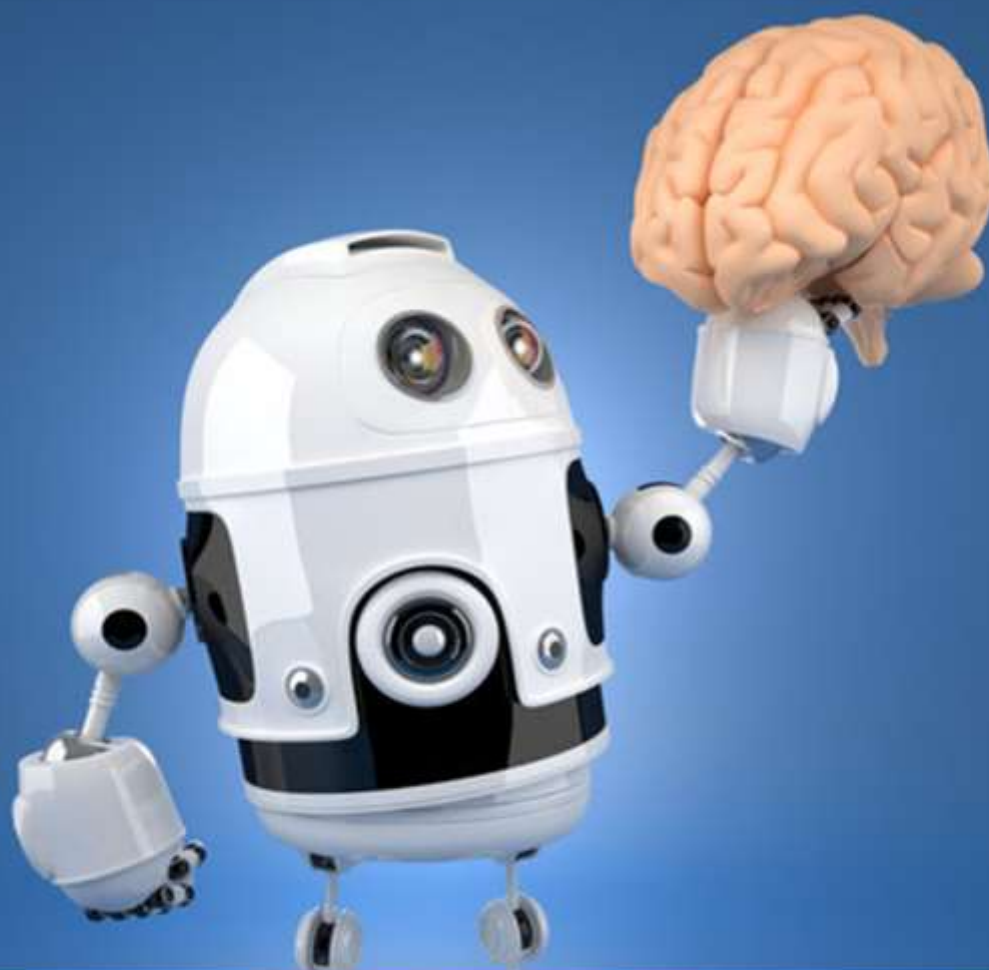


TOPPER'S SOLUTIONS

....In Search of Another Topper



ARTIFICIAL INTELLIGENCE & SOFT COMPUTING (BE - COMPUTER)

7 SEM

As per Revised Syllabus w.e.f 2019-20

Aug 2020 Edition

TOPPER'S SOLUTIONS

...In Search of Another Topper

There are many existing paper solution available in market, but Topper's Solution is the one which students will always prefer if they refer... ;) Topper's Solutions is not just paper solutions, it includes many other important questions which are important from examination point of view. Topper's Solutions are the solution written by the Toppers for the students to be the upcoming Topper of the Semester.

It has been said that **"Action Speaks Louder than Words"** So Topper's Solutions Team works on same principle. Diagrammatic representation of answer is considered to be easy & quicker to understand. So our major focus is on diagrams & representation how answers should be answered in examinations.

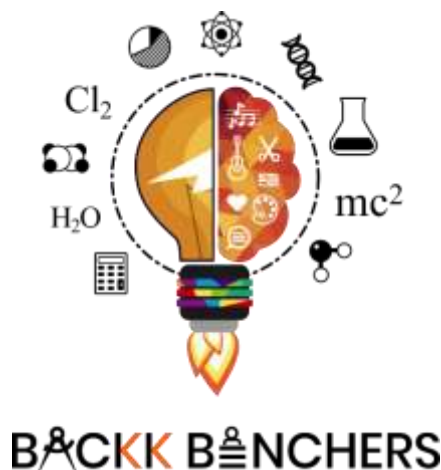
Why Topper's Solutions:

- ❖ Point wise answers which are easy to understand & remember.
- ❖ Diagrammatic representation for better understanding.
- ❖ Additional important questions from university exams point of view.
- ❖ Covers almost every important question.
- ❖ In search of another topper.

"Education is Free.... But its Technology used & Efforts utilized which we charge"

It takes lot of efforts for searching out each & every question and transforming it into Short & Simple Language. Entire Community is working out for betterment of students, do help us.

Thanks for Purchasing & Best Luck for Exams



♥ Handcrafted by BackkBenchers Community ♥

**EVERY MORNING YOU HAVE TWO CHOICES;
CONTINUE TO SLEEP WITH YOUR DREAMS
OR WAKE UP AND CHASE THEM.**

---- By Anonymous.

Syllabus:

Exam	TT-1	TT-2	AVG	Term Work	Oral/Practical	End of Exam	Total
Marks	20	20	20	25	25	80	150

#	Module	Details Contents	No.
1.	Introduction to Artificial Intelligence(AI) and Soft Computing	<ul style="list-style-type: none"> Introduction and Definition of Artificial Intelligence. Intelligent Agents: Agents and Environments, Rationality, Nature of Environment, Structure of Agent, types of Agent. Soft Computing: Introduction of soft computing, soft computing vs. hard computing, various types of soft computing techniques. 	01
2.	Problem Solving	<ul style="list-style-type: none"> Problem Solving Agent, Formulating Problems, Example Problems. Uninformed Search Methods: Depth Limited Search, Depth First Iterative Deepening (DFID), Informed Search Method: A* Search. Optimization Problems: Hill climbing Search, Simulated annealing, Genetic algorithm. 	15
3.	Knowledge, Reasoning and Planning	<ul style="list-style-type: none"> Knowledge based agents First order logic: syntax and Semantic, Knowledge Engineering in FOL Inference in FOL: Unification, Forward Chaining, Backward Chaining and Resolution. Planning Agent, Types of Planning: Partial Order, Hierarchical Order, Conditional Order. 	47
4.	Fuzzy Logic	<ul style="list-style-type: none"> Introduction to Fuzzy Set: Fuzzy set theory, Fuzzy set versus crisp set, Crisp relation & fuzzy relations, membership functions. Fuzzy Logic: Fuzzy Logic basics, Fuzzy Rules and Fuzzy Reasoning Fuzzy inference systems: Fuzzification of input variables, defuzzification and fuzzy controllers. 	68
5.	Artificial Neural Network	<ul style="list-style-type: none"> Introduction – Fundamental concept– Basic Models of Artificial Neural Networks – Important Terminologies of ANNs – McCulloch-Pitts Neuron. Neural Network Architecture: Perceptron, Single layer Feed Forward ANN, Multilayer Feed Forward ANN, Activation functions, Supervised Learning: Delta learning rule, Back Propagation algorithm. Un-Supervised Learning algorithm: Self Organizing Maps. 	85
6.	Expert System	<ul style="list-style-type: none"> Hybrid Approach - Fuzzy Neural Systems Expert system: Introduction, Characteristics, Architecture, Stages in the development of expert system. 	109



This E-Book is Published Specially for **Last Moment Tutions** Viewers

For Video Lectures visit: <https://lastmomenttutions.com/>

CHAP - 1: INTRODUCTION TO ARTIFICIAL INTELLIGENCE (AI) AND SOFT COMPUTING

Q1. Differentiate between Soft & Hard computing

Ans

[5M | Dec-19]

DISTINGUISH BETWEEN SOFT COMPUTING & HARD COMPUTING:

Table 1.1: Soft Computing v/s Hard Computing

Soft Computing	Hard Computing
It is also known as Computational Intelligence .	It is also known as Conventional Computing .
Soft Computing can evolve its own programs.	Hard Computing requires programs to be written.
It uses multivalued or fuzzy logic.	It uses two-valued logic.
It can deal with ambiguous and noisy data.	It requires exact input data.
It allows parallel computations	It is strictly sequential.
It gives approximate answers.	It gives precise answer.
It is deterministic.	It is stochastic.
It uses soft constraints.	It uses real – time constraints.
It is useful for routine tasks that are not critical.	It is useful in critical systems.
Need of robustness rather than accuracy.	Need of accuracy & precision in calculations and outcomes.

Q2. State PEAS Description for Online English Tutor

Ans

[5M | Dec-19]

PEAS:

1. PEAS stands for **Performance, Environment, Actuators, and Sensors**.
2. Based on these properties of an agent, they can be grouped together or can be differentiated from each other.
3. Each agent has these following properties defines for it.
 - a. **Performance Measure (P):** It specifies the performance expected by the agent.
 - b. **Environment (E):** It specifies the surrounding condition where the agent has to perform a task.
 - c. **Actuators (A):** It specifies the tool available for the agent to complete the task.
 - d. **Sensors (S):** It specifies the tool required to sense the work environment.

PEAS DESCRIPTION FOR ONLINE ENGLISH TUTOR:

1. **Performance Measure (P):** Maximize student's score on test.
2. **Environment (E):** Set of students.
3. **Actuators (A):** Screen display (exercises, suggestions, corrections).
4. **Sensors (S):** Keyboard, Typed Words.

-- EXTRA QUESTIONS --**Q1. Define AI. What are applications of AI?****Ans:** [P | Medium]**AI:**

1. AI Stands for **Artificial Intelligence**.
2. The term AI was coined by John McCarthy in 1956.
3. **John McCarthy** defines AI as "the branch of computer science that aims to create intelligent machines."
4. AI is usually the science of making computers to do things that require intelligence when done by humans.

GOALS OF AI:

1. **To Create Expert Systems:** The systems which exhibit intelligent behavior, learn, explain, and advice its users is known as expert system.
2. **To Implement Human Intelligence in Machines:** Creating systems that understand, think, learn, and behave like humans.

APPLICATIONS:

1. **Gaming:** AI plays crucial role in strategic games such as chess, poker, tic-tac-toe, etc., where machine can think of large number of possible positions based on heuristic knowledge.
 2. **Natural Language Processing:** It is possible to interact with the computer that understands natural language spoken by humans.
 3. **Expert Systems:** There are some applications which integrate machine, software, and special information to impart reasoning and advising. They provide explanation and advice to the users.
 4. **Vision Systems:** These systems understand, interpret, and comprehend visual input on the computer.
 5. **Speech Recognition:** Some intelligent systems are capable of hearing and comprehending the language in terms of sentences and their meanings while a human talks to it.
 6. **Handwriting Recognition:** The handwriting recognition software reads the text written on paper by a pen or on screen by a stylus
 7. **Intelligent Robots:** Robots are able to perform the tasks given by a human.
-

Q2. Intelligent Agents**Ans:** [P | Low]

1. Intelligence is broadly divided into:
 - a. Natural Intelligence (Human).
 - b. Artificial Intelligence (Machines/Robots).
2. Natural Intelligence is further divided into:
 - a. Thought Process.
 - b. Action.

3. Artificial Intelligence is further divided into:
 - a. Think.
 - b. Action.
4. **Intelligence System** is the system which thinks & acts like a **human** as well as the system which thinks & acts **rationally**.

DEFINITIONS OF INTELLIGENT AGENT:

1. "The art of creating machines that performs functions that require intelligence when performed by people."
2. "To study of how to make computers do things at which, at the moment people are better."
3. "The automation of activities that we associate with human thinking, activities such as decision making, problem solving, learning."
4. "The branch of the computer science that is concerned with the automation of intelligent behavior."

Q3. Draw and explain the basic building blocks of Learning Agent

Ans: **[P | High]**

LEARNING AGENT:

1. Learning can be considered as the result of interaction between the agent and the world.
2. Turing proposed a new method to design the agent is to **build learning machines** and then **to teach them**.
3. A learning agent receives the percepts from its working environment.
4. It then analyses the feedback.
5. Later on it refines its actions with the help of the feedback received.
6. In Learning Agent, Learning refers to **improving initial knowledge** by operating in unknown environment.

LEARNING AGENT MODEL:

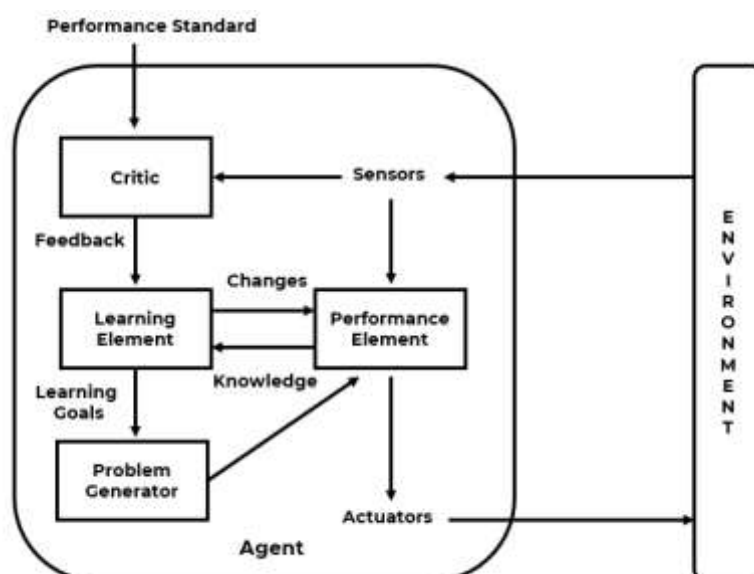


Figure 1.1: Learning Agent Model.

Figure 1.1 shows Model of Learning Agent. Learning Agent consists of four measure components.

I) Performance Element:

1. Performance Element is an agent in itself.
2. Performance Element takes the percept and decides an action.
3. It has **condition-action rules**.
4. This rules can be changed by Learning Element depending on the experience and feedback from Critic.

II) Critic:

1. Critic provides **feedback** to the learning element.
2. It determines how the performance element should be modified to do better in the future.
3. Critic tells the learning element how well the agent is doing with respect to a fixed performance standard.
4. It guides the agent from taking wrong steps in future by comparing its performance with the percepts.

III) Learning Element:

1. It is consider as the **heart of Learning Agent**.
2. Learning Element is responsible for making improvements.
3. It learns and adapts to the changing needs of the environment.
4. Learning Element can modify condition-action rules of the agent.
5. It **guides the problem generator**.

IV) Problem Generator:

1. Problem Generator is responsible for **suggesting actions** that will lead to new and informative experiences.
2. It generates problems to explore the world.

EXAMPLE:

Automated Taxi Driver:

1. When taxi goes out on the road, the critic element observes the world and passes information to the learning element.
 2. If taxi makes a quick left turn across three lanes of traffic, the critic observes the reaction of another driver.
 3. From this experience, the learning element is able to formulate a rule saying this was a bad action, and the performance element is modified by installing the new rule.
 4. Problem generator might identify certain areas of behavior in need of improvements and suggest experiments, such as trying out the brakes on different road surfaces under different conditions.
-

Q4. Define Rationality and Rational Agent. Give an example of rational action performed by any intelligent agent.

Ans: [P | Medium]

RATIONALITY:

1. Rationality is nothing but status of being reasonable, sensible, and having good sense of judgment.
2. Rationality implies the conformity of one's beliefs with one's reasons to believe, or of one's actions with one's reasons for action.
3. It is concerned with expected actions and results depending upon what the agent has perceived.
4. Performing actions with the **aim of obtaining useful information** is an important part of rationality.

RATIONAL AGENT:

1. A rational agent is an agent which has **clear preferences**.
2. It models uncertainty via expected values.
3. A rational agent can be anything that makes decisions, typically a person, firm, machine, or software.
4. A rational agent always performs right action, where the right action means the action that causes the agent to be most successful in the given percept sequence.
5. Rational agent is capable of **taking best possible action** in any situation.

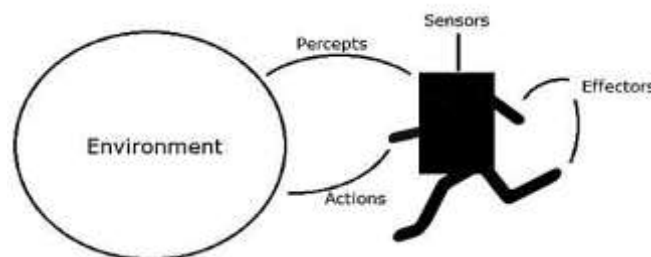


Figure 1.2: Agents Interact With Environment.

EXAMPLE OF RATIONAL ACTION PERFORMED BY ANY INTELLIGENT AGENT:

Automated Taxi Driver:

1. **Performance Measure:** Safe, fast, legal, comfortable trip, maximize profits.
2. **Environment:** Roads, other traffic, customers.
3. **Actuators:** Steering wheel, accelerator, brake, signal, horn.
4. **Sensors:** Cameras, sonar, speedometer, GPS, odometer, engine sensors, keyboard.

Q5. Draw and describe the architecture of goal based agent.

Ans: [P | High]

GOAL BASED AGENTS:

1. Goal Based Agent is one of the type of **Agent Program**.
2. Goal Based Agent expand the capabilities of Model Based Agent by using **Goal information**.
3. Goal Based Agent maintains a Goal or Destination information.

4. Goal information describes situations that are desirable.
5. This allows the agent a way to choose among multiple possibilities, selecting the one which reaches the goal state.
6. Searching and planning are used to achieve the agent's goal.
7. Goal Based Agent is based on **Reflex**.
8. It uses **Condition - Action Rule**.
9. It keeps the **track of world**.
10. Figure 1.3 shows the architecture of goal based agent.

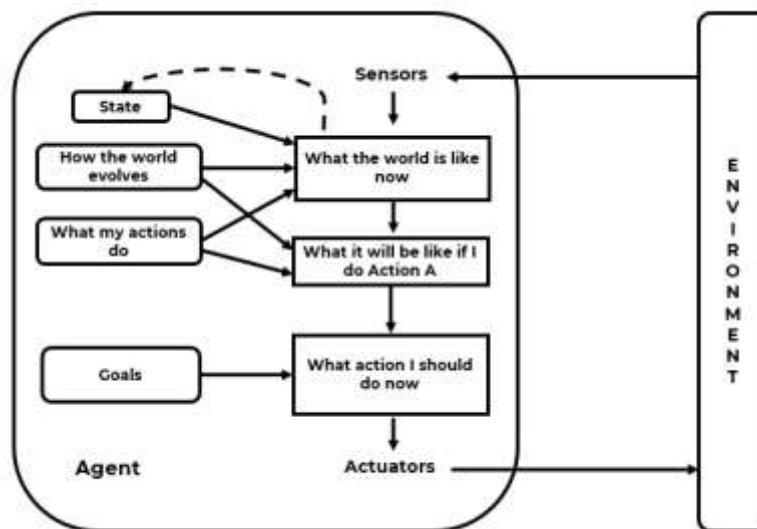


Figure 1.3: Goal Based Agent.

EXAMPLE:

1. At a road junction, the taxi can go left, turn right, or go straight.
2. The correct decision depends on where the taxi is trying to get to (i.e. destination or goal).
3. Such agent differs from the reflex agent because in reflex agent we are having static table of condition - action and in the model based it involves consideration of the future status – “What will happen if I do such – and – such?”
4. Goal based agent appears less efficient but more flexible due to its dynamic nature.
5. If it starts to rain, the agent can update its knowledge of how effectively its brakes will operate.

Q6. Explain Model-Based Reflex Agents

Ans:

[P | Medium]

MODEL-BASED REFLEX AGENT:

1. Model Based Agent keeps track of world.
2. It is based on reflex.
3. It also uses Condition-Action Rule.
4. It can work in Partial Observable Environment.
5. It maintains some sort of internal state that depends on the percept history and thereby reflects at least some of the unobserved aspects of the current state.

6. Updating the internal state information as time goes by requires two types of knowledge to be encoded in the agent program.
 - a. How the world evolves independently of the agent.
 - b. **For example:** Overtaking car will be closer behind than it was a moment ago.
 - c. How the agent's own actions affect the world.
 - d. **For example:** When the agent turns the steering wheel clockwise, the car turns to the right.
7. Such knowledge if implemented in scientific theory or Boolean circuits is called model and the agent that uses such a model is called a model-based agent.

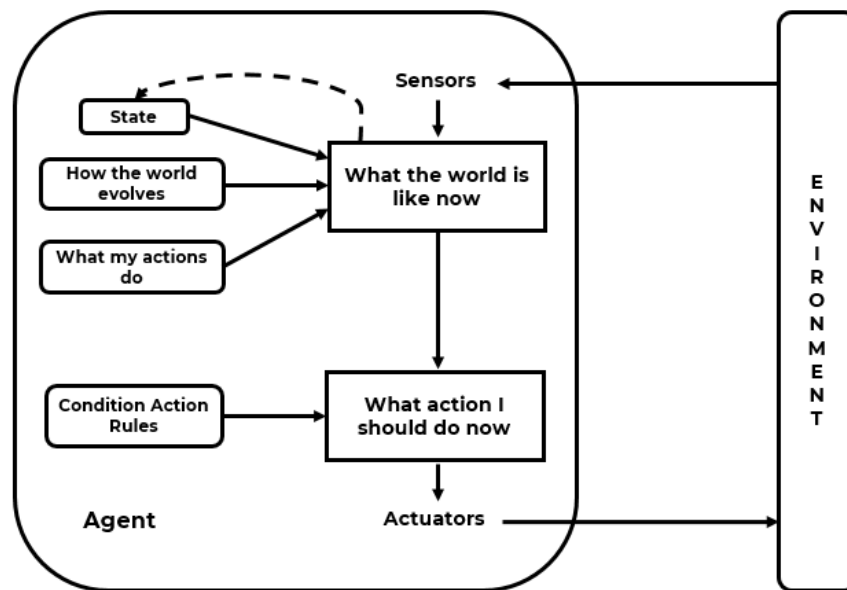


Figure 1.4: Model Based Reflex Agent.

Q7. Explain Simple Reflex Agent

Ans:

[P | Medium]

SIMPLE REFLEX AGENT:

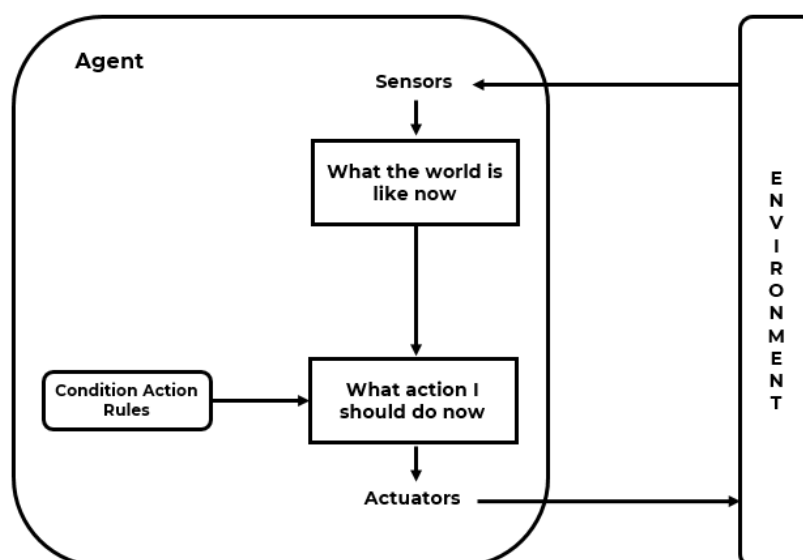


Figure 1.5: Simple Reflex Agent

1. This is improved agent compared to Table Driven Agent.
2. It uses Condition-Action rule instead of Percept-Action.
3. Such agents select action on the basis of the current percept, ignoring the rest of the percept history.
4. Having admirable property of being simple, but they turn out to be of very limited intelligent.
5. Suitable only when environment is fully observable.
6. **Example:** If car in front is braking **then** initiate- braking.
7. Figure 1.5 shows the schematic of the simple reflex agent.

Q8. Draw and Describe the Architecture of Utility based agent.

Ans:

[P | High]

UTILITY BASED AGENT:

1. Goal-based agents only distinguish between goal states and non-goal states.
2. It is possible to define a measure of how desirable a particular state is.
3. This measure can be obtained through the use of a utility function.
4. Utility based agent choose actions based on a **preference (utility)** for each state.
5. Such agents try to maximize the performance with high quality behavior.
6. Performance measure = Degree of happiness.
7. Utility based agent **keeps the track of world**.
8. It can also work in **Partial Observable Environment**.
9. **Example:** There can be multiple action sequences that will get the taxi to its destination but some are quicker, safer, more reliable, or cheaper than others.
10. Utility agent examines “how happy I will be in such a state?”
11. Figure 1.6 present utility based agent.

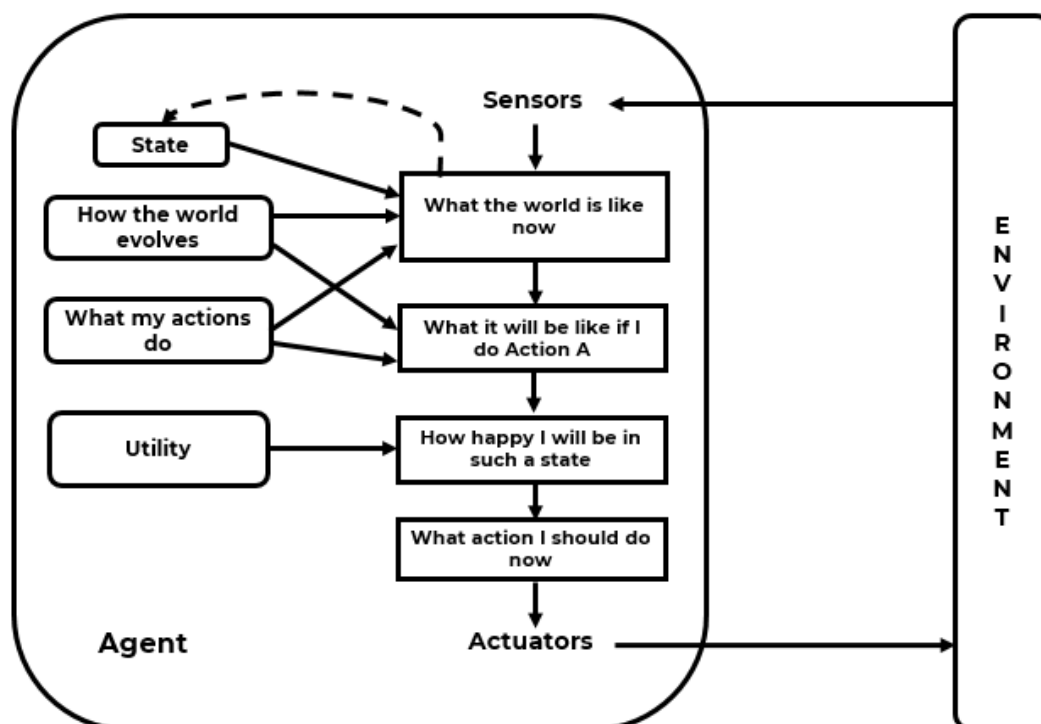


Figure 1.6: Utility Based Agent.

Q9. Explain Types of Agents**Ans:****[P | Medium]****I) TABLE DRIVEN AGENT:**

1. It is the simplest type of Agent.
2. It uses Look-up table for percept & action mapping.
3. It is not suitable for real life problems.
4. Example: Chess playing Agents which has total combination of 35100
5. It is difficult & tedious to prepare Look-up table.
6. It requires large memory space to store table.
7. An agent has no Autonomy or Intelligence.

II) SIMPLE REFLEX AGENTS:

Refer Q7

III) MODEL BASED REFLEX AGENTS:

Refer Q6

IV) GOAL BASED AGENTS:

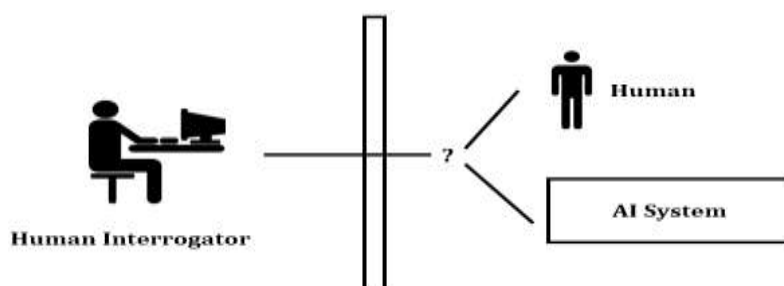
Refer Q5.

V) UTILITY BASED AGENTS:

Refer Q8.

Q10. Explain Turing Test designed for satisfactory operational definition of intelligence.**Ans:****[P | Medium]****TURING TEST:**

1. Turing Test was proposed by **Alan Turing in 1950**.
2. It was designed to provide a satisfactory operational definition of intelligence.
3. It measures the performance of intelligent machine against that of human being.
4. The computer possess the test if a human interrogator, after posing some written questions, cannot tell whether the written responses come from a person or not.
5. Therefore Turing Test can be considered as the art of creating machines that performs functions that require intelligence when performed by people.
6. Figure 1.7 shows the example of Turing Test.

**Figure 1.7: Turing Test Example.**

TO PASS A TURING TEST, A COMPUTER MUST HAVE FOLLOWING CAPABILITIES:

1. **Natural Language Processing (NLP)**: Must be able to communicate successfully in English.
2. **Knowledge Representation**: To store what it knows and hears. .
3. **Automated Reasoning**: Answer the Questions based on the stored information.
4. **Machine Learning**: Must be able to adapt in new circumstances.

PROBLEM:

Turing test is not reproducible, constructive, or amenable to mathematical analysis.

Q11. Define Intelligent Agent. What are the characteristics of Intelligent Agent?**Ans:****[P |Medium]****INTELLIGENT AGENT:**

1. An intelligent agent is a **component of artificial intelligence**.
2. An intelligent agent is an autonomous entity which observes through sensors and acts upon an environment using actuators and directs its activity towards achieving goals.
3. Intelligent agents may also learn or use knowledge to achieve their goals.
4. They may be very simple or very complex.
5. **Siri & Alexa** are the example of intelligent agent.

SOME DEFINITIONS OF INTELLIGENT AGENT:

1. "The art of creating machines that performs functions that require intelligence when performed by people."
2. "To study of how to make computers do things at which, at the moment people are better."
3. "The automation of activities that we associate with human thinking, activities such as decision making, problem solving, learning."
4. "The branch of the computer science that is concerned with the automation of intelligent behavior."

CHARACTERISTICS OF INTELLIGENT AGENT:**Internal characteristics are as follows:**

1. **Learning/reasoning**: An agent has the ability to learn from previous experience and to successively adapt its own behavior to the environment.
2. **Reactivity**: An agent must be capable of reacting appropriately to information from its environment.
3. **Autonomy**: An agent must have both control over its actions and internal states.
4. **Goal-oriented**: An agent has well-defined goals and gradually influence its environment.

External characteristics are as follows:

1. **Communication**: An agent often requires an interaction with its environment to fulfill its tasks, such as human, other agents, and arbitrary information sources.

2. **Cooperation:** Cooperation of several agents permits faster and better solutions for complex tasks that exceed the capabilities of a single agent.
3. **Mobility:** An agent may navigate within electronic communication networks.
4. **Character:** Like human, an agent may demonstrate an external behavior with many human characters as possible.

Q12. What are PEAS descriptors? Give PEAS descriptors for a robot meant for cleaning the house.

Ans: [P | Low]

PEAS:

1. PEAS stands for **Performance, Environment, Actuators, and Sensors**.
2. Based on these properties of an agent, they can be grouped together or can be differentiated from each other.
3. Each agent has these following properties defines for it.
 - a. **Performance Measure (P):** It specifies the performance expected by the agent.
 - b. **Environment (E):** It specifies the surrounding condition where the agent has to perform a task.
 - c. **Actuators (A):** It specifies the tool available for the agent to complete the task.
 - d. **Sensors (S):** It specifies the tool required to sense the work environment.

PEAS FOR ROBOT MEANT FOR CLEANING THE HOUSE:

1. **Performance Measure (P):** Maximize energy consumption, Maximize Dirt Pick up, Percentage of precision of cleaning.
2. **Environment (E):** House, Dirt distribution unknown, assume actions are **deterministic** and environment is **static**.
3. **Actuators (A):** Jointed Arm and hand, View Detector for Robot, Left, Right, Suck and NoOp.
4. **Sensors (S):** Camera, Orientation & Touch Sensors, Sensors to identify the dirt and Potentiometric Sensor.

ENVIRONMENT:

Partially Observable, Stochastic, Sequential, Dynamic, Continuous and Multi Agent.

Q13. Give PEAS description for an Autonomous Mars Rover. Characterize its environment

Ans: [P | Low]

PEAS FOR AUTONOMOUS MARS ROVER:

1. **Performance Measure (P):** Terrain explored and reported, samples gathered and analyzed.
2. **Environment (E):** Launch vehicle, lander and Mars.
3. **Actuators (A):** Wheels/Legs, sample collection device, analysis devices and radio transmitter.
4. **Sensors (S):** Camera, Touch Sensors, Accelerometers, Orientation Sensors, Wheels/Joint Encoders and Radio Receiver.

ENVIRONMENT:

Partially Observable, Stochastic, Sequential, Dynamic, Continuous and Single Agent.

Q14. Give PEAS description for a Robot Soccer Player. Characterize its environment.

Ans: [P | Low]

PEAS FOR ROBOT SOCCER PLAYER:

1. **Performance Measure (P):** To Play, Make Goal & Win the Game.
2. **Environment (E):** Soccer, Team Members, Opponents, Referee, Audience and Soccer Field.
3. **Actuators (A):** Navigator, Legs of Robot, View Detector for Robot.
4. **Sensors (S):** Camera, Communicators and Orientation & Touch Sensors.

ENVIRONMENT:

Partially Observable, Stochastic, Sequential, Dynamic, Continuous and Multi Agent.

Q15. Dimensions/ Properties of Environment?

Ans: [P | Low]

Used to categorize the environment and should be considered while designing particular agent.

I) Fully Observable vs. Partially Observable:**1. Fully Observable:**

- a. Entire information can be accessed using sensors.
- b. Easy to work for agent.
- c. No need to keep track of world.

2. Partially Observable:

- a. Entire information is not available.
- b. Difficult to work for Agent.
- c. Agent needs to keep track of world.

3. Example: Automated taxi cannot see what other drivers are thinking.**II) Deterministic vs. Stochastic:**

1. **Deterministic:** Next state can be uniquely defined if current state and action or input is known.
2. **Stochastic:** Next state can't be uniquely defined. Hence difficult to write.
3. Taxi driving is clearly stochastic.

