Chapter 5: Knowledge Representations

What is knowledge representation?

Humans are best at understanding, reasoning, and interpreting knowledge. Humans know things, which is knowledge and as per their knowledge they perform various actions in the real world. **But how machines do all these things comes under knowledge representation and reasoning**. Hence we can describe Knowledge representation as following:

- Knowledge representation and reasoning (KR, KRR) is the part of Artificial intelligence which is concerned with AI agents thinking and how thinking contributes to intelligent behavior of agents.
- It is responsible for representing information about the real world so that a computer can understand and can utilize this knowledge to solve complex real world problems such as diagnosing a medical condition or communicating with humans in natural language.
- It is also a way which describes how we can represent knowledge in artificial intelligence. Knowledge representation is not just storing data into some database, but it also enables an intelligent machine to learn from that knowledge and experiences so that it can behave intelligently like a human.

What to Represent:

Following are the kind of knowledge which needs to be represented in AI systems:

- **Object:** All the facts about objects in our world domain. E.g., Guitars contain strings, trumpets are brass instruments.
- **Events:** Events are the actions which occur in our world.
- **Performance:** It describes behavior which involves knowledge about how to do things.
- **Meta-knowledge:** It is knowledge about what we know.
- **Facts:** Facts are the truths about the real world and what we represent.
- **Knowledge-Base:** The central component of the knowledge-based agents is the knowledge base. It is represented as KB. The Knowledgebase is a group of the

Sentences (Here, sentences are used as a technical term and not identical with the English language).

Knowledge: Knowledge is awareness or familiarity gained by experiences of facts, data, and situations. Following are the types of knowledge in artificial intelligence:

Types of knowledge

Following are the various types of knowledge:

1. Declarative Knowledge:

- Declarative knowledge is to know about something.
- It includes concepts, facts, and objects.
- It is also called descriptive knowledge and expressed in declarative sentences.
- It is simpler than procedural language.

2. Procedural Knowledge

- It is also known as imperative knowledge.
- Procedural knowledge is a type of knowledge which is responsible for knowing how to do something.
- It can be directly applied to any task.
- It includes rules, strategies, procedures, agendas, etc.
- Procedural knowledge depends on the task on which it can be applied.

3. Meta-knowledge:

- Knowledge about the other types of knowledge is called Meta-knowledge.

4. Heuristic knowledge:

- Heuristic knowledge is representing knowledge of some experts in a field or subject.
- Heuristic knowledge is rules of thumb based on previous experiences, awareness of approaches, and which are good to work but not guaranteed.

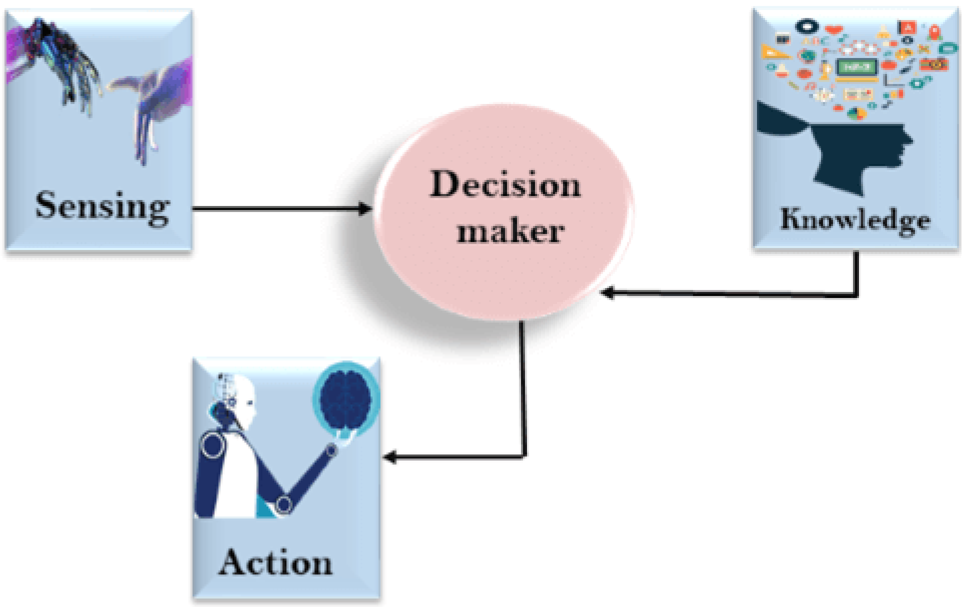
5. Structural knowledge:

- Structural knowledge is basic knowledge of problem-solving.
- It describes relationships between various concepts such as kind of, part of, and grouping of something.
- It describes the relationship that exists between concepts or objects.

The relation between knowledge and intelligence:

- Knowledge of real-worlds plays a vital role in intelligence and the same for creating artificial intelligence. Knowledge plays an important role in demonstrating intelligent behavior in AI agents. An agent is only able to accurately act on some input when he has some knowledge or experience about that input.
- Let's suppose if you met some person who is speaking in a language which you don't know, then how you will be able to act on that. The same thing applies to the intelligent behavior of the agents.

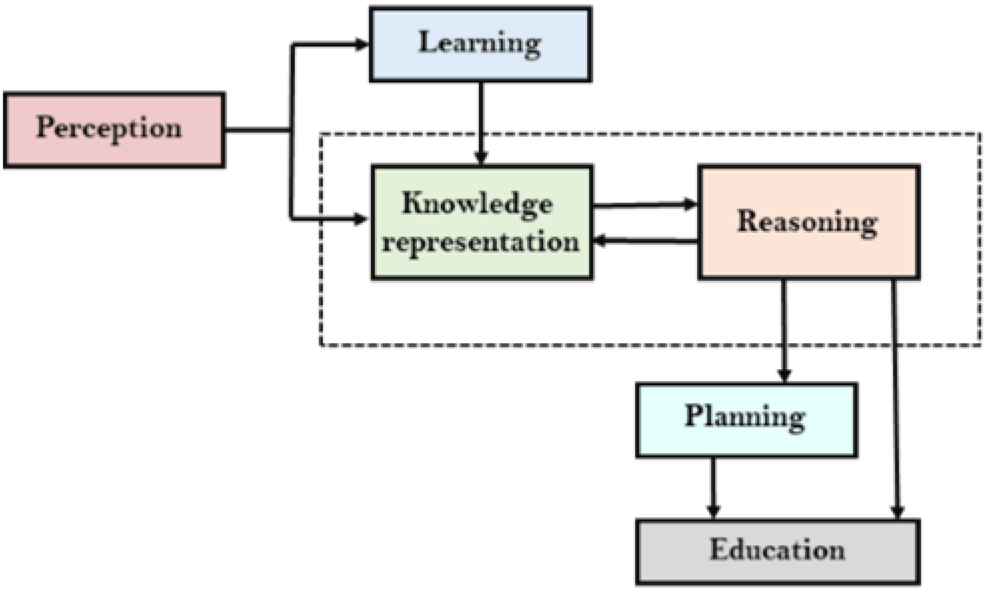
- As we can see in the diagram below, there is one decision maker which acts by sensing the environment and using knowledge. But if the knowledge part is not present then, it cannot display intelligent behavior.



AI knowledge cycle:

An Artificial intelligence system has the following components for displaying intelligent behavior:

- **Perception**
- **Learning**
- **Knowledge Representation and Reasoning**
- **Planning**
- **Execution**



The above diagram is showing how an AI system can interact with the real world and what components help it to show intelligence. AI systems have a Perception component by which it retrieves information from its environment. It can be visual, audio or another form of sensory input. The learning component is responsible for learning from data captured by Perception comportment. In the complete cycle, the main components are knowledge representation and Reasoning. These two components are involved in showing intelligence in machine-like humans. These two components are independent with each other but also coupled together. The planning and execution depend on analysis of Knowledge representation and reasoning.

Approaches to knowledge representation:

There are mainly four approaches to knowledge representation, which are given below:

1. Simple relational knowledge:

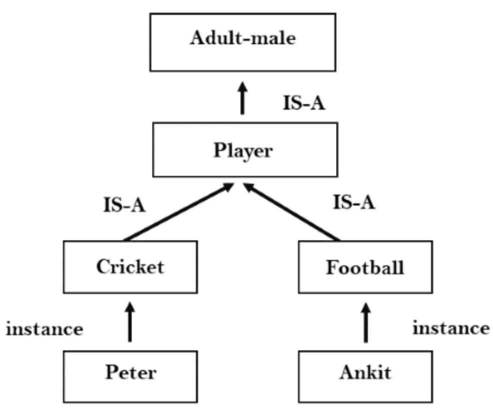
- It is the simplest way of storing facts which uses the relational method, and each fact about a set of the object is set out systematically in columns.
- This approach of knowledge representation is famous in database systems where the relationship between different entities is represented.
- This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

Example: The following is the simple relational knowledge representation.

