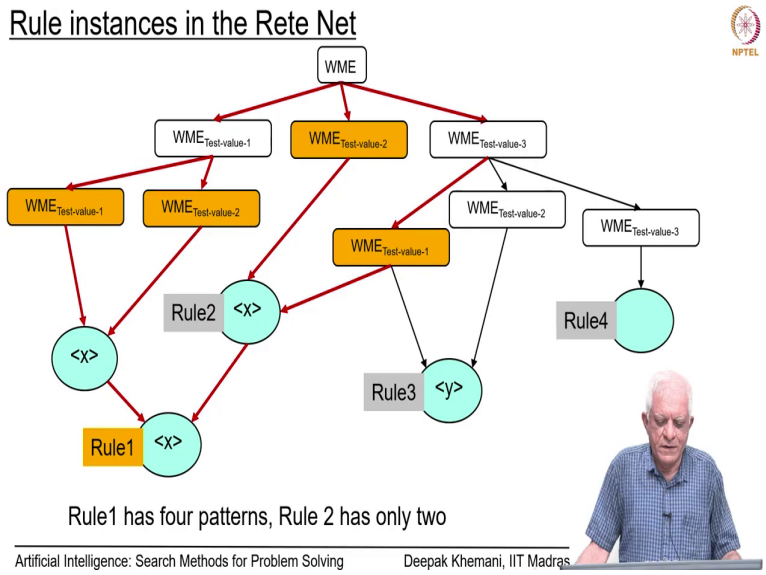
Artificial Intelligence: Search Methods for Problem Solving
Prof. Deepak Khemani
Department of Computer Science and Engineering
Indian Institute of Technology, Madras

Chapter – 06
A First Course in Artificial Intelligence
Lecture – 78
Rule Based Expert Systems
Rete Algorithm: Optimizing the Match

(Refer Slide Time: 00:14)

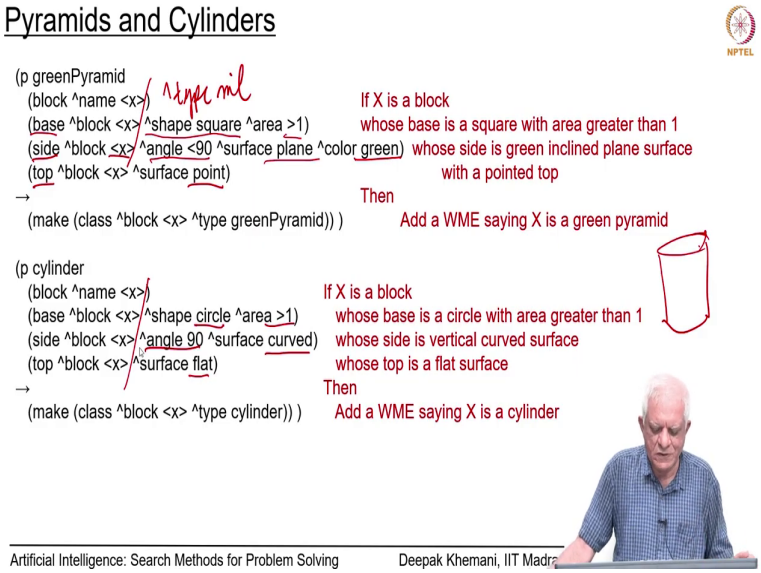


So, we are back again and we have been looking at a Rete network. And at the end of the last session, we saw this example network, which is a network for four rules that we have not written; but you can imagine by looking at the diagram as to what those four rules would be. So, rule 4 for example, is a rule which has only one pattern, which it needs for matching; and

therefore it the beta node has only one pattern and if that one pattern exist, the rule will be ready to fire.

Rule 1 is the other extreme which needs four patterns as depicted by the four parts that are converging on to the node which we have added the label of rule 1 essentially. Let us look at some simple rules and what the network associated with them would look like?

(Refer Slide Time: 01:13)



So, here is a small set of rules which kind of categorizes shapes into either pyramids or cylinders or wands and things like that. And we assume that the shapes are described by different aspects of their description, each specified by some pattern. So, let us start with something called a green pyramid. So, when do we say that some object is a green pyramid? This is the rule that we would like to write; we say that if X is a block, so the left hand.

