Knowledge Representation (Part 2)

Dr. Mohammad Kazemifard

Session 4

Outlines

- Frames
  - Structure
  - Inheritance
  - Facets
- Logic
  - Propositional logic
  - Predicate calculus

Frames

- A natural extension of the semantic network is a schema.
- A schema is a unit that contains typical knowledge about some concept or object, and includes both declarative and procedural knowledge.
  - Schema of a bird: it has wings and legs, and how it hunt for food
- Expert system designers refer to the schema as a frame
- Frames is a data structure for representing stereotypical knowledge of some concept

Frames (Cont.)

- Minsky (first proposed frames) describes a frame as follows:
- *"When one encounters a new situation (or makes a substantial change in one's view of a problem) one selects from memory a structure called a "frame." This is a remembered framework to be adapted to fit reality by changing details as necessary."*

Structure of a Frame

- Frame name: Bob
- Class is optional and it shows object1 IS-A object2
- i.e. Class=Human

GENERAL FRAME STRUCTURE

Frame Name:	Object1	
Class:	Object2	
Properties:	Property1	Value1
	Property2	Value2
	Property3	Value3
	…	…

Class Frame

- A class frame represents the general characteristics of some set of common objects.

Frame Name: Bird

Properties:

Color	Unknown
Eats	Worms
No._Wings	2
Flies	True
Hungry	Unknown
Activity	Unknown

Instance Frame

- **Instance Frame** describes a specific instance of a class frame. The frame **inherits** both properties and property values from the class.

Frame Name: Tweety

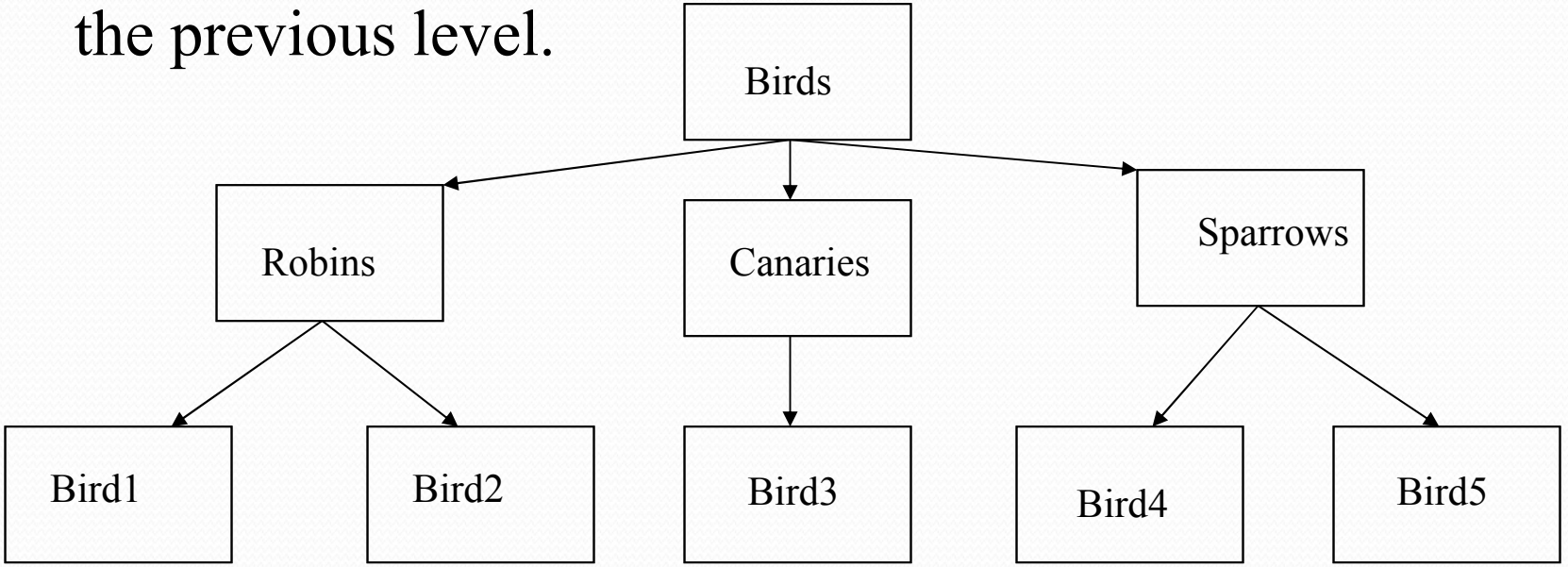
Class: Bird

Properties:

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

Hierarchical Structure

- You can create complex frame structures organizing by different levels of abstraction. Each level inherits from the previous level.



