Unit III

Knowledge representation : types of knowledge in AI, issues in knowledge representation, logic representation, propositional logic, predicate logic, forward chaining and backward chaining

Knowledge Representation

Knowledge Representation

- Representing information about the real world so that a computer can understand and can utilize this knowledge to solve the complex real world problems such as diagnosis a medical condition or communicating with humans in natural language.
- Knowledge representation enables an intelligent machine to learn from that knowledge and experiences so that it can behave intelligently like a human.
- What to Represent:
    - Object
    - Events
    - Performance
    - Meta-knowledge
    - Facts
    - Knowledge-Base

- Knowledge representation in AI refers to the process of encoding knowledge in a manner that enables an artificial intelligence system to reason, understand, and make decisions. It involves capturing information about the world that an AI system needs to perform tasks such as problem-solving, decision-making, understanding natural language, and learning.

Types of Knowledge

- Procedural Knowledge: Procedural knowledge involves knowledge about how to perform specific tasks or actions. Examples include driving a car or baking a cake. Includes Rules, Strategies, Procedures etc.

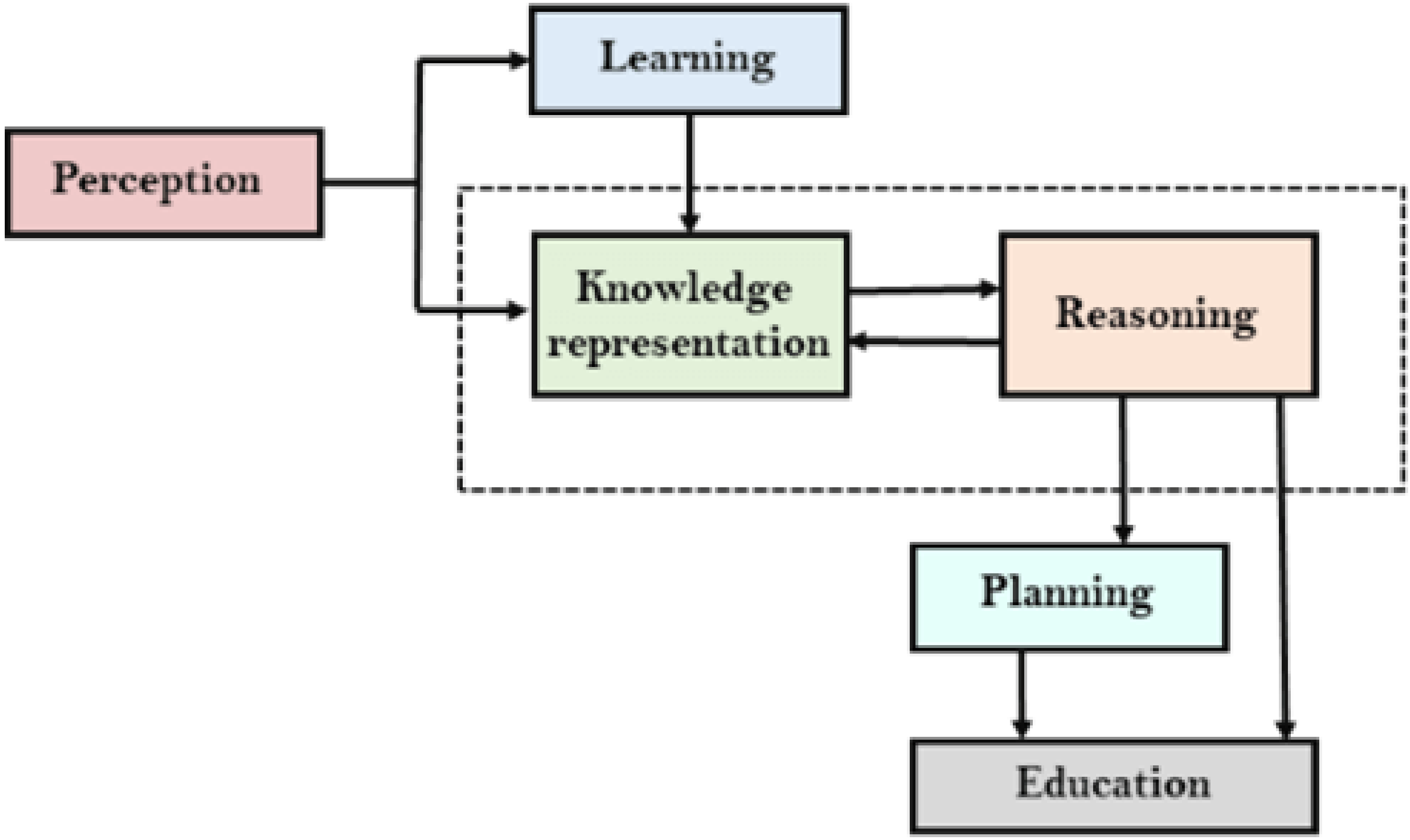
- Declarative Knowledge: Declarative knowledge consists of factual information about the world. Examples include "Paris is the capital of France" or "Water freezes at 0 degrees Celsius." Includes Concepts, Facts, Objects.

