ANNAMACHARYA UNIVERSITY, RAJAMPET
(ESTD UNDER AP PRIVATE UNIVERSITIES (ESTABLISHMENT AND REGULATIONS) ACT, 2016
RAJAMPET, Annamayya District, AP, INDIA

Course : Introduction to Artificial Intelligence

Course Code : **24BCOM35T**

Branch : **B.Sc (AIML, AI&DA, Programming & Data Science)**

Prepared by : **S. Mastan**

Designation : **Assistant Professor**

Department : **MCA**

Title of the Course : Introduction to Artificial Intelligence

Category : PC

Course Code : 24BCOM35T

Branches : **AIML, AI&DA, Programming & Data Science**

Semester : III Semester

Lecture Hours	Tutorial Hours	Practice Hours	Credits
3	0	0	3

Course Objectives: This course aims to provide a comprehensive understanding of the foundational building blocks of Artificial Intelligence (AI) through the concept of intelligent agents. It explores key approaches in AI, including heuristic search, game search, and logical inference, while delving into decision theory and planning processes for agents. Additionally, the course focuses on analyzing various types of objects and scenarios in uncertain environments and equipping learners with the ability to identify solutions using reasoning in the context of uncertain knowledge.

Course Outcomes:

At the end of the course, the student will be able to
1. understand the importance of artificial Intelligence in real world environment
2. apply the artificial intelligence algorithms for problem solving
3. analyze the various reasoning and knowledge representation techniques
4. solve the problems using classification and planning techniques
5. apply knowledge and reasoning techniques in uncertain environment for the obtain solutions

Unit 1-Introduction to Artificial Intelligence **9**

Introduction to AI, History of AI, Emergence of Intelligent Agents, and Intelligent Agents: PEAS- Representation for an Agent, Types of Agents, Types of Agent Environments, Concept of Rational Agent, Structure of Intelligent agents, defining the Problem as a State Space Search, Problem Characteristics.

Unit 2-Problem Solving **9**

Solving problems by searching, Problem Formulation, Uninformed Search Techniques -DFS, BFS, Informed search methods –heuristic Functions, Hill Climbing, A*,Performance Evaluation. Constrained Satisfaction Problems: Constraint Satisfaction Problems like – map Coloring, Back tracking for CSP, Local Search.

Unit 3-Knowledge and Reasoning **9**

A knowledge Based Agent, Introduction to Logic, Propositional Logic, Reasoning in Propositional logic, First Order Logic: Syntax and Semantics, Extensions and Notational Variation, Inference in First Order Logic, Unification, Forward and Back ward chaining, Resolution.

Unit 4-Knowledge Engineering and Planning **9**

Knowledge Engineering: Ontology, Categories and Objects, Mental Events and Objects.
Planning: Planning problem, Planning with State Space Search, Partial Order Planning, Hierarchical Planning, Conditional Planning.

Unit 5-Uncertain Knowledge and Reasoning **9**

Uncertain Knowledge and Reasoning: Uncertainty, Acting Under Uncertainty, Basic Probability Notation, The Axioms of Probability, Inference Using Full Joint Distributions, Independence, Bayes Rule and Its Use.

Prescribed Textbooks:

1. Stuart Russell and Peter Norvig*, Artificial Intelligence: A Modern Approach. 2nd Ed., Pearson Publication.*

Reference Books:

1. George Lugar, *AI-Structures and Strategies for Complex Problem Solving*. 4th Ed., Pearson Education, 2002.
2. RobertJ. Schalkolf, *Artificial Intelligence: an Engineering approach*. McGraw-Hill, 1990
3. Patrickh. Winston, *Artificial Intelligence*. 3rd Ed., Pearson.
4. E. Rich, and K. Knight, *Artificial Intelligence*, Tata Mc Graw-Hill.

Artificial Intelligence

D

UNIT-I

Introduction to AI :-

Artificial Intelligence is a branch of science which deals with helping machines (computers) finding solutions to complex problems in a more Human Like Fashion.

Artificial Intelligence refers to the development of computer systems that can perform tasks that typically acquire human intelligence. These tasks include learning, problem solving, decision making and perception. AI aims to create machines that can mimic human thinking abilities.

AI systems are designed to perform tasks that humans can do, such as understanding language, recognizing images and making decisions. AI has ability to learn from data, identify patterns, and improve its performance.

History of AI:-

AI is not a new word and not a new technology for researchers. This technology is much older than you would imagine.

* Even these are the myths of Mechanical men in Ancient Greek and Egyption Myths.

Following are some milestones in the history of AI.

Maturation of AI:(1943-1952):-

* In 1943 the first work which is now recognized as AI, was done by "Warren Mc Culloch" and "Walter Pitts". They proposed a model of Artificial neurons.
* In 1949 "Donald Hebb" demonstrated an updating rule for modifying the connection strength btn neurons. His rule is now called as "Hebbian Learning".
* In 1950 the "Alan Turing" who was an English mathematician and pioneered Machine learning in 1950.
* Alan Turing publishes "computing Machinery and Intelligence" in which he proposed a test.

* The test can check the machine's ability to exhibit intelligent behavior equivalent to human intelligence called a Turing test.

→ The birth of AI (1952-1956):-

* In the year 1955 "Allen Newell" and "Herbert A. Simon" created the "First AI program", which was named as "Logic Theorist".
* This program had proved 38 of 52 Mathematical theorems, and find new and more elegant proofs for some theorems.
* In 1956 the term AI was coined by an American computer scientist "John McCarthy" at the "Dartmouth conference".

→ The golden years- Early enthusiasm (1956-1974):-

* In 1966 the researchers emphasized developing algorithms, which can solve mathematical problems.
* "Joseph Weizenbaum" created the first chatbot in 1966 which was named as "ELIZA".
* In 1972 the first intelligent humanoid robot was built-in Japan, which was named as "WABOT-1"

→ The first AI winter (1974-1980):-

* AI winter refers to the time period, where computer scientists dealt (behave) with a severe shortage of funding from government for AI researchers.
* During AI winters an interest of publicity an AI was decreased.

→ A boom of AI (1980-1987):-

* After AI winter duration the AI came back with "Expert Systems."
* Expert Systems were programmed, that emulate (try to do something better than others) the decision-making ability of a human expert.

→ The Second AI winter (1987-1993):-

* Again investors and government stopped in funding for AI research, as due to high cost but not efficient result.
* The expert systems such as XCON was very cost effective.

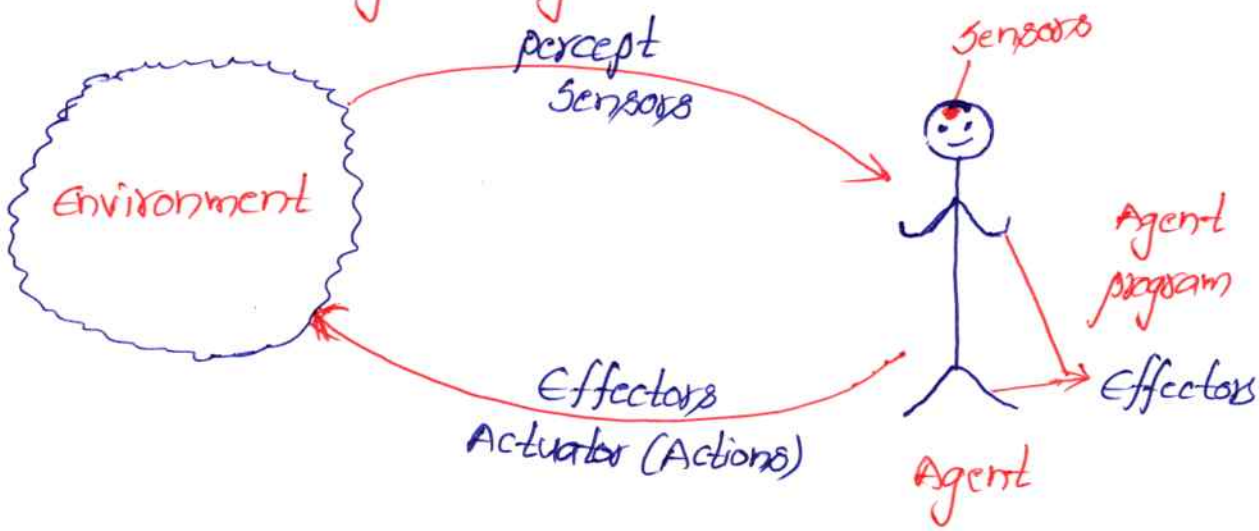
The emergence of Intelligent Agents:-

* In 1997 "IBM Deep Blue" beats world chess champion "Gary Kasparov" and became the first computer to beat a world chess champion.
* In 2002 for the first time AI entered the home in the form of "Roomba", a vacuum cleaner.
* Companies like Facebook, Twitter, and Netflix also started using AI.
* In the year 2011 "IBM's Watson" won Jeopardy, a quiz show where it had to solve the complex questions as well as riddles.
* Watson had proved that it could understand natural language and can solve tricky questions quickly.
* Now AI has developed to a remarkable level.
* Now-a-days companies like Google, Facebook, IBM, and Amazon are working with AI and creating amazing devices.

Emergence of Intelligent Agent:-

Intelligent agents represent a subset of AI systems demonstrating intelligent behavior including adaptive learning, planning and problem-solving. These agents dynamically adjust their behavior learning from past experiences to improve their approach and aiming for accurate solutions.

Intelligent Agent Architecture



* **Environment:-** Environment is the area around the agent that it interacts with. An environment can be anything like a physical space, a room or a virtual space like a game world or Internet.

* **Sensors:-** Sensors are tools that AI agent uses to perceive their environment. They can be any physical like cameras, microphones, temperature sensors

or a software sensor that read data from files.

* Actuators:- Actuators are tools that AI agent uses to interact with their environment through some actions. They can be any physical actuators like wheels, motors, robotic hands, or computer screens or they can be software actuators that send messages.

* Effectors:- Effectors take instructions from decision making mechanism and translates them into actions and these actions are performed through actuators.

PEAS Representation of AI:-

PEAS stands for Performance measure, Environment, Actuators and Sensors. It is a framework that is used to describe an AI agent. It is a structured approach to design and understand AI systems.

* Performance measure:- Performance measure is a criteria that measures the success of the agent. It is used to evaluate how well the agent is achieving its goal. For example in a spam filter systems, the performance measure could be minimizing the no. of spam e-mails reaching the inbox.

