Artificial Intelligence

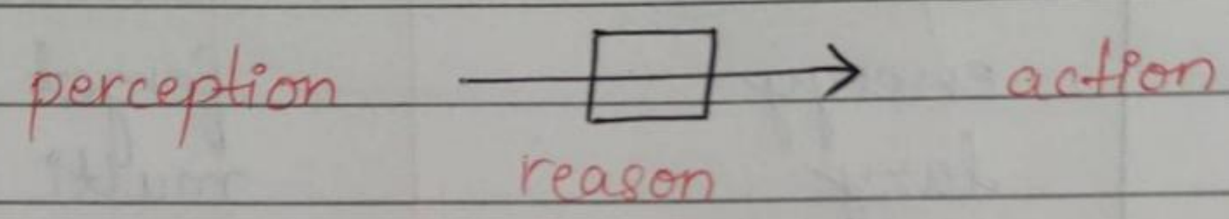
Date
Page

	Features of Human Being		Features of Computer
	Decision	wisdom	Accuracy
	Sense	aging factor	Speed
	emotion	growth	dependent
	consiouness	experience	limitation
	creativity	energy	forget things
	action	lazy	multi taskings
data	vision	freedom	assistant
↓	thinking person	language-understand	discriminate
info	imagination		distraction
↓	reasoning	relationship	problem-solving
know.	knowledge		
↓	feelings		
wisdom	interaction		
intelligent	communication		

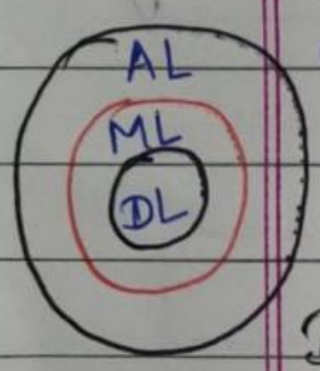
Scanned by TapScanner

Def^n: AI is the study of making computers to do things at present which human being are better.

Def^n: AI is the study of computations that makes it possible to Percive, Reason and Act.



Def^n: AI is the area of computer science, concerned with designing intelligent computer system i.e. systems exibits the characteristic we normally associate with intelligence in human being.



Def^n: AI is the branch of comp. sci. dealing with symbolic, non-algorithmic, methods of problem-solving.

Scanned by TapScanner

History of AI:

Date
Page

1614 → Scot mathematician.
→ created a intelligent machine.
→ This machine do basic calculation like addition, subtraction with overflow.

1620 → Francis Bacon
→ He developed / found search technic for extracting features from entities.

1642 → French mathematician and philosper
[more better version of 1614] → C Blaise Pascal.
→ It was made modified version of 1614 which can do all basic calculation with carries and borrows.

1694 → Mathematician Leibnitz
→ made Leib wheel it perform all complex mathematical calculations.

1642-1936 → Charlie's Analytical engine.

1834 → Tom Stendage
→ wrote reasearch paper on thinking machine.

1840 → Joseph Faber
→ first talking machine named Euphonia.

1921 → C Zech Author - Karel Capek
→ R.U.R [Rossum's Universal Robots]
→ It was a Science Fiction Play.

Scanned by TapScanner

explain turing test ?

Date
Page

1950 → Year of Turing Test
→ Alan Turing

1957 → Neweel and Simon
→ They predicted that one day computers will beat human being in chess.

1997 → Deep Blue developed by IBM computer.
→ Play Chess
→ The match was of 6 match
- In first match was played in Philadephia and the score was 4 - 2 → deepblue
  ↓ Garry Kasparov
- In second match was played in New York and the score was 3 - 2 (1 draw)
  ↑ deepblue

Scanned by TapScanner

Applications of AI:

(i) Game Plane:

Reasons: (i) easy to understand.
(ii) well defined set of rules
(iii) structured in a way or nature of a game is not ambiguous (confusing).
(iv) no financial or ethical burden.
(v) no expert knowledge is required.
(vi) easy to generate state space. → set of all possible configuration of a system.

List of games:

- Tic-Tac-Toe
- Chess
- Two Water Jug Problem
- Three Water Jug Problem
- Tower of Hanoi
- Crypt-Arimatic Problem.
- Travelling Sales man Problem
- 8-puzzle problem
- Farmers, Fox, goat, grass.
- 8-queen problem.
- Monkey-Banana Problems

(ii) Automated Reasoning and Theorem Proving:

- It is a subfield of automated reasoning and mathematical logic dealing with proving mathematical theorem by computer programs.

Scanned by TapScanner

Date
Page

(iii) Natural language processing (NLP):

1) Understanding (NLU) → machine understand human language.
→ subtype of NLP.

2) Translation → the language convert 1 medium to 2 medium

3) Generation (NLG) → subtype of NLP.

(iv) Robotics:

- A Remotely operated vehicle (ROV) is an unoccupied underwater robot that is connected to a ship by a series of cables.
  → cabels transmit command and control signals between the operator and the ROV.
- Fully Autonomous is a robot that acts without resource to human control. eg. Roomba
- Semi Autonomous is a term for automation that can make decisions and perform action without direction, i.e. they need human in certain situation.
- Three types Robots: Land, Water, Ariel.

Scanned by TapScanner

Date
Page

(v) Expert System:

it should evolve, experience

- Expert System has knowledge base and ~~then~~ after match they take decision/inference.

should increase

- Specific and domain knowledge system should open to learn.

- Knowledge base
  ↓
  Inference engine/rules of engine
  ↓
  User Input or inference

example of expert system:

MYCIN
- It take sample of a blood and finds the bacteria that causes the disease in the body.

DENDRAL
- It is chemical analysis expert system used in organic chemistry to detect unknown organic molecules

PROSPECTOR
- It was used by zoologist to find minerals at particular area by taking samples.

Scanned by TapScanner

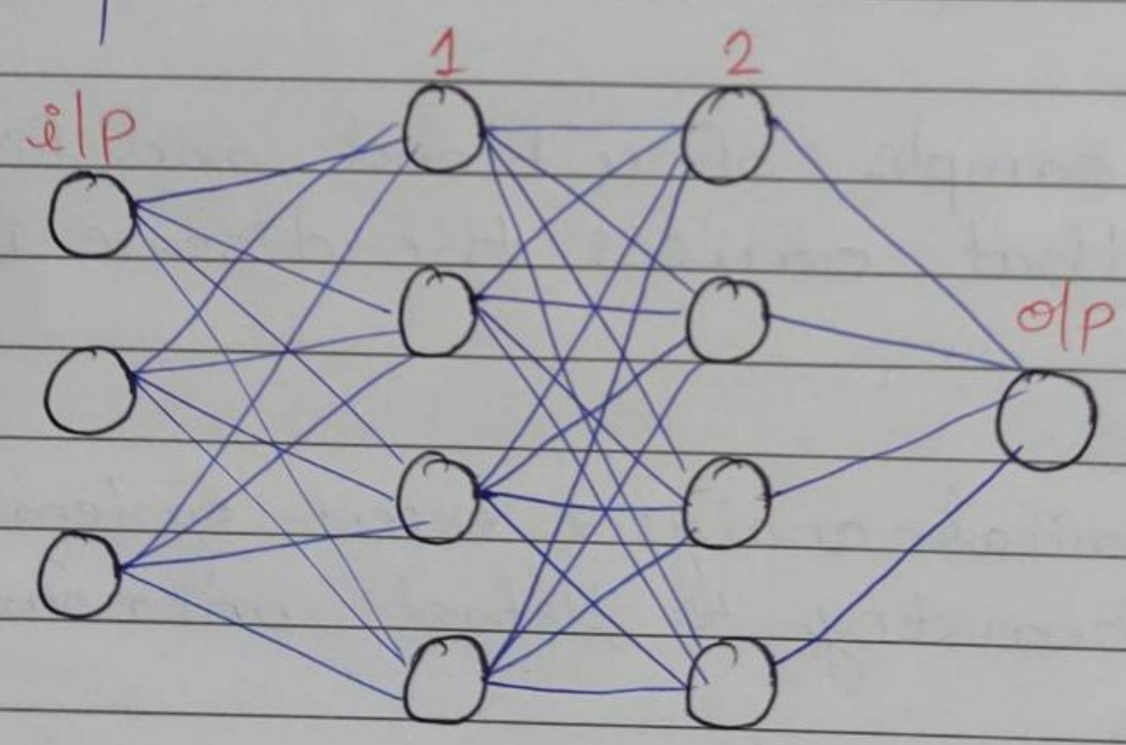
(vi) Neural Network: (Part of deep learning)

- Neural Network is also known as artificial neural networks (ANNs)

- ANN is based on a collection of connected units or nodes called artificial neurons, which loosely model the neurons in a biological brain.

- Biological Neural Network (BNN) is a network of neurons that are connected together by axons and dendrites.

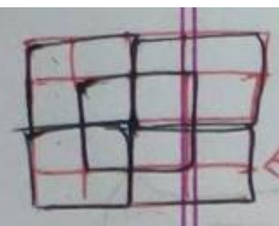
- The connections between neurons are made by synapses.



at least one and output or upto n layer

Def^n: AI is branch of computer science that deals with ways of representating knowledge using symbols rather than numbers with rule of thumbs for processing information.

Scanned by TapScanner



pixels should be smaller than image & it will be odd e.g 3x3, 5x5 - -

Def^n: AI is the study of developing algorithms enables by constraints imposed by representation that support models targeted at thinking, perception and action.

Task domains of AI:

Repetative task

(1) Mundand task:

we want machine to perform all this task.

(i) Perception:
- speech
- vision

(ii) Natural language understanding:
- Translation
- Generation

(iii) Reasoning:
- Common sense
- Interation

Def^n: AI works on pattern matching method which attempt to describe objects, events or process in terms of their qualitative features and logical and computational relationship among them.

Scanned by TapScanner

Date
Page

(2) Formal task:

(i) Problem - Solving

(ii) Mathematical Prospective.

(3) Engineering task:

(i) designing intelligent system

e.g. ADAS in MG

↓

Advanced Driver Assistance System → 2nd level automation system (Partial Automation)

(ii) Fault-finding / Fault diagonist

e.g. autonomus car → it calculate wrong turn and dash the thing/person.

rectify the fault and solve the problem.

(iii) expert-analysis (prediction):

it is not limited. It require knowledge that many of us do not have.

e.g. in car → engine analysis, tyre analysis

Scanned by TapScanner

Date
Page

Characteristic of AI problems:

(1) It is Voluminous (Size).

(2) It is constantly changing.

(3) It is ~~hard~~ noteasy to characterise

(4) Its rules will change based on data.

(5) It is very dynamic.

(6) Choice of intelligence/knowledge.

Scanned by TapScanner

Date
Page

What is an AI technique?

AI technique is a method that exploits knowledge that should be represented in such a way:
(1) the knowledge captures generalization.
(2) It can be modified to correct errors and to adopt the changes in the environment.
e.g. Tic Tac Toe.

Characteristic of AI technique:

(1) It can be used in many situations even it is not totally accurate or correct

(2) It can be used to overcome its own sheer-bulk by narrowing the range of possibilities.

Scanned by TapScanner

AI Techniques:

Tic-Tae-Toe:

Strategy 1:

Data Structure:

(i) Board: A nine element vector representing the board where the elements of the vector correspond to the board position.

1	2	3
4	5	6
7	8	9

(ii) Notation: zero, 1, 2

zero → Blank, 1 → X, 2 → O

(iii) Move-Table:

① Index ② Current Board Position ③ New/Next Board Position.

Index	CBP	NBP
0	000 000 000	000010000
1	000 000 001	020000001
2	000 000 002	000100002

Scanned by TapScanner

Date
Page

Algorithm:

Step 1: View the vector board as a ternary number and convert it to a decimal no.

Step 2: Use the number computed in (Step 1) as an index into home table and excess the vector store them.

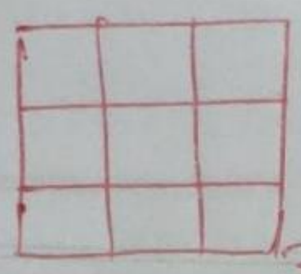
Step 3: The vector selected in a (step 2) represents the way the board will look after the move that should be made.

So, set the board equal to that vector.

Scanned by TapScanner

Advantages	DisAdvantages
- You can play all the moves	- requires lot of space.
- optimized move	- requires a lot of time to prepare move table.
- possibilities of update on move table	- It is not scaleable or inextensible.
- keep an eye on oppo-nent move	- Hard work is require to prepare such a move table.

Scanned by TapScanner



3x3

cells - 9

player - 1 → X

player - 2 → O

→ A player has 3 consecutive marks either it is hori, vertical, diagon

↓ 3 ↓ 3 ↓ 2

→ 2D board structure

1	2	3
4	5	6
7	8	9

→

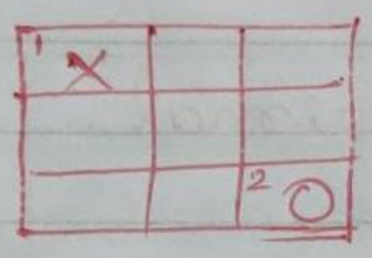
-	e	-
e	c	e
-	e	-

corner cells (-)

edge cells (e)

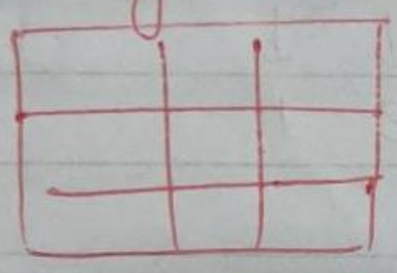
centre cells (c)

→ Algorithm

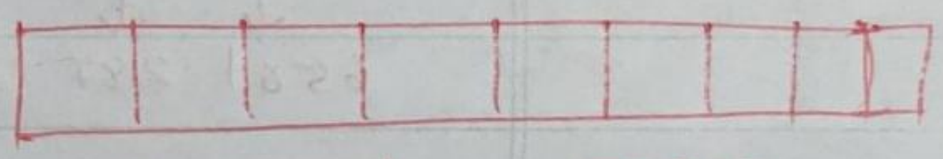


first move will be of X then O will take move and X will see the possibilities of O.

→ In system 2D convert in 1D i.e matrix to vector.



← IBP [Initial Board Position]



in vector IBP

→ goal state will be either X X X or O O O

→ IBP is represented by

0	0	0
0	0	0
0	0	0

zero.

→ If 5 position we will put X in 2D

1	2	3
4	5X	6
7	8	9

then

0	0	0	0	1	0	0	0	0

in 1D on 5 position it will 1.

→ 3^9 → 9 cells & 3 possibilites

↓

0, X, O

Scanned by TapScanner

(2) $A_2 \begin{bmatrix} 1 & 2 & 3 \\ 2 & 5 & 3 \\ 1 & 0 & 8 \end{bmatrix}$

$\cancel{[A \mid I]}$

ternary to decimal.

(i) $000\ 000\ 000_3$

$\downarrow$

$3^8\ 3^7\ 3^6\ 3^5\ 3^4\ 3^3\ 3^2\ 3^1\ 3^0$

$\downarrow\ \downarrow\ \downarrow\ \downarrow\ \downarrow\ \downarrow\ \downarrow\ \downarrow\ \downarrow$

6561 2187 729 243 81 27 9 3 1

$\downarrow$

0 0 0 0 0 0 0 0 0

$\downarrow\ \downarrow\ \downarrow\ \downarrow\ \downarrow\ \downarrow\ \downarrow\ \downarrow\ \downarrow$

6561 2187 729 243 81 27 9 3 1

$\downarrow$

000 000 000

$\downarrow$

summation

0_{10}

I	CBP	NBP
0	000000000	000010000

(ii) $000000001_3 \rightarrow$ Ans: 1_{10}

$15h + 7 - 25 = 0$

$15h - 18 = 0$

$15h = 18$

$h = \frac{\cancel{18}\ 3\times\cancel{6}}{\cancel{15}\ 3\times5}$

$\begin{array}{r} 25 \\ 7 \\ \hline 18 \end{array}$ 15

Scanned by TapScanner

Strategy - 2:

Data Structure:

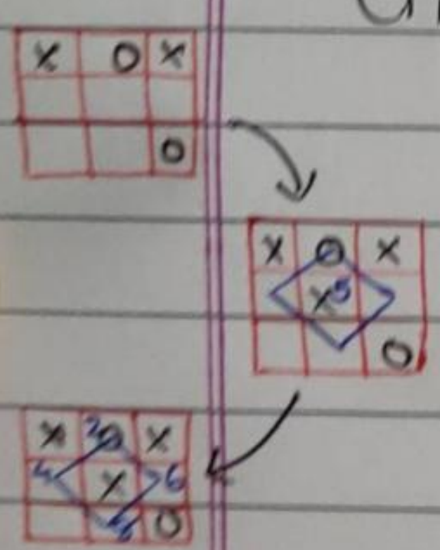
(i) Board: 9 elements board

1	2	3
4	5	6
7	8	9

(ii) Notation:

2 → blank
3 → X
5 → O

(iii) Make(5): (make move 5)



```
if
    board[5] = 2 [Blank]
        return 5
else
    return 2,4,6,8
```

we have to make possibility of best and winning move together...

we have to check nearby cell

(iv) Possw^m(P): (Possibilities)

```
if
    player p can't win in next move
    return 0
else
    return the tile to make next move i.e. winning move.
```

→ This fun^n will unable the program or player p both to win and to block the oponents wins.

