LTCE ①

10/10

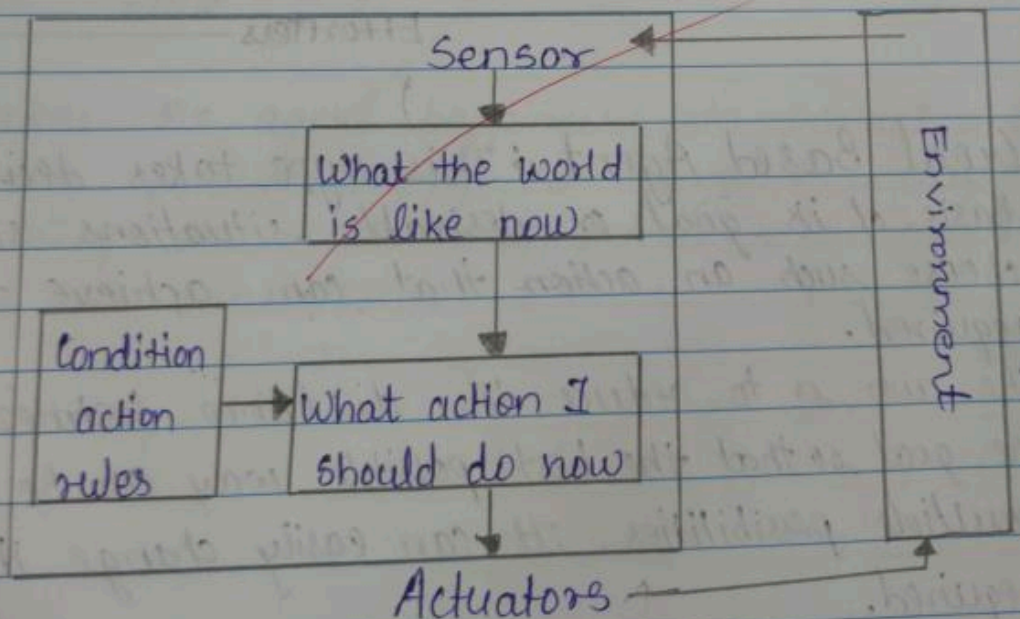
Q1] Define Agents. What are different types of agents with diagram?

→ An agent is anything that can be viewed as perceiving its environment through sensors and acting upon that environment through effectors.

Types of Agents :-

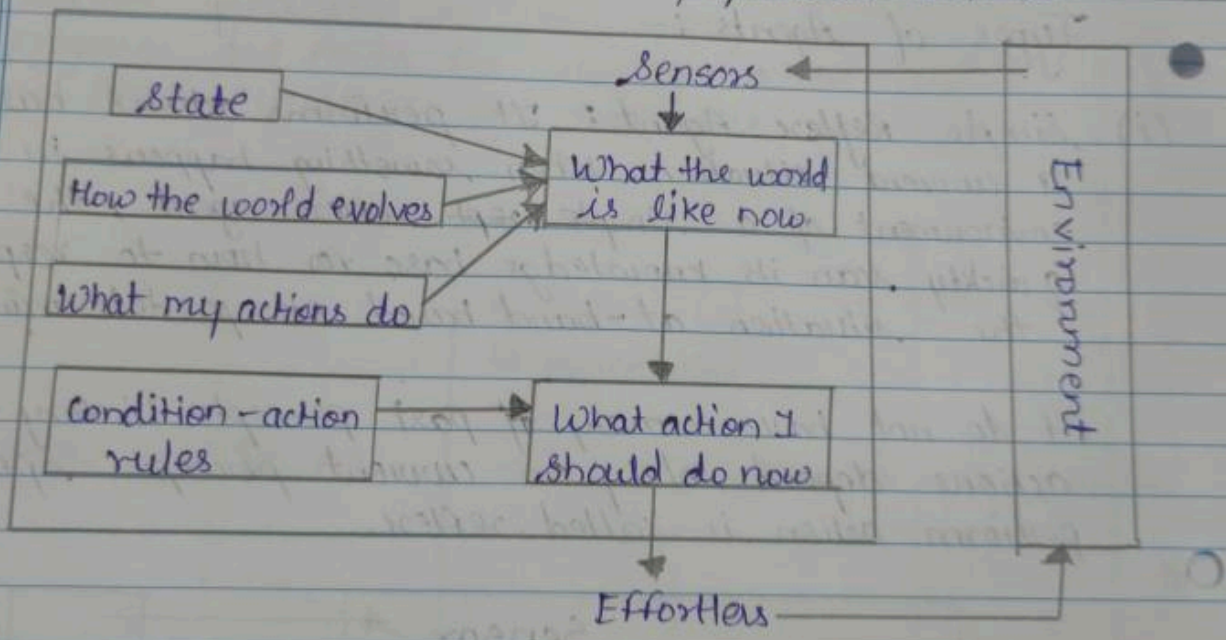
(i) Simple Reflex Agent : It performs actions based on a current situation. When something happens in the environment of a simple ~~rept~~ reflex agent, the agent quickly scan its knowledge base for how to respond to the situation at-hand based on predetermined rules.

It do not have memory of past percepts history. So, actions depend solely on current percept. Spontaneously perform action is called reflex.

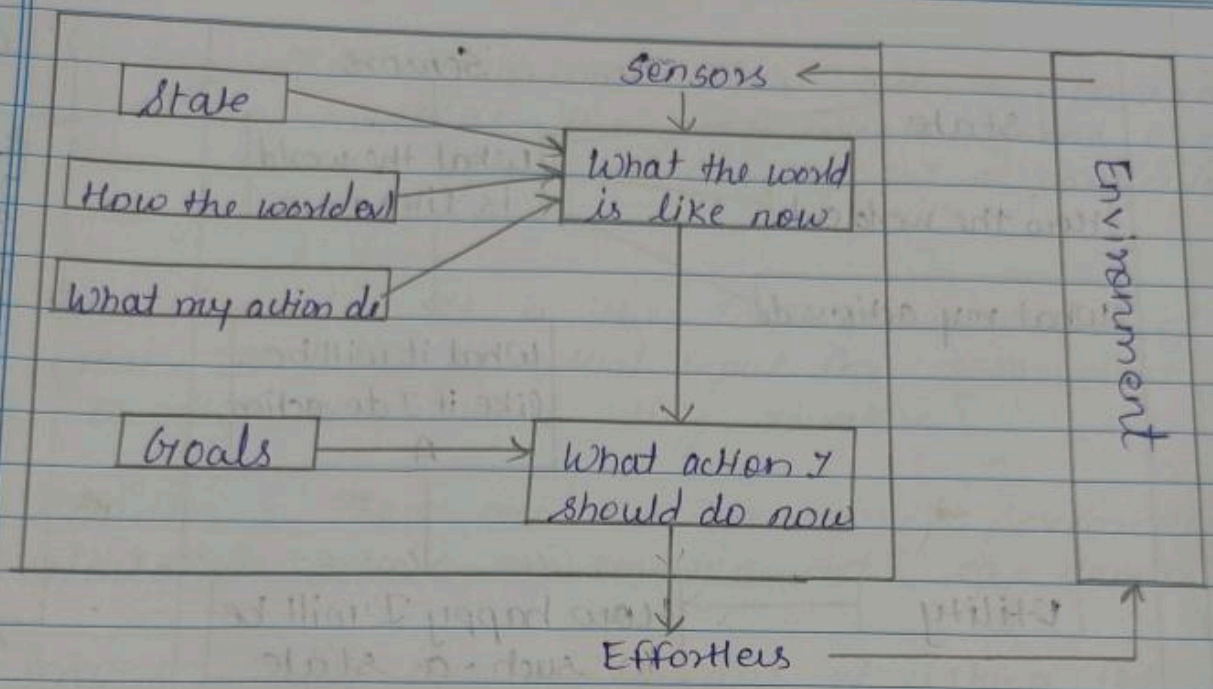


LTCE

(ii) Model Based Agent : It can work in a partially observable environment, and track the situation. This knowledge about how the world evolves is called as model of the world. It store history in Internal Mode.
These agents have the model, which is knowledge of world and based on the model they perform actions.

