

Chapter-1

what is AI ?

→ Artificial Intelligence is the simulation of human intelligence by machine like computer system

process

Learning Reasoning Selfconnection
(using the rules to reach goals)

four goals of AI

Thinking Humanly

→ cognitive
Science
approach

Thinking Rationally

→ Think well

Acting Humanly

→ Turing
test
approach

Acting Rationally

→ ACT well
→ heuristic

what is Heuristic? Method / Rule

→ Heuristic is a rule of thumb, strategy, trick, simplification on any kind of device which drastically limits search for solutions in large problem.

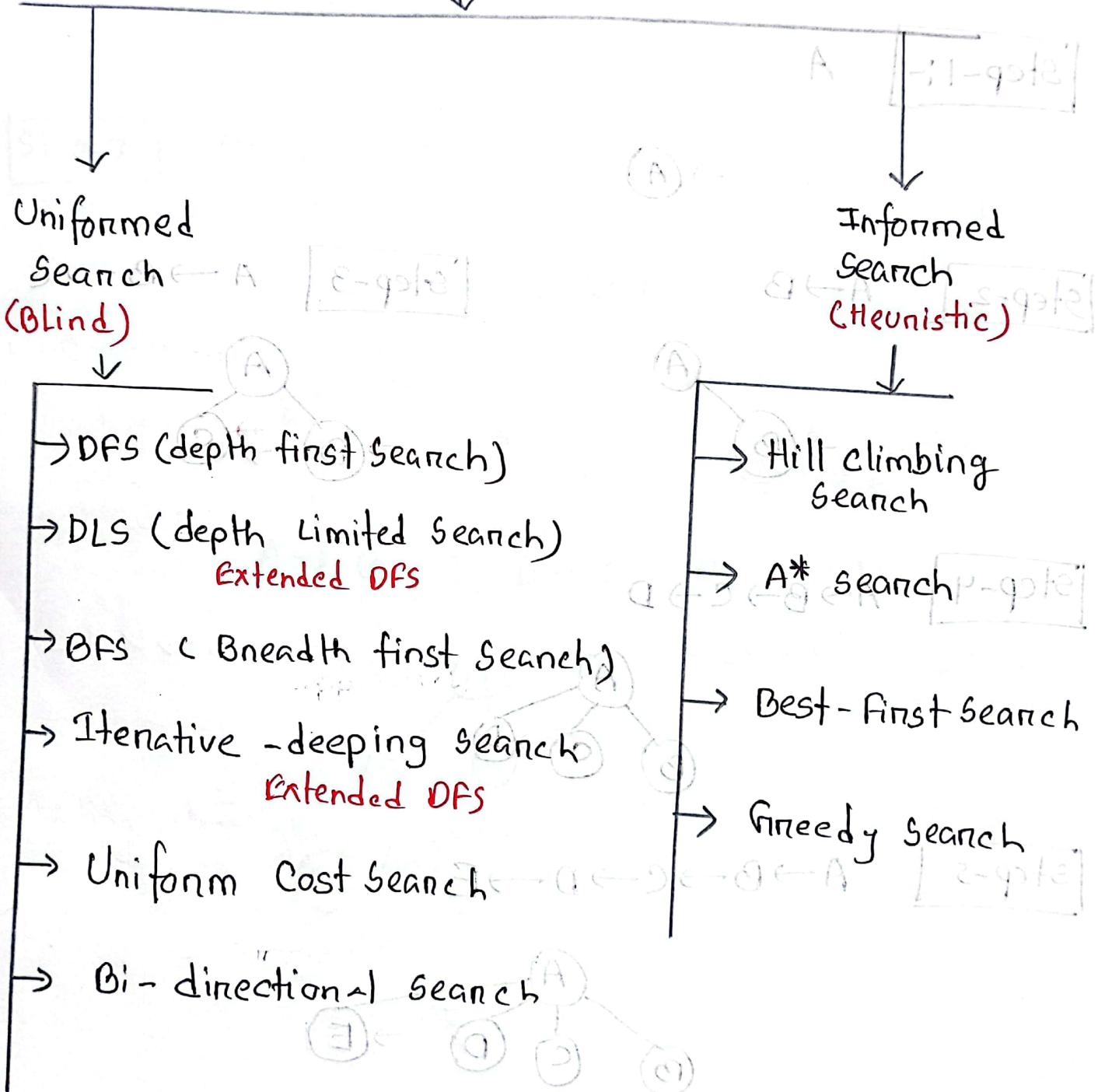
→ Heuristic method enable AI to produce approximate solutions rather than exact ones.

→ Instead of looking for perfect solution, heuristic method looks for quick solution that falls within acceptable range of accuracy.

chapter-3

27/01

⑥ Searching Techniques in AI - ~~Robotics~~



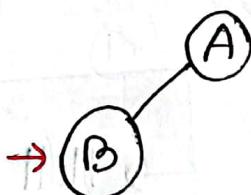
BFS

Start Node :- (A) Final state :- (O)

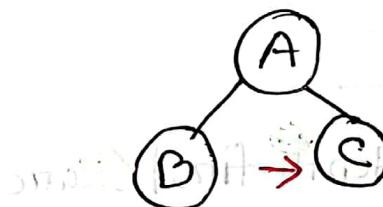
Step-1 :- A



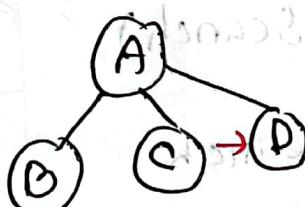
Step-2 $A \rightarrow B$



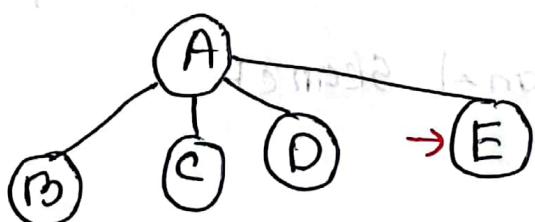
Step-3 $A \rightarrow B \rightarrow C$



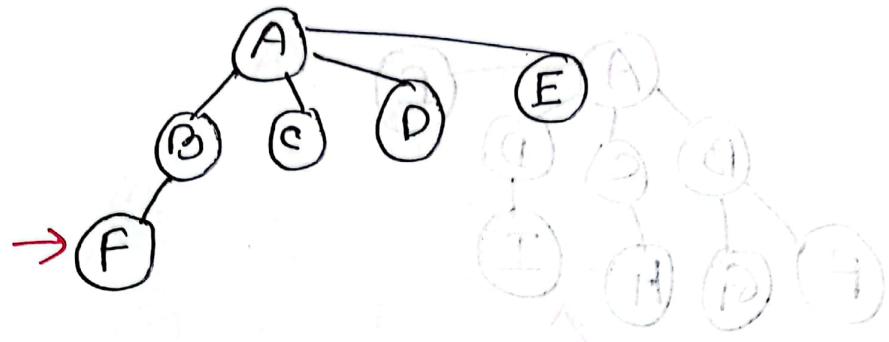
Step-4 $A \rightarrow B \rightarrow C \rightarrow D$



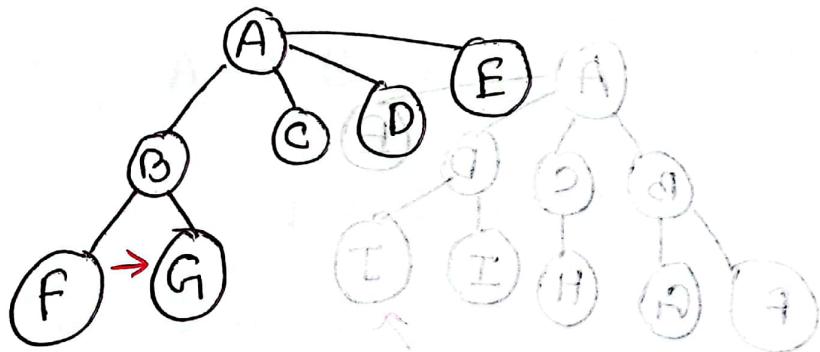
Step-5 $A \rightarrow B \rightarrow C \rightarrow D \rightarrow E$



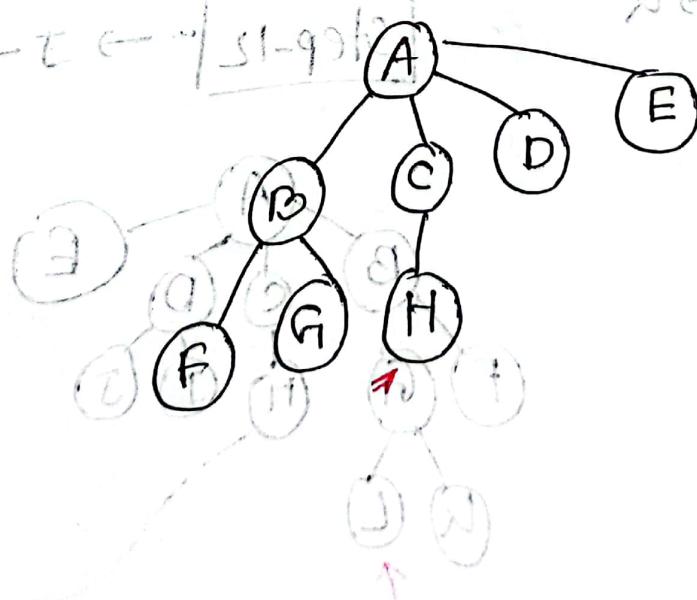
Step-6 $A \rightarrow B \rightarrow C \rightarrow D \rightarrow E \rightarrow F \rightarrow G \rightarrow H \rightarrow I \rightarrow J$ | 8-ply



Step-7 $A \rightarrow B \rightarrow C \rightarrow D \rightarrow E \rightarrow F \rightarrow G$ | 9-ply

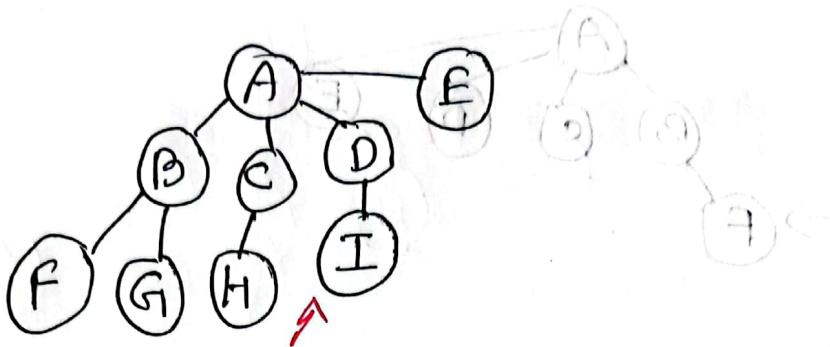


Step-8 $A \rightarrow B \rightarrow C \rightarrow D \rightarrow E \rightarrow F \rightarrow G \rightarrow H$ | 10-ply



