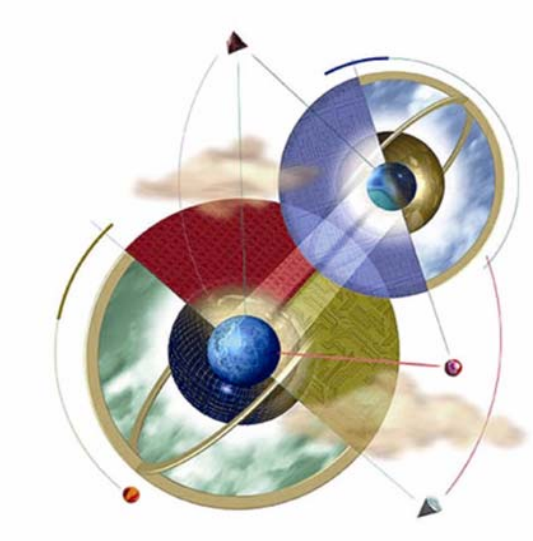


Chapter 2: The Representation of Knowledge

Expert Systems: Principles and Programming, Fourth Edition

Objectives

- Introduce the study of logic
- Learn the difference between formal logic and informal logic
- Learn the meaning of knowledge and how it can be represented
- Learn about semantic nets
- Learn about object-attribute-value triples

Objectives Continued

- See how semantic nets can be translated into Prolog
- Explore the limitations of semantic nets
- Learn about schemas
- Learn about frames and their limitations
- Learn how to use logic and set symbols to represent knowledge

Objectives Continued

- Learn about propositional and first order predicate logic
- Learn about quantifiers
- Explore the limitations of propositional and predicate logic

What is the study of logic?

- Logic is the study of making inferences – given a set of facts, we attempt to reach a true conclusion.
- An example of informal logic is a courtroom setting where lawyers make a series of inferences hoping to convince a jury / judge .
- Formal logic (symbolic logic) is a more rigorous approach to proving a conclusion to be true / false.

Why is Logic Important

- We use logic in our everyday lives – “should I buy this car”, “should I seek medical attention”.
- People are not very good at reasoning because they often fail to separate word meanings with the reasoning process itself.
- Semantics refers to the meanings we give to symbols.

The Goal of Expert Systems

- We need to be able to separate the actual meanings of words with the reasoning process itself.
- We need to make inferences w/o relying on semantics.
- We need to reach valid conclusions based on facts only.

Knowledge in Expert Systems

- Knowledge representation is key to the success of expert systems.
- Expert systems are designed for knowledge representation based on rules of logic called inferences.
- Knowledge affects the development, efficiency, speed, and maintenance of the system.

Definitions of Knowledge

a)

(1) the fact or condition of knowing something with familiarity gained through experience or association (2)acquaintance with or understanding of a science, art, or technique

b)

(1) the fact or condition of being aware of something

(2) the range of one's information or understanding

c) the circumstance or condition of apprehending truth or fact through reasoning : cognition

d) the fact or condition of having information or of being learned

Epistemology

- Epistemology is the formal study of knowledge .
- Concerned with nature, structure, and origins of knowledge.

Categories of Epistemology

- Philosophy
- A priori
- A posteriori
- Procedural
- Declarative
- Tacit

A Priori Knowledge

- Also called “theoretical knowledge”
- “That which precedes”
- Independent of the senses
- Universally true
- Cannot be denied without contradiction
- e.g., coin flips will give 50% heads and 50% tails

A Posteriori Knowledge

- Also called “empirical knowledge”
- “That which follows”
- Derived from the senses
- Now always reliable
- Deniable on the basis of new knowledge w/o the necessity of contradiction
- E.g., 100 coin flips give only 39 heads – what can you conclude?

Procedural Knowledge

Knowing how to do something:

- Fix a watch
- Install a window
- Brush your teeth
- Ride a bicycle

Declarative Knowledge

- Knowledge that something is true or false
- Usually associated with declarative statements
- E.g., “Don’t touch that hot wire.”

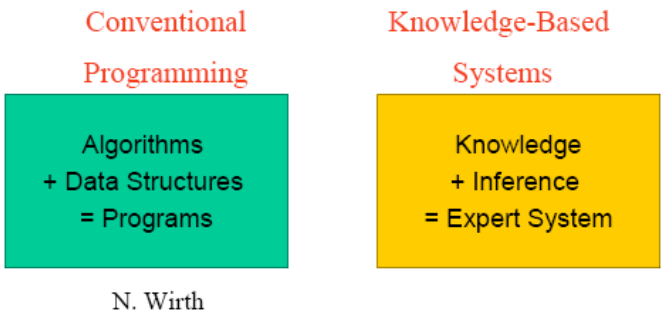
Tacit Knowledge

- Unconscious knowledge
- Cannot be expressed by language
- E.g., knowing how to walk, breath, etc.

