

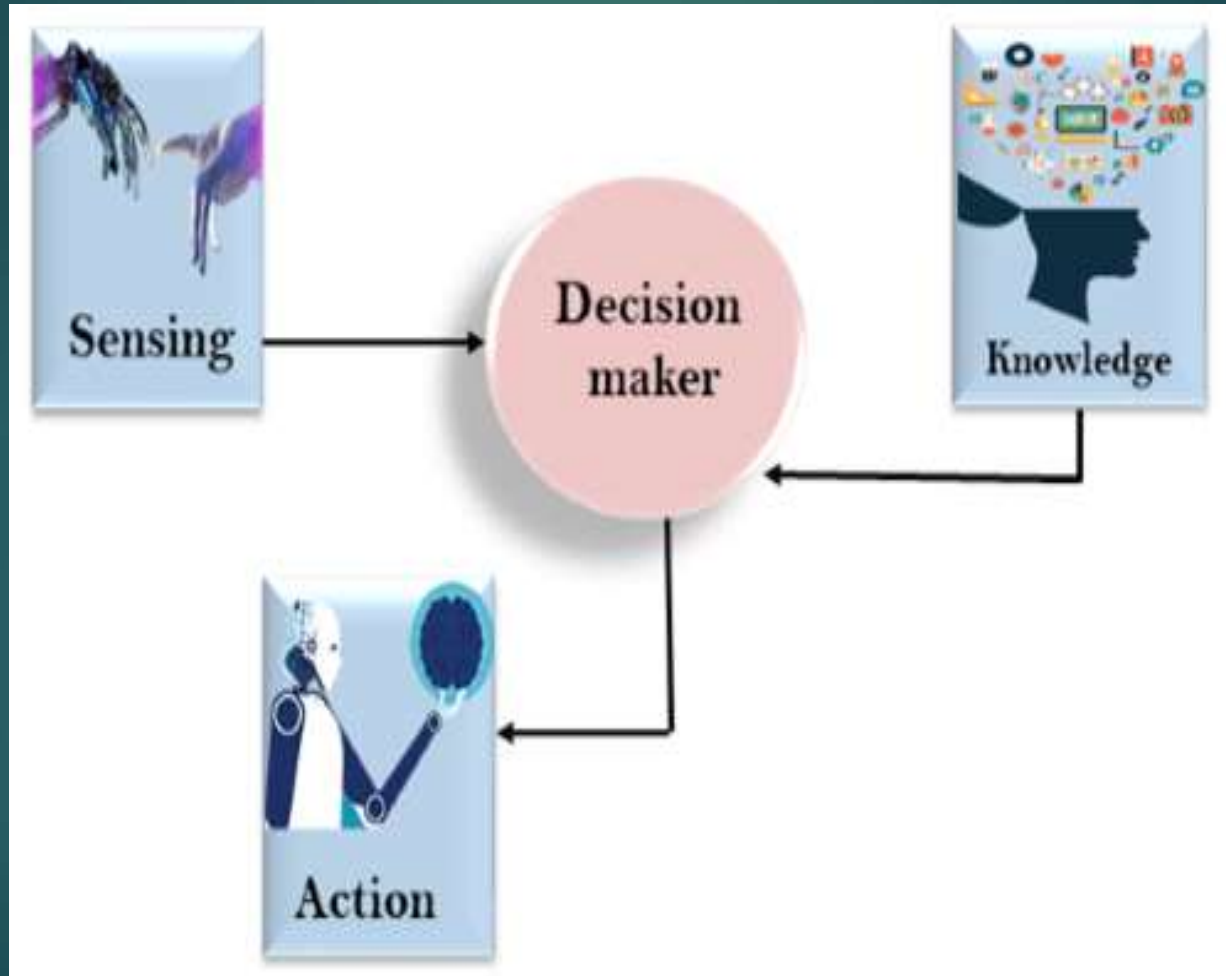
Unit-2

KNOWLEDGE REPRESENTATION

What is knowledge representation?

