

Artificial Intelligence
Prof. Anupam Basu
Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur
Lecture # 25
Rule Based Expert System

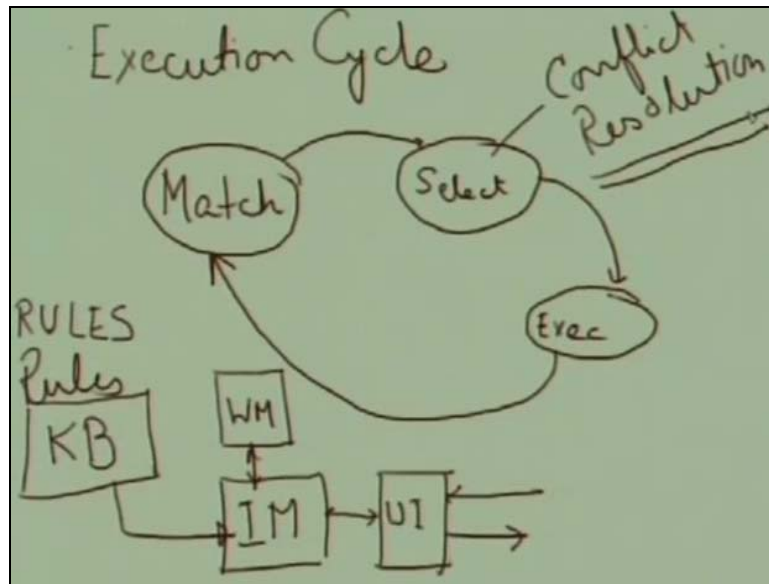
We will be discussing about Rule-Based Expert System and we will also mention some things about certainty factors. Although we will have separate lectures on certainty factors that demands little more discussion. While discussing about Rule Based Systems we had talked about expert systems. And if we quickly recapitulate what we learnt about Rule Based Systems then there are some salient points that we should remember.

First let us quickly have a look at what are the things we need to discuss about Rule Based Systems. First is the execution cycle. This is just for your recapitalization. This means that there are three phases; one is match, one is select and the other one is execute. And after that we come back to the match phase only. The architecture of a Rule Based System consists of a knowledge base which you can write in short as KB. There is an inference mechanism which we are writing as IM, inference mechanism or control structure and there is a working memory. In this working memory all the current facts are being stored.

There is another user interface as it should be for any system and the user interactions are through this user interface. And that is communicating through the inference mechanism and the inference machine reads from and writes to the working memory. The knowledge base consists of rules in case of Rule Based System. And it is needless to say at this point that rules are of the structure if antecedent then consequent. Now what is done in the match phase?

The inference mechanism looks at the data for the current problem from the working memory and tries to match them with the contents of the knowledge base to see which rules are enabled to be fired. There may be more than one rule which is enabled to be fired.

(Refer Slide Time: 04:18)

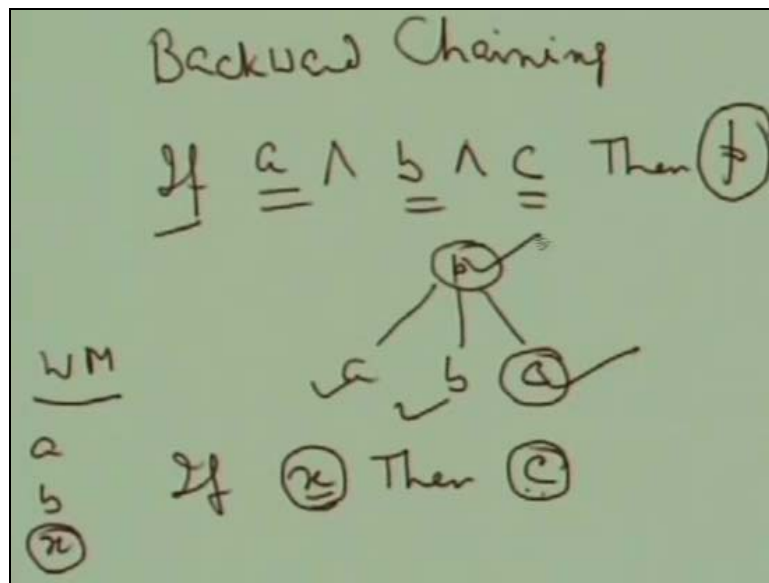


When is a rule enabled to be fired?

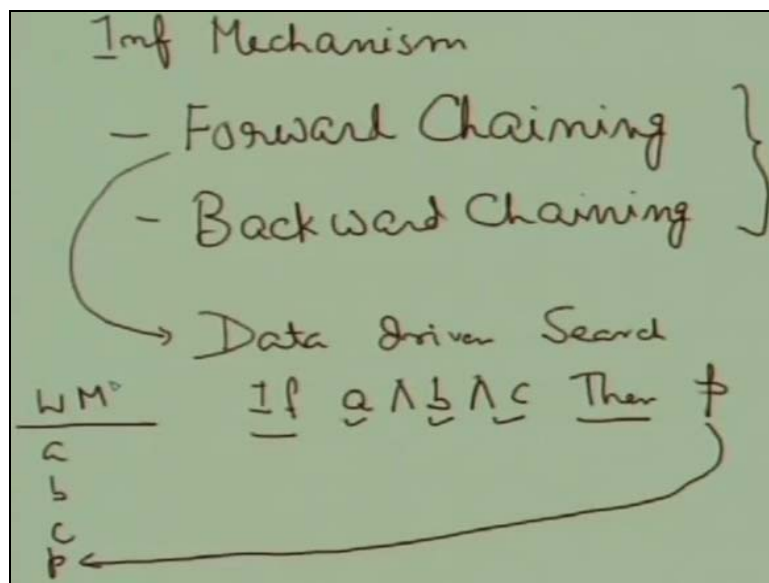
A rule is said to be enabled to be fired when the entire antecedents match or the consequents match depending on whether we are going to do forward chaining or backward chaining. If more than one rules match or applicable at a particular cycle then in this phase the select phase which is also known as the conflict resolution phase a subset of rules are selected maybe one or maybe two. And those rules which are selected are only fired and as this rules are fired by the inference machine new facts are generated which come to the working memory here. That will enable other rules to match and this cycle will go on until no further rule matches or the goal with which we started the system is met. These are the two points but besides this there are a couple of other points which we need to recapitulate. It is about the inference mechanism.

There are two types of reasoning we talked about; one is forward chaining and the other is backward chaining. In forward chaining which is also known as data driven search we have got a rule like this; If a and b and c then p. Now, given a rule like this and with whatever is there in the working memory say a b c we start matching the contents of the antecedents and if all of them match then we say this rule is enabled and if this rule is fired then the firing will produce this p as a new fact in the working memory and this can enable further search.