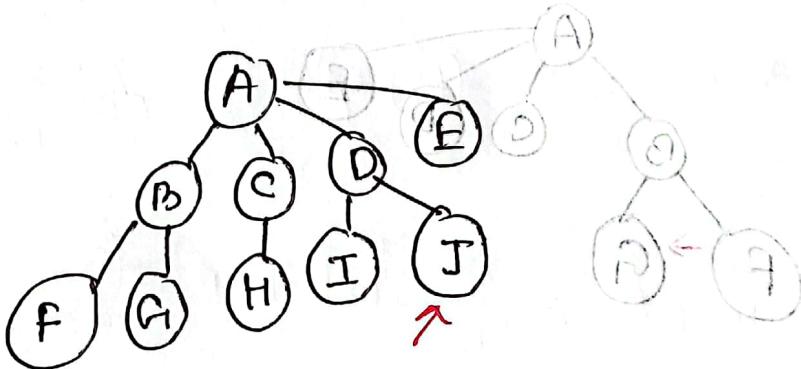
Step-9

$A \rightarrow B \rightarrow C \rightarrow D \rightarrow E \rightarrow F \rightarrow G \rightarrow H \rightarrow I$ [B-9P]



$A \leftarrow I \leftarrow B \leftarrow G \leftarrow H \leftarrow D \leftarrow E \leftarrow A$ [F-9P]

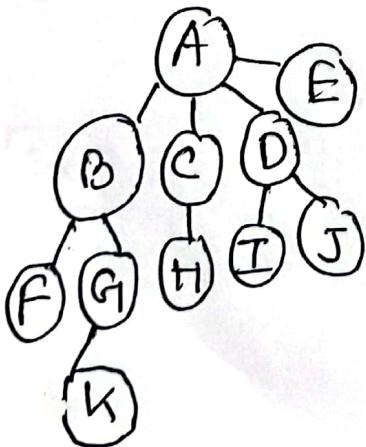
$\dots \rightarrow I \rightarrow J$



$H \leftarrow I \leftarrow F \leftarrow G \leftarrow B \leftarrow A$ [8-9P]

Step-11

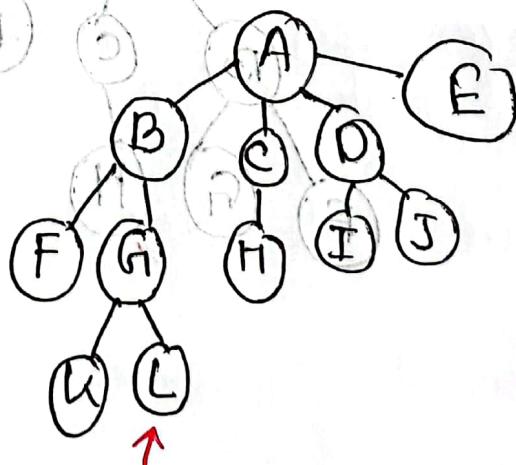
$\dots \rightarrow I \rightarrow J \rightarrow K$



1

Step-12

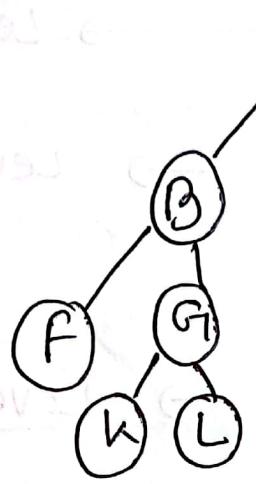
$\dots \rightarrow J \rightarrow K \rightarrow L$



1

Step-13

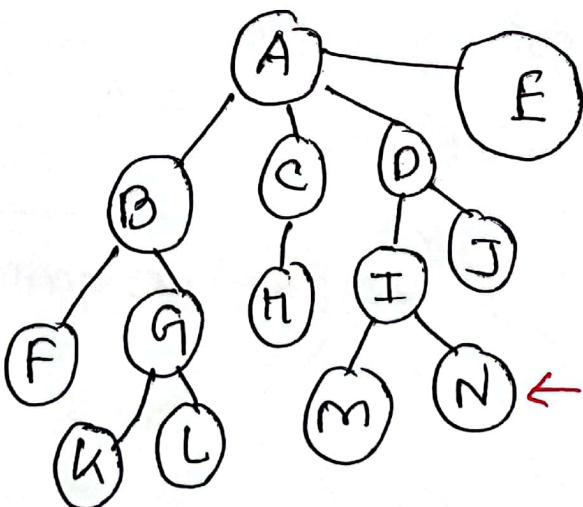
$A \rightarrow \dots \rightarrow J \rightarrow K \rightarrow M$



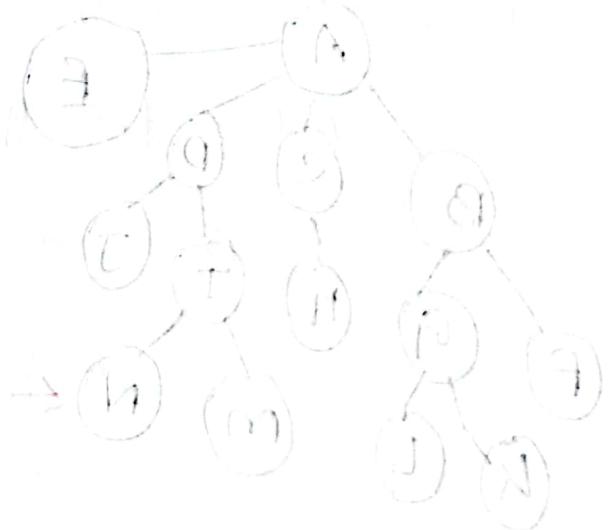
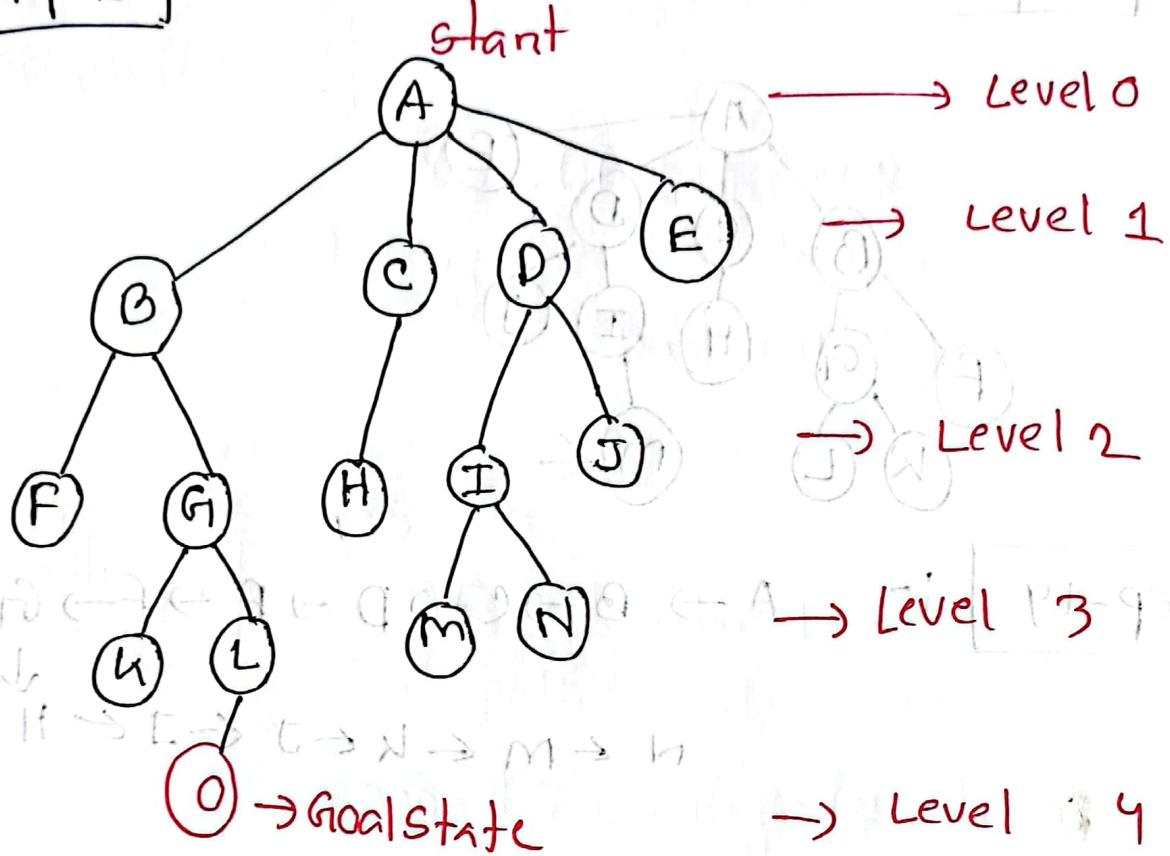
Step-14

$i) A \rightarrow B \rightarrow C \rightarrow D \rightarrow E \rightarrow F \rightarrow G$

$N \leftarrow M \leftarrow K \leftarrow J \leftarrow I \leftarrow H$



Step-15



BFS

dis

Advantage :-

(store each level)

→ requires lots of memory

Space problem

→ needs lots of time

if goal Node is far from root node

(d init. $\approx d$)

Time complexity

$O(b^{d+1})$

ni ei feb → depth of maximum branching factor in a tree

branch out → depth of shallowest

($d = \text{depth}$ of solution)

Space complexity

$O(b^d)$

keeps every node in memory

298

(Chalkboard notes)

Completeness?

smriti Yes, BFS is complete
mont not ei it's open loop
it always reaches to goal if
open loop
(b is finite)

Optimality?

yes

BFS is optimal

if path cost is in
minimum
non-decreasing function

of depth of the node

Ex:- (Step cost = 1)