III) Episodic vs. Sequential:

1. In an episodic environment, the agent's experience is divided into atomic episodes. Each episode consists of the perceiving and then performing a single action.
2. In episodic environment, the choice of action in each episode depends only on the episode itself.
3. In sequential environment, on the other hand, the current decision could affect all future decisions.

4. **Example:** Detecting defective parts on an assembly line is episodic task while chess and taxi driving are the sequential tasks.

IV) Static vs. Dynamic:

1. Static:

- If the environment cannot change while an agent is deliberating, then we say the environment is static.
- Static environment is easy to deal with.

2. Dynamic:

- If the environment can change while an agent is deliberating, then we say the environment is dynamic.
- Difficult to work for agent.
- Action may become redundant.

3. **Example:** Taxi driving is clearly dynamic. Crossword puzzles are static.

V) Discrete vs. Continuous:

- Such distinction is applied to the environment according to the way in which time is handled.
- Example:** Chess game is discrete-time and taxi driving is continuous.

VI) Single Agent vs. Multi Agent:

- Depends upon no. of agents present in the environment.
- An agent solving a crossword puzzle by itself is clearly in a single-agent environment, whereas an agent playing chess is two-agent environment.

Q16. Compare and Contrast problem solving agent and planning agent.

Ans: [P | High]

Table 1.2: Problem Solving Agent v/s Planning Agent

Problem Solving Agent	Planning Agent
Problem Solving Agent is one kind of Goal Based Agent .	Planning Agent is the combination of Problem Solving Agents & Knowledge-based Agents .
Problem Solving Agent decides what to do by finding sequences of actions that lead to desirable states.	Planning Agent constructs the plans that achieve its goals, and then executes them.
Problem Solving Agent does not use domain-independent heuristic function .	Planning Agent use domain-independent heuristic function .
Problem Solving Agent demonstrate one specific solution.	Planning Agent can be viewed as the producer or generator of the solution.
Problem Solving Agent is concerned with plan execution .	Planning Agent is concerned with plan generation .
Limitations of the Problem Solving Approach motivates the design of planning systems.	Limitations of the Planning Approach does not motivates the design of Problem Solving systems.
Example: Vacuum World.	Example: STRIPS

Q17. Compare Model based Agent with Utility based Agent.**Ans:****[P | Medium]****Table 1.3: Model Based Agent v/s Utility Based Agent**

Model Based Agent	Utility Based Agent
Model Based Agent choose actions based on model of the world.	Utility Based Agent choose actions based on preference (utility) for each state.
Model Based Agent does not examines "how happy I will be in such a state?"	Utility Based Agent examines "how happy I will be in such a state?"
It does not use utility function.	It uses utility function.
It is Reflex Agent.	It is not a Reflex Agent.
It uses condition action rule.	It does not condition action rule.
Diagram: Refer Figure 1.4	Diagram: Refer Figure 1.6

Q18. Define Soft Computing? Explain Applications of Soft Computing?**Ans:****[P | High]****SOFT COMPUTING:**

1. Soft Computing is sometimes referred as **Computational Intelligence**.
2. The main goal of soft computing is to develop **intelligence machines** to provide solutions to **real world problems**, which are not modeled or too difficult to model mathematically.
3. It is an emerging approach to the computing.
4. The Role Model for Soft Computing is **Human Mind**.

DEFINITION:

Soft Computing is an Computational Intelligence which uses the remarkable ability of human mind to **reason & learn** in an environment of uncertainty & imprecision.

SOFT COMPUTING PARADIGMS:

1. **Fuzzy Set:** For Knowledge Representation via Fuzzy IF – Then Rule.
2. **Neural Networks:** For learning & adaptation.
3. **Genetic Algorithms:** For Evolutionary Computation.

APPLICATIONS:

1. Hand writing recognition.
2. Image processing & Data compression.
3. Power systems.
4. Fuzzy Logic Control.
5. Speech & Vision Recognition Systems.
6. Machine Learning Applications.

CHAP - 2: PROBLEM SOLVING

Q1. Explain Hill-Climbing algorithm.

Ans:

[5M | Dec-19]

HILL CLIMBING ALGORITHM:

1. Hill climbing algorithm is a **local search algorithm** which continuously moves in the direction of increasing elevation/value to find the peak of the mountain or best solution to the problem.
2. It terminates when it reaches a peak value where no neighbor has a higher value.
3. Hill climbing algorithm is a technique which is used for optimizing the mathematical problems.
4. One of the widely discussed examples of Hill climbing algorithm is **Traveling-salesman Problem** in which we need to minimize the distance traveled by the salesman.
5. It is also called greedy local search as it only looks to its good immediate neighbor state and not beyond that.
6. A node of hill climbing algorithm has two components which are **state and value**.
7. Hill Climbing is mostly used when a good heuristic is available.
8. In this algorithm, we don't need to maintain and handle the search tree or graph as it only keeps a single current state.

FEATURES OF HILL CLIMBING:

1. **Generate and Test variant:** Hill Climbing is the variant of Generate and Test method. The Generate and Test method produce feedback which helps to decide which direction to move in the search space.
2. **Greedy approach:** Hill-climbing algorithm search moves in the direction which optimizes the cost.
3. **No backtracking:** It does not backtrack the search space, as it does not remember the previous states.

STATE-SPACE DIAGRAM FOR HILL CLIMBING:

1. The state-space landscape is a graphical representation of the hill-climbing algorithm which is showing a graph between various states of algorithm and Objective function/Cost.
2. On Y-axis we have taken the function which can be an objective function or cost function, and state-space on the x-axis.
3. If the function on Y-axis is cost then, the goal of search is to find the global minimum and local minimum.
4. If the function of Y-axis is Objective function, then the goal of the search is to find the global maximum and local maximum.

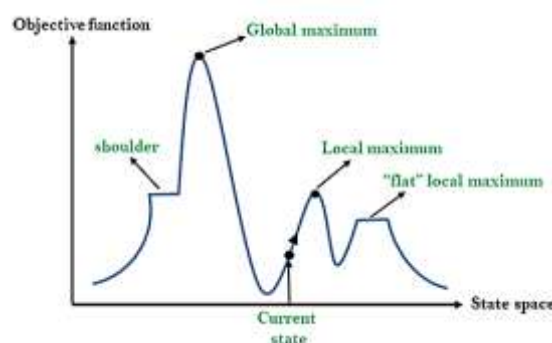
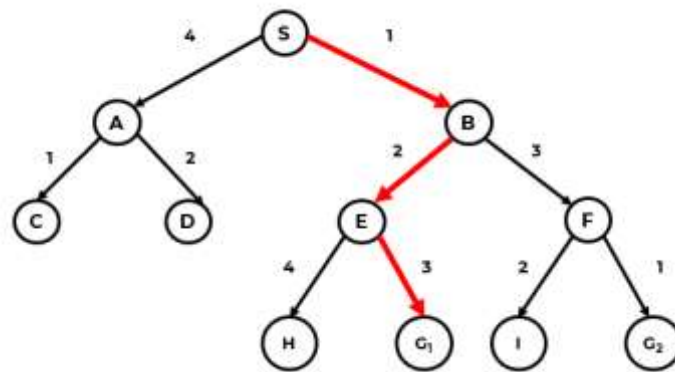


Figure 2.1: Hill Climbing State Space Diagram.

DIFFERENT REGIONS IN THE STATE SPACE LANDSCAPE:

1. **Local Maximum:** Local maximum is a state which is better than its neighbor states, but there is also another state which is higher than it.
2. **Global Maximum:** Global maximum is the best possible state of state space landscape. It has the highest value of objective function.
3. **Current state:** It is a state in a landscape diagram where an agent is currently present.
4. **Flat local maximum:** It is a flat space in the landscape where all the neighbor states of current states have the same value.
5. **Shoulder:** It is a plateau region which has an uphill edge.

EXAMPLE:**Figure 2.2: Hill Climbing Example.**

Consider the Example of Hill Climbing Algorithm shown in Figure 2.2.

Search Path = [S, B, E, G₁]

$S \rightarrow B \rightarrow E \rightarrow G_1$

Cost = $1 + 2 + 3$
= 6

Q2. Write a short note on genetic algorithm.

Ans:

[5M | Dec-19]

GENETIC ALGORITHM:

1. Genetic Algorithms are a part of evolutionary computing, which is a rapidly growing area of Artificial Intelligence.
2. Genetic Algorithms are inspired by **Darwin's theory about evolution**.
3. It is an **intelligent random search technique**.
4. It is used to **solve optimization problem**.
5. Genetic Algorithm use encode solutions as fixed length "bit strings".
6. **Example:** 101110, 111111, 000101
7. Genetic Algorithm works by testing any string and getting a score indicating how good that solution is.
8. The set of all possible solutions [0...1000] is called as search space or state space.

GENETIC ALGORITHM PSEUDO CODE:

```

Generate an initial population of individuals
Evaluate the fitness of all individuals
While termination condition not met do
    Select fitter individuals for reproduction.
    Recombine between individuals
    Mutate individuals
    Evaluate the fitness of the modified individuals
    Generate a new population
End While
  
```

GENETIC ALGORITHM FLOWCHART:

Figure 2.3 shows the flowchart for genetic algorithm.



Figure 2.3: Genetic Algorithm Flowchart.

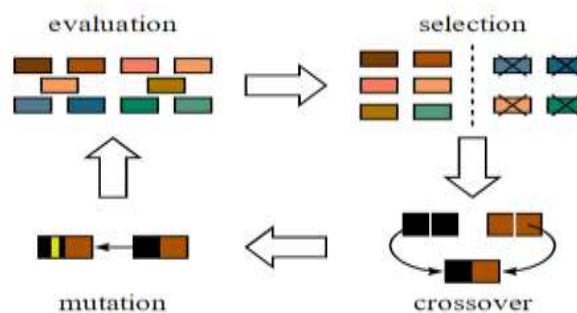
COMPONENTS OF GENETIC ALGORITHM:

Figure 2.4: Genetic Algorithm Components.

1. For example, in Tournament Selection, the algorithm selects an individual with the highest fitness value from a random subset of the population.

2. Genetic Algorithm's evolution operates in three stages.
3. Selection, where it chooses a relatively fit subset of individuals for breeding.
4. Crossover, where it recombines pairs of breeders to create a new population.
5. Mutation, where it potentially modifies portions of new chromosomes to help maintain the overall genetic diversity.
6. Arrows in the figure 2.4 indicate transitions into the next genetic operation within one generation.

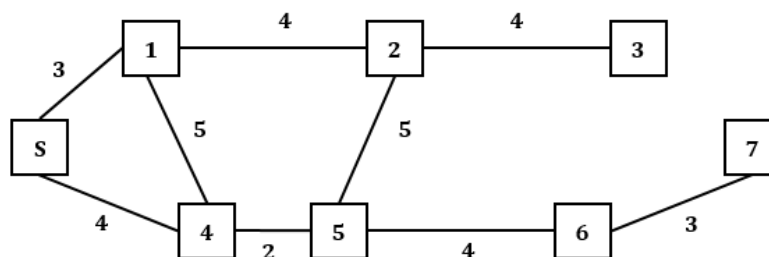
ADVANTAGES:

1. It is easy to understand.
2. Chance of getting optimal solution is higher.

LIMITATIONS OF GENETIC ALGORITHM:

1. Cross over rate should be 80-95%.
2. Fitness function must be accurate.

Q3. Consider the graph given in Figure 1 below. Assume that the-initial state is S and the goal state is 7. Find a path from the initial state to the goal state using A* Search. Also report the solution cost. The straight line distance heuristic estimates for the nodes are as follows: $h(1)=14$, $h(2)=10$, $h(3)=8$, $h(4)=12$, $h(5)=10$, $h(6)=10$, $h(S)=15$



Ans:

[10M | Dec-19]

EXAMPLE:

Consider the figure 2.5 shown below. Here 'S' is the initial state & '7' is the goal state. The heuristic/cost is given along branches & links.

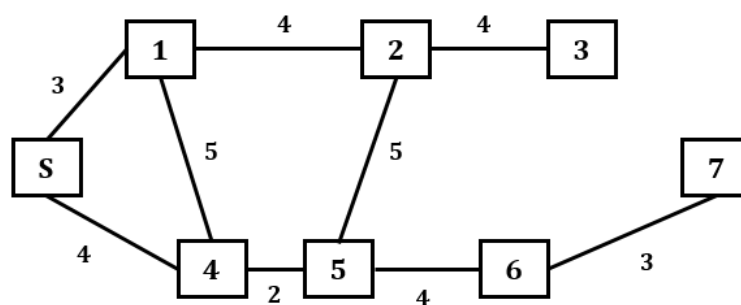


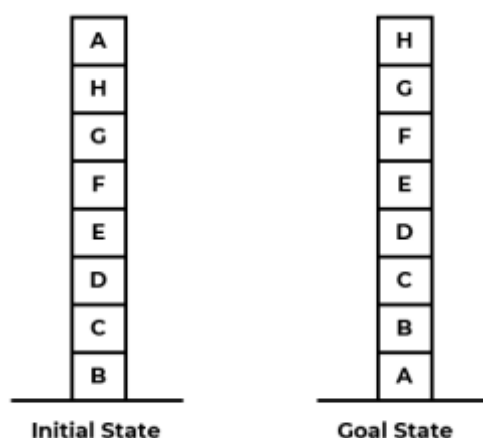
Figure 2.5: Example of A*

Steps:

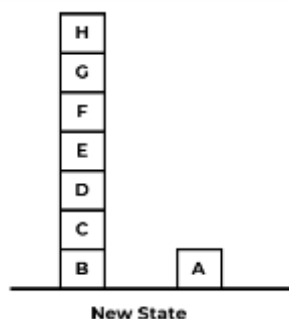
Tree		Cost	Open Queue	Close Queue
1		$F(1) = G(1) + H(1)$ $= 3 + 14$ $= 17$ $F(4) = G(4) + H(4)$ $= 4 + 12$ $= 16$	[S] [1 4]	[S] [S 4]
2		$F(1) = G(1) + H(1)$ $= 5 + 14$ $= 19$ $F(5) = G(5) + H(5)$ $= 2 + 10$ $= 12$	[S] [S 4] [S 4 1 5]	[S] [S 4] [S 4 5]
3		$F(2) = G(2) + H(2)$ $= 5 + 10$ $= 15$ $F(6) = G(6) + H(6)$ $= 4 + 10$ $= 14$	[S] [S 4] [S 4 5] [S 4 5 2 6]	[S] [S 4] [S 4 5] [S 4 5 6]
4		$F(7) = G(7) + H(7)$ $= 3 + 0$ $= 3$	[S] [S 4] [S 4 5] [S 4 5 6] [S 4 5 6 7]	[S] [S 4] [S 4 5] [S 4 5 6] [S 4 5 6 7]
"Success" Hence we have found the Optimal Solution for the problem.				

Optimal Cost = 4 + 2 + 4 + 3 = 13

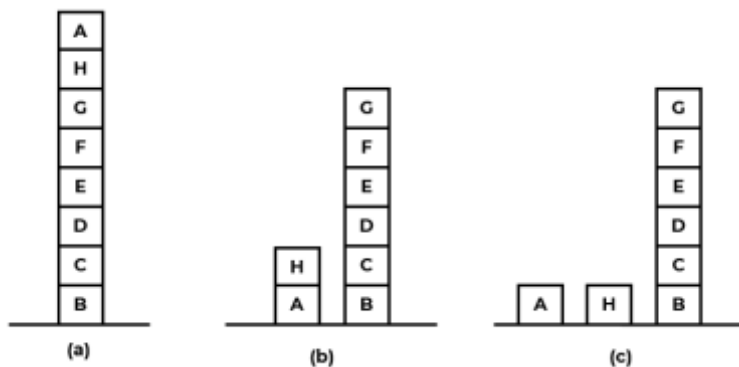
Optimal Path = S → 4 → 5 → 6 → 7

Q4. Give Local and Global heuristic function for block world problem**Ans:****[5M | Dec-19]****Figure 2.6: Block world problem****LOCAL HEURISTIC FUNCTION:**

1. Add one point for every block that is resting on the thing it is supposed to be resting on.
2. Subtract one point for every block that is sitting on the wrong thing.
3. Using this function, the goal state has a score of 8.
4. The initial state has a score of 4 (since it gets one point added for blocks C, D, E, F, G and H and one point subtracted for blocks A & B).
5. There is only one move from the initial state, namely to move block A to the table as shown below.



6. That produces a state with a score of 6 (since now A's position causes a point to be added rather than subtracted).
7. From the new state, there are three possible moves, leading to the three states shown in Figure 2.7.
8. These states have the scores: (a) 4, (b) 4, & (c) 4.

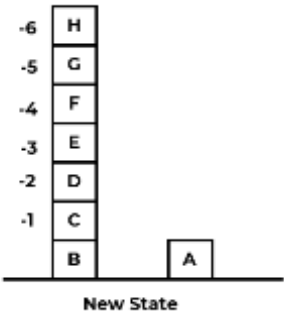


9. All 3 cases a, b, c have no better score than the current state = 6; so we stop.

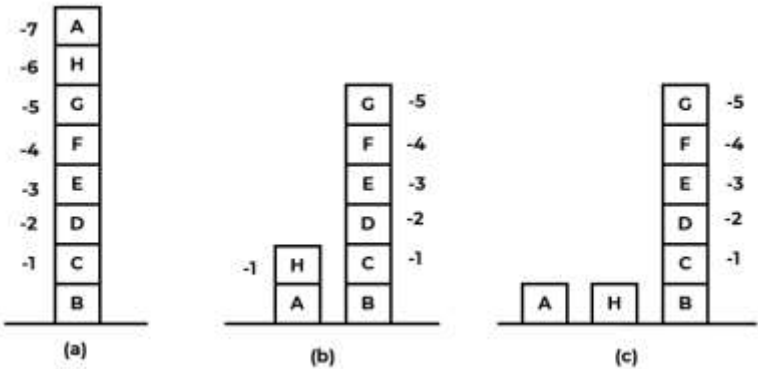
- 10. The process has reached a local maximum that is not the global maximum.
- 11. The problem is that by purely local examination of support structures, the current state appears to be better than any of its successors because more blocks rest on the correct objects.

GLOBAL HEURISTIC FUNCTION:

- 1. For each block that has the correct support structure, add one point for every block in the support structure.
- 2. For each block that has an incorrect support structure, subtract one point for every block in the existing support structure.
- 3. Using this function, the goal state has the score 28.
B = 1, C = 2, D = 3, E = 4, F = 5, G = 6, H = 7
- 4. The initial state has the score is -28.
C = -1, D = -2, E = -3, F = -4, G = -5, H = -6, A = -7
- 5. Moving A to the table yields a state with a score of -21.
- 6. Since A no longer has seven wrong blocks under it.



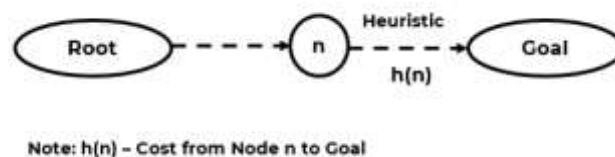
- 7. The three states that can be produced next now have the following scores: (a) -28, (b) -16, and (c) -15.



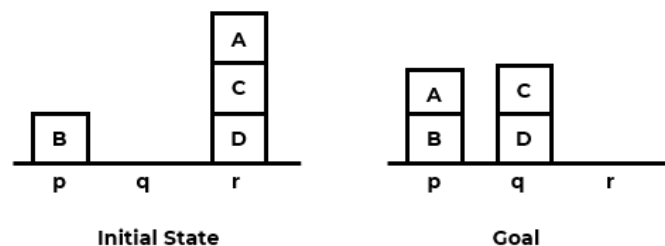
- 8. Here we will choose move (c); Which is the correct one.
- 9. This new heuristic function captures the two key aspects of this problem: incorrect structures are bad and should be taken apart; and correct structures are good and should be built up.

-- EXTRA QUESTIONS --**Q1. Explain Heuristic Function with Example?****Ans:****[P | Medium]****HEURISTIC FUNCTION:**

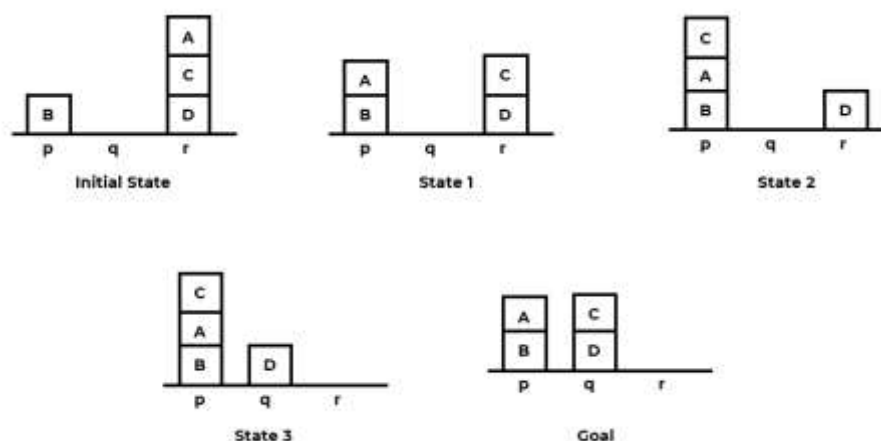
1. Heuristic Function is the function that gives an estimation on the cost of getting from node to the goal state.
2. Heuristic Function is used in **Informed Search Technique**.
3. It is used in a decision process to try to make the best choice of a list of possibilities.
4. Best move is the one with the least cost.
5. Heuristic function helps in **implementing goal oriented search**.
6. It is usually used to increase the efficiency of the search process.
7. Figure 2.7 represents Heuristic Function.

**Figure 2.7: Heuristic Function.****EXAMPLE OF HEURISTICS FUNCTION FOR BLOCKS WORLD PROBLEM:**

Consider the example of Block World Shown Below:



As shown below the heuristic value of $H(s) = 4$. Since we take 4 move to reach from initial state to Goal State.



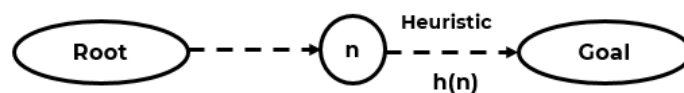
Q2. Define heuristic function. Give an example heuristics function for 8-puzzle problem.

Ans:

[P | Medium]

HEURISTIC FUNCTION:

1. Heuristic Function is the function that gives an estimation on the Cost of getting from node to the goal state.
2. Heuristic Function is used in **Informed Search Technique**.
3. It is used in a decision process to try to make the best choice of a list of possibilities.
4. Best move is the one with the least cost.
5. Heuristic function helps in **implementing goal oriented search**.
6. It is usually used to increase the efficiency of the search process.
7. Figure 2.8 represents Heuristic Function.



Note: $h(n)$ – Cost from Node n to Goal

Figure 2.8: Heuristic Function.

HEURISTICS FUNCTION FOR 8-PUZZLE PROBLEM:

For 8-Puzzle Problem, two heuristics are commonly used. i.e. No. of misplaced tiles and Manhattan Distance.

I) No. of Misplaced tiles:

1. In this case, heuristic function is determined using number of misplaced titles (not including the blank space).

2. For Example:

3. Consider the example given below. It consists of initial state and goal state.

Current State			Goal State		
1	2	3	1	2	3
4	5	6	4	5	6
7		8	7	8	

4. As shown in example, only "8" is misplaced, so the heuristic function evaluates to 1.

Current State			Goal State		
1	2	3	1	2	3
4	5	6	4	5	6
7		8	7	8	

1	2	3
4	5	6
7	8	

\therefore **Heuristic $h(n) = 1$**

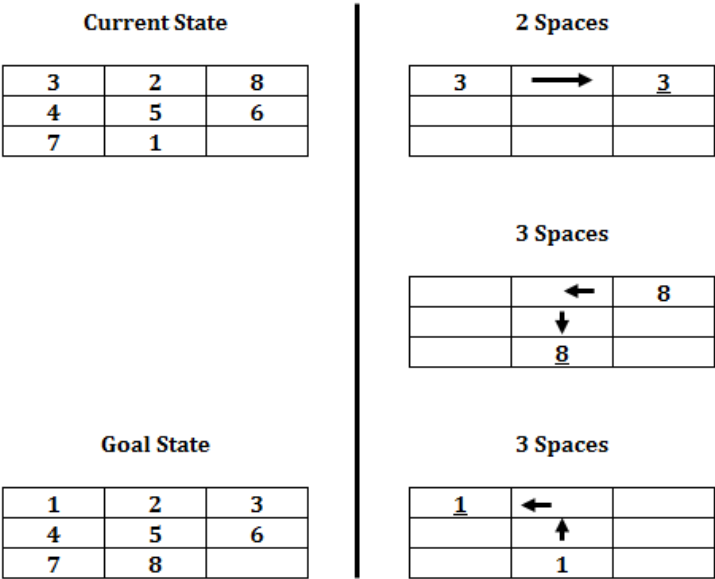
II) Manhattan Distance:

1. Manhattan Distance is the sum of the distances of the tiles from their goal positions.

2. For Example:

3. Consider the example given below. It consists of initial state and goal state.

4. In this case, only the “3”, “8” and “1” tiles are misplaced by 2, 3 and 3 spaces respectively. So the heuristic function evaluates to 8.
5. Therefore, Heuristic $h(n) = 2 + 3 + 3 = 8$



Q3. Design a planning agent for a Blocks World problem. Assume suitable initial state and final state for the problem.

Ans: [P | Medium]

PLANNING AGENT:

1. Planning Agent has Ideal Planner along with knowledge base inside its memory.
2. Agent tells the knowledge base about the percept.
3. But instead of asking the action to knowledge base, it ask the Goal.
4. Once Goal is provided by the knowledge base, it forwards it to Ideal Planner.
5. Ideal Planner prepares a Plan to achieve the Goal.
6. The steps of this plan are then implemented as the action.
7. Figure 2.9 represents the planning agent.

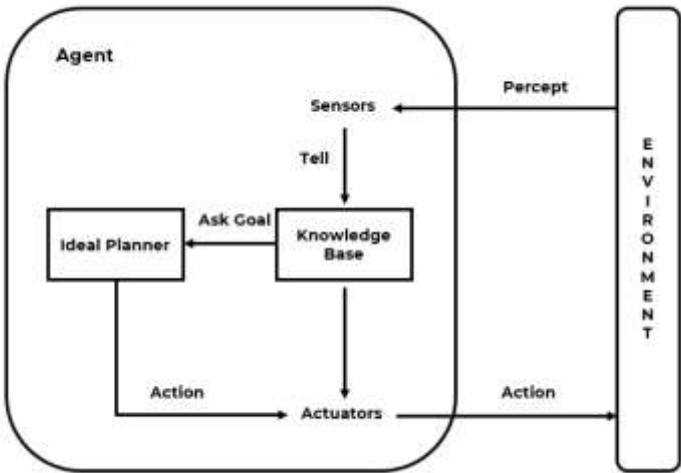


Figure 2.9: Planning Agent.

PLANNING AGENT FOR A BLOCKS WORLD PROBLEM:

Consider the Initial State and Goal State as shown below in figure 2.10 for Block World Problem.

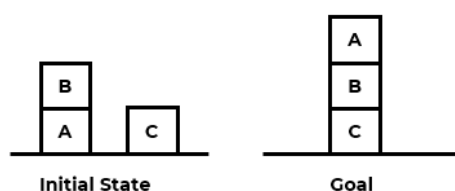


Figure 2.10: Example of Block World Problem.

Planning Agent Component for Block World Problem:

Initial State	Goal	A Plan
Clear (A)	On (B, C)	Pickup (B)
Clear (B)	On (A, B)	Stack (B, C)
Clear (C)		Pickup (A)
onTable (A)		Stack (A, B)
onTable (B)		
onTable (C)		

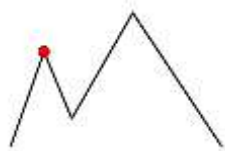
Q4. What are the problems/frustration that occur in hill climbing technique? Illustrate with an example.

Ans:

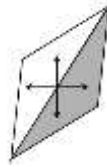
[P | High]

PROBLEMS IN HILL CLIMBING TECHNIQUE:**I) Local Maxima:**

1. Local Maxima is a state that is better than all of its neighbours.
2. But it is not better than some other states far away.
3. Hill Climbing Algorithm tends to find **only local maxima**.
4. This problem can be solved using **Backtracking**.

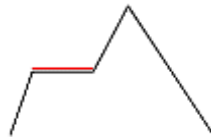
**II) Ridges:**

1. Ridge is a curve in the search space that can **lead to maxima**.
2. The orientation of the high region, compared to the set of available moves, makes it impossible to climb up.
3. However, two moves executed serially may increase the height.
4. Move to the several direction at once can help in dealing with the ridges.



III) Plateau:

1. Plateau is a flat area of the search space in which all neighbouring states have the same value.
2. The heuristic of Plateau region has **same value**.
3. In plateau it is not possible to determine the best direction by using local comparison.
4. Solution to plateau is to take a big jump to any direction to get to a new search space.



EXAMPLE:

Depending on the initial state, Hill-Climbing gets stuck in Local Maxima.

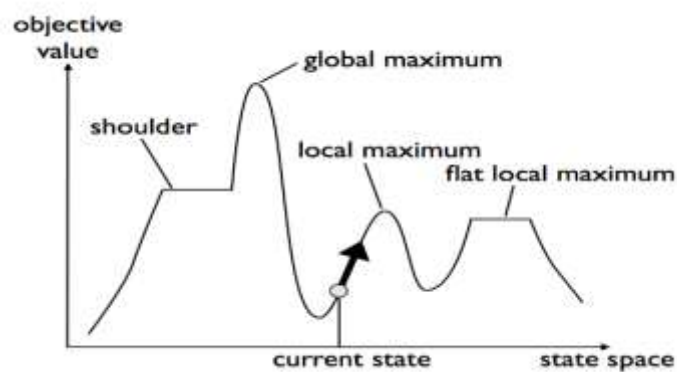


Figure 2.11: Local Maxima.

Consider the example of 8 Puzzle using Hill Climbing. As shown in figure 2.11 it get stuck in Local Maxima. Hence we need to backtrack.

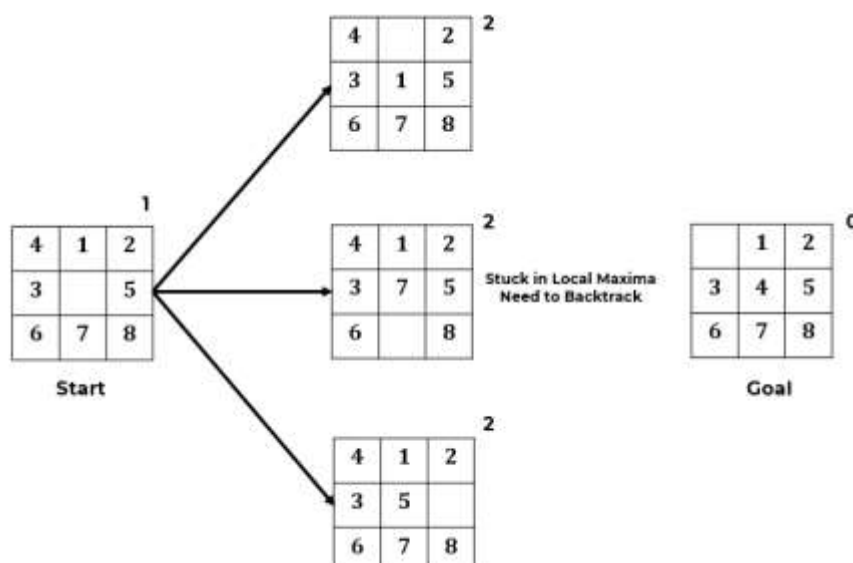


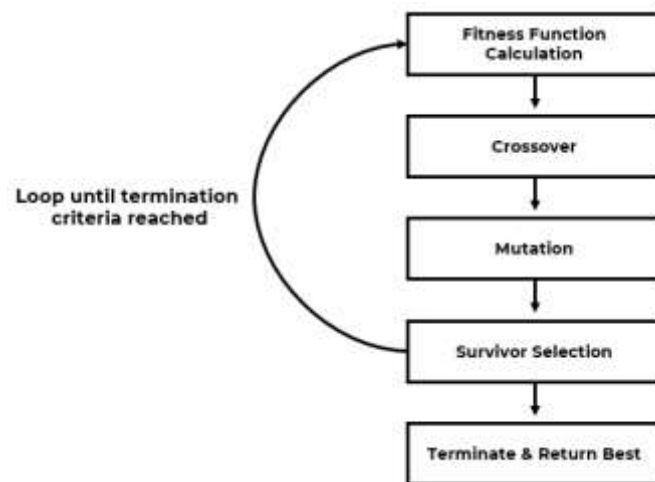
Figure 2.12: Example of Local Maxima

Q5. Explain how genetic algorithm can be used to solve a problem by tasking a suitable example

Ans: [P | Medium]

MAXONE PROBLEM:

Genetic Algorithm steps to solve problem:



1. Produce an initial population of individuals.
2. Evaluate the fitness of all individuals.
3. While termination condition not met do.
 - a. Select fitter individuals for reproduction.
 - b. Recombine between individuals.
 - c. Mutate individuals.
 - d. Evaluate the fitness of the modified individuals.
 - e. Generate a new population.
4. End while.

Encoding:

1. An individual is encoded (naturally) as a string of 'L' binary digits
2. Let's say $L = 10$.
3. Then, $1 = 0000000001$ (10 bits)

Produce an initial population of individuals:

1. We start with a population of n random strings. Suppose that $l = 10$ and $n = 6$.
2. We toss a fair coin 60 times and get the following initial population:

$S_1 = 1111010101$

$S_2 = 0111000101$

$S_3 = 1110110101$

$S_4 = 0100010011$

$S_5 = 1110111101$

$S_6 = 0100110000$

Evaluate the fitness of all individuals:

Now we evaluate the fitness based on 1's.

$S_1 = 1111010101 \quad f(S_1) = 7$

$S_2 = 0111000101 \quad f(S_2) = 5$

$$S_3 = 1110110101 \quad f(S_3) = 7$$

$$S_4 = 0100010011 \quad f(S_4) = 4$$

$$S_5 = 1110111101 \quad f(S_5) = 8$$

$$S_6 = 0100110000 \quad f(S_6) = 3$$

Therefore $f(S) = 34$

Selection:

1. Next we apply fitness proportionate selection with the roulette wheel method.
2. We repeat the extraction as many times as the number of individuals.
3. We need to have the same parent population size (6 in our case)
4. Suppose that, after performing selection, we get the following population:

$$S_1' = 1111010101 \quad (S1)$$

$$S_2' = 1110110101 \quad (S3)$$

$$S_3' = 1110111101 \quad (S5)$$

$$S_4' = 0111000101 \quad (S2)$$

$$S_5' = 0100010011 \quad (S4)$$

$$S_6' = 1110111101 \quad (S5)$$

Crossover:

1. For each couple we decide according to crossover probability (for instance 0.6) whether to actually perform crossover or not.
2. Suppose that we decide to actually perform crossover only for couples $(s1', s2')$ and $(s5', s6')$.
3. For each couple, we randomly extract a crossover point, for instance 2 for the first and 5 for the second.