Knowledge in Rule-Based Systems

- Knowledge is part of a hierarchy.
- Knowledge refers to rules that are activated by facts or other rules.
- Activated rules produce new facts or conclusions.
- Conclusions are the end-product of inferences when done according to formal rules.

Knowledge in Rule-Based Systems II



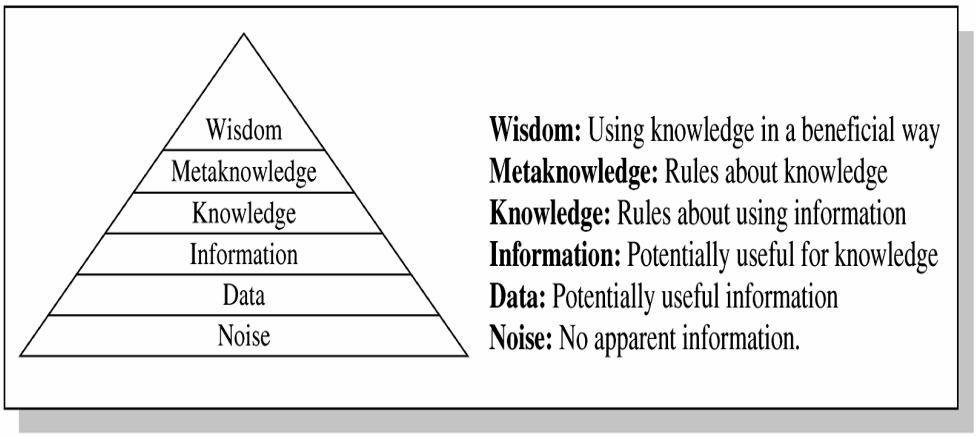
Expert Systems vs. ANS

- ANS does not make inferences but searches for underlying patterns.
- Expert systems
  - **Draw inferences using facts**
  - **Separate data from noise**
  - **Transform data into information**
  - **Transform information into knowledge**

Metaknowledge

- Metaknowledge is knowledge about knowledge and expertise.
- Most successful expert systems are restricted to as small a domain as possible.
- In an expert system, an ontology is the metaknowledge that describes everything known about the problem domain.
- Wisdom is the metaknowledge of determining the best goals of life and how to obtain them.

Figure 2.2 The Pyramid of Knowledge



Knowledge Representation Methods

A number of knowledge-representation techniques have been devised:

- Production Rules
- Semantic nets
- Frames
- Scripts
- Logic
- Conceptual graphs

Production Rules

- Frequently used to formulate the knowledge in expert systems.
- A formal variation is Backus-Naur form (BNF)
  - metalanguage for the definition of language syntax
  - a *grammar* is a complete, unambiguous set of production rules for a specific language
  - a *parse tree* is a graphic representation of a sentence in that language
  - provides only a syntactic description of the language
    - not all sentences make sense

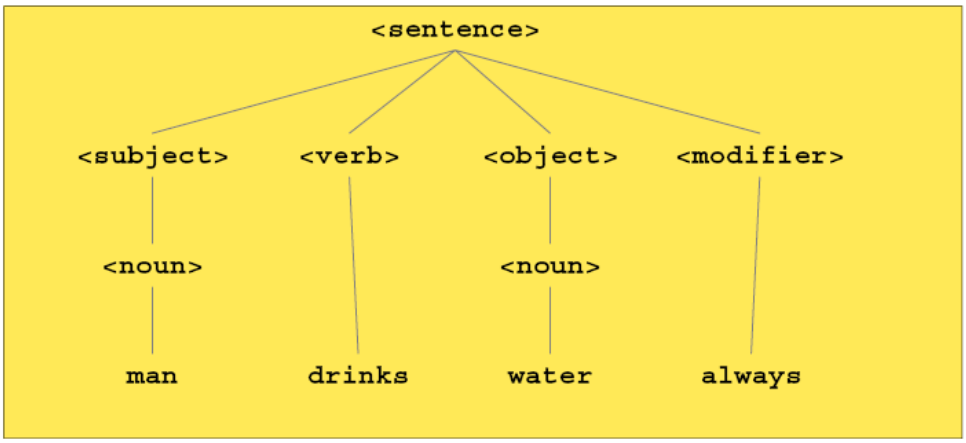
Example: Production Rules

(for a subset of the English language)

```
Grammar
<sentence> -> <subject> <verb> <object> <modifier>
<subject> -> <noun>
<object> -> <noun>

Lexicon
<noun> -> man | cat | water
<verb> -> drinks | walks
<modifier> -> always | sometimes
```

Example: Parse Tree of a Sentence

- Example sentence:
  Man drinks water always.