- Meta-Knowledge: Meta-knowledge refers to knowledge about knowledge. It includes information about the reliability, sources, or relationships between different pieces of knowledge.

- Structural Knowledge: Structural knowledge represents the relationships and organization of knowledge. Examples include hierarchies, networks, or ontologies.

- Heuristic Knowledge: It is representing knowledge of some experts in a field or subject. Describes rule of thumb that guides reasoning process. Based on previous knowledge, approaches etc.

AI Knowledge Cycle



Approaches to Knowledge Representation

- Simple relational knowledge
- Inheritable knowledge
- Inferential knowledge
- Procedural knowledge

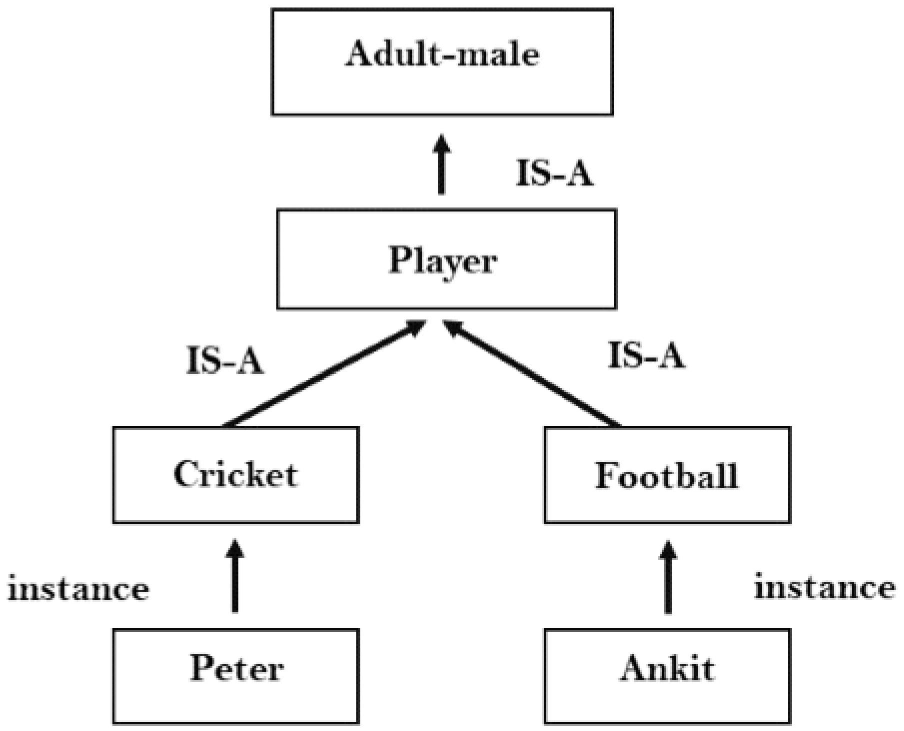
Simple Relational Knowledge

- It is the simplest way of storing facts which uses the relational method, and each fact about a set of the object is set out systematically in columns.
- This approach of knowledge representation is famous in database systems where the relationship between different entities is represented.
- This approach has little opportunity for inference.

