

- Definition:

- Involves gathering expertise for use in expert systems, organizing it into structured formats like IF-THEN rules.
- Also refers to the process of absorbing information into memory, focusing on effective retrieval.

- Procedure:

- Identification: Break down complex issues into parts.
- Conceptualization: Define key concepts.
- Formalization: Organize knowledge formally for programmatic use.
- Implementation: Code the structured knowledge.
- Testing: Check and validate the system's functionality.

Fact:

"Book X is available in the library."

Representation in Logical Structure:

Using a logical statement, this fact can be represented as:

`Available(Book X)`

Characteristics Applied:

- **Syntax and Semantics:** The function `Available()` clearly defines that it checks for the availability of a specific book.
- **Expressive Capacity:** This structure allows the system to represent the availability of any book in the library by substituting "Book X" with any book's name.
- **Efficiency:** Such representation supports quick queries about book availability, facilitating efficient information retrieval and decision-making processes in the library's computer system.

Chapter-3 (KNOWLEDGE REPRESENTATIONS)

Knowledge Representation

- **Definition**: It's the study of how knowledge can be represented in a computer system to mimic human thought and reasoning.
- **Components**:
 - Facts: True information relevant to a specific domain.
 - Representation: How facts are formally expressed, often using logical structures.
- **Example**: For the fact "Charlie is a dog," a logical representation could be Dog(Charlie).
- **Characteristics**:
 - **Syntax and semantics**: Clearly defined rules for structure and meaning.
 - **Expressive capacity**: Ability to convey all necessary domain knowledge.
 - **Efficiency**: Supports effective computing and reasoning.

- All birds fly: $\forall x (\text{Bird}(x) \rightarrow \text{Fly}(x))$
- Tweety is a bird: $\text{Bird}(\text{Tweety})$

We want to prove that Tweety can fly. Using resolution:

1. We negate the query $\neg \text{Fly}(\text{Tweety})$ and add it to our knowledge base.

2. Our clauses in clausal form are:

- $\neg \text{Bird}(x) \vee \text{Fly}(x)$ (from $\forall x (\text{Bird}(x) \rightarrow \text{Fly}(x))$)
- $\text{Bird}(\text{Tweety})$
- $\neg \text{Fly}(\text{Tweety})$ (negated query)

$$P \rightarrow Q \equiv \neg P \vee Q$$

3. Now we apply resolution:

- We resolve $\neg \text{Fly}(\text{Tweety})$ with $\neg \text{Bird}(x) \vee \text{Fly}(x)$ using unification by substituting x with Tweety.
- This gives us $\neg \text{Bird}(\text{Tweety}) \vee \text{Fly}(\text{Tweety})$.

4. Finally, we resolve $\text{Bird}(\text{Tweety})$ with $\neg \text{Bird}(\text{Tweety}) \vee \text{Fly}(\text{Tweety})$, which gives us $\text{Fly}(\text{Tweety})$, the desired fact.

Since we started with $\neg \text{Fly}(\text{Tweety})$ and ended up with $\text{Fly}(\text{Tweety})$, we have derived a contradiction, meaning our original negation of the query must be false, thus proving that, according to our knowledge base, Tweety can fly.

WWW.KNOWLEDGEGATE.IN

Chapter-3 (KNOWLEDGE REPRESENTATIONS)

-

Chapter-3 (KNOWLEDGE REPRESENTATIONS)

Forward Chaining

1. Data-driven approach: begins with known facts.
2. Bottom-up reasoning: builds up from facts to a conclusion.
3. Uses Modus Ponens: applies rules to derive new information.
$$\begin{array}{l} p \rightarrow q \\ p \\ \hline q \end{array}$$
4. Continues until no new information can be derived.
5. Common in real-time systems and production rule systems.

Backward Chaining

1. Goal-driven approach: starts with the desired conclusion.
2. Top-down reasoning: works backwards from goal to facts.
$$\begin{array}{l} p \rightarrow q \\ \sim q \\ \hline \sim p \end{array}$$
3. Uses Modus Tollens: checks if conclusions can lead to the goal.
4. Continues until it reaches the starting facts or rules.
5. Used in diagnostic systems where the end goal is known.

Example: Deciding on a Postgraduate Course

- **Goal:** A student wants to determine the best postgraduate course to enroll in.
- **Question:** What course should the student choose for a successful career?
- **Check Desired Outcome:** The desired outcome is a fulfilling and prosperous career in a field the student is passionate about.
- **Compare Interests:** The student is interested in technology and innovation.
- **Search for Career Paths:** The system considers various career paths that align with technology and innovation.
- **Hypothesis 1:** Is a Master's in Computer Science suitable?
- **Check for Evidence:** The system checks the student's academic background, skills, and job market trends for computer science graduates.
- **Hypothesis 2:** Could an MBA in Technology Management be an alternative?
- **Check for Evidence:** The system evaluates the student's leadership potential, market demand for technology managers, and the student's long-term career goals.
- **Solution if Matched:** If the evidence strongly supports one path over the other based on the student's profile and job market trends, the system recommends that course.
- **Outcome:** The student chooses the recommended course which aligns with their interests, background, and market demand, leading to a more focused educational path towards a desired career.

- Starts with a goal: looks at the desired outcome first. ✓
- Works backward: checks if current knowledge supports the goal.
- Goal-driven: each step is aimed at proving the end objective.
- Deduces necessities: determines what conditions must be met for the goal.
- Stops when it reaches the beginning facts or rules, or the goal is proven or disproven.

