AI / ML

Agent
An autonomous entity which act when an environment using a sensor and actuator.

Actuator
Component of machine which is responsible for moving and controlling a system or some mechanism.

Goals of Agent
1. High Performance
2. Optimized Results
3. Rational Action.

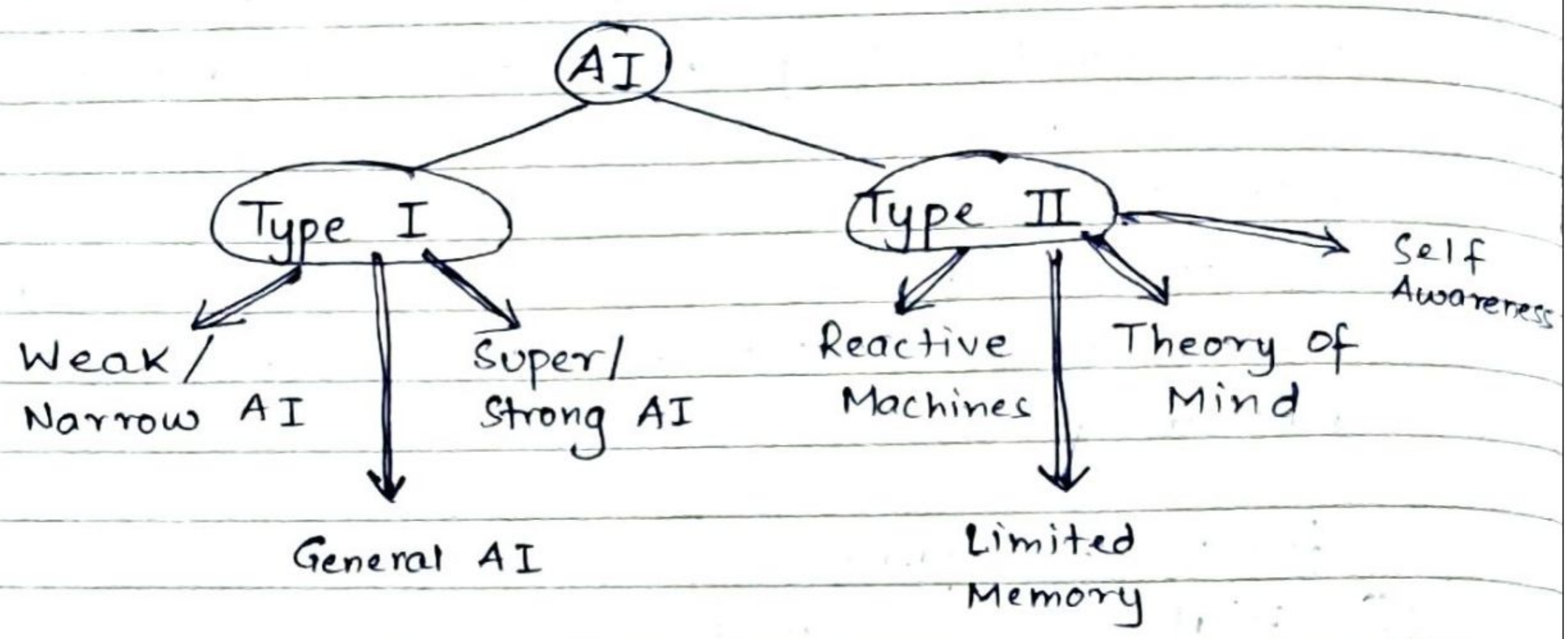
Parameters for Categories of Agent.
1. Performance
2. Environment
3. Actions
4. Sensors

} PEAS

Applications of AI.
1. HealthCare
2. Entertainment
3. Social Media
4. Travel and Transport
5. Data Security
6. E-Commerce
7. Gaming.
8. Finance.
9. Education
10. Agriculture.

Scanned by TapScanner

Types of AI.



Imp

Types of Agents

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

19/01/2023

(1) **SIMPLE REFLEX AGENTS**

1. These agents work in fully observable enviornment. The simple reflex agents take decisions on the basis of current percepts and ignore rest of the percept history.

2. The Simple Reflex agent work on condition - action rule which mean it maps the current State to action.

Cons/ Problems

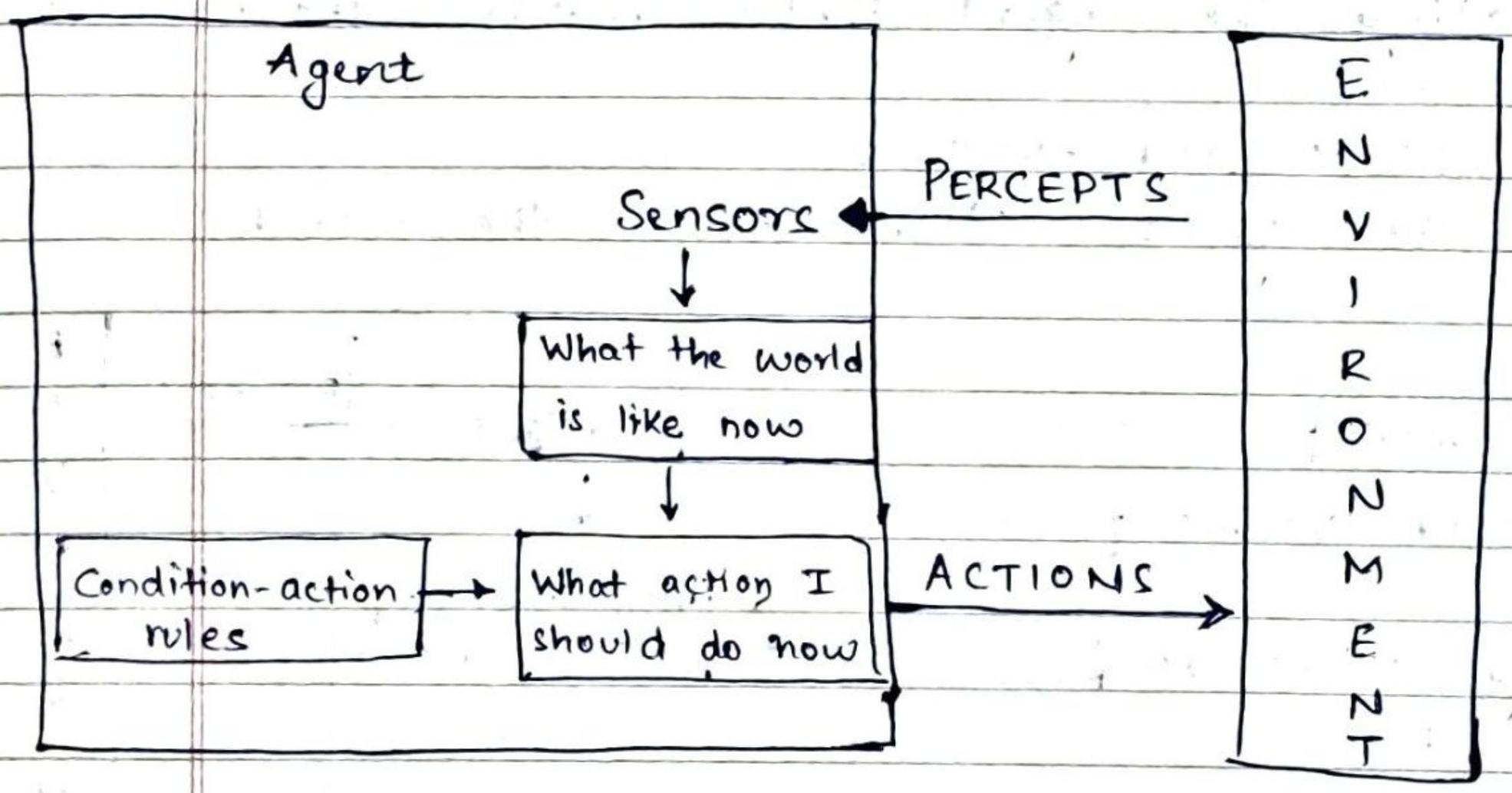
Problems with these agents are as follows :-

Scanned by TapScanner

1. They have limited intellegence.
2. They don't use any previous history/information/learning.
3. Mostly the bigger size and specific task is problem with these agents.
4. These agents are not flexible enough for any chances in enviornment.

<u>Example</u>:

1. Dirt Cleaner.



Scanned by TapScanner

REFLEX

(2) MODEL BASED AGENTS

These agents work in partially observable enviornment so they are able to track the situation.

These agents have two important factors :-

(1) MODEL

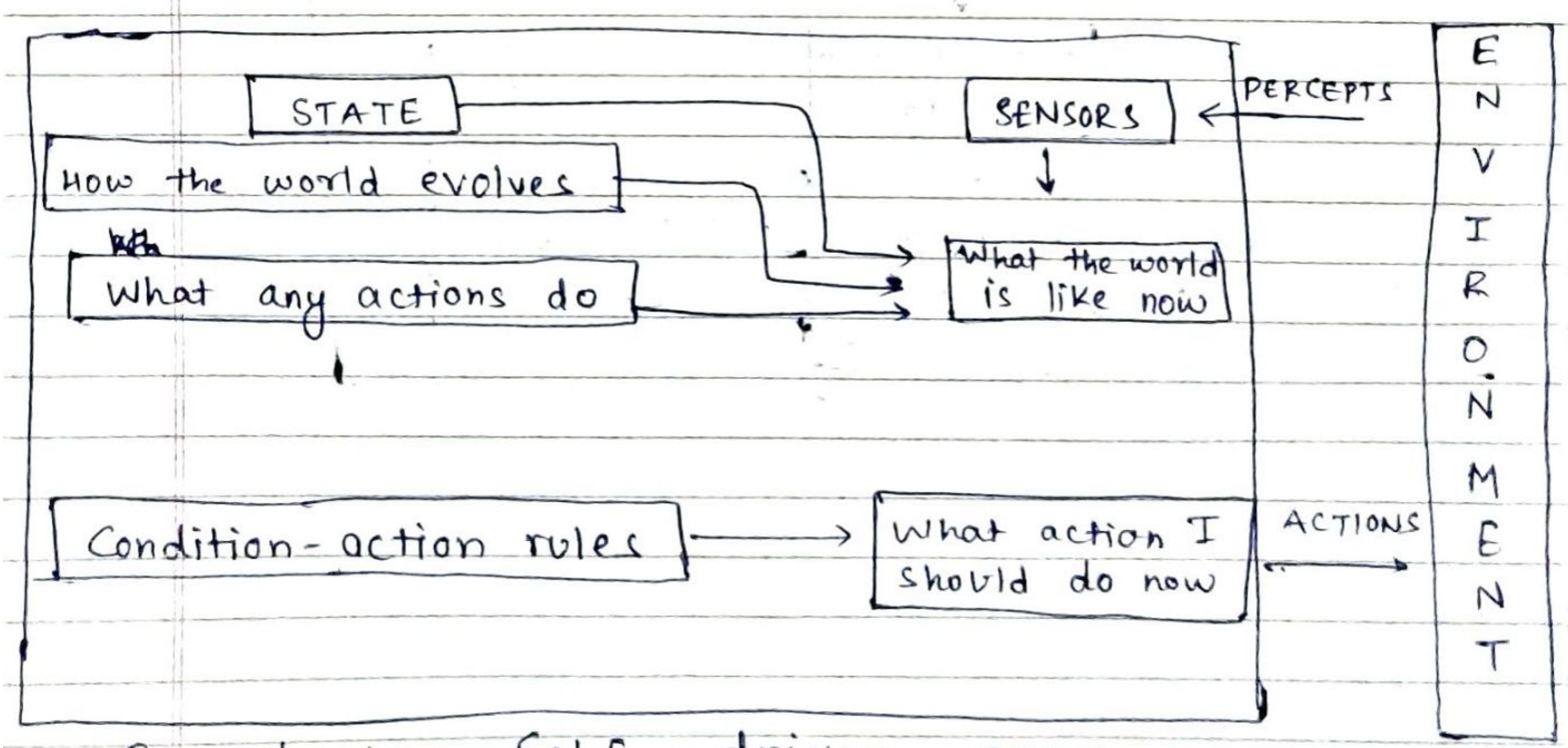
It is the Knowledge about "How things happen in the world", so it is called a model based agent

(2) INTERNAL STATE

It is the representation of the current state based on percept history.

