UNIT-4

Knowledge Representation

Knowledge Representation (ज्ञान अभिविन्यास / प्रस्तुतीकरण) कृत्रिम बुद्धिमत्ता (AI) का वह क्षेत्र है जिसमें वास्तविक दुनिया के तथ्यों, वस्तुओं, अवधारणाओं, संबंधों और नियमों को ऐसे व्यवस्थित और संरचित रूप में प्रस्तुत किया जाता है, जिसे एक कंप्यूटर प्रणाली समझ सके, उस पर तर्क कर सके, निर्णय ले सके और समस्याएँ हल कर सके।

दूसरे शब्दों में, Knowledge Representation वह तरीका है जिसके माध्यम से मानव ज्ञान को मशीन-पठनीय (machine readable) रूप में बदलकर AI सिस्टम को मानव जैसी बुद्धिमत्ता प्रदर्शित करने योग्य बनाया जाता है।

यह सिर्फ डेटा संग्रह नहीं है, बल्कि ज्ञान को सार्थक (meaningful) तरीके से संगठित करना है ताकि मशीन:

- ज्ञान संग्रहित कर सके,
- उसे याद रख सके,
- सही समय पर निकाल सके,
- उसके आधार पर तर्क (inference) कर सके,
- और नई जानकारी मिलने पर अपने ज्ञान को अपडेट कर सके।

Knowledge Representation AI का एक महत्वपूर्ण घटक है, जो विभिन्न प्रकार के ज्ञान—जैसे तथ्य (facts), नियम (rules), संबंध (relationships), वस्तुओं के गुण (attributes) और प्रक्रियाएँ (procedures)—को औपचारिक (formal) और सुसंगत (structured) रूप में व्यवस्थित करता है। इसका उद्देश्य ज्ञान को ऐसे प्रतीकों और संरचनाओं में बदलना है जिन्हें कंप्यूटर समझ सके और उनमें निहित अर्थ को logically process कर सके।

AI सिस्टम इस ज्ञान का उपयोग करके:

- नए निष्कर्ष निकालता है,
- निर्णय लेता है,
- प्राकृतिक भाषा को समझता है,
- योजना बनाता है,
- समस्याओं का समाधान करता है,
- और मानव विशेषज्ञ जैसा व्यवहार प्रस्तुत करता है।

Knowledge Representation इस बात पर केंद्रित होती है कि **ज्ञान को किस प्रकार दर्शाया जाए** ताकि वह मशीन के लिए अधिकतम उपयोगी, प्रभावी और तर्कसंगत बन सके।

Approaches in Artificial Intelligence (AI)

AI में "Approaches" का मतलब होता है – वह तरीके या method जिनसे AI system सोचना, सीखना और problem solve करना सीखता है।
हर approach की अपनी एक philosophy होती है कि machine को कैसे intelligent बनाया जाए – कुछ human thinking की नकल करते हैं, कुछ data learning पर आधारित होते हैं और कुछ logic व search methods पर।

1. Symbolic / Knowledge-Based Approach (Top-Down Approach)

इस approach में AI system को human knowledge symbols, facts, और rules के रूप में दिया जाता है। System इन rules को apply करके reasoning करता है और decisions लेता है।
इसमें सारी जानकारी logically define की जाती है।

Example:

- Expert Systems जैसे MYCIN
- "If fever AND cough → Possible flu"

2. Sub-Symbolic / Machine Learning Approach (Bottom-Up Approach)

इस approach में AI को manually rules देने की बजाय उसे data से खुद सीखने दिया जाता है।
Neural Networks, Deep Learning, SVM, Decision Trees जैसी techniques इसी category में आती हैं।
AI खुद patterns सीखकर performance improve करता है।

Example:

- Image Recognition (cat-dog पहचानना)
- Voice Assistants जैसे Siri, Alexa

3. Search-Based Approach

इस approach में AI किसी भी problem को एक **state space** की तरह मानता है और solution पाने के लिए states को **search algorithms** (BFS, DFS, A*, आदि) से explore करता है।
Goal तक पहुँचने के लिए विभिन्न paths check किए जाते हैं।

Example:

- Google Maps shortest route निकालता है
- 8-puzzle, chess move calculation

4. Statistical and Probabilistic Approachइस approach में uncertainty वाली situations को handle करने के लिए **probability** और **statistics** का उपयोग किया जाता है। Bayesian Networks, HMM, MDP जैसे models इसी में आते हैं।