(b_d)₀

Optimal

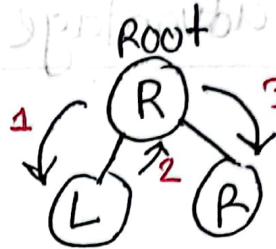
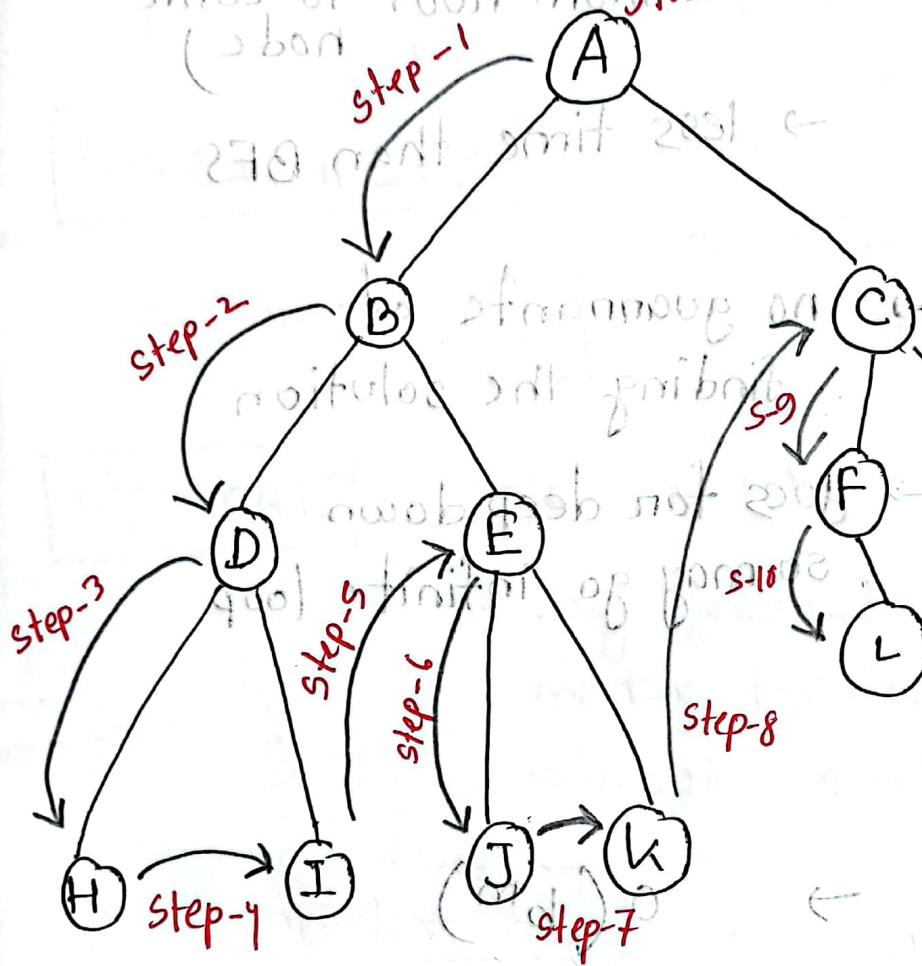
in short moves

program

Depth first search

DFS

state space graph
flag no solution to
mod of tree from start node
(abn)



Optimal node

→ Goal state

Best goal state

Path found: A → B → D → H → I → E

depth: 5

state space: 6 - M

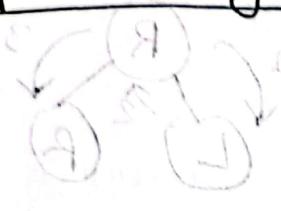
nodes

visited: 6 - M

memory

available: 6 - M

advantage :-



→ uses less memory

Only store a stack
of nodes on path

(shortest
from root to curr
node)

→ less time than BFS

disadvantage

→ no guarantee of
finding the solution

→ goes for deep down
so may go infinite loop

Time complexity

→ $O(b^m)$

$b \rightarrow$ maximum branching
factor

$m \rightarrow$ maximum depth
of node

$m > b \rightarrow$ worst condition

space complexity

$O(bm)$

Linear space

completeness?

NO

fails infinite depth
space

Optimal?

NO

it may generate a large
number of steps / high cost
to reach a goal node

list

repetition

DLS

Depth Limited Search

limit = 2

L-0

Step-1

Root

S

L-1

depth limit reached

A

B

L-2

S-2

C

D

I

J

S-3

S-4

S-5

S-6

E

F

G

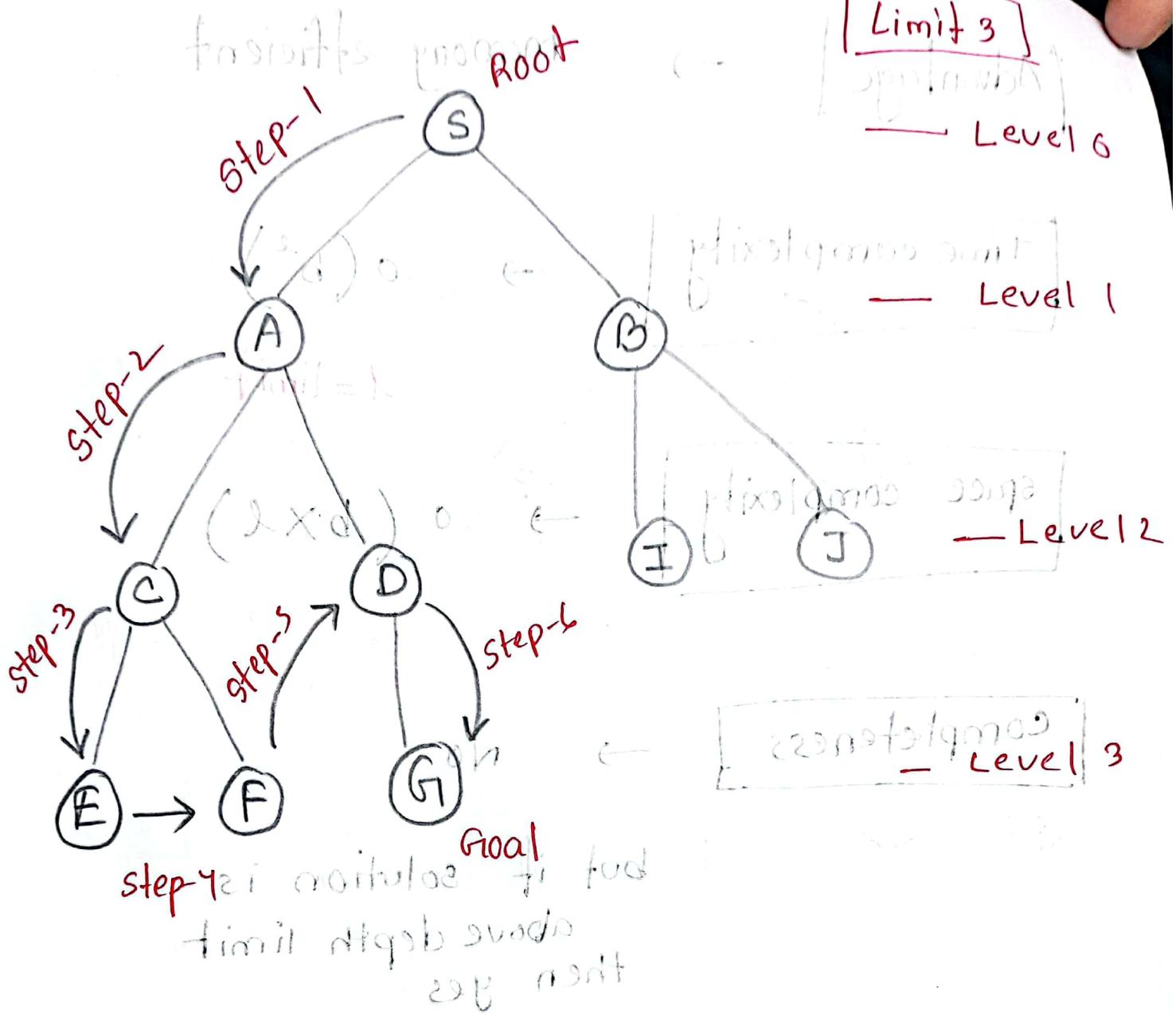
Goal

as limit
is 2

S → A → C → D → B → I → J → X

fail

no solution



$S \rightarrow A \rightarrow C \rightarrow E \rightarrow F \rightarrow D \rightarrow G$

$b < 2$ finds

SUCCESS
 Reached
 goal

Advantage

→ memory efficient

Time complexity

→ $O(b^l)$

$l = \text{limit}$

Space complexity

→ $O(b \times l)$

Completeness

→ No

but if solution is found
above depth limit
then yes

Optimal

220000
branch & bound
1308

even if $l > d$

~~IDS~~

Iterative Deepening Search

Step-1

Level 0

DLS bound with = 0

(A)

(B)

(C)

— Level 0

(A)

(B)

(C)

(D), (E)

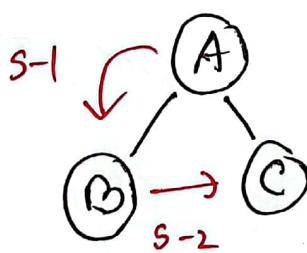
Iteration :- A → B → C → D → E

∴ Goal not found

∴ increase Level = 0 + 1

Step-2

Level 1 ; DLS bound = 1



— Level 0

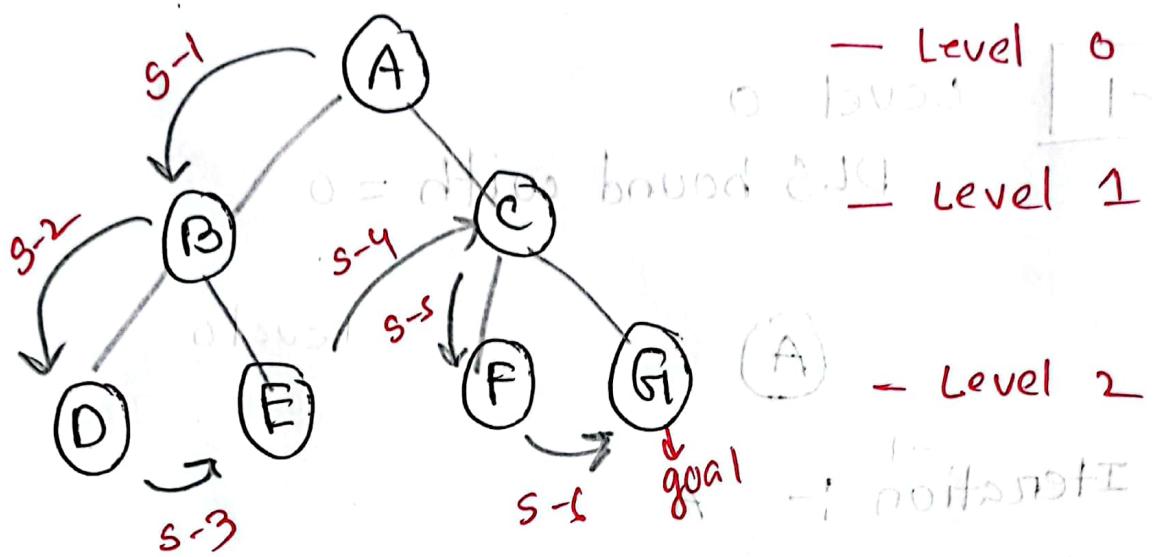
— Level 1

Iteration :- A → B → C

∴ Goal not found increasing level

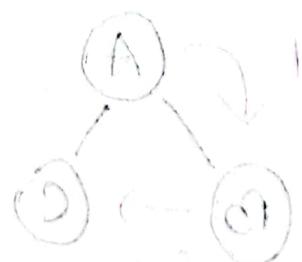
Step-2

Level : 2 DLS bound : 2



Iteration :- $A \rightarrow B \rightarrow D \rightarrow E \rightarrow C \rightarrow F \rightarrow H$
-3

