05/08

Page No.
Date

Name:- Bhushan Mukund Kor
Class:- D15C Roll NO:- 28
Subject:- AIDS

Assignment No-1

Q.1) What is AI? Considering the COVID-19 pandemic situation, how AI helped to survive and renovated our way of life with different applications?

→ AI (Artificial Intelligence) refers to computer systems that can perform tasks requiring human intelligence, such as learning, reasoning, problem-solving, and decision-making.

AI's Role During COVID-19:-

1) Healthcare & Diagnosis- AI helped detect COVID-19 from X-rays/CT scans and predicted outbreaks.

2) Vaccine Development:- Accelerated drug discovery and vaccine trials.

3) Contact Tracing - AI-powered apps tracked virus spread and alerted users.

4) Remote Work & Education:- AI-driven tools enabled virtual meetings, online learning, and automation.

5) Chatbots & Customer Support:- Automated helplines reduced strain on healthcare and business.

6) Robotics & Automation - AI-powered robots assiste in sanitization, delivery and hospital care.

7) Fake News Detection - AI identified misinformatio and ensured reliable information sharing.

AI transformed work, healthcare, and communicatio making life more efficient during the pandemic.

Q.2) What are AI Agents terminology, explain with examples.

→ AI Agents Terminology:

An AI Agent is a system that perceives its environm and takes actions to achieve a goal.

Key Terminologies:-

1) Agent :- A software or hardware entity that interacts with the environment.
eg. A self-driving car

2) Environment :- The external system with which the agent interacts.
eg. Traffic, pedestrains for a self driving car.

3) perception :- The agent's ability to collect data from sensors.
eg. Cameras & LiDAR in autonomous vehicles.

4) Actuators:- The components that execute actions.
eg. Motors & steering in a robotic vacuum cleaner.

5) Sensors:- Devices that gather information from the environment.
eg. Temperature sensors in smart thermostats.

6) Rationality:- The ability of an agent to take the best possible action.
eg. A chess AI choosing the best move.

7) Autonomy:- The degree to which an agent operates without human intervention.
eg. A robotic assistant that learns user preferences over time.

Q. 3) How AI technique is used to solve 8 puzzle problem?

→ The 8-puzzle problem is solved using AI search techniques like Brute force search and Heuristic search.

AI Techniques for Solving the 8-puzzle Problem:-

1) Uninformed search:- (Brute Force):

- Breadth-first search (BFS) - Explores all possible moves level by level. (Guaranteed solution but slow)
- Depth-first search (DFS) - Explores deep paths first. (Might get stuck in infinite loops).

2) Informed Search (Heuristic search):-

Date

- Best-First search (Greedy Algorithm) - Chooses the best move based on heuristics. (Faster but may not be optimal).

- A search Algorithm :- Uses a heuristic function.
  - Heuristic ($h(n)$) = Number of misplaced tiles.
  - Cost ($g(n)$) = steps taken so far
  - $f(n) = g(n) + h(n)$ (Ensures shortest path)

eg.

A* is Best because it guarantees optimal solution & balances exploration and efficiency.

AI helps solve the 8-puzzle by efficiently searching for the shortest path to reach the goal state.

Q.4) What is PEAS descriptor? Give PEAS descriptor for following:-

1) Taxi Driver
2) Medical diagnosis system
3) A music composer
4) An aircraft autolander
5) An essay evaluator
6) A robotic sentry gun for the Keck Lab.

→ PEAS (Performance measure, Environment, Actuators, sensors) is a framework to define an AI agent's components.