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

2. 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 hierarchical manner.
- In this approach, we apply inheritance property.
- Elements inherit values from other members of a class.
- This approach contains inheritable knowledge which shows a relation between instance and class, and it is called instance relation.
- Every individual frame can represent the collection of attributes and its value.
- In this approach, objects and values are represented in Boxed nodes.
- We use Arrows which point from objects to their values.
- **Example:**



3. 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:
  a. Marcus is a man
  b. All men are mortal
  Then it can represent as;

  man(Marcus)
  ∀x = man (x) ----------> mortal (x)s

4. Procedural knowledge:

- Procedural knowledge approach uses small programs and codes which describe how to do specific things, and how to proceed.
- In this approach, one important rule is used which is the 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.
- But it is not necessary that we can represent all cases in this approach.

Requirements for knowledge Representation system:

A good knowledge representation system must possess the following properties.

1. Representational Accuracy:
The KR system should have the ability to represent all kinds of required knowledge.

2. Inferential Adequacy:
The KR system should have the ability to manipulate the representational structures to produce new knowledge corresponding to existing structures.

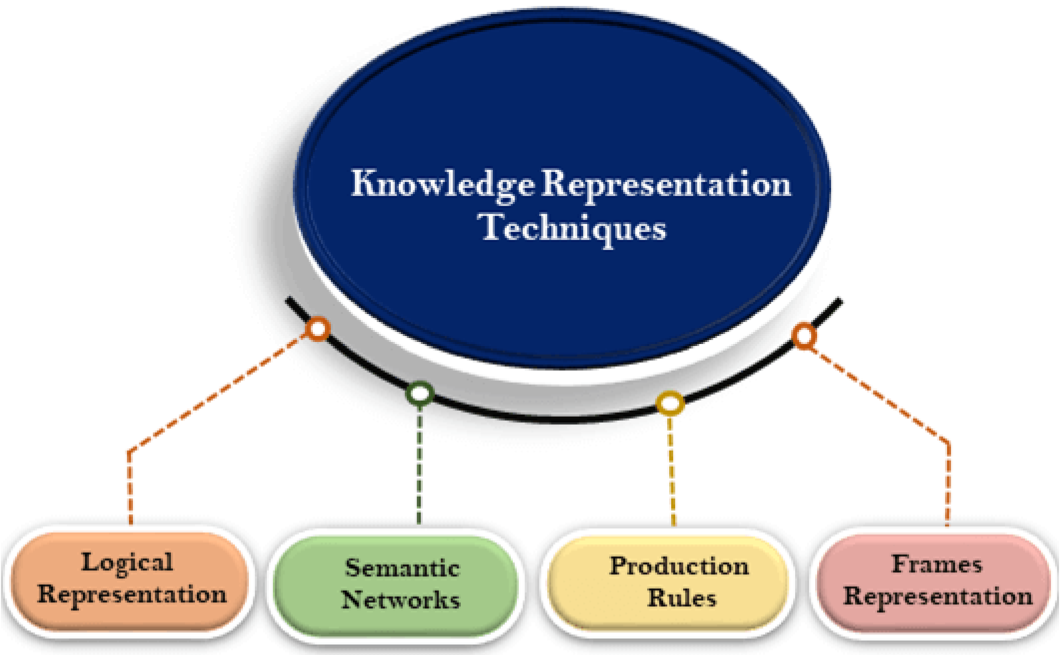
3. Inferential Efficiency:
The ability to direct the inferential knowledge mechanism into the most productive directions by storing appropriate guides.

4. Acquisitional efficiency- The ability to acquire new knowledge easily using automatic methods.

Techniques of knowledge representation

There are mainly four ways of knowledge representation which are given as follows:

1. Logical Representation
2. Semantic Network Representation
3. Frame Representation
4. Production Rules



1. Logical Representation

Logical representation is a language with some concrete rules which deals with propositions and has no ambiguity in representation. Logical representation means drawing a conclusion based on various conditions. This representation lays down some important communication rules. It consists of precisely defined syntax and semantics which supports the sound inference. Each sentence can be translated into logic using syntax and semantics.

Syntax:

- Syntaxes are the rules which decide how we can construct legal sentences in logic.
- It determines which symbol we can use in knowledge representation.
- How to write those symbols.

Semantics:

- Semantics are the rules by which we can interpret the sentence in the logic.
- Semantic also involves assigning a meaning to each sentence.

Logical representation can be categorized into mainly two logics:

a. Propositional Logics
b. Predicate logics

> Note: We will discuss Propositional Logics and Predicate logics in later chapters.

Advantages of logical representation:

1. Logical representation enables us to do logical reasoning.
2. Logical representation is the basis for the programming languages.

Disadvantages of logical Representation:

1. Logical representations have some restrictions and are challenging to work with.
2. Logical representation technique may not be very natural, and inference may not be so efficient.

> Note: Do not be confused with logical representation and logical reasoning as logical representation is a representation language and reasoning is a process of thinking logically.

2. Semantic Network Representation

Semantic networks are alternatives to predicate logic for knowledge representation. In Semantic networks, we can represent our knowledge in the form of graphical networks. This network consists of nodes representing objects and arcs which describe the relationship between those objects. Semantic networks can categorize the object in different forms and can also link those objects. Semantic networks are easy to understand and can be easily extended.

This representation consist of mainly two types of relations:

a. IS-A relation (Inheritance)
b. Kind-of-relation

Example: Following are some statements which we need to represent in the form of nodes and arcs.

Statements:

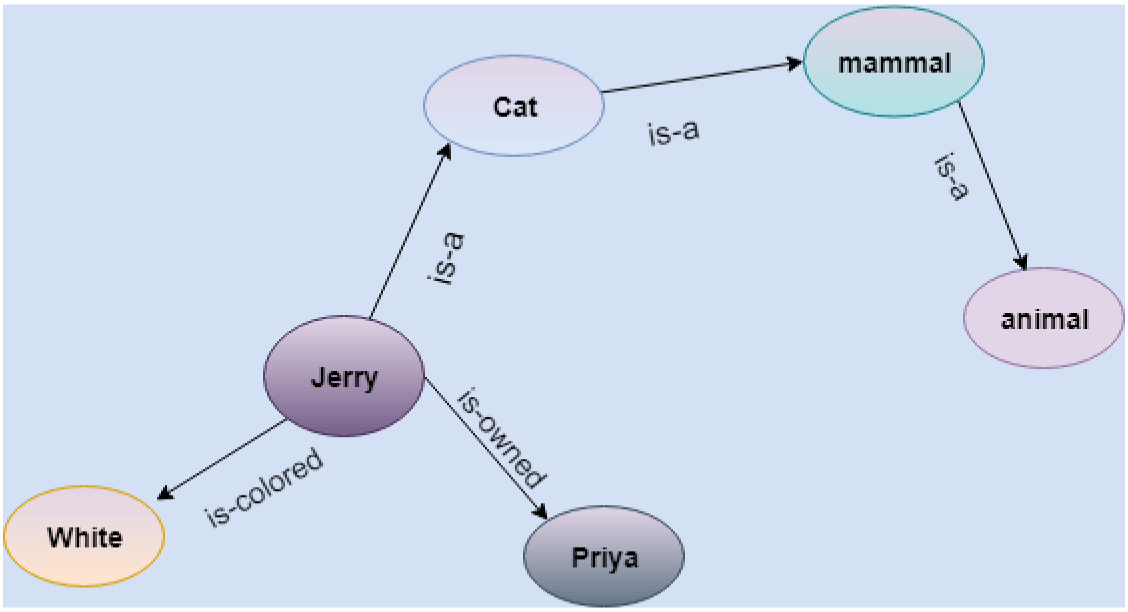
a. Jerry is a cat.

b. Jerry is a mammal

c. Jerry is owned by Priya.

d. Jerry is brown colored.

e. All Mammals are animals.



In the above diagram, we have represented the different type of knowledge in the form of nodes and arcs. Each object is connected with another object by some relation.

Drawbacks in Semantic representation:

1. Semantic networks take more computational time at runtime as we need to traverse the complete network tree to answer some questions. It might be possible in the worst case scenario that after traversing the entire tree, we find that the solution does not exist in this network.