Goal node found in 3rd iteration



Advantage

- BFS (fast branch)
- DFS (memory efficiency)

Time complexity

- $O(b^d)$

Space complexity

- $O(bd)$

Completeness?

- Yes

complete when
b is finite

Optimal ?

- Yes

if step cost = 1

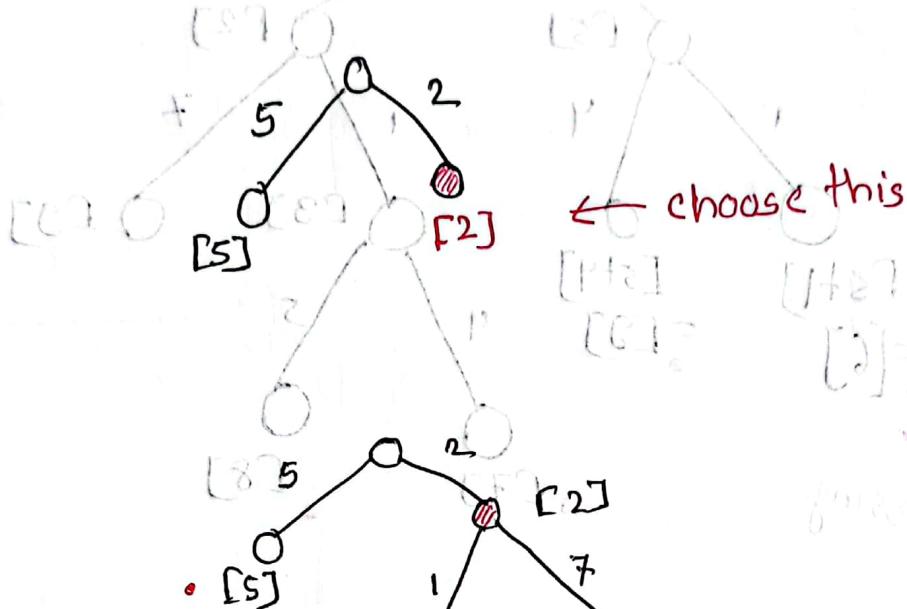
or

increasing function
of depth

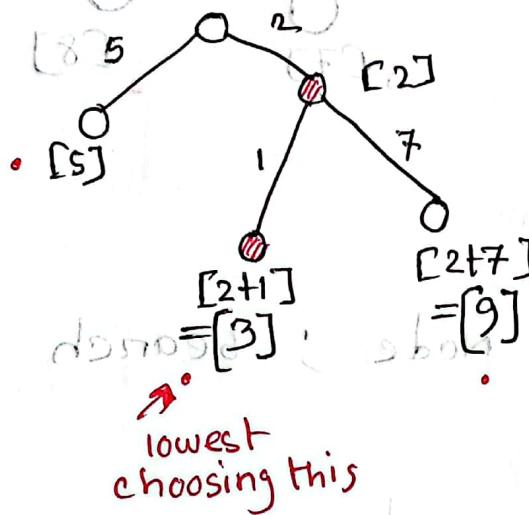
Uniform Cost

Expand node with smallest cost $g(n)$

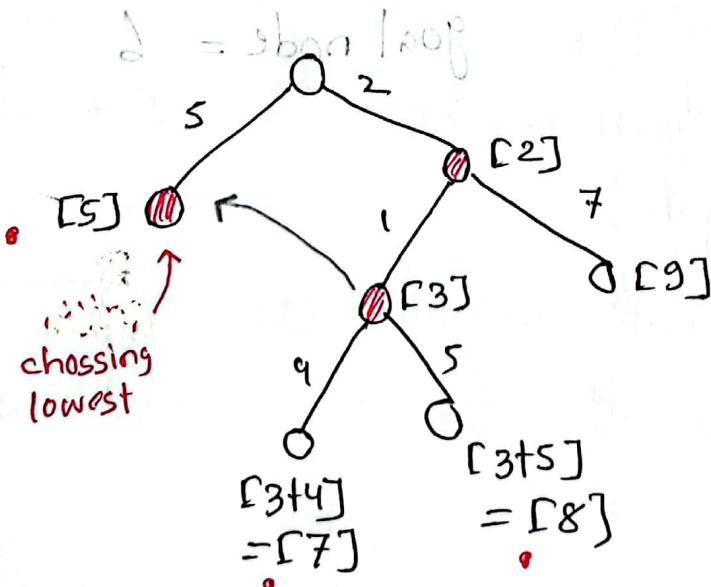
Step-1



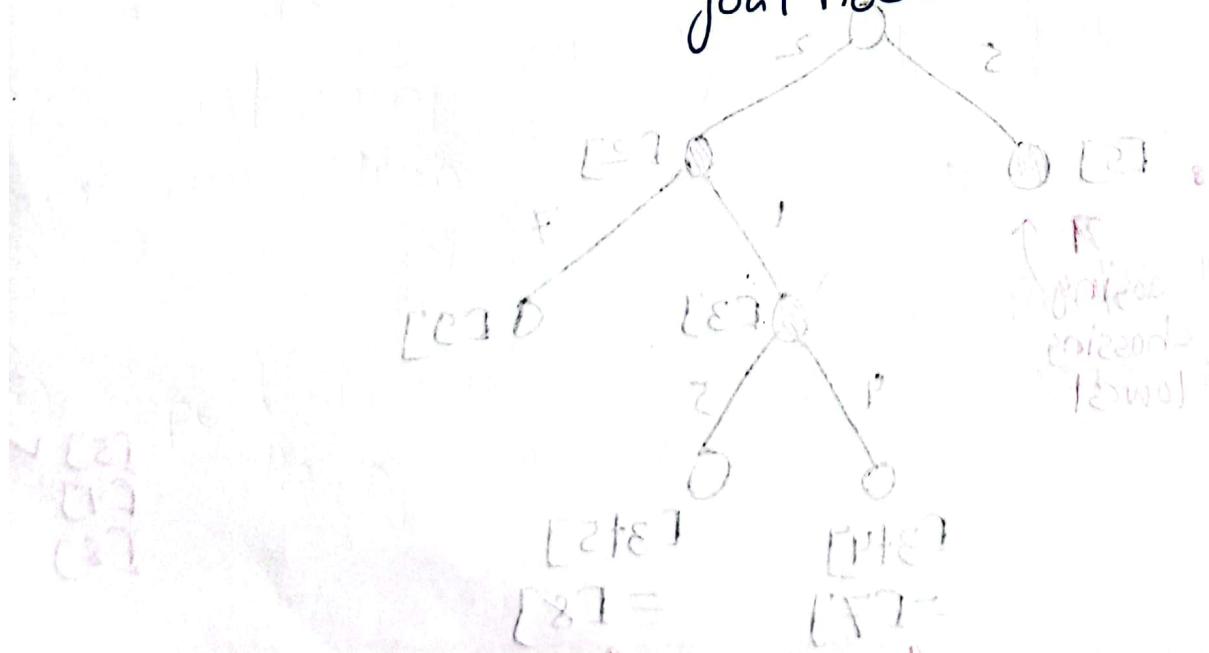
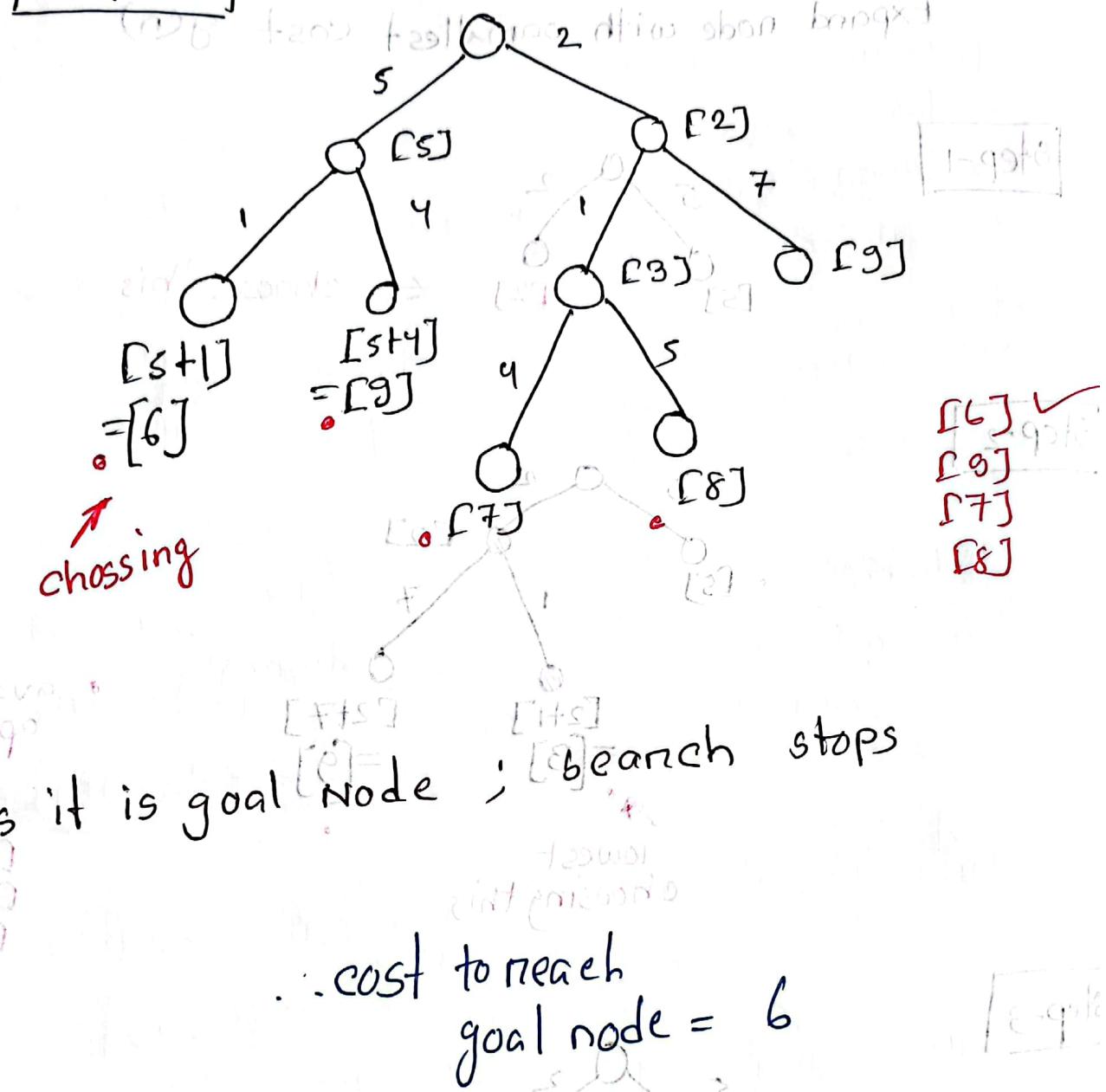
Step-2



