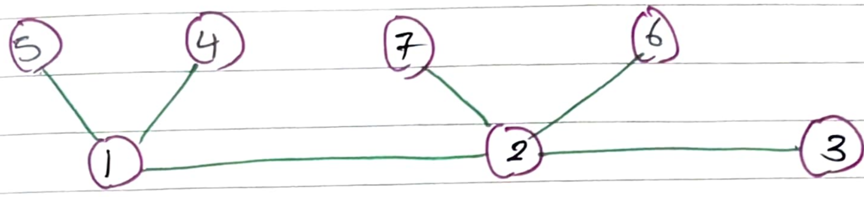
EcoNāma Page No.
Date / /

AI Unit 2

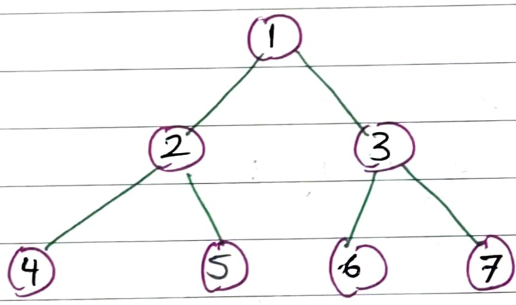
→ BFS and DFS

1]



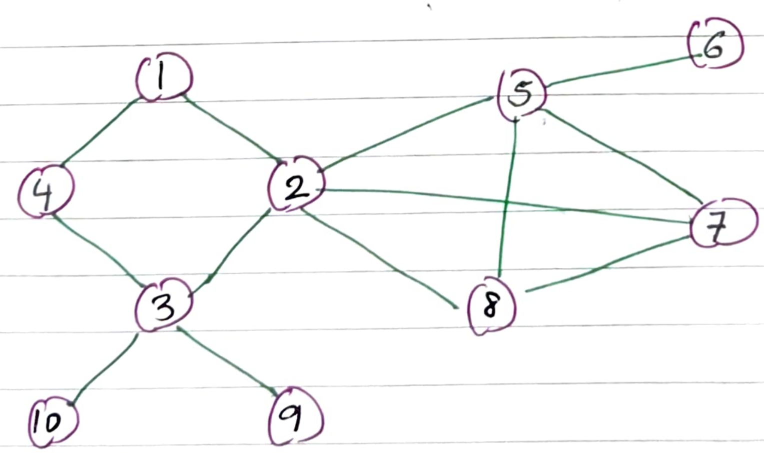
BFS : 1, 2, 4, 5, 3, 7, 6
DFS : 1, 2, 3, ~~4~~, ~~5~~ 6, 7, 4, 5

2]



BFS : 1, 2, 3, 4, 5, 6, 7 [Level order]
DFS : 1, 2, 4, 5, 3, 6, 7 [Preorder]

3]



EcoNama Page No.
Date / /

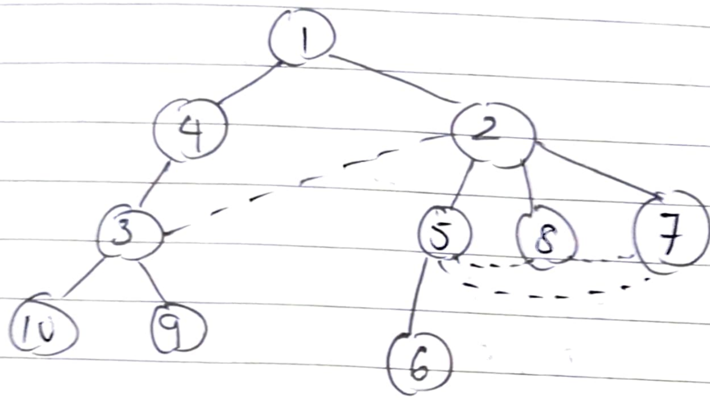
BFS:

Queue:

~~1~~	~~4~~	~~2~~	~~3~~	~~5~~	7	8	10	9	6

Ans: 1, 4, 2, 3, 5, 7, 8, 10, 9, 6

BFS Spanning tree:

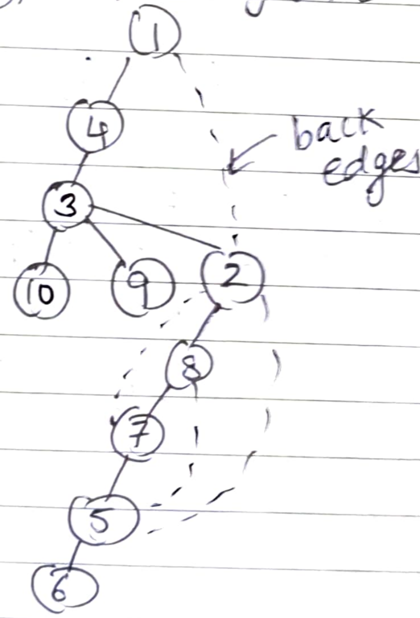


DFS:

Ans: 1, 4, 3, 10, 9, 2, 8, 7, 5, 6

Stack

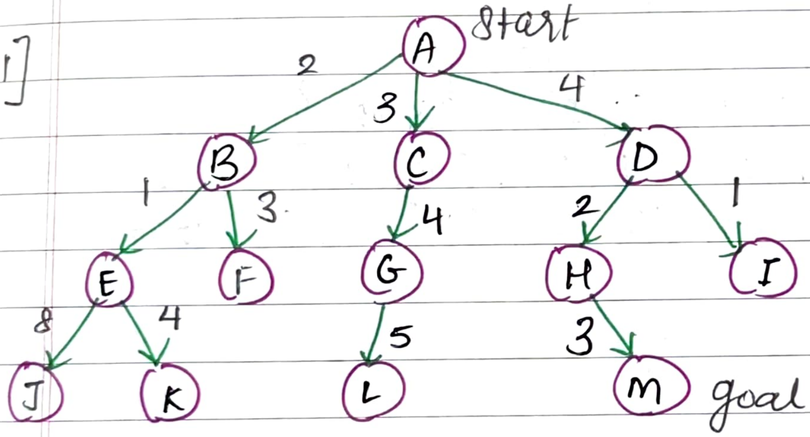
DFS spanning tree



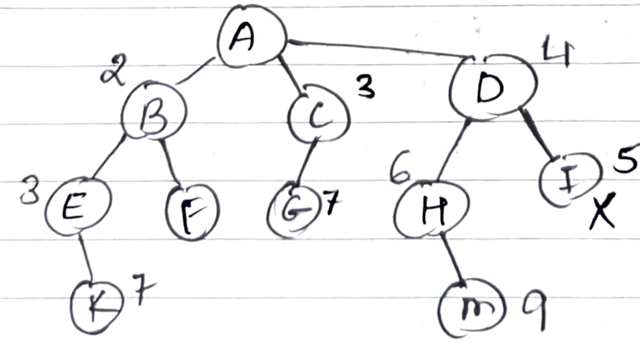
EcoNama Page No.
Date / /

→ Uniform cost search
1] Optimal solution when costs are non-negative
2] Priority queue is used (min heap).
3] It is an uninformed search algorithm
4] Used to find the least-cost path from a start to goal node in a weighted graph.
5] Variant of Dijkstra's algorithm
6] Explores nodes with lowest cumulative cost first
7] The optimal solution is not guaranteed within reasonable time and can get stuck in an infinite loop.