2. Semantic networks try to model human-like memory (Which has 1015 neurons and links) to store the information, but in practice, it is not possible to build such a vast semantic network.
3. These types of representations are inadequate as they do not have any equivalent quantifier, e.g., for all, for some, none, etc.
4. Semantic networks do not have any standard definition for the link names.
5. These networks are not intelligent and depend on the creator of the system.

Advantages of Semantic network:

1. Semantic networks are a natural representation of knowledge.
2. Semantic networks convey meaning in a transparent manner.
3. These networks are simple and easily understandable.

3. Frame Representation

A frame is a record-like structure which consists of a collection of attributes and its values to describe an entity in the world. Frames are the AI data structure which divides knowledge into substructures by representing stereotypical situations. It consists of a collection of slots and slot values. These slots may be of any type and sizes. Slots have names and values which are called facets.

Facets: The various aspects of a slot are known as **Facets**. Facets are features of frames which enable us to put constraints on the frames. Example: IF-NEEDED facts are called when data of any particular slot is needed. A frame may consist of any number of slots, and a slot may include any number of facets and facets may have any number of values. A frame is also known as **slot-filter knowledge representation** in artificial intelligence.

Frames are derived from semantic networks and later evolved into our modern-day classes and objects. A single frame is not very useful. Frames systems consist of a collection of frames which are connected. In the frame, knowledge about an object or event can be stored together in the knowledge base. The frame is a type of technology which is widely used in various applications including Natural language processing and machine visions.

Example: 1

Let's take an example of a frame for a book

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

Example 2:

Let's suppose we are taking an entity, Peter. Peter is an engineer as a profession, and his age is 25, he lives in the city of London, and the country is England. So following is the frame representation for this:

Slots	Filter
Name	Peter
Profession	Doctor
Age	25
Marital status	Single
Weight	78

Advantages of frame representation:

1. The frame knowledge representation makes the programming easier by grouping the related data.

2. The frame representation is comparably flexible and used by many applications in AI.
3. It is very easy to add slots for new attributes and relations.
4. It is easy to include default data and to search for missing values.
5. Frame representation is easy to understand and visualize.

Disadvantages of frame representation:

1. In frame system inference mechanism is not easily processed.
2. Inference mechanism cannot be smoothly proceeded by frame representation.
3. Frame representation has a much generalized approach.

4. Production Rules

Production rules system consists 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

In production rules agent checks for the condition and if the condition exists then production rule fires and corresponding action is carried out. The condition part of the rule determines which rule may be applied to a problem. And the action part carries out the associated problem-solving steps. This complete process is called a recognize-act cycle.

The working memory contains the description of the current state of problems-solving and rules can write knowledge to the working memory. This knowledge matches and may fire other rules.

If there is a new situation (state) generated, then multiple production rules will be fired together, this is called conflict set. In this situation, the agent needs to select a rule from these sets, and it is called a conflict resolution.

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:

1. The production rules are expressed in natural language.
2. The production rules are highly modular, so we can easily remove, add or modify an individual rule.

Disadvantages of Production rule:

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

Propositional logic in Artificial intelligence

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:

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

Following are some basic facts about propositional logic:

- Propositional logic is also called Boolean logic as it works on 0 and 1.

- In propositional logic, we use symbolic variables to represent the logic, and we can use any symbol to represent a proposition, such as A, B, C, P, Q, R, etc.
- Propositions can be either true or false, but it cannot be both.
- Propositional logic consists of an object, relations or function, and **logical connectives**.
- These connectives are also called logical operators.
- The propositions and connectives are the basic elements of the propositional logic.
- Connectives can be said as a logical operator which connects two sentences.
- A proposition formula which is always true is called **tautology**, and it is also called a valid sentence.
- A proposition formula which is always false is called **Contradiction**.
- A proposition formula which has both true and false values is called
- Statements which are questions, commands, or opinions are not propositions such as "**Where is Rohini**", "**How are you**", "**What is your name**", are not propositions.

Syntax of propositional logic:

The syntax of propositional logic defines the allowable sentences for the knowledge representation. There are two types of Propositions:

a. **Atomic Propositions**
b. **Compound propositions**

- **Atomic Proposition:** Atomic propositions are simple propositions. It consists of a single proposition symbol. These are the sentences which must be either true or false.

Example:

1. a) 2+2 is 4, it is an atomic proposition as it is a **true** fact.

2. b) "The Sun is cold" is also a proposition as it is a **false** fact.

- **Compound proposition:** Compound propositions are constructed by combining simpler or atomic propositions, using parentheses and logical connectives.

Example:

1. a) "It is raining today, and the street is wet."
2. b) "Ankit is a doctor, and his clinic is in Mumbai."

Logical Connectives:

Logical connectives are used to connect two simpler propositions or represent a sentence logically. We can create compound propositions with the help of logical connectives. There are mainly five connectives, which are given as follows:

1. **Negation:** A sentence such as $\neg$ P is called negation of P. A literal can be either Positive literal or negative literal.
2. **Conjunction:** A sentence which has $\wedge$ connective such as, **P $\wedge$ Q** is called a conjunction.
   Example: Rohan is intelligent and hardworking. It can be written as,
   P= Rohan is intelligent,
   Q= Rohan is hardworking. → P$\wedge$ Q.
3. **Disjunction:** A sentence which has $\vee$ connective, such as **P $\vee$ Q**. is called disjunction, where P and Q are the propositions.
   Example: "Ritika is a doctor or Engineer",
   Here P= Ritika is Doctor. Q= Ritika is Doctor, so we can write it as **P $\vee$ Q**.
4. **Implication:** A sentence such as P → Q, is called an implication. Implications are also known as if-then rules. It can be represented as

   If it is raining, then the street is wet.

   Let P= It is raining, and Q= Street is wet, so it is represented as P → Q

5. **Biconditional:** A sentence such as **P⇔ Q is a Biconditional sentence, example If I am breathing, then I am alive**

   P= I am breathing, Q= I am alive, it can be represented as P ⇔ Q.

Following is the summarized table for Propositional Logic Connectives:

Connective symbols	Word	Technical term	Example
∧	AND	Conjunction	A ∧ B
∨	OR	Disjunction	A ∨ B
→	Implies	Implication	A → B
⇔	If and only if	Biconditional	A⇔ B
¬ or ~	Not	Negation	¬ A or ¬ B

Truth Table:

In propositional logic, we need to know the truth values of propositions in all possible scenarios. We can combine all the possible combinations with logical connectives, and the representation of these combinations in a tabular format is called **Truth table**. Following are the truth table for all logical connectives:

For Negation:

P	¬ P
True	**False**
False	**True**

For Conjunction:

P	Q	P∧ Q
True	True	**True**
True	False	**False**
False	True	**False**
False	False	**False**

For disjunction:

P	Q	P V Q.
True	True	**True**
False	True	**True**
True	False	**True**
False	False	**False**

For Implication:

P	Q	P→ Q
True	True	**True**
True	False	**False**
False	True	**True**
False	False	**True**

For Biconditional:

P	Q	P⇔ Q
True	True	**True**
True	False	**False**
False	True	**False**
False	False	**True**

Truth table with three propositions:

We can build a proposition composing three propositions P, Q, and R. This truth table is made-up of 8n Tuples as we have taken three proposition symbols.

P	Q	R	¬R	Pv Q	PvQ→¬R
True	True	True	False	True	False
True	True	False	True	True	True
True	False	True	False	True	False
True	False	False	True	True	True
False	True	True	False	True	False
False	True	False	True	True	True
False	False	True	False	False	True
False	False	False	True	False	True

Precedence of connectives:

Just like arithmetic operators, there is a precedence order for propositional connectors or logical operators. This order should be followed while evaluating a propositional problem. Following is the list of the precedence order for operators:

Precedence	**Operators**
First Precedence	Parenthesis
Second Precedence	Negation
Third Precedence	Conjunction(AND)
Fourth Precedence	Disjunction(OR)
Fifth Precedence	Implication
Six Precedence	Biconditional

Note: For better understanding use parentheses to make sure of the correct interpretations. Such as $\neg R \vee Q$, It can be interpreted as $(\neg R) \vee Q$.

Logical equivalence:

Logical equivalence is one of the features of propositional logic. Two propositions are said to be logically equivalent if and only if the columns in the truth table are identical to each other.