Before Crossover:

$$S_1' = 1111010101 \quad S_5' = 0100010011$$

$$S_2' = 1110110101 \quad S_6' = 1110111101$$

After Crossover:

$$S_1'' = 1110110101 \quad S_5'' = 0100011101$$

$$S_2'' = 1110110101 \quad S_6'' = 1110110011$$

Mutation:

1. The final step is to apply random mutation.
2. For each bit that we are to copy to the new population we allow a small probability of error (for instance 0.1)
3. Causes movement in the search space (local or global)
4. Restores lost information to the population

Before applying mutation:

$$S_1'' = 1110110101$$

$$S_2'' = 1111010101$$

$$S_3'' = 1110111101$$

$$S_4'' = 0111000101$$

$$S_5'' = 0100011101$$

$$S_6'' = 1110110011$$

After applying mutation:

$$S_1''' = 1110100101$$

$$S_2''' = 1111110100$$

$$S_3''' = 1110101111$$

$$S_4''' = 0111000101$$

$$S_6''' = 1110110001$$

$$S_5''' = 0100011101$$

Evaluate the fitness of the modified individuals:

After Applying Mutation:

$$S_1''' = 1110100101 \quad f(S_1''') = 6$$

$$S_2''' = 1111110100 \quad f(S_2''') = 7$$

$$S_3''' = 1110101111 \quad f(S_3''') = 8$$

$$S_4''' = 0111000101 \quad f(S_4''') = 5$$

$$S_5''' = 0100011101 \quad f(S_5''') = 5$$

$$S_6''' = 1110110001 \quad f(S_6''') = 6$$

Therefore $f(S) = 37$

In one generation, the total population fitness changed from 34 to 37, thus improved by ~9%

At this point, we go through the same process all over again, until a stopping criterion is met.

Q6. Define the terms chromosome, fitness function, crossover and mutation as used in Genetic algorithms.

Ans:

[P | Medium]

CHROMOSOME:

1. In genetic algorithm, a chromosome is a set of parameters which define a proposed solution to the problem that the genetic algorithm is trying to solve.
2. The set of all solutions is known as the **population**.

FITNESS FUNCTION:

1. A fitness function is a particular type of **objective function**.
2. The fitness function is a function which takes a candidate solution to the problem as input and produces as output how "fit" or how "good" the solution is with respect to the problem in consideration.

CROSSOVER:

1. Crossover is also known as **recombination**.
2. Crossover is a genetic operator used to combine the genetic information of two parents to generate new offspring.

FITNESS FUNCTION:

1. Mutation is a genetic operator used to maintain genetic diversity from one generation of a population of genetic algorithm chromosomes to the next.
2. It is analogous to biological mutation.

Q7. Explain A* Algorithm with example**Ans:****[P | Medium]****A*:**

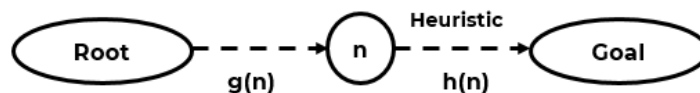
1. A* Algorithm is form of **Best First Search**.
2. It is an **Informed Search Technique**.
3. Additional knowledge in terms of heuristic function is made available.
4. A* Algorithm uses **Priority Queue**.
5. New members are added in the Queue in the ascending order at Evaluation Function $f(n)$.
6. A* uses heuristic function $h(n)$ as well as $g(n)$ for Evaluation.

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

$$g(n) = \text{cost so far to reach } n$$

$$h(n) = \text{estimated cost from } n \text{ to goal}$$

$$f(n) = \text{estimated total cost of path through } n \text{ to goal}$$
7. Heuristics Function $h(n)$ gives estimated cost of reaching Goal from node n . while $g(n)$ gives cost of reaching node n from root.

**Note: $h(n)$ – Cost from Node n to Goal****ALGORITHM:**

1. Create a single member Queue comprising of Root Node.
2. If first member of Queue is Goal, then go to step 5.
3. If first member of Queue is not Goal then remove it from the Queue.
Consider its Successor if any, and add them to the Queue in ascending order of Evaluation Function $f(n)$.
4. If the Queue is not empty, then go to step 2.
If the Queue is empty, then go to step 6.
5. Print "SUCCESS" & Stop.
6. Print "FAILURE" & Stop.

ADVANTAGES:

1. A* algorithm is complete.
2. It is optimal due to admissible heuristic.

LIMITATIONS:

1. Large memory space needed as it generates nodes which are not used.
2. It never opens nodes for which $f(n) > C^*$, where C^* is optimal cost.

Q8. Prove that A* is admissible if it uses a monotone heuristic.

Ans:

[P | Medium]

A*:

1. A* is an **informed search technique**.
2. An admissible heuristic is one that never overestimates the cost to reach the goal.
3. A* uses heuristics function $h(n)$ as well as $g(n)$ for evaluation.

$$F(n) = G(n) + H(n)$$

4. A heuristic is monotone, if the heuristic value is non-decreasing along any path from start to goal.
5. All monotone heuristics are **admissible**.
6. Heuristic Value of A* technique at the Goal node is $H(n^*) = 0$
7. So one step away from the goal,

$$H(n) \leq H(n') + \text{Cost}(n, n') \leq \text{Cost}(n, n^*)$$

8. By induction, $H(n) \leq \text{Cost}(n, n^*)$ from any node.
9. Thus it is admissible since it always underestimates the path cost to the goal state.
10. Therefore if a node is expanded using a monotone heuristic, A* has found the optimal route to that node.
11. A* is admissible if it uses an optimistic heuristic estimate,
12. Therefore all A* Algorithms are admissible.

Q9. Explain BFS.

Ans:

[P | Medium]

BREADTH-FIRST SEARCH (BFS):

1. BFS is **uninformed search technique**.
2. It is used for solving problem by Tree search.
3. Only tree is specified without any additional knowledge of favorable nodes.
4. Its uses two Queues for its implementation.
 - a. Open Queue.
 - b. Close Queue or Visited Queue.
5. Successors are added in Open Queue from Back End [FIFO Queue].

Algorithm:

1. Create a single member Queue comprising of Root Node.
2. If first member of Queue is Goal, then go to step 5.
3. If first member of Queue is not Goal then remove it from the Queue and add to the Visited or Closed Queue.
Consider its Successor if any, and add them to the Queue from Back/Rear End [FIFO].
4. If the Queue is not empty, then go to step 2.
If the Queue is empty, then go to step 6.
5. Print "SUCCESS" & Stop.

6. Print "FAILURE" & Stop.

Performance of BFS:

1. **Completeness:** Yes.
2. **Optimality:** It gives shallowest goal. Optimal, if path cost is not decreasing with depth.
3. **Space Complexity:** $O(b^{d+1})$
4. **Time Complexity:** $O(b^{d+1})$ (d: Depth of the tree & b: Branching factor of tree)

Example:

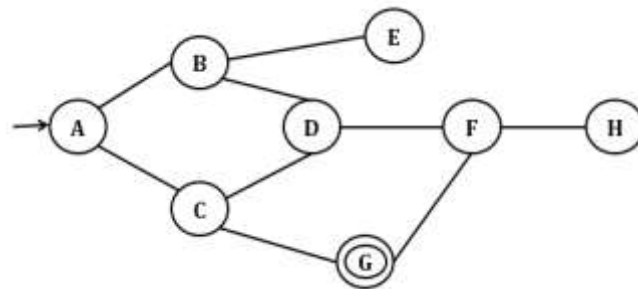


Figure 2.13: BFS Example.

Consider the Example as shown in figure 2.13. Here 'A' is the Initial Node & 'G' is the Goal.

Open Queue	Close Queue
[A]	[]
[B C]	[A]
[C D E]	[A B]
[D E G]	[A B C]
[E G F]	[A B C D]
[G F]	[A B C D E]
[G F]	[A B C D E G]
"Success"	

Q10. Explain DFS.

Ans:

[P | Medium]

DEPTH-FIRST SEARCH (DFS):

1. DFS is an **uninformed search technique**.
2. Depth First Search (DFS) is an algorithm for traversing or searching a tree, tree structure, or graph.
3. It starts from root node of the search tree and going deeper and deeper until a goal node is found, or until it hits a node that has no children.
4. Then the search backtracks, returning to the most recent node it hasn't finished exploring.
5. Its uses two Queues for its implementation.
 - a. Open Queue.
 - b. Close Queue or Visited Queue.

6. Successors are added in Open Queue from Front End [LIFO Queue].

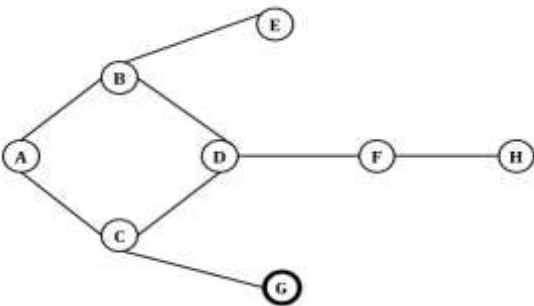
Algorithm:

- 1. Create a single member Queue comprising of Root Node.
- 2. If first member of Queue is Goal, then go to step 5.
- 3. If first member of Queue is not Goal then remove it from the Queue and add to the Visited or Closed Queue.
Consider its Successor if any, and add them to the Queue from Front End [LIFO].
- 4. If the Queue is not empty, then go to step 2.
If the Queue is empty, then go to step 6.
- 5. Print "SUCCESS" & Stop.
- 6. Print "FAILURE" & Stop.

Performance of DFS:

- 1. **Completeness:** No. (Due to dead end)
- 2. **Optimality:** No.
- 3. **Space Complexity:** $O(b * m)$ Where m = maximum depth.
- 4. **Time Complexity:** $O(b^m)$

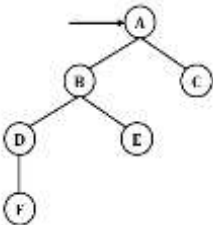
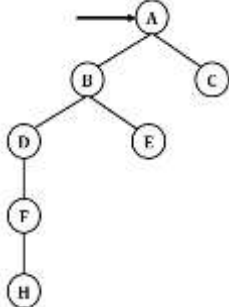
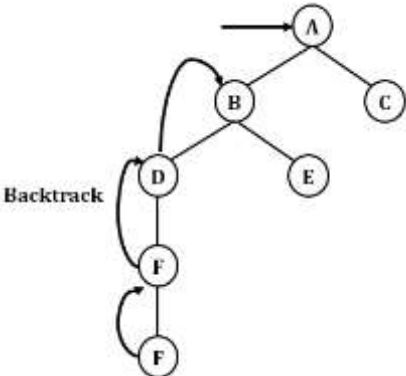
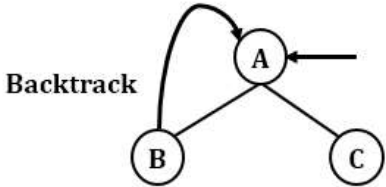
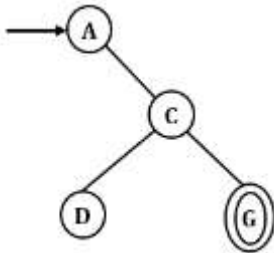
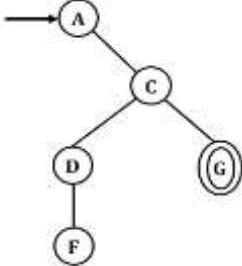
Example: Consider the Example as shown in Question 18.
Consider the figure shown below. Here 'A' is the initial state & 'G' is the goal state.



Assuming that the alphabetically smaller node is expanded first to break ties.

STEPS:

	Tree Representation	Open Queue	Close Queue
1		[A]	[]
2		[B C]	[A]
3		[D E C]	[A B]

4		[F E C]	[A B D]
5		[H E C]	[A B D F]
Since Node 'H' Does not have any Child, hence Backtrack.			
6		[E C]	[A B]
7		[C]	[A]
8		[D G]	[A C]
9		[F G]	[A C D]

10		[H G]	[A C D F]
Since Node 'H' Does not have any Child, hence Backtrack.			
11		[G]	[A C]
Success.... Goal Reached and DFS Path: A → C → G			

Suppose the Cost Between the nodes be as shown in table below

State	Next State	Cost
A	B	4
A	C	1
B	D	3
B	E	8
C	D	2
C	G	6
D	F	3
F	H	2

Therefore Solution Cost of Path

$$A \rightarrow C \rightarrow G = 1 + 6 = 7$$

Q11. Explain DLS.

Ans:

[P | Medium]

DEPTH-LIMITED SEARCH (DLS):

1. Depth Limited Search is used to avoid Dead-End problem of DFS.
2. It puts Limit on maximum depth to which search is allowed.
3. Beyond that limit, search is not performed.
4. It then explores other branch.
5. Depth of each state is recorded as it is generated.
6. When picking the next state to expand, only those with depth less or equal than the current depth are expanded.
7. Once all the nodes of a given depth are explored, the current depth is incremented.

Algorithm:

1. Create a single member Queue comprising of Root Node.
2. If first member of Queue is Goal, then go to step 5.
3. If first member of Queue is not Goal then remove it from the Queue and add to the Visited or Closed Queue.

Consider its Successor if any, and add them to the Queue from Front End [LIFO].

4. If the Queue is not empty, then go to step 2.
If the Queue is empty, then go to step 6.
5. Print "SUCCESS" & Stop.
6. Print "FAILURE" & Stop.

Performance of DLS:

1. **Completeness:** No. (Due to dead end)
2. **Optimality:** Yes if $l \geq d$
3. **Space Complexity:** $O(b * l)$ Where l = Depth limit.
4. **Time Complexity:** $O(b^l)$

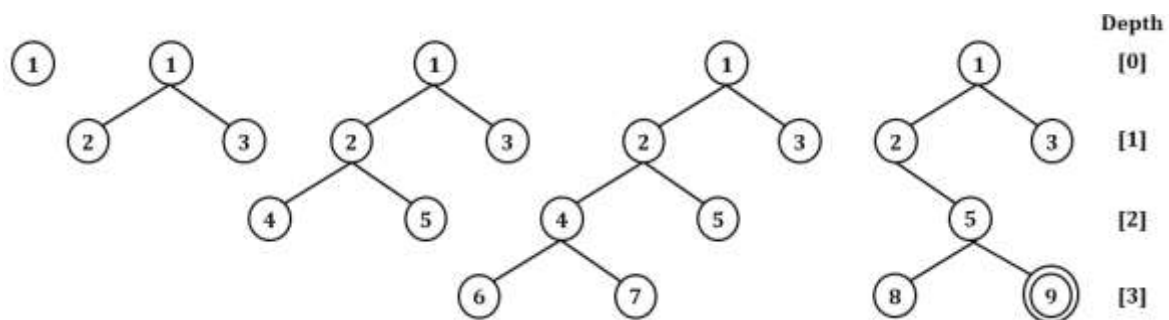
Example:

Figure 2.14: DLS Example.

Consider the Example as shown in figure 2.14. Here '1' is the Initial Node & '9' is the Goal. Since the Depth Limit is 3, it explores the node till 3rd depth.

Q12. Explain Iterative Deepening Depth-First Search.

Ans:

[P | Medium]

ITERATIVE DEEPENING DEPTH-FIRST SEARCH:

1. It is variation of DFS.
2. It combines the advantages of BFS like completeness & optimality.
3. The depth limit is iteratively increased, starting from 0 till Goal is reached.
4. Hence, it becomes complete and optimal like BFS.
5. Also its memory space requirement is less like DFS.
6. Successors are added in Open Queue from Front End [LIFO Queue].

Algorithm:

1. Create a single member Queue comprising of Root Node.
2. If first member of Queue is Goal, then go to step 5.

3. If first member of Queue is not Goal then remove it from the Queue and add to the Visited or Closed Queue.
Consider its Successor if any, and add them to the Queue from Front End [LIFO].
4. If the Queue is not empty, then go to step 2.
If the Queue is empty, then go to step 6.
5. Print "SUCCESS" & Stop.
6. Print "FAILURE" & Stop.

Performance:

1. **Completeness:** Yes.
2. **Optimality:** Yes (If path cost is not decreasing with depth).
3. **Space Complexity:** $O(b * d)$ Where b = branching factor & d = depth of the solution.
4. **Time Complexity:** $O(b^d)$

Example:

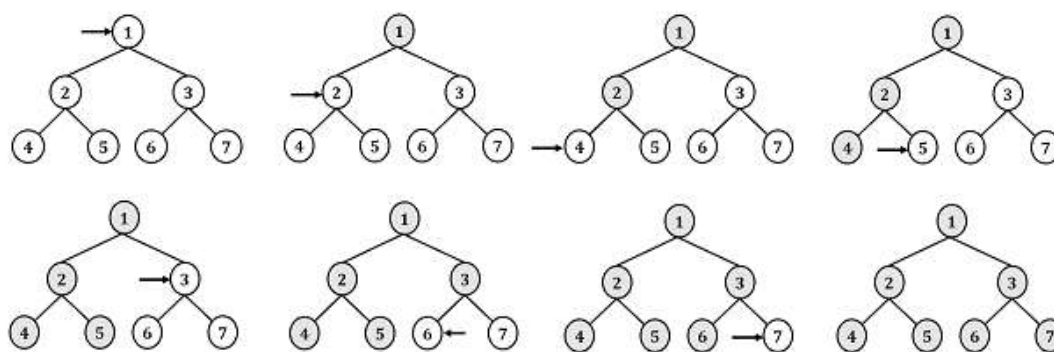


Figure 2.15: Iterative Deepening Depth First Search Example.

Consider the Example as shown in figure 2.15. Since the Depth Limit is 2, it explores the node till 2nd depth.

Q13. Compare Greedy Best first search and A* search algorithms based on performance measure with justification: Complete, Optimal, Time and Space complexity

Ans:

[P | High]

GREEDY BEST FIRST:

Greedy best-first search expands the nodes that appears to be closest to the goal.

Performance Measure	Greedy Best First	Justification
Complete	No	Can get Stuck in loops.
Optimal	No	Usually Path Selected by Algorithm is Longest as compared to optimal Solution.
Time Complexity	Best: $O(d)$ and Worst: $O(b^m)$	But a good heuristic can give dramatic improvement. (m = Maximum Depth)

Space Complexity	$O(b^m)$	Keeps all nodes in memory.
------------------	----------	----------------------------

A*:

A* Algorithm avoid expanding paths that are already expansive.

Performance Measure	A*	Justification
Complete	Yes	Unless there are infinitely many nodes with $F \leq F(G)$
Optimal	Yes	If the heuristic is admissible and No Algorithm with the same heuristic is guaranteed to expand fewer nodes.
Time Complexity	$O(b^d)$	Exponential.
Space Complexity	$O(b^d)$	Keeps all nodes in memory.

RECURSIVE BEST-FIRST (RBFS):

RBFS keeps the track of Evaluation Value of the best alternative path available.

Performance Measure	RBFS	Justification
Complete	Yes	It is Better Efficient than IDA*
Optimal	Yes	If the heuristic is admissible
Time Complexity	$O(b^d)$	Exponential.
Space Complexity	$O(b^d)$	Keeps all nodes in memory.

Q14. Given a full 4-gallon jug and an empty 3- gallon jug, the goal is to fill the 4-gallon jug with exactly 2-gallons of water. Give state space representation.

Ans:

[P | Medium]

- The state space for this problem can be represented by set of ordered pairs of integers **(X, Y)**
- Where,
 - X represents the quantity of water in the 4 - gallon jug.
 - Y represents the quantity of water in 3 - gallon jug.
- State Space = (X, Y)**
- Such that:**

$$X = \{0, 1, 2, 3, 4\} \text{ \& } Y = \{0, 1, 2, 3\}$$
- As given the start state is: **Start State: (4, 0)**
- The goal state is 2 gallons of water in the 4 gallon jug: **Goal State: (2, 0)**
- Now we have to generate production rules for the water jug problem.

PRODUCTION RULES:

Rule	State	Process
1	$(X, Y \mid X < 4)$	(4, Y) {Fill 4 gallon jug}
2	$(X, Y \mid Y < 3)$	(X, 3) {Fill 3 gallon jug}

3	$(X, Y \mid X > 0)$	$(0, Y)$ {Empty 4 gallon jug}
4	$(X, Y \mid Y > 0)$	$(X, 0)$ {Empty 3 gallon jug}
5	$(X, Y \mid 0 < X + Y \leq 4 \text{ and } Y > 0)$	$(4, Y - (4 - X))$ { Pour water from 3 gallon jug to fill 4 gallon jug }
6	$(X, Y \mid 0 < X + Y \leq 3 \text{ and } X > 0)$	$(X - (3 - Y), 3)$ { Pour water from 4 gallon jug to fill 3 gallon jug }
7	$(X, Y \mid 0 < X + Y \leq 4 \text{ and } Y \geq 0)$	$(X + Y, 0)$ { Pour all of water from 3 gallon jug into 4 gallon jug }
8	$(X, Y \mid 0 < X + Y \leq 3 \text{ and } X \geq 0)$	$(0, X + Y)$ { Pour all of water from 4 gallon jug into 3 gallon jug }
9	$(0, 2)$	$(2, 0)$ {Pour 2 gallon water from 3 gallon jug into 4 gallon jug}

INITIALIZATION:

Start State: $(4, 0)$

Apply Rule 3: $(X, Y \mid X > 0) \rightarrow (0, 0)$

{Empty 4 - gallon jug}

Now the state is $(0, 0)$

Iteration 1:

Start State: $(0, 0)$

Apply Rule 2: $(X, Y \mid Y < 3) \rightarrow (X, 3)$

{Fill 3 - gallon jug}

Now the state is $(X, 3)$

Iteration 2:

Current State: $(X, 3)$

Apply Rule 7: $(X, Y \mid 0 < X + Y \leq 4 \text{ and } Y \geq 0) \rightarrow (X+Y, 0)$

{Pour all water from 3-gallon jug into 4-gallon jug}

Now the state is $(3, 0)$

Iteration 3:

Current State: $(3, 0)$

Apply Rule 2: $(X, Y \mid Y < 3) \rightarrow (3, 3)$

{Fill 3-gallon jug}

Now the state is $(3, 3)$

Iteration 4:

Current State: $(3, 3)$

Apply Rule 5: $(X, Y \mid 0 < X + Y \leq 4 \text{ and } Y > 0) \rightarrow (4, Y - (4 - X))$

{Pour water from 3-gallon jug into 4-gallon jug until 4-gallon jug is full}

Now the state is (4, 2)

Iteration 5:

Current State: (4, 2)

Apply Rule 3: $(X, Y \mid X > 0) \rightarrow (0, Y)$

{Empty 4-gallon jug}

Now state is (0, 2)

Iteration 6:

Current State: (0, 2)

Apply Rule 9: $(0, 2) \rightarrow (2, 0)$

{Pour 2 gallon water from 3 gallon jug into 4 gallon jug}

Now the state is (2, 0)

Goal Achieved.

STATE SPACE TREE:

Figure 2.16 represents the state space tree for the problem.

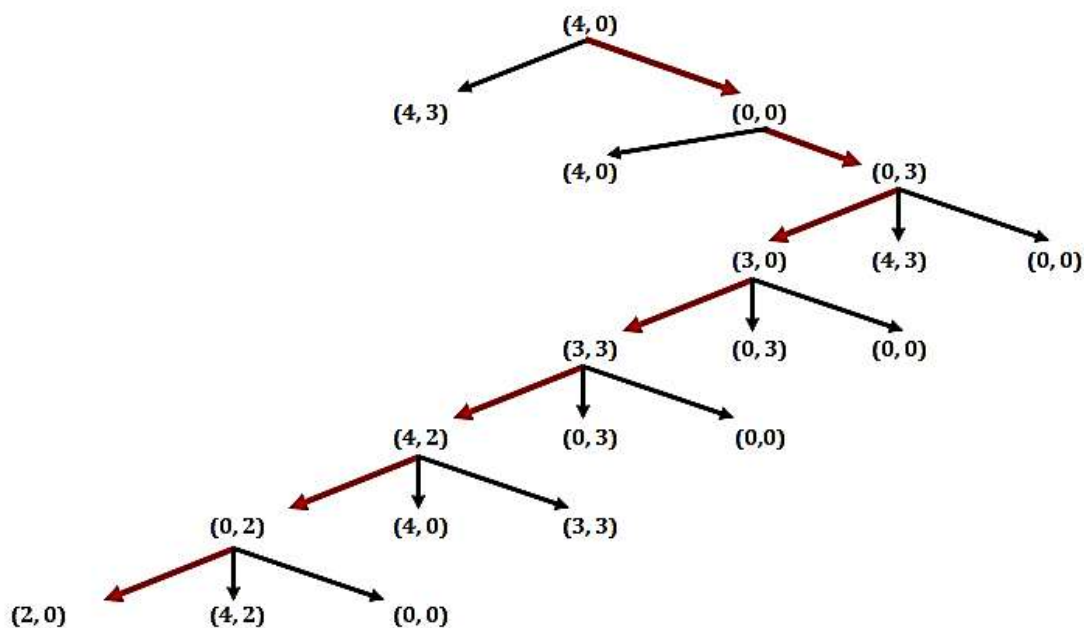


Figure 2.16: State Space Tree.

Q15. Compare and contrast uninformed search strategies with respect to solving 8-puzzle problem.

Ans:

[P | Medium]

8-PUZZLE PROBLEM:

1. In the 8-puzzle problem we have a 3×3 square board and 8 numbered tiles.
2. The board has one blank position.
3. Blocks can be slid to adjacent blank positions.
4. We can alternatively and equivalently look upon this as the movement of the blank position up, down, left or right.

5. The objective of this puzzle is to move the tiles starting from an initial position and arrive at a given goal configuration.

COMPARISON OF UNIFORMED SEARCH ALGORITHM:

Criterion	Breadth First	Uniform Cost	Depth First	Depth Limited	Iterative Deepening
Complete	Yes ^a	Yes ^{a, b}	No	No	Yes ^a
Time	$O(b^d)$	$O(b^{1 + \lceil C^*/\epsilon \rceil})$	$O(b^m)$	$O(b^L)$	$O(b^d)$
Space	$O(b^d)$	$O(b^{1 + \lceil C^*/\epsilon \rceil})$	$O(bm)$	$O(bL)$	$O(bd)$
Optimal	Yes ^c	Yes	No	No	Yes ^c

Where,

$b \rightarrow$ Branching Factor.

$d \rightarrow$ Depth of the shallowest solution.

$m \rightarrow$ Maximum Depth.

$L \rightarrow$ Depth Limit.

Superscript caveats are as follows:

$a \rightarrow$ Complete if 'b' is finite.

$b \rightarrow$ Complete if step costs $\geq \epsilon$

$c \rightarrow$ Optimal if step costs are all identical.

Q16. Local Search Algorithms

Ans:

[P | Medium]

LOCAL SEARCH ALGORITHMS:

1. In many optimization problems, the path to the goal is irrelevant; the goal state itself is the solution.
2. In such cases, we can use **local search algorithms**.
3. Local search algorithms can start from a prospective solution and then move to a neighboring solution.
4. It can return a valid solution even if it is interrupted at any time before they end.
5. Local search algorithm includes:

I) Hill Climbing Search:

Refer Q5.

II) Local Beam Search:

1. Local Beam Search is a **heuristic search algorithm**.
2. It explores a graph by expanding the most promising node in a limited set.
3. Local Beam search is an optimization of best-first search that reduces its **memory requirements**.
4. In this algorithm, it holds k number of states at any given time.
5. At the start, these states are generated randomly.
6. The successors of these k states are computed with the help of objective function.
7. If any of these successors is the maximum value of the objective function, then the algorithm stops.
8. Otherwise the (initial k states and k number of successors of the states = 2k) states are placed in a pool.

9. The pool is then sorted numerically.
10. The highest k states are selected as new initial states.
11. This process continues until a maximum value is reached.

III) Simulated Annealing:

1. Simulated Annealing (SA) is an effective and general form of **optimization**.
2. It is useful in finding global optima in the presence of large numbers of local optima
3. Annealing is the process of heating and cooling a metal to change its internal structure for modifying its physical properties.
4. When the metal cools, its new structure is seized, and the metal retains its newly obtained properties.
5. In simulated annealing process, the temperature is kept variable.
6. We initially set the temperature high and then allow it to 'cool' slowly as the algorithm proceeds.
7. When the temperature is high, the algorithm is allowed to accept worse solutions with high frequency.

IV) Travelling Salesman Problem:

1. The Travelling Salesman Problem (TSP) is a classic algorithmic problem in the field of computer science.
2. It is focused on **optimization**.
3. In this context better solution often means a solution that is cheaper.
4. TSP is a mathematical problem.
5. It is most easily expressed as a graph describing the locations of a set of nodes.

EXAMPLE:

Consider the following graph of cities as shown in figure 2.17.

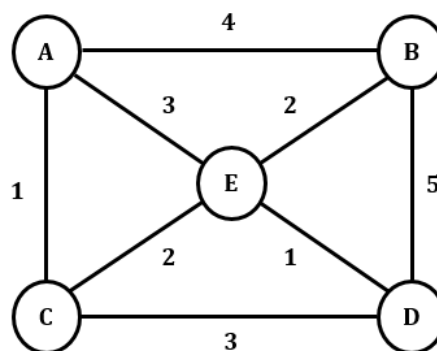


Figure 2.17: Graph of Cities.

FORMULATION:

States: Cities.

Initial state: A

Successor function: Travel from one city to another connected by a road.

Goal test: The trip visits each city only once that starts and ends at A.

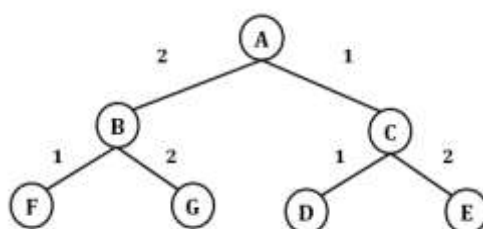
Path cost: Traveling time.

Q17. Uniform Cost Search?**Ans:****[P | Medium]****UCS:**

1. UCS is same as BFS.
2. It is **uninformed search technique**.
3. It is used to find worst case space & time complexity.
4. Uniform cost search is a search algorithm used to traverse, and find the shortest path in weighted trees and graphs.
5. Uniform Cost Search or UCS begins at a root node and will continually expand nodes, taking the node with the smallest total cost from the root until it reaches the goal state.
6. Uniform cost search doesn't care about how many steps a path has, only the total cost of the path.
7. UCS with all path costs equal to one is identical to breadth first search.
8. Let, $C^* \rightarrow$ Optimal Cost & $E \rightarrow$ Smallest step cost.
9. Maximum no. of steps = C^* / E

PERFORMANCE OF BFS:

1. **Completeness:** Yes.
2. **Optimality:** Yes.
3. **Space Complexity:** $O(b^{1 + \lceil C^* / E \rceil})$
4. **Time Complexity:** $O(b^{1 + \lceil C^* / E \rceil})$

EXAMPLE:**Figure 2.18: UCS Example.**

Consider the Example as shown in figure 2.18. Here 'A' is the Initial Node & 'G' is the Goal.

Open Queue	Close Queue
[A] 0	[]
[B C] 2 1	[A]
[B D E] 2 2 3	[A C]
[D E F G] 2 3 3 4	[A C B]
[E F G] 3 3 4	[A C B D]

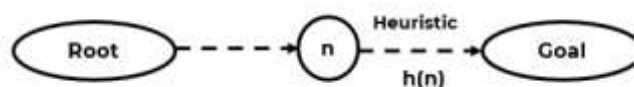
[F G] 3 4	[A C B D E]
[G] 4	[A C B D E F]
"Success"	

Q18. Best First Search?**Ans:****[P | Medium]****BEST FIRST SEARCH:**

1. Best First Search is an informed search technique.
2. Additional knowledge in terms of heuristic function is made available.
3. It uses Priority Queue.
4. New members are added in the Queue in the ascending order at Evaluation Function $f(n)$.
5. Best First Search uses heuristic function $h(n)$ itself as Evaluation Function.

$$f(n) = h(n)$$

6. Heuristic Function $h(n)$ gives estimated cost for reaching Goal from Node "n"



Note: $h(n)$ - Cost from Node n to Goal

ALGORITHM:

1. Create a single member Queue comprising of Root Node.
2. If first member of Queue is Goal, then go to step 5.
3. If first member of Queue is not Goal then remove it from the Queue.
Consider its Successor if any, and add them to the Queue in ascending order of Evaluation Function $f(n)$.
4. If the Queue is not empty, then go to step 2.
If the Queue is empty, then go to step 6.
5. Print "SUCCESS" & Stop.
6. Print "FAILURE" & Stop.

ADVANTAGES & LIMITATIONS:

1. Best First Search is complete but not optimum.
2. It considers only $h(n)$ to explore nodes.

PERFORMANCE OF BEST FIRST SEARCH:

1. **Completeness:** No (Can get stuck in loops).
2. **Optimality:** No.
3. **Space Complexity:** $O(b^m)$
4. **Time Complexity:** $O(b^m)$

EXAMPLE:

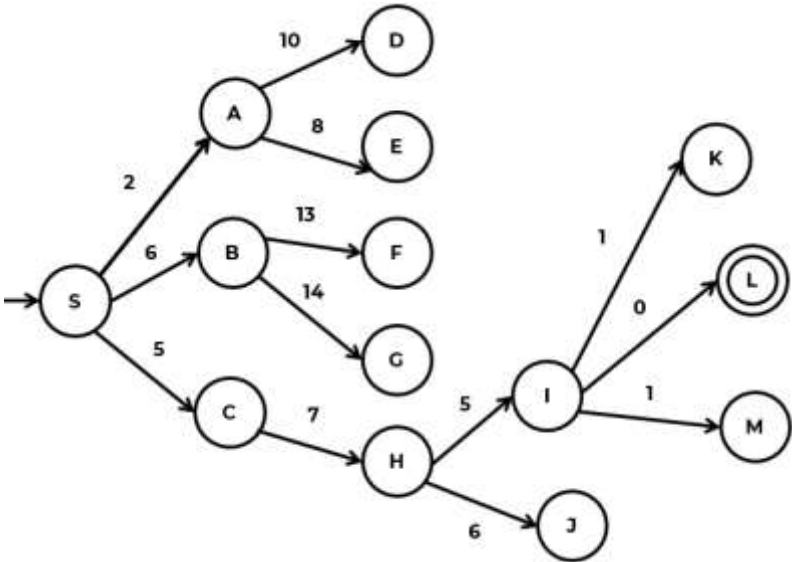


Figure 2.19: Best First Search Example.

Consider the Example as shown in figure 2.19. Here 'S' is the Initial Node & 'L' is the Goal.