Scanned by TapScanner

→ Here the fun^n will test an entire row, column and diagonal to see if it is a possible win by multiplying the values of its squares/tile together.

for example:

If the product is 8

i.e. $\underset{X}{3} \times \underset{X}{3} \times \underset{\text{Blank}}{2}$, X can win

If the product is 50

i.e. $\underset{O}{5} \times \underset{O}{5} \times \underset{\text{Blank}}{2}$, O can win.

for example:

X	O	X
X		O

if we put X on 7, then we have changes of win

Here, we can see we got one column and one diagonal which has product of 18 and now O has next move which will try to restrict changes of winning but still X is winning here.

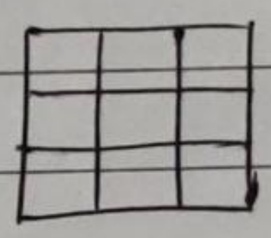
Scanned by TapScanner

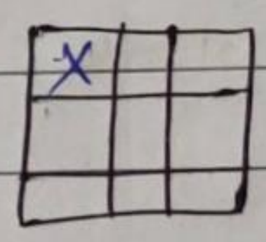
odd
nd even
umber
on board

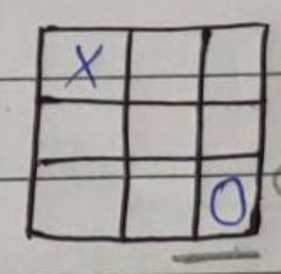
1	2	3
4	5	6
7	8	9

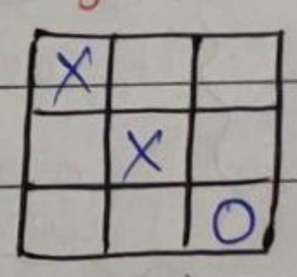
→ Go(n) = occupied tile number.

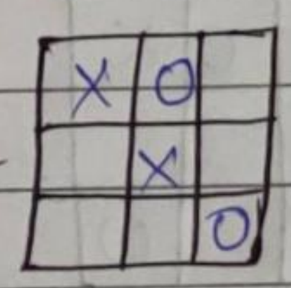
if turn is odd
Board[n] = 3 $\qquad$ 7(3) = X
turn = turn + 1.
else
turn is even $\qquad$ this even ~~date~~ represent what? [does it represent that O is 2nd move & that is even]
Board[n] = 5 $\qquad$ 4(5) = O
turn = turn + 1.

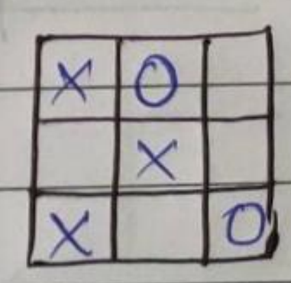
T_0 = 
zero mark already

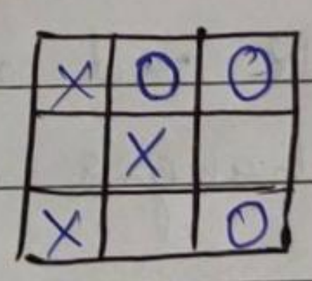
T_1 = 
one mark already

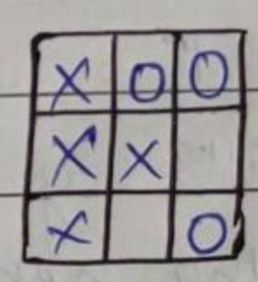
T_2 = 
why we have taken odd no. of cell

T_3 = 

T_4 = 

T_5 = 

T_6 = 

T_7 =  → Goal state

Scanned by TapScanner

Strategy : 2'

Data Structure :

(i) Board : Different board position and it is known as magic square.

8	3	4	15
1	5	9	15
6	7	2	15
15	15	15	15

(ii) Notation :

2 → Blank
3 → X
5 → O

(iii) Make (s) :

```
if
    board[s] = 2 (Blank)
    return s
else
    return 2,4,6,8
```

(iv) Posswin (P) :

```
if
    player p can't win in next move
    return 0
else
    return the tile to make next move
```

Scanned by TapScanner

→ To check for a position win for one player we consider each pair of squares own by that player and ~~complet~~ compute the difference between 15 and the sum of two squares. If this difference is negative ($<=0$) or if it is greater than 9 then the original two squares were not collinear and so be ignored.

<u>example:</u>

X^8		
	O	
	X^7	O

$15 - 15 = 0$

<u>example:</u>

		X
X	O	
		O

$15 - 5 = 10 > 9$

O	X	
		O
	X	

$15 - 11 = 4 < 9$

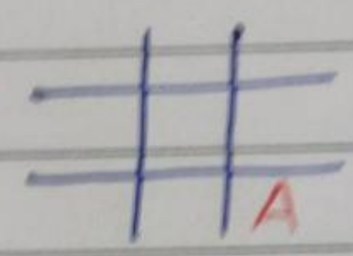
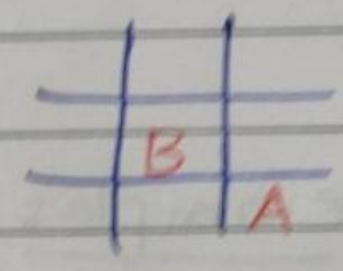
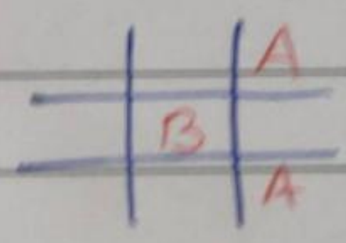
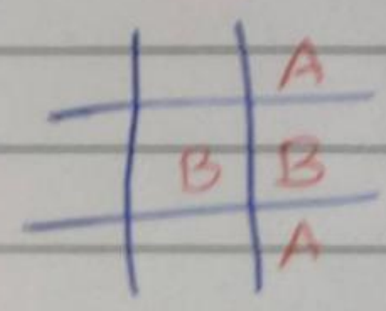
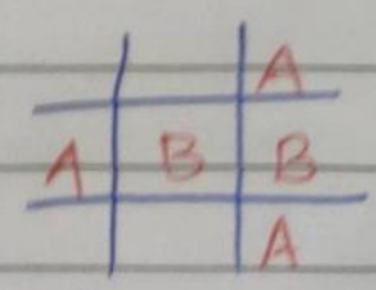
[if less than 9 then we will consider the move.]

Scanned by TapScanner

example:

2	7	6
9	5	1
4	3	8

solution:

A	B	Move	Board Position
8	-	Random Start	
-	5	15-8=7 ∴ 5,2 or 6,1 or 4,3	
6	-	A have 6,1 or 4,3	
-	1	A have 8+6=14 B have 15-14=1	
9	-	B have 5+1=96 A have 15-6=9	

So, the game is tie.

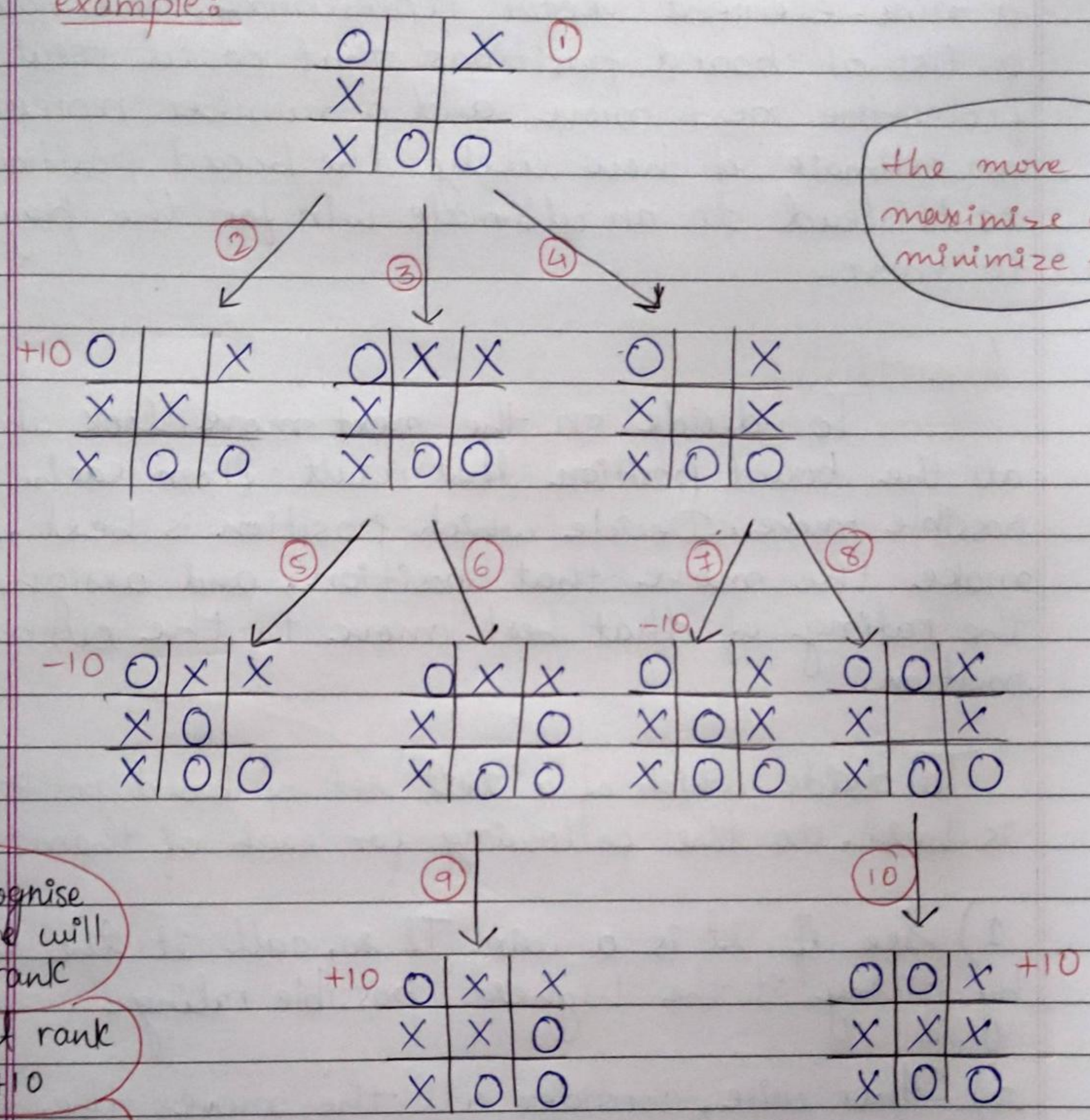
Scanned by TapScanner

Min-Max

Date
Page

Strategy-3

example:



the move will maximise and minimize win

to recognise win we will give rank

highest rank win +10

lowest rank win -10

Scanned by TapScanner

Data Structure:

Board Position:

A structure containing a nine-element vector representing the board, a list of board positions that could result from the next move, and a number representing an estimate of how likely the board position is to lead to an ultimate win for the player to move.

Algorithm:

To decide on the next move, look ahead at the board position that result from each possible move. Decide which position is best, make the move that position, and assign the rating of that best move to the current position.

To decide which of a ~~test~~ set of board position is best, do the following for each of them:

1) See if it is a win. If so, call it the best by giving it the highest possible rating.

2) Other wise, consider all the moves the opponent could make next. See which of them is worst for us. Assume the opponent will make that move. What ever rating that move has, assign it to the node we are considering.

3) The best node is then the one with the highest rating.

Scanned by TapScanner

- This algorithm will look ahead at various sequence of moves in order to find a sequence that leads to a win.

- It attempts to maximize the likelihood of winning, while assuming that the opponent will try to minimize that likelihood.

- This algorithm is called the minimax procedure.

Scanned by TapScanner

Unit 2

Problems, Problem Spaces and search

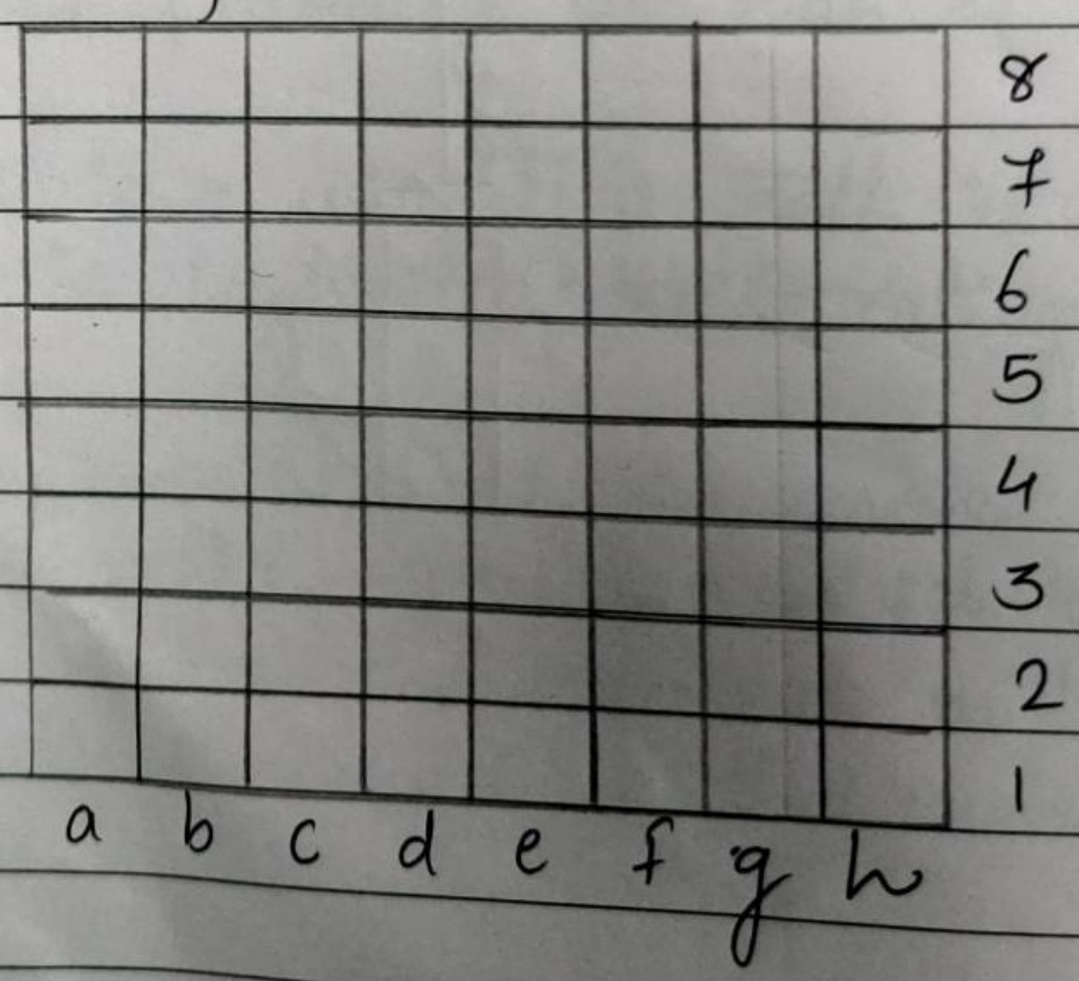
To built a system to solve a particular problem we need to do four things:

1. Define the problem
2. Analyze the problem.
3. Represent the task knowledge to solve the problem.
4. Choose the best technique(s) to solve the problem

Defining the problem as a state space search:

Part-I State space moves:

1. Visual representation.



Scanned by TapScanner

(2) Symbolic representation.