Player	Weight	Age
Player1	65	23
Player2	58	18
Player3	75	24

Inheritable Knowledge

- In the inheritable knowledge approach, all data must be stored into a hierarchy of classes.
- All classes should be arranged in a generalized form or a hierarchal manner.
- Elements inherit values from other members of a class.
- Every individual frame can represent the collection of attributes and its value.



Inferential Knowledge

- Inferential knowledge approach represents knowledge in the form of formal logics.
- This approach can be used to derive more facts.
- It guaranteed correctness.
- Example: Let's suppose there are two statements:
  - Marcus is a man
  - All men are mortal
- Then it can represent as;
  - man(Marcus)
  - ∀x = man (x) ----------> mortal (x)s

Procedural Knowledge

- Procedural knowledge approach uses small programs and codes which describes how to do specific things, and how to proceed.
- In this approach, one important rule is used which is If-Then rule.
- In this knowledge, we can use various coding languages such as LISP language and Prolog language.
- We can easily represent heuristic or domain-specific knowledge using this approach.

Issues in Knowledge Representation

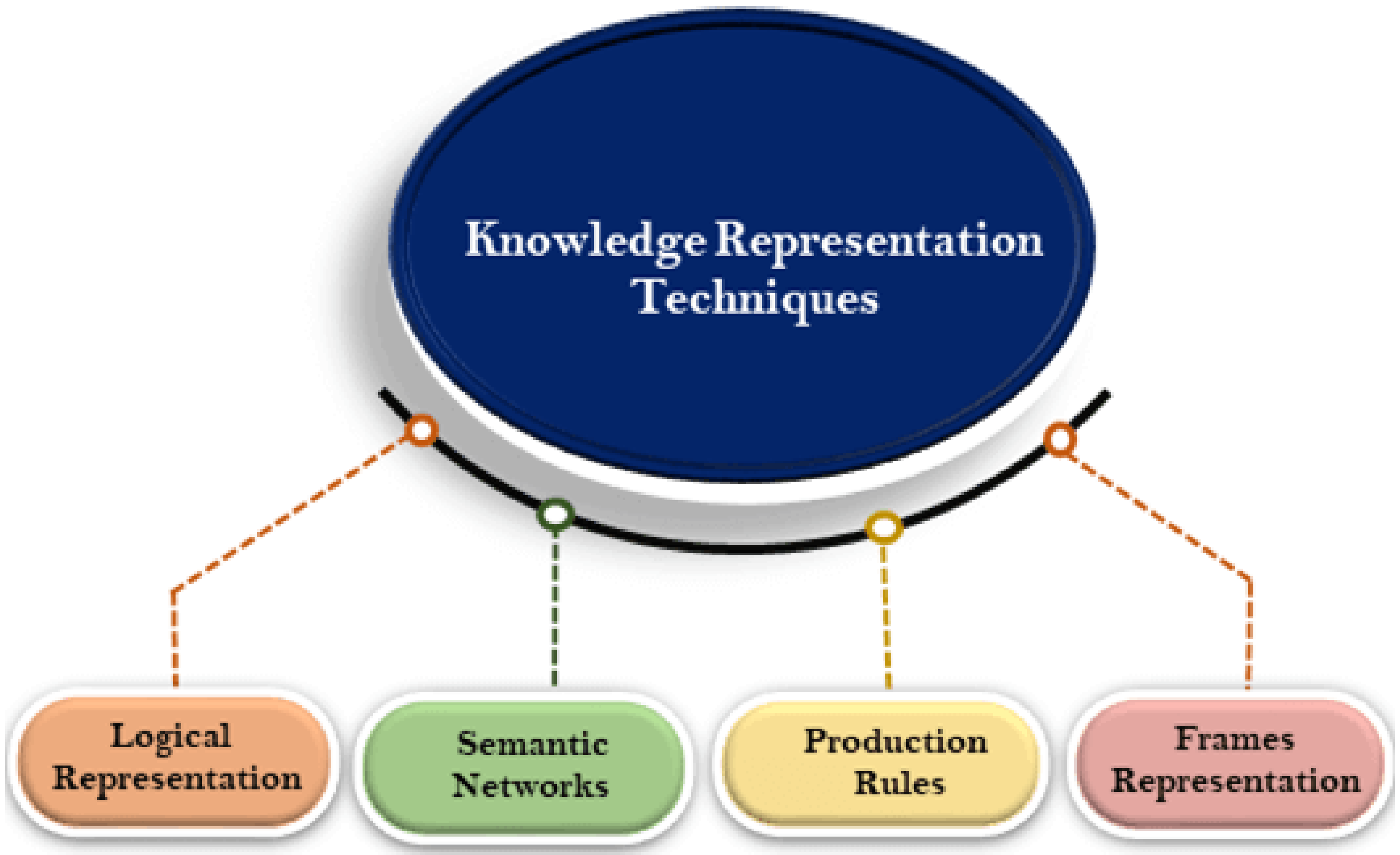
- **Complexity:**
  - Representing complex knowledge domains in a structured and computationally efficient manner poses challenges due to the sheer volume and interconnectedness of information.
- **Uncertainty:**
  - Real-world knowledge often contains uncertainties, probabilities, and fuzzy boundaries, making it challenging to represent and reason with confidently.

