Rule Based Systems

Rule-Based Systems
- Forward and Backward Chaining
- Exemplars
- Business Side of AI

Expert Systems

Expert systems were sold as a way to capture specialist human expertise which was in short supply. Eg..,

- Medical expertise
- Computer configuration expertise.
- Expertise for oil exploration.

Aim was to develop systems capturing this expertise, so the knowledge could be deployed where experts were unavailable.

Rule-Based Systems

In Logic (and semantic nets) we represent knowledge in a declarative, static way - as some facts and rules that are true.

Rules in logic say what is TRUE given some conditions.

Rule-based systems are based on rules that say what to DO, given various conditions.

IF <this is the case> THEN <do this>

A special ENGINE controls when rules are invoked.

Rules vs Logic

Simple examples are very similar to ➔ rules in logic.

Less concern about precise semantics and sound inference.

More about programmed operations based on explicit and editable knowledge

Forward, Backward, Mixed

Two main kinds of rule-based systems: forward chaining and backward chaining.

Forward chaining starts with the facts, and sees what rules apply (and hence what should be done) given the facts.

Backward chaining starts with something to find out, and looks for rules that will help in answering it given current fact (or requiring new inputs, eg. Diagnostic tests).

Forward chaining

In a forward chaining system:

- Facts are held in a *working memory*
- *Condition-action* rules represent actions to take when specified facts occur in working memory. IF condition THEN action.
- Typically the actions involve adding or deleting facts from working memory.

Production System is forward-chaining

Control cycle called *recognise-act* cycle.

Repeat:

- Find all rules which have satisfied conditions given facts in working memory.
- Choose one, using *conflict resolution strategies*.
- Perform actions (e.g. modify working memory).

Until no rules can fire, or “halt” symbol added to working memory.

Backward Chaining

Same rules/facts may be processed differently, using *backward chaining* engine.

Start with possible hypothesis.

Set this as a goal to prove or its inverse to disprove (as in resolution).

Backward Chaining

Basic algorithm:

To prove goal G:

If G is in the initial facts, it is proven.

Otherwise, find a rule which can be used to conclude G, and try to prove each of that rule's conditions.

Applications

Backward chaining systems have been fairly widely used in E.g., medical expert systems, where start with set of hypotheses on possible diseases - try to prove each one, asking additional diagnostic questions when fact is unknown.

Value of KBS?

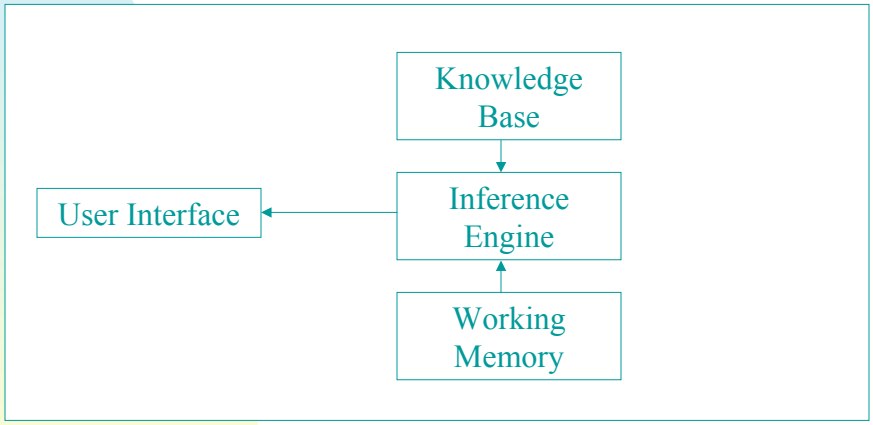
Expert systems provide expert quality advice, diagnoses and recommendations on real world problems

Designed to perform function of a human expert

Examples:

- Medical diagnosis - program takes place of a doctor; given a set of symptoms the system suggests a diagnosis and treatment
- Car fault diagnosis - given car's symptoms, suggest what is wrong with it

Structure of Expert System



Rules

The knowledge base of an expert system is often ***rule based*** – the system has a list of rules which determine what should be done in different situations

These rules are initially designed by human expert/s

The rules are called ***production rules***

Each rule has two parts, the ***condition-action pair***

1. **Condition** – what must be true for the rule to fire
2. **Action** – what happens when the condition is met

Can also be thought of as IF-THEN rules

Rule-Based Systems – Working Memory

The contents of the working memory are constantly compared to the production rules

When the contents match the condition of a rule, that rule is fired, and its action is executed

