

# Expert Systems

## 4. KNOWLEDGE REPRESENTATION

IFK15037, 3 credits



YUITA ARUM SARI, S.Kom, M.Kom  
yuita@ub.ac.id

# OUTLINE

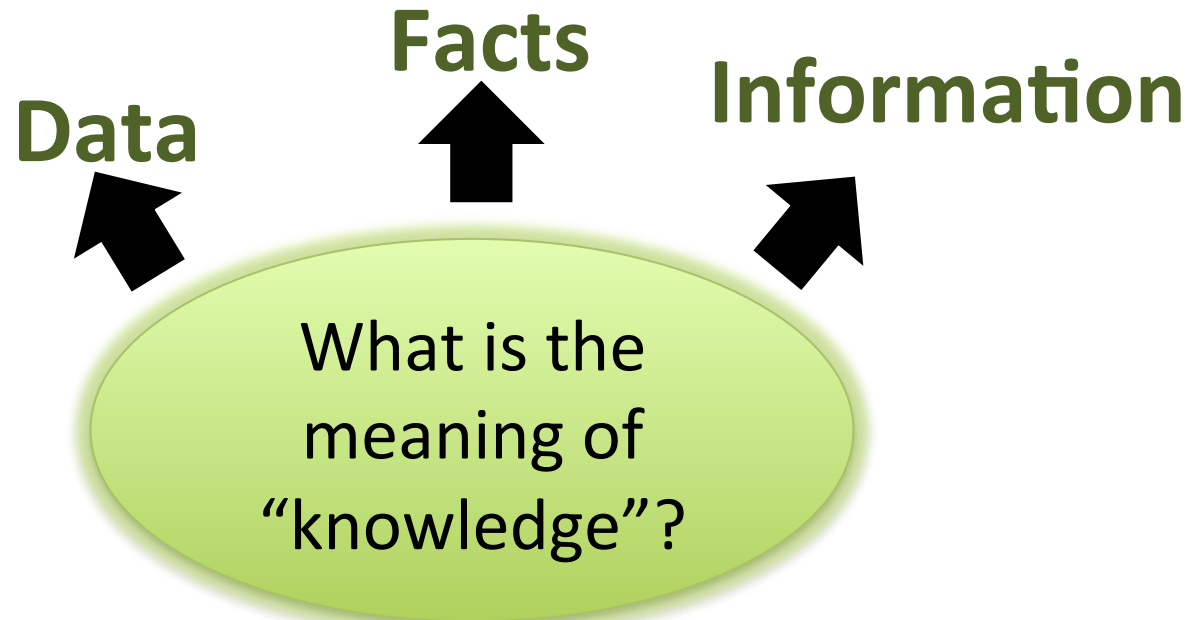
**Logic**

**Frame**



**Semantic  
Nets**

**Script**



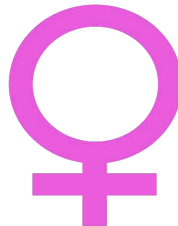
# Knowledge Representation

What is the meaning of “representation”?

Symbols standing for things in the world



**First aid**



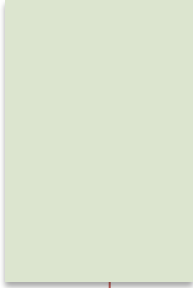
**women**

**“John loves Mary”**




The proposition that John loves Mary

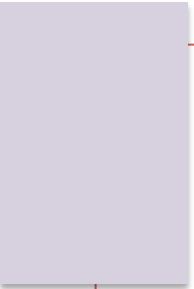
# Knowledge vs Expert Systems



**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.

An argument refers to the formal way facts and rules of inferences are used to reach valid conclusions.

The process of reaching valid conclusions is referred to as **logical reasoning.**

# Hierarchy of Knowledge



- 1) **Wisdom** : using knowledge in beneficial way
- 2) **Meta knowledge** : Rules about knowledge
- 3) **Knowledge** : Rules about using information
- 4) **Information** : potentially useful for knowledge
- 5) **Data** : potentially useful information
- 6) **Noise** : no apparent information

- ◆ Meta knowledge 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 meta knowledge** that describes everything known about the problem domain.
- ◆ **Wisdom** is the meta knowledge of determining the best goals of life and how to obtain them.



**Algorithm + Data Structures = Programs**

**Knowledge + Inference = Expert Systems**



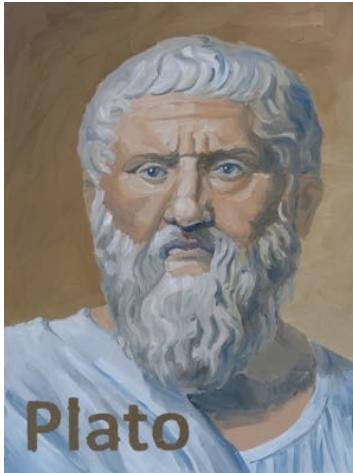
Expert systems :