Open Queue	Close Queue
[S] 0	[]
[A C B] 2 5 6	[S]
[C B E D] 5 6 8 10	[S A]
[B H E D] 6 7 8 10	[S A C]
[H E D F G] 7 8 10 13 14	[S A C B]
[I J H E D F G] 5 6 7 8 10 13 14	[S A C B H]
[L K M I J H E D F G] 0 1 1 5 6 7 8 10 13 14	[S A C B H I]
[K M I J H E D F G] 1 1 5 6 7 8 10 13 14	[S A C B H I L]
"Success"	

Q19. IDA* Algorithm?

Ans:

[P | Medium]

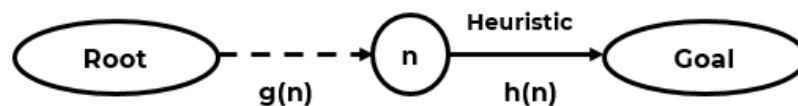
IDA*:

- 1. IDA* stands for Iterative Deeping A* Algorithm.
- 2. It is used to remove memory space problem of A* Algorithm.
- 3. A* generates the nodes without using them.

4. It never opens nodes for which $f(n) > C^*$.
5. Hence, they need to be flushed out from memory. But C^* is not known Beforehand.
6. IDA* iteratively increases limit of C^* starting from zero till Goal is reached.
7. It is Informed Search & uses Priority Queue.
8. New members are added in the Queue in the ascending order at Evaluation Function $f(n)$.
9. IDA* uses heuristic function $h(n)$ as well as $g(n)$ for Evaluation.

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

10. Heuristics Function $h(n)$ gives estimated cost of reaching Goal from node n . while $g(n)$ gives cost of reaching node n from root.



Note: $h(n)$ – Cost from Node n to Goal

ALGORITHM:

1. Create a single member Queue comprising of Root Node.
2. If first member of Queue is Goal, then go to step 6.
3. If first member of Queue is not Goal then remove it from the Queue.
Consider its Successor if any, and add them to the Queue in ascending order of Evaluation Function $f(n)$.
4. If the Queue is not empty, then go to step 2.
If the Queue is empty, then go to step 7.
5. Nodes for which $f(n) > C$ is flushed.
6. Print "SUCCESS" & Stop.
7. Print "FAILURE" & Stop.

ADVANTAGES:

1. IDA* algorithm is complete.
2. It is optimal due to admissible heuristic.
3. It is memory efficient.

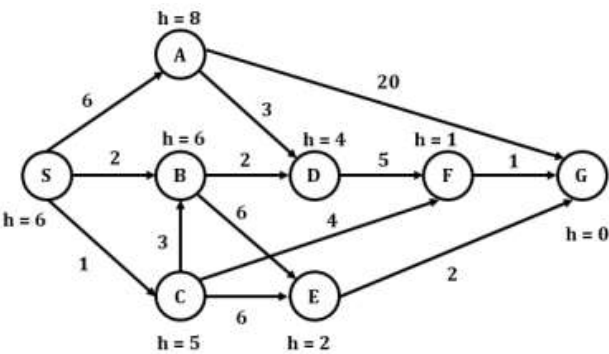
LIMITATIONS:

Due to its iterative nature it takes more time to search.

EXAMPLE:

Same as A* Algorithm. Show the Flushing for $f(n) > C^*$ at end

Q20. Consider the search problem below with start state S and goal state G. The transition costs are next to the edges and the heuristic values are next to the states. What is the final cost using A* search.



Ans:

[P | Medium]

EXAMPLE:

Consider the figure 2.20 shown below. Here 'S' is the initial state & 'G' is the goal state. The heuristic/cost is given along branches & links.

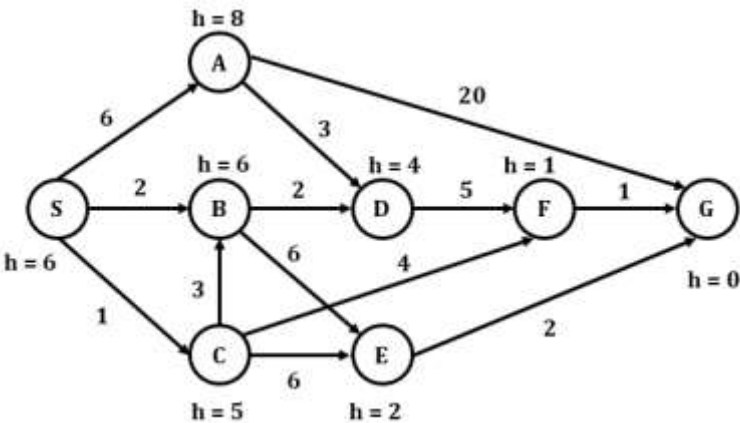
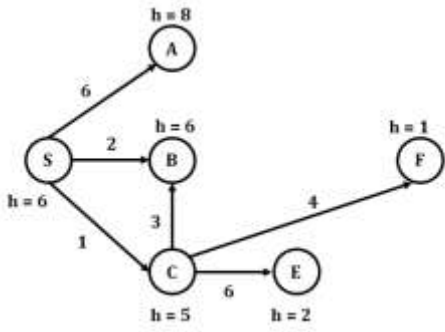
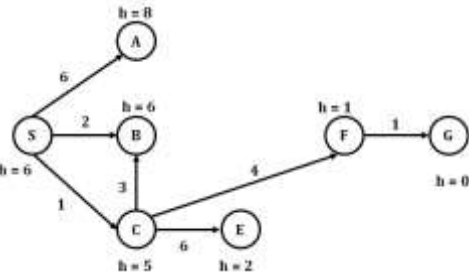


Figure 2.20: Example of A*

Steps:

Tree	Cost	Open Queue	Close Queue
<div><div>1</div><div></div></div>	<div><div><div><div>$F(A) = G(A) + H(A)$</div><div>$= 6 + 8$</div><div>$= 14$</div></div><div><div>$F(B) = G(B) + H(B)$</div><div>$= 2 + 6$</div><div>$= 8$</div></div><div><div>$F(C) = G(C) + H(C)$</div><div>$= 1 + 5$</div><div>$= 6$</div></div></div></div>	<div><div>[S]</div><div>[S A B C]</div></div>	<div><div>[S]</div><div>[S C]</div></div>

2		$F(B) = G(B) + H(B)$ $= 3 + 6$ $= 9$ $F(F) = G(F) + H(F)$ $= 4 + 1$ $= 5$ $F(E) = G(E) + H(E)$ $= 6 + 2$ $= 8$	[S] [S C] [S C B F E]	[S] [S C] [S C F]
3		$F(G) = G(G) + H(G)$ $= 1 + 0$ $= 1$	[S] [S C] [S C F] [S C F G]	[S] [S C] [S C F] [S C F G]
<p align="center">“Success” Hence we have found the Optimal Solution for the problem.</p>				

Optimal Path = S → C → F → G

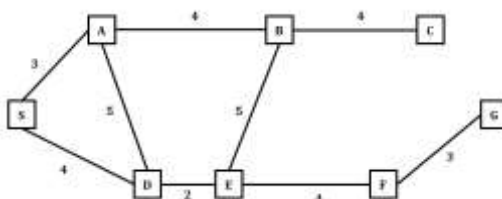
Optimal Cost = 1 + 4 + 1 = 6

Q21. Consider the graph given in Figure below. Assume that the initial state is A and the goal state is G. Show how Greedy Best first Search would create a search tree to find a path from the initial state to the goal state:

At each step of the search algorithm, show which node is being expanded, and the content of fringe. Also report the eventual solution found by the algorithm, and the solution cost.

Assuming the straight-line distance as the heuristics function:

$h(S)=10.5$, $h(A)=10$, $h(B)=6$, $h(C)=4$, $h(D)=8$, $h(E)=6.5$, $h(F)=3$ and $h(G)=0$



Ans:

[P | Medium]

GREEDY BEST FIRST SEARCH:

1. Greedy Best First Search is **Informed (Heuristic) Search Technique**.
2. It is a basic search that uses path-cost to determine the next node to expand.
3. It expands the node that is estimated to be closest to goal.
4. It expands nodes based on $f(n) = h(n)$.
5. It is implemented using **priority queue**.

EXAMPLE:

	Tree	Cost	Open Queue	Close Queue
1		$F(A) = H(A) = 10$ $F(D) = H(D) = 8$	[S] [A D]	[S] [S D]
2		$F(E) = H(E) = 6.5$	[S] [A D] [S D E]	[S] [S D] [S D E]
3		$F(B) = H(B) = 6$ $F(F) = H(F) = 3$	[S] [A D] [S D E] [S D E B F]	[S] [S D] [S D E] [S D E B]
4		$F(G) = H(G) = 0$	[S] [A D] [S D E] [S D E B F] [S D E B G]	[S] [S D] [S D E] [S D E B] [S D E B G]
"Success" Hence we have found the Solution for the problem.				

Cost = 8 + 6.5 + 3 + 0 = 17.5 & Optimal Path = S → D → E → F → G

CHAP - 3: KNOWLEDGE, REASONING & PLANNING

Q1. Wumpus world knowledge base

Ans:

[5M | Dec-19]

WUMPUS WORLD:

1. Wumpus world is a **computer game**.
2. It is used as test bed for an intelligent agent.
3. Agent is allowed to explore the Wumpus world and its performance is judged based on the point it scored.
4. Highest point = more intelligent
5. Nice example for the reasoning.
6. It is a 4 x 4 Grid of the rooms as shown in the figure 3.1.

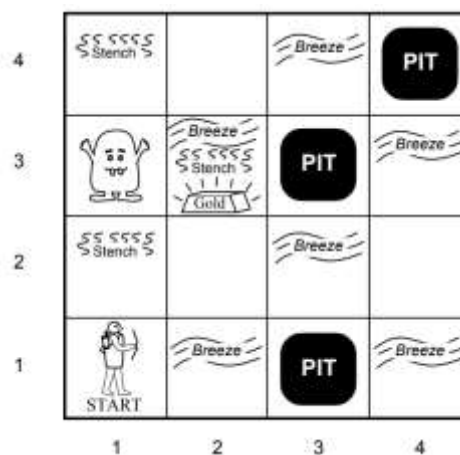


Figure 3.1 Wumpus World Environment

7. The Wumpus world is a cave consisting of rooms connected by passage ways.
8. Wumpus is hidden in one of the room and eats anyone who enters its room.
9. Wumpus can be shot by the agent but the agent has only one arrow.
10. Three rooms contain bottomless PITs.
11. The only mitigating feature of living in this environment is the possibility of finding a heap of gold.

KNOWLEDGE BASE:

1. Knowledge-base is a central component of a knowledge-based agent, it is also known as KB.
2. It is a collection of sentences (here 'sentence' is a technical term and it is not identical to sentence in English).
3. These sentences are expressed in a language which is called a knowledge representation language.

REPRESENTATION OF KNOWLEDGE BASE FOR WUMPUS WORLD:

Following is the Simple KB for wumpus world when an agent moves from room [1, 1], to room [2, 1]:

$\neg W_{11}$	$\neg S_{11}$	$\neg P_{11}$	$\neg B_{11}$	$\neg G_{11}$	V_{11}	OK_{11}
$\neg W_{12}$	----	$\neg P_{12}$	-----	----	$\neg V_{12}$	OK_{12}
$\neg W_{21}$	$\neg S_{21}$	$\neg P_{21}$	B_{21}	$\neg G_{21}$	V_{21}	OK_{21}

1. Here in the first row, we have mentioned propositional variables for room[1, 1], which is showing that room does not have wumpus($\neg W_{11}$), no stench ($\neg S_{11}$), no Pit($\neg P_{11}$), no breeze($\neg B_{11}$), no gold ($\neg G_{11}$), visited (V_{11}), and the room is Safe(OK_{11}).
2. In the second row, we have mentioned propositional variables for room [1, 2], which is showing that there is no wumpus, stench and breeze are unknown as an agent has not visited room [1,2], no Pit, not visited yet, and the room is safe.
3. In the third row we have mentioned propositional variable for room[2, 1], which is showing that there is no wumpus($\neg W_{21}$), no stench ($\neg S_{21}$), no Pit ($\neg P_{21}$), Perceives breeze(B_{21}), no glitter($\neg G_{21}$), visited (V_{21}), and room is safe (OK_{21}).

Q2. The law says that it is a crime for an American to sell weapons to hostile nations. The country Nono, an enemy of America, has some missiles, and all of its missiles were sold to it by Colonel West, who is American. Prove that Col. West is a criminal using resolution technique

Ans: [10M | Dec-19]

Represent the above sentences in first order predicate logic (FOPL) and convert it to CNF:

1. It is a crime for an American to sell weapons to hostile nations
FOPL: $\text{American}(x) \wedge \text{Weapon}(y) \wedge \text{Sells}(x, y, z) \wedge \text{Hostile}(z) \Rightarrow \text{Criminal}(x)$
CNF: $\neg \text{American}(x) \vee \neg \text{Weapon}(y) \vee \neg \text{Sells}(x, y, z) \vee \neg \text{Hostile}(z) \vee \text{Criminal}(x)$
2. Nono has some missiles, i.e., $\exists x \text{ Owns}(\text{Nono}, x) \wedge \text{Missile}(x)$
FOPL: $\text{Owns}(\text{Nono}, M1) \wedge \text{Missile}(M1)$
CNF: $\text{Owns}(\text{Nono}, M1) \wedge \text{Missile}(M1)$
3. All of its missiles were sold to it by Colonel West
FOPL: $\text{Missile}(x) \wedge \text{Owns}(\text{Nono}, x) \Rightarrow \text{Sells}(\text{West}, x, \text{Nono})$
CNF: $\neg \text{Missile}(x) \vee \neg \text{Owns}(\text{Nono}, x) \vee \text{Sells}(\text{West}, x, \text{Nono})$
4. Missiles are weapons:
FOPL: $\text{Missile}(x) \Rightarrow \text{Weapon}(x)$
CNF: $\neg \text{Missile}(x) \vee \text{Weapon}(x)$
5. An enemy of America counts as "hostile"
FOPL: $\text{Enemy}(x, \text{America}) \Rightarrow \text{Hostile}(x)$
CNF: $\neg \text{Enemy}(x, \text{America}) \vee \text{Hostile}(x)$
6. West, who is American
FOPL: $\text{American}(\text{West})$
CNF: $\text{American}(\text{West})$
7. The country Nono, an enemy of America.
FOPL: $\text{Enemy}(\text{Nono}, \text{America})$
CNF: $\text{Enemy}(\text{Nono}, \text{America})$

Prove that "West is Criminal" using resolution technique:

1	$\neg \text{American}(x) \vee \neg \text{Weapon}(y) \vee \neg \text{Sells}(x, y, z) \vee \neg \text{Hostile}(z) \vee \text{Criminal}(x)$		
2	$\neg \text{Missile}(x) \vee \neg \text{Owns}(\text{Nono}, x) \vee \text{Sells}(\text{West}, x, \text{Nano})$		
3	$\neg \text{Enemy}(x, \text{America}) \vee \text{Hostile}(x)$		
4	$\neg \text{Missile}(x) \vee \text{Weapon}(x)$		
5	$\text{Owns}(\text{Nono}, \text{M1})$		
6	$\text{Missile}(\text{M1})$		
7	$\text{American}(\text{West})$		
8	$\text{Enemy}(\text{Nano}, \text{America})$		
9	$\neg \text{Criminal}(\text{West})$		
10	$\neg \text{American}(\text{West}) \vee \neg \text{Weapon}(y) \vee \neg \text{Sells}(\text{West}, y, z) \vee \neg \text{Hostile}(z)$	1, 9	{x/West}
11	$\neg \text{Weapon}(y) \vee \neg \text{Sells}(\text{West}, y, z) \vee \neg \text{Hostile}(z)$	7, 10	{x/West}
12	$\neg \text{Missile}(y) \vee \neg \text{Sells}(\text{West}, y, z) \vee \neg \text{Hostile}(z)$	4, 11	{x/y}
13	$\neg \text{Sells}(\text{West}, \text{M1}, z) \vee \neg \text{Hostile}(z)$	6, 12	{y/M1}
14	$\neg \text{Missile}(\text{M1}) \vee \neg \text{Owns}(\text{Nono}, \text{M1}) \vee \neg \text{Hostile}(\text{Nano})$	2, 13	{x/M1, z/Nano}
15	$\neg \text{Owns}(\text{Nono}, \text{M1}) \vee \neg \text{Hostile}(\text{Nano})$	6, 14	{}
16	$\neg \text{Hostile}(\text{Nano})$	5, 15	{}
17	$\neg \text{Enemy}(\text{Nano}, \text{America})$	3, 16	{x/Nano}
18		8, 17	{}

Resolution Tree:

Figure 3.2 shows the resolution tree for the above example.

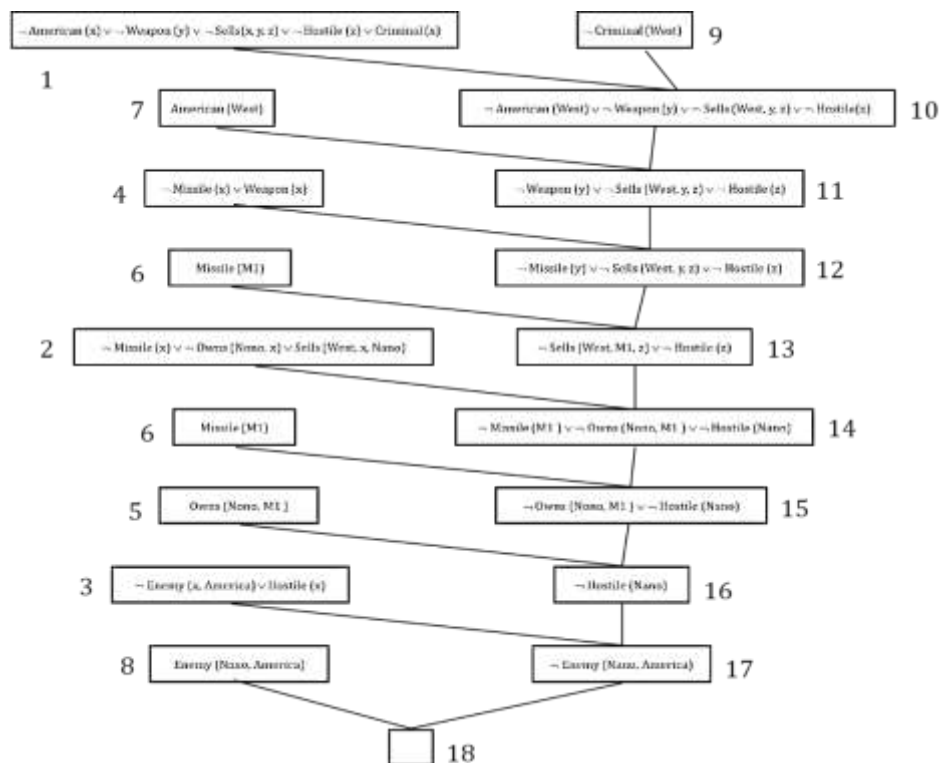


Figure 3.2: Resolution Tree

Q3. Explain planning problem in AI. What are different types of planning? Consider problem of changing a flat tire. The goal is to have a good spare tire properly mounted on to the car's axle, where the initial state has a flat tire on the axle and a good spare tire in the trunk. Give the ADL description for the problem

Ans:

[10M | Dec-19]

PLANNING PROBLEM:

1. The planning problem in Artificial Intelligence is about the decision making performed by intelligent creatures like robots, humans, or computer programs when trying to achieve some goal.
2. It involves choosing a sequence of actions that will transform the state of the world, step by step, so that it will satisfy the goal.
3. The world is typically viewed to consist of atomic facts (state variables), and actions make some facts true and some facts false.
4. Goal can be specified as a union of sub-goals.
5. Consider the example of Ping Pong Game where, points are assigned to opponent player when a player fails to return the ball within the rules of ping-pong game.
6. There can a best 3 of 5 matches where, to win a match you have to win 3 games and in every game you have to win with a minimum margin of 2 points.
7. **Examples of Planning Agent:** Search-based problem-solving agent & Logical planning agent.

TYPES OF PLANNING:

I) Progression Planners:

1. **Forward State Space Search** is also called as "Progression Planner".
2. It is deterministic Planning Technique.
3. In Progression Planners, we plan sequence of actions starting from the initial state in order to attain the goal.
4. With forward state space searching method we start with the initial state and go to final goal state.
5. While doing this, we need to consider the probable effects of the actions taken at every state.
6. Thus the prerequisite for this type of planning is to have initial world state information, details of the available actions of the agent, and description of the goal state.

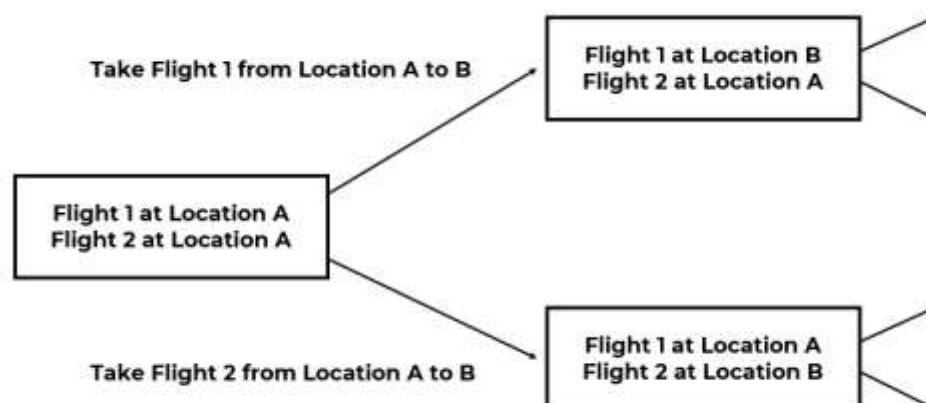


Figure 3.3: State Space Graph of Progression Planner

7. In Figure 3.3, it gives a state-space graph of progression planner for a simple example where flight 1 is at location A and flight 2 is also at location A.
8. These flights are moving from location A to location B.
9. In 1st case only flight 1 moves from location A to location B, so the resulting state shows that after performing that action flight 1 is at location B whereas flight 2 is at its original location A.
10. Similarly In 2nd case only flight 2 moves from location A to location B and the resulting state shows that after performing that action flight 2 is at location B while flight 1 is at its original.
11. **Disadvantage:** Large branching factor.

II) Regression Planner:

1. **Backward state-space search** is also called as Regression Planner.
2. In Regression Planner, processing will start from the finishing state and then you will go backwards to the initial state.
3. So basically we try to backtrack the scenario and find out the best possibility, in-order to achieve the goal.
4. In forward state space search we used to need information about the successors of the current state now, for backward state-space search we will need information about the predecessors of the current state.
5. Here the problem is that there can be many possible goal states which are equally acceptable.
6. That is why this approach is not considered as a practical approach when there are large numbers of states which satisfy the goal.
7. Consider the below example in figure 3.4, here you can see that the goal state is flight 1 is at location B and flight 2 is also at location B.

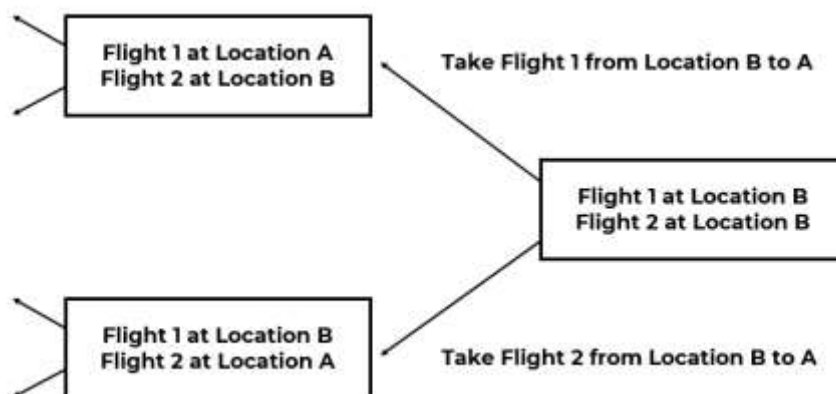


Figure 3.4: State Space Graph of Regression Planner

8. If this state is checked backwards; we have two acceptable states in one state only flight 2 is at location B, but flight 1 is at location A and similarly in 2nd possible state flight 1 is already at location B, but flight 2 is at location A.
9. As we search backwards from goal state to initial state, we have to deal with partial information about the state, since we do not yet know what actions will get us to goal.
10. This method is complex because we have to achieve a conjunction of goals.

THE SPARE TIRE PROBLEM:

1. There are just four actions:
 - a. Removing the spare from the trunk.
 - b. Removing the flat tire from the axle.
 - c. Putting the spare on the axle.
 - d. Leaving the car unattended overnight.
2. We assume that the car is parked in a particularly bad neighborhood, so that the effect of leaving it overnight is that the tires disappear.
3. The ADL description of the problem is shown below.
4. It is purely propositional.
5. **Initial State:** A flat tire on the axle and a good spare tire in the trunk.
6. **Goal State:** Have a good spare tire properly mounted onto the car's axle.

Init($\text{At(Flat, Axle)} \wedge \text{At(Spare, Trunk)}$)

Goal(At(Spare, Axle))

Action(Remove(Spare, Trunk),
 PRECOND: At(Spare, Trunk)
 EFFECT: $\neg \text{At(Spare, Trunk)} \wedge \text{At(Spare, Ground)}$)

Action(Remove(Flat, Axle),
 PRECOND: At(Flat, Axle)
 EFFECT: $\neg \text{At(Flat, Axle)} \wedge \text{At(Flat, Ground)}$)

Action(PutOn(Spare, Axle),
 PRECOND: $\text{At(Spare, Ground)} \wedge \neg \text{At(Flat, Axle)}$
 EFFECT: $\neg \text{At(Spare, Ground)} \wedge \text{At(Spare, Axle)}$)

Action(LeaveOvernight,
 PRECOND:
 EFFECT: $\neg \text{At(Spare, Ground)} \wedge \neg \text{At(Spare, Axle)} \wedge \neg \text{At(Spare, Trunk)}$
 $\wedge \neg \text{At(Flat, Ground)} \wedge \neg \text{At(Flat, Axle)}$)

-- EXTRA QUESTIONS --**Q1. Give PEAS Descriptors for WUMPUS World****Ans:****[P | Medium]****PEAS Description of the Wumpus World Environment:****Performance Measure:**

1. +1000 for picking up the gold,
2. -1000 for falling into a pit,
3. -1 for each action taken,
4. -10 for using up the arrow.

Environment:

1. 4x4 Grid of rooms.
2. Initially agent is in room labeled [1, 1] facing to the right.
3. Locations of PITs, wumpus and gold are chosen randomly.
4. Each square other than the start can be a pit with probability 0.2.

Actuators: Agent can perform five different actions.

1. Move forward/ backward.
2. Turn left 90 degrees.
3. Turn right 90 degrees.
4. Grab gold.
5. Shoot wumpus.

Sensors: Agent has five different sensors.

1. Breeze Sensor (Room adjacent to pit)
2. Stench Sensor (Room adjacent to wumpus)
3. Glitter Sensor (Room with gold)
4. Bump Sensor (Collide with wall)
5. Scream Sensor (Wumpus dead)

WUMPUS World Agent has following characteristics:

Fully Observable, Deterministic, Episodic, Static, Discrete & Single Agent.

Q2. Explain the steps involved in converting the propositional logic statement into CNF with a suitable example.**Ans:****[P | Medium]****PROPOSITIONAL LOGIC:**

1. Propositional Logic is the **simplest logic**.
2. All higher order logic is based on it.
3. It has less expression abilities.

4. The world is expressed as facts.
5. Fact can either be true or false.

CAUSAL NORMAL FORM:

1. It is also known as **Conjunctive Normal Form**.
2. CNF is the simplest representation of sentences.
3. CNF is the easy to work for computer.
4. CNF is used in **resolution by Refutation Method**.

Rules for converting sentences in CNF:

1. Convert bi-condition into implications or eliminate all \leftrightarrow connectives.

$$(P \leftrightarrow Q) \Rightarrow ((P \rightarrow Q) \wedge (Q \rightarrow P))$$

2. Remove Implication or eliminate all \rightarrow connectives.

$$(P \rightarrow Q) \Rightarrow (\neg P \vee Q)$$

3. Bring negation inside (DE Morgan's Law)

$$\neg \neg P \Rightarrow P$$

$$\neg (P \vee Q) \Rightarrow \neg P \wedge \neg Q$$

$$\neg (P \wedge Q) \Rightarrow \neg P \vee \neg Q$$

$$\neg (\forall x)P \Rightarrow (\exists x) \neg P$$

$$\neg (\exists x)P \Rightarrow (\forall x) \neg P$$

4. **Universal Elimination:** Drop universal Quantifiers as it is.

Example: Everyone likes Ice-cream.

$$\forall x \text{ Likes } (X, \text{Ice-cream})$$

$$\text{CNF: Likes } (X, \text{Ice-cream})$$

5. **Existential Elimination:** First replace the variable by skolem constant and then drop Existential Quantifiers.

Example: Somebody likes Ice-cream

$$\exists x \text{ likes } (X, \text{Ice-cream})$$

$$\text{CNF: Likes } (\text{Sachin}, \text{Ice-cream})$$

Q3. Explain Resolution by Refutation

Ans: [P | Medium]

RESOLUTION BY REFUTATION:

1. Resolution by Refutation is the **most powerful method of reasoning**.
2. It is used in **PROLOG**.
3. It begins by contradicting / refuting the fact.
4. If fact "F" is to be proved then it starts with " $\neg F$ "
5. It contradicts all other rules in the knowledge base.

6. The process stops when it returns "Null Clause"
7. Once Null or Empty Clause is reached, the fact is said to be proved.
8. It uses **CNF sentences**.
9. Facts which are connected by AND (\wedge) are treated as two separate facts.
10. The procedure becomes easy for "Horn Clause"
11. In Horn Clause at least one term is positive.
12. **Example:** $(\neg P \vee Q)$ is a Horn Clause.

Rules:

1. Fact can either be true or false, but not both simultaneously.
2. $P \wedge Q$ is true means P is true and Q is true.
3. $(\neg P \vee Q)$ is true.

Q4. Consider the following axioms:

All people who are graduating are happy. All happy people smile.

Someone is graduating.

- (i) **Represent these axioms in first order predicate logic.**
- (ii) **Convert each formula to clause form.**
- (iii) **Prove that "Is someone smiling?" using resolution technique. Draw the resolution tree.**

Ans: **[P | Medium]**

FIRST ORDER PREDICATE LOGIC (FOPL):

1. First Order Predicate Logic is **based on top – of propositional logic**.
2. It is also called as first order logic calculus.
3. It is more expensive than predictive logic.
4. It uses all **five connectors** of propositional logic.
5. It uses two quantifiers: **Universal Quantifier ($\forall x$) and Existential Quantifier ($\exists x$)**

EXAMPLE:

- a. All people who are graduating are happy.
FOPL: $\forall (x) \text{ Graduating } (x) \rightarrow \text{Happy } (x)$
- b. All happy people smile.
FOPL: $\forall (x) \text{ Happy } (x) \rightarrow \text{Smiling } (x)$
- c. Someone is graduating.
FOPL: $\exists (x) \text{ graduating } (x)$

CLAUSE FORM:

1. It is also known as **Conjunctive Normal Form**.
2. CNF is the simplest representation of sentences.
3. CNF is the easy to work for computer.
4. CNF is used in **resolution by Refutation Method**.

Steps:**1. Eliminate all \rightarrow connectives:**

- $$\forall (x) \neg \text{Graduating}(x) \vee \text{Happy}(x)$$
- $$\forall (x) \neg \text{Happy}(x) \vee \text{Smiling}(x)$$
- $$\exists (x) \text{Graduating}(x)$$

2. Standardize variable apart:

- $$\forall (x) \neg \text{Graduating}(x) \vee \text{Happy}(x)$$
- $$\forall (y) \neg \text{Happy}(y) \vee \text{Smiling}(y)$$
- $$\exists (z) \text{Graduating}(z)$$

3. Eliminate \exists (Skolemization):

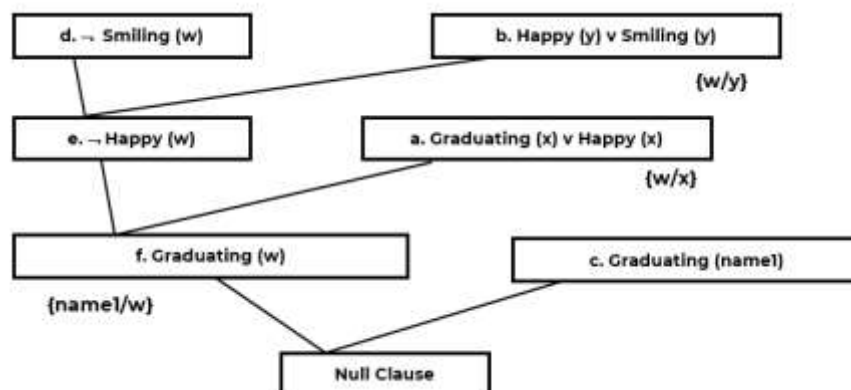
- $$\forall (x) \neg \text{Graduating}(x) \vee \text{Happy}(x)$$
- $$\forall (y) \neg \text{Happy}(y) \vee \text{Smiling}(y)$$
- $$\text{Graduating}(\text{name1})$$
- Note: name1 is the Skolemization constant.

4. Drop all \forall :

- $$\neg \text{Graduating}(x) \vee \text{Happy}(x)$$
- $$\neg \text{Happy}(y) \vee \text{Smiling}(y)$$
- $$\text{Graduating}(\text{name1})$$

RESOLUTION TREE:

- Resolution is the most powerful method of reasoning.
- It is used in PROLOG.
- It begins by contradicting / refuting the fact.
- If fact "F" is to be proved then it starts with " $\neg F$ ".
- It contradicts all other rules in the knowledge base.
- The process stops when it returns "Null Clause".
- Figure 3.5 shows the resolution tree for the given example.

**Figure 3.5: Resolution Tree**

Q5. Represent the following statement into FOPL.

- (i) Anyone who kills an animal is loved by no one.
- (ii) A Square is breezy if and only if there is a pit in a neighboring square (Assume the wumpus world environment)
- (iii) Give the PEAS description for an Internet Shopping Agent. Characterize its environment.

Ans: [P | Medium]

a. Anyone who kills an animal is loved by no one.

FOPL: $\forall (x) (\exists (y) \text{Animal}(y) \wedge \text{Kills}(x, y)) \rightarrow (\forall (z) \neg \text{Loves}(z, x))$

b. A Square is breezy if and only if there is a pit in a neighboring square (Assume the wumpus world environment)

FOPL: In Wumpus World Problem, let P be Pit & B be Breeze.

For R₂: $B(x, y) \leftrightarrow (P(x, y) \vee P(y, x))$

For R₃: $B(y, x) \leftrightarrow (P(x, x) \vee P(x, y) \vee P(z, x))$

c. Give the PEAS description for an Internet Shopping Agent. Characterize its environment.