Example in CLIPS: This program produces all possible sentences according to the production rules (i.e. grammar), although most of them will not be meaningful.

```
(defrule sentence-rule
   (subj ?s)
   (verb ?v)
   (obj ?o)
   (adverb ?d)
=>
   (assert (sentence ?s ?v ?o ?d)))

(defrule subject-rule
   (noun ?n)
=>
   (assert (subj ?n)))

(defrule object-rule
   (noun ?n)
=>
   (assert (obj ?n)))
```

```
(deffacts lexicon
   (noun man)
   (noun cat)
   (noun water)
   (verb drinks)
   (verb walks)
   (adverb always)
   (adverb sometimes))
```

```
CLIPS> (dribble-on)
CLIPS> (reset)
CLIPS> (run)
CLIPS> (facts)
f-0     (initial-fact)
f-1     (noun man)
f-2     (noun cat)
f-3     (noun water)
f-4     (verb drinks)
f-5     (verb walks)
f-6     (adverb always)
f-7     (adverb sometimes)
f-8     (subj water)
f-9     (obj water)
f-10    (sentence water drinks water sometimes)
f-11    (sentence water drinks water always)
f-12    (sentence water walks water sometimes)
f-13    (sentence water walks water always)
f-14    (subj cat)
f-15    (sentence cat walks water sometimes)
f-16    (sentence cat walks water always)
f-17    (sentence cat drinks water sometimes)
f-18    (sentence cat drinks water always)
f-19    (obj cat)
f-20    (sentence water drinks cat sometimes)
f-21    (sentence water drinks cat always)
f-22    (sentence water walks cat sometimes)
f-23    (sentence water walks cat always)
f-24    (sentence cat drinks cat sometimes)
f-25    (sentence cat drinks cat always)
f-26    (sentence cat walks cat sometimes)
f-27    (sentence cat walks cat always)
f-28    (subj man)
f-29    (sentence man walks cat sometimes)
f-30    (sentence man walks cat always)
f-31    (sentence man walks water sometimes)
f-32    (sentence man walks water always)
f-33    (sentence man drinks cat sometimes)
f-34    (sentence man drinks cat always)
f-35    (sentence man drinks water sometimes)
f-36    (sentence man drinks water always)
f-37    (obj man)
f-38    (sentence water drinks man sometimes)
f-39    (sentence water drinks man always)
f-40    (sentence water walks man sometimes)
f-41    (sentence water walks man always)
f-42    (sentence cat drinks man sometimes)
f-43    (sentence cat drinks man always)
f-44    (sentence cat walks man sometimes)
f-45    (sentence cat walks man always)
f-46    (sentence man drinks man sometimes)
f-47    (sentence man drinks man always)
f-48    (sentence man walks man sometimes)
f-49    (sentence man walks man always)
For a total of 50 facts.
CLIPS> (dribble-off)
```


Advantages and Disadvantages of Production Rules

Advantages:

- simple and easy to understand
- straightforward implementation
- formal foundations for some variants

Disadvantages:

- simple implementations are very inefficient
- some types of knowledge are not easily expressed in such rules
- large sets of rules become difficult to understand and maintain

Semantic Nets

- A classic representation technique for propositional information (sometimes called propositional net)
- Propositions – a form of declarative knowledge, stating facts (true/false)
- Propositions are called “atoms” – cannot be further subdivided.
- Semantic nets consist of nodes (objects, concepts, situations) and arcs or links (relationships between them).
- For nodes
  - Labels indicate the name
  - Nodes can be instances (individual objects) or classes (generic nodes)

Links-Semantic Nets

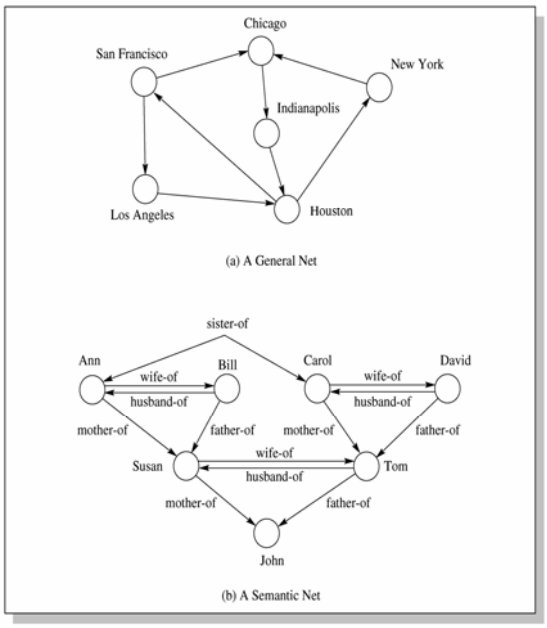
Links represent relationships

- The relationships contain the structural information of the knowledge to be represented
- The label indicates the type of the relationship