Updating the agent state requires information about -

1) How the world evolves
2) How the agents action affects the world



Example :- Self-driven cars

Scanned by TapScanner

(3) GOAL BASED AGENTS.

① The Knowledge of the current state enviornment is not always sufficient to decide for an agent to what to do.

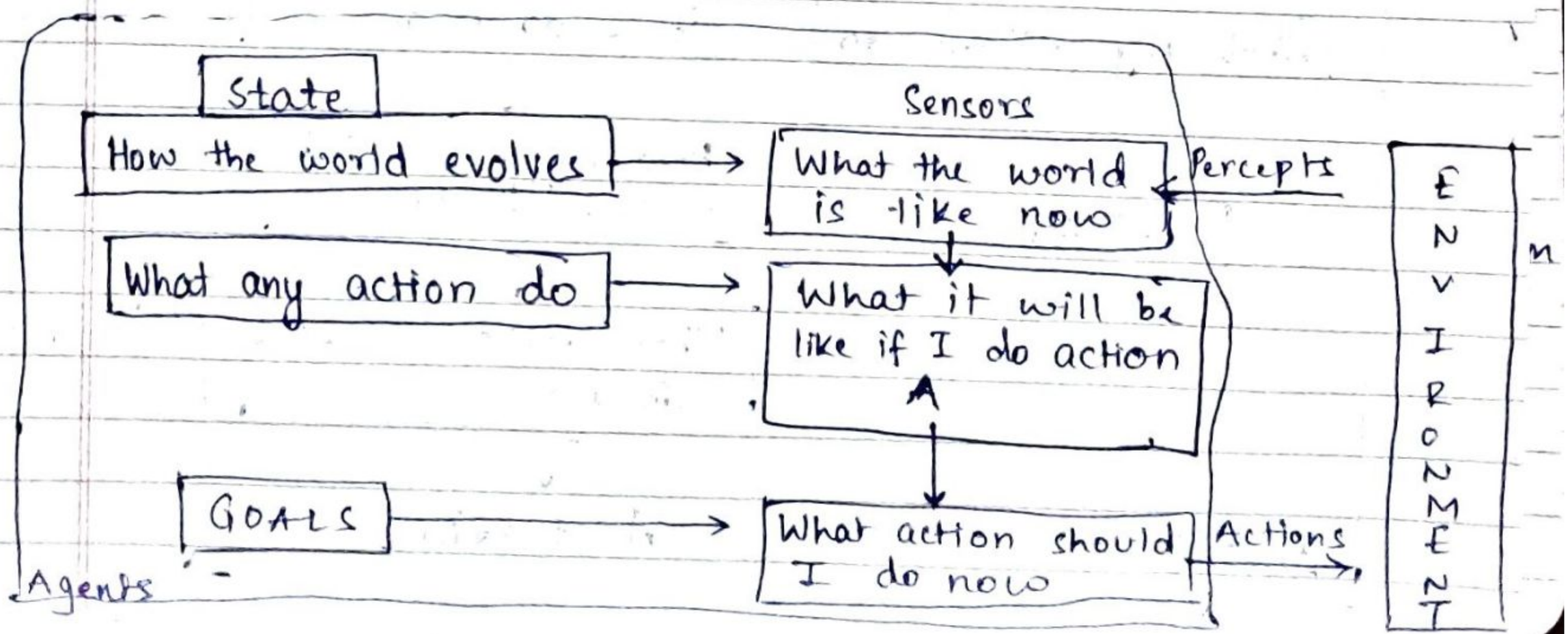
② The agents needs to know its goal which describes desirable situation,

③ Goal based agents expands the capability of the model based agents by having the information of "Goal".

④ They choose an action so that they can achieve the goal.

IMP ⑤ These agents have to consider a long sequence of possible action before deciding whether the goal is achieved or not. Such condition of different scenario are called Searching and Planning.

⑥ G robots for home delivery are used by Alibaba.



Scanned by TapScanner

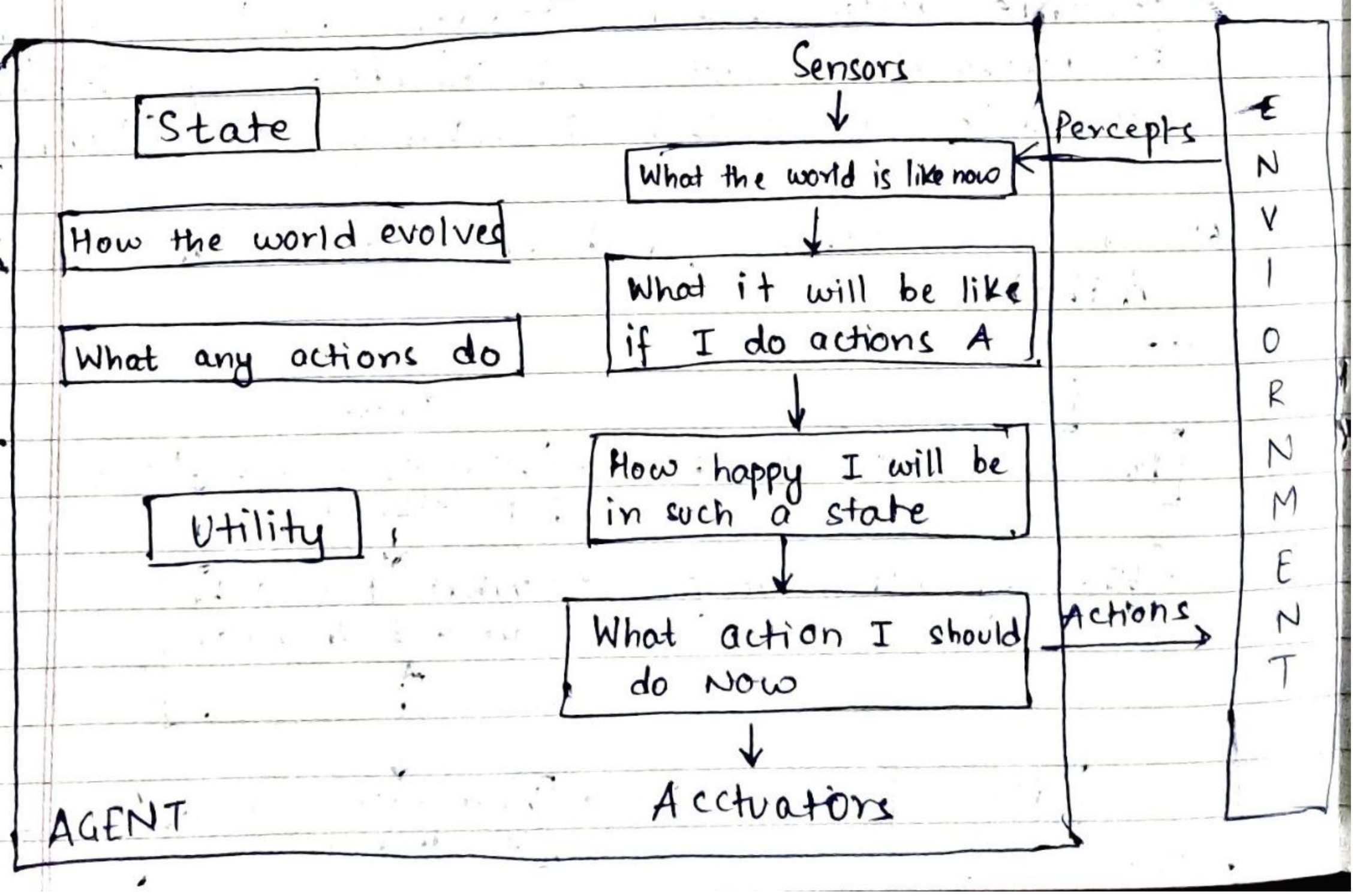
(4) Utility Based Agents

① These agents are the expansion in model based agent by providing one extra component of Utility. which makes them different by providing a measure of success at the given state.

② Utility based agent act based not only on goals but also the best way to achieve the goal.

③ These agente are more useful when their are multiple possible alternatives and agent is expected to chose the best one.

④ These agents match each state to a Real number to check how efficiently they have taken the action to achieve the goal.



Scanned by TapScanner

(5) <u>Learning Agent</u>

A learning agent in AI is the type of agent which can learn from its past experiences or it has the learning capabilities.

IMP It is start to act with basic knowledge and then able to act and adapt automatically through learning.

A learning agent has mainly four components as follows :-

1) Learning Element
It is responsible for making improvements by learning from enviornment.

2) Critic
Learning Element takes feedback from critic which describes that how well the agent is doing with respect to a fixed performance standard.

3) Performance Element
It is responsible for selecting external action.

