Rule-based Expert Systems

- Rules as a Knowledge Representation technique
- Structure of a rule-based expert system
- Characteristics of an expert system
- Forward/Backward chaining
- Conflict Resolution
Rules as a knowledge representation technique

- The human mental process is internal, it is too complex to be represented as an algorithm. However, most experts are capable of expressing their knowledge in the form of rules for problem solving.

- IF-THEN structure – relates given information or facts in the IF part to some action in the THEN part (description of how to solve a problem).

- Any rule consists of two parts:
  - the IF part (antecedent – premise or condition)
  - The THEN part (consequent – conclusion or action)

- A rule have multiple antecedents joined by AND (conjunction) or OR (disjunction)

- The antecedent of a rule incorporates: the object and its value are linked by an operator.
What rules can represent

- Relations
  IF the ‘fuel tank’ is empty
  THEN the car is dead

- Recommendation
  IF the season is autumn
  AND the sky is cloudy
  AND the forecast is drizzle
  THEN the advice is ‘take an umbrella’

- Directive
  IF the car is dead
  AND the ‘fuel tank’ is empty
  THEN the action is ‘refuel the car’

What rules can represent

- Strategy
  IF the car is dead
  THEN the action is ‘check the fuel tank’;
    step 1 is complete

  IF step 1 is complete
  AND the ‘fuel tank’ is full
  THEN the action is ‘check the battery’;
    step 2 is complete

- Heuristic
  IF the spill is liquid
  AND the ‘spill PH’ < 6
  AND the ‘spill smell’ is vinegar
  THEN the ‘spill material’ is ‘acetic acid’
Structure of rule-based Expert System

- A production system model - Newell and Simon (CMU), 1970s.

- Solve problems by applying their knowledge (expressed as production rules) to a given problem represented by problem-specific information.

- Production rules (stored in the long term memory) and problem-specific information, facts (in the short-term memory)
The Knowledge base – contains the domain knowledge useful for problem solving.
- Each rule specifies a relation, recommendation, directive, strategy, heuristics.
- IF (condition) THEN (action) – When the condition part of the rule is satisfied, the rule is said to fire and the action part is executed.

The Database includes a set of facts used to match against the IF (condition) parts of the rules stored in the knowledge base.

The Inference Engine carries out the reasoning whereby the expert system reaches a solution. It links the rules given in the knowledge base with the facts provided in the database.
The explanation facilities enable the user to ask the expert system how a particular condition is reached and why a specific fact is needed. An expert system must be able to explain its reasoning and justify its advice, analysis or conclusion.

The user interface is the means of communication between a user seeking a solution and an expert system.

Characteristics of an expert system

- High-quality performance (speed of reaching a solution) - built to perform at a human expert level in a narrow, specialized domain.
- Apply heuristics to guide the reasoning and thus reduce the search area for a solution
- Explanation capability – enable the experts to review its own reasoning and explain its decisions
- Employ symbolic reasoning when solving a problem (symbol – used to represent different types of knowledge such as facts, rules, concepts)
Characteristics of an expert system

- Knowledge is separated from the processing (the knowledge base and the inference engine are split up)
  - Conventional program – a mixture of knowledge and the control structure to process this knowledge, difficulties in understanding the program code, any change to the code effects both the knowledge and its processing.

- With an expert system shell
  - Enter rules in knowledge base,
  - Each new rule adds some new knowledge and make the expert system smarter.

Inference chains – Forward/Backward

- The inference engine compares each rules stored in KB with facts contained in DB.
- If (condition) part of the rule matches a fact, the rule is fired and its THEN (action) part is executed.
- The matching of the rule IF parts to the facts produces inference chains. The inference chain indicates how the expert system applies the rules to reach a conclusion.
Inference engine cycles via a match-fire procedure

An example of an inference chain

Rule 1: IF Y is true
        AND D is true
        THEN Z is true

Rule 2: IF X is true
        AND B is true
        AND E is true
        THEN Y is true

Rule 3: IF A is true
        THEN X is true
Forward Chaining

- Data-driven reasoning
- Starts from the known data and proceeds forward with that data.
- Each time only the topmost rule is executed.
- When fired, the rule adds a new fact in the database.
- Any rule can be executed only once.
- The match-fire cycle stops when no further rules can be fired.

