THE REPRESENTATION OF KNOWLEDGE

Table of Content

Introduction

- Knowledge in an expert system is of little value without a way to organize and load into a computer
- The know-how to program a computer to mimic the thought processes of an expert through an appropriate representation scheme is called **knowledge representation**
- It involves knowledge of a shell or a programming language that will represent the expert knowledge

Introduction

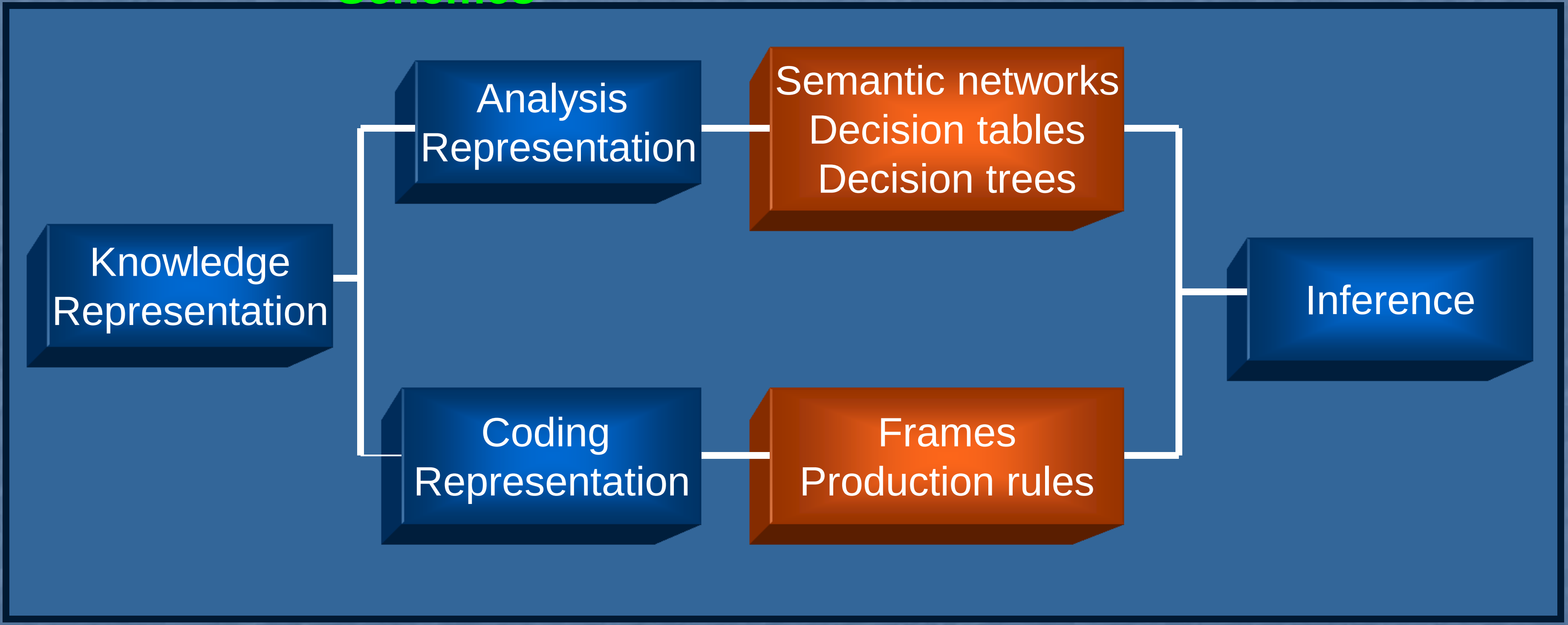
- A number of schemes have been developed over the years that share 2 common characteristics:
  a. They contain facts that can be used in reasoning through shell inference engine
  b. They can be programmed with existing computer languages

Introduction

- There are generally 2 types of knowledge representations schemes:
  a. Analysis representation
     - Support knowledge acquisition during scope establishment and initial knowledge gathering
     - Most techniques are pictorial such as semantic networks, decision trees and tables
  b. Coding representation
     - The working code of the ES either in the form of frames or production rules

