INSTITUTE OF INFORMATION TECHNOLOGY

JAHANGIRNAGAR UNIVERSITY

Number of Assignment : 02

Submission Date :

Course Tittle : Artificial Intelligence

Course Code : ICT - 4101

Submitted To

Dr. Mohammad Shahidul Islam

Professor

IIT – JU

Submitted By

Md. Shakil Hossain

Roll – 2023

4th year 1st Semester

IIT – JU

1. Define artificial intelligence. Explain all four approaches.

Ans: Artificial Intelligence (AI) is the study of how to make computers do things which, at the moment, people do better. In other words, Artificial Intelligence is the branch of computer science in which intelligence is incorporated in an artificial device. Purpose of AI is to create a Human clone.

2. What can AI do today? Briefly explain all.

Ans: Autonomous planning and scheduling: A hundred million miles from Earth, NASA's Remote Agent program became the first on-board autonomous planning program to control the scheduling of operations for a spacecraft. Remote Agent generated plans from high-level goals specified from the ground, and it monitored the operation of the spacecraft as the plans were executed-detecting, diagnosing, and recovering from problems as they occurred.

Game playing: IBM's Deep Blue became the first computer program to defeat the world champion in a chess match when it bested Garry Kasparov by a score of 3.5 to 2.5 in an exhibition match.

Autonomous control: The ALVINN computer vision system was trained to steer a car to keep it following a lane. It was placed in CMU's NAVLAB computer-controlled minivan and used to navigate across the United States-for 2850 miles it was in control of steering the vehicle 98% of the time.

Diagnosis: Medical diagnosis programs based on probabilistic analysis have been able to perform at the level of an expert physician in several areas of medicine.

Logistics Planning: During the Persian Gulf crisis of 1991, U.S. forces deployed a Dynamic Analysis and Replanning Tool, DART, to do automated logistics planning and scheduling for transportation. This involved up to 50,000 vehicles, cargo, and people at a time, and had to account for starting points, destinations, routes, and conflict resolution among all parameters.

Robotics: Many surgeons now use robot assistants in microsurgery. HipNav is a system that uses computer vision techniques to create a three-dimensional model of a patient's internal anatomy and then uses robotic control to guide the insertion of a hip replacement prosthesis.

Language understanding and problem solving: PROVERB is a computer program that solves crossword puzzles better than most humans, using constraints on possible word fillers, a large database of past puzzles, and a variety of information sources including dictionaries and online databases such as a list of movies and the actors that appear in them.

3. Define the terms: agent, agent function.

Ans:

• **Agent:** An agent is something that acts. An agent acts on behalf of a person. It is an entity that acts in response to the environmental issues.

• **Agent function:** Agent function is the action performed by an agent in response to the environmental issues.

4. How agents interact with environments through sensors and actuators? Use a vacuum-cleaner world with just two locations to explain those interactions.

Ans:

An **agent** is anything that can be viewed as perceiving its **environment** through **sensors** and SENSOR acting upon that environment through **actuators.** This simple idea is illustrated in Figure 1.2.

- A human agent has eyes, ears, and other organs for sensors and hands, legs, mouth, and other body parts for actuators.
- A robotic agent might have cameras and infrared range finders for sensors and various motors for actuators.
- A software agent receives keystrokes, file contents, and network packets as sensory inputs and acts on the environment by displaying on the screen, writing files, and sending network packets.

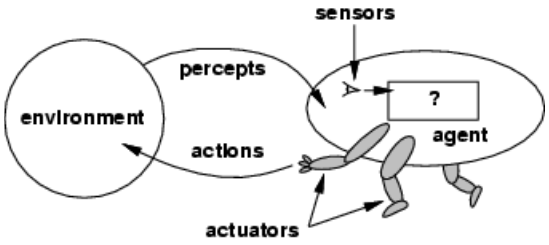


Figure 1.2 Agents interact with environments through sensors and actuators.

Percept

We use the term **percept** to refer to the agent's perceptual inputs at any given instant.

Percept Sequence

An agent's **percept sequence** is the complete history of everything the agent has ever perceived.

Agent function

Mathematically speaking, we say that an agent's behavior is described by the **agent function** that maps any given **percept sequence** to an action.

$f : P^* \rightarrow A$

Agent program

Internally, the agent function for an artificial agent will be implemented by an **agent program.** It is important to keep these two ideas distinct. The agent function is an abstract mathematical description; the agent program is a concrete implementation, running on the agent architecture.

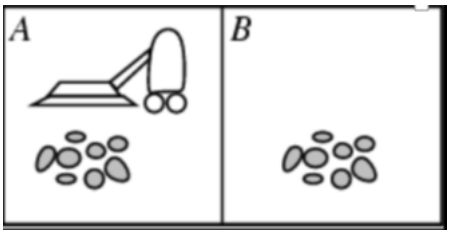


Fig: A vacuum-cleaner world with just two locations:

5. Write the PEAS description of the task environment for an

a. Automated taxi,
b. Medical diagnosis system,
c. Part-picking robot, and
d. Refinery controller

Ans:

Agent Type	**Performance Measure**	**Environments**	**Actuators**	**Sensors**
Taxi driver	Safe: fast, legal, comfortable trip, maximize profits	Roads, other traffic, pedestrians, customers	Steering, accelerator, brake, Signal, horn, display	Cameras, sonar, Speedometer, GPS, Odometer, engine sensors, keyboards, accelerometer

Agent Type	Performance Measure	Environment	Actuators	Sensors
Medical diagnosis system	Healthy patient, minimize costs, lawsuits	Patient, hospital, staff	Display questions, tests, diagnoses, treatments, referrals	Keyboard entry of symptoms, findings, patient's answers
Satellite image analysis system	Correct image categorization	Downlink from orbiting satellite	Display categorization of scene	Color pixel arrays
Part-picking robot	Percentage of parts in correct bins	Conveyor belt with parts; bins	Jointed arm and hand	Camera, joint angle sensors
Refinery controller	Maximize purity, yield, safety	Refinery, operators	Valves, pumps, heaters, displays	Temperature, pressure, chemical sensors
Interactive English tutor	Maximize student's score on test	Set of students, testing agency	Display exercises, suggestions, corrections	Keyboard entry

6. Briefly explain all the properties of task environments.

Ans:

Properties of task environments

- Fully observable vs. partially observable
- Deterministic vs. stochastic
- Episodic vs. sequential
- Static vs. dynamic
- Discrete vs. continuous
- Single agent vs. multiagent

Fully observable vs. **partially observable**

If an agent's sensors give it access to the complete state of the environment at each point in time, then we say that the task environment is fully observable. A task environment is effectively fully observable if the sensors detect all aspects that are *relevant* to the choice of action. An environment might be **partially observable** because of noisy and inaccurate sensors or because parts of the state are simply missing from the sensor data.

Deterministic vs. **stochastic**

If the next state of the environment is completely determined by the current state and the action executed by the agent, then we say the environment is deterministic; other- wise, it is stochastic.

Episodic vs. **sequential**

In an **episodic task environment**, the agent's experience is divided into atomic episodes. Each episode consists of the agent perceiving and then performing a single action. **Crucially, the next episode does not depend on the actions taken in previous episodes.** For example, an agent that has to spot defective parts on an assembly line bases each decision on the current part, regardless of previous decisions; In **sequential environments**, on the other hand, the current decision could affect all future decisions. Chess and taxi driving are sequential.

Discrete vs. **continuous.**

The discrete/continuous distinction can be applied to the *state* of the environment, to the way *time* is handled, and to the *percepts* and *actions* of the agent. For example, a discrete-state environment such as a chess game has a finite number of distinct states. Chess also has a discrete set of percepts and actions. Taxi driving is a continuous state and continuous time problem: the speed and location of the taxi and of the other vehicles sweep through a range of continuous values and do so smoothly over time. Taxi-driving actions are also continuous.

Single agent vs. **multiagent.**

An agent solving a crossword puzzle by itself is clearly in a single-agent environment, whereas an agent playing chess is in a two-agent environment.

7. List the properties of the following familiar environments.

a. Taxi driving
b. Medical diagnosis system,
c. Part-picking robot, and
d. Refinery controller

Ans:

Task Environment	Observable	Deterministic	Episodic	Static	Discrete	Agents
Crossword puzzle	Fully	Deterministic	Sequential	Static	Discrete	Single
Chess with a clock	Fully	Strategic	Sequential	Semi	Discrete	Multi
Poker	Partially	Stochastic	Sequential	Static	Discrete	Multi
Backgammon	Fully	Stochastic	Sequential	Static	Discrete	Multi
Taxi driving	Partially	Stochastic	Sequential	Dynamic	Continuous	Multi
Medical diagnosis	Partially	Stochastic	Sequential	Dynamic	Continuous	Single
Image-analysis	Fully	Deterministic	Episodic	Semi	Continuous	Single
Part-picking robot	Partially	Stochastic	Episodic	Dynamic	Continuous	Single
Refinery controller	Partially	Stochastic	Sequential	Dynamic	Continuous	Single
Interactive English tutor	Partially	Stochastic	Sequential	Dynamic	Discrete	Multi

8. What is the difference between agent program and agent function?

Ans:

A. An agent function is an abstract mathematical mapping from percept sequences to actions; an agent program is a specific implementation in a system.
B. An agent function processes information from the agent's sensors; an agent program controls the agent's actuators.
C. An agent function applies to a finite set of inputs; an agent program applies to an infinite set of inputs

9. What are the drawbacks of table driven agent?

Ans:

- **Table lookup** of percept-action pairs defining all possible condition-action rules necessary to interact in an environment
- **Too big** to generate and to store (Chess has about 10^{120} states, for example)
- **No knowledge** of non-perceptual(cannot perceive) parts of the current state
- **Not adaptive** to changes in the environment; requires entire table to be updated if changes occur
- **Looping**: Can't make actions conditional

- Take a long time to build the table
- No autonomy
- Even with learning, need a long time to learn the table entries

10. How many types of agents are there? Briefly explain them.

Ans:

- **Table-driven agents** – use a percept sequence/action table in memory to find the next action. They are implemented by a (large) **lookup table**.
- **Simple reflex agents** – are based on **condition-action rules**, implemented with an appropriate production system. They are **stateless devices which do not have memory** of past world states.
- **Agents with memory** – have **internal state**, which is used to keep track of past states of the world.
- **Agents with goals** – are agents that, in addition to state information, have **goal information** that describes desirable situations. Agents of this kind take future events into consideration.
- **Utility-based agents** – base their decisions on **classic axiomatic utility theory** in order to act rationally.
- **Simple Reflex Agent -** The simplest kind of agent is the **simple reflex agent.** These agents select actions on the basis of the current percept, ignoring the rest of the percept history.
- **Model-based reflex agents -** The most effective way to handle partial observability is for the agent to keep track of the part of the world it can't see now. That is, the agent should maintain 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. Updating this internal state information as time goes by requires two kinds of knowledge to be encoded in the agent program.

11. Write short notes on

a. Table-driven agents
b. Simple reflex agents
c. Model-based reflex agents
d. Goal-based agents
e. Utility-based agents

Ans:

a. **Table-driven agents** – use a percept sequence/action table in memory to find the next action. They are implemented by a **lookup table**.
b. **Simple reflex agents** – are based on **condition-action rules**, implemented with an appropriate production system. They are **stateless devices which do not have memory** of past world states.
c. **Model-based reflex agents -** The most effective way to handle partial observability is for the agent to keep track of the part of the world it can't see now. That is, the agent should maintain 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.
d. **Goals-based agent** – are agents that, in addition to state information, have **goal information** that describes desirable situations. Agents of this kind take future events into consideration.
e. **Utility-based agents** – base their decisions on **classic axiomatic utility theory** in order to act rationally.

