

**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 – 76**  
**Rule Based Expert Systems**  
**Business Rule Management Systems**

(Refer Slide Time: 00:14)



Towards an efficient implementation  
of  
a forward chaining rule based system



So, we are back looking at rule-based production systems. In the last few sessions, we saw this kind of a cognitive architecture which is used for problem solving in which the user or the programmer has to only specify the rules and the system which we call as an inference engine takes care of picking the rules, matching them with data and selecting one rule to

apply and keeps doing this in a cycle that we saw, the cycle if you remember was called match resolve and execute.

Now, it is turned out that if you look at this from very basic principles ah, it is the match part which is a most expensive part essentially because what do you have to do? You have to look at every rule and every rule may have some number of patterns so, you have to look at every pattern in every rule and match it with every working memory element which is a record in your working memory or which is a piece of data.

So, the number of patterns multiplied by the number of elements in the working memory is the time that you would spent if you did brute force match essentially. The other parts are relatively computationally cheaper. The resolved part basically takes the conflict set which is a set of matching rules along with data and has to pick one and the execute part is even less expensive that it has to take the selected rule and execute the actions on the right hand side essentially.

And we saw some aspects of this which talked about things like what are the conflict resolution strategies, we saw that refractoriness ah, lexical order, specificity, recency and mean sense analysis. We will look at them again in terms of implementation and we want to now focus on what is a good way of implementing rule based forward chaining rule based systems essentially.

(Refer Slide Time: 02:37)

## The Rete Network



The origin of the name *rete* is based on the English word, *riet* in middle English, itself derived from the Latin word *rete* meaning network, for "An anatomical mesh or network, as of veins, arteries, or nerves." -- [www.yourdictionary.com/rete](http://www.yourdictionary.com/rete)

### Definition of *rete*

1 : a network especially of blood vessels or nerves : [plexus](http://www.yourdictionary.com/plexus)

2 : an anatomical part resembling or including a network

<https://www.merriam-webster.com/dictionary/rete>



So, we will look at something called the rete network and the rete algorithm that is associated with that. Just to begin with a little bit of history about this is, the word rete itself comes from an old English word riet or riet which itself is derived from the Latin word which is actually curiously the same as this rete, but basically it means an anatomical mesh of network as of veins and arteries or nerves essentially.

If you look at the definition given by Merriam Webster dictionary, it is the same. A network especially of blood vessels or nerves essentially. An anatomical; anatomical parts assembling or including a network. We will see that the implementation of our rule based systems is going to follow this analogy quite closely. Just as our blood vessels, they carry blood from our heart to various parts of our body carrying oxygen and they come back to the heart to replenished again and then, go back into the circulation.

In a same way we will see that our implementation of the rule based production system which is going to be called the rete algorithm using this network called the rete network is going to be very similar, we will distribute information which is the new working memory elements that are being added into the system and we will collect them together to trigger rules that are going to be applicable.

(Refer Slide Time: 04:27)

### Business Rule Management Systems



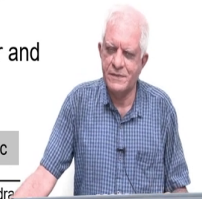
Charles L. Forgy developed the Rete algorithm in the late 70's but it took a little while for the algorithm to become widely popular, when Business Rules technology finally emerged.

- Most BRMS vendors developed their own algorithm that are known in the elite circles as XRete or Uni-Rete..
- Rete 2 is the version that Forgy developed which broke the legendary "Rete wall" which referred to the dramatic performance degradation when the number of objects in working memory increased.
- Rete 3 is the evolution of Rete 2 developed for RulesPower and subsequently integrated with Blaze Advisor.

From [this](#) blog (2011) by Carole-Ann Berlioz of Sparkling Logic

Artificial Intelligence: Search Methods for Problem Solving

Deepak Khemani, IIT Madras



So, a little bit of history as I said it was developed by Charles Forgy, he was a PhD student at CMU and it was part of his PhD thesis you know and you can look up a paper published by him it is easily available on the net.

But it became more popular when the emphasis shifted from expert systems to business rules essentially, I have mentioned this business rules earlier and the idea is that the business community at various kinds they are interested in specifying the rules of when what should be

done and things like that they are not so much concerned in writing the algorithms which will implement those rules essentially.

And when they caught on in the business community, then the rete net algorithm became very popular. So, most business rule management systems that is BRMS vendors develop their own algorithms which are known in elite circles as XRete or Uni-Rete.