Introduction

Selected Knowledge Representation Schemes

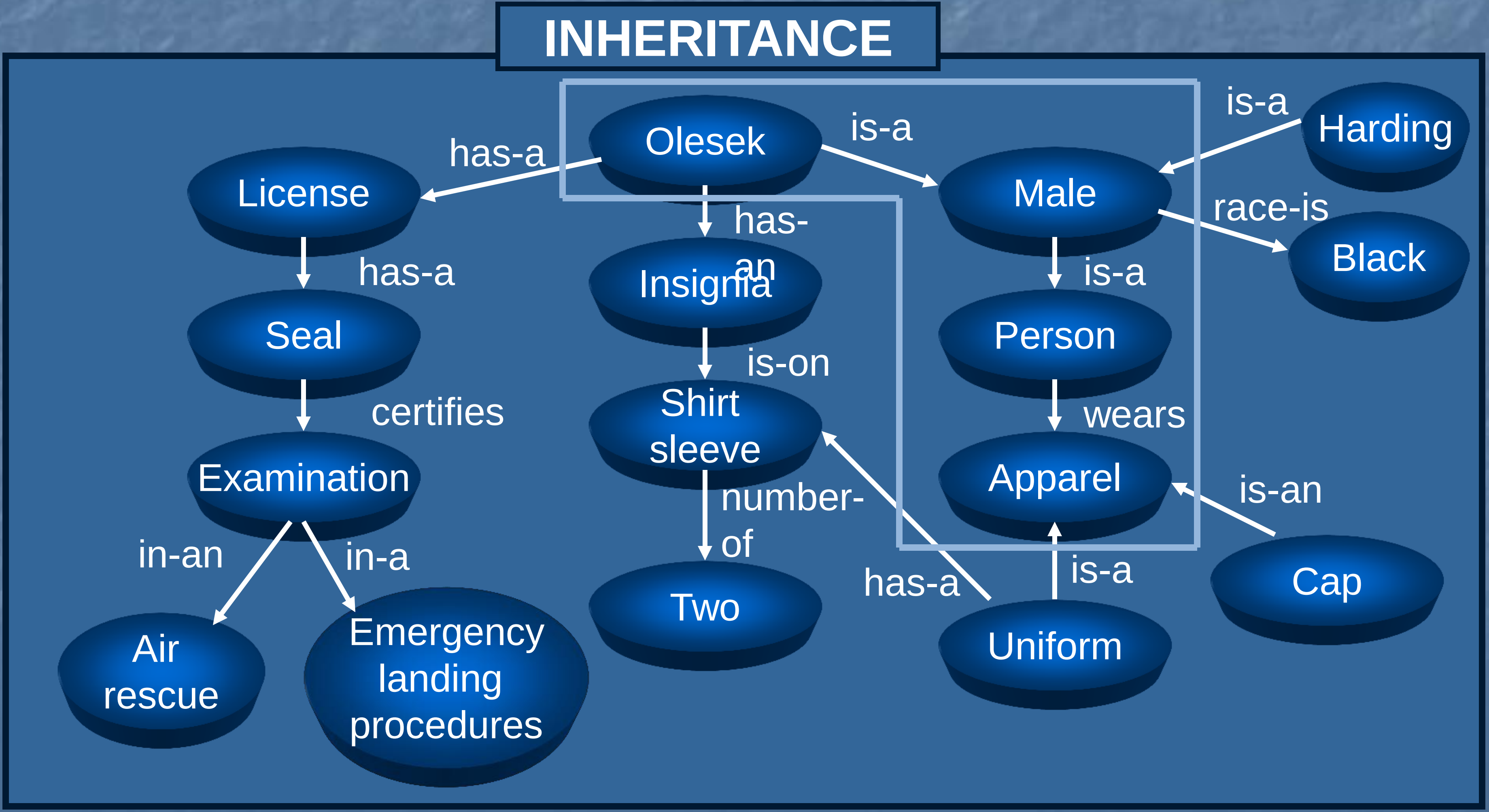


Semantic networks

- Semantic networks is the most general representation scheme, and also one of the oldest in AI [Harmon85, Castillo91].
- Basically graphical representation of knowledge that show hierarchical relationships between objects.
- Made up of a network of nodes and arc.
- The nodes represent objects and the arc the relationships between objects.

Semantic networks

- Example:



Semantic networks

- Nodes represent the objects, concepts, or events in the world.
- Names of the arcs
  - correspond to names of relations
  - indicate which concepts or objects are linked by the relations.

Semantic networks

- The 2 common arcs used are:
  - IS-A is used to show class relationship.
  - HAS-A is used to identify characteristics or attributes of the object nodes
  - other arcs are used for definitional purpose only

Decision Tables

- Organized in a spreadsheet format, using columns and rows
- The table divided into two parts:
  - A list of attributes is developed and for each attribute all possible values are listed
  - A list of possible conclusions. The different configurations of attributes are match against the conclusion.
- Knowledge for the table is collected in KA sessions

