CHAPTER-1

1

Expert Systems

Grading:

- Midterm Exam %25
- Project/Assignments/Quizzes %50
- Final Exam %25

Textbook:

Joseph Giarratano and Gary Riley.
Expert Systems: Principles and Programming.
3rd edition, PWS Publishing, Boston, MA,1998.

2

Course Topics

1.Introduction
2.CLIPS ES shell:
Pattern Matching, Variables, Functions, Expressions, Constraints
Templates, Facts, Rules, Salience; Inference Engine
3.Knowledge Representation Methods:
Production Rules, Semantic Nets, Schemata and Frames, Logic
4.Reasoning and Inference:
Predicate Logic, Inference Methods, Resolution
Forward-chaining, Backward-chaining
5.Reasoning with Uncertainty:
Probability, Bayesian Decision Making
6.Approximate / Fuzzy Reasoning
7.Expert System Design
8. Expert System Examples

3

Project Groups

- Each group will contain 2 students.
- Groups will find their own topics.

- At the end of semester, submit only a diskette containing:

1) Project report document (5-8 pages)
2) Source code

4

Possible Project Topics

Bacterial Infections Diagnosis
Car Repair System
Tutorial System for Teaching English
Television Malfunction Diagnosis
Refrigerator Malfunction Diagnosis
Fire Emergency System
Earthquake Emergency System
Intelligent Information Discovery
Knowledge Discovery
Data Mining .... (Others)

5

What is an Expert System (ES)?

- Giarratano & Riley:
  A computer system that emulates the decision-making ability of a human expert in a restricted domain.

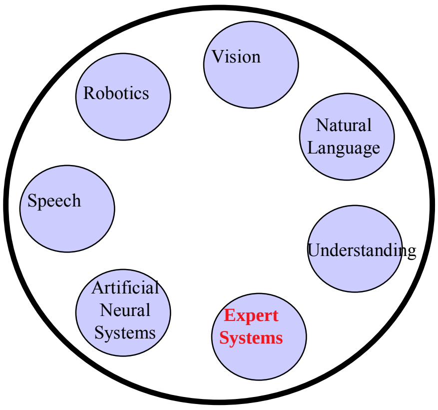
- Edward Feigenbaum:
  An intelligent computer program that uses knowledge and inference procedures to solve problems that are difficult enough to require significant human expertise for their solutions.

The term *Knowledge-Based System (KBS)* is often used synonymously.

6

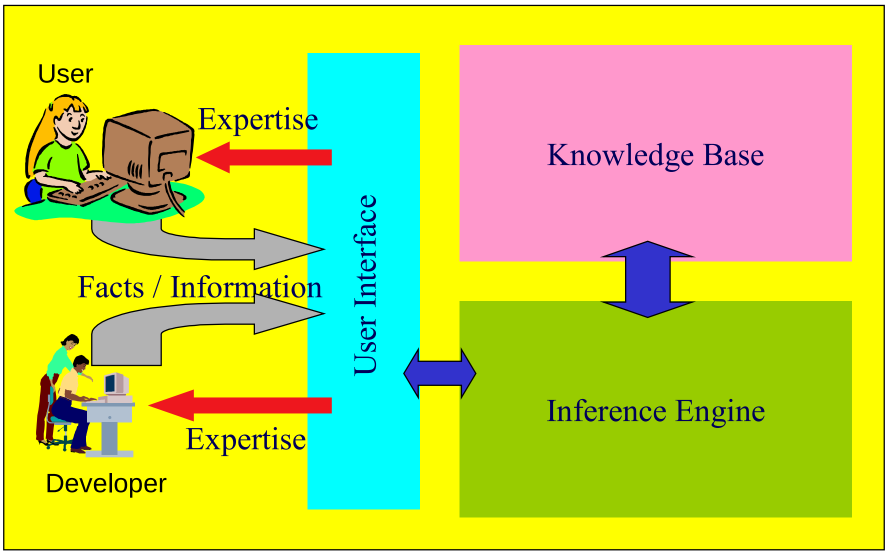
Some areas of Artificial Intelligence

Computing Intelligence
- Expert Systems
- Soft computing
- AI sub-areas



7

Main Components of an ES



8

Main Components of ES

- knowledge base
  - contains essential information about the problem domain
  - often represented as facts and rules
- inference engine
  - mechanism to derive new knowledge from the knowledge base and the information provided by the user
  - often based on the use of rules
- user interface
  - interaction with end users
  - development and maintenance of the knowledge base