So, the rule is on the left and the English description is on the right here in the red. So, if we have a block and let us call it X; X is a variable remember, whose base is a square with area greater than 1 and all that is described by attributes and values here, whose base has a shape square and the area is more than 1 essentially.

Whose side is a given, is a green inclined plane surface. So, we are talking about the side of block X; again remember x is a variable and we are talking about the same block all the time. The side is at an angle which is not vertical less than 90; it is a planar surface, which means it is not curved and it has a color which is green essentially.

So, a block whose base is a square with an area greater than 1, whose side is green and inclined and it is a plane surface and whose top is a point essentially or it is a block with a pointed top. So, the top has a shape of being a point essentially. If we have such a block, which satisfies these three conditions; then we can say, label it by saying that it is a green pyramid essentially.

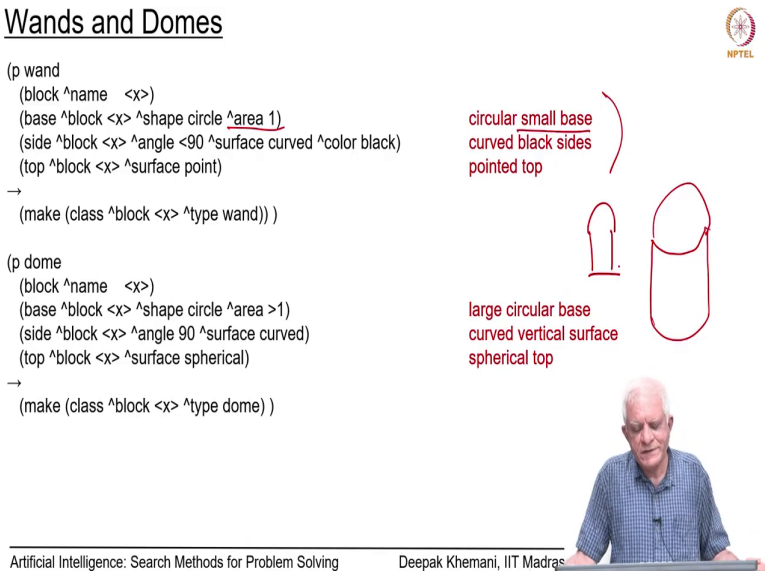
So, in this example we are saying, make a new memory element or add a new working memory element; but we could have modified the first working memory element also, if we had defined an attribute called type as part of that. So, that could easily have been done, but this is just for illustration. So, here you could have a type nil to start with; but later on you can change it to green pyramid and then you would instead of make, you would use modify.

But we are interested in the left hand side, not so much in the right hand side in the rete net. Rete net is concerned largely with optimizing the match part of the algorithm. Here is another rule which says, which defines a cylinder; it says that if X is a block, whose base is a circle with area greater than 1.

So, it is a circle with area greater than 1; whose side is vertical, but curved. So, the angle is 90, but the surface is curved, and whose top is a flat surface, as you can imagine a cylinder this thing right. How do the cylinder look? Like this essentially.

Then we are saying that this particular block; again we are referring to each block by its name; in this rule it says that, whatever it is name is. But whatever identifier we are using or name if you want to call, it must match in all the four patterns essentially. And we have seen that, what that is what happens in the beta nodes in the network; that the X value in different patterns must have the same value essentially.

(Refer Slide Time: 05:21)