* **Environment:-** The environment represents the domain or context in which the agent operates and interact. This can range from physical spaces like rooms to virtual environments such as game worlds or online platforms like internet.

* **Actuators:-** Actuators are the mechanisms through which the AI agent performs actions or interacts with its environment to achieve its goals. These can include physical actuators like motors and robotic hands, as well as digital actuators like computer screens and text-to-speech converters.

* **Sensors:-** Sensors enable the AI agent to gather information from its environment, providing data that informs its decision-making process and actions. These sensors can capture various environmental parameters such as temperature, sound, movement, or visual input. Examples of sensors include cameras, microphones, temperature sensors, and motion sensors.

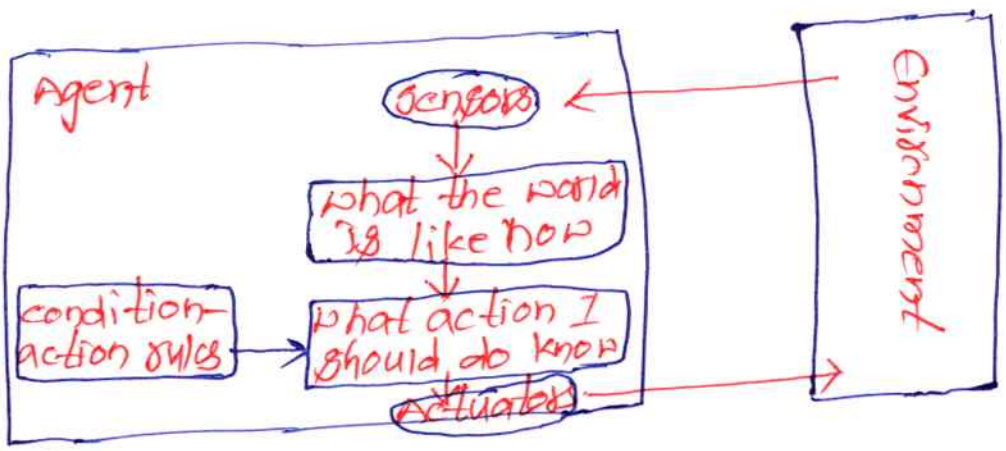
Types of Agents:-

There are five main types of agents.

They are:

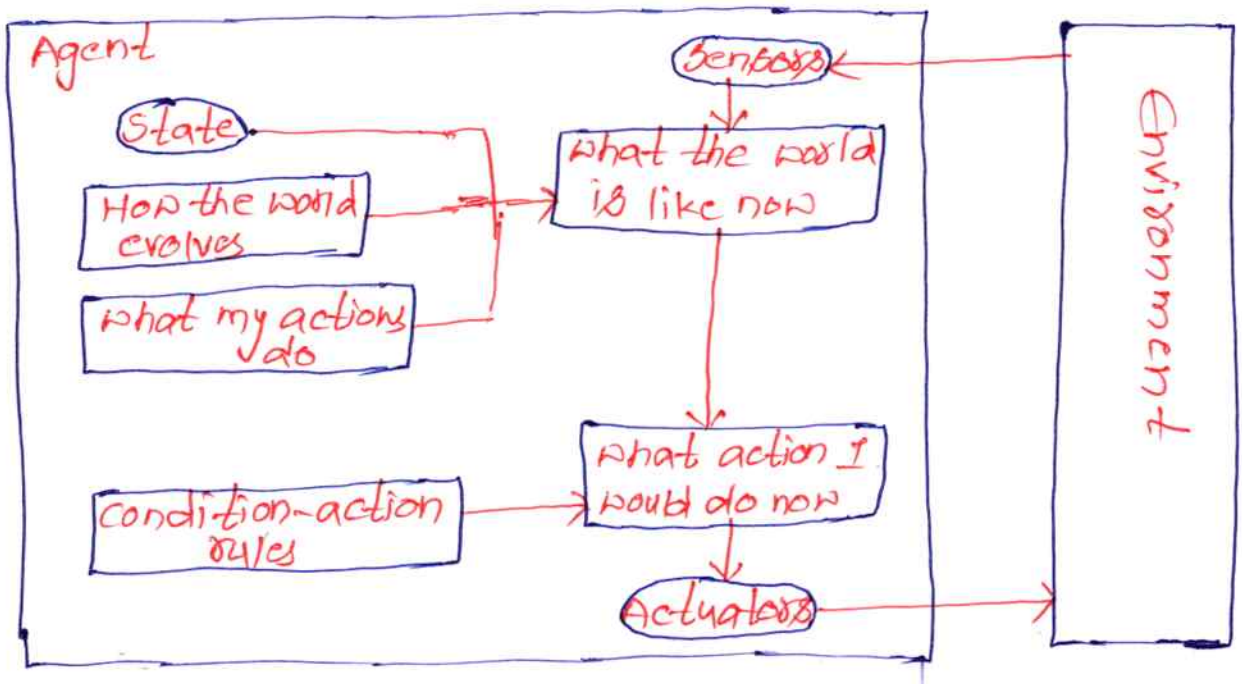
* Simple Reflex Agents
* Model-Based Reflex Agents
* Goal-Based Agents
* Utility-Based Agents
* Learning-Agents.

* Simple Reflex Agents:-



Simple Reflex Agents act based solely on current perceptions using condition-action rules. These agents respond directly without considering any past experiences or potential future states. They operate on basic "if-then" logic. If a specific condition is detected, execute a corresponding action. Ex:- Traffic light control systems.