9

General Concepts of ES

- knowledge acquisition (knowledge elicitation)
  - transfer of knowledge from humans to computers
  - sometimes knowledge can be acquired directly from the environment
    - machine learning
- knowledge representation
  - storing and processing knowledge in computers
- inference
  - mechanism that allows the generation of new conclusions from existing knowledge in a computer
- explanation
  - illustrates to the user how and why a particular solution was generated

10

History of ES

- strongly influenced by cognitive science and mathematics (Newell & Simon)
  - the way humans solve problems
  - formal foundations, especially logic and inference
- production rules as representation mechanism
  - `IF … THEN` type rules
  - reasonably close to human reasoning
  - can be manipulated by computers
  - knowledge "chunks" are manageable both for humans and for computers

11

Application Areas 1

Domain	General area
– Configuration	– Assemble components of a system in the proper way
– Diagnosis	– Infer underlying problems based on observed evidence
– Instruction	– Intelligent teaching so that a student can ask Why, How, What if, questions as if a human was teaching.
– Interpretation	– Explain observed data

12

Application Areas 2

Domain	General area
– Monitoring	– Compares observed data to expected data to judge performance
– Planning	– Devises actions to yield a desired outcome
– Prognosis	– Predict the outcome of a given situation –
– Remedy	– Prescribe treatment for a problem
– Control	– Regulate a process - may require most of the above.

13

When Not to Use ES

- expert systems are not suitable for all types of domains and tasks
  - conventional algorithms are known and efficient
  - the main challenge is computation, not knowledge
  - knowledge cannot be captured easily
  - users may be reluctant to apply an expert system to a critical task

14

How to decide appropriate domain?

- Can the problem be solved by conventional programming?
- Is the domain well bounded?
- Is there a need for an expert system?
- Is there at least one human expert willing to help?
- Can the expert explain his knowledge so that the knowledge engineer can understand it?
- Is the knowledge mainly heuristic & uncertain?

15

Differences between expert systems and conventional programs 1

Characteristic	Conventional Program	Expert System
Control by …	Statement order	Inference engine
Control & Data	Implicit integration	Explicit separation
Control Strength	Strong	Weak
Solution by …	Algorithm	Rules & Inference
Solution search	Small or none	Large
Problem solving	Algorithm	Rules

16

Differences between expert systems and conventional programs 2

Characteristic	Conventional Program	Expert system
Input	Assumed correct	Incomplete, incorrect
Unexpected input	Difficult to deal with	Very responsive
Output	Always correct	Varies with the problem
Explanation	None	Usually
Applications	Numeric, file & text	Symbolic reasoning
Execution	Generally sequential	Opportunistic rules

17

Differences between expert systems and conventional programs 3

Characteristic	Conventional Program	Expert System
Program Design	Structured design	Little or no structure
Modifiability	Difficult	Reasonable
Expansion	Done in major lumps	Incremental

18

Examples of Commercial Expert Systems

- •XCON/R1
  - –configuration of DEC VAX computer systems
- •MYCIN
  - –diagnosis of illnesses
- •PROSPECTOR
  - –analysis of geological data for minerals
  - –discovered a mineral deposit
- •DENDRAL
  - –identification of chemical constituents

Others:

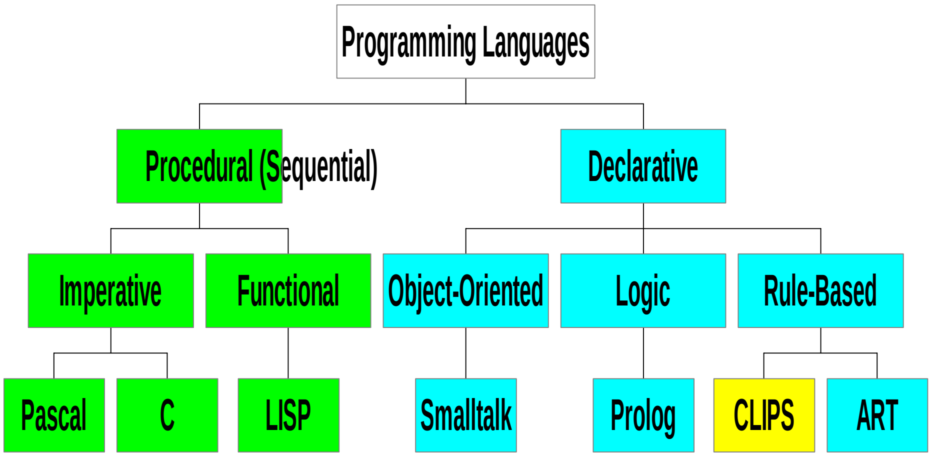
- CRYSALYS
- MOLGEN
- ACE
- MUD
- TEIRESIAS
- HEARSAY
- COMPASS
- ONCOCIN

19

ES Tools

- ES languages
  - higher-level languages specifically designed for knowledge representation and reasoning
  - SAIL, KRL, KQML, DAML
- ES shells
  - an ES development tool/environment where the user provides the knowledge base
  - separation of knowledge and inference
  - allows the re-use of the “machinery” for different domains
  - CLIPS, JESS, Mycin, Babylon

20



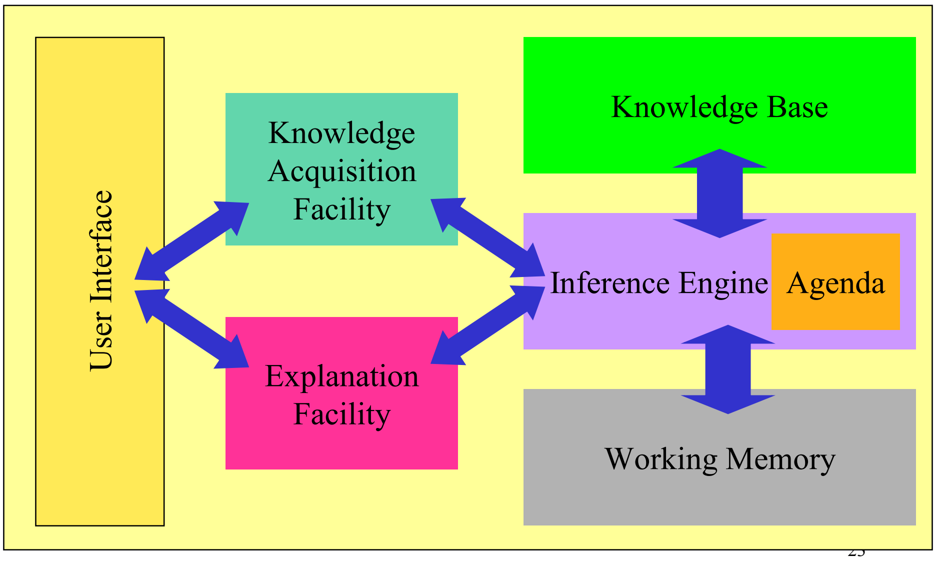
21

ES Elements

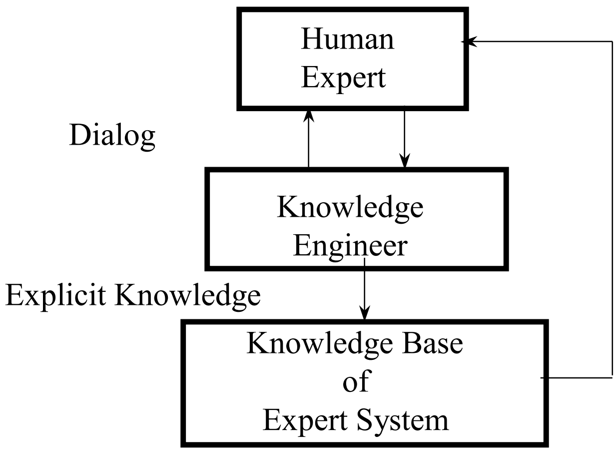
- knowledge base
- inference engine
- working memory
- agenda
- explanation facility
- knowledge acquisition facility
- user interface

