Expt. No. • Chem

Date. / /
Page No.

AI

1. Approach to AI (Heuristic Search, Game Playing)
   [A*, AO*, Best First] [MinMax, α-β]
   Constraint Satisfaction, DFS, BFS, Hill Climbing
2. Knowledge Representation (Approaches, Predicate Logic, Reasoning)
   Forward Backward

AI

Giving power to machines, like human so that they can also think and can take decision itself like human after learning

Applications of AI

Google Assistant, Alexa, Tesla Smart Cars, Recommendation in Netflix

History

Evolution of AI - during world war 2 to break the German communication a computer is made in which Alen Turing has great role. ~~Alen Turing~~ Evolution of AI comes in the picture when a newspaper is published in which the headline is "Can Machine Thinks" that was written by Alen Turing.

Teacher's Signature :

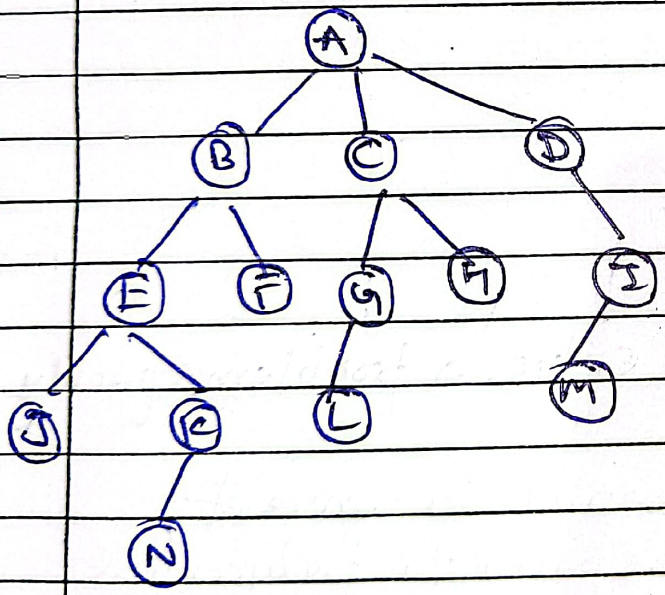
	Uninformed Searching	Informed Searching
①	Search without Information	Search with information (TSP)
②	No knowledge (Brootforce)	Use knowledge to find steps for solution
③	Time Consuming	Quick Solution
④	Optimal	Not always optimal
⑤	More Complexity (Time, Space)	Less Complexity (Time, Spac
⑥	DFS, BFS etc	A*, Heuristic DFS, Best First Search

Heuristic Function $h(n)$

Expt. No. Date. / /
Page No.

Breadth First Search (BFS)

- Uninformed Search Technique (Blind Tech)
  Brute force
- FIFO (Queue) Level Order Traversal
- Shallowest Node
- Complete (It will always give answer)



A | B | C | D | E | F | G | H | I | J | K | L | — —

- Optimal
- Time Compl (O(V+E))
  In AI $O(b^d)$
  b → branch factor (no of children)

Teacher's Signature :

Depth First Search (DFS)

1. Uninformed Search Tech
2. Stack (LIFO)
3. Deepest Node
4. Incomplete
5. Non Optimal
6. Time Complexity
   $O(b^d)$

To guess
↑
Heuristic in AI

It is technique designed to solve a problem quickly

Heuristic is used when we want to convert non polynomial (time & state space) to polynomial (time & state space)

It reduces time to solve a problem.

It cannot give optimal solution always.

In informed searching heuristic is used as Guider.

Expt. No. _____ Date. / /
Page No. _____

Best First Search Algo

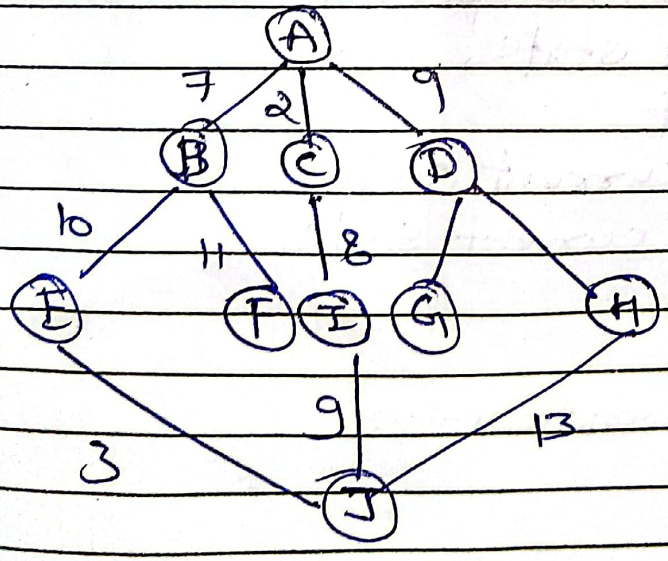
- It gives always good soln but does not gurantee optimal soln.
- Informed Searching
- Heuristic Technique
- Greedy Method (Lesser heuristic val)