V	H	R	Q	K	R	H	V	8 B
S	S	S	S	S	S	S	S	7
								6
			S					5
				S				4
								3
S	S	S	S	S	S	S	S	2 W
V	H	R	K	Q	R	H	V	1
a	b	c	d	e	f	g	h	

Total 10^{120} possible board moves

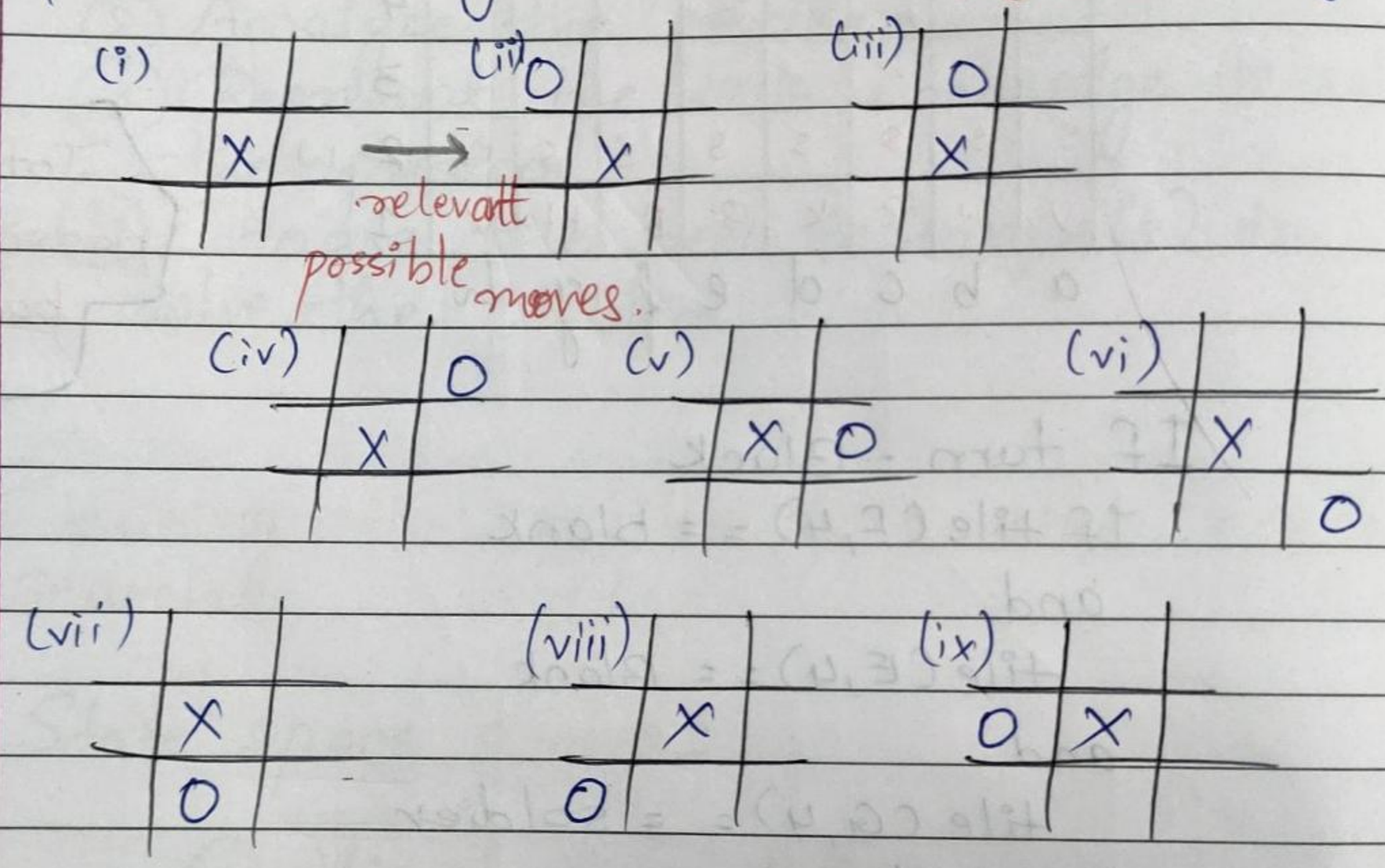
If turn = Black
  If tile (F,4) == blank
  and
    tile (E,4) == Blank
  and
    tile (G,4) == soldier
  then tile
    (E,4) == soldier.

Scanned by TapScanner

Date
Page

Formal description of a problem requires the following:

(1) Definition of a state space containing all possible configuration of relevant object.



(2) Specification of one or more states within the space giving description of problem solving start states known as initial states.

Note: Initial state would be first start.

Scanned by TapScanner

Date
Page

(3) Specification of a one or more states from the spaces giving description of acceptable solution to the problem known as final state [goal state].

e.g. tictactoe.

→ goals state can be many it depends on rules
→ initial state will be one

(4) Specification of set of rules to describe available actions.

→ If then rule [knowledge base]

next move for action

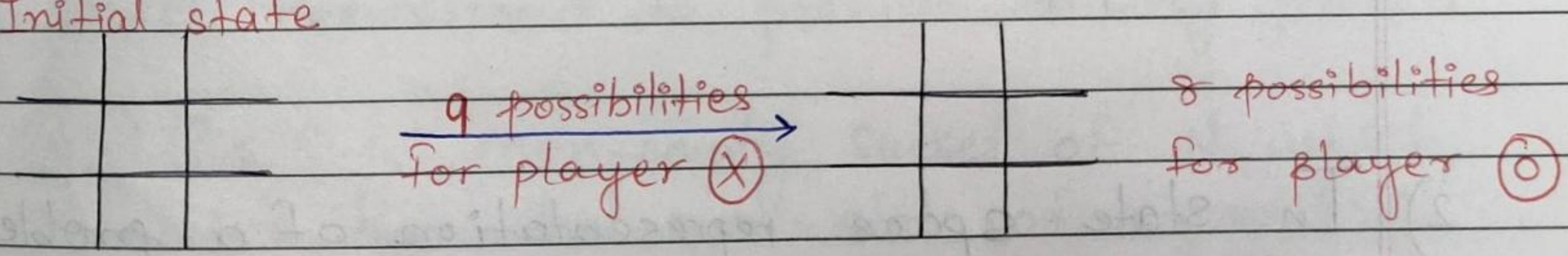
Part-II

Problem spaces:

State space representation:

e.g. tic-tac-toe, 3x3 → board

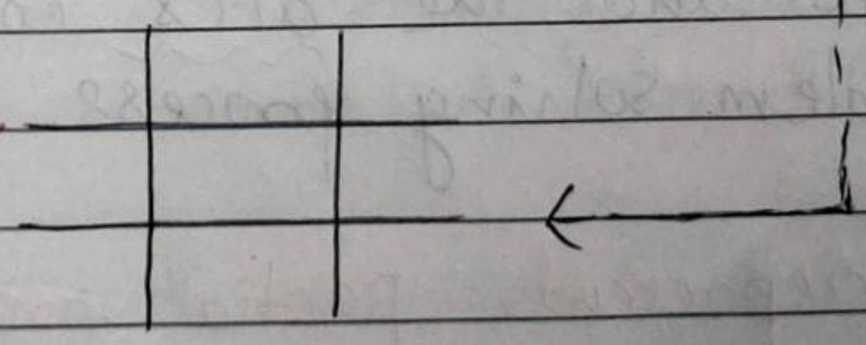
Initial state



and so on....

In visual rep. we take both I.S & G.S

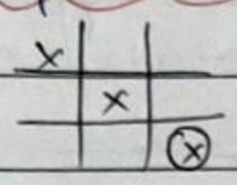
Goal state



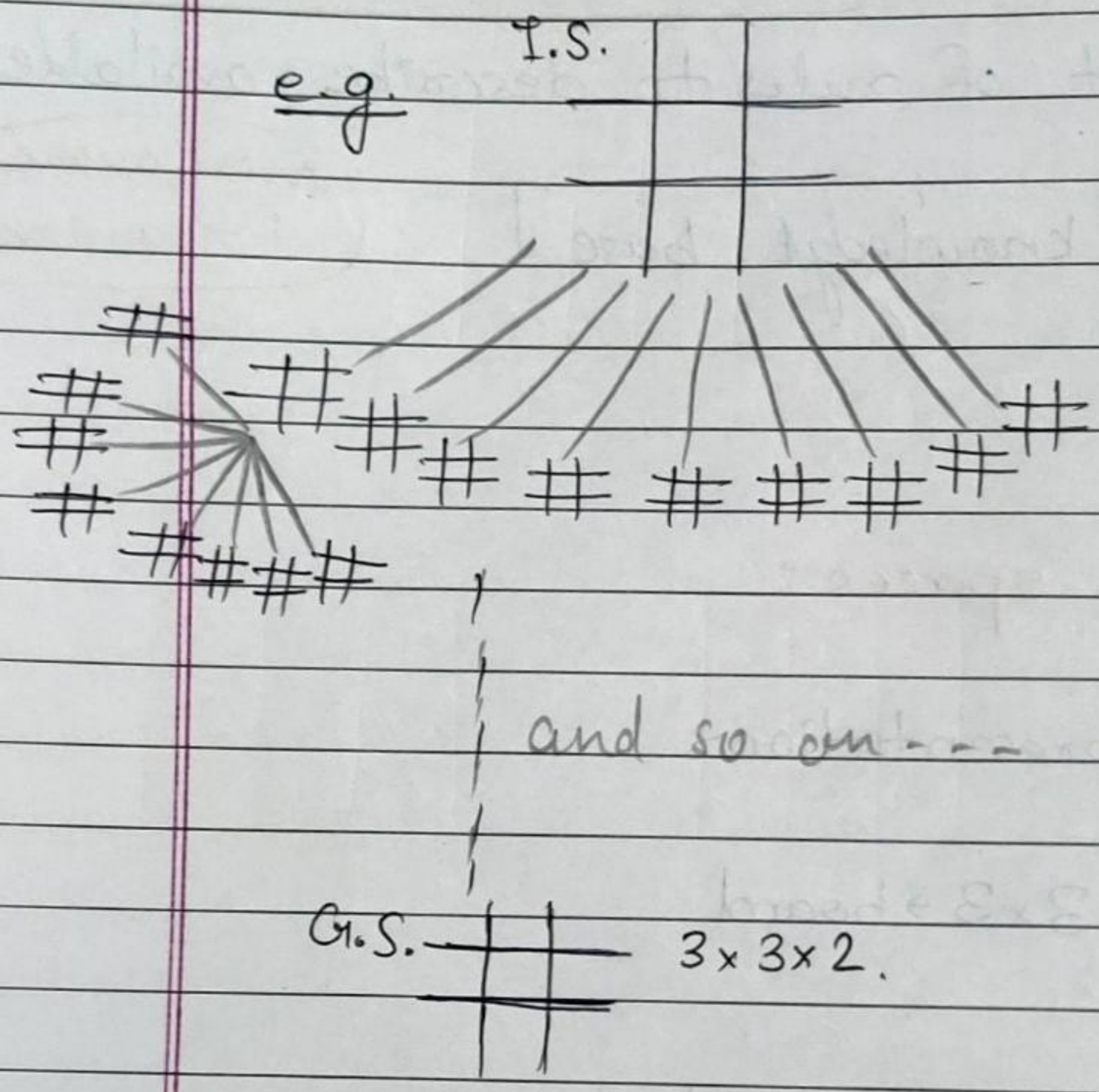
Scanned by TapScanner

only play legal move correspond to game

1) A representation of state of a problem where each state ~~the~~ b. corresponds to a ~~leago~~ legal possible move on the board.

e.g.  $3 \times 3 \times \underline{2} = 18$

e.g.



$3 \times 3 \times 2$.

How many goal state in tic-tac-toe?

2) In state space representation of a problem the nodes of a graph correspond to a partial problem solution state and the arcs corresponds to step in the problem solving process.

→ each node represents partial problem solving

→ to take arcs we have to ~~analys~~ analyize move and rules.

3) The graph also represent:
(I) Initial state corresponding to the given board forming the root node [initial state].
(II) one or more goal state which are solutions to the problem forming leaves of the graph.
[terminal]
[end of goal state]

Part-III

State space search:

1) A state space is representing by four tuple i.e. [N, A, S, GD] where,

(i) N : N is set of nodes states corresponding to problem solving process states

(ii) A: A is a set of arcs between the nodes which correspond to problem solving process

(iii) S : S is non-empty subset of N which contains the start ~~spat~~ state of a problem

(iv) GD: GD is not empty subset of N which contains goal state of a problem.

2) A solution path in this graph is from node S to node GD.

Scanned by TapScanner

3) State space search consist of some searching technique to reach to a goal state starting from initial state following a possible path in a state space representation.

example: 8 puzzle problem.

I.S.

3	7	5
4	2	8
	1	6

→

G.S

1	2	3
4	5	6
7	8	

3	7	5
	2	8
4	1	6

3	7	5
4	2	8
1		6

and move will be so on...

example: Two Jug Problem

we have a less move to finish a task

one Jug → 4 litre
second Jug → 3 litre

Jug moves → (F, E, P)
F → fill, E → empty, P → Pour

(0,0) → (2,0)
I.S

Solution: (0,0) → (0,3) pour 3l in 2 Jug
↓
(3,0) pour 3l of 2 Jug in 1 Jug
↓
(3,3) again fill 2 Jug of 3l
↓
(4,2) → (0,2) → (2,0) G.S

Pour 1l in 4l of 1 Jug

Now, empty 1 Jug

pour 2l in 1 Jug of 2 Jug

Scanned by TapScanner

Three Jug Problem.

1 - 8 litre 2 - 5 litre 3 - 3 litre

(0, 0, 0) → (4, 4, 0)

	8 moves	9 moves	11 moves
0	0 0 0		0 0 0
1	8 0 0		0 0 3
2	3 5 0		3 0 0
3	3 2 3		3 0 3
4	6 2 0		6 0 0
5	6 0 2		1 5 0
6	1 5 2		1 2 3
7	1 4 3		1 2 0
8	4 4 0		1 0 2
9			1 5 2
10			1 4 3
11			4 4 0

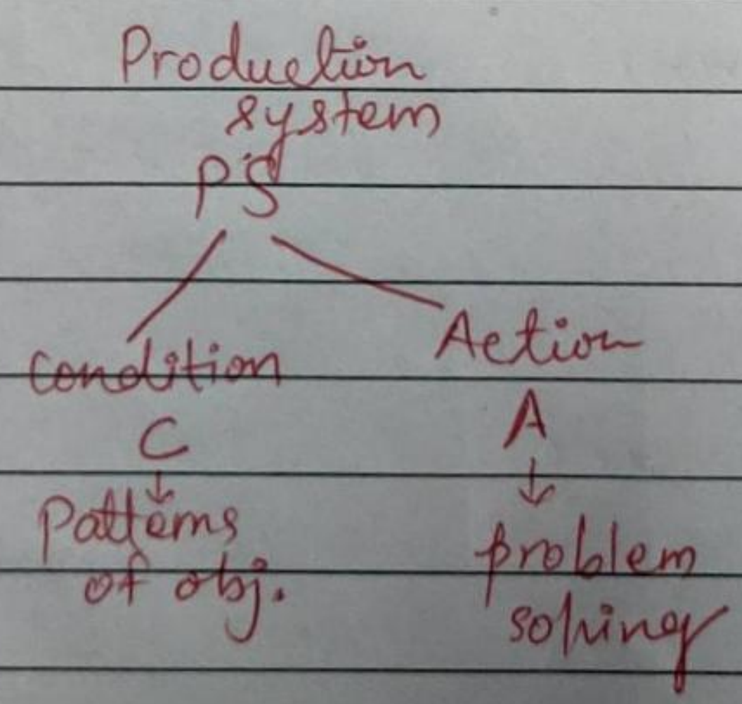
Scanned by TapScanner

Production system:

1) A production system is a model of computation used in AI for implementing search algorithm and for modeling human problem solving process.
[search forms the core of many intelligent processes, it is useful to structure AI programs in a way that facilities describing and performing the search process].
[search algorithm will give optimal path and direction].

2) It provides control over problem solving process and it consists of:

(i) A set of rules called production rules consisting of condition-action pair.
(I) Condition is a pattern which determines when a rule may be applied to the object.
(II) Action it determine the problem solving step.



Scanned by TapScanner

(ii) One or more knowledge data base which contains the description of the current ~~sit~~ state of the world. The description is matched against the condition part of production rule to select appropriate problem solving action.
- When a match is found the action associated with condition is perform to alter the knowledge data base.

3) A control strategy which is a mechanism which loops in a simple cycle to decide which rule can be applied.
- The control strategies has two requirements:
(i) It should be simple, not too complex.
(ii) It should not loop endlessly (∞).

4) Rule applier [T, B, L, R]

Top → T Bottom → B Left → L Right → R

e.g 8 puzzle game. in this game we only take move only top, bottom, left and right and after taking this move we will win the game.

6	7	8
3	4	5
	1	2

Scanned by TapScanner

Date
Page

e.g. e.g.

Heuristic : [are like tour guide] [travelling sells man. problem] → The intial state and goal state is given and we have to give min. path time and not repeating path

- Heuristic come from greek word heurika which means i found something.

Heuristic → search algorithm → find direction or path → in graph search [should be low]

best path or optimal path → to reach goal state.

e.g. parking me gaye and 1,3,9 position khali he but hum 9 par dekha and gaye the

best state → The best state ~~is~~ ~~which~~ is that in which the cost is less.

good state → The good state is that in which the cost is more but first we have seen.

vo good state hua and 1,3 are best state

- Heuristic is a technique which improves the efficiency of a search process possibly by sacrificing the claims of completeness.
[it leads near to goal state but not a goal state].

- In graph we will search goal state.

e.g. In 8-puzzle we will see what will be goal state in graph and in the term of cost.

Scanned by TapScanner

- The move of taking the number to the goal state position is known as cost.

- We have initial state and going to goal state we need cost and it should be less if cost is decreasing then we can say that we are in correct direction.

Hueristic can be applied in one of the following situation:

1) A problem may not have an exact solution because of inherant ambiquies in the problem statement or available data.

e.g. medical diagonistic system.

reasons: If data incomplete then we can't reach to goal state. [just like if we have half info of any disease then we can't say what exact the problem is the to the patients].