PEAS for Internet Shopping Agent:

1. **Performance Measure:** Price, quality, appropriateness, efficiency.
2. **Environment:** Current and future WWW sites, vendors, shippers.
3. **Actuators:** Display to user, follow URL, fill in form.
4. **Sensors:** HTML pages (text, graphics, and scripts)

Environment Type:

Partially Observable, Deterministic, Sequential, Dynamic, Discrete and Multiagent.

Q6. Write first order logic statement for following statements:

- (i) If a perfect square is divisible by a prime p then it is also divisible by square of p.
- (ii) Every perfect square is divisible by some prime.
- (iii) Alice does not like Chemistry and History.
- (iv) If it is Saturday and warm, then Sam is in the park.
- (v) Anything anyone eats and is not killed by is food.

Ans: [P | Medium]

EXAMPLE:

a. If a perfect square is divisible by a prime p then it is also divisible by square of p.

FOL: $\forall (x, y) \text{Perfect_Square}(x) \wedge \text{Prime}(y) \wedge \text{Divides}(x, y) \rightarrow \text{Divides}(x, \text{Square}(y))$

b. Every perfect square is divisible by some prime.

FOL: $\forall (x) \exists (y) \text{Perfect_Square}(x) \wedge \text{Prime}(y) \wedge \text{Divides}(x, y)$

c. Alice does not like Chemistry and History.

FOL: $\neg \text{Likes}(\text{Alice}, \text{Chemistry}) \wedge \neg \text{Likes}(\text{Alice}, \text{History})$

d. If it is Saturday and warm, then Sam is in the park.

FOL: $\text{Day}(\text{Saturday}) \wedge \text{Warm}(\text{Saturday}) \rightarrow \text{Location}(\text{Sam}, \text{Park})$

e. Anything anyone eats and is not killed by is food.

FOL: $\forall (x) (\exists (y) \text{Eats}(y, x) \wedge \neg \text{KilledBy}(y, x)) \rightarrow \text{Food}(x)$

Q7. Write a PROLOG program to find Fibonacci Series.

Ans:

[P | Medium]

PROLOG:

1. PROLOG stands for **Programming in logic**.
2. PROLOG is one of the most widely used programming languages in artificial intelligence research.
3. It is a **declarative programming language**.
4. Instead of giving all statements in program, only rules are provided, which is used to answer any query.
5. PROLOG is not useful for solving mathematical problems.

FIBONACCI SERIES:

1. Fibonacci Series is a series of numbers in which each number is the sum of the two preceding numbers.
2. The simplest is the series 1, 1, 2, 3, 5, 8, etc.

FLOW CHART:

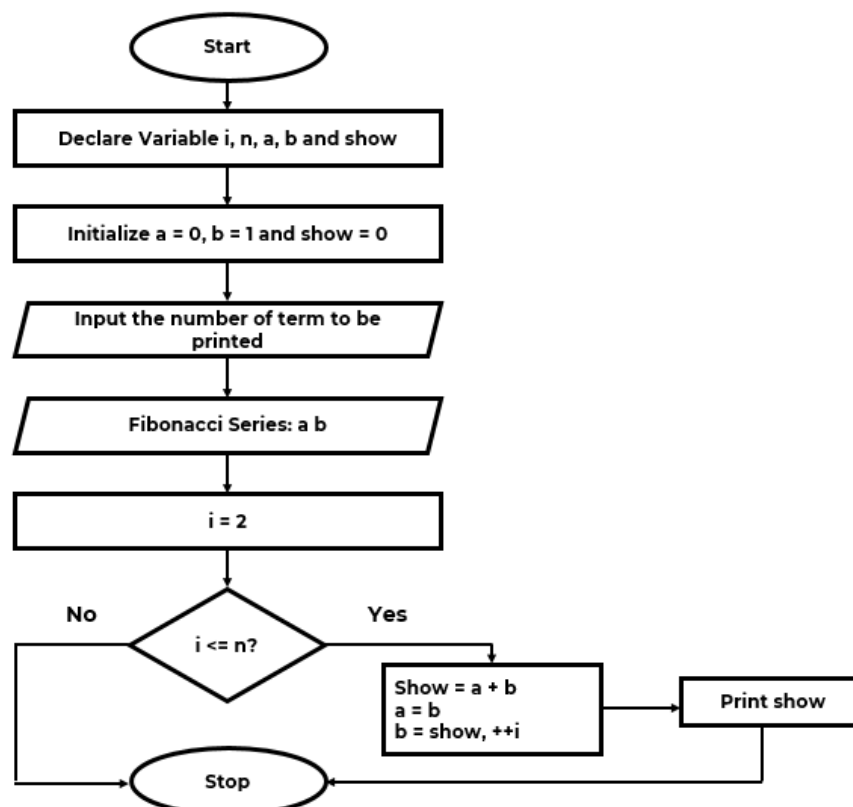


Figure 3.6: Flow Chart for Fibonacci Series using PROLOG.

ALGORITHM:

1. Start.
2. Declare variable I, a, b and show.
3. Initialize the variable, a = 0, b = 1 and show = 1.
4. Enter the number of terms of Fibonacci series to be printed.

5. Print First Two terms of series.
6. Use loop for following steps:
 - a. Show = a + b
 - b. a = b
 - c. b = show
 - d. Increment the value of l by 1 each time
 - e. Print the show variable value

PROGRAM:

```

Domains
    x = integer
Predicates
    Fibonacci (x)
Clauses
    Fibonacci (l).
Fibonacci (N):-
    N1 = N - 1,
    N1 >= 0, !,
    Fibonacci (N1),
    Write (F1, " ,"),
    F = F1 + N.
  
```

Input:

Fibonacci (10)

Output:

1, 1, 2, 3, 5, 8, 13, 21, 44, 63

Q8. What is Uncertainty? Explain Bayesian Network with example.

Ans:

[P | Medium]

UNCERTAINTY:

1. Problems with First-order logic is that the agents almost never have access to the whole truth about their environment.
2. So Agent cannot find a **categorical answer** to some important questions.
3. The agent therefore acts under **Uncertainty**.
4. **Uncertainty** is a summary of all that which is not explicitly taken into account in the agent's Knowledge Base.
5. **For Example**, a wumpus agent will unable to discover which of two squares contains a pit. If those squares en route to the gold then agent might take a chance and enter one of the two squares.

SOURCES OF UNCERTAINTY:

1. **Incompleteness and Incorrectness** in agents understanding of properties of the environment.

2. **Laziness and Ignorance** in storing knowledge as it is inescapable in complex, dynamic or in-accessible world.

TYPES:

I) Uncertainty in prior knowledge:

E.g., some causes of a disease are unknown and are not represented in the background knowledge of a medical-assistant agent.

II) Uncertainty in actions:

E.g., actions are represented with relatively short lists of preconditions, while these lists are in fact arbitrary long.

III) Uncertainty in perception:

E.g., sensors do not return exact or complete information about the world; a robot never knows exactly its position.

HANDLING THE UNCERTAINTY:

1. Consider the example of Diagnosis for medicine, which is a task which involves uncertainty.
2. Dental Diagnosis system using First-order Logic as follows:

$$\forall p \text{ symptom}(p, \text{Toothache}) \rightarrow \text{disease}(p, \text{cavity})$$

3. But not all patients have toothache because of cavity some have other problems as-

$$\forall p \text{ symptom}(p, \text{Toothache}) \rightarrow \text{disease}(p, \text{cavity}) \vee \text{disease}(p, \text{gum disease}) \vee \text{disease}(p, \text{Impacted_Wisdom_tooth}) \dots$$

4. Hence, to make the rule true, an unlimited list of possible causes must be added.

BAYESIAN BELIEF NETWORK:

1. Bayesian Belief Network is also known as **Bayes Network**.
2. It is a **probabilistic graphical model**.
3. It represents a set of random variables and their conditional dependencies via a directed acyclic graph (DAG).
4. It is used to represent probabilistic relationships between different classes.

EXAMPLE OF BAYESIAN BELIEF NETWORK:

Consider the example where we want to reason about the relationship between smoking and lung cancer. Figure 3.7 shows Bayesian Belief Network which includes 5 Boolean random variables representing:

- a. Has lung cancer (C)
- b. Smokes (S)
- c. Has a reduced life expectancy (RLE)
- d. Exposed to secondhand smoke (SHS)
- e. At least one parent smokes (PS)

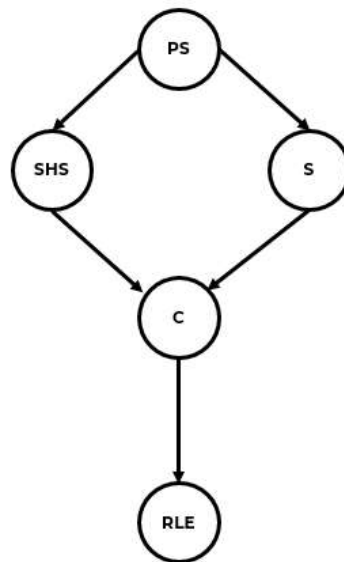


Figure 3.7: Bayesian Belief Network.

Q9. The gauge reading at a nuclear power station shows high values if the temperature of the core goes very high. The gauge also shows high value if the gauge is faulty. A high reading in the gauge sets an alarm off. The alarm can also go off if it is faulty. The probability of faulty instruments is low in a nuclear power plant.

(i) Draw the Bayesian Belief Network for the above situation.

(ii) Associate a conditional probability table for each node.

Ans:

[P | Medium]

BAYESIAN BELIEF NETWORK:

Refer Q1 – BBN Part

EXAMPLE:

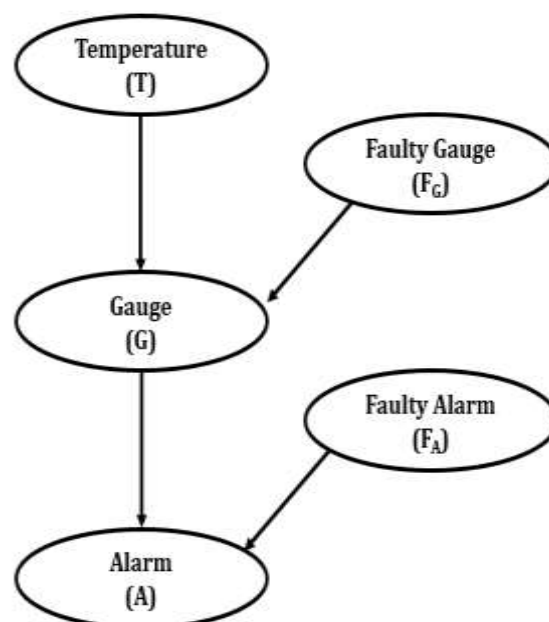


Figure 3.8: Bayesian Belief Network.

Consider the Boolean variables:

- T → Core temperature (normal, high).
 G → Gauge that records T (normal, high).
 F_G → Faulty gauge (true, false).
 F_G → Faulty alarm (true, false).
 A → Alarm that sounds when gauge reading is high (on, off).

1. Now situation is that the gauge is more likely to fail when the core temperature gets too high.
2. Probability that G gives correct temperature when it is working is x, and when faulty, y.
3. Alarm works correctly when not faulted. But when faulted, it never rings.
4. Figure 3.8 shows the Bayesian Belief Network for above example.

CONDITIONAL PROBABILITY TABLE (CPT):

1. Conditional Probability Table (CPT) is defined for a set of discrete random variables to demonstrate marginal probability of a single variable with respect to the others.
2. For example, assume there are three random variables X_1 , X_2 and X_3 where each have K states.
3. Then, the conditional probability table of X_1 provides the marginal probability values for $P(X_1 | X_2, X_3)$
4. Conditional Probability Table for each node is shown below.

	T = Normal		T = High	
	F_G	$\neg F_G$	F_G	$\neg F_G$
G = Normal	Y	X	$1 - Y$	$1 - X$
G = High	$1 - Y$	$1 - X$	Y	X

	G = Normal		G = High	
	F_A	$\neg F_A$	F_A	$\neg F_A$
T = Normal	0	0	0	1
T = High	1	1	1	0

Q10. Illustrate forward chaining and backward chaining in propositional logic with example.**Ans:****[P | Medium]****FORWARD CHAINING:**

1. Forward chaining is also known as **Forward Reasoning**.
2. It is one of the two main methods of reasoning.
3. It can be described logically as repeated application of **modus ponens**.
4. Forward-chaining is a **data-driven approach**.
5. Forward chaining starts with the available data and uses inference rules to extract more data until a goal is reached.
6. Forward chaining is a popular implementation strategy for expert systems, business and production rule systems.

EXAMPLE OF FORWARD CHAINING IN PROPOSITIONAL LOGIC:

Figure 3.9 shows example of forward chaining in propositional logic.

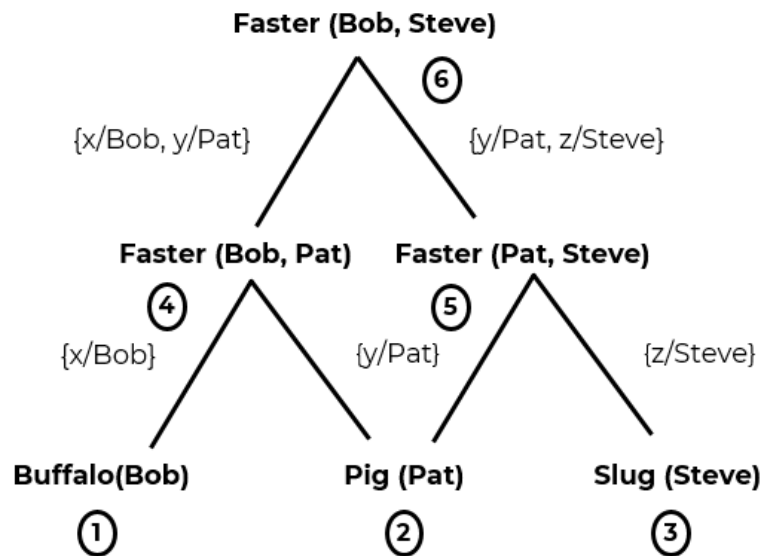


Figure 3.9: Example of forward chaining in propositional logic.

Rules:

1. $\text{Buffalo}(x) \wedge \text{Pig}(y) \Rightarrow \text{Faster}(x, y)$
2. $\text{Pig}(y) \wedge \text{Slug}(z) \Rightarrow \text{Faster}(y, z)$
3. $\text{Faster}(x, y) \wedge \text{Faster}(y, z) \Rightarrow \text{Faster}(x, z)$

Facts:

1. $\text{Buffalo}(\text{Bob})$
2. $\text{Pig}(\text{Pat})$
3. $\text{Slug}(\text{Steve})$

New facts:

1. $\text{Faster}(\text{Bob}, \text{Pat})$
2. $\text{Faster}(\text{Pat}, \text{Steve})$
3. $\text{Faster}(\text{Bob}, \text{Steve})$

BACKWARD CHAINING:

1. Backward chaining is also known as **Backward Reasoning**.
2. It is one of the two main methods of reasoning.
3. Backward chaining is an inference method that can be described as working backward from the goal(s).
4. It is used in automated theorem provers, inference engines, proof assistants and other artificial intelligence applications.
5. Backward chaining is implemented in logic programming by **SLD resolution**.
6. Backward chaining systems usually employ a depth-first search strategy.

EXAMPLE OF BACKWARD CHAINING IN PROPOSITIONAL LOGIC:

Figure 3.10 shows example of backward chaining in propositional logic.

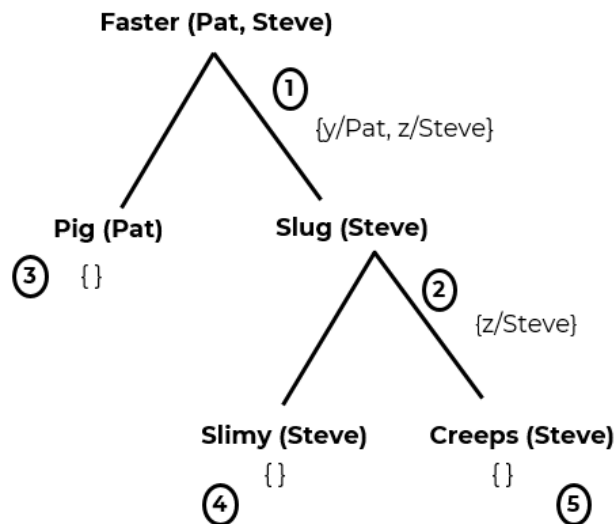


Figure 3.10: Example of backward chaining in propositional logic.

Rules:

1. $\text{Pig}(y) \wedge \text{Slug}(z) \Rightarrow \text{Faster}(y, z)$
2. $\text{Slimy}(a) \wedge \text{Creeps}(a) \Rightarrow \text{Slug}(a)$

Facts:

1. $\text{Pig}(\text{Pat})$
2. $\text{Slimy}(\text{Steve})$
3. $\text{Creeps}(\text{Steve})$

New facts:

1. $\text{Faster}(\text{Pat}, \text{Steve})$
2. $\text{Slug}(\text{Steve})$

Q11. Define Belief Network. Explain conditional Independence relation in Belief Network with example

Ans: [P | Medium]

BAYESIAN BELIEF NETWORK:

1. Bayesian Belief Network is also known as **Bayes Network**.
2. It is a **probabilistic graphical model**.
3. It represents a set of random variables and their conditional dependencies via a directed acyclic graph (DAG).
4. It is used to represent probabilistic relationships between different classes.

EXAMPLE OF BAYESIAN BELIEF NETWORK:

Consider the example where we want to reason about the relationship between smoking and lung cancer. Figure 3.11 shows Bayesian Belief Network which includes 5 Boolean random variables representing:

- a. Has lung cancer (C)
- b. Smokes (S)
- c. Has a reduced life expectancy (RLE)
- d. Exposed to secondhand smoke (SHS)
- e. At least one parent smokes (PS)

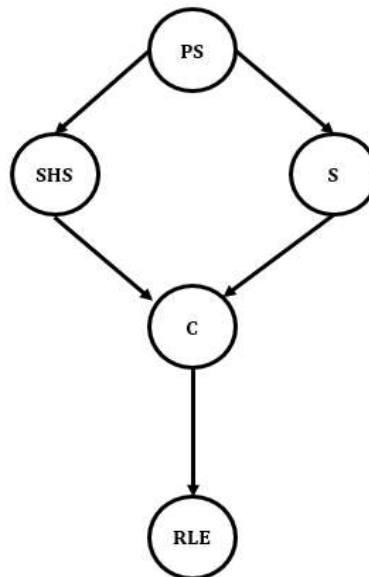


Figure 3.11: Bayesian Belief Network.

CONDITIONAL INDEPENDENCE RELATION IN BELIEF NETWORK:

1. A Bayesian network is a graphical representation of conditional independence and conditional probabilities.
2. **Causal chains** give rise to conditional independence.
3. **Example:** Smoking causes cancer, which causes dyspnoea.

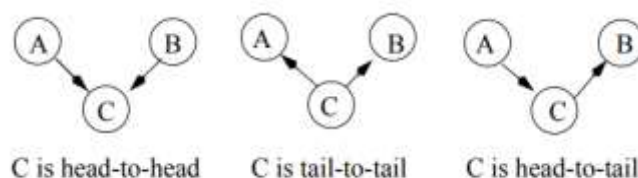


4. Common causes or ancestors also give rise to conditional independence.
5. **Example:** Cancer is the common cause of the two symptoms i.e. positive Xray & dyspnoea.
6. So informally, a variable is conditionally independent of another, if your belief in the value of the latter wouldn't influence your belief in the value of the former.
7. By the same token, a variable is conditionally independent of another, given some information, if your belief in the value of the second variable wouldn't influence your belief in the value of the first, given that you have the information.

Example:

1. Let A, B and C be three disjoint sets of variables.
2. A is said to be conditionally independent of B given C iff

$$P(A | B, C) = P(A | C) \text{ for all possible values of A, B and C.}$$
3. If every undirected path from A to B is blocked by C then **$I(A, B | C)$**
4. For conditional independence, **d-separation test** is used.



5. A path is blocked if:
 - a. There is a node C which is head-to-tail with respect to the path.
 - b. There is a node C which is tail-to-tail with respect to the path.
6. There is a node that is head-to-head and neither the node, nor any of its descendants, are in C.

Q12. Explain a partial order planner with an example.

Ans: **[P | High]**

PARTIAL ORDER PLANNER:

1. Planning is the process of thinking about and organizing the activities required to achieve a desired goal.
2. Any planning algorithm that can place two actions into a plan without specifying which comes first is called as a **Partial Order Planner**.
3. A Partial Order Planner searches through plane space.
4. It starts with an initial plan representing the start and finishing steps, and on each iteration adds one more step.
5. If it leads to incomplete plan, it backtracks and tries another branch of search space.
6. The solution is represented as a graph of actions, not a sequence of actions.

Example:

Let us define following two macros for the sake of simplicity for block world example.

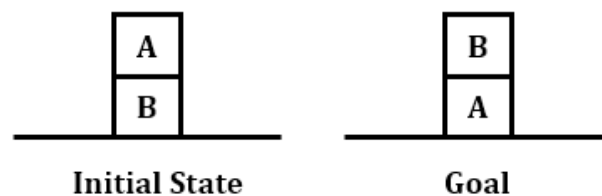


Figure 3.12: Partial Order Planning Block World Example.

Note: MOT (Move on Table)

Macro operator	Description
MOT (A)	Move A onto table
MOVE (B, A)	Move B onto A

To achieve the Goal State ON (A, B)

- a. Move 'A' onto table should occur before move 'B' to 'A'.
- b. Hence MOT (A) should come before MOVE (B, A) in the Final Plan.

PARTIAL ORDER PLANNER GRAPH:

1. It contains the dummy actions START & FINISH to mark the beginning and end of the plan in the graph.
2. The planner can generate total plans from the graph.
3. Figure 3.13 Partial Order Planning Graph for Block World Example.

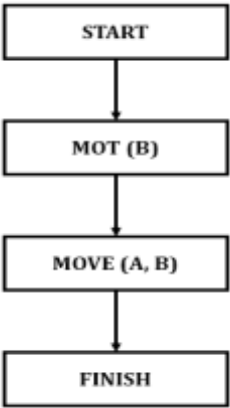


Figure 3.13: Partial Order Planning Graph.

Q13. Differentiate between forward and backward chaining

Ans: [P | Medium]

Table 3.1: Forward chaining v/s Backward chaining.

Forward Chaining	Backward Chaining
It starts with new data.	It starts with some goal or hypothesis.
It asks few questions.	It asks many questions.
It examines all rules.	It examines some rules.
Slow approach.	Fast approach.
Gather larger information from small amount of data.	It produce small amount of information from available data.
Forward Chaining is primarily data driven.	Backward Chaining is primarily Goal Driven.
It uses its input. It searches rules for answers.	It proves the considered hypothesis.
It is a form of Top-Down reasoning.	It is a form of bottom up reasoning.
Works forward to find conclusions from facts.	Works backward to find facts that support the hypothesis.
It tends to breath – first.	It tends to depth – first.

CHAP - 4: FUZZY LOGIC

Q1. Give different membership functions of fuzzy logic

Ans:

[5M | Dec-19]

MEMBERSHIP FUNCTION:

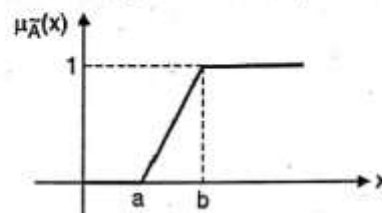
1. Membership functions were first introduced in 1965 by Lofti A. Zadeh in his first research paper "fuzzy sets".
2. A function that specifies the degree to which a given input belongs to a set is known as **Membership Function**.
3. Membership functions characterize fuzziness, whether the elements in fuzzy sets are **discrete or continuous**.
4. Membership functions are used in the fuzzification and defuzzification steps of a FLS (fuzzy logic system), to map the non-fuzzy input values to fuzzy linguistic terms and vice versa
5. Membership functions are **represented by graphical forms**.

TYPES OF MEMBERSHIP FUNCTION:

I) Increasing MFs (T Function):

An increasing MF is specified by two parameters (a, b) as follows:

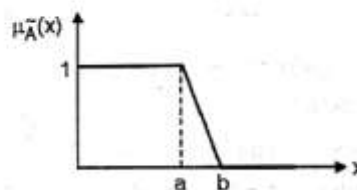
$$T(x; a, b) = \begin{cases} 0 & ; x \leq a \\ (x - a)/(b - a) & ; a \leq x \leq b \\ 1 & ; x \geq b \end{cases}$$



II) Decreasing MF (L Function):

A decreasing MF is specified by two parameters (a, b) as follows:

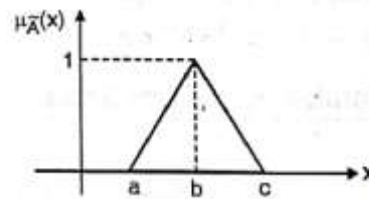
$$L(x; a, b) = \begin{cases} 1 & ; x \leq a \\ (b - x)/(b - a) & ; a \leq x \leq b \\ 0 & ; x \geq b \end{cases}$$



III) Triangular MF (^ Function):

A triangular MF is specified by three parameters (a, b, c) as follows:

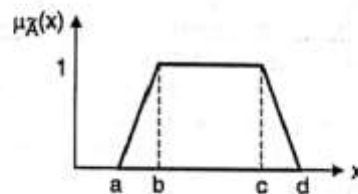
$$\mu_{\tilde{A}}(x; a, b, c) = \begin{cases} 0 & ; x \leq a \\ (x-a)/(b-a) & ; a \leq x \leq b \\ (c-x)/(c-b) & ; b \leq x \leq c \\ 0 & ; x \geq c \end{cases}$$



IV) Trapezoidal MF (π Function):

A Trapezoidal MF is specified by four parameters (a, b, c, d) as follows:

$$\text{Trapezoid}(x; a, b, c, d) = \begin{cases} 0 & ; x \leq a \\ (x-a)/(b-a) & ; a \leq x \leq b \\ 1 & ; b \leq x \leq c \\ (d-x)/(d-c) & ; c \leq x \leq d \\ 0 & ; x \geq d \end{cases}$$



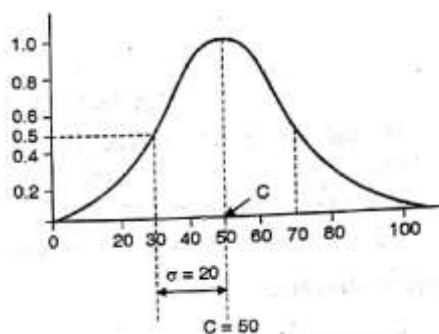
An alternative expression using min and max can be given as,

$$\mu_{\text{trapezoid}} = \max\left(\min\left(\frac{x-a}{b-a}, 1, \frac{d-x}{d-c}\right), 0\right)$$

V) Gaussian MFs:

A Gaussian MF is specified by two parameters { c, σ }

$$\text{Gaussian}(x; c, \sigma) = e^{-1/2 \left(\frac{x-c}{\sigma} \right)^2}$$

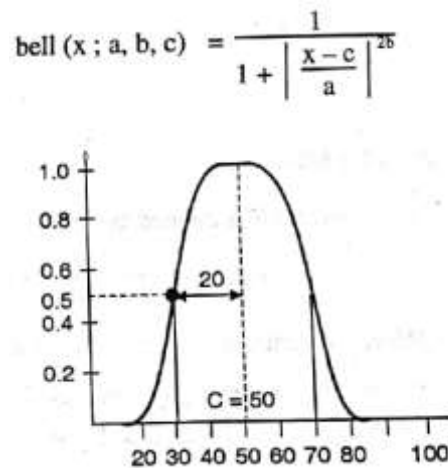


Gaussian (x; 50, 20) MF

- c represents MFs center
- σ determines MFs Width

VI) Generalized Bell MF / Cauchy MF:

A Generalized Bell MF is specified by three parameters (a, b, c)

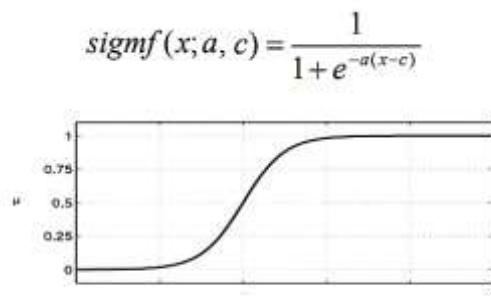


A desired generalized bell MF can be obtained by a proper selection of the parameters a, b, c

- c specifies the center of a bell MF
- a specifies the width of a bell MF
- b determines the slope at the crossover points

VII) Sigmoidal MFs:

A sigmoidal MF is defined by,



Where, a controls the slope at the crossover point $x = c$.

- Depending on the sign of the parameter a, a sigmoidal MF is open right or open left and thus is appropriate for representing concepts such as “Very Large” or “Very Negative”
- They are widely used as the activation function in artificial neural networks.

Q2. ANFIS

Ans:

[5M | Dec-19]

ANFIS:

1. ANFIS stands for **Adaptive Neuro Fuzzy Inference System**.
2. It is a kind of artificial neural network that is based on Takagi–Sugeno fuzzy inference system.
3. ANFIS was introduced by Jang.
4. ANFIS is used for modeling, controlling, and parameter estimation in complex systems.
5. ANFIS is a combination of artificial neural network (ANN) and fuzzy inference system (FIS).

- Combining the ANN and fuzzy-set theory can provide advantages and overcome the disadvantages in both techniques.

FEATURES OF ANFIS:

- It refines fuzzy IF-THEN rules to describe the behavior of a complex system;
- It does not require prior human expertise;
- It is easy to implement;
- It enables fast and accurate learning;
- It offers desired data set; greater choice of membership functions to use; strong generalization abilities; excellent explanation facilities through fuzzy rules; and
- It is easy to incorporate both linguistic and numeric knowledge for problem solving.

ANFIS ARCHITECTURE:

- In order to explain the ANFIS architecture, we assumed that there are two inputs: x and y .
- Two fuzzy if-then rules for a first-order Sugeno fuzzy model can be expressed as follows:
Rule 1: If x is A_1 and y is B_1 , then $f_1 = p_1x + q_1y + r_1$,
Rule 2: If x is A_2 and y is B_2 , then $f_2 = p_2x + q_2y + r_2$,
- Where A_i and B_i are the fuzzy sets, f_i is the output, and p_i , q_i , and r_i are the design parameters that are determined during the training process.
- The ANFIS architecture used to implement the two rules is shown in figure 4.1

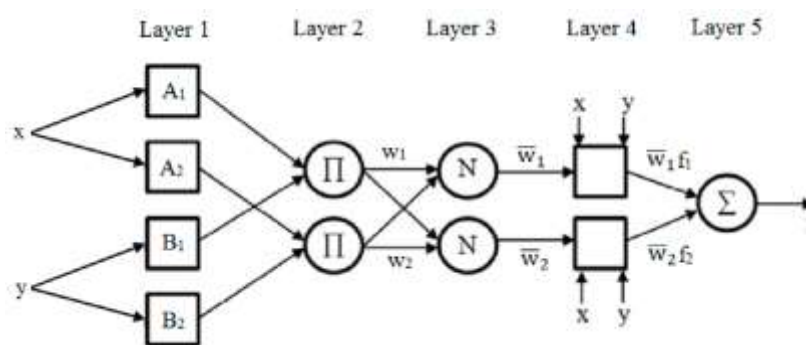


Figure 4.1: ANFIS Architecture with two inputs, one output, and two rules

- The structure of ANFIS includes five layers that can be explained as follows, where O_i^j represents the output of the i^{th} node and j^{th} layer.
- Layer 1:** In this layer, each node represents a node function:

$$O_i^1 = \mu_{A_i}(x), \quad \text{for } i = 1, 2$$

Where x is the input to the i^{th} node. A_i is the linguistic label (cold, warm, etc.) characterized by proper membership functions. Some of the membership functions are triangular, trapezoidal, and Gaussian.

- Layer 2:** In this layer, each node calculates the firing strength of a rule by multiplication:

$$O_i^2 = w_i = \mu_{A_i}(x) * \mu_{B_i}(y), \quad i = 1, 2$$

- Layer 3:** In this layer, firing strengths that were evaluated in the previous layer are normalized to distinguish between the firing strengths of each rule from the total firing strengths of total rules:

$$O_i^3 = \bar{w}_i = \frac{w_i}{w_1 + w_2}, \quad i = 1, 2$$

9. **Layer 4:** In this layer, node i calculates the contribution of i^{th} rule to the overall output:

$$O_i^4 = \bar{w}_i * f_i = \bar{w}_i(p_i * x + q_i * y + r_i)$$

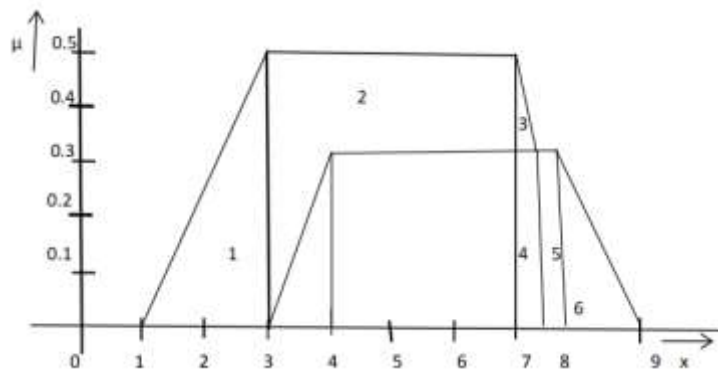
Where \bar{w}_i is the output of Layer 3, and the parameter set is $\{p_i, q_i, r_i\}$.

10. **Layer 5:** In this layer, the single node calculates the overall output as the total contribution from each rule:

$$O_1^5 = \sum_i \bar{w}_i * f_i = \frac{\sum_i w_i * f_i}{\sum_i w_i}$$

11. The ANFIS has two learning algorithms, back-propagation and hybrid methods, which try to minimize the error between the observed and predicted data.

Q3. Explain defuzzification techniques. Apply defuzzification by using Center of Gravity (CoG) method on the following:



Ans:

[10M | Dec-19]

DEFUZZIFICATION:

1. Defuzzification is the process of converting a fuzzy set into a crisp value.
2. The defuzzified value in FLC (Fuzzy Logic Controller) represents the action to be taken in controlling the process.
3. Defuzzification process can also be treated as the rounding off process, where fuzzy set having a group of membership values on the unit interval reduced to a single scalar quantity.
4. The output of a fuzzy process may be union of two or more fuzzy membership functions.
5. In such cases we need to find crisp value as a representative of the entire fuzzy MF.

DEFUZZIFICATION TECHNIQUES:

I) Centroid Method:

1. It is also known as center of Area Method.
2. This method provides a crisp value based on the center of gravity of the fuzzy set.
3. The total area of the membership function distribution used to represent the combined control action is divided into a number of sub-areas.