Let's take two propositions A and B, so for logical equivalence, we can write it as A⇔B. In below truth table we can see that column for ¬A∨ B and A→B, are identical hence A is Equivalent to B

A	B	¬A	¬A∨ B	A→B
T	T	F	**T**	**T**
T	F	F	**F**	**F**
F	T	T	**T**	**T**
F	F	T	**T**	**T**

Properties of Operators:

- **Commutativity:**
  - P∧ Q= Q ∧ P, or
  - P ∨ Q = Q ∨ P.
- **Associativity:**
  - (P ∧ Q) ∧ R= P ∧ (Q ∧ R),
  - (P ∨ Q) ∨ R= P ∨ (Q ∨ R)
- **Identity element:**
  - P ∧ True = P,
  - P ∨ True= True.
- **Distributive:**
  - P∧ (Q ∨ R) = (P ∧ Q) ∨ (P ∧ R).
  - P ∨ (Q ∧ R) = (P ∨ Q) ∧ (P ∨ R).
- **DE Morgan's Law:**

- $\neg (P \land Q) = (\neg P) \lor (\neg Q)$
    - $\neg (P \lor Q) = (\neg P) \land (\neg Q)$.
- **Double-negation elimination:**
    - $\neg (\neg P) = P$.

Limitations of Propositional logic:

- We cannot represent relations like ALL, some, or none with propositional logic. Example:
    a. **All the girls are intelligent.**
    b. **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.

Rules of Inference in Artificial intelligence

Inference:

In artificial intelligence, we need intelligent computers which can create new logic from old logic or by evidence, **so generating the conclusions from evidence and facts is termed as Inference**.

Inference rules:

Inference rules are the templates for generating valid arguments. Inference rules are applied to derive proofs in artificial intelligence, and the proof is a sequence of the conclusion that leads to the desired goal.

In inference rules, the implication among all the connectives plays an important role. Following are some terminologies related to inference rules:

- **Implication:** It is one of the logical connectives which can be represented as P → Q. It is a Boolean expression.
- **Converse:** The converse of implication, which means the right-hand side proposition goes to the left-hand side and vice-versa. It can be written as Q → P.
- **Contrapositive:** The negation of converse is termed as contrapositive, and it can be represented as ¬ Q → ¬ P.
- **Inverse:** The negation of implication is called inverse. It can be represented as ¬ P → ¬ Q.

From the above term some of the compound statements are equivalent to each other, which we can prove using truth table:

P	Q	P → Q	Q→ P	¬ Q → ¬ P	¬ P → ¬ Q.
T	T	T	T	T	T
T	F	F	T	F	T
F	T	T	F	T	F
F	F	T	T	T	T

Hence from the above truth table, we can prove that P → Q is equivalent to ¬ Q → ¬ P, and Q→ P is equivalent to ¬ P → ¬ Q.

Types of Inference rules:

1. Modus Ponens:

The Modus Ponens rule is one of the most important rules of inference, and it states that if P and P → Q is true, then we can infer that Q will be true. It can be represented as:

Notation for Modus ponens: $\frac{P \rightarrow Q, \quad P}{\therefore Q}$

Example:

Statement-1: "If I am sleepy then I go to bed" ==> P→ Q
Statement-2: "I am sleepy" ==> P
Conclusion: "I go to bed." ==> Q.
Hence, we can say that, if P→ Q is true and P is true then Q will be true.

Proof by Truth table:

P	Q	$P \rightarrow Q$
0	0	0
0	1	1
1	0	0
1	1	1

2. Modus Tollens:

The Modus Tollens rule states that if P→ Q is true and ¬ **Q is true, then ¬ P** will also true. It can be represented as:

Notation for Modus Tollens: $\dfrac{P \rightarrow Q,\ \sim Q}{\sim P}$

Statement-1: "If I am sleepy then I go to bed" ==> P→ Q
Statement-2: "I do not go to the bed."==> ~Q
Statement-3: Which infers that "**I am not sleepy**" => ~P

Proof by Truth table:

P	Q	$\sim P$	$\sim Q$	$P \rightarrow Q$
0	0	1	1	1
0	1	1	0	1
1	0	0	1	0
1	1	0	0	1

3. Hypothetical Syllogism:

The Hypothetical Syllogism rule states that if P→R is true whenever P→Q is true, and Q→R is true. It can be represented as the following notation:

Example:

Statement-1: If you have my home key then you can unlock my home. **P→Q**
Statement-2: If you can unlock my home then you can take my money. **Q→R**
Conclusion: If you have my home key then you can take my money. **P→R**

Proof by truth table:

P	Q	R	$P \to Q$	$Q \to R$	$P \to R$
0	0	0	1	1	1 ←
0	0	1	1	1	1 ←
0	1	0	1	0	1
0	1	1	1	1	1 ←
1	0	0	0	1	1
1	0	1	0	1	1
1	1	0	1	0	0
1	1	1	1	1	1 ←

4. Disjunctive Syllogism:

The Disjunctive syllogism rule state that if P∨Q is true, and ¬P is true, then Q will be true. It can be represented as:

Notation of Disjunctive syllogism: $\frac{P \lor Q,\ \neg P}{Q}$

Example:

Statement-1: Today is Sunday or Monday. ==>P∨Q
Statement-2: Today is not Sunday. ==> ¬P
Conclusion: Today is Monday. ==> Q

Proof by truth-table:

P	Q	$\neg P$	$P \lor Q$
0	0	1	0
0	1	1	1 ←
1	0	0	1
1	1	0	1

5. Addition:

The Addition rule is one the common inference rule, and it states that If P is true, then P∨Q will be true.

Notation of Addition: $\frac{P}{P \vee Q}$

Example:

Statement: I have a vanilla ice-cream. ==> P
Statement-2: I have Chocolate ice-cream.
Conclusion: I have vanilla or chocolate ice-cream. ==> (P∨Q)

Proof by Truth-Table:

P	Q	$P \vee Q$
0	0	0
1	0	1
0	1	1
1	1	1

6. Simplification:

The simplification rule state that if **P∧ Q** is true, then **Q or P** will also be true. It can be represented as:

Notation of Simplification rule: $\frac{P \wedge Q}{Q}$ Or $\frac{P \wedge Q}{P}$

Proof by Truth-Table:

P	Q	$P \wedge Q$
0	0	0
1	0	0
0	1	0
1	1	1

7. Resolution:

The Resolution rule state that if P∨Q and ¬ P∧R is true, then Q∨R will also be true. **It can be represented as**

Notation of Resolution $\dfrac{P \vee Q,\ \neg P \wedge R}{Q \vee R}$

Proof by Truth-Table:

P	¬ P	Q	R	$P \vee Q$	¬ P∧R	$Q \vee R$
0	1	0	0	0	0	0
0	1	0	1	0	0	1
0	1	1	0	1	1	1
0	1	1	1	1	1	1
1	0	0	0	1	0	0
1	0	0	1	1	0	1
1	0	1	0	1	0	1
1	0	1	1	1	0	1

Rule of Inference	Tautology	Name
p $p \rightarrow q$ $\therefore q$	$(p \wedge (p \rightarrow q)) \rightarrow q$	Modus Ponens
$\neg q$ $p \rightarrow q$ $\therefore \neg p$	$(\neg q \wedge (p \rightarrow q)) \rightarrow \neg p$	Modus Tollens
$p \rightarrow q$ $q \rightarrow r$ $\therefore p \rightarrow r$	$((p \rightarrow q) \wedge (q \rightarrow r)) \rightarrow (p \rightarrow r)$	Hypothetical syllogism
$\neg p$ $p \vee q$ $\therefore q$	$(\neg p \wedge (p \vee q)) \rightarrow q$	Disjunctive Syllogism
p $\therefore (p \vee q)$	$p \rightarrow (p \vee q)$	Addition
$(p \wedge q) \rightarrow r$ $\therefore p \rightarrow (q \rightarrow r)$	$((p \wedge q) \rightarrow r) \rightarrow (p \rightarrow (q \rightarrow r))$	Exportation
$p \vee q$ $\neg p \vee r$ $\therefore q \vee r$	$((p \vee q) \wedge (\neg p \vee r)) \rightarrow q \vee r$	Resolution

Q.1) State which rule of inference is the basis of the following argument:

"It is below freezing now. Therefore, it is either below freezing or raining now."

<u>Solution:</u>

Let, p :"*It is below freezing now*"

q :"*It is raining now.*"

Then this argument is of the form:

$$\frac{p}{\therefore p \lor q}$$

This is an argument that uses the **addition rule.**

Q.2) State which rule of inference is the basis of the following argument:

"It is below freezing and raining now. Therefore, it is below freezing now."