Example:

- Weather prediction
- Spam mail detection

Issues in Artificial Intelligence (AI)

AI में "Issues" का मतलब है – **वह challenges, limitations या problems** जो intelligent systems बनाने और चलाने में आती हैं।
ये issues technical भी होते हैं और ethical भी।

1. Knowledge Representation Problem

AI systems को दुनिया की बहुत सारी knowledge को structured रूप में represent करना पड़ता है। लेकिन complex real-world knowledge को **rules, symbols या logic** में बदलना बहुत मुश्किल होता है।

Example:

- Common sense knowledge ("ice is cold") machine को समझाना कठिन है।

2. Reasoning and Inference Issue

AI system को available facts से conclusion निकालना होता है।
लेकिन कई बार जानकारी **incomplete**, **uncertain**, या **contradictory** होती है, जिससे reasoning मुश्किल हो जाती है।

Example:

- मेडिकल AI को symptoms अधूरे होने पर भी diagnosis करना पड़ता है।

3. Learning from Limited Data

Machine Learning models को efficient बनने के लिए बहुत ज़्यादा data चाहिए।
अगर data कम, गलत या biased हो तो AI गलत output देगा।

Example:

- Facial recognition कम data पर गलत पहचान करता है।

4. Natural Language Understanding Issue

Human language complex, ambiguous और context-dependent होती है।
AI के लिए sentences, emotions, sarcasm आदि समझना बहुत कठिन है।

Example:

- "I saw a man with a telescope" — telescope किसके पास है, AI समझ नहीं पाता।

5. Perception and Vision Issues

Computer Vision या sensors real world को accurately sense नहीं कर पाते क्योंकि environment unpredictable होता है।

Example:

- Self-driving car को fog या low light में objects समझने में दिक्कत होती है।

6. Ethical and Social Issues

AI से privacy loss, job loss, fairness और misuse जैसी समस्याएँ पैदा हो सकती हैं।
अगर data biased हो तो AI भी biased decisions लेगा।

Example:

- Hiring AI system female profiles reject कर दे - क्योंकि training data biased था।

7. High Computational Cost

AI systems को heavy hardware (GPU/TPU), high memory और long training time चाहिए, जिससे cost बढ़ती है।

Example:

- Deep Learning model train करने में कई हजार रुपये या डॉलर खर्च हो जाते हैं।

8. "Black Box" Issue (Lack of Explainability)

Deep Learning models अपने decision का reason explain नहीं कर पाते।
User नहीं जान पाता कि prediction कैसे आया, जिससे trust कम होता है।

Example:

- AI बताता है कि "tumhe diabetes ka high risk hai",
  लेकिन *kaise decide किया – explain नहीं करता।*

Frame in Artificial Intelligence

Frame AI का एक **knowledge representation structure** है, जिसमें किसी object, situation या concept से जुड़ी सारी जानकारी को **एक organized template** के रूप में store किया जाता है। Frame एक तरह का **data structure** होता है जिसमें **slots** (attributes) और उनके **values** (details) होते हैं।

Frame का मुख्य उद्देश्य वास्तविक दुनिया की चीज़ों को **structured and hierarchical form** में represent करना है ताकि AI system आसानी से:

- object के properties समझ सके,
- inheritance apply कर सके (parent से child में properties आना),
- reasoning कर सके,
- और missing information को default values से fill कर सके।

Frames complex knowledge को छोटे-छोटे **units** में divide करके store करते हैं, जिससे AI को information को process करना आसान हो जाता है।
यह **Semantic Networks** की तरह होते हैं, लेकिन और अधिक structured और powerful होते हैं।

Frames में तीन महत्वपूर्ण चीज़ें होती हैं:

1. **Frame Name** – किस concept/object के लिए frame है
2. **Slots** – उस object की properties
3. **Fillers** – slots के values

AI में Frames का उपयोग **expert systems, natural language understanding, planning, scene representation, vision systems** आदि में किया जाता है।

Frame एक ऐसा **template** है जिसमें किसी चीज़ से जुड़ी सारी जानकारी "slots" में रखी जाती है, ताकि AI उसे समझकर reasoning कर सके।

Example of Frame (Human Example)

Frame: PERSON

Frame Name: PERSON

Slots and Values:

Name: (Aarti, Rohan, etc.)