2) A problem may have an exact solution but the cost of finding that solution is prohibited.

e.g. chess board. [we see the oponent move]

⇒ distance of is that which tells the how much move we have take to reach goal state

Date
Page

Purpose:

The purpose of heuristic function is to guide the search process in the most profitable direction by suggesting which path to follow first when more than one is available.

Heuristic function:

- Heuristic function maps a problem state description to the measure of desirability. This function provides an estimate of wheather a path is good or bad.

- Two types of Heuristic function:
  1) local heuristic function
  2) global heuristic function

Advantages of Heuristic:

1) It eliminates obviously bad path.
2) It provides guidelines for perspective path.

Scanned by TapScanner

Date
Page

Heuristic for 8-puzzle problem:

Algorithm:

1) Find the distance of each ~~tile~~ tile from its goal ~~state~~ position.

2) Find the sum of all distances and reject the one has greatest distance.

3) Choose the possibility which minimize the distance in order to select a tile for making the next move, use the following steps:

3.1) Consider each tile from the current state that can be move say $t_1, t_2, ----, t_k$.

- Make the move and get the resulting states say $S_1, S_2, ----, S_k$.

3.2) Evaluate each state $S_1, S_2, ----, S_k$ as follows:

3.2.1) In each state S_i consider each tile 1/1.

3.2.2) Find its distance from the destination position in the goal state and add this distances for each tile.

3.2.3) Let the resulting distances be $D_1, D_2, ----, D_k$.

3.2.4) Find the smallest distance D_i out of all distances to make next move.

Scanned by TapScanner

Note: Heuristic is actually our strategy $\xrightarrow{\text{numerical value}}$ apply heuristic function

Date
Page

example:

Initial State

1	7	4
3		6
8	5	2

Goal State

1	2	3
4	5	6
7	8	

to reach goal state how much distance we need:

Tile to Move (distance)

tile	1	2	3	4	5	6	7	8
distance	0	3	3	3	1	0	3	1

Total of all distance = 14

1	7	4
3		6
8	5	2

$\xrightarrow{NS_{14}}$

1	7	4
3	5	6
8		2

NS_{11}

1		4
3	7	6
8	5	2

NS_{12}

1	7	4
	3	6
8	5	2

NS_{13}

1	7	4
3	6	
8	5	2

$D(hNS_{11}) \rightarrow$

1	2	3	4	5	6	7	8
0	3	3	3	1	0	2	1

$\rightarrow 13$

$D(hNS_{12}) \rightarrow$

1	2	3	4	5	6	7	8
0	3	2	3	1	0	3	1

$\rightarrow 13$

Scanned by TapScanner

Date
Page

$D(hNS_{13}) \rightarrow$

1	2	3	4	5	6	7	8
0	3	3	3	1	1	3	1

$\rightarrow 15$

$D(hNS_{14}) \rightarrow$

1	2	3	4	5	6	7	8
0	3	3	3	0	0	3	1

$\rightarrow 13$

Scanned by TapScanner

Note: 7 questions will be there and write the answer in yes or no and then Justify.

Date
Page

Problem Characteristic: [Total 7]

(divide in parts or not)

1) Is the problem decomposable.

No → e.g. tic-tac-toe, 8 puzzle problem, Travelling sales man problem, Water Jug Problem, Chess, Farmer.

Yes → e.g. 8 queen problem, a mathematical problem.

2) Can the solution step be ignored or undone?

i) Recoverable ii) irrecoverable iii) ignore

e.g. 8 puzzle problem → Recoverable
tic-tac-toe → Irrecoverable
water-Jug → ignore
farmer → Irrecoverable
Chess → Irrecoverable [50% ignore]
8-queen problem → Irrecoverable.
Tower of Hanoi → Recoverable.

3) Is the problem's universe is predictable?

e.g. tic-tac-toe, chess we can't predictable.

opponent more is predictable

e.g. 8 puzzle problem, Travelling sales man Problem, Tower of Hanoi, 8 queen problem we can predictable.

Scanned by TapScanner

4) Is the good solution absolute or relative?

absolute: 8-puzzle problem, Tic-Tac-Toe, Chess

(fix)

relative: Water Jug Problem, Farmer, Travelling Sales Man Problem.

(not fix, dynamic)

5) Is the solution a state or path?

state - tic-tac-toe [fix board position], Chess

path - TSMP, Water Jug, Farmer

(where we can decide the route)

6) What is the role of knowledge?

Level of knowledge

1) Less → Tic-Tac-Toe

2) Moderate → Sales Man Problem, Tower of Hanoi, 8-queen Problem.

3) High → Chess → 8-queen Problem

8 queen can be moderate or high

7) Does the task require interaction with person?

Two types

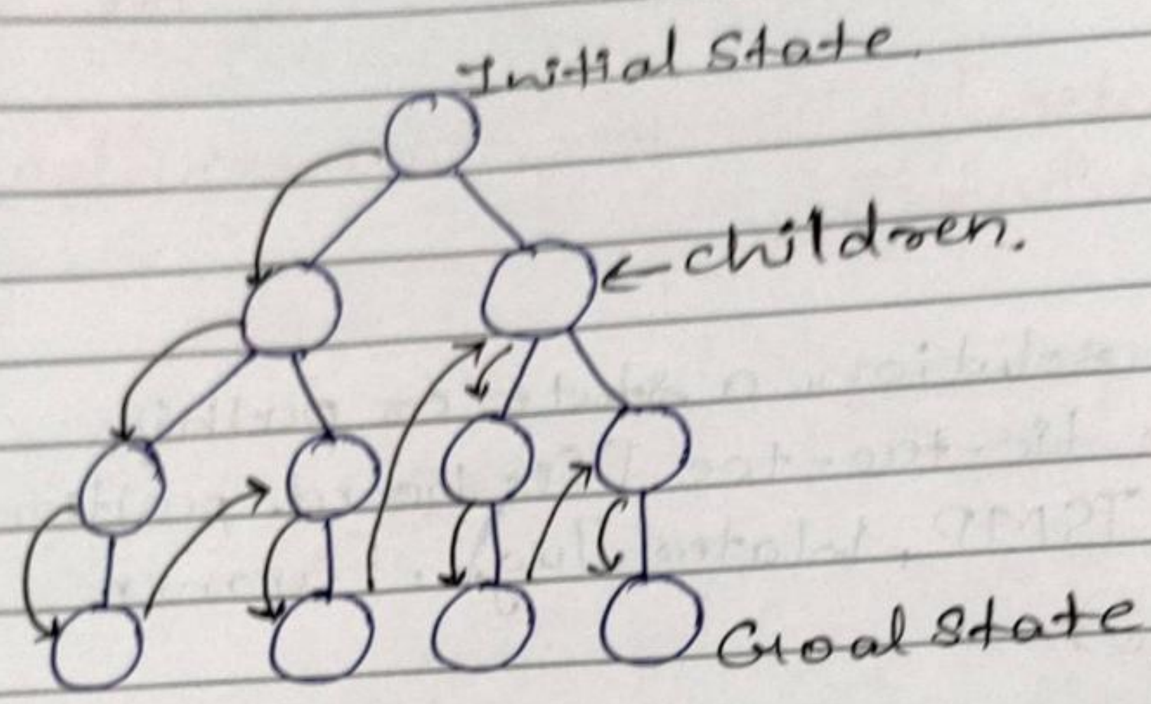
1) Conversational → Tic Tac Toe, Chess.
(2 player)

2) Solitary → Farmer, Water Jug, 8-puzzle.
(we decide where should go i.e. 1 player)

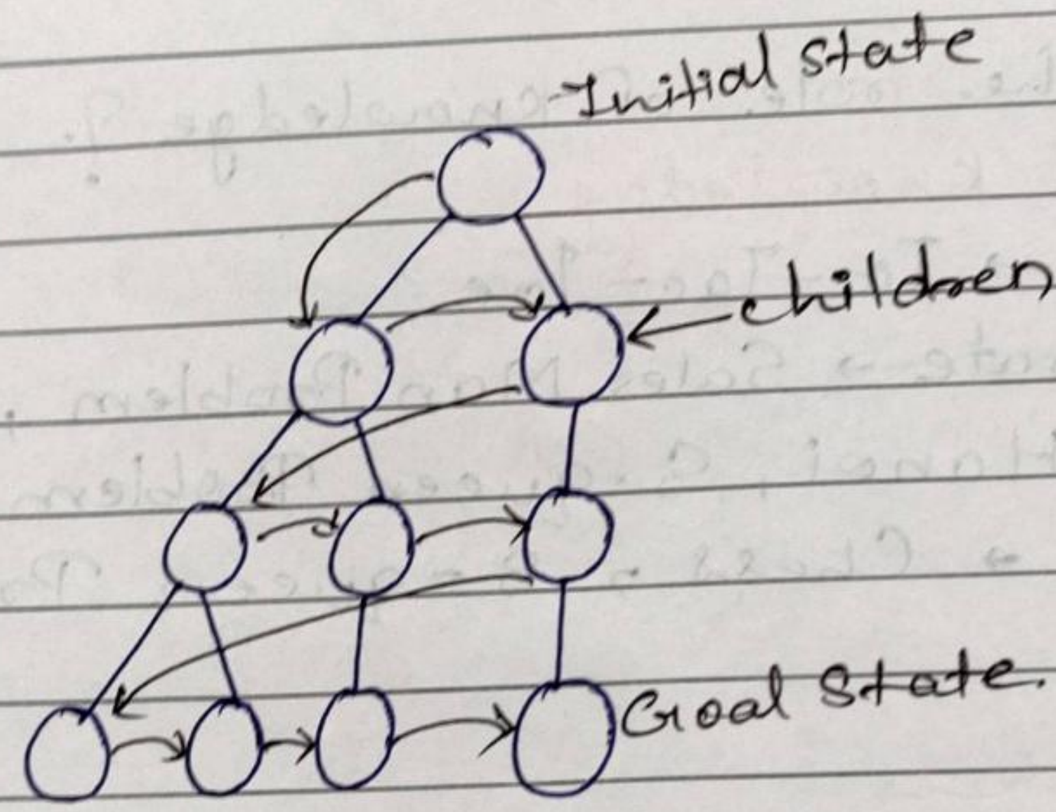
Scanned by TapScanner

Breadth First Search
Depth First Search.

DFS →



BFS →



DFS:

Algorithm:

1) Place starting node "S" in the queue.
2) If the queue is empty report failure & stop.
3) If the first element of the queue is goal report success & stop.
4) Remove first element of the queue expand it and place all the children in front of the queue.
5) Go to step 2.

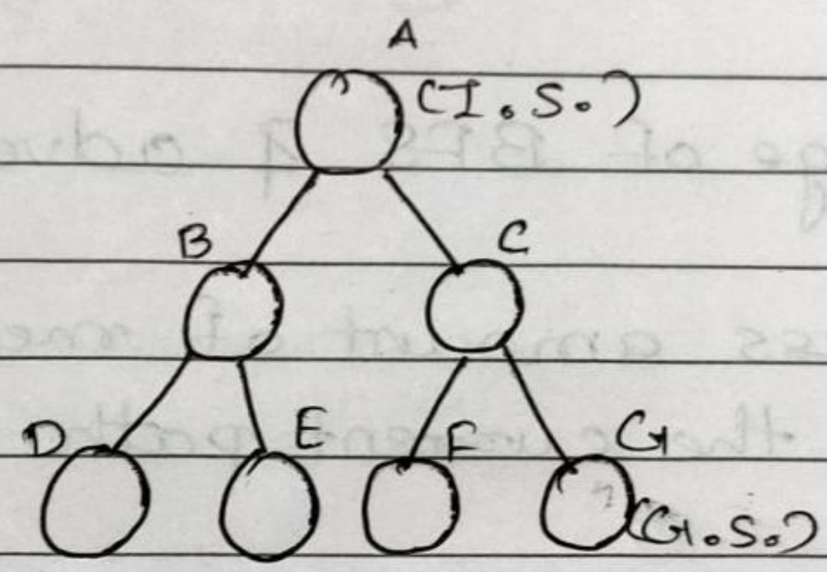
Scanned by TapScanner

BFS:

Algorithm:

1) Place starting node "S" in the queue.
2) If the queue is empty report failure and stop.
3) If the first element of the queue is goal report success & stop.
4) Remove first element of the queue expanding it and place all the children in ~~front~~ rear of the queue.
5) Go to step 2.

<u>example</u>:



DFS: [~~A~~]

[~~B~~ C]

[~~D~~ E C] D & E will be kept front of C.

[~~E~~ C] E doesn't have any children i.e. [C]

[~~C~~]

[~~F~~ G]

[G] → stop

Report = success

Traversal path: [path to follow goal state]

[A B D E C F G]

Scanned by TapScanner

BFS:

[A]
[B C]
[~~C~~ D E]
[~~D~~ E F G]
[~~E~~ F G]
[~~F~~ G]
[G] → stop
Report = success

Traversal path:
[A B C D E F G]

Disadvantage of BFS & advantange of DFS:

1) It require less amount of memory because the nodes only in the current path are to be store.

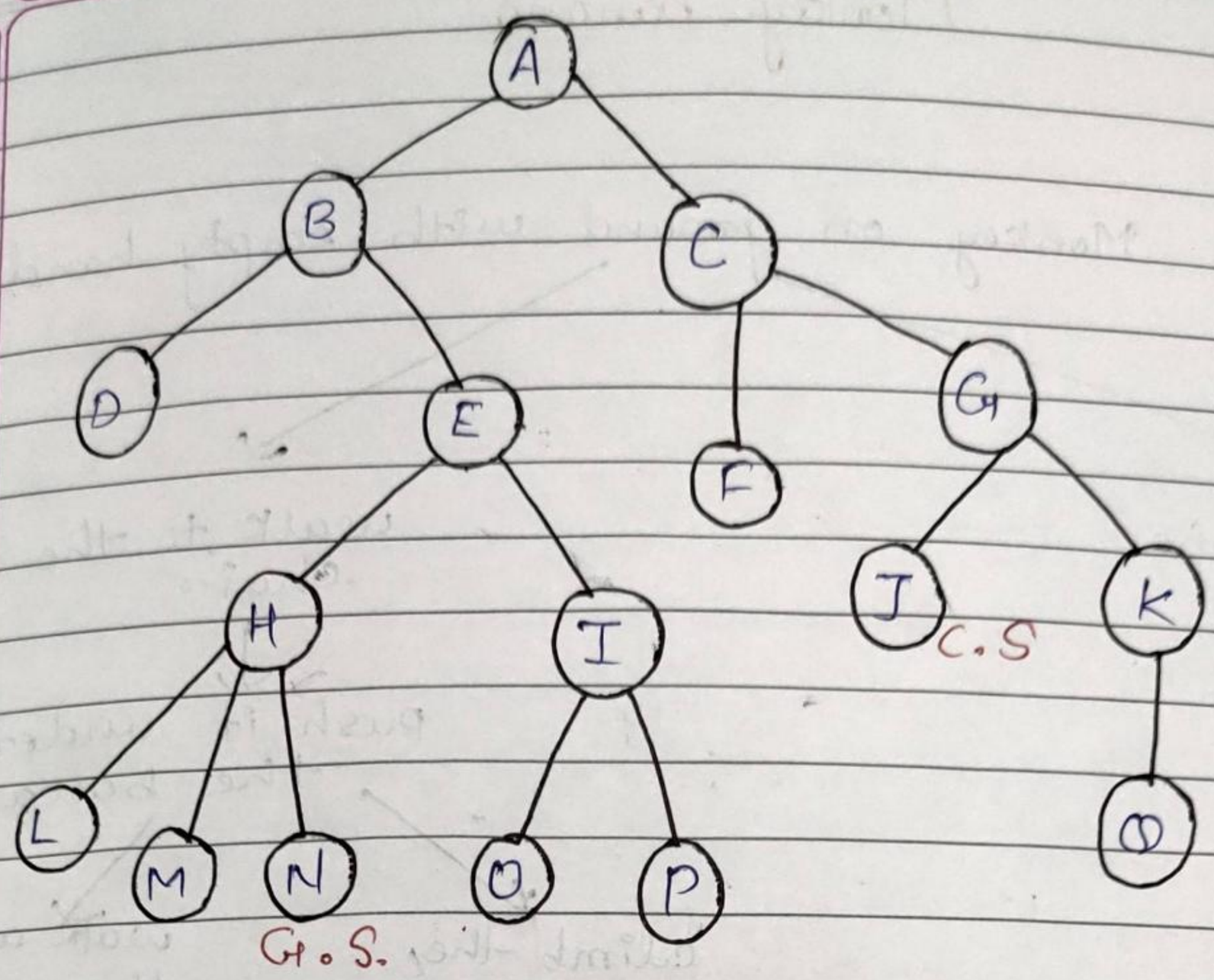
2) It can get to a solution quickly without examining much of the search tree if the solution lies in the left branch of the tree.

Advantage of BFS & Disadvantage of DFS:

1) It always find a solution if it exist. Larger path are never explored if shorter ones exists.

2) It never gets tracked into a blind valley. (we don't know about next node).