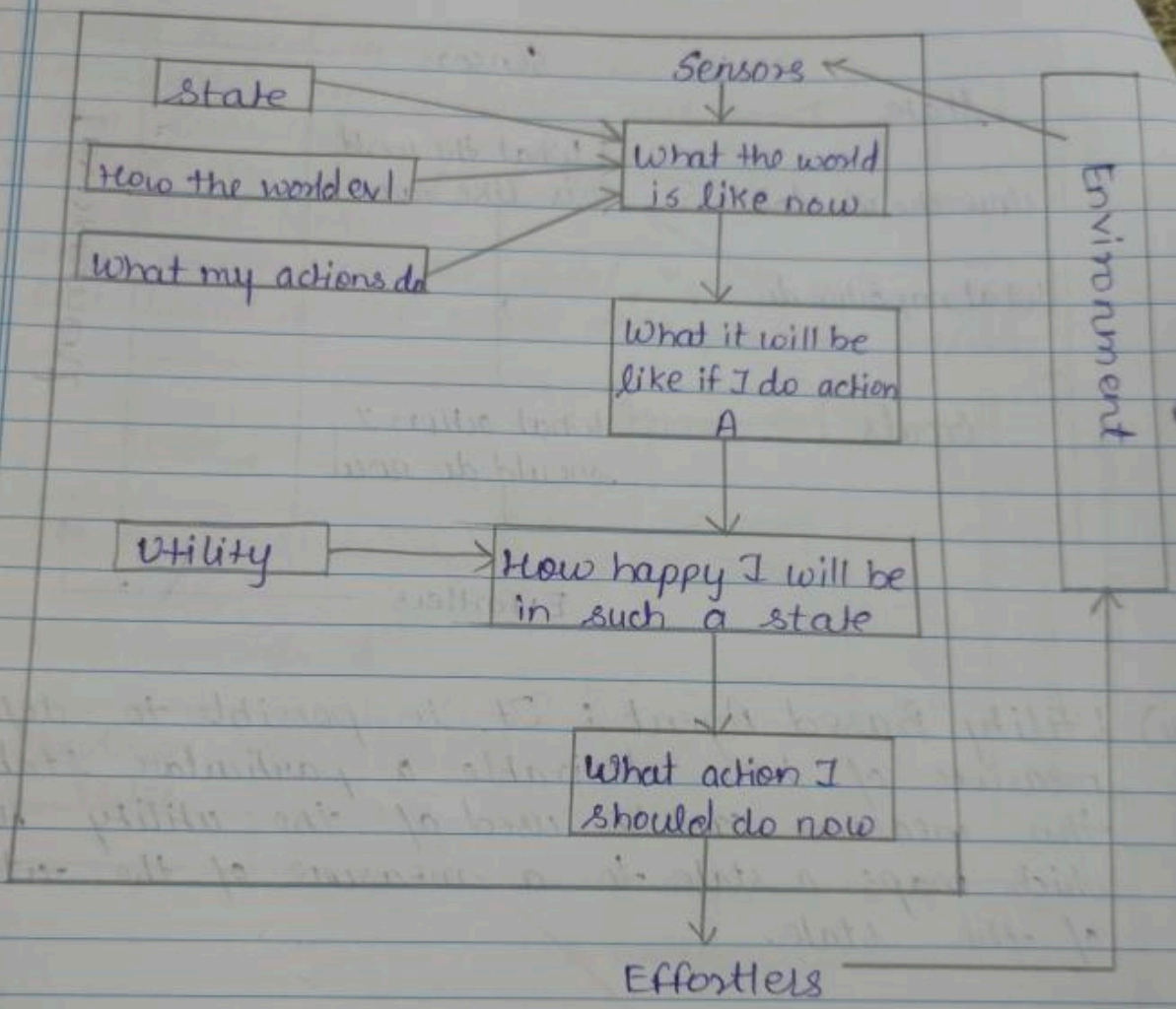


(iii) Goal Based Agent : This type takes decisions on the basis of its goal or desirable situations so that it can choose such an action that can achieve the goal required.
The aim is to reduce the distance between action and the goal so that the best possible way can be chosen from multiple possibilities. It can easily change its behaviour if required.



(iv) Utility Based Agent : It is possible to define a measure of how desirable a particular state is this measure can be used of the utility function which maps a state to a measure of the ~~utt~~ utility of the state.

It takes the agent happiness into account and gives an idea of how happy the agent is because of the utility and hence the action with maximum utility is considered.



Q2] Describe PEAS with example ?

→ PEAS stands for Performance, Environment, Actuators and sensors.

PEAS is an AI agent representation system that focus on evaluating the performance of the environment, sensors and actuators.

LTCE ⑤

(i) Performance Measures: These are the parameters used to measure the performance of the agent. How well the agent is carrying out a particular assigned task.

(ii) Environment: It is the task environment of the agent. It takes perceptual input from environment & acts on the environment using actuators.

(iii) Actuator: These are the means of performing calculated actions on environment. eg: Hands & legs.

(iv) Sensors: These are the means of taking the input from the environment. eg: ears, nose & eyes.

Example: Shopping for books on the internet.

(i) Performance:
→ Price of book
→ cost minimization
→ Quality of book

(ii) Environment:
→ Internet websites
→ vendors / sellers
→ shippers

(iii) Actuators:
→ Filling in the forms
→ Display to the user
→ Follow URL

(iv) Sensors:-
→ key board entry
→ HTML
→ Browser used to find web page.

LTCE

Q3] Write a short note on :-

(i) Fully absorable & partially absorable environment

→ In a fully observable environment, the agent is familiar with the complete state of the environment at a given time. There will be no portion of the environment that is hidden for the agent.

Example : While running a car on the road (Environment) the driver (Agent) is able to see - road conditions, signboard & pedestrains on the road at a given time and drive accordingly. So Road is fully observable environment for a driver while driving the car.

In a partially observable environment, the agent is not familiar with the complete environment at a given time.

Example : Playing card games is a perfect example, as the other parts of the environment : opponent, game name, etc are known for the player (Agent).

(ii) Deterministic and Stochastic Environment

→ Deterministic are the environments where the next state is observable at a given time. So there is no ~~necessary~~ uncertainity in environment.

Example : The traffic signal is a deterministic environment where the next signal is k/s pedestrian (Agent).

The Stochastic environment is the opposite of a deterministic environment. The next state is totally unpredictable for the agent. So randomness exists in the environment.

Example : The radio station where the listner is not aware about the next song or playing a soccer is stochastic environment.

Q 4] Describe Structure of agents.

→ Agents in Artificial Intelligence follow this simple structural formula.