22

ES Structure



Development of an Expert System



Rule-Based ES

- knowledge is encoded as **`IF … THEN`** rules
  - these rules can also be written as *production rules*
- the inference engine determines which rule antecedents are satisfied
  - the left-hand side must "match" a fact in the working memory
- satisfied rules are placed on the agenda
- rules on the agenda can be activated ("fired")
  - an activated rule may generate new facts through its right-hand side
  - the activation of one rule may subsequently cause the activation of other rules

25

Example Rules

IF … THEN Rules

```
Rule: Red_Light
  IF      the light is red
  THEN    stop
Rule: Green_Light
  IF      the light is green
  THEN    go
```

antecedent (left-hand-side)

consequent (right-hand-side)

Production Rules

```
the light is red ==> stop
the light is green ==> go
```

antecedent (left-hand-side)

consequent (right-hand-side)

Example Rule

Human-Readable Format

IF the stain of the organism is gram negative
AND the morphology of the organism is rod
AND the aerobiocity of the organism is gram anaerobic
THEN there is strongly suggestive evidence (0.8)
that the class of the organism is enterobacteriaceae

MYCIN Format

```
IF      (AND (SAME CNTEXT GRAM GRAMNEG)
             (SAME CNTEXT MORPH ROD)
             (SAME CNTEXT AIR AEROBIC)
THEN (CONCLUDE CNTEXT CLASS ENTEROBACTERIACEAE TALLY .8)
```

27

Inference Engine Cycle

- The inference engine determines the execution of the rules by the following cycle:
  - conflict resolution
    - select the rule with the highest priority from the agenda
  - execution (Act)
    - perform the actions on the consequent of the selected rule
    - remove the rule from the agenda
  - match
    - update the agenda
      - add rules whose antecedents are satisfied to the agenda
      - remove rules with non-satisfied agendas
- the cycle ends when no more rules are on the agenda, or when an explicit stop command is encountered

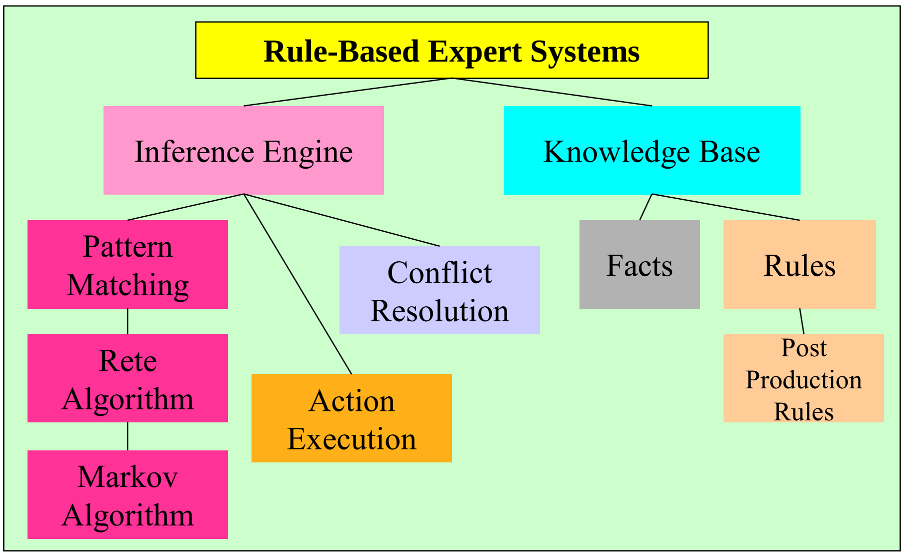
28

Forward and Backward Chaining

- different methods of rule activation
  - forward chaining (data-driven)
    - reasoning from facts to the conclusion
    - as soon as facts are available, they are used to match antecedents of rules
    - a rule can be activated if all parts of the antecedent are satisfied
    - often used for real-time expert systems in monitoring and control
    - examples: CLIPS, OPS5
  - backward chaining (query-driven)
    - starting from a hypothesis (query), supporting rules and facts are sought until all parts of the antecedent of the hypothesis are satisfied
    - often used in diagnostic and consultation systems
    - examples: EMYCIN (Empty MYCIN)

29

Foundations of Expert Systems



30

Post Production Systems

- production rules were used by the logician Emil L. Post in the early 40s in symbolic logic
- Post's theoretical results
  - any system of mathematics or logic could be represented by production rule system
- basic principle of production rules
  - a set of rules governs the conversion of a set of strings into another set of strings
    - these rules are also known as rewrite rules
    - simple syntactic string manipulation
    - no understanding or interpretation is required
    - also used to define grammars of languages–e.g. BNF grammars of programming languages
    - no control strategy

Production Systems (cont.)