- **Incompleteness:**
  - Knowledge representation systems may lack complete information about the domain, leading to gaps and limitations in reasoning and decision-making.

- **Context-dependency:**
  - The meaning and interpretation of knowledge can vary depending on the context in which it is applied, necessitating context-aware representation and reasoning.

- **Scalability:**
  - Representing and managing large volumes of knowledge in AI systems requires scalable solutions to ensure efficient processing and storage capabilities.

Techniques of Knowledge Representation



Logical Representation

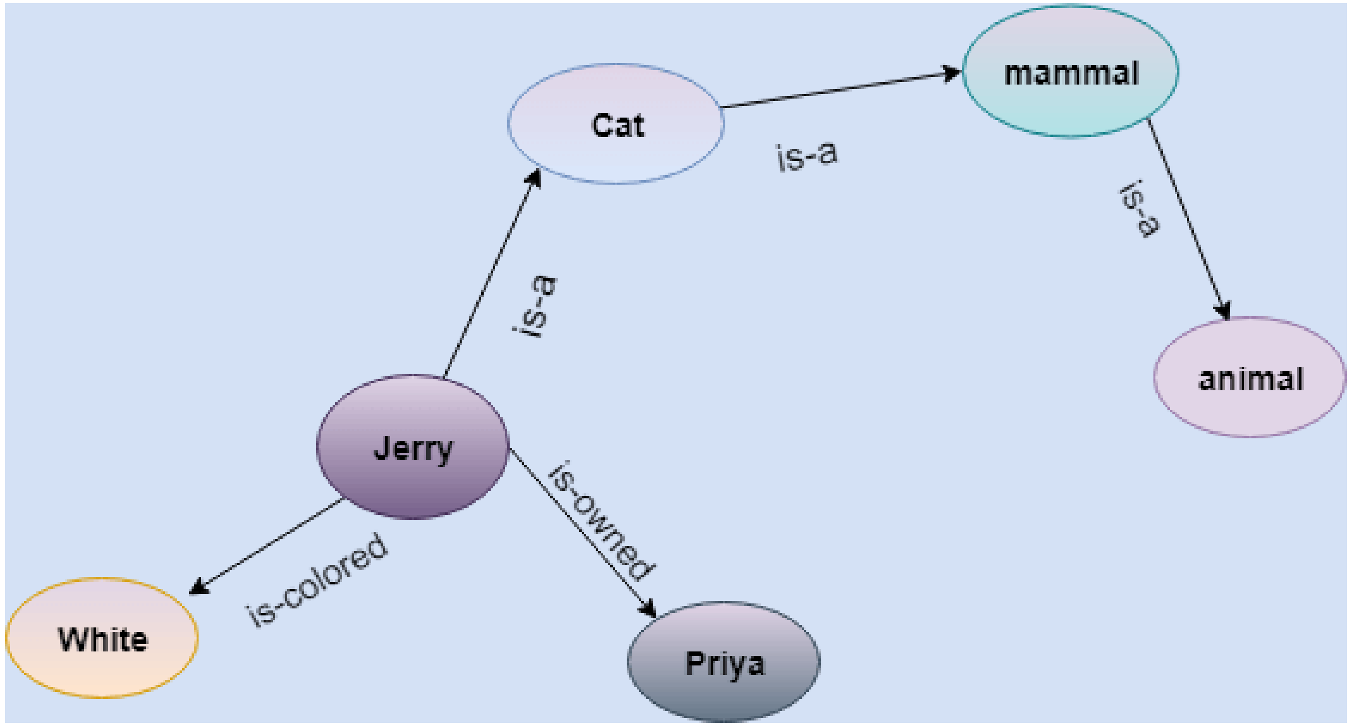
- Uses concrete rules which deals with propositions and has no ambiguity in representation.
- Means drawing a conclusion based on various conditions.
- Logical representation can be categorized into mainly two logics:
  - Propositional Logics
  - Predicate logics