Age: (value)

Gender: (Male/Female)

Occupation: (Teacher, Student, Engineer)

Address: (City name)

Height: (value)

Weight: (value)

इस Frame में:

- **PERSON** object है
- **Slots** = Name, Age, Gender, Occupation
- **Slot Values** = specific व्यक्ति की जानकारी

अगर हम PERSON Frame को किसी specific व्यक्ति के लिए भरें:

Frame Example: PERSON–AARTI

Frame Name: PERSON-AARTI

Slots:

Name: Aarti

Age: 25

Gender: Female

Occupation: Computer Teacher

Address: Raipur

AI इस Frame के आधार पर reasoning कर सकता है, जैसे:

- Aarti is a Person → inherits all properties of PERSON frame
- If Occupation = Teacher → role-based reasoning कर सकता है

Example of Frame (Animal Example)

Frame: DOG

Frame Name: DOG

Slots:

Type: Mammal

Sound: Bark

Legs: 4

Color: (variable)

Breed: (variable)

Food: Non-vegetarian

Frame: DOG–TOMMY

Frame Name: DOG-TOMMY

Slots:

Color: Brown

Breed: Labrador

Age: 3

AI automatically समझेगा:

- TOMMY is a DOG → DOG के default slots inherit करेगा
- Sound = Bark
- Legs = 4

Conceptual Dependency (CD) in AI

Conceptual Dependency (CD) Artificial Intelligence में एक **knowledge representation model** है जिसे 1970 में **Roger Schank** द्वारा develop किया गया था। इसका main उद्देश्य **natural language sentences** को ऐसे **concept-level representation** में बदलना है जो:

- language-independent हो
- ambiguity कम करे
- computer को human-like understanding करने में मदद करे
- actions और events को primitive conceptual units में represent करे

इसका मतलब:
CD किसी भी sentence को उसके मूल अर्थ (meaning) में convert करता है, चाहे वह sentence किसी भी language में लिखा हो।

मुख्य उद्देश्य (Objectives of CD)

1. **Universal meaning representation**
   एक ही meaning को अलग-अलग languages में लिखे गए sentences से भी **same conceptual structure** मिल सके।
2. **Ambiguity removal**
   Sentences के confusion और multiple meaning को कम करता है।
3. **Deep understanding**
   Computer को सिर्फ शब्द नहीं, बल्कि उनका real meaning समझने देता है।
4. **Inference making**
   चूँकि CD में deep semantic structure होता है, AI system logically infer भी कर सकता है।

Conceptual Dependency की Key Concepts

CD representation **primitive ACTs (Actions)** और **conceptual roles** पर आधारित होती है।

1. Primitive ACTs

ये ऐसे basic actions होते हैं जिनसे सभी complex actions बने होते हैं।

कुछ important primitive ACTs:

Primitive ACT	Meaning
ATRANS	transfer of possession (देना-लेना)
PTRANS	physical movement
MTRANS	mental transfer (सोचना, जानकारी देना)
MBUILD	formation of thoughts
PROPEL	applying physical force
INGEST	खाना-पीना
EXPEL	बाहर निकालना
MOVE	body part movement
SPEAK	बोलना

2. Conceptual Roles

ये बताते हैं कि action में कौन क्या कर रहा है।

कुछ major roles:

- **Agent** – Action कौन कर रहा है
- **Object** – किस पर action हुआ
- **Recipient** – किसे benefit मिला
- **Instrument** – action कैसे किया
- **Direction / Location** – कहाँ हुआ

CD Representation Example (Detailed)

Sentence:
“Ram gave a book to Sita.”

Conceptual Dependency Representation:

- Action: **ATRANS** (transfer of possession)
- Agent: **Ram**
- Object: **Book**
- Recipient: **Sita**

CD diagrammatically:

ATRANS

Agent: Ram

Object: Book

From: Ram

To: Sita

इसका मतलब actual language matter नहीं करती।
अगर यह sentence हिंदी में होता:

"राम ने सीता को एक किताब दी।"

तो भी CD representation **same** ही रहेगा।

Another Example:

Sentence:
“He hit the ball.”

CD representation:

- Action: **PROPEL** (force applied)
- Agent: **He**
- Object: **Ball**
- Instrument: (माना कि हाथ से मारा)

PROPEL

Agent: He

Object: Ball

Instrument: Hand

Why CD is Important?