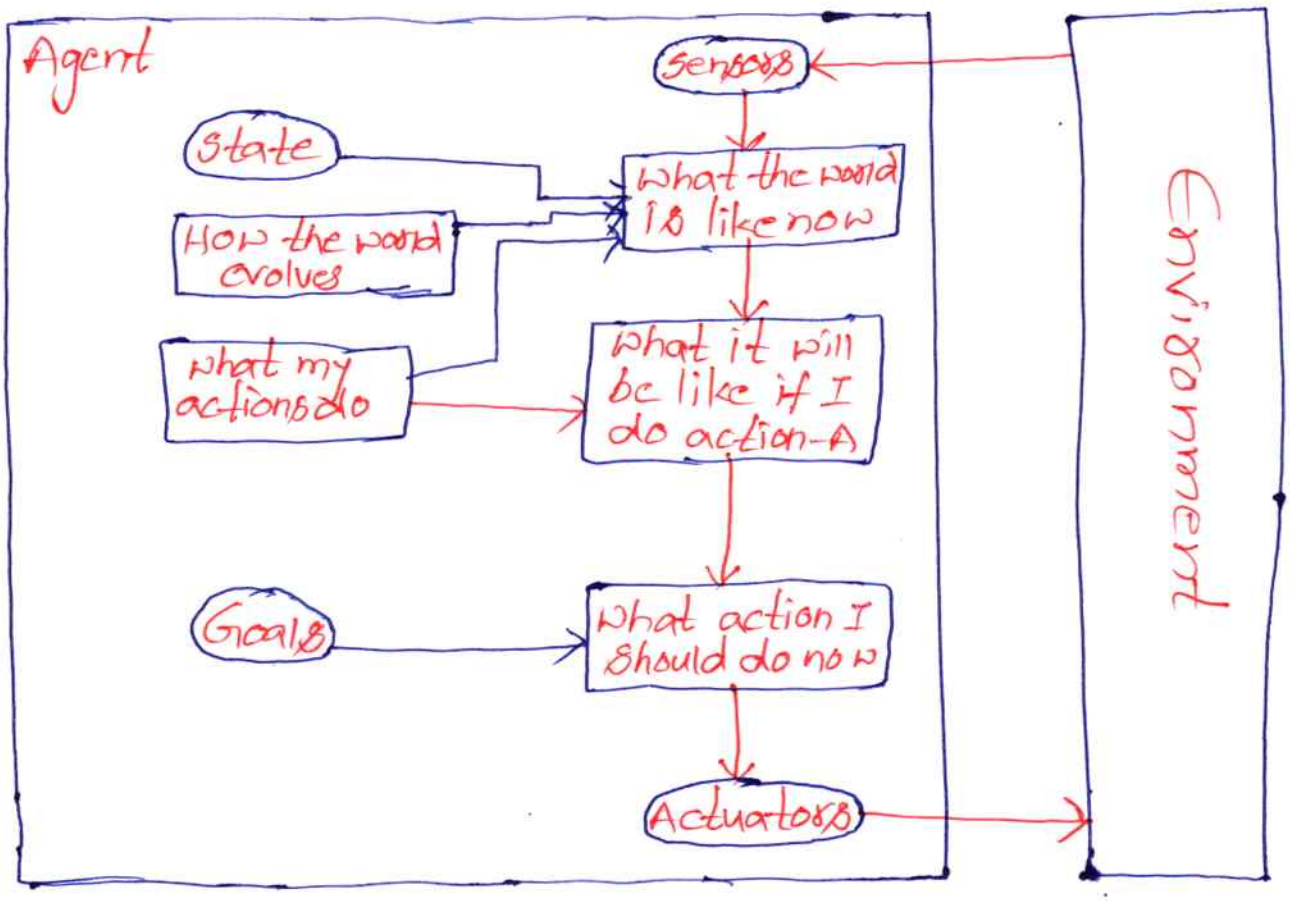
* Model-Based Reflex Agents:-



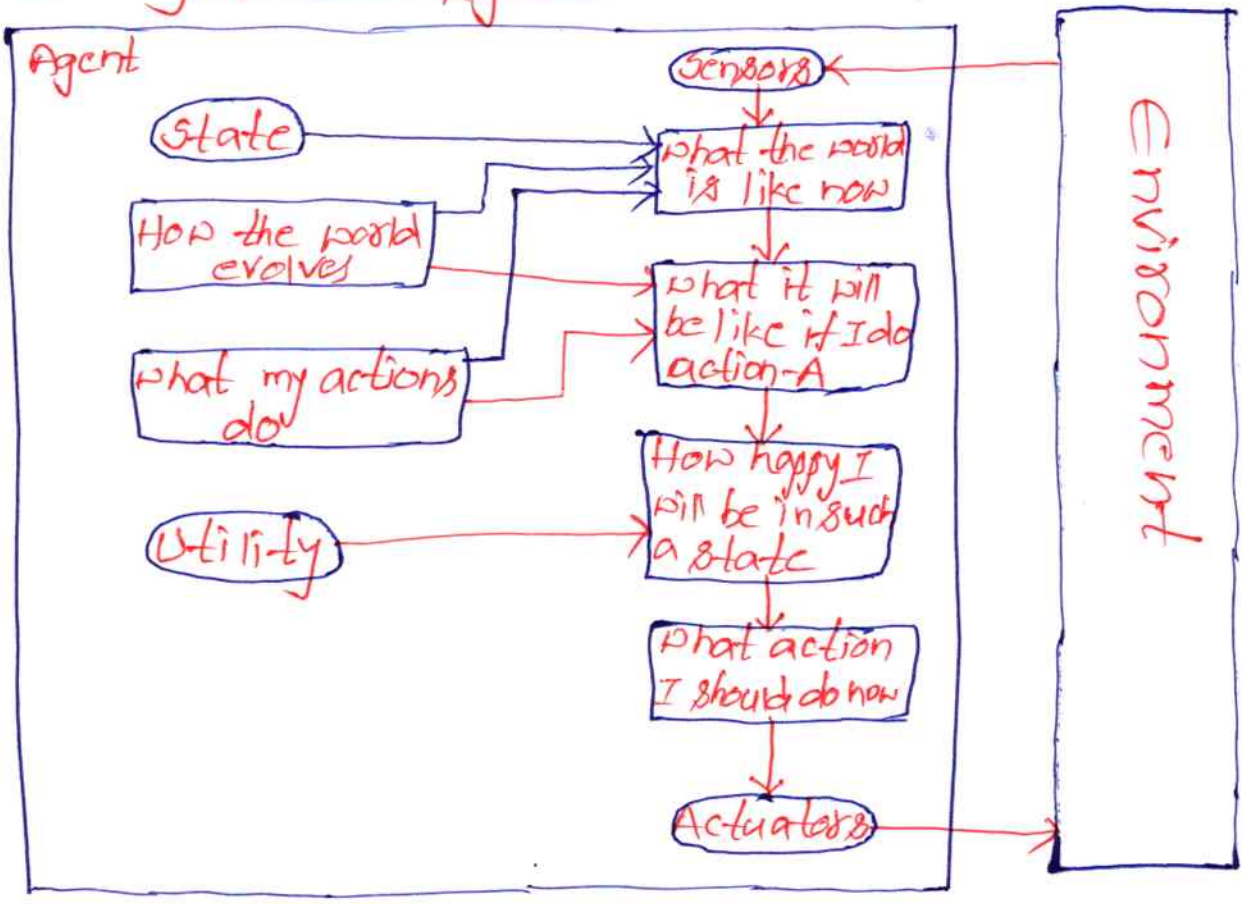
Model-Based Reflex agents maintain an internal representation of the world, allowing them to track aspects of the environment they cannot directly observe. This internal model helps them make more informed decisions by considering how the world evolves and how their accitions affect it. Ex:- Robot vacuum cleaners.

* Goal-Based Agents:-

Goal-Based Agents plan their actions with a specific objective in mind. Unlike Reflex agents that respond to immediate stimuli (something that causes activity), goal-based agents evaluate how different action seavences might lead towards their defined goal, selecting the path that appears most promising.
Ex:- Google maps for finding best route.

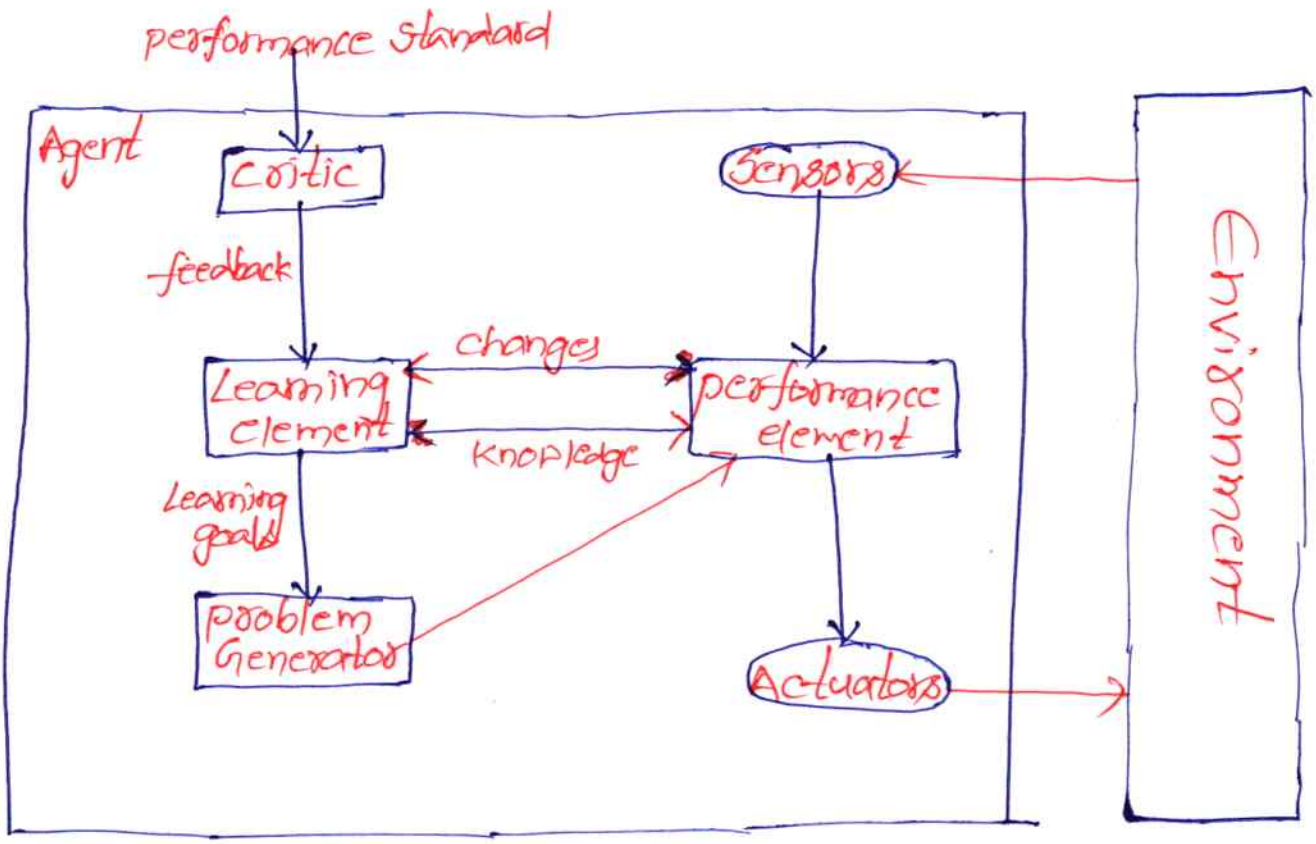


* Utility-Based Agents:-



Utility-Based Agents extend goal-based thinking by evaluating actions based on how well they maximize a utility function - essentially a measure of "happiness" or "satisfaction".

* Learning Agent:-



Learning Agents improve their performance over time based on experiences. They modify their behavior by observing the consequences of their actions, adjusting their internal actions, adjusting their internal models and decision-making approaches to achieve better outcomes in future interactions.

→ Types of Agent Environments:-

Environments in AI can be classified based on how much information the agent can access. Some of the basic types are:

* Fully observable v/s partially observable
* Deterministic v/s stochastic
* Episodic v/s Seavential
* Single-agent v/s Multi-agent
* Static v/s Dynamic
* Discrete v/s continuous

*. **Fully observable:-** The agent has complete and accurate knowledge of the environment's state at any given time.

Ex:- In chess the agent can see all the pieces on the board.

* **partially observable:-** the agent has only access to partial information about the environment. Some aspects of the environment may be hidden or unknown.

Ex:- In poker, players cannot see ~~all the~~ each other's cards, making the game partially observable.

* Deterministic:- In deterministic environments, the outcome of every action is certain. The agent can predict the exact result of any action.

Ex:- In chess every move has a known predictable outcome.

* Stochastic:- In stochastic environments, the outcome of action is uncertain and can vary. The agent cannot predict exactly what will happen after taking an action.

Ex:- In Dice games the outcome of rolling a die is unpredictable.

* Episodic:- In episodic environment the agent's experience is divided into separate episodes. Each episode is independent of the others.

Ex:- A robot performing a series of tasks in different environments.

* Sequential:- In sequential environment the agent requires memory of past actions to determine the next best action. The current decision could affect all future decisions.

Ex:- In chess every move is based on past actions. If a move is made wrong it will affect future actions.

* **Single-Agent:-** In a Single-Agent environment, only one agent interacts with the environment. There are no other agents to compete.

  Ex:- In Maze solving a single robot explores and navigates a maze by itself.

* **Multi-Agent:-** In a Multi-Agent environment, multiple agents interact with each other. These agents can either cooperate or compete.

  Ex:- Multiple players in football interact with one another either as teammates or opponents.

* **Static:-** In a static environment, the state of the environment remains unchanged unless the agent takes an action.

  Ex:- The chess board environment remains the same until a player makes a move.

* **Dynamic:-** In a dynamic environment, the state of the environment can change independently of the agent's actions.

  Ex:- In self-driving cars the traffic predictions and road conditions are constantly changing.

* **Discrete:-** In discrete environment, the no. of possible states and actions ~~is~~ are finite and countable.

  Ex:- In Tic-Tac-Toe game it has a finite no. of moves and board positions.

* **Continuous:-** In continuous environment, the actions performed cannot be numbered i.e it is not discrete, it is said to be continuous.

  Ex:- In Robot Navigation, the robot moving in real-world space can take any position or direction.

➡ Concept of Rational Agent:-

* A rational agent is one that does the right thing.
* what does it mean to do the right thing?
* There should be some criteria that the agent is doing right thing.
* When we put an agent in an environment, it generates a sequence of actions according to the percepts it receives.
* This sequence of actions causes the environment to go through a sequence of states.
* Here sequence of states means starting from the initial state to reach Goal state.
* If the sequence is desirable, then the agent has performed well, if the sequence is not desirable, then the agent has not performed well.

* Here performance measure that evaluates any given sequence of environment states to achieve the desirability.
* There is no any fixed performance measure for all tasks and agents.