Step-3



Step-4



Disadvantage

- doesn't care about number of steps only concerned about path cost

may stuck infinity loop
and back again

non-monotonic function

Time complexity

$$O(b^{C^*/\epsilon})$$

b = branching factor

C^* = cost of optimal solution

ϵ = each step to get closer to goal node

$$O(b^{C^*/\epsilon})$$

Space complexity

too much

Completeness?

Yes

if step cost $\geq \epsilon$

otherwise stuck in loop

optimality

Yes

for any step cost

Uninformed Search

- ① is a searching technique that has no additional information about the distant from the current state to the goal

Informed Search

- ① is a searching technique that has additional information about the distant from the current state to goal.

② No use of knowledge

② Use knowledge to find steps to the solution

③ medium efficiency

③ Highly efficient

④ cost comparatively High

④ Cost Low

⑤ Speed slower than informed

⑤ Find solution more quickly

Informed

exploit information about the problem domain

Uninformed

exploit no information about the problem domain

⑥ Example Algorithm

Best first search

FIFO search but no information about the problem domain

A* search

state loop of a search move to

④

Upcoming best

known as
Heuristic Search

best action from set

known as Blind Search

farm

⑤

less length

while

to (implementation)

⑥ Best for more length

use

heuristic

/ domain

Don't know ledge

to choose

the best

move

state from set

choose from set

best move to

state from set

(n) d = (n) +

too set for no information set = (n) d
loop of a move to set

what is heuristic function?

Heuristic function is a function, $f(n)$, that gives an estimation on the cost of from node n to goal state - so the node with least f cost among all possible choices can be selected for expansion first.

Approaches of defining f

① f measure the value (goodness) of current state

② f measure the estimated cost of getting to the goal from the current state

$$f(n) = h(n)$$

$h(n)$ = is the estimation of the cost to get from n to goal

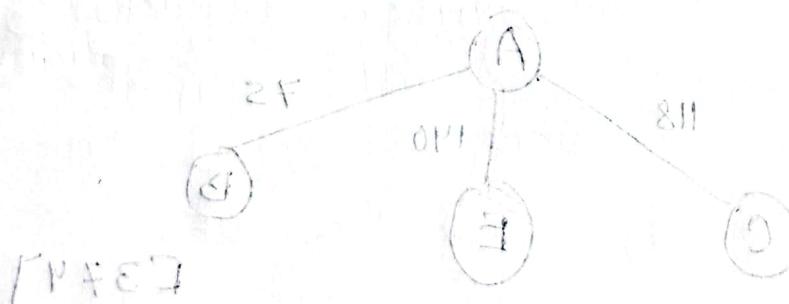
(3) f measures the estimated cost of getting to goal state from current state and the cost of existing path to it.

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

$g(n)$ = the cost to get n

from initial state

$h(n)$ = an estimate of cost to get from n to goal



$$0H = (n)$$

[PAST]

Greedy Best First Search

→ follow the lowest $h(n)$ → don't care about cost
 $f(n) = h(n)$

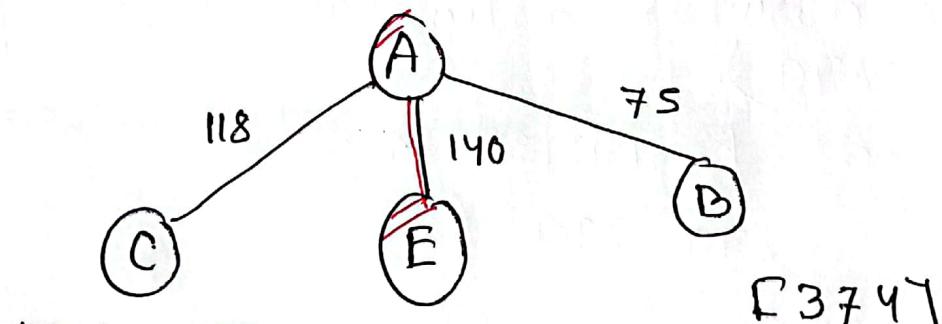
Step-1

A start

$$h(n) = 366$$

[366]

Step-2



$$h(n) = 329$$

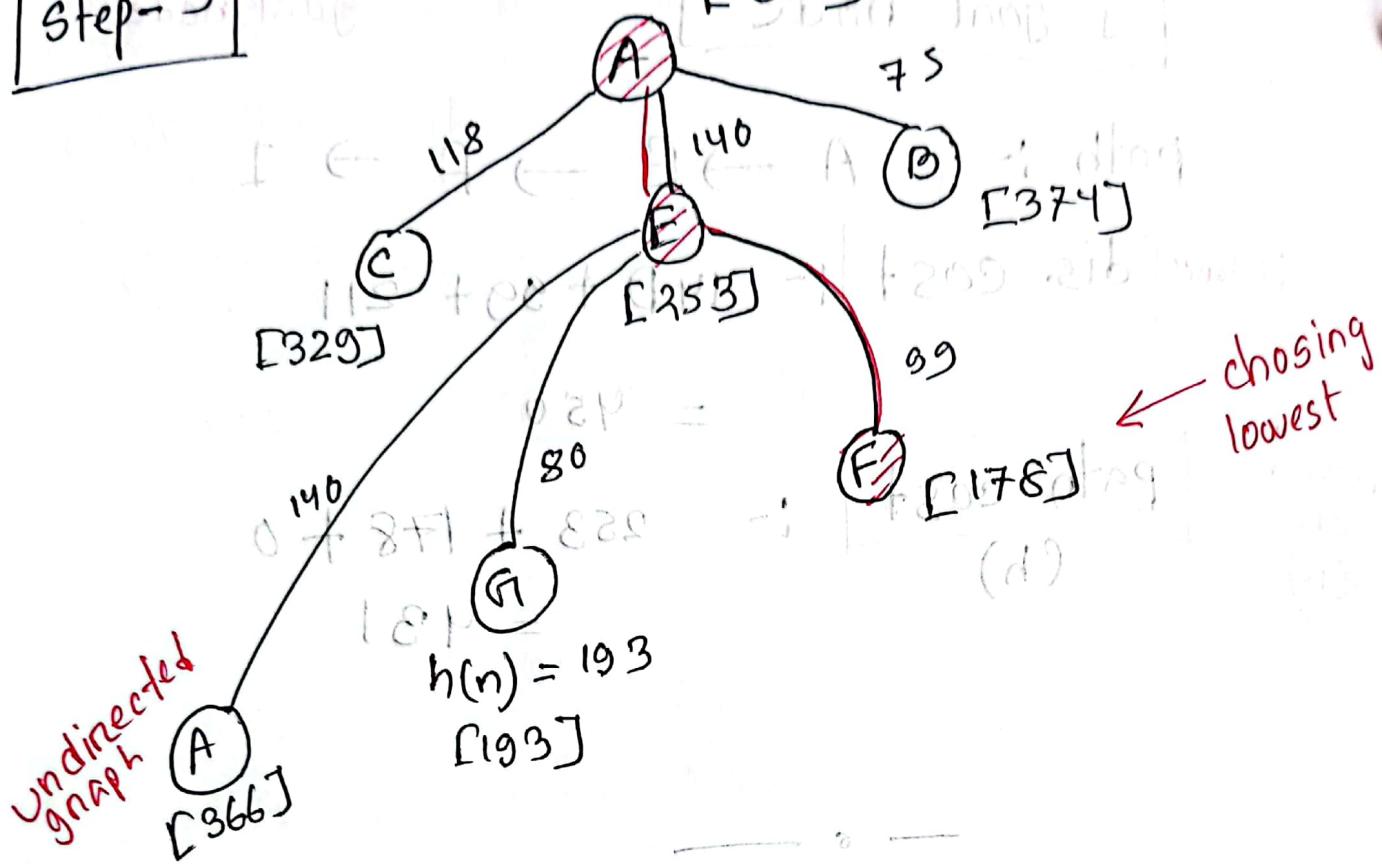
[329]

[253]