- Advantages of logical representation:
  - Logical representation enables us to do logical reasoning.
  - Logical representation is the basis for the programming languages.

- Disadvantages of logical Representation:
  - Logical representations have some restrictions and are challenging to work with.
  - Logical representation technique may not be very natural, and inference may not be so efficient.

Semantic Network Representation

- Represent knowledge in the form of graphical networks.
- Statements:
  - Jerry is a cat.
  - Jerry is a mammal
  - Jerry is owned by Priya.
  - Jerry is brown colored.
  - All Mammals are animal.



- Advantages of Semantic network:
  - Semantic networks are a natural representation of knowledge.
  - Semantic networks convey meaning in a transparent manner.
  - These networks are simple and easily understandable.

- Drawbacks in Semantic representation:
  - Semantic networks take more computational time at runtime as we need to traverse the complete network tree to answer some questions.
  - Semantic networks try to model human-like memory to store the information, but in practice, it is not possible to build such a vast semantic network.
  - Semantic networks do not have any standard definition for the link names.
  - These networks are not intelligent and depend on the creator of the system.

Frame Representation

- Record like structure which consists of a collection of attributes and its values to describe an entity in the world.

Slots	Filters
Title	Artificial Intelligence
Genre	Computer Science
Author	Peter Norvig
Edition	Third Edition
Year	1996
Page	1152

- Advantages of frame representation:
  - The frame knowledge representation makes the programming easier by grouping the related data.
  - It is very easy to add slots for new attribute and relations.
  - It is easy to include default data and to search for missing values.
  - Frame representation is easy to understand and visualize.

- Disadvantages of frame representation:
  - In frame system inference mechanism is not be easily processed.
  - Inference mechanism cannot be smoothly proceeded by frame representation.

Production Rules

- Production rules system consist of (condition, action) pairs which mean, "If condition then action". It has mainly three parts:
  - The set of production rules
  - Working Memory
  - The recognize-act-cycle
- Example:
  - IF (at bus stop AND bus arrives) THEN action (get into the bus)
  - IF (on the bus AND paid AND empty seat) THEN action (sit down).
  - IF (on bus AND unpaid) THEN action (pay charges).
  - IF (bus arrives at destination) THEN action (get down from the bus).

- Advantages of Production rule:
  - The production rules are expressed in natural language.
  - The production rules are highly modular, so we can easily remove, add or modify an individual rule.

- Disadvantages of Production rule:
  - Production rule system does not exhibit any learning capabilities, as it does not store the result of the problem for the future uses.
  - During the execution of the program, many rules may be active hence rule-based production systems are inefficient.

Forward Chaining

- The Forward-chaining algorithm starts from known facts, triggers all rules whose premises are satisfied, and add their conclusion to the known facts.
- This process repeats until the problem is solved.

Forward Chaining:

- Forward chaining, also known as data-driven or goal-oriented reasoning, starts with the available facts and uses them to infer new conclusions until a goal is reached. Here's how it works:
- **Initialization**: Forward chaining begins with an initial set of known facts or data.
- **Pattern Matching**: The system matches the known facts against the conditions specified in the rules (also known as production rules or if-then rules).
- **Rule Application**: If a rule's conditions match the available facts, the rule's consequent (the "then" part) is executed, adding new facts to the knowledge base.