- Markov algorithms (1954)
  - *ordered* group of productions
  - termination on: (1) last production not applicable to a string, or (2) production ending with period applied
  - can be applied to substrings, beginning at left
  - Features: null string = ^; single-char vars (a,b,etc.); Greek letters = punctuation

Markov Algorithm Example

(1) $\alpha xy \rightarrow y\alpha x$
(2) $\alpha \rightarrow$ ^
(3) ^ $\rightarrow \alpha$

Rule	Success or Failure	String
1	F	ABC
2	F	ABC
3	S	α ABC
1	S	B α AC
1	S	BC α A
1	F	BC α A
2	S	BCA

Table 1.11 Execution Trace of a Markov Algorithm

Production Systems (cont.)

- Markov
  - too inefficient to be used with many rules
- Rete
  - Charles Forgy--Carnegie-Mellon Univ. (1979)
  - fast pattern matcher
  - looks only for changes in matches (ignores static data)

Procedural vs. Non-procedural Languages

- Procedural
  - programmer must specify *exactly* how the problem is to be solved
- Non-procedural
  - programmer specifies the *goal*

Procedural Languages

- Imperative
  - statements are commands
  - rigid control structure
  - top-down design
  - not efficient symbol manipulators
- Functional
  - function-based (association, domain, co-domain); $f : S \rightarrow T$
  - bottom-up

LISP

- Leading AI language
  - symbolic expressions (lists or atoms)
  - primitives (CAR, CDR, etc.)
  - predicates

Function	**Predicates**
QUOTE	ATOM
CAR	EQ
CDR	NULL
CPR	
CTR	
CONS	
EVAL	
COND	
LAMBDA	
DEFINE	
LABEL	

Table 1.12 Original LISP Primitives and Functions

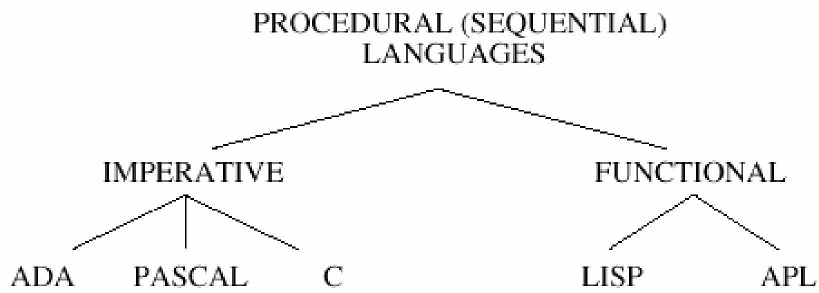


Figure 1.8 Procedural Languages

Non-procedural Languages

- Declarative: seperated the goal from the methods used to achive it
- Object-oriented
  - design vs. programming
- Logic
  - theorem proving
- Expert Systems (declarative)
- Induction-based: (Non declarative) program learns by examples