More than one production rule may match the working memory

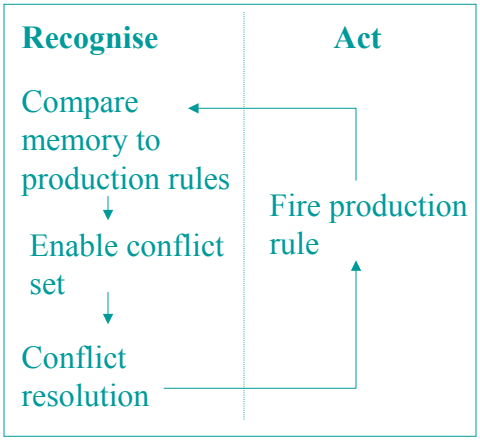
Production System Recognise-Act Cycle

The system cycles around in the ***recognise-act cycle***

Sometimes there is more than one rule to "fire"

Whenever a condition is matched, it is added to the ***conflict set*** – all the rules which are currently matched

The system must then decide which rule within the conflict set to fire – ***conflict resolution***



Why Conflict Resolution?

Adds procedural direction to what seems undirectional processing

Random choice of rules lead to programs which do not run the same way twice

Apply to newest members of WM

Apply oldest firing rule

Pattern Matching Efficiency

- At face value forward chaining is very inefficient
- Techniques such as RETE have been developed which keep perfect track of state of a WM and which rules to fire.
- Allow undoing and backtracking for more sophisticated reasoning.

Expert System Examples

- R1
- Mycin
- Internist

R1

An Expert Computer Configurer

PROBLEM: Specify a machine with certain pieces (e.g. CPU, memory, periphs), find the way to configure it into boxes, with floorplane, backplanes, wiring, etc

BUT:

There are many possible ways

Cost and flexibility matter

Certain pieces give weird constraints

Elements of R1

Written in OPS5

Uses Constraint Satisfaction

Large Database of Parts

Large Knowledgebase of Constraints

Organized as a Sequence of "Tasks"

1) Verify Order

2) Configure CPU Processor

3) configure Bus Modules

4) Count Cabinets

5) Devise Floor Plan

6) Custom manufacture Cabling

MYCIN

MYCIN's "knowledge base" consisted of set of IF-THEN rules, e.g.,

IF the infection is primary-bacteremia

AND the site of the culture is one of the sterile sites

AND the suspected portal of entry is the gastrointestinal tract

THEN there is suggestive evidence (0.7) that infection is bacteroid.

MYCIN: Reasoning and Problem Solving Strategy

- MYCIN could use backward chaining to find out whether a possible bacteria was to blame.
- This was augmented with “certainty factors” that allowed an assessment of the likelihood, if no one bacteria was certain.
- MYCIN’s problem solving strategy was simple:
  - For each possible bacteria:
    - Using backward chaining, try to prove that it is the case, finding the certainty.
  - Find a treatment which “covers” all the bacteria above some level of certainty.

MYCIN: Problem Solving

- When trying to prove a goal through backward chaining, system could ask user certain things.
  - Certain facts are marked as “askable”, so if they couldn’t be proved, ask the user.
- This results in following style of dialogue:

MYCIN: Has the patient had neurosurgery?
USER: No.
MYCIN: Is the patient a burn patient?
USER: No.
…
MYCIN: It could be Diplococcus..

Modelling Human Diagnostic Strategies.

Problem Solving Strategy used in MYCIN only works when small number of hypotheses (e.g., bacteria).

For hundreds of possible diseases, need a better strategy.

Later medical diagnostic systems used an approach based on human expert reasoning.

Diagnostic Reasoning: Internist

Internist is a medical expert system for general disease diagnosis.

Knowledge in system consists of disease profiles, giving symptoms associated with disease and strength of association.

E.g., ECHINOCOCCAL CYST of LIVER

Finding	Strength
Cough	*1*
Fever	*2*
Jaundice	*4*

Problem Solving in Internist

Use initial data (symptoms) to suggest, or trigger possible diseases.

Determine what other symptoms would be expected given these diseases.

Gather more data to *differentiate* between these hypotheses. Either:

- If one hypothesis most likely, try to confirm it.
- If many possible hypotheses, try to rule some out.
- If a few hypotheses, try to discriminate between them.

Summary

Effective systems have been developed that capture expert knowledge in areas like medicine, equipment repair, tax law.

Typically combine rule-based approaches, and some top level control of the problem solving process; augmented with probabilistic forms of reasoning

The KBS Business

- Corporations and Govt jumped on the bandwagon, creating the first “AI” industry
- Many faculty formed/joined companies
  - Business Model sell “Expert System Shell”
  - Consultantcy “Knowledge Engineering”
- Most Businesses eventually collapsed, tho many ES are still in use. Why?
  - Rule systems didn’t scale up to complexity
  - Humans don’t want to be replaced.