(Refer Slide Time: 07:55)



(Refer Slide Time: 8:50)



On the other hand, in the case of backward chaining we start with the goal not the data. Unlike the forward chaining in the backward chaining case we start with, for example we have got a rule like backward chaining, If we have got a rule like if a and b and c then p and there are other rules also and we find that I have to prove p. Then I start with p and that tells me I have to prove a b c. And maybe in the working memory a and b are true so these are solved but I have to find a rule which has got c in its consequent.

Suppose x is also there then c. Then after this since c is my goal then I start with c and see whether this is proved or not. For that I have to fire this rule c is true if x is true so in order to prove c you have to prove x but x is already proved so therefore c is proved therefore p is proved, that is backward chaining. Now all these things we have

discussed in our discussion about Rule Based System. Rule Based Systems have been applied in expert systems.

What are expert systems?

Expert systems are said to be AI's greatest commercial success. An expert system uses knowledge specific to a problem domain. This is a very important point that you should try to understand that it uses knowledge that is very narrow specific to a particular problem maybe fault diagnosis of a two wheeler or maybe fault diagnosis of a particular type of car. So the narrower you make the domain the possibility of a success of an expert system is more because what we try to do in an expert system is to capture the human expert's knowledge in the form of rules.

Human expert knowledge is a little complicated than can be really captured in the form of rules. So, if we can restrict our domain then probably we can acquire proper knowledge and encode it in the form of rules and then we will find that for small domain in order to capture all possible cases we can have a huge number of rules. But we are not really trying to match the human expertise or the human capability in any way in expert systems then the question can be asked then why at all we go for an expert system?

The reason is, human expertise is scarce it is costly for all the aspects of problems you will not find enough number of experts who can cater to the large number of queries that come up.

Think of specialist doctors. Now specialist doctors are really scarce in number when compared to the demand that we have from a large number of patients for example, really good expert cardiologists. Now all the patients really do not need that level of expert advice all the time. There are many cases which can be treated by less than a human expert. So, if it be possible to capture the expert's knowledge to some extent in such systems then obviously we can provide better service to the patients. Everybody need not line up in front of the doctors chamber, need not just wait for weeks and months to get an appointment. Maybe through some computer system they can be given routine advices wherever that is good.

Although expert systems aim or attempt to capture human expertise the gap will always be there. Nevertheless, for many applications for example diagnosing the situation under a mine when we have put in some sensors and we really want to interpret the data either I take out the data at an upper level and try to look at it but proper interpretation of data at runtime may not be feasible for an expert to do. Therefore an expert system can do that job quite well provided we keep the domain narrow.

(Refer Slide Time: 14:18)

Expert systems

- expert systems are AI's greatest commercial success
- an expert system uses knowledge specific to a problem domain to provide "expert quality" performance in that application area
 - DENDRAL (1967) determine molecular structure based on mass spectrograms
 - MYCIN (1976) diagnosis & therapy recommendation for blood diseases
 - PROSPECTOR (1978) mineral exploration (found a \$100M ore deposit)
 - XCON (1984) configure VAX and PDP-11 series computer systems (saved DEC \$70M per year)

What is an expert system?

Expert system is a system that uses knowledge specific to a problem domain to provide expert quality performance in that specific application area. There have been a number of successful expert systems.

Here we can see some of the names; Dendral which was developed in 1967 is an expert system that determines molecular structures based on mass spectrograms. Mycin which is a very popular expert system developed at Stanford University by doctor E H Shortliffe in 1976 as part of his PhD dissertation. Mycin works for the area diagnosis and therapy recommendation for blood diseases. And later on Mycin was done for different infectious diseases, bacteria, infections as well. Prospector which came very close to Mycin in 1978 is an expert system that was used for mineral exploration. That means it tried to estimate where a particular mineral is likely to be there. And it found about a \$100M ore deposit successfully.

Xcon is an expert system that was developed in 1984. But the predecessors of Xcon are known as R1. And that expert system was again developed at a University and later on it was taken up by the Digital E- Corporation to configure how to install them large VAX and P D P 11 series computer systems. Installation of such large systems requires a lot of expertise about the layout, the cabling and other conditions. But that can be done judiciously by Xcon. And this saved Digital Equipment Corporation DEC \$70M per year. That shows the commercial success. Now let us look at the common characteristics of expert systems.

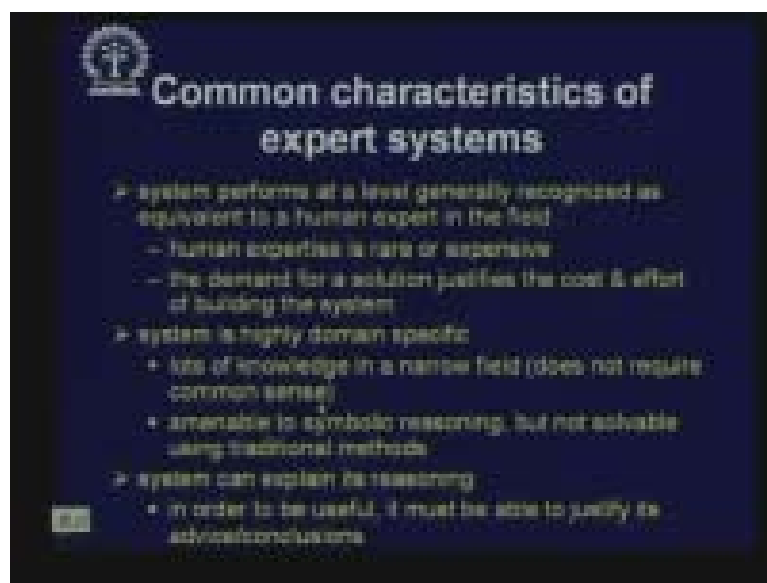
What are the common characteristics?

System performs at a level that is generally recognized as equivalent to a human expert in the field. It is better that you keep the domain narrow, keep the set of problems that an expert system can address limitedly rather than be too ambitious. But the success of an expert system would be measured by if I had employed a skilled person and expert there paying a huge salary he who would have done some particular work and for that work if the expert system can do and show a performance that is

close to this then the expert system is ok for that domain. The reasons as I said human expertise is rare or expensive. And there has to be a demand for the solution that justifies the cost and the effort of building the expert system.

Expert system building although we know the technology now the Rule Based System is one. But it takes a lot of effort, time, man power, programming time, testing time in order to make a real successful expert system and so there has to be a demand for it. The system is highly domain specific because even in a narrow field lots of knowledge is there and we have to capture that. And it should be narrow enough so that it does not depend so much on common sense. Now we have to run it using a machine so it has to be amenable to symbolic reasoning.