Facets

- Provide you with additional control over property values
- Can define a constraint on a property value - i.e.
  - Limit a numeric property value to some set range
  - limit a string value to one of a few possible values
  - restrict the type of data that can be stored in a
    - property value, i.e. string, numeric, or boolean
- Can instruct a property how to obtain its value or what to do if its value changes
  - IF-NEEDED facets
  - IF-CHANGED facets

Methods

- A *method* is a procedure that executes whenever a property value is needed.
- Property value can be obtained through a series of computations, or from an external database or spreadsheet
- *Method* is typically written in the procedural language of the chosen shell and attached to the property's IF-NEEDED facet

IF-NEEDED Facet

1. Assume property value “Flies” in the “Tweety” frame is “Unknown”.
2. You want the system to determine if tweety can fly.

Method: IF Tweety has less than two wings
THEN Tweety can’t fly

IF Tweety has two wings
THEN Tweety can fly

- To implement method, we need the property value of “No._Wings” from the “Tweety” frame.

IF-NEEDED Facet Cont.

Shells differ but one example is

Frame:Property

Using the code on the previous “Tweety frame, we have

Tweety:No._Wings = 1

Now we need to assert a value of **True** or **False** to the “Flies” property depending on Tweety’s wing count.

IF-NEEDED Facet Cont.

- Frame:Property = Value

- IF Tweety:No._Wings < 2
- THEN Tweety:Flies = False

- IF Tweety:No._Wings = 2
- THEN Tweety:Flies = True

- This method is attached to the IF-NEEDED facet of property "Flies" of the frame "Tweety"

IF-NEEDED Facet Cont.

- To avoid write this method for many other birds, many shells permit to use a variable in place of the frame's name

IF Self: No_Wings < 2

THEN Self: Flies=FALSE

IF Self: No_Wings = 2

THEN Self: Flies=TRUE

IF-CHANGED Facet

IF Self: Hungry=TRUE

THEN Self: Activity= Eating # Self: Eats

- Self: Eats is “eating worms”

Logic

- Logic is the oldest form of knowledge representation in a computer
- One most often linked to intelligent systems are **propositional logic** and **predicate calculus**
- Techniques use symbols to represent knowledge
- Operators applied to the symbols produce logical reasoning

Proposition Logic

- Represent and reason with propositions
- Propositions are true or false statements
- Capturing facts or rules in symbolic forms and operates on them through the use of logical operators

IF The car will not start → A
AND It is too far to walk to work → B
THEN I will miss work today → C

A ^ B → C

Logical operator and symbols

Operator	*Symbol*
AND	$\wedge$, &, $\cap$
OR	$\vee$, $\cup$, +
NOT	$\neg$, ~
IMPLIES	$\supset$, $\rightarrow$
EQUIVALENCE	$\equiv$

A	B	A AND B
F	F	F
F	T	F
T	F	F
T	T	T

A	NOT A
T	F
F	T

A	B	A OR B
F	F	F
F	T	T
T	F	T
T	T	T

A	B	A $\equiv$ B
F	F	T
F	T	F
T	F	F
T	T	T

Implication

A	B	A→B
F	F	T
F	T	T
T	F	F
T	T	T

$$A \rightarrow B \equiv \neg A \vee B$$

IF The battery is dead

THEN The car won't start

- Evaluate implication table row by row

1. battery not dead → car will start; implication T
2. battery not dead → car won't start; implication T
3. battery dead → car will start; implication F
4. battery dead → car won't start; implication T

- For many problems it can be difficult to assert a truth value to an entire statement.