- **Iterative Process**: The process continues iteratively, applying rules and adding new facts to the knowledge base, possibly triggering the application of more rules.
- **Goal Achievement**: The process continues until a specific goal or set of goals is achieved or until no further inferences can be made.
- **Example**: If a rule states "If it is raining, then take an umbrella," and the fact "It is raining" is known, forward chaining would infer the action "Take an umbrella."
- Forward chaining is often used in systems where the data is readily available, and the goal is to derive conclusions or take actions based on that data.

Backward Chaining

- A backward chaining algorithm is a form of reasoning, which starts with the goal and works backward, chaining through rules to find known facts that support the goal.
- Backward chaining, also known as goal-driven reasoning or backward reasoning, starts with a goal and works backward to find a sequence of steps or rules that lead to the satisfaction of that goal. Here's how it works:

- **Goal Specification**: Backward chaining begins with a specified goal that the system wants to achieve.
- **Pattern Matching**: The system matches the goal against the consequents of the available rules (i.e., the "then" parts).
- **Rule Selection**: If a rule's consequent matches the goal, the system recursively applies backward chaining to determine if the conditions of that rule can be satisfied.
- **Iterative Process**: The process continues iteratively, working backward through the rules and goals until it reaches a point where all conditions are satisfied.
- **Example**: If the goal is "Take an umbrella," backward chaining would work backward to determine if there are any rules that lead to this goal, such as "If it is raining, then take an umbrella."
- Backward chaining is often used in systems where the desired outcome or goal is known, and the system needs to determine what actions or conditions would lead to that outcome.

Logic Representation

How to represent the knowledge into logical form?

- There are two ways to represents the knowledge:
- Propositional Logic
- Predicate Logic

Propositional logic

- Propositional logic (PL) is the simplest form of logic where all the statements are made by propositions.
- A proposition is a declarative statement which is either true or false.
- It is a technique of knowledge representation in logical and mathematical form.

- Example:
  a) It is Sunday.
  b) The Sun rises from West (False proposition)
  c) 3+3= 7(False proposition)
  d) 5 is a prime number.

Kinds of Proposition

- **Atomic Propositions:**
- Atomic propositions are the simple propositions. It consists of a single proposition symbol. These are the sentences which must be either true or false.
  - 2+2 is 4, it is an atomic proposition as it is a true fact.
  - "The Sun is cold" is also a proposition as it is a false fact.
- **Compound propositions**
- Compound propositions are constructed by combining simpler or atomic propositions, using parenthesis and logical connectives.
  - "It is raining today, and street is wet."
  - "Ankit is a doctor, and his clinic is in Mumbai."

Commonly used connectives

Name	Connective	Symbol
Conjunction	AND	&
Inclusive Disjunction	OR	V
Negation	NOT	~
Material Condition	IMPLIES	$\rightarrow$
Material Biconditional	IFF	$\leftrightarrow$
Exclusive Disjunction	XOR	+
Joint Denial	NAND	|
Disjoint Denial	NOR	$\downarrow$

Syntax/Rules of Propositional Logic

- Atomic proposition are represented as A…Z and are known as well form atomic proposition.
- If A and B are well form proposition then, ~A, (A&B), (AVB), (A->B), (A<->B), (A+B), (A|B), (A↓B) are also well form proposition.
- Nothing else is a well form proposition.

Truth table

Table 5.2 Truth tables for logical connectives

A	B	¯A	¯B	AVB	A&B	A→B	A↔B	A⊕B	A:B	A:B
T	T	F	F	T	T	T	T	F	F	F
F	T	T	F	T	F	T	F	T	T	F
T	F	F	T	T	F	F	F	T	T	F
F	F	T	T	F	F	T	T	F	T	T

Logical Equivalence

- Two propositions are logically equivalence if they have the same truth table for all combinations.
- $(A \to B) \Leftrightarrow (\sim A \lor B)$