<u>Solution:</u>

let, p:"*It is below freezing now*"

q :"*It is raining now*"

This argument is of the form:

$$\frac{p \land q}{\therefore p}$$

This argument uses the **simplification rule.**

Q.3) State which rule of inference is the basis of the following argument:

"If you have a current password, then you can log onto the network. You have a current password. Therefore, You can log onto the network."

<u>Solution:</u>

Let, p :"*you have a current password*"

q :"*you can log onto the network*"

Then this argument is of the form:

$$\begin{array}{l} p \rightarrow q \\ \underline{p \qquad} \\ \therefore q \end{array}$$

This is an argument that uses the **Modus Ponens rule.**

Q.4) State which rule of inference is the basis of the following argument:

"If it rains today, then we will not have a barbecue today. If we do not have a barbecue today, then we will have a barbecue tomorrow. Therefore, if it rains today, then we will have a barbecue tomorrow"

<u>Solution:</u>

let, p:"*it rains today*"

q :"*We will have a barbecue today*"

r :"*we will have a barbecue tomorrow*"

This argument is of the form:

$$\begin{array}{l} p \rightarrow \neg q \\ \underline{\neg q \rightarrow r \quad} \\ \therefore p \rightarrow r \end{array}$$

This argument uses the **Hypothetical rule.**

Q.5) Show that the premises: "*If I play football then I am tired the next day*", "*I will take rest if I am tired*", "*I did not take rest*" will lead to the conclusion "*I did not play football*".

Solution:

Let, p:"*If I play football*"

q:"*I am tired*"

r:"*I will take rest* "

Hypothesis: i) $p \to q$

ii) $q \to r$

iii) $\neg r$

Conclusion: $\therefore \neg p$

s.n.	STEPS	REASONS
1.	$p \to q$	Given Hypothesis
2.	$q \to r$	Given Hypothesis
3.	$p \to r$	HYPOTHTICAL SYLLOGISM IN 1 & 2
4.	$\neg r$	Given Hypothesis
5.	$\neg p$	MODUS TOLLENSON 3 & 4

Q.6) Show that the premises "*It is not sunny this afternoon and it is colder than yesterday*". "*We will go swimming only if it is sunny.*" "*If we do not go swimming, then we will take a canoe trip.*" and "*If we take a canoe trip, then we will be home by sunset*" lead to the conclusion "*We will be home by sunset.*"

Solution:

Let, p:"*It is sunny this afternoon*"

q:"*It is colder than yesterday*"

r:"*We will go swimming*"

s:"*We will take canoe trip*"

t:"*We will be home by sunset*"

Hypothesis: i) $\neg p \wedge q$

ii) $r \to p$

iii) $\neg r \to s$

iv) $s \to t$

Conclusion: $\therefore t$

s.n.	STEPS	REASONS
1.	$\neg p \wedge q$	Given Hypothesis
2.	$\neg p$	SIMPLIFICATION ON 1
3.	$r \to p$	Given Hypothesis
4.	$\neg r$	MODUS TOLLENS ON 2 & 3
5.	$\neg r \to s$	Given Hypothesis
6.	s	MODUS PONENS ON 4 & 5
7.	$s \to t$	Given Hypothesis
8.	t	MODUS PONENS ON 6 & 7

Q.7) Show that the premises *"If you send me an e-mail message, then I will finish writing the program", "If you do not send me an e-mail message, then I will go to sleep early," "If I go to sleep early, then I will wake up feeling refreshed"* lead to the conclusion *"If I do not finish writing the program, then I will wake up feeling refreshed."*

Solution:

Let, p: *"you send me an e-mail message"*
q: *"I will finish writing the program"*
r: *"I will go to sleep early"*
s: *"I will wake up feeling refreshed"*

Hypothesis: i) $p \rightarrow q$
ii) $\neg p \rightarrow r$
iii) $r \rightarrow s$

Conclusion: $\therefore \neg q \rightarrow s$

s.n.	STEPS	REASONS
1.	$p \rightarrow q$	Given Hypothesis
2.	$\neg q \rightarrow \neg p$	CONTRAPOSITIVE ON 1
3.	$\neg p \rightarrow r$	Given Hypothesis
4.	$\neg q \rightarrow r$	Hypothetical syllogism using (2) and (3)
5.	$r \rightarrow s$	Given Hypothesis
6.	$\neg q \rightarrow s$	Hypothetical syllogism using (4) and (5)

Q.8) Show that the premises *"If it does not rain or if it is not foggy, then the sailing race will be held and the lifesaving demonstration will go on," "If the sailing race is held, then the trophy will be awarded,"* and *"The trophy was not awarded"* imply the conclusion *"It rained."*

Solution:

Let, p: *"It rains"*
q: *"It is foggy"*
r: *"The sailing race is held"*
s: *"Life saving demonstration is done"*
t: *"Trophy is awarded"*

Hypothesis: i) $(\neg p \vee \neg q) \rightarrow (r \wedge s)$
ii) $r \rightarrow t$
iii) $\neg t$

Conclusion: $\therefore p$

s.n.	STEPS	REASONS
1.	$r \rightarrow t$	Given Hypothesis
2.	$\neg t$	Given Hypothesis
3.	$\neg r$	MODUS TOLLENS ON 1 & 2
4.	$(\neg p \vee \neg q) \rightarrow (r \wedge s)$	Given Hypothesis
5.	$\neg r \vee \neg s$	Addition on 3
6.	$\neg(r \wedge s)$	DE-MORGAN'S LAW on 5
7	$\neg(\neg p \vee \neg q)$	MODUS TOLLENS ON 4 and 6
8	$p \wedge q$	DE-MORGAN'S LAW on 7
9	p	SIMPLIFICATION on 8

Q.9) Show that the premises *"If the interest rate drops , the housing market will improve", "The federal discount rate will drop or the housing market will not improve", "Interest rate will drop"* imply the conclusion *"The federal discount rate will drop"*

Solution:

Let, p: *"the interest rate drops "*
q: *"the housing market will improve"*
r: *"The federal discount rate will drop"*

Hypothesis: i) $p \rightarrow q$
ii) $r \vee \neg q$
iii) p
Conclusion: $\therefore r$

STEPS	REASONS
1. $p \rightarrow q$	Given Hypothesis
2. $\neg p \vee q$	Implication on 1
3. $r \vee \neg q$	Given Hypothesis
4. $\neg p \vee r$	Resolution From 2 and 3
5. p	Given hypothesis
6. r	From 4 and 5

Q.10) Show that the premises *"If my cheque book is in office, then I have paid my phone bill", "I was looking for phone bill at breakfast or I was looking for phone bill in my office", "If was looking for phone bill at breakfast then my cheque book is on breakfast table" , "If I was looking for phone bill in my office then my cheque book is in my office", "I have not paid my phone bill"* imply the conclusion *"My cheque book is on my breakfast table"*

Solution:

Let, p: *"my cheque book is in office"*
q: *"I have paid my phone bill"*
r: *"I was looking for phone bill at breakfast "*
s: *"I was looking for phone bill in my office"*
t: *"my cheque book is on breakfast table"*

Hypothesis: i) $p \rightarrow q$
ii) $r \vee s$
iii) $r \rightarrow t$
iv) $s \rightarrow p$
v) $\neg q$
Conclusion: $\therefore t$

Hypothesis: i) $p \rightarrow q$
ii) $r \vee s$
iii) $r \rightarrow t$
iv) $s \rightarrow p$
v) $\neg q$
Conclusion: $\therefore t$

STEPS	REASONS
1. $p \rightarrow q$	Given Hypothesis
2. $\neg q$	Given Hypothesis
3. $\neg p$	Modus Tollens on 1 and 2
4. $r \vee s$	Given Hypothesis
5. $r \rightarrow t$	Given Hypothesis
6. $\neg r \vee t$	Implication on 5
7. $s \vee t$	Resolution from 4 and 6
8. $s \rightarrow p$	Given Hypothesis
9. $\neg s \vee p$	Implication on 8
10. $t \vee p$	Resolution from 7 and 9
11. t	From 3 and 10

PROOF BY RESOLUTION:

- Propositional Resolution works only on expressions in *clausal form*. Before the rule can be applied, the premises and conclusions must be converted to this form.