Ex:- performance measure for an Automated Taxi Driver include:

→ Getting to the correct destination.
→ Minimizing fuel consumption.
→ Minimizing the trip time and cost.
→ Minimizing violation of traffic laws.
→ Maximizing safety and passenger comfort.
→ Maximizing profits.

* For each possible percept sequence, a rational agent should select an action that is expected to maximize its performance measure, given the evidence provided by the percept sequence and whatever built-in knowledge the agent has.
* Components of Rational agent are:
Sensors, Actuators, performance measure, Agent program, Internal State etc.

Structure of Intelligent agents:-

* The architecture, agent function, and agent programme are the three fundamental components of the AI structure.
* AI's job is to create an agent program that performs the agent function. The architecture and agent software combine to form the framework of an intelligent agent.

Architecture + Agent programme = Agent.

The following are the three most essential terms in the construction of an AI agent.

* Architecture is the equipment on which an AI agent operates, by cameras and other sensors.
* The agent function is used to map a percept to a specific action.
* Agent programme:- An agent programme is a programme that performs the function of an agent. They take the current percept as input from the sensors and return an action to the actuators.

Defining the problem as a State Space Search:-

State Space Search is a method used widely in AI and computer Science to find a solution to a problem by searching through the set of possible states of the problem.

* A state space search algorithm uses the state space to navigate from the initial state to the goal state.
* The first step is to initialize the search by setting the initial state as the current state.
* If current state is not the goal state we generate the set of possible states to reach the goal state.
* If the current state is a goal state, we terminate the algorithm and return the result.

Ex:- A common example of a state space search is the 8-puzzle problem.

The 8-puzzle is a sliding puzzle that consists of 8-numbered tiles in a 3×3 grid and one blank space. The goal is to rearrange the tiles from a given initial state to a final goal state by sliding the tiles into the blank space.

We are taking a practicle example of an 8-puzzle problem. Let us pick current and target states of this example.

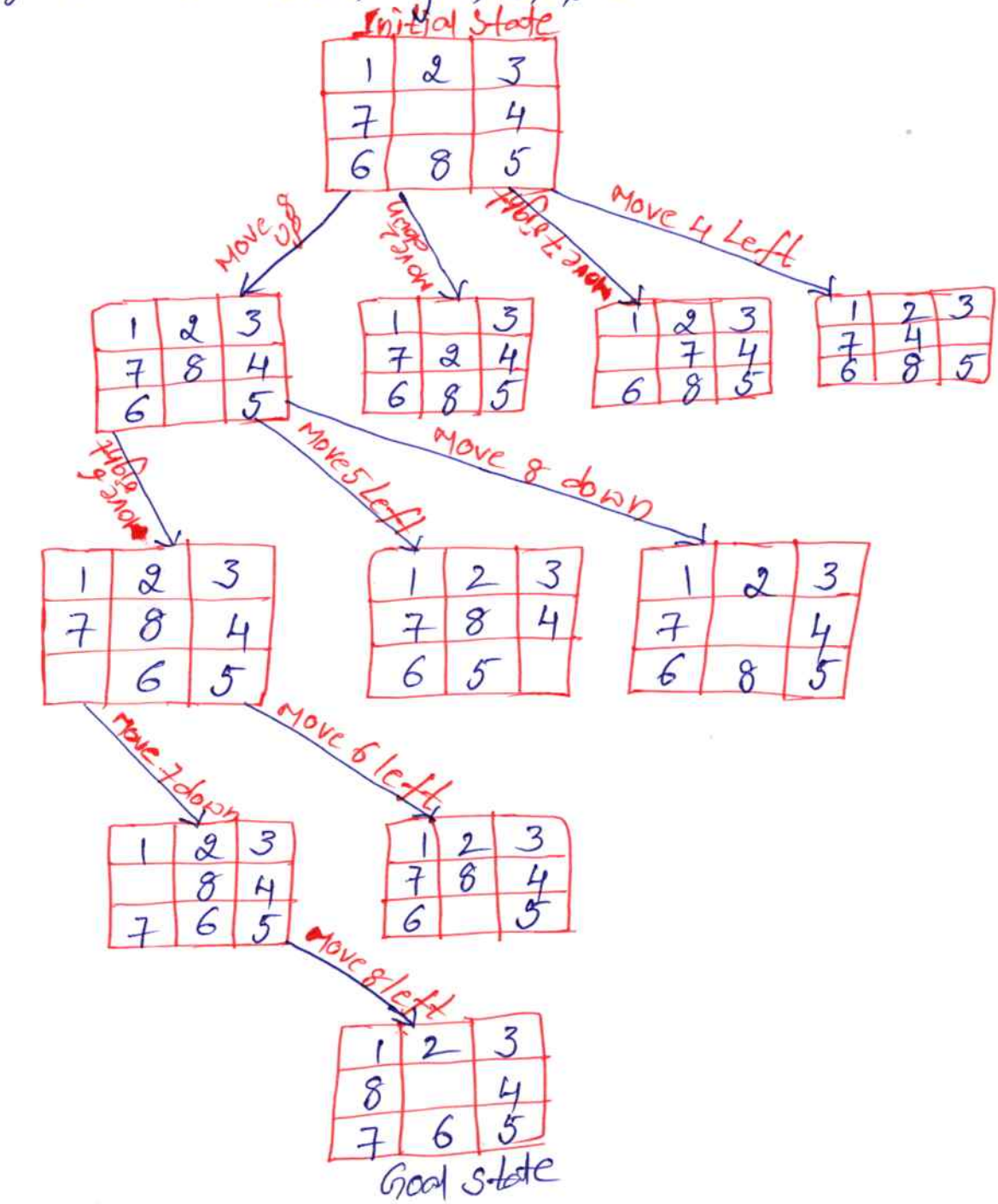
1	2	3
7		4
6	8	5

Initial state

1	2	3
8		4
7	6	5

Goal state.

Here in 8-puzzle problem the actions we can perform are: Left, Right, Up & Down.



→ Problem Characteristics:-

A problem may have different aspects of representation and exploration. In order to choose an appropriate method for a particular problem first we need to categorize the problem based on the following characteristics.

* **Complexity:-** AI problems often involve dealing with complex systems or large amount of data. AI systems must be able to handle this complexity efficiently to produce meaningful results.

* **Learning and adaption:-** AI systems should be capable of learning from data or experiences and adapting their behaviour accordingly. This enables them to improve performance and handle new situations more effectively.

* **Uncertainty:-** AI systems frequently operate in environments where outcomes are uncertain or incomplete information is available. They must be equipped to make decisions or predictions under such conditions.

* **Dynamism:-** Environments in which AI systems operate can change over time. These changes may occur unpredictably or according to specific rules, requiring AI systems to continuously adjust their models.

* Interactivity:- Many AI applications involve interaction with users or other agents. Effective AI systems should be able to perceive, interpret, and respond to these interactions in a meaningful way.

* Context dependence:- The performance of AI systems may depend on the context in which they operate. Understanding and responding to different contexts is essential for achieving desired outcomes.

* Multi-disciplinary:- AI problems often require knowledge and techniques from multiple disciplines, including computer science, mathematics, statistics, psychology and more. These fields are necessary for developing effective AI solutions.

* Goal-oriented Design:- AI systems are designed to achieve specific objectives or goals. Designing AI systems with clear objectives in mind helps guide the development process and ensures that the resulting systems are focused on achieving meaningful outcomes.

Solving Problems by Searching :-

* Problem solving is a process of generating solutions from ~~based~~ observed data.
* Problem is characterized by a set of goals, set of objects, and set of operations.
* The method of solving problem through AI involves the process of
  - → Defining the search space.
  - → Deciding initial state and goal states.
  - → Finding the path from initial state to goal state through search space.
  - → Here search space or problem space is abstract space which has all valid states that can be generated by the combination of operators or objects.
  - → A problem space can have one or more solutions.
  - → Solution is a combination of objects and operations that achieve the goals.
  - → Search refers a search of solution in problem space.

Problem Formulation:-

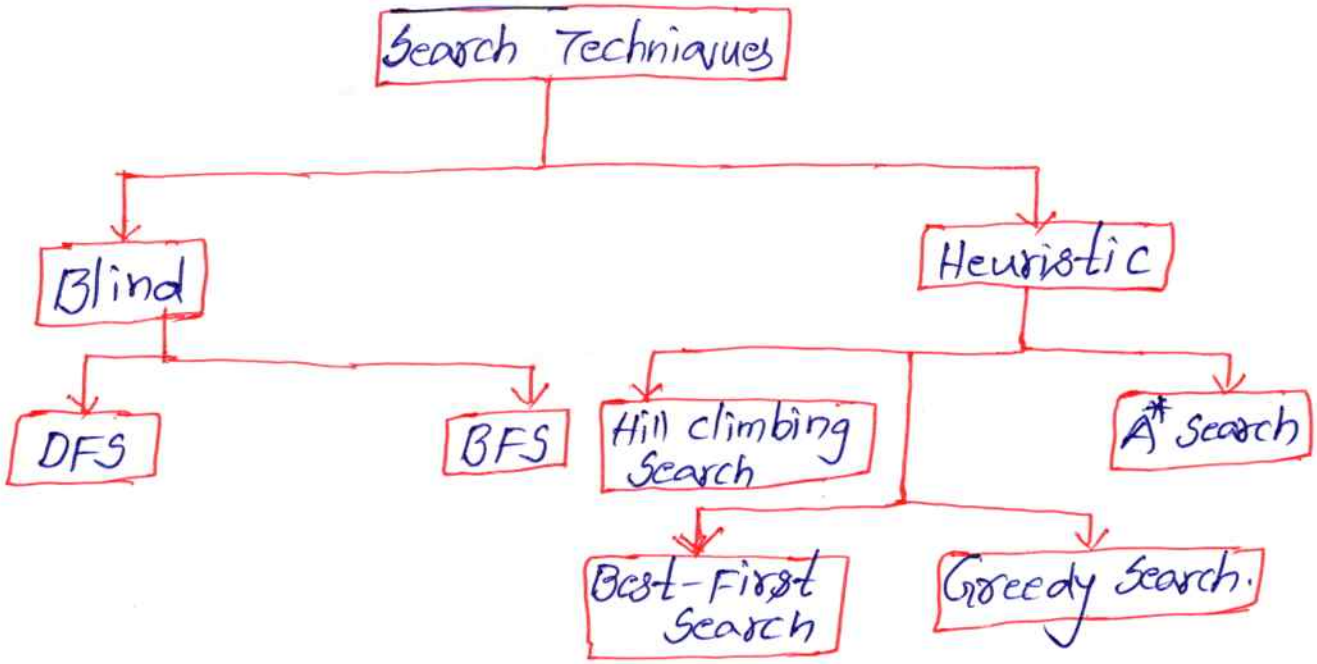
Problem formulation is the crucial process of taking a real-world challenge and turning it into a structured, computational problem that an AI system can handle.