4) Problem Generator
This component is responsible for suggesting actions that will lead to new and informative experience

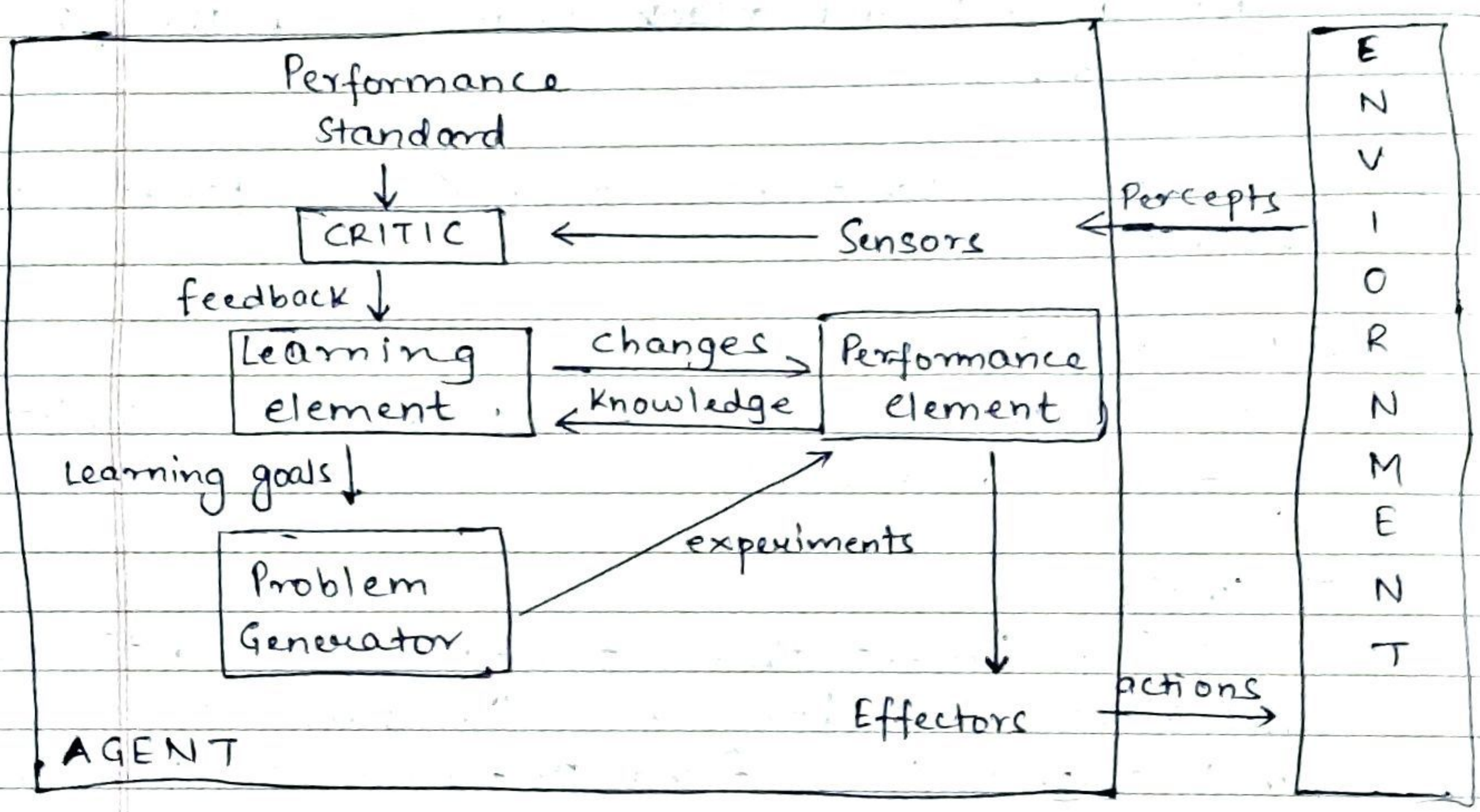
Thus learning agents can learn, analyze and look new ways to improve the performance.

Scanned by TapScanner

~~State~~

~~How the world evolves~~

~~What any actions do~~



Scanned by TapScanner

1) Fully Observable V/S Partially Observable.

2) Determinastic V/S Stiochastic.

3) Competitive V/S Collaborative.

4) Single Agent V/S Multi-Agent.

5) Dynamic V/S Static

6) Discrete V/S Continuous.

7) Episodic V/S Sequential

8) Known V/S Unknown

Imp 9) State Space Search

★★ S : { S , A , Action(S), Result(S, a), Cost(S, a)}

S = Total number of States (Start, intermediate, Goal)

A = Set of Actions / Possible Actions

★ Note : (1) Define a problem precisiely, (2) analyze the problem
(3) Take necessary action (use of knowledge)
(4) Choose the best ~~option~~ action.

~~Represe~~

Scanned by TapScanner

Representing a problem in State Space Search

1. Define the problem in formal language.
2. Stating operations (initial, intermediate, desired condition)
3. Goal State (result of search technique)

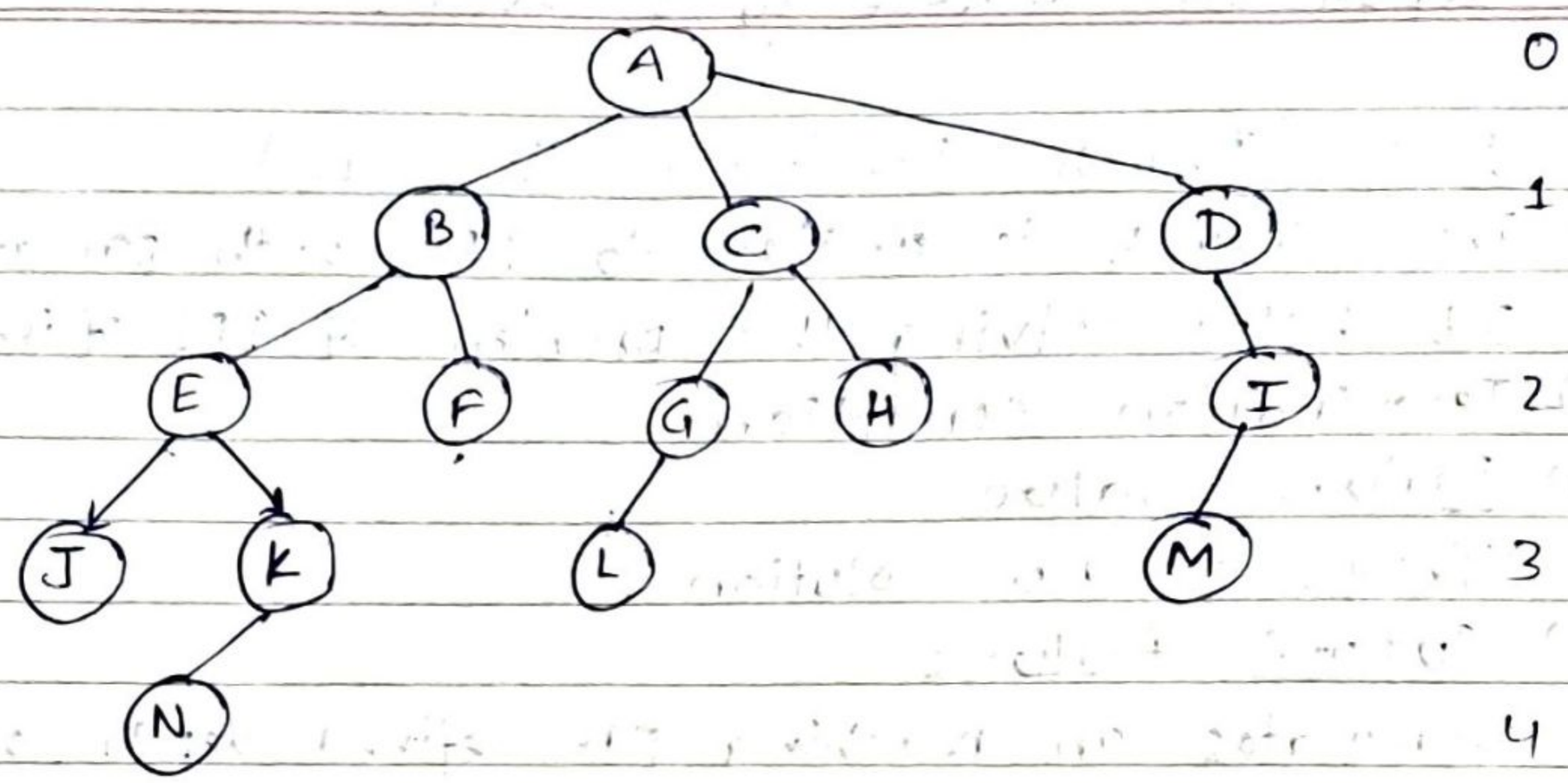
Searching Techniques

Uninformed Search	Informed Search
(1) Search without information	(1) Search with information
(2) No knowledge	(2) With some knowledge
(3) More time consuming	(3) Less time consuming
(4) More complexity (Time, Space)	(4) Less complexity
(5) Eg. BFS, DFS, TSP,	(5) Eg. Theoristic Approach, Greedy Approach

(A) BFS (Breadth First Search) or Level Order or Level Search Technique.

1. Uninformed Search
2. Uses Queue. (FIFO)
3. Shallowest node.
4. Complete
5. Optimal
6. Time Complexity : b^d b: branch factor, d: depth

Scanned by TapScanner



~~A~~	~~B~~	~~C~~	~~D~~	~~E~~	~~F~~	~~G~~	~~H~~	~~I~~	~~J~~	~~K~~	~~L~~	~~M~~	~~N~~

A B C D E F G H I J K L M N

↑

Maximum Branches : 3

Depth of G : 2

Time complexity to find G = 3^2

(B) <u>DFS (Depth First Search)</u>

(1) Uninformed Search

(2) Uses stack (LIFO)

(3) Deepest Node

(4) Incomplete (maybe infinite in cyclic graph or maybe infinite if search space is unlimited

(5) Non-optimal

(6) Time Complexity : b^d

Scanned by TapScanner

** Iterative Deeping
similar to DLS, but in each iteration depth will inverse by 1 level.

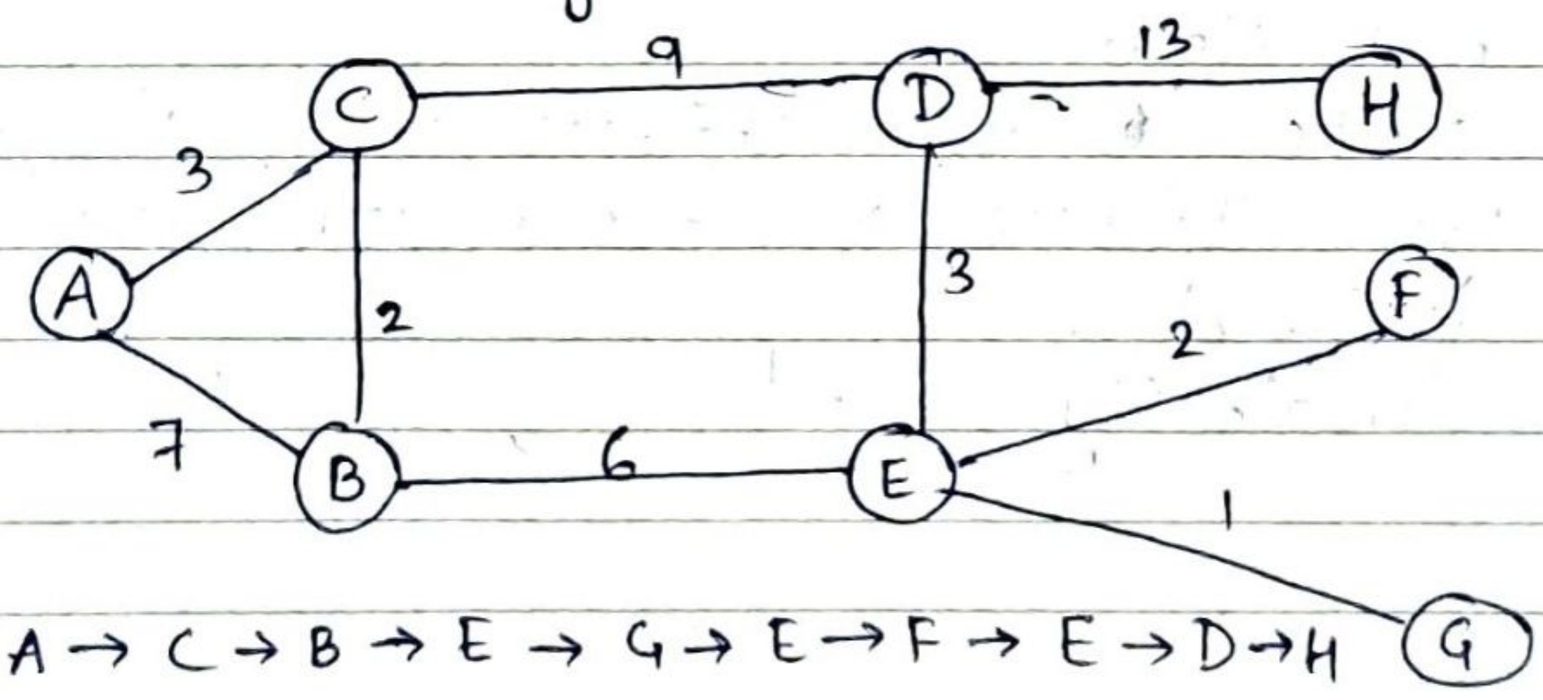
(C) DLS (Depth Limited Search)