Predicate calculus

- It is an extension of proposition logic that provides a finer representation of the knowledge
- Instead of representing an entire proposition with a single symbol, such as A= Ball's color is red, it used, color(ball, red) or ball(color, red).
- Symbols:
  - Constants
  - Predicates
  - Variables
  - Functions
- Operate on symbols with propositional logic operations

Constants

- They are used to name specific object or properties about the problem
- Begin with lowercase
- Examples:
  - ali, ahmad, temperature,x

Predicates

- A proposition or fact is divided into two parts: a predicate and an argument
- The argument represents the object or objects of the proposition
- The predicate is an assertion about the object denoting a relationship between arguments
- Examples:
  - Proposition: "Ali likes football"
  - likes(ali, football)

Variables

- They represent general classes of objects or properties.
- Begin with uppercase
- Examples:

  “Ali likes football”

  “Ahmad likes basketball”

  likes(X,Y)

  X= ali, X= ahmad

  Y=football, Y=basketball

Functions

- A mapping from entities of a set to a unique element of another set.
- Examples:
  - father (javad) = reza
  - father(hasan)=ali
  - friends(father(javad), father(hasan))=friends(reza,ali)

Operations

- The same operators found in the proposition logic
- Examples:
  - father(hasan, ali), father (hosain,ali)
  - hasan and hosain are brothers
- father(X,Z) ^ father (Y, Z) → brother(X, Y)

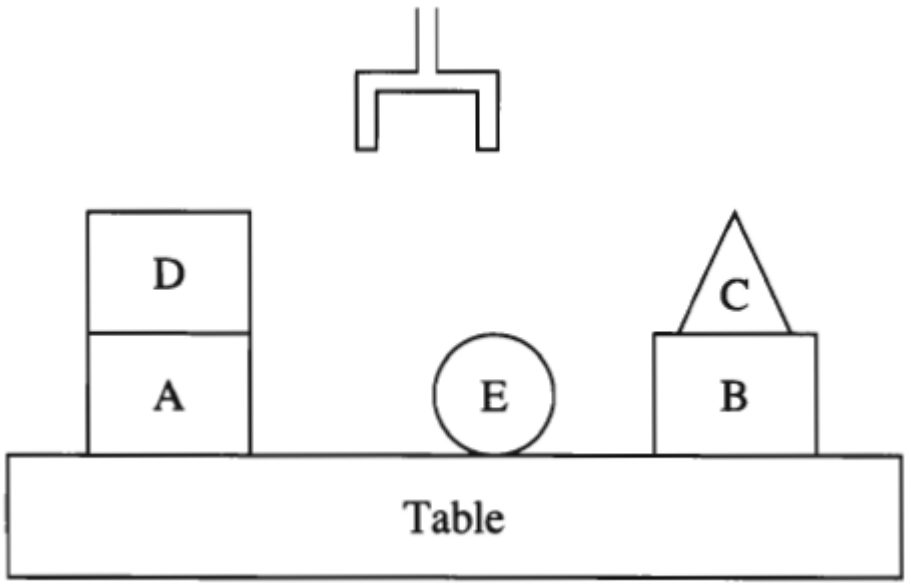
Operations: variable quantifiers

- Universal quantifier ($\forall$) indicates that the expression is TRUE for **all** values of the variable. For Example:
  - For all values of X, the statement is TRUE, that is everyone likes football
  - $\forall$X likes(X, football)
- Existential quantifier ($\exists$) indicates that the expression is TRUE for **some** values of the variable (at least one value exist). For example:
  - At least someone likes football
  - $\exists$X likes(X, football)

Robot control example

- The function of the robot is to move a specific block to a specific location

- cube(a), cube(b), cube(d), pyramid(c), sphere(e), hand(hand), table(table1)
- on(a, table1), on(b, table1), on(d,a), on(c,b), on(e, table1)
- holding(hand, nothing)



World understanding

- Goal, for example, put block b on block a:

put_on(b,a)

holding(hand, b) ^ clear(a) → put_on(b,a)

∀X ∃Y (holding(hand, X) ^ clear(Y) → put_on(X,Y)

∀X (~∃Y on(Y, X)→ clear(X))

- clear(c), clear(d), clear(e)

Conclusion

- O-A-V triples, Rules, and Semantic network were discussed.
- Frames
  - Structure
  - Inheritance
  - Facets
- Logic
  - Propositional logic
  - Predicate calculus