Agent = Architecture + Agent Program

These are the terms most associated with agent structure :-

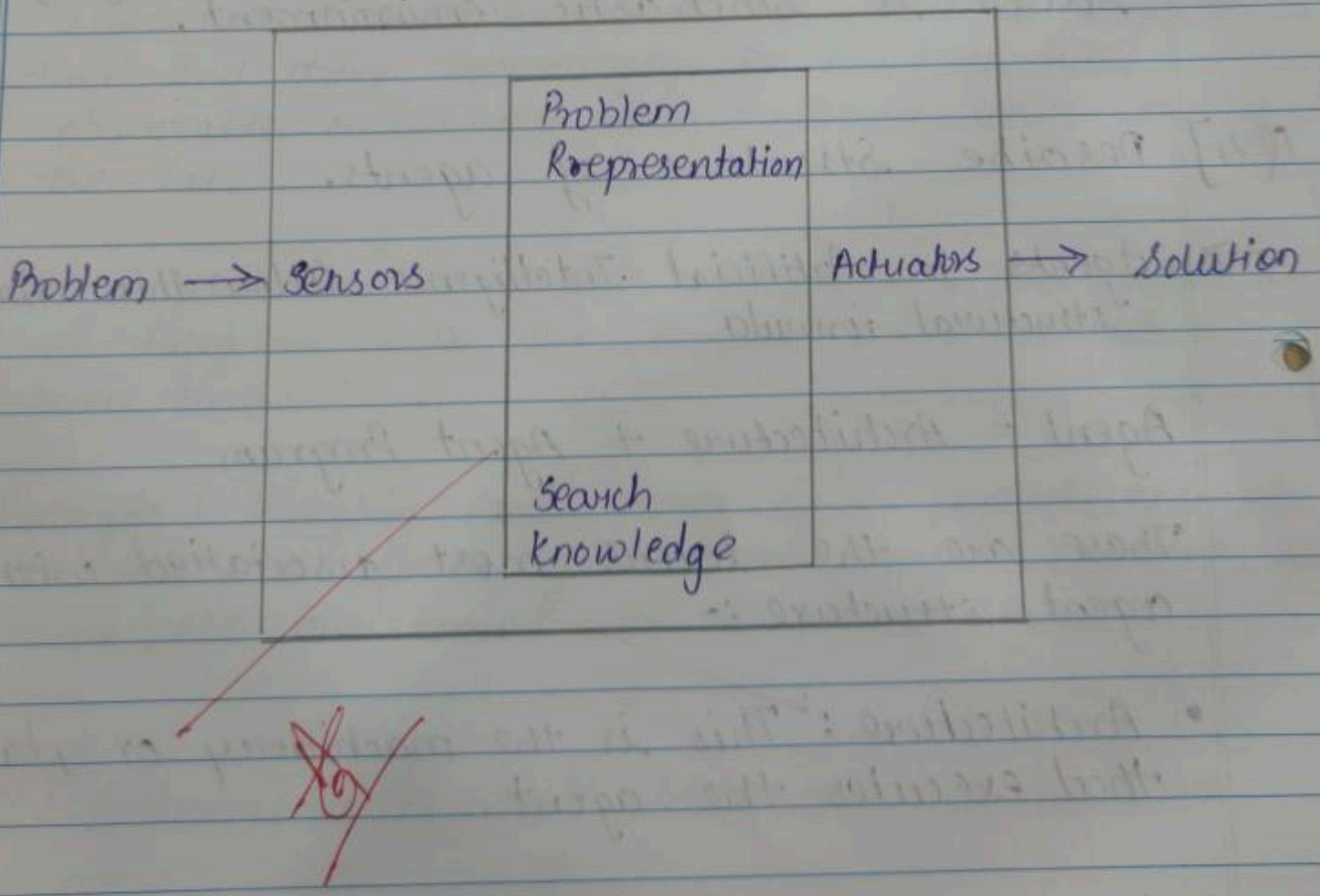
- Architecture : This is the machinery or platform that executes the agent.
- Agent Function : The agent function maps a percept to the Action, represented by the

LTCE

following formula : $f = P^* \to A$

- Agent Program : The agent program is an implementation of the agent function. The agent program produces function f by executing on the physical architecture.

The architecture makes the percepts of from the sensor available to the program, runs the program and feeds the program action choices to the actuators as they are generated.



Q1. Define chromosome, selection, fitness function, cross-over and mutation as used in Genetic Algorithm. Explain how genetic algorithm works.

→ The genetic algorithm is a method that is applied to solve both constrained and unconstrained optimisation problems.

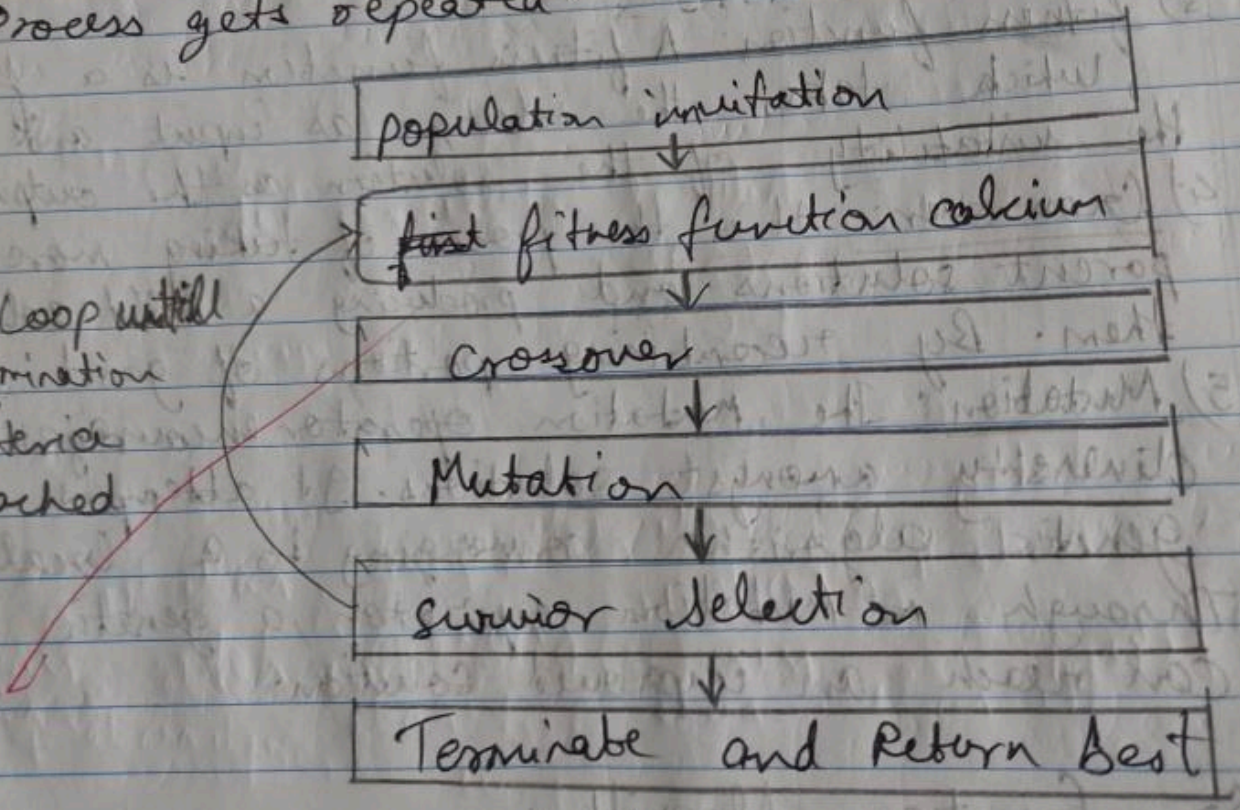
1) Chromosome: Chromosome is regarded as solution to the given problem.
2) Selection: During each successive generation, a portion of the existing population is selected to obtain a new generation.
3) fitness function: A fitness function is a function which takes the solution as input and produces the suitability of the solution as the output.
4) Cross-over: It is the process of taking more than one parent solutions and producing a child solution from them. By recombining portions of good solutions.
5) Mutation: The mutation operator encourages genetic diversity amongst solutions. It attempts to prevent genetic algorithm converging to a local minimum. Through ~~a~~ mutation operator, a genetic algorithm can reach an improved solution.