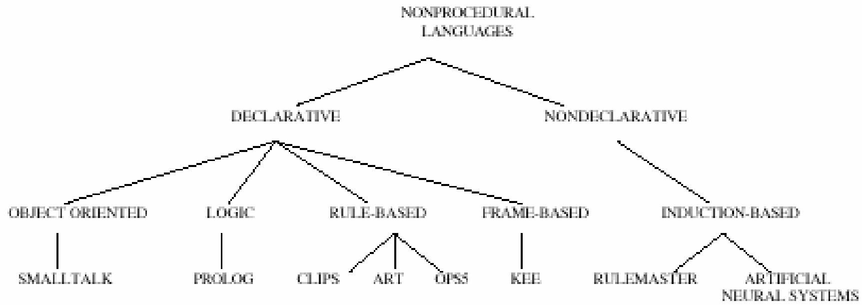


Figure 1.9 Nonprocedural Languages

Characteristic	Conventional Program	Expert System
Control by ...	Statement order	Inference engine
Control and data	Implicit integration	Explicit separation
Control Strength	Strong	Weak
Solution by ...	Algorithm	Rules and inference
Solution search	Small or none	Large
Problem solving	Algorithm is correct	Rules
Input	Assumed correct	Incomplete, incorrect
Unexpected input	Difficult to deal with	Very responsive
Output	Always correct	Varies with problem
Explanation	None	Usually
Applications	Numeric, file, and text	Symbolic reasoning
Execution	Generally sequential	Opportunistic rules
Program design	Structured design	Little or no structure
Modifiability	Difficult	Reasonable
Expansion	Done in major jumps	Incremental

Table 1.13 Some Typical Differences between Conventional Programs and Expert Systems

Artificial Neural Systems

- Connectionism
- Real-time response to complex pattern recognition problems
- Analog computer with simple processing elements
- Element weights--key

Number of Cities	Routes
1	1
2	1–2–1
3	1–2–3–1
	1–3–2–1
4	1–2–3–4–1
	1–2–4–3–1
	1–3–2–4–1
	1–3–4–2–1
	1–4–2–3–1
	1–4–3–2–1

Table 1.14 Traveling Salesman Problem Routes

$$I \equiv \text{Neuron Input}_i = \sum_j W_{ij} I_j$$

$$\text{Neuron Output} = \frac{1}{1 + e^{-(I-\theta)}}$$

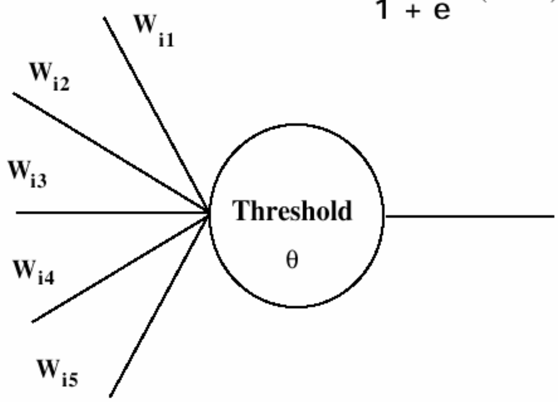


Figure 1.10 Neuron Processing Element

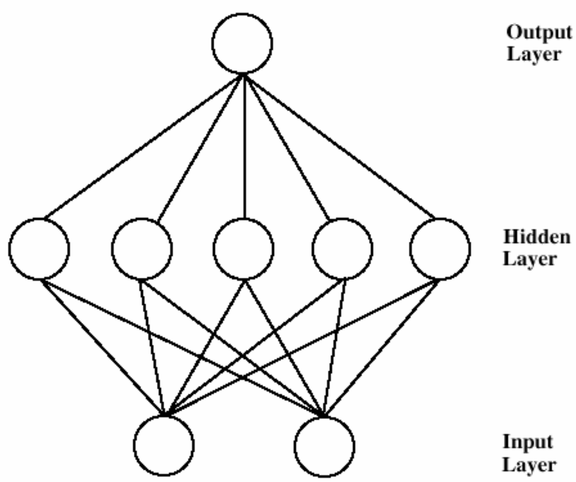


Figure 1.11 A Back-propagation Net

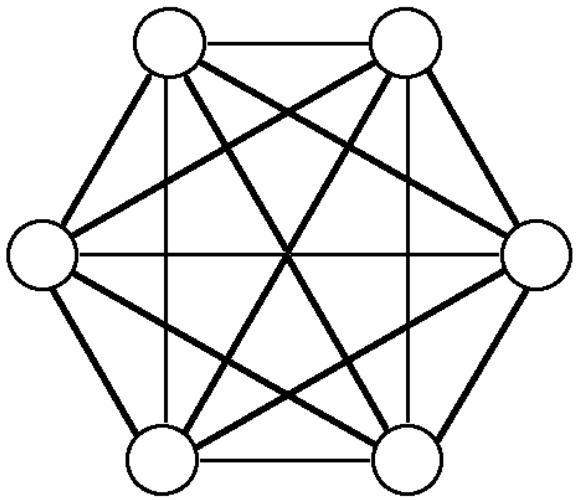


Figure 1.12 Hopfield Artificial Neural Net

Connectionist ES

- Use ANS to build ES
- ANS --> knowledge base constructed by training examples
- Add explanation capability to ANS
- Inductive learning to reduce knowledge acquisition bottleneck