12. Describe a well-defined problem.

Ans: A well-defined problem can be described by:

1. **Initial state**
2. **Operator or successor function** - for any state x returns s(x), the set of states reachable
3. from x with one action
4. **State space** - all states reachable from initial by any sequence of actions
5. **Path** - sequence through state space
6. **Path cost** - function that assigns a cost to a path. Cost of a path is the sum of costs of individual actions along the path
7. **Goal test** - test to determine if at goal state

13. What is meant by problem solving agent? Briefly explain goal formulation and problem formulation.

Ans: A Problem-solving agent is a **goal-based** agent. It decides what to do by finding sequence of actions that lead to desirable states. The agent can adopt a goal and aim at satisfying it. To illustrate the agent's behavior, let us take an example where our agent is in the city of Arad, which is in Romania. The agent has to adopt a **goal** of getting to Bucharest.

Goal formulation, based on the current situation and the agent's performance measure, is the first step in problem solving. The agent's task is to find out which sequence of actions will get to a goal state.

Problem formulation is the process of deciding what actions and states to consider given a goal.

14. What an agent-design assumes its environment is, if it does not have any idea of it?

Ans: The agent design assumes the Environment is

- **Static** : The entire process carried out without paying attention to changes that might be occurring in the environment.
- **Observable :** The initial state is known and the agent's sensor detects all aspects that are relevant to the choice of action
- **Discrete :** With respect to the state of the environment and percepts and actions so that alternate courses of action can be taken
- **Deterministic :** The next state of the environment is completely determined by the current state and the actions executed by the agent. Solutions to the problem are single sequence of actions

15. Explain in short all the components of a 'problem'.

Ans: A **problem** can be formally defined by **four components**:

I. The **initial state** that the agent starts in. The initial state for our agent of example problem is described by *In(Arad)*

II. A **Successor Function** returns the possible **actions** available to the agent. Given a state x,SUCCESSOR-FN(x) returns a set of {action,successor} ordered pairs where each action is one of the legal actions in state x,and each successor is a state that can be reached from x by applying the action.

For example,from the state In(Arad),the successor function for the Romania problem would return

{ [Go(Sibiu),In(Sibiu)],[Go(Timisoara),In(Timisoara)],[Go(Zerind),In(Zerind)] }

1. **State Space** : The set of all states reachable from the initial state. The state space forms a graph in which the nodes are states and the arcs between nodes are actions.
2. A **path** in the state space is a sequence of states connected by a sequence of actions.
3. iii. The **goal test** determines whether the given state is a goal state.
4. iv. A **path cost** function assigns numeric cost to each action. For the Romania problem the cost of path might be its length in kilometers.
5. The **step cost** of taking action a to go from state x to state y is denoted by c(x,a,y). The step cost for Romania are shown in figure 1.18. It is assumed that the step costs are non negative.
6. A **solution** to the problem is a path from the initial state to a goal state.
7. An **optimal solution** has the lowest path cost among all solutions.

16. Write short notes on

a. 8-queens problem
b. Airline travel problem
c. The 8-puzzle

Ans:

a. 8-queens problem

The goal of 8-queens problem is to place 8 queens on the chessboard such that no queen attacks any other.(A queen attacks any piece in the same row,column or diagonal). Figure 2.5 shows an attempted solution that fails: the queen in the right most column is attacked by the queen at the top left.

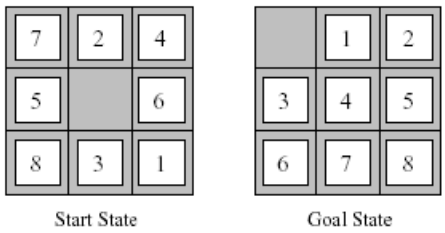
An **Incremental formulation** involves operators that augments the state description,starting with an empty state.for 8-queens problem,this means each action adds a queen to the state. A **complete-state formulation** starts with all 8 queens on the board and move them around. In either case the path cost is of no interest because only the final state counts.

b. The **airline travel problem** is specifies as follows **:**

- **States :** Each is represented by a location(e.g.,an airport) and the current time.
- **Initial state :** This is specified by the problem.
- **Successor function :** This returns the states resulting from taking any scheduled
- flight(further specified by seat class and location),leaving later than the current time plus
- the within-airport transit time,from the current airport to another.
- **Goal Test :** Are we at the destination by some prespecified time?
- **Path cost :** This depends upon the monetary cost,waiting time,flight time,customs and immigration procedures,seat quality,time of dat,type of air plane,frequent-flyer mileage awards, and so on.

c.

An 8-puzzle consists of a 3x3 board with eight numbered tiles and a blank space. A tile adjacent to the blank space can slide into the space. The object is to reach the goal state, as shown in figure.



17. What is meant by node and fringe in a search tree? What are the states of a node?

Ans: A **node** is a data structure with five components:

- STATE : a state in the state space to which the node corresponds;
- PARENT-NODE : the node in the search tree that generated this node;
- ACTION : the action that was applied to the parent to generate the node;
- PATH-COST :the cost,denoted by g(n),of the path from initial state to the node,as
- indicated by the parent pointers; and
- DEPTH : the number of steps along the path from the initial state.

Fringe

Fringe is a collection of nodes that have been generated but not yet been expanded. Each element of the fringe is a leaf node that is a node with no successors in the tree. The fringe of each tree consists of those nodes with bold outlines. The collection of these nodes is implemented as a queue.

18. State the general tree search algorithm.

Ans:

```
function TREE-SEARCH(problem, strategy) returns a solution, or failure
    initialize the search tree using the initial state of problem
    loop do
        if there are no candidates for expansion then return failure
        choose a leaf node for expansion according to strategy
        if the node contains a goal state then return the corresponding solution
        else expand the node and add the resulting nodes to the search tree
```

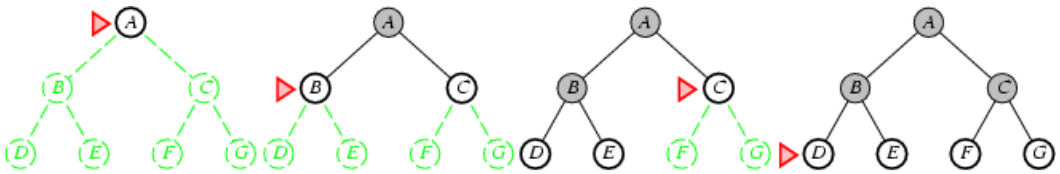
19. Write five uninformed search strategies and explain any two with examples.

Ans: There are five uninformed search strategies as given below.

- Breadth-first search
- Uniform-cost search
- Depth-first search
- Depth-limited search
- Iterative deepening search

Breadth-first search

Breadth-first search is a simple strategy in which the root node is expanded first,then all successors of the root node are expanded next,then their successors,and so on. In general,all the nodes are expanded at a given depth in the search tree before any nodes at the next level are expanded.



Iterative deepening depth-first search

Iterative deepening search is a general strategy often used in combination with depth-first-search,that finds the better depth limit. It does this by gradually increasing the limit – first 0,then 1,then 2, and so on – until a goal is found. This will occur when the depth limit reaches d,the depth of the shallowest goal node.

20. Find the time complexity for breadth-first-search. Assume every state has p successors.

Ans: Time complexity for BFS

Assume every state has p successors. The root of the search tree generates p nodes at the first level,each of which generates p more nodes,for a total of p^2 at the second level. Each of these generates p more nodes,yielding p^3 nodes at the third level,and so on. Now suppose,that the solution is at depth d. In the worst case,we would expand all but the last node at level d,generating p^{d+1} - p nodes at level d+1. Then the total number of nodes generated is $p + p^2 + p^3 + \ldots + p^d + (p^{d+1} + p) = O(p^{d+1})$.

Every node that is generated must remain in memory,because it is either part of the fringe or is an ancestor of a fringe node. The space compleity is,therefore ,the same as the time complexity

21.What will be the final state of vacuum problem if it is sensor less? Explain with appropriate figure.

Ans: the final state of vacuum problem if it is sensor less

- No sensor
- Initial State(1,2,3,4,5,6,7,8)
- After action [Right] the state (2,4,6,8)
- After action [Suck] the state (4, 8)
- After action [Left] the state (3,7)
- After action [Suck] the state (7)
- Answer : [Right,Suck,Left,Suck] coerce the world into state 7 without any sensor
- Belief State: Such state that agent belief to be there

22. What are three distinct problem types lead by partial information?

Ans:

Sensor less or conformant problem
–Agent may have no idea where it is; solution (if any) is a sequence.

contingency problem
– Percepts provide *new* information about current state; solution is a tree or policy; often interleave search and execution.
– If uncertainty is caused by actions of another agent: *adversarial problem*

exploration problem
– When states and actions of the environment are unknown.

23. What is meant by heuristic search strategy? What is the significance of heuristic function?

Ans:

Informed(Heuristic) Search Strategies

Informed(Heuristic) search strategy is one that uses problem-specific knowledge beyond the definition of the problem itself. It can find solutions more efficiently than uninformed strategy.

Heuristic functions

A **heuristic function** or simply a **heuristic** is a function that ranks alternatives in various search algorithms at each branching step basing on an available information in order to make a decision which branch is to be followed during a search. The key component of Best-first search algorithm is a **heuristic function**, denoted by h(n):

h(n) = estimated cost of the **cheapest path** from node n to a **goal node**.

24.Explain greedy best-first search. What are the properties of greedy best-first search?

Ans:

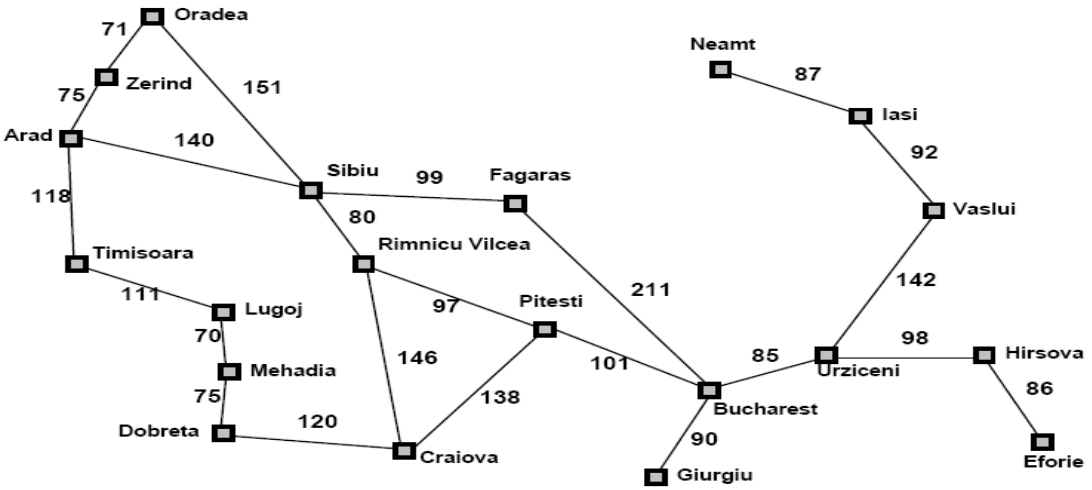
Greedy Best-first search

Greedy best-first search tries to expand the node that is closest to the goal, on the grounds that this is likely to a solution quickly. It evaluates the nodes by using the heuristic function **f(n) = h(n).**