Genetic Algorithm:-

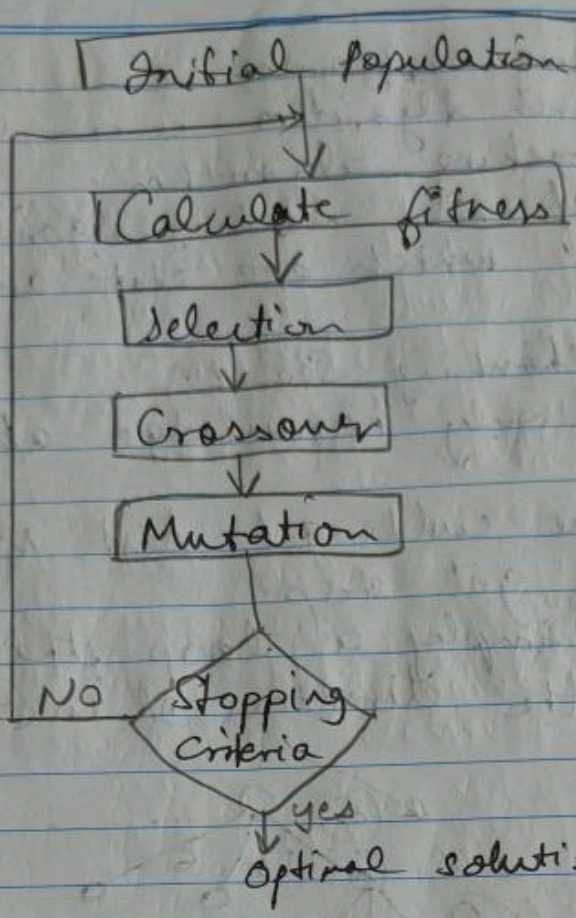
An algorithm is a progression of steps for solving a problem. A genetic algorithm is a problem solving technique that uses genetics as its model

of problem-solving. It is based on natural selection.

Basic structure: The basic structure of GA is
i) We begin with an initial population, it may be generated at random or seeded by other heuristics.
ii) Select parents from this population for mating.
iii) Apply crossover and mutation operators on the parents to generate new off-springs.
iv) Finally these off-springs replace the existing individuals in the population and.
v) Process gets repeated.



* Basic structure of GA



Q2.) Explain Hill Climbing algorithm with example. Explain the problems faced by Hill-climbing algorithm.

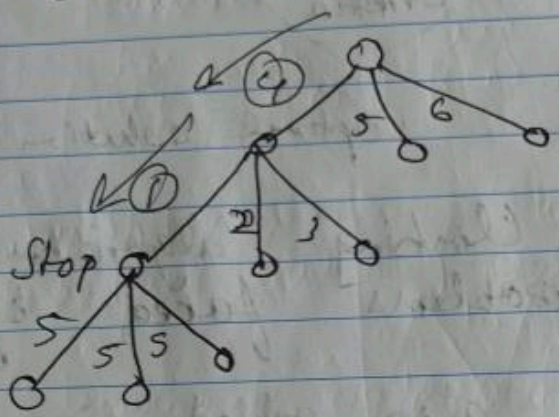
→ This algorithm also called discrete optimization algorithm, uses a simple heuristic function the amount of distance the node is from the goal. The ordering of choices is a heuristic measure of the remaining distance one has to traverse to reach the goal node.

LTCE

Hill climbing algorithm
→ local search Algorithm is a Greedy Approach
→ No backtracking algorithm.
→ A node of Hill climbing algorithm has two components which are state and value.

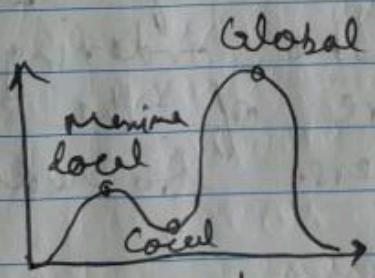
* Hill climbing Search :-
(1) There is only knowledge about local domain.
(2) Often we will get best move then the process will get stop.
(3) It is simply a loop that continuously moves in the direction of increasing value.

eg.



problems faced by Hill climbing algorithm :-

(1) local maxima



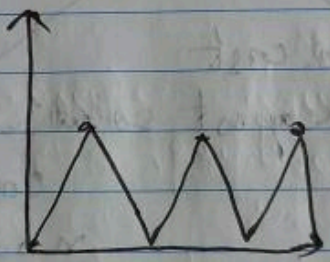
A local maximum is a state which is better than its neighbouring state, however it is not the best possible state as the

exists a state where objective function value is higher.

(2) Plateau:

Plateau obstacle can be solved by taking ~~making~~ making a big jump from the current state which will land you in non-plateau region.

(3) Ridge:



~~Rid~~ Ridge occurs when there are multiple peaks and all have the same value or in other words, there are multiple local maxima which are same as global maxima.

(Q3) Explain A* algorithm in detail.

→

A* Algorithm is the most commonly known form of best-first search. It uses heuristic function h(n) and cost to reach the node n from the start state g(n). It has combined features of UCS and greedy best-first search, by which it solve the problem efficiently.

In A* search algorithm we use search heuristic as well as the cost to reach the node. Hence we can combine both costs as following and this sum is called as a fitness number.

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

- $f(n)$: Estimated cost of the cheapest solution
- $g(n)$: cost to reach node n from start state
- $h(n)$: cost to reach from node n to goal node.

Algorithm:

1. Start at current node or from root node
2. While there are place to explore:
   - → Pick the place with the lowest total estimated cost (current cost + estimated remaining cost).
   - → Move to the place.
   - → If you have reached your destination, Exit.
   - → Or search neighboring places / nodes.

(3) Once you have explored all nodes or reached the goal state:
→ Trace your path backward from goal to the start node using the lowest-cost path.

(4) You have found the shortest path from root node to goal node/state.

(5) End.

Advantages:
(1) A* search algo. is the best algo than others as it is optimal and complete.
(2) This algorithm can solve complex problems.

Disadvantages:
(1) It does not always produce the shortest path as it mostly based on heuristics and approxim

(2) The main drawback is memory requirement as it keeps all generated nodes in the memory, so it is not practical for various large-scale problems.

* Time & space complexity is $O(b^d) = O(b^d)$.

Continue Next page →

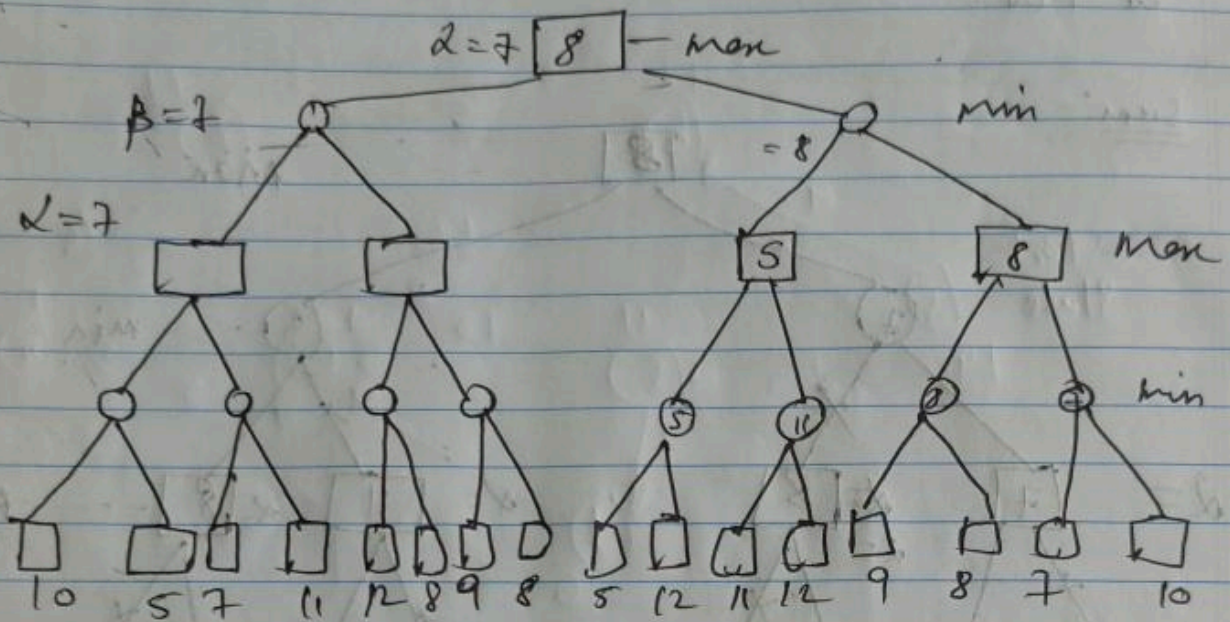
LTCE

Q5) Differentiate between informed search & uniformed search algorithms.

