# Knowledge Representation

# Session Meta Data

| | |
|---|---|
| **Author** | **Dr. D. Thenmozhi** |
| Reviewer | |
| Version Number | 1.2 |
| Release Date | 17 October 2022 |

# Session Objectives

- Understanding knowledge representation in expert system.
- Learn about representing knowledge using if-then rules.

# Session Outcomes

- At the end of this session, participants will be able to
  - explain knowledge representation in expert system using if-then rules

# Agenda

- Knowledge representation
  - If-then rule
  - Example

# REPRESENTING KNOWLEDGE WITH IF-THEN RULES

- Traditionally the most popular form of knowledge representation in expert systems

- Examples of rules:

  - if  condition P then conclusion C
  - if situation S then action A
  - if conditions C1 and C2 hold then condition C does not hold

# Development of an Expert System

- Steps
  - Consult actual experts for that domain and learn a great deal about it yourself.
  - Extracting some understanding of the domain from experts and literature and moulding this understanding in to a chosen knowledge-representation formalism is called the art of knowledge engineering
- Example:
  - Consider a knowledge base.
  - It consists of simple rules that help identify animals from their basic characteristics assuming that the identification problem is limited just to a small number of animals.
  - Rules in this knowledge base are of the form:
    - RuleName : if Condition then Conclusion

# Animal knowledge base

```
% A small knowledge base for identifying animals

:- op( 100, xfx, [has, gives, 'does not', eats, lays, isa] ).
:- op( 100, xf, [swims, flies] ).

rule1 : if
        Animal has hair
        or
        Animal gives milk
    then
        Animal isa mammal.

rule2 : if
        Animal has feathers
        or
        Animal flies and
        Animal lays eggs
    then
        Animal isa bird.

rule3 : if
        Animal isa mammal and
        ( Animal eats meat
          or
          Animal has pointed teeth and
          Animal has claws and
          Animal has 'forward pointing eyes')
    then
        Animal isa carnivore.

rule4 : if
        Animal isa carnivore and
        Animal has 'tawny colour' and
        Animal has 'dark spots'
    then
        Animal isa cheetah.
```

# Animal knowledge base

```
rule5 : if
            Animal isa carnivore and
            Animal has 'tawny colour' and
            Animal has 'black stripes'
        then
            Animal isa tiger.

rule6 : if
            Animal isa bird and
            Animal 'does not' fly and
            Animal swims
        then
            Animal isa penguin.

rule7 : if
            Animal isa bird and
            Animal isa 'good flyer'
        then
            Animal isa albatross.
```

# Animal knowledge base

```
fact : X isa animal  :-
  member( X, [cheetah, tiger, penguin, albatross] ).

askable( _ gives _, 'Animal' gives 'What').
askable( _ flies, 'Animal' flies).
askable( _ lays eggs, 'Animal' lays eggs).
askable( _ eats _, 'Animal' eats 'What').
askable( _ has _, 'Animal' has 'Something').
askable( _ 'does not' _, 'Animal' 'does not' 'DoSomething').
askable( _ swims, 'Animal' swims).
askable( _ isa 'good flier', 'Animal' isa 'good flier').
```

# Developing the shell

- To get the above rules to work, rewrite the rules as Prolog rules

```
Animal isa mammal  :-
  Animal has hair;
  Animal gives milk.

Animal isa carnivore  :-
  Animal isa mammal,
  Animal eats meat.

...
```

A tiger called peter can be confirmed a tiger by adding Prolog facts and properties

```
peter has hair.
peter is lazy.
peter is big.
peter has 'tawny colour'.
peter has 'black stripes'.
peter eats meat.
```

```
?- peter isa tiger.

yes

?- peter isa cheetah.

no
```

# Summary

- Knowledge representation
  - If-then rule
- Identifying animal from the given knowledge base.
  - Facts
  - Rules
  - Askables

# Check your understanding

- Assume that there are no operators defined.
- Rewrite the rules 1 to 7 with unary predicates of flies and swims and binary predicates for all other actions of the animals from animal knowledge base using Prolog.