Algorithm:

→ Let OPEN be a priority queue containing initial state

→ Loop
  If OPEN is empty return failure
  NODE ← Remove-First (OPEN)
    If Node is a Goal
      then return the path from initial to Node
    else generate all the successors of Node
      put the newly generated Node into OPEN
      according to their heuristic values

→ END



Heuristic value
A → J = 40
B → J = 32
C → J = 17

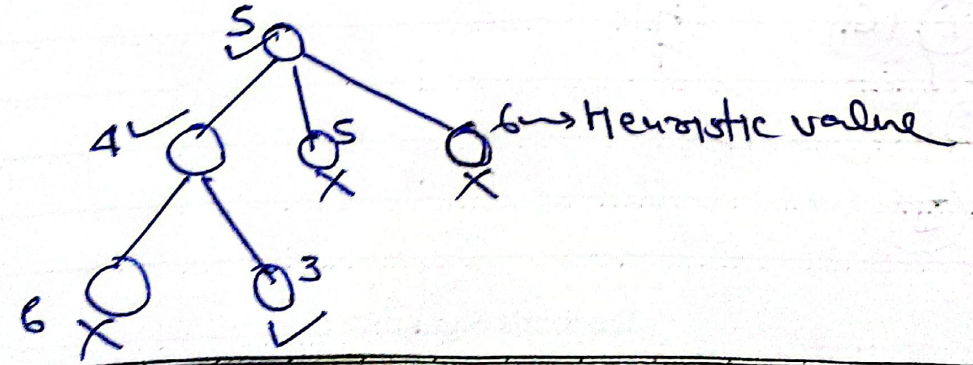
Teacher's Signature : _____

Hill Climbing Algo

- Local Search Algo
  (It has knowledge of local domain only not the knowledge of global domain (problem).
- Greedy Approach (Take best first)
- No Backtracking
  (It moves while it is finding best if it is not finding best it will stop it will not backtrack)

Algo

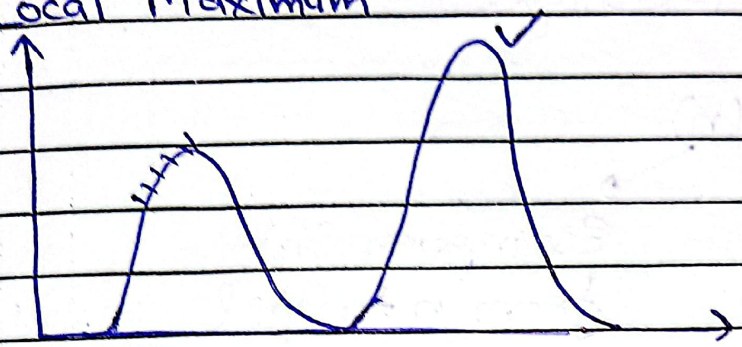
→ Evaluate the initial state
→ Loop until a solution is found or there are no operators left
  Select and apply a new operator
  Evaluate the new state
  If goal then quit
  If better than current state then it is new current state



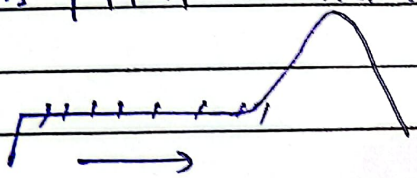
Expt. No. _____ Date. / /
Page No. _____