1. Separate from noise
2. Transform data into information
3. Transform information to knowledge

# The Philosophers of Knowledge



**Aristoteles**



**Plato**



**Descartes**



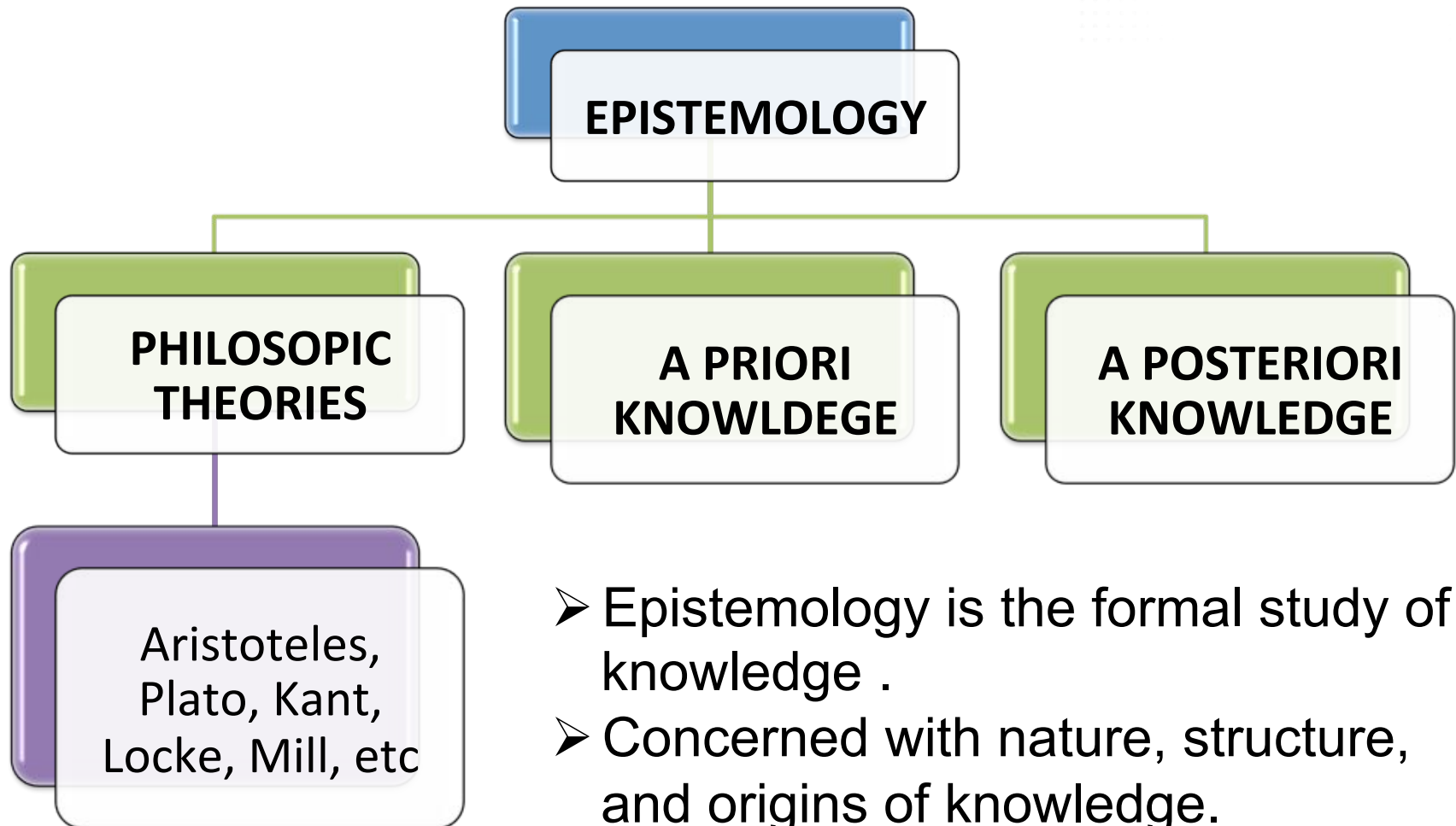
**David Hume**



**Kant**

*...and many others*

*Expert Systems: Principles and Programming, Fourth Edition*



## A priori

“That which precedes”

Independent of the senses

Universally true

Cannot be denied without contradiction

## A posteriori

“That which follows”

Derived from the senses

Now always reliable

Deniable on the basis of new knowledge w/o the necessity of contradiction

# Further Classifying of Knowledge

## PROCEDURAL

Knowing how to do something:

Fix a watch

Install a window

Brush your teeth

Ride a bicycle

## DECLARATIVE

Knowledge that something is true or false

Usually associated with declarative statements

E.g., “Don’t touch that hot wire.”

## TACIT

Unconscious knowledge (insensible)

Cannot be expressed by language

E.g., knowing how to walk, breath, etc

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.