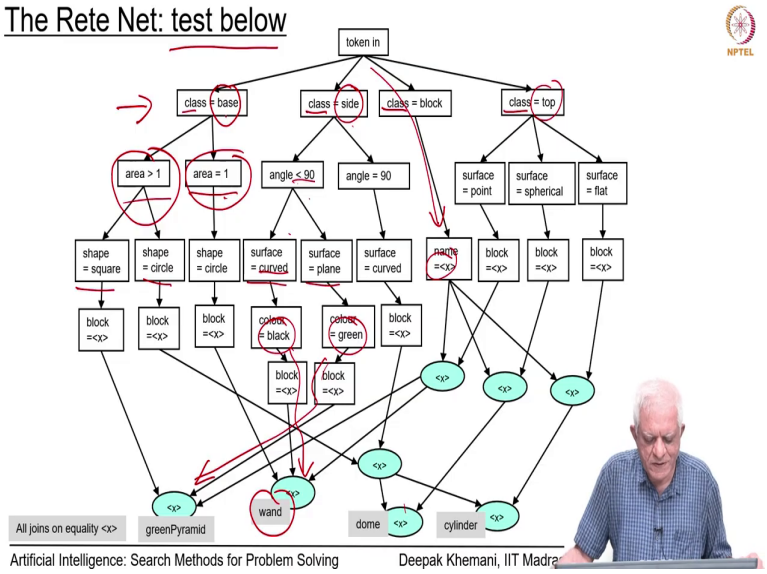
So, there are two more rules here for, about 10 years ago wands were very popular amongst children and I imagine some of the audience here would be familiar with what we are talking about. So, what is a wand? A wand is block with a circular small base.

So, as described here with curved black sides; we are assuming that wands are black and has a pointed top. A small base is depicted by saying, area is 1 essentially. So, it is a very, it is like a stick, except that it as a pointed top essentially, black stick I think.

And for the sake of illustration we have something called the dome, whose base is a circle, whose balls are vertical and curved and whose surface is spherical. So, you can imagine something like this; then you have this and then you have something like this.

You can imagine this, I am not sure whether they have drawn it well enough; but the side view would look like this essentially. It is a dome; it has got vertical walls, flat base and a spherical top essentially.

(Refer Slide Time: 06:52)



Supposing we had these four rules and we construct a rete network for them; how would it look? So, here is a rete network that we have. You can see that, the green pyramid rule has three conditions and we will see this highlighted shortly.

But, let us just focus on the on the network here; remember that the way that the algorithm operates is that, we put in positive tokens or negative tokens as the case may be. In our case may be they are just positive tokens here; because there are no delete actions in our actions here. So, we put the token and it travels down the network.

What is the first test it does? The first level test it does is on the class name. So, the class is either base or side or block or top essentially. You can see that there is a block part is not doing much except that providing a name; that we are talking about this particular block essentially.

And the other three branches flowing down are talking about three different properties of that particular block essentially; one is the fact as what is the base look like, other says what are the side look like and other says what are the top look like essentially.

Then if you remember, for base we had two kinds of blocks, either the area was greater than 1 or the area was 1. And then the shape was a square or a circle as the case may be; likewise for side, we said that it is either a curved surface or a plane surface. And there was an additional form of color for the sides which was there in the green pyramid rule and in the wand rule; in the wand rule the color was black, in the green pyramid the colour was green.

So, you can see that this token which is going to come down here is going to go here and the token if it is satisfies this facts that, the side is angle less than 90; that means it is inclined, it is curved and its color is black, then it will flow down here which is the wand rule essentially. So, you can imagine; this is the network that we construct for the four rules that we had for the green pyramid, the wand, the dome and the cylinder. And this is simply a representation of the rules; we have not talked about data so far.

Given those four rules, if you were to convert this into data; then this is a kind of network that we can construct. This is not the unique network; you could have done for example, the shape test before the area test.

So, we in the case of a base, we have done area test first and the shape test later; but that is something that you have to somehow figure out what is the right order of doing the test. So, for the base, we are doing area first, then shape; for the side, we are doing angle first and then whether a surface is plane or curved and then if there is a color involved and things like that.