(1) This search is similar to DFS with pre-defined limit.
(2) It helps solving the problem of DFS of infinite loop.
(3) Termination condition.
(a) Failure value
There is no solution.
(b) Cutoff failure
Terminates on reaching pre-defined depth so the solution is not ~~guaranteed~~ guaranteed.
(4) This algorithm is memory efficient.
(5) Not an optimal solution provider
(6) Incomplete.

**

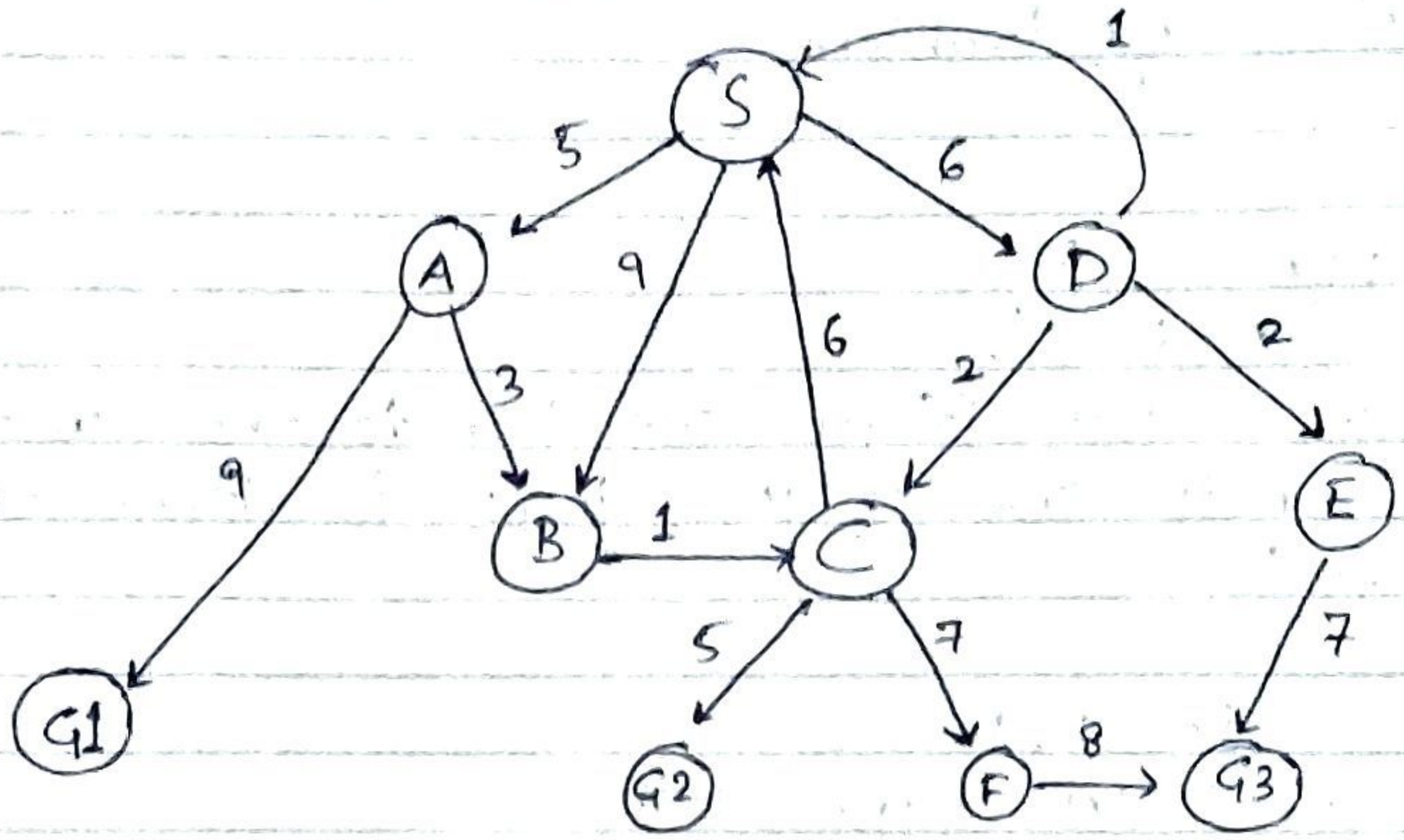
(D) Cheapest First Search ~~First~~ (Uniform cost search)

(1) Used for weighted tree / graphs traversal.
(2) Optimal Path (Lowest cost).
(3) Backtracking
(4) Node expansion is based on path cost.
(5) Priority queue is used for implementation. Highest priority is given to the lowest cost
(6) It provides optimal solution
(7) May stuck in infinite loop.
(8) It does not care about the number of steps to search the goal node.



A → C → B → E → G → E → F → E → D → H

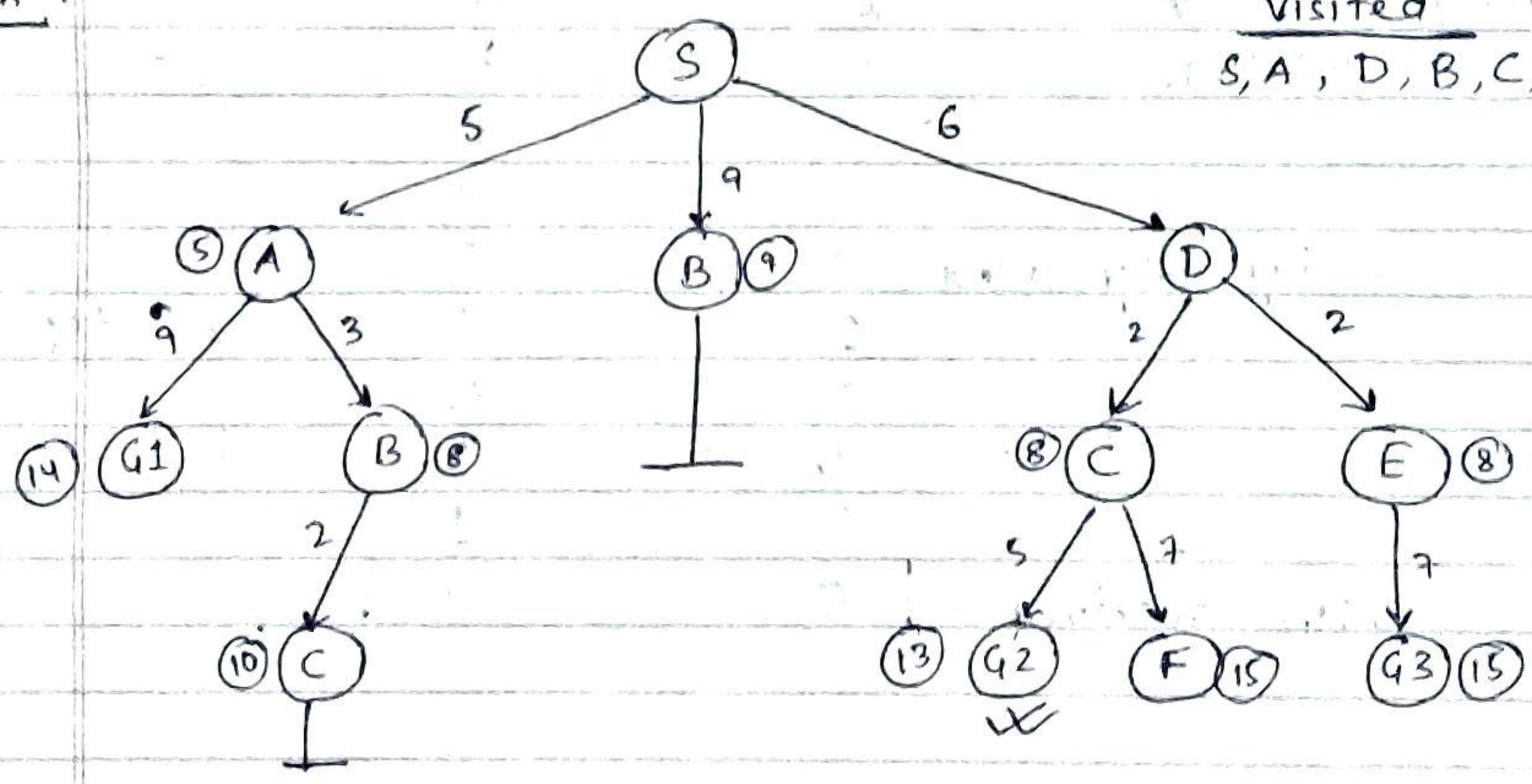
Scanned by TapScanner



Soln :

Visited

S, A , D , B , C , E



Scanned by TapScanner

Water Jug Problem

Rules :

(1) Empty a jug
(2) Fill a jug
(3) Pour water from one jug to the another jug until one of the jugs is either empty or full.

Example $d = 4$ ltrs
$x = 3$ Ltrs $y = 5$ Ltrs

Case 1

	x	y
Initial	0	0
	3	0
	0	3
x jug fully filled	3	3
	1	5
	1	0
	0	1
	3	1
4 ltrs received	0	4

8 Steps.

Case 2

	x	y
Initial	0	0
	0	5
	3	2
	0	2
	2	0
	2	5
4 ltrs achieved	3	4

6 steps.

Scanned by TapScanner

Questions

(1) Implement water jug problem using DFS
(2) Represent water jug problem in state space search.
(3) Can water jug problem be implemented by DFS If yes or NO then why ?

A^* Algorithm

(1) Find shortest path efficiently

(2) Optimal Path

(3) Extension in Dijkstra Algorithm.