A number of knowledge-representation techniques have been devised:

Rules

Semantic Nets

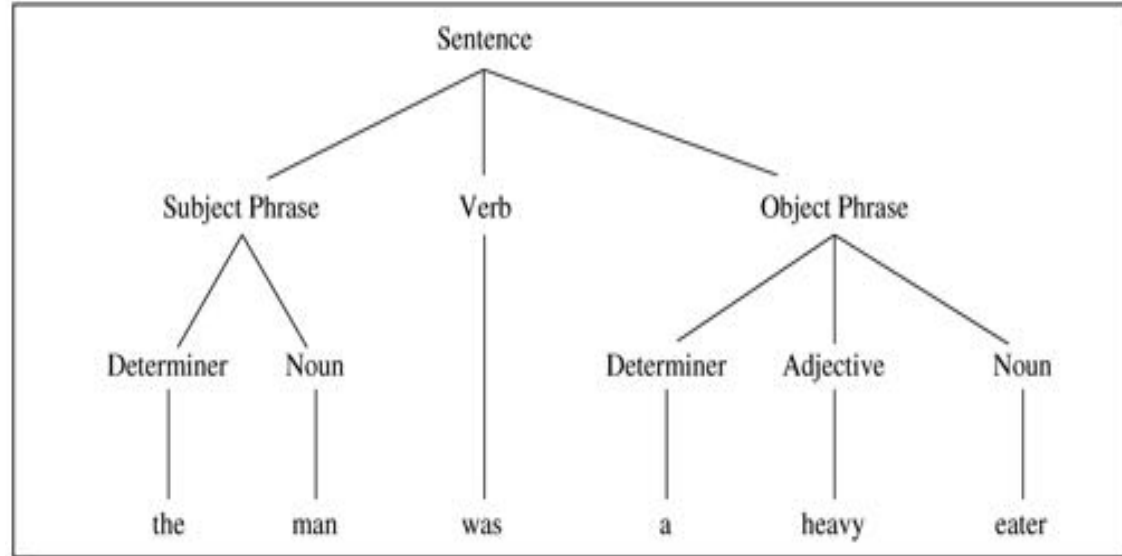
Frame

Scripts

Logic

Conceptual Graph

Parse tree of a sentence :



A parse tree or derivation tree is **graphic representation** of a sentence.

- 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.
- 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.*

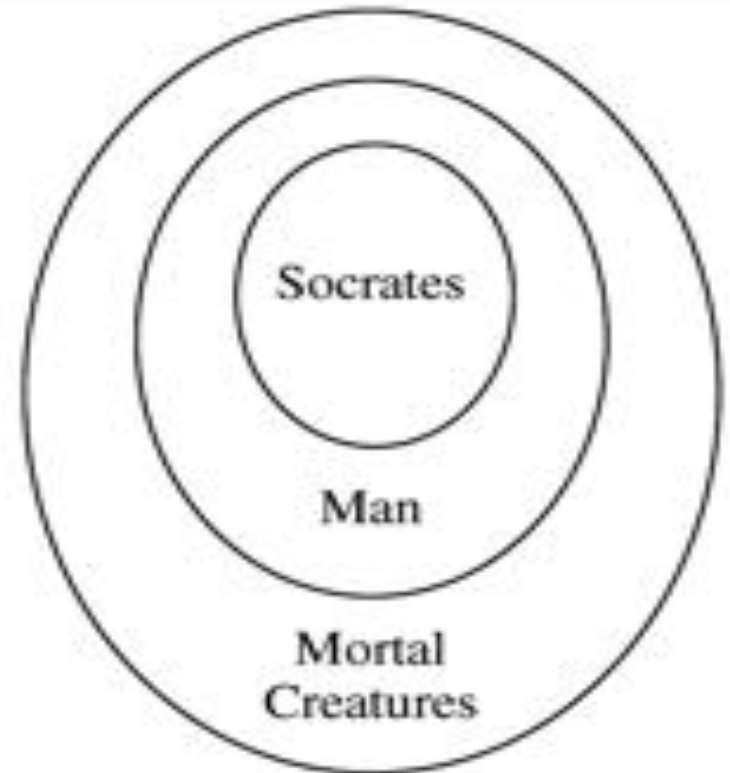
Premise: *Garfield is a cat.*

Conclusion: *Garfield is a climber.*



# Logic – Venn Diagram

- 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.



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.”

Propositional logic uses true statements to form or prove other true statements.

- **Representation (syntax):** How to represent a proposition.
- **Reasoning (algorithm):** How to create or prove new propositions.

## Representation of propositional logic

A propositional symbol and *connectives* ( $!$ ,  $*$ ,  $+$ ,  $\Rightarrow$ ,  $\Leftrightarrow$  )

Example:

$C$  = “It’s cold outside” ;  $C$  is a proposition

$O$  = “It’s October” ;  $O$  is a proposition

If  $O$  then  $C$  ; *if it’s October then it’s cold outside*

- 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.
- 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  $p$ .

# The Feature of Propositional Logic

- ① **Tautology** – a statement that is true for all possible cases.
- ② **Contradiction** – a statement that is false for all possible cases.
- ③ **Contingent statement** – a statement that is neither a tautology nor a contradiction.