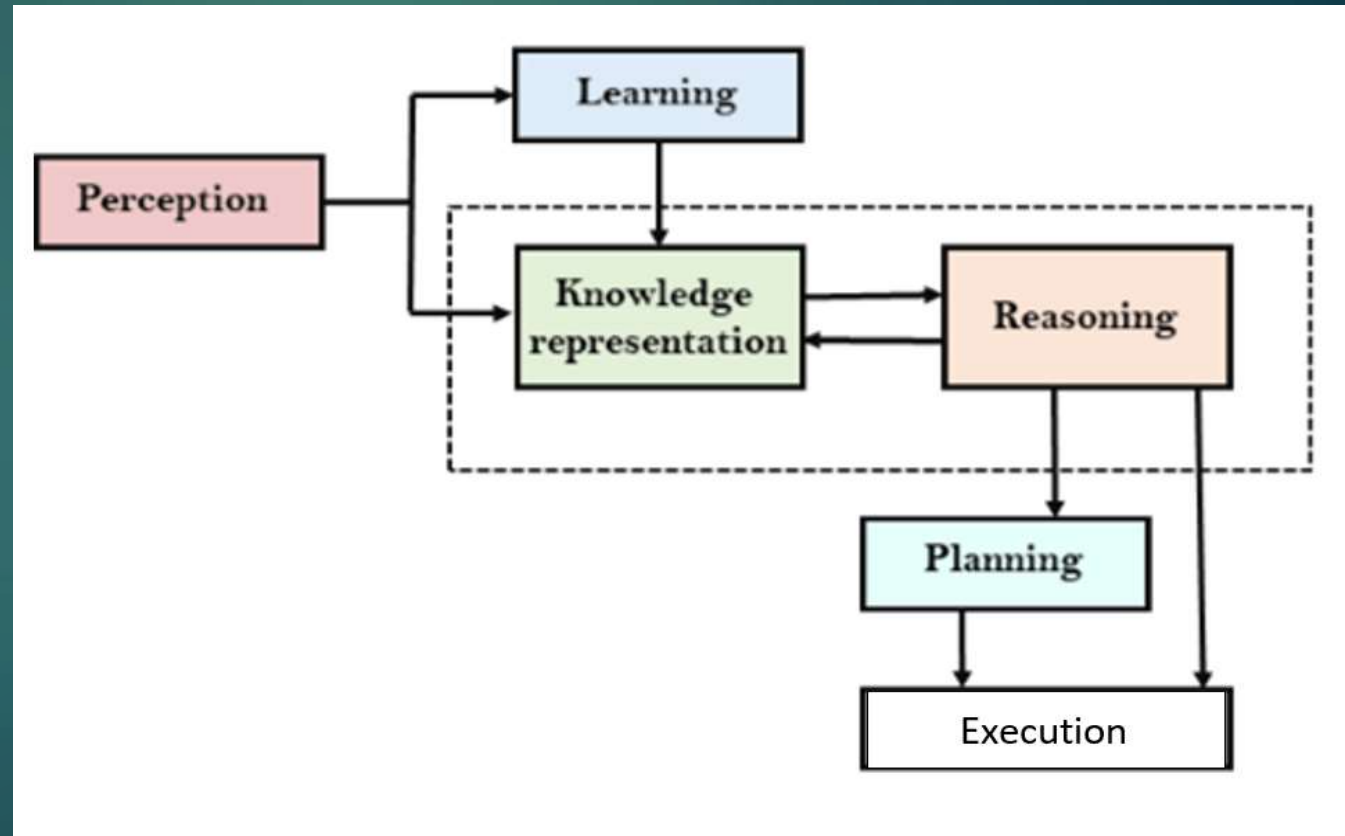
- ❑ Humans are best at understanding, reasoning, and interpreting knowledge.
- ❑ **But how machines do all these things comes under knowledge representation and reasoning (KRR).**
- ❑ We can describe Knowledge representation as following:
 - concerned with AI agents thinking and how thinking contributes to intelligent behavior of agents.
 - representing information about the real world.
 - Not just storing data into some database, but it also enables an intelligent machine to learn from that knowledge and experiences

The relation between knowledge and intelligence



AI knowledge cycle

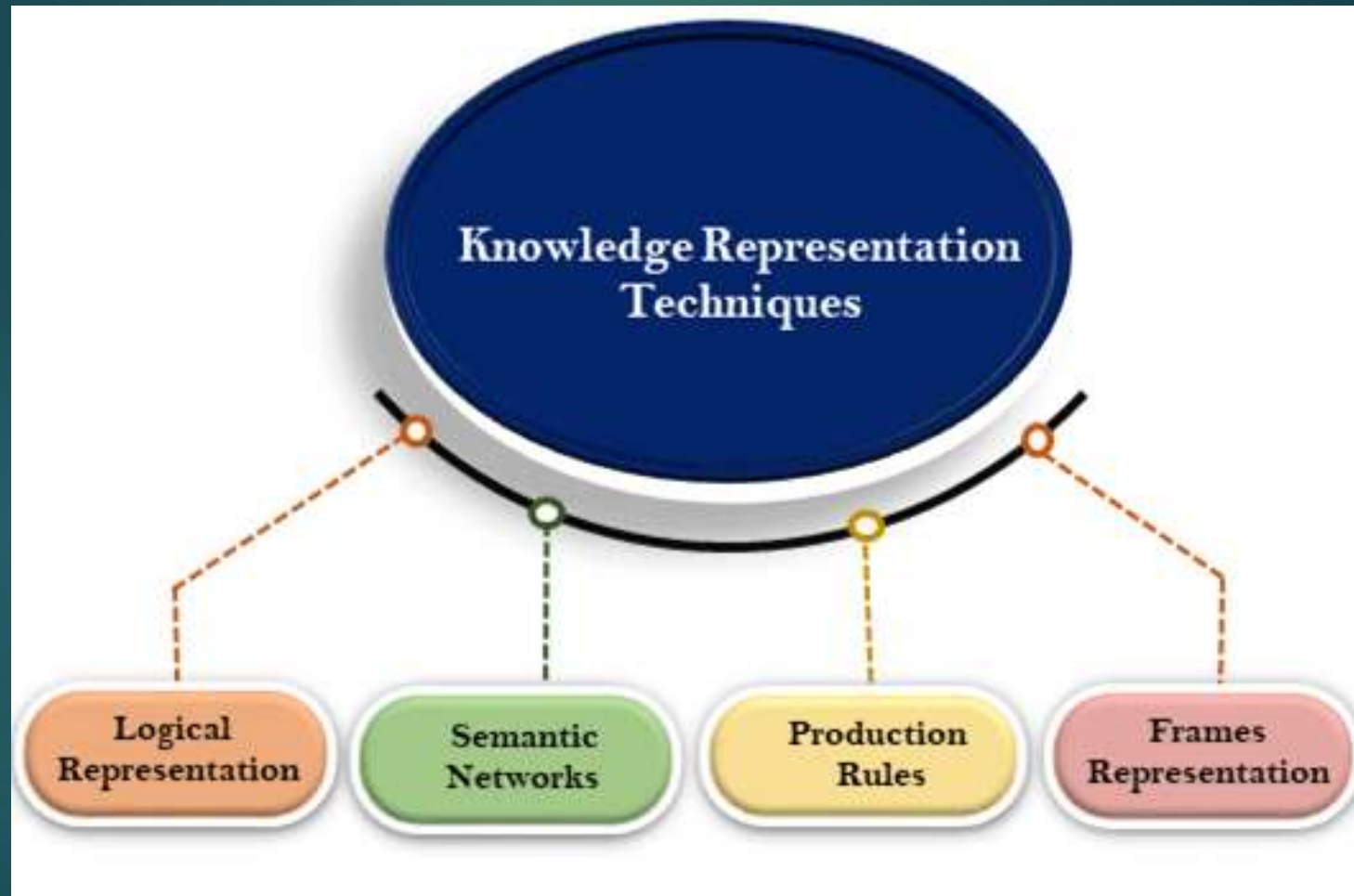
- Perception
- Learning
- Knowledge Representation and Reasoning
- Planning
- Execution





Techniques of knowledge representation

What is knowledge representation?