A *literal* is either an atomic sentence or a negation of an atomic sentence. For example, if p is a logical constant, the following sentences are both literals.

$$p$$
$$\neg p$$

A *clausal sentence* is either a literal or a disjunction of literals. If p and q are logical constants, then the following are clausal sentences.

$$p$$
$$\neg p$$
$$\neg p \vee q$$

A *clause* is the set of literals in a clausal sentence. For example, the following sets are the clauses corresponding to the clausal sentences above.

$$\{p\}$$
$$\{\neg p\}$$
$$\{\neg p, q\}$$

CONVERTING TO CAUSAL FORM:

- Examples:
  1. $p \rightarrow q \;\; = \neg p \vee q$
  2. $p \leftrightarrow q \; = (\neg p \vee q) \wedge (\neg q \vee p)$-------{CNF}
  3. $\neg(p \wedge q) = \neg p \vee \neg q$
  4. $\neg(p \vee q) = \neg p \wedge \neg q$

CNF: CNF (Conjunctive normal form) if it is a ∧(Conjunction) of ∨(Disjunction s) of literals (variables or their negation.)

DNF: DNF (Disjunctive normal form) if it is a ∨(Disjunction s) of ∧(Conjunction) of literals (variables or their negation.)

Example: 1. $\neg p \vee \neg q$

2. $(p \wedge q) \vee (p \wedge r)$

Prove:

i) P→Q

ii) ¬P→R

iii) R→S

∴ ¬Q→S

<u>Solution:</u>

The causal form of hypothesis and Conclusion are:

i) ¬P v Q

ii) P v R

iii) ¬R v S

∴ Q v S

STEPS	REASONS
1. ¬P v Q	1. Given Hypothesis
2. P v R	2. Given Hypothesis
3. Q v R =R v Q	3. Using Resolution
4. ¬R v S	4. Given Hypothesis
5. Q v S	4. Using Resolution

Using Graphical Method:

Prove:

i) P→Q

ii) ¬P→R

iii) R→S

∴ ¬Q→S

STEP 1. The causal form of hypothesis and Conclusion are:

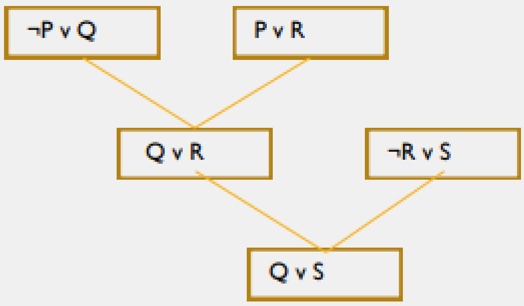
i) ¬P v Q

ii) P v R

iii) ¬R v S

∴ Q v S

STEP 2.



Using Graphical Method:

Prove:

i) P

ii) P→R

iii) R→S

∴ S

STEP 1. The causal form of hypothesis and Conclusion are:

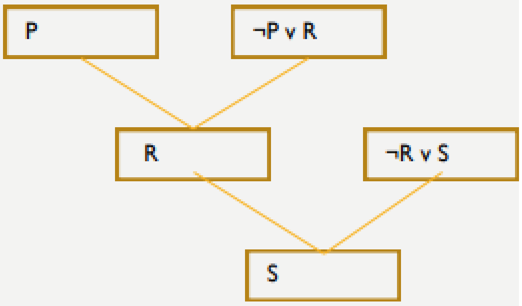
i) P

ii) ¬P v R

iii) ¬R v S

∴ S

STEP 2.



- Using resolution principle, prove that the hypotheses: "If today is Tuesday then I will have a test in Discreet Math or Microprocessor". "If my Microprocessor teacher is sick then I will not have a test in Microprocessor." "Today is Tuesday and my Microprocessor teacher is sick." lead to the conclusion that "I will have a test in Discrete Math"

Solution:

Let, p : "Today is Tuesday"

q : "I will have test in Discrete Math"

r : "I will have test in Microprocessor"

s : "My Microprocessor teacher is sick"

Hypothesis: i) $p \rightarrow (q \vee r)$

ii) $s \rightarrow \neg r$

iii) $p \wedge s$

Conclusion: q

The Causal Forms are:

Hypothesis:

i) $\neg p \vee (q \vee r)$

ii) $\neg s \vee \neg r$

iii) p

iv) s

Conclusion:

$\therefore$ q

The Causal Forms are:

Hypothesis:

i) $\neg p \vee (q \vee r)$

ii) $\neg s \vee \neg r$

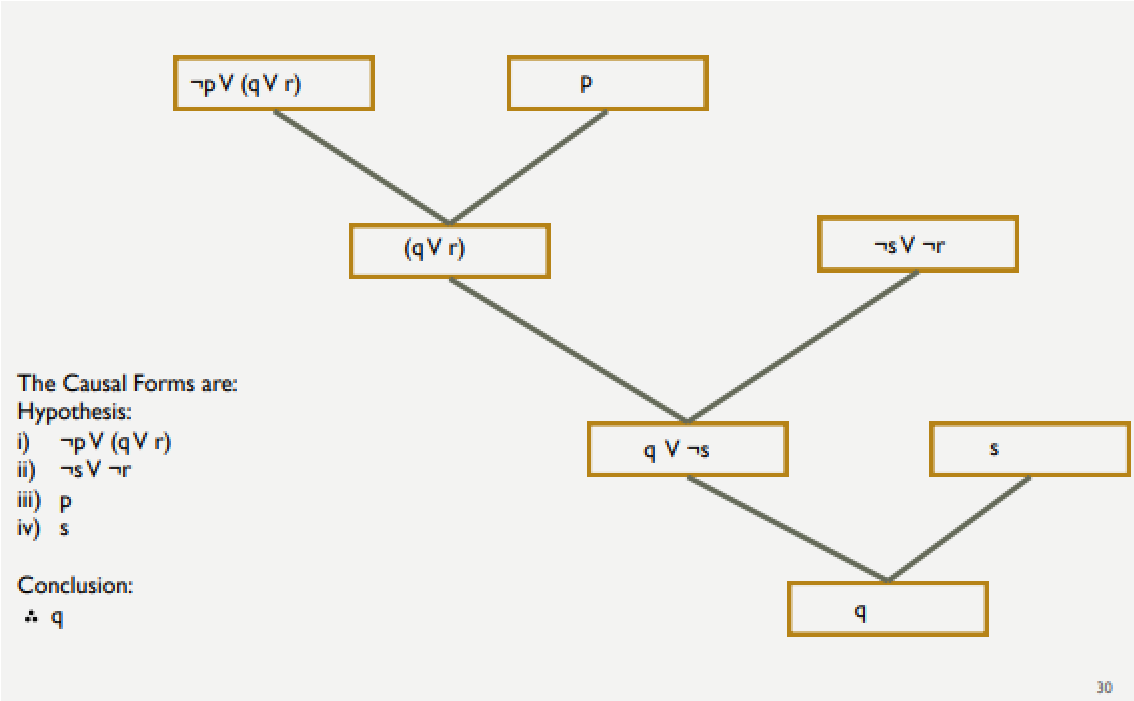
iii) p

iv) s

Conclusion:

$\therefore$ q

STEPS	REASONS
1. $\neg p \vee (q \vee r)$	Given Hypothesis
2. p	Given Hypothesis
3. $q \vee r$	From 1 and 2
4. $\neg s \vee \neg r$	Given Hypothesis
5. $q \vee \neg s$	From 3 and 4
6. s	Given Hypothesis
7. q	From 5 and 6



First-Order Logic in Artificial intelligence

In the topic of Propositional logic, we have seen how to represent statements using propositional logic. But unfortunately, in propositional logic, we can only represent the facts, which are either true or false. PL is not sufficient to represent complex sentences or natural language statements. The propositional logic has very limited expressive power. Consider the following sentence, which we cannot represent using PL logic.

- **"Some humans are intelligent", or**
- **"Sachin likes cricket."**

To represent the above statements, PL logic is not sufficient, so we require some more powerful logic, such as first-order logic.

First-Order logic:

- First-order logic is another way of knowledge representation in artificial intelligence. It is an extension to propositional logic.
- FOL is sufficiently expressive to represent the natural language statements in a concise way.

- First-order logic is also known as **Predicate logic or First-order predicate logic**. First-order logic is a powerful language that develops information about the objects in a more easy way and can also express the relationship between those objects.
- First-order logic (like natural language) does not only assume that the world contains facts like propositional logic but also assumes the following things in the world:
    a. **Objects:** A, B, people, numbers, colors, wars, theories, squares, pits, wumpus, ...
    b. **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
    c. **Function:** Father of, best friend, third inning of, end of, ...
- As a natural language, first-order logic also has two main parts:
    a. **Syntax**
    b. **Semantics**

Syntax of First-Order logic:

The syntax of FOL determines which collection of symbols is a logical expression in first-order logic. The basic syntactic elements of first-order logic are symbols. We write statements in short-hand notation in FOL.