Example: A Rice Crop Management System

- **Initial Observation:** A farmer in Odisha notices that some rice plants are shorter than usual and have yellow leaves.
- **Rule 1:** If rice plants are short and have yellow leaves, they might lack nitrogen.
- **Rule 2:** If rice plants lack nitrogen, then using nitrogen-rich fertilizer may help.
- **Rule 3:** If the season is monsoon and the problem is nitrogen deficiency, then use urea fertilizer, which is suitable for wet conditions.

Forward Chaining Process:

1. **Observing Symptoms:** The farmer inputs into the system that the rice plants are short with yellow leaves.
2. **Applying Knowledge:** The system uses Rule 1 to deduce that the plants might be suffering from nitrogen deficiency.
3. **Taking Action:** Based on Rule 2, the system suggests that applying a nitrogen-rich fertilizer could be beneficial.
4. **Considering Conditions:** Seeing that it is the monsoon season, Rule 3 advises the farmer to use urea fertilizer specifically.
5. **Outcome:** The farmer follows the advice, applies urea, and over time the rice plants become healthier and grow taller.

- Begins with what is known: starts with basic facts or data.
- Moves forward: applies rules to these facts to derive new information
- Data-driven: new conclusions are made as data is processed.
- Adds knowledge: builds upon existing information incrementally.
- Stops when no further deductions can be made or a goal is reached.

Universal Generalization

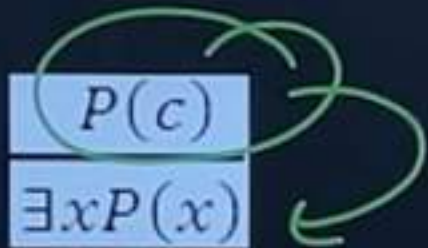
- Universal generalization is a valid inference rule which states that if premise $P(c)$ is true for any arbitrary element c in the universe of discourse, then we can have a conclusion as $\forall x P(x)$.
- It can be represented as:
$$\frac{P(c)}{\forall x P(x)}$$
- This rule can be used if we want to show that every element has a similar property. In this rule, x must not appear as a free variable.
- **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.

$$U = \{1, 2, 3, 4\}$$

$$P(1) \quad P(2) \quad P(3) \quad P(4)$$

$$\forall x P(x)$$

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 some element in the universe which has the property P .
- It can be represented as: The diagram shows a box containing $P(c)$ above a box containing $\exists xP(x)$. A green arrow points from the $P(c)$ box to the $\exists xP(x)$ box. Another green arrow starts from the $\exists xP(x)$ box, loops around, and points back to the $P(c)$ box, indicating a valid inference.
- **Example:** "Priyanka got good marks in English."
- "Therefore, someone got good marks in English."

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.
- 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.

$$x = \{1, 2, 3, 4\}$$

- It can be represented as:

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

- 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:

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

$$x = \{1, 2, 3, 4\}$$

Example: "Every person like ice-cream":

$$\forall x P(x)$$

$$P(2)$$

So we can infer that "John likes ice-cream": $P(c)$

3. CNF requires \neg to appear only in literals, so we move \neg inwards by repeated application of the following equivalences

In the example, we require just one application of the last rule:

- $(\neg A \vee B \vee C) \wedge ((\neg B \wedge \neg C) \vee A).$

4. Now we have a sentence containing nested \wedge and \vee operators applied to literals. We apply the distributivity law, distributing \vee over \wedge wherever possible.

- $(\neg A \vee B \vee C) \wedge (\neg B \vee A) \wedge (\neg C \vee A).$

- The original sentence is now in CNF, as a conjunction of three clauses.

- A formula is in conjunctive normal form (CNF) or clausal normal form if it is a conjunction of one or more clauses, where a clause is a disjunction of literals; otherwise put, it is an AND of ORs.
- **Procedure for converting Propositional logic to CNF:**
- **Example:** $A \Leftrightarrow (B \vee C)$ into CNF.
 1. Eliminate \Leftrightarrow , replacing $\alpha \Leftrightarrow \beta$ with $(\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha)$.
 $(A \Rightarrow (B \vee C)) \wedge ((B \vee C) \Rightarrow A)$
 2. Eliminate \Rightarrow , replacing $\alpha \Rightarrow \beta$ with $\neg \alpha \vee \beta$:
 $(\neg A \vee B \vee C) \wedge (\neg(B \vee C) \vee A)$.

$$\underline{ab} + \underline{cd}$$

$$\underline{(a+b)} \cdot \underline{(c+d)}$$

Negation

- $\neg [\forall_x P(x)] = \exists_x \neg P(x)$
- $\neg [\exists_x P(x)] = \forall_x \neg P(x)$

- Existential quantifiers: - with existential quantifier of a propositional that is true if and only if $P(x)$ is true for at least one value of x in the universe of discourse.
- There exists an element x in the universe of discourse such that $P(x)$ is true.
- $\exists_x P(x)$, i.e. for at least one value of x $P(x)$ is true

- Let try some other statement 'Some Indian like samosa'

- if i say four Indian are there I_1, I_2, I_3, I_4
- I_1 like samosa $\vee I_2$ like samosa $\vee I_3$ like samosa $\vee I_4$ like samosa
- $\text{Samosa}(I_1) \vee \text{Samosa}(I_2) \vee \text{Samosa}(I_3) \vee \text{Samosa}(I_4)$
- $\exists_x \text{Samosa}(x)$, if we confine x to be Indian then it means some x likes samosa.

- Universal quantifiers: - The universal quantification of a propositional function is the proposition that asserts
 - $P(x)$ is true for all values of x in the universe of discourse.
 - The universe of discourse specifies the possible value of x .
 - $\forall_x P(x)$, i.e. for all value of a $P(x)$ is true