PEAS Descriptors for Given Systems :-

	Agents	Performance	Environment	Actuators	Sensors
1)	Taxi Driver	Safety, speed, fuel efficiency, customer satisfaction.	Roads, traffic, weather, passengers.	Steering, accelerator, brakes, indicators.	GPS, cameras, speed sensor, fuel gauge.
2)	Medical Diagnosis system	Accuracy, diagnosis speed, patient satisfaction.	Patient symptoms, medical reports, lab reports.	Display diagnosis, suggest treatment, notify doctors.	Patient input, medical history, test results.
3)	Music Composer AI	Harmony, creativity, genre accuracy, listener ratings.	Music theory, existing songs, user preferences.	Generates sheet music, audio files, MIDI output.	User feedback, musical databases, instruments.
4)	Aircraft Autolander	Safe landing, smoothness, accuracy, efficiency.	Runway, weather, altitude, wind speed.	Landing gear, brakes, flaps, engine throttle.	Radar, GPS, altimeter, speed sensors.
5)	Essay Evaluator AI	Accuracy, grammar check, coherence, objectivity.	Essays, language rules, grading rubrics.	Score assignment, feedback generation, Plagarism check.	Input text, grammar databases, vocabulary rules.
6)	Robotic Sentry Gun for Kerk Lab	Accuracy, threat detection, response time, safety.	Surrounding area, intruders, authorized personnel.	Rotating turret, firing mechanism, alarm system.	Motion detecting cameras, thermal sensors.

Page No.
Date

Q.5) Categorize a shopping bot for an offline bookstore according to each of the six dimensions (fully/partially observable, deterministic/stochastic, episodic/sequential, static/dynamic, discrete/continuous, single/multi agent)

→ Categorize of a Shopping Bot for an Offline Bookstore

1) Observability : Partially Observable
- The bot may not have full information about book availability, customer preferences, or store layout.

2) Deterministic vs. stochastic : Stochastic
- Customer behavior and stock availability are uncertain, making outcomes unpredictable.

3) Episodic vs. Sequential : Sequential
- The bot's actions (eg. suggesting books, handling purchase) affect future interactions.

4) Static vs. Dynamic : Dynamic
- The bookstore environment changes with customer movements, stock updates, and price adjustments.

5) Discrete Vs Continuous : Discrete
- The bot processes distinct inputs (book selection, payment) but may have continuous interactions like voice recognition.

6) Single vs Multi-Agent :- Multi-Agent

• The bot interacts with multiple customers and bookstore staff, making it a multi-agent system.

This classification helps in designing an efficient AI based shopping assistant.

Q.6) Differentiate Model based & Utility based Agent.

→

Model based Agent	Utility based Agent
1) Uses an internal model of the environment to decide actions.	1) Uses a Utility function to choose the best action based on maximum benefit.
2) Maintains knowledge of how the world works and updates it over time.	2) Evaluates different possible actions and picks the one that maximizes utility.
3) Decision Based on past and current perceptions.	3) Decision Based on a computed utility value.
4) Works towards achieving a goal state.	4) Works towards maximizing long-term performance
5) Eg. A self-driving car using a map to navigate.	5) Eg: A stock trading AI choosing the most profitable investment.

7) Explain the architecture of a knowledge based ag and learning Agent:

→ Knowledge - Based Agent:

Architecture

A Knowledge-Based Agent (KBA) uses knowledge stored in a Knowledge Base (KB) to make decisions

Components:

1) Knowledge Base (KB) - stores facts and rules.
2) Inference Engine - Applies reasoning to derive conclusions.
3) Perception (sensors) - Collects environmental data.
4) Actuators - Executes actions.
5) Learning Module - Updates KB overtime.

Example: An expert medical diagnosis system that suggests treatments based on stored medical knowledge.

Learning Agent Architecture:

A Learning Agent improves its performance over time by learning from past experiences.

components:

1) Learning Element - Learns patterns from past actions.
2) Performance Element - Makes decisions based on learned data.
3) Critic - Evaluates agent performance and give feedback
4) Problem Generator - Suggests new experiences for learning.

Example :- A self-driving car that refines its driving skills through real-world experience.

Q. 8) Convert the following to predicates:
a) Anita travels by car if available otherwise travels by bus.
b) Bus goes via Andheri and Goregaon.
c) car has puncture so is not available.
Will Anita travel via Goregaon? Use forward reasoning.

→ Step 1:- Convert sentences into predicates.