So, there were all variations of rete. Rete 2 was developed by Forgy himself and it broke what is used to be called as a legendary rete wall which refer to the dramatic performance drop or degradation when the number of objects in the working memory increase.

The whole game that we are talking about efficiency is that if you have a large number of rules and if you have a large amount of data, how to do this match efficiently? Especially, given the fact that the cycle says that it is a cycle in which you do the match every time, then you do resolve, then you execute and you keep doing it again and again. So, if in every cycle, you have to do match once with huge amounts of data how does one manage that is a idea.

Rete 3 was an evolution of Rete 2 and developed by this company called Sparkling Logic which use in their system called blaze advisor and some of this history that I am talking about I got it from this blog by Carole-Ann Berlioz who was working in Sparkling Logic essentially.

(Refer Slide Time: 06:36)

### Rete-NT (a trade secret)



- The new OPSJ Rete-NT engine from Production Systems Technology uses the latest Rete incarnation from Charles Forgy.
- The Rete-NT engine works with any Rete-based BRMS, including those from IBM, Oracle, Fair Isaac, Red Hat, and Pegasystems.
- It is available to the vendors (licensed at \$5,000 per CPU) or can be used directly by PST clients with the OPSJ rule syntax.
- The new generation of his algorithm called RETE-NT is much faster than Rete3.
- How did he do it? Well, he would not say ...

Trade secret...!

From [this](#) blog (2011) by Carole-Ann Berlioz of Sparkling Logic

Artificial Intelligence: Search Methods for Problem Solving

Deepak Khemani, IIT Madras



Now, Rete net Rete--NT in those days NT was a popular trademark extension you might say. At one point, there was a operating systems called windows NT which came in the 80's as well anyway. So, this Rete-NT was the new OPSJ Rete-NT engine from production systems technology which is the company uses the latest incarnation from Charles Forgy.

So, Charles Forgy did the they did the advancements in his this thing and he started this company and it became a big hit and all that. The engine works with rete based some people call it rete, some people call it rete I prefer to call it rete so, rete base business rule management systems including those with IBM, Oracle and so on as you can see it was a very popular technology. They made it available to people licensed at 5000 dollar per CPU or it could be used directly in their production systems technology applications.

The new generation of this algorithm which is called Rete-NT is much faster than Rete3 essentially. The from the academic communities point of view, unfortunately we do not know how why it is faster and because it had become a trade secret by then and the company benefited directly from keeping it secret, it has not come into public knowledge, but it is a good problem that you can worry about and see how can we make the algorithm that we are going to discuss now a little bit more efficient essentially.

(Refer Slide Time: 08:27)

### Faster Processing with the Rete Net



The best usage of the Rete network I have seen in a business environment was likely Alarm Correlation and Monitoring.

- ... a "stateful" kind of execution where alarms are received over time as they occur.
- ... the actual Telecom network is made of thousands of pieces of equipment to keep in "mind" while processing the alarms, there is no wonder that Rete outperforms brute force.
- When one alarm is raised on one Router, the Rete network does not have to reprocess all past events to realize that we reached the threshold of 5 major alerts on the same piece of equipment and trigger the proper treatment, eventually providing a probable diagnostic.
- Hours of processing time turned into seconds in Network Management Systems. Fabulous.

From [this](#) blog (2011) by Carole-Ann Berlioz of Sparkling Logic



So, what are the benefits that people got from using the rete algorithm and the Rete-Net? Again described by Carole-Ann Berlioz, the best usage of rete network I have seen she says is in a business environment which is to deal with alarms which come out of systems correlation and monitoring essentially and so, she says that it is a stateful kind of execution where alarms are received over time.

So, just imagine that you are monitoring some large network of things in this case, they were monitoring network; networks or network management systems and at some point, some device somewhere would raise an alarm or would feel or something like that so, the whole question was when to raise what kind of alarms essentially.

So, the actual telecom network is made up of thousands of pieces of equipment which have to be kept in mind so, to speak while processing the alarms essentially, there is no wonder that Rete outperforms brute force. You cannot keep cycling through every piece of equipment or every piece of data that comes out of it essentially.

So, when one alarm is raised on one router, the Rete network does not have to reprocess all the past events to realize that we have reached the threshold of 5 major alerts on the same piece of equipment and trigger the proper treatment essentially and eventually providing a probable diagnostic.

So, you can see that it is like pieces of data cropping up somewhere over your large network and you have to make sense of it essentially and you do not want to every time a new message comes you do not want to look at all the older messages again and again.

And she claims that hours of processing time turned into seconds in network management systems essentially. So, you can see that it is clearly something which is worthy of attention of people who want to work in developing software which can be use a situations like this.

