KNOWLEDGE REPRESENTATION

Knowledge Representation :- (KR)

It is the process of representing real world knowledge into symbols so AI can understand, store and manipulate

goal: enable AI to think an reason about problems, make decisions, and understand situations by leveraging stored knowledge.

Components :

1 Knowledge Base (KB): repo where knowledge is stored in structured format.

2 Representation Language : formal language or set of symbols used to encode the knowledge. It must be unambiguous and machine understandable.

3 Interface Engine : set of procedures and algorithms that uses the KB to derive new facts, answer queries, or make decision.

Properties of Good KR Schema :

1. Representation Adequacy
2. Inferential Adequacy
3. acquisitional efficiency

types

1> declarative knowledge (facts)
2> Procedural knowledge (How-to)
3> Temporal knowledge (time related inf
4> Uncertain knowledge (info i.e x uncerta
5> Meta- knowledge (knowledge about know-edge
6> Heuristic knowledge (problem -solving)

Representation Schemas

1> Sematic Networks
- represented as graph
- Nodes: obj, ideas, entities)
- Links : relationship between nodes

2> Frame
- more like OOP class
- can hold values, default values, and even procedures

3> Production Rules
- in if - then structure

4> logic - based method
- uses formal logic systems (algo math reasoning etc)

ISSUES IN KNOWLEDGE REPRESENTATION

1> Getting Knowledge : how do get all the lar
and needed and perfect data? hein?

2> Storing large amount of data

3> Updating the large amount of data when needed

4> Mainting rules and avoiding ambiguity

5> Different Representation

6> Frame Problem in logic based KB

7> Sharing knowledge between diff AI system

8> Common Sense stating which leads to large sizes

13> Choosing way to represent

9> Context understanding

12> Balancing Details and Speed

10> Maintaining Consistency

11> Enabling Reasoning

FIRST ORDER LOGIC

a.k.a Predicate Logic is a powerful and expressive formal system used for KR allows us to represent objects, properties of objects, and relationships between objects along with universal and existential qualificat

Before FOI we need a brief about
Propositional Logic: represents facts as atomic propositions. It cannot express relationships between objects
cannot express general statements (all humans are mortal ×)
(we'll need state individually)

components
1 constants: specific objects / individual
e.g. Plato, Delhi, 3, a, John
2. Predicates: properties of objects
relationship between objects
(like boolean functions)
e.g. IsHuman(Socrates) → True
Likes(John, Mary) → True
Greater(5, 3)
3. Functions: represents mapping from one or more object to another object.

e.g. FatherOf (John) → (refers to father ob
Sum(2,3) → refers 5 obj
ColorOf(Sky) → refers to Blue ob

4. Variables: unspecified objects or any o
e.g. x, y, z, p

5. Terms: a obj, constant, function,
e.g. Socrates, x, FatherOf(John), Sum(
3)

6. Atomic Sentences: formed by predicate symbol followed by a parenthesized list of items, they are basic True/Fa statement.
e.g. IsHuman(Socrates)
Likes(John, FatherOf(mary))

7. Connectives (Logical Operators)

AND	^	P^Q	(are true if both true)
OR	v	PvQ	(are true if any true)
NOT	~	~P	true ↔ False
IMPLIES	⇒	P⇒Q	IF P THEN Q
IFF	<=>	P⇔Q	(biconditional) IDK :)

8. Quantifiers
↳ universal (for all) ∀ (forall x)
↳ existential (there exists) ∃

Sematics (Meaning)
define the truth of a FOI sentence in a specific model (or interpretation)

A model has:

1> domain (D) - non-empty set of objects that the constants, variables can refer to

2> Interpretation Function - assigns meaning to each symbol

Each constant/variable refers to D

Each predicate refers to relation over D

Each function refers to a function from D to D

A sentence is true in a model if its meaning, given the intrepretation, evaluates to true.

COMPUTABLE FUNCTIONS AND PREDICATES

computability: concept from theoritical computer science that asks 'Can a problem be by an algorithm?'

It defines the limits of what comps can do

In AI it means whether our represented knowledge can be effectively processed by machine?

Computable Function is a function for which an algorithm exists that can compute its output for any valid input in finite time.