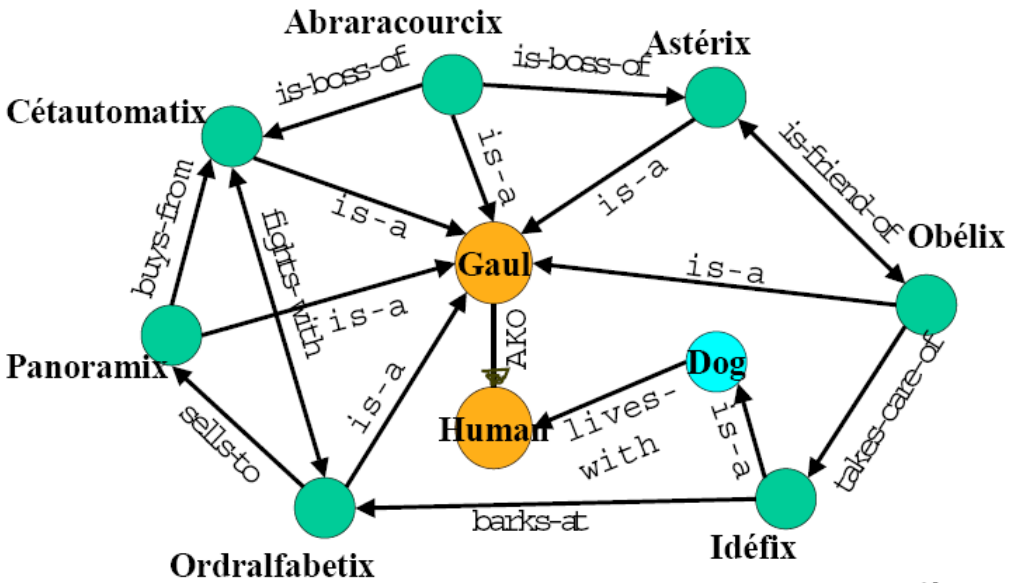
Common types of links:

- IS-A – relates an instance or individual to a generic class
- A-KIND-OF – relates generic nodes to generic nodes

Figure 2.4 Two Types of Nets



Semantic Net Example



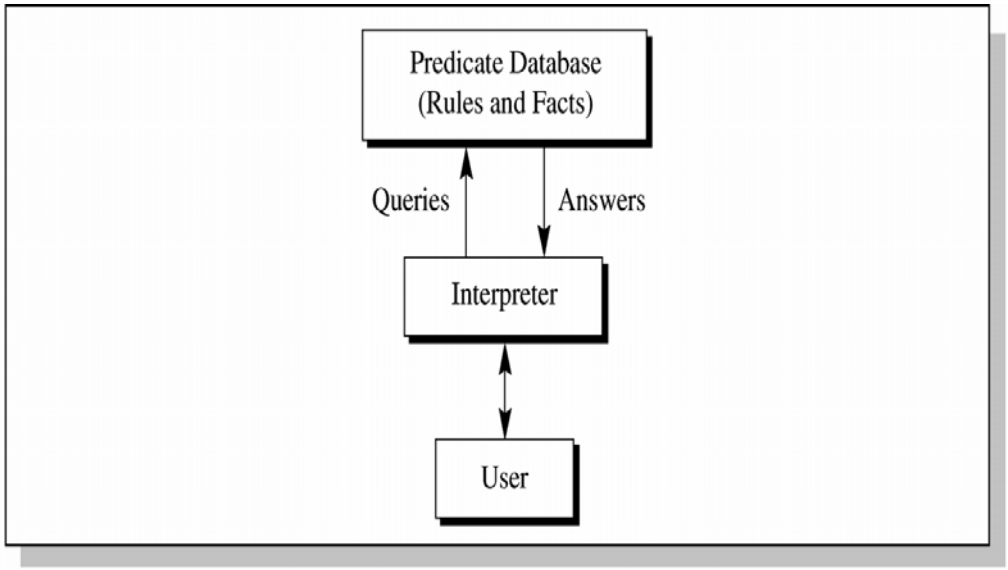
Example in CLIPS

```
(deffacts  initial-facts-of-semantic-net
  (is-a           Asterix         Gaul)
  (is-a           Obelix          Gaul)
  (is-a           Abraracourcix   Gaul)
  (is-a           Cetautomatix    Gaul)
  (is-a           Panoramix       Gaul)
  (is-a           Ordralfabetix   Gaul)
  (is-a           Idefix          Dog)
  (lives-with     Dog             Human)
  (AKO            Gaul            Human)
  (is-boss-of     Abraracourcix   Asterix)
  (is-boss-of     Abraracourcix   Cetautomatix)
  (is-friend-of   Asterix         Obelix)
  (buys-from      Panoramix       Cetautomatix)
  (sells-to       Ordralfabetix   Panoramix)
  (fights-with    Cetautomatix    Ordralfabetix)
  (takes-care-of  Obelix          Idefix)
  (barks-at       Idefix          Ordralfabetix)
)
```



```
(defrule humans
  (is-a  ?name  Gaul)
 =>
  (assert (AKO ?name  Human)))

(defrule mutual-friends
  (is-friend-of  ?name1  ?name2)
 =>
  (assert (is-friend-of  ?name2  ?name1)))

(defrule mutual-fighting
  (fights-with  ?name1  ?name2)
 =>
  (assert (fights-with  ?name2  ?name1)))

(defrule shopping1
  (buys-from  ?name1  ?name2)
 =>
  (assert (sells-to  ?name2  ?name1)))

(defrule shopping2
  (sells-to  ?name1  ?name2)
 =>
  (assert (buys-from  ?name2  ?name1)))
```

