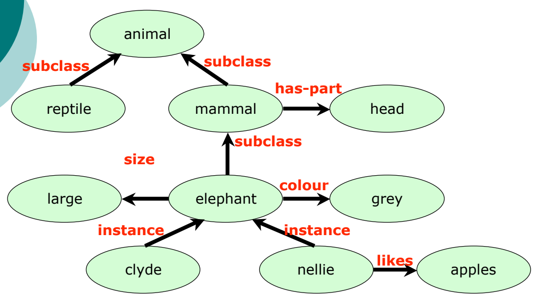
Expert Systems (ES) Week 3

- Semantic networks
- Frames

Semantic networks: An example



Semantic networks (cont.)

- Often referred to as *Semantic Nets*.
- Diagram has 3 types of relationship:
  - **Subclass**: relationship which can be read as "is a kind of" i.e. inheritance e.g. relationship between mammal and animal.
  - **Instance**: relationship which can be read as "is a" or "is an" e.g. clyde is an elephant.
  - **Property**: relationship which is neither subclass or instance but is a feature of an entity e.g. the colour of an elephant is grey.

Semantic networks (cont.)

- A selection of facts represented by the network example are:

  (subclass mammal animal)
  (subclass elephant mammal)
  (large elephant)
  (instance nellie elephant)
  (likes nellie apples)

- Can infer facts such as (has-part nellie head)

Semantic networks (cont.)

- Link classes via subclass relations (kind-of)
- Link classes and objects via instance relations (is-a or is-an)

Tip: if not sure what the link is, just ask yourself whether *kind-of* or *is-a* best describes the link.
E.g. *Nellie is a* kind-of *elephant?* ✗ [Indian/African]
Nellie is-an *elephant?* ✓

- Describe class properties via links to nodes e.g. the link Animal has_legs 4

Semantic networks: Classification

There are many classifications of semantic network. Here are 6 of the most common:

1. Definitional networks.
2. Assertional networks.
3. Implicational networks.
4. Executable networks.
5. Learning networks.
6. Hybrid networks.

Semantic networks: Classification (cont.)

1. Definitional networks

- Emphasise *subtype* i.e. *is-a* relationships.
- Support inheritance to copy all superttype (parent) properties to the subtypes (children).
- Often called a *generalisation* or *subsmption* hierarchy.

2. Assertional networks

- Designed to assert propositions.
- Some networks of this type proposed as models of conceptual structures used in natural language semantics.

Semantic networks: Classification (cont.)

3. Implicational networks

- Use implication as the primary relation for connecting nodes.
- Used to represent patterns of belief, causality, or inferences.

4. Executable networks

- Include a mechanism (e.g. marker passing or attached procedures) to perform inferences, pass messages, or search for patterns and associations.

Semantic networks: Classification (cont.)

5. Learning networks

- Build/extend their representations by acquiring more knowledge from examples.
- New knowledge can add or delete nodes or arcs, or by modifying values (known as *weights*) associated with arcs.

6. Hybrid networks

- Combine two or more of network types 1..5 as either a single network, or two separate but closely interacting networks.

[Taken from *Semantic Networks* by J.F. Sowa]

Semantic networks: Advantages and disadvantages

- ☑ Easy to understand.
- ☑ Easy to construct.
- ☑ Can represent a number of different concepts, e.g. learning by near miss.

- ☒ Complex for more complex systems.
- ☒ Limited domain of usefulness.
- ☒ Less easy to understand for computers than a textual format.

Frames

- Are used to store the information relevant to a particular concept.
- Are similar to record structures.
- Support inheritance.
- Use
  - objects to represent a concept e.g. mammal.
  - slots to represent properties e.g. colour, size.
  - slot-values to represent values e.g. grey, large.
- Are easy to derive from semantic networks.

Frames: An example

```
Mammal:
        subclass:           Animal
        warm-blooded:       yes
        *furry:             yes

Elephant:
        subclass:           Mammal
        has-trunk:          yes
        *colour:            grey
        *size:              large
        *furry:             no

Nellie:
        instance:           Elephant
        colour:             pink
        owner:              Fred
        size:               small
```

Frames: Slots and slot values

Objects: mammal, elephant, Nellie.

For mammal:
- Slots: subclass, warm-blooded furry.
- Slot-values: animal, yes.

Note:
1. "*" indicates a slot whose default value can be overridden.
2. There are other ways of representing default values (see later in the lecture).

Frames: An inference algorithm

To find `value(O,A)`

If `slot-value(O,A)` returns a value V, then return V
Else, if `slot-value(O,subclass) or slot-value(O,instance)` returns C then find `value(C,A)`
Otherwise FAIL.

Note: can be coded in Lisp with suitable data structures to represent objects (*see later*).