4. The area and the center of gravity or centroid of each sub-area is calculated and then the summation of all these sub-areas is taken to find the defuzzified value for a discrete fuzzy set.
5. For discrete membership function, the defuzzified value denoted as X^* using COG is defined as:

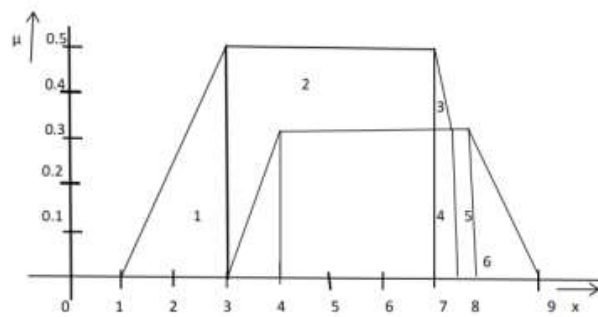
$$x^* = \frac{\sum_{i=1}^n x_i \cdot \mu(x_i)}{\sum_{i=1}^n \mu(x_i)}$$

Here x_i indicates the sample element, $\mu(x_i)$ is the membership function, and n represents the number of elements in the sample.

6. For continuous membership function, X^* is defined as:

$$x^* = \frac{\int x \mu_A(x) dx}{\int \mu_A(x) dx}$$

7. This method is most preferred and physically appealing of all the Defuzzification methods.
8. **Example:**



The defuzzified value X^* using COG is defined as:

$$x^* = \frac{\sum_{i=1}^N A_i \times \bar{x}_i}{\sum_{i=1}^N A_i}$$

- Here N indicates the number of sub-areas, A_i and \bar{x}_i represents the area and centroid of area, respectively, of i^{th} sub-area.
- In the aggregated fuzzy set as shown above, the total area is divided into six sub-areas.
- For COG method, we have to calculate the area and centroid of area of each sub-area.
- These can be calculated as below.

The total area of the sub-area 1 is $\frac{1}{2} \times 2 \times 0.5 = 0.5$

The total area of the sub-area 2 is $(7-3) \times 0.5 = 4 \times 0.5 = 2$

The total area of the sub-area 3 is $\frac{1}{2} \times (7.5-7) \times 0.2 = 0.5 \times 0.5 \times 0.2 = .05$

The total area of the sub-area 4 is $0.5 \times 0.3 = 0.15$

The total area of the sub-area 5 is $0.5 \times 0.3 = 0.15$

The total area of the sub-area 6 is $\frac{1}{2} \times 1 \times 0.3 = 0.15$

- Now the centroid or center of gravity of these sub-areas can be calculated as:

Centroid of sub-area 1 will be $(1+3+3)/3 = 7/3 = 2.333$

Centroid of sub-area 2 will be $(7+3)/2 = 10/2 = 5$

Centroid of sub-area 3 will be $(7+7+7.5)/3 = 21.5/3 = 7.166$

Centroid of sub-area 4 will be $(7+7.5)/2 = 14.5/2 = 7.25$

Centroid of sub-area 5 will be $(7.5+8)/2 = 15.5/2 = 7.75$

Centroid of sub-area 6 will be $(8+8+9)/3 = 25/3 = 8.333$

- Now we can calculate A_i and \bar{x}_i and is shown in below table 4.1

Sub Area Number	Area (A_i)	Centroid of Area (\bar{x}_i)	$A_i \bar{x}_i$
1	0.5	2.333	1.1665
2	02	5	10
3	0.05	7.166	0.3583
4	0.15	7.25	1.0875
5	0.15	7.75	1.1625
6	0.15	8.333	1.2499

The defuzzified value X^* will be

$$X^* = \frac{\sum_{i=1}^N A_i \times \bar{x}_i}{\sum_{i=1}^N A_i}$$

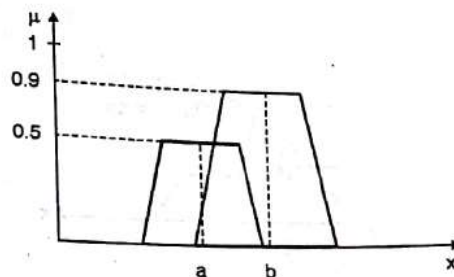
$$\begin{aligned}
 &= (1.1665 + 10 + 0.3583 + 1.0875 + 1.1625 + 1.2499) / (0.5 + 2 + 0.05 + 0.15 + 0.15 + 0.15) \\
 &= (15.0247)/3 \\
 &= 5.008 \\
 &X^* = 5.008
 \end{aligned}$$

II) Weighted Average Method:

- This method is valid for fuzzy sets with symmetrical output membership functions and produces results very close to the COA method.
- This method is less computationally intensive.
- Each membership function is weighted by its maximum membership value.
- The defuzzified value is defined as

$$X^* = \frac{\sum_{i=1}^n \mu_{\bar{F}}(x_i) \cdot x_i}{\sum_{i=1}^n \mu_{\bar{F}}(x_i)}$$

5. Example:



$$\text{Defuzzified value } X^* = [(a \times 0.5) + (b \times 0.9)] / (0.5 + 0.9)$$

III) Center of Sums:

1. This is the most commonly used defuzzification technique.
2. This method involves the algebraic sum of individual output fuzzy sets, instead of their union.
3. In this method, the overlapping area is counted twice.
4. The defuzzified value x^* is defined as:

$$x^* = \frac{\sum_{i=1}^N x_i \cdot \sum_{k=1}^n \mu_{A_k}(x_i)}{\sum_{i=1}^N \sum_{k=1}^n \mu_{A_k}(x_i)}$$

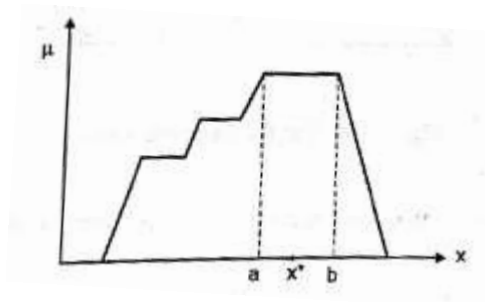
Here, n is the number of fuzzy sets, N is the number of fuzzy variables, $\mu_{A_k}(x_i)$ is the membership function for the k -th fuzzy set.

IV) Mean of Max (Middle of Maxima):

1. In this method, the defuzzified value is taken as the element with the highest membership values.
2. When there are more than one element having maximum membership values, the mean value of the maxima is taken.
3. Let A be a fuzzy set with membership function $\mu_A(x)$ defined over $x \in X$, where X is a universe of discourse.
4. The defuzzified value is let say x^* of a fuzzy set and is defined as,

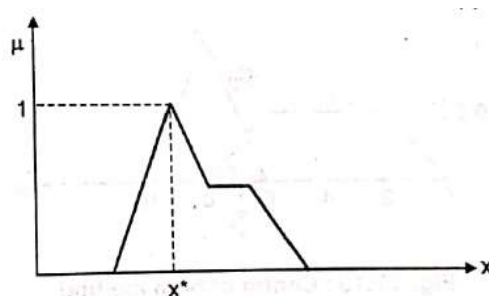
$$x^* = \frac{\sum_{x_i \in M} x_i}{|M|}$$

Here, $M = \{x_i \mid \mu_A(x_i) \text{ is equal to the height of the fuzzy set } A\}$ and $|M|$ is the cardinality of the set M

5. Example:

Defuzzified value:

$$x^* = \frac{a+b}{2}$$

V) Max Membership Principle:

1. This method is limited to peak output functions.
2. It uses the individual clipped or scaled central outputs.
3. The defuzzified value X^* is defined as:

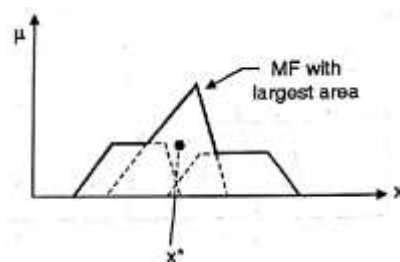
$$\mu_{\tilde{c}}(x^*) \geq \mu_{\tilde{c}}(x) \text{ for all } x \in X$$

VI) Center of Largest:

1. It is used only for non-convex Fuzzy set.
2. If multiple waveforms are there then calculate the area of individual waveform and take the center of largest area as the final answer.
3. The defuzzified value X^* is defined as:

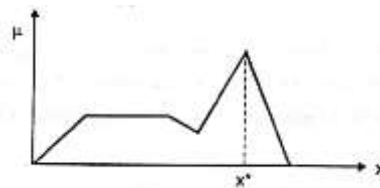
$$x^* = \frac{\int \mu_{\tilde{c}_m}(x) \cdot x \, dx}{\int \mu_{\tilde{c}_m}(x) \cdot dx}$$

Where C_m is the convex fuzzy subset that has the largest area.



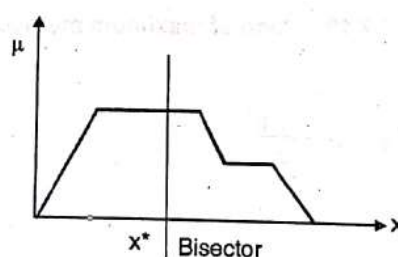
VII) First/Last of Maxima:

1. It is very similar to mean of max only that first and last of maximize is consider.
2. This method uses the overall output (i.e. Union of all individual output MF)
3. When there is only one height; $X^* = \text{Height}$ it is same as mean of max.



VIII) Center of Area / Bisector of Area Method (BOA):

1. This method uses the vertical line that divides the region into two equal area as shown below.
2. This Line is known as Bisector.
3. This method calculates the position under the curve where the areas on both sides are equal.



Q4. Explain fuzzy controller system for a tipping example. Consider service and food quality rated between 0 and 10. Use this to leave a tip of 25%

Ans: [10M | Dec-19]

THE TIPPING PROBLEM:

1. The 'tipping problem' is commonly used to illustrate the power of fuzzy logic principles to generate complex behavior from a compact, intuitive set of expert rules.
2. We would formulate this problem as:

a. Inputs:

▪ Service:

- **Universe (i.e., crisp value range):** How good was the service of the wait staff, on a scale of 0 to 10?
- **Fuzzy set (i.e., fuzzy value range):** poor, acceptable, amazing

▪ Food Quality:

- **Universe:** How tasty was the food, on a scale of 0 to 10?
- **Fuzzy set:** bad, decent, great.

b. Outputs:

▪ Tip:

- **Universe:** How much should we tip, on a scale of 0% to 25%
- **Fuzzy set:** Low, Medium & High.

c. Rules:

- IF service is poor or the food is rancid, THEN tip is cheap.
- IF service is good, THEN tip is average.
- IF service is excellent or food is delicious, THEN tip is generous.

d. Usage:

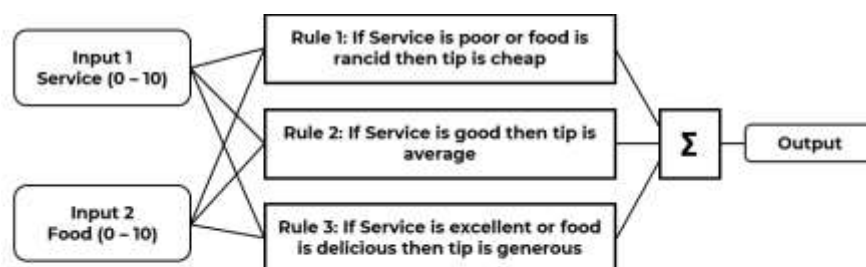
▪ If I tell this controller that I rated:

- The service as 3, and
- The quality as 8,

▪ It would recommend I leave:

- A 16.7% tip.

FUZZY LOGIC SYSTEM WITH 2 INPUTS & 1 OUTPUT FOR TIPPING PROBLEM:

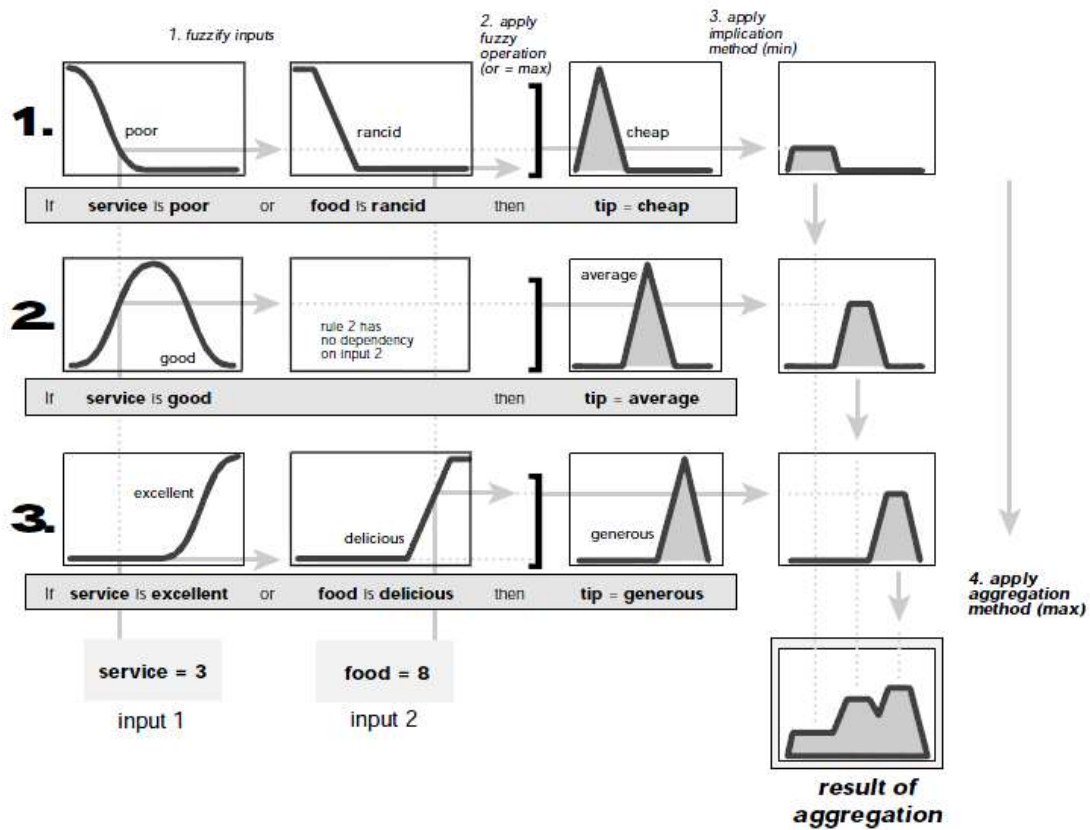


The inputs are crisp (non-fuzzy) numbers limited to a specific range

All rules are evaluated in parallel using fuzzy reasoning

The results of the rules are combined and distilled (defuzzified)

The result is a crisp (non-fuzzy) number

IMPLEMENTATION OF FLS:

Q5. Determine (alfa) α -level sets and strong α -level sets for the following fuzzy sets.
 $A = \{ (1, 0.2), (2, 0.5), (3, 0.8), (4, 1), (5, 0.7), (6, 0.3) \}$

Ans:

[5M | Dec-19]

The following are α -level sets:

$$A_{0.2} = \{ 1, 2, 3, 4, 5, 6 \}$$

$$A_{0.3} = \{ 2, 3, 4, 5, 6 \}$$

$$A_{0.5} = \{ 2, 3, 4, 5 \}$$

$$A_{0.7} = \{ 3, 4, 5 \}$$

$$A_{0.8} = \{ 3, 4 \}$$

$$A_1 = \{ 4 \}$$

Following are strong α -level sets:

$$A_{0.2}' = \{ 2, 3, 4, 5, 6 \}$$

$$A_{0.3}' = \{ 2, 3, 4, 5 \}$$

$$A_{0.5}' = \{ 3, 4, 5 \}$$

$$A_{0.7}' = \{ 3, 4 \}$$

$$A_{0.8}' = \{ 4 \}$$

$$A_1' = \emptyset$$

-- EXTRA QUESTIONS --**Q1. Fuzzy Set****Ans:****[P | Medium]****FUZZY SET:**

1. Fuzzy set is a set having degrees of membership between 1 and 0.
2. Fuzzy sets are represented with tilde character (~).
3. For example, Number of cars following traffic signals at a particular time out of all cars present will have membership value between [0, 1].
4. Partial membership exists when member of one fuzzy set can also be a part of other fuzzy sets in the same universe.
5. The degree of membership or truth is not same as probability, fuzzy truth represents membership in vaguely defined sets.
6. A fuzzy set \tilde{A} in the universe of discourse, U , can be defined as a set of ordered pairs and it is given by:

$$\tilde{A} = \{(x, \mu_{\tilde{A}}(x)) | x \in X\}$$

7. When the universe of discourse, U , is **discrete and finite**, fuzzy set \tilde{A} is given by:

$$\tilde{A} = \sum_{i=1}^n \frac{\mu_{\tilde{A}}(x_i)}{x_i} = \frac{\mu_{\tilde{A}}(x_1)}{x_1} + \frac{\mu_{\tilde{A}}(x_2)}{x_2} + \dots + \frac{\mu_{\tilde{A}}(x_n)}{x_n}$$

$$\tilde{A} = \int \frac{\mu_{\tilde{A}}(x)}{x}$$

8. Where "n" is a finite value.
9. Fuzzy sets also satisfy every property of classical sets.

COMMON OPERATIONS ON FUZZY SETS:Given two Fuzzy sets \tilde{A} and \tilde{B} **I) Union:**Fuzzy set \tilde{C} is union of Fuzzy sets \tilde{A} and \tilde{B}

$$\tilde{C} = \tilde{A} \cup \tilde{B}$$

$$\mu_{\tilde{C}}(x) = \max(\mu_{\tilde{A}}(x), \mu_{\tilde{B}}(x))$$

II) Intersection:Fuzzy set \tilde{D} is intersection of Fuzzy sets \tilde{A} and \tilde{B}

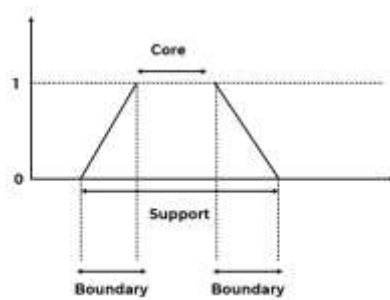
$$\tilde{D} = \tilde{A} \cap \tilde{B}$$

$$\mu_{\tilde{D}}(x) = \min(\mu_{\tilde{A}}(x), \mu_{\tilde{B}}(x))$$

III) Complement:Fuzzy set \tilde{E} is complement of Fuzzy set \tilde{A}

$$\tilde{E} = \complement_{\tilde{A}} X$$

$$\mu_{\tilde{E}}(x) = 1 - \mu_{\tilde{A}}(x)$$

Q2. Features of Membership function**Ans:****[P | Medium]****FEATURES OF MEMBERSHIP FUNCTIONS:****I) Core:**

For any fuzzy set \tilde{A} , the core of a membership function is that region of universe that is characterized by full membership in the set. Hence, core consists of all those elements y of the universe of information such that,

$$\mu_{\tilde{A}}(y) = 1$$

II) Support:

For any fuzzy set \tilde{A} , the support of a membership function is the region of universe that is characterized by a nonzero membership in the set. Hence, core consists of all those elements y of the universe of information such that,

$$\mu_{\tilde{A}}(y) > 0$$

III) Boundary:

For any fuzzy set \tilde{A} , the boundary of a membership function is the region of universe that is characterized by a nonzero but incomplete membership in the set. Hence, core consists of all those elements y of the universe of information such that,

$$1 > \mu_{\tilde{A}}(y) > 0$$

Q3. Explain Fuzzy Inference System.**Ans:****[P | Medium]****FUZZY INFERENCE SYSTEM (FIS):**

1. Fuzzy Inference System is the key unit of a fuzzy logic system having decision making as its primary work.
2. It uses the "IF...THEN" rules along with connectors "OR" or "AND" for drawing essential decision rules.

CHARACTERISTICS:

1. The output from FIS is always a fuzzy set irrespective of its input which can be fuzzy or crisp.
2. It is necessary to have fuzzy output when it is used as a controller.
3. A defuzzification unit would be there with FIS to convert fuzzy variables into crisp variables.

FIS ARCHITECTURE:

Figure 4.2 shows the FIS Architecture.

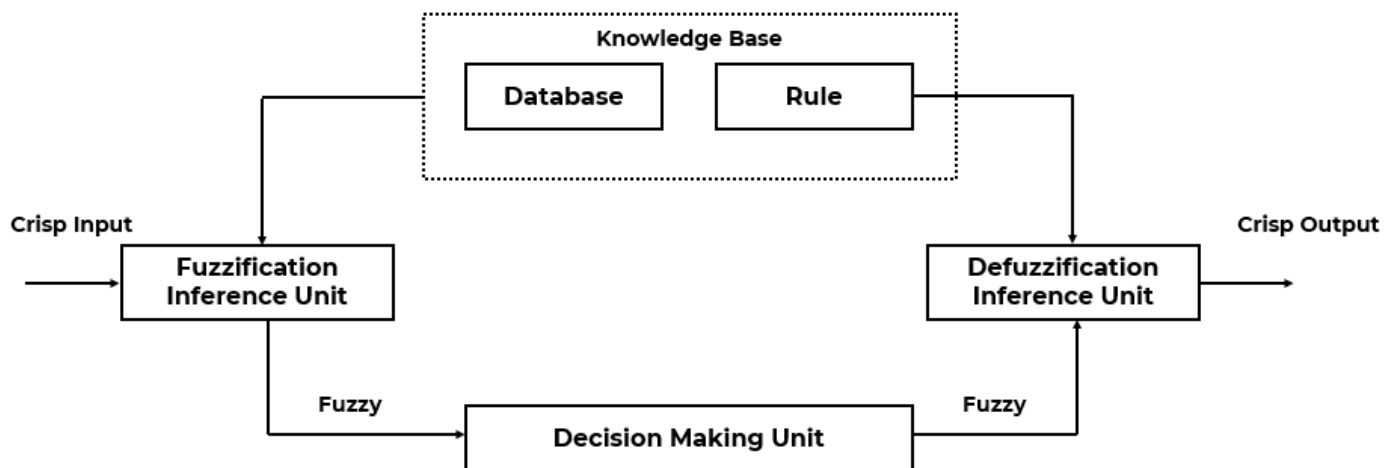


Figure 4.2: FIS Architecture.

FUNCTIONAL BLOCKS OF FIS:

The following five functional blocks will help you understand the construction of FIS:

1. **Rule Base:** It contains fuzzy IF-THEN rules.
2. **Database:** It defines the membership functions of fuzzy sets used in fuzzy rules.
3. **Decision-making Unit:** It performs operation on rules.
4. **Fuzzification Interface Unit:** It converts the crisp quantities into fuzzy quantities.
5. **Defuzzification Interface Unit:** It converts the fuzzy quantities into crisp quantities. Following is a block diagram of fuzzy interference system.

WORKING:

1. A Fuzzification unit supports the application of numerous fuzzification methods, and converts the crisp input into fuzzy input.
2. A knowledge base - collection of rule base and database is formed upon the conversion of crisp input into fuzzy input.
3. The defuzzification unit fuzzy input is finally converted into crisp output.

Q4. Explain mamdani type of fuzzy inference systems in details

Ans:

[P | Medium]

MAMDANI FIS:

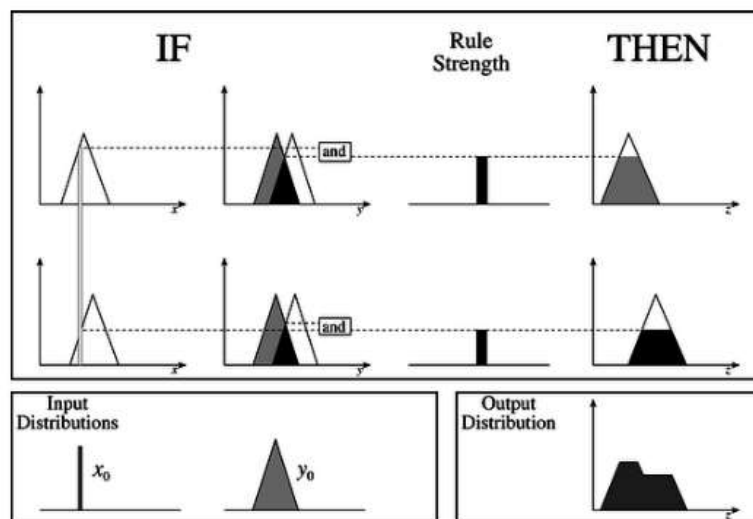
1. The Mamdani Fuzzy Inference System was proposed by Ebrahim Mamdani in 1975.
2. Mamdani FIS is used to control a steam engine and boiler combination by synthesizing a set of fuzzy rules.

To compute the output of this FIS given the inputs, six steps has to be followed:

1. Determine a set of Fuzzy Rules.
2. Fuzzily the inputs using the input membership functions.
3. Combine the fuzzified inputs according to the fuzzy rules to establish a rule strength (Fuzzy Operations).
4. Determine the consequence of the rule by combining the rule strength and the output membership function (Implication).
5. Combining the consequences to get an output distribution (Aggregation).
6. Defuzzify the output distribution (This step is only required if a crisp output (class) is needed).

Fuzzy Rule Composition in Mamdani Model:

1. In Mamdani FIS, The fuzzy rules are formed using IF-THEN statements and AND/OR connectives.
2. The consequent of the rule can be obtained in two steps.
 - a. By computing the strength of each rule.
 - b. By clipping the output membership function at the rule strength.
3. The outputs of all the fuzzy rules are then combined to obtain the aggregated fuzzy output.
4. Finally, defuzzification is applied on to the aggregated fuzzy output to obtain a crisp output value.
5. Consider two inputs, two rule Mamdani fuzzy inference system as shown in figure 4.3.
6. Assume two inputs are crisp value x and y .
7. Assume the following two rules:
 - a. **Rule 1:** If x is A_1 and y is B_1 then z is C_1
 - b. **Rule 2:** If x is A_2 and y is B_2 then z is C_2

**Figure 4.3: A Two Input, Two Rule Mamdani FIS with a Fuzzy Input.**

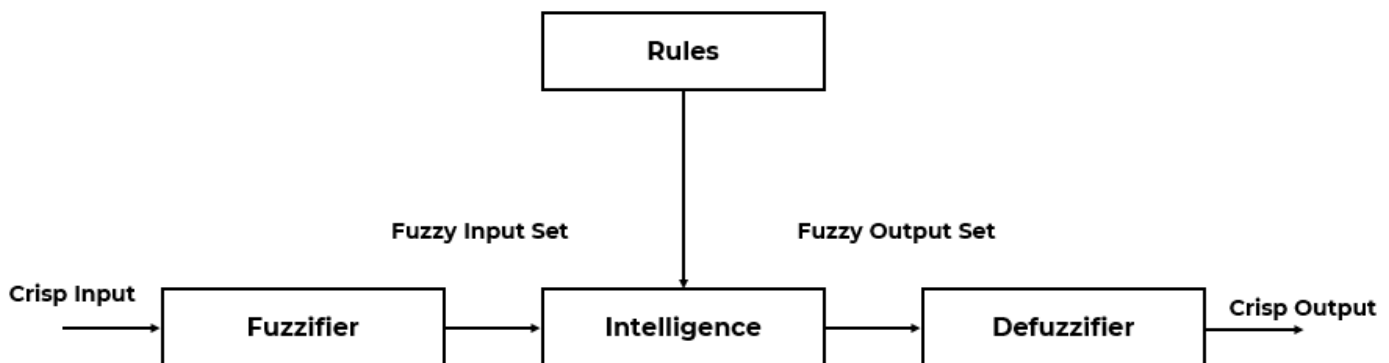
8. It Fuzzifies the two inputs by finding the intersection of the two crisp input values with the input membership function.
9. It uses the minimum operator to compute the Fuzzy Input “and” for combining the two fuzzified inputs to obtain a rule strength.
10. The output membership function is clipped at the rule strength.
11. Finally, the maximum operator is used to compute the Fuzzy Output “or” for combining the outputs of the two rules.

Q5. Explain Fuzzy Logic.**Ans:****[P | Medium]****FUZZY LOGIC:**

1. The term fuzzy mean things which are not very clear or vague.
2. In real life, we may come across a situation where we can't decide whether the statement is true or false.
3. At that time, fuzzy logic offers very valuable flexibility for reasoning.
4. We can also consider the uncertainties of any situation.
5. Fuzzy logic algorithm helps to solve a problem after considering all available data.
6. Then it takes the best possible decision for the given the input.
7. The FL method imitates the way of decision making in a human which consider all the possibilities between digital values T and F.

CHARACTERISTICS:

1. Flexible and easy to implement machine learning technique.
2. Helps you to mimic the logic of human thought.
3. Logic may have two values which represent two possible solutions.
4. Highly suitable method for uncertain or approximate reasoning.
5. Fuzzy logic views inference as a process of propagating elastic constraints.
6. Fuzzy logic allows you to build nonlinear functions of arbitrary complexity.
7. Fuzzy logic should be built with the complete guidance of experts.

ARCHITECTURE:**Figure 4.4: Fuzzy Logic Architecture.**

Fuzzy Logic architecture has four main parts as shown in the figure 4.4.

I) Rule Base:

1. It contains all the rules and the if-then conditions offered by the experts to control the decision-making system.
2. The recent update in fuzzy theory provides various methods for the design and tuning of fuzzy controllers.
3. This updates significantly reduce the number of the fuzzy set of rules.

II) Fuzzification:

1. Fuzzification step helps to convert inputs.
2. It allows you to convert, crisp numbers into fuzzy sets.
3. Crisp inputs measured by sensors and passed into the control system for further processing.
4. Example like Room temperature, pressure, etc.

III) Inference Engine:

1. It helps you to determine the degree of match between fuzzy input and the rules.
2. Based on the % match, it determines which rules need implement according to the given input field.
3. After this, the applied rules are combined to develop the control actions.

IV) Defuzzification:

1. At last the Defuzzification process is performed to convert the fuzzy sets into a crisp value.
2. There are many types of techniques available, so you need to select it which is best suited when it is used with an expert system.

Q6. Explain Fuzzy Reasoning.**Ans:****[P | Medium]****FUZZY REASONING**

1. The basic rule of inference in traditional two-value topic is modus ponens, according to which we can infer the truth of a proposition B from the truth of A and the implication $A \rightarrow B$.
2. For instance, if A is identified with "the tomato is red" and B with "the tomato is ripe," then if it is true that "the tomato is red," it is also true that "the tomato is ripe".
3. This concept is illustrated as follows:
 Premise 1 (fact): x is A
 Premise 2 (rule): if x is A then y is B
 Consequence (conclusion): y is B
4. However, in much of human reasoning, modus ponens is employed in an approximate manner.
5. For example, if we have the same implication rule "if the tomato is red, then it is ripe" and we know that "the tomato is more or less red," then we may infer that "the tomato is more or less ripe".
6. This is written as:
 Premise 1 (fact): x is A'
 Premise 2 (rule): if x is A then y is B
 Consequence (conclusion): y is B'
7. Where A' is close to A and B' is close to B.
8. When A, B, A', and B' are fuzzy sets of approximate universes, the foregoing inference procedure is called **approximate reasoning** or **fuzzy reasoning**;
9. It is also called **generalized modus ponens**, since it has modus ponens as special case.

CHAP - 5: ARTIFICIAL NEURAL NETWORK

Q1. Different types of Neural Networks

Ans:

[5M | Dec-19]

Note: For better understanding we have explained the below answer in detail; kindly cut shot the answer as per your understanding.

ARTIFICIAL NEURAL NETWORK (ANN):

1. An artificial neural network may be defined as an information-processing model that is inspired by the way biological nervous system, such as brain, process information.
2. ANN is a **computational model**.
3. This model tries to replicate only most basic functions of the brain.
4. ANN is composed of a large number of highly interconnected processing elements (neurons) working in unison to solve specific problems.
5. ANN like people, learn by example.
6. ANNs have three layers that are interconnected.
7. The first layer consists of input neurons.
8. Those neurons send data on to the second layer, which in turn sends the output neurons to the third layer.
9. Figure 5.1 shows the structure of Artificial Neural Network.

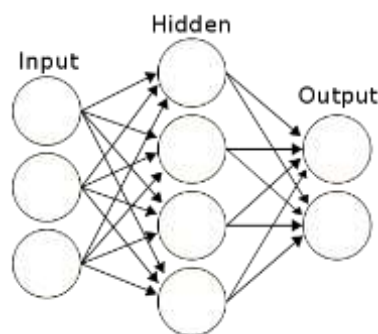


Figure 5.1: Structure of Artificial Neural Network

TYPES:

I) Single Layer Feed Forward Networks:

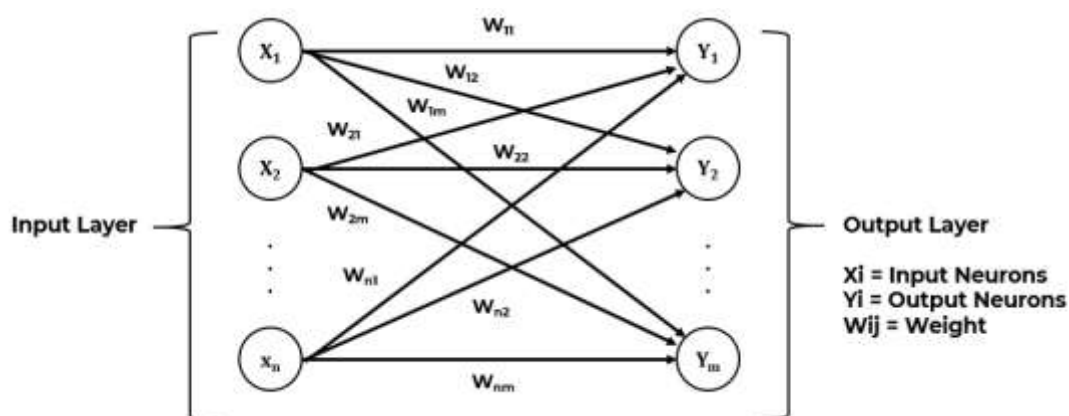


Figure 5.2: Single layer feedforward network.

1. Figure 5.2 represents single layer feedforward network.
2. A single layer feed forward network consists of neurons arranged in two layers.
3. The first layer is called the **input layer** and the second layer is called the **output layer**.
4. The input signals are fed to the neurons of the input layer and neurons of the output layer produce output signals.
5. Every input neuron is connected to every output neuron **via synaptic links or weights**.

II) Multilayer Feed Forward Network:

1. In a multilayer feed forward network, there are multiple layers.
2. Thus, besides having input layer and output layer, this network has one or more intermediary layers called **hidden layers**.
3. The computational units of the hidden layer are known as the **hidden neurons or hidden units**.
4. Before directing the input to the output layer the hidden layer performs useful intermediary computations.
5. Neurons in input layer are connected to the neurons in hidden layer and the weights on these connections are referred to as "**Input-Hidden Layer Weights**".
6. Similarly, the hidden layer neurons are connected to the output layer neurons and the corresponding weights are referred to as "**Hidden-Output Layer Weights**".
7. A multi-layer feed forward network with m input neurons, n_1 neurons in the first hidden layer, n_2 neurons in the second hidden layer and k output neurons is written as $m - n_1 - n_2 - k$ network.
8. Figure 5.3 represents multi-layer feedforward network.

