ARTIFICIAL INTELLIGENCE

AI

__/__/__

Man made
Intelleigence → Thinking power

- Resoning
  - Setat process
  - Judgement
  - Bredection
  - Making Dicission
- learning
  - gaining knowledge
- Perseption
  - Processor
  - accuring on interpeting, or
  - selecting or senceiring or organary.
- Problem solving
  - Ability
- Linguistic Intellegence
  - Communication

Artificial Intelligence :-

AI is a Simmulation at human Intelligence process by machine.

AI VS ML

AI is an umbrella term which minimises human cognition to perform complex tasks and learn from them.

ML is a Subset at AI, it is a method to train a computer to learn from data and inputs without explicit Programming for every scenario or circumstance.

- ML helps a computer to Achieve AI.

DLL :- It has more complexity than ML.
(imagine if this is more than 1000 hidden layers)

Turing Test :-

- By Alan Turing in 1950
- Turing test is used to determine whether or not machine ~~think~~ can think intelligently like human.
- There will be a human interogetup in one room and in other room there will be one machine and one human.
- When human interogetup is unable to distinguish whether the response is from machine and human, the machine is intellegent.

Rational :- Based on logical Thoughts (does not takes the dicission Emotionally)

Agent :- Came from the latin word 'Agere'. It means "to do".

Rational Agent :- Computer program may do something but computer agents are expected to do more such as operate autonomasly perceive from the environment, persists over a prolont time period adupt to change, create an pursive goals.
— A rational agent is one that acts as to achieve the best outcome.
Eg:- Self driving Car – intelligent agent.

Intelligent Agent :- The Intelligent Agent Sense the Environment through Sensors & act through the actuators.
an agent runs on cycle at perceiving, thinking & acting.

Agent = Architecture + Agent program.

Vaccum Cleaner agent:- (As a problem Solving Agent)

- PSA Solve the problem by following certain path to reach the goal State.
- Important Statement for PSA is Problem formulation
- Problem formulation Shows actions to the agent to achieve goal.

① Initial Step —

- from where we will Start
- Any State Can be Considered as an initial State.
- In our Eg:— we may Start from Room A, Room B.

② Action Step — in our example each State has 3 actions like left, right & ~~Sucking~~ Sucking.

— Success funⁿ for a Vaccum cleaner Agent:—

Result (S, a)
S = State
a = action.

- After taking action 'a', the PSA will lead to which ~~State~~ State (that function will return).
- Eg:- ~~Room~~ First the agent is in 'Room A' (State) having dust in the room, So action Sucking & after that it will return that State (clean state)

__/__/__

- then the agent will check whether this state is a 'goal state' or not. ~~If it is not~~

- If it is not the goal state, so the same funⁿ will be again applied from the current state.

- in the next state the room A is "clean" so agent will move to right (action), so that it will go to the next state return the state.

- In the third state, room is 'B' & action = " such the dust"

- After sucking, it will return the state i.e. empty or goal state.

③ Goal State (Test State) :- After performing action, agent is in which state known as goal state.

- If both rooms are clean, agent reach the goal state.

④ Path Cost State :-

- Cost b/w initial state to goal state is called as Path cost. Cost may be assigned a numerical value (generally 1)

- The path cost is essential, to verify the efficiency of the agent.

Agents

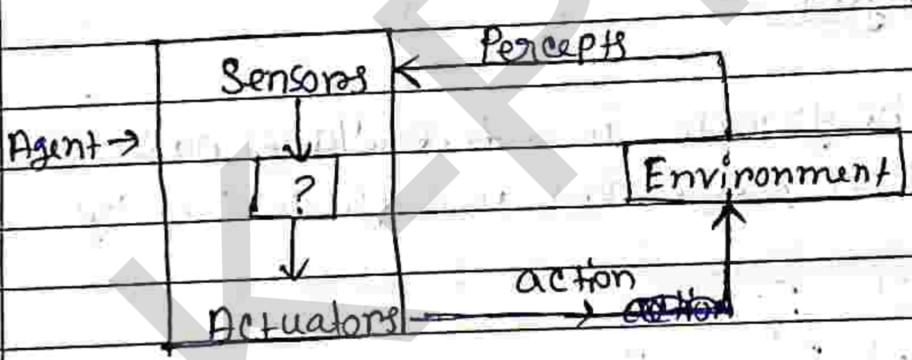
Agents and environment:-

An agent is anything that can be viewed as preceiving its environment through sensors and acting upon that envn't through actuators.

Human Agents:- eyes, ears → Sensors
hand, leg, mouth → actuators.

Robotic Agents:- Cameras, infrared range finders → S
motors → A

Software Agents:- file, network packets, Keystrokes → S
Screen, Sending massage files etc → A



Rational Agents:-

A rational agent could be anything that makes dicisions, such as a person, firm, machine or software. It carries out an action with the best outcome after considering past and current percept.

Requirements for a Rational agent :-

~~Rationality~~ :- 1. Rationality
2. Information gathering Capability
3. Ability to adupt.
4. An agent is autonomous if its behaviour is determined its own experience.

~~Types of Environments in AI :-~~

~~An environment in AI is the surrounding of the agent. The agent takes the input from the env^t through Sensors and delivers the output to the env^t through acutators. The different type of env^t are —~~

Intelligent Agent :-

Agents can be grouped into five classes based on their degree of preceived intelligence and capability :-

(I) Simple Reflex Agent
(II) Model-Based Reflex Agent.
(III) Goal Based Agents
(IV) Utility-Based Agents
(V) Learning Agent