→

Informed Search algorithm	Unformed Search algorithm
1) It uses knowledge for the searching process.	1) It ~~also~~ doesn't use knowledge for the searching process
2) It is also known as ~~Heuristic~~ Heuristic search.	2) It is also known as Blind search.
3) Due to a quicker search it consumes much less time.	3) Due to slow searches, it consumes comparatively more time.
4) Cost is low	4) Cost is high
5) Having a wide scope in terms of handling large search problems.	5) Solving a massive search task is challenging
6) It is more efficient.	6) It is less efficient
7) Finding solution is quicker	7) Finding the solution is slower.
8) Eg. (1) Greedy Search (2) A* Search (3) AO* Search (4) Hill climbing algo.	8) eg. (1) DFS (2) BFS (3) Branch & Bound.

(14) Explain α, β pruning algorithm. Apply α, β pruning on following example considering first node as max.



α (Alpha) β (Beta) pruning is a technique used in game tree search algorithms, such as the minmax algorithm, to significantly reduce the number of nodes that need to be evaluated during the search.

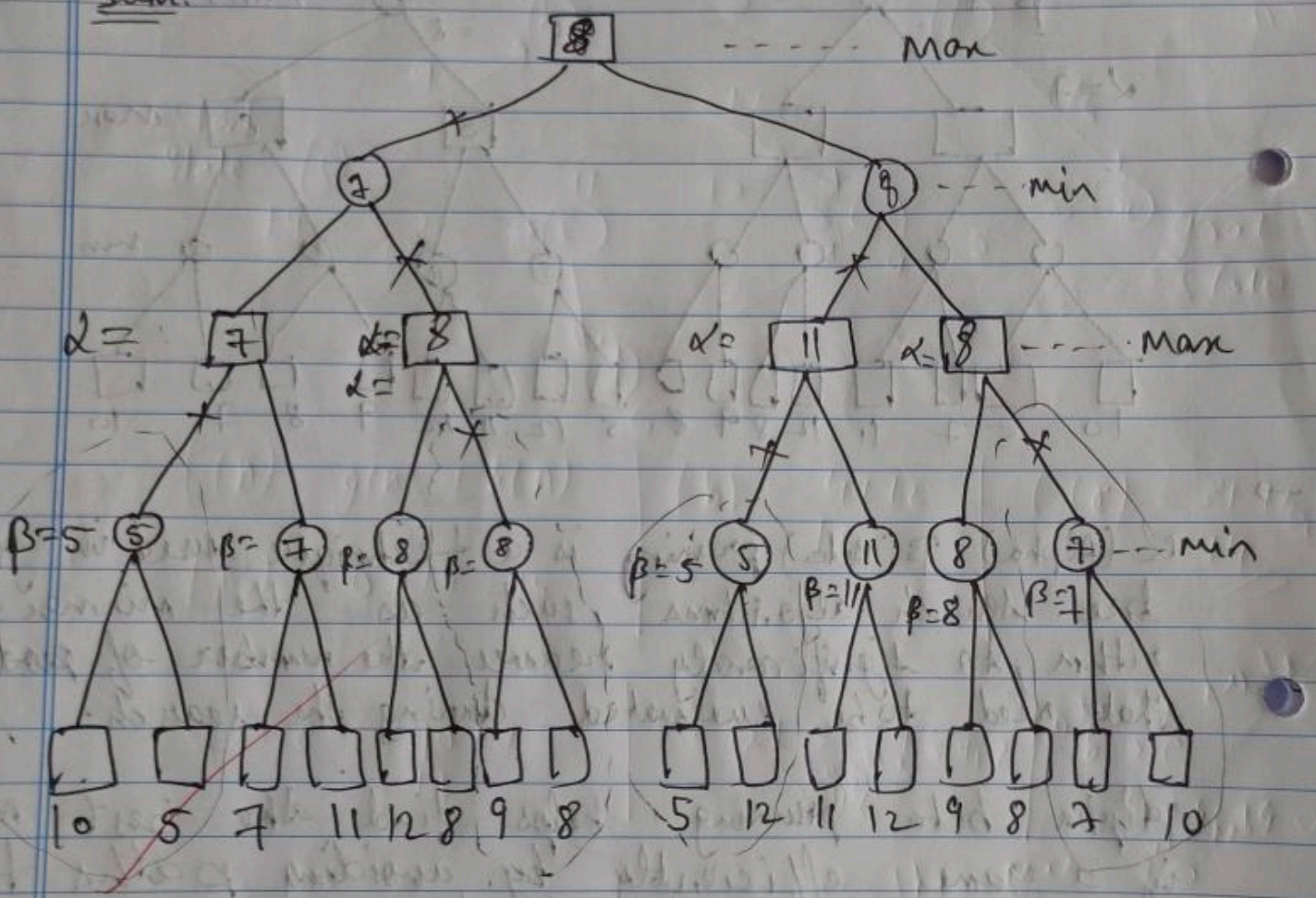
Alpha beta pruning helps find the best move in a game efficiently by avoiding paths that won't affect the final decision, making the search process faster without sacrificing accuracy.

Two parameter can be defined as:

(a) Alpha: The best (highest value) choice we have found so far at any point along the path of Maximizer. The initial value of α alpha is -∞.

(b) Beta: The best (lowest value) choice we have found so far at any point along the path of Minimizer. The initial value of beta is $+\infty$.

Sum:



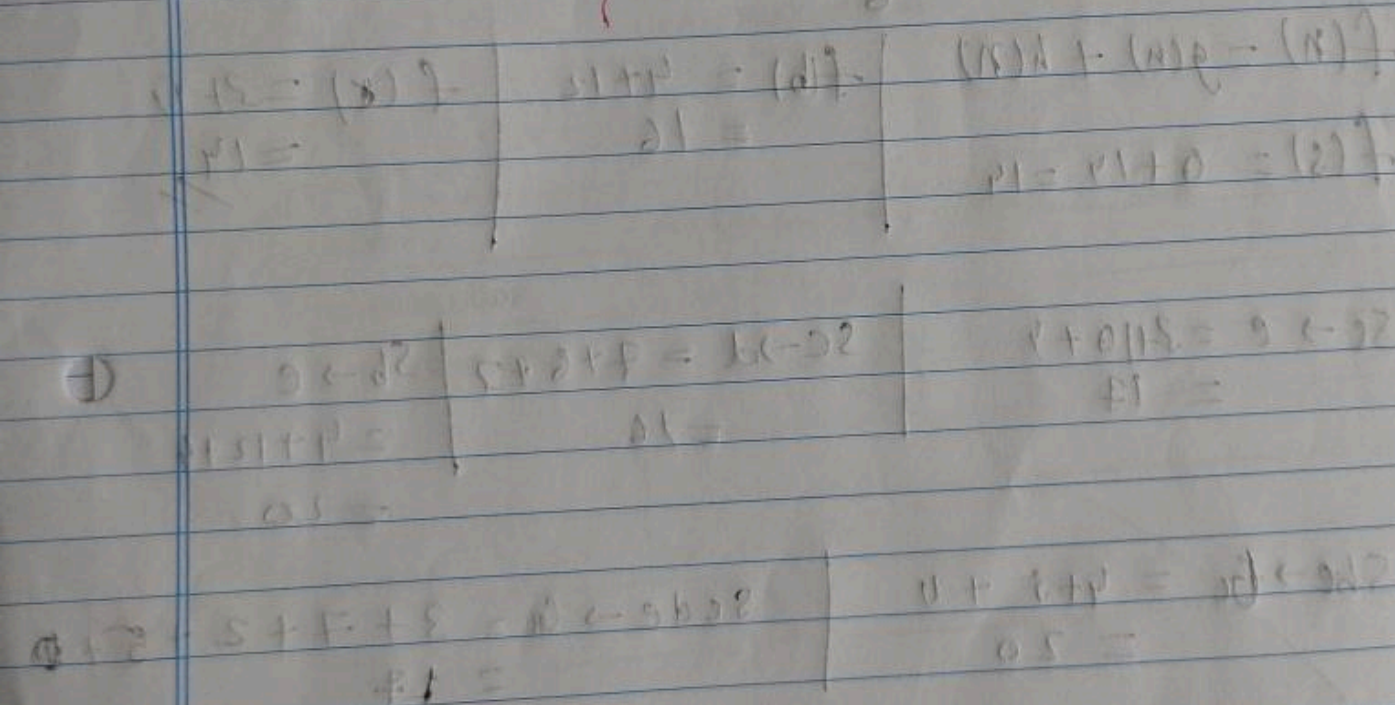
Step 1: In min (β) Beta we take min value of the node ~~node 5, node 8, node 7 is pruned~~ is taken. So node 10 ~~is pruned~~, node 11 is, node 12

Step 2: node 9, node 12, node 12, node 9, node 10 is pruned.

Step2: In this max value is checked (α), so node 5, node 8, node 5, node 7 is prun

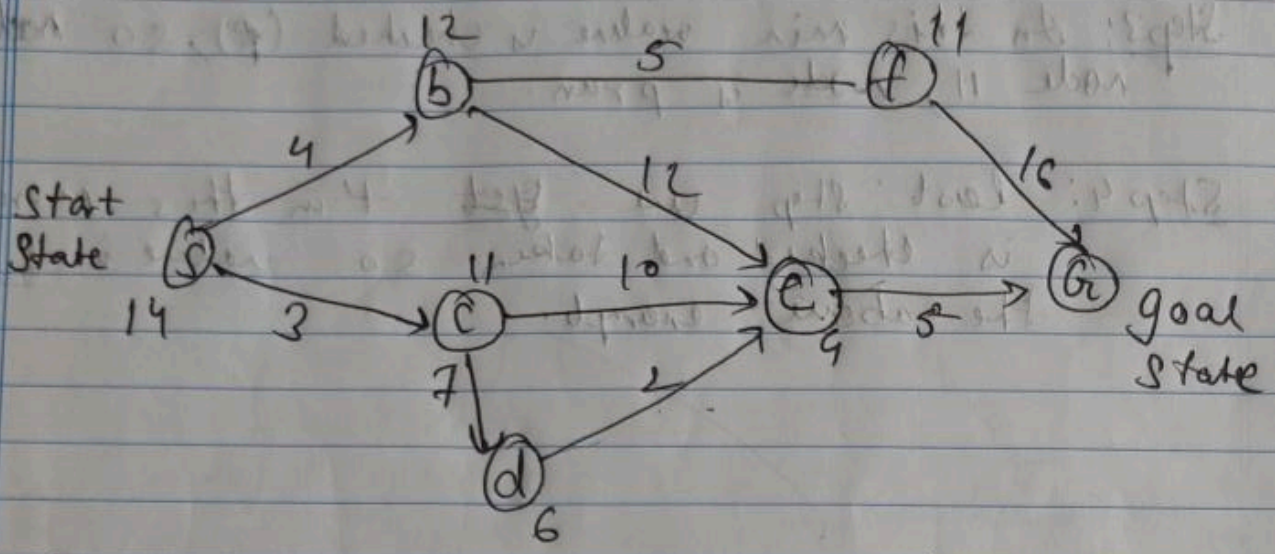
Step3: In this min value is checked (β), so node 8, node 11, ~~node~~ is prun.

Step 4: Last step we ~~get~~ 4 in this max value is checked and taken so value is 8 in the above example.



Q3)

→ **example of A* algorithm:-**



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

$f(s) = 0 + 14 = 14$

$f(b) = 4 + 12 = 16$

$f(c) = 3 + 11 = 14$ //

$Sc \to e = 3 + 10 + 4 = 17$

$Sc \to d = 7 + 6 + 2 = 16$

$Sb \to e = 4 + 12 + 4 = 20$

$Sbe \to G = 4 + 5 + 4 = 20$

$Scde \to G = 3 + 7 + 2 + 5 + 0 = 17$ //

⇒ Sbe G is the path for the above example.

31/8/23

Q] Illustrate forward chaining & backward chaining in propositional logic with an example?

→ Forward Chaining :-

The inference engine goes through all the facts, conditions and derivatives before deducing the outcome i.e. When based on available data a decision is taken then the process is called as Forward Chaining. It works from an initial state and reaches to the goal (final decision).

It is a down-up approach, as it moves from bottom to top. It is a process of making a conclusion based on known facts or data, by starting from initial state and reaches the goal state.

It is also called as data-driven as we reach to the goal using available data. It is commonly used in expert system, such as CLIPS, bussiness and production rule system.

Example :- A
$A \rightarrow B$
B

He is running
If he is running, he sweats.
He is sweating

Backward Chaining :-

In this, the inference system knows the final decision or goal, and works backwards to determine what facts must be asserted so that the goal can be achieved, i.e. it works from goal and reaches the initial state.

It is a top-down approach and based on modus ponens inference rule.

The goal is broken into sub-goal. It is called as goal-driven approach, as a lists of goals decides which rules are selected and used.

It is used in game theory, inference engines, proof assistants and various AI applications.

It is mostly used a depth-first search strategy for proof.

Example :-

B

$A \rightarrow B$

A

He is sweating.

If he is running, he sweats.

He is running.

Q2] Define Belief Network ? Describe the steps of constructing Belief Network with example ?

→ Belief Network is also called as Bayesian Network which is a probabilistic graphical model which represents a set of variables and their conditional dependencies using a directed acyclic graph.

It ~~represent~~ consists of nodes that represent random variables and edges that represent probabilistic dependencies between these variables.

Steps to construct a Belief Network with an example :-

⇒ Step 1 : Define the Variables - identify the random variables that you want to model.
Lets use two variables : Umbrella & Weather

- Umbrella (U) represents whether a person is carrying a ~~and~~ umbrella (Yes or No)
- Weather (W) represents the weather conditions (Rainy, Sunny or Cloudy)

⇒ Step 2 : Establish the structure - Create a graphical structure to represent the dependencies among the variables using directed acyclic graph(DA

"Weather" influences "Umbrella", so we have to direct edges from "weather" to "Umbrella".