Logical Representation

- ❑ Logical representation is a language with some concrete rules which deals with propositions.
- ❑ Logical representation means drawing a conclusion based on various conditions.
- ❑ It consists of precisely defined syntax and semantics which supports the sound inference.
- ❑ Syntax:
 - Syntaxes are the rules which decide how we can construct legal sentences in the logic.
- ❑ Semantics:
 - Semantics are the rules by which we can interpret the sentence in the logic.

Logical Representation

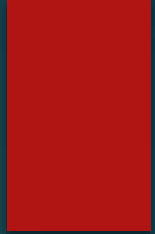


- Logical representation can be categorized into mainly two logics:
 - Propositional Logics
 - Predicate logics
- Advantages of logical representation:
 - Enables us to do logical reasoning.
 - basis for the programming languages.
- Disadvantages of logical Representation:
 - Have some restrictions and are challenging to work with.
 - may not be very natural, and inference may not be so efficient.



Propositional Logic

Propositional logic (PL)

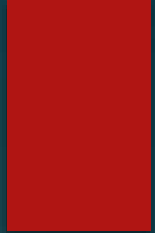


- PL is the simplest form of logic where all the statements are made by propositions.
- A proposition is a declarative statement which is either true or false.
- Example:
 - It is Sunday.
 - The Sun rises from West (False proposition)
 - $3+3=7$ (False proposition)
 - 5 is a prime number.

Basic Facts about PL

- 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 for representing a proposition, such **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**.
- 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.
- ❑ **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.
 - ❖ **Example:**
 - $2+2$ 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.
 - ❖ **Example:**
 - "It is raining today, and street is wet."
 - "Ankit is a doctor, and his clinic is in Mumbai."

Logical Connectives:



- Logical connectives are used to connect two simpler propositions or representing a sentence logically. We can create compound propositions with the help of logical connectives.

- There are mainly five connectives, which are given as follows:
 - Negation.
 - **Conjunction.**
 - **Disjunction.**
 - **Implication.**
 - **Biconditional**

Logical Connectives:

- ❑ **Negation:** A sentence such as $\neg P$ is called **negation of P**. A literal can be either Positive literal or negative literal.
- ❑ **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, $P = \text{Rohan is intelligent}$, $Q = \text{Rohan is hardworking}$. $\rightarrow P \wedge Q$.
- ❑ **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 $P = \text{Ritika is Doctor}$. $Q = \text{Ritika is Engineer}$, so we can write it as $P \vee Q$.
- ❑ **Implication:** A sentence such as $P \rightarrow Q$, is called an implication. Implications are also known as **if-then rules**.
 - **Example:** If it is raining, then the street is wet.
 - Let $P = \text{It is raining}$, and $Q = \text{Street is wet}$, so it is represented as $P \rightarrow Q$
- ❑ **Biconditional:** A sentence such as $P \Leftrightarrow Q$ is a **Biconditional sentence**.
 - **Example:** If I am breathing, then I am alive.
 - $P = \text{I am breathing}$, $Q = \text{I am alive}$, it can be represented as $P \Leftrightarrow Q$.

Summary of Propositional Logic Connectives:

Connective symbols	Word	Technical term	Example
\wedge	AND	Conjunction	$A \wedge B$
\vee	OR	Disjunction	$A \vee B$
\rightarrow	Implies	Implication	$A \rightarrow B$
\Leftrightarrow	If and only if	Biconditional	$A \Leftrightarrow B$
\neg or \sim	Not	Negation	$\neg A$ or $\neg B$

Truth Tables: Truth values of propositions in all possible scenarios

For Negation:

P	$\neg P$
True	False
False	True

For Conjunction:

P	Q	$P \wedge Q$
True	True	True
True	False	False
False	True	False
False	False	False

For disjunction:

P	Q	$P \vee Q$
True	True	True
False	True	True
True	False	True
False	False	False

Truth Tables:

For Implication:

P	Q	$P \rightarrow Q$
True	True	True
True	False	False
False	True	True
False	False	True

For Biconditional:

P	Q	$P \leftrightarrow Q$
True	True	True
True	False	False
False	True	False
False	False	True

Truth table with three propositions:

P	Q	R	$\neg R$	$P \vee Q$	$P \vee Q \rightarrow \neg R$
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

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 $A \Leftrightarrow B$. In below truth table we can see that column for $\neg A \vee B$ and $A \rightarrow B$, are identical hence A is Equivalent to B

A	B	$\neg A$	$\neg A \vee B$	$A \rightarrow B$
T	T	F	T	T
T	F	F	F	F
F	T	T	T	T
F	F	T	T	T

Properties of Operators:

□ Commutativity:

- $P \wedge Q = Q \wedge P$, 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:

- $P \wedge \text{True} = P$,
- $P \vee \text{True} = \text{True}$.

□ 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:

- $\neg (\neg P) = P$.

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.



Propositional Logic Rules of Inference

Inference and its Rules

□ 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.**

Inference and its Rules

□ Following are some terminologies related to inference rules:

Implication: It is one of the logical connectives which can be represented as $P \rightarrow Q$. 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 $Q \rightarrow P$.

Contrapositive: The negation of converse is termed as contrapositive, and it can be represented as $\neg Q \rightarrow \neg P$.

Inverse: The negation of implication is called inverse. It can be represented as $\neg P \rightarrow \neg Q$.