(Refer Slide Time: 19:19)



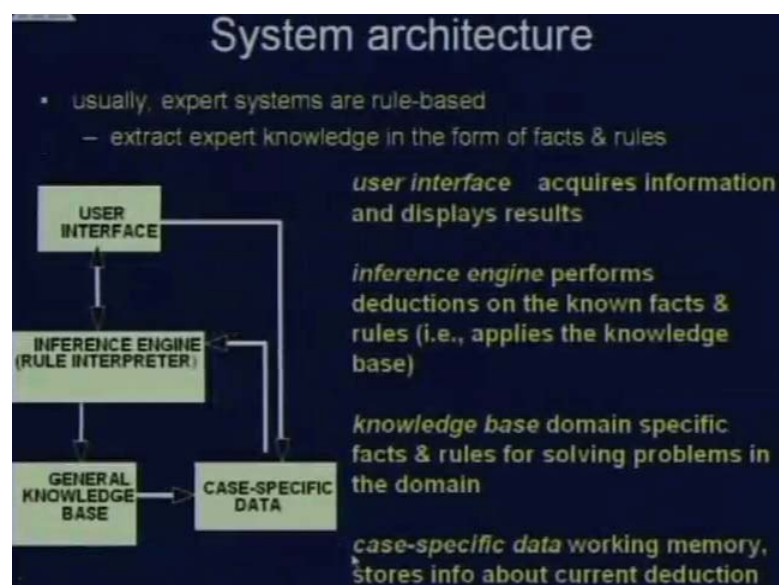
We should be able to reason using symbolic methods like predicate calculus, propositional calculus where we can represent the facts and rules in the form of symbols as we have seen earlier. But it should also have a challenge that the solution is not straight forward. You cannot employ an expert system to sort a file or sort an array or even do some pure image processing activities for which there are very good algorithms available. The methodology is absolutely known, you really do not need much of decision making expertise.

The rules in a Rule Based Expert System are, if the situation is this then this will provide those critical judgments in an expert system. If those critical judgments were not there in that case expert system is not required. Again if you recollect that match, select, execute cycle of a rule based system in the select phase we are doing conflict resolution that means we are selecting a subset of rules by some judgment by some heuristics. We may assume that if we select this rule that is bringing me closer to the goal so let me select this. It is not a straight forward algorithmic decision and so for all those cases expert systems or expert system approach can be used.

Presently there are systems which are a little hybrid in nature in the sense that they use some part of rule based expert system and some parts are solved algorithmically and these two together provide a very good result. System can explain its reasoning.

Think of a scenario that you have gone to an expert and the expert as told you that you have to undergo a surgery or you have to undergo a set of blood tests. You can ask the expert why you want me to undergo this blood test. The expert will be able to explain to you why or what is the reasoning that went behind him. Maybe that he is suspecting certain results, there are some possibilities in his mind and he wants to verify that or eliminate some of the possibilities and he will be able to explain that to you. Just as a human expert will be able to do that it is expected that an expert system will also provide that sort of explanation at least close to that sort of explanation. We should attempt that it should give similar explanations. Therefore, in order to be useful it must be able to justify its advice or conclusions.

(Refer Slide Time: 22:55)



The basic architecture that we had said about Rule Based System remains unaltered here. Usually expert systems are rule based although there are expert systems which have been built on frames semantic nets and other technologies but mostly Rule Based Systems have found its popularity.

What are these rules for?

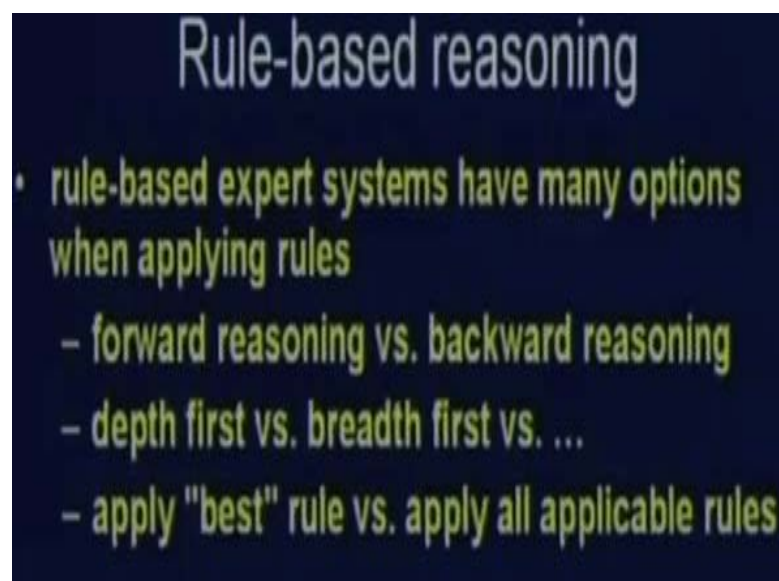
The rules are to extract the expert knowledge in the form of facts and rules and that is a real task. Extracting knowledge from experts is really important and critical and this process is known as knowledge acquisition. In order that an expert system becomes successful the knowledge as to be acquired properly and it is not a very straight forward case because no one is sure whether the human thinking moral is rule based or what sort of structure it maintains so there is a lot of debate about it. So often the experts talk from a level which may be intuition or which may be very deeply embedded in their knowledge and they are so habituated they can click upon a particular suggestion and they say that. But what is the reasoning process that went through to lead to this inference is often not very clear even to the experts themselves.

Therefore it is necessary that a knowledge engineer as we call it must talk to the experts, understand the domain and should be able to extract that knowledge and represent that knowledge in the form of rules so that it can be embedded in an expert system.

In the blocks there are:

The user interface, one is the inference engine, there is a general knowledge base and there is a case specific data. Now you can think of where is the working memory. The working memory is this for the time being. This is the rule interpreter, inference engine, inference machine, control engine or whatever you call which is the path and this is the knowledge base. And we have not deviated from the earlier architectures we showed except for this might be that the case specific data can be directly fed here but the interactions are the same that the inference machine will read the data, find out which rules are enabled, apply them and on application new data will be generated over here and that will come. So, user interface acquires information and displays results. Inference engine performs deductions on the known facts and rules that applies the knowledge base. And knowledge base stores the domain specific facts and rules for solving problems in the domain and case specific data working memory or working memory stores information about the current deduction. Therefore these are the four blocks.

(Refer Slide Time: 26:44)



Now, as we know rule based expert systems have many options when applying rules. At the beginning of this lecture I was mentioning about forward chaining and backward chaining which is also known as forward reasoning and backward reasoning. And in forward reasoning we start from the available facts and try to arrive at the goal. On the other hand, in case of backward reasoning we start with a conclusion and try to prove the conclusion by finding the sub goals. If x and y and z then p means that in order to prove p I have to prove x , I have to prove y and I have to prove z . So p is true if x is true, y is true and z is true. Might be x is already known to be true but for y and z I have to test them. So again I will have to find a rule whose