Scanned by TapScanner



Scanned by TapScanner

Monkey banana

Monkey on ground with empty hand

- walk to the stick → took / Grab the stick → walk and stand below the banana → wave the stick → didn't get banana
- walk to the chair → push it under the banana
  - climb the chair → it's on the chair with empty hand → get off the chair with empty hand → Didn't get banana
  - walk and get the stick → walk and climb the chair → on chair with stick → wave the stick → Banana falls to ground → get off chair → drop the stick → monkey has (G.S.) banana

BFS	DFS
[A]	[A]
[B C]	[B C]
[C D E]	[D E C]
[D E F G]	[E C]
[E F G]	[H I C]
[F G H I]	[L M N I C]
[G H I]	[M N I C]
[H I J K]	[N I C]
[I J K L M N]	N is goal state
[J K L M N O P]	Traversal path
J is the goal state	[A B D E H L M N]

Traversal path

[A B C D E F G H I J]

Scanned by TapScanner

For n queen n×n board
For 8 queen
8×8 board.

8-queen ~~game~~ problem

→ The eight queens puzzle is the problem of placing eight chess queens on an 8×8 chess board so that no two queens threaten each other; thus, a solution requires that no two queens share the same row, column or diagonal.

→ There are 92 solutions.

a	b	c	d	e	f	g	h	
			q					8
						q		7
		q						6
							q	5
	q							4
				q				3
q								2
					q			1

Step-1:
If the remainder from dividing n by 6 is not 2 or 3 then the list is simply all even numbers followed by all odd numbers not greater than n.

Step-2:
Otherwise, write seperate lists of even and odd numbers
(2, 4, 6, 8 – 1, 3, 5, 7)

Scanned by TapScanner

Step-3:
If the remainder is 2, swap 1 and 3 in odd list and move 5 to the end
(3, 1, 7, 5)

Step-4:
If the remainder is 3, move 2 to the end of even list and 1, 3 to the end of odd list (4, 6, 8, 2 – 5, 7, 1, 3)

Step-5:
Append odd list to the even list and place queens in the rows given by these num., from left to right
(a_2, b_4, c_6, d_8, e_3, f_1, g_7, h_5).

Scanned by TapScanner

Date
Page

Farmer, Fox, Goat, Grass Problem

Condition:

(i) Goat and Grass cannot be together, otherwise it will eat the grass.

(ii) Fox catch goat alone then fox will eat the goat.

(iii) Farmer is the only driver of boat

(iv) Boat has capacity of two.

Notations:

Farmer → F_a Goat → G_1

Fox → F_o Grass → G_2

Start	River	
F_a, F_o, G_1, G_2		O
F_o, G_2	$\xrightarrow{F_a, G_1}$ $\xleftarrow{F_a}$	G_1
F_o	$\xrightarrow{F_a G_2}$	$F_a\ G_1, G_2$
$F_a F_o G_1$	$\xleftarrow{F_a G_1}$	G_2
G_1	$\xrightarrow{F_a F_o}$	$F_a F_o\ G_2$
$F_a G_1$	$\xleftarrow{F_a}$	$F_o G_2$
O	$\xrightarrow{F_a G_1}$	$F_a F_o G_1 G_2$ Goal State

Scanned by TapScanner

3 Missionaries and 3 Cannibals Problems

Condition:

(i) On any side of river. No. of cannibals (C) should not be more than no. of missionaries (M). Otherwise (C) hunt (M).

(ii) Boat has capacity of two person. ~~Notation~~.

Notation:

For missionaries → M
For cannibals → C

Start Page	3M 3C	River	0
~~Step 1~~			
Step 1	3M 1C	$\xrightarrow{2C}$	2C
Step-2	3M 2C	$\xleftarrow{C}$	1C
Step-3	3M	$\xrightarrow{2C}$	3C
Step-4	3M1C	$\xleftarrow{C}$	2C
Step-5	1M 1C	$\xrightarrow{2M}$	2M2C
Step-6	2M 2C	$\xleftarrow{1M1C}$	1M1C
Step-7	2C	$\xrightarrow{2M}$	3M1C
Step-8	3C	$\xleftarrow{C}$	3M
Step-9	1C	$\xrightarrow{2C}$	3M2C
Step-10	2C	$\xleftarrow{C}$	3M1C
Step-11	0	$\xrightarrow{2C}$	3M 3C Goal State

Scanned by TapScanner

Unit - 3

Generators and Tester

Generator → generates possible sol^n
↓
two ways
(1) random → easy way to search
↓
it picks any solution ← doesn't gaurantee goal state

(2) systematic → completedness
↓
also known as DFS

Tester → test the fun^n or solution wheather it is acceptable or not
↓
if not then again go to the generators

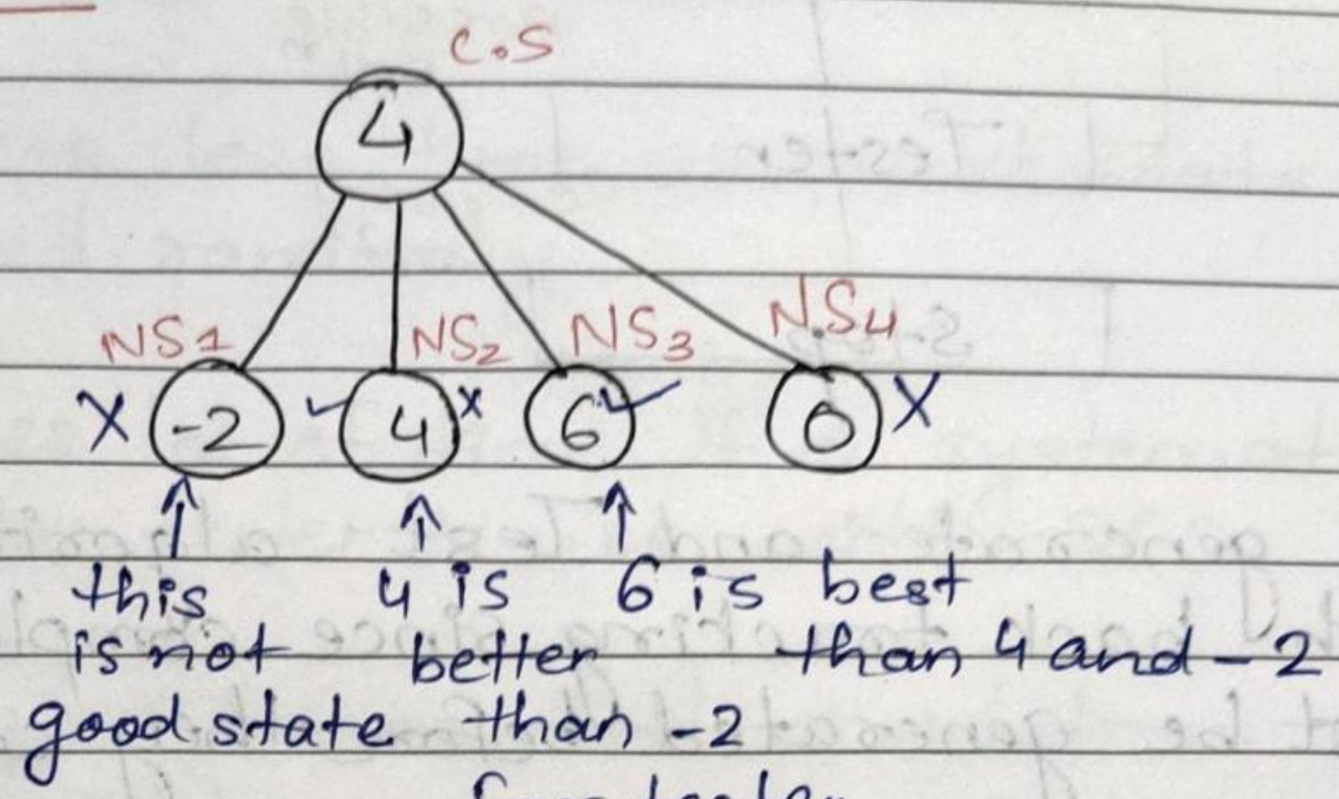
Tester will show we are on right path or not by saying yes or no.

→ from current state (c.s.) generator will generate the next state (n.s) then the tester will check wheather the c.s is better or n.s. is better to each other.

→ I.S → C.S → N.S

Scanned by TapScanner

example:



Now, 6 will become C.S.

Generate and Test:

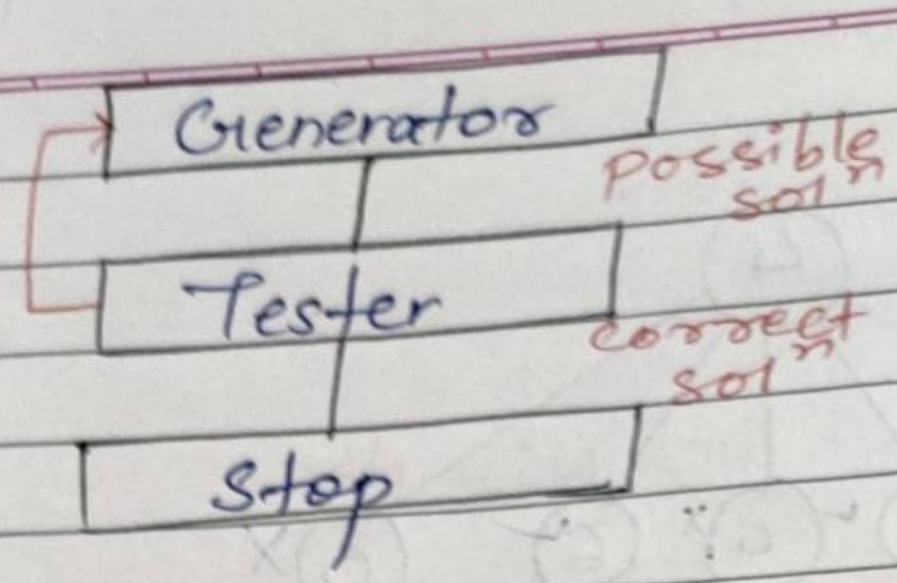
Generate and Test search algorithm is a very simple algorithm that gaurantees to find a solution if done systematically there exists a solution.

Generate - Test is applying method of searching goal state even an initial state. It consists of two function

1) Generator 2) Tester

Generator Funⁿ generates a possible next ~~move~~ node while the tester funⁿ compares the next state with acceptable goal state and report with yes/no to indicate wheather the goal state is reached.

Scanned by TapScanner



- The generate and Test algorithm is DFS with back tracking since complete solution must be generated before they can be tested.

- In its most systematic form it is simply an exhausting search of the problem space.

- Generate and Test, operates by generating solution randomly but then there is no gaurantee that a solution will ever be found. It is also known as British museum algorithm, a reference to a method for finding a object in a british museum by wondering randomly but some path are not considered because these seem unlikely to have to a solution. This evolution is performed by heuristic function.

Scanned by TapScanner

Date
Page

Characteristic:

- It takes less time as next state are generated randomly.
- It takes lot of time if a systematic generates tester is use with DFS.
- DFS gaurantees that a solution would be found quickly.
- Random search didn't consider those paths a which seen unlikely to have a solution.
- The evolution by tester function is performed using hueristic function.

Scanned by TapScanner

Hill Climbing

→ In Hill Climbing if the test fun^n is augmented with a heuristic function that provides an estimate of how close a given state is a goal state.

→ Hill Climbing is often used when a good heuristic function is available for evaluating states but when no other useful knowledge is available.

e.g.

suppose you are unfamiliar city without a map and you want to get downtown. You simply aim for tall buildings.

Hill Climbing Technique:

1) Simple hill climbing.
2) Steepest Ascent hill climbing.

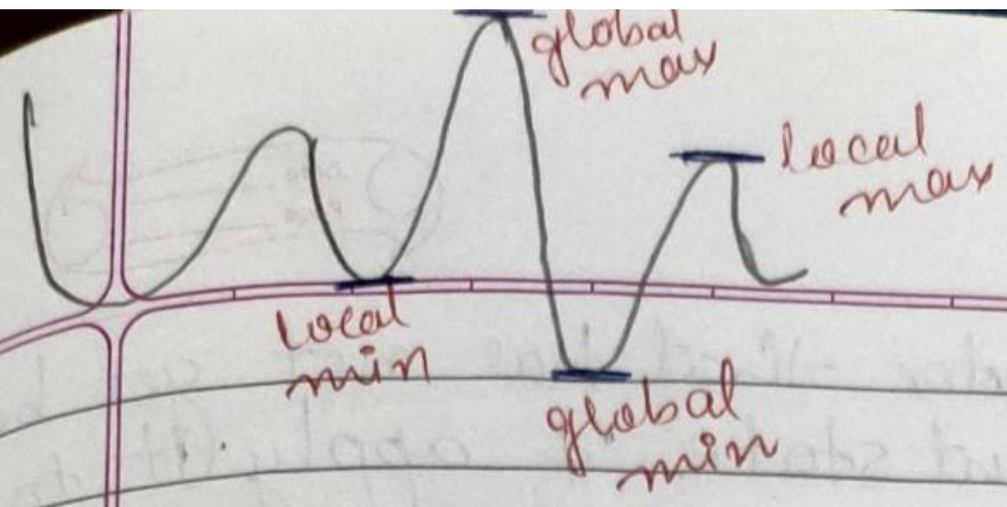
Introduction of Hill-Climbing:

→ Hill Climbing algorithm is a local search algorithm which continuously moves in the direction of increasing value to find the peak of the mountain or best solution of the problem.

→ It terminates when it reaches a peak value where no neighbour has a higher value.

(The neighbour (next move) value should be best when we move ahead)

Scanned by TapScanner



→ It is also called greedy local search as it only looks to its good imediate neighbour state and not beyond that.

→ Hill climbing is mostly use when a good heuristic is available and when no other useful knowledge is available.

$$\underset{\text{evaluation}}{Eva} = \underset{\text{Tester}}{T} + \underset{\text{good heuristic function}}{G.H}$$

Features:

→ Less time consuming
→ no back tracking
→ Less optimal sol^n & sol^n is not gauranteed.

Algorithm: Simple Hill Climbing:

1) Evaluate the initial state. If it is also a goal state, then return it and quit. Otherwise, continue with the initial state as the current state.

2) Loop ~~unt~~ until a solution is found or until there are no new operators left to be applied in the current state:

Scanned by TapScanner

(a) Select an operator that has not yet been applied to the current state & apply it to produce a new state

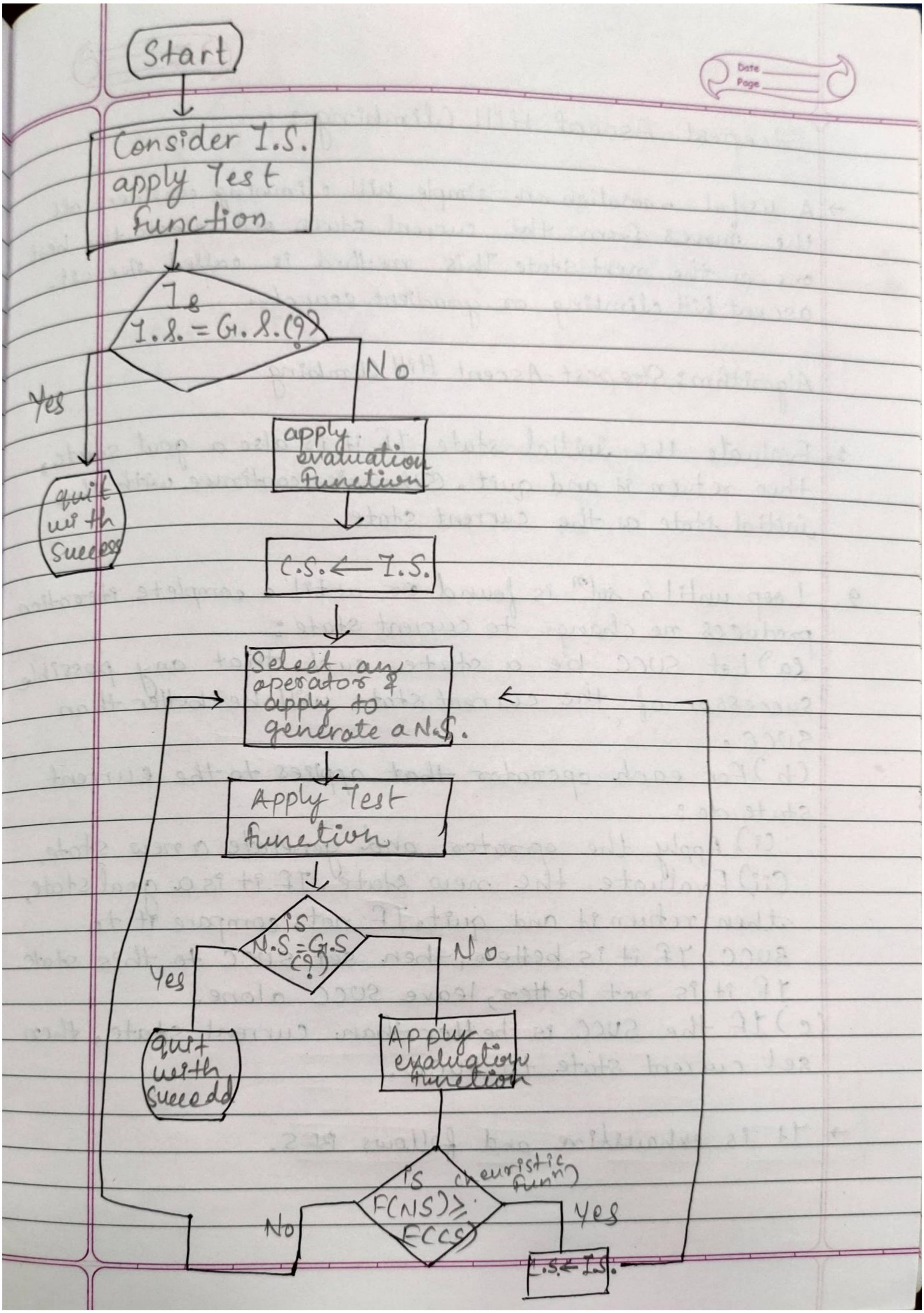
(b) Evaluate the new state

(i) If it is a goal state, then return it & quit.