Properties of greedy best-first search

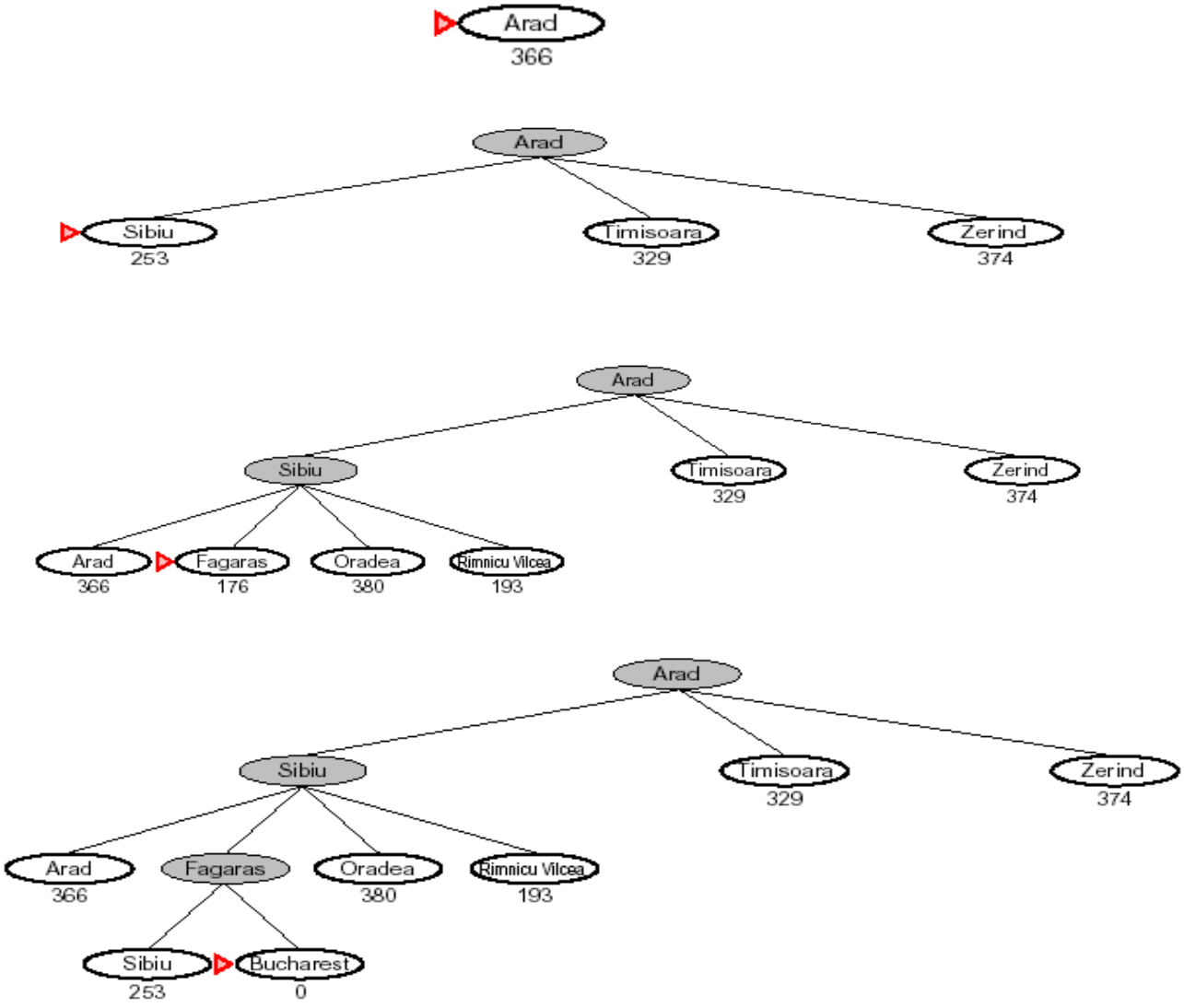
- **Complete??** No–can get stuck in loops, e.g.,
  Iasi ! Neamt ! Iasi ! Neamt !
  Complete in finite space with repeated-state checking
- **Time??** $O(b^m)$, but a good heuristic can give dramatic improvement
- **Space??** $O(b^m)$—keeps all nodes in memory
- **Optimal??** No

25. Find and explain the path from Zerind to Bucharest using greedy best-first search.



Straight–line distance to Bucharest	
Arad	366
Bucharest	0
Craiova	160
Dobreta	242
Eforie	161
Fagaras	178
Giurgiu	77
Hirsova	151
Iasi	226
Lugoj	244
Mehadia	241
Neamt	234
Oradea	380
Pitesti	98
Rimnicu Vilcea	193
Sibiu	253
Timisoara	329
Urziceni	80
Vaslui	199
Zerind	374

Ans:



26. Write short note on A* search. Use the bellow figure to find the way from 'Arad' to 'Bucharest' using A* search. Show the stages using figure

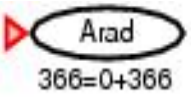
Ans: A* Search is the most widely used form of best-first search. The evaluation function f(n) is obtained by combining

(1) **g(n) =** the cost to reach the node,and

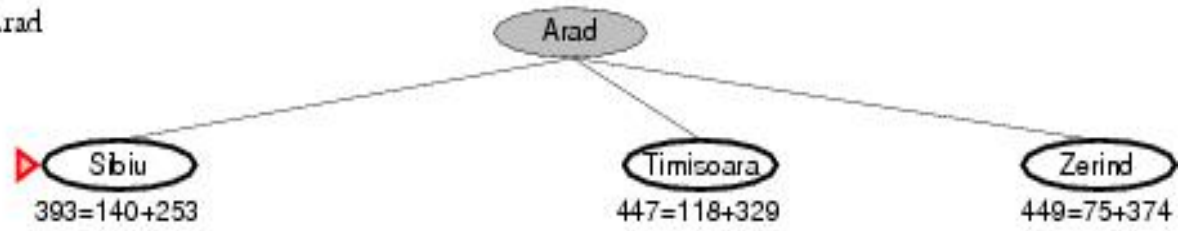
(2) **h(n) =** the cost to get from the node to the **goal** :