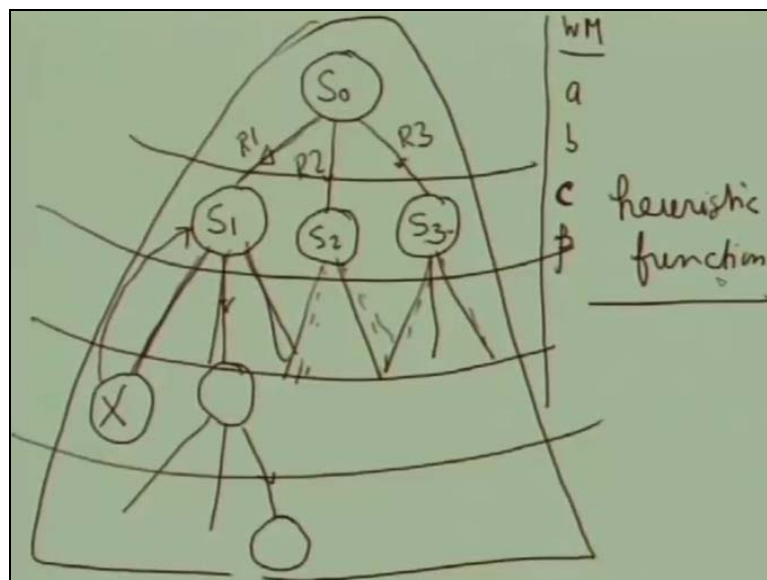
consequent has got y as I was explaining at the beginning of the lecture. Similarly, depth first, breadth first etc are again different ways in which this thing can be done.

What is a search space? For example, I am in a particular state say S_0 how is that state defined?

That state is defined by the state of the working memory that is some of the facts I know of a, b, c etc. Now, given this state S_0 where a, b and c are known I may have a possibility of different rules to be fired R_1, R_2, R_3 . And firing each of these rules will lead me to different states S_1, S_2, S_3 because each of these rules will derive new inferences might be p will be known to be true if I fire this. Now on this new state I will again have other rules to fire.

If I had selected this rule then probably I would have had some other different rules to fire which I have not traversed as yet. So after this I have got these three rules to fire, I fire this rule and I come to this state and that had three different paths and I select this state and come to another state etc. And while at every level and some paths I have left unexplored. So this entire phase is the possible search space in which the path towards the solution lies and my job is to find that out. And in order to search that particular path out of these three rules I first take this, then I take this and in this way I go on. And when I come to a dead end I have not found the solution I backtrack and follow the other unexplored paths. That is the depth first.

(Refer Slide Time: 31:08)



In the case of breadth first what would be the case?

At this level I will fire all these rules and come to this state. Again at this level I will fire all the possibilities and come to this state and I will try to explore the space in a breadth-wise manner. Obviously you know the difficulties of depth first that it will need a lot of nodes and lot of new children will be generated.

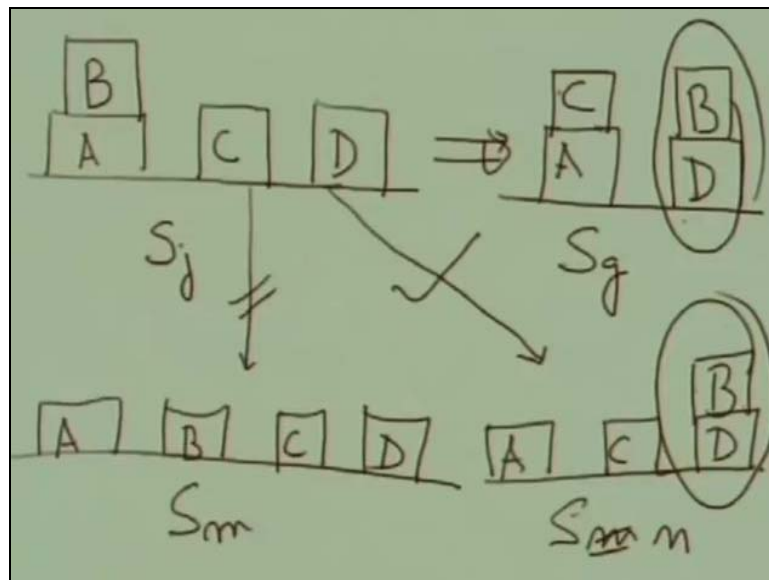
Another option is heuristic search **which has been discussed in this course**. Heuristics means with some partial information and knowledge. I would like to find a better way possibly an intelligent way to explore the path.

Now in that case what do you do?

In that case probably at this level I try to evaluate a heuristic function. Now what that heuristic function will be is very much dependent on my application and based on the heuristic function I can select a particular path. For example, I have A on B and C is here and D is here. It is some particular state say S_j this is the state of this and I want to come to another state that is my desired state which should be A and on that there should be C and D to go below B and now that is my objective. Now I have got different options to do that.

Suppose I want to come to this goal state S_g then one way I can try is I cannot really take C over here C on top of B. So one rule can be that you come to a state where A is here, B is here, C is here, D is here that is one possibility. And from there you take C above this and take B above this. Suppose if this be the case then I can straight away come to this where A will be here C is here and I directly put D on B. Now, suppose my rules allow me that I can directly come over here therefore this is another state say S_m and this is S_n . now which rule I should fire?

(Refer Slide Time: 35:44)



If I have got a distance function then this S_n is closer to this because this part of the problem is already solved in one move and this will require more number of moves. So, if that be my heuristic function I will apply this rule instead of this because that will reduce my distance to the goal. Now these sort of judgments are used in order to have breadth first search. Therefore applying the best rule is one approach. Another approach is you can apply all applicable rules. Here is an inference example.

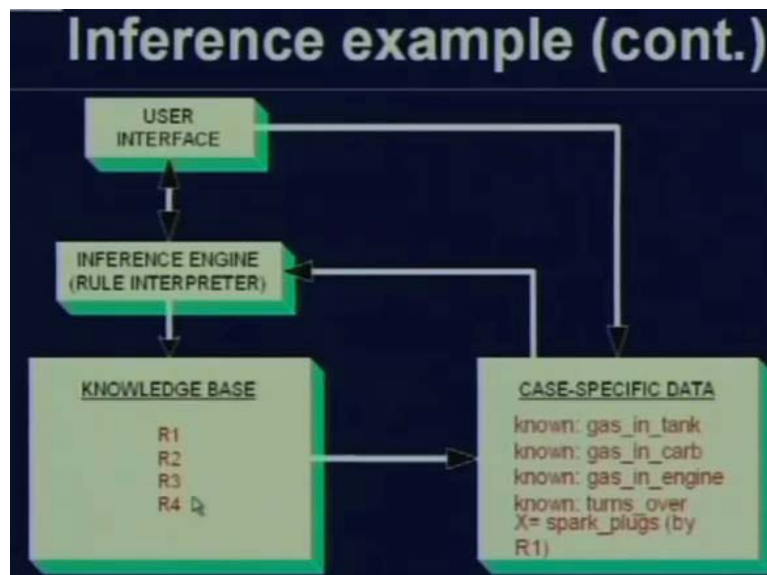
(Refer Slide Time: 36:29)