Problems in Hill Climbing

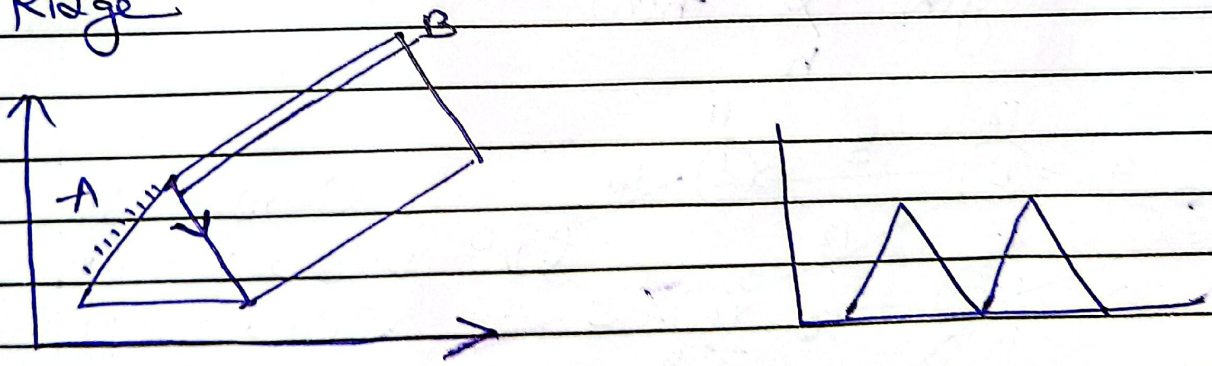
① Local Maximum



② Plateau / Flat Maximum



③ Ridge



Teacher's Signature : _____

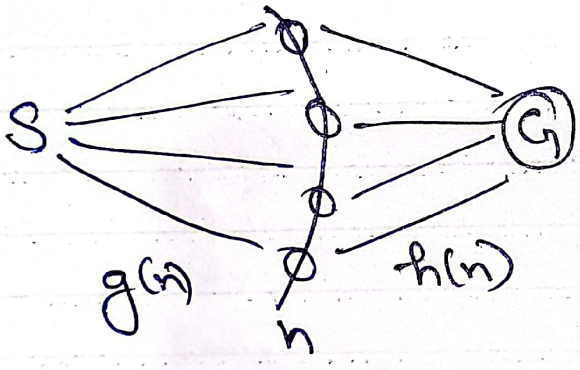
A* Algorithm

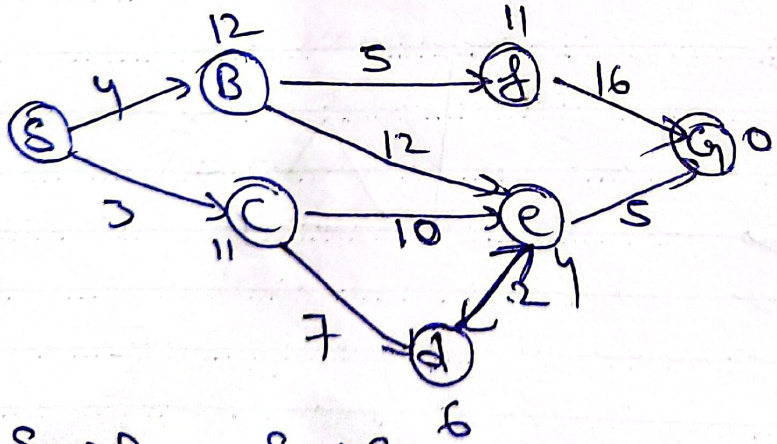
Admissible (It guarantees optimal solution)

Informed Searching Technique

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

- $g(n)$: Actual Cost from start node to n
- $h(n)$: Estimation Cost from n to Goal Node





S → B
16

S → C
14

✓ S → B = 16
✓ S → C = 14
SC → e = 17
✓ SC → d = 16
SB → f = 20
SB → e = 20
✓ SCd → e = 16
SCde → G = 17

Expt. No. Date, / / Page No.