__/__/____

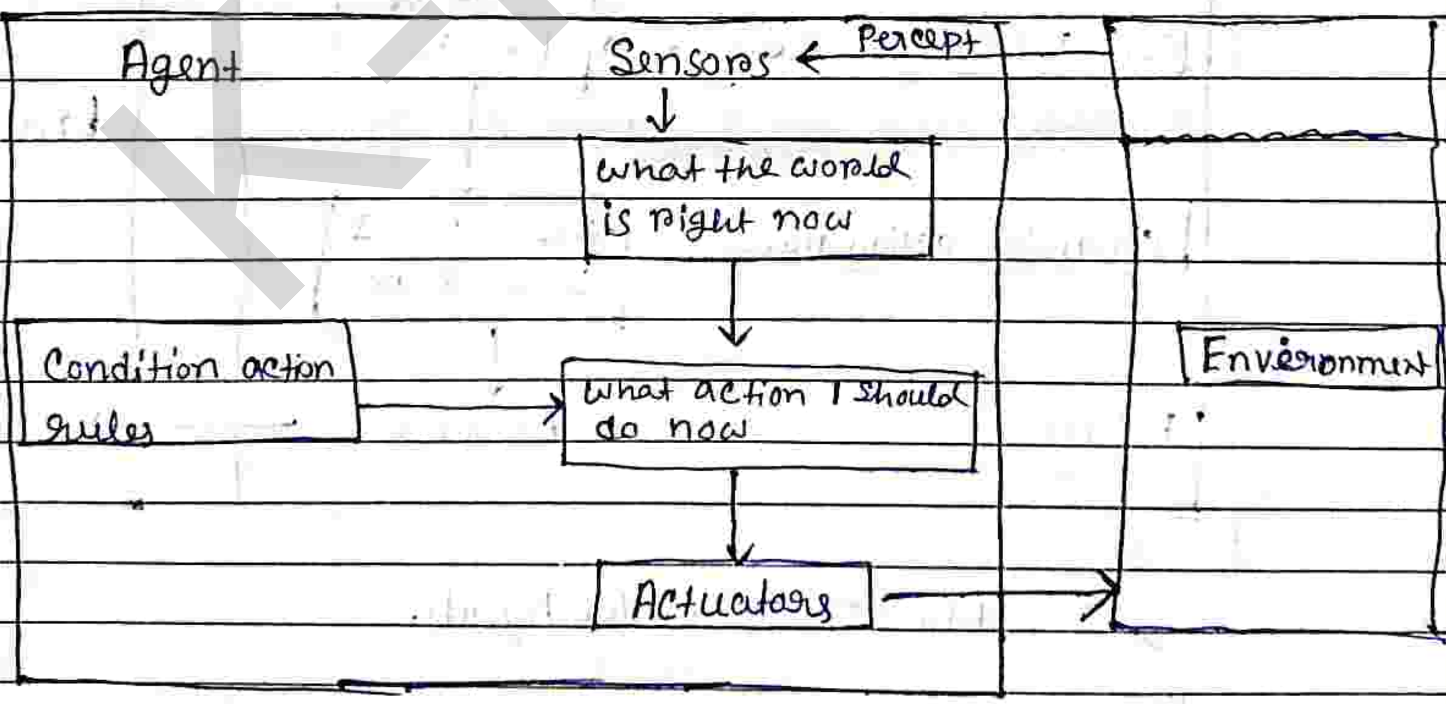
1. Simple Reflex Agents:-

The agent act only on the basis of current percept.
The agent function is based on Condition-action rule.
A condition-action rule is a rule that maps a State i.e... a a condition to an action.

if the condition is true
    action is taken
else
    no action.

Limitations:-

- Very limited intelligence.
- No knowledge about non-perceptual parts of the State.
- To big to generate and Store.
- Infinite loops are unavoidable.

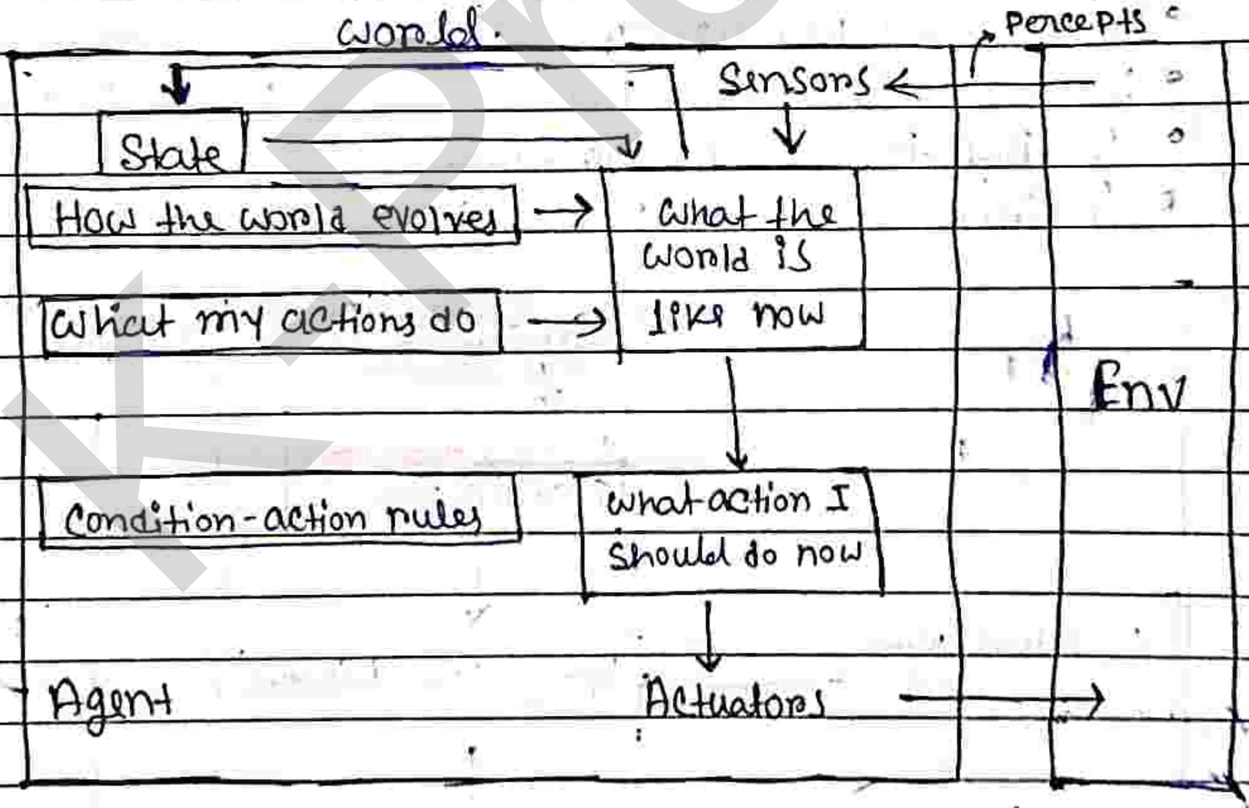


Simple Reflex Agents.

Eg:- A thermostat that turns on the heater when the temperature drops below a certain threshold ~~below~~ but doesn't consider previous temperature reading or long term weather forecasts.

2. Model Based Reflex agents:-

It works by finding a rule whose condition matches the current situation.
- Can handel partially observable env.
- updating the state requires information about-
  - How the world evolves independently from the agent.
  - How do the agent's actions affect the world.



Model-Based Reflex Agents.

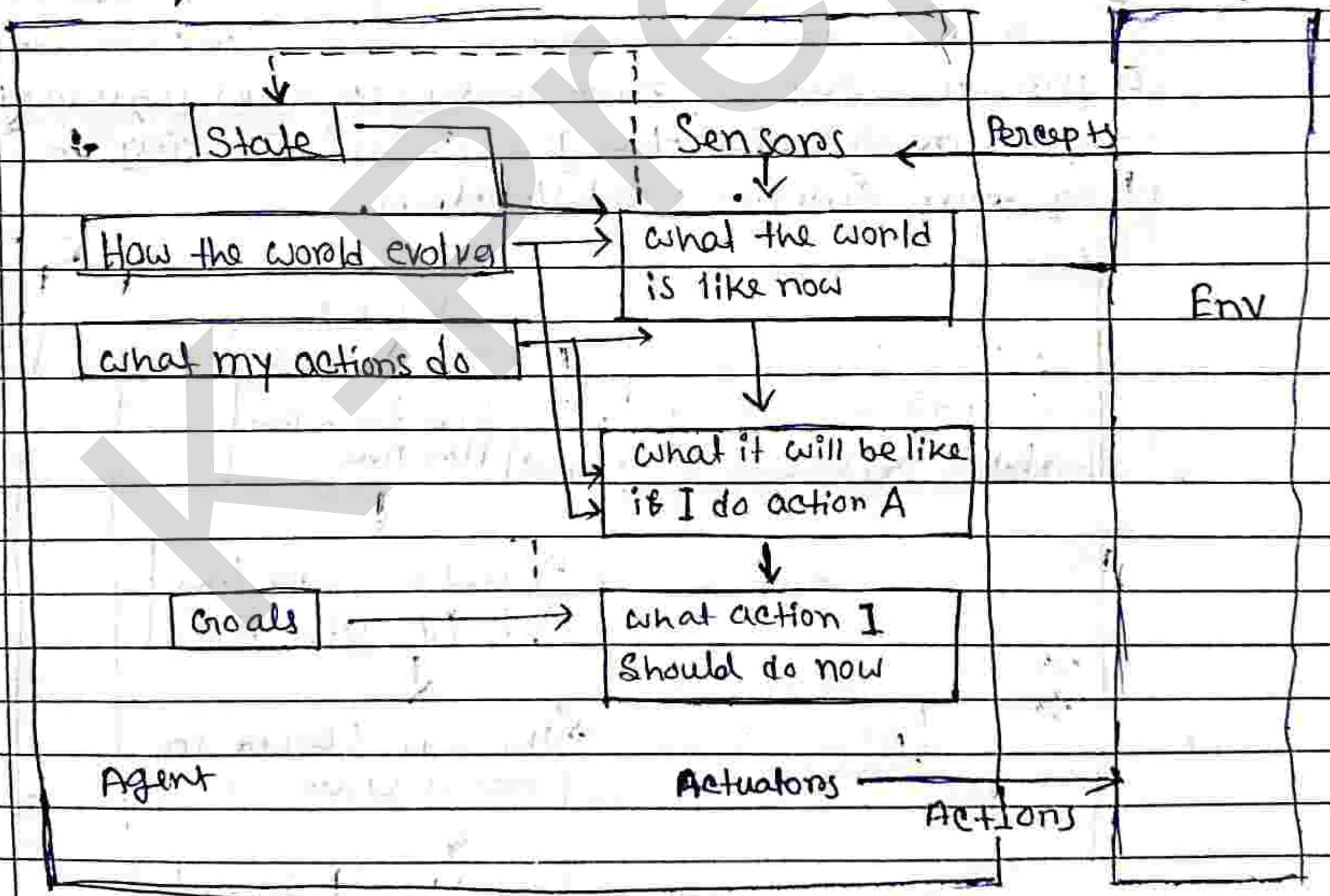
explain the diagram at your own lines.

