have to buy tickets and all this kind of stuff and so on.

In this case as we saw there were two options that, your friends wanted to go to the Beach clearly because all the options with Mall were rejected so that is essentially not a candidate. And also at least in the case of the Matrix the film we saw that Pizza Hut was rejected, but Saravana Bhavan was selected.

But we had seen in the space that we were looking at earlier that Saravana Bhavan that that Bhuvan Shome would have been a candidate film as well and that is kind of reflected here in saying that it is a another possible choice here. So, you could have a two solutions and the movie choice could have been either Matrix or Bhuvan shome and you would have a sub tree which is a solution.

(Refer Slide Time: 13:50)



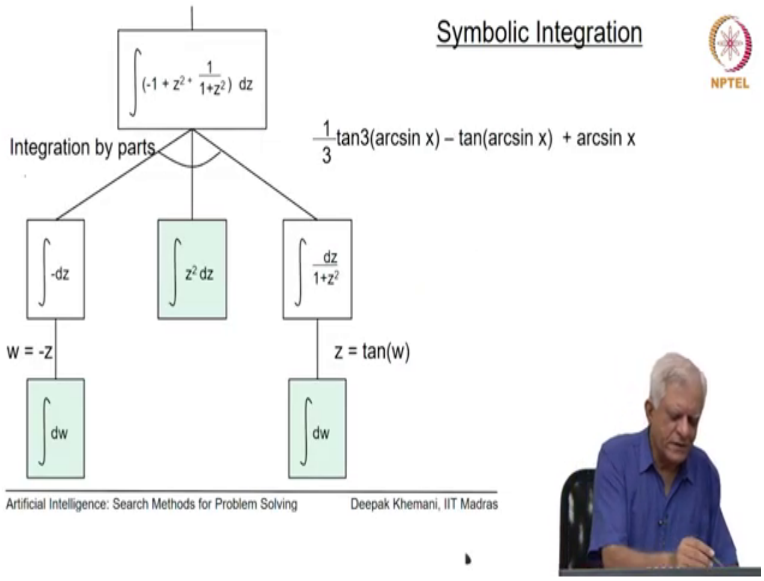
Now, from the very beginning and this is more than 50 years ago people have realized that AND OR tress are a good way to represent many problems. And one of the problems that has been addressed using AND OR trees is the problem of symbolic integration which you must have surely studied in your early engineering days or late school days.

And it is a thing which requires a tremendous amount of knowledge about how to transform problems into other problems which can be solved easily. And this is the flavor of problem decomposition that you have a problem to solve and can you transform it to another problem which can be solved more easily. And you repeat this process till the transform problem is so primitive that you do not need to refine it any further that is a flavor of and solving AND OR trees.

So, in this example we will not go into the details of this particular integration problem, but it seems a difficult enough looking at this thing and there are various things that you could try.

For example, you could substitute x with sin y and you get the second part which is sin raise to 4 divide by cos raise to 4; which you could either solved it by cot inverse or tan raise to 4 or break it up into something which looks even more do not thing and then this tan raise to 4 let us say we go by this choice we have reduce it to z raise to 4 by 1 plus z raise to 4.

(Refer Slide Time: 15:30)



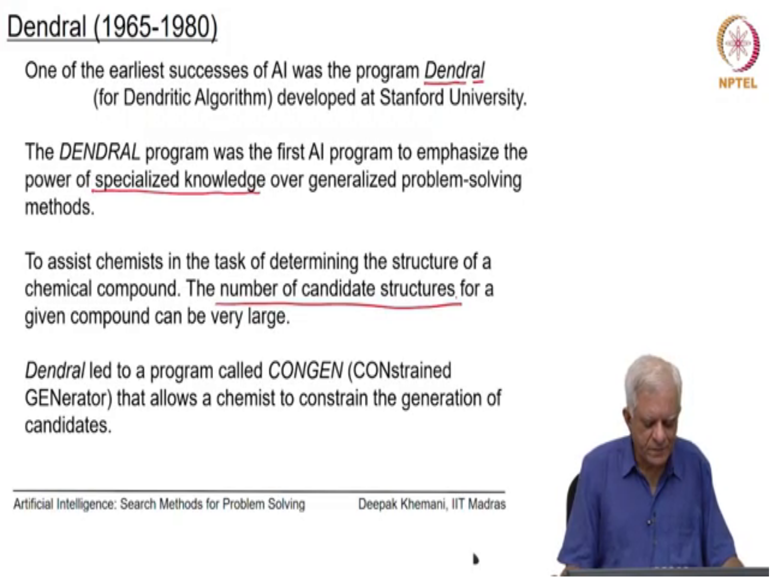
And that itself you can do using integration by parts which means that now this larger problem that you have, you have broken up into three parts, which is minus 1 z square and 1 over 1 plus z square. And these three parts are connected by then and arc and then eventually you come down to a very primitive distinct.