**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**

$p$	$\sim p$
T	F
F	T

# The propositional expression of P,Q,R

$$\neg (\neg \mathbf{P}) \equiv \mathbf{P}$$

$$(\mathbf{P} \vee \mathbf{Q}) \equiv (\neg \mathbf{P} \rightarrow \mathbf{Q})$$

the contrapositive law:  $(\mathbf{P} \rightarrow \mathbf{Q}) \equiv (\neg \mathbf{Q} \rightarrow \neg \mathbf{P})$

de Morgan's law:  $\neg (\mathbf{P} \vee \mathbf{Q}) \equiv (\neg \mathbf{P} \wedge \neg \mathbf{Q})$  and  $\neg (\mathbf{P} \wedge \mathbf{Q}) \equiv (\neg \mathbf{P} \vee \neg \mathbf{Q})$

the commutative laws:  $(\mathbf{P} \wedge \mathbf{Q}) \equiv (\mathbf{Q} \wedge \mathbf{P})$  and  $(\mathbf{P} \vee \mathbf{Q}) \equiv (\mathbf{Q} \vee \mathbf{P})$

the associative law:  $((\mathbf{P} \wedge \mathbf{Q}) \wedge \mathbf{R}) \equiv (\mathbf{P} \wedge (\mathbf{Q} \wedge \mathbf{R}))$

the associative law:  $((\mathbf{P} \vee \mathbf{Q}) \vee \mathbf{R}) \equiv (\mathbf{P} \vee (\mathbf{Q} \vee \mathbf{R}))$

the distributive law:  $\mathbf{P} \vee (\mathbf{Q} \wedge \mathbf{R}) \equiv (\mathbf{P} \vee \mathbf{Q}) \wedge (\mathbf{P} \vee \mathbf{R})$

the distributive law:  $\mathbf{P} \wedge (\mathbf{Q} \vee \mathbf{R}) \equiv (\mathbf{P} \wedge \mathbf{Q}) \vee (\mathbf{P} \wedge \mathbf{R})$

Same connectives as propositional logic

Propositions have structure: Predicate/Function + arguments.

$R, 2$  ; *Terms. Terms are not individuals, not propositions*

$\text{Red}(R), (\text{Red } R)$  ; *A proposition, written in two ways*

$(\text{southOf UnicornCafe UniHall})$  ; *a proposition*

$(+ 2 2)$  ; *Term, since the function + ranges over numbers*

Quantifiers enable general axioms to be written

(forall ?x

(iff (Triangle ?x) (and (polygon ?x)

(numberOfSides ?x 3)))

# Logic – Predicate Logic

Although Propositional Logic is  
complete ...  
It is still inadequate.

All A are B  
n is A  
n is B

All men are mortal.  
Socrates is a man.  
Socrates is mortal.

VALID

To capture what makes this form valid we need to notate the subject and **predicate** of a proposition.

**Note:** We need logic laws that work for statements involving quantities like “some” and “all”.



A Predicate is a declarative sentence whose true/false value depends on one or more variables.

The statement “ $x$  is greater than 3” has two parts:

- the subject:  $x$  is the subject of the statement
- the predicate: “is greater than 3” (a property that the subject can have).

We denote the statement “ $x$  is greater than 3” by  $P(x)$ , where  $P$  is the predicate “is greater than 3” and  $x$  is the variable. The statement  $P(x)$  is also called the value of propositional function  $P$  at  $x$ .

Assign a value to  $x$ , so  $P(x)$  becomes a proposition and has a truth value:

$P(5)$  is the statement “5 is greater than 3”, so  $P(5)$  is true.

$P(2)$  is the statement “2 is greater than 3”, so  $P(2)$  is false.

Given each propositional function determine its true/false value when variables are set as below.

1.  $\text{Prime}(x)$  = “ $x$  is a prime number.”
  - $\text{Prime}(2)$  is true, since the only numbers that divide 2 are 1 and itself
  - $\text{Prime}(9)$  is false, since 3 divides 9.
2.  $C(x, y)$  = “ $x$  is the capital of  $y$ ”.
  - $C(\text{Ottawa}, \text{Canada})$  is true.
  - $C(\text{Buenos Aires}, \text{Brazil})$  is false.
3.  $E(x, y, z)$  = “ $x + y = z$ ”.
  - $E(2, 3, 5)$  is ...
  - $E(4, 4, 17)$  is ...

# Logic – Predicate Logic

greater\_than(2, 3)

Predicate  
symbol

term (constant)

mother\_of(joe, susan)

mother\_of(sister\_of(joe), susan)

- Production rules are one of the most popular and widely used knowledge representation languages.
- Production rule system consists of three components
  - **working memory** contains the information that the system has gained about the problem thus far.
  - **rule base** contains information that applies to all the problems that the system may be asked to solve.
  - **interpreter** solves the control problem, i.e., decide which rule to execute on each selection-execute cycle.
- Used both for KR and Problem solving system

- ❑ A classic representation technique for propositional information
- ❑ 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 (relationships between them)

## Intuition base:

An important feature of human memory is the high number of connections or associations between the different pieces of information contained in it.