⇒ Step 3: Define Conditional Probability Tables - Specify the CPTs for each variables :-

- For "Umbrella" → Yes or No
- For "Weather" → Sunny, Cloudy and Rainy

⇒ Step 4: Assign Probabilities :-

- For Weather ⇒ P(Weather = Sunny) = 0.6
  P(Weather = Rainy) = 0.3
  P(Weather = Cloudy) = 0.1

- For Umbrella we will define probabilities based on weather conditions
  P(Umbrella = Yes | Weather = Sunny) = 0.1
  P(Umbrella = Yes | Weather = Rainy) = 0.4
  P(Umbrella = Yes | Weather = Cloudy) = 0.8
  P(Umbrella = No | Weather = Sunny) = 0.9
  P(Umbrella = No | Weather = Rainy) = 0.6
  P(Umbrella = No | Weather = Cloudy) = 0.2

⇒ Step 5: Inference & Probability Calculations - you can use the network to calculate probabilities or answer questions.

What is the probability that someone is carrying an umbrella on a rainy day?

→ P(Umbrella = Yes | Weather = Rainy) = 0.4

⇒ Step 6: Updating Probabilities

If you receive new information, you can update the probabilities in the network accordingly.

Q3] Explain Total Order Planning & Partial Order Planning in detail and example?

→ (a) Total Order Planning :-

- It is a straight forward approach where all the actions in a plan are strictly ordered from start to finish.
- This means that each action in the plan has a fixed position and cannot be executed in parallel with other actions. It is also know as Linear Planning.
- Preconditions and Effects : In Total Order Planning, actions have defined preconditions and effects. An action can only be executed if its preconditions are met, and it changes the state of the world according to its effects.
- Sequential Execution : Actions are executed in a strict sequence, and the plan proceeds from the initial state to the goal state one step at a time.

Example :- Consider the task of making a sandwich. In a total order plan, you have to follow sequence :

(1) Take bread
(2) Spread peant butter
(3) Place bananas
(4) Close the sandwich

This actions must occur in specific order and no action can overlap.

(B) Partial Order Planning :-

- Partial Order planning is more flexible approach to planning, allowing for actions to be executed in parallel and without any strict ordering.
In this approach :

(i) Causal Links : Instead of a strict order, actions are linked by causal relationships. One action can enable or inhibit another action.

(ii) Concurrency : Actions can be executed in parallel if there are no conflicts in the causal links. These links leads to a more efficient and flexible plan.

Example :- lets say you need to plan a simple morning routine that includes three tasks :-
(1) Brushing your teeth
(2) Making a cup of cofee
(3) Checking your email

All there actions can be done parallel and at a same time.

Q4] Consider the following axioms
(i) All people who are graduating are happy
(ii) All happy people simple.
(iii) Someone is graduating.
Prove that Is someone Smiling ? Using resolution technique draw resolution tree.

→ Step 1 : Symbolic logic :-
x = people
G = people graduating
H = happy people
S = smiling people

Step 2 : First Order Propositional logic

(i) $\forall x\, G \;/\; \forall x\, H$ — $\forall x\, [\text{Graduating}(x) \rightarrow \text{happy}(x)]$
(ii) $\forall x\, H \;/\; \forall x\, S$ — $\forall x\, [\text{happy}(x) \rightarrow \text{smile}(x)]$
(iii) $\exists x\, G(x)$ — $\exists x\, \text{Graduating}(x)$

Prove : Someone is Smiling $\Rightarrow \exists x\, \text{Smile}(x)$
negative $\Rightarrow \neg \exists x\, \text{Smile}(x)$

2023/10/18 11:05

$\alpha \rightarrow \beta \equiv \neg\alpha \vee \beta$

Step-3 : Convert FOL to CNF

① Eliminate Implications

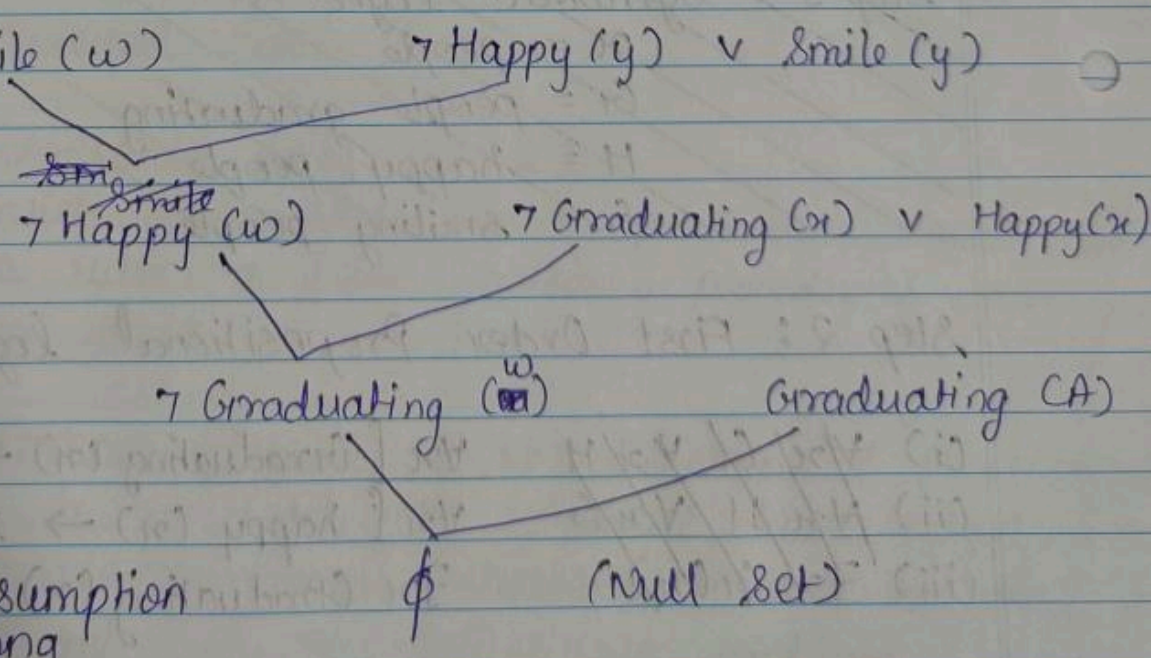
(i) $\forall x [\neg \text{Graduating}(x) \vee \text{Happy}(x)]$
(ii) $\forall x [\neg \text{Happy}(x) \vee \text{Smile}(x)]$
(iii) $\exists x \text{ Graduating}(x)$
(iv) $\neg \exists x \text{ Smile}(x)$

② Standardize Variables Apart

(i) $\forall x [\neg \text{Graduating}(x) \vee \text{Happy}(x)]$
(ii) $\forall y [\neg \text{Happy}(y) \vee \text{Smile}(y)]$
(iii) $\exists z \text{ Graduating}(z) \Rightarrow \text{Graduating}(A)$
(iv) $\neg \exists w \text{ Smile}(w) \Rightarrow \forall w \neg \text{Smile}(w)$

③ Drop Universal Quantifier

(i) $\neg \text{Graduating}(x) \vee \text{Happy}(x)$
(ii) $\neg \text{Happy}(y) \vee \text{Smile}(y)$
(iii) $\text{Graduating}(A)$
(iv) $\neg \text{Smile}(w)$



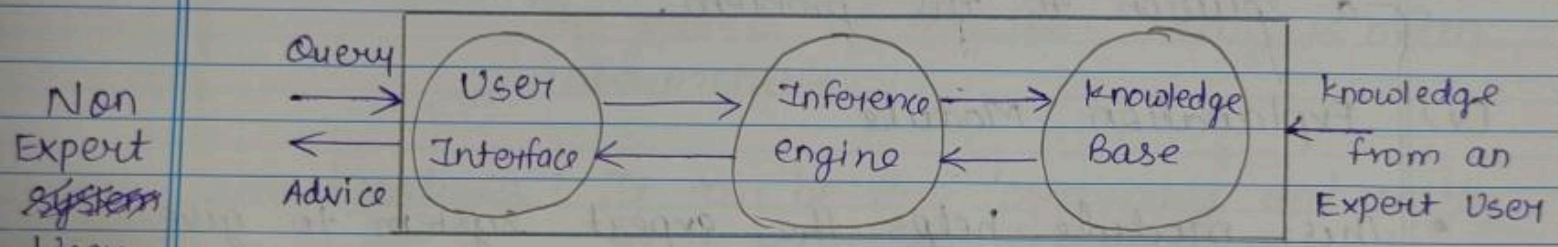
∴ Our assumption is wrong

Hence Someone is smiling

Q5] Draw and explain architecture of expert System.

