U23ITT62

ARTIFICIAL INTELLIGIENCE AND MACHINE LEARNING

UNIT-I PROBLEM SOLVING

Introduction to AI - AI Application - Problem Solving agents - Search algorithm - Uninformed Search Strategies - Heuristic Search Strategies - Local Search and optimization Problems - adversarial Search - Constraint Satisfaction Problem.

L1: Introduction to AI

AI is the branch of cs focused on building machines and ability of Computer to do task like human intelligence, such as making decision, Problem solving etc..

⟹ father of AI - John McCarthy.

Here, Artificial defines, "man made" and Intelligence defines "thinking Power".

Tasks & foundation of AI:-

1) Understanding Language (NLP)
   ex:- Chat GPT, Google translate.
2) Learning from data (ML)
   ex:- Netflix recommendation based on watching
3) Recognizing images or objects (computer vision)
   ex: Face recognizion, Imag classification.
4) Making Decision:-
   ex: Self driving car.

5) Reasoning And Problem ...
ex: Games like chess.

Tools In AI :-

1) Programming language.
   - Python (Most popular lang for AI).
   - R (used. in statistics and data analysis)
2) Machine Learning Lib (Build model from data)
   - Tensorflow (Deep learning, Neaural Network)
   - pytorch (Research friendly deep learning)
   - Matplot (Grap Rep).
3) Computer vision tool (Image Reco).
   - open cv (Image Processing).
4) Natural Language Processing (understand human lang)
   - Spacy (fast NLP tasks)
5) Reinforcement learning tools (Games/Robotic
   - open AI Gym (Train agents).
6) Data & visualization tools.
   - Pandas (Data analysis).
   - Matplot (Seaborn Data vaise Graph)
   - Numpy (Numerical computing).

Advantages of AI :-

→ High Accuracy with less error & efficiency
→ 24/7 Availability
→ High Speed / faster decision making.
→ Reduce human Risks.
→ useful as a public utility.

Dis Advantage of AI:-

→ High cost
→ Job Displacement
→ Lack of human Judgment
→ Data Privacy issues
→ No feelings and emotions.

AI Application:-

1) Healthcare
   - Disease Diagnosis (cancer, heart)
   - Medical imaging (x-ray, MRI, ECG, EEG)
2) Robotics:-
   - Autonomous Robots (Industry, delivery)
   - Surgical Robots (operation).
   - Service Robots (hotels)
3) E-Commerce:
   - Recommendation
   - Virtual Shopping (Amazon).
4) Smartphone:-
   - Face Recognition in clock.
5) Self driving car. (Autonomous vehicles)
6) Banking a finance
   - fraud detection
   - Credit Scoring.
7) Education:
   - Personalize learning.
8) NLP
   - Translation
   - voice to text
   - Sentiment Analysis.

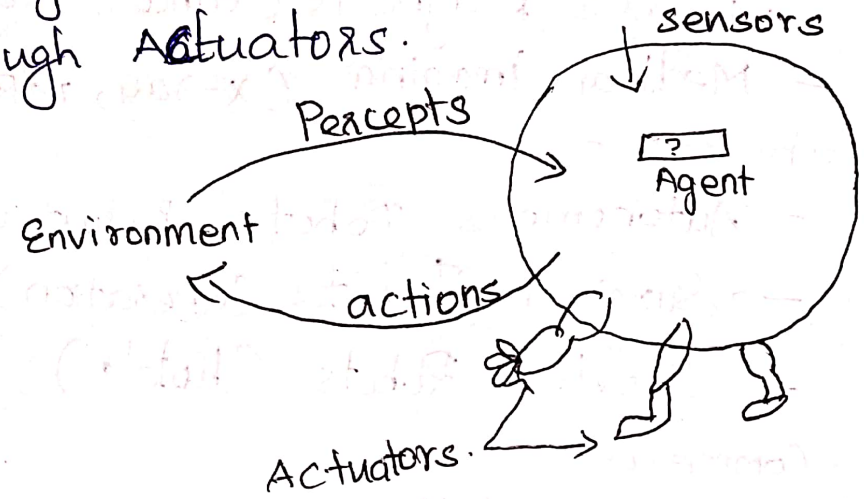
9) Government & Public services.
- Traffic System.
- Public safety monitoring.