So, for example, integral of dw you do not have to worry about any further and integral of z square. For example, we assume that there is a lookup table or something where we know the values of these things and likewise for the third part which is 1 by z 1 plus z square you can substitute z equal to tan w and eventually reduce it again to dw essentially.

Now, this substitutions that you have made on the way we will have to be back substituted to construct the solution. And eventually if you are really inclined you can go back and see that the solution has those three parts which we got from integration by part and the what we see here is a result of back substitution.

So, symbolic mathematics is very popular nowadays and there are many approaches we started with a program called saint if long time ago then somebody interested in a program called sin. Then there was a commercially available piece of software called maxima and now MATLAB of course, allows you to do this sort of a thing.

(Refer Slide Time: 16:47)



Another problem where AND OR trees really shown was in this program called Dendral which was called an expert system and around the time the 70s and the 80s in the last century was in some sense the wave of expert system. Just like nowadays it is a wave of deep neural networks at that time there was a wave of expert systems and Dendral was one of the pioneering programs which led that whole activity.

In fact, it is seen as one of the earliest successes of AI; developed at Stanford University, this Dendral program is basically in two parts Dendritic and algorithms so, it got its name from there. It was one of the first programs may emphasize the power of specialized knowledge over generalized problem solving methods.

So, we have seen throughout this course that, search based approaches to problem solving which are general approaches problem solving. Run into this what we call as the monster called comebacks or combinatorial explosion or exponential growth and we had to been struggling to find various ways of trying to overcome that one of them was the use of heuristic functions.

But Dendral took this to one step further and it said that you can use specialized knowledge to solve problem. And this specialized knowledge that we are talking about is something that in the community with that was working on expert systems said that we will extract from human experts.

So, the idea was that you that you talk to the experts and there used to be protocols for extracting knowledge and extract that knowledge and put it in the form of some standardized representation typically rules we will look at rule based systems a little bit later.

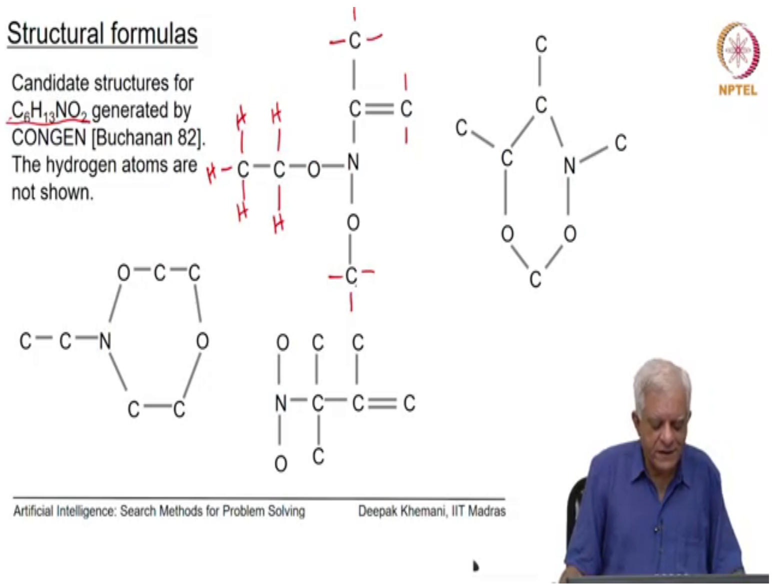
And then use some general purpose algorithms to work with that knowledge essentially. So, that was the notion of specialized knowledge we will come back to that later in this course. The task that Dendral was working was to assist chemists in the task of determining the structure of a chemical compound. Now, if you have studied chemistry, you would know that you have molecular formula, then you have structural formula.

Structural formulas tell you how the atoms in that particular compound are arranged. Now given one molecular formula there may be many different structural formulas associated with it and that would result in different materials and for example, carbon is a classic example it can be coal or it can be a diamond, but it is all carbon arranged in different ways.

So, the problem with the chemists are facing was that the number of candidate structures can be very large running into hundreds of thousands or even more. And it was a painstaking task for them to propose relevant structures and verify and verification was done by taking the spectrogram of that structure and testing whether it is really matching with the spectrogram of the real material that you are working with.

So, anyway the problem was that the number of structures that they wanted to explore would be very large. And this was where Dendral came in it led to a program called CONGEN which stands for CONstrained GENerator that allows the chemist to constrain the generation of candidate structures; in such a way that they had to inspect only a few towards the end essentially.