(ii) If it is not a goal state but it is better than the current state, then make it the current state.

(iii) If it is not better than the current state, then continue in the loop.

Scanned by TapScanner



Scanned by TapScanner

Steepest-Ascent Hill Climbing:

→ A useful variation on simple hill climbing considers all the moves from the current state and selects the best one as the next state. This method is called steepest-ascent hill climbing or gradient search.

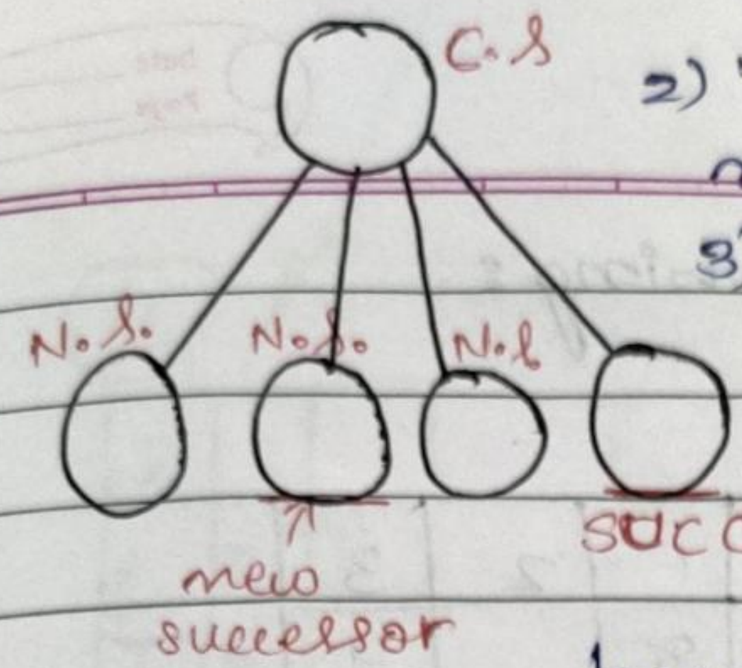
Algorithm: Steepest-Ascent Hill Climbing

1. Evaluate the initial state. If it is also a goal state, then return it and quit. Otherwise, continue with the initial state as the current state.

2. Loop until a sol^n is found or until a complete iteration produces no change to current state:
   (a) Let SUCC be a state such that any possible successor of the current state will be better than SUCC.
   (b) For each operator that applies to the current state do:
      (i) Apply the operator and generate a new state.
      (ii) Evaluate the new state. If it is a goal state, then return it and quit. If not, compare it to SUCC. If it is better, then set SUCC to this state. If it is not better, leave SUCC alone.
   (c) If the SUCC is better than current state, then set current state to SUCC.

→ It is exhaustive and follows BFS.

Scanned by TapScanner



1) In C.S we apply operator and the worst value (best move) is called successor.

2) Then the successor compare ~~itself~~ from next state.

3) Then we will check whether the N.S. have best value then successor.

→ If any N.S. is better than successor then that N.S will become new successor and that will compare with C.S.

→ And at last the new successor will become the C.S.

→ Iteration and current state change hona khatam tho search khatam.

Scanned by TapScanner

Date
Page

example of Simple Hill Climbing:

I.S.

2	8	3
1	6	4
7	5	

→

G.S.

1	2	3
8		4
7	6	5

Here I.S = G.S.

h(I.S.) = 6

1	2	3	4	5	6	7	8
1	1	0	0	1	1	0	2

Check F(CS) ≤ F(NS)

(8)

2	8	3
1	6	4
7		5

2	8	3	
1	6		(6)
7	5	4	

(5)

2	8	3
1		4
7	6	5

(6)

2	8	3
1	6	4
	7	5

(6)

2	8	3
1	6	4
7	5	

C.S of heuristis is less then above C.S

(3)

2		3
1	8	4
7	6	5

(5)

2	8	3
	1	4
7	6	5

(5)

2	8	3
1	4	
7	6	5

(5)

2	8	3
1	6	4
7		5

2 1 0 0 0 0 0 2

1 1 0 0 0 0 0 1

1 1 0 1 0 0 0 2

1 1 0 0 0 1 0 2

Scanned by TapScanner

Date
Page

(4)

2	3	
1	8	4
7	6	5

(2)

	2	3
1	8	4
7	6	5

1 00000001

1 1 1 0 0 0 0 0 1

(1)

1	2	3
	8	4
7	6	5

(3)

2	·	3
1	8	4
7	6	5

(0)

1	2	3
8		4
7	6	5

(2)

1	2	3
7	8	4
	6	5

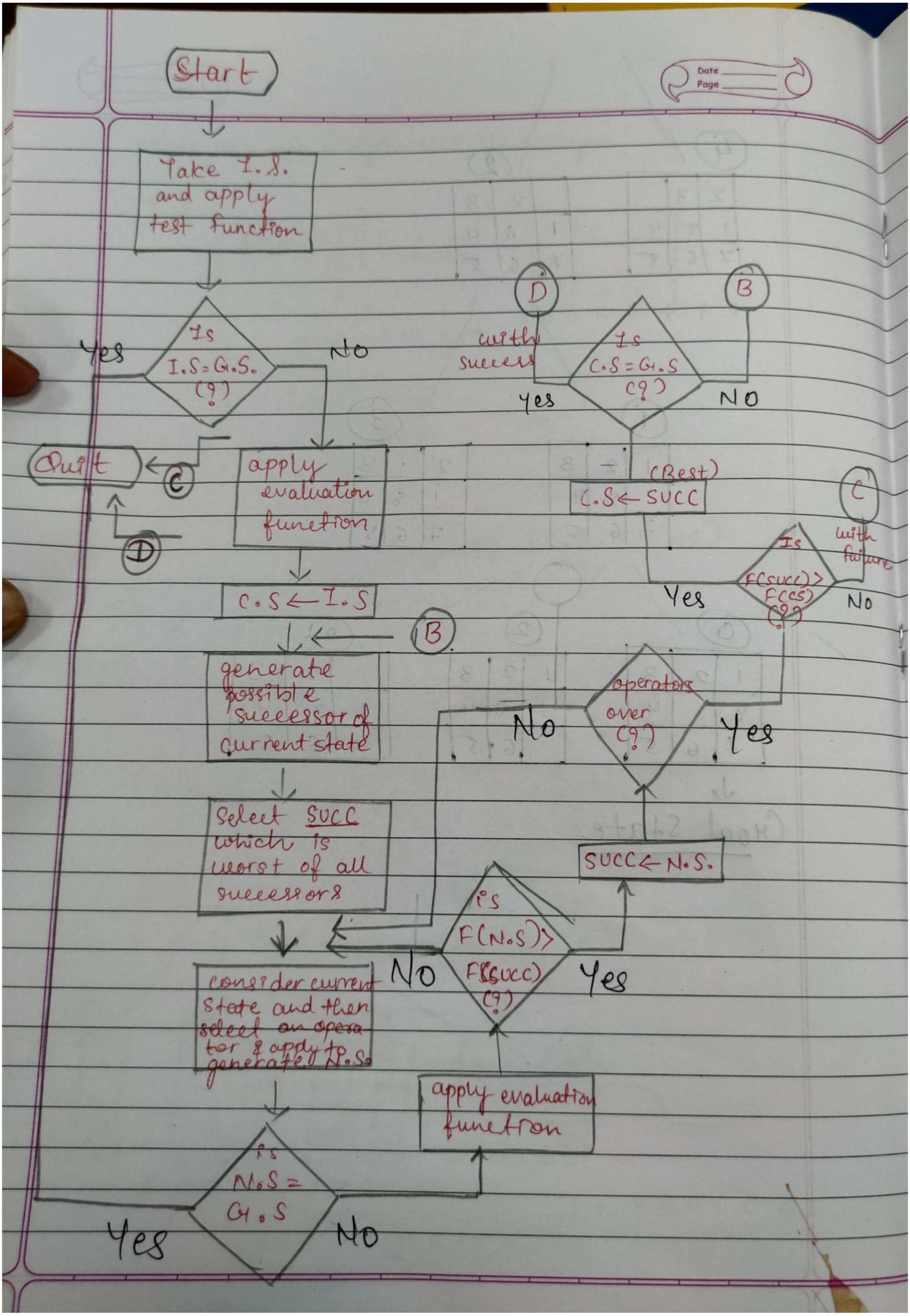
(2)

	2	3
1	8	4
7	6	5

↓

Goal State

Scanned by TapScanner

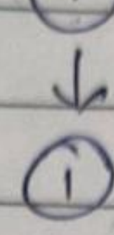


Scanned by TapScanner

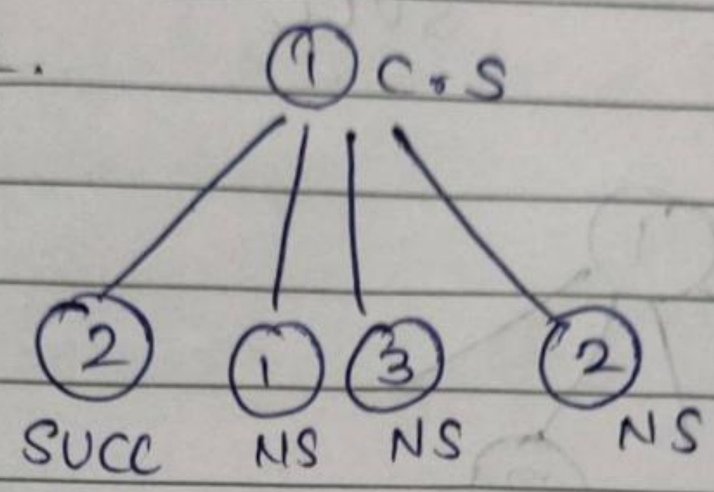
example: (1) → (4)
I.S. G.S.

steepest ascent →

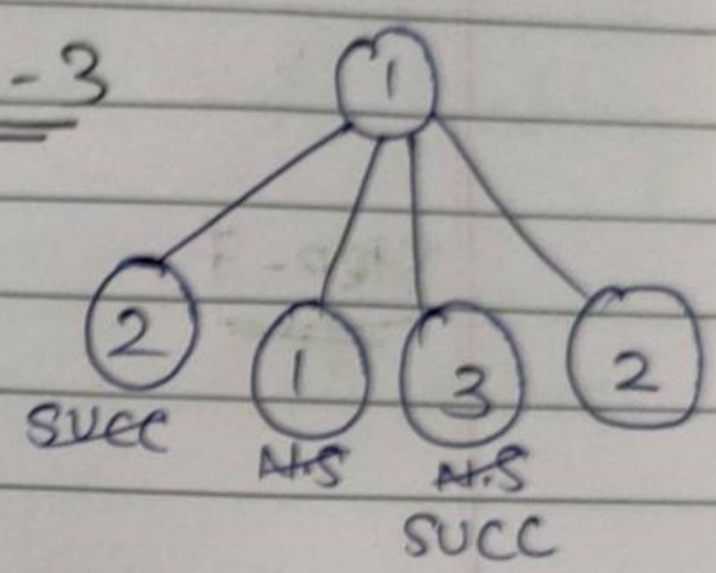
Step-1.



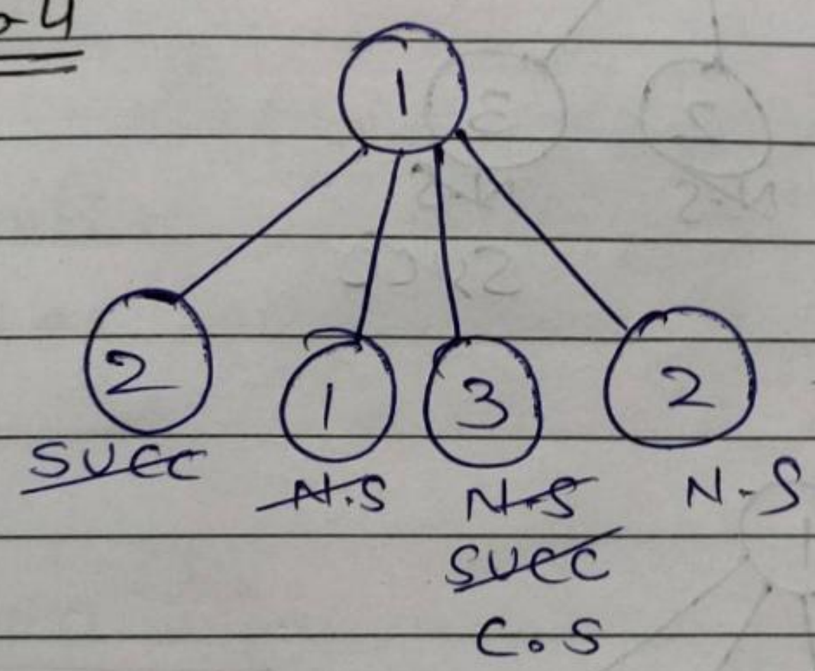
Step-2.