__/__/__

eg: A Self-driving System not only responds to present road conditions but also takes into account its Knowledge of traffic rules, road maps, and past experience to negative Safety.

3. Goal-Based Agents :-

- Extended form of model based reflex agent.
- baised on Predefined objective or goals that they aim to achieve.
- They use Search and Planning Methods to create Sequence of action than enhance decision making.



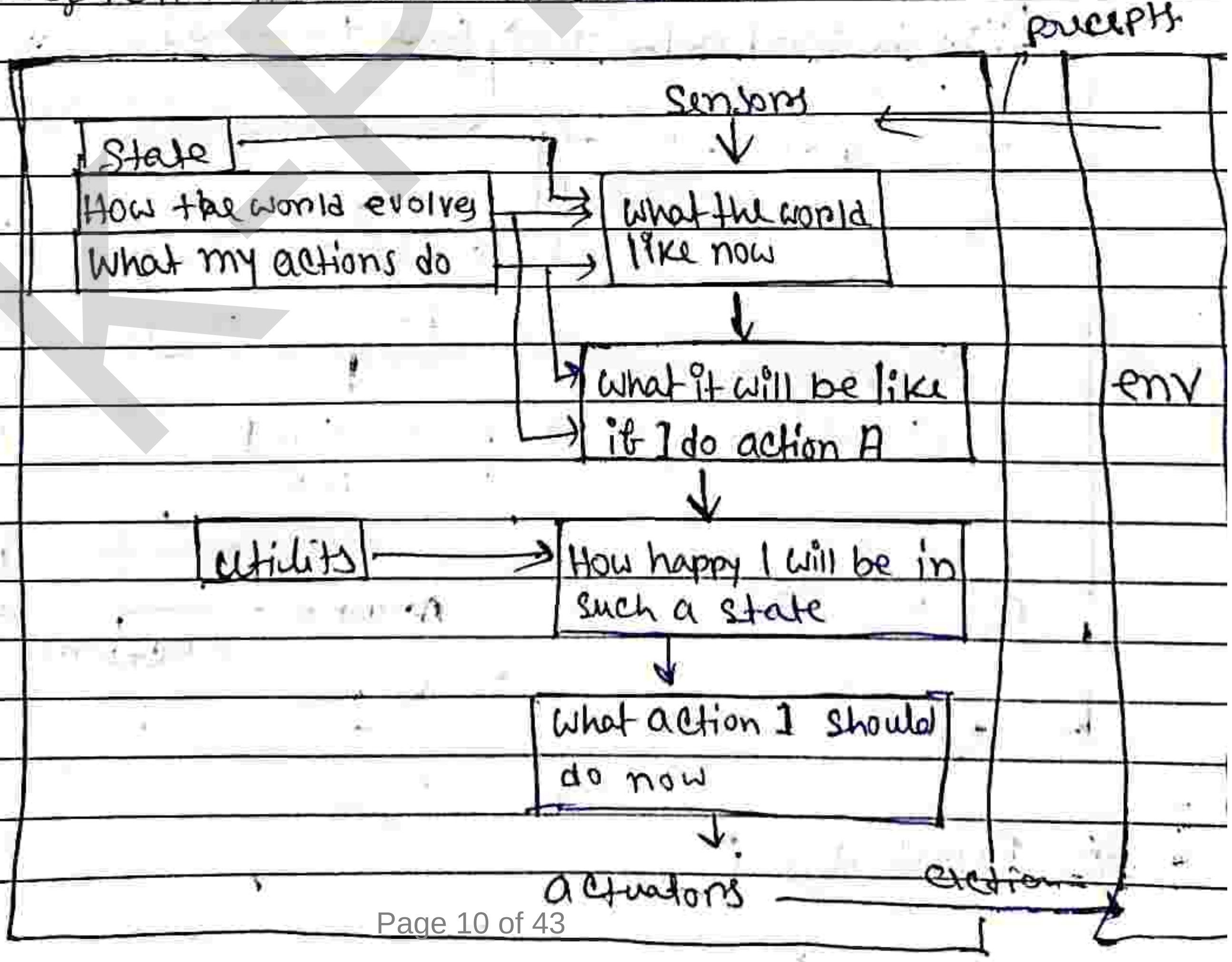
Goal Based Agents:-

Explain the diagram.

4. <u>Utility - Based Agents</u> :-

- Focus on ~~utility~~ utility not goal.
- utility based agents go ~~by~~ beyond basic goal-oriented methods by taking into account not only the accomplishment of goals, but also the quality of outcomes.
- They use utility functions to value various states, enabling detailed comparisons and trade-offs among different goals.

Eg:- An Investment advisor algorithm suggests investment options by considering factors such as a potential return, risk tolerance, and liquidity requirements, with the goal of maximizing the investors long-term financial satisfaction.



__/__/__

5. Learning Agents :-

— learning agents are a key idea in the field at AI, with the goal at developing systems that can improve their performance over time through experience.

→ These agents made up ~~of~~ at a few important parts —

1. learning element
2. Performance element
3. critic
4. Problem generator.

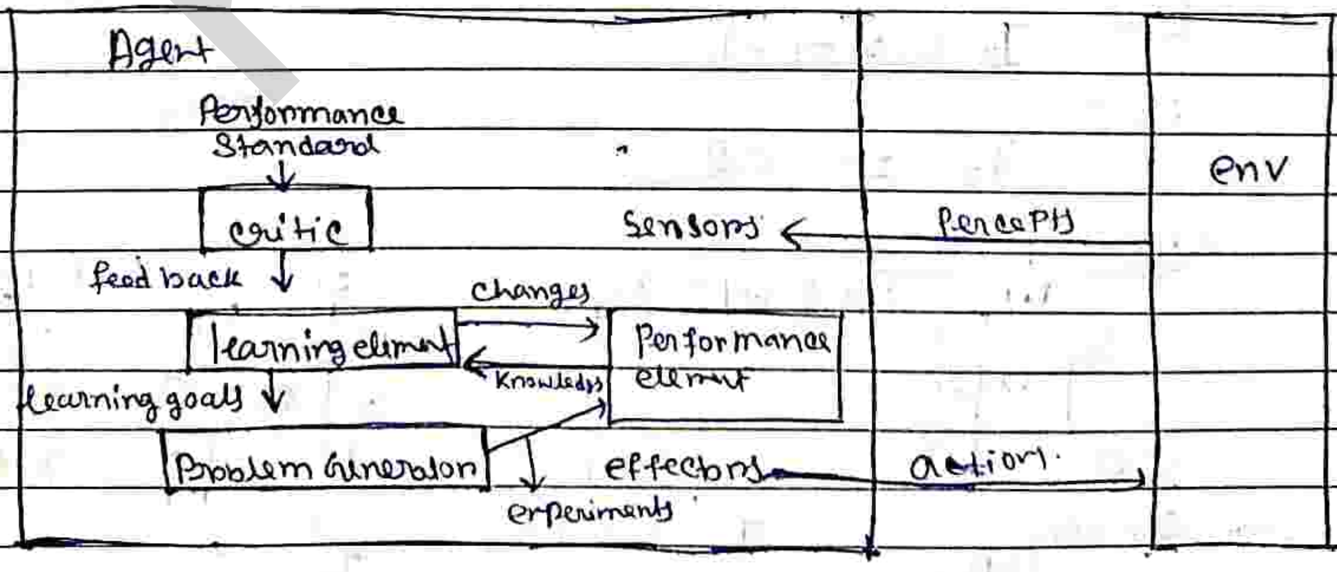
LE:- It is an element ~~action~~ which is responsible for making improvements by learning from the environment (when to do what)

PE:- It is responsible for selecting external action. (How to do)

PG:- generated problems. (carryout experiences)

Critic:- Try to reduce error. (from agent action)

Actuators:- it will act in-to the physical environment.