Figure 2.6: General Organization of a PROLOG System



PROLOG and Semantic Nets

- In PROLOG, predicate expressions consist of the predicate name, followed by zero or more arguments enclosed in parentheses, separated by commas.
- Example:

  mother(becky,heather)

  means that becky is the mother of heather

PROLOG Continued

- Programs consist of facts and rules in the general form of goals.
- General form: `p:- p1, p2, …, pN`

  `p` is called the rule's head and the p_i represents the subgoals
- Example:

```
spouse(x,y) :- wife(x,y)
```

  `x` is the spouse of `y` if `x` is the wife of `y`

Types of Relationships

- relationships can be arbitrarily defined by the knowledge engineer
  - allows great flexibility
  - for reasoning, the inference mechanism must know how relationships can be used to generate new knowledge
    - inference methods may have to be specified for every relationship
- frequently used relationships
  - IS-A
    - relates an instance (individual node) to a class (generic node)
  - AKO (a-kind-of)
    - relates one class (subclass) to another class (superclass)

Objects and Attributes

- attributes provide more detailed information on nodes in a semantic network
  - often expressed as *properties*
    - combination of attribute and value
  - attributes can be expressed as relationships
    - e.g. has-attribute

Implementation Questions

- simple and efficient representation schemes for semantic nets
  - tables that list all objects and their properties
  - tables or linked lists for relationships
- conversion into different representation methods
  - predicate logic
    - nodes correspond variables or constants
    - links correspond to predicates
  - propositional logic
    - nodes and links have to be translated into propositional variables and properly combined with logical connectives

Object-Attribute-Value Triple

- One problem with semantic nets is lack of standard definitions for link names (IS-A, AKO, etc.).
- The OAV triplet can be used to characterize all the knowledge in a semantic net.

Object	Attribute	Value
Astérix	profession	warrior
Obélix	size	extra large
Idéfix	size	petite
Panoramix	wisdom	infinite

Method-1:

```
(deftemplate OAV
        (slot       Object)
        (slot       Attribute)
        (multislot  Value))

(deffacts objects
  (OAV (Object Asterix)    (Attribute profession) (Value warrior))
  (OAV (Object Obelix)     (Attribute size)       (Value extra large))
  (OAV (Object Idefix)     (Attribute size)       (Value petite))
  (OAV (Object Panoramix) (Attribute wisdom)     (Value infinite)))
```

Method-2:

```
(deffacts objects
      (Asterix   profession warrior)
      (Obelix    size       "extra large")
      (Idefix    size       petite)
      (Panoramix wisdom     infinite))
```

Problems with Semantic Nets

- To represent definitive knowledge, the link and node names must be rigorously defined.
  - A solution to this is extensible markup language (XML) and ontologies.
- Problems also include combinatorial explosion of searching nodes, inability to define knowledge the way logic can, and heuristic inadequacy.