Decision Tables

- Decision Table for Gift Problem

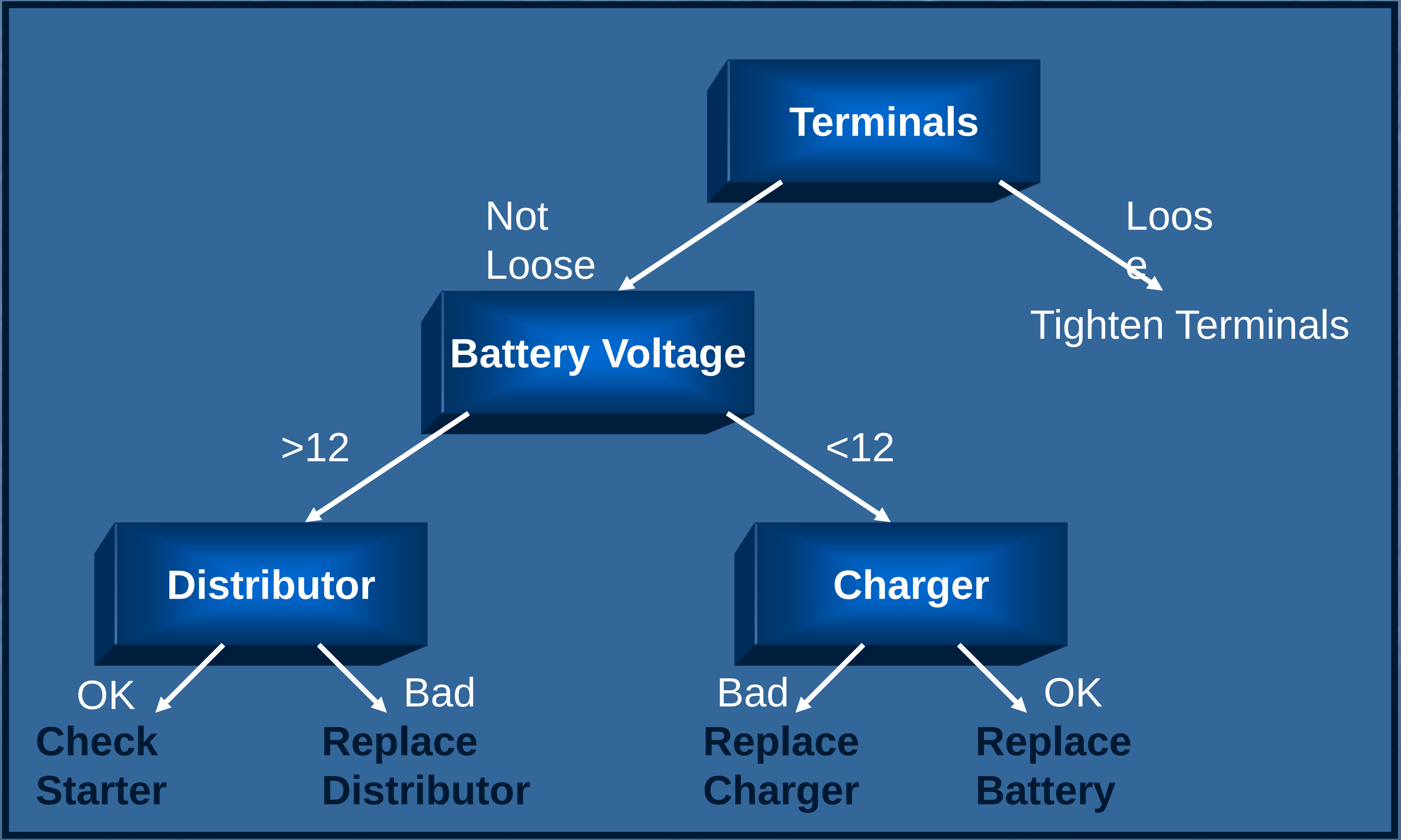
Decision Factors		**Result**
Money	**Age**	**Gift**
Much	Adult	Car
Much	Child	Computer
Little	Adult	Toaster
Little	Childs	Calculator

Decision Trees

- A hierarchical arranged semantic network and is closely related to a decision table
- It is composed of nodes representing goals and links representing decisions
- Rules can be extracted from the decision tree, that can be executed by computer program
- A major advantage can simplify knowledge acquisition process

Decision Trees

- Decision tree for electrical system diagnosis



Frames

- A frame is a data structure for representing common concepts and situations (stereotype knowledge).

- Like semantic nets, frames can be organized in a hierarchy with general concepts near the top and specific concepts placed at the lower levels.

Frames

- Unlike semantic nets, each frame or node in this hierarchy can be very rich in supplementary information, thus eliminating many of the nodes that are required in a semantic network.
- Values that describe one object are group together into single unit.

Frames

- Values that describe one object are group together into single unit.
- Knowledge partition into slots.
- Each slot describe:
  - declarative knowledge (colour of a car)
  - procedural knowledge (activate a certain rule if the value exceed certain value)

Frames

- Frames describe an object in great detail. The detail in form of slots that describe the various and characteristic of the object or situation.

Frames

- Basic frame design

Frame Name:	Object1
Class:	Object2
Properties:	

Property1	Value1
Property2	Value2
***	***
***	***

Frames

Class frame

- Represent the general characteristic of some set of common objects.
- For example, class frames such as cars, boats, birds.
- Define those properties that are common to all the object within the class, and possibly default property values.
- 2 types of properties:
  - static: describe an object feature whose value doesn't change
  - dynamic: is a feature whose value is likely to change during operation of the system

Frames

- Class Frame

Frame Name:	Bird

Properties:

Color	Unknown
Eats	Worms
No Wings	2
Flies	Try
Hungry	Unknown
Activity	Unknown

Frames

Instance Frame

- Use to describe a <u>specific instance</u> (sub-class or examples) of a class frame.
- This frames <u>inherits</u> both properties and property values from class.
- The property values can be <u>change</u> to tailor the object represented in instance frame.
- Can create as many instances of the class and immediately inherits the class information.
- Speed up system coding.