__/__/__

Tic Tac Toe Problem

let

O	O	X
X		O
		X

f_g = X's Possibility to win
O's Possibility to win.

$T_1 = 0 \rightarrow$ Draw
$T_2 = 1 \rightarrow$ win.

1st step only x term :-

X can be placed at 3 places

fig 1

O	O	X
X	X	O
		X

O	O	X
X		O
X		X

O	O	X
X		O
	X	X

fig 1 :-
Here T_2 = if you plot X to both vacant plots on fig 1 the possiblity ab win is 2
if you plot O at both vacant place on fig 1 then possibility ab win = 0.
So $T_2 = 2 - 0 = 2$

fig 2
$T_2 = 2 - 1 = 1$

fig 3
$T_2 = 2 - 0 = 2$

2nd Step O turn :-

1.1 fig 1 expanded

O	O	X
X	X	O
O		X

	O	

$T_2 = 0 - 0 = 0$
Draw.

$T_2 = 1 - 0 = 1$

2.1 fig 2 Expanded.

O	O	X
X	O	O
X		X

2.2

O	O	X
X		O
X	O	X

$T_2 = 1 - 1 = 0$
$T_2 = 1 - 1 = 0$

fig expended

3.1

O	O	X
X	O	O
	X	X

$T_2 = 1 - 0 = 1$

3.3

O	O	X
X		O
O	X	X

Draw

$T_2 = 0 - 0 = 0$

1,1 & 3.3 Draw

X turn

fig 1.2

O	O	X
X	X	O
X	O	X

$T_2 = 1$

fig 2.1

O	O	X
X	O	O
X	X	X

$T_2 = 1$

fig 2.2

O	O	X
X	X	O
X	O	X

$T_2 = 1$

fig 3.1

O	O	X
X	O	O
X	X	X

$T_2 = 1$

Now we have to field X possibility
(we wait to win only on intention to draw)
So it will backtrack by observe the current moves of X

thus fig 2 left bottom corner is the best Plan for X to win.

* Water Jug Problem

Q You are given two Jugs, a 4-liter one and a 3-liter one, you have unlimited water to fill the Jugs.
~~Jugs~~ Neither Jugs has any measuring marks on it.
Q How can you exactely measure 2 liters of water in the 4-lit Jug.?

→ Here is two variables,
X and Y

X represent the amount of water in the 4-lit Jug
Y represent " " " " " 3-lit Jug.

$x \leq X \leq 4$
$0 \leq Y \leq 3$

Initial state (0,0)
goal state (2,y) y may be $0 \leq Y \leq 3$

$X = 0$ $Y = 0$
$X = 0$ $Y = 3$
$X = 3$ $Y = 0$
$X = 3$ $Y = 3$
$X = 4$ $\boxed{Y = 2}$
$X = 0$ $Y = 2$
$X = 2$ $Y = 0$

Operation

1. fill 4 lit Jug — (X, Y) — $(4, y)$

   X must be less than 4 so we make it 4

2. fill 3 lit Jug — (X, y) — $(X, 3)$

   y must be less than 3

3. Empty 4 lit Jug — (x, y) — $(0, y)$

   x must be > 0 to make it empty

4. Empty 3 lit Jug — (x, y) — $(x, 0)$

   y must be > 0 make it empty

5. add water from 3 lit Jug to fill 4 lit Jug until 4 lit Jug is full — (x, y) — $(4, (y - (4 - x)))$

   $0 < X + Y \geq 4$ and $4 > 0$

6. add water from 4 to 3 until 3 is full — (x, y) — $(X - (3 - Y), 3)$

   $x > 0$ and $X + Y \geq 3$

7. add all water from 3 to 4 — (x, y) — $(x, y, 0)$

   $x + y \leq 0$ and $Y > 0$

8. add all water from 4 to 3 — (x, y) — $(0, X + y)$

   $X > 0$ add $X + Y \leq 0$

Q

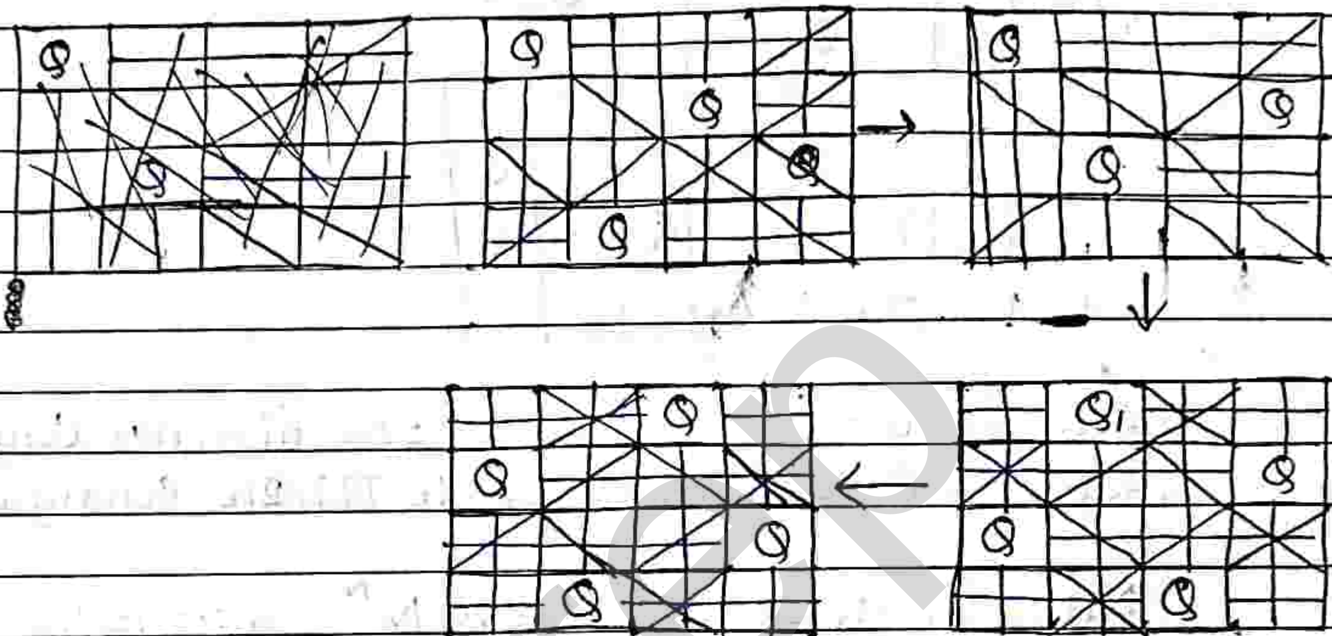
A	B	C
8	5	3

Initial 8 0 0   goal 4 4 0

A	B	C
8	0	0
3	5	0
3	2	3
6	2	0
6	0	2
1	5	2
1	4	3
4	4	0

* N - Queen Problem

4-Queen Problem:-



Dont do like this. Just write the Q only.