(Refer Slide Time: 20:41)

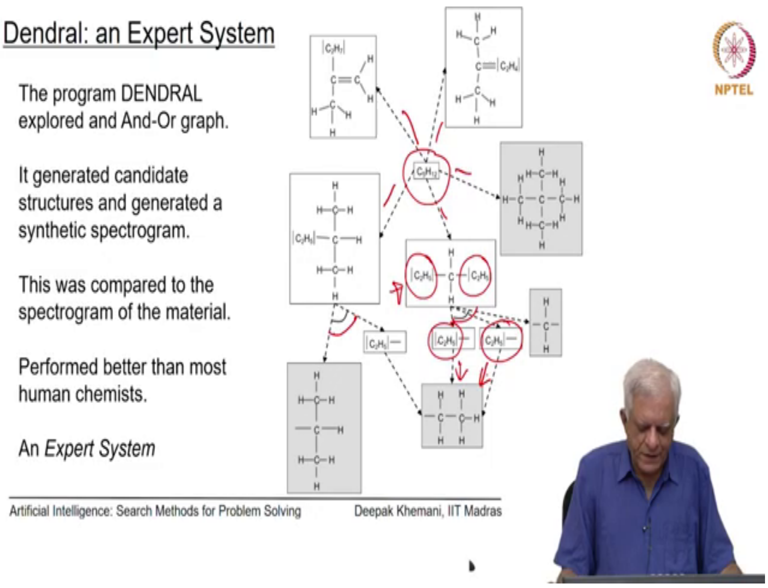


So, here is an example of what you mean by structural formula. So, the compound here is as you can see C 6 H 13 NO 2 that is a molecular formula for the compound. But those atoms the 6 carbon atom, the 13 hydrogen atoms, 1 nitrogen atom and 2 oxygen atoms can be explode can be arranged in different ways essentially.

So, in this diagram that I have drawn here which has been taken from this paper were Buchanan we are not drawn the hydrogen atoms. But you can imagine that given that the valency of carbon is 4 they would be 1 hydrogen atom here, another here, another here, one here, one here and so on essentially.

So, 3 here and 2 here and 3 here essentially. So, I am also not drawing the entire set, but you can imagine the that would be the final this thing, but since hydrogen atoms can; obviously, be filled in in the diagram we have not shown there. So, which of these is really the material that we are looking at? That is whether task that Dendral was helping the chemists to solve essentially.

(Refer Slide Time: 21:57)



Now, this kind of depicts the kind of search space that DENDRAL was exploring. So, here we have drawn it for this modern formula which is C 5 H 12 and you can see that there are in this diagram there are 5 different choices of how to try and explore different formulas.

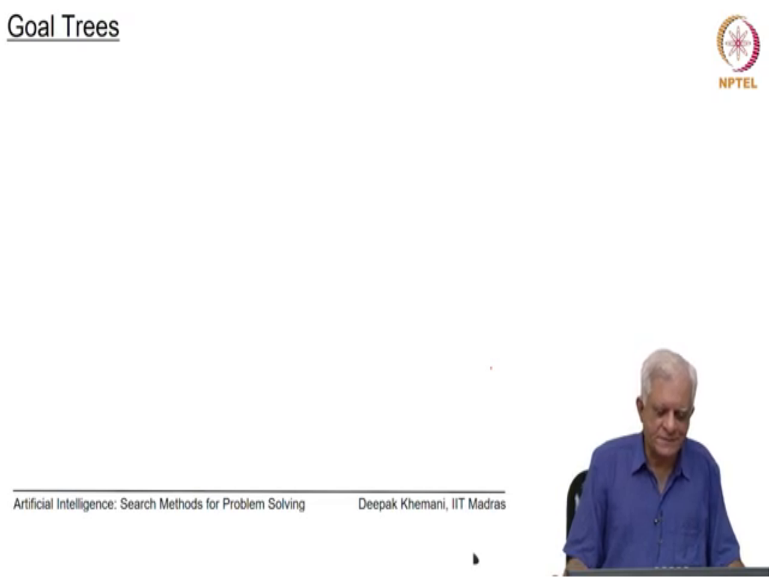
So some of those are or choices like the one in the center, but some of them are and choices like for example, here or for example, here. And so this and choice says that if you want to look at this particular compound then you have to solve for C 2 H 5, you have to solve for C 2

H 5 which is shown here C 2 H 5 here and C 2 H 5 here. And both have the same solution and that is why as I said some time ago there is a search space can be a graph as opposed to a tree does not have to be a tree.

But the ones shaded here are completely solved, which means there is nothing unknown about the structure and those could be candidate solutions that DENDRAL would explore. So, the program DENDRAL essentially explode And-Or graph these also called AO graphs and it generated candidate structures and then generated a synthetic spectrogram for each structure.

And compare the synthetics spectrogram with the real spectrogram of the material and based on that it would decide whether it was the good hypothesis or not. It turned out that it performed better than most trained chemists and it is an example of what was then called an expert system essentially. So, these are two examples which are there to motivate us to study how to explore And-Or graphs.

(Refer Slide Time: 23:55)



And we will do that in the next session where we will look at this idea of Goal Tress. So, meet you then.