A problem can be defined by 5-components.

1. **Initial state:-** This is the starting point of the problem where the AI begins its process.
2. **Action:-** At this stage AI ~~defines~~ identifies all the possible actions it can take from the initial state. Each action has an impact on how the system moves closer to solving the problem.
3. **Transition:-** This refers to how the system changes from one state to another after an action is taken. Transition modelling helps how the agent's actions influence the next steps.
4. **Goal Test:-** Once an action is taken, AI checks if it has reached its goal. If the goal is achieved, the problem-solving process stops and the solution is considered complete.

5. Cost Function:- This step assigns a numerical value to the cost of achieving the goal. The cost can include resources like time, energy or money and helps decide the most efficient way to reach the goal.

Types of Search Techniques:-



* **Blind/Uninformed Search:-** Blind Search is also called as exhaustive or Brute-Force search uses no information about the problem to guide the search and therefore may not be very efficient. This algorithms are particularly useful in situations where no heuristic information is available. These algorithms are foundational in AI problem-solving and applied in various fields such as pathfinding, puzzle solving and network routing.

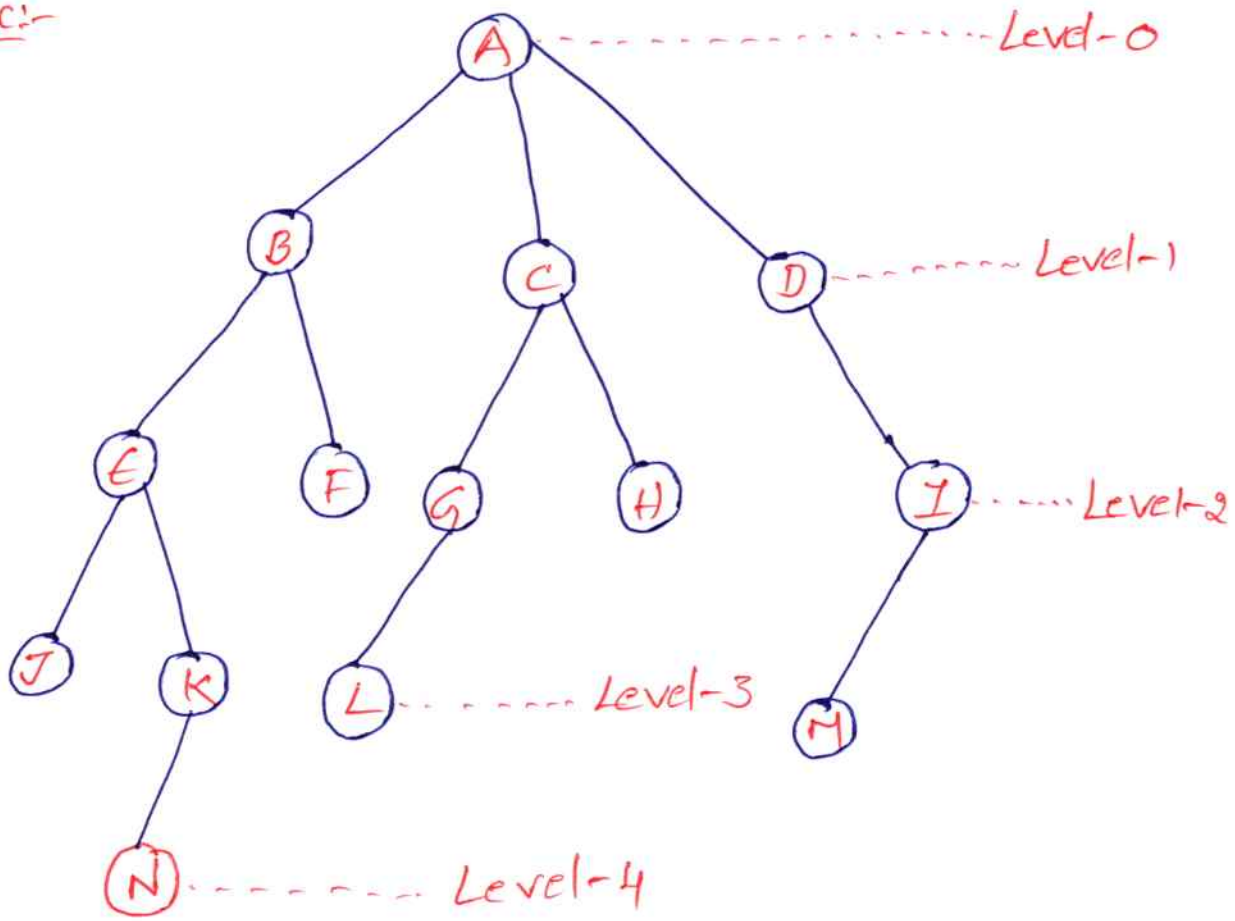
BFS:-

In AI Breadth First search algorithm is an essential search technique for exploring and navigating various problem spaces. The BFS traversal algorithm used to satisfy a given property by searching the tree or graph datastructure.

→ BFS Traversal:-

* BFS starts from a 'start' or 'Root' node and places it into a queue.
* This queue follows the First-In-First-out(FIFO) principle.
* while the queue is not empty, the algorithm dequeues a node.
* It then identifies all unvisited neighbors of the dequeued node.
* Each unvisited neighbor is marked as visited and added to the end of the queue.
* This process continues, expanding nodes level by level until the goal node is found or the queue becomes empty.
* BFS is guaranteed to find a solution if one exists assuming the search space is finite.
* In unweighted graphs, BFS finds the shortest path to the goal node.

Ex:-



A	B	C	D	E	F	G	H	I	J	K	L	M	N	

Queue

Result:- A, B, C, D, E, F, G, H, I, J, K, L, M, N

If Goal node = N then path from root node to goal node is

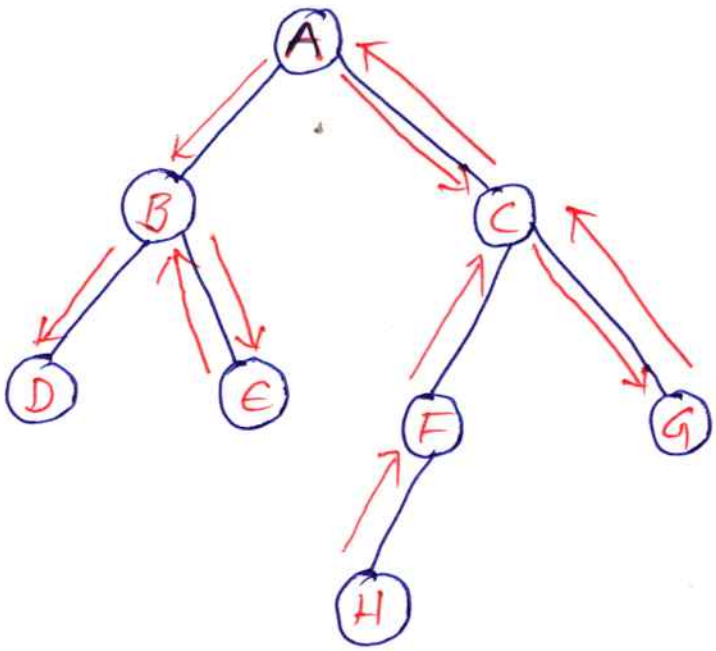
A → B → E → K → N

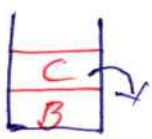
➡ DFS :- DFS in AI is a fundamental graph traversal algorithm used for exploring or searching through graph and tree data structure. Its main principle involves exploring as deeply as possible.

* DFS Traversal :-

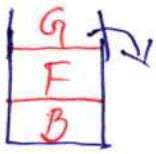
* DFS starts from a "start" or "Root" node and places it into a stack.
* This stack follows the Last-In-First-out (LIFO) principle.
* From the current node it explores one of its unvisited neighbors and then continuous to explore that neighbor's unvisited neighbors, and so on, going as deep as possible along that particular path.
* When it reaches a node with no unvisited neighbors i.e (dead end) or a node that has no successors.
* Then it "backtracks" to the most recently visited node that still has unvisited neighbors.
* The backtracking proccs continuous until all reachable nodes have been visited or a specific goal condition is met.

Ex:-

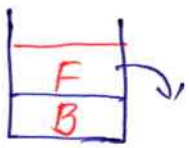




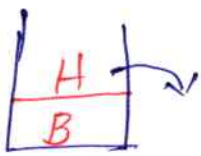
A



AC



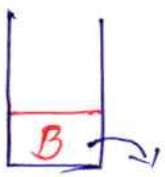
ACG



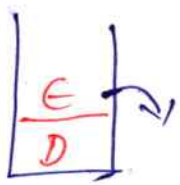
ACGF



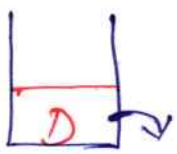
~~ACGFHB~~



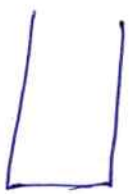
ACGFH



ACGFHB



ACGFHBE



ACGFHBED

Heuristic Functions:-

* A Heuristic function in AI is essential to estimate the cost or distance from a given initial state to a goal state in a search problem.
* Instead of exploring all possibilities heuristic function guide the search by narrowing down the most promising paths.
* This makes problem solving more efficient, especially in complex situations where exact solutions are too costly.
* Heuristic search algorithms uses heuristic functions to make more intelligent decisions during the search process.

* Characteristics and roles:-

* **Estimation:-** A heuristic function provides an estimation, not the exact cost, of reaching the goal. This estimation helps to quickly narrow down the search space.
* **Guidance:-** Heuristics are integral to informed search algorithms like A* search, Greedy search, Best First search etc. This search techniques estimates the cost to determine the order in which to explore nodes in the search space.