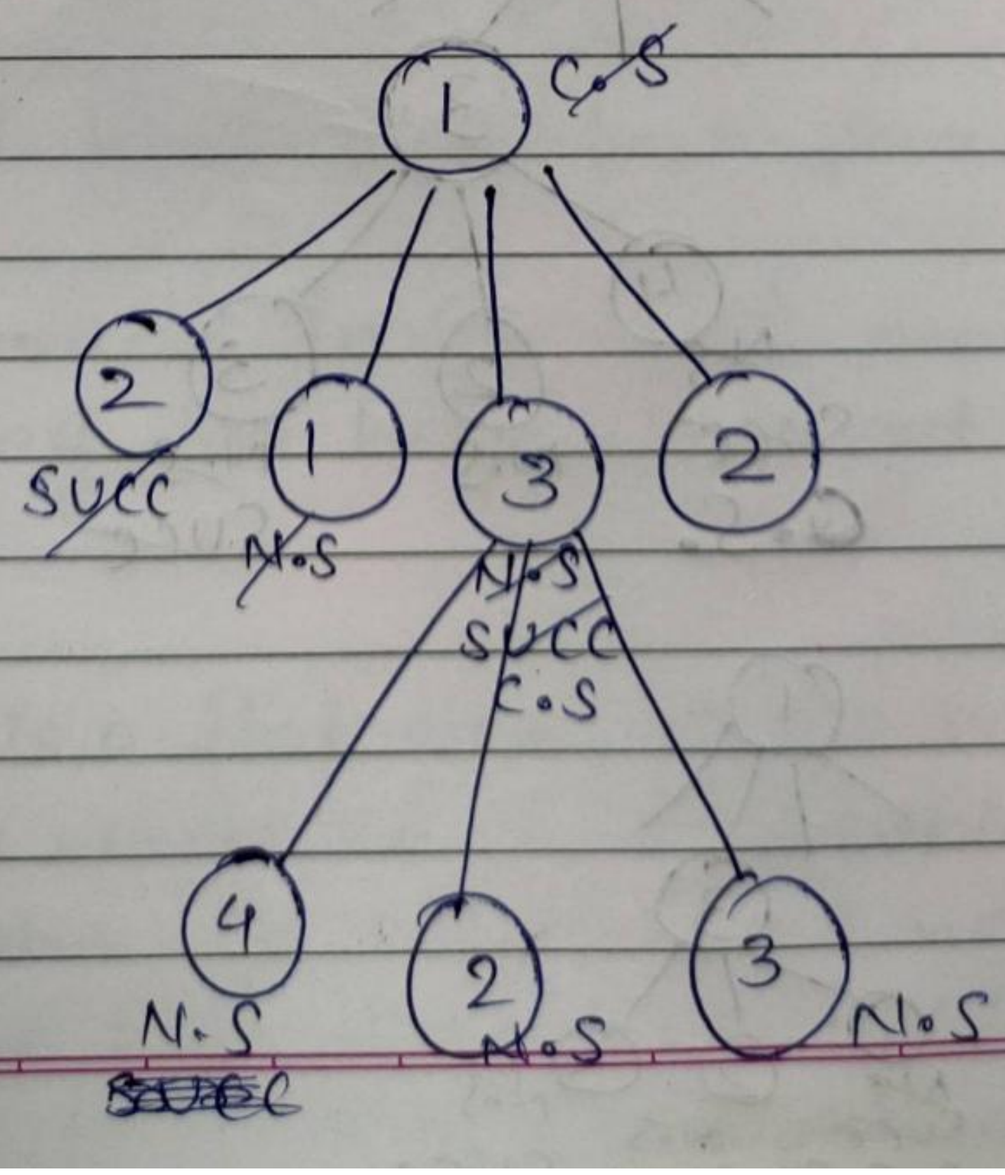
Step-3



Step-4



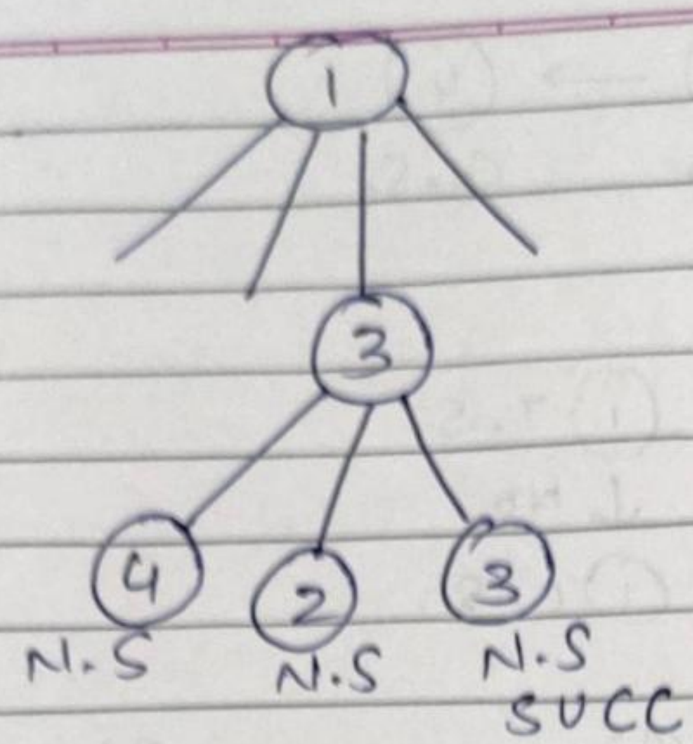
Step-5



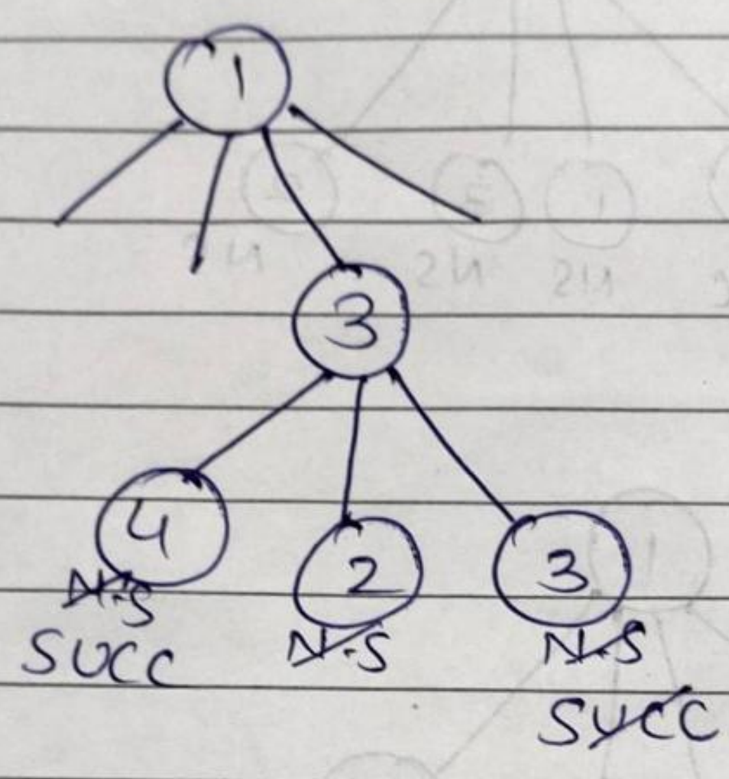
Scanned by TapScanner

Date
Page

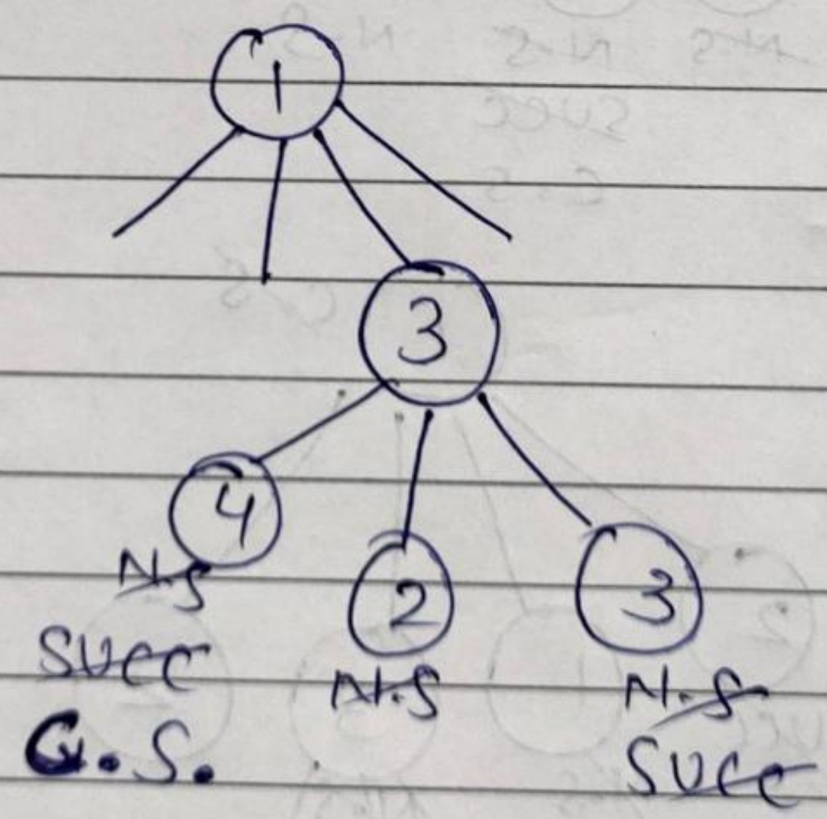
Step-6.



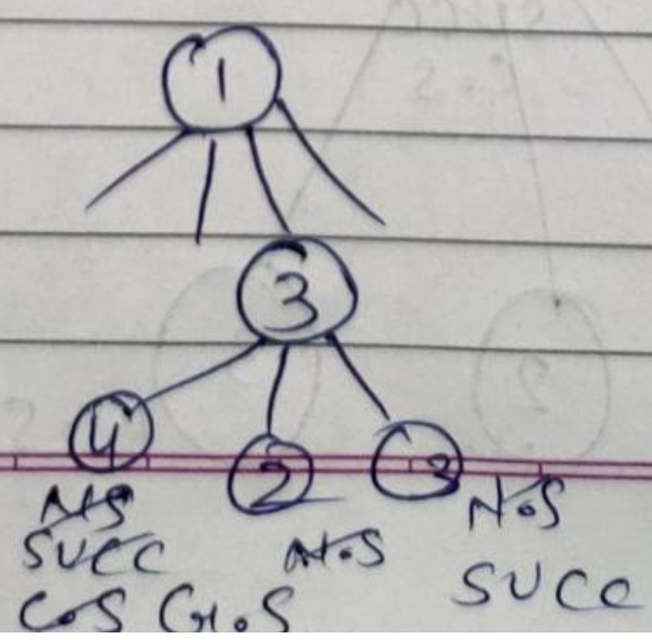
Step-7.



Step-8.



Step-9.



Scanned by TapScanner

- It is a variation of simple Hill Climbing technique.

- It consider all the moves from the current state instead of moving to the first state i.e. better, move to the best possible state i.e. one move away.

- It donot just climb to a better state but climbing up the steepest slope.

- It has elements of the breadth first algorithm.

Note:

Steepest ascent differs from the basic Hill Climbing algorithm by choosing the best successor rather than the first successor i.e. better.

Common Issue:

- It cannot backtrack to it parent node.

- If it gets track in the local maxima/minima, then nothing can help it to get out of that situation.

- A plateau is a flat area of the search space in which a whole set of neighboring states have the same value. On a plateau, it is not possible to determine the best direction in which to move by making local comparisons. (straigh area)

Scanned by TapScanner

- A ridge is a special kind of local maximum. It is an area of the search space that is higher than surrounding areas and that itself has a slope (which one would like to climb). But the orientation of the high region, compared to the set of available moves and the directions in which they move, makes it impossible to traverse a ridge by single moves.

(1) A state that is better than all of its neighbours but not then some other states far away.

Solⁿ: backtracking technique can be solution of local maxima or minima.

(2) plateau: A flat area of the search space in which all neighbouring states have the same value for that algorithm does not find any best direction to move.

Solⁿ: The solution is to take big steps or very little step while searching to solve problem.

(3) ridge: A small small peaks:
The orientation of high region compare to the sets of available move make it impossible to climb up.

Solⁿ: at the same time move in all directions and check which is better next state or which state is better.

Scanned by TapScanner

Block world problem:

Points to remember:

→ Add one point for every block i.e. resting on the thing it is suppose to be resting on it.

→ Subtract one point for every block i.e. sitting on the wrong thing

Rules
→ not more than three blocks on ground.
→ at a time one block can move.

G.S:

H
G
F
E
D
C
B
A

h(GS) = 8

I.S:

A	−1
H	+1
G	+1
F	+1
E	+1
D	+1
C	+1
B	−1

h(IS) = 4

+1	H
+1	G
+1	F
+1	E
+1	D
+1	C
−1	B

A	+1

h(NS) = 6

So, we will quit the game with failure.

There are total 3 possibility for Next state but all three are not better more

* → optimal solution

use in

A Algorithm:

- A^* is a corner stone name of many AI system.
- It is based on Best First Search algorithm.
- A^* search finds the shortest path through a search space to a goal state using heuristic function.
- This technique finds minimal cost solutions to reach to a goal state.
- In A^* the symbol * represent optimality
- A^* requires heuristic function to evaluate the cost of path that passes through the particular state.

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

where,

$g(n)$ = actual cost to reach n.

$h'(n)$ = estimate of cost to reach goal from n.

$f'(n)$ = estimate of total cost along path through n.

Scanned by TapScanner

Date
Page

Best First Search: DFS & BFS

[most promising
a model successor]

OR graph → multiple path to solve problem but we want optimal
→ maintain list
→ easily track by parent node

Scanned by TapScanner

$f' = g + h'$

where

g = cost of getting from the initial state to the current node.

h' = an estimate of the additional cost of getting from the current node to a goal state.

We need two list of nodes: list that through out problem we have to use

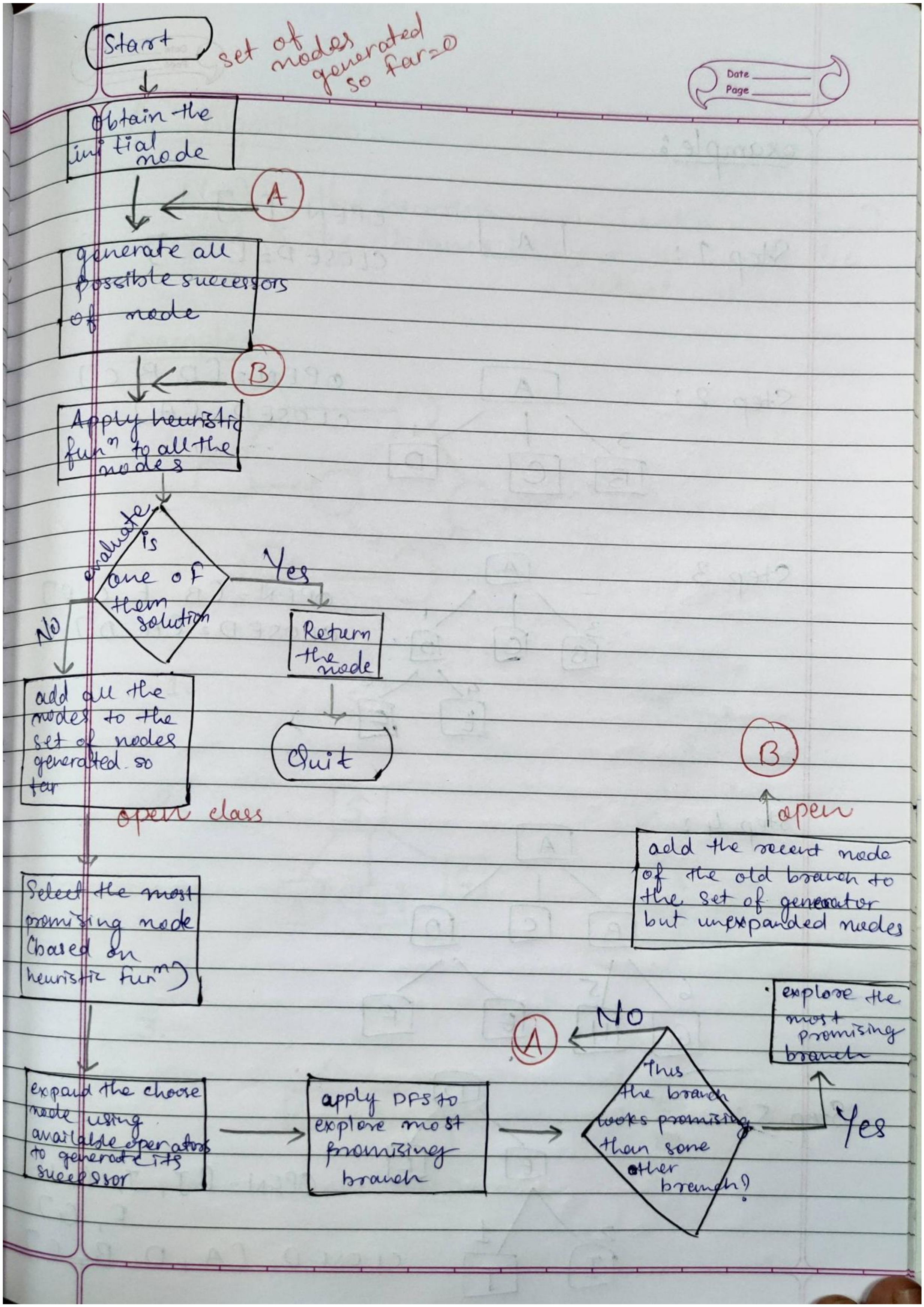
OPEN:

nodes that have been generated and have the heuristic function applied to them but which have not yet been examine.

CLOSED:

nodes that have already been examine.