- Natural language समझने के लिए structured approach देता है
- Machine Translation, Question Answering, Chatbots में उपयोगी
- Human cognitive model जैसा structure बनाता है
- Inferences निकालना easy बनाता है
  (जैसे अगर किसी ने खाना खाया, तो वह पहले खाना अपने पास रखता था → ATRANS inference)

Advantages

- Language independent
- Deep semantic understanding
- Powerful inference support

Limitations

- Representation complex हो जाती है
- बड़ी knowledge-base maintain करना मुश्किल
- Natural language की सभी complexities को fully cover नहीं करता

Semantic Net in AI

Semantic Network (Semantic Net) Artificial Intelligence में उपयोग होने वाला एक **knowledge representation technique** है, जिसमें **objects, concepts, events** और उनके बीच के **relationships** को एक **graph structure** के रूप में represent किया जाता है। इस representation में **nodes** concepts को दिखाते हैं और **links/edges** उनके बीच के relationships को।

Semantic Net का उद्देश्य है कि AI systems किसी भी knowledge को **structured, connected, और meaningful form** में store करके logically **reasoning**, **inference**, और **knowledge retrieval** कर सकें।

Semantic Net को सबसे पहले **Ross Quillian (1966)** ने human memory representation को model करने के लिए प्रस्तावित किया था। Semantic Net एक associative network बनाता है जो human brain की तरह interconnected knowledge store करता है, यानी information isolated नहीं रहती बल्कि concept से concept linked रहती है।

Semantic Net की प्रमुख विशेषताएँ (Key Characteristics)

1. Graph-based Structure

- Node = Concept/Object/Event
- Edge/Link = Relationship between nodes
  इससे knowledge visually समझना बहुत आसान हो जाता है।

2. Inheritance Mechanism

Semantic Net **property inheritance** को support करता है।
Example:
अगर "Bird" node के पास property है **can fly**,
तो उसके child node "Sparrow" automatic वही property inherit करेगा।

3. Expressive Representation

Semantic Nets conceptual knowledge, object relationships, category hierarchy और actions तक represent कर सकते हैं।

4. Easy Visualization

Graph-style representation होने से complex knowledge भी आसानी से समझ आता है।

5. Supports Inference

Machine किसी भी node से linked nodes को trace करके reasoning कर सकती है।

Semantic Network का Structure

Semantic Net दो मुख्य components पर आधारित होता है:

1. Nodes

ये represent करते हैं:

- Objects (e.g., Dog, Car)
- Concepts (e.g., Animal, Vehicle)
- Attributes (e.g., Color, Size)
- Events (e.g., Running, Eating)

2. Links (Relationships)

Important relationship types:

Relationship	Meaning
IS-A	Category relationship
PART-OF	किसी चीज़ का हिस्सा होना
HAS-A	possession
INSTANCE-OF	specific example
CAUSE-EFFECT	causality relation
AGENT, OBJECT, LOCATION	action-based relations

Semantic Net Example

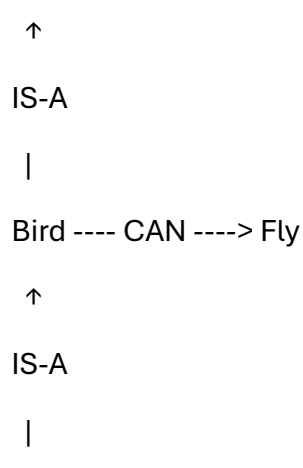
Sentence:
"A robin is a bird. A bird is an animal. Birds can fly."

Semantic Net representation:

- Robin → **IS-A** → Bird
- Bird → **IS-A** → Animal

- Bird → **CAN** → Fly

Graphical form:



Inference Example

इस network से AI खुद infer कर सकती है:

- Robin is an animal
- Robin can fly

क्योंकि Robin ने Bird की properties inherit की।

Semantic Nets के Applications

1. **NLP (Natural Language Processing)**
   Meaning extraction और concept mapping में उपयोग होता है।
2. **Expert Systems**
   Medical diagnosis, legal reasoning जैसे domains में knowledge representation के लिए।
3. **Machine Reasoning**
   Nodes और relations को trace करके logical inference निकाला जाता है।
4. **Semantic Web & Ontology Design**
   Knowledge graph, linked data, RDF आदि में इस्तेमाल होता है।
5. **Cognitive Modeling**
   Human memory और concept association को model करने में।

Advantages (लाभ)

- Knowledge visualization आसान
- Property inheritance → कम redundancy
- Strong relationship representation
- Logical inference संभव
- Concept-based searching और reasoning आसान