Problems with Semantic Nets II

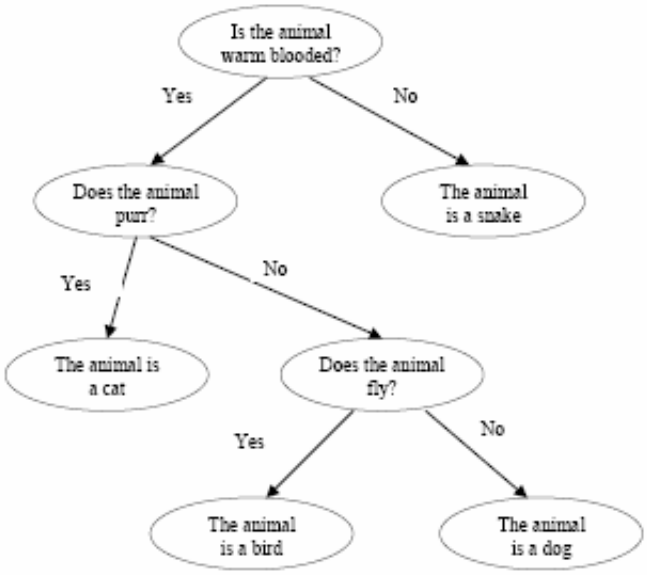
Disadvantages of semantic nets could be classified as:

- Expressiveness
  - no internal structure of nodes
  - relationships between multiple nodes
  - no easy way to represent heuristic information
  - extensions are possible, but cumbersome
  - best suited for binary relationships
- Efficiency
  - may result in large sets of nodes and links
  - search may lead to combinatorial explosion
    - especially for queries with negative results
- Usability
  - lack of standards for link types
  - naming of nodes
    - classes, instances

Decision Trees

- A decision tree is a semantic net in which:
  - Each node is connected to a set of possible answers.
  - Each nonleaf node is connected to a test that splits its set of possible answers into subsets corresponding to different test results.
  - Each branch carries a particular test result's subset to another node.
  - Each leaf represents a possible answer.

Example-1 (Binary decision tree)



SOLUTION-1 (Simple but inefficient)

```
(deffunction ask-user (?question)
  (printout t ?question " (y/n) " )
  (bind ?answer (read))
  (while (and (neq ?answer y) (neq ?answer n)) do
      (printout t "(y/n) ? " )
      (bind ?answer (read)))
  (return ?answer))


(defrule start
=>
 (assert (node1))


(defrule node1
 ?n <- (node1)
=>
 (retract ?n)
 (if (eq (ask-user "Is the animal warm blooded?") n) then
    (printout t "The animal is a snake" crlf)
  else  (assert (node2)) ) )
```



```
(defrule node2
 ?n <- (node2)
=>
 (retract ?n)
 (if (eq (ask-user "Does the animal purr?") y) then
    (printout t "The animal is a cat" crlf)
  else
    (assert (node3)) ) )


(defrule node3
 ?n <- (node3)
=>
 (retract ?n)
 (if (eq (ask-user "Does the animal fly?") y) then
    (printout t "The animal is a bird" crlf)
  else
    (printout t "The animal is a dog" crlf) ) )
```

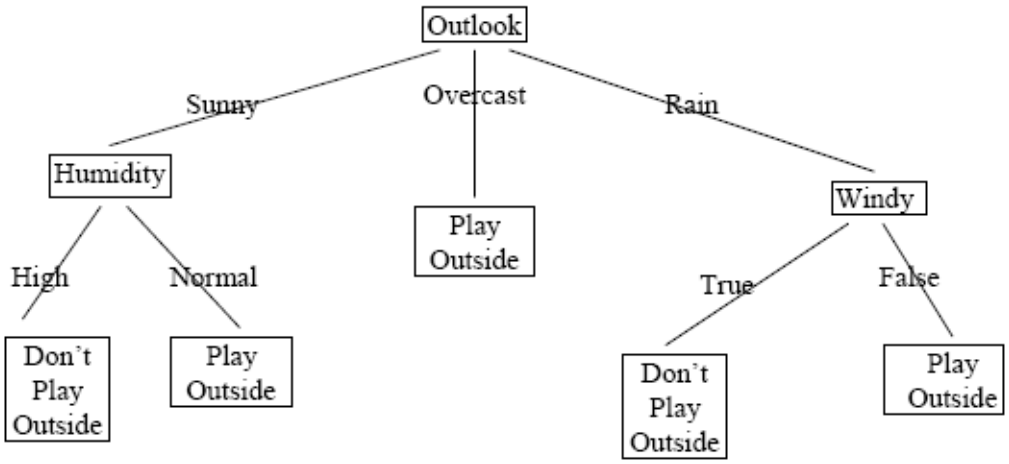
SOLUTION-2 (Generalized and better)

```
(deffunction ask-user (?question)
  (printout t ?question " (y/n) " )
  (bind ?answer (read))
  (while (and (neq ?answer y) (neq ?answer n)) do
    (printout t "(y/n) ? " )
    (bind ?answer (read)))
  (return ?answer))

; "answer" nodes :
;       (node <name> answer <value>)
; "decision" nodes :
;       (node <name> decision <question> <yes-node> <no-node>)
(deffacts decision-tree
  (node root decision "Is the animal warm blooded?" node1 node3)
  (node node1 decision "Does the animal purr?" node4 node2)
  (node node2 decision "Does the animal fly?" node5 node6)
  (node node3 answer "The animal is a snake")
  (node node4 answer "The animal is a cat")
  (node node5 answer "The animal is a bird")
  (node node6 answer "The animal is a dog"))
```



```
(defrule start
=>
 (assert (current-node root)))

(defrule do-decision-node
 ?n <- (current-node ?name)
 (node ?name decision ?question ?yes-branch ?no-branch)
=>
 (retract ?n)
 (if (eq (ask-user ?question) y)
    then (assert (current-node ?yes-branch))
    else (assert (current-node ?no-branch))  ) )

(defrule do-answer-node
 ?n <- (current-node ?name)
 (node ?name answer ?value)
=>
 (retract ?n)
 (printout t ?value crlf))
```