(4) Heuristic approach (inform search)

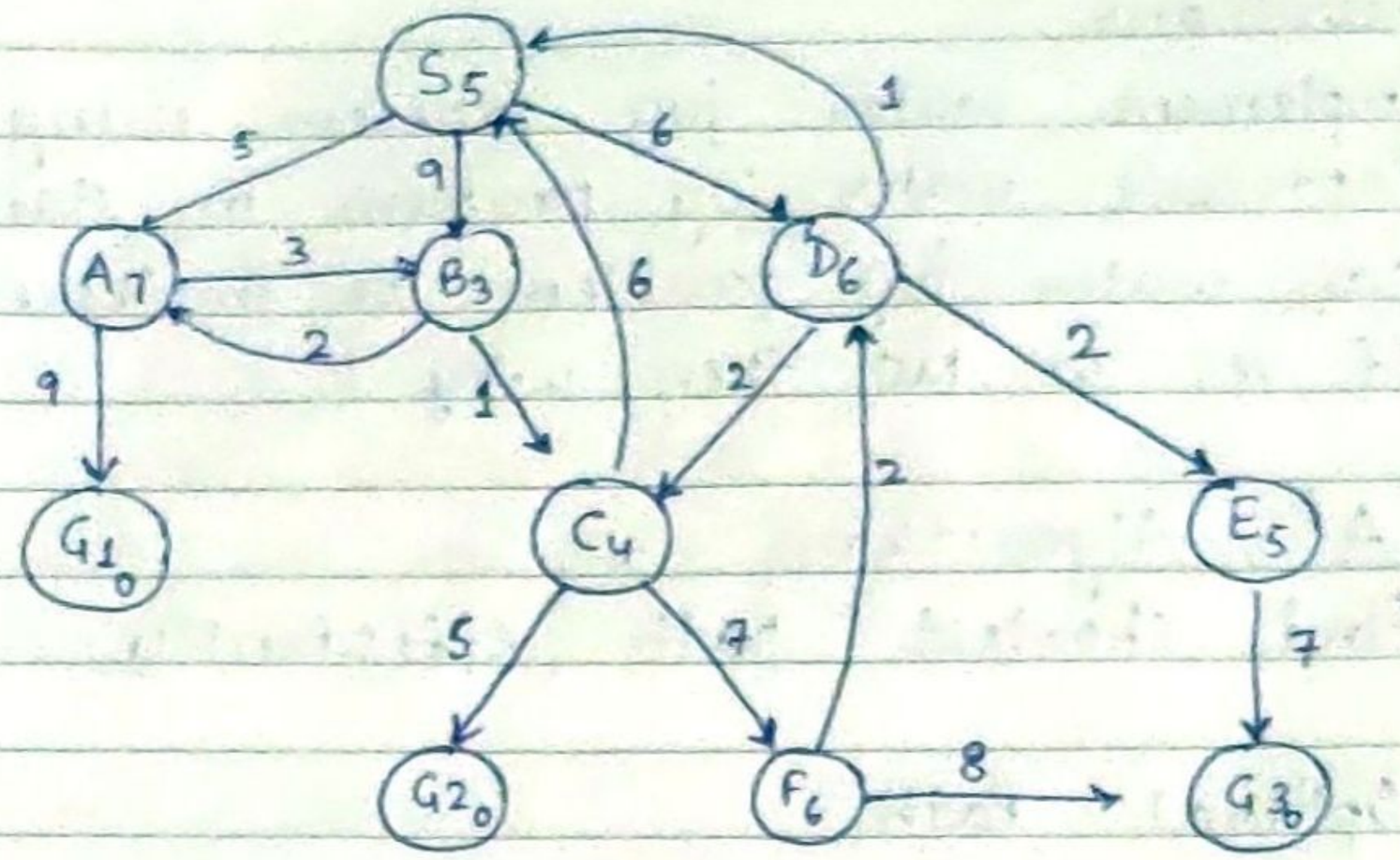
(5) In A^* algo, star (aestric *) tells admissible algorithm, which means A^* takes guarantee to provide optimal solution.

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

$g(n) \Rightarrow$ Actual cost from start node to n.

$h(n) \Rightarrow$ Estimated cost from n to goal node.

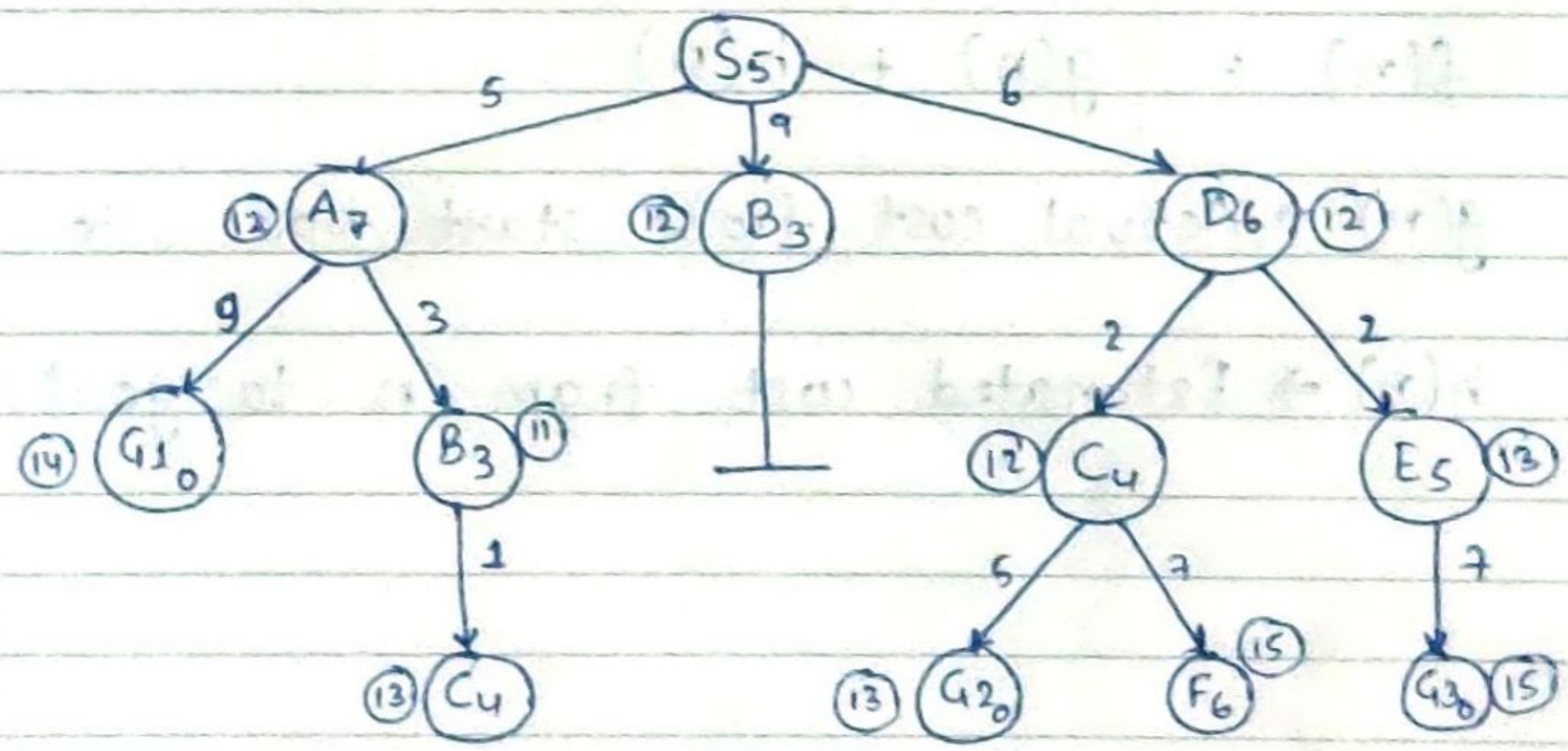
Scanned by TapScanner



From D to G_3 total cost is 9 whereas D tells the cost 6 so this shows "underestimate" to find the shortest path algo must use a heuristic that never overestimates the cost.

visited

S(5), A(12), B(11), D(12), C(12), E(13), G2(13)



Scanned by TapScanner

Best First Search

(1) This is the combination of BFS and DFS.
(2) Follows greedy approach.
(3) It provides good solution but may or may not provide optimal solution.

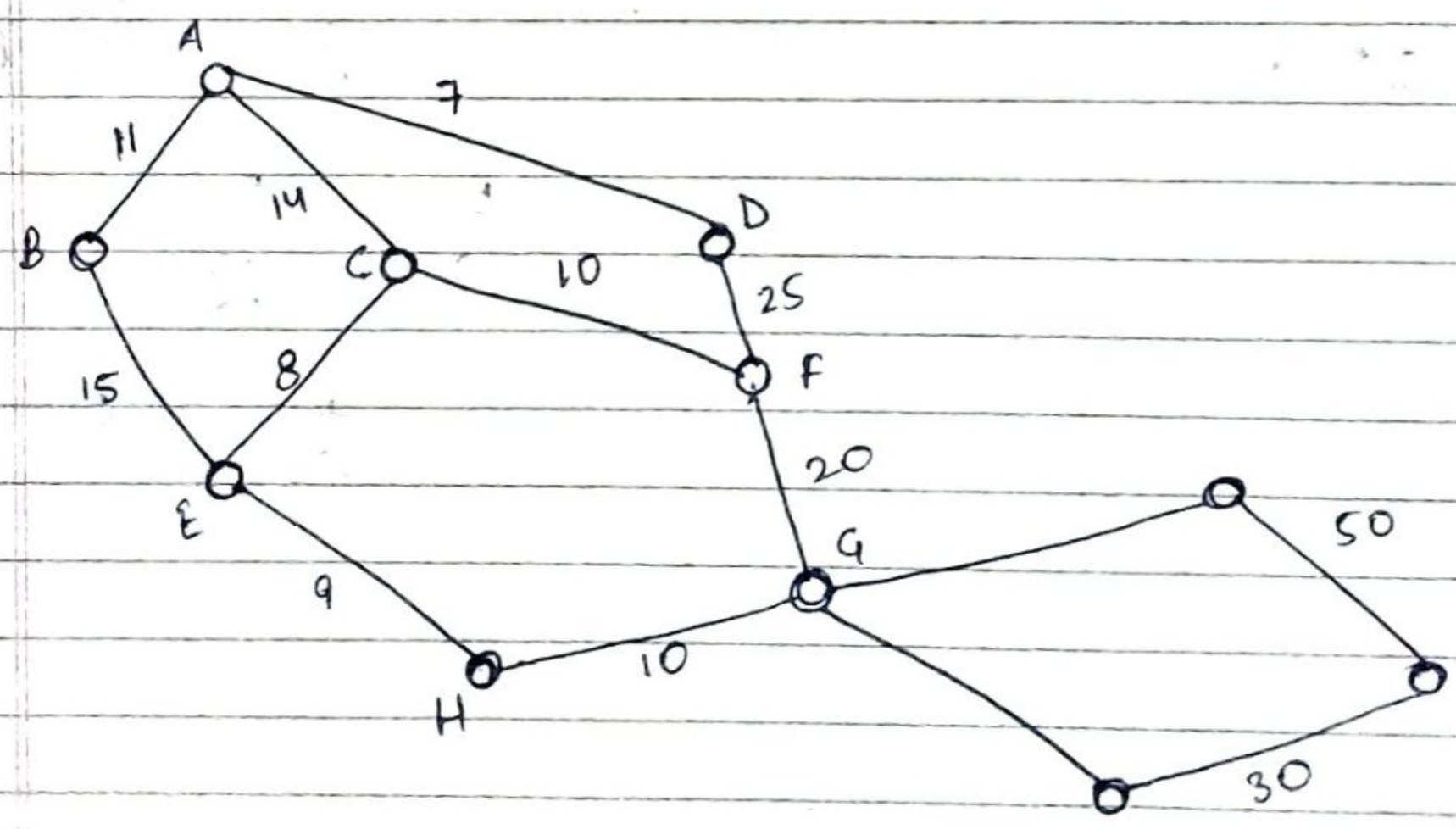
<u>Algorithm</u>

(1) Let "open" be the a priority queue containing initial state.
(2) LOOP → If "open" is empty return failure.
(3) Insert first node in "open" and name the "node".