1) Anita's travel options:-
- If a car is available, Anita travels by car.
- If a car is not available, Anita travels by bus.

Predicate form:-
- ~~Travels (Anita, car)~~
- Available (car) → Travels (Anita, car)
- ¬Available (car) → Travels (Anita, Bus)

2) Bus Route Information:
- The bus goes via Andheri and Goregaon.

Predicate form:-
- GoesVia (Bus, Andheri)
- GoesVia (Bus, Goregaon)

3) Car Availability:
- The car has a puncture, so it is not available.

Predicate form:

- Puncture (Car)
- ¬ Available (Car)

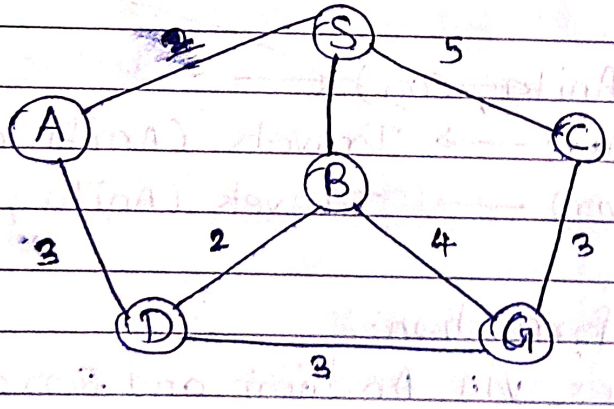
Step 2:- Apply Forward Reasoning.

1) Given ¬ Available (Car), the first rule implies:
   - Travels (Anita, Bus)

2) Since Travels (Anita, Bus) and we know Goes Via (Bus, Goregoan) and Goes Via (Bus, Andheri).

∴ Anita will travel via both Andheri and Goregaon.
And Answer is yes Anita will travel via Goregaon.

Q. 9) Find the route from S to G using BFS.



→ Step 1:- Apply BFS

BFS explores nodes level by level (FIFO order).

Step 1:- Start from 'S' node

Queue = [S]

Explore = {S}

Expand S → Add [A, B, C] to queue

Updated Queue = [A, B, C]

Step 2:- Explored = {S, A}

Expand A → Add [D] to queue

updated Queue : [B, C, D]

Step 3:- Explored = {S, A, B}

Expand B → Add [G] to queue.

(Goal found)

updated Queue = [C, D, G]

Since we have got our goal no need for further exploration.

Path we can get by backtrack.

G came from B.

B came from S.

Path :- S → B → G

Page No.
Date

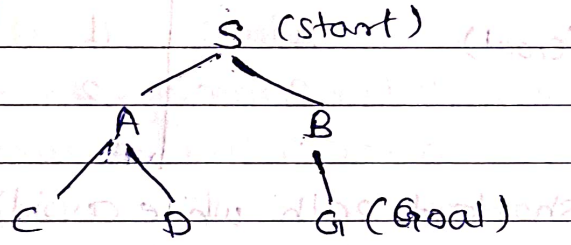
Q.10) What do you mean by depth limited search? Explain Iterative Deepening search with example.

→ **Depth-Limited Search (DLS)**

Definition:

- Depth-Limited Search (DLS) is a modified Depth-First search (DFS) with a predefined depth limits.
- It avoids infinite loops in graphs with cycles and saves memory.
- If a goal is not found within the limit, it fails.

eg.



If depth = 1
Depth 0 : S
Depth 1 : A, B
we cannot find goal
but If depth = 2
Depth 2 : C, D, G
with depth limit = 2, we can find G.

Time complexity = $O(b^l)$ where b = branching factor & l = depth limit.

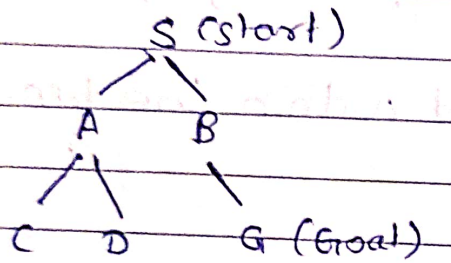
Space complexity = $O(l)$ (Better than BFS)

Iterative Deepening Search (IDS)

Definition:

- IDS is a combination of Depth-First search (DFS) and Breadth-First search (BFS).
- It runs DFS repeatedly, increasing the depth limit one step at a time.
- It finds the shortest path like BFS but uses less memory.

eg.