Table 5.3 Some commonly used logical equivalences

1.	A	is logically equivalent to	(~~A)
		is logically equivalent to	(A & A)
2.	(A & B)	is logically equivalent to	(B & A)
3.	(A V B)	is logically equivalent to	(B V A)
4.	(A & (B & C))	is logically equivalent to	((A & B) & C)
	(AV(BVC))	is logically equivalent to	((AVB)VC)
5.	(A & (BVC))	is logically equivalent to	((A & B)V(A & C))
	(AV(B & C))	is logically equivalent to	((AVB) & (AVC))
6.	~(A & B)	is logically equivalent to	(~A V ~B)
	~(AVB)	is logically equivalent to	(~A & ~B)
7.	(A→B)	is logically equivalent to	(~A V B)
	(A→B)	is logically equivalent to	~(A & ~B)
	(A→B)	is logically equivalent to	(~B → ~A)
8.	(A→(B→C))	is logically equivalent to	((A→B)→C)
9.	(A ↔ B)	is logically equivalent to	(A & B)V(~A & ~B)
	(A ↔ B)	is logically equivalent to	(A→B) & (B→A)

Tautology and Contradiction

- If the result of a proposition is TRUE for all possible combination, it is known as tautology.
- The dog is either brown, or the dog is not brown.
- $(X \rightarrow Y) \vee (Y \rightarrow X)$
- If the result of a proposition is FALSE for all possible combination, it is known as contradiction.
- $\sim(X \rightarrow Y) \vee (Y \rightarrow X)$

Properties of Operators

- Commutativity:
  - $P \wedge Q = Q \wedge P$, or
  - $P \vee Q = Q \vee P$.
- Associativity:
  - $(P \wedge Q) \wedge R = P \wedge (Q \wedge R)$,
  - $(P \vee Q) \vee R = P \vee (Q \vee R)$
- Identity element:
  - $P \wedge True = P$,
  - $P \vee True = True$.
- Distributive:
  - $P \wedge (Q \vee R) = (P \wedge Q) \vee (P \wedge R)$.
  - $P \vee (Q \wedge R) = (P \vee Q) \wedge (P \vee R)$.
- DE Morgan's Law:
  - $\neg (P \wedge Q) = (\neg P) \vee (\neg Q)$
  - $\neg (P \vee Q) = (\neg P) \wedge (\neg Q)$.
- Double-negation elimination:
  - $\neg (\neg P) = P$.

Show that the following proposition is a Tautology.

- $\{(P \vee \sim Q) \wedge (\sim P \vee \sim Q)\} \vee Q$

Limitations of Propositional logic

- We cannot represent relations like ALL, some, or none with propositional logic. Example:
  - All the girls are intelligent.
  - Some apples are sweet.
- Propositional logic has limited expressive power.
- In propositional logic, we cannot describe statements in terms of their properties or logical relationships.

First-Order Logic or Predicate Logic

- A predicate is defined as a relation that binds two atom together.
- E.g.: Bhaskar like airplane can be written as: **LIKES(Bhaskar, airplane)**
- Here, LIKES is a **predicate** and Bhaskar and airplane are **atoms**.
- So the general form can be written as: **LIKES(x,y)**
- E.g.: Ravi’s father is Rohits’s father**.** **FATHER(father(Ravi), Rohit)**
- Terms: These are the arguments in a predicate.

- Terms can be defined as:
    - A constant is a term.
    - A variable is a term.
    - A function f defined as f(x1, x2…) is also a term.
- It assumes the following things in the world:
    - Objects: A, B, people, numbers, colors, wars, theories, squares, pits, wumpus, ...
    - Relations: It can be unary relation such as: red, round, is adjacent, or n-any relation such as: the sister of, brother of, has color, comes between
    - Function: Father of, best friend, third inning of, end of, ...