Inference example

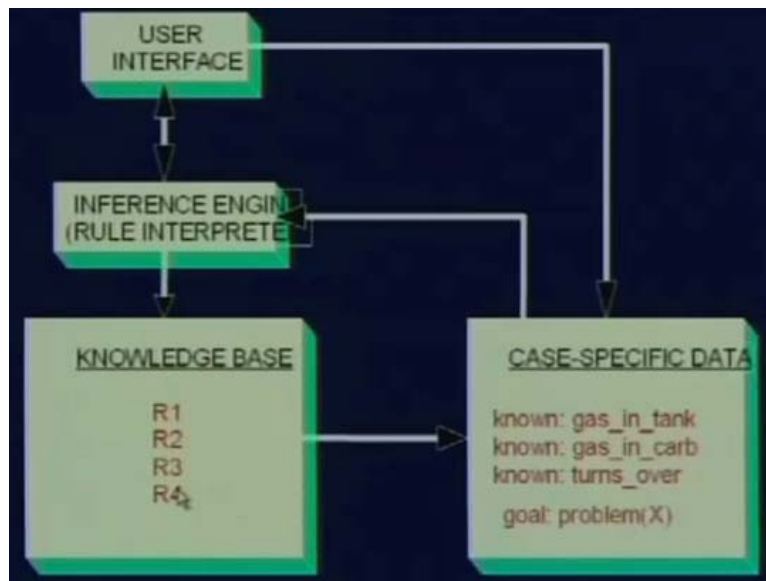
- Consider the following rules about diagnosing auto problems
 - (R1) if gas_in_engine and turns_over, then problem(spark_plugs).
 - (R2) if not (turns_over) and not(lights_on), then problem(battery).
 - (R3) if not(turns_over) and lights_on, then problem(starter).
 - (R4) if gas_in_tank and gas_in_carb, then gas_in_engine.

Consider the following rules about diagnosis of an auto car problem. The rule R1 says if there is gas in engine and it turns over. That means when you put it in it is making that sound but it is not starting then the problem is with spark plugs. As you turn the key it is not turning over so the initial sound is not coming and the lights are also not on then the problem is with battery. If it does not turn over and the lights are on then the problem is with the starter. Fourth rule says, there is gas in tank and there is gas (petrol) in tank and gas in carburetor then there is gas in engine.

(Refer Slide Time: 37:29)



(Refer Slide Time: 38:00)



Suppose these four rules are known to us and this is the status of our database. We know that there is gas in tank and there is gas in carburetor and that it is turning over so that sort of sound as been made. Our goal is to find what the problem is. And we have got all these four rules in our knowledge base so we have to find out what **the underlying** problem is.

Is it problem spark plugs, problem battery, problem starter or what is it?

Out of these rules we can always find out which rule will be enabled. Now we can see that rule R1 is enabled. Given this gas in tank, gas in carburetor and turning over scenario rule one is gas in engine because rule four says that gas in tank and gas in carburetor then gas in engine. Therefore you first fire rule R4 then R1 then you know there is gas in engine and turns over. Then we can find that using rule R1 here the problem is with spark plugs. Then we come up with a solution that x is spark plugs by R1 so we fire two rules. That is a very simplistic case of an expert system.

Although we solved this problem with forward chaining we started finding the rule that was matching with this gas in tank, gas in carburetor so R4 was enabled. And then in that cycle we came to know the gas in engine. In the second cycle this R1 was gas in engine and turns over match so R1 is fired. But if we had done with backward chaining how would you have done it?

To make it more specific, suppose our goal is to prove that the problem is with battery then with these same set of initial facts and rules provided to us we can try to solve it with backward chaining. We will see how it works and you will have to draw the search space based on the rules fired. **You need not draw the entire search space but the path that is traversed.**

But how did the problem get solved?

You need to draw the tree structure. Now, as I say a very core part of developing any expert system is acquiring the knowledge from the expert and that process is known as knowledge engineering. Knowledge acquisition is the bottleneck in developing expert systems.

(Refer Slide Time: 41:11)

Knowledge engineering

- **knowledge acquisition is the bottleneck in developing expert systems**
 - often difficult to codify knowledge as facts & rules
 - extracting/formalizing/refining knowledge is long and laborious
- **known as *knowledge engineering***

For different reasons often it is difficult to codify knowledge as facts and rules that is first of all to extract them. Extracting and formalizing or refining knowledge is a long difficult and laborious process. This process is known as knowledge engineering.

(Refer Slide Time: 41:44)

Explanation facilities are imperative for acceptance

- **TEIRESIAS (1977) front-end for MYCIN, supported knowledge acquisition and explanation**
- **could answer**
 - WHY is that knowledge relevant**
 - HOW did it come to that conclusion**
 - WHAT is it currently trying to show.**

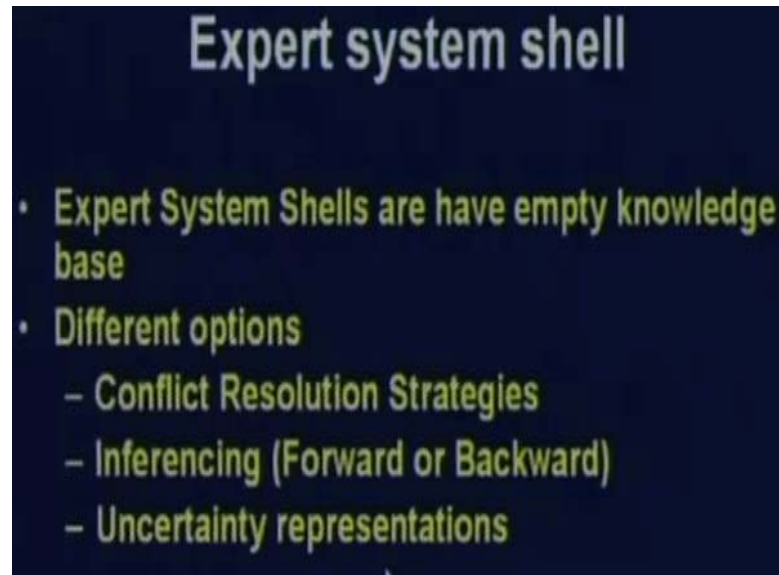
Can add new rules and adjust existing rules

The other thing is explanation facilities. These are imperative for acceptance of an expert system. Now, the MYCIN expert system has nice front end developed known as TEIRESIAS that supported knowledge acquisition and explanation. And it could answer questions like, why is that knowledge relevant? How did that system come to that conclusion? What is it currently trying to show?

This interface also allows because when you start developing an expert system. It will be too much to expect that at the very initial phase your knowledge base will be so complete that you would not require any change or you would not need any knowledge to be put in. We always learnt to do so therefore an expert system should

also learn accordingly. So there should always be a facility by which you can add knowledge to an expert system. And that addition can be done by TEIRESIAS in case of MYCIN.

(Refer Slide Time: 43:00)



What is an expert system shell?

This is also another very important criterion. As you can think of that an expert system consist of the knowledge base, the inference mechanism, the working memory, user interface etc. Now as we change the domain of application the first thing we will change for sure is the knowledge base. But may be that when I have taken so much trouble in developing one expert system and developed its inference mechanism, its conflict resolution strategy, its working memory structure and everything then I would not like to throw out everything instead I would rather like to change the knowledge base and apply in another application area if possible. That will reduce the development life cycle of an expert system.