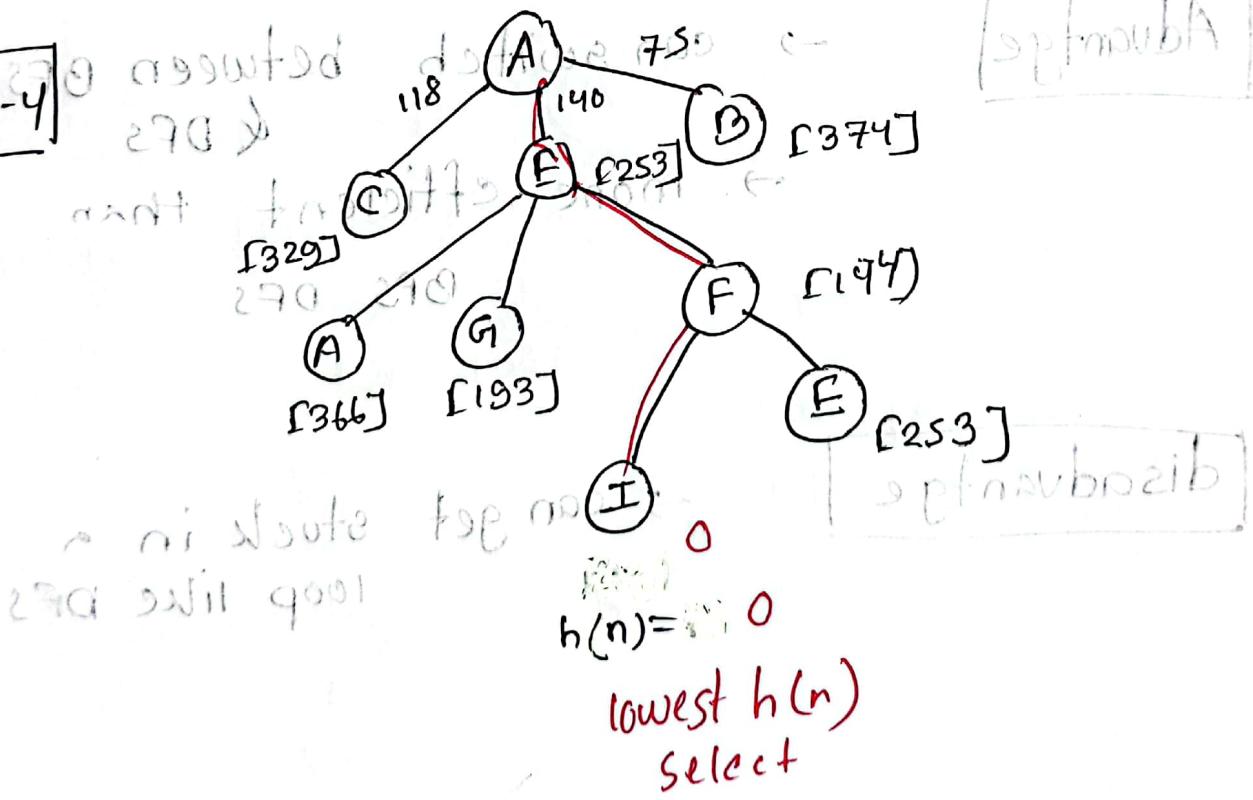
choosing

cause lowest $h(n)$

Step-3



Step-4



I goal node

$h(n) = 0$ means goal node reached

path :- A → E → F → I

distance / dis. cost :- 84 + 99 + 211

based
on
 $h(n)$

path cost (h) :- 253 + 178 + 0

$$= 431$$

801 = (0) d
[seen]

Advantage

→ can switch between BFS & DFS

→ more efficient than

disadvantage

→ can get stuck in a loop like DFS

Time complexity

$$O(b^m)$$

b = branching factor

m = maximum path length

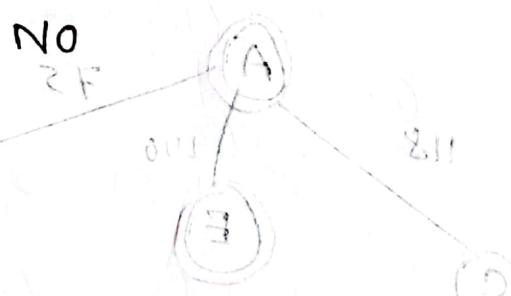
Space complexity

$$O(b^m)$$

Completeness

No (even if state space finite)
incomplete cause don't
check for repeated states

Optimal



is opt

$$P(F+E+F) = (n)^4$$

$$[exp] =$$

$$2 \times 2 + 0 \times 1 = (n)^4$$

$$[Exp] =$$

$$2 \times 8 + 8 \times 1 = (n)^4$$

$$[Exp] =$$

Result
(n)

Combination of
Uniform cost
and greedy

A* Search

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

exact
cost to
reach n node

estimation
to reach
goal
node

(Initial state A moves) ON

Goal state

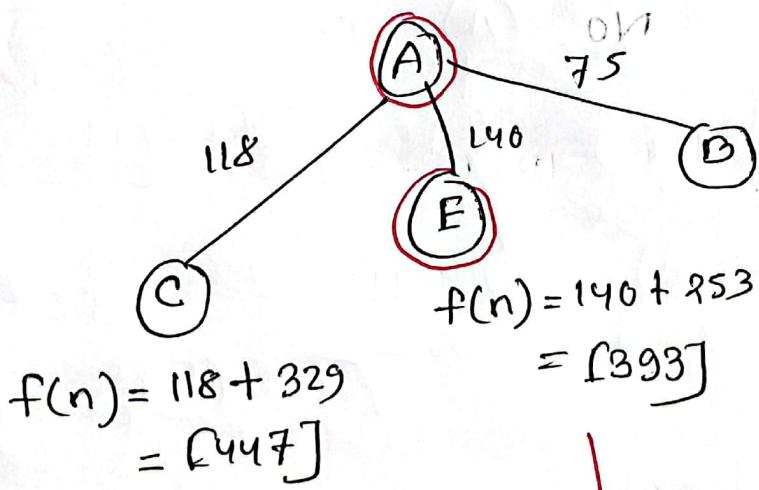
Step-1

(A)

start

$$f(n) = 0 + 366 = [366]$$

Step-2

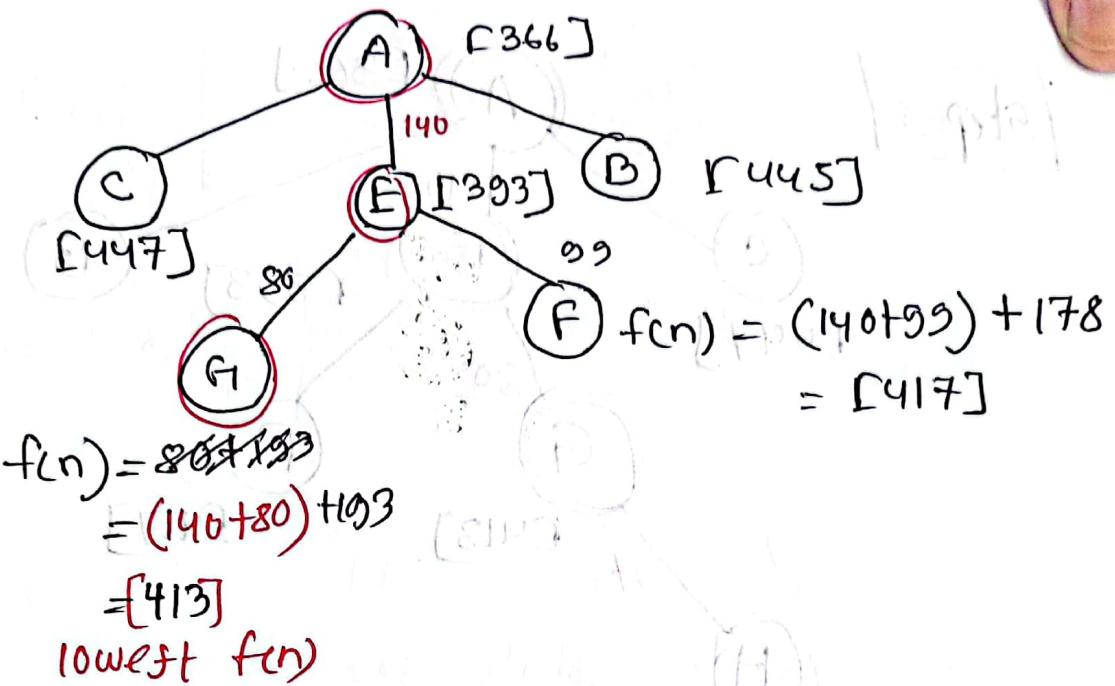


Initial

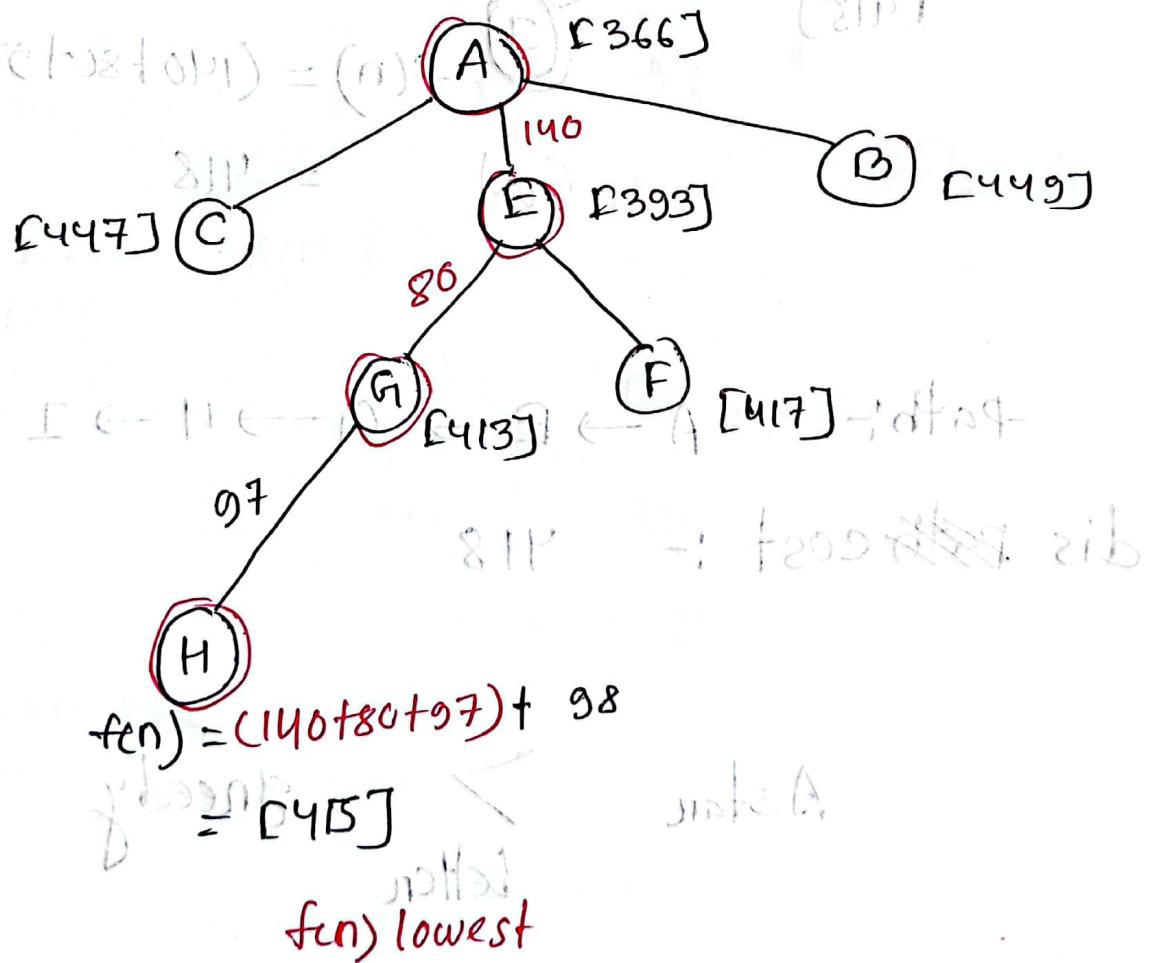
$$\begin{aligned} f(n) &= 175 + 378 \\ &= [545] \end{aligned}$$