How to ~~Set~~ make A^* Admissible

Estimated Actual

$h(n) \leq h^*(n)$ → underestimation

$h(n) \geq h^*(n)$ → overestimation

~~(optimal)~~

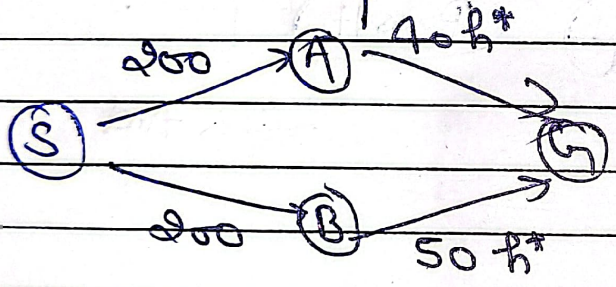
$h(n)$ → heuristic value

$h(n)$ → Estimated

S → n → G

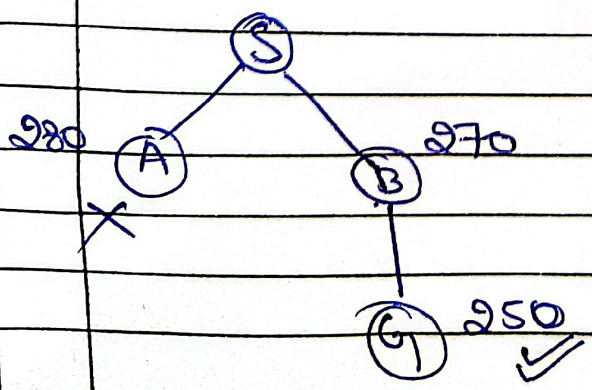
$g(n)$ $h^*(n)$ → Actual

In case of underestimation it gives optimal solution always



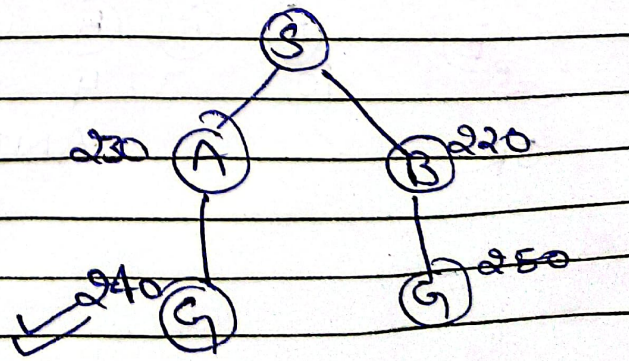
Case I Overestimation

$h(A) = 80$
$h(B) = 70$
$> h^*$



Case II Underestimation

$h(A) = 30$
$h(B) = 20$
$< h^*$



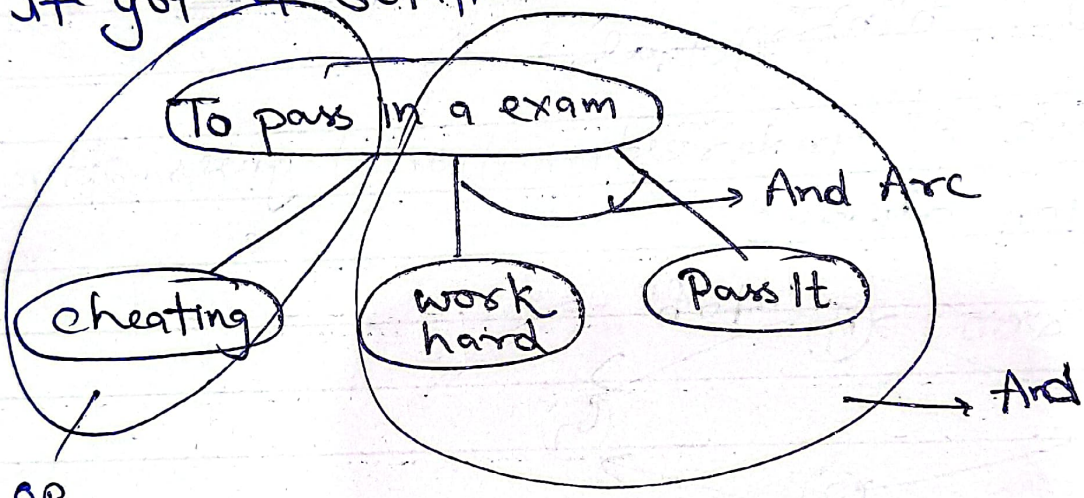
Teacher's Signature :