* Efficiency:- By guiding the search towards more promising paths, heuristic functions can significantly reduce the computational time and resources required to find a solution especially in large complex problems.

* Admissibility:- An admissible (that can be accepted) heuristic never overestimates the actual cost to reach the goal. This property is crucial for algorithms like A* to guarantee finding an optimal solution.

* Consistency or Monotonicity:- A consistent heuristic satisfies the triangle inequality i.e the estimated cost from a node to the goal is less than or equal to the cost of moving to an adjacent node plus the estimated cost from that adjacent node to the goal.

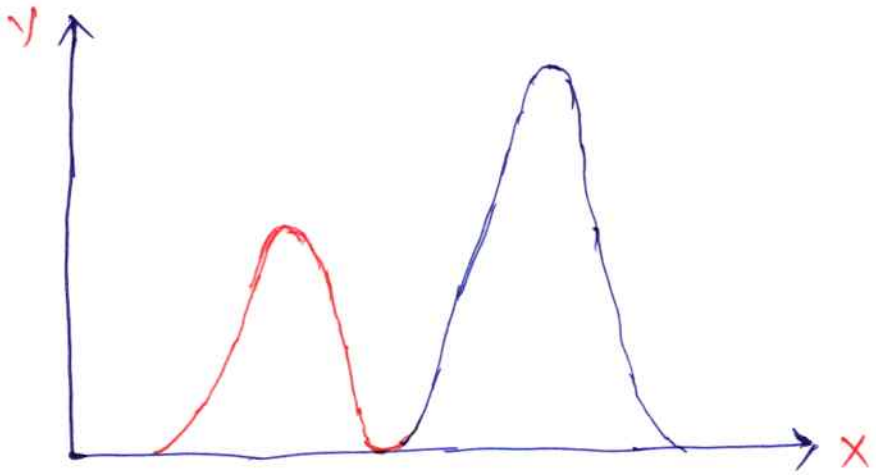
Hill Climbing:-

The Hill climbing algorithm is a local search optimization technique used in AI to find an optimal or near-optimal solution to a problem. It follows Greedy approach, uses Heuristic function and no Backtracking is available.

* The algorithm starts with a randomly generated choosen initial state/current state.
* It then explores the "neighbors" of the current state which are ~~signifi~~ slightly modified versions of the current solution.
* Each neighbor is evaluated using an objective function to determine its "value" or "fitness". The goal is typically to maximize an objective function or minimize a cost function.
* The algorithm moves to the neighbor that offers the best improvement compared to the current state.
* This process of generating, evaluating and selecting neighbors continuous iteratively as long as a better neighbor can be found.
* The algorithm terminates when no neighboring state offers an improvement, indicating that a local optimum has been reached.

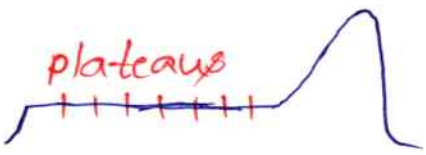
* Problems/Disadv of Hill climbing:-

1. Local Maxima:-



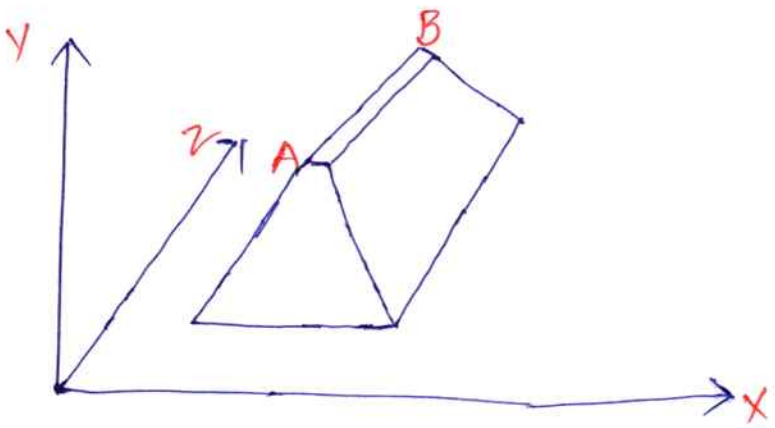
A local maxima is a suboptimal solution that appears better than all its immediate neighbors but it is not the global best. Hill climbing moves upwards and gets stuck here, unable to explore beyond.

2. Plateaus:-



A plateau is a flat region where all neighboring states yield the same objective function value. The algorithm cannot decide which direction to move, leading to stagnation. (the state of being still, or not moving).

3. Ridges:-

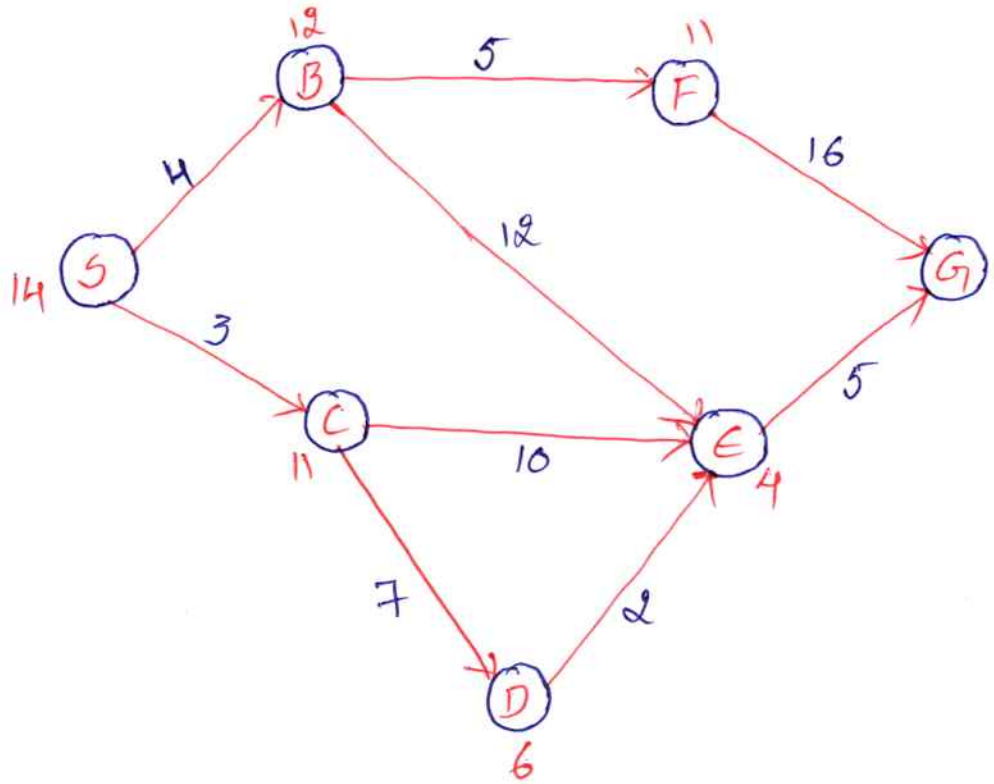


A ridge is a sequence of local maxima that are not directly adjacent. Hill climbing ~~that~~ struggles here because moves are restricted to immediate neighbors making upward progress difficult.

A* Algorithm:-

* The A* Algorithm is a widely used pathfinding and graph traversal algorithm in AI and computer science.
* It is designed to find the shortest path b/n. a starting node and goal node in a graph.
* A* is an informed search algo. i.e it uses a heuristic function to guide its search.
* This heuristic estimates the cost from the current node to goal node.
* A* guarantees finding the optimal shortest path.

Ex:-



$f(N) = g(N) + h(N)$

$g(N)$ = Actual cost from start node to 'N'.

$h(N)$ = Estimated cost from 'N' to goal node.

* Initial state → $f(N) = 0 + 14 = 14$

* $S \rightarrow B$ $\quad$ $S \rightarrow C$

$f(N) = 4 + 12 = 16$ $\quad$ $f(N) = 3 + 11 = 14$ ✓

* $SC \rightarrow E$ $\quad$ $SC \rightarrow D$

$S \rightarrow C \rightarrow E$ $\quad$ $S \rightarrow C \rightarrow D$

$f(N) = 3 + 10 + 4$ $\quad$ $f(N) = 3 + 7 + 6$

$= 17$ $\quad$ $= 16$ ✓

* SCD → E

S → C → D → E

f(N) = 3 + 7 + 2 + 4

= 16 ✓

* SCDE → G

S → C → D → E → G

f(N) = 3 + 7 + 2 + 5 + 0

= 17

* SB → E

S → B → E

f(N) = 4 + 12 + 4

= 20

* SB → F

S → B → F

f(N) = 4 + 5 + 11

= 20 ✓

* SBF → G

S → B → F → G

f(N) = 4 + 5 + 16

f(N) = 25

Performance Evaluation:-

Performance Evaluation in AI refers to the systematic process of assessing an AI model's effectiveness and accuracy using quantative metrics. which can include accuracy, precision, recall and F1-scores etc for different models. choosing the right metric is crucial for performance evaluation. For Classification models, common metrics are:

* **Accuracy:-** Accuracy is the percentage of correct predictions by the model out of the total prediction.
* **Precision:-** The proportion of positive prediction that were actually correct.
* **Recall:-** The proportion of actual positives that were actually correct.
* **F1-Score:-** A balanced measure of precision and recall.
* **Confusion Matrix:-** A table summarizing correct and incorrect predictions to show types of errors.

Constraint Satisfaction Problem: Map coloring

Constraint Satisfaction Problem (CSP) is a mathematical problem where the goal is to find values for a set of variables that satisfy a given set of constraints. These constraints define the permissible combinations of values for the variables. By using CSP we can solve complex problems in AI.

* The goal is to find a complete and consistent assignment where all constraints are satisfied simultaneously.
* CSPs are solved using methods like backtracking search and constraints.
* These techniques are applied to real-world problems such as Sudoku puzzles, Map coloring etc.

Components of CSP:-

* **Variables:-** These are the unknown elements or objects in a problem that need to be assigned values.

Ex:- Variables – A set $\{V_1, V_2, V_3 \ldots V_n\}$

* **Domains:-** Each variable has a domain, which is a predefined set of permissible values that can be assigned to it.

Ex:- Domains – A set $\{D_1, D_2, D_3 \cdots D_n\}$

* **constraints:-** These are the rules or relationships that define which combinations of values for different variables are valid and which are not.