lowest
 $f(n)$

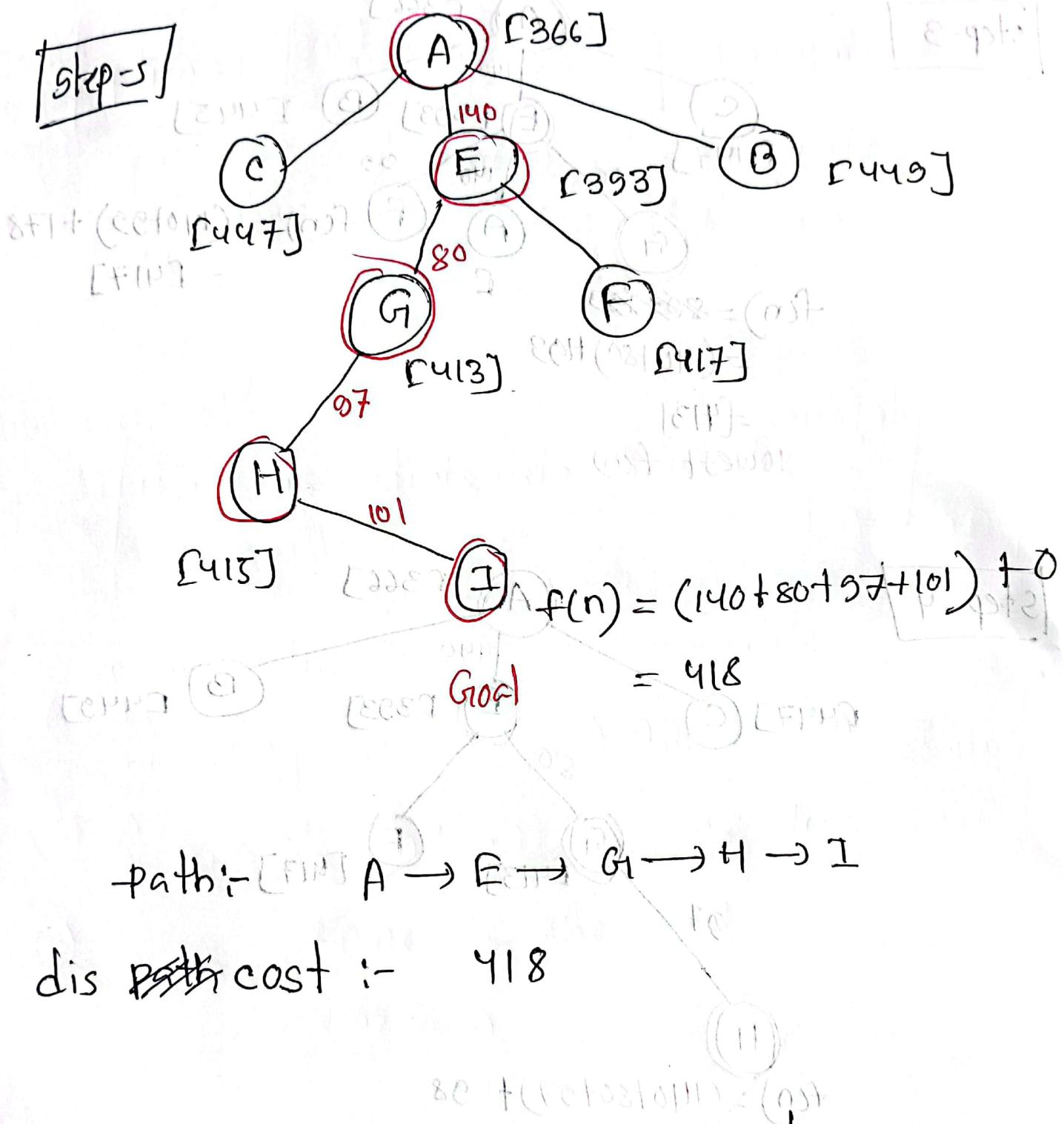
Step-3



Step-4



Step-5



Astar > greedy
better

advantage

- best of all
- can also solve complex problems

disadvantage

Highen memory requirement

- not always produce shortest path

Time complexity

$$O(b^d)$$

depends on $b^{(n)}$

Space complexity

$$O(b^d)$$

completeness

Yes if

($b = \text{finite}$)

(cost is fixed)

optimal

We have to find -

value of node -

Yes

and optimum

if $h(n)$ is admissible

(known as admissible heuristic)

From this

is admissible

subgoal priority function

After heuristics

what is admissible heuristic ?

a heuristic $h(n)$ is admissible

if for

every node n , $h(n) \leq h^*(n)$

where $h^*(n)$ is the true cost

to reach the goal state

from n

if g

(start = g)

(exit = g)

What are the concepts in a search?

① Initial state

② Action / Transition Model

③ State space graph

④ Goal test

⑤ Solution / optimal solution

State space search

Def-1

A state space search is a method used in AI to find solution to a problem by searching through the set of possible states.

Def-2 Formal def

State space is represented by 4 types

[N, A, S, GD]

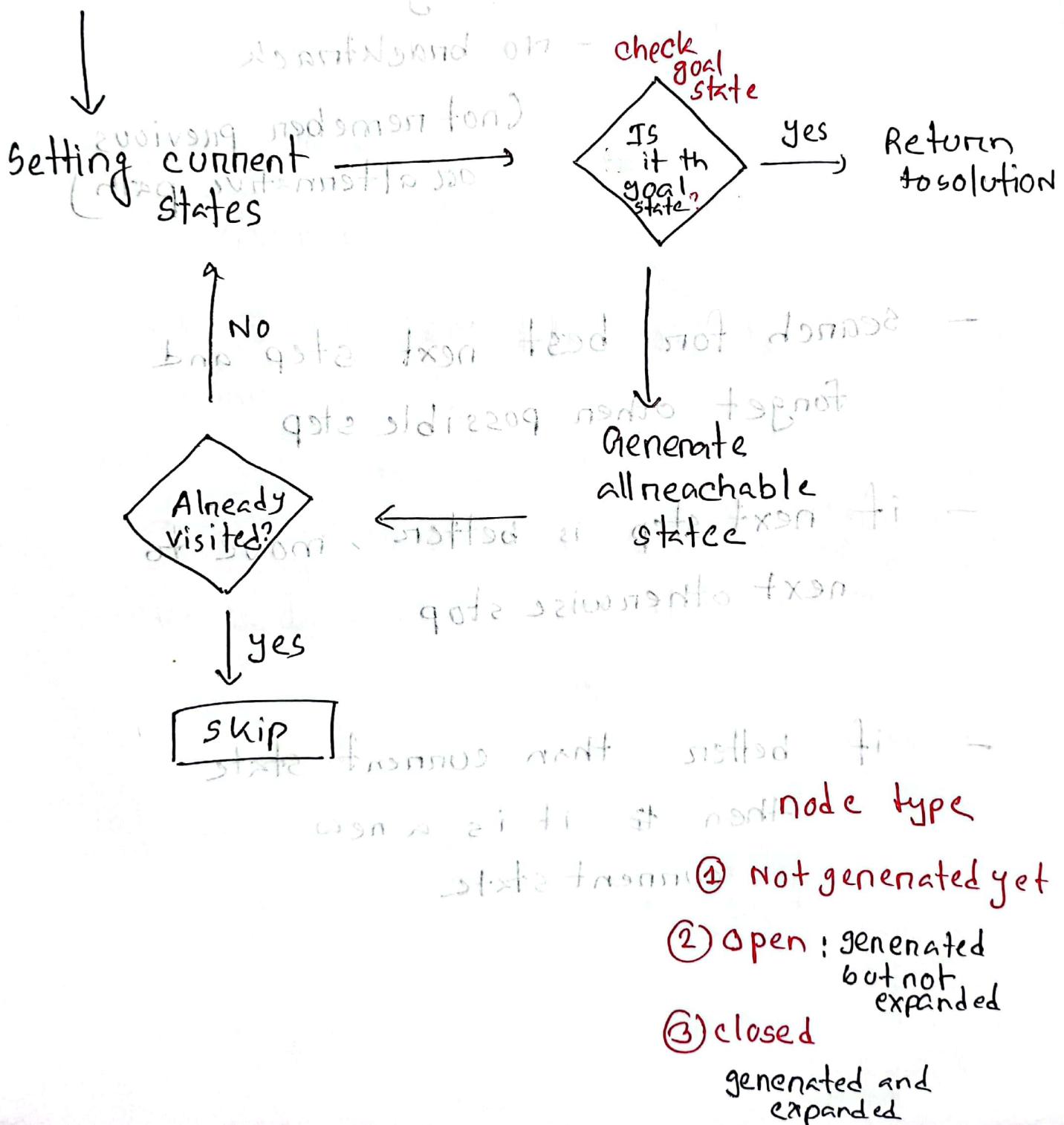
- ① N → set of nodes or states of graph
- ② A → set of arch or links between nodes
an instance of an action
- ③ S → ^{subset} contain start state of a problem
- ④ GD → ^{subset} containing goal state of problem