Frames

- Instance frame

Frame Name: Tweety

Class: Bird

Properties:

Color	Yellow
Eats	Worms
No Wings	1
Flies	False
Hungry	Unknown
Activity	Unknown
Lives	Cage

Frames

Frame Inheritance

- From example “Tweety” an instance of bird class.
- Like most bird Tweety eat worms, but has only one wing and cannot fly.
- Can allow an instance to accept the class default values or provide values unique to the instance.
- Can also provide unique properties. e.g. if Tweety live in a cage.

Frames

Frame Inheritance

- Inheriting Behaviour
- Beside inheriting descriptive information from its class, an instance also inherits it behaviour.
- Need to include a procedure (method) within class frame that define some action the frame performs.

Frames

Facets

- Facets (subslots) describe some knowledge or procedures about the attribute in the slot.
- Can control property values
- Can be used to direct reasoning process.
- May takes many form such as:
  - a constraint value; for example, the slot 'age', would be constraint age has to be an integer between 0 and 120
  - a default value; for example, unless there is contrary evidence it is assumes that all people like sambal belacan

Frames

Facts (continue…)

- May takes many form such as: (cont..)
  - If-ADDED Facet: Executes when new information is placed in the slots.
  - If-REMOVED Facet: Executes when information is deleted from the slot.
  - If-NEEDED Facet: Executes when new information is needed from the slot, but the slot is empty.
  - If-CHANGED Facet: Executes when information changes.

Frames

If-NEEDED Facet

- Need to define a procedure or method that executes whenever a property value is needed.
- Example:
  - Consider the property 'Flies' in Tweety frame of previous slide (slide 23). Assume this property is unknown and you want the system to determine if tweety can fly. Birds can fly if they have 2 wings.

Frames

If-NEEDED Facet

- A general rule:

```
IF       Tweety has less than 2 wings
THEN     Tweety can't fly

IF       Tweety has two wings
THEN     Tweety can fly
```

- or more specifically:

```
IF       Tweety:No_wings < 2
THEN     Tweety:Flies = False

IF       Tweety:No_wings = 2
THEN     Tweety:Flies = True
```

Frames

If-CHANGED Facet

- Perform some action whenever its value changes.
- A general rule:

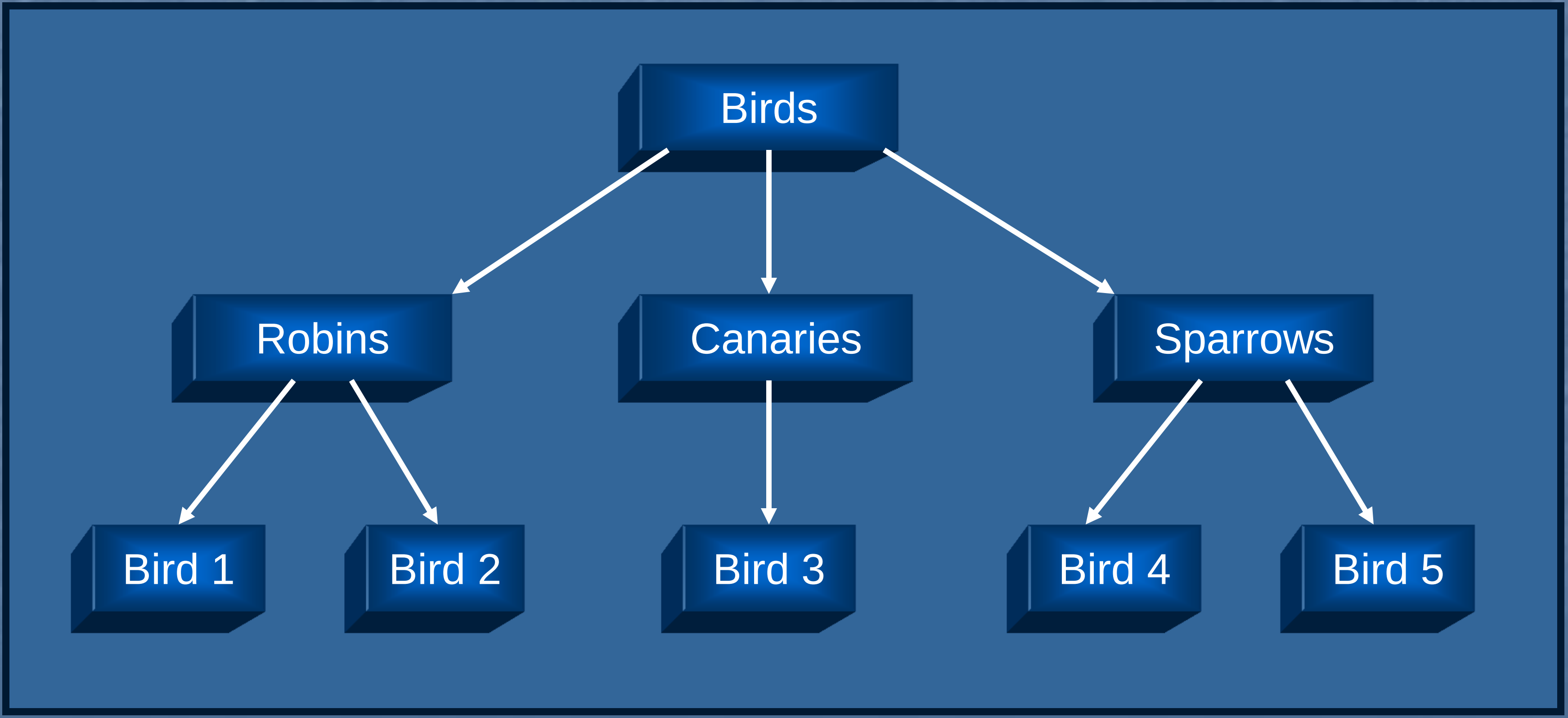
```
IF      a bird is hungry
THEN  the bird eat
```