Limitations (सीमाएँ)

- बड़े networks बहुत complex और confusing हो जाते हैं
- Procedural knowledge represent करना कठिन
- Ambiguous knowledge handle करना मुश्किल
- Inference computation कभी-कभी slow हो जाता है

Scripts in AI

Artificial Intelligence में **Script** एक **structured knowledge representation framework** है, जिसका उपयोग किसी भी **common, everyday situation** में होने वाले **sequence of events** को represent करने के लिए किया जाता है। Scripts का concept सबसे पहले **Roger Schank और Robert Abelson (1977)** ने दिया था, जो उनकी **Conceptual Dependency Theory** का हिस्सा था।

Script वास्तविक दुनिया में "जैसे चीज़ें सामान्य रूप से होती हैं" उनका एक **standardized, organized और stereotyped sequence** provide करता है। Example के तौर पर, जब कोई व्यक्ति restaurant जाता है, तो events का एक fixed pattern होता है – inside जाना, seat लेना, order देना, खाना, bill देना और बाहर आना। Script ऐसे ही sequences को formally represent करती है, जिससे AI systems इन्हें समझ सकें, interpret कर सकें और reasoning कर सकें, भले ही कुछ details input में missing हों।

Scripts AI को **gap-fill करने**, **unstated events infer करने**, और **natural language को context के साथ समझने** में मदद करती हैं। इन्हें हम ऐसे templates कह सकते हैं जो बताते हैं कि किसी भी specific situation में आमतौर पर events कैसे होते हैं।

Scripts की Key Characteristics

1. Predefined Event Sequence

Script के अंदर किसी situation में होने वाले events का **pre-arranged order** define किया जाता है।

2. Slots and Values

Scripts कई **slots** से बने होते हैं, जैसे:

- Entry conditions
- Props (objects involved)
- Roles (people involved)
- Scenes (sub-events)
- Results or outcomes

हर slot के पास defined या default values होती हैं।

3. Predictive Inference

यदि user input में कुछ events missing हों, तो AI Script के आधार पर उन्हें खुद infer कर सकता है।
Example:
अगर story में लिखा है:
"John paid the bill and left the restaurant."
AI समझ सकता है कि उसने खाना भी खाया होगा, भले ही explicitly लिखा न हो।

4. Hierarchical Structure

Scripts में बड़ी situation कई छोटे **scenes** में विभाजित होती है।
Example: Restaurant Script → Ordering Scene → Eating Scene → Payment Scene

5. Domain-Specific

हर अलग situation का अपना script होता है:

- Airport Script
- Classroom Script
- Hospital Script
- Movie Theatre Script
- Doctor Visit Script

Components of a Script

1. Entry Conditions

Script शुरू होने से पहले जो conditions true होनी चाहिए।

2. Props

Objects जो use होते हैं (जैसे menu, bill, table, food आदि)

3. Roles

Actors/People involved:

- Customer
- Waiter
- Cook
- Cashier

4. Scenes

Main script के अंदर sub-events का detailed sequence।

5. Results

Script complete होने के बाद final outcomes।

Restaurant Script Example

Entry Conditions:

- Customer hungry है
- Customer के पास पैसे हैं

Props:

- Menu
- Table
- Food
- Bill

Roles:

- Customer

- Waiter
- Cook
- Cashier

Scenes:

1. Enter करना और seat मिलना
2. Food order करना
3. Food खाना
4. Bill मांगना और payment करना
5. बाहर निकलना

Result:

- Customer अब hungry नहीं है
- Customer के पैसे कम हुए हैं

Scripts AI में क्यों महत्वपूर्ण हैं?

1. **Natural Language Understanding (NLU)**
   AI को sentences के पीछे की real-world meaning समझने में मदद मिलती है।
2. **Inference Generation**
   Missing events को automatically fill कर सकता है।
3. **Ambiguity Removal**
   Context में बैठाकर ambiguous statements को clear करता है।
4. **Knowledge Organization**
   मानव experiences को structured और reusable form में store करता है।
5. **Human Behavior Understanding**
   Cognitive modeling और story understanding में उपयोगी।

Advantages of Scripts

- Common events को efficiently represent करता है
- Missing details को infer करना आसान