Agent function: Input O/P Sequence to

L2: Problem Solving Agent:-

⇒ Agent and Environment.

An Agent is anything that can be viewed as perceiving its environment through sensors & acting upon that environment through Actuators.



$F: P^* \rightarrow A$

F:- Agent function.

P - Percept Sequence

A - Action.

Sensor:- used to observes the current environment.

Actuators: Acting a environment through. actuators.

Ex:- In human Agent: eye, ears, and other. organs are sensors, and hands, legs, month & other body parts for actuators.

In Robotic Agent:- Cameras and infrared range finder for sensors and various motors for actuators.

⇒ Input devices - sensor.
O/P devices - Actuators.

Agent function: It is used to maps the percept sequence to an action.

Agent are having dataset that contain (historical data of environment) + (map the current environment status.

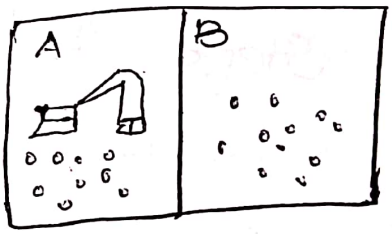
Agent Program:-

The Agent function for artificial agent will be implemented by an agent program.

The agent program is a concrete implementation, running on the agent architecture.

Vaccum Cleaner world:-

In this we have 2 location. one is A & B.



* Percepts :- location, content (A, B) eg [A, Dirty].

* Action :- left, Right, Suck, No operation.

* Tabulation of vaccum Agent function :-

Percept Sequence	Actions
A, clean	Right
A, Dirty	Suck
B, clean	left
B, Dirty	SUCK
If both are clean	No operation.

Types of Agent:-

* Simple Reflex Agent
* Model based reflex Agent
* Goal based Agent
* Utility based Agent
* Learning Agent.

L3: Problem Solving Agent:-

A kind goal based agent - choose their action in order to achieve Goals.

This allows the agent to a way of choosing among multiple possibilities, selecting the one which reaches a Goal state.

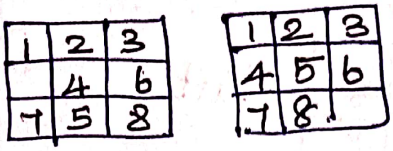
It requires Searching and planning techniques : eg: GPS - finding a path to certain destination.

Steps Performed by problem Solving agent.

Goal formulation
↓
Problem formulation.
↓
Search Solution.
↓
Execution.

Goal formulation:- It is the first and simplest step in problem solving. It organizes the steps and sequence required to formulate one goal out of multiple goals.

Problem



Problem Formulation:-

It is the most important step of problem solution which descides what action should be taken to achieve the formulated goal.

→ Initial State:- It is the starting state of the agent towards its goal.
(1,2,3/b,4,6/7,5,8).

→ Action:- It is the possible action available to the agent. (Left, Right, up, down).

→ Goal Test:- It determine if the given state is a goal state. (1,2,3/4,5,6/7,8,b)

