What is an expert system?

- An expert system is computer software that embodies a significant portion of the specialized knowledge of a human expert in a specific, narrow domain, and emulates the decision-making ability of the human expert.
- The technology is based on the premise that what makes a person an expert is years of experience that enables him to recognize certain patterns in a problem as being similar to patterns he has seen previously.
- It is one of the early (large-scale) successes of artificial intelligence.

Advantages and disadvantages

<table>
<thead>
<tr>
<th>Human Expert</th>
<th>Expert System</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pro Expert System</td>
<td>Pro Human</td>
</tr>
<tr>
<td>perishable</td>
<td>creative</td>
</tr>
<tr>
<td>unpredictable</td>
<td>adaptive</td>
</tr>
<tr>
<td>slow reproduction</td>
<td>broad focus</td>
</tr>
<tr>
<td>expensive</td>
<td>common sense</td>
</tr>
<tr>
<td>slow processing</td>
<td>lacks inspiration</td>
</tr>
<tr>
<td>affordable</td>
<td>needs instruction</td>
</tr>
<tr>
<td>fast processing</td>
<td>narrow focus</td>
</tr>
<tr>
<td>machine knowledge</td>
<td></td>
</tr>
</tbody>
</table>

Many applications

- control (air traffic)
- debugging
- design (computer configuration)
- medical diagnosis
- instruction/training
- interpretation (speech)
- monitoring (nuclear plant)
- planning (mission planning)
- Factory scheduling
- prediction (weather)
- repair (telephone)
- control (air traffic)
- debugging
- design (computer configuration)
- medical diagnosis
- instruction/training
- interpretation (speech)
- monitoring (nuclear plant)
- planning (mission planning)
- Factory scheduling
- prediction (weather)
- repair (telephone)

Expert system technology

- Consists of:
  - Knowledge-base (+ language for encoding knowledge)
  - Inference engine (algorithms for reasoning)

Rule-based knowledge representation

- A rule consists of two parts: condition (antecedent) part and conclusion (action, consequent) part. I.e.:
  - IF (conditions) THEN (actions)
- Antecedent part of the rule describes the facts or conditions that must exist for the rule to fire
- Consequent describes
  - the facts that will be established, or
  - the action that will be taken or conclusion that will be made
### Structure of Rule-Based Expert System

![Structure of Rule-Based Expert System Diagram]

#### Components of rule-based system
- **Working memory (Fact memory)**
  - data structures representing the current state of the system (Facts = positive literals with no variables)
- **Knowledge base (Rule memory)**
  - set of condition action rules
  - `<WM pattern> → <WM changes>`
  - Rules can add and delete facts from working memory
- **Inference engine (Rule interpreter)**
  - applies production rules to the working memory.

### Components of rule-based system

- **Inference Engine**
  - Agenda
  - Working Memory (Facts)
- **Knowledge Base**
  - Rules
- **Explanation Facility**
  - User Interface
- **Knowledge Acquisition Facility**

### Pattern Matching
- Find all rules that are eligible to be “fired” by matching left-hand side of rules to facts in working memory
- If rules have variables, matching requires unification (the RETE algorithm can do this efficiently)

### Conflict resolution
- When several rules are eligible to fire, there are different criteria for selecting which to fire next:
  - Fire rule with the highest priority
  - Fire rule with the most specific condition
  - Fire the most recently used rule
  - Fire rule with the most recently used variable
  - Fire the most recently added rule

### Forward Chaining
- **Match phase:** Find rules eligible to fire
  - Is there one? → No → STOP
- **Conflict resolution phase:** Choose an applicable rule
- **Act phase:** Fire the Rule

### Rule-based system cycle

1. **Pattern Matching**
2. **Conflict Resolution (select rule)**
3. **Fire rule**
**Advantages of rule-based expert systems**
- Good for complex problems where humans have expert knowledge
- Expressiveness and intuitiveness
  - Rules can be understood by a non-programmer
- Simplicity (uniform KR)
- Modularity and modifiability
  - Individual rules can be changed and added
  - But it is hard to analyze the effect of each new rule on performance.