Example-2



Schemata

- Knowledge Structure – an ordered collection of knowledge – not just data.
- Semantic Nets – are shallow knowledge structures – all knowledge is contained in nodes and links.
- Schema is a more complex knowledge structure than a semantic net.
- In a schema, a node is like a record which may contain data, records, and/or pointers to nodes.

Frames

- One type of schema is a frame (or script – time-ordered sequence of frames).
- Frames are useful for simulating commonsense knowledge.
- Semantic nets provide 2-dimensional knowledge; frames provide 3-dimensional.
- Frames represent related knowledge about narrow subjects having much default knowledge.

Frames Continued

- A frame is a group of slots and fillers that defines a stereotypical object that is used to represent generic / specific knowledge.
- Commonsense knowledge is knowledge that is generally known.
- Prototypes are objects possessing all typical characteristics of whatever is being modeled.
- Problems with frames include allowing unrestrained alteration / cancellation of slots.

Logic and Sets

- Knowledge can also be represented by symbols of logic.
- Logic is the study of rules of exact reasoning – inferring conclusions from premises.
- Automated reasoning – logic programming in the context of expert systems.

Figure 2.8 A Car Frame

Slots	Fillers
manufacturer	General Motors
model	Chevrolet Caprice
year	1979
transmission	automatic
engine	gasoline
tires	4
color	blue

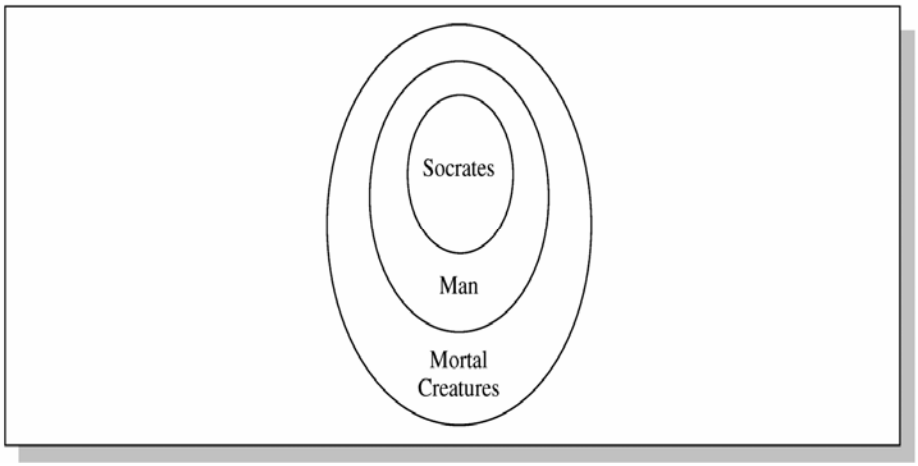
Forms of Logic

- Earliest form of logic was based on the syllogism – developed by Aristotle.
- Syllogisms – have two premises that provide evidence to support a conclusion.
- Example:
  - Premise: *All cats are climbers.*
  - Premise: *Garfield is a cat.*
  - Conclusion: *Garfield is a climber.*

Venn Diagrams

- Venn diagrams can be used to represent knowledge.
- Universal set is the topic of discussion.
- Subsets, proper subsets, intersection, union , contained in, and complement are all familiar terms related to sets.
- An empty set (null set) has no elements.