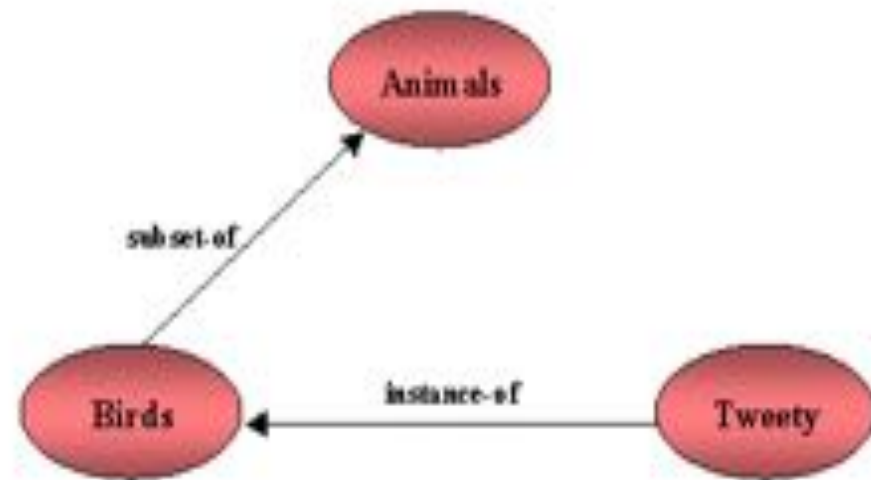
## There are two types of primitive

- **Nodes** correspond to objects, or classes of objects, in the world
- **Links** are unidirectional connections between nodes and correspond to relationships between these objects

# Semantic Nets

Major problem with semantic nets is that although the name of this knowledge representation language is semantic nets, there is not, ironically, clear semantics of the various network representations. For the above example,

- it can be interpreted as the representation of a specific bird named Tweety,
- it can be interpreted as a representation of some relationship between Tweety, birds and animals.



# Semantic Nets

## Common Used Links :

- IS-A
- PART-OF
- MODIFIERS: on, down, up, bottom, move to,...
- Link types are set up for specific domain knowledge

## Example of Semantic Nets :

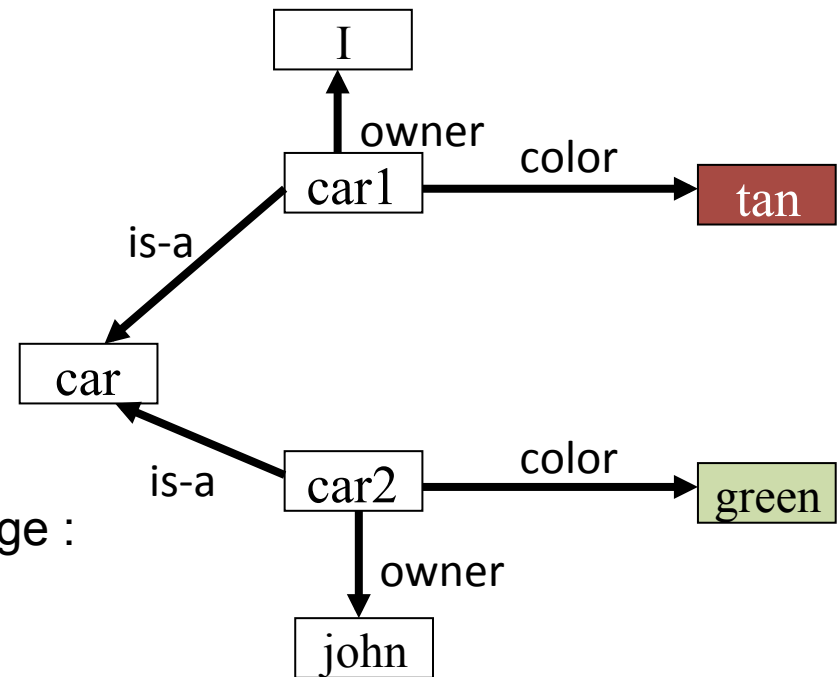
My car is tan and John's car is green

### For particular domain :

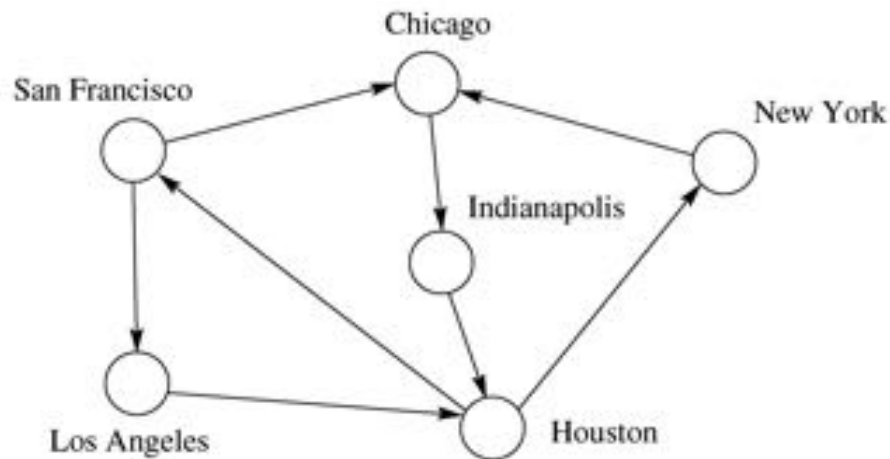
- make up a set of link-types
- create a set of nodes
- connect them together
- ascribe meaning