**Difficulties**
- “Knowledge acquisition bottleneck”
  - Difficulty in acquiring expert knowledge and putting it in system
- Not good at common-sense reasoning
- Traditional systems not good at handling uncertainty
  - Recent techniques have overcome this

**Expert system shell**
- A tool for building an expert system
- A software package that includes an inference engine, knowledge representation language, user interface, and all the code used by an expert system – regardless of the domain
- All you have to add is the knowledge, i.e., the rules and facts used by an expert to solve problems in a certain domain
- CLIPS is an example of an expert system shell

**CLIPS: basic elements**
- **Fact-list**: global memory for data that contains the facts
- **Knowledge-base**: contains all the rules.
- **Inference engine**: forward chaining

**Syntax for facts**
- (single-field)
- (two fields)
- (speed 38 mph)
- (cost 78 dollars 23 cents)
- (name “John Doe”)
- **Adding facts**:
  (assert <facts>)
- **Removing facts**:
  (retract <fact-index>)

**Syntax for rules**
- (defrule <rule name> [<optional comment>]
  <<patterns>> =>
  <<actions>>)
- (defrule fire-emergency “An example rule”
  (emergency fire)
  =>
  (assert (action activate-sprinkler-system)))
Syntax for variables
- ?speed
- ?sensor
- ?value

(defrule grandfather
  (is-a-grandfather ?name)
  =>
  (assert (is-a-man ?name)))

E.g.: (is-a-grandfather John) → ?name = John
(is-a-grandfather Joe) → ?name = Joe

Wildcards
(person <name> <eye-color> <hair-color>)
(person John brown black)
(person Joe blue brown)

(defrule find-brown-haired-people
  (person ?name ?brown)
  =>
  (printout t ?name " has brown hair")

States that eye color doesn't matter.

Expert Systems vs Conventional Programs

<table>
<thead>
<tr>
<th>Characteristic</th>
<th>Conventional Program</th>
<th>Expert System</th>
</tr>
</thead>
<tbody>
<tr>
<td>Control by...</td>
<td>Information &amp; control integrated</td>
<td>Knowledge separate from control</td>
</tr>
<tr>
<td>Solution by...</td>
<td>Algorithm</td>
<td>Rules &amp; inference</td>
</tr>
<tr>
<td>Representation</td>
<td>Numeric</td>
<td>Symbolic</td>
</tr>
<tr>
<td>Execution</td>
<td>Generally sequential</td>
<td>Opportunistic rules</td>
</tr>
</tbody>
</table>

Important difference:
-- Expert system is developed by considering how a human expert would perform a task.
-- Conventional program is developed by considering how a computer would perform a task.

Knowledge Representation
- There are other ways of representing declarative knowledge besides first-order logic and rule-based systems
- Semantic networks and frames are closely related formalisms that graphically represent taxonomies of objects and their properties
- They often provide a clearer view of represented knowledge, and allow more efficient inference, than an equivalent logical representation

Semantic networks
- Graph structures that encode taxonomic knowledge of objects and their properties
  -- objects represented as nodes
  -- relations represented as labeled edges
- Inheritance = form of inference in which subclasses inherit properties of superclasses

Semantic Network Example
Adapted from Figure 9.1 Page 252 of Rich and Knight
Frames

- A limitation of semantic networks is that additional structure is often necessary to distinguish
  - statements about an object’s relationships
  - properties of the object
- A frame is a node with additional structure that facilitates differentiating relationships between objects and properties of objects.
- Sometimes called a “slot-and-filler” representation

Multiple inheritance

CycL

- CYC originally used a frame-based system to represent knowledge, but has since developed its own knowledge representation language, CycL, which is an extension of first-order logic
- All the knowledge in CYC is represented declaratively, as facts and rules
- CYC presently has close to a million facts and rules from which its inference engine can derive new conclusions using deductive reasoning
- The present estimate (which keeps being revised upwards) is that it needs ten to twenty million facts and rules to have common sense

CYC (from enCYClopediа) Project

- Began at MCC (Microelectronics and Computer Technology Corporation) in Austin, TX, in 1984, as a ten-year project with a $35 million grant
- Since 1995 has been continued by a private company, CYCOP
- A massive knowledge base and inference engine designed to overcome the limitations of expert system technology by formalizing common sense knowledge

Examples of common-sense knowledge

- “Cars in motion generally have a driver”
- “Police in most countries are armed”
- “If you drop a glass, it will break”
Natural language interface

- Development of a natural language interface for CYC is ongoing
- The goal is for CYC to learn by reading books and articles, or by having people tell it things in English
- Current natural language interface is useful but very primitive (this is a *hard* problem)

Applications of CYC

- Although CYC is far from having common sense, the techniques developed in the course of this project for knowledge representation and inference have a number of applications, including:
  - Heterogenous database browsing and integration
  - Captioned image retrieval
  - Natural language processing