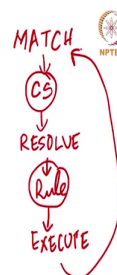


(Refer Slide Time: 10:43)

### Intra-cycle savings

```
(p grade-AI-C
(student ^name <n> ^rollNo <r> ^age <a> ^year <y>)
-> (course ^student <r> ^subject AI ^marks {<m> >59 <= 70} ^grade nil)
-> (Modify 2 ^grade C)

(p grade-ML-C
(student ^name <n> ^rollNo <r> ^age <a> ^year <y>)
-> (course ^student <r> ^subject ML ^marks {<m> >49 <= 60} ^grade nil)
-> (Modify 2 ^grade C)
```



So, let us now first turn our attention to what are the savings that we can do essentially. So, just to again recap, let me draw it here our cycle is simple you do match, you produce the conflict set so, let me put this in circle and then you do resolve, you produce a rule and then you do execute and the output of this execute basically goes back into match essentially.

So, this is a cycle that we are talking about. You now you keep going through the cycle, in every cycle, you will match all the rules with all the data, pick one rule, execute it, make the changes that you want to make in the working memory and go back and do match again.

So, there are two kinds of savings that we can think of doing here one is called savings within the cycle itself and one which has we will call as intra cycle savings and the second is savings

across cycles which means from one cycle to next what is it that we can carry forward so that we have to do less amount of processing.

So, let us first start with intra cycle savings and we will go back to our simple examples. Here is a small example which is meant for grading. So, supposing faculty member is using some kind of a software for grading and the software is a rule-based software, then you may have rules like this.

So, what is this rule say the first pattern says that if there is a student whose name is  $n$ , whose roll number is  $r$ , no remember that  $n$  and  $r$  are variables here essentially because they come in angular brackets. So, these are applicable to all students and all roll numbers and everything.

So, supposing there is a student essentially, the second one says that if that student so, remember that if I use the same variable name in the same rule, then it must match with the same constant essentially. So, if that student that we are talking about in this and in the subject AI, he is got marks which are more than 59, remember that this is the marks he is got and this is more than 59 and this is less than equal to 70.

So, essentially between 60 and 70, if he has got marks and if he does not have a grade, then you assign a grade C to that student. How do we do that? We do that by modifying this element and modifying the grade C which was originally nil, we change it to C essentially. So, this is one rule which looks at the AI course and looks at the C grade. So, this is one rule for one grade in this thing and likewise you can imagine there are other rules for other grades and other courses as well.

Just to illustrate the idea, we will see shortly that that we do not have to be so specific while specifying the rules, but this is just to illustrate the idea of how this whole thing happens ok. So, I may need this piece of the board so, let me rub that off here. Hopefully, you remember what is the cycle that we are talking about.

Here is another; another course and in this case, now the course is different course let us say its machine learning and its cut offs are 49 and 60 which are different from this thing and this is also a rule for C gradient machine learning course.

So, you can imagine that for every course, for every grade you could write such a rule and then the whole system could go inside and you would enter all the data of all the students as to in which course how much marks he or she has got and the system would do the grading essentially.

(Refer Slide Time: 15:22)

**Intra-cycle savings**

```
(p grade-AI-C
(student ^name <n> ^rollNo <= > ^age <a> ^year <y>)
→ (course ^student <r> ^subject AI ^marks {<m> >59 <= 70} ^grade nil)
→ (Modify 2 ^grade C))


(p grade-ML-C
(student ^name <n> ^rollNo <r> ^age <a> ^year <y>)
→ (course ^student <r> ^subject ML ^marks {<m> >49 <= 60} ^grade nil)
→ (Modify 2 ^grade C))
```

WMEs

- 1
- 2
- 3
- 4
- 5
- 6

CS = { ..., <grade-AI-C,1,3>, <grade-AI-C,2,5>, <grade-ML-C,1,4>, ... }

1. (student ^name Sneha ^rollNo 201835 ^age 20 ^year 3) ... in 5 courses
2. (student ^name Sunil ^rollNo 201706 ^age 21 ^year 4) ... in 3 courses
3. (course ^student 201835 ^subject AI ^marks 66 ^grade nil)
4. (course ^student 201835 ^subject ML ^marks 50 ^grade nil)
5. (course ^student 201706 ^subject AI ^marks 62 ^grade nil)
6. (course ^student 201706 ^subject ML ^marks 69 ^grade nil)



Artificial Intelligence: Search Methods for Problem Solving
Deepak Khemani, IIT Madras

So, we just take a tiny example here. In our example, we are just looking at a subset of the data and there are 6 working memory elements in this. Remember these are timestamps 1, 2,

3, 4, 5, 6 and there are two students we are talking about Sneha and Sunil and we are talking about two both these courses because we are talking about both these rules.