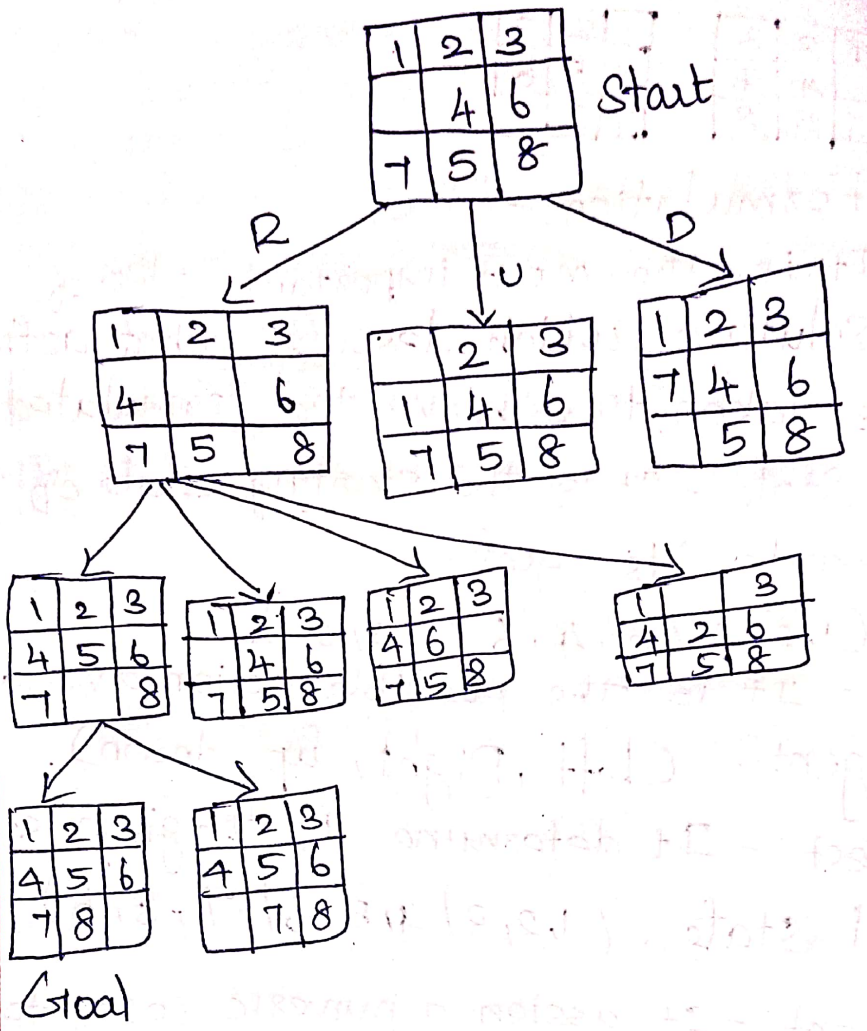
→ Path Cost:- It assign a numeric cost to each path that follows the goal.
(cost / per action).

Search Solution:-

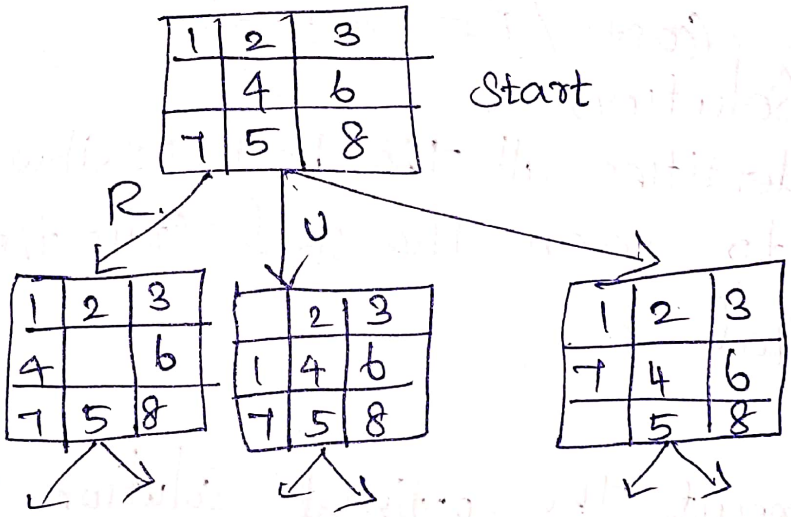
It identifies all the best posible sequence of action to reach the goal state from the current state:-

Execution:-

It execute the optimal solution from the searching algorithm to reach the goal state from the current state. An optimal solution has the lowest path cost among all the solutions.



Execution:



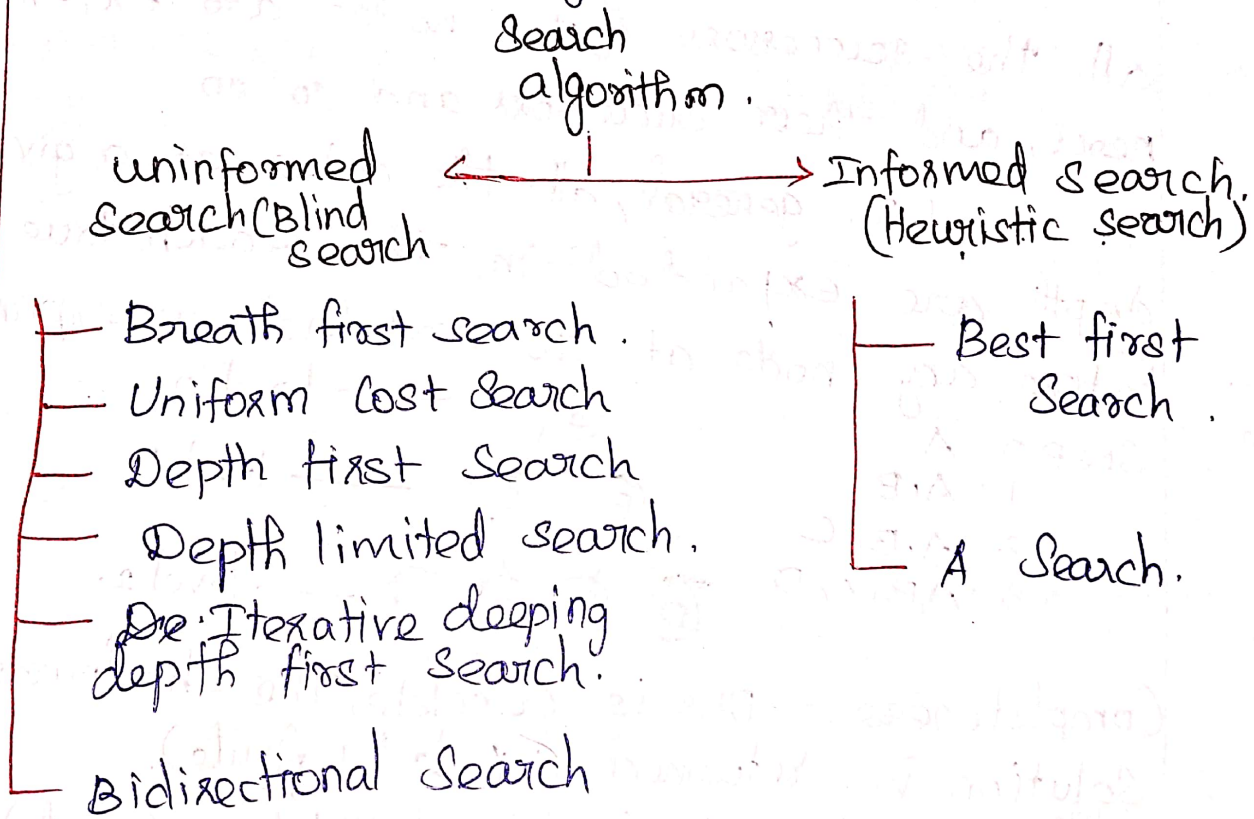
Search form the core component of many intelligent process.

Search is the systematic examination of states to find path from the start/root state to the goal state.

Many traditional search algorithm are used in AI application.

For complex problem, the traditional algorithm are unable to find the solution within some practical time and space limits.

Consequently, many spacial techniques are developed using heuristic function.



15. Uniformed Search strategies:-

→ uniformed search algorithm generates the search tree without using any domain knowledge. (get knowledge only from the knowledge definition).

→ This algorithm generates successors using the Successor fanction.

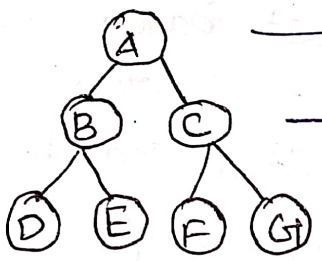
→ knowledge about Goal state and not Information about the path cost from the Current State to goal state.

i) Breadth First Search:-

→ The Root node is expanded first and then all the successors of the nodes are expended. next, and their successors and so on.

→ In general, all the nodes at a given depth are expanded in the search true before any node at the next level are expanded.

Step 0: A
1: A, B
2: A, B, C
3: A, B, C, D



Completeness:- BFS is Complete, The shallowest. Solution is returned (if b is finite).

Time Complexity:- $b + b_2 + b_3 + \cdots + b_d = O(b, d)$.

b - node at every State

d - depth of shallowes solution.

→ Time requirement is still a major factor.

based on memory size
→ space is the bigger problem (more than time),
optimal :- BFS is optimal if path cost is a non decreasing function of the depth of the node.

2) Uniform Cost Search :-

Instead of expanding the shallowest node, Uniform Cost Search expand the node n with the lowest path cost.

Uniform Cost doesn't not care about the number of steps a path has only about their total Cost.

Completeness :- UCS is complete, such if there is a solution UCS will find it.