$P \rightarrow Q$ is equivalent to $\neg Q \rightarrow \neg P$ and vice versa

P	Q	$P \rightarrow Q$	$Q \rightarrow P$	$\neg Q \rightarrow \neg P$	$\neg P \rightarrow \neg Q$
T	T	T	T	T	T
T	F	F	T	F	T
F	T	T	F	T	F
F	F	T	T	T	T

□ From the above truth table, we can prove that $P \rightarrow Q$ is equivalent to $\neg Q \rightarrow \neg P$, and $Q \rightarrow P$ is equivalent to $\neg P \rightarrow \neg Q$

Types of Inference rules:

1. Modus Ponens
2. Modus Tollens
3. Hypothetical Syllogism
4. Disjunctive Syllogism
5. Addition
6. Simplification
7. Resolution

Modus Ponens

- The Modus Ponens rule is one of the most important rules of inference, and it states that if P and $P \rightarrow Q$ is true, then we can infer that Q will be true.

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

□ Example:

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

Proof by Truth table:

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

Modus Tollens

- The Modus Tollens rule state that if $P \rightarrow Q$ is true and $\neg Q$ is true, then $\neg P$ will also true.

Notation for Modus Tollens:
$$\frac{P \rightarrow Q, \neg Q}{\neg P}$$

□ Example:

- **Statement-1:** "If I am sleepy then I go to bed" $\Rightarrow P \rightarrow Q$
- **Statement-2:** "I do not go to the bed." $\Rightarrow \neg Q$
- **Statement-3:** Which infers that "I am not sleepy" $\Rightarrow \neg P$

Proof by Truth table:

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

Hypothetical Syllogism

□ The Hypothetical Syllogism rule state that if $P \rightarrow R$ is true whenever $P \rightarrow Q$ is true, and $Q \rightarrow R$ is true.

□ Example:

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

Proof by truth table:

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

Disjunctive Syllogism

- The Disjunctive syllogism rule state that if $P \vee Q$ is true, and $\neg P$ is true, then Q will be true.

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

Example:

- **Statement-1:** Today is Sunday or Monday. $\implies P \vee Q$
- **Statement-2:** Today is not Sunday. $\implies \neg P$
- **Conclusion:** Today is Monday. $\implies Q$

Proof by truth-table:

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

Addition

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

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

Example:

- **Statement:** I have a vanilla ice-cream. $\implies P$.
- **Statement-2:** I have Chocolate ice-cream.
- **Conclusion:** I have vanilla or chocolate ice-cream. $\implies (P \vee Q)$

Proof by Truth-Table:

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


Simplification

- The simplification rule state that if $P \wedge Q$ is true, then Q or P will also be true.

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





First-Order Logic

Problems with PL

- ❑ In the topic of PL, we have seen that 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 the 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.
- ❑ Example:
 - "Some humans are intelligent", or
 - "All humans are intelligent",
- ❑ To represent the above statements, PL logic is not sufficient, so we required some more powerful logic, such as first-order logic.

First-Order logic

- ❑ 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



- **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:
 - **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, ...

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	\wedge , \vee , \neg , \Rightarrow , \Leftrightarrow
Equality	$=$
Quantifier	\forall , \exists

Atomic and Compound sentences

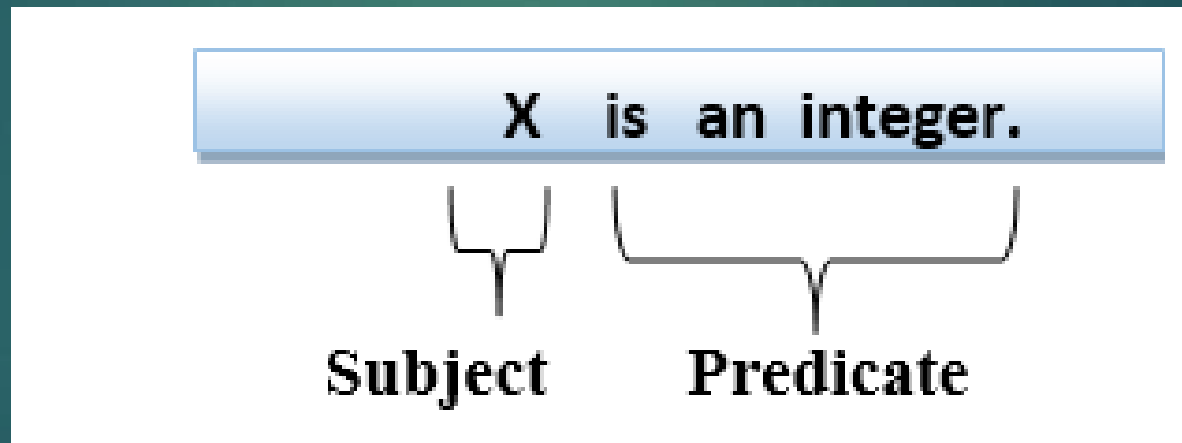
- ❑ Atomic sentences are the most basic sentences of first-order logic.
- ❑ These sentences are formed from a predicate symbol followed by a parenthesis with a sequence of terms.
- ❑ We can represent atomic sentences as **Predicate (term1, term2,, term n)**.
 - ❑ **Example:**
 - Ravi and Ajay are brothers: => **Brothers(Ravi, Ajay)**.
 - Chinky is a cat: => **cat (Chinky)**.

