More on Expert Systems
Knowledge Engineering

The process of building an expert system:

1. The knowledge engineer establishes a dialog with the human expert to elicit knowledge.
2. The knowledge engineer codes the knowledge explicitly in the knowledge base.
3. The expert evaluates the expert system and gives a critique to the knowledge engineer.
Development of an Expert System
Uncertainty

• Uncertainty = having limited knowledge (more than possible outcomes)

• Both human experts and expert systems must be able to deal with uncertainty.

• It is easier to program expert systems with shallow knowledge than with deep knowledge.

• Shallow knowledge – based on empirical and heuristic knowledge.

• Deep knowledge – based on basic structure, function, and behavior of objects.
Limitations of Expert Systems

- **Limitation 1**: typical expert systems cannot generalize through analogy to reason about new situations in the way people can.

- **Solution 1 for limitation 1**: repeating the cycle of interviewing the expert.

- **Limitation raised form Solution 1**: A knowledge acquisition bottleneck results from the time-consuming and labor intensive task of building an expert system.
Early Expert Systems

- **DENDRAL** – used in chemical mass spectroscopy to identify chemical constituents
- **MYCIN** – medical diagnosis of illness
- **DIPMETER** – geological data analysis for oil
- **PROSPECTOR** – geological data analysis for minerals
- **XCON/R1** – configuring computer systems
## Broad Classes of Expert Systems

<table>
<thead>
<tr>
<th>Class</th>
<th>General Area</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuration</td>
<td>Assemble proper components of a system in the proper way.</td>
</tr>
<tr>
<td>Diagnosis</td>
<td>Infer underlying problems based on observed evidence.</td>
</tr>
<tr>
<td>Instruction</td>
<td>Intelligent teaching so that a student can ask why, how, and what if questions just as if a human were teaching.</td>
</tr>
<tr>
<td>Interpretation</td>
<td>Explain observed data.</td>
</tr>
<tr>
<td>Monitoring</td>
<td>Compares observed data to expected data to judge performance.</td>
</tr>
<tr>
<td>Planning</td>
<td>Devise actions to yield a desired outcome.</td>
</tr>
<tr>
<td>Prognosis</td>
<td>Predict the outcome of a given situation.</td>
</tr>
<tr>
<td>Remedy</td>
<td>Prescribe treatment for a problem.</td>
</tr>
<tr>
<td>Control</td>
<td>Regulate a process. May require interpretation, diagnosis, monitoring, planning, prognosis, and remedies.</td>
</tr>
</tbody>
</table>
Problems with Algorithmic Solutions

- Conventional computer programs generally solve problems having algorithmic solutions.

- Algorithmic languages include C, Java, and C#.

- Classic AI languages include LISP and PROLOG.
Considerations for Building Expert Systems

- Can the problem be solved effectively by conventional programming?
- Is there a need and a desire for an expert system?
- Is there at least one human expert who is willing to cooperate?
- Can the expert explain the knowledge to the knowledge engineer in a way that can understand it.
- Is the problem-solving knowledge mainly heuristic and uncertain?
Languages, Shells, and Tools

- Expert system languages are post-third generation.
- Expert system languages (e.g. CLIPS) focus on ways to represent knowledge.
- Tool = language + utility program (code generator, graphics editor, debuggers, etc.).
- Shell: is a special purpose tool designed for certain types of applications in which the user must supply the knowledge base. Example, EMYCIN (empty MYCIN)
Elements of an Expert System

- User interface – mechanism by which user and system communicate.
- Exploration facility – explains reasoning of expert system to user.
- Working memory – global database of facts used by rules.
- Inference engine – makes inferences deciding which rules are satisfied and prioritizing.
Elements Continued

- Agenda – a prioritized list of rules created by the inference engine, whose patterns are satisfied by facts or objects in working memory.
- Knowledge acquisition facility – automatic way for the user to enter knowledge in the system bypassing the explicit coding by knowledge engineer.
Structure of a Rule-Based Expert System
Production Rules

• Knowledge base is also called production memory.

• Production rules can be expressed in IF-THEN pseudocode format.

• In rule-based systems, the inference engine determines which rule antecedents are satisfied by the facts.
Inference engine operates on recognize-act cycle

While not done
  conflict resolution:
    act:
    match:
    check for halt:
End-while
Inference engine operates on recognize-act cycle

- conflict resolution: if there are activations then select the one with the highest priority. Else done.
- act: sequentially perform the actions. Update the working memory. Remove the fired activations.
- match: Update the agenda by checking if there are activation or remove activations if there LHS is no longer satisfied.
- check for halt: if an halt action is performed or break command given, then done.
General Methods of Inferencing

- Forward chaining – reasoning from facts to the conclusions resulting from those facts – best for prognosis, monitoring, and control.

- Backward chaining – reasoning in reverse from a hypothesis, a potential conclusion to be proved to the facts that support the hypothesis – best for diagnosis problems.
Production Systems

• Rule-based expert systems – most popular type today.
• Knowledge is represented as multiple rules that specify what should/not be concluded from different situations.
• Forward chaining – start w/facts and use rules do draw conclusions/take actions.
• Backward chaining – start w/hypothesis and look for rules that allow hypothesis to be proven true.
Post Production System

• Basic idea – any mathematical / logical system is simply a set of rules specifying how to change one string of symbols into another string of symbols.

• Basic limitation – lack of control mechanism to guide the application of the rules.
What are Expert Systems?

Can be considered declarative languages:

- Programmer does not specify how to achieve a goal at the algorithm level.

- Induction-based programming – the program learns by generalizing from a sample.