(4) If "node" is a ~~goal~~ "goal" return the path from initial state to "node". else generate all successors of "node" and insert them into "open" (according to there 'F' value)

(5) END LOOP

BFS is informed search (heuristic approach)



Scanned by TapScanner

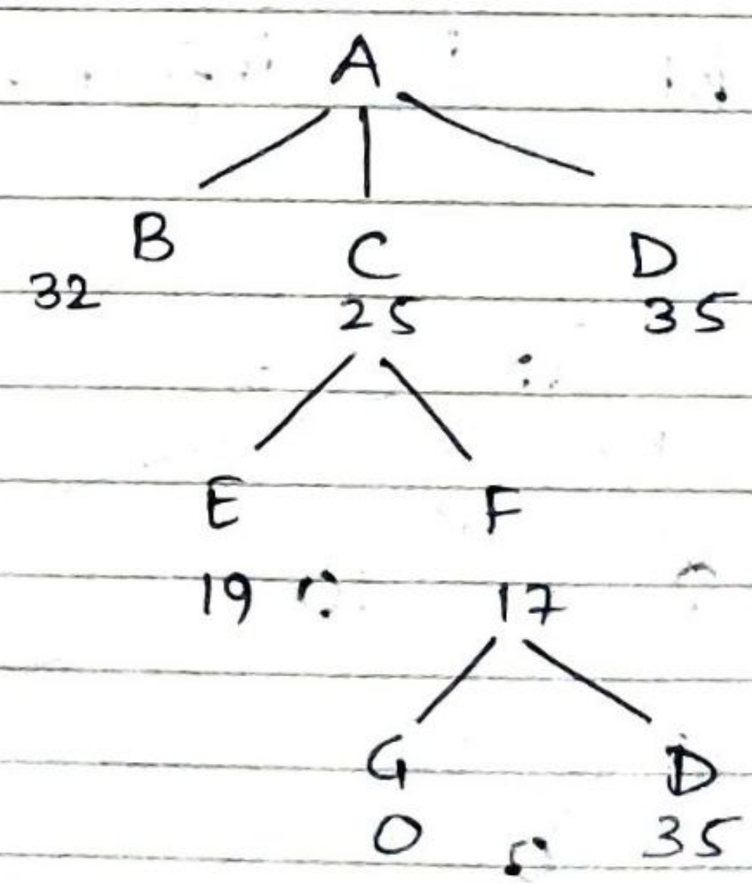
Straight Line Distance

A → G	40
B → G	32
C → G	25
D → G	35
E → G	19
F → G	17
H → G	10
G → G	0

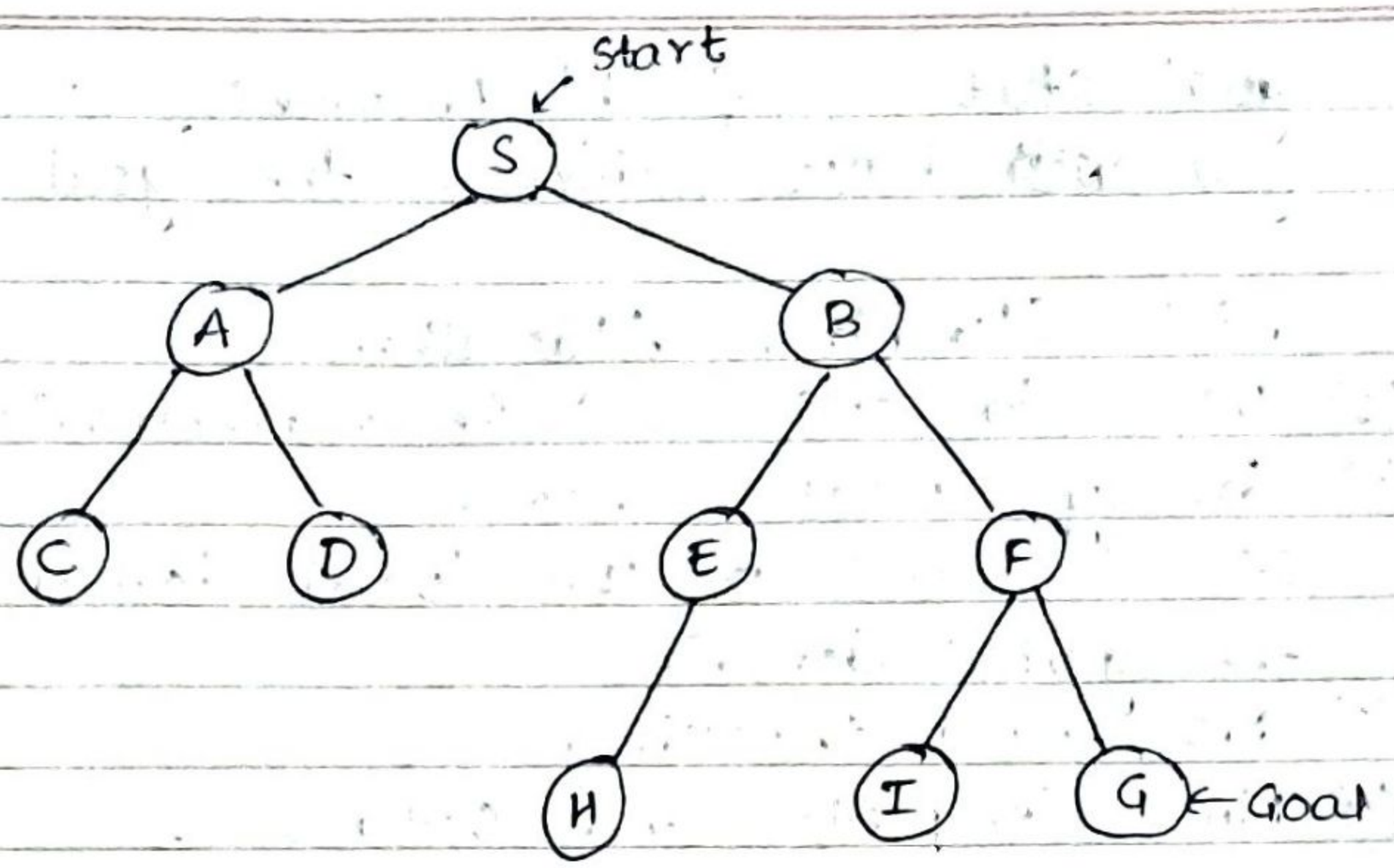
∴ open

A	C	B	D	F	E

ACFG



Scanned by TapScanner



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

Apply state space search to solve 8 puzzle problem using A^* Algorithm.

2	8	3
1	6	4
7		5

1	2	3
8		4
7	6	5

Imp

'g' will be the depth and 'h' will be the difference in between the current state and the

Scanned by TapScanner

goal state. If 'h' becomes 0 it mean that you have achieved the goal state.

Hill Climbing Algorithm.

(1) Used to optimize the mathematical problem.
(2) Greedy Local Search
It will search in the local domain but not in the global domain
(3) NO back tracking
(4) Not Complete and Optimal.

Algorithm.

(1) Take the problem as input
(2) Local variables are current and neighbour
(3) Current ← Make-node (Initial State (Problem))
(4) Loop until goal state is achieved or no more operators to apply on current state.
(5) do

```
neighbour ← a highest valued successor of
                                current state
if value [ neighbour ] ≤ value [current]
then return state [current]
else
    current ← neighbour
```

Types Of Hill-Climbing

1) Simple
2) Sterepest Ascent
3) Stiochastic

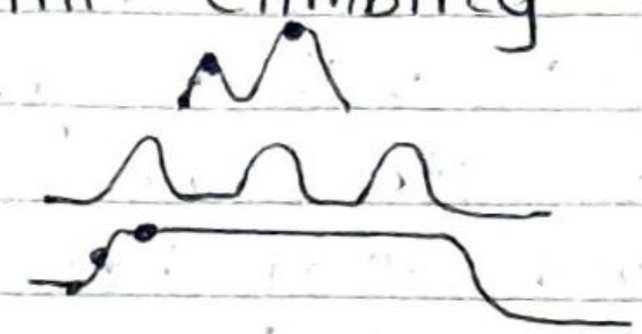
Scanned by TapScanner

Problem with Hill - Climbing

(1) Local maxima

(2) Ridge

(3) Plateau



Hill Climbing : Local heuristic problem.

(1) (+1) for each block that is resting upon the right block as it was supposed to be (-1) for each block that is resting upon the wrong block.

Initial

A
D
C
B

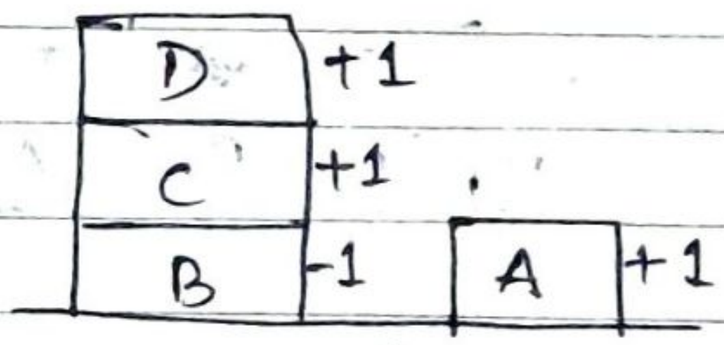
Score → 0

Goal

D
C
B
A

Score → 4

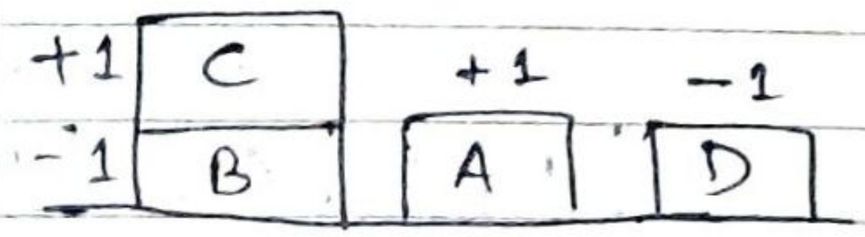
Case 2



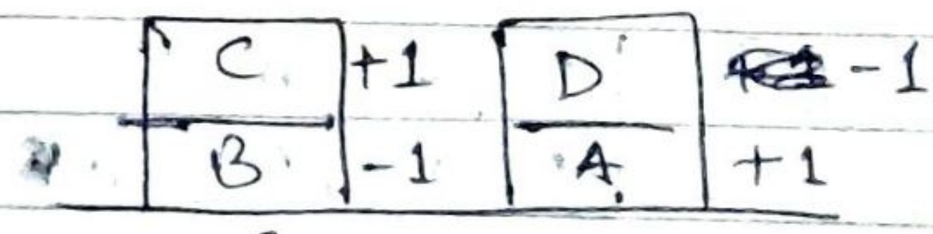
Score → 2

Case 3

a)



Score → 0



Score → 0

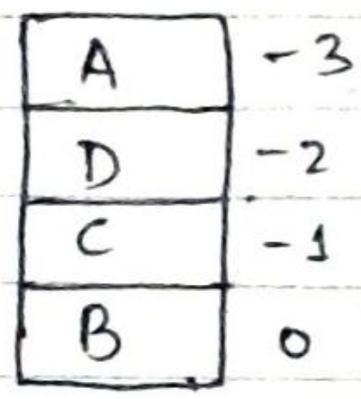
Scanned by TapScanner

Global heuristic :

For each block that has the correct support structure (+1) will be given to every block in the support structure.