- 
- ❑ Complex sentences 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:
 - ❑ **Universal Quantifier, (for all, everyone, everything)**
 - ❑ **Existential quantifier, (for some, at least one).**

Universal Quantifier

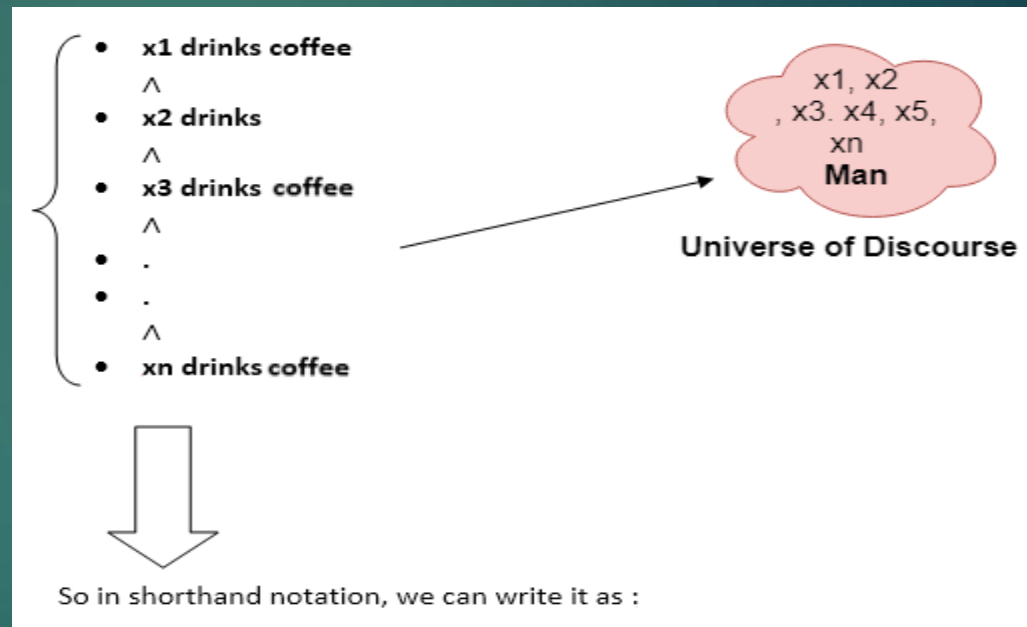
- Universal quantifier is a symbol of logical representation, which specifies that the statement within its range is true for everything or every instance of a particular thing.
- The Universal quantifier is represented by a symbol \forall , which resembles an inverted A.
- If x is a variable, then $\forall x$ is read as:
 - For all x
 - For each x
 - For every x .

Universal Quantifier

□ Example:

□ **All man drink coffee.**

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



□ $\forall x \text{ man}(x) \rightarrow \text{drink}(x, \text{coffee}).$

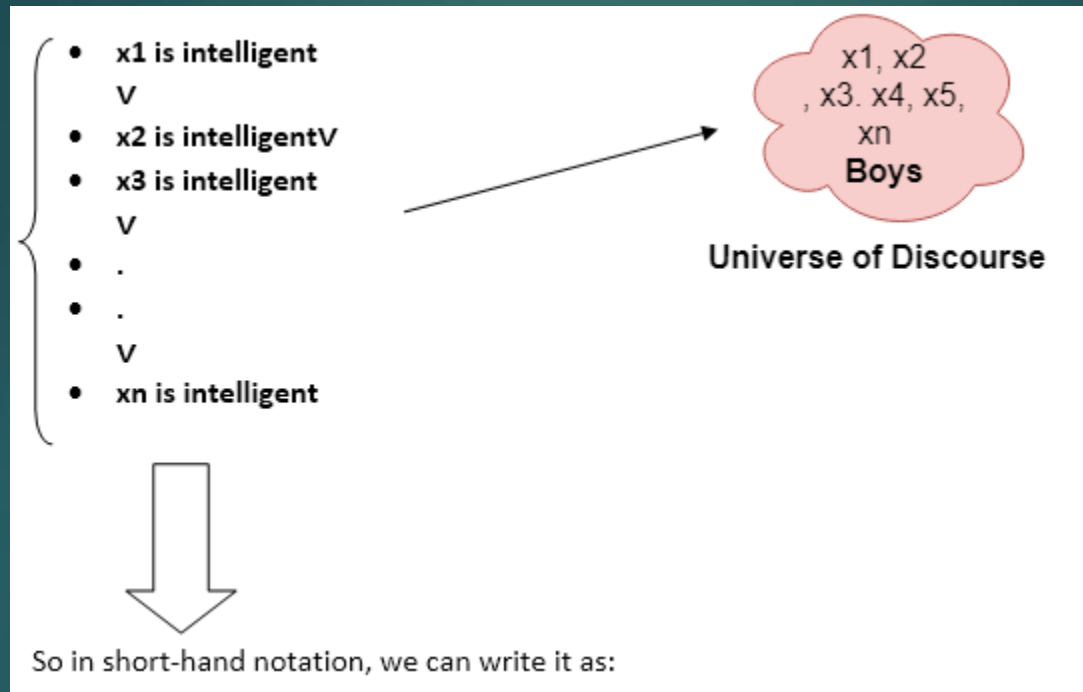
□ It will be read as: There are all x where x is a man who drink coffee.

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.
- ❑ 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.



- $\exists x: \text{boys}(x) \wedge \text{intelligent}(x)$
- 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 $\forall y \exists x$.

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 so it will be represented as follows.

$$\forall x \text{ bird}(x) \rightarrow \text{fly}(x).$$

1. Every man respects his parent.

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:

$$\forall x \text{ man}(x) \rightarrow \text{respects}(x, \text{parent}).$$

Examples of FOL using quantifier

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:**


$$\exists x \text{ boys}(x) \rightarrow \text{play}(x, \text{cricket}).$$

4. All students do not 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:

$$\neg \forall (x) [\text{student}(x) \rightarrow \text{like}(x, \text{Mathematics}) \wedge \text{like}(x, \text{Science})].$$



Inference in First-Order Logic

Inference in FOL

- ❑ Inference in First-Order Logic is used to deduce new facts or sentences from existing sentences.
- ❑ Before understanding the FOL inference rule, let's understand some basic terminologies used in FOL
- ❑ **Substitution:**
 - Substitution is a fundamental operation performed on terms and formulas.
 - It occurs in all inference systems in first-order logic.
 - The substitution is complex in the presence of quantifiers in FOL.
 - If we write $F[a/x]$, so it refers to substitute a constant "a" in place of variable "x".