Basic Elements of First-order logic:

Following are the basic elements of FOL syntax:

Constant	1, 2, A, John, Mumbai, cat,....
Variables	x, y, z, a, b,....
Predicates	Brother, Father, >,....
Function	sqrt, LeftLegOf, ...
Connectives	$\land$, $\lor$, $\neg$, $\Rightarrow$, $\Leftrightarrow$

Equality	==
Quantifier	∀, ∃

Atomic sentences:

- Atomic sentences are the most basic sentences of first-order logic. These 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: Ravi and Ajay are brothers: => Brothers(Ravi, Ajay).
Chinky is a cat: => cat (Chinky).

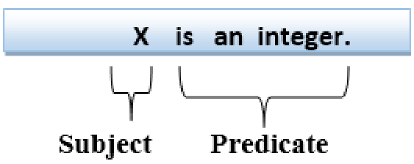
Complex Sentences:

- 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.



Quantifiers in First-order logic:

- A quantifier is a language element which generates quantification, and quantification specifies the quantity of specimen in the universe of discourse.
- These are the symbols that permit to determine or identify the range and scope of the variable in the logical expression. There are two types of quantifier:
  a. **Universal Quantifier, (for all, everyone, everything)**
  b. **Existential quantifier, (for some, at least one).**

Universal Quantifier:

Universal quantifier is a symbol of logical representation, which specifies that the statement within its range is true for everything or every instance of a particular thing.

The Universal quantifier is represented by a symbol $\forall$, which resembles an inverted A.

Note: In universal quantifiers we use the implication "→".

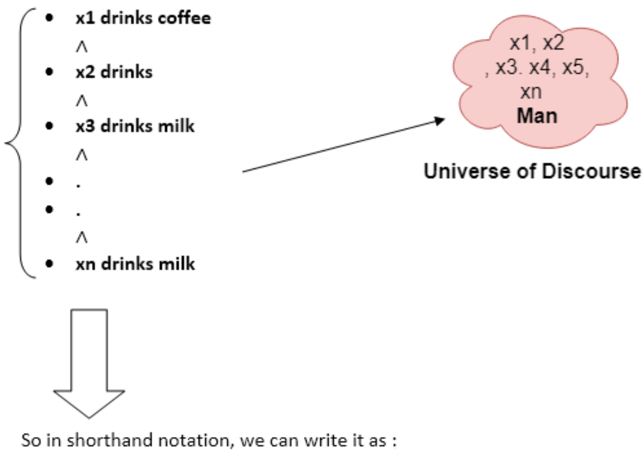
If x is a variable, then $\forall$x is read as:

- **For all x**
- **For each x**
- **For every x.**

Example:

All men drink coffee.

Let a variable x which refers to a cat so all x can be represented in UOD as below:



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

It will be read as: There are all x where x is a man who drinks coffee.

Existential Quantifier:

Existential quantifiers are the type of quantifiers, which express that the statement within its scope is true for at least one instance of something.

It is denoted by the logical operator $\exists$, which resembles as inverted E. When it is used with a predicate variable then it is called as an existential quantifier.

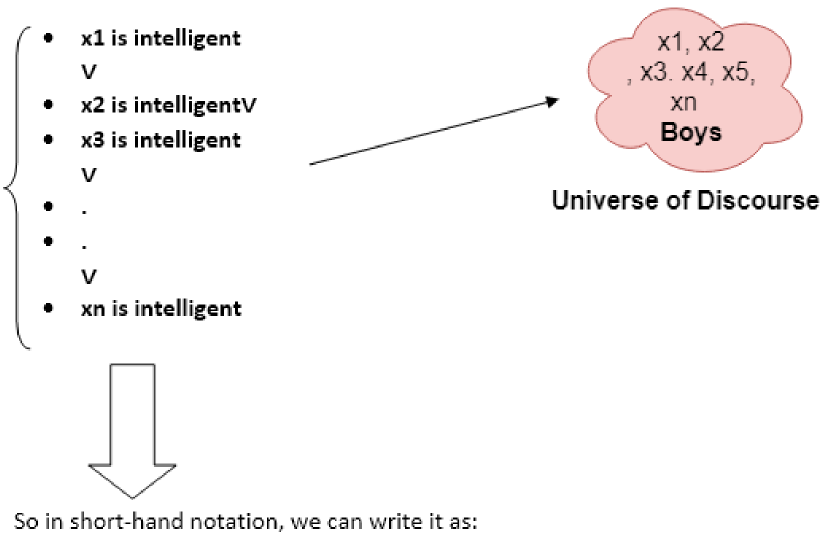
Note: In Existential quantifier we always use AND or Conjunction symbol ($\wedge$).

If x is a variable, then existential quantifier will be $\exists$x or $\exists$(x). And it will be read as:

- **There exists a 'x.'**
- **For some 'x.'**
- **For at least one 'x.'**

Example:

Some boys are intelligent.



$\exists$x: boys(x) $\wedge$ intelligent(x)

It will be read as: There are some x where x is a boy who is intelligent.

Points to remember:

- The main connective for universal quantifier $\forall$ is implication $\rightarrow$.
- The main connective for existential quantifier $\exists$ is and $\wedge$.

Properties of Quantifiers:

- 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 $\forall y \exists x$.

Some Examples of FOL using quantifier:

1. All birds fly.
In this question the predicate is "**fly(bird)**."
And since there are all birds who fly, it will be represented as follows.
$\forall$**x bird(x) $\rightarrow$fly(x)**.

2. Every man respects his parents.
In this question, the predicate is "**respect(x, y)," where x=man, and y= parent**.

Since there is every man so will use $\forall$, and it will be represented as follows:

$\forall$**x man(x) → respects (x, parent)**.

3. Some boys play cricket.

In this question, the predicate is "**play(x, y)**," where x= boys, and y= game. Since there are some boys so we will use $\exists$**, and it will be represented as**:

$\exists$**x boys(x) → play(x, cricket)**.

4. Not all students like both Mathematics and Science.

In this question, the predicate is "**like(x, y)," where x= student, and y= subject**.

Since there are not all students, so we will use $\forall$ **with negation, so** following representation for this:

¬$\forall$ (x) [student(x) → like(x, Mathematics) $\wedge$ like(x, Science)].

5. Only one student failed in Mathematics.

In this question, the predicate is "**failed(x, y)," where x= student, and y= subject**.

Since there is only one student who failed in Mathematics, so we will use following representation for this:

$\exists$(x) [student(x) → failed (x, Mathematics) $\wedge$ $\forall$ (y) [¬(x==y) $\wedge$ student(y) → ¬failed (x, Mathematics)].

Free and Bound Variables:

The quantifiers interact with variables which appear in a suitable way. There are two types of variables in First-order logic which are given below:

Free Variable: A variable is said to be a free variable in a formula if it occurs outside the scope of the quantifier.

Example: $\forall$x $\exists$(y)[P (x, y, z)], where z is a free variable.

Bound Variable: A variable is said to be a bound variable in a formula if it occurs within the scope of the quantifier.

Example: $\forall$x [A (x) B(y)], here x and y are the bound variables.

Convert the Following Sentences into FOPL.

1. All students are smart.

   $\forall$ x (Student(x) $\Rightarrow$ Smart(x))

2. There exists a student.

$\exists$ x Student(x).

3. There exists a smart student.

   $\exists$ x (Student(x) $\wedge$ Smart(x))

4. Every student loves some student.

   $\forall$ x (Student(x) $\Rightarrow$ $\exists$ y (Student(y) $\wedge$ Loves(x,y)))

5. Every student loves some other student.

   $\forall$ x (Student(x) $\Rightarrow$ $\exists$ y (Student(y) $\wedge$ $\neg$ (x = y) $\wedge$ Loves(x,y)))

6. There is a student who is loved by every other student.

   $\exists$ x (Student(x) $\wedge$ $\forall$ y (Student(y) $\wedge$ $\neg$(x = y) $\Rightarrow$ Loves(y,x)))

7. Bill is a student.

   Student(Bill)

8. Bill takes either Analysis or Geometry (but not both)

   Takes(Bill, Analysis) $\Leftrightarrow$ $\neg$ Takes(Bill, Geometry)

9. Bill takes Analysis or Geometry (or both).

   Takes(Bill, Analysis) $\vee$ Takes(Bill, Geometry)

10. Bill takes Analysis and Geometry.

    Takes(Bill, Analysis) $\wedge$ Takes(Bill, Geometry)

11. Bill does not take Analysis.

    $\neg$ Takes(Bill, Analysis).