or more specifically:

```
IF      Bird:Hungry = True
THEN  Bird: Activity = Eating
```

Frames

- Frame World of Birds



Exercise on Frames

- Prepare a frame of an automobile that you know, show 2 level of hierarchies. Fill some property and property values. (static and dynamic)

Production Rules

- Form of <u>procedural knowledge</u> that describe how to solve a problem.
- The procedural and/or factual knowledge is represented as rules, called productions, in the form of condition-action pairs.
- Is stated as follows:
  - "IF this condition occurs, THEN do this action; or this result (or conclusion or consequence) will occur.

Production Rules

- Examples

  IF flammable liquid was spilled,
  THEN call the fire department.

  IF the pH of the spill is less than 6,
  THEN the spill material is an acid.

  IF the spill material is an acid,
  and the spill smells like vinegar,
  THEN the spill material is acetic acid.

Production Rules

- When the IF portion of a rule is satisfied by the facts, the action specified by the THEN is performed.
- When this happens the rule is said to "fire" or "execute".

Types of Rules

- **Relationship**
  - IF The battery is dead
    THEN The car will not start
- **Recommendation**
  - IF The car will not start
    THEN take a cab
- **Directive**
  - IF the car will not start
    AND the fuel is okay
    THEN check out the electrical system

Types of Rules

- **Strategy**
  - IF The car will not start
    THEN first check out the fuel system then check out electrical system
- **Heuristic**
  - IF the car will not start
    AND The car is a 1957 Ford
    THEN check the float

Types of Rules

- **Uncertain Rules**
  - IF inflation is high
    THEN *Almost certainly* interest rates are high

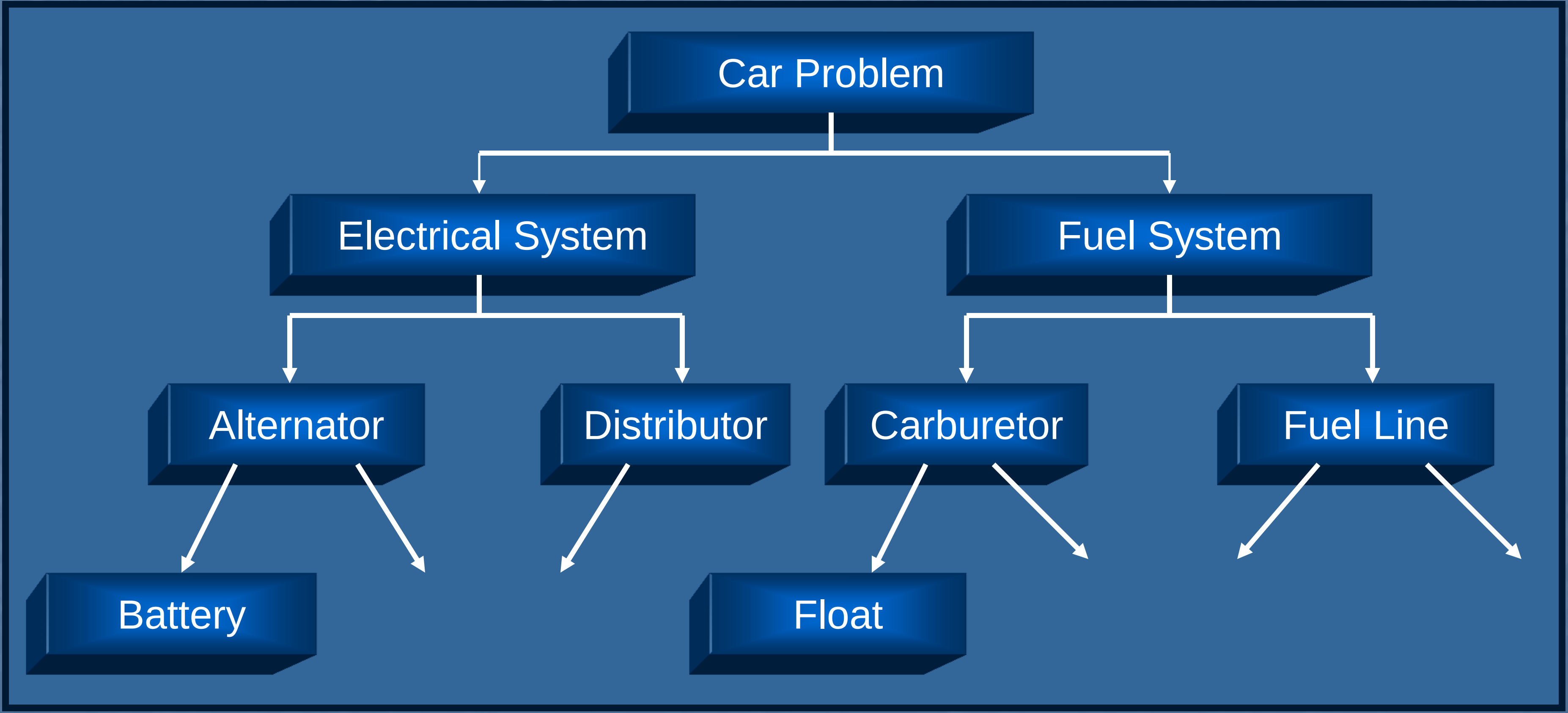
- Can assign *Certainty Factors*:
  - IF inflation is high
  - THEN interest are high CF=0.8