- Story/narrative को समझना आसान
- Ambiguous sentences clarify करता है

Limitations of Scripts

- केवल stereotypical events को अच्छी तरह handle कर पाता है
- Unusual या unexpected events handle नहीं कर पाता
- Real-world scenarios infinite हैं – हर एक के लिए script बनाना मुश्किल
- Rigid structure dynamic situations में fail हो सकता है

Propositional Logic in AI

Propositional Logic, जिसे **Propositional Calculus** या **Boolean Logic** भी कहा जाता है, Artificial Intelligence में एक बहुत महत्वपूर्ण logical system है, जिसका उपयोग **facts**, **truth statements**, और **logical reasoning** को represent करने के लिए किया जाता है।

Propositional Logic में basic units को **propositions** कहते हैं – ये simple declarative statements होती हैं जो या तो **TRUE (T)** होती हैं या **FALSE (F)**, लेकिन दोनों कभी नहीं। Propositions को represent करने के लिए **propositional variables** जैसे P, Q, R आदि का उपयोग किया जाता है, और इन्हें logically combine करने के लिए **logical connectives** जैसे AND, OR, NOT, IF–THEN, और IF AND ONLY IF उपयोग किए जाते हैं।

Propositional Logic AI systems को एक **formal, mathematical और syntax-based** method देता है जिससे machine known facts से नए facts derive कर सके। चूँकि यह सिर्फ पूरे statements पर काम करता है, उनके internal structure पर नहीं, इसलिए इसे simple लेकिन powerful logic माना जाता है। यही logic आगे चलकर **Predicate Logic**, **Resolution-based reasoning**, **Expert Systems**, और **Automated Theorem Proving** का आधार बनता है।

Propositional Logic की Key Characteristics

1. Truth-valued Statements

हर proposition का truth value होना चाहिए:

- TRUE (T)
- FALSE (F)

Examples:
“Sky is blue” → True
“4 > 10” → False

2. Propositional Variables

Statements को simplify करके variables से represent किया जाता है, जैसे P, Q, R।

Example:
P = “It is raining.”
Q = “I will carry an umbrella.”

3. Logical Connectives

Connective	Symbol	Meaning
AND	∧	दोनों statements true हों
OR	∨	कम से कम एक true हो
NOT	¬	Negation
IMPLICATION	→	If...then संबंध
BICONDITIONAL	↔	If and only if

Example:
P → Q
“If it is raining, then I will carry an umbrella.”

4. Syntax और Semantics

- **Syntax:** Valid logical expressions कैसे बनेंगी
- **Semantics:** Expressions के truth values कैसे evaluate होंगे

5. Truth Tables

Compound propositions के truth values calculate करने के लिए truth tables का उपयोग होता है।

Inference in Propositional Logic (AI Reasoning)

AI inference rules से नए facts derive करती है, जैसे:

1. Modus Ponens

If (P → Q) and P is true → Q must be true.

2. Modus Tollens

If (P → Q) and Q is false → P is false.

3. Resolution Rule

Automated theorem proving में सबसे ज्यादा उपयोग होता है।

Detailed Example of Propositional Logic

मान लीजिए:
P = "It is sunny."
Q = "We will go to the park."

Statement:
If it is sunny, then we will go to the park.

Logical form:
P → Q

अब यदि P true है, तो **Modus Ponens** के अनुसार:

- P → Q
- P true है
  → इसलिए Q भी true होगा।

यह AI reasoning process है।

Applications of Propositional Logic in AI

1. **Expert Systems**
   IF...THEN rules के लिए (जैसे IF symptoms THEN disease)
2. **Automated Theorem Proving**
3. **Knowledge-Based Systems**
4. **Model Checking**
5. **Digital Circuit Design**
   Boolean expressions का उपयोग logic gates में होता है।
6. **AI Planning और Decision Making**
7. **Game Playing और State-Space Reasoning**

Advantages (लाभ)

- Simple और आसान
- Mathematically precise
- Small problems के लिए efficient
- Advanced logics की foundation
- Truth tables और resolution से automated reasoning संभव

Limitations (सीमाएँ)

- Statements के internal structure represent नहीं कर सकता
  Example: "All humans are mortal" represent नहीं हो सकता।
- Objects के बीच relationships express नहीं कर सकता
- बड़े problems के लिए बहुत सारे variables चाहिए
- Complex world knowledge represent करने में weak
- Quantifiers (∀, ∃) नहीं हैं