Iteration	Depth Limit	Nodes Explored
1	0	S
2	1	S, A, B
3	2	S, A, B, C, D, G

IDS finds the shortest path while avoiding deep infinite loops.

Time complexity = $O(b^d)$ where b = branching factor & d = depth of the goal

Space complexity = $O(b^d)$

11) Explain Hill climbing and its drawbacks in detail with example. Also state limitations of steepest-ascent hill climbing.

→ **Hill Climbing Algorithm:-**

Hill Climbing is a heuristic search algorithm that continuously moves toward higher valued states (better solutions) until it reaches a peak. It is commonly used in optimization problems.

Types of Hill Climbing:

1) Simple Hill Climbing
2) Steepest-Ascent Hill Climbing
3) Stochastic Hill Climbing

Drawbacks of Hill Climbing:-

1) Local Maxima
- The algorithm may get stuck at a peak that is not the global maximum.
- Eg. In a landscape with multiple peaks, the algorithm might stop at a lower peak.

2) Plateaus
- A flat area where all neighboring states have the same value.
- The algorithm cannot decide where to move next.

eg. In an 8-Queens problem, the heuristic value might not change even after moving a queen.

3) Ridges
- A path of increasing values that cannot be climbed because moves are limited.
- Eg. If an algorithm can move only in 4 directions but the optimal path requires diagonal moves, it may get stuck.

4) Lack of Backtracking
- Once it moves forward, it does not revisit past states.
- Solution: Random restarts or Simulated Annealing.

Limitations of steepest-Ascent Hill Climbing :-

- Computationally expensive: It checks all possible moves before choosing the best one.
- Still suffers from local maxima and plateaus.
- May oscillate between states if two equally good options exist.

Q. 12) Explain simulated annealing and write its algorithm.

→ Simulated Annealing (SA) :-

Simulated Annealing is a metaheuristic optimization algorithm that helps in escaping local maxima by sometimes accepting worse solutions to explore a better global solution.

It is inspired by the annealing process in metallurgy, where metals are heated and cooled slowly to reach a stable structure.

Algorithm:-

1) Initialize the solution and set the initial temperature
2) Repeat until temperature is low:
   - Generate a new neighbor state.
   - Calculate ΔE (change in cost function).
   - If $\Delta E > 0$, accept the new state.
   - Else, accept with probability $P = e^{(\frac{\Delta E}{T})}$
   - Reduce temperature using a cooling function.
3) Return the best solution found.

Q.13) Explain A* Algorithm with an example.

→ A* is an informed search algorithm that finds the shortest path from a start node to a goal node using a combination of cost(g) and heuristics (h).

Formula:- $f(n) = g(n) + h(n)$

g(n):- Cost from the start node to node n.
h(n):- Estimated cost from node n to the goal (heuristic)
f(n):- Total estimated cost (priority for selection)

Algorithm:-

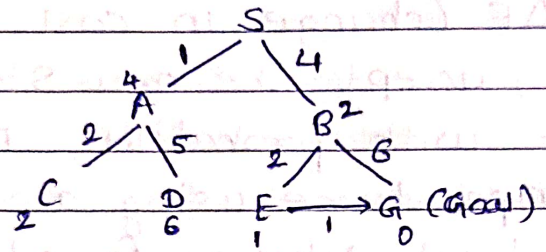
1) Initialize: Add the start node to the open list.
2) Loop until Goal is found:
   - Pick the node with the lowest f(n) from the open list.
   - Move it to the closed list (visited).
   - Generate neighboring nodes.

Date

- Calculate their $f(n) = g(n) + h(n)$.
- If a better path is found, update $g(n)$.

3) Repeat until the goal is reached or open list is empty.