Types of Rules

- **Meta-Rules**
- A rule that describe how other rules should be used.
  - IF the car will not start
    AND the electrical system is operating normally
    THEN use rules concerning the fuel system

Rules Set

- Expert formed several sets of rules to a given problem through experience



Advantages of Rules

- Rules are easy to understand
- Inference and explanation are easy to derive
- Modifications and maintenance are relatively easy
- Uncertainty is easily combined with rules
- Each rule is usually independent of all others

Limitation of Rules

- Complex knowledge requires thousand of rules: problems in using and maintaining it
- Builders like rules rather than looking for more appropriate representations
- System with many rules may have a search limitation in the control program: difficulty in evaluating and making inferences

Exercise on Rules

- Try to crank the starter. If it is dead or cranks slowly, turn on the headlights. If the headlights are bright (or dim only slightly), the trouble is either in the starter itself, the solenoid, or in the wiring. To find the trouble, short the two large solenoid terminals together (not to ground). If the starter cranks normally, the problem is in the wiring or in the solenoid; check them up to the ignition switch. If the starter does not work normally, check the bushings (see section 7-3 of the manual for instructions). If the bushings are good send the starter to the test station or replace it. If the headlights are out or very dim, check the battery (see section 7-4 for instructions). If the battery and connecting wires are not at fault, turn the headlights on and try to crank the starter. If the lights dim drastically, it is probably because the starter is shorted to the ground. Have the starter tested or replace it. (Based on Carrice et al. [5]).

Logic

- Oldest form of knowledge representation in a computer is logic
- Logic is concerned with the truthfulness of a chain of statements.
- An argument is true if and only if, when all assumptions are true, then all conclusions are also true.
- 2 kinds of logic:
  - Propositional Logic
  - Predicate Calculus
- Both use symbols to represent knowledge and operators applied to the symbols to produce logical reasoning

Propositional Logic (PL)

- Propositional logic represents and reasons with propositions.
- P.L. assigns symbolic variable to a proposition such as
  - A = The car will start
- In. P. L. if we are concern with the truth of the statement, we will check the truth of A.

Propositional Logic (PL)

Operators	**Symbol**
AND	∧, &, ∩
OR	∨, ∪, +
NOT	¬, ~
IMPLIES	⊃, →
EQUIVALENT	≡

Propositional Logic (PL)

- Propositions that are linked together with connectives, such as AND, OR, NOT, IMPLIES, and EQUIVALENT, are called compound statements.

- Example:

IF	The Students Work Hard	$\rightarrow$	A
AND	Always come to lectures	$\rightarrow$	B
AND	Do all their homework's $\rightarrow$	C	
THEN	they will get an A	$\rightarrow$	D

- Using the symbols: $A \wedge B \wedge C$ -> D

- Propositional logic is concerned with the truthfulness of compound statements, depending on the connectives.

Propositional Logic (PL)

Truth Table

A	B	A and B	A or B	Not A	A ≡ B
F	F	F	F	T	T
F	T	F	T	T	F
T	F	F	T	F	F
T	T	T	T	F	T

Propositional Logic (PL)

- **Implies Operator**: C = A $\rightarrow$ B
- For implication C, if A is true, then B is implied to be true
- The implies return a F when A is TRUE and B is FALSE Otherwise it return a TRUE.