Frames: Using the algorithm

Given

```
Elephant:
      subclass:          Mammal
      has-trunk:         yes
      colour:            grey
      size:              large
      habitat:           jungle

Nellie:
      instance:          Elephant
```

Try to find

```
      value (Nellie, colour)
```

Frames: Lisp representation

Can represent frames as association lists in the form

```
defvar frame
`((mammal (limbs, 4))
  (elephant (subclass . mammal)
            (colour . grey))
  (nellie   (instance . elephant)
            (tusks . 2)))
```

Or sets of tuples)

```
`((limbs mammal 4)
  (subclass elephant mammal)
  (colour elephant grey)
  (instance nellie elephant)
  (tusks nellie 2))
```

Frames: Subframes and demons

- Can have subframes as slot values to describe a concept in more detail.
- Can use demons to calculate information as required. The following shows a demon (called *if_needed*) to determine the type of cycle

```
Type:
      If_needed:  IF [one_wheeled] THEN [unicycle]
                  IF [two_wheeled] THEN [bicycle]
                  IF [three_wheeled] THEN [tricycle]
```

Type is a slot and its value resolves to *unicycle*, *bicycle*, or *tricycle*.

Frames (cont.)

- Can use frames for *learning by near miss*.
  - A slot name represents a property.
  - A slot value represents a property value.
- Requires slots to be added or modified (by reinforcement) to represent the learning by near miss concept.
- Have slots which are unordered, unlike scripts which require all slots to be listed in the order in which the subscripts are to be activated.

Frames: Default slot values

- Can use default values for slots for typical situations.
- Can specify/restrict types.
- Default values modified by *cancellation* in a *sub-frame*.

```
Car
    wheels:     integer (default → 4)    [default]
    seats:      integer (default → 4)
    …

Roadster
    subset_of:  car
    seats:      2                         [cancellation]
    …                                     [wheels=4 implicit]
```

Frames: Default slot values (cont.)

- Useful to specify values in a frame which are most likely to be used in a sub-frame.
- Increases efficiency as sub-frames which use the slot with its default value don't have to specify the slot – its inclusion is implicit.

Frames: As slots

Can use a frame as a slot to represent concepts in greater detail, e.g. for a frame describing a restaurant:

Restaurant frame

```
subclass:       Business_establishment
Types:
    Range:     (Cafeteria, Seat_yourself, Wait_to_be_seated)
    Default:    Wait_to_be_seated
    If_needed: IF[stack_of_trays] THEN [Cafeteria]
               IF[wait_here_sign OR reservations_made]
                 THEN [Wait_to_be_seated]
                 OTHERWISE [Seat_yourself]
```

Frames: Slot values

- *Types* is a frame consisting of slots Range, Default, If_needed.
- *Range* shows a set of values. Could also be a valid range of numbers.
- *If_needed* illustrates a conditional slot value using keywords IF, THEN, and OTHERWISE.

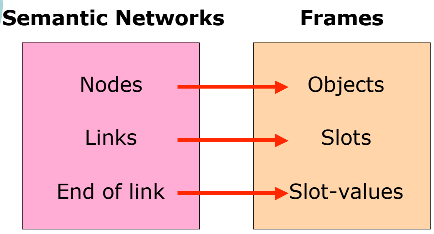
Frames: Meta Knowledge

- Is knowledge about the knowledge represented.
- Is a level of abstraction.
- Meta-K can be captured in a frame: slot becomes the type of meta-knowledge, slot value is the value of the meta-knowledge.
- Meta-K in frames can be used to represent information about the frame itself, for example the person responsible for creating it and when
- Could represent a gauge as to the reliability of the information in the frame.

Frames: Problems

Difficult to represent the following:
- negations
- non-taxonomic knowledge.

Semantic networks and frames: Linking the two representations



Multiple inheritance

Supported by semantic nets and frames.

Example (see next slide for diagram)

- Could represent Clyde as being an elephant and a circus animal.
- Can conclude he's large, has a trunk, and can balance.
- What about habitat?

Multiple inheritance (cont.)

```
Elephant:
      subclass:          Mammal
      has-trunk:         yes
      *colour:           grey
      *size:             large
      *habitat:          jungle

Circus-Animal:
      subclass:          Animal
      habitat:           tent
      skills:            balancing   CONFLICT

Clyde:
      instance:          Circus-Animal  Elephant
      colour:            pink
      owner:             Fred
```

Multiple inheritance: problem resolution

- It's not good enough to define precedence by ordering definitions. *Why?*
- Solution: Need to create a class for Circus-Elephant.

Rule of thumb

Avoid multiple inheritance wherever possible: no multiple inheritance ⇒ no problem

Semantic networks and frames: Advantages and disadvantages

TIP: Think of frames as a being a superset of the functionality of semantic networks.

Advantages

- Simple semantics.
- Easy understand.
- Can be used for inference.
- Can be searched in a tree-like fashion.

Disadvantages

- No support for NOT.
- Can be problematic for multiple inheritance.
- Cannot represent uncertain information.