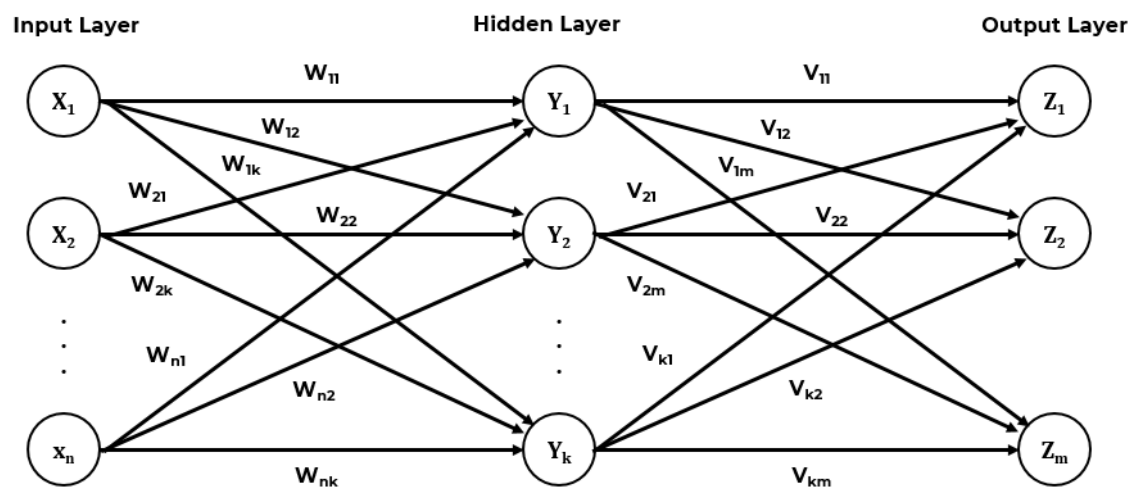


Figure 5.3: Multilayer feedforward network.

III) Single Layer Recurrent Network:

1. A network is said to be a feed-forward network if no neuron in the output layer is an input to a node in the same or in the preceding layer.
2. On the other hand, when outputs can be directed back as inputs to same or preceding layer nodes then it results in the formation of feedback networks.
3. If the feedback of the output of the processing elements is directed back as input to the processing elements in the same layer then it is called lateral feedback.
4. Recurrent networks are feedback networks with closed loop.

5. Figure 5.4, shows a simple recurrent neural network having a single neuron with feedback to itself.
6. Figure 5.5 shows a single-layer network with a feedback connection in which a processing element's output can be directed back to the processing element itself or to the other processing element or to both.



Figure 5.4: Single Node with own feedback.

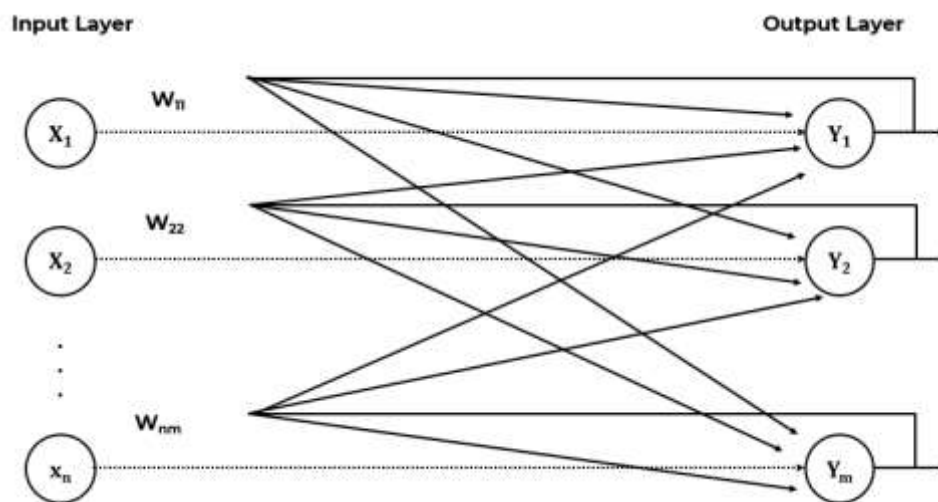


Figure 5.5: Single Layer Recurrent Network

IV) Multi-Layer Recurrent Network:

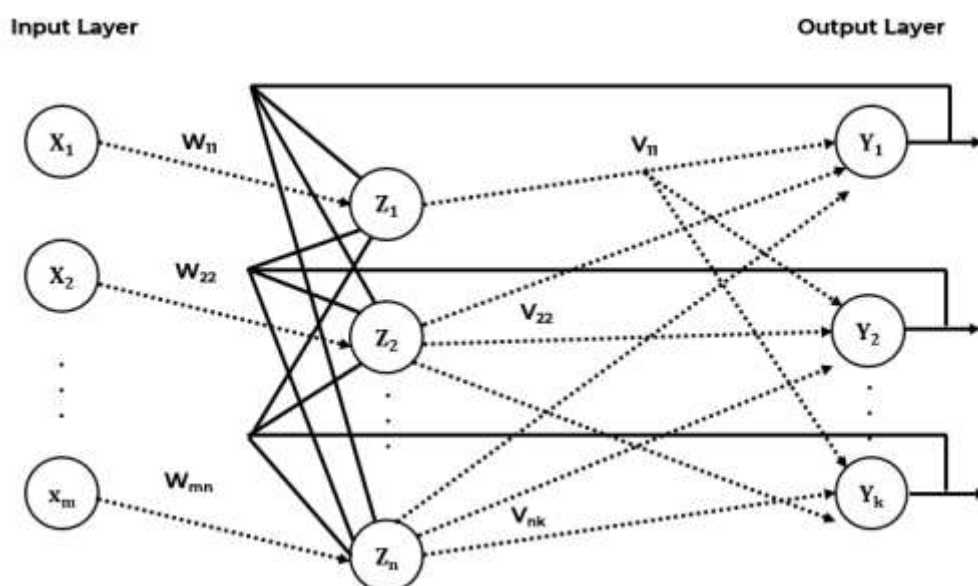


Figure 5.6: Multi-Layer Recurrent Network.

1. Figure 5.6 shows architecture of multilayer feedback network.
2. It can be noted that a processing element output can be directed back to the nodes in a preceding layer, forming a multilayer recurrent network.
3. They perform the same task for every element of a sequence, with the output being depended on the previous computations.
4. Inputs are not needed at each time step.
5. Also, in these networks, a processing element output can be directed back to the processing element itself and to other processing elements in the same layer.

Q2. Design a Mc-Culloh Pitts model for XOR Gate

Ans:

[10M | Dec-19]

MCCULLOCH-PITTS:

1. The first Computational Model for an artificial Neuron was proposed by McCulloch and Pitts in 1943.
2. The inputs and outputs are binary (exclusively ones and zeros), the nodes produce only binary results.
3. The McCulloch-Pitts model was an extremely simple artificial neuron.
4. The McCulloch-Pitts Neural Model is also known as **Linear Threshold Gate**.
5. It is a neuron of a set of inputs $I_1, I_2 \dots I_m$ and one output Y .
6. The linear threshold gate simply classifies the set of inputs into two different classes.
7. Thus the output Y is binary.
8. Such a function can be described mathematically using these equations:

$$Sum = \sum_{i=1}^N I_i W_i,$$

$$y = f(Sum)$$

W_1, W_2, \dots, W_m are weight values normalized in the range of either (0, 1) or (-1, 1).

9. Sum is the weighted sum, and T is a threshold constant.
10. The function 'f' is a linear step function at threshold T .
11. The symbolic representation of the linear threshold gate is shown in figure 5.7

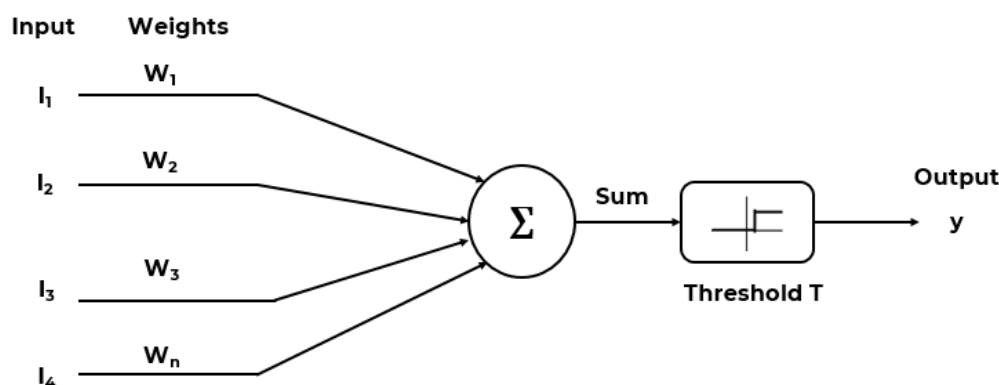


Figure 5.7: McCulloch – pitts neuron Representing Signal flow and the Operation.

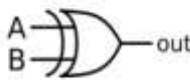
DESIGN A MC-CULLOH PITTS MODEL FOR XOR GATE:

1. It is sometimes called as XOR gate or exclusive or gate.
2. It gives a true output when the number of true inputs is odd.

3. If both the inputs are true and both are false then the output is false.
4. These are used to implement binary addition in computers.
5. The truth table and symbol are shown below:

X_1	X_2	Y
0	0	0
0	1	1
1	0	1
1	1	0

Truth Table



XOR Gate

Implementation:

XOR function cannot be represented by simple and single logic function; it is represented as

$$y = x_1 \bar{x}_2 + \bar{x}_1 x_2$$

$$y = z_1 + z_2$$

Where;

$$z_1 = x_1 \bar{x}_2$$

$$z_2 = \bar{x}_1 x_2$$

$$y = z_1 (OR) z_2$$

A single layer net is not sufficient to represent the function. An intermediate layer is necessary.

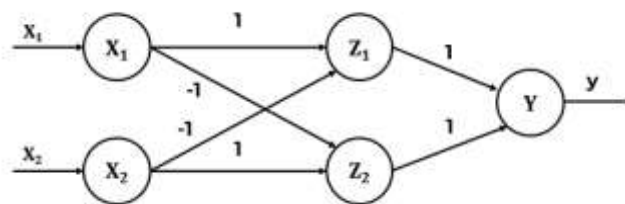


Figure 5.8: Neural Net for XOR Function

FIRST FUNCTION ($z_1 = x_1 \bar{x}_2$):

The truth table for function z_1 is shown below.

x_1	x_2	z_1
0	0	0
0	1	0
1	0	1
1	1	0

The net representation is given as

Case 1: Assume both weights as excitatory i.e.,

$$W_{11} = W_{21} = 1$$

Calculate the net inputs. For inputs,

$$(0, 0), z_{1in} = 0 \times 1 + 0 \times 1 = 0$$

$$(0, 1), z_{1in} = 0 \times 1 + 1 \times 1 = 1$$

$$(1, 0), z_{1in} = 1 \times 1 + 0 \times 1 = 1$$

$$(1, 1), z_{1in} = 1 \times 1 + 1 \times 1 = 2$$

Hence, it is not possible to obtain function z_1 using these weights.

Case 2: Assume one weights as excitatory and the other as inhibitory, i.e.,

$$W_{11} = 1; W_{21} = -1$$

Calculate the net inputs. For inputs,

$$(0, 0), z_{1in} = 0 \times 1 + 0 \times -1 = 0$$

$$(0, 1), z_{1in} = 0 \times 1 + 1 \times -1 = -1$$

$$(1, 0), z_{1in} = 1 \times 1 + 0 \times -1 = 1$$

$$(1, 1), z_{1in} = 1 \times 1 + 1 \times -1 = 0$$

On the basis of this calculated net input, it is possible to get the required output. Hence,

$$W_{11} = 1$$

$$W_{21} = -1$$

$$\Theta \geq 1 \text{ for the } z_1 \text{ neuron}$$

SECOND FUNCTION ($z_2 = \overline{x_1}x_2$):

The truth table for function z_2 is shown below.

x_1	x_2	z_2
0	0	0
0	1	1
1	0	0
1	1	0

The net representation is given as

Case 1: Assume both weights as excitatory i.e.,

$$W_{12} = W_{22} = 1$$

Calculate the net inputs. For inputs,

$$(0, 0), z_{2in} = 0 \times 1 + 0 \times 1 = 0$$

$$(0, 1), z_{2in} = 0 \times 1 + 1 \times 1 = 1$$

$$(1, 0), z_{2in} = 1 \times 1 + 0 \times 1 = 1$$

$$(1, 1), z_{2in} = 1 \times 1 + 1 \times 1 = 2$$

Hence, it is not possible to obtain function z_2 using these weights.

Case 2: Assume one weights as excitatory and the other as inhibitory, i.e.,

$$W_{12} = -1; W_{22} = 1$$

Calculate the net inputs. For inputs,

$$(0, 0), z_{2in} = 0 \times -1 + 0 \times 1 = 0$$

$$(0, 1), z_{2in} = 0 \times -1 + 1 \times 1 = 1$$

$$(1, 0), z_{2in} = 1 \times -1 + 0 \times 1 = -1$$

$$(1, 1), z_{2in} = 1 \times -1 + 1 \times 1 = 0$$

On the basis of this calculated net input, it is possible to get the required output. Hence,

$$W_{12} = -1$$

$$W_{22} = 1$$

$$\Theta \geq 1 \text{ for the } z_2 \text{ neuron}$$

THIRD FUNCTION (z_1 (OR) z_2):

The truth table for this function is shown below.

x_1	x_2	y	z_1	z_2
0	0	0	0	0
0	1	1	0	1
1	0	1	1	0
1	1	0	0	0

Here the net input is calculated using

$$y_{in} = z_1 v_1 + z_2 v_2$$

Case 1: Assume both weights as excitatory i.e.,

$$v_1 = v_2 = 1$$

Calculate the net inputs. For inputs,

$$(0, 0), y_{in} = 0 \times 1 + 0 \times 1 = 0$$

$$(0, 1), y_{in} = 0 \times 1 + 1 \times 1 = 1$$

$$(1, 0), y_{in} = 1 \times 1 + 0 \times 1 = 1$$

$$(1, 1), y_{in} = 0 \times 1 + 0 \times 1 = 0 \text{ (Because for } x_1 = 1 \text{ and } x_2 = 1, z_1 = 0 \text{ and } z_2 = 0)$$

Setting a threshold of $\Theta \geq 1$, $v_1 = v_2 = 1$, which implies that the net is recognized. Therefore, the analysis is made for XOR function using McCulloch-Pitts Neurons. Thus for XOR function, the weights are obtained as:

$$W_{11} = W_{22} = 1 \text{ (Excitatory)}$$

$$W_{12} = W_{21} = -1 \text{ (Inhibitory)}$$

$$v_1 = v_2 = 1 \text{ (Excitatory)}$$

Q3. Implement AND function using perceptron networks for bipolar inputs and targets**Ans:****[10M | Dec-19]****PERCEPTRON NETWORKS:**

1. Perceptrons are a type of artificial neuron that predates the sigmoid neuron.
2. Perceptron networks come under single layer feed forward networks.
3. It is also known as simple perceptrons.
4. The perceptron network consists of three units, namely, sensory units, associator unit, and response unit.
5. The sensory units are connected to associator units with fixed weights having values 1, 0 or -1, which are assigned at random.
6. The binary activation function is used in sensory unit and associator unit.
7. The response unit has an activation of 1, 0 or -1
8. The output of the perceptron network is given by

$$y = f(y_{in})$$

Where $f(y_{in})$ is activation function and is defined as

$$f(y_{in}) = \begin{cases} 1 & \text{.... If } y_{in} > \theta \\ 0 & \text{.... If } -\theta \leq y_{in} \leq \theta \\ -1 & \text{.... If } y_{in} < -\theta \end{cases}$$

IMPLEMENT AND FUNCTION USING PERCEPTRON NETWORKS FOR BIPOLAR INPUTS AND TARGETS:

The truth table for AND function with bipolar inputs and targets is shown below:

x_1	x_2	t
1	1	1
1	-1	-1
-1	1	-1
-1	-1	-1

The perceptron network, which uses perceptron learning rule, is used to train the AND function. The network architecture is shown below in figure 5.9. The input patterns are presented to the network one by one. When all the four input patterns are presented, then one epoch is said to be completed. The initial weights and threshold are set to zero, i.e., $w_1 = w_2 = b = 0$ and $\theta = 0$. The learning rate α is set to 1.

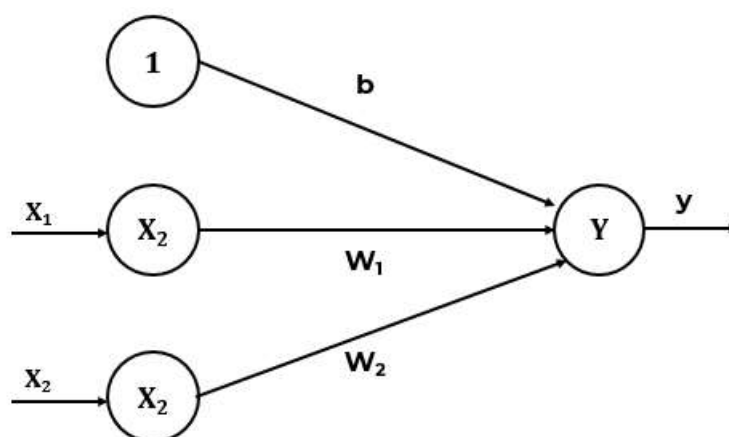


Figure 5.9: Perceptron network for AND function

For the first input pattern, $x_1 = 1$, $x_2 = 1$ and $t = 1$ with weights and bias, $w_1 = 0$, $w_2 = 0$ and $b = 0$

Calculate the net input:

$$\begin{aligned} y_{in} &= b + x_1 w_1 + x_2 w_2 \\ &= 0 + 1 \times 0 + 1 \times 0 \\ &= 0 \end{aligned}$$

The output y is computed by applying activations over the net input calculated:

$$\begin{aligned} f(y_{in}) &= 1 \quad \dots \text{If } y_{in} > 0 \\ &= 0 \quad \dots \text{If } y_{in} = 0 \\ &= -1 \quad \dots \text{If } y_{in} < 0 \end{aligned}$$

Here we have taken $\theta = 0$. Hence, when, $y_{in} = 0$, $y = 0$.

Check whether $t = y$. Here, $t = 1$ and $y = 0$, so $t \neq y$, hence weight updation takes place:

$$\begin{aligned} w_i(\text{new}) &= w_i(\text{old}) + \alpha t x_i \\ w_1(\text{new}) &= w_1(\text{old}) + \alpha t x_1 = 0 + 1 \times 1 \times 1 = 1 \\ w_2(\text{new}) &= w_2(\text{old}) + \alpha t x_2 = 0 + 1 \times 1 \times 1 = 1 \\ b(\text{new}) &= b(\text{old}) + \alpha t = 0 + 1 \times 1 = 1 \end{aligned}$$

Here, the changes in weights are

$$\begin{aligned} \Delta w_1 &= \alpha t x_1 \\ \Delta w_2 &= \alpha t x_2 \\ \Delta b &= \alpha t \end{aligned}$$

The weights $w_1 = 1$, $w_2 = 1$, $b = 1$ are the final weights after first input pattern is presented. The same process is repeated for all the input patterns. The process can be stopped when all the targets become equal to the calculated output or when a separating line is obtained using the final weights for separating the positive responses from negative responses. Table 5.1 shows the training of perceptron network until its target and calculated output converge for all the patterns.

Table 5.1

Input			Target	Net Input	Calculated Output	Weight changes			Weights		
x_1	x_2	1	(t)	(y_{in})	(y)	Δw_1	Δw_2	Δb	w_1 (0)	w_2 (0)	b (0)
EPOCH - 1											
1	1	1	1	0	0	1	1	1	1	1	1
1	-1	1	-1	1	1	-1	1	-1	0	2	0
-1	1	1	-1	2	1	1	-1	-1	1	1	-1
-1	-1	1	-1	-3	-1	0	0	0	1	1	-1

EPOCH - 2											
1	1	1	1	0	1	0	0	0	1	1	-1
1	-1	1	-1	1	-1	0	0	0	1	1	-1
-1	1	1	-1	2	-1	0	0	0	1	1	-1
-1	-1	1	-1	-3	-1	0	0	0	1	1	-1

The final weights and bias after second epoch are

$$w_1 = 1, w_2 = 1, b = -1$$

Since the threshold for the problem is zero, the equation of the separating line is

$$x_2 = -\frac{w_1}{w_2}x_1 - \frac{b}{w_2}$$

Here

$$w_1 x_1 + w_2 x_2 + b > 0$$

$$w_1 x_1 + w_2 x_2 + b > 0$$

Thus, using the final weights we obtain

$$x_2 = -\frac{1}{1}x_1 - \frac{(-1)}{1}$$

$$x_2 = -x_1 + 1$$

It can be easily found that the above straight line separates the positive response and negative response region, as shown in Figure 5.9. The same methodology can be applied for implementing other logic functions such as OR, AND NOT, NAND, etc. If there exists a threshold value $\theta \neq 0$, then two separating lines have to be obtained, i.e., one to separate positive response from zero and the other for separating zero from the negative response.

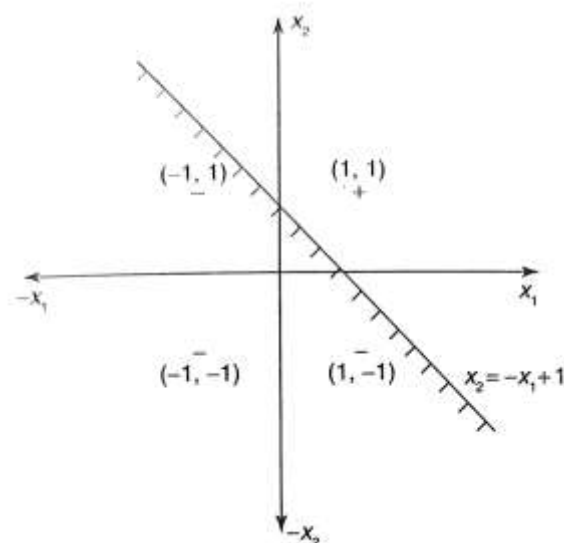


Figure 5.10: Decision boundary for AND function in perceptron training ($\theta = 0$)

Q4. Construct kohonen Self-organizing map to cluster the four given vectors

[0 0 1 1], [1 0 0 0], [0 1 1 0] and [0 0 0 1]. The number of cluster formed is two. Assume an initial learning rate of 0.5

Ans:**[10M | Dec-19]**

The number of input vectors is four and number of clusters to be formed is two. Thus, $n = 4$ and $m = 2$. The architecture of the Kohonen self-organizing feature map is given by Figure 5.11

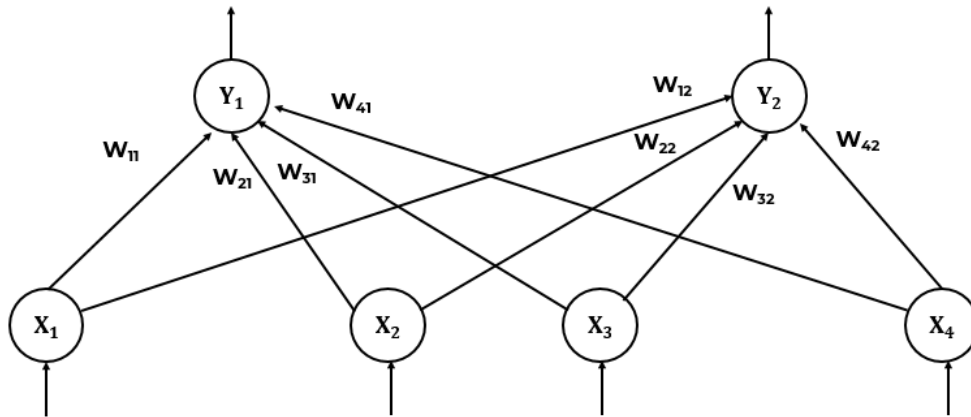


Figure 5.11: Architecture of the Kohonen self-organizing feature map

Step 0: Initialize the weights randomly between 0 and 1

$$w_1 = \begin{bmatrix} 0.2 & 0.9 \\ 0.4 & 0.7 \\ 0.6 & 0.5 \\ 0.8 & 0.3 \end{bmatrix} \quad R = 0; \alpha(0) = 0.5$$

First input vector:

Step 1: For $x = [0 \ 0 \ 1 \ 1]$ perform Steps 2-4.

Step 2: Calculate the Euclidean distance:

$$D(j) = \sum_i (w_{ij} - x_i)^2$$

$$D(1) = \sum_{i=1}^4 (w_{i1} - x_i)^2$$

$$\begin{aligned} &= (0.2 - 0)^2 + (0.4 - 0)^2 + (0.6 - 1)^2 + (0.8 - 1)^2 \\ &= 0.04 + 0.16 + 0.16 + 0.04 \\ &= 0.4 \end{aligned}$$

$$D(2) = \sum_{i=1}^4 (w_{i2} - x_i)^2$$

$$\begin{aligned} &= (0.9 - 0)^2 + (0.7 - 0)^2 + (0.5 - 1)^2 + (0.3 - 1)^2 \\ &= 0.81 + 0.49 + 0.25 + 0.49 \\ &= 2.04 \end{aligned}$$

Step 3: Since $D(1) < D(2)$, therefore $D(1)$ is minimum. Hence the winning cluster unit is Y_1 i.e., $j = 1$

Step 4: Update the weights on the winning cluster unit $j = 1$

$$w_{ij}(\text{new}) = w_{ij}(\text{old}) + \alpha[x_i - w_{ij}(\text{old})]$$

$$w_{i1}(\text{new}) = w_{i1}(\text{old}) + 0.5[x_i - w_{i1}(\text{old})]$$

$$\begin{aligned}
 w_{11}(n) &= w_{11}(0) + 0.5[x_1 - w_{11}(0)] \\
 &= 0.2 + 0.5[0 - 0.2] \\
 &= 0.1
 \end{aligned}$$

$$\begin{aligned}
 w_{21}(n) &= w_{21}(0) + 0.5[x_2 - w_{21}(0)] \\
 &= 0.4 + 0.5[0 - 0.4] \\
 &= 0.2
 \end{aligned}$$

$$\begin{aligned}
 w_{31}(n) &= w_{31}(0) + 0.5[x_3 - w_{31}(0)] \\
 &= 0.6 + 0.5[1 - 0.6] \\
 &= 0.8
 \end{aligned}$$

$$\begin{aligned}
 w_{41}(n) &= w_{41}(0) + 0.5[x_4 - w_{41}(0)] \\
 &= 0.8 + 0.5[1 - 0.8] \\
 &= 0.9
 \end{aligned}$$

The updated weight matrix after presentation of first input pattern is

$$w_1 = \begin{bmatrix} 0.1 & 0.9 \\ 0.2 & 0.7 \\ 0.8 & 0.5 \\ 0.9 & 0.3 \end{bmatrix}$$

Second input vector:

Step 1: For $x = [1 \ 0 \ 0 \ 0]$ perform Steps 2-4.

Step 2: Calculate the Euclidean distance:

$$D(j) = \sum_i (w_{ij} - x_i)^2$$

$$\begin{aligned}
 D(1) &= \sum_{i=1}^4 (w_{i1} - x_i)^2 \\
 &= (0.1 - 1)^2 + (0.2 - 0)^2 + (0.8 - 0)^2 + (0.9 - 0)^2 \\
 &= 0.81 + 0.04 + 0.64 + 0.81 \\
 &= 2.3
 \end{aligned}$$

$$\begin{aligned}
 D(2) &= \sum_{i=2}^4 (w_{i2} - x_i)^2 \\
 &= (0.9 - 1)^2 + (0.7 - 0)^2 + (0.5 - 0)^2 + (0.3 - 0)^2 \\
 &= 0.01 + 0.49 + 0.25 + 0.09 \\
 &= 0.84
 \end{aligned}$$

Step 3: Since $D(2) < D(1)$, therefore $D(2)$ is minimum. Hence the winning cluster unit is Y_2 i.e., $j = 2$

Step 4: Update the weights on the winning cluster unit $j = 2$

$$w_{ij}(\text{new}) = w_{ij}(\text{old}) + \alpha[x_i - w_{ij}(\text{old})]$$

$$w_{i2}(\text{new}) = w_{i2}(\text{old}) + 0.5[x_i - w_{i2}(\text{old})]$$

$$\begin{aligned}
 w_{12}(n) &= w_{12}(0) + 0.5[x_1 - w_{12}(0)] \\
 &= 0.9 + 0.5[1 - 0.9] \\
 &= 0.95
 \end{aligned}$$

$$\begin{aligned}
 w_{22}(n) &= w_{22}(0) + 0.5[x_2 - w_{22}(0)] \\
 &= 0.7 + 0.5[0 - 0.7] \\
 &= 0.35
 \end{aligned}$$

$$\begin{aligned}
 w_{32}(n) &= w_{32}(0) + 0.5[x_3 - w_{32}(0)] \\
 &= 0.5 + 0.5[0 - 0.5] \\
 &= 0.25
 \end{aligned}$$

$$\begin{aligned}
 w_{42}(n) &= w_{42}(0) + 0.5[x_4 - w_{42}(0)] \\
 &= 0.3 + 0.5[0 - 0.3] \\
 &= 0.15
 \end{aligned}$$

The updated weight matrix after presentation of first input pattern is

$$w_1 = \begin{bmatrix} 0.1 & 0.95 \\ 0.2 & 0.35 \\ 0.8 & 0.25 \\ 0.9 & 0.15 \end{bmatrix}$$

Third input vector:

Step 1: For $x = [0 \ 1 \ 1 \ 0]$ perform Steps 2-4.

Step 2: Calculate the Euclidean distance:

$$D(j) = \sum_i (w_{ij} - x_i)^2$$

$$\begin{aligned}
 D(1) &= \sum_{i=1}^4 (w_{i1} - x_i)^2 \\
 &= (0.1 - 0)^2 + (0.2 - 1)^2 + (0.8 - 1)^2 + (0.9 - 0)^2 \\
 &= 0.01 + 0.64 + 0.04 + 0.81 \\
 &= 1.5
 \end{aligned}$$

$$\begin{aligned}
 D(2) &= \sum_{i=2}^4 (w_{i2} - x_i)^2 \\
 &= (0.95 - 0)^2 + (0.35 - 1)^2 + (0.25 - 1)^2 + (0.15 - 0)^2 \\
 &= 0.9025 + 0.4225 + 0.5625 + 0.0225 \\
 &= 1.91
 \end{aligned}$$

Step 3: Since $D(1) < D(2)$, therefore $D(1)$ is minimum. Hence the winning cluster unit is Y_1 i.e., $j = 1$

Step 4: Update the weights on the winning cluster unit $j = 1$

$$w_{ij}(\text{new}) = w_{ij}(\text{old}) + \alpha[x_i - w_{ij}(\text{old})]$$

$$w_{i1}(\text{new}) = w_{i1}(\text{old}) + 0.5[x_i - w_{i1}(\text{old})]$$

$$\begin{aligned}
 w_{11}(n) &= w_{11}(0) + 0.5[x_1 - w_{11}(0)] \\
 &= 0.1 + 0.5[0 - 0.1] \\
 &= 0.05
 \end{aligned}$$

$$\begin{aligned}
 w_{21}(n) &= w_{21}(0) + 0.5[x_2 - w_{21}(0)] \\
 &= 0.2 + 0.5[1 - 0.2] \\
 &= 0.6
 \end{aligned}$$

$$\begin{aligned}
 w_{31}(n) &= w_{31}(0) + 0.5[x_3 - w_{31}(0)] \\
 &= 0.8 + 0.5[1 - 0.8] \\
 &= 0.9
 \end{aligned}$$

$$\begin{aligned}
 w_{41}(n) &= w_{41}(0) + 0.5[x_4 - w_{41}(0)] \\
 &= 0.9 + 0.5[0 - 0.9] \\
 &= 0.45
 \end{aligned}$$

The updated weight matrix after presentation of first input pattern is

$$w_1 = \begin{bmatrix} 0.05 & 0.95 \\ 0.6 & 0.35 \\ 0.9 & 0.25 \\ 0.45 & 0.15 \end{bmatrix}$$

Fourth input vector:

Step 1: For $x = [0 \ 0 \ 0 \ 1]$ perform Steps 2-4.