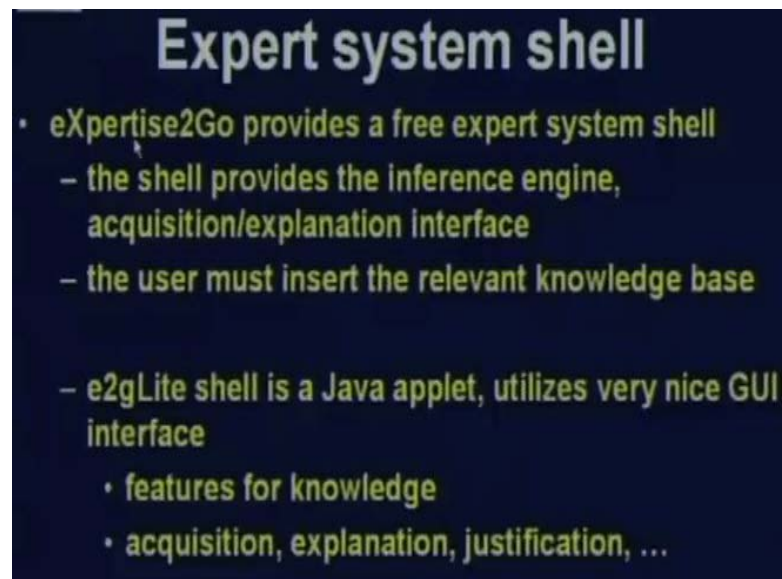
In order to meet this end expert system shells have been proposed. Expert system shells are expert systems where the knowledge base part is empty. So out of that you just take the knowledge base part make it empty and allow somebody else to put in some other knowledge base on that and that is expert system shell. And such expert system shells have got the flexibility of giving you different conflict resolution strategies from which you can select depending on the domain such as different sorts of inferencing, forward and backward reasoning etc.

Another thing we have not yet mentioned at all is representation of uncertainties. The rules we use in an expert system are often very uncertain. The logic is not as simple as 2 plus 2 is equal to 4. For example, given a particular soil quality a particular vegetable that will grow over there experts can say that this relationship of a particular soil quality a particular monsoon prediction will yield approximately this amount of vegetable of a particular crop. But all these are not absolutely certain and there are some uncertainties always involved. The uncertainties come from different sources such as uncertainties in data, the uncertainties in our expertise also.

For example, when a patient comes to a doctor or doctor prescribes a medicine, he has got estimation from this expertise that this medicine will reduce the symptom within seven days to a particular level. But he cannot always be certain that within these seven days this will reduce which is one uncertainty and the second thing is to how much it will reduce is not always possible to quantify. The frequency of fever or headache will come down but how much?

Therefore there are uncertainties in different levels. But in order to be successful an expert system shell must provide some means of representing uncertainty.

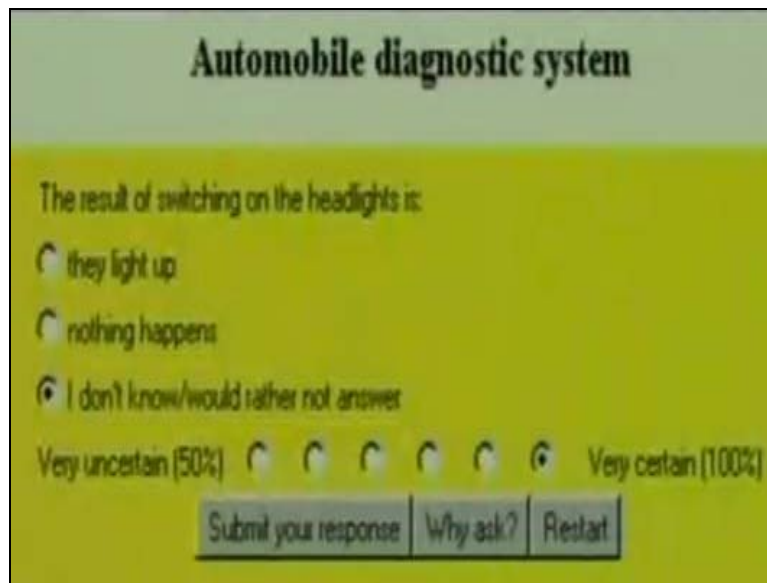
(Refer Slide Time: 46:49)



Expertise2go: It is a free expert system shell where the shell provides the inference engine knowledge acquisition and explanation interface. The user must insert the relevant knowledge base.

Expertise2go Lite version is a java applet where a very nice GUI interface is available and there are features for knowledge acquisition, explanation, justification etc.

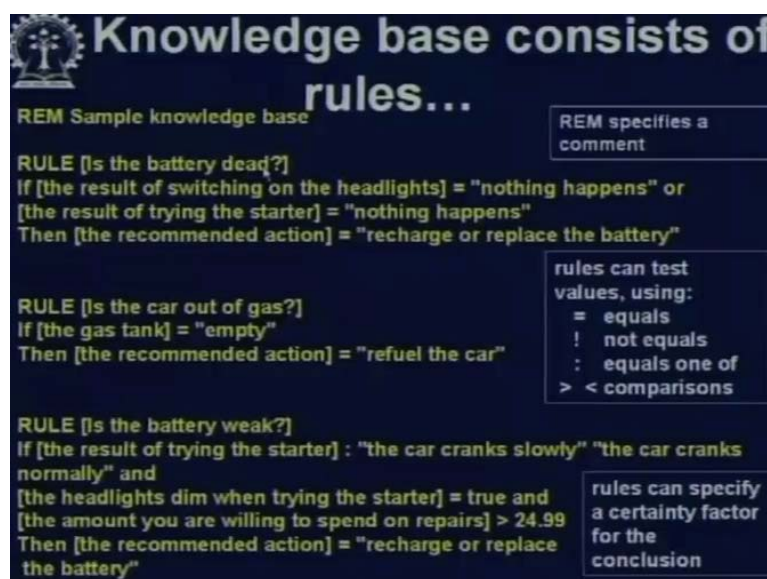
(Refer Slide Time: 47:40)



Here is a typical GUI where the expert system shell has been used to develop an automobile diagnostic system. This is a GUI which is shown to the user so the user can select the result of switching on the headlight is: we have just shown the example: they light up, nothing happens, I do not know. This sort of facility is also given. Therefore your rule base must be able to capture those also. Either you are certain or very uncertain you can submit your response. You can also ask the system why you are being questioned like this and the system should be able to explain it to you. Therefore these are the typical GUI.

In this expertise2go let us see typical rules that are present. This REM is a remark. We are showing a sample knowledge base here. REM specifies a comment. We also put in a comment because this is the rule. The rule is going to check, is the battery dead?