A	B	C
F	F	T
F	T	T
T	F	F
T	T	T

Propositional Logic (PL)

- Example to illustrate **Implies**

IF	The battery is dead (A)
THEN	The car won’t start (B)

Propositional Logic (PL)

EQUIVALENCE

Idempotent Laws	$A \rightarrow B \equiv \neg A \cup B$ $A \cap \neg A \equiv F$ $A \cup \neg A \equiv T$
Commutative Laws	$A \cap B \equiv B \cap A$ $A \cup B \equiv B \cup A$
Distributive Laws	$A \cap (B \cup C) \equiv (A \cap B) \cup (A \cap C)$ $A \cup (B \cap C) \equiv (A \cup B) \cap (A \cup C)$
Associative Laws	$A \cap (B \cap C) \equiv (A \cap B) \cap C$ $A \cup (B \cup C) \equiv (A \cup B) \cup C$
Absorptive Laws	$A \cup (A \cap B) \equiv A$ $A \cap (A \cup B) \equiv A$
DeMorgan's Laws	$\neg(A \cap B) \equiv \neg A \cup \neg B$ $\neg(A \cup B) \equiv \neg A \cap \neg B$

Propositional Logic (PL)

- Used to transform complex statement into simpler statement and equivalent.
- Example:

$\neg(A \cap B) \cup B$	
$(\neg A \cup \neg B) \cup B$	DeMorgan's Laws
$\neg A \cup (\neg B \cup B)$	Associative Law
$\neg A \cup T$	Idempotent Law
T	Identity Law

Propositional Logic (PL)

- P.L. offers techniques for capturing facts or rules in symbolic form and then operates on them through use of logical operators.
- PL provide method of managing statements that are either TRUE or FALSE.
- Prolog is based on PC

Predicate Calculus (PC)

- Some P.L. weakness:

1. Limited ability to express knowledge and lose much of their meanings.

   The Pacific Ocean contains water.

   Florida is a state within the USA.

   Only assigning true value without making any statement about ‘oceanhood’ or ‘statehood’.

Predicate Calculus (PC)

- Some P.L. weakness:

  2. Not all statements can be represented.

     All *men are mortals.*

     Some *dogs like cats.*

- Thus, need a more general form of logic capable of representing the details.

Predicate Calculus (PC)

- Enhances processing by allowing the use of variables and functions.
- Use symbols that represent
    - constants
    - predicates
    - variables
    - functions
- Operate on these symbols using PL operators.

Predicate Calculus (PC)

- Constant

- Specific objects or properties about a problem.
- Begin with lower case.
- Example: ahmad, elephant and temperature
- ahmad represent object Ahmad. Can also use A or X instead of Ahmad.

Predicate Calculus (PC)

- <u>Predicates</u>

- Divide proposition into 2 parts:
- predicate: assertion about object
- argument: represent the object
- Example: To represent Azizi teach TN6023,
  - teach (azizi, tn6023)
  - teach is a predicate, denoting relationship between arguments. The 1st letter must be in lower case.

Predicate Calculus (PC)

- <u>Variables</u>

- Represent general classes of objects or properties
- Written as symbols beginning with upper case.
- To capture the proposition Azizi teach TN6023, we write:
  - teach (X,Y)
  - X = Azizi and Y = TN6023

Predicate Calculus (PC)

- <u>Functions</u>

- Permits symbol to be used to represent function.
- A function denotes a mapping from entities of a set to a unique element of another set.
  - father(azizi) = zakaria   mother(azizi) = zaharah.
  - Can be also used within predicates. For example:
  - husband (father(azizi), mother(azizi)) = husband(zakaria,zaharah)

Predicate Calculus (PC)

- Operations
- PC uses the same operators found in P.L.
- Proposition: John likes Mary likes(john,mary)
- Bob likes Mary likes(bob,mary)
- 2 persons like Mary. To account for jealousy:
  - likes (X,Y) AND likes(Z,Y) implies NOT likes (X,Z)

    or
  - likes (X,Y) $\wedge$ likes (Z,Y) $\rightarrow$ $\neg$likes(X,Z)

Predicate Calculus (PC)

- P.C. introduce 2 symbols called variable quantifiers.
  1. ∀ universal quantifier: for all.
  2. ∃ existential quantifier: there exist

Predicate Calculus (PC)

- ∀ Indicates the expression is TRUE for all values of designated variable.
- Example:
  - ∀X likes (X,mary)
  - means for all values of X, the statement is true, everybody likes Mary

Predicate Calculus (PC)

- ∃ indicates the expression is TRUE for some values of the variable; at least one value exist that makes the statement true:
- ∃X likes (X,mary)

Predicate Calculus (PC)

- Parentheses are used to indicate the scope of quantification
- $\forall X$ (likes(X,mary) $\wedge$ nice(mary) $\rightarrow$ nice (X))
  - determines all instances of X who like Mary and if Mary is nice, then it is implied that those who like Mary are also nice.

Exercise

- Beberapa orang anak Pak Dolah berjaya melanjutkan pelajaran ke seberang laut manakala yang selebihnya melanjutkan pelajaran di dalam negeri
- Semua burung laut suka makan ikan.
- Terdapat juga spesis burung yang tidak boleh terbang.
- Jika esok tidak hujan, Ali akan pergi ke Pasaraya.