So, you can see that let Sneha has got in AI, she is got 66 marks in ML she is got 50 marks. Now, you can see that in that both these grades kind of match, both these marks match these two rules that we have stated here. So, you expect that she would get C in both the courses and Sunil has got 62 in AI and 69 in ML essentially. So, the AI course he should get a C grade.

So, in all these data, we expect to see three instances in the conflict set, but of course, Sunil will get some other grade probably B grade or something because 69 marks is more than 60 so, maybe that is in the B range, but that rule we have not shown here essentially. What we want to show is the intra cycle savings that you can do if you implement the system properly.

So, look at the first rule and with the first pieces of data. The 1st working memory element is that Sneha is a student, the 3rd working memory element says that in AI, she got 66 marks now, because 66 is between 59 and 70 it matches the 2nd pattern and the 1st pattern matches this and that will go into the conflict site essentially.

But it so, happens that the 2nd rule also matches for Sneha. Sneha has got as you can see 62 marks no sorry 50 marks in ML which is between 49 and 60 so, both these rules will match. In fact, for every student, for every course, some rule will match because every course that they are doing they would get one grade somewhere and so, there would be some grade some rule matching that one grade.

Now, what we are talking about intra cycle savings here is that the first pattern in both these rules is same and in fact, will be same in all the rules. So, that if we match rule 1, then we match rule 2, then we match rule 3 and so on and so forth, we do not want to match the 1st pattern again and again and again ok. Having matched it once, we know that there is a student called Sneha we are talking about, we should be able to cut down on the matches for the other rules.

So, the question is can we for all the rules in which that pattern occurs do the match only once that would be the ideal situation. It is not just that the complete pattern should be repeated. Even, if part of the pattern is repeated, we would still like to share the work done in matching that particular pattern. So, if the first three attributes of a pattern are the same, then if we can match those first three attribute only once, then we are doing some savings.

Now, notice that I use the word first three whereas, I had said earlier that the order in which you may mention the attributes do not matter. In fact, they do not matter, but this ordering that I am talking about the first three will come through the implementation that we are going to shortly see essentially. So, the basic idea of intra cycle savings is that you if you can share the work done for matching different patterns, then go ahead and do so that is the goal of intra cycle savings.

So, in this particular example as I said there are three rules which will match the other rule that matches shown in blue for Sunil is saying that Sunil will also get a C grade in AI because that like working memory number 5 says that he has got 62 marks and that is in the range of 59 and 70. The last pattern will match something, but it will not match a rule, it will not match one of these two rules it will match a rule that we have not mentioned here essentially.

So, as far as the data that we are saying on this screen, the conflict set will get three rules one says that Sneha will get C in AI that is the first one here. The second one says that she will get a C in ML and the third rule sorry the second one says that Sunil will get a C in AI and the third one says that Sneha will get a C in ML essentially. So, that is one side of the story.

(Refer Slide Time: 20:39)

### In practice only one grading rule



```
(p grade-assignment
(cutoffs ^subject <s> ^low <low> ^high <h> ^grade <g>)
(student ^name <n> ^rollNo <r> ^age <a> ^year <y>)
(course ^student <r> ^subject <s> ^marks > <low> <= <h> ^grade nil)
→
(Modify 3 ^grade <g> )
```

1. (student ^name Sneha ^rollNo 201835 ^age 20 ^year 3)
2. (student ^name Sunil ^rollNo 201706 ^age 21 ^year 4)
3. •
4. •
5. (course ^student 301835 ^subject AI ^marks 66 ^grade nil)
6. (course ^student 301835 ^subject ML ^marks 50 ^grade nil)
7. (course ^student 201706 ^subject AI ^marks 62 ^grade nil)
8. (course ^student 201706 ^subject ML ^marks 69 ^grade nil)
9. •
10. •
11. (cutoffs ^subject AI ^low 59 ^high 70 ^grade C) ←
12. •
13. •



So, as I said this was just for illustration purposes that I use two separate rules to assign C grade for each course essentially. In practice one does not need to do that essentially, you can just write one rule and that will take care of all the courses and all the grades essentially, but of course, you have to have additional data here.

So, if you look at this rule and compare this rule has got three patterns. The rule that we saw earlier, the two rules that we saw had two patterns one pattern said that this is a student and the second pattern says that this is the range in which C grade is given.