1]

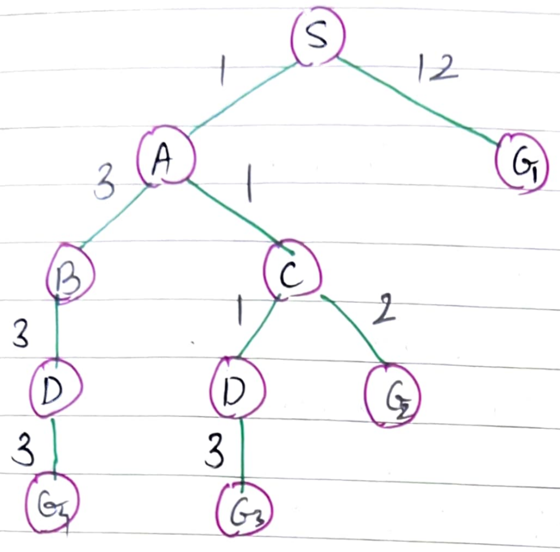


PQ : (B,2), (C,3), (D,4), (E,3), (F,5), (J,11), (K,7), (G,7), (H,6), (I,5), (M,9), (L,12)



A → D → H → M   cost = 9.

2]



Initialization : { (S,0) }
Iteration 1 : { (A, 1), (G1, 12) }
Iteration 2 : { (B, 4), (C, 2), {(G1, 12)} }
Iteration 3 : { (B, 4), (D, 3), (G2, 4), (G1, 12) }
Iteration 4 : { (B, 4), (G3, 6), (G2, 4), (G1, 12) }
Iteration ~~5 : {~~

∴ Ans : S → A → C → G_2.

→ A* Algorithm

1] Informed searching technique (or heuristic)

2] Informed search means that there is prior knowledge of the problem, this knowledge is called heuristic value. The goal state is kept into consideration.

3] $f(N) = g(N) + h(N)$

$g(N)$ ↓ Actual cost from start node to n

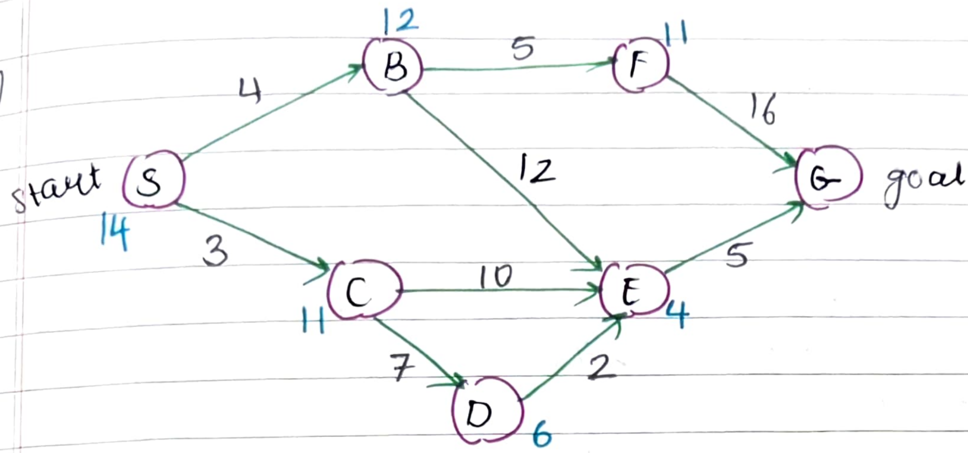
$h(N)$ ↓ Estimation cost from n to goal node

4] It is an admissible algorithm i.e. an optimal solution is guranteed.

5] T.C. = $O(V+E)$ or $O(b^d)$

EcoNāma Page No
Date / /

1]



$f(S) = 0 + 14 = 14$

$S \to B : f(B) = 4+12 = 16$ ✓

$S \to C : f(C) = 3+11 = 14$ ✓

$SC \to E : f(E) = 3+10+4 = 17$

$SC \to D : f(D) = 3+7+6 = 16$ ✓

$SB \to F : f(F) = 4+5+11 = 20$

$SB \to E : f(E) = 4+12+4 = 20$.