Note: First-order logic is capable of expressing facts about some or all objects in the universe.

Inference in FOL

□ Equality:

- First-Order logic does not only use predicate and terms for making atomic sentences but also uses another way, which is equality in FOL.
 - Sentence: Brother of John is Smith
 - $\text{Brother}(\text{John}, \text{smith})$
 - **Example:** $\text{Brother}(\text{John}) = \text{Smith}$.
- As in the above example, the object referred by the **Brother (John)** is similar to the object referred by **Smith**. The equality symbol can also be used with negation to represent that two terms are not the same objects.
- **Example:** $\neg(x=y)$ which is equivalent to $x \neq y$.

FOL inference rules for quantifier

- As propositional logic we also have inference rules in first-order logic, so following are some basic inference rules in FOL:
 - **Universal Generalization**
 - **Universal Instantiation**
 - **Existential Instantiation**
 - **Existential introduction**

Universal Generalization

- Universal generalization is a valid inference rule which states that if premise $P(c)$ is true for any arbitrary element c in the universe of discourse, then we can have a conclusion as $\forall x P(x)$.

- It can be represented as:

$$\frac{P(c)}{\forall x P(x)}$$

- This rule can be used if we want to show that every element has a similar property.
- **Example:** Let's represent, $P(c)$: "**A byte contains 8 bits**", so for $\forall x$ $P(x)$ "**All bytes contain 8 bits.**", it will also be true.

Universal Instantiation

- ❑ Universal instantiation is also called as **universal elimination** or **UI** is a valid inference rule. It can be applied multiple times to add new sentences.
- ❑ As per UI, **we can infer any sentence obtained by substituting a ground term for the variable.**
- ❑ The UI rule state that we can infer any sentence $P(c)$ by substituting a ground term c (a constant within domain x) from $\forall x P(x)$ for any object in the universe of discourse.
- ❑ It can be represented as:

$$\frac{\forall x P(x)}{P(c)}$$

Universal Instantiation

□ Example 1:

- IF "Every person like ice-cream" $\Rightarrow \forall x P(x)$ so we can infer that "John likes ice-cream" $\Rightarrow P(c)$

□ Example: 2:

- Let's take a famous example.
- "All kings who are greedy are Evil."
- So let our knowledge base contains this detail as in the form of FOL:
- $\forall x \text{ king}(x) \wedge \text{greedy}(x) \rightarrow \text{Evil}(x),$
- So from this information, we can infer any of the following statements using Universal Instantiation:
- $\text{King}(\text{John}) \wedge \text{Greedy}(\text{John}) \rightarrow \text{Evil}(\text{John}),$
- $\text{King}(\text{Richard}) \wedge \text{Greedy}(\text{Richard}) \rightarrow \text{Evil}(\text{Richard}),$
- $\text{King}(\text{Father}(\text{John})) \wedge \text{Greedy}(\text{Father}(\text{John})) \rightarrow \text{Evil}(\text{Father}(\text{John})),$

Existential Instantiation

- ❑ Existential instantiation is also called as **Existential Elimination**, which is a valid inference rule in first-order logic.
- ❑ It can be applied only once to replace the **existential sentence**.
- ❑ The new KB is not logically equivalent to old KB, but it will be satisfiable if old KB was satisfiable.
- ❑ This rule states that **one can infer $P(c)$** from the formula given in the form of **$\exists x P(x)$ for a new constant symbol c** .
- ❑ The restriction with this rule is that c used in the rule must be a new term for which $P(c)$ is true.
- ❑ It can be represented as:

$$\frac{\exists x P(x)}{P(c)}$$

Existential introduction

- ▶ An existential introduction is also known as an **existential generalization**, which is a valid inference rule in first-order logic.
- ▶ This rule states that if there is **some element c in the universe of discourse** which has a **property P** , then we can infer that there exists **something in the universe which has the property P** .

- It can be represented as

$$\frac{P(c)}{\exists x P(x)}$$

- **Example:** Let's say that,
"Priyanka got good marks in English."
"Therefore, someone got good marks in English."



Unification

What is Unification?

- Unification is a process of making two different logical atomic expressions identical by finding a substitution.
- Unification depends on the substitution process.
- It takes two literals as input and makes them identical using substitution.
- Let Ψ_1 and Ψ_2 be two atomic sentences and σ be a unifier such that, $\Psi_1\sigma = \Psi_2\sigma$, then it can be expressed as **UNIFY**(Ψ_1, Ψ_2).
- **Example 1:** Unify{King(x), King(John)}
- Let $\Psi_1 = \text{King}(x)$, $\Psi_2 = \text{King}(\text{John})$, **Substitution** $\theta = \{x/\text{John}\}$ is a unifier for these atoms and applying this substitution, and both expressions will be identical.

Unification:

- Let's say there are two different expressions, $P(x, y)$, and $P(a, f(z))$.
 - In this example, we need to make both above statements identical to each other. For this, we will perform the substitution.
 - $$\begin{array}{ll} P(x, y) & \text{..... (i)} \\ P(a, f(z)) & \text{..... (ii)} \end{array}$$
 - Substitute x with a , and y with $f(z)$ in the first expression, and it will be represented as a/x and $f(z)/y$.
 - With both the substitutions, the first expression will be identical to the second expression and the substitution set will be: $[a/x, f(z)/y]$.

Conditions for Unification:

□ Following are some basic conditions for unification:

- Predicate symbol must be same, atoms or expression with different predicate symbol can never be unified.
- Number of Arguments in both expressions must be identical.
- Unification will fail if there are two similar variables present in the same expression.

Example 1

□ UNIFY(knows(Richard, x), knows(Richard, John))

Here, $\psi_1 = \text{knows}(\text{Richard}, x)$, and $\psi_2 = \text{knows}(\text{Richard}, \text{John})$

$S_0 \Rightarrow \{ \text{knows}(\text{Richard}, x); \text{knows}(\text{Richard}, \text{John}) \}$

SUBST $\theta = \{x/\text{John}\}$

$S_1 \Rightarrow \{ \text{knows}(\text{Richard}, \text{John}); \text{knows}(\text{Richard}, \text{John}) \}$, **Successfully Unified.**

Unifier: $\{x/\text{John}\}$.