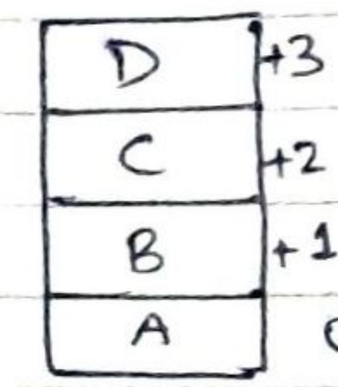
For each block that has the wrong support structure (-1) will be given to every block in the support structure

Initial



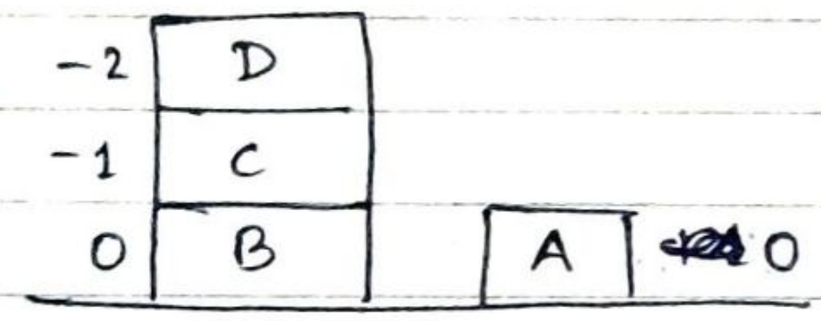
Score → -6

Goal



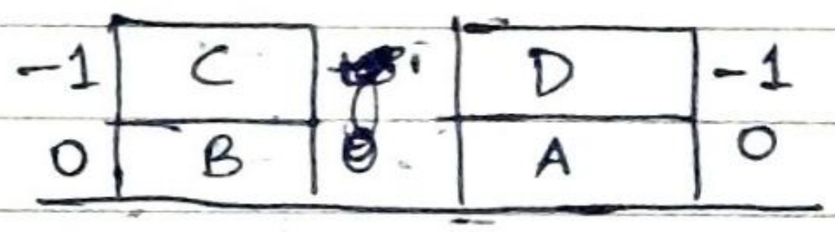
Score → 6

Case 1 :



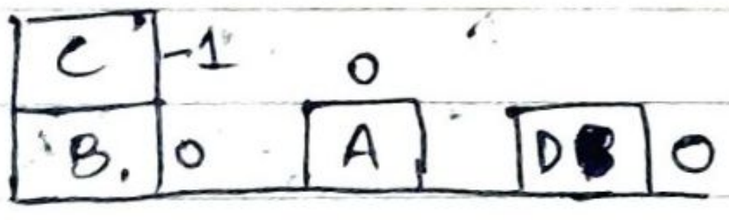
Score → -3

Case 2 :



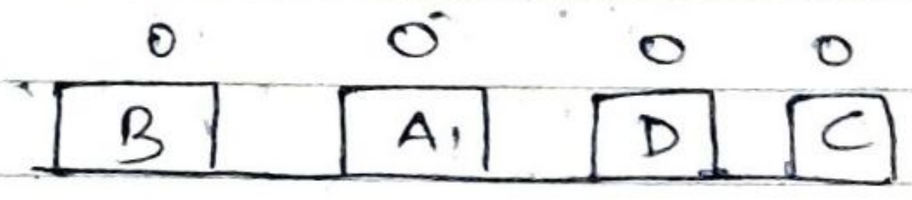
Score → -2

Case 3 :



Score → -1.

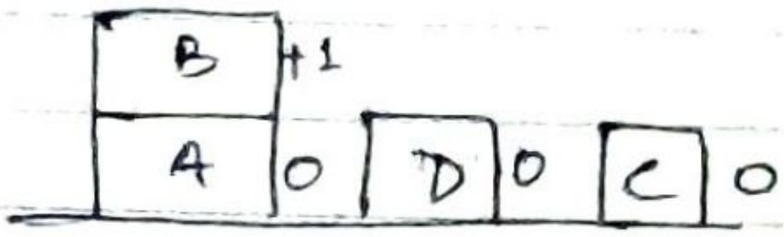
Case 4 :



Score → 0

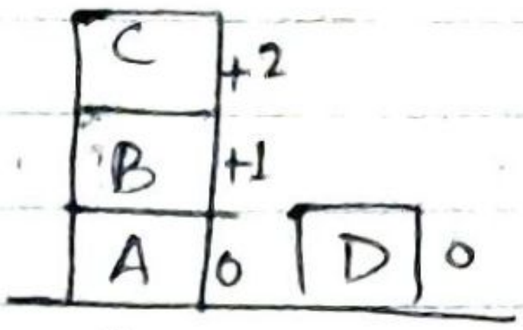
Scanned by TapScanner

Case 5 :



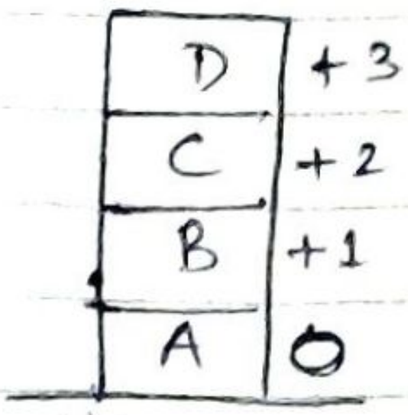
Score → +1 .

Case 6 :



Score → +3

Case 7 :

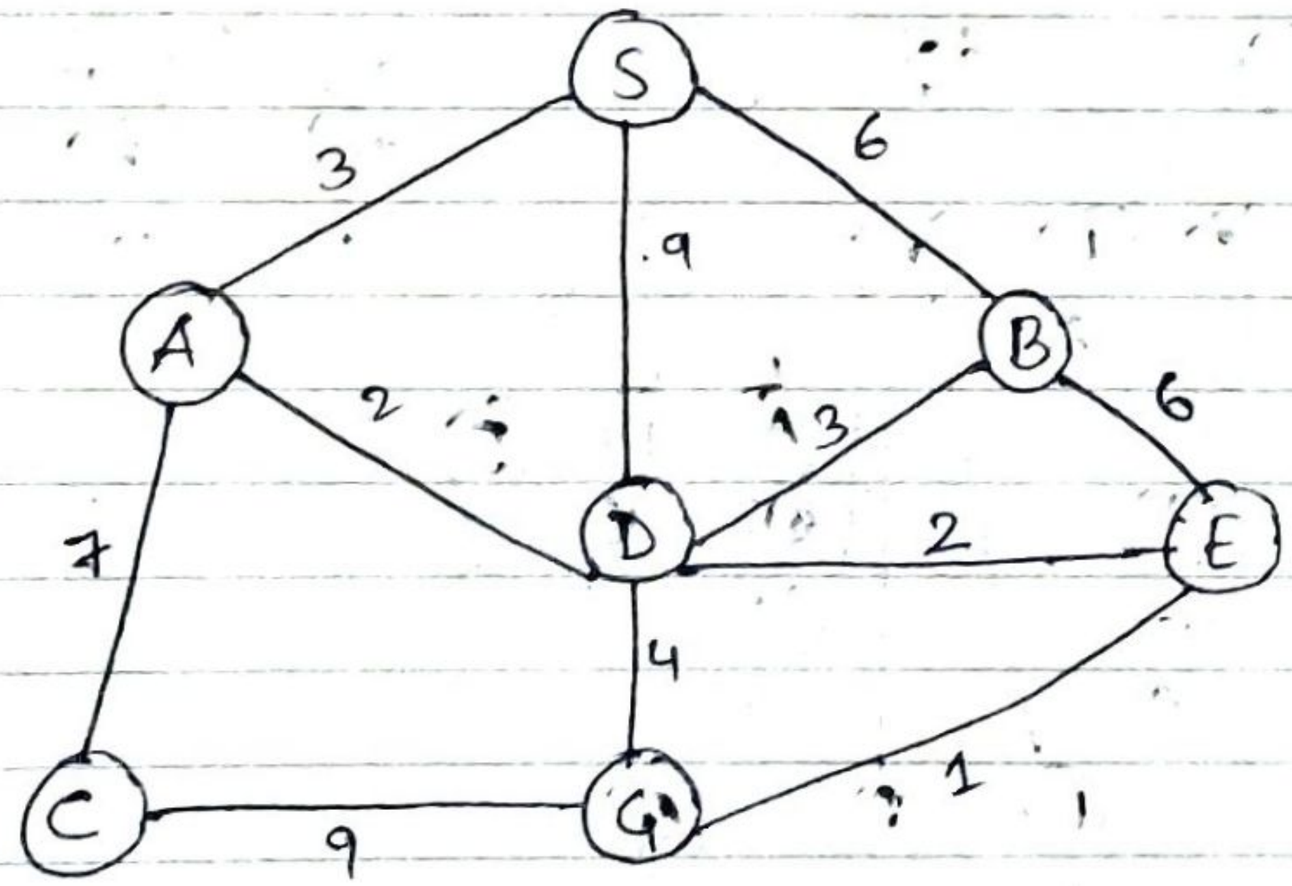


Score → 6 .