Basic Elements of First-order logic

Constant	1, 2, A, John, Mumbai, cat,....
Variables	x, y, z, a, b,....
Predicates	Brother, Father, >,....
Function	sqrt, LeftLegOf, ...
Connectives	∧, ∨, ¬, ⇒, ⇔
Equality	==
Quantifier	∀, ∃

Quantifiers

- It is a symbol that permits to declare or identify the range or scope of the variable in a logical expression.
- There are 2 quantifiers:
- Universal Quantifier (∀): a) All a b) Each a c) Every a
- Existential Quantifier (∃): a) Some b b) at least one b c) exists a b

All man drink coffee

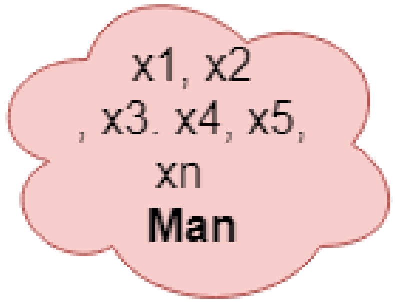
- **x1 drinks coffee**

  ∧
- **x2 drinks**

  ∧
- **x3 drinks milk**

  ∧
- .
- .

  ∧
- **xn drinks milk**



Universe of Discourse

So in shorthand notation, we can write it as :

$\forall x\ man(x) \rightarrow drink\ (x, coffee).$

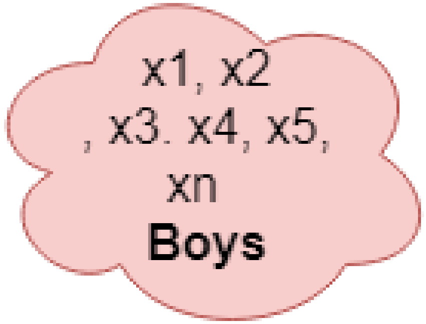
Some boys are intelligent

- **x1 is intelligent**

  V
- **x2 is intelligentV**
- **x3 is intelligent**

  V
- **.**
- **.**

  V
- **xn is intelligent**



Universe of Discourse

∃x: boys(x) ∧ intelligent(x)

So in short-hand notation, we can write it as:

Points to remember

- The main connective for universal quantifier $\forall$ is implication $\rightarrow$.
- The main connective for existential quantifier $\exists$ is and $\wedge$.
- In universal quantifier, $\forall x \forall y$ is similar to $\forall y \forall x$.
- In Existential quantifier, $\exists x \exists y$ is similar to $\exists y \exists x$.
- $\exists x \forall y$ is not similar to $\forall y \exists x$.

Free and Bound Variable

- Free: If its occurrence is outside the scope of quantifier
- Bound: If its occurrence is within the scope of quantifier
- E.g.: $\forall x\ (A(x) \rightarrow B(x))$, change in the x will also effect the values of A(x) and B(x), hence the variable x is a bound variable.
- E.g.: $\forall x\ (A(x, y) \rightarrow B(x, y))$, change in the x will not effect the values of y, hence the variable y is a free variable but x is bound variable.

Atomic sentences

- Atomic sentences are formed from a predicate symbol followed by a parenthesis with a sequence of terms.
- We can represent atomic sentences as Predicate (term1, term2, ..., term n).
- Example: Rohit and Rahul are brothers: => Brothers(Ravi, Ajay).

  Rita is a cat: => cat (Rita).

Complex Sentences

- Complex sentences are made by combining atomic sentences using connectives.
- First-order logic statements can be divided into two parts:
  - Subject: Subject is the main part of the statement.
  - Predicate: A predicate can be defined as a relation, which binds two atoms together in a statement.
- Consider the statement: "x is an integer.", it consists of two parts, the first part x is the subject of the statement and second part "is an integer," is known as a predicate.

Some Examples

- All birds fly.
- $\forall x \; bird(x) \rightarrow fly(x)$.

- Every man respects his parent.
- $\forall x \; man(x) \rightarrow respects\,(x, parent)$.

- Some boys play cricket.
- $\exists x \; boys(x) \rightarrow play(x, cricket)$.

- Not all students like both Mathematics and Science.
- $\neg\forall\,(x)\,[\,student(x) \rightarrow like(x, Mathematics) \wedge like(x, Science)]$.

- Only one student failed in Mathematics.
- $\exists(x)\,[\,student(x) \rightarrow failed\,(x, Mathematics) \wedge \forall\,(y)\,[\neg(x{==}y) \wedge student(y) \rightarrow \neg failed\,(x, Mathematics)]$.

Thank You !

Any Queries ?