Time Complexity :-

Let c^* is cost of the optimal solution, ε is a goal node. Then no of steps is $c^*/\varepsilon+1$

Here we have taken +1, as we start from State 0 & end to c^*/ε

worst case time complexity of UCS is $O(b^{1+}(c^*/\varepsilon))$

Space complexity :-

same logic for space complexity, so the worst case space Complexity of UCS is $O(b^{1+}(L^*/\varepsilon))$.

Optimal :-

UCS is always optimal as it only select a path with the lowest path Cost.

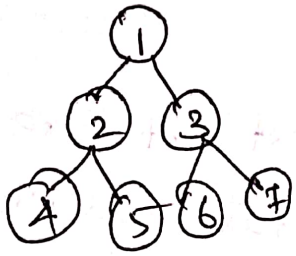
3) DEPTH FIRST SEARCH

Start with root node and complexity exp... left most child node, before exploring its sib...

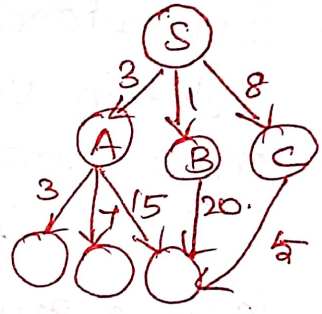
Siblings are explored in left to right fa...

Depth first search always expands the ~~request~~ deepest node in the current frontier of the search tree.

Depth 1st traversal. 1 → 2 → 4 → 5 → 3 → 6 → 7



Expanded node	Nodes list.
	So
So	A3 B1 C8
A3	D6 E10 G18 B1 C8
~~D6~~	E10 G18 B1 C8
~~D6~~	
E10	G18 B1 C8
G18	B1 C8



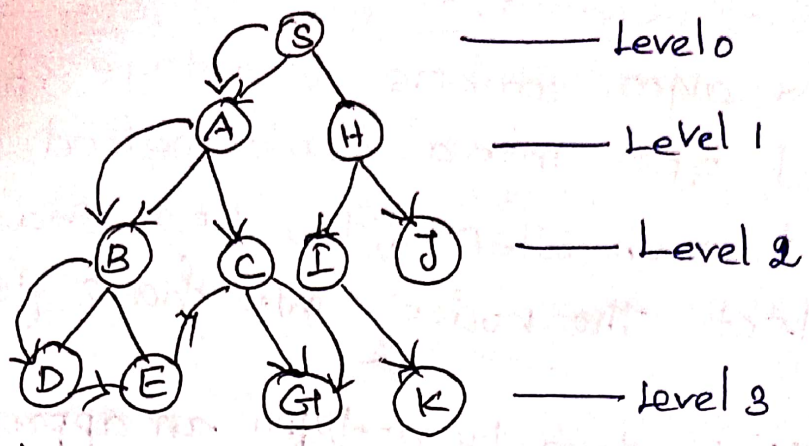
Solution path found is SAG1, Cost is

No of nodes expanded {including goal nodes} = 5

4) Depth limited Search Algorithm:-

Depth limited search is the combination of DFS and Limits for level.

Depth limited search is also similar to the DFS as it also implements "Last in first Out (LIFO) Stack data Structure" but in addition it has a level limit.



Completeness :-

DFS is complete within finite state space as expand every node within a limited search tree.

Time Complexity :- Time complexity of DFS will be equivalent to the node traversed by the algorithm $O(b-1)$

$$T(n) = 1 + n_2 + n_3 + \cdots + n_m = O(n_m)$$

m - maximum depth.

Space Complexity :-

DFS needs to store only single path from the root node, Hence space complexity of DFS is equivalent to the ~~single~~ size of the fringe set. which is $(O(b^* - 1))$

Optimal : DFS is a non optimal, it generate large no of steps or high cost to reach the goal node.

1-b Informed Search Algorithm :-

This informed search strategy is one that uses problem - specific knowledge beyond the definition of the problem itself. It can find solution more efficiently than an uninformed strategy.

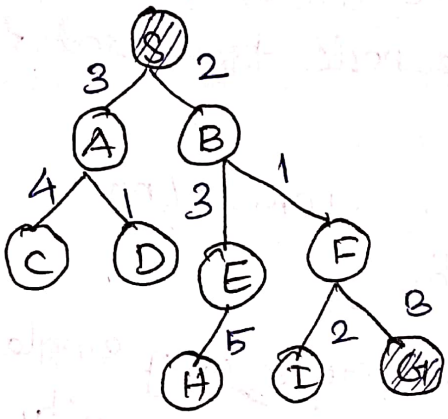
1) Best first Search:-

BFS algm combine advantage of DFS and BFS into a single method.