Conversion to Clausal Form in AI

Conversion to Clausal Form AI और First-Order Logic में इस्तेमाल होने वाली एक systematic प्रक्रिया है, जिसमें किसी भी complex logical expression को एक standard format **clausal form** में बदला जाता है।
Clausal form एक ऐसी structure होती है जिसमें पूरी formula को **clauses** के set के रूप में लिखा जाता है, और हर clause **disjunction (OR)** of literals होता है।
Literal का मतलब है:

- Atomic proposition (जैसे P(x))
- या उसका negation (जैसे ¬P(x))

यह conversion बहुत जरूरी है क्योंकि AI में resolution, theorem proving, inference, refutation method, logic programming आदि techniques केवल **clausal form** पर ही काम करती हैं। यह प्रक्रिया formula की logical meaning को same रखती है, लेकिन structure को मशीन द्वारा process करने योग्य बना देती है।

Clausal Form की Main Characteristics

किसी भी formula को **clausal form** में तब कहा जाता है जब:

1. पूरी formula **conjunction (AND)** of clauses हो।
2. हर clause **disjunction (OR)** of literals हो।
3. सारे quantifiers Skolemization से हटाए जा चुके हों।
4. Formula में कोई implication (→) या biconditional (↔) न हो।
5. Negation (¬) सिर्फ atomic propositions के आगे हो (NNF form)।
6. Final result एक **set of clauses** के रूप में लिखा जाए।

Steps for Conversion to Clausal Form (Detailed Hindi + English Terms)

Step 1: Eliminate Implications (→) और Biconditionals (↔)

Replace rules:

- $P \rightarrow Q = \neg P \vee Q$
- $P \leftrightarrow Q = (P \rightarrow Q) \wedge (Q \rightarrow P)$

Step 2: Move NOT (¬) Inward using De-Morgan's Laws

ये step formula को **Negation Normal Form (NNF)** में लाता है।

Rules:

- ¬∀x P = ∃x ¬P
- ¬∃x P = ∀x ¬P
- ¬(P ∧ Q) = ¬P ∨ ¬Q
- ¬(P ∨ Q) = ¬P ∧ ¬Q

Step 3: Standardize Variables

सभी quantifiers को unique variables दें ताकि confusion न हो।

Example:

∀x P(x) ∧ ∀x Q(x)

बनता है

∀x P(x) ∧ ∀y Q(y)

Step 4: Skolemization (Remove Existential Quantifiers)

Existential quantifiers (∃) को **Skolem constants** या **Skolem functions** से replace किया जाता है।

- यदि ∃ किसी universal variable पर depend नहीं करता → **Skolem constant**
- यदि depend करता है → **Skolem function**

Example:

∀x ∃y Loves(x, y)

Skolemized form:

∀x Loves(x, f(x))

Step 5: Remove Universal Quantifiers

अब formula में केवल universal quantifiers रहते हैं जिन्हें हम drop कर देते हैं क्योंकि clauses में variables implicitly universally quantified होते हैं।

Step 6: Convert to CNF (Conjunctive Normal Form)

Formula को AND of ORs की form में बदला जाता है।
जैसे:

$$P \vee (Q \wedge R)$$

convert होता है:

(P ∨ Q) ∧ (P ∨ R)

Step 7: Split into Clauses

जो CNF मिला है, उसे अलग-अलग clauses के रूप में लिखते हैं।

Complete Worked Example (Hindi + English Terms)

Convert करें:

Given:

∀x (P(x) → ∃y Q(x, y))

Step 1: Remove implication

∀x (¬P(x) ∨ ∃y Q(x, y))

Step 2: Negation inward

Already ok.

Step 3: Standardize variables

Already unique.

Step 4: Skolemization

∃y को Skolem function f(x) से replace करेंगे:

∀x (¬P(x) ∨ Q(x, f(x)))

Step 5: Remove universal quantifier

¬P(x) ∨ Q(x, f(x))

Step 6: CNF form

Already in CNF.