Forward Chaining

- A technique for gathering information and then inferring from it whenever can be inferred.
- Many rules can be executed that have nothing to do with established goal.
- If our goal is to infer only one particular fact, the forward chaining inference technique would not be efficient.
Backward chaining

- The goal-driven reasoning
- The goal (a hypothetical solution) - the inference engine attempts to find the evidence to prove it.
  - The knowledge base is searched to find rules that might have the desired solution
  - Such rules must have the goal in their THEN (action) parts.
  - If such a rule is found and its IF (condition) part matches data in the database, then the rule is fired and the goal is proved. However, this is rarely the case.

Backward chaining

- The engine puts aside the rule it is working with (“stack”) and set up a new goal, a subgoal, to prove the IF part of this rule.
- Then the knowledge base is searched again for rules that can prove the subgoal.
- The engine repeats the process of stacking the rule until no rules are found in the knowledge base to prove the current subgoal.
How to choose between forward and backward chaining?

- If an expert first needs to gather some information and then tries to infer from it whatever can be inferred, choose the **forward chaining** inference engine.

- If your expert begins with a hypothetical solution and then attempts to find facts to prove it, choose the **backward chaining** inference engine.

Conflict resolution

Assume that there are two simple rules for crossing a road. Let us now add third rule.

- Rule 1:
  
  IF the ‘traffic light’ is green
  THEN the action is go

- Rule 2:
  
  IF the ‘traffic light’ is red
  THEN the action is stop

- Rule 3:
  
  IF the ‘traffic light’ is red
  THEN the action is go
Conflict resolution

- We have two rules, rule 2 and rule 3 with the same IF part.
- Both of them can be set to fire when the condition part is satisfied.

Conflict set

- The engine must determine which rule to fire from such as a set
- Conflicting resolution – a method for choosing a rule to fire when more than one rule can be fired in a given cycle.

Conflict resolution

In forward chaining

- Both rules would be fired
- Rule 2 is fired as the topmost one and as a result, its THEN part is executed and linguistic object action obtains value stop.
- However, rule 3 is also fired because the condition part of this rule matches the fact ‘traffic light’ is red, which is in the database.
- As a consequence, object action takes new value go.
Methods used for conflict resolution

- Fire the rule with the highest priority
  - In simple applications, the priority can be established by placing the rules in an appropriate order in the KB (~ 100 rules)

- Fire the most specific rule.
  - The longest matching strategy
  - Based on the assumption that a specific rule processes more information than a general one.

- Fire the rule that uses the data most recently entered in the database.
  - Relies on tags attached to each fact in the database.
  - In the conflict set, the expert system first fires the rule whose antecedent uses the data most recently added to the database.
Metaknowledge

- Metaknowledge can be simply defined as knowledge about knowledge. Metaknowledge is knowledge about the use and control of domain knowledge in an expert system.
- In rule-based expert systems, metaknowledge is represented by metarules. A metarule determines a strategy for the use of task-specific rules in an expert system.

Metarules

- Metarule 1:
  Rules supplied by experts have higher priorities than rules supplied by novices

- Metarule 2:
  Rules governing the rescue of human lives have higher priorities than rules concerned with clearing overloads on power system equipment.
Advantages of rule-based expert system

- Natural knowledge representation – an expert usually explains the problem-solving procedure with “In such-and-such situation, I do so-and-so”. → represented quite naturally as IF-THEN production rules.
- Uniform structure: production rules have uniform IF-THEN structure. Each rule is an independent piece of knowledge (self-documented)

Advantages of rule-based expert system

- Separation of knowledge from its process
  The structure of a rule-based expert system provides an effective separation of the knowledge base from the inference engine. This makes it possible to develop different applications using the same expert system shell.

- Dealing with incomplete and uncertain knowledge
  Most rule-based expert systems are capable of representing and reasoning with incomplete and uncertain knowledge
Disadvantages of rule-based expert system

- Opaque relations between rules.
  Although individual production rules are relatively simple and self-documented, their logical interactions within a large set of rules may be opaque. Rule-based systems make it difficult to observe how individual rules serve the overall strategy.

- Ineffective search strategy
  The inference engine applies an exhaustive search through all the production rules during each cycle with a large set of rules (over 100 rules) can be slow, and thus large rule-based systems can be unsuitable for real-time applications.

- Inability to learn
  In general, rule-based expert systems do not have an ability to learn from experience.
  Unlike a human expert, who knows when to “break the rules”, an expert system cannot automatically modify its knowledge base, or adjust existing rules or add new ones.
  The knowledge engineer is still responsible for revising and maintaining the system.