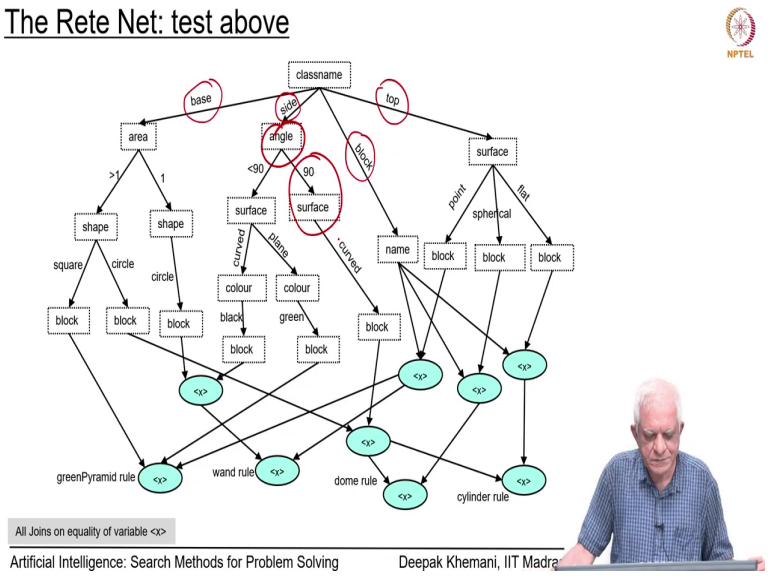
So, the order of test can be anything, you know and all of them will be equivalent; it is just that some networks would be larger than other networks essentially. And the same test may be repeated in more than one place essentially.

For example, this test of area greater than 1 is done only once here; and whether it is for the green pyramid, where it is a square or whether it is for the dome which is a, whose base is a circle. In both cases all we are asking is the area 1 or is the area more than 1 and that test in the way that we have ordered this test is done only once.

If you are tested first shape first and area next; then this area test would have been there along both the branches where it would have succeeded essentially. So, that is the kind of optimization we want to talk about.

Again this is the representation as I said concise representation, where the test is shown and the node which is below; the node is testing saying that the class name is base, so get that token here; this is saying the area is equal to 1, so get that token from there and so on. We have not drawn the tests separately and we implement of course we have to do that.

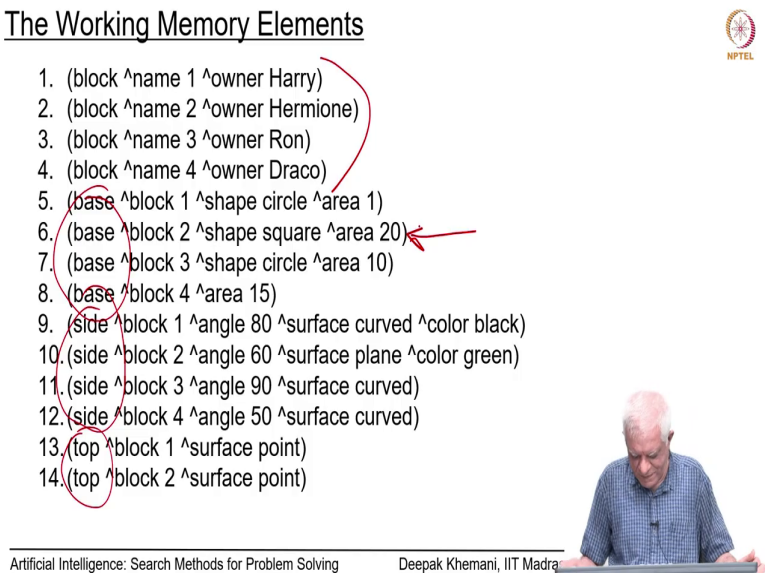
(Refer Slide Time: 11:38)



This is the alternative representation, where the test is done on the parent node; the test is named on the parent node and the value of the test is mentioned in the edges that flow, ok. It is a matter of preference based diagram you like better, but these are just compact ways of representing the network. So, this network is the same network as the previous network, except that the edges have labels here which are the values of the test; in this network, the values of a test are in the target node or the child node itself essentially.

The test as well as its value is given in the child node. In this network the test is on the top for example, angle. What is the angle is being tested here? And if it says 90; then it is labeled on the edge and then it comes down essentially, but the two networks are equivalent essentially.

(Refer Slide Time: 12:37)



So, let us say that we have some data that we want to talk about and we want to see how this data is consumed by the rete network. Remember that we said that, the idea is to feed in the changes into the working memory from the top; then the tokens go down, trickle down and then they are assimilated by the beta network and then some rules may triggers essentially.