Algorithm

Ex :- A*

BFS, DFS
Hill climbing

Initialization



HILL climbing

- Local Search Algo

- Greedy

- No backtracking

(not remember previous or alternative path)

- search for best next step and forget other possible step

- if next step is better, move to next otherwise stop

- if better than current state

then it is a new

current state

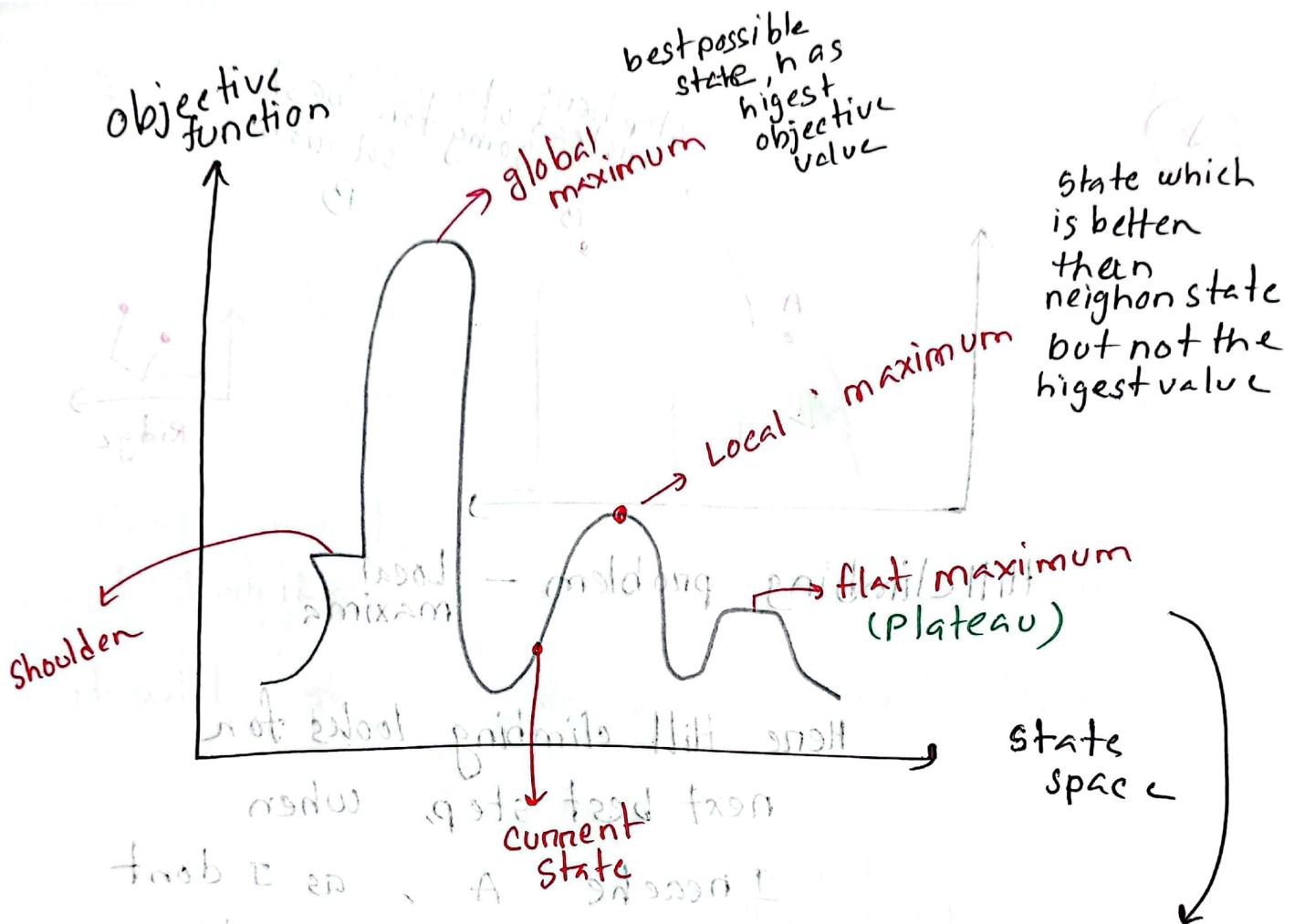
b321978 : 0996 (5)

for loop

b321978

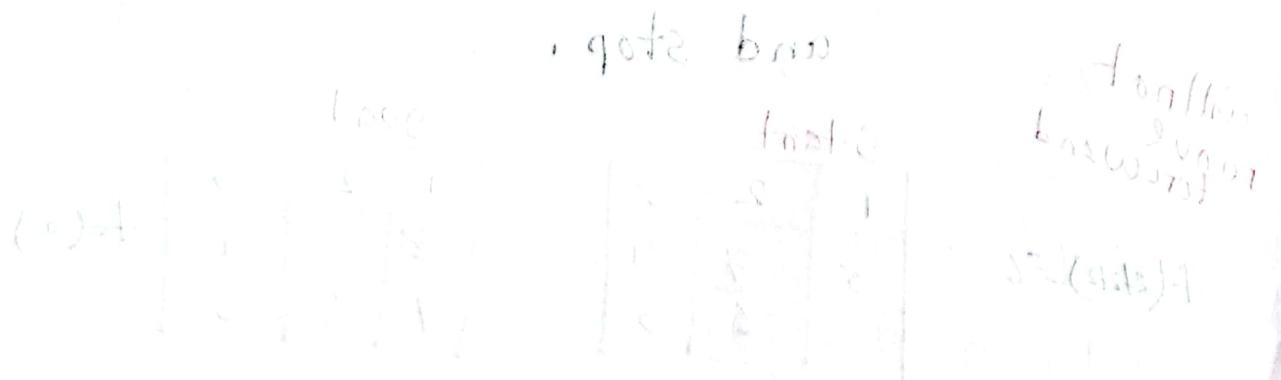
b32010 (5)

b321978
is brang



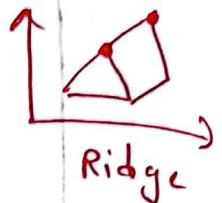
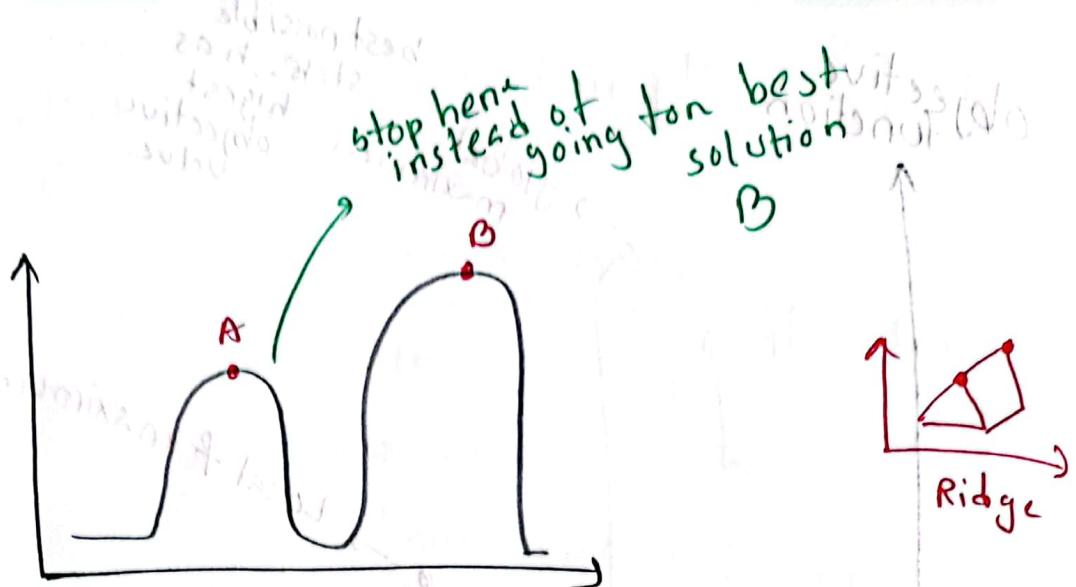
radius of state feed from
task c en. A straight line
to a shoulder sinks and
positive feed in A straight line

all neighbour
states has
same value



(+) + 0.5
task
task C

①
initial state
method A
state comparison
addition and subtraction
but very rapid



Hill climbing problem - Local maxima

Hence Hill climbing looks for next best step, when I reach A, as I don't have other knowledge, it will think A is best solution and stop.

will not move forward

$$f(2+2+2) = 6$$

Start		
1	2	5
8	7	4
6	3	

goal		
1	2	3
8		4
7	6	5

$$f = 0$$

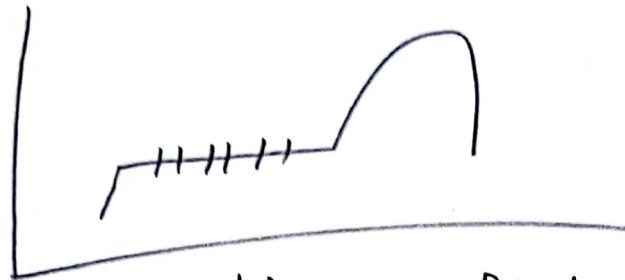
$$\begin{aligned} f &= (1+2+2+2) \\ &= 7 \end{aligned}$$

1	2	5
7	4	
8	6	3

1	3	5
8	7	4
6		3

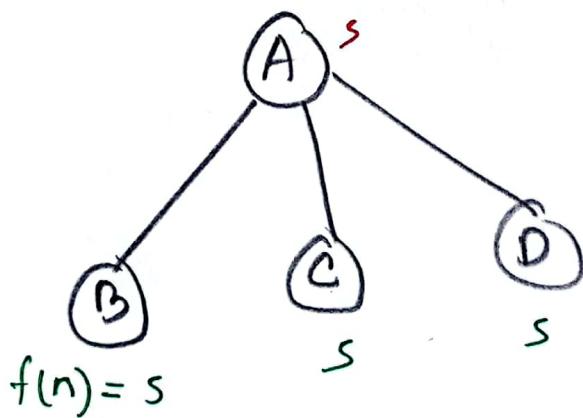
~~$f = -7$~~

(2)



Hill climbing :- flat maximum / plateau problem

∴ if all neighbouring states has same value
it will not move for next step and stops.



Algorithm stop
to go next
step

cause all has
same value

if only search
for next best option

- # starts from level 0. Search for goal node through iteration.
- # if not found increase level by 1.
now search for goal node
- # if not found increase level by 1
and search through iteration until you
found the goal node