8- Queen Problem:-

Q							
				Q			
	Q						
					Q		
		Q					
						Q	
			Q				
							Q

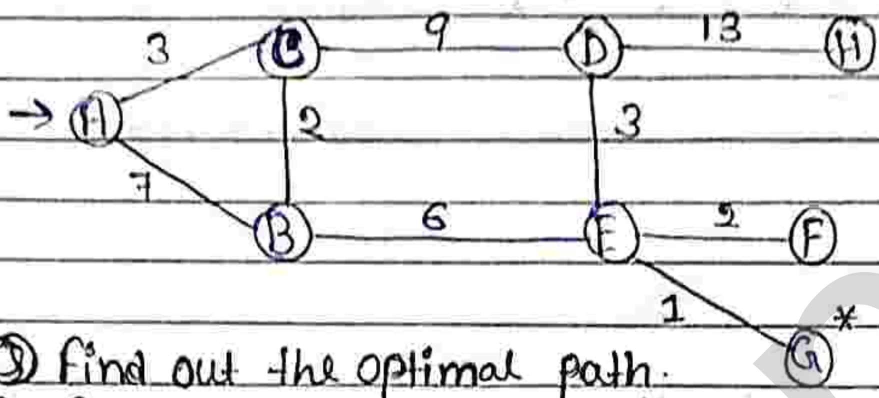
	Q		X				
				Q	X		
						Q	X
	X		Q				
Q						X	
X							Q
		X			Q		
		Q		X			

for 4-Queen, there are 256 distinct Configuration.
for 8-Queen, there are 16,777,216 Configuration.

NB:- In general, we have N^N Configeuration for N-Queen
$O(N!)$ Time complexity.

UCS (Uniform Crossed Search)

1. use priority Queue Data Structure
2. use Concept of Back Tracking.



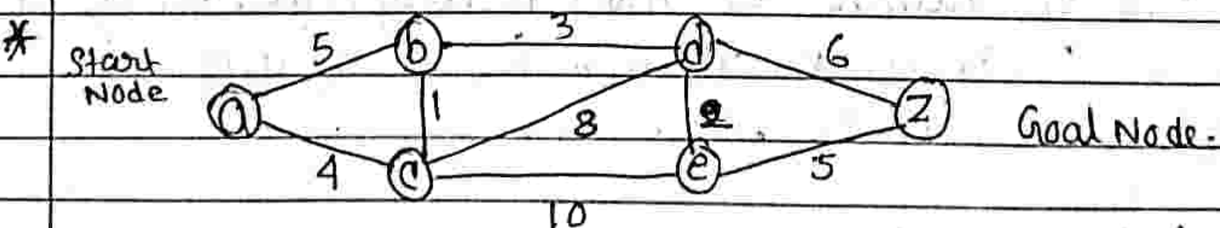
Q) Find out the optimal path.
Q) Goal Node H*

(1) A → C → B → E → G
3 + 2 + 6 + 1
= 12

Goal node Changed to H

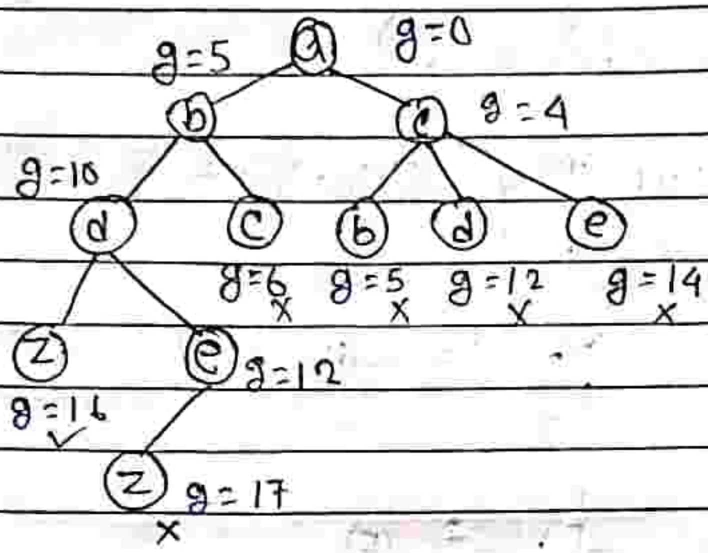
A → C → D → H
3 + 9 + 13 = 25

A → C → B → E → G → E → F → E
Back tracking.



a → c → b → d → z
4 + 1 + 5 + 6 = 16

_/_/_



Analysis:-

Complete : yes (getting the Solution)

Problem : May Stuck within ∞ loop.

Optinium reasult : yes.

$TC = O(b^{1+\text{ceiling}(C^*/\varepsilon)})$

Depth Limited Search (DLS)

Demerits of DFS :-

- If the number of nodes is very high, the agent will get stuck in an infinite loop going deeper in a single direction (for DFS only)

- To avoid the above problem, we apply a limit of searching in one direction or in a particular depth.

- Working principle is same with DFS, but addition is a predetermined limit.

Termination Condition :-

I. The Solution is found OR There is no Solution
II. Terminate on reaching Predetermined depth. This Condition is K/a cut of failure.

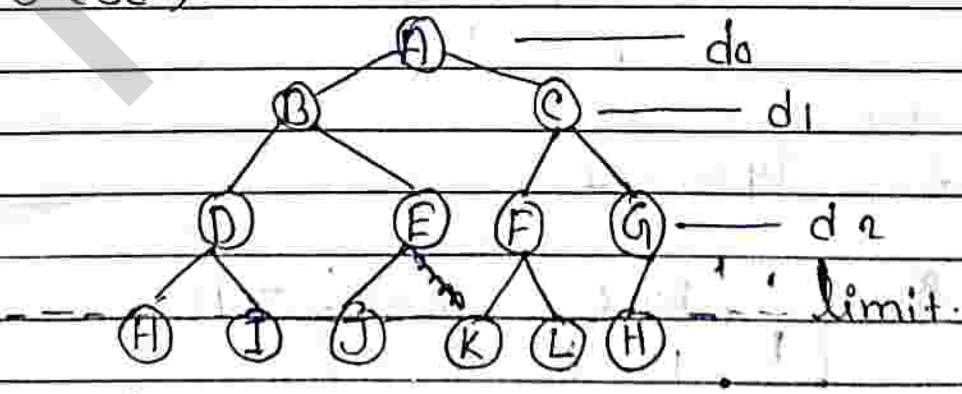
Advantage :-

1. Memory efficient
2. Improved Performance
3. Resource Optimization
4. Prevention of Infinite Loops.
5. Goal-Oriented Search.

Disadvantage :-

1. Incomplete & Not complete → Can be terminated w/o finding a Solution.
   ↓
   May not give the least result.

TC : $O(b^d)$
SC : $O(bd)$



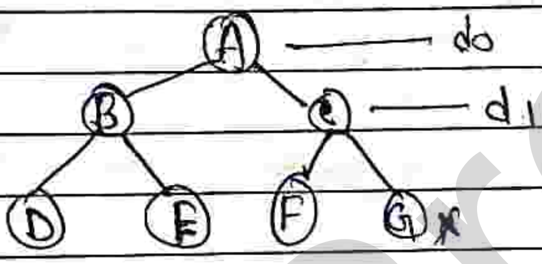
Predetermined Depth = 2
USE DFS → Stack
→ LIFO

Iterative Deepening Search :-

(I) Combination of BFS & DFS
(II) Best Depth limit is found out by gradual increase in limit. (in each iteration, increase by 1)

Advantages:- Utilizes adv of BFS & DLS.