Scanned by TapScanner

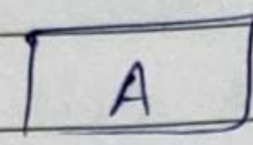


Scanned by TapScanner

Date
Page

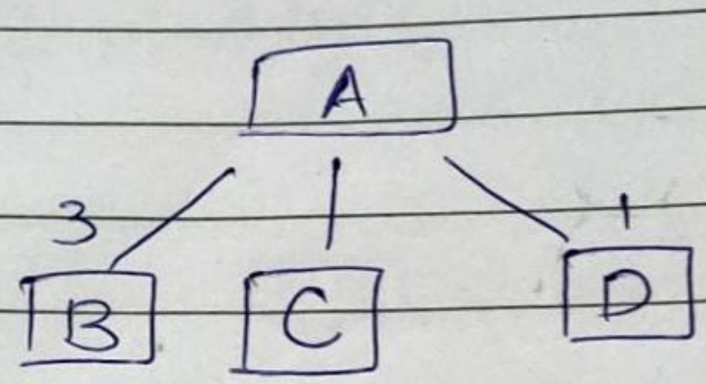
example:

Step 1:



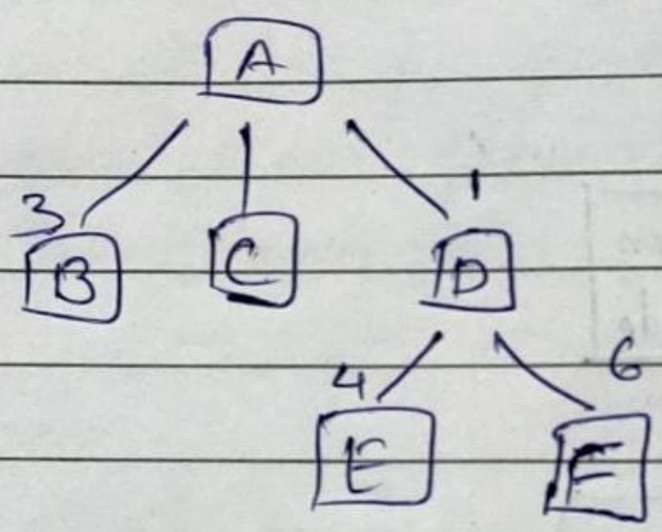
OPEN = [A]
CLOSED = [ ]

Step 2:



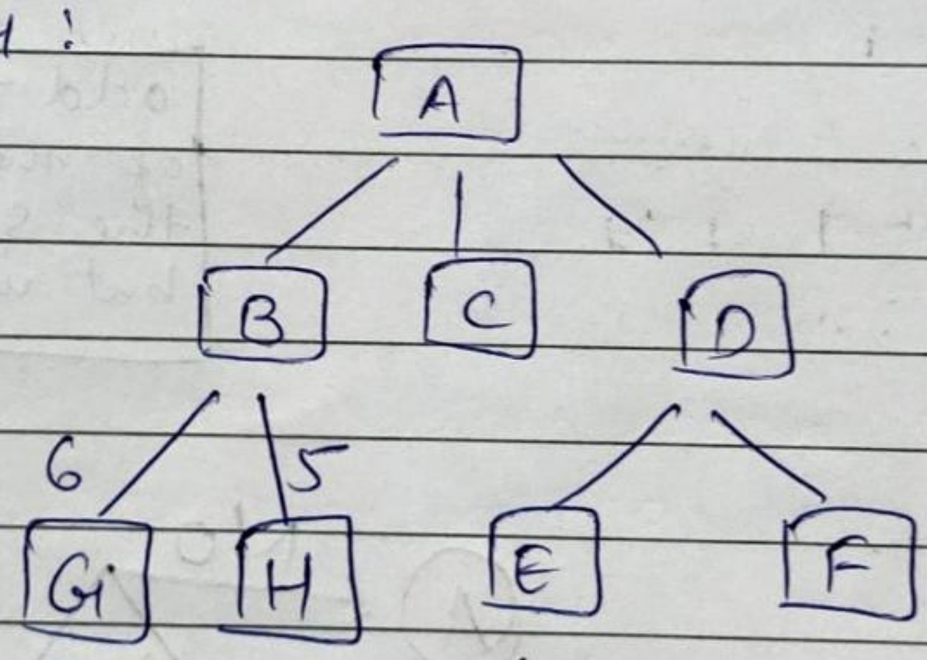
OPEN = [D B C]
CLOSED = [A]

Step 3:

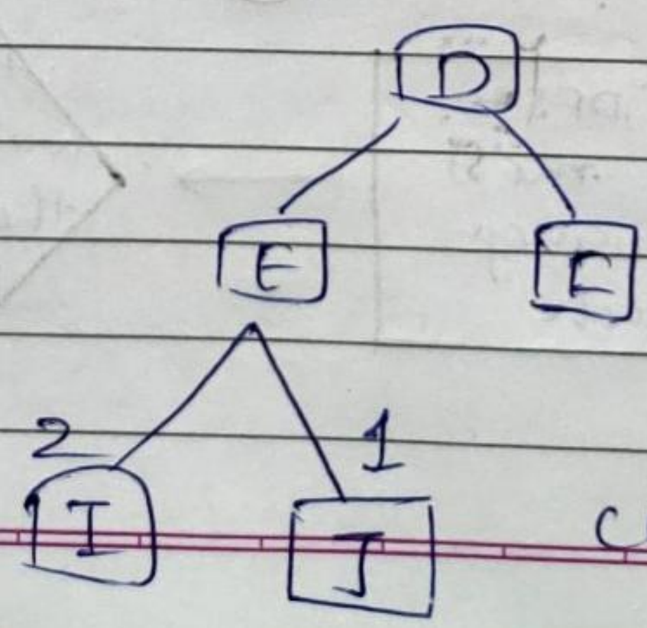


OPEN = [B E C F]
CLOSED = [A D]

Step 4:



Step 5:



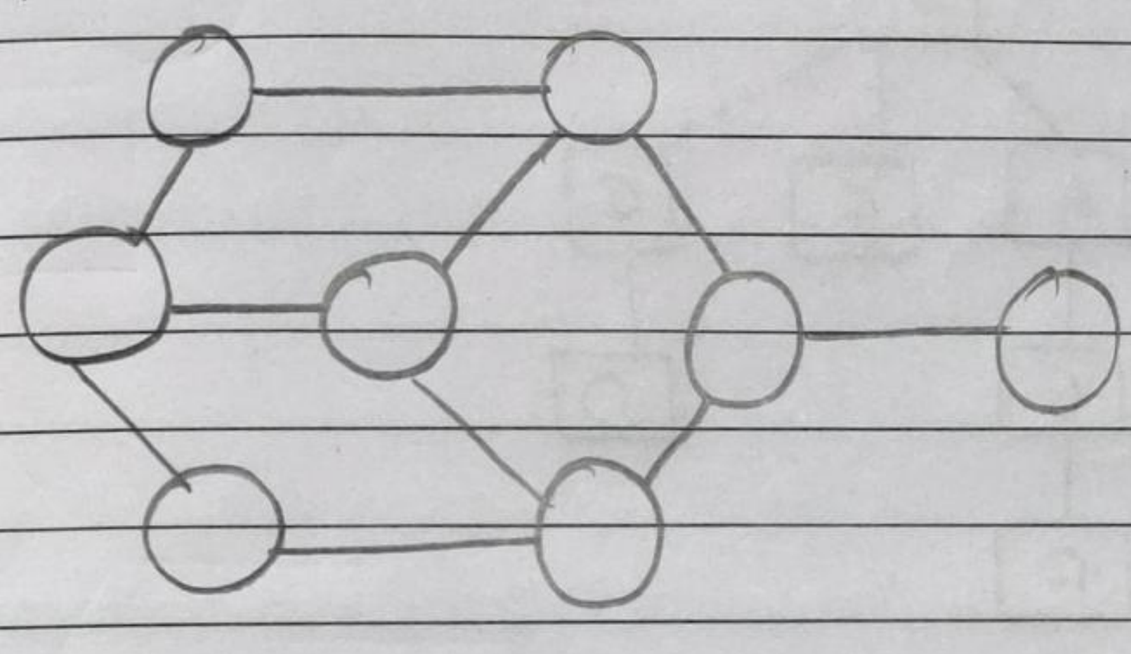
OPEN = [J, I, C, H, F, G]
CLOSED = [A, D, B, E]

Scanned by TapScanner

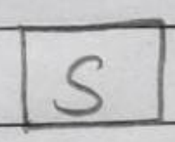
A* Algorithm:

h(n) is the heuristic value (estimate value)
g(n) is the distance between two node lie (actual value)

example:

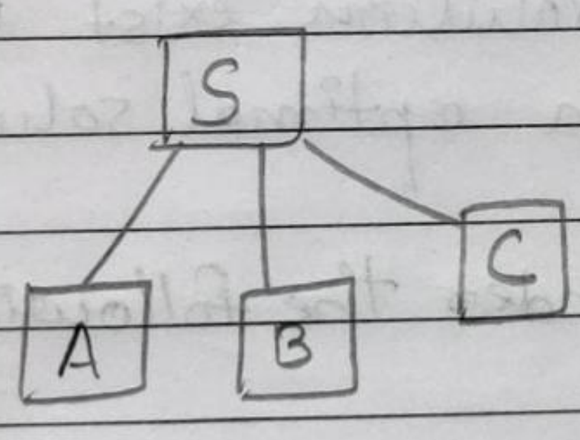


1.



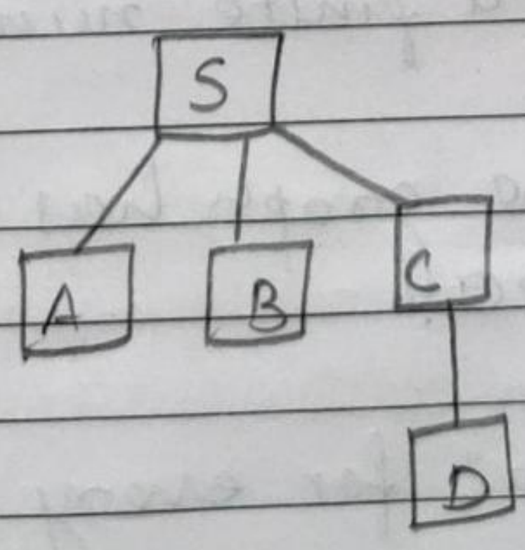
Queue
S

2.



Queue
SC
SA
SB

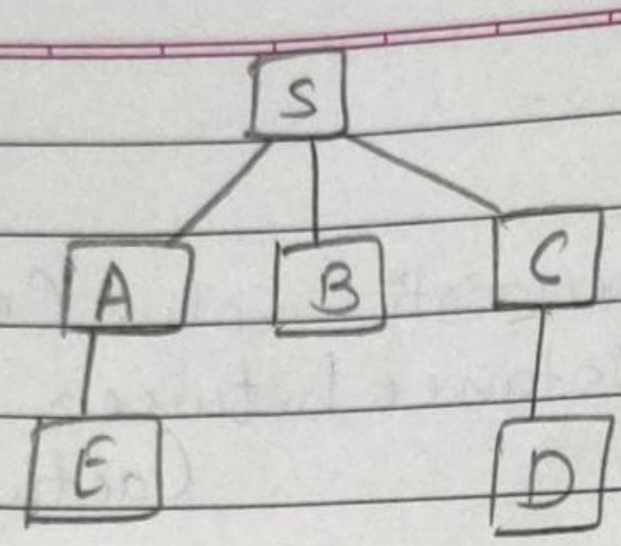
3.



Queue
SA
[redacted]
[redacted]
SCD
SB

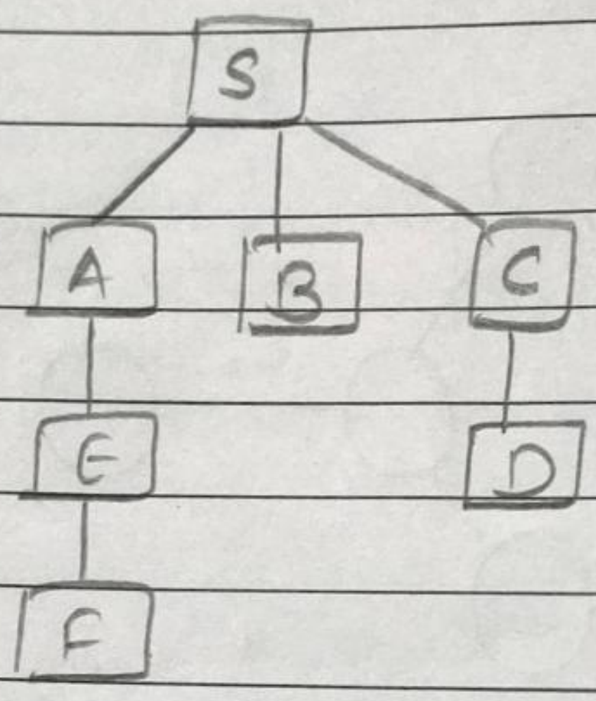
Scanned by TapScanner

4.



Queue
SAE
SCD
SB

5.



A^* search properties.

admissible → The algorithm is A^* is admissible, this means that provided a solution exist the 1st solution found by A^* is an optimal solution

A^* is admissible under the following conditions:

- In the state space graph
  1) every node has a finite number of successor.
  2) every arc in the graph has a cost greater than some $\epsilon > 0$.
  3) heuristic function: for every node small n, $h^*(n) \leq h(n)$.

Scanned by TapScanner

Complete A^* is also complete under the above condition.

Scanned by TapScanner

Knowledge Representation

Date
Page

Data
↓
Info
↓
Know
↓
Wisdom

1) Proportional / propositional logic

Characteristic

→ simple to deal
→ It is able to represent "some" real world facts (dynamic)
→ always written in capital letter.

e.g. This is a man = MAN
The sun is rising = SUNRISING
Man is mortal = MORTALMAN

Advantages

→ It is easy to represent & understand as well
→ It require less memory.

Disadvantages

→ It is very hard to distinguish between name and characteristic.

e.g.
Virat Kohli is a crickter = CRICKTER
VIRAT KOHLI

→ Fails to capture the relationship of the real world facts.

Solution: To predicate logic of 1st order predicate logic - it can represent real world facts is the relationship between the entity and other thing.

Scanned by TapScanner

e.g. The sun is rising

(Rising) (Sun)

predicate argument

e.g. Man is mortal → Mortal (man)

e.g. Virat Kohli is a crickter.
→ crickter (virat kohli).

e.g. Sun is rising in the east.
→ rising (sun, east)

e.g. Viral Kohli is left handed crickter.
→ crickter (virat kohli, left handed)

① Today is a sunny day = Today (sunny day)

② I like robotics = likes (x, robotics)

③ John is brother of Alex = brother (John, alex)

④ John loves to play guitar = love-to-play (John, guitar)

⑤ I scored II marks in AI = scored-ai (x, II)

Scanned by TapScanner

By using symbol → (Quantifiers). to quantify or calculate

universal (apply to everyone/ everybody)
symbol forall (∀)

existantial
symbol = ∃ (there exists)

other symbol:

" → " → if then condition (implication)
" ∧ " → and
" ∨ " → or
" ¬ " → not (negative (not like).

1) All man are mortal =
$\forall x : man(x) \rightarrow mortal(x)$

2) All students likes AI =
$\forall x : student(x) \wedge likes(x, ai)$
↓
represent facts not a condition

3) Ram likes to eat all kind of food.
$\forall x : food(x) \wedge likes_to_eat(ram, x)$

If all kind of food
then $\forall x : food(x) \wedge likes(ram, x)$

Scanned by TapScanner

4) All yellow mushrooms are poisonous.
$\forall x$: mushroom $(x) \wedge$ colour $(x$, yellow$) \rightarrow$ poisonous.

5) When it started raining all student carry umbrella.
$\forall x$: student $(x) \wedge$ start (raining) $\rightarrow$ carry $(x$, umbrella).

6) Some purple mushrooms are poisonous.
$\exists x$: mushroom $(x) \wedge$ colour $(x$, purple$) \rightarrow$ poisonous (x).

7) Not all students have AI book.
$\exists x$: student $(x) \wedge$ have $(x$, ai_book$)$

8) Some students do not have AI book.
$\exists x$: student $(x) \wedge \neg$ has $(x$, ai-book$)$.

9) Mohan eats peanuts and goes to school.
eats (mohan, peanuts) $\wedge$ goto (mohan, school).
eats (mohan, peanuts) $\wedge$ go-to-school (mohan)

10) Venus is near sun but does not have tail
near (venus, sun) $\wedge \neg$ has (venus, tail)

11) Anything anyone watches in multiplex is a movie.
$\forall x : \forall y$: person $(x) \wedge$ thing $(y) \wedge$ watch_in_multiplex $(x, y) \rightarrow$ movie(y)

Scanned by TapScanner

12) Alex watches terminator in multiplex.

watches_in_multiplex (x,y)
alex terminator

① Hardik Pandya is an all rounder cricketer who stays at Baroda.

→ cricketer (hardik pandya) ∧ all rounder (hardik Pandya) ∧ stay (hardik pandya, baroda)

② Everyone is loyal to someone.

→

③ Laxman eats everything that ram eats.

→ $\forall x$: food(x) ∧ eat (ram, x) → eat (lax, x)

④ Anything anyone eats and is not killed by a food.

→ $\forall x : \forall y$: person(x) ∧ thing (y) ∧ eat (x,y) ∧ ¬ is_not_killed_by (x,y) → food (y)

Scanned by TapScanner

(5) One who knows riding does not know skatting.

→ $\forall x$: person $(x) \wedge$ knows (x, riding) → $\neg$ knows (x, skatting)

(1) Jack went to school and brought maggie in lunch box.

→ go to (Jack, schoo) $\wedge$ brought_maggie (Jack, lunch box)

(2) While laughing sue fell in swimming pool and does not know swimming.

→ laughing (sue) $\wedge$ fell in swimming pool (Sue) $\wedge \neg$ knows (sue, swimming)

(3) All those who did not prepare for the test are absent in the class.

→ All x (person) $\wedge \neg$ prepare (x, test) → absent (x, class)

(4)

Scanned by TapScanner

Story

Date
Page

(1) Ram was mighty a king.
→ king (Ram, mighty).

(2) Ram has a crown on his head.
→ has_crown (Ram, head)

(3) The crown has beautiful multicolour gems on it.
(multicolour, gem) ∧ (beautiful, gem) · (gem, on_a_crown)

(4) All courtiers are attracted towards ram's crown.
→ All (courtiers) ∀x : courtiers (x) ∧ attracted_towards (x, crown)

(5) Ram slept in night and next morning found his crown missing.
→ x : (slept, night) ∧ time (next_morning) ∧ ¬ find (

(6) Ram's instruct the chief minister to investigate
→ instruct (Ram, chief minister) ∧ chief_minister (investigate)

(7) The chief minister suspect three courtiers Rohan, Shyam and Mohan.
→ courtiers (Ram) ∧ courtiers (Shyam) ∧ courtiers (Mohan) ∧ suspect (chief minister, Ram) ∧ suspect (chief minister, Shyam) ∧ suspect (chief minister, Mohan)

Scanned by TapScanner