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

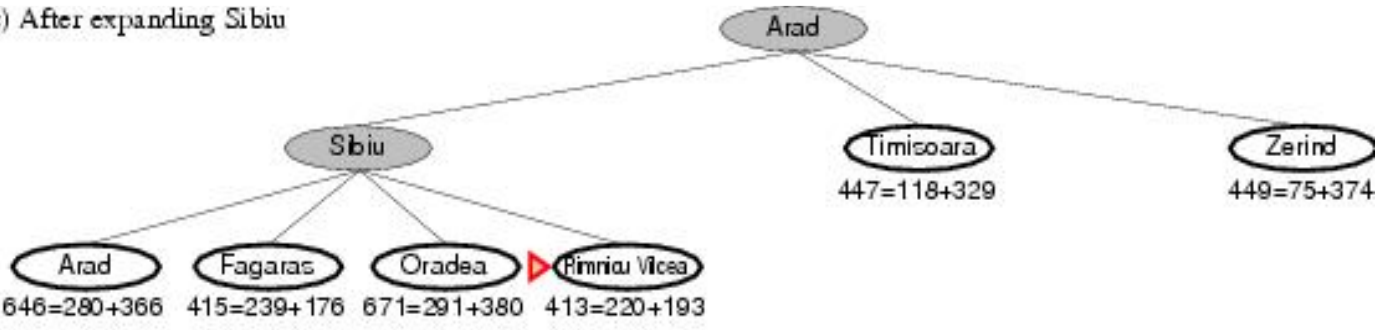
(a) The initial state



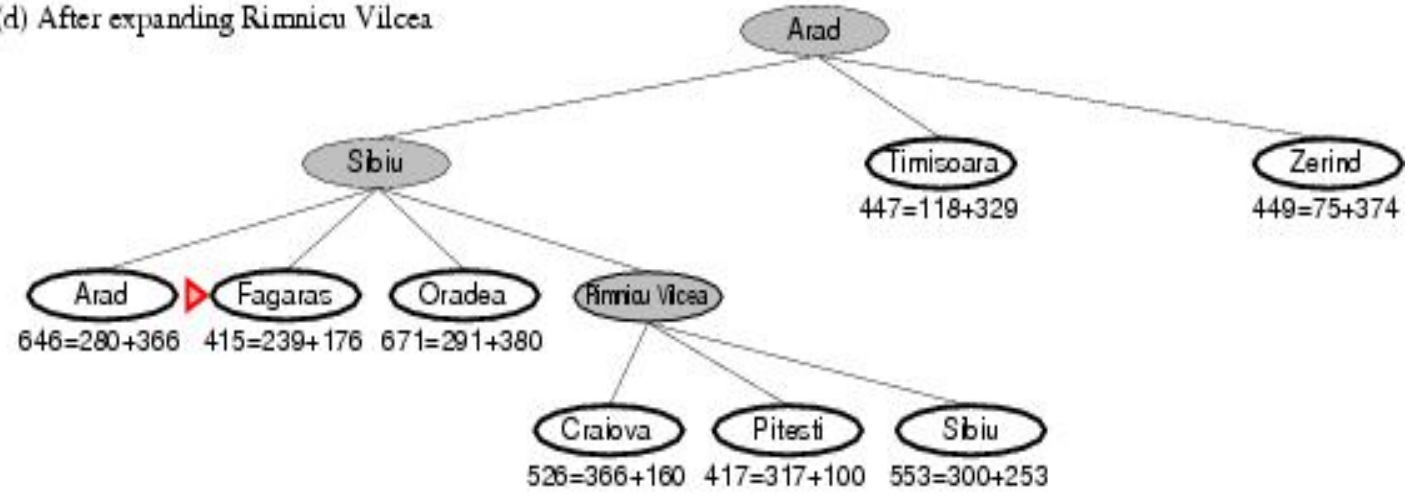
After expanding Arad



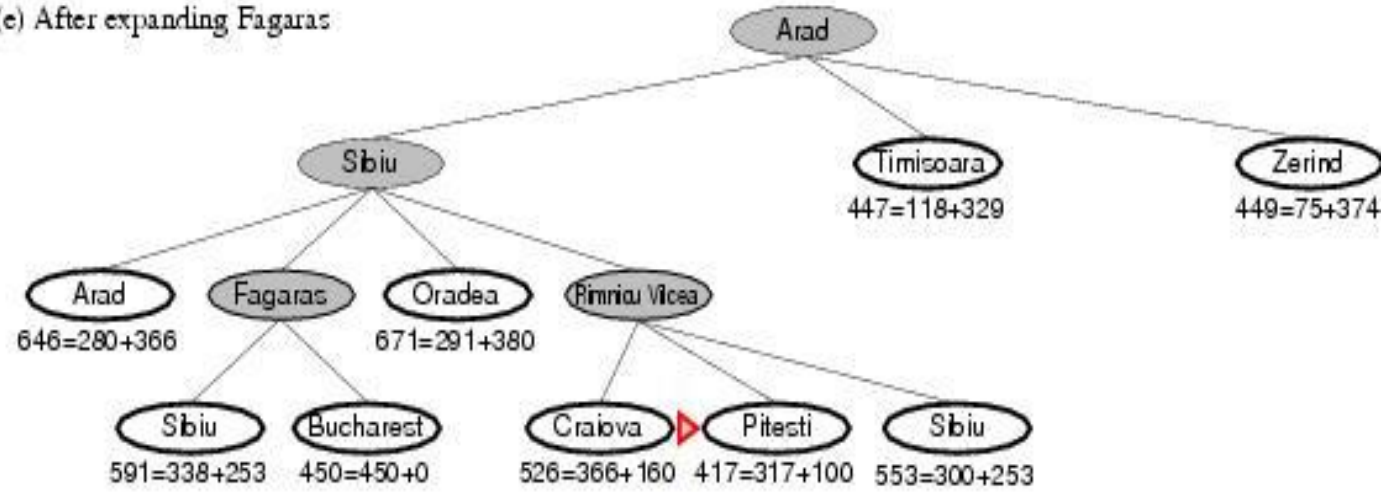
(c) After expanding Sibiu



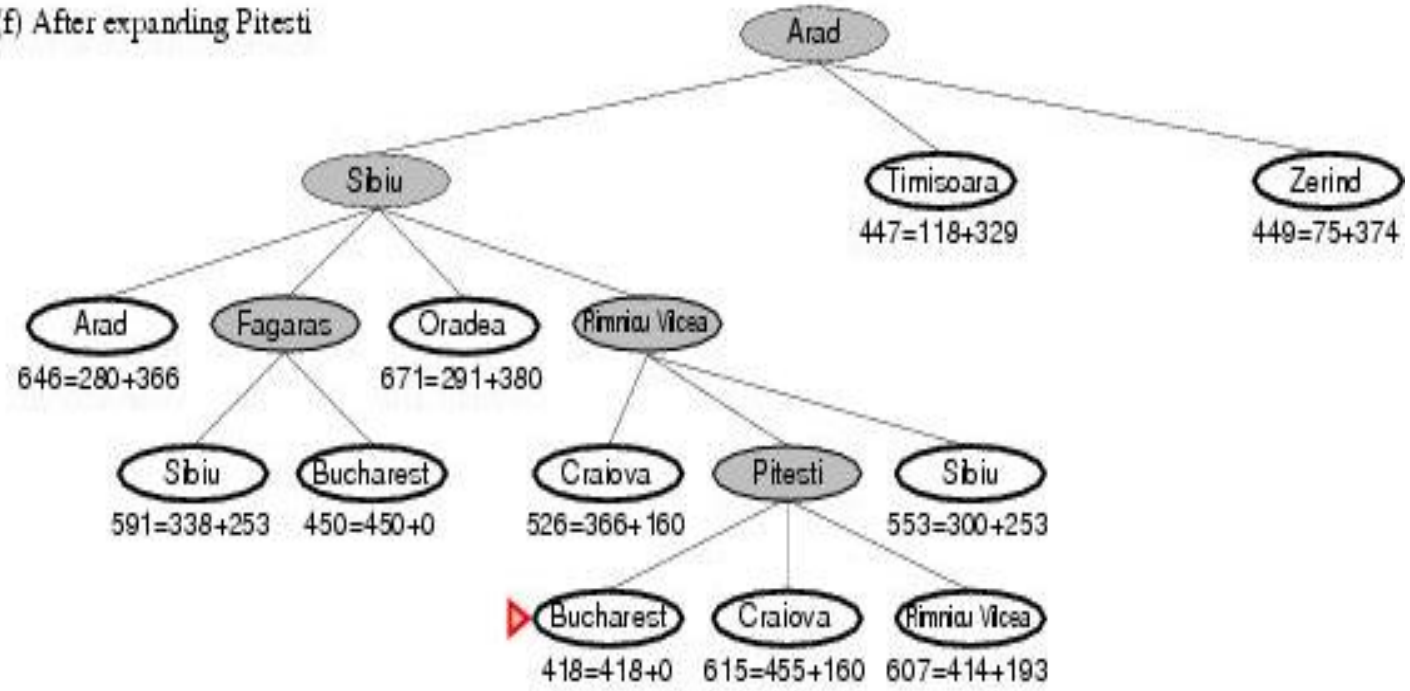
(d) After expanding Rimnicu Vilcea



(e) After expanding Fagaras



(f) After expanding Pitesti



27. Describe A* search and give the proof of optimality of A*

Ans:

A* **Search** is the most widely used form of best-first search. The evaluation function f(n) is obtained by combining

(1) **g(n) =** the cost to reach the node,and

(2) **h(n) =** the cost to get from the node to the **goal** :

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

A* Search is both optimal and complete. A* is optimal if h(n) is an admissible(Deserving to be admitted) heuristic. The obvious example of admissible heuristic is the straight-line distance h_{SLD}. It cannot be an overestimate.

A* Search is optimal if h(n) is an admissible heuristic – that is, provided that h(n) never overestimates the cost to reach the goal.

28. Explain Min-Max algorithm and Alpha –beta pruning.

Ans:

The minimax Algorithm

The minimax algorithm computes the minimax decision from the current state. It uses a simple recursive computation of the minimax values of each successor state, directly implementing the defining equations. The recursion proceeds all the way down to the leaves of the tree ,and then the minimax values are **backed up** through the tree as the recursion unwinds**.** For example in Figure 2.19,the algorithm first recourses down to the three bottom left nodes,and uses the utitliy function on them to discover that their values are 3,12,and 8 respectively. Then it takes the minimum of these values,3,and returns it as the backed-up value of node B. A similar process gives the backed up values of 2 for C and 2 for D. Finally,we take the maximum of 3,2,and 2 to get the backed-up value of 3 at the root node.

The minimax algorithm performs a complete depth-first exploration of the game tree. If the maximum depth of the tree is *m*,and there are b legal moves at each point,then the time complexity of the minimax algorithm is O(bm). The space complexity is O(bm) for an algorithm that generates successors at once.

Alpha-Beta Pruning

The problem with minimax search is that the number of game states it has to examine is **exponential** in the number of moves. Unfortunately, we can't eliminate the exponent, but we can effectively cut it in half. By performing **pruning,**we can eliminate large part of the tree from consideration. We can apply the technique known as **alpha beta pruning** ,when applied to a minimax tree ,it returns the same move as **minimax** would,but **prunes away** branches that cannot possibly influence the final decision.

Alpha Beta pruning gets its name from the following two parameters that describe bounds on the backed-up values that appear anywhere along the path:

- α : the value of the best choice we have found so far at any choice point along the path of MAX.
- β: the value of best choice we have found so far at any choice point along the path of MIN.

Alpha Beta search updates the values of α and β as it goes along and prunes the remaining branches at anode as soon as the value of the current node is known to be worse than the current α and β value for MAX and MIN,respectively.

The effectiveness of alpha-beta pruning is highly dependent on the order in which the successors are examined. It might be worthwhile to try to examine first the successors that are likely to be the best. In such case,it turns out that alpha-beta needs to examine only O(bd/2) nodes to pick the best move,instead of O(bd) for minimax. This means that the effective branching factor becomes sqrt(b) instead of b – for chess,6 instead of 35. Put anotherway alpha-beta cab look ahead roughly twice as far as minimax in the same amount of time.

29. How does the brain work? Explain how the information transmits at the synapses.

Ans:

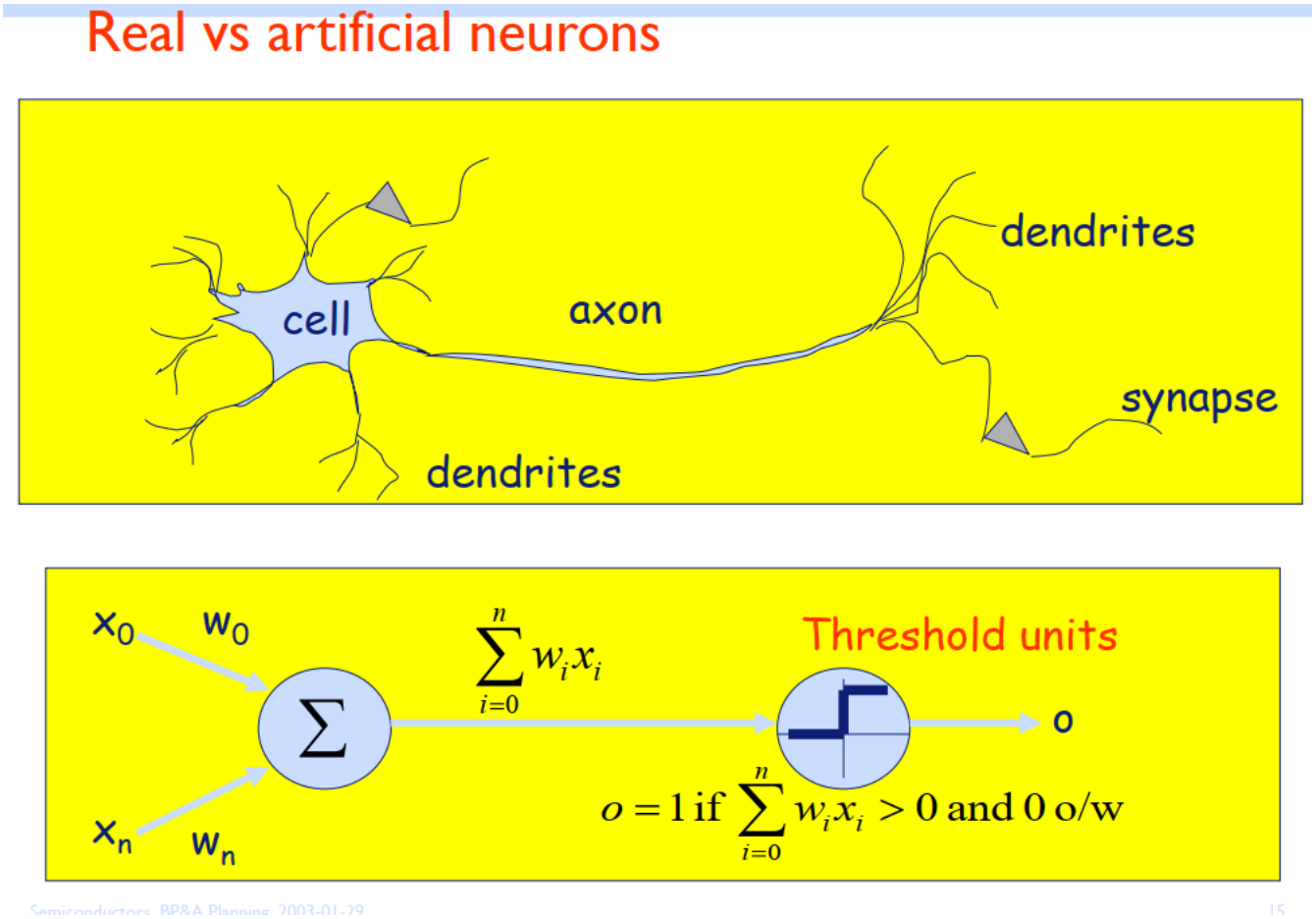
NEURON

-The cell that perform information processing in the brain
-Fundamental functional unit of all nervous system tissue
-Each consists of : SOMA, DENDRITES, AXON, and SYNAPSE

The information transmission happens at the synapses.
-The spikes travelling along the axon of the pre-synaptic neuron trigger the release of neurotransmitter substances at the synapse.
-The neurotransmitters cause excitation or inhibition in the dendrite of the post-synaptic neuron.
-The integration of the excitatory and inhibitory signals may produce spikes in the post-synaptic neuron.
-The contribution of the signals depends on the strength of the synaptic connection.

30. Draw and differentiate between real vs artificial neurons.

Ans:



31. Write short note on

a. single perceptron
b. Linear Neurons
c. Gaussian Neurons
d. Sigmoidal Neurons

Ans:

a. **single perceptron**

It's a single-unit network

Change the weight by an amount proportional to the difference between the desired output and the actual output.

$$\Delta W_i = \eta * (D-Y).I_i$$

b. **Linear Neurons**

- Obviously, the fact that threshold units can only output the values 0 and 1 restricts their applicability to certain problems.
-We can overcome this limitation by eliminating the threshold and simply turning f_i into the **identity function** so that we get:

$$o_i(t) = \text{net}_i(t)$$

-With this kind of neuron, we can build networks with m input neurons and n output neurons that compute a function

$f: R^m \rightarrow R^n$.

-Linear neurons are quite popular and useful for applications such as interpolation.
-However, they have a serious limitation: Each neuron computes a linear function, and therefore the overall network function $f: R^m \rightarrow R^n$ is also **linear**.
-This means that if an input vector x results in an output vector y, then for any factor ϕ the input $\phi \cdot x$ will result in the output $\phi \cdot y$.
-Obviously, many interesting functions cannot be realized by networks of linear neurons.

c. **Gaussian Neurons**

Another type of neurons overcomes this problem by using a **Gaussian** activation function:

$$f_i(\text{net}_i(t)) = e^{\frac{\text{net}_i(t)-1}{\sigma^2}}$$

-Gaussian neurons are able to realize **non-linear** functions.

-Therefore, networks of Gaussian units are in principle unrestricted with regard to the functions that they can realize.

-The drawback of Gaussian neurons is that we have to make sure that their net input does not exceed 1.

-This adds some difficulty to the learning in Gaussian networks.

d. **Sigmoidal Neurons**

-**Sigmoidal neurons** accept any vectors of real numbers as input, and they output a real number between 0 and 1.

-Sigmoidal neurons are the most common type of artificial neuron, especially in learning networks.

-A network of sigmoidal units with m input neurons and n output neurons realizes a network function $f: R^m \rightarrow (0,1)^n$

-The parameter τ controls the slope of the sigmoid function, while the parameter θ controls the horizontal offset of the function in a way similar to the threshold neurons.

$$f_i(\text{net}_i(t)) = \frac{1}{1+e^{-(\text{net}_i(t)-\theta)/\tau}}$$

32. What do you mean by supervised and unsupervised learning? Differentiate between them.

Ans:

Supervised learning is a machine learning approach that's defined by its use of labeled datasets. These datasets are designed to train or "supervise" algorithms into classifying data or predicting outcomes accurately. Using labeled inputs and outputs, the model can measure its accuracy and learn over time.

Unsupervised learning uses machine learning algorithms to analyze and cluster unlabeled data sets. These algorithms discover hidden patterns in data without the need for human intervention.

The main distinction between the two approaches is the use of labeled datasets. To put it simply, supervised learning uses labeled input and output data, while an unsupervised learning algorithm does not.

Goals: In supervised learning, the goal is to predict outcomes for new data. You know up front the type of results to expect. With an unsupervised learning algorithm, the goal is to get insights from large volumes of new data. The machine learning itself determines what is different or interesting from the dataset.

Applications: Supervised learning models are ideal for spam detection, sentiment analysis, weather forecasting and pricing predictions, among other things. In contrast, unsupervised learning is a great fit for anomaly detection, recommendation engines, customer personas and medical imaging.

Complexity: Supervised learning is a simple method for machine learning, typically calculated through the use of programs like R or Python. In unsupervised learning, you need powerful tools for working with large amounts of unclassified data. Unsupervised learning models are computationally complex because they need a large training set to produce intended outcomes.

Drawbacks: Supervised learning models can be time-consuming to train, and the labels for input and output variables require expertise. Meanwhile, unsupervised learning methods can have wildly inaccurate results unless you have human intervention to validate the output variables.

33. State the back-propagation algorithm for updating weights in a multilayer network.

Ans:

1. Initialize the weights in the network
2. **repeat**

   for each example e in the training set **do**

   i. O = neural-net-output(network, e) ; forward pass
   ii. T = teacher output for e
   iii. Calculate error (T - O) at the output units
   iv. Compute **wj = wj +α * Err * Ij** for all weights from hidden layer to output layer;backward pass
   v. Compute **wj = wj +α * Err * Ij** for all weights from input layer to hidden layer; backward pass continued
   vi. Update the weights in the network

   end

3. **until** all examples classified correctly or stopping criterion met
4. **return**(network)

34. **What are the advantages and disadvantages of neural networks.**

Ans:

Advantage:

1. Distributed representations
2. Simple computations
3. Robust with respect to noisy data
4. Robust with respect to node failure
5. Empirically shown to work well for many problem domains
6. Parallel processing

Disadvantages:

1. Training is slow
2. Interpretability is hard
3. Network topology layouts ad hoc
4. Can be hard to debug
5. May converge to a local, not global, minimum of error
6. May be hard to describe a problem in terms of features with numerical values

35. **State all six steps of training in a back propagation network.**

Ans:

1. Initialize weights with small, random values

2. While stopping condition is not true

for each training pair (input/output):

each input unit broadcasts its value to all hidden units

each hidden unit sums its input signals & applies activation function to compute its output signal

each hidden unit sends its signal to the output units

each output unit sums its input signals & applies its activation function to compute its output signal

3. Each output computes its error term, its own weight correction term and its bias(threshold) correction term & sends it to layer below

4. Each hidden unit sums its delta inputs from above & multiplies by the derivative of its activation function; it also computes its own weight correction term and its bias correction term

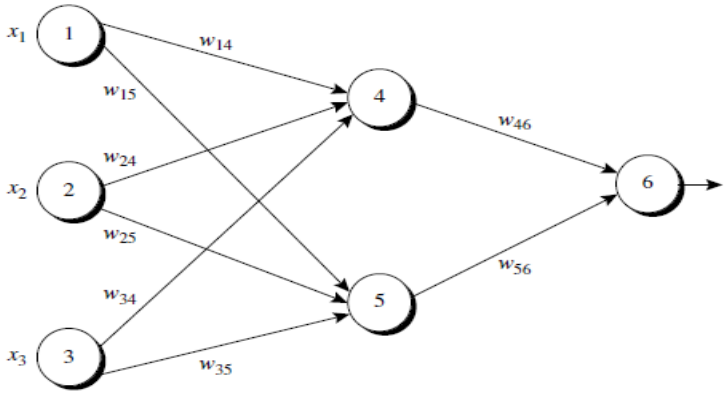
5. Each output unit updates its weights and bias

6. Each hidden unit updates its weights and bias

Each training cycle is called an epoch. The weights are updated in each cycle

It is not analytically possible to determine where the global minimum is. Eventually the algorithm stops in a low point, which may just be a local minimum.

36. Figure shows a multilayer feed-forward neural network. Let the learning rate be 0.9. The initial weight and bias values of the network are given in Table, along with the first training tuple, *X = (1, 0, 1), with a class label of y=1.*



Initial Input, Weight, and Bias Values

x_1	x_2	x_3	w_{14}	w_{15}	w_{24}	w_{25}	w_{34}	w_{35}	w_{46}	w_{56}	θ_4	θ_5	θ_6
1	0	1	0.2	−0.3	0.4	0.1	−0.5	0.2	−0.3	−0.2	−0.4	0.2	0.1

Find out the following:

(a) Net Input and Output (b) Error at Each Node (c) Weight and (d) Bias Updating

Ans:

Table 9.2. Net Input and Output Calculations

Unit, j	Net Input, I_j	Output, O_j
4	$0.2 + 0 - 0.5 - 0.4 = -0.7$	$1/(1 + e^{0.7}) = 0.332$
5	$-0.3 + 0 + 0.2 + 0.2 = 0.1$	$1/(1 + e^{-0.1}) = 0.525$
6	$(-0.3)(0.332) - (0.2)(0.525) + 0.1 = -0.105$	$1/(1 + e^{0.105}) = 0.474$

Table 9.3. Calculation of the Error at Each Node

Unit, j	Err_j
6	$(0.474)(1 - 0.474)(1 - 0.474) = 0.1311$
5	$(0.525)(1 - 0.525)(0.1311)(-0.2) = -0.0065$
4	$(0.332)(1 - 0.332)(0.1311)(-0.3) = -0.0087$

Table 9.4. Calculations for Weight and Bias Updating

Weight or Bias	New Value
w_{46}	$-0.3 + (0.9)(0.1311)(0.332) = -0.261$
w_{56}	$-0.2 + (0.9)(0.1311)(0.525) = -0.138$
w_{14}	$0.2 + (0.9)(-0.0087)(1) = 0.192$
w_{15}	$-0.3 + (0.9)(-0.0065)(1) = -0.306$
w_{24}	$0.4 + (0.9)(-0.0087)(0) = 0.4$
w_{25}	$0.1 + (0.9)(-0.0065)(0) = 0.1$
w_{34}	$-0.5 + (0.9)(-0.0087)(1) = -0.508$
w_{35}	$0.2 + (0.9)(-0.0065)(1) = 0.194$
θ_6	$0.1 + (0.9)(0.1311) = 0.218$
θ_5	$0.2 + (0.9)(-0.0065) = 0.194$
θ_4	$-0.4 + (0.9)(-0.0087) = -0.408$

37. Write short notes on cross-validation.

Ans:

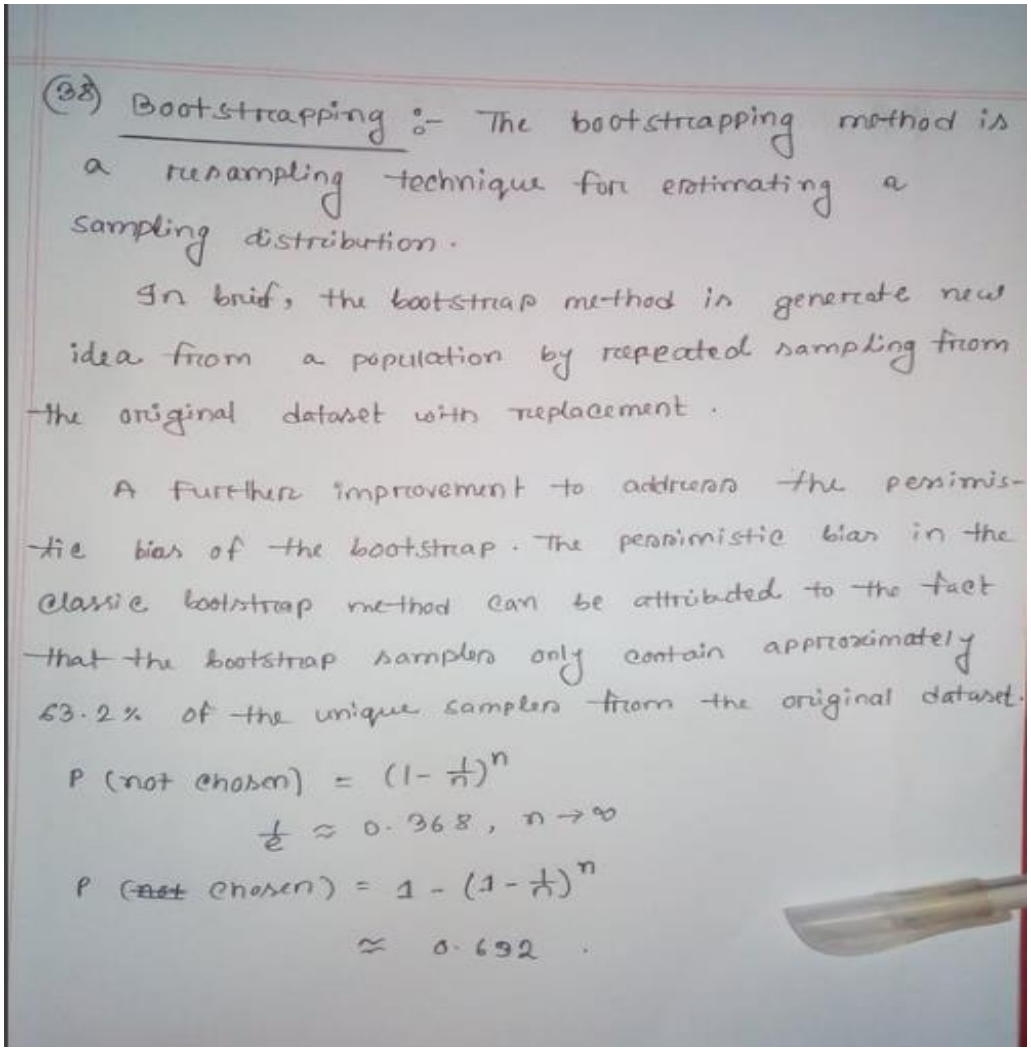
Cross validation defined as:

"A statistical method or a resampling procedure used to evaluate the skill of machine learning models on a limited data sample."

It is mostly used while building **machine learning models**. It compares and selects a model for a given predictive modeling problem, assesses the models' predictive performance. Later judges how they perform outside to a new data set, also known as **test data**. The motivation to use cross validation techniques is that we are **holding** it to a training dataset when we fit a model.

38. What is Bootstrapping? What is the significance of 0.632 bootstrapping technique and Why it is so called?

Ans:



39. The data tuples of table are sorted by decreasing probability value, as returned by a classifier. For each tuple, compute the values for the number of true positives (*TP*), false positives (*FP*), true negatives (*TN*), and false negatives (*FN*). Compute the true positive rate (*TPR*) and false positive rate (*FPR*). Plot the ROC curve for the data.

Tuple #	*Class*	*Probability*
1	*P*	0.95
2	*N*	0.85
3	*P*	0.78
4	*P*	0.66
5	*N*	0.60
6	*P*	0.55
7	*N*	0.53
8	*N*	0.52
9	*N*	0.51
10	*P*	0.40

Ans:

TP: A **true positive** is an outcome where the model *correctly* predicts the *positive* class.

0.95+0.78+0.66+0.55+0.40=3.34

TPR: True positive rate is 0.668

TN: A **true negative** is an outcome where the model *correctly* predicts the *negative* class.

0.85+0.60+0.53+0.52+0.51=3.01

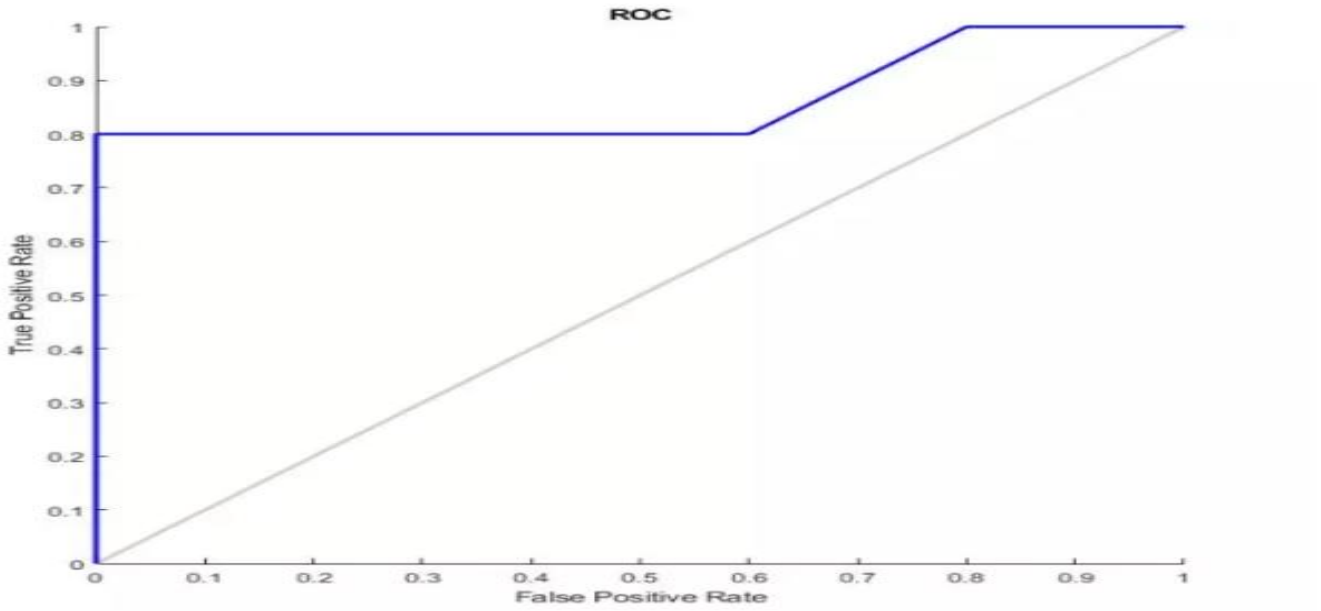
FP: A **false positive** is an outcome where the model *incorrectly* predicts the *positive* class.

0.05+0.22+0.34+0.45+0.60=1.66

FPR: False Positive rate is 0.332

FN: A false negative is an outcome where the model *incorrectly* predicts the *negative* class.

0.15+0.40+0.47+0.48+0.49=1.99



---

40. What are the four basic types of agent program in any intelligent system? Explain how did you convert them into learning agents?.

Ans:

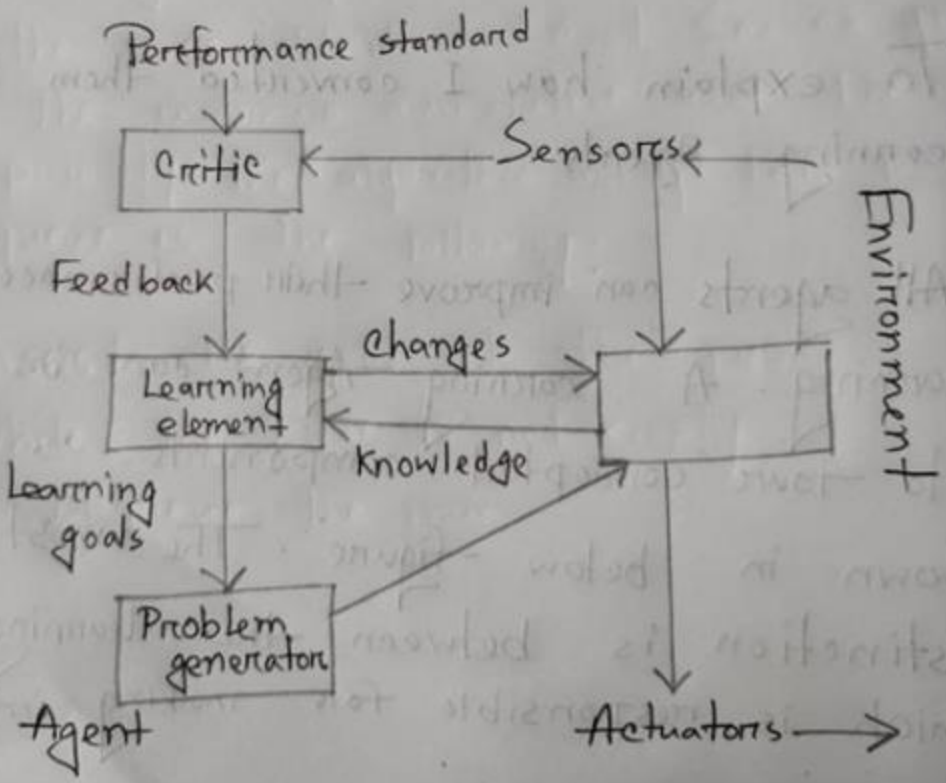
The four basic types of agent program in any intelligent system are :-

1. Simple Reflex Agents.
2. Model- Based Reflex Agents.
3. Goal- Based Agents
4. Utility- Based Agents.

To explain how I converted them into learning Agents

All agents can improve their performance through learning. A learning Agent can be divided into four conceptual components. as shown in below figure. The most important distinction is between the learning element, which is responsible for making improvements.

and the performance element, which is responsible for making external actions. The performance element is what we have previously considered to be the entire agent: it takes the in percepts and decides on actions. The last component of the learning agent is the problem generator. It is responsible for suggesting actions that will lead to new and informative experiences.



41. Explain the following uninformed search strategies with examples.

i. Breadth First Search. (4)
ii. Uniform Cost Search (4)
iii. Depth First Search (4)
iv. Depth Limited Search (4)

Ans:

i. Breadth First Search: Breadth First Search (BFS) is an uninformed search strategy that explores all the vertices of a graph at a given depth level before moving to the next level. It starts with the initial state and expands all possible successor states before moving to their successors. It uses a queue to keep track of the unexplored states.

Example:
Consider a maze with the following representation:
S - Start
G - Goal

- Wall
| S | | | |

| G | # | | G |

| | | | G |

To find the path from the start to the goal using BFS, the algorithm would expand the nodes in the following order: S, G1, G2, G3, G4.

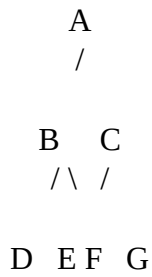
ii. Uniform Cost Search: Uniform Cost Search (UCS) is an uninformed search strategy that expands the nodes in order of their path cost from the initial state to the current node. It always selects the node with the lowest total cost so far and explores it next.

Example:
Consider a map with the following path costs:
A -> B: 4
A -> C: 2
C -> D: 1
B -> E: 3
E -> G: 5
C -> F: 2
F -> G: 4

To find the path from node A to G using UCS, the algorithm would expand the nodes in the following order: A, C, D, F, G.

iii. Depth First Search: Depth First Search (DFS) is an uninformed search strategy that explores a branch of the search tree as deeply as possible before backtracking. It goes as deep as possible exploring each successor before backtracking to explore remaining successors.

Example:
Consider a binary tree with the following structure:

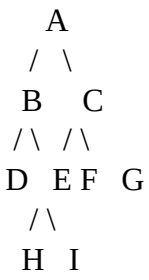


To traverse this tree using DFS, the algorithm would explore the nodes in the following order: A, B, D, E, C, F, G.

iv. Depth Limited Search : Depth Limited Search is an uninformed search strategy that combines the advantages of depth-first search and breadth-first search. It explores the search space by traversing down each branch until a specified depth limit is reached before moving to the next branch.

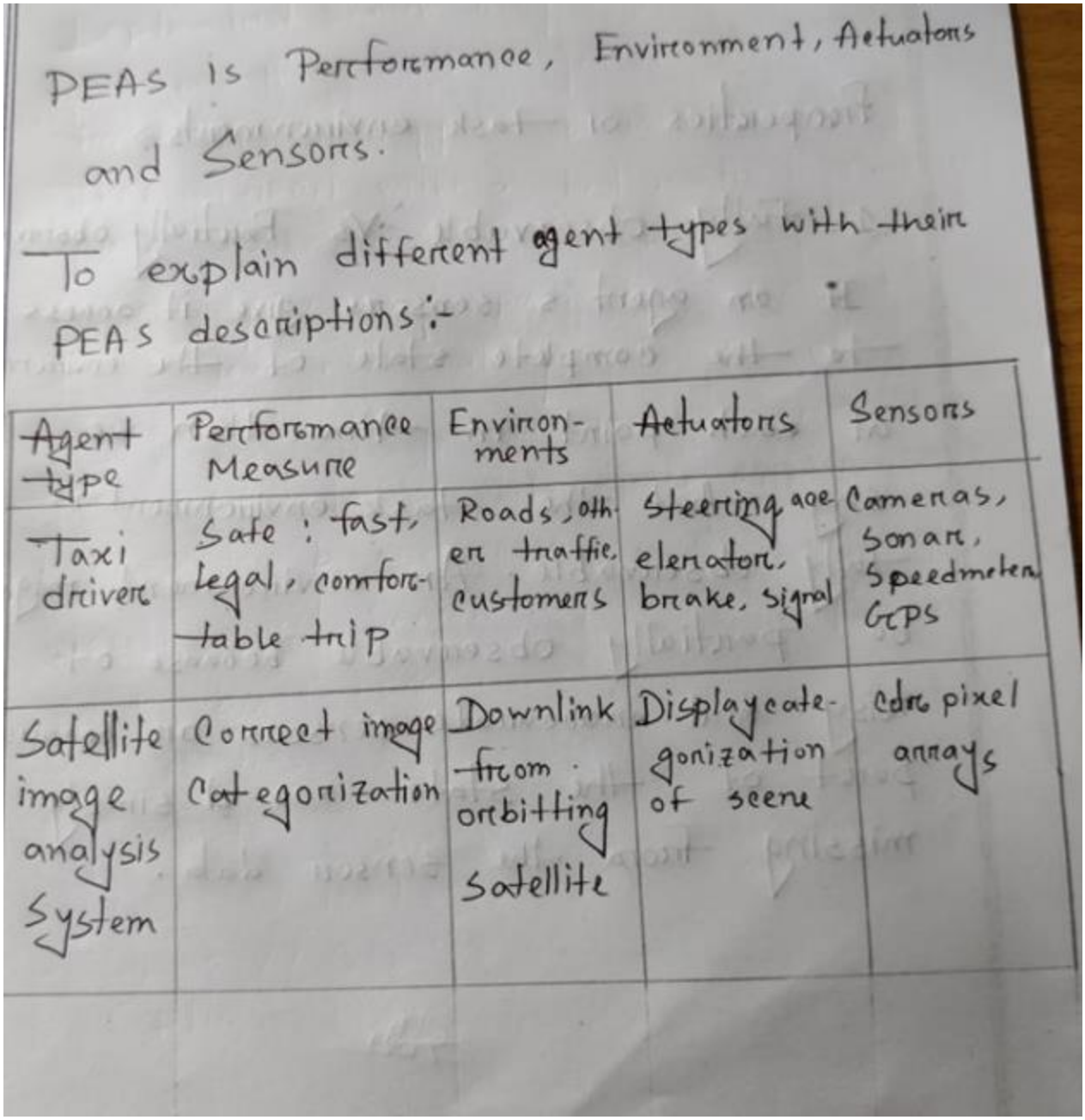
Example:

Consider a scenario where we have a binary tree representing a search space with various nodes. The goal is to find a specific node within the tree using Depth Limited Search with a depth limit of 3.



42. What is PEAS? Explain different agent types with their PEAS descriptions.

Ans:



PEAS is Performance, Environment, Actuators and Sensors.

To explain different agent types with their PEAS descriptions:-

Agent type	Performance Measure	Environments	Actuators	Sensors
Taxi driver	Safe : fast, legal, comfortable trip	Roads, other traffic, customers	Steering, accelerator, brake, signal	Cameras, sonar, speedometer, GPS
Satellite image analysis system	Correct image categorization	Downlink from orbitting satellite	Display categorization of scene	color pixel arrays

43. Explain in detail the properties of Task Environments.

Ans: Fully observable vs. **partially observable**

If an agent's sensors give it access to the complete state of the environment at each point in time, then we say that the task environment is fully observable. A task environment is effectively fully observable if the sensors detect all aspects that are *relevant* to the choice of action. An environment might be **partially observable** because of noisy and inaccurate sensors or because parts of the state are simply missing from the sensor data.

Deterministic vs. **stochastic**

If the next state of the environment is completely determined by the current state and the action executed by the agent, then we say the environment is deterministic; other- wise, it is stochastic.

Episodic vs. **sequential**

In an **episodic task environment**, the agent's experience is divided into atomic episodes. Each episode consists of the agent perceiving and then performing a single action. **Crucially, the next episode does not depend on the actions taken in previous episodes.** For example, an agent that has to spot defective parts on an assembly line bases each decision on the current part, regardless of previous decisions;

In **sequential environments**, on the other hand, the current decision could affect all future decisions. Chess and taxi driving are sequential.

Discrete vs. **continuous.**

The discrete/continuous distinction can be applied to the *state* of the environment, to the way *time* is handled, and to the *percepts* and *actions* of the agent. For example, a discrete-state environment such as a chess game has a finite number of distinct states. Chess also has a discrete set of percepts and actions. Taxi driving is a continuous state and continuous time problem: the speed and location of the taxi and of the other vehicles sweep through a range of continuous values and do so smoothly over time. Taxi-driving actions are also continuous (steering angles, etc.).

Single agent vs. **multiagent.**

An agent solving a crossword puzzle by itself is clearly in a single-agent environment, whereas an agent playing chess is in a two-agent environment.

44. Define a problem and its components. Explain how a problem solving agent works?

Ans:

A well-defined problem can be described by:

- **Initial state**
- **Operator or successor function** - for any state x returns s(x), the set of states reachable
- from x with one action
- **State space** - all states reachable from initial by any sequence of actions
- **Path** - sequence through state space
- **Path cost** - function that assigns a cost to a path. Cost of a path is the sum of costs of individual actions along the path
- **Goal test** - test to determine if at goal state

A **problem** can be formally defined by **four components**:

- The **initial state** that the agent starts in. The initial state for our agent of example problem is described by *In(Arad)*
- A **Successor Function** returns the possible **actions** available to the agent. Given a state x,SUCCESSOR-FN(x) returns a set of {action,successor} ordered pairs where each action is one of the legal actions in state x,and each successor is a state that can be reached from x by applying the action.

For example,from the state In(Arad),the successor function for the Romania problem would return

{ [Go(Sibiu),In(Sibiu)],[Go(Timisoara),In(Timisoara)],[Go(Zerind),In(Zerind)] }

- **State Space** : The set of all states reachable from the initial state. The state space forms a graph in which the nodes are states and the arcs between nodes are actions.
- A **path** in the state space is a sequence of states connected by a sequence of actions.
- Thr **goal test** determines whether the given state is a goal state.
- A **path cost** function assigns numeric cost to each action. For the Romania problem the cost of path might be its length in kilometers.
- The **step cost** of taking action a to go from state x to state y is denoted by c(x,a,y). The step cost for Romania are shown in figure 1.18. It is assumed that the step costs are non negative.
- A **solution** to the problem is a path from the initial state to a goal state.
- An **optimal solution** has the lowest path cost among all solutions.

Problem-solving agents

A Problem-solving agent is a **goal-based** agent. It decides what to do by finding sequence of actions that lead to desirable states. The agent can adopt a goal and aim at satisfying it.

45. Explain real-world problems with examples.

Ans:

Real-world problems refer to challenges or issues that individuals, communities, or societies face in their everyday lives. Here are some examples of real-world problems along with their explanations:

1. **Poverty:** This problem occurs when individuals or families lack sufficient income or resources to meet their basic needs, such as food, shelter, healthcare, and education. Poverty leads to a variety of hardships and reduces opportunities for upward mobility. For instance, a family struggling to afford nutritious meals or unable to access quality education due to financial constraints.

2. Global warming: This problem refers to the increase in the Earth's average temperature due to excessive greenhouse gas emissions, primarily from human activities. Global warming leads to climate change, causing various environmental issues like melting polar ice, increased severe weather events, rising sea levels, and disruptions to ecosystems. An example would be the impact of global warming on coral reefs, which are experiencing bleaching and degradation due to increased ocean temperatures.

3. Unemployment: This problem arises when individuals capable of working and seeking employment are unable to find suitable jobs, leading to financial strains and reduced quality of life. High levels of

unemployment can lead to social and economic instability. For example, during an economic recession, numerous individuals lose their jobs and struggle to find new opportunities, causing financial distress and societal unrest.

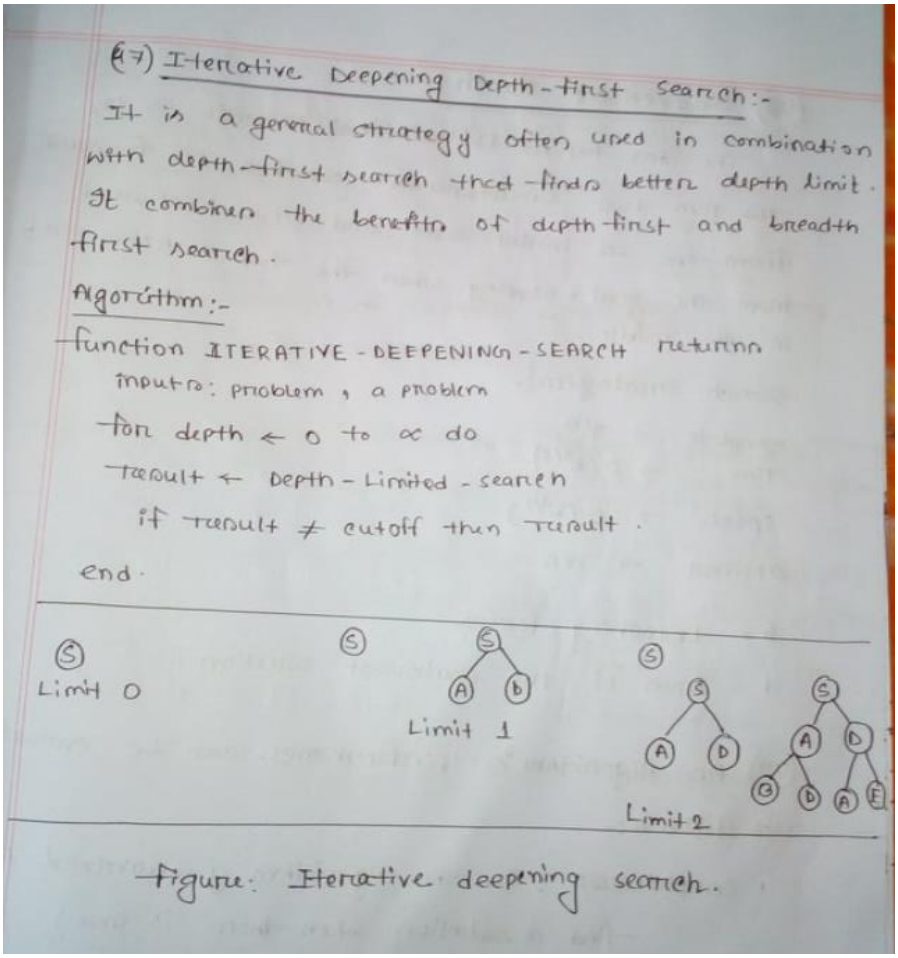
4. Gender inequality: This problem highlights the unequal treatment and opportunities based on gender, where women often face discrimination and limited access to education, healthcare, employment, and participation in decision-making processes. An example is the gender pay gap, where women are paid less than men for similar work, leading to financial disparities and perpetuating the cycle of inequality.

5. Food insecurity: This problem occurs when individuals or communities lack consistent access to adequate and nutritious food to maintain an active and healthy life. Food insecurity affects individuals globally, with examples including malnourished children in developing countries or individuals living in food deserts — areas without access to affordable and nutritious food sources, commonly found in low-income neighborhoods.

46. Explain in detail with examples

47**. Iterative deepening search**

Ans:



48. Bidirectional search (8)

Ans:

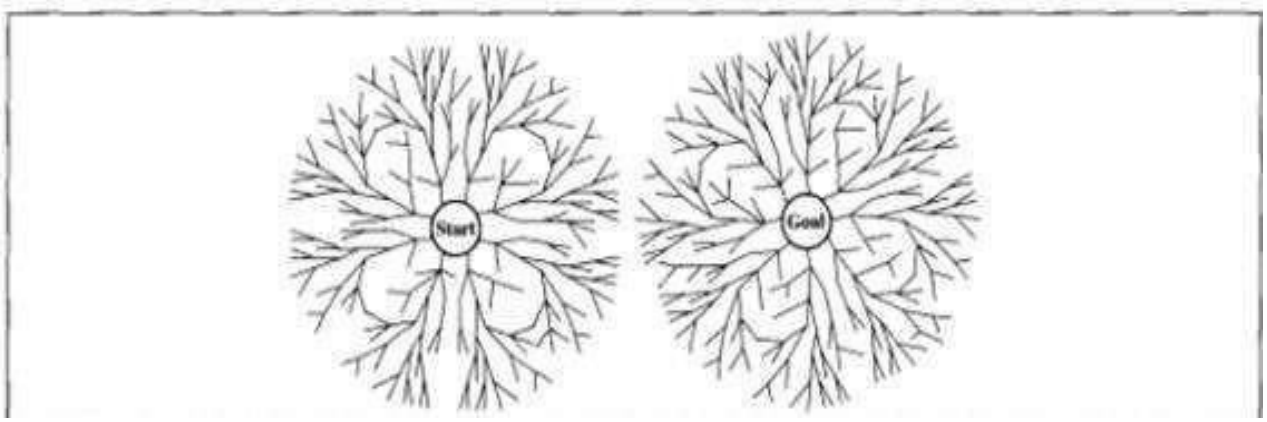
(48) Bidirectional search :-

The idea behind bidirectional search is to run two simultaneous searches one forward from the initial state and other backward from the goal, stopping when the two searches meet in the middle.

Search Strategies :-

complete → yes
Time → $O(b^{d/2})$
space → $O(b^{d/2})$
optimal → Yes.

b = branching factor
d = depth of the shallowest solution.



49. How an algorithm's performance is evaluated? Compare different uninformed search strategies in terms of the four evaluation criteria.

Ans:

The algorithm's performance can be measured in four ways :

- **Completeness** : Is the algorithm guaranteed to find a solution when there is one?
- **Optimality** : Does the strategy find the optimal solution
- **Time complexity** : How long does it take to find a solution?
- **Space complexity** : How much memory is needed to perform the search?

compares search strategies in terms of the four evaluation criteria

Criterion	Breadth-First	Uniform-Cost	Depth-First	Depth-Limited	Iterative Deepening	Bidirectional (if applicable)
Complete?	Yes[a]	Yes[a,b]	No	No	Yes[a]	Yes[a,d]
Time	$O(b^{d+1})$	$O(b^{1+\lfloor C^*/\epsilon \rfloor})$	$O(b^m)$	$O(b^\ell)$	$O(b^d)$	$O(b^{d/2})$
Space	$O(b^{d+1})$	$O(b^{1+\lfloor C^*/\epsilon \rfloor})$	$O(bm)$	$O(b\ell)$	$O(bd)$	$O(b^{d/2})$
Optimal?	Yes[c]	Yes	No	No	Yes[c]	Yes[c,d]

Figure 1.38 Evaluation of search strategies, b is the branching factor; d is the depth of the shallowest solution; m is the maximum depth of the search tree; l is the depth limit. Superscript caveats are as follows: a complete if b is finite; b complete if step costs >= E for positive E; c optimal if step costs are all identical; d if both directions use breadth-first search.

50. What is Greedy Best First Search? Explain with an example the different stages of Greedy Best First search.

Ans:

Greedy Best-first search

Greedy best-first search tries to expand the node that is closest to the goal, on the grounds that this is likely to a solution quickly.

It evaluates the nodes by using the heuristic function **f(n) = h(n).**

Stages: question 25 er naswer

51. What is A* search? Explain various stages of A* search with an example. (16)

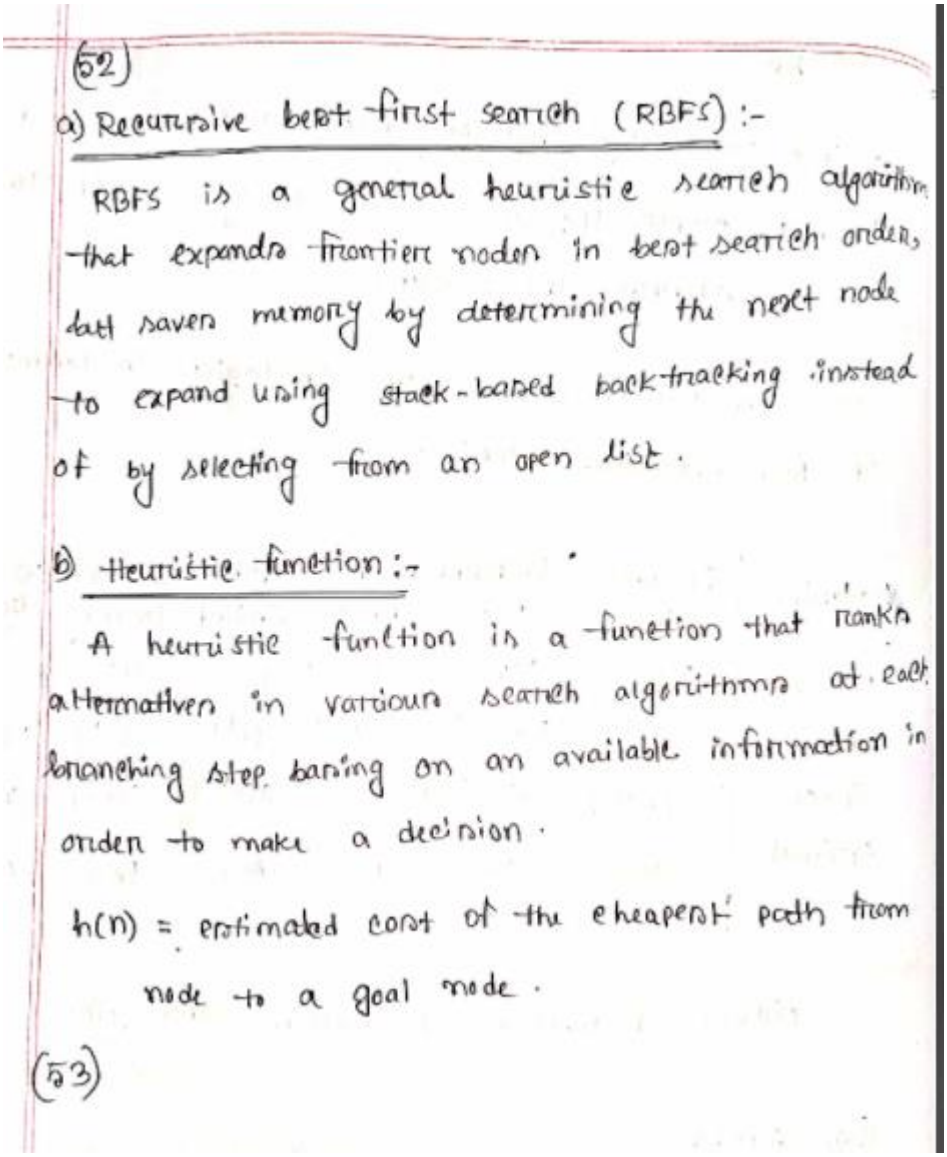
Ans: question 26 er naswer

52. Explain in detail with examples

a. Recursive Best First Search(RBFS) (8)

b. Heuristic Functions (8)

Ans:



(52)

a) Recursive best-first search (RBFS) :-

RBFS is a general heuristic search algorithm that expands frontier nodes in best search order, but saves memory by determining the next node to expand using stack-based back-tracking instead of by selecting from an open list.

b) Heuristic function :-

A heuristic function is a function that ranks alternatives in various search algorithms at each branching step basing on an available information in order to make a decision.

h(n) = estimated cost of the cheapest path from node to a goal node.

(53)

53. Explain the following local search strategies with examples.

a. Hill climbing (4)

b. Genetic Algorithms (4)

c. Simulated annealing (4)

d. Local beam search (4)

Ans:

(53)

a. Hill Climbing :- Hill climbing algorithm is a local search algorithm which continously moves the direction of increasing elevation to find the peak of the best solution to the problem. It is a technique which is used for optimizing the mathematical problems. For Example, Traveling salesman problem in which we need to minimize the distance traveled by the salesman.

b. Genteic Algorithm: Genetic algorithms are adaptive heuristic search algorithms that belong to the larger part of evolutionary algorithms. They are based on the ideas of natural selection and genetics. For Example, Recurrent neural network, mutation testing, code breaking, filtering and signal processing, learning fuzzy rule base etc.

c. Simulated annealing:- Annealing is the process of heating and cooling a metal to change its

internal structure for modifying its physical properties. In simulated annealing process, the temperature is kept variable.

d. Local Beam search:- In this algorithm, it holds k number of states at any given time. At the start these states are generated randomly. The successors of these k states are computed with help of objective function.

function BeamSearch (problem, k) returns solution

Start with k randomly generated

Loop

generate all o successors of all k states

If any of the states = solution, return

else select the k best successors.

end.

54. Define constraint satisfaction problem(CSP). How CSP is formulated as a search problem? Explain with an example. (16)

Ans:

(54)

Constraint satisfaction problem (CSP):-

Constraint satisfaction problem (CSP) is a technique where a problem is solved when its value satisfy certain constraints or rules of the problem. Such type of technique leads to a deeper understanding of the problem structure as well as its complexity. For example, Sudaku playing, n-queen problem, crossword.

In local state-spaces, the choice is only one to search for a solution. But in CSP, we have two the choices either: we can search for a solution or we can perform a special type of interference called constraint propagation.

Example: Graph colouring : colour the graph of each region with same color. WA, NT, Q, NSW, V, SA, T are regions. The domain set of color {Red, green, blue}.

{WA = red, NT = green, Q = red, NSW = green}.

55. Explain with examples

a. Constraint graph (4)

b. Cryptarithmetic problem (4)

c. Adversarial search problem (4)

d. Game (4)

Ans:

(55)

(a) Constraint Graph:- The nodes of the graph correspond to variables of the problem and the arcs correspond to constraints.

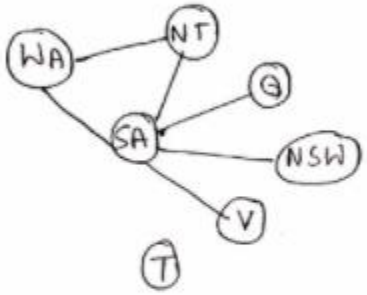
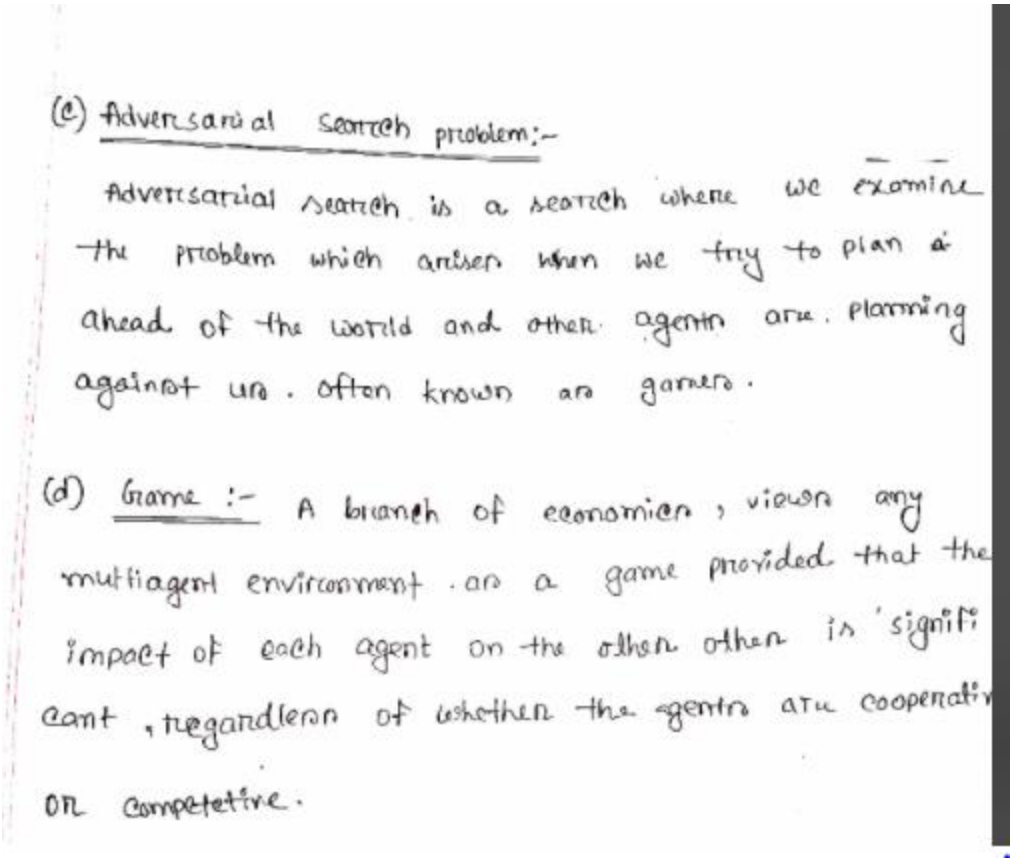


Figure: constraint graph.

(b) Cryptarithmetic problem:- It is a type of constraint satisfaction problem where the game is about digits and its unique replacement either with alphabets or other symbols.


(c) Adversarial Search problem:-

Adversarial search is a search where we examine the problem which arises when we try to plan ahead of the world and other agents are planning against us. often known as games.

(d) Game :- A branch of economics, views any multiagent environment as a game provided that the impact of each agent on the other other is 'significant, regardless of whether the agents are cooperative or competetive.

56. Explain with algorithm and example :

i. Minimax algorithm

ii. Alpha-Beta Pruning

Ans:

(56) Explain with algorithm and example:

(i) Miniman Algorithm:-

The minimax algorithm computes the minimax decision from the current state.

function MINIMAX-DECISION (state) returns

Input state, current state

← MAX-VALUE (state)

return the action in Successors

function MAX-VALUE (state) returns

if TERMINAL-TEST (state) then return UTILITY (state)

$v \leftarrow -\infty$

do

return v

function MIN-VALUE (state) returns

if TERMINAL-TEST (va state) then return -UTILITY (st

$v \leftarrow \infty$

do

return v.

(ii) Alpha-Beta Pruning :- Alpha-Beta Pruning is a modified version of the minimax algorithm.

Condition of Alpha-Beta pruning is $\alpha >= \beta$.

-function ALPHA-BETA-SEARCH (State) returns

inputs : State, current state in game

$v \leftarrow$ MAX-VALUE (State, $-\infty$, $+\infty$)

return the action in SUCCESSORS (State).

-function MAX-VALUE (State, α, β) returns

inputs : State, current state in game

α, the value of best alternative for MAX

β, the value of best alternative for MIN.

$v \leftarrow -\infty$

-for a, s in SUCCESSORS (State) do

$v \leftarrow$ MAX (v, MIN-VALUE) (s, α, β)

if $v \geq \beta$ then return v

$\alpha \leftarrow$ MAX (α, v)

return v.