12. No student loves Bill.

    $\neg$ $\exists$ x (Student(x) $\wedge$ Loves(x, Bill))

13. Bill has at least one sister.

    $\exists$ x SisterOf(x,Bill)

14. Bill has no sister.

    $\neg$ $\exists$ x SisterOf(x,Bill)

15. Bill has at most one sister.

$\forall$ x, y (SisterOf(x, Bill) $\land$ SisterOf(y, Bill) $\Rightarrow$ x = y)

16. Bill has exactly one sister.

    $\exists$ x (SisterOf(x, Bill) $\land$ $\forall$ y (SisterOf(y, Bill) $\Rightarrow$ x = y))

17. Bill has at least two sisters.

    $\exists$ x, y (SisterOf(x, Bill) $\land$ SisterOf(y, Bill) $\land$ $\neg$ (x = y))

18. Every student takes at least one course.

    $\forall$ x (Student(x) $\Rightarrow$ $\exists$ y (Course(y) $\land$ Takes(x,y)))

19. Only one student failed History.

    $\exists$ x (Student(x) $\land$ Failed(x, History) $\land$ $\forall$ y (Student(y) $\land$ Failed(y, History) $\Rightarrow$ x = y))

20. No student failed Chemistry but at least one student failed History.

    $\neg$ $\exists$ x (Student(x) $\land$ Failed(x, Chemistry)) $\land$ $\exists$ x (Student(x) $\land$ Failed(x, History))

21. Every student who takes Analysis also takes Geometry.

    $\forall$ x (Student(x) $\land$ Takes(x, Analysis) $\Rightarrow$ Takes(x, Geometry))

22. No student can fool all the other students.

    $\neg$ $\exists$ x (Student(x) $\land$ $\forall$ y (Student(y) $\land$ $\neg$ (x = y) $\Rightarrow$ Fools(x,y)))

Using resolution in below data:

1. John likes all kind of foods.
2. Apple and vegetable are foods.
3. Anything anyone eats and not killed is food.
4. Anil eats peanuts and still alive.
5. Harry eats everything that anil eats.

Prove by resolution that: John likes peanuts

Steps 1 : Converting into FOPL

a) $\forall x$: food(x) → likes(John, x)

b) food(Apple) ∧ food(vegetable)

c) $\forall x \forall y$: eats (x,y) ∧¬killed(x) → food(y)

d) eats(anil, peanuts) ∧ alive(anil)

e) $\forall x$: eats(anil, x) →eats(Harry, x)

Added predicates :

¬killed(x) → alive(x)

alive(x) →¬killed(x)

Conclusion : likes(John, peanuts)

Steps 2(i) : Changing FOPL into conjunctive normal form (CNF)

a) x :¬food(x) V likes(John, x)

b) food(Apple) ∧ food(vegetable)

c) $\forall x \forall y$:¬[eats (x,y) ∧¬killed(x)] → food(y)

$\forall x \forall y$:¬eats (x,y) V killed(x) V food(y)

d) eats(anil, peanuts) ∧ alive(anil)

e) $\forall x$: eats(anil, x) V eats(Harry, x)

Added predicates :

killed(x) → alive(x)

¬alive(x) →¬killed(x)

Conclusion : likes(John, peanuts)

Steps 2(ii) :

Rename the variable at its standard form and

Remove ∃x quantifier and also drop ∀x quantifier

a) ¬food(x) V likes(John, x)

b) food(apple)

   food(vegetable)

c) ¬t

d) eats(anil, peanuts)

   alive(anil)

e) ¬eats(anil, a) V eats(Harry, a)

   Added predicates :

   killed(b) V alive(b)

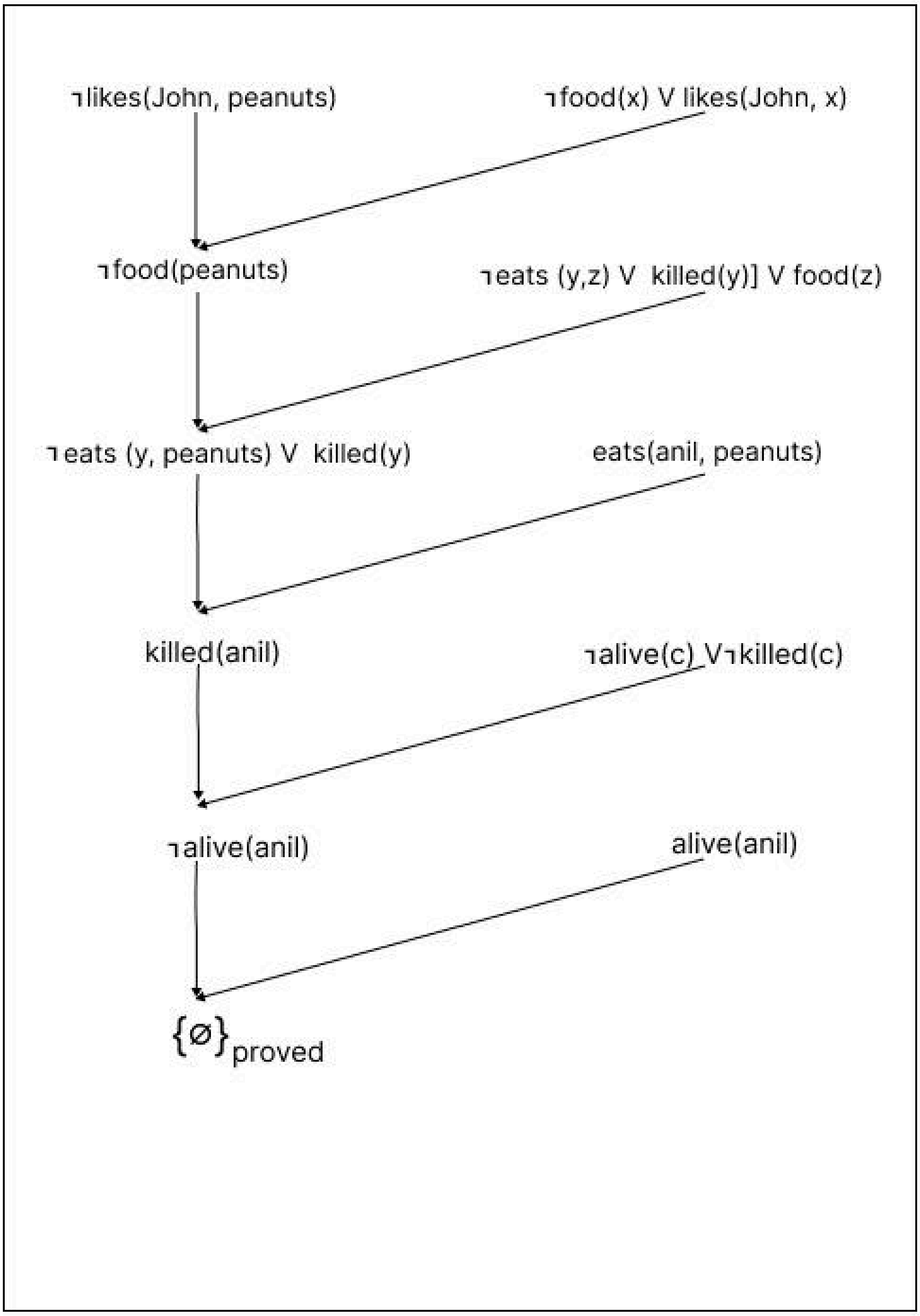
   ¬alive(c) V¬killed(c)

   Conclusion : likes(John, peanuts)

Steps 3 : Negate the conclusion

   Conclusion :¬likes(John, peanuts)

Steps 4: Resolution graph



Using resolution in below data:

1. If it is a sunny and warm day you will enjoy it.
2. If it is raining you will get wet.
3. It is a warm day.
4. It is a rainy day.
5. It is sunny.

Prove by resolution that: you will enjoy it

Steps 1 : Converting into FOPL

a) sunny Λ warm → enjoy

b) raining → wet

c) warm

d) rainy

e) sunny

Conclusion : enjoy(you)

Steps 2(i) : Changing FOPL into conjunctive normal form (CNF)

a) ┐sunny V ┐warm V enjoy

b) ┐raining Λ wet

c) warm

d) rainy

e) sunny

Conclusion : enjoy(you)

Steps 2(ii) :

Rename the variable at its standard form and

Remove ∃x quantifier and also drop ∀x quantifier

a) ¬sunny V ¬warm V enjoy

b) ¬raining

wet

c) warm

d) rainy

e) sunny

Conclusion : enjoy(you)

Steps 3 : Negate the conclusion

Conclusion :¬enjoy(you)

Steps 4: Resolution graph