(Refer Slide Time: 48:55)



Look at these brackets which are the antecedents. If the result of switching on the headlights is equal to nothing happens. Now this is a peculiar case where we are allowing disjunctions also. The result of trying the starter equals nothing happens. Then the recommendation actions are recharge or replace the battery. Now depending on the shell that you are using the syntax of the rule will change. This is a particular syntax here. So in this particular case say the shell is providing me with some operators like equals, not equals, equals one of or comparisons so look at the rule; is the car out of gas? That is what I want to test.

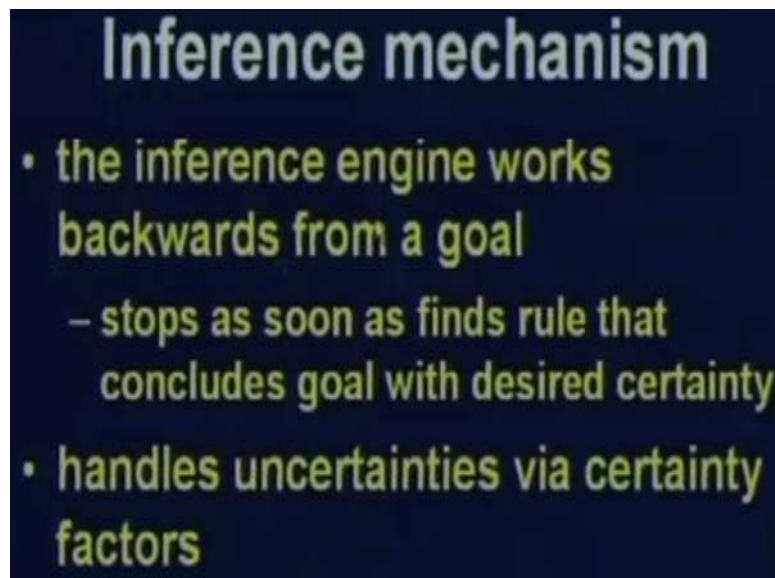
If the gas tank equals empty then the recommended action is refuel the car. In that way there are rules. And as the system fires the rules it also prompts for acquiring information, what sort of information are those?

Prompt: the result of trying the starter.

What happens when you turn the key to try to start the car?


The car cranks normally, the car cranks slowly and nothing happens. This is being shown to the user and the user selects one as we have shown through the GUI. For example, it is prompting, a multiple choice like the smell of gasoline is present when trying the starter, not present when trying the starter so the user is asked to select. And these sort of prompt actions or (f...) are really activating this sort of GUI as shown here.

(Refer Slide Time: 50:55)



The inference engine in this e2glite shell is the inference engine works backwards from a goal, it is backward chaining. Obviously it stops as soon as it finds the rule that concludes goal with desired certainty. It handles uncertainties via certainty factors. Now these certainty factors are really important.

(Refer Slide Time: 51:22)



Rule Based Inferencing

Many ways to handle uncertainty

- probabilities
specify likelihood of a conclusion, apply Bayesian reasoning
- certainty factors
a certainty factor is an estimate of confidence in conclusions
not probabilistically precise, but effective

This is a very important method. But before that let me state that there are different ways of handling uncertainty. In the following couple of lectures we will be dealing with different techniques of uncertainty management. The different ways of handling uncertainty: One can be probabilities where we specify likelihood of a conclusion or apply Bayesian reasoning. And certainty factor is an estimate of confidence in conclusions. It is an estimate provided by the experts. It is not as mathematically precise as is the case of probabilities but it is also very effective.

(Refer Slide Time: 52:25)



Mycin

- medical expert system for analysis of meningitis infections
- backward chaining rules-based expert system
 - produces focused questions
 - easy to produce an NL interface that need only interpret answers to specific questions

MYCIN is a medical expert system for analysis of meningitis infections. MYCIN uses backward chaining rule based expert system. It produces focused questions. It also produces a natural language like interface for communication with the users.

(Refer Slide Time: 52:55)

```
Rule-Based Systems: Mycin
IF: (1) the stain of the organism is gram-positive, and
     (2) the morphology of the organism is coccus, and
     (3) the growth conformation of the organism is clumps.
THEN
     there is suggestive evidence(0.7) that the identity of
     the organism is staphylococcus.

PREMISE: ($AND (SAME CNTXT GRAM GRAMPOS)
             (SAME CNTXT MORPH COCCUS)
             (SAME CNTXT CONFORM CLUMPS))
ACTION: (CONCLUDE CNTXT IDENT STAPHYLOCOCCUS TALLY
        .7)
```

Some typical rules of Mycin:

The syntax has now changed compared to the earlier one we saw. If the stain of organism is gram positive and there is no disjunction law here and the morphology of the organism is coccus and the growth conformation of the organism is clumps. If these three are true then remember it is a medical case we cannot serially with certainty so what we will say is there is suggestive evidence of some value .7 that the identity of the organism is staphylococcus.

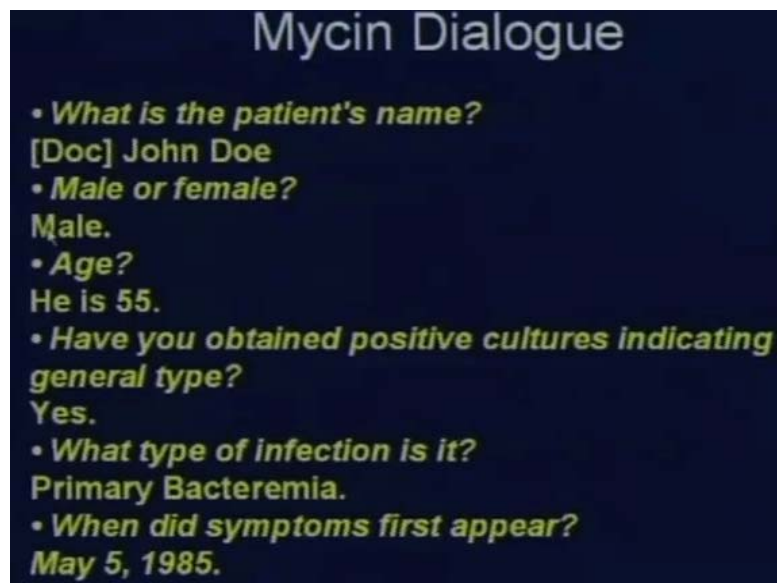
That is a particular organism.

Now this rule in MYCIN can be translated in a Lisp syntax like structure but the premise of the rule is and there is a conjunction of in the same context that is the organism and the strain of the organism is gram positive, morphology is coccus and conformation is clumps. Then action is conclude context and the conclude is the staphylococcus with a tally .7.

What is this .7?

That is a very key factor the certainty factor of Mycin. But the beauty of MYCIN is the way it converses with the user.

(Refer Slide Time: 54:50)



Mycin Dialogue

- *What is the patient's name?*
[Doc] John Doe
- *Male or female?*
Male.
- *Age?*
He is 55.
- *Have you obtained positive cultures indicating general type?*
Yes.
- *What type of infection is it?*
Primary Bacteremia.
- *When did symptoms first appear?*
May 5, 1985.