Characteristics

1> algo existance : there must be step by step procedure

2> termination : algo must end

3> deterministic : for same i/p → same o/p

e.g add(x, y) = x + y

multiply (x, y) = x * y

Computable Predicates

is a predicate for which an algo exist that can determine whether predicate is true or false for i/p in finit time.

(special func return True / False)

Characteristics :

Algorithm Existance

termination

Deterministic

Eg IsEven(x) → True / False

Importance of CF and CP

i) foundation for automated Reasoning
ii) practical implementation
iii) Effective Knowledge Representation
iv) Define AI's capabilities

FORWARD / BACKWARD REASONING

The choice of how AI system thinks or reaches a conclusion is often determined by its reasoning strategy.

Forward | Backward

(Data-Driven / Antecedent Driven)

strategy starts with initial facts or data available and applies rules to conclude new facts and continues process unit goal is reached or no more facts (new)

1. looks a its current set of known facts
2. Scans through its set of rules (IF A and B then C)
3. If the 'IF' part matches the current facts, the 'THEN' part is added to the set of facts
4. This process repeats with new added fact sets

5 stops when goal reached or
no new facts can be added

Refer example from PDF notes

Backward Reasoning
(Goal-Driven or Consequent-Driven)
Strategy starts with a specific goal or hypothesis and works backward to find the initial facts or conditions that would prove the goal.
'What do I need to know, to prove this?'

1. starts with goal it wants to prove
2. looks for rules whose 'THEN' part matches the current goal.
3. To prove the goal, it attempts to prove the 'IF' part of one of these matching rules. so these 'IFs' become new subgoals
4. This process continues recursively, trying to prove subgoals, until it reaches facts that are already known to be true

UNIFICATION AND LIFTING

unification and lifting are core concepts in knowledge representation and automated Reason using FOL, they help AI system reason with general rules.

Unification

- process of finding a substitution that mak two logical expressions identical.
- helps applying general rules to specific facts.
- components — variables (x, y, z) — constants $(A, \text{John}, 3)$ — functions — predicate
- goal is find most general Unifier (MGU) (simplest valid substitution)
- Rules
  1. Identical terms → ✓ (no substitution)
  2. Variable v/s Term → substitute if var doesn't appear inside term
  3. Variable v/s variable → substitute one for other
  4. complex term → unify only if same predicate / function name and same entity

e.g. $P(x, A)$ and $P(B, y)$ → MGU $\{x/B, y/A\}$

$P(x, x)$ and $P(A, B)$ → fail

$P(x, f(y))$ and $P(A, f(B))$ → MGU $\{x/A, y/B\}$

Lifting : applying inference rules to general FOL statement (with variables)

- ↑ efficiency and generalization
- avoid creating separate rules for each instance.

uses unification to match rules premises with facts.

Rule : $P1(x) \wedge P2(y) \Rightarrow Q(x,y)$

Fact : $P1(A)$, $P2(B)$

Unify : $\{x/A, y/B\}$

Infer : $Q(A,B)$

matching tool unification = pattern - matching engine (find subs)

reasoning process using the maching tool lifting = reasoning strategy that uses unification to apply general rules

RESOLUTION PROCEDURE

process by which AI concludes facts from new data

resolution is a single, complete inference rule used in automated theorem proving and AI reasoning with FOL

It works through refutation

proving a conclusion by showing that negation leads to a contradiction

requirements

1 Knowledge in FOL
2. all stm must be in Conjuctive Normal Form (AND IF NOT)
3. unification for matching

Steps

1. Convert all statements to CNF
2. Negate the Goal $P \rightarrow \sim P$
3. Apply Resolution Rule
   - find complementary literals (P and $\sim P$)
   - unify if variable exit
   - combine the remaining literals into a new resolvent clause
4. check for empty clause

   ~~fact~~ If found contradiction, its proven.

Applications

Que-Ans Sys   Automated Theorem Proving

Logic Programming   Knowledge based expert Systems.

LOGIC PROGRAMMING

declarative programming paradigm based on formal logic

It focuses on 'what the problem' is not 'how to solve it?'

components

facts (known truths) parent(john, marry)

rules: conditional statements

~~questions~~ queries: questions asked to system

execution process:

- uses inference mechanism like backward chaining
- employs unification to match queries with facts/rules.

example

father(john, marry). → fact

child(x, y) :- father(y, x) → rule

?- child(mary, john) → query

Programming By Stating facts and rules, not procedures.