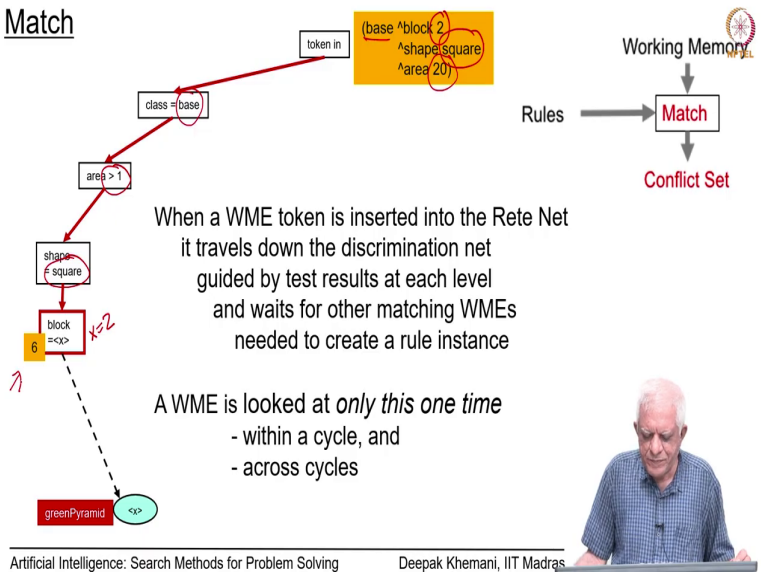
And let us say this is the data, initially of course everything is the change in the working memory; because before we add this data, the working memory is empty. So, we add this one by one and this is the order in which we add them.

There are four blocks that we are talking about, we have called them by block 1; we have given them you know names 1, 2, 3, 4 and we have also mentioned owners for the benefit of

the people who have you photo fans. Then we have described the base in some this thing; we have described the side in something, we have described top for some of them.

So, you see in this particular data working memory, there are only two descriptions of top. So, we expect that only at most rule 1 and 2 will fire or of the rule that matches this pointed tops, which is the wand and the pyramid will fire, and the others may not, I mean you can add more data. So, where does, how does the rete net handled this data is a question that we have want to ask?

(Refer Slide Time: 14:17)



So, remember that originally what was happening was it, you put the working memory and you got conflict set. But in the Rete Net, when a working memory token is inserted into the Rete Net; it travels down the discrimination network that is the alpha network, guided by the test at each level. We have seen that alpha nodes have test associated with them and

eventually waits for other matching working memory elements needed to create a rule instance.

So, in the alpha network, it goes to as far down as it can and then it is waiting to be summoned by a beta node essentially. When will a beta node summon it? If the beta node has two parents for example; if both its parents have got a working memory element, then beta node will say yes I have both the elements needed.

And so, conceptually the token will travel downwards; in practice if you think about the implementation, maybe you want to keep it that at the alpha level, that is a final point we will not discuss here.

So, as far as the match part goes, you put in a working memory element on the top and it travels down the alpha network and eventually the beta network, at least conceptually. So, let us illustrate this process; you put in a token and let us say this token is one of the tokens that we mentioned in the working memory last time, it does a tests at each level in the alpha network and travels down essentially.

So, let us look at this in a little bit detail. The first test is that the class name is base and you can verify that indeed the class name of the token is base. The second test says that the area should be greater than 1 and you can verify that yes the area is 20 which is greater than 1. The third test says, the shape must be square and you can verify that yes indeed the shape is square.

The fourth test simply says that we are calling it the name, but remember that at this point this has a value 2. So, where here we know x equal to 2, x is bound to 2 as we say sometimes. And this token which was token number 6 in our, in our working memory that we mentioned; remember that there was a whole set of tokens and this is token number 6 that we are talking about here, and it has gone down doing this different tests and come here.

Now, the important thing that is really the crux of this entire approach is this, so let me do this again. A working memory element is looked at only this one time essentially, whether it is within a cycle or whether it is across cycles.