Step 7: Final Clausal Form

{ ¬P(x) , Q(x, f(x)) }

Inference Rules in AI

Inference Rules AI में वे formal, logically valid procedures होते हैं जिनकी मदद से पुराने facts (premises) से नए facts (conclusions) derive किए जाते हैं। ये rules यह बताते हैं कि एक reasoning system कौन-से *legal logical steps* का उपयोग करके ऐसे नए statements निकाल सकता है जो logically true हों, **यदि original statements true हैं**।
AI में inference rules इसीलिए जरूरी हैं क्योंकि वे machine को **automated reasoning**, **knowledge expansion**, और **consistent logical conclusions** निकालने की क्षमता देते हैं।

Inference rules का उपयोग निम्न AI systems में किया जाता है:

- Expert Systems
- Knowledge-Based Systems
- Automated Theorem Provers

- Logic Programming (जैसे Prolog)
- Deductive Databases
- Semantic Reasoning Systems

ये rules ensure करते हैं कि AI system केवल logically valid conclusions ही derive करे और contradictions न बने।

Importance of Inference Rules in AI

AI को हमेशा वो माताfacts नहीं दिए जाते जो conclusions के लिए ज़रूरी हों।
इसलिए inference rules help करते हैं:

✓ Automated reasoning में
✓ Incomplete information से भी conclusion निकालने में
✓ Knowledge base को logically consistent रखने में
✓ नए facts generate करने में
✓ Theorem prove करने में
✓ Logical decision-making में

Inference rules को intelligent behavior का **logical backbone** माना जाता है।

Main Types of Inference Rules in AI

1. Modus Ponens (Most Important Rule)

Rule:
यदि "P → Q" true है और P true है, तो Q भी true होगा।

Formula:

P → Q

P

———————

∴ Q

Example:
"It is raining → Road is wet"
"It is raining"
⇒ "Road is wet"

2. Modus Tollens

यदि P → Q और Q false है (¬Q), तो P भी false होगा।

P → Q

¬Q

———————

∴ ¬P

3. Hypothetical Syllogism

दो implications को combine करता है।

P → Q

Q → R

———————

∴ P → R

4. Disjunctive Syllogism

यदि "P ∨ Q" true है और P false है, तो Q true होगा।

P ∨ Q

¬P

———————

∴ Q

5. Addition Rule

P

———————

∴ P ∨ Q

6. Simplification Rule

P ∧ Q

———————

∴ P

7. Conjunction Rule

P

Q

———————

∴ P ∧ Q

8. Resolution Rule (सबसे जरूरी Rule in AI)

Resolution automated theorem proving, clausal form reasoning और Prolog में सबसे ज़्यादा इस्तेमाल होता है।

Definition:
अगर एक clause में कोई literal है और दूसरे clause में उसका negation है, तो दोनों को resolve करके नया clause निकाला जा सकता है।

Formula (Propositional):

(P ∨ A), (¬A ∨ Q)

———————————————

P ∨ Q

Example:

(¬Rain ∨ Wet)

(Rain)

———————————

Wet

यह rule **sound और complete** दोनों है, इसलिए AI में सबसे powerful logical inference rule माना जाता है।

9. Generalized Resolution (Predicate Logic)

यह First-Order Logic के लिए है और इसमें:

- **Unification**
- **Substitution**
- **Variable Standardization**

का प्रयोग होता है।

Example:

¬P(x) ∨ Q(x)

P(a)

———————

Q(a)

10. Universal Instantiation

Universal statement से specific case निकाला जाता है।

∀x P(x)

————————

P(a)

11. Existential Instantiation

Existential statement से एक नया constant introduce किया जाता है।

∃x P(x)

————————

P(c)

Inference rules **logical entailment** के सिद्धांत पर आधारित होते हैं।
Entailment का मतलब:

यदि premises true हैं, तो conclusion हर possible situation में true होगा।

Inference rules नया ज्ञान invent नहीं करते–
ये उस ज्ञान को **derive** करते हैं जो original statements में *implicitly hidden* होता है।

AI का reasoning system इन rules को बार-बार apply करके:

- new conclusions generate करता है
- logical proof बनाता है
- knowledge base expand करता है
- contradictions detect करता है

Inference rules को दो गुणों का पालन करना जरूरी है:

1. Soundness

कभी false conclusion derive न हो, यदि premises true हैं।

2. Completeness

सभी logically true conclusions derive करने की क्षमता हो।

इसलिए **Resolution Rule** AI में सबसे ज्यादा important है क्योंकि यह sound और complete दोनों है।

Example: Multiple Inference Rules Applied

Given:

1. ∀x (Human(x) → Mortal(x))
2. Human(Socrates)

Step 1 (Universal Instantiation):

Human(Socrates) → Mortal(Socrates)

Step 2 (Modus Ponens):
क्योंकि Human(Socrates) true है:

Mortal(Socrates)

Thus, AI system successfully infers that **Socrates is mortal**.