Disadvantages:- Repeates the previous Stages everything
Working principle is LIFO



Step 1:- depth set to 0

A

stack → check whether (A) is goal is state or not → No

Stop ↑ yes

increment depth limit += 1
∴ depth = 1

Step 2:- depth = 1

~~A~~

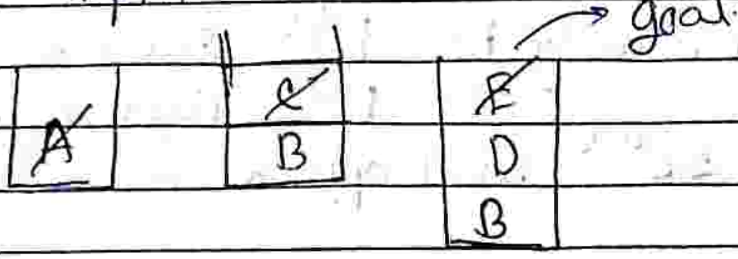
~~C~~
~~B~~

Pop C → goal ? → NO
Pop B → goal ? → NO

Depth lime += 1.

Step 3:-

depth = 2



Complete → yes
optimal → yes.

$TC = O(b^d)$
$SC = O(bd)$

Bi-Directional Search:-

Involves two searches that are executed simultaneously

- Forward Search (Start to Goal)
- Backward Search (Goal to Search)

Demerits:-
Start and goal both start.

- Single Search graph is traversed.
- any Searching technique can be applied that is BFS, DFS
- Stopping Condition:- The Search terminates when the two Search graphs intersect or meet.

Advantage:-
(I) Consumes less memory)
(II) Faster than traditional single-direction Searches.

Disadvantage:-
(I) Implementation can be more Complex
(II) The goal State must be known in advance.

__/__/__

4- Character measure:—

1. Time Complexity i.e. $b^{d/2}$ Start to intercept
   $b^{d/2}$ (Goal to interupt)

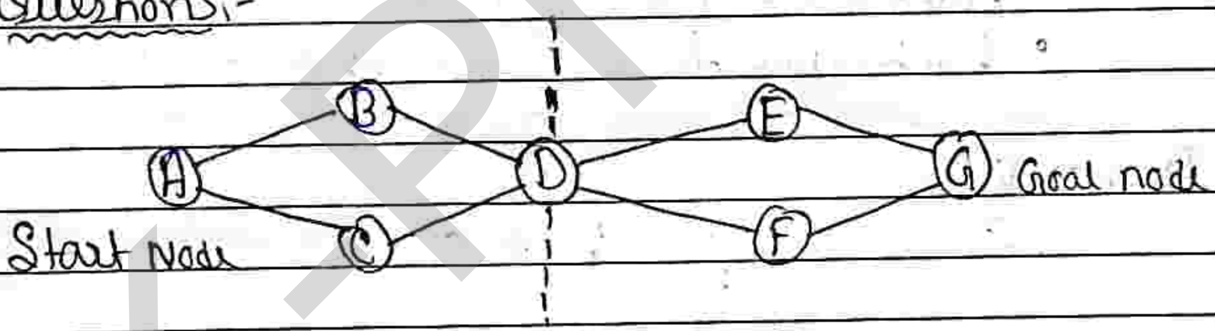
$b^{d/2} + b^{d/2} = 2b^{d/2}$ most approx.

II. Space Complexity $= b^d$

III. Optimality = yes if BFS
No if using DFS

(IV) Complete = Yes, if using BFS
No if using DFS

Questions:-



Solve Bidirectional Search, it is optimal so use BFS

ABCD Forward

~~A~~

~~B~~ C

~~C~~ D

D

⟵ Stop ⟶

Backward

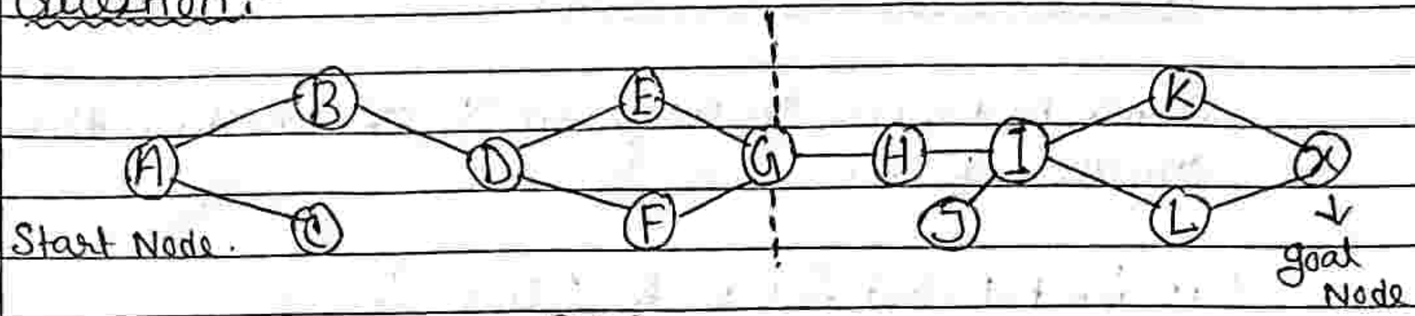
~~G~~

~~E~~ F

~~F~~ D

D

Question :-



BFS

Forward

~~A~~
~~B~~ C
~~C~~ D
D
~~E~~ F
~~F~~ G
G D

Backward.

~~X~~
~~K~~ L
~~L~~ I
~~I~~
H I
~~I~~ G
G

Page 198 - Slide 2nd & 3rd point Important Short Question.

___/___/___

Informed Search Strategy :-

Some heuristic value given, based on this find smallest path.

1. Best First Search & A star Search

BFS :

→ uses greedy technique.

→ It always selects the path which appears at the moment.

→ uses a heuristic function called $h(n)$, called as heuristic cost.

$h^*(n)$ → estimated cost.

→ Implemented by priority ~~queue~~ structure.

Algorithm (BFS)

I. Place the starting Node into the open list.

II. If the open list is empty stop and written fellow.

III. Remove the node n from the open list which has lowest value at $h(n)$ means heuristic cost and place it in the closed list.

IV. Expand the ~~node~~ noodle and generate succes of node n.

V. Check each success up at node n and find whether node is a goal node or not.

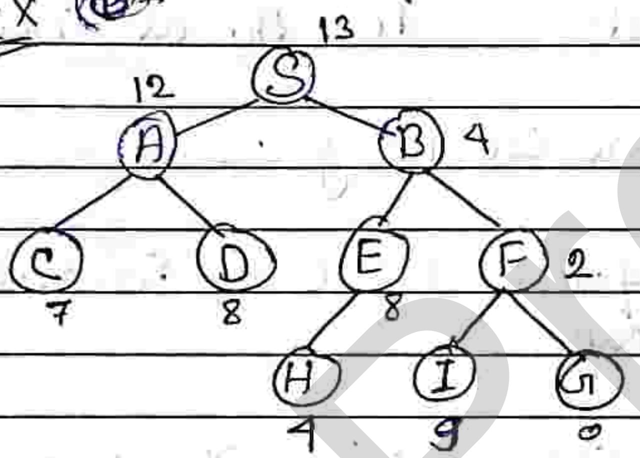
If any successor node is a goal node than return Success and terminate the Search.

(VI) for each successor node, algorithm checks for evalution in f(n). Check if the node hasbeen in either open or close list.

If the node hasn't been both list, then add it to the open list.

(VII) Return to Step 2.

Ex



Best first search

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

Solution -

Step 1 - Open [S] { }

Step 2 -

Open [A, B]	Close [S]
open [A]	Close [S, B]
open [E, F, A]	Close [S, B]
Open [E, A]	Close [S, B, F]
open [E, A, I, G]	Close [S, B, F]
Open [E, A, I]	Close [S, B, F, G]

g is goal node —

so path is - S → B → F → G

__/__/__

Explaination:-

→ Start from root node S and check successor of S i.e. A and B. priority of A=12, B=4

→ proceed through A & B,
So, S is closed ~~state~~ & A & B in open.