Example 2

□ **UNIFY**(prime(11), prime(y))

Here, $\Psi_1 = \{\text{prime}(11)\}$, and $\Psi_2 = \{\text{prime}(y)\}$

$S_0 \Rightarrow \{\text{prime}(11), \text{prime}(y)\}$

$\text{SUBST } \theta = \{Y/11\}$

$S_1 \Rightarrow \{\text{prime}(11), \text{prime}(11)\}$, Successfully unified.

Unifier: $\{Y/11\}$.

Example 3

□ **UNIFY**{**p(f(a), g(Y))** and **p(X, X)**}

Sol: $S_0 \Rightarrow$ Here, $\Psi_1 = p(f(a), g(Y))$, and $\Psi_2 = p(X, X)$

SUBST $\theta = \{X/f(a)\}$

$S1 \Rightarrow \Psi_1 = p(f(a), g(Y))$, and $\Psi_2 = p(f(a), f(a))$

SUBST $\theta = \{g(y)/f(a)\}$, **Unification failed.**

Unification is not possible for these expressions.

Example 4

□ **UNIFY**{ **$p(b, X, f(g(Z)))$ and $p(Z, f(Y), f(Y))$** }

Here, $\psi_1 = p(b, X, f(g(Z)))$, and $\psi_2 = p(Z, f(Y), f(Y))$

$S_0 \Rightarrow \{ p(b, X, f(g(Z))); p(Z, f(Y), f(Y)) \}$

SUBST $\theta = \{Z/b\}$

$S_1 \Rightarrow \{ p(b, X, f(g(b))); p(b, f(Y), f(Y)) \}$

SUBST $\theta = \{X/f(Y)\}$

$S_2 \Rightarrow \{ p(b, f(Y), f(g(b))); p(b, f(Y), f(Y)) \}$

SUBST $\theta = \{Y/g(b)\}$

$S_2 \Rightarrow \{ p(b, f(g(b)), f(g(b))); p(b, f(g(b)), f(g(b))) \}$

Unified Successfully.

And Unifier = $\{ b/Z, f(Y) /X, g(b) /Y \}$.

Example 5

► UNIFFY {p (X, X), and p (Z, f(Z))}

Here, $\psi_1 = \{p(X, X)\}$, and $\psi_2 = \{p(Z, f(Z))\}$

$S_0 \Rightarrow \{p(X, X), p(Z, f(Z))\}$

$SUBST \theta = \{Z/X\}$

$S_1 \Rightarrow \{p(Z, Z), p(Z, f(Z))\}$

$SUBST \theta = \{Z/f(Z)\}$, **Unification Failed.**

Hence, unification is not possible for these expressions.

Example 6

□ **UNIFY $Q(a, g(X, a), f(Y)), Q(a, g(f(b), a), X)$**

Here, $\psi_1 = Q(a, g(x, a), f(y))$, and $\psi_2 = Q(a, g(f(b), a), x)$

$S_0 \Rightarrow \{Q(a, g(x, a), f(y)); Q(a, g(f(b), a), x)\}$

SUBST $\theta = \{x/f(b)\}$

$S_1 \Rightarrow \{Q(a, g(f(b), a), f(y)); Q(a, g(f(b), a), f(b))\}$

SUBST $\theta = \{y/b\}$

$S_1 \Rightarrow \{Q(a, g(f(b), a), f(b)); Q(a, g(f(b), a), f(b))\}$, **Successfully Unified.**

Unifier: $[a/a, x/f(b), y/b]$.

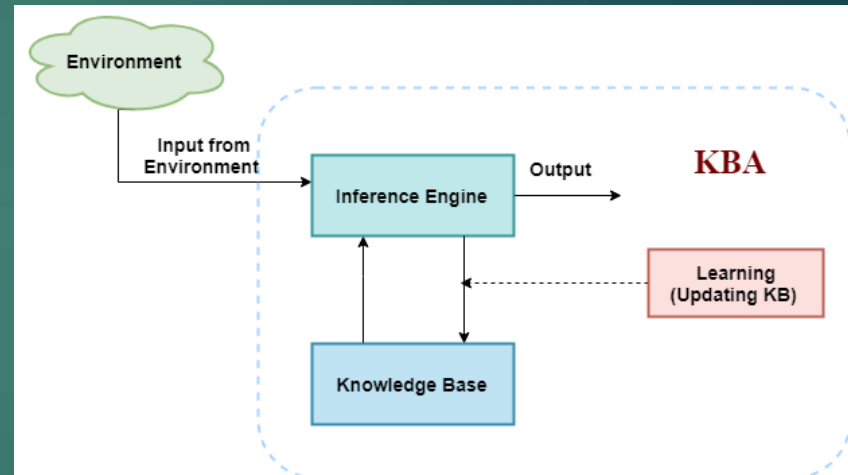


Forward and Backward chaining

FC and BC in AI

- ❑ In artificial intelligence, forward and backward chaining is one of the important topics, but before understanding forward and backward chaining lets first understand Inference Engine.

❑ Inference engine:



- ❑ The inference engine is a part of the expert system. Inference engine commonly proceeds in two modes, which are:
 - **Forward chaining**
 - **Backward chaining**

Forward Chaining

- ❑ Forward chaining is also known as a forward deduction or forward reasoning method when using an inference engine.
- ❑ Forward chaining is a form of reasoning which start with atomic sentences in the knowledge base and applies inference rules (Modus Ponens) in the forward direction to extract more data until a goal is reached.
- ❑ 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.