Class	General Area
Configuration	Assemble proper components of a system in the proper way.
Diagnosis	Infer underlying problems based on observed evidence.
Instruction	Intelligent teaching so that a student can ask *Why*, *How* and*What If* type questions just as if a human was teaching.
Interpretation	Explain observed data.
Monitoring	Compares observed data to expected data to judge performance.
Planning	Devise actions to yield a desired outcome.
Prognosis	Predict the outcome of a given situation.
Remedy	Prescribe treatment for a problem.
Control	Regulate a process. May require interpretation, diagnosis,monitoring, planning, prognosis, and remedies.

Table 1.3 Broad Classes of Expert Systems

Name	Chemistry
CRYSALIS	Interpret a protein's 3-D structure
DENDRAL	Interpret molecular structure
TQMSTUNE	Remedy Triple Quadruple Mass Spectrometer (keep it tuned)
CLONER	Design new biological molecules
MOLGEN	Design gene-cloning experiments
SECS	Design complex organic molecules
SPEX	Plan molecular biology experiments

Table 1.4 Chemistry Expert Systems

Name	Electronics
ACE	Diagnosis telephone network faults
IN-ATE	Diagnosis oscilloscope faults
NDS	Diagnose national communication net
EURISKO	Design 3-D microelectronics
PALLADIO	Design and test new VLSI circuits
REDESIGN	Redesign digital circuits to new
CADHELP	Instruct for computer aided design
SOPHIE	Instruct circuit fault diagnosis

Table 1.5 Electronics Expert Systems

Name	Medicine
PUFF	Diagnosis lung disease
VM	Monitors intensive-care patients
ABEL	Diagnosis acid-base/electrolytes
AI/COAG	Diagnosis blood disease
AI/RHEUM	Diagnosis rheumatoid disease
CADUCEUS	Diagnosis internal medicine disease
ANNA	Monitor digitalis therapy
BLUE BOX	Diagnosis/remedy depression
MYCIN	Diagnosis/remedy bacterial infections
ONCOCIN	Remedy/manage chemotherapy patients
ATTENDING	Instruct in anesthetic management
GUIDON	Instruct in bacterial infections

Table 1.6 Medical Expert Systems

Name	Engineering
REACTOR	Diagnosis/remedy reactor accidents
DELTA	Diagnosis/remedy GE locomotives
STEAMER	Instruct operation - steam powerplant

Table 1.7 Engineering Expert Systems

Name	Geology
DIPMETER	Interpret dipmeter logs
LITHO	Interpret oil well log data
MUD	Diagnosis/remedy drilling problems
PROSPECTOR	Interpret geologic data for minerals

Table 1.8 Geology Expert Systems

Name	Computer Systems
PTRANS	Prognosis for managing DEC computers
BDS	Diagnosis bad parts in switching net
XCON	Configure DEC computer systems
XSEL	Configure DEC computer sales order
XSITE	Configure customer site for DEC computers
YES/MVS	Monitor/control IBM MVS operating system
TIMM	Diagnosis DEC computers

Table 1.9 Computer Expert Systems

Advantages of ES

- economical
  - lower cost per user
- availability
  - accessible anytime, almost anywhere
- response time
  - often faster than human experts
- reliability
  - can be greater than that of human experts
- explanation
  - reasoning steps that lead to a particular conclusion

Disadvantages of ES

- limited knowledge
  - “shallow” knowledge
    - no “deep” understanding of the concepts and their relationships
  - no “common-sense” knowledge
  - no knowledge from possibly relevant related domains
  - “closed world”
    - ES knows only what it has been explicitly “told”
    - it doesn’t know what it doesn’t know
- mechanical reasoning
  - may not have or select the most appropriate method for a particular problem
  - some “easy” problems are computationally very expensive