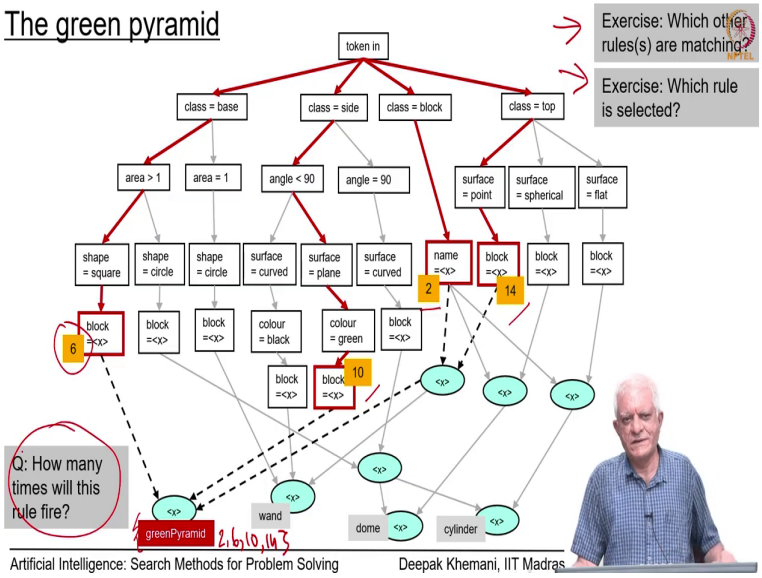
So, if you think carefully about what is happening here is that, when you add a working memory element to the network; it just travels down this path that we have drawn here and it is looked at only once, every test is an exactly once. Once it has reached this alpha node which has a label 6 here; it stays there waiting for other working memory elements to come which are matching.

In this particular case, we know that it will match the green pyramid rule; but it has to wait for three other tokens before that can happen. Whether it is for this cycle, which means match all the rules with all the data or whether it is across cycles; which means the next cycle again match all the rules with all the data, this particular working memory element will be inspected only once by the network. And you can imagine the kind of savings that this will do in the match process essentially, ok.

So, I think we should take a pause here or you should pause the lecture for a little while and ponder over this a little bit, and realize that this is a key to the Rete Network of the Rete algorithm being efficient essentially. And we know that Charles Forgy, in fact made improvements to this as well and Rete NT was apparently hundreds of time faster than the simple Rete Network that we are describing here essentially.

So, this is a key to that every token, every working memory element is looked at only once and that happens when it is dropped into the network and travels down certain path essentially. It is never matched again and again either within the cycle or between different cycles the only time it is looked at is when it is added to the working memory, ok. So, that is a key to the efficiency.

(Refer Slide Time: 20:06)



Let us look at this network a little bit more. In particular the green pyramid rule, we saw this token which is travelled down on this token number 6, we saw its journey in the last seen, you can imagine the other tokens also doing the same thing.

There are four tokens that are needed for the green pyramid rule; token number 2, in fact that is the first one that would be inserted into the system, then token number 6, then token number 10 and then token number 14 once. These four tokens are inserted into the system and they are all talking about the same particular block, where x is the value 2 in this case if I remember correctly.

Then the green pyramid rule will be activated; it will go into the conflict set and it will go into the conflict set by saying that this rule with the data 2, 6, 10 and 14, is (Refer Time: 21:09). Now, remember that there might be more than one green pyramid into the system, you know

there are some block number 15 may be a green pyramid rule; in which case another instance of the green pyramid rule come into play here.

So, what you are looking at here is a rule which the matching data here. So, there are some things I want you to think about a little bit; first one is here on the top right, which other rules are matching? You should look at the data or the working memory and look at the network or look at the rules equivalently; remember that the rules and network are the same and see which are the rules are matching given the data.

And then depending on what strategy that you are using for conflict resolution, which rule will be selected? So, first find out which are the rules which are matching; then find out which of them will be selected provided you choose a particular conflict resolution strategy essentially.

So, let us focus further movement on two strategies; one is specificity and the other is recency, ok. Another question I would like to ask you is that, supposing this green pyramid rule with this data 2, 6, 10 and 14 is selected; then how many times will this rule fire? Now, if you look at this rule; it simply says that add a new working memory element saying that, 2 is a green pyramid, there is no relation happening.

So, this same rule will continue to be matching in the conflict set; even after it is executed, it still matches. And it was selected the first time; is there any reason why it should not be selected the second time?

So, how many times will it fire? Think about this and think about the property of refractoriness that we spoke about, ok. So, will take a break here and we will come back and look at a couple of more examples. And will also pay some attention to how the different conflict resolution strategies can be implemented essentially?

So, see you after in the next session.