Example :-



Step-by-step Execution

1) Start at S, $f(S) = g(S) + h(S) = 0 +$ heuristic
2) Expand S, select A ($f = 1+4 = 5$) and B ($f = 4+2 = 6$)
3) Expand A, check C ($f = 3+2 = 5$) and D ($f = 6+6 = 12$).
4) Expand B, check E ($f = 6+1 = 7$) and G ($f = 10+0 = 10$).
5) Expand E, finds G with $f = 7+0 = 7$

Shortest path found: $S \rightarrow B \rightarrow E \rightarrow G$
cost = 7.

14) Explain Min max. Explain Minimax Algorithm and draw game tree for Tic TAC Toe game.

→ Minimax is a decision-making algorithm used in two-player games like Tic-Tac-Toe, Chess, and checkers. It assumes that:

- One player (maximizer) tries to maximize the score.
- The other player (minimizer) tries to minimize the score.

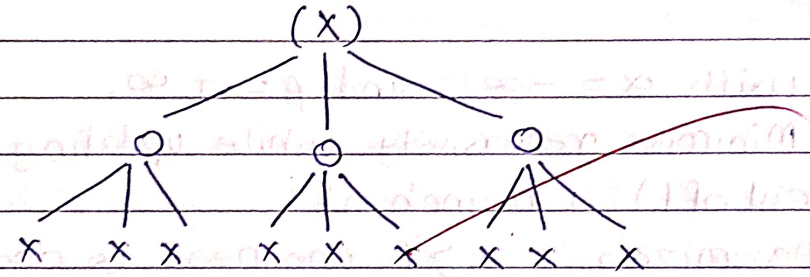
Page No.
Date

Algorithm steps:-

1) Generate the game tree up to a certain depth.
2) Evaluate terminal nodes using a heuristic function.
3) Backpropagate scores:
   - Maximizer selects the maximum score.
   - Minimizer selects the minimum score.
4) Repeat until the best move is found.

Minimax in Tic-Tac-Toe:-

Game Tree (X starts first)



- Maximizer (X) tries to win (score = +1).
- Minimizer (O) tries to block or win (score = -1).
- Draw = 0.

Q. 15) Explain Alpha beta pruning algorithms for adversarial search with example.

→ ## Alpha-Beta Pruning Algorithm:-

Alpha-Beta Pruning is an optimization technique for the Minimax algorithm in adversarial search. It skips unnecessary branches in the game tree, improving

efficiency without affecting the result.

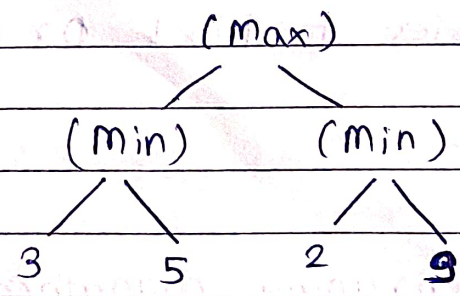
Key concepts:-

- Alpha (α) : Best maximum score for the maximizer (initialized as $-\infty$).
- Beta (β) : Best minimum score for the minimizer (initialized as $+\infty$).
- Pruning : Stops evaluating a branch if it's worse than the current best option.

Algorithm

1) Start with $\alpha = -\infty$ and $\beta = +\infty$.
2) Apply Minimax recursively while updating α and β.
3) Prune (cut off) a branch if:
   - For Maximizer : $\alpha \geq \beta$ (no need to check further).
   - For Minimizer : $\beta \leq \alpha$ (no need to check further).
4) Continue until the best move is found.

E.g.



Maximizer (X) wants max value.
Minimizer (O) wants min value.

1) Left branch : Min(3,5) = 3, so $\beta = 3$
2) Right branch : First value 2 → $\beta = 2$ (less than α)
3) Prune node (9) as it's unnecessary.

Final choice: Max (3,2) = 3
Skipped evaluation of (9), making it faster.