$SCD \to E : f(E) = 3+7+2+4 = 16$ ✓

$SCDE \to G : f(G) = 3+7+2+5 = \underline{\underline{17}} \to$ Min value.

$\therefore$ Path $= S \to C \to D \to E \to G$

→ How to make A^* admissible

1] $h(n) \leq h^*(n)$ – underestimation

$h(n) \geq h^*(n)$ – overestimation

Here,

$h(n)$ – estimated value

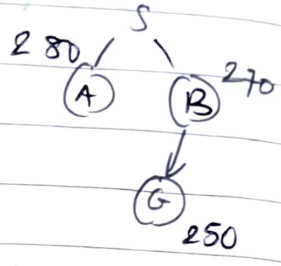
$h^*(n)$ – actual/optimal value.

$g(A) = 200$

$g(B) = 200$

Case I: overestimation

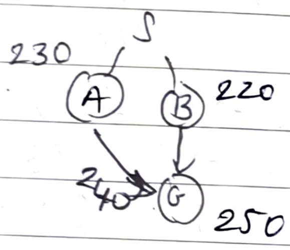
$h(A) = 80$
$h(B) = 70$ $\}$ $> h^*$



$f(A) = 200 + 80 = 280$
$f(B) = 200 + 70 = 270$
$f(G) = g(G) + h(G)$
$g(G) = 200 + 50 = 250$
$\therefore f(G) = 250 + 0 = 250.$
$250 < 270$ → overestimation

Case II: underestimation

$h(A) = 30$
$h(B) = 20$
$f(A) = 200 + 30 = 230$
$f(B) = 200 + 20 = 220$
$g(G) = 200 + 50 = 250$
$f(G) = 250 + 0 = 250$



~~250~~ → through B. $250 > 220, 230.$
$f(G) = 200 + 40 = 240$
↳ through A
$\therefore S \to A \to G.$

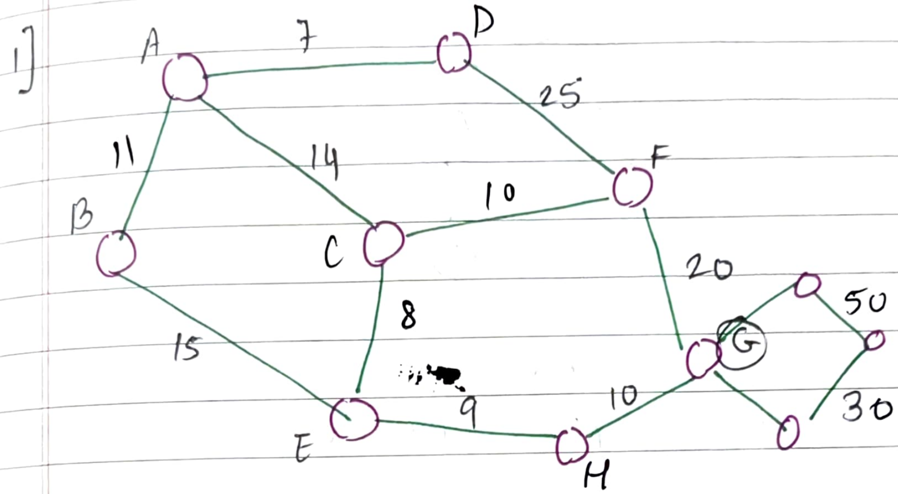
Conclusion: Use underestimation for accurate results than and more optimal solution.

→ Greedy Best First Search Algorithm

1] Informed, Heuristic algorithm

2] $f(n) = h(n)$.

1]



Straight Line dist. $= \sqrt{(x_2-x_1)^2 + (y_2-y_1)^2}$

A → G = 40.