Now, we have three patterns and the first pattern shown here basically says that in some course *s* which is now a variable here, earlier we had AI separate and ML separate, there is some low cut off, there is some high cut off associated with some grade *g* essentially and whichever student we are talking about if the low matches, what is there in the first pattern

and if the high matches, what is there in the first pattern, then you should give the grade which is specified in the first pattern.

So, there is an association between the first pattern and the third pattern essentially. The first pattern specifies the cut offs and the grade, and the third pattern says that this student is within that range essentially. If you implement a system like this, then one rule will be enough to do all the work that you want to do.

So, there were two students we talked about Sneha and Sunil, there may be others and then, we talked about four courses here, but there will be other courses for example, Sunil would have got a B grade in ML and other students would have got something.

In addition, what the instructor or the user needs to do is to specify the cut offs for every grade that she is giving in her course. So, essentially somehow these kind of working memory elements need to be added to the working memory. So, for example, if I were teaching AI, I would say for C grade my range is between 59 and 70, for B grade my range is so on and so forth and so, every; every instructor would specify this range.

You can imagine that you can build a software system whereby the instructor says this is my course, these are my cutoffs and here is I am uploading the list of marks and do the grading for me automatically such systems are common in fact, we use one system like this in IIT Madras and the whole thing can work in the rule-based fashion essentially. So, there is compactness to be achieved if you write the rules in a more general enough fashion essentially ok.

(Refer Slide Time: 24:02)

### Inter cycle savings



Let us say there are 300 students  
who between them have taken 1200 courses

Initially there are 1200  
instances of grade-assignment rule in the conflict set

One of these instances is selected, and fires

The remaining 1199 are **still** matching

**why** compute their match again?!



Now, the other part we said was inter cycle savings. Between cycles can we do some saving of work essentially and what are what is it that we are referring to? We are referring to the match part. We have said that match takes a predominantly the most amount of time in the cycle of match resolve and execute essentially.

So, let us look at this simple argument. Supposing there are 300 students in your institute and between them, they have taken 1200 courses. Now, so, let us say each student takes about 4 courses on the average. So, there are 1200 courses with 300 students which means that 1200 grades have to be assigned essentially and if you had this rule that we or the rules that we had talked spoken about there would be 1200 instances of grade assignment rule in the conflict set.

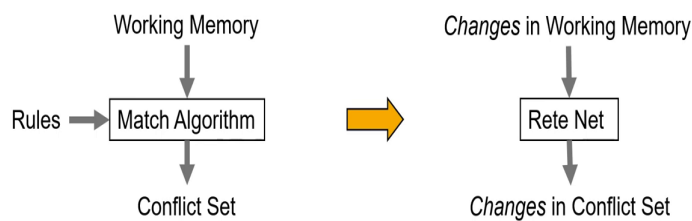


Depending on whatever strategy that you have implemented for resolving those conflict, one rule would be selected, and that rule would fire essentially. But the remaining 1199 rules are still matching nothing has changed about the other students marks or cut offs or anything like that the question is why should we do that match for the remaining students all over again and as you will see shortly.

The rete network is organization of data which allows to do both this intra cycle savings as well as inter cycle savings in one go. It is quite a neat algorithm and definitely worth studying carefully.

(Refer Slide Time: 26:01)

### No more brute force Match



So, what is it that we are going to do? We are going to replace the brute force match. What is the brute force match? It says that take every pattern in every rule and match it or compare it

with every working memory element in the working memory essentially. So, every pattern in every rule with every piece of data.

So, you can see the amount of work that is being done is quite huge instead we will replace it by a structure which is called the Rete network and what the Rete network will do is that instead of taking the working memory as input, it would take only changes in the working memory as input and instead of generating the conflict set as the output, it would only generate changes in the conflict set as the output essentially.

So, you can see that what is happening here or what is going to happen here is that when the execute part of a rule happens, it says add certain working memory elements, delete certain working memory elements, they are changes in the working memory so, those would be inserted into the Rete net.

The Rete net as we will see is going to be a compilation of the rules. So, on the left hand side of the figure, you have rules and working memory as input to the match algorithm, on the right-hand side there are no rules and that the reason is that, the rules have been compiled into the Rete network essentially. So, it only takes the changes in the working memory and changes produces changes in the complex settings essentially.

So, we will stop here. I hope you will think about what we want to do and try to imagine how you would do this, but we will take it up in the next session, next video essentially. So, see you after hopefully a short break I think.