SNS COLLEGE OF ENGINEERING

Kurumbapalayam(Po), Coimbatore – 641 107
Accredited by NAAC-UGC with 'A' Grade
Approved by AICTE, Recognized by UGC & Affiliated to Anna University, Chennai

Department of AI &DS

Course Name – 23ADT201 ARTIFICIAL INTELLIGENCE

II Year / III Semester

UNIT 4

LOGICAL REASONING

Topic: Knowledge Engineering Overview

CASE STUDY:

- **IBM Watson for Oncology:** IBM Watson uses Knowledge Engineering to analyze patient data and medical literature, providing oncologists with evidence-based treatment options for cancer patients. It leverages vast medical knowledge and learning algorithms to assist in making accurate diagnoses and recommending personalized treatment plans.

Knowledge Engineering in FOL

- Knowledge Engineer is someone who
- investigates a particular domain,
- learns what concepts are important in that domain, and
- creates a formal representation of the objects and relations in the domain.

Knowledge Engineering in FOL...

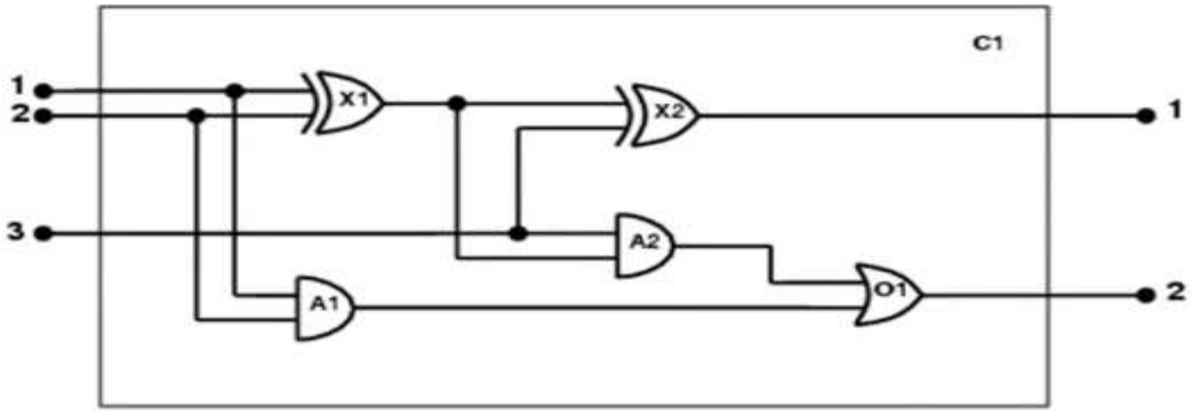
- General purpose knowledge base
  - Support queries about full range of human knowledge.
  - In this we can expect any kind of query, which knowledge base will have to infer.
- Special purpose knowledge base
  - Which has restricted domain (problem specific), here expected queries are known in advance.

Steps in Knowledge Engineering Process

1. Identify the task
2. Assemble the relevant knowledge
3. Decide on a vocabulary of predicates, functions, and constants
4. Encode general knowledge about the domain
5. Encode a description of the specific problem instance
6. Pose queries to the inference procedure and get answers
7. Debug the knowledge base

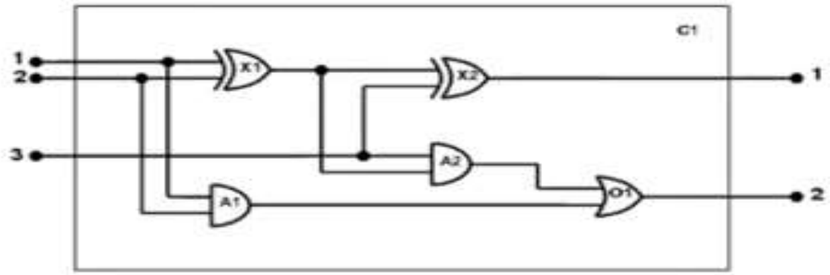
KE in FOL for Electronic Circuits Domain

- One-bit full adder



1. Identify the task.

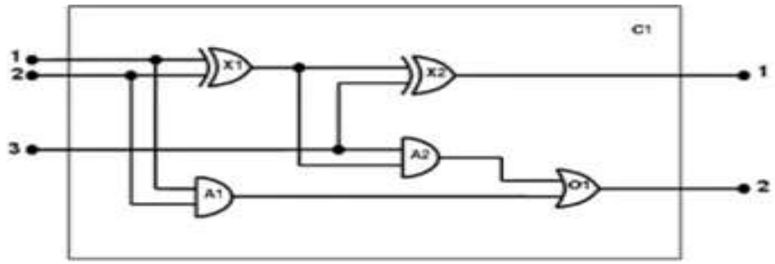
- Identify the task similar to PEAS design.
- Knowledge engineer must describe the range of question that the KB will support
- Find the facts that available for each specific problem instance
  - Does the circuit actually add properly? (circuit verification)



2. Assemble the relevant knowledge

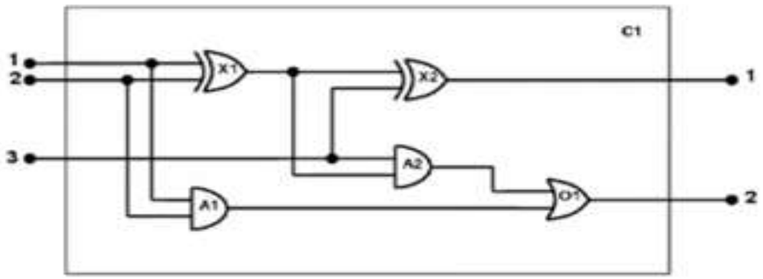
- Composed of wires and gates;
- Types of gates (AND, OR, XOR, NOT)

- Irrelevant: size, shape, color, cost of gates



3. Decide on a Vocabulary

- Translate the important domain level concepts into logic level names.
- Once the choice among predicates, functions and constants have been made, the result is vocabulary, which is Ontology of domain.