→ An expert system is AI software that uses knowledge ~~base~~ stored in a knowledge base to solve problems that would usually require a human expert thus preserving a human expert's knowledge in its knowledge base.

Architecture of an Expert System



(i) Knowledge Base

- It represents facts and rules.
- It consists of knowledge in a particular domain as well as rules to solve problem, procedures and instrinsic data.

(ii) Inference Engine - The function of the inference engine is to fetch the relevant knowledge from the knowledge base, interpret it and to find a solution relevant to the users problem.

(iii) Knowledge Acquisation and Learning Module

- The function of this component is to allow the expert system and find a solution to acquire more knowledge from various sources and store it in the knowledge base.

(iv) User Interface

- This module makes it possible for a non-expert user to interact with the expert system and find a solution to the problem.

(v) Explanation Module

- This module helps the expert system to give the user an explanation about how the expert system reached a particular conclusion.

Q6] Explain different types of learning in AI?

→ (i) Supervised Learning

- It is also known as Supervised Machine Learning.
- Supervised learning is defined as when a model gets trained on a "labelled Dataset".
- Labelled datasets have both input and output parameter

Example:- Image classifier to differentiate between cats and dogs. Using labelled images from datasets of dogs and cats, machine will learn to classify between cat or a dog.

When we input new dog or cat images that it has never seen before, it will use the learned algorithm and predict.

Advantages : It have high accuracy.
The process of decision-making is often interpretable.

Disadvantages : It can be time consuming and costly as it relies on labelled dataset.

Applications: Image classification, NLP, Speech recognization

Categories :-

(i) Classification - The classification algorithms are used to solve the classification problems in which the output variable is categorical, such as "Yes" or "No", Male or Female, Red or Blue, etc

(ii) Regression - There are used to predict continuous output variables such as market trends, weather prediction, etc.

(ii) Un Supervised Learning

→ In unsupervised learning, the machine is trained using the unlabelled dataset, and the machine predicts the output without any supervision.

→ The main aim is to group or categorize the unsorted dataset according to the similarities, patterns, and differences.

→ Example : ~~Customer Segmentation~~ Suppose a basket of fruit images, input into ~~no~~ ML model. The image is unknown to model so it has to find patterns and categorises of the objects. So it will discover its patterns, color difference, shape, etc and predict the output.

→ Categories : (i) Clustering : It is a technique to find the inherent groups from data. It groups similar in one group and no similarities in another group.

(ii) Association :- It is used to find dependency of one data item on another data item and map those variables accordingly so that it can generate maximum profit.

→ Advantages - Helps to discover hidden patterns and various relationship between data

→ Disadvantages : Without using labels, it may be difficult to predict.

→ Applications : Data Preprocessing, Exploratory Data Analysis.

(iii) Semi-Supervised Learning

→ It works between supervised and unsupervised learning so it uses both labelled and unlabelled data.

→ This is useful when obtaining labelled data as it is expensive or time-consuming.

→ Example :- Sentiment Analysis - you have small labelled reviews dataset (negative or positive) and large unlabelled reviews. So you can use Semi-Supervised learning to improve sentiment analysis by incorporating both labelled & unlabelled.

→ ~~Categories~~ Advantages : It is simple and easy to understand.
It is highly efficient.

→ Disadvantages : Iterations results may not be stable.
Accuracy is low.

→ Application :- Recommendation System, Translator.

LTCE

(iv) Reinforcement Learning

→ It is a learning method that interacts with the environment by producing actions and discovering errors.

→ Trial, error, and delay are the most relevant characteristics of reinforcement learning.

→ Example : Consider AI agent ~~vs~~ to play chess.

The agent explores different moves and receives positive or negative feedbacks based on outcomes.

It finds applications in which they learn to perform task by interacting with their surrounding.

→ Advantages : Helps in solving real-world problems.

→ Disadvantages : It requires huge data and computations.

→ Applications : Robotics, Virtual Reality, etc.

Q7] Explain Bayes Theorem with example?

→ Bayes Theorem is used to determine the conditional probability of event A when event B has already happened.

Statement : "The conditional probability of an event A, given the occurrence of another event B, is equal to the product of the event of B, given A and the probability of A divided by the probability of event B." i.e.

$$P(A/B) = P(B/A) \cdot \frac{P(A)}{P(B)}$$

where,
P(A/B) is the probability of event A when event B happens
P(B/A) is the probability of event B when A happens.
P(A) and P(B) are probabilities of events A & B. It is never equal to zero.

Example :- What is the probability that person has disease Dengue with neck pain?

$P(B) = \frac{1}{30,000}$ $P(A) = 0.02$ $P(A/B) = 0.8$

find $P(B/A)$?

Solution :-

As per the formula $P(A/B) = P(B/A) \cdot \frac{P(A)}{P(B)}$

so $P(B/A) = P(A/B) \cdot \frac{P(B)}{P(A)}$

$= 0.8 \times \frac{1}{30,000 \times 0.02}$

$= 0.001333$ //

Q8] Explain the steps involved in converting the propositional logic statement into CNF with example ?

→ Rules to convert P2 to CNF

① Remove Biconditional using rule

$$p \leftrightarrow q \equiv (p \rightarrow q) \wedge (q \rightarrow p)$$

② Remove Implication using rule

$$p \rightarrow q \equiv \sim p \vee q$$

③ Move negation inwards

$$\sim (p \vee q) \equiv \sim p \wedge \sim q$$
$$\sim (p \wedge q) \equiv \sim p \vee \sim q$$
$$\sim (\sim p) \equiv p$$

(1) Apply Distributive and commutative law

$p \wedge (q \vee r) \equiv (p \wedge q) \vee (p \wedge r)$
$p \vee (q \wedge r) \equiv (p \vee q) \wedge (p \vee r)$ } Distributive.

$p \wedge q \equiv q \wedge p$
$p \vee q \equiv p \wedge q$ } commutative

Example :- $A \leftrightarrow (B \vee C)$

Step 1 : Remove Biconditional

$[A \rightarrow (B \vee C)] \wedge [(B \vee C) \rightarrow A]$

Step 2 : Remove Implication

$[\sim A \vee (B \vee C)] \wedge [\sim (B \vee C) \vee A]$

Step 3 : Use Negation

$[\sim A \vee (B \vee C)] \wedge [(\sim B \wedge \sim C) \vee A]$

Step 4 : Apply Distributive law

$[\sim A \vee B \vee C] \wedge [(\sim B \vee A) \wedge (\sim C \vee A)]$

$(\sim A \vee B \vee C) \wedge (\sim B \vee A) \wedge (\sim C \vee A)$ //

Q16] Write a PROLOG program for factorial of given number.

→

```
run :- write ('This is a PROLOG program, to
              find factorial of a number'),
       start.

start :- nl, nl, write ('Please enter the number
                         X = '), read (X),
         entered (X, Result),
         write ('The factorial of the number is '),
         write (Result), nl.

entered (0, 1).
entered (X, Result) :-
    X > 0,
    X1 is X - 1,
    entered (X1, SubResult),
    Result is SubResult * X.
```

Output:-

This is a Prolog program to find factorial of a nu
Please enter the number X = 5.
The factorial of the number is 120