Time Complexity:-

$n.\log n$ (best) n = no of nodes.

- For the worst case we may search all the nodes.
- priority ~~heap~~ queue is using in mean heap or max heap

So insertion and deletion operations take $O(\log n)$ ~~time~~

- Such operations carried out for n times.

Space Complexity:- $O(b^d)$

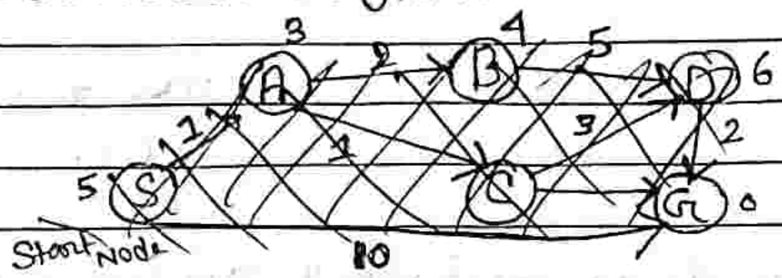
Complete → NO
Optimal → NO

Advantages:- Better than uninformed search which has exponential time complexity (b^d)

_/_/_

In Comparison to that BFS takes polynomial time complexity (if choosen better furistics),
if Choosen bad heuristics, it also shows exponential time.

A* Search Algorithm :- ***



State	h(x)
S	5
A	3
B	4
C	2
D	6
G	0

→ It finds the sortest path through the search space using the huristic function, it uses h(n), cost to reach the node A from the start node that is g(n)

$$\boxed{f(n) = g(n) + h(n)}$$

g(n) = ~~8/1~~ Cost from Start node to a particular node
h(n) = huristic value associate a node.
f(n) = estimate cost of cheapest solution.

A* algorithm Steps :-

① Let the Starting node in open list.
② Check the open list, if the list is Empty, return falure or Stop.
③ Select the node from the open list which has the Smaller value evelution function, if node n is the goal node, return Success and stop.
④ Expand the Selected node n and Generate all of its Successors. Add the Selected node n to the closed list.

— for each successor nod n,

• Check whether n is already open or close list.

• If it is not, compute evalute function for n, and add n to the open list.

⑤ else if nod n is already in open or Close lists, then it should be attach to the back point which reflex the lowest g(n) Value

⑥ Return to Step ②.

Advantages:—

• Combines benefits of BFS and Dijkstra's algorithm.

Admissibility of A* A:—

① A* is efficient because it balances between g(n) and h(n)

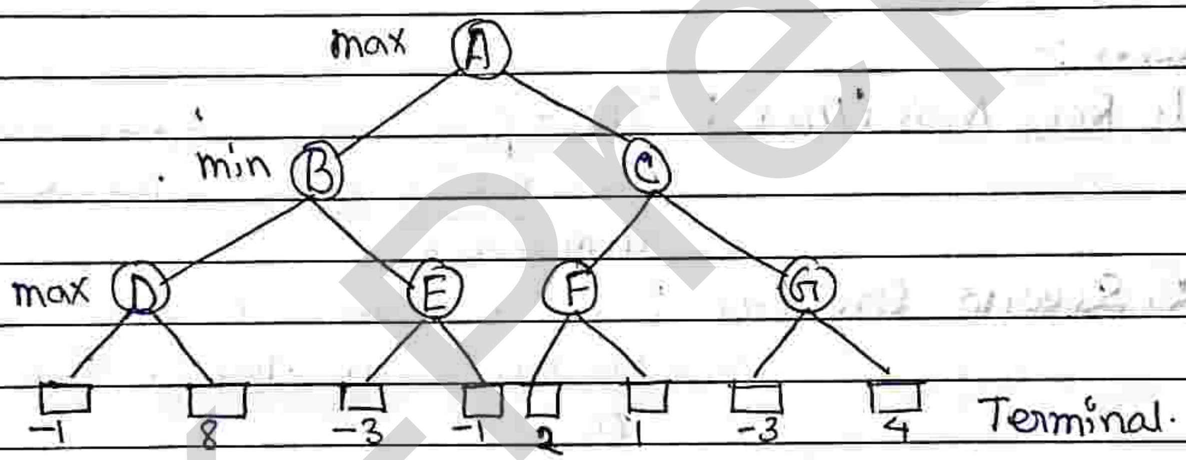
② Correctness depends on huristic function (h(n)) if, h(n) ≤ true cost, A* is optimal, otherwise it is not admissable.

Analysis

Time Complexity = $O(b^d)$
Space Complexity = $O(b^d)$
Completeness → Yes
optimal → yes optimal (if admissible)

Minmax - algorithm

1. Its a backtracking algorithm also
2. Best move Stratigy used.
3. Max will try to maximise the utility (best move)
4. min will try to minimize the utility (worst case)
5. Mainly used for game playing.
6. it is a recursive or back tracking algo, which is used in decision making and game theory.
7. It provides an optimal moves for the player, assuming that the opponent player also playing optimally.



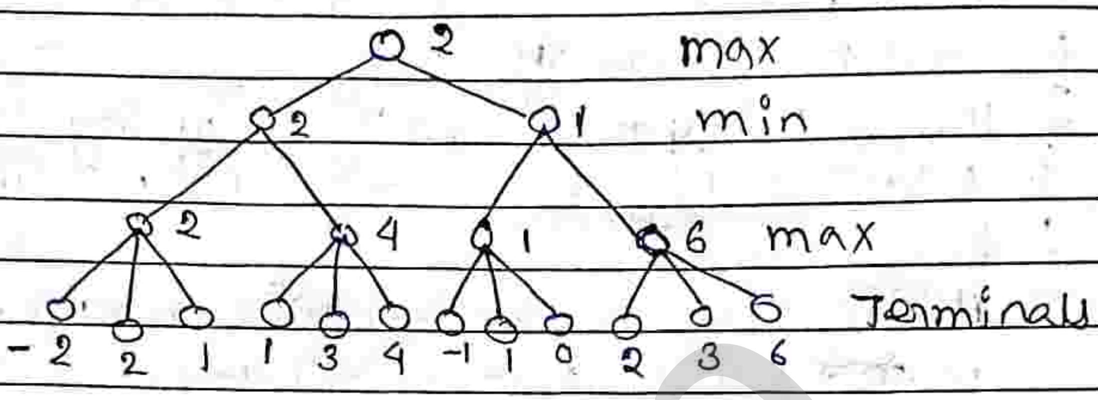
game tree.

Utility = 2

Time complexity = $O(b^d)$

- Min max —
- Two player game like Tic Tac Toe, Chess, checker.
- Before Starting the game what is the max and min value is initially.

Max = $-\infty$
Min = ∞



Understanding the game Test (above):-

levels:-

1. Root Node (max): The topmost level where the maximizing player makes the decision.
2. Second level (min): The minimizing players choose the minimum value among child nodes.
3. Leaf Nodes (Terminal values):- The final outcomes of the game, given at bottom level.

Steps of Solution:-

Step 1:- Comparision
-2 compare with $-\infty$, -2 is greater
then -2 compare with 2, 2 is greater
then 2 compare with 1, 2 is greater.

The first level (from bottom) we get 2, 4, 1, 6

_/_/_

Step-2

2 compare with ∞ , 2 is lower
then 2 with 4, 2 is lower.

likewise we got 2, 1.

Step-3

2 ans.