Page No.
Date

Q. 16) Explain WUMPUS world environment giving its PEAS description. Explain how percept sequence is generated?

→ Wumpus World is a grid-based environment used to demonstrate AI agents working in an unknown, partially observable world. The agent navigates a cave with hazards while searching for gold.

PEAS for WUMPUS:-

Performance Measure :- +1000 for finding gold, -1000 for falling into a pit or encountering wumpus, -100 per move to encourage efficiency.

Environment :- Grid based cave with pits, wumpus, gold, walls and agent's starting position.

Actuators :- Move forward, turn left/right, grab gold, shoot arrow (to kill wumpus), climb out.

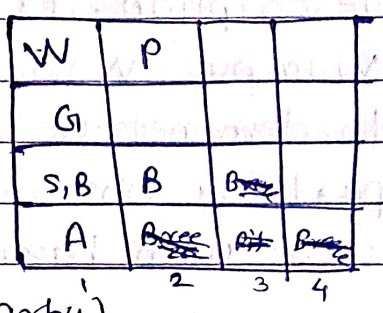
Sensors :- Stench (near wumpus), Breeze (near pits), Glitter (gold present), Bump (wall hit), Scream (wumpus killed).

Percept Sequence:-

- Percepts are generated based on the agent's current location.
- Example: If the agent is next to a Wumpus, it perceives a stench.

• The percept sequence is a list of observations over time.

eg



1 (1, 1) No percept
2 (1, 2) Breeze (pit nearby)
3 (2, 2) Stench, Breeze (Wumpus & pit nearby)
4 (2, 3) Glitter (Gold is here)

Q. 17) Solve the following Crypto-arithmetic problem 1.
SEND + MORE = MONEY.

→ Let's solve the crypto-arithmetic problem.

```
  SEND
+ MORE
-------
 MONEY
```

Step 1: Assign Unique Digits to Letters
Each letter represents a unique digit (0-9). The sum must satisfy the equation.

Step 2:- Identify Constraints

M = 1 (since Money has 5 digits) & M is 1st)
O = 0 (as it's the second digit.

Find S

S + M ≥ 10 (to cause a carry), so

S = 9

Page No.
Date

Find E and N

No carry from column 3 → E = 5, N = E + 1 = 6.

Find R

Carry exists from column 1 → R = 8.

Find D and Y

D + E = 10 + Y ⟶ D = 7, Y = 2

Final answer:-

S = 9, E = 5, N = 6, D = 7, M = 1, O = 0, R = 8, Y = 2

Verify:-

```
  S E N D        9 5 6 7
+ M O R E      + 1 0 8 5
-----------    ---------
M O N E Y      1 0 6 5 2
```

Ans :⟶ 10652

Q. 18) Consider the following axioms: All people who are graduating are happy. All happy people are smiling. someone is graduating.
Explain the following:-
1) Represent these axioms in first order predicate logic.
2) Convert each formula to clause form.
3) Prove that "Is someone smiling?" using resolution technique. Draw the resolution tree.

→ Step 1:- Represent Axioms in First-Order Predicate Logic.

Let:-

- G(x) → x is graduating
- H(x) → x is happy.
- S(x) → x is smiling.

Given axioms:

1) All people who are graduating are happy.
   $\forall x (G(x) \rightarrow H(x))$

2) All happy people are smiling.
   $\forall x (H(x) \rightarrow S(x))$

3) Someone is graduating
   $\exists x\, G(x)$

Step 2:- Convert to Clause form

Standardize variable (Remove Universal Quantifiers)

1) $G(x) \rightarrow H(x)$ becomes $\neg G(x) \vee H(x)$
2) $H(x) \rightarrow S(x)$ becomes $\neg H(x) \vee S(x)$
3) $\exists x\, G(x)$ becomes $G(A)$ (Introduce a skolem constant A)

Clause form:

- $\neg G(x) \vee H(x)$
- $\neg H(x) \vee S(x)$
- $G(A)$

Date

Step 3:- Prove "Is someone smiling?"

We need to prove $\exists x\, S(x)$ or show that $S(A)$ is true using resolution.