Type(X_1) = XOR
Type(X_1, XOR)
XOR(X_1)

Type(X_2) = XOR
Type(X_2, XOR)
XOR(X_2)

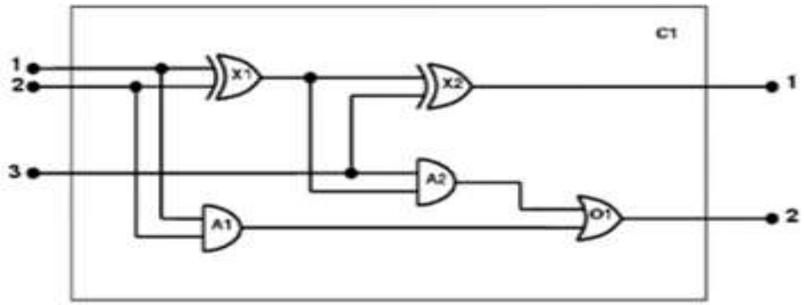
Type(A_1) = AND
Type(A_1, AND)
AND(A_1)

Type(A_2) = AND
Type(A_2, AND)
AND(A_2)

Type(O_1) = OR
Type(O_1, OR)
OR(O_1)

4. Encode General Knowledge Of The Domain

- $\forall t_1,t_2$ Connected(t_1, t_2) $\Rightarrow$ Signal(t_1) = Signal(t_2) (t=terminal, g=gate)
- $\forall t$ Signal(t) = 1 $\vee$ Signal(t) = 0, 1 $\neq$ 0
- $\forall t_1,t_2$ Connected(t_1, t_2) $\Rightarrow$ Connected(t_2, t_1)
- $\forall g$ Type(g) = OR $\Rightarrow$ Signal(Out(1,g)) = 1 $\Leftrightarrow$ $\exists n$ Signal(In(n,g)) = 1
- $\forall g$ Type(g) = AND $\Rightarrow$ Signal(Out(1,g)) = 0 $\Leftrightarrow$ $\exists n$ Signal(In(n,g)) = 0
- $\forall g$ Type(g) = XOR $\Rightarrow$ Signal(Out(1,g)) = 1 $\Leftrightarrow$ Signal(In(1,g)) $\neq$ Signal(In(2,g))
- $\forall g$ Type(g) = NOT $\Rightarrow$ Signal(Out(1,g)) $\neq$ Signal(In(1,g))

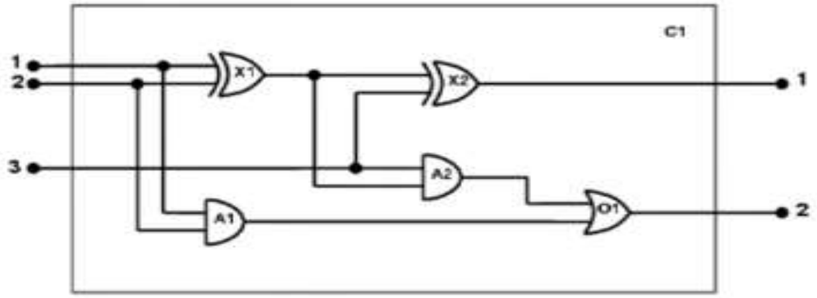


5. Encode The Specific Problem Instance

Type(X_1) = XOR
Type(X_2) = XOR
Type(A_1) = AND
Type(A_2) = AND
Type(O_1) = OR
Type(C_1) = Circuit

Connected(Out(1,X_1),In(1,X_2))
Connected(Out(1,X_1),In(2,A_2))
Connected(Out(1,A_2),In(1,O_1))
Connected(Out(1,A_1),In(2,O_1))
Connected(Out(1,X_2),Out(1,C_1))
Connected(Out(1,O_1),Out(2,C_1))

Connected(In(1,C_1),In(1,X_1))
Connected(In(1,C_1),In(1,A_1))
Connected(In(2,C_1),In(2,X_1))
Connected(In(2,C_1),In(2,A_1))
Connected(In(3,C_1),In(2,X_2))
Connected(In(3,C_1),In(1,A_2))



6. Pose Queries To The Inference Procedure

- What are the possible sets of values of all the terminals for the adder circuit?

$$\exists i_1,i_2,i_3,o_1,o_2\ \text{Signal}(\text{In}(1,C_1)) = i_1 \wedge \text{Signal}(\text{In}(2,C_1)) = i_2 \wedge \text{Signal}(\text{In}(3,C_1)) = i_3 \wedge \text{Signal}(\text{Out}(1,C_1)) = o_1 \wedge \text{Signal}(\text{Out}(2,C_1)) = o_2$$

- There are substitution of variables i1,i2,i3 with values (1/0).
- The final query will return complete with given Input and Output for the device.
- It should be used to check that add inputs correctly
- This is called as circuit verification.

7. Debug the knowledge base

- We have to see the knowledge base in different ways
  - System unable to give output in no signals
  - If all inputs are 000, then the output also 00,
  - And etc.
- May have omitted the assertions like 1 ≠ 0

THANK YOU