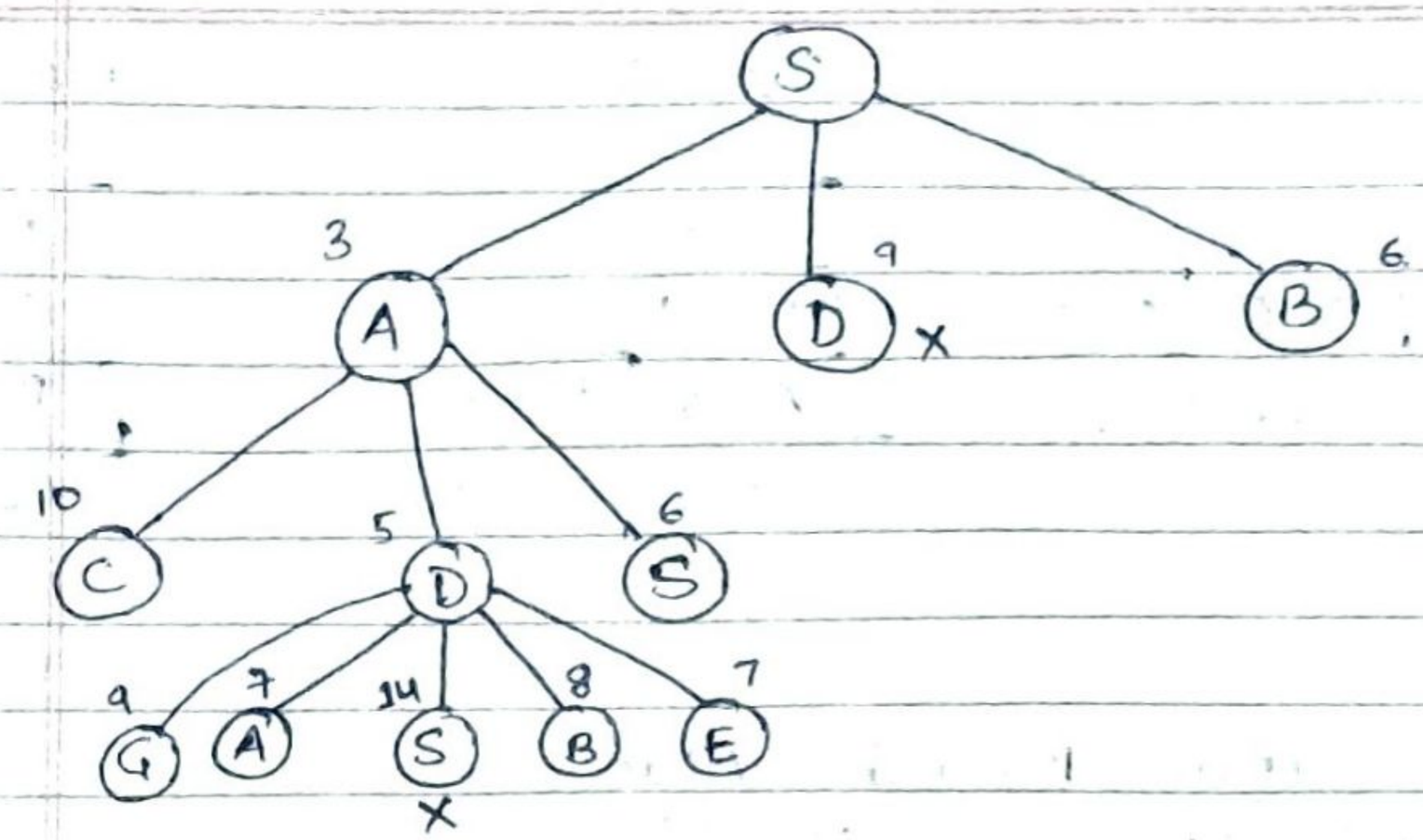
Branch and Bound Algo :

(1) It provides optimal solution
(2) Branching is the process of generating sub-problems.
(3) Bounding refers to be ignoring partial solution that cannot be better than the current best solution .
~~(4) Pruning~~

Pruning
It ~~eliminates~~ eliminates those parts of a search which does not contain better solution .

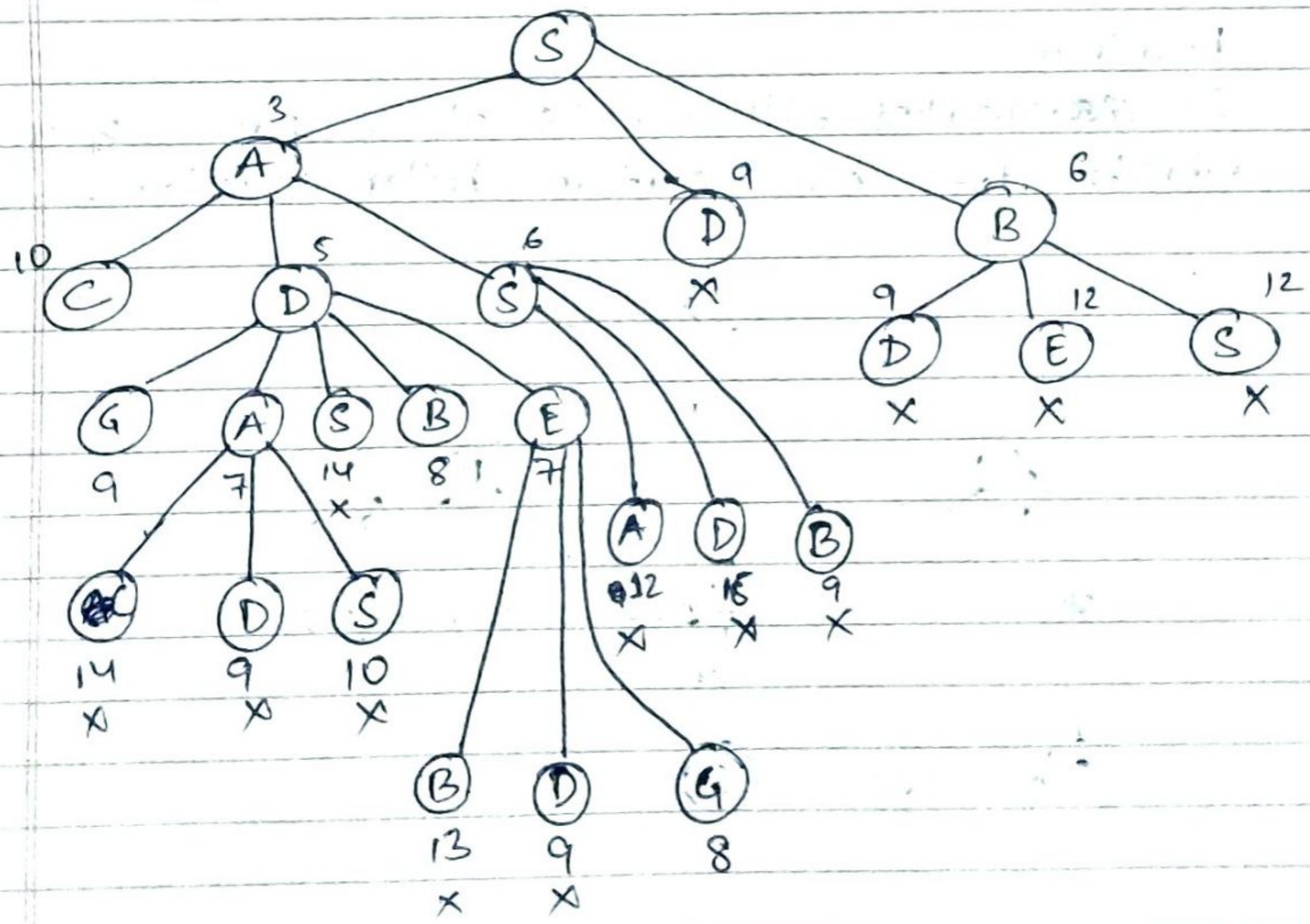


Scanned by TapScanner



we have achieved 'G' state and as of now the cost to reach G is 9.

Using Pruning we can discard 'D' and 'S' having cost 9 and 14 respectively.



S → A → D → E → G

Scanned by TapScanner

Constraint Satisfaction Problem

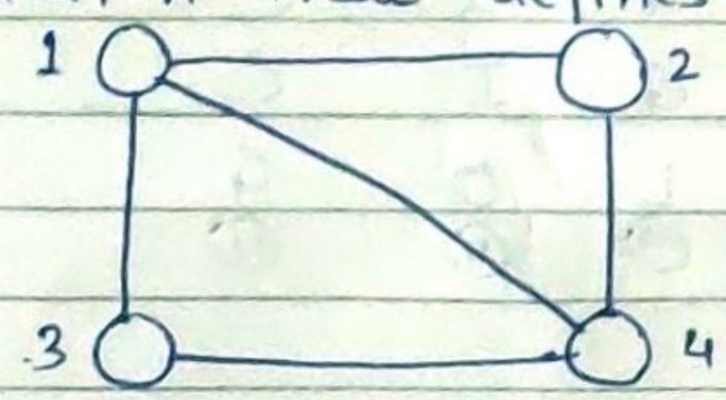
(1) It is used for map ~~colour~~ colouring or graph colouring.

(2) It contains 3 components.
V ; C.

(3) V is a set of variables $\{V_1, V_2 \ldots V_n\}$
D is a set of Domain $\{D_1, D_2, D_3 \ldots D_n\}$
which ~~contains~~ will be assigned one for each variable.

(4) C is a set of constraints that specify allowable combinations of values. C_i [scope, Rel]
where scope is a set of variables that participates in constraint.

(5) Rel is relation that defines the value that variable can take.



(6) It uses intelligent backtracking.

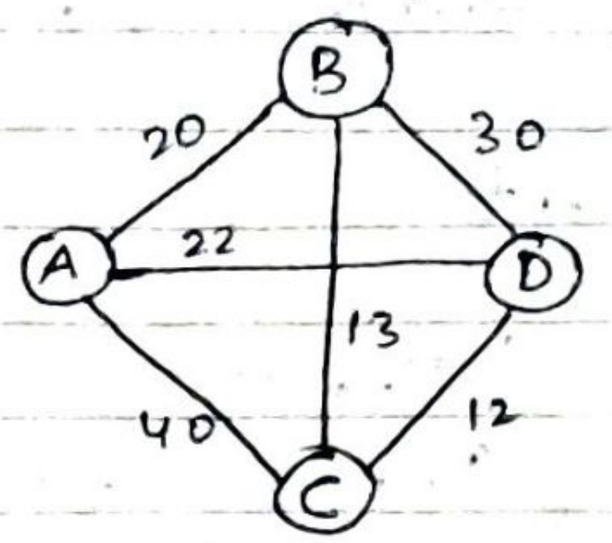
$V = \{1, 2, 3, 4\}$

$D = \{R, G, B\}$

$C = \{1 \neq 2, 1 \neq 3, 1 \neq 4, 2 \neq 4, 3 \neq 4\}$

		Intelligent Backtracking				
		1	2	3	4	
Initial Domain		RGB	RGB	RGB	RGB	
1	R	R	GB	GB	GB	
2	G	R	G	GB	B	
3	B	R	G	B	B	Empty.
4	G	R	G	G	B	

Scanned by TapScanner



Travelling Salesman Problem by appling State Space Search.

Crypt Arithmetic problem.

	L	O	G	I	C	~~O~~	O
carry ①	L	O	G	I	C	~~X~~	P
① P	R	O	L	O	G	~~2~~	C
	⑧	⓪	⑨	⓪	④	3	
						~~4~~	G
	L	E	T	S		~~5~~	I
	W	A	V	E		6	
L	A	T	E	R		7	
						~~8~~	R
						~~9~~	L

Scanned by TapScanner