### Programs to manipulate the knowledge :

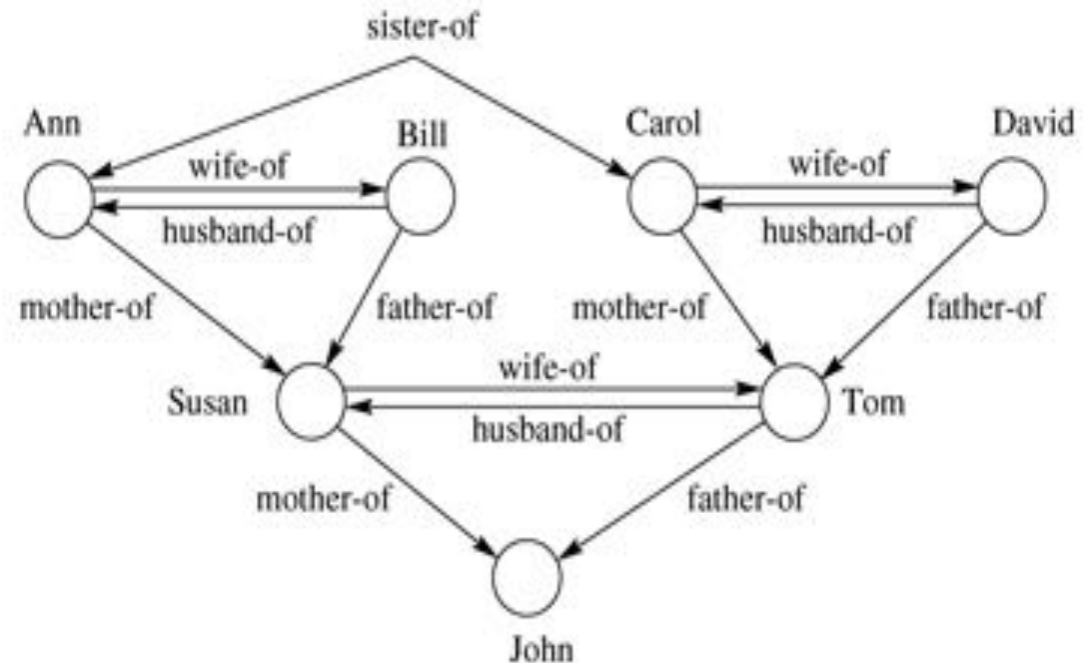
- LISP
- CL



# Two Types of Semantic Nets



(a) A General Net



(b) A Semantic Net



- ❖ **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

- 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.
- 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.

- Frame: a knowledge representation technique which attempts to organize concepts into a form which exploits inter relationships and common beliefs
- frame-based KR is analogous to object-oriented programming; the difference is the entities encoded
- A frame is similar to a record data structure or database record:
- Frame has slot names and slot fillers, and usually arranged in a hierarchy

## Structure of frame (1)

### Frame name

**slot:** value , value, .....

·  
·  
·

### slot:

**facet:** value, value, .....

**facet:** value, value, .....

---

Frame: printer

superset: office-machine

subset: {laser-printer, ink-jet-printer}

energy-source: wall-outlet

maker: Epson

date: 1-April-2003

Frames often allowed slots to contain procedures.

- “if-needed” procedures, run when value needed
- if-added” procedures, run when a value is added (to update rest of data, or inform user).

# Frame- Class and Instances



- (frame) instance: representing” lowest-level” object; a single object or entity
- (frame) class: describes different frames (either instances or classes)
- every instance has an “is-a” link, pointing to its class
  - possibly more than one “is-a”

# Example of Frame (1)

Frame Name:

Bird

Properties:

Colour	Unknown
Wings	2
Flies	True

← Class frame

Instance frame →

Frame Name:

Tweety

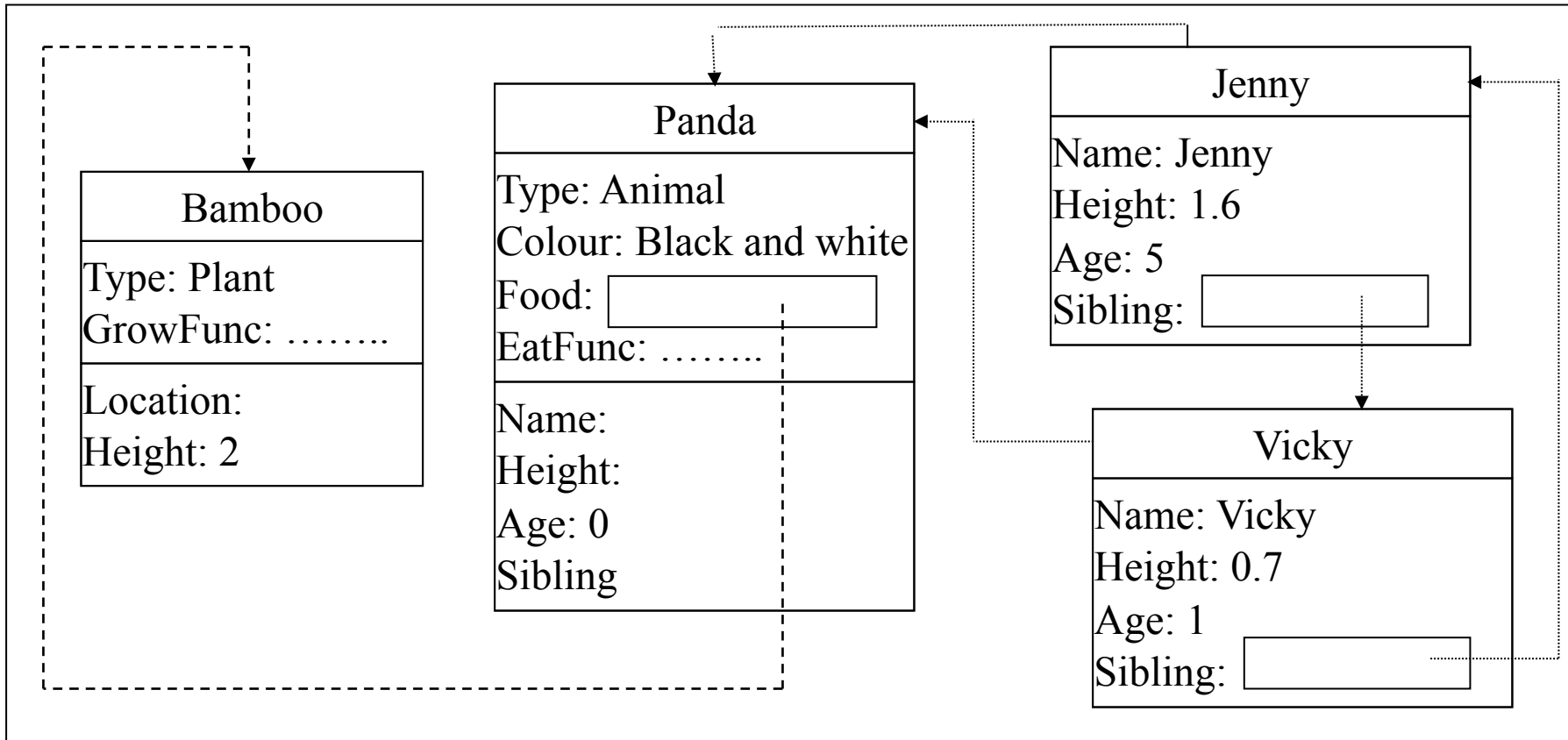
Class:

Bird

Properties:

Colour	Yellow
Wings	1
Flies	False

## Example of Frame (2)



- Rather similar to frames: uses inheritance and slots; describes stereotypical knowledge, (i.e. if the system isn't told some detail of what's going on, it assumes the "default" information is true), but concerned with events.
- Somewhat out of the mainstream of expert systems work. More a development of natural-language-processing research.
- A script is a remembered precedent, consisting of tightly coupled, expectation-suggesting primitive-action and state-change frames [Winston, 1992]
- A script is a structured representation describing a stereotyped sequence of events in a particular context [Luger, Stubblefield, 1998]



## Why Scripts ?

- ✓ Because real-world events do follow stereotyped patterns. Human beings use previous experiences to understand verbal accounts; computers can use scripts instead.
- ✓ Because people, when relating events, do leave large amounts of assumed detail out of their accounts. People don't find it easy to converse with a system that can't fill in missing conversational detail
- ✓ Scripts predict unobserved events.
- ✓ Scripts can build a coherent account from disjointed observations.
- ✓ Applications
  - This sort of knowledge representation has been used in intelligent front-ends, for systems whose users are not computer specialists.
  - It has been employed in story-understanding and news-report-understanding systems.

# Component of Scripts

- Script name
  - Entry conditions:
  - Roles
  - Props
  - Scene 1
  - Scene 2
  - ...
  - Results

# Example of Scripts

Script: **RESTAURANT**

Track: Coffee Shop

Props: Tables

Menu

Food

Check

Money

Roles: Customer

Waiter

Cook

Cashier

Owner

Entry conditions:

Customer is hungry

Customer has money

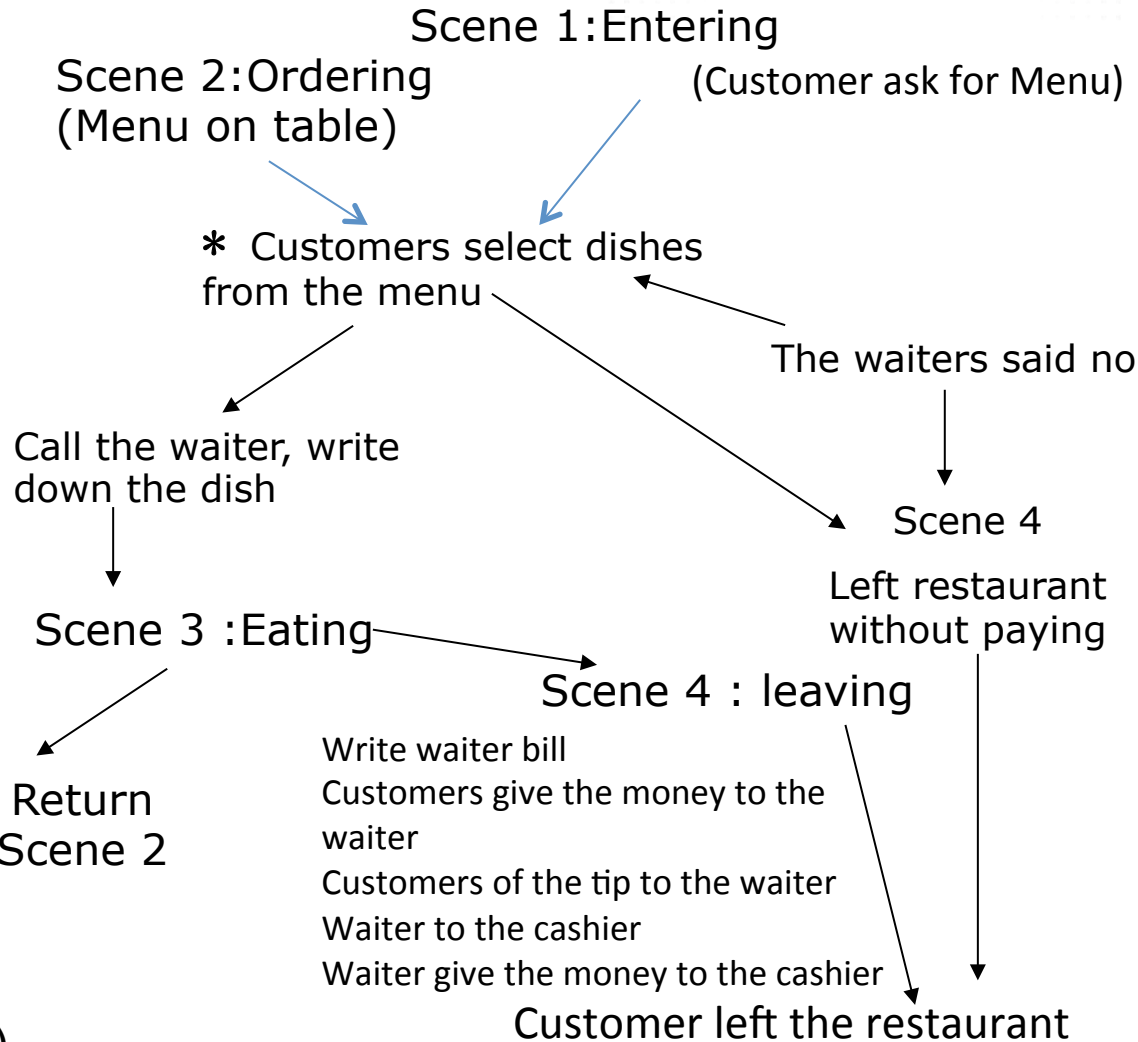
Results:

Customer has less money

Owner has more money

Customer is not hungry

Customer is pleased(optional)



# Pros and Cons of KR Methodology

Methodology	Pros	Cons
Logic	<ul style="list-style-type: none"><li>• With a semantics</li><li>• Expressiveness</li></ul>	<ul style="list-style-type: none"><li>• Inefficient</li><li>• Unable to express procedural knowledge</li><li>• Unable to do default reasoning</li></ul>
Production Systems	<ul style="list-style-type: none"><li>• Naturalness of expression</li><li>• Modularity</li><li>• Restricted syntax</li><li>• Ability to Represent Uncertain Knowledge</li></ul>	<ul style="list-style-type: none"><li>• Inefficient</li><li>• Less expressive</li></ul>
Frame representation	<ul style="list-style-type: none"><li>• Domain knowledge model reflected directly</li><li>• Support default reasoning</li><li>• Efficient</li><li>• Support procedural knowledge</li></ul>	<ul style="list-style-type: none"><li>• Lack of semantics</li><li>• Expressive limitations</li></ul>

# REFERENCES

- Peter Szoiovits, Knowledge base systems.
- Xiu-jun GONG (Ph. D), School of Computer Science and Technology, Tianjin University, [gongxj@tju.edu.cn](mailto:gongxj@tju.edu.cn)  
<http://cs.tju.edu.cn/faculties/gongxj/course/ai/>
- Joseph C. Giarratano, Gary D. Riley, Expert Systems: Principles and Programming, Published November 14th 2004 by Course Technology Inc.

**Thank You....**