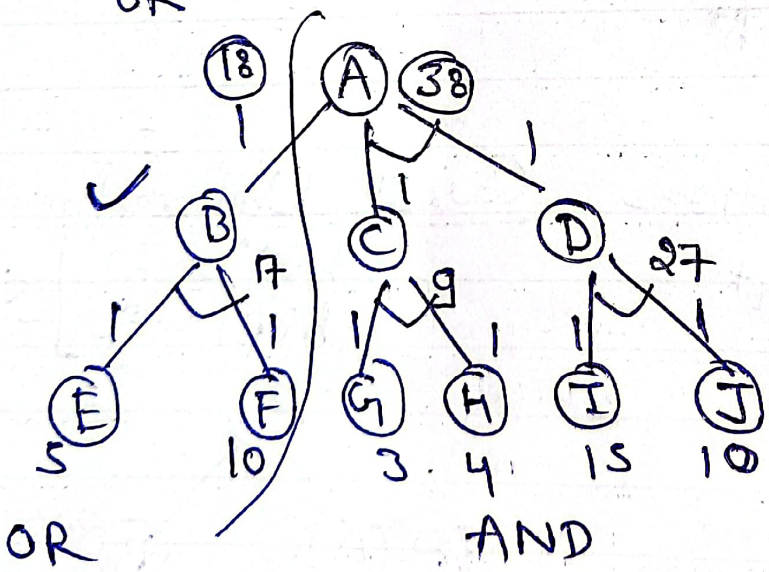
AO* Algorithm

It is based on AND/OR Graph

It works based on Problem Decomposition (Breakdown into smaller pieces)

AO* does not explore all the solution paths once it got a solution.



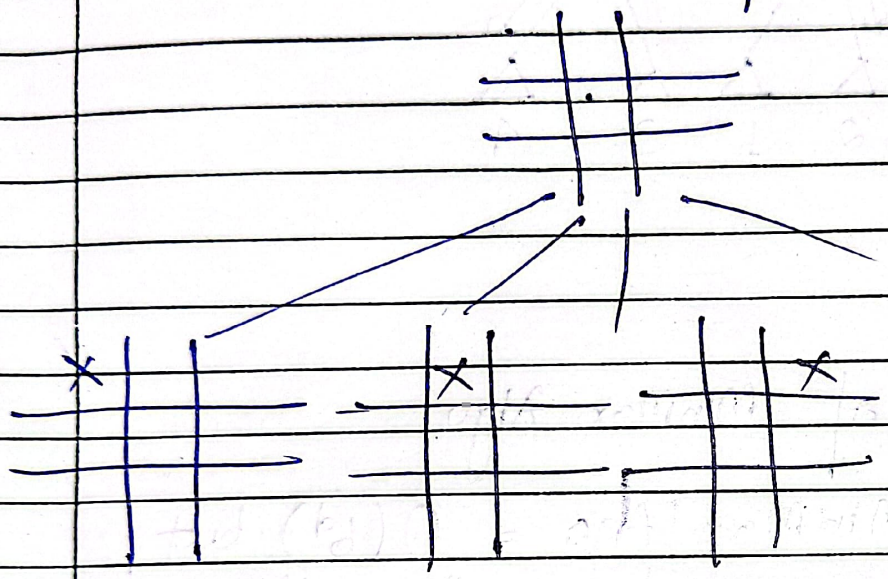


Expt. No. 21/3/26 Date. / /
Page No.