B → G = 32

C → G = 25

D → G = 35

E → G = 19

F → G = 17

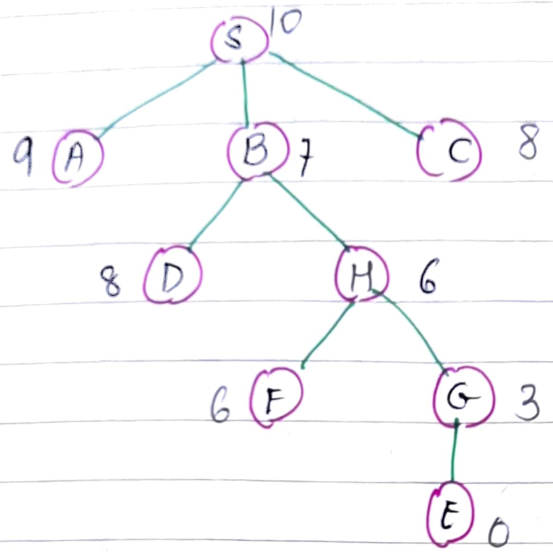
H → G = 10

G → G = 0.

Answer : A → C → F → G

EcoNāma Page No. Date / /

2]



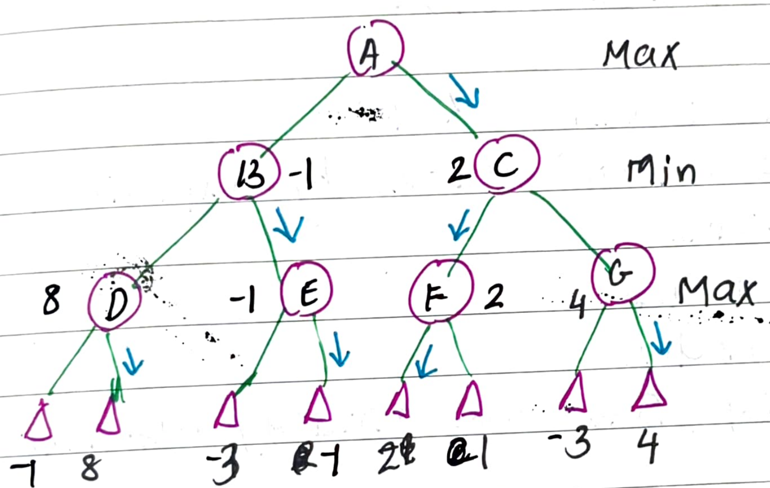
	Open		Closed	
	Node	h(n)	Node	parent
1.	S	10		
2.			S	
	A	9		
	B	7 ✓		
	C	8		
3.			B	S
	C	8		
	A	9		
	H	6 ✓		
	D	8		
4.	~~C 8~~		H	B
	C	8		
	D	8		
	A	9		
	G	3 ✓		
	F	6		
5.	F	6	G	H
	D	8		
	A	9		
	E	0 ✓		

Answer : S → B → H → G → E

Minimax Algorithm

1] Back tracking algorithm
2] Best move strategy used
3] Max → Best move (maximize utility)
Min → worst move (minimize utility)
4] Time complexity: $O(b^d)$
b - branching factor (no. of possible children).
d - depth

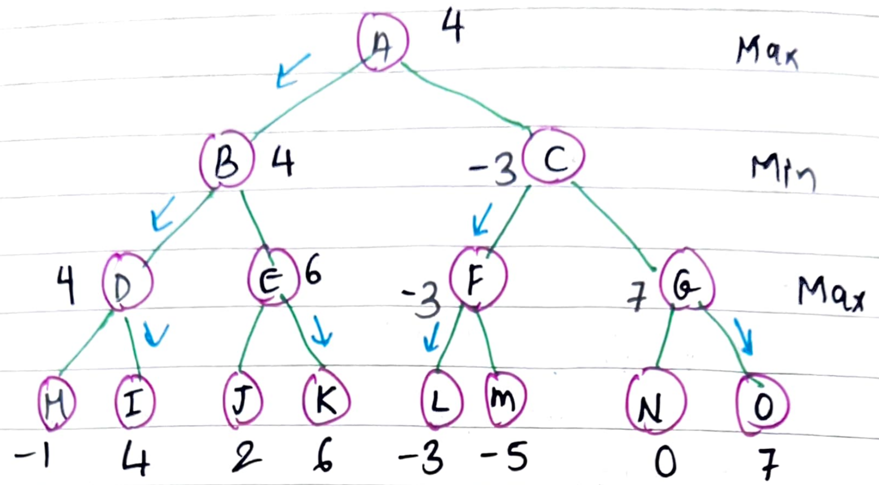
1]