Figure 2.13 Venn Diagrams



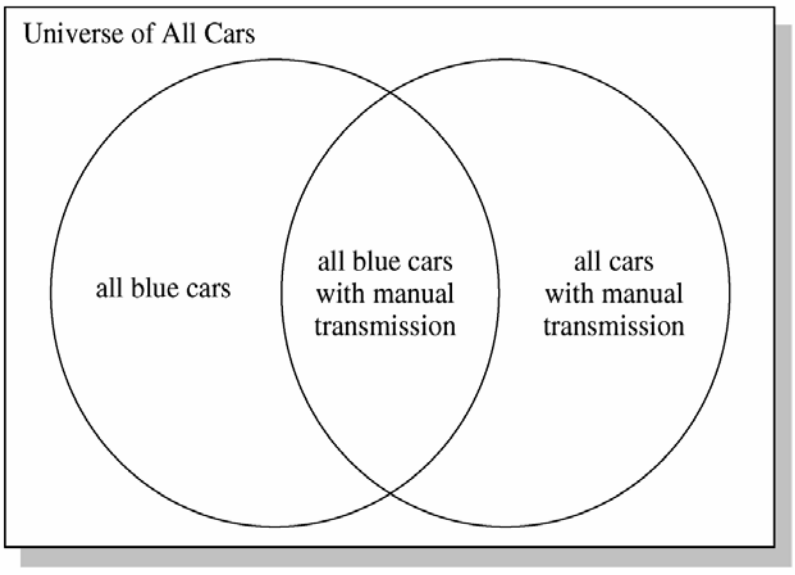
Propositional Logic

- Formal logic is concerned with syntax of statements, not semantics.
- Syllogism:
  - All goons are loons.
  - Zadok is a goon.
  - Zadok is a loon.
- The words may be nonsense, but the form is correct – this is a “valid argument.”

Boolean vs. Aristotelian Logic

- Existential import – states that the subject of the argument must have existence.
- "All elves wear pointed shoes." – not allowed under Aristotelian view since there are no elves.
- Boolean view relaxes this by permitting reasoning about empty sets.

Figure 2.14 Intersecting Sets



Boolean Logic

- Defines a set of axioms consisting of symbols to represent objects / classes.
- Defines a set of algebraic expressions to manipulate those symbols.
- Using axioms, theorems can be constructed.
- A theorem can be proved by showing how it is derived from a set of axioms.

Other Pioneers of Formal Logic

- Whitehead and Russell published ***Principia Mathematica***, which showed a formal logic as the basis of mathematics.
- **Gödel** proved that formal systems based on axioms could not always be proved internally consistent and free from contradictions.

Features of Propositional Logic

- Concerned with the subset of declarative sentences that can be classified as true or false.
- We call these sentences “statements” or “propositions”.
- Paradoxes – statements that cannot be classified as true or false.
- Open sentences – statements that cannot be answered absolutely.

Features Continued

- Compound statements – formed by using logical connectives (e.g., AND, OR, NOT, conditional, and biconditional) on individual statements.
- Material implication – $p \rightarrow q$ states that if p is true, it must follow that q is true.
- Biconditional – $p \leftrightarrow q$ states that p implies q and q implies p.

Features Continued

- Tautology – a statement that is true for all possible cases.
- Contradiction – a statement that is false for all possible cases.
- Contingent statement – a statement that is neither a tautology nor a contradiction.

Truth Tables

Table 2.4 Truth Table of the Binary Logical Connectives

p	q	$p \wedge q$	$p \vee q$	$p \rightarrow q$	$p \leftrightarrow q$
T	T	T	T	T	T
T	F	F	T	F	F
F	T	F	T	T	F
F	F	F	F	T	T

Table 2.5 Truth Table of Negation Connectives

p	~p
T	F
F	T

Universal Quantifier

- The universal quantifier, represented by the symbol ∀ means “for every” or “for all”.

$$(\forall x)\ (x \text{ is a rectangle} \rightarrow x \text{ has four sides})$$

- The existential quantifier, represented by the symbol ∃ means “there exists”.

$$(\exists x)\ (x - 3 = 5)$$

- Limitations of predicate logic – *most* quantifier.

Summary

- We have discussed:
  - Elements of knowledge
  - Knowledge representation
  - Some methods of representing knowledge
- Fallacies may result from confusion between form of knowledge and semantics.
- It is necessary to specify formal rules for expert systems to be able to reach valid conclusions.
- Different problems require different tools.