Introduction to Game Playing

1. Min Max Algorithm
2. Alpha Beta (α-β) Pruning

Game Tree / Search Tree



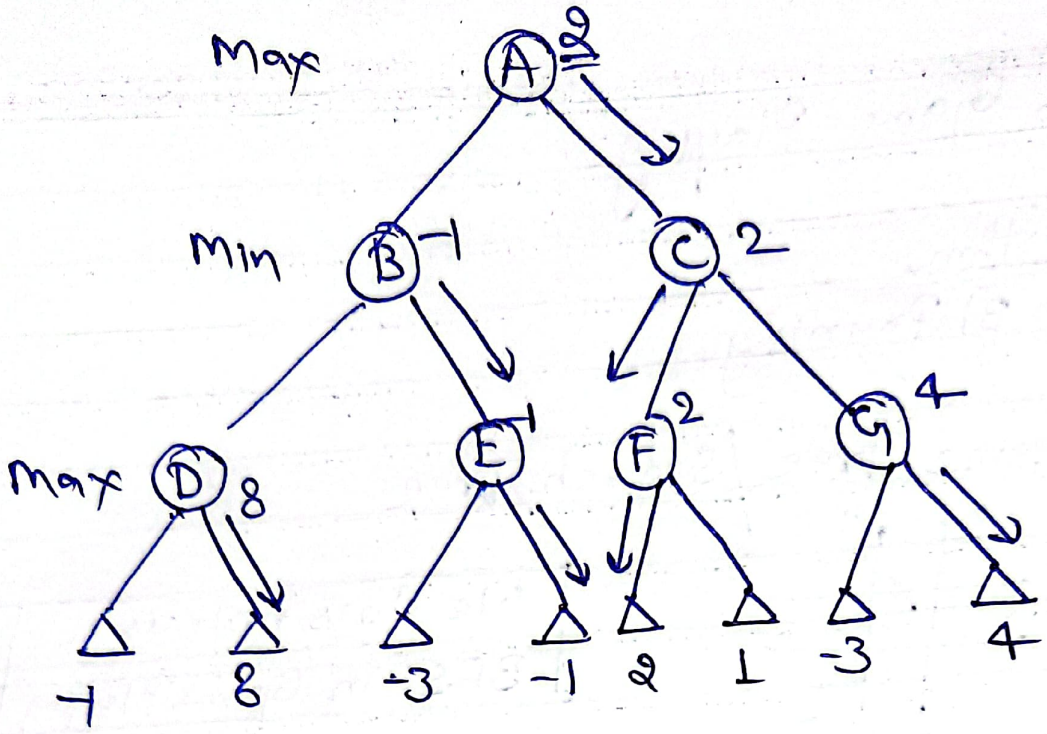
We does not use BFS in Game Playing

Utility means rewards after winning

Min Max Algorithm

1. It is a backtracking algorithm.
2. It uses best move strategy for winning.
3. Max will try to maximize its utility (Best Move)
4. Min will try to minimize its utility (Worst Move)

Teacher's Signature :



$O(b^d)$

α-β Pruning

→ Enhanced version of MinMax Algo

→ Time Comp of MinMax Algo = $O(b^d)$ but α-β pruning time comp is lesser than this.

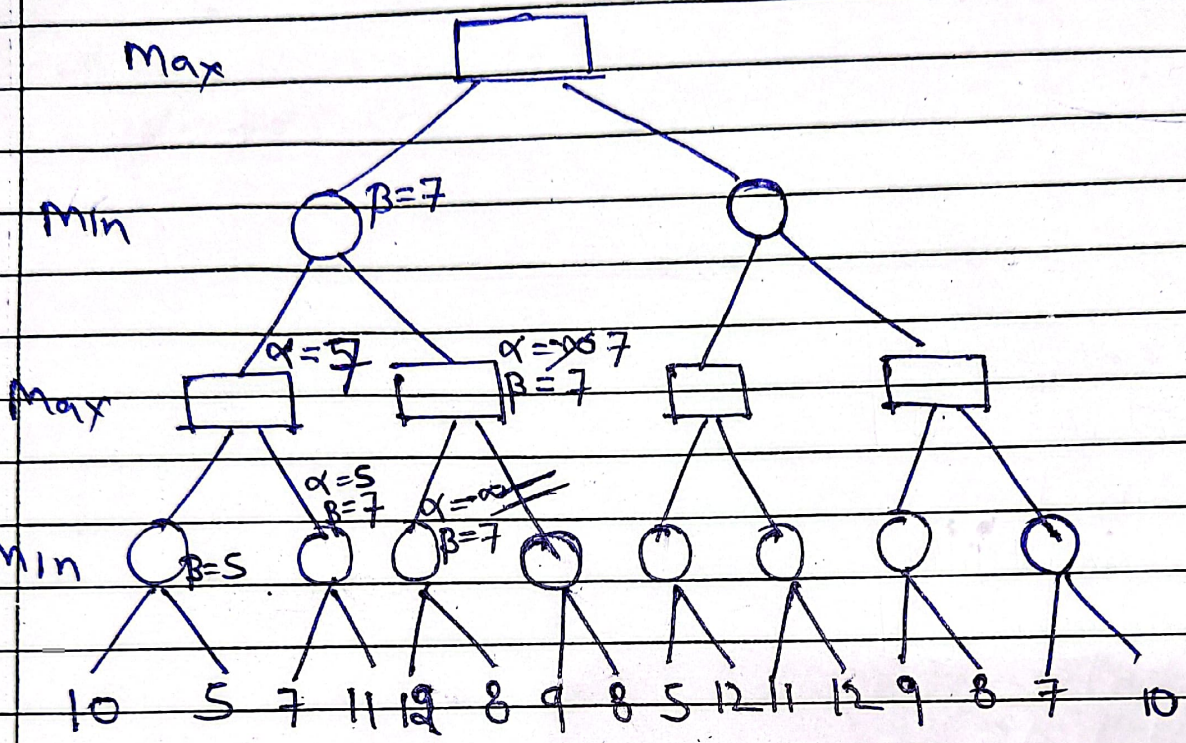
→ Cutt off search by exploring less no of nodes

α - β
| |
Max Min

→ If we have found one path or solution then we will prun (cut off) other paths.

Expt. No. _____ Date. / /
Page No. _____

We have to pruning (ignore) the branch if $\alpha \geq \beta$



Teacher's Signature : _____