Step 2: Calculate the Euclidean distance:

$$D(j) = \sum_i (w_{ij} - x_i)^2$$

$$\begin{aligned}
 D(1) &= \sum_{i=1}^4 (w_{i1} - x_i)^2 \\
 &= (0.05 - 0)^2 + (0.6 - 0)^2 + (0.9 - 0)^2 + (0.45 - 1)^2 \\
 &= 0.0025 + 0.36 + 0.81 + 0.3025 \\
 &= 1.475
 \end{aligned}$$

$$\begin{aligned}
 D(2) &= \sum_{i=2}^4 (w_{i2} - x_i)^2 \\
 &= (0.95 - 0)^2 + (0.35 - 0)^2 + (0.25 - 0)^2 + (0.15 - 1)^2 \\
 &= 0.9025 + 0.1225 + 0.0625 + 0.7225 \\
 &= 1.81
 \end{aligned}$$

Step 3: Since $D(1) < D(2)$, therefore $D(1)$ is minimum. Hence the winning cluster unit is Y_1 i.e., $j = 1$

Step 4: Update the weights on the winning cluster unit $j = 1$

$$w_{ij}(\text{new}) = w_{ij}(\text{old}) + \alpha[x_i - w_{ij}(\text{old})]$$

$$w_{i1}(\text{new}) = w_{i1}(\text{old}) + 0.5[x_i - w_{i1}(\text{old})]$$

$$\begin{aligned}
 w_{11}(n) &= w_{11}(0) + 0.5[x_1 - w_{11}(0)] \\
 &= 0.05 + 0.5[0 - 0.05] \\
 &= 0.025
 \end{aligned}$$

$$\begin{aligned}
 w_{21}(n) &= w_{21}(0) + 0.5[x_2 - w_{21}(0)] \\
 &= 0.6 + 0.5[0 - 0.6] \\
 &= 0.3
 \end{aligned}$$

$$\begin{aligned}
 w_{31}(n) &= w_{31}(0) + 0.5[x_3 - w_{31}(0)] \\
 &= 0.9 + 0.5[0 - 0.9] \\
 &= 0.45
 \end{aligned}$$

$$\begin{aligned}
 w_{41}(n) &= w_{41}(0) + 0.5[x_4 - w_{41}(0)] \\
 &= 0.45 + 0.5[1 - 0.45] \\
 &= 0.475
 \end{aligned}$$

The updated weight matrix after presentation of first input pattern is

$$w_1 = \begin{bmatrix} 0.025 & 0.95 \\ 0.3 & 0.35 \\ 0.45 & 0.25 \\ 0.475 & 0.15 \end{bmatrix}$$

Since all the four given input patterns are presented, this is end of first iteration or 1-epoch. Now the learning rate can be updated as:

$$\alpha(t + 1) = 0.5 \alpha(t)$$

$$\alpha(1) = 0.5 \alpha(0)$$

$$= 0.5 \times 0.5$$

$$= 0.25$$

With this learning rate you can proceed further up to 100 iterations or till radius becomes zero or the weight matrix reduces to a very negligible value. The net with updated weights is shown by figure 5.11

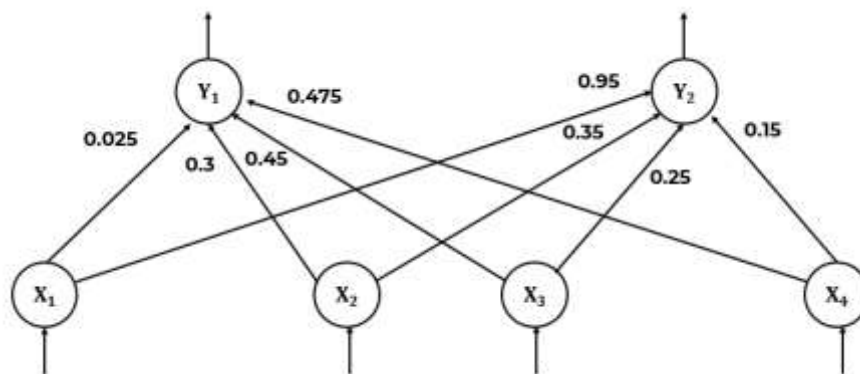
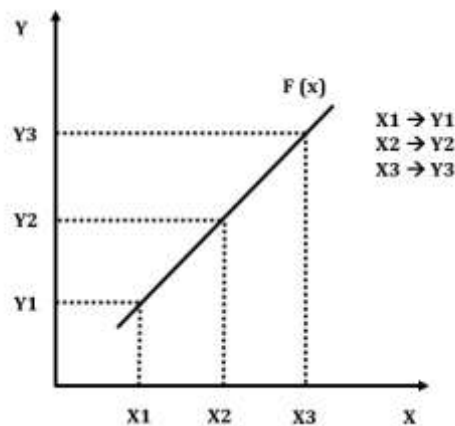


Figure 5.12

-- EXTRA QUESTIONS --**Q1. Explain Supervised & Unsupervised Learning?****ANS:****[P | Medium]****SUPERVISED LEARNING:**

1. It is also known as **Inductive Learning**.
2. Supervised learning deals with learning a function from available training data.
3. A supervised learning algorithm analyzes the training data and produces an inferred function, which can be used for mapping new examples.
4. Common examples of supervised learning include:
 - a. Classifying e-mails as spam.
 - b. Labeling webpages based on their content.
 - c. Voice recognition.
5. There are many supervised learning algorithms such as neural networks, Support Vector Machines (SVMs), and Naive Bayes classifiers.

Example: Perception Learning Rule.**Figure 5.13: Supervised Learning Example****UNSUPERVISED LEARNING:**

1. Unsupervised learning is the machine learning task of inferring a function to describe hidden structure from "unlabeled" data.
2. Since the examples given to the learner are unlabeled, there is no evaluation of the accuracy of the structure that is output by the relevant algorithm.
3. Unsupervised learning is an extremely powerful tool for analyzing available data and look for patterns and trends.
4. It is most commonly used for clustering similar input into logical groups.
5. Common approaches to unsupervised learning include:
 - a. K - Means.
 - b. Self-organizing maps.
 - c. Hierarchical clustering.

Example: Clustering.

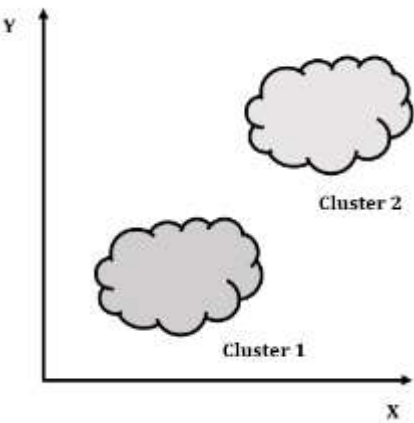


Figure 5.14: Unsupervised Learning Example.

Q2. Differentiate between Supervised & Unsupervised Learning

Ans:

[P | Medium]

Table 5.2: Supervised Learning v/s Unsupervised Learning

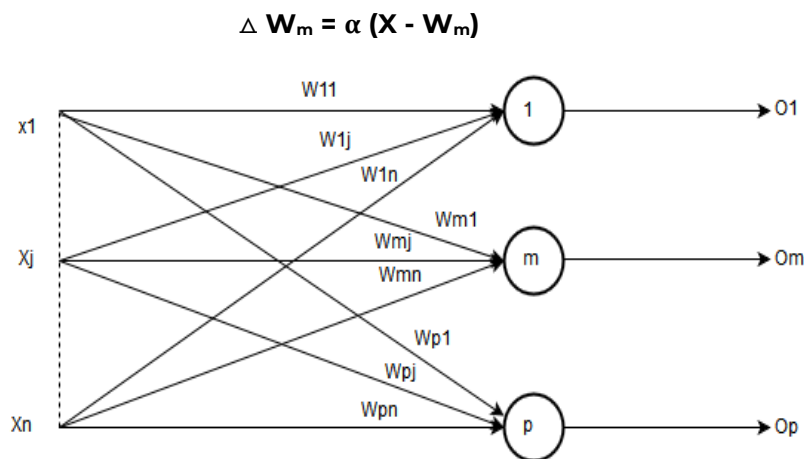
Supervised Learning	Unsupervised Learning
Supervised Learning can corresponds to learning in class room in the presence of teacher. Thus questions can be answered.	Unsupervised Learning corresponds to learning from a video tape lecture. Thus questions cannot be answered.
Supervised Learning usually provides the classification and recognition of the patterns.	Unsupervised Learning usually preforms the clustering of the training patterns.
In Supervised Learning, Desired output is known.	In Unsupervised Learning, desired output is unknown.
Error information can be used.	Error information cannot be used.
A training set is required.	A training set is not required.
A reward/punishment scheme is used.	It uses observation for learning.
Neural Network Model for Supervised Learning is Perception & Feed – Forward Neural Network.	Neural Network Model for Unsupervised Learning is Self-Organizing Maps.
In supervised Learning, one set of observations, called inputs, is assumed to be the cause of another set of observations, called outputs.	In Unsupervised Learning all observations are assumed to be caused by a set of latent variables.
Example: Perceptron Learning Rule.	Example: Hebbian Learning Rule.

Q3. Winner take all learning rule in neural network.**Ans:****[P | Medium]**

1. Winner-Take-All is a strategy used in **Competitive Learning**.
2. It is used for **Unsupervised Network Training**.
3. Winner Takes All Learning Rule is originated from **Outstar Learning Rule**.
4. This learning rule differs substantially from all other learning rules.
5. It is used for learning statistical properties of inputs.
6. The learning is based on the premise that one of the neurons in the layer, say N^{th} neuron has maximum response due to input X as shown in figure 5.15.
7. This neuron is declared as the Winner Neuron with a weight.

$$\mathbf{W}_m = [\mathbf{W}_1 \ \mathbf{W}_2 \ \mathbf{W}_3 \ \dots \mathbf{W}_{mn}]$$

8. The Increment is computed as follows:

**Figure 5.15: Winner Takes All.**

9. In figure 5.10, 'm' is winner neuron.
10. The winner selection is based on the following criterion of maximum activating among all 'p' neurons participating in competitive environment.

$$\mathbf{W}_m^t \mathbf{X} = \text{Max} (\mathbf{W}_i^t \mathbf{X}) \dots \dots \dots (\text{Where } i = 1, 2, \dots, p)$$

Q4. Learning Vector Quantization.**Ans:****[P | Medium]****LEARNING VECTOR QUANTIZATION:**

1. Learning Vector Quantization was developed by **Kohonen**.
2. It is one of the most frequently used **unsupervised clustering algorithms**.
3. It is a supervised version of **vector quantization**.
4. Learning Vector Quantization (LVQ), is a prototype-based supervised classification algorithm.
5. It is used for pattern classification.
6. In LVQ each output unit represents a particular class or category.
7. In LVQ, Classes are predefined and we have a set of labelled data.
8. The goal is to determine a set of prototypes that best represent each class.

9. An LVQ system is represented by prototype $W = \{w(i), \dots, w(n)\}$ which are defined in the feature space of observed data.
10. LVQ can be a source of great help in classifying text documents.

ARCHITECTURE:

1. Following figure 5.16 shows the architecture of LVQ which is quite similar to the architecture of KSOM.
2. As we can see, there are “n” number of input units and “m” number of output units.
3. The layers are fully interconnected with having weights on them.

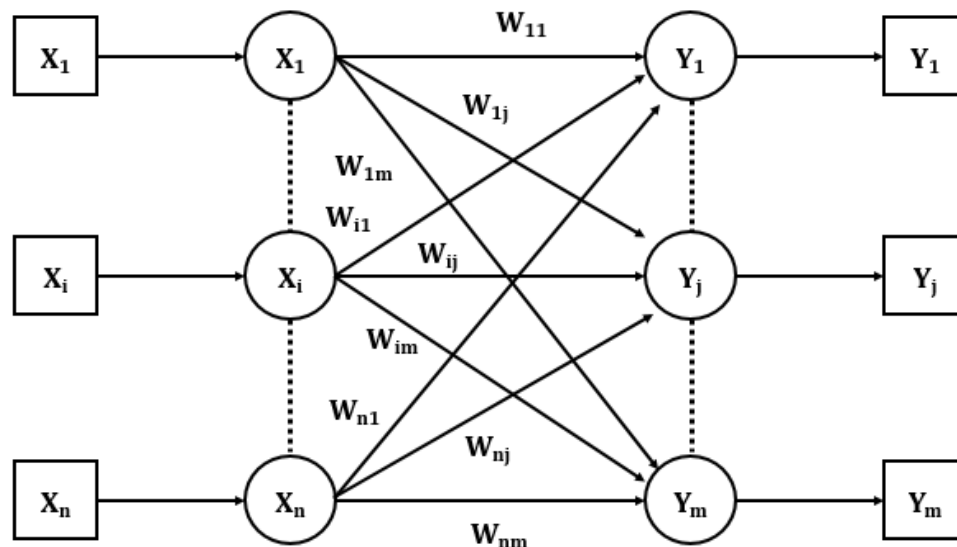


Figure 5.16: LOV Architecture.

Parameters Used:

Following are the parameters used in LVQ training process as well as in the flowchart

1. \mathbf{x} = training vector ($x_1, \dots, x_i, \dots, x_n$)
2. \mathbf{T} = class for training vector \mathbf{x}
3. \mathbf{w}_j = weight vector for j^{th} output unit
4. \mathbf{C}_j = class associated with the j^{th} output unit

Advantages:

1. LVQ creates prototypes that are easy to interpret for experts.
2. It is used in a variety of practical applications.

Disadvantage:

1. A key issue in LVQ is the choice of an appropriate measure of distance or similarity for training and classification.

Q5. Explain Linear separable and Non-linearly separable pattern with example.

Ans: **[P | Medium]**

1. If a subset of the input pattern belongs to one class (Say X_1) and the remaining subset of the input pattern to another class (Say X_2) then the objective in a pattern classification problem is to determine a set of weights W_1, W_2, \dots, W_m

2. Such that if the weighted sum

$$\sum_{i=1}^M w_i x_i > \theta, \text{ then } x = (x_1, x_2, \dots, x_M)^T$$

Belongs to Class X_1

3. And if

$$\sum_{i=1}^M w_i x_i < \theta, \text{ then } x = (x_1, x_2, \dots, x_M)^T$$

Belongs to Class X_2

4. The dividing surface between the two classes is given by

$$\sum_{i=1}^M w_i x_i = \theta$$

5. If $W^T X > 0$, Then X belongs to Class X_1 .
6. If $W^T X < 0$, Then X belongs to Class X_2 .
7. The equation for dividing linear hyper line is $W^T X = 0$
8. If such a weight vector exists such that $W^T X > 0$ for each $X \in X_1$ and $W^T X < 0$ for each $X \in X_2$ then the patterns sets X_1 and X_2 are **Linearly Separable**.
9. If no weight vector exists such that $W^T X > 0$ for each $X \in X_1$ and $W^T X < 0$ for each $X \in X_2$ then the patterns sets X_1 and X_2 are **Non-Linearly Separable**.
10. Figure 5.17 shows the example of Linearly separable pattern.

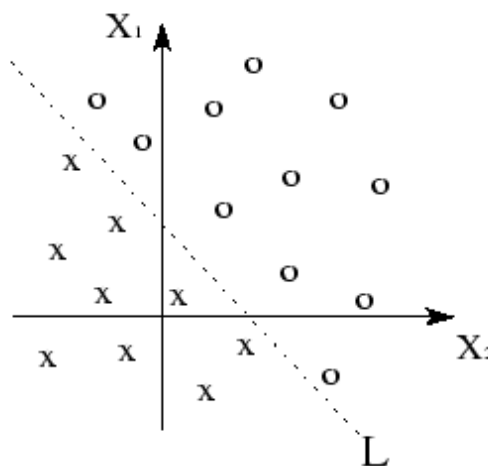


Figure 5.17: Linearly Separable pattern.

Example of Linearly Separable Pattern:

AND

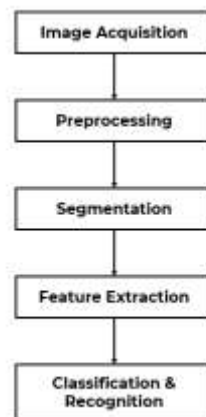
X	Y	Class
0	0	0
0	1	0
1	0	0
1	1	1

OR

X	Y	Class
0	0	0
0	1	1
1	0	1
1	1	1

Example of Non-Linearly Separable Pattern:**XOR**

X	Y	Class
0	0	0
0	1	1
1	0	1
1	1	0

Q6. Character recognition using neural network.**Ans:** [P | Medium]**CHARACTER RECOGNITION:****Figure 5.18: Character Recognition Technique.**

1. Character Recognition is an application of the **Error Back Propagation Algorithm**.
2. Character Recognition techniques are used for both printed & hand written characters.
3. Figure 5.18 shows the block diagram of character recognition.

I) Image Acquisition:

1. In this Step, the original image is obtained & scanned.

II) Pre-Processing:

1. The pre-processing is a series of operations performed on scanned input image.
2. The role of pre-processing is to segment the interesting pattern from the background.
3. Generally, noise filtering, smoothing and normalization should be done in this step.

III) Segmentation:

1. After scanning the document, the document image is subjected to pre-processing for background noise elimination, and Binarization to generate the bit map image of the text.
2. The pre-processed image is divided into lines, words and characters.
3. Segmentation Includes:
 - a. **Line segmentation:** To separate the text lines, line segmentation is used.
 - b. **Word segmentation:** Word segmentation is provide the space between words.
 - c. **Character Recognition:** It is providing the Spacing between the characters. So, it is called segmentation.

IV) Feature Extraction:

1. This approach is useful when image sizes are large.
2. A reduced feature representation is required to quickly complete tasks such as image matching and retrieval.

V) Classification & Recognition:

1. Classification is used to take segmented input samples and classify them into group.
2. Finally the segmented image is classified & recognition using various image recognition techniques.

Q7. Explain error back proportional algorithm.**Ans:****[P | Medium]****ERROR BACK PROPORTIONAL ALGORITHM:**

1. Error Back Proportional Algorithm is a **Supervised Learning Algorithm**.
2. It is common method of training **Artificial Neural Networks** used in conjunction with an optimization method such as **Gradient Descent**.
3. Back propagation requires a known, desired output for each input value in order to calculate the loss function gradient.
4. It is therefore usually considered to be a supervised learning method, although it is also used in some unsupervised networks such as **auto encoders**.
5. The algorithm allows experiential acquisition of input mapping knowledge within multi-layer networks.
6. Back propagation requires that the activation function used by the artificial neuron be differentiable.

STEPS:

Given are P training sets $\{(Z_1, D_1), (Z_2, D_2), \dots, (Z_p, D_p)\}$

Where $Z = i \times 1$, $D = K \times i$, $Y = j \times 1$ & $O = K \times 1$

i^{th} Component of $Z = -1$ & j^{th} Component of $Y = -1$

Step 1: $\eta > 0$ and some E_{\max} is choose W & V some small random values

$$W = K \times j$$

$$V = j \times i$$

$$q = 1, p = 1 \text{ \& } E = 0$$

Step 2: Training Cycle starts here

$$\text{Input: } Z = Z \times p \text{ \& } D = D \times p$$

$$\text{Output: } Y_j = F(V_j \times Z)$$

$$O_k = F(W_k \times Y)$$

Step 3: Compute error

$$E = \frac{1}{2} (d_k - O_k)^2 + E$$

Step 4: Compute Error Signal

$$\delta O_k = \frac{1}{2} (d_k - O_k) (1 - O_k^2)$$

$$\delta y_j = \frac{1}{2} (1 - y_j^2) \sum_{k=1}^K \delta O_k \cdot W_{kj}$$

Step 5: Output layer weights are adjusted

$$W_{kj} = W_{kj} + n\delta O_k Y_j$$

Step 6: Hidden layer weights are adjusted

$$V_{ji} = V_{ji} + \eta\delta y_i z_i$$

Step 7:

If $p < P$
 $P = P + 1$
 $q = q + 1$
 & go to step 2
 Else step 8

Step 8:

If $E < E_{\max}$ Terminate
 Else If $E > E_{\max}$
 $E = 0$
 $p = 1$
 & go to step 2

Q8. What is Self-Organizing Map? Draw & Explain Architecture of Kohonen Self Organization Feature Map.

Ans: [P | Medium]

SELF-ORGANIZING MAP:

1. Self-Organizing Map is also called as **Self-Organizing Feature Map (SOFM)**.
2. It is a type of **Artificial Neural Network**.
3. SOFM is **Unsupervised Learning Technique**.
4. SOFM's are **neural networks** that employ unsupervised learning methods.
5. It maps their weights to conform to the given input data with a goal of representing multidimensional data.
6. The output data is easier and understandable form for the human eye in SOFM's.

KOHONEN SELF ORGANIZATION FEATURE MAP:

1. Kohonen's SOFM is called a **Topology-Preserving Map**.
2. Kohonen's SOFM's are a type of **Unsupervised Learning**.
3. The goal is to discover some underlying structure of the data.
4. KSOFM consists of a two dimensional array of output units connected to all input nodes.
5. It works based on the property of Topology Preservation.
6. Nearby input patterns should activate nearby output units on the map.
7. KSOFM can be recognized as a **special competitive learning**.
8. Only the weight vectors of winner and its neighbor units are updated

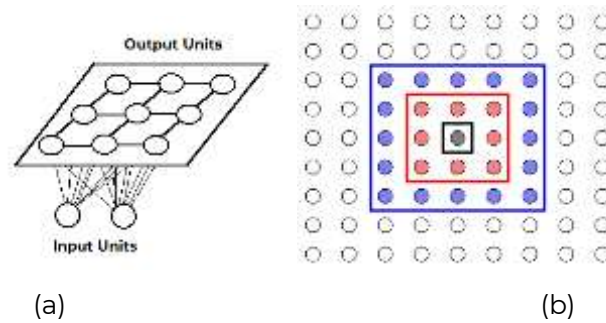


Figure 5.19: (a) KSOFM (b) Neighborhoods (R) for a rectangular matrix of cluster units.

KSOFM Working:

The Self-Organizing Map algorithm can be broken up into 6 steps:

1. Each node's weights are initialized.
2. A vector is chosen at random from the set of training data and presented to the network.
3. Every node in the network is examined to calculate which ones' weights are most like the input vector. The winning node is commonly known as the Best Matching Unit (BMU).
4. The radius of the neighborhood of the BMU is calculated. This value starts large. Typically it is set to be the radius of the network, diminishing each time- step.
5. Any nodes found within the radius of the BMU, calculated in 4), are adjusted to make them more like the input vector (Equation 3a, 3b). The closer a node is to the BMU, the more its' weights are altered.
6. Repeat 2) for N iterations.

CHAP - 6: EXPERT SYSTEM

Q1. Draw and describe the architecture of expert system.

Ans:

[P | High]

EXPERT SYSTEM:

1. Expert System are **Artificial Intelligence (AI) tools**.
2. It capture the expertise of knowledge workers (Experts) and provide advice to usually non-experts in a given domain.
3. In artificial intelligence, an expert system is a computer system that emulates the decision-making ability of a human expert.
4. Expert systems are designed to solve complex problems by reasoning about knowledge, represented primarily as if-then rules rather than through conventional procedural code.
5. Expert System are implemented with artificial Intelligence technology, often called **Expert System Shells**.

ARCHITECTURE OF EXPERT SYSTEM:

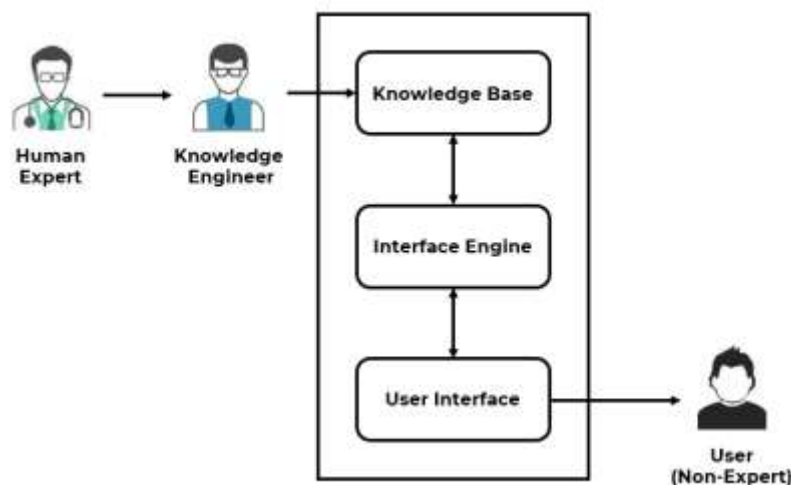


Figure 6.1: Expert System Architecture.

CHARACTERISTICS OF EXPERT SYSTEMS

1. High performance
2. Understandable
3. Reliable
4. Highly responsive

COMPONENTS:

I) Knowledge Base:

1. Knowledge Base is **database of rules**.
2. It contains domain-specific and high-quality knowledge.
3. Knowledge is required to exhibit intelligence.
4. The success of any Expert System majorly depends upon the collection of highly accurate and precise knowledge.

II) Interface Engine:

1. Interface Engine acquires and manipulates the knowledge from the knowledge base to arrive at a particular solution.
2. Interface Engine use **Forward & Backward Chaining Strategies**.

III) User Interface:

1. User interface provides interaction between user of the Expert System and the Expert System itself.
2. It is generally Natural Language Processing, so as to be used by the user who is well-versed in the task domain.
3. The user of the Expert System need not be necessarily an expert in Artificial Intelligence.
4. It explains how the Expert System has arrived at a particular recommendation.

ADVANTAGES:

1. **Availability:** They are easily available due to mass production of software.
2. **Less Production Cost:** Production cost is reasonable. This makes them affordable.
3. **Speed:** They offer great speed. They reduce the amount of work an individual puts in.
4. **Less Error Rate:** Error rate is low as compared to human errors.
5. **Reducing Risk:** They can work in the environment dangerous to humans.
6. **Steady response:** They work steadily without getting motional, tensed or fatigued.

DISADVANTAGES:

No technology can offer easy and complete solution. Large systems are costly, require significant development time, and computer resources. ESs have their limitations which include –

1. Limitations of the technology.
 2. Difficult knowledge acquisition
 3. ES are difficult to maintain.
 4. High development costs.
-

Q2. Explain Neuro Fuzzy Hybrid system**Ans:****[P | Medium]****NEURO FUZZY HYBRID SYSTEM:**

1. Integration of Neural Network and Fuzzy Logic is called as Neuro Fuzzy System.
2. It is based on fuzzy system which is trained on the basis of working of neural network theory.
3. The learning process operates only on the local information and causes only local changes in the underlying fuzzy system.
4. A Neuro-fuzzy system can be seen as a **3-layer feed forward neural network**.
5. The first layer represents input variables, the middle (hidden) layer represents fuzzy rules and the third layer represents output variables.
6. Fuzzy sets are encoded as connection weights within the layers of the network, which provides functionality in processing and training the model.

7. Figure 6.2 represents Neuro Fuzzy System

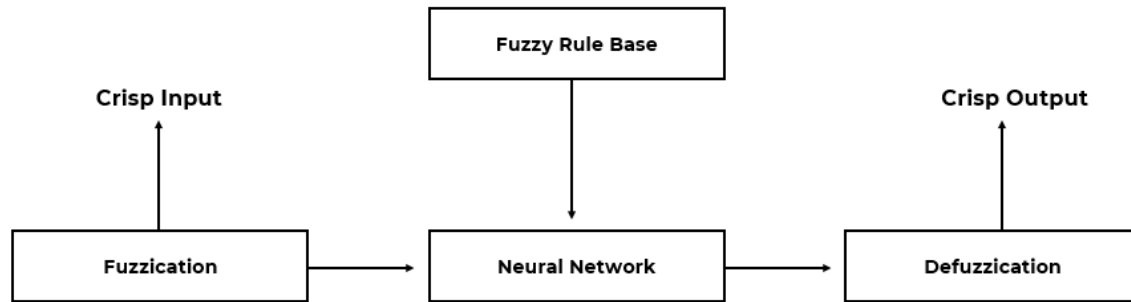


Figure 6.2: Neuro Fuzzy System

WORKING:

1. In input layer, each neuron transmits external crisp signals directly to the next layer.
2. Each Fuzzification neuron receives a crisp input and determines the degree to which the input belongs to input fuzzy set.
3. Fuzzy rule layer receives neurons that represent fuzzy sets.
4. An output neuron, combines all inputs using fuzzy operation UNION.
5. Each defuzzification neuron represents single output of Neuro-Fuzzy System.

ADVANTAGES:

1. It can resolve conflicts by collaboration and aggregation.
2. It can handle numeric, linguistic and logic kind of information.
3. It has self-learning, self-organizing and self-tuning capabilities.
4. It can manage imprecise, partial, vague or imperfect information.
5. It can mimic human decision-making process.

DISADVANTAGES:

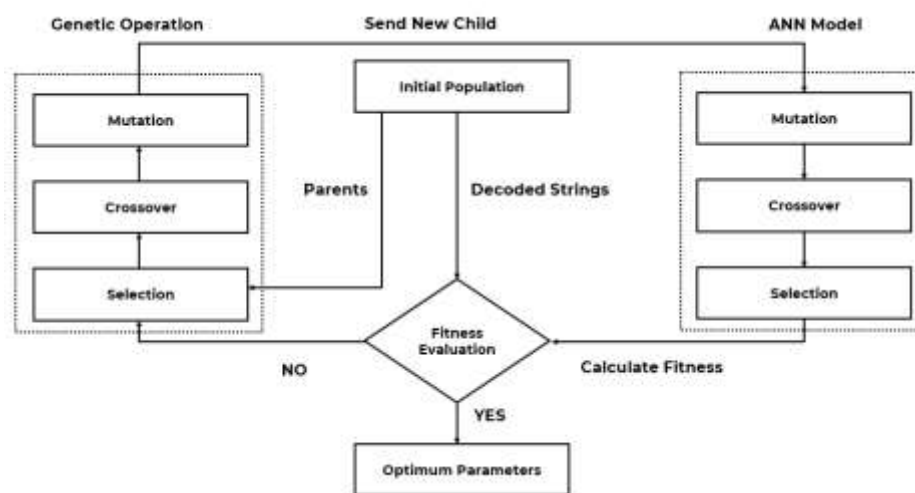
1. Hard to develop a model from a fuzzy system.
2. Neural networks cannot be used if training data is not available.
3. Problems of finding suitable membership values for fuzzy systems

APPLICATIONS:

1. Student Modelling.
2. Medical systems.
3. Traffic control systems.
4. Forecasting and predictions.

Q3. Explain Neuro Genetic Hybrid system**Ans:****[P | Medium]****NEURO GENETIC HYBRID SYSTEM:**

1. A Neuro Genetic hybrid system is a system that combines **Neural Networks and Generic Algorithm**.
2. **Neural Network** is capable to learn various tasks from examples, classify objects and establish relation between them.
3. **Genetic algorithm** serves as important search and optimization techniques.
4. Genetic algorithms can be used to improve the performance of Neural Networks and they can be used to decide the connection weights of the inputs.
5. These algorithms can also be used for topology selection and training network.
6. Figure 6.3 represents Neuro Genetic System.

**Figure 6.3: Neuro Genetic System****WORKING:**

1. GA repeatedly modifies a population of individual solutions.
2. GA uses three main types of rules at each step to create the next generation from the current population:
 - a. **Selection** to select the individuals, called parents, that contribute to the population at the next generation
 - b. **Crossover** to combine two parents to form children for the next generation
 - c. **Mutation** to apply random changes to individual parents in order to form children
3. GA then sends the new child generation to ANN model as new input parameter.
4. Finally, calculating of the fitness by developed ANN model is performed.

ADVANTAGES:

1. GA is used for topology optimization i.e. to select number of hidden layers, number of hidden nodes and interconnection pattern for ANN.
2. In GAs, the learning of ANN is formulated as a weight optimization problem, usually using the inverse mean squared error as a fitness measure.

3. Control parameters such as learning rate, momentum rate, tolerance level, etc. are also optimized using GA.
4. It can mimic human decision-making process.

DISADVANTAGES:

1. Highly complex system.
2. Maintenance costs are very high.
3. Accuracy of the system is dependent on the initial population.

APPLICATIONS:

1. Face recognition.
2. DNA matching.
3. Animal and human research.
4. Behavioral system.

Mumbai University – AISC – Dec-19

Q1. Attempt any four: [20]

(a) State PEAS Description for online English tutor.

Ans: [Chapter No. 1 | Page No. 01]

(b) Differentiate between Soft and Hard computing.

Ans: [Chapter No. 1 | Page No. 01]

(c) Give Local and Global heuristic function for block world problem.

Ans: [Chapter No. 2 | Page No. 20]

(d) Give different membership functions of fuzzy logic.

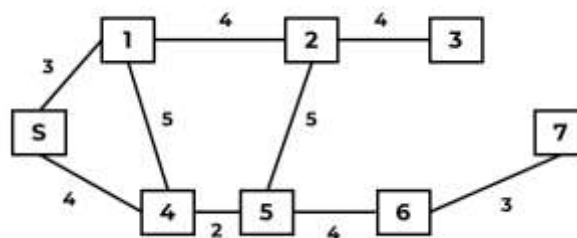
Ans: [Chapter No. 4 | Page No. 68]

(e) Determine (alfa) α -level sets and strong α -level sets for the following fuzzy sets.

$A = \{ (1, 0.2), (2, 0.5), (3, 0.8), (4, 1), (5, 0.7), (6, 0.3) \}$

Ans: [Chapter No. 4 | Page No. 78]

Q2. (a) Consider the graph given in Figure 1 below. Assume that the initial state is S and the goal state is 7. Find a path from the initial state to the goal state using A* Search. Also report the solution cost. The straight line distance heuristic estimates for the nodes are as follows: $h(1)=14$, $h(2)=10$, $h(3)=8$, $h(4)=12$, $h(5)=10$, $h(6)=10$, $h(S)=15$. [10]



Ans: [Chapter No. 2 | Page No. 18]

(b) The law says that it is a crime for an American to sell weapons to hostile nations. The country Nono, an enemy of America, has some missiles, and all of its missiles were sold to it by Colonel West, who is American. Prove that Col. West is a criminal using resolution technique. [10]

Ans: [Chapter No. 3 | Page No. 48]

Q3. (a) Implement AND function using perceptron networks for bipolar inputs and targets. [10]

Ans: [Chapter No. 5 | Page No. 92]

(b) Explain fuzzy controller system for a tipping example. Consider service and food quality rated between 0 and 10. Use this to leave a tip of 25%. [10]

Ans: [Chapter No. 4 | Page No. 77]

Q4. (a) Design a Mc-Culloch Pitts model for XOR Gate. [10]

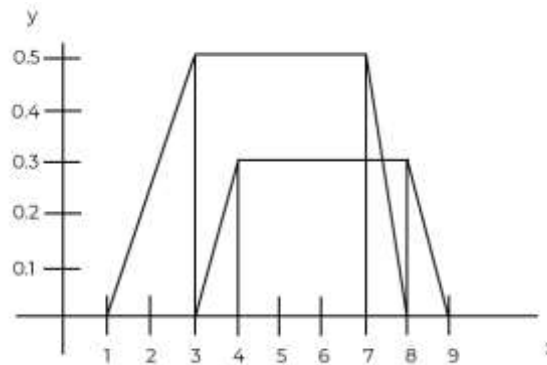
Ans: [Chapter No. 5 | Page No. 88]

(b) Construct kohonen Self-organizing map to cluster the four given vectors

$[0\ 0\ 1\ 1]$, $[1\ 0\ 0\ 0]$, $[0\ 1\ 1\ 0]$ and $[0\ 0\ 0\ 1]$. The number of cluster formed is two. Assume an initial learning rate of 0.5. [10]

Ans: [Chapter No. 5 | Page No. 99]

Q5. (a) Explain defuzzification techniques. Apply defuzzification by using Center of Gravity (CoG) method on the following: [10]



Ans: [Chapter No. 4 | Page No. 72]

(b) Explain planning problem in AI. What are different types of planning? Consider problem of changing a flat tire. The goal is to have a good spare tire properly mounted on to the car's axle, where the initial state has a flat tire on the axle and a good spare tire in the trunk. Give the ADL description for the problem. [10]

Ans: [Chapter No. 3 | Page No. 50]

Q6. Write Short notes on following (Any Four) [20]

(a) Genetic algorithm

Ans: [Chapter No. 2 | Page No. 16]

(b) ANFIS

Ans: [Chapter No. 4 | Page No. 70]

(c) Hill Climbing algorithm

Ans: [Chapter No. 2 | Page No. 15]

(d) Wumpus world knowledge base

Ans: [Chapter No. 3 | Page No. 47]

(e) Different types of Neural Networks

Ans: [Chapter No. 5 | Page No. 85]

Join **BackkBenchers Community** & become the **Student Ambassador** to represent your college & earn 15% Discount.



Be the **Technical Content Writer** with BackkBenchers and earn upto 100 Rs. per 10 Marks Questions.



Buy & Sell **Final Year Projects** with BackkBenchers. Project Charge upto 10,000.



Follow us on **Social Media Profiles** to get notified



BackkBenchersCommunity



+91-9930038388



BackkBenchersCommunity

E-Solutions are Available @BackkBenchers Website.