Properties of Forward-Chaining:

- It is a **down-up approach**, as it moves from bottom to top.
- It is a process of making a conclusion based on known facts or data, **by starting from the initial state and reaches the goal state**.
- Forward-chaining approach is also called as **data-driven** as we reach to the goal using available data.
- Forward -chaining approach is commonly used in the expert system, such as business, and production rule systems.

Example:

- ❑ Consider the following sentences:
 - a) John likes all kinds of food
 - b) Anything anyone eats and not killed is food.
 - c) Anil eats peanuts and is still alive.
 - d) Harry eats everything that Anil eats.

- ❑ Translate the above sentences into predicate logic & using FC and BC prove that:
 - John likes peanuts

Conversion of Facts into FOL

- a) John likes all kinds of food
- b) Anyone anything eats and not killed is food.
- c) Anil eats peanuts and is still alive.
- d) Harry eats everything that Anil eats.

▪ John likes peanuts

- $\forall x \text{ food}(x) \rightarrow \text{likes}(\text{John}, x)$
 - $\forall x \forall y: \text{eats}(x, y) \wedge \neg \text{killed}(x) \rightarrow \text{food}(y)$
 - $\text{eats}(\text{Anil}, \text{Peanuts}) \wedge \text{alive}(\text{Anil})$
 - $\forall x: \text{eats}(\text{Anil}, x) \rightarrow \text{eats}(\text{Harry}, x)$
 - $\forall x \neg \text{killed}(x) \rightarrow \text{alive}(x)$
 - $\forall x \text{ alive}(x) \rightarrow \neg \text{killed}(x)$
-
- Needs to prove:
 - $\text{likes}(\text{John}, \text{Peanuts})$.

■ Proof using Forward chaining:

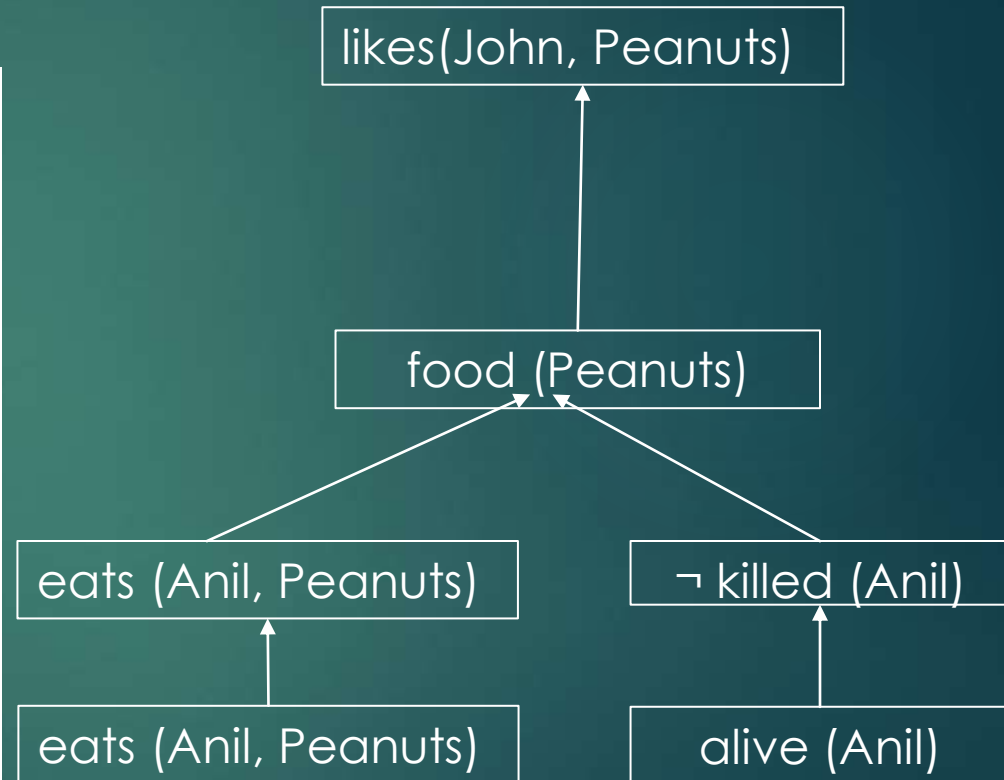
Modus
Ponens

$P \rightarrow Q, P$	$P \rightarrow Q, Q$
-----	-----
Q	P

- $\forall x \text{ food}(x) \rightarrow \text{likes}(\text{John}, x)$
- $\forall x \forall y: \text{eats}(x, y) \wedge \neg \text{killed}(x) \rightarrow \text{food}(y)$
- $\text{eats}(\text{Anil}, \text{Peanuts}) \wedge \text{alive}(\text{Anil})$
- $\forall x: \text{eats}(\text{Anil}, x) \rightarrow \text{eats}(\text{Harry}, x)$
- $\forall x \neg \text{killed}(x) \rightarrow \text{alive}(x)$
- $\forall x \text{ alive}(x) \rightarrow \neg \text{killed}(x)$

□ Needs to prove:

- $\text{likes}(\text{John}, \text{Peanuts})$.



Proof using Backward chaining:

$\frac{P \rightarrow Q, P}{Q}$	$\frac{P \rightarrow Q, Q}{P}$
--------------------------------	--------------------------------

- $\forall x \text{ food}(x) \rightarrow \text{likes}(\text{John}, x)$
- $\forall x \forall y: \text{eats}(x, y) \wedge \neg \text{killed}(x) \rightarrow \text{food}(y)$
- $\text{eats}(\text{Anil}, \text{Peanuts}) \wedge \text{alive}(\text{Anil})$
- $\forall x: \text{eats}(\text{Anil}, x) \rightarrow \text{eats}(\text{Harry}, x)$
- $\forall x \neg \text{killed}(x) \rightarrow \text{alive}(x)$
- $\forall x \text{ alive}(x) \rightarrow \neg \text{killed}(x)$

□ Needs to prove:

- $\text{likes}(\text{John}, \text{Peanuts})$.