* Min tries to minimize your own utility so that you can't win. Min is the opponent.

A → C → F (guarantee of getting 2 utility).

2]



A → B → D → I (4).

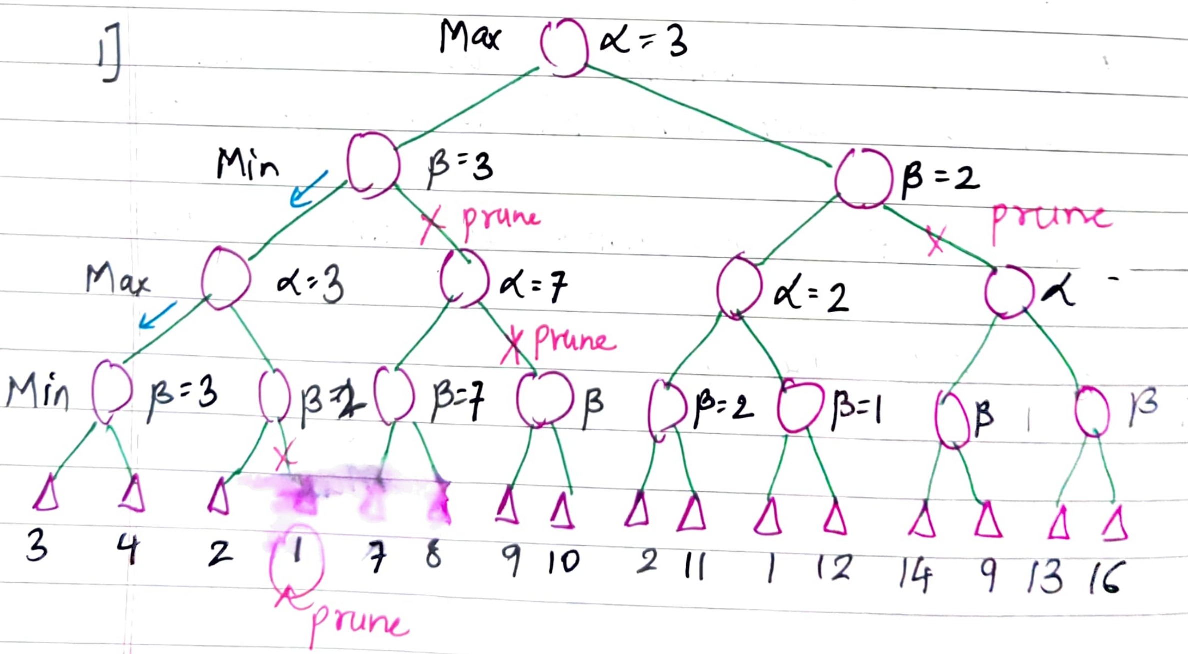
→ Alpha Beta Pruning ($\alpha - \beta$)

1] It helps to cut off search by exploring less no. of nodes.

2) α - Max nodes
β - Min nodes.

~~3] Backtracking, Best move strategy algorithm.~~

1]



Pruning condition: $\alpha \geq \beta$

1] If (Max, Min) is not given start from Max at root.

2] Give ~~∞~~ $\alpha = -\infty$, $\beta = \infty$ from root and Keep giving same values to the left child.

3] Fill the entire leftmost tree with these values.

4] Chang α values at max level and β values at min level.