Ex:- Constraints – {Scope, relation}

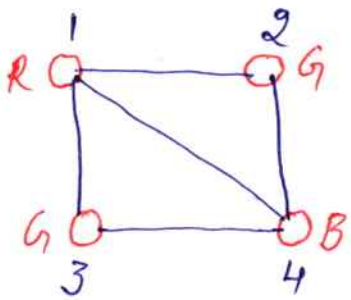
→ Scope is a set of variables that participate in constraint.

→ Relation defines the values that variables can take.

Ex:- Map coloring / Graph coloring

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

$D = \{Red, Green, Blue\}$



constraints $\{1 \neq 2, 1 \neq 3, 1 \neq 4, 2 \neq 4, 3 \neq 4\}$

(or)

constraint = No any adjacent nodes should be colored with the same color.

	1	2	3	4
Initial →	RGB	RGB	RGB	RGB
1=R	R	GB	GB	GB
2=G	R	G	GB	B
3=B	R	G	B	B

Back-track.

	1	2	3	4
Initial	RGB	RGB	RGB	RGB
1=R	R	GB	GB	GB
2=G	R	G	GB	B
3=G	R	G	G	B
4=B	R	G	G	B

∴ 1 = Red

2 = Green

3 = Green

4 = Blue

→ Backtracking Search for CSP:-

* Backtracking search is a fundamental algorithm used to solve CSP in AI.
* It explores the solution space by building a partial assignment of values to variables and checking for consistency with the problem's constraints.
* It operates as a depth First search strategy.

→ Working of Backtracking:-

* **Initialization:-** At the starting state no variables have been assigned a value.
* **Variable selection:-** An unassigned variable is choosen. Heuristic like minimum values are used to select the variable that is most constrained.
* **Value Assignment:-** A value from the chosen variable's domain is assigned to it. Heuristic like Least constraining value can be used to select a value.
* **Consistency check:-** The current partial assignment is checked to ensure it satisfies all relevant constraints. This involves checking if the newly assigned value conflicts with any previously assigned values.

* Recursion:- If the assignment is consistent, the algorithm recursively proceeds to assign a value to the next unassigned variable.

* Backtracking:- If the assignment is not consistent or if further assignments do not lead to a solution, undo the last assignment and try the next possible value.

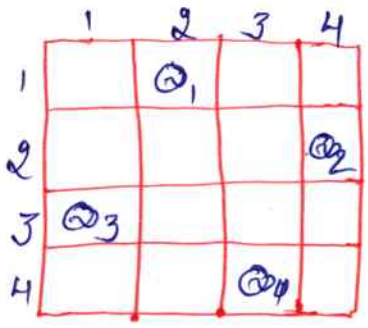
* Termination:- The process continues either a complete and consistent assignment is found, or all possible assignments have been explored without finding a solution, indicating that no solution exist.

Ex:- Four Queen problem

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

$D = \{Q_1, Q_2, Q_3, Q_4\}$

C = First Queen in first row,
Second " " 2nd "
..... Fourth Queen in 4th row.



No any two Queens are in the same row same column and same diagonal.

Local Search:-

* Local Search Algorithm in AI is an optimization technique used to find optimal or near optimal solutions to problems.
* Instead of evaluating every possible solution, these algorithms iteratively explore neighboring solutions in the solution space, making incremental improvements until a satisfactory solution is found.

Basic concepts:-

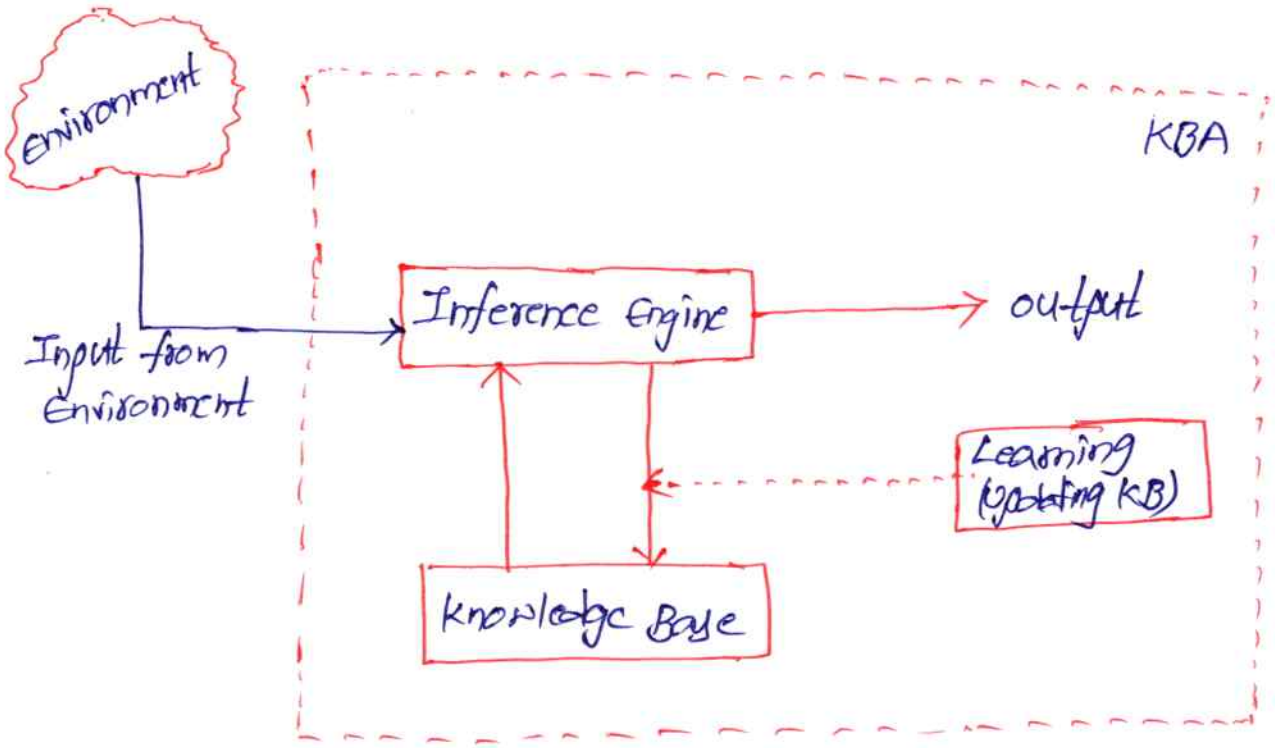
1. Iterative Improvement:- Local search algorithm start with an initial solution and repeatedly move to a better neighboring solution.
2. Neighborhood:- The neighborhood of a solution refers to the set of solutions that are better than the current solution.
3. Compare:- It looks around at all neighbors to see if any better solution is available than the current solution.
4. Move:- If a better neighbor exists, move to it making it our new "current" solution.
5. Heuristic:- A function evaluates the quality of a solution, guiding the search towards better options.

* Local optima:- These algorithms can get stuck in local optima, which are solutions that are better than their immediate neighbors but not necessarily the global optimal solution.

Knowledge-based Agent in AI:-

* A knowledge-based Agent is an AI system that doesn't work on patterns from vast data-sets but they work on a predefined knowledge repository.
* This concept relates back to the early days of AI development.
* We can appreciate their significant role in mimicking human decision-making processes.
* These agents are similar to chess grandmasters, but an extensive memory to make decisions.
* One of the earliest success stories of KBAs was the MYCIN system developed in the 1970's.
* MYCIN helped doctors diagnose bacterial infections by drawing on a stored knowledge base of symptoms and treatments.
* This early break-through demonstrated the potential for AI in critical decision-making areas like healthcare.

* Architecture of a KBA:-



* **Knowledge base:-** The knowledge base serves as the brain of knowledge-base agent. It is filled with information that helps the agent in understanding its environment. However unlike the human brain this knowledge doesn't evolve on its own. Updating the knowledge base is a manual and often labor-intensive task.

* **Inference Engine:-** The inference engine is the agent's component that applies logical rules to the knowledge base to deduce additional information. It essentially acts as a system of logical deduction

that uses the contents of the knowledge base to reach decisions. The inference engine is not always as efficient as one might hope. Especially when dealing with complex problems that requires more computational power.

* Environment Perception:- The agent's operation commences with observing its environment, which could be sensors or other input methods.

* Action determination:- Depending on the output of the Inference engine, the agent decides on the most suitable action. This action is then implemented within the environment.

* Learning from outcomes:- Finally the agent gains insight from the outcomes of its actions. It uses this feedback to modify its knowledge base and hone (perfects) its future decision-making process.

Propositional Logic :-

* Propositional logic is used for solving complex problems using simple statements.
* These statements can be either true or false but cannot be both at same time.
* These propositions form knowledge representation, reasoning and decision-making in AI systems.
* These propositions represents facts or conditions about a situation.
* We use symbols to represent the propositions and logical operations to connect those propositions.
* It helps us understand how different facts are related to each other in complex problems.
* **Building Blocks of Propositional Logic:-**

1. **Atomic Proposition:-** These are basic statements that represent individual facts or conditions. Atomic propositions are simple statements, they cannot be divided further.

   Ex:- P: It is raining.
   Q: The door is closed.

2. (*) In page (7)

* Logical connectives:- connectives are used to combine atomic propositions to form compound propositions. There are five types of Logical connectives.

1. AND (∧) :- True if both propositions are true.

Ex:- The boy promised to buy a pen AND pencil.

P	Q	P∧Q
T	T	T
T	F	F
F	T	F
F	F	F

2. OR (∨) :- True if atleast one proposition is true.

Ex:- The boy promised to buy a pen OR pencil.

P	Q	P∨Q
T	T	T
T	F	T
F	T	T
F	F	F

3. NOT (¬):- Negates the truth value of a proposition.

Ex:- It is not raining.

P	¬P
T	F
F	T

4. IF-THEN(→):- True unless the first is true and second is false.

Ex:- If it rains, then the ground gets wet.

P	Q	P → Q
T	T	T
T	F	F
F	T	T
F	F	T

5. If and only If(↔):- True if both propositions are either true or false.

Ex:- She is happy IF AND ONLY IF she passed the exam.

P	Q	P ↔ Q
T	T	T
T	F	F
F	T	F
F	F	T

(*)
2. Compound Proposition:- A compound proposition is formed by combining two or more atomic propositions using logical connectives like AND, OR, NOT, etc.