What is the patient's name?

The doctor types in John Doe. Male or female and these italicized parts shown here is the system's response. The doctor says male. What is the age?

He is 55. The system also understands this. The doctor cannot write that to my mind he is approximately 55 so it is not free flowing English but he can realize some restrictions that he can type in he is 55.

Have you obtained positive cultures indicating general type?

Yes the doctor says.

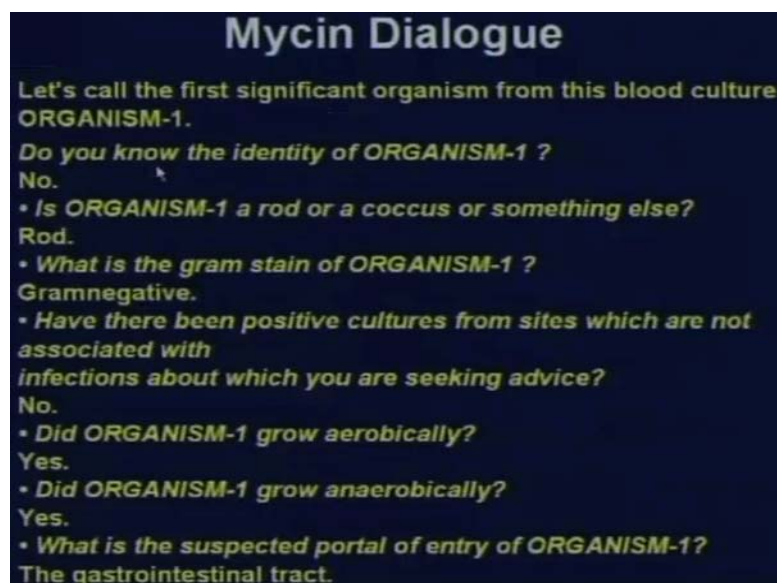
What type of infection is it?

It is primary bacteremia.

When did the symptoms first appear?

May 5 1985.

(Refer Slide Time: 55:29)



Mycin Dialogue

Let's call the first significant organism from this blood culture ORGANISM-1.

Do you know the identity of ORGANISM-1 ?
No.

- *Is ORGANISM-1 a rod or a coccus or something else?*
Rod.
- *What is the gram stain of ORGANISM-1 ?*
Gramnegative.
- *Have there been positive cultures from sites which are not associated with infections about which you are seeking advice?*
No.
- *Did ORGANISM-1 grow aerobically?*
Yes.
- *Did ORGANISM-1 grow anaerobically?*
Yes.
- *What is the suspected portal of entry of ORGANISM-1?*
The gastrointestinal tract.

Now it goes on.

Do you know the identity of the organism one?

No

Is organism one a rod or a coccus or something else?

The doctor says rod. In this way a very naturally flowing discussion goes on.

(Refer Slide Time: 55:55)

Mycin Recommendation

My recommendation will be based on the opinion that the identity of ORGANISM-1 may be:

- 1 Pseudomonas-aeruginosa,*
- 2 Klebsiella-pneumoniae,*
- 3 E. Coli,*
- 4 Bacteroides-fragilis,*
- 5 Enterobacter, or*
- 6 Proteus-non-mirabilis.*

On a scale of 0 to 4, where higher numbers indicate greater severity, how would you rate the patient's degree of sickness?

3

And ultimately MYCIN comes up with a recommendation. My recommendation will be based on the opinion that the identity of organism may be, note this may be Pseudomonas Aeruginosa, E. Coli and other such possibilities. Then it furthers. So this is my provisional diagnosis. On a scale of 0 to 4 it is asking the doctor where higher numbers indicate greater severity.

(Refer Slide Time: 56:25)

Example Mycin Recommendation

Does the patient have a clinically significant allergic reaction to any antimicrobial agent?

No.

How much does the patient weigh?

70 kilograms.

My first recommendation is as follows:

*In order to cover for items 1, 2, 3, 5, and 6, give Gentamycin using a dose of 119 mg (1.7 mg/kg) for 10 days.
Modify dose in renal failure.*

And in order to cover for item 4, give Clindamycin using a dose of 595 mg (8.5 mg/kg) for 14 days.

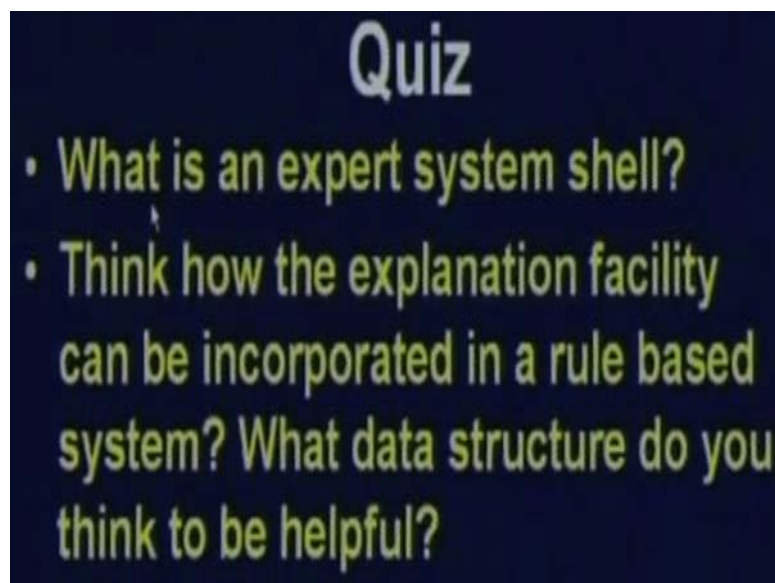
How would you rate the patient's degree of sickness?

The doctor says 3 and the MYCIN further goes on asking more questions and then comes up with a recommendation that there were so many possibilities and just as an expert doctor it says, in order to cover for items 1 2 3 5 and 6 give Gentamycin using a dose of 119 milligram 1.7 milligram/Kg for ten days, modify dose in the case of renal failure and in order to cover for item 4 give Cindamycin using a dose of so and so for 14 days. Just see how naturally a system can communicate and that is what an expert system can do in order to come to a reasonable **conclusion** once it is nicely built and it is restricted in a proper domain.

Here is a quiz as an exercise.

What is an expert system shell and the **second problem is really challenging on your part.**

(Refer Slide Time: 57:55)



Forward chaining and backward chaining concepts are made clear for your understanding. Now, Think how explanation facility can be incorporated in a rule based system. By explanation facility find out that if at any particular point a rule has stated a conclusion. You have got fever, you have got malaria for example. Then if the user says why then the system explanation facility should be able to present that rule which really concluded malaria. And might be that had some antecedents which have been found true then how did these antecedents were evaluated to be true or how is that these antecedents were found to be true that should also be found out such that which rules made these antecedents true.

What is the data structure you think will be helpful?

Data structure stack that you know of will be immensely helpful.