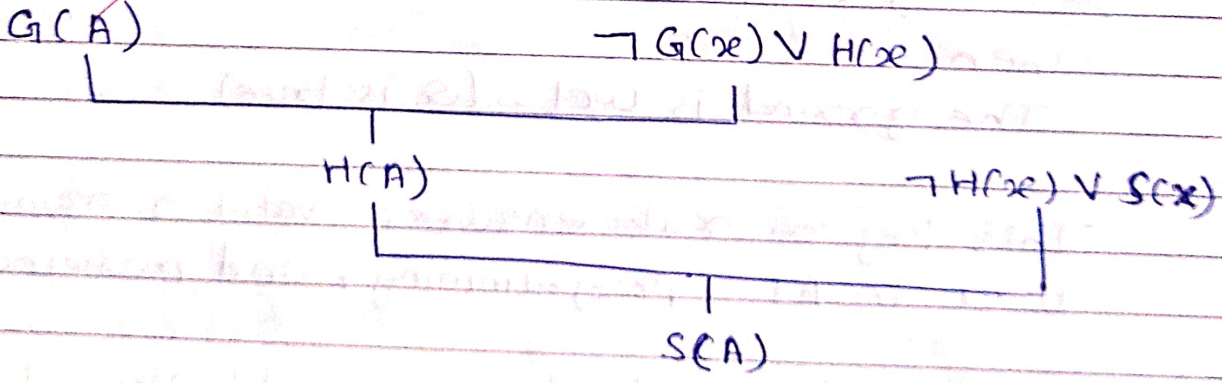
Resolution steps

1) From $G(A)$ and $\neg G(x) \vee H(x)$
   - $G(A)$ unifies with $G(x) \rightarrow H(A)$ is derived.

2) From $H(A)$ and $\neg H(x) \vee S(x)$
   - $H(A)$ unifies with $H(x) \rightarrow S(A)$ is derived.

Since we derived $S(A)$, we conclude that someone is smiling.

Step 4: Resolution Tree



Thus, Someone is smiling is proved using resolution.

Page No.
Date

Q.19) Explain Modus ponen with suitable example.

→ Modus Ponens (Law of Detachment)

Modus ponens is a fundamental rule of inference in logic that states:

If:

1) $P \rightarrow Q$ (If P is true, then Q is true)

2) P (P is true)

Then:

Q must be true.

Example

Premises:-

1) If it rains, the ground will be wet.

$P \rightarrow Q$ (If Rain, then wet ground)

2) It is raining.

P (Rain is true)

then

The ground is wet. (Q is true)

This logical rule ensures valid reasoning and widely used in AI, programming, and mathematical proofs.

Q.20) Explain forward chaining and backward chaining algorithm with the help of example.

→ 1) Forward Chaining (Data Driven)

• Starts with known facts and applies rules to infer

new facts until the goal is reached.
- Used in expert systems, AI planning, and production systems.

eg.

Rules:

1) If it rains, the ground will be wet.
   Rain ⟶ Wet ground
2) If the ground is wet, people carry umbrellas.
   wet Ground ⟶ Umbrella.

Given Fact:

It is raining.

Forward Chaining Steps:

1) It is raining triggers Rule 1 ⟶ The ground is wet.
2) The ground is wet triggers Rule 2 ⟶ People carry umbrellas.

Conclusion: People carry umbrellas.

2) Backward Chaining (Goal-Driven)

- Starts with a goal and works backwards to check if known facts support it.
- Used in AI problem-solving and diagnostic systems.

Date

Example:-

Goal: Are people carrying Umbrellas?

Rules:
1) If the ground is wet, people carry umbrellas
   Wet Ground → Umbrella.
2) If it rains, the ground is wet:
   Rain → Wet Ground.

Backward Chaining steps:-

1) Are people carrying umbrellas? → Check if the ground is wet.
2) Is the ground wet? → Check if it is raining.
3) Is it raining ? → Yes (Given Fact).

Conclusion: Yes, people carry umbrellas.