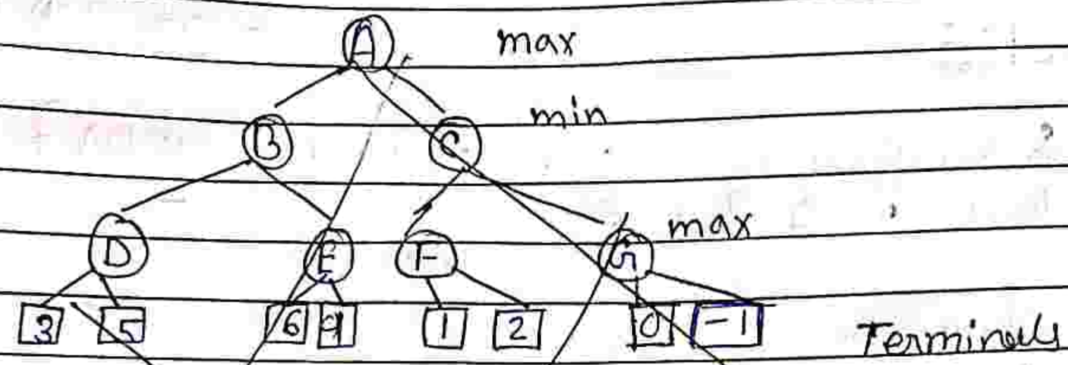
α-β Pruning:-

In minmax algorithm the no of states to be examined increases exponentially with the depth of the tree. We can reduce it to half by a technique called Pruning ~~tes~~ which eliminates nodes without checking.

It has two threshold parameters α and β

→ α is the best choice or largest value we have found so far. its initial value is −∞
→ β is the best choice or the lowest value we have found so far. its inital value is +∞

PTO



- The initial call starts from A. The value at α is $-\infty$ and the value at β is $+\infty$. At A the maximizer must choose max at B and C, so A calls B first.

- At B it the minimizer must choose min at D and E

Rules

Max updates α and min updates β.

It removes all the nodes which are not affecting the final decision by a process called pruning and the cond^n for pruning is $\alpha \geq \beta$.

The only value pass the α, β values to the child nodes using DFS.

While backtracking the node values are passed to the upper nodes.

__/__/__

Simulated Annealing:—

1. Variation of Hill climbing.
   - Here downhill moves may be possible.
2. The term objective function is used in place of huristic function.
3. Our attempt ~~to~~ is to maximize the objective function.
4. Anneling is ~~to~~ the process in which physical Substances such as metals are melted and then gradually Cooled until reach the Stable State.
5. The probablity that the metal will Jump to a higher energy given by.

$$P = e^{-\Delta E/KT}$$

where K = Boltzmann's constant
ΔE = Change in the Value of the objective function.
T = Temperature.

Revised formula

$$P = e^{-\Delta E/T}$$

Difference b/w Simple hill climbing and Simulated Anneling:-

1. The anneling Schedule must be maintained
2. Moves to worse State are Sometimes accepted.
3. It is good idea to maintain, in addiction to the current State, the best State solution So far.

/ /

Algorithm:-

1. Evaluate the initial State
   - if it is the goal State, return it as the output an stop
   - Otherwise Continuous with the initial State as the current State.

2. Initialize BEST-SO-FAR to current State.
3. Initialize T (temperature) according to the anniling Schedule.

4. loop until a Solution is found or until there are no new operator left to be applied in the current State.

   (a) Select an operator that has not yet been applied to the current State and apply it to produce a new State.

   (b) Evalute the new State, compute the Change in the objective function -

$$\Delta E = (\text{value at current State}) - (\text{value at new State})$$

- If new state is a goal State then return and quit.

- If it is not a goal State but is better than the current state, than make it the current state also update BEST-SO-FAR to this new State.

- If new State is not better than current State, then make the current State with probablity p as defined.

(c) Revise T as necessary according to the anneling schedule.

5. Return, BEST-SO-FAR as the answer.

Genetic Algorithms :-

(i) Genetic Algorithm begins with a set of randomly generated states called population.

(ii) Each state or individual is represented as a finite string of 0's and 1's in most cases. However, they can have problem specific representation as well.

(i) 8-Queen problem by Genetic Algorithm :-

8×8 box

		Q3					
			Q4				
					Q6		
Q1							
	Q2			Q5			
						Q7	Q8

The initial population is given in the figure.

★ 1st Step →

→ Represent the individuals of a population.

→ Make an array of size 8 [top to bottom]

{6, 7, 2, 4, 7, 5, 8, 8} respective queen positions of the column-row

→ backwards = {3, 2, 7, 5, 2, 4, 1, 1} [bottom to top]

As, most of the Queens are resided at the bottom part of the 8x8 board
∴ we will work with the bottom to top array
the motto is to work with small numbers.

★ 2nd step → Generate other initial population of random arrangements of Queens on the boards.

Considered for our example, the following strings

A: { 3, 2, 7, 5, 2, 4, 1, 1 }

B: { 3, 2, 5, 4, 3, 2, 1, 3 }   Assuming B, C, D

C: { 2, 4, 7, 4, 8, 5, 5, 2 }

D: { 2, 4, 4, 1, 5, 1, 2, 4 }

★ 3rd step :- Apply fitness function which gives the probability of being chosen to reproduce the offspring.

In the 8-Queen problem, the fitness fun^n is given by the no. of non-attacking pairs of Queens.

Fitness Score	6 +	5 +	4 +	3 +	3 +	2
	Q_1-Q_3	Q_2-Q_3	Q_3-Q_4	Q_4-Q_5	Q_5-Q_6	Q_6-Q_7
	Q_1-Q_4	Q_2-Q_4	Q_3-Q_5	Q_4-Q_6	Q_5-Q_7	Q_6-Q_8
	Q_1-Q_5	Q_2-Q_6	Q_3-Q_7	Q_4-Q_7	Q_5-Q_8	
	Q_1-Q_6	Q_2-Q_7	Q_3-Q_8		+0 +0	
	Q_1-Q_7	Q_2-Q_8				
	Q_1-Q_8				= 23	

☆ Max^m fitness score any population can have is 28 [7+6+5+4+3+2+1+0]

Consider the fitness scores for string B to be 11 for C to be 24 and for D it is 20.

String	A	B	C	D	
Initial value	32752411	32543213	24748552	24415124	
Fscore	23	11	24	20	78
F%	$\frac{23}{78}\times100$	$\frac{11}{78}\times100$	$\frac{24}{78}\times100$	$\frac{20}{78}\times100$	
	= 29.48 %	14.10 %	30.76 %	25.64 %	

fitness func^n only

☆ 4th Step → ~~[illegible]~~ Select two highest fitness score strings and make pairs — Selection.

A – C , B – D

☆ 5th Step → perform Crossover for each pair by randomly choosing a crossover point.

$\boxed{A}$: 327|52411
$\boxed{C}$: 247|48552

Let the crossover point be 3 interchange the part after 3rd.

A: 3 2 7 4 8 5 5 2 → E
C: 2 4 7 5 2 4 1 1 → F

E & F are offsprings

B: 3 2 5|4 3 2 1 3
D: 2 4 4|1 5 1 2 4

H/W:- Four Queen Problem by Hill climbing [4×4]

Q_1	Q_2		
			Q_4
		Q_3	

Objective funᶜ. is given by no. of non-attacking pairs.

$$\text{Obj. fun}^c = \underset{\substack{Q_1-Q_3 \\ Q_1-Q_4}}{2} + \underset{Q_2-Q_3}{1} + \underset{Q_3}{0} + 0 = 3$$

Randomly move any of the queens to any possible posⁿ in their respective columns.

	Q_2		
Q_1			
			Q_4
		Q_3	

Goal value = 6

$$\text{Obj. func} \rightarrow \underset{Q_1-Q_4}{1} + \underset{Q_2-Q_3}{1} + 0 + 0$$

$$= 2$$

For 4-Queen by Hill climbing is not a good approach.

	Q_2		
Q_1			Q_4
		Q_3	

$\rightarrow 3$

	Q_2		Q_4
Q_1			
		Q_3	

$= 3+1+1+0$

$= 5$

	Q_2		
			Q_4
Q_1			
		Q_3	

$= 3+2+1+0 = 6$

→ It is not Complete.
→ It is not optimal.