→ Inference:-

* Inference in reasoning is the process of reaching a logical conclusion or educated guess based on given information, evidence or premises.
* It involves clues from a text or situation, combined with existing background knowledge.
* Inferences are fundamental to critical reasoning ~~and can be~~

How Inference work:-

* Using clues:- Identifying hints and details within the given information or text.
* Applying knowledge:- Applying our general knowledge and understanding the world to interpret these clues.
* Forming a Logical Step:- Taking a logical step to connect the clues and your knowledge arriving at a conclusion that is not directly stated but is supported by the evidence.

* Types of Inferences:-

1. **Deductive Inference:-** Moves from a general principle or rule to a specific conclusion.

Ex:- All flowers are purple.
All roses are flowers.
∴ All roses are purple.

2. **Inductive Inference:-** Moves from specific instances or evidence to a general principle or conclusion.

Ex:- The sun has risen every day in the past.
Therefore the sun will rise tomorrow.

3. **Abductive Inference:-** Arrives at the most likely explanation from a set of evidences or facts.

Ex:- If you see a person with a bloody nose.
You might infer that they were in a fight.

(**Modus Ponens:-** It states that if a conditional stmt "If P, then Q" is true, and its first part is also true, then the second part must logically be true.
Ex:- If it is raining (P) then the ground is wet (Q).)

Reasoning in Propositional Logic:-

* Reasoning in propositional Logic involves using deductive arguments to determine the truth of conclusions from given premises.
* It formulates reasoning by treating statements as atomic propositions (P, Q) that are either true or false.
* By applying inference rules such as Modus ponens these propositions are combined with Logical connectives (AND, OR, NOT, IMPLIES) to build complex expressions and derive new truths.

* How Reasoning works:-

1. **Representing Knowledge:-** Statements about the world are translated into atomic propositions, represented by letters like P, Q, or R.
2. **Forming Arguments:-** A set of true premises is used to logically infer a conclusion.
3. **Applying Inference Rules:-** Inference rules are applied in a sequence of steps.
   * **Modus Ponens:-** If you have P and you have (P → Q) you can infere Q.

* **Resolution:-** A powerful technique used in automated reasoning to prove a hypothesis from a set of premises.

4. **Valid Arguments:-** A deductive argument is valid if the conclusion is guaranteed to be true whenever all the premises are true.

Ex:- premises-1: P is "It is raining."

premises-2: $P \rightarrow Q$, where Q is "The ground is wet". This means "If it is raining then the ground is wet."

Reasoning:- Using Modus Ponens on premises-1 and premises-2 you can infer Q.

Conclusion: Therefore the ground is wet.

First-Order Logic / Predicate Logic:-

* First-order Logic is also known as Predicate Logic helps us represent knowledge, reason through problems and understand language.
* Propositional Logic deals with complete statements that are either true or false, without breaking them down into objects and their properties.
* FOL extends PL by allowing reasoning about variables and their relationships, making it much more powerful and expressive for complex scenarios.

* FOL plays an essential role in natural language processing, automated reasoning and knowledge representation.

Syntax of FOL and Semantics of FOL:-

* **objects/constants:** These are individual items or entities within a given domain.
  Ex:- People, Numbers, colors etc.
* **Variables:-** Variables are symbols that can take several values a particular domain.
  Ex:- x, y, z etc.
* **Predicate:-** Predicates are statements that describe properties of objects or their relationships. They accept at least one parameter which may be a constant or a variable.
  Ex:- Father(John, Robert).
  Here father is predicate which tells the relation btn. John and Robert.
* **Functions:-** Functions map elements to distinct values. They take arguments in the form of input elements and return a single value as output.
  Ex:- Age(Alice) = 25 ∴ Alice age is 25.
* **Quantifiers:-** These define the scope of variables.
  1. Universal quantifier (∀): Applies a predicate to all elements. Ex:- ∀x (person(x) → mortal(x)) means "All persons are mortal."

2. Existential Quantifier (∃):- Shows the existence of at least one element.

Ex:- ∃x (Person(x) ∧ Likes (x, Ice cream)) means "Someone likes Ice cream".

* Logical connectives:- operators like AND(∧), OR(∨), NOT(¬) IMPLIES (→) and IF AND ONLY IF (↔) are used to build complex sentences.

→ Extensions and Notational Variation:-

* Extensions refer to adding capabilities to existing logical framework such as non-monotic reasoning.
* Notational variations are different ways of representing the same logical concept, like using "∨" or "or" for disjunction.
* These variations are crucial for handling real-world ~~variations~~ uncertainity problems.

Extensions:-

1. Non-monotonic reasoning:- This is an extension that allows for defeasible reasoning, where conclusions can be retracted if new information contradicts them. For example in a default reasoning system if you know a bird can fly, you assume it does, unless you are told it is a Penguin.

2. Default Reasoning:- A common extension where an AI system uses default assumptions to make conclusions based on minimal sets of facts, like circumscribing (restrict within limits) predicates to their minimal true extensions.

* Notational Variations:-

1. propositional Logic connectives:- These are the basic building blocks of logical statements.
   → conjuction (AND):- Represented as AND, V, &. For example Bob owned a car AND color is Blue.
   → Disjunction (OR):- Represented as 'OR' or 'V'. For example PVQ means P OR Q.
   → Negation (¬):- Represented as "NOT" or "¬". For example "NOT" color car Red.
   → Implication (IF):- Represented as "IF" or "→". For example IF Location of Bob Mars (Able Bob Fly)

* Probabilistic Notation:- Used to represent Uncertainity in AI, providing tools for learning from data and making predictions.
   → Quantifiers:- Used to specify the scope of variables.
   * ∀ (forall) Denotes a universal quantifier, meaning every element in a set satisfies a condition.
   * ∃ (there exist): Denotes an existential quantifier, meaning at least one element in a set satisfies a condition.

* Symbolic Notations:- Used in knowledge-based systems to represent facts, beliefs and goals. For instance a fact like "If the temperature is below 18°c, turn on the heater" can be represented using symbolic structures in a knowledge base.

→ Inference in First-Order-Logic:-

* Inference in First-order-Logic is the process of deriving new and valid conclusions from a set of known facts or sentences.
* Unlike Propositional Logic, which can only reason with simple true or false statements, FOL uses predicates, variables and quantifiers to represent more complex knowledge and more powerful inference methods.
* Concepts for FOL inference:- Several key techniques and concepts are used in FOL inference.
  1. Unification:- The core mechanism for finding a substitution that makes two logical expressions identical.
     * Substitution:- It is a fundamental operation performed on terms and formulas. It occurs in all inference systems in FOL.

$F[a/x]$

Substituting a constant 'a' in the place of variable 'x'.

For example if I want to substitute a constant 'b' in place of 'x' then it can be represented as

$F(b/x)$.

This is used to match variables with constants or other variables to apply inference rules such as resolution.

2. ~~Representation~~

* Equality:- In FOL atomic sentences are constructed not only using predicates and terms but also by incorporating equality. Equality symbols are used to indicate that two terms refer to the same object.

Ex:- Brother (John) = Smith.

Here the object refered by Brother (John) is similar to the object refered by Smith. The equality symbol can also be used with negation to represent that two terms are not the same objects.

Ex:- $\neg(x=y)$ which is equivalent to $x \neq y$.

2. Resolution:- Resolution in FOL is a refutation-based (proves that something is false) deductive proof technique used in AI for automated theorem proving. It involves converting logical statements into a simplified form called clausal form, negating the conclusion and repeatedly applying the resolution inference rule to derive a contradiction (the empty clause) thereby proving the original theorem.

3. Skolemization:- The process of removing existential quantifiers ($\exists$) by replacing them with new functions or constants known as Skolem functions or constants. This is crucial preprocessing step for converting FOL sentences into a clausal form suitable for resolution. Ex:- $\exists x$ Doctor(x) ∴ Doctor(d1) d1 represents the specific.

4. Lifting:- A generalization of Propositional inference rules, like Modus Ponens, to operate on FOL sentences with variables. It allows for more efficient reasoning by creating a single, general inference step rather than generating numerous instances.

Ex:- "For all x, If x is a king and x is greedy, then x is evil" ($\forall x$ King(x) ∧ Greedy(x) ⇒ Evil(x)) by unifying the specific instance "King(John) ∧ Greedy(John) to infere "Evil(John).

→ **Inference Methods:-** Inference is performed using different strategies which are the most common being Forward and Backward chaining.

* **Forward chaining:-** Forward chaining is a data-driven inference technique. It starts with the available data and applies rules to infer new data until a goal is reached. This method is commonly used in situations where the initial data set is extensive and the goal is to derive conclusion from it.

Working of Forward Chaining:-

* **Start with known Facts:-** The inference engine begins with the known facts in the knowledge base.
* **Apply Rules:-** It looks for rules whose conditions are satisfied by the known facts.
* **Infer New Facts:-** When a rule is applied, new facts are inferred and added to the knowledge base.
* **Repeat:-** This process is repeated until no more rules can be applied or a specified goal is achieved.

Ex:- Consider a medical diagnosis system where rules are used to diagnose diseases based on symptoms.

* Fact:- The patient has a fever
* Rule:- If a patient has a fever and a rash, they might have measles.

From the above two statements i.e starting with the known fact (fever), the system checks for other symptoms (rash). If the patient also has a rash, the system infers the posibility of measles.

* Backward chaining:- Backward chaining is a goal-driven inference technique. It starts with the goal and works backward to determine which facts must be true to achieve that goal. This method is ideal for situations where the goal is clearly defined and the path to reach it needs to be established.

Working of Backward chaining:-

* Starts with a Goal:- The inference engine begins with the goal or hypothesis (An idea that is possible for explanation but has not yet been found true or correct) it wants to prove.
* Identify Rule:- It looks for rules that could conclude the goal.

* Check conditions:- For each rule, it checks if the conditions are met, which may involve proving additional sub-goals.

* Recursive Process:- This process is recursive, working backward through the rule set until the initial facts are reached or the goal is deemed unattainable.

Ex:- In a troubleshooting system for network issues.

Goal:- Determine why the network is down.

Rule:- If the Router is malfunctioning, the network will be down.

Here the system starts with the goal i.e "network down" and works backward to check if the router is malfunctioning, verifying the necessary conditions to confirm the hypothesis.