Exercise

- Ada sesetengah orang suka makan durian.
- Kalau kaca menjadi intan, kayalah penjual botol.
- Hanya jauhari mengenal manikam.
- Kalau ada sumur diladang, boleh saya menumpang mandi, kalau ada umur   panjang, boleh kita berjumpa lagi.

Reasoning with logic

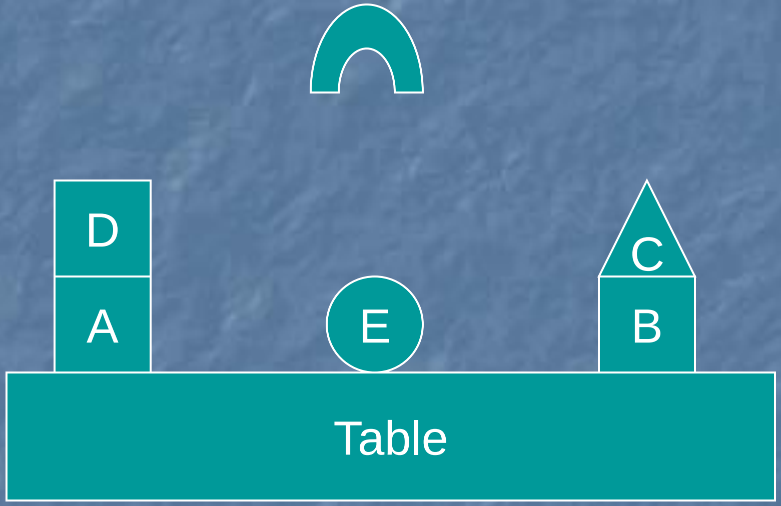
- PC can provide reasoning capability to intelligent system
- Reasoning requires the ability to infer conclusions from available facts.
- One simple form of inference is modus ponen:
  - **IF** A is true

    AND A $\rightarrow$ B is true

    THEN B is true

Reasoning with logic

- Robot Control Example
- The function of the robot is move a specified block to some specified location



Reasoning with logic

- Robot Control Example

- Description of the block world using PC using the following logical assertions:

1. cube(a), cube(b), cube(d), pyramid(c), sphere(e), hand(hand), table (table1)
2. on(a,table1), on(b,table1), on(d,a), on(c,b)
3. holding(hand,nothing)

Reasoning with logic

- Robot Control Example

- The goal might be to put some block on other block, for example put block **b** onto block **a**:

  put_on(b,a)

- To accomplish this the robot need to obtain block **b** and make certain that block **a** is clear:

  $hand\_holding(b) \wedge clear(a) \rightarrow put\_on(b,a)$

- To move any block, in variable form:

  $\forall X\ \exists Y\ (hand\_holding(X) \wedge clear(Y) \rightarrow put\_on(X,Y))$

  where X is the block to be move and Y is the target block

Reasoning with logic

- Robot Control Example

- One of the robot task when instructed to pick up and move some blocks is to determine if it is clear.

- If not clear, need to remove any item on the block:

  $\forall X\ (\neg \exists Y\ on(Y,X) \rightarrow clear(X))$

  For all X, X is clear if there does not exist a Y such that Y is on X, would produce the following assertions:

  clear(c), clear(d)

Multiple Knowledge Representation

- No single knowledge representation method is ideally suited by itself for all tasks.
- Some recent expert system shells use two or more knowledge representation schemes.
- A rather successful combination of knowledge representation method is production rules and frames.

Multiple Knowledge Representation

- By themselves, production rules do not provide a totally effective representation facility for many expert system applications.
- Their expressive power is inadequate:
  - for defining terms
  - for describing domain objects and static relationships among objects.

Multiple Knowledge Representation

- Their major inadequacies are effectively handled by frames.
- Frames provide:
  - a rich structural language for describing the objects referred to in the rules
  - the partitioning, indexing and organising a system's production rules.

Knowledge Representation

Scheme	Advantages	Disadvantages
Production Rules	▪simple syntax ▪easy to understand simple interpreter ▪highly modular ▪flexible (easy to add or modify)	▪hard to follow hierarchy ▪inefficient for large systems ▪not all knowledge can be expressed as rules ▪poor at representing structure descriptive knowledge
Semantic Networks	▪easy to follow hierarchy easy to trace association ▪flexible	▪meaning attached to nodes might be ambiguous ▪exception handling is difficult

Knowledge Representation

Scheme	Advantages	Disadvantages
Frames	▪expressive power ▪easy to setup slots for new properties and relations ▪easy to create specialised procedures ▪easy to include default information and detect missing values	▪difficult to Program ▪difficult for inference ▪lack of software
Formal Logic	▪facts asserted independently of use ▪assurance that all and only valid consequences are asserted (precision) ▪Completeness	▪separation of representation and processing ▪inefficient with large data sets ▪very slow with large knowledge base