At each step of the BFS search we select the nodes we have generated so far.

This is done by applying an appropriate heuristic function to each of them.

Then expand the chosen node by using the rules to generate its successors.



Node	H(n)
A	12
B	4
C	7
D	3
E	8
F	2
H	4
I	9
S	13
G	0

2) A* Search:-

↳ It is most widely used form of BFS.

The evaluation function f(n) is obtained by.

g(n) - Cost to reach the node.

h(n) = Cost to get from node to goal

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

Local Search and optimization Problem:-

Local Search Algorithm:-

The LSA Search only the final state not the path to get there.

Ex: 8 Queens problem (we can only about finding a valid final configuration of 8 queens (8 queen arranged on chess board and no queen can attack other 8 queens) and not the path from initial state to final state.

Local Search algorithm operate by searching from a start state to neighboring states

Without keeping track of a path, nor the set of states that have been reached.

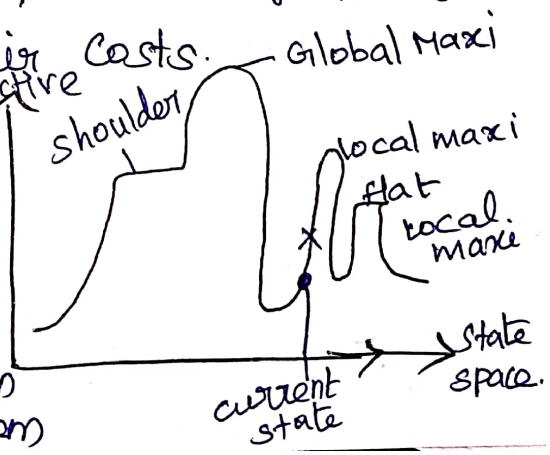
They might never explore a portion of the search space when a solution actually resides,

They search only the final state.

Ex: Hill Climbing Search algorithm:-

A state space landscape is a graph of states associated with their costs.

Hill climbling algorithm is a heuristic search algorithm which continuously move in the direction of increasing value to find the peak of the mountain or best solution to the problem



It keep track of one current state each iteration move to the neighbouring with highest value. that is if head direction that provides the steepest

App of Local Search Algorithm:-

- integrated Circuit Algorithm
- Factory floor layout
- Automatic Programming
- Telecommunication network optimization.
- Crop planning and Portfolio management.

Adv of LSA:-

→ use very little memory

→ They can often find reasonable Solution in large or infinite state space for which systematic algorithms are unsuitable.

→ Local Search algorithm can also Solve optimization problem

→ They find best state awarding to an objective function.

→ It is a game playing techniques where the agents are surrounded by a competitive environment → A conflicting goal is given to theagents (multiagent).

→ This agents complete with one another and try to defect one another in order to win the game.

→ Such conflicting goal giverise to the adversarial search.

→ Here, games playing means discussing those games when human intelligence and logic factor is used, excluding other factors such as luck factor.

→ Tic-tac-toe, chess, cheekers etc.. such type of games where no luck factor works, only mind works.

Game theory:-

→ According to game theory, a game played b/w two players, To complete the game one has to win the game and other to looser automatically.

Consider a game with 2 player Max and Min. Max move 1st followed by Min.

A game can be formally defined as a search problem with the following elements.

→ So the initial state.

→ player(s): Define which player has to move in a state(s).

→ Action(s) - Return the set of legal move in a state.

→ Result(s,a) - The transition model which define the result of a move.

→ Terminal - Test(s) - True when game is over, false otherwise.

→ utility (s,p) - This function also objective function or playoff function.

→ Defines the final numeric value for a game that ends in a terminal state s for player p.

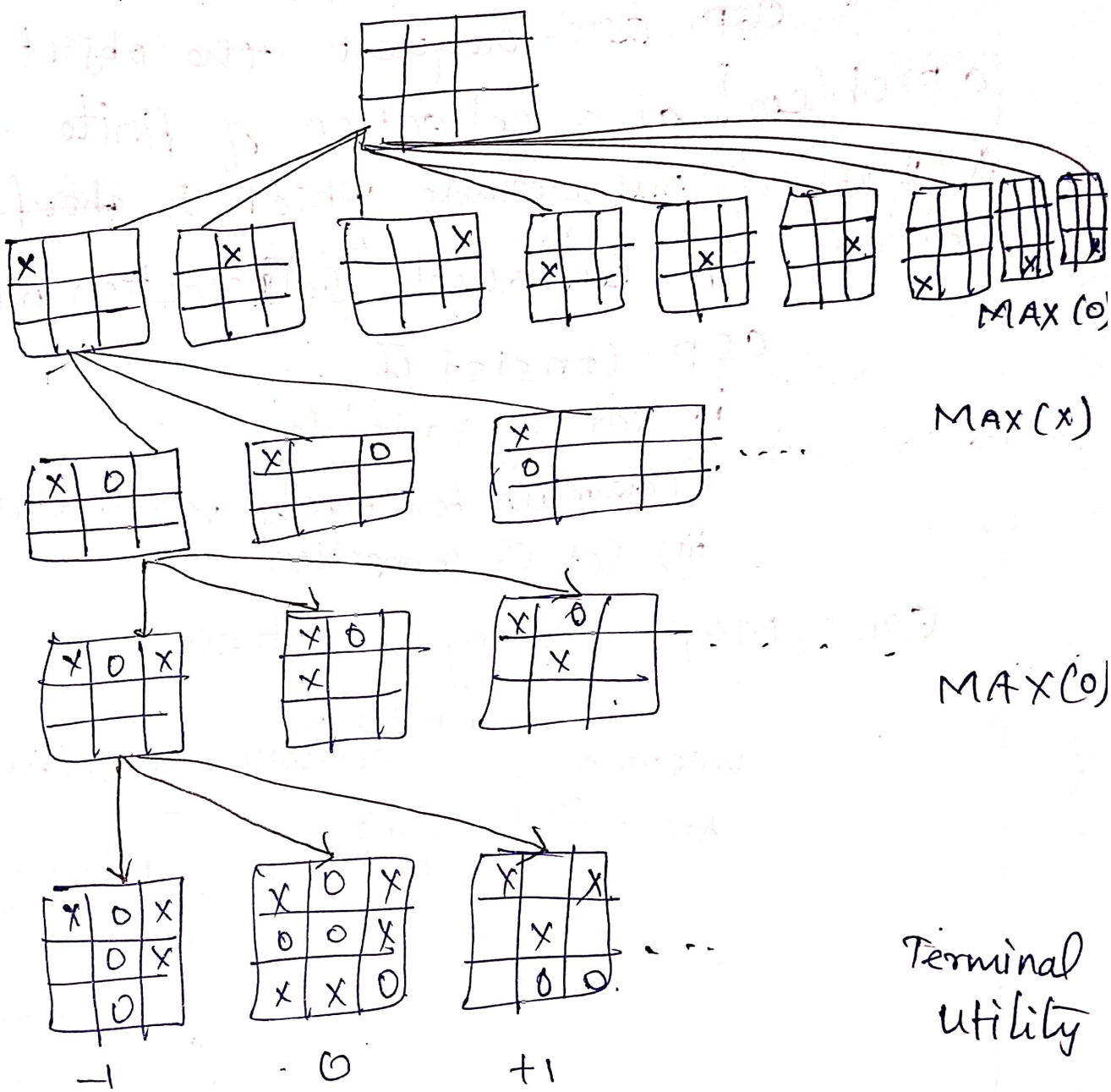
eg:- In tic-tac-toe, the outcome is win, loss or draw with values (-1, 0, or 1)

Max has 9 possible moves from the initial state.

(10)

play alternates b/w Max placing an X and Min placing an O.

The number in each leaf node indicates the utility value of terminal state from the point of view of MAX (Good for MAX, bad for Min).



Lg) CSP - Constraint Satisfaction

CSPs are mathematical question defined as a set of objects whose state must satisfy a number of constraints or limitation.

CSPs are request the object in a problem, as a collection of finite constraints out variable which is ~~should~~ solved by constraints satisfaction methods.

CSP Consist of
i) set of variable
ii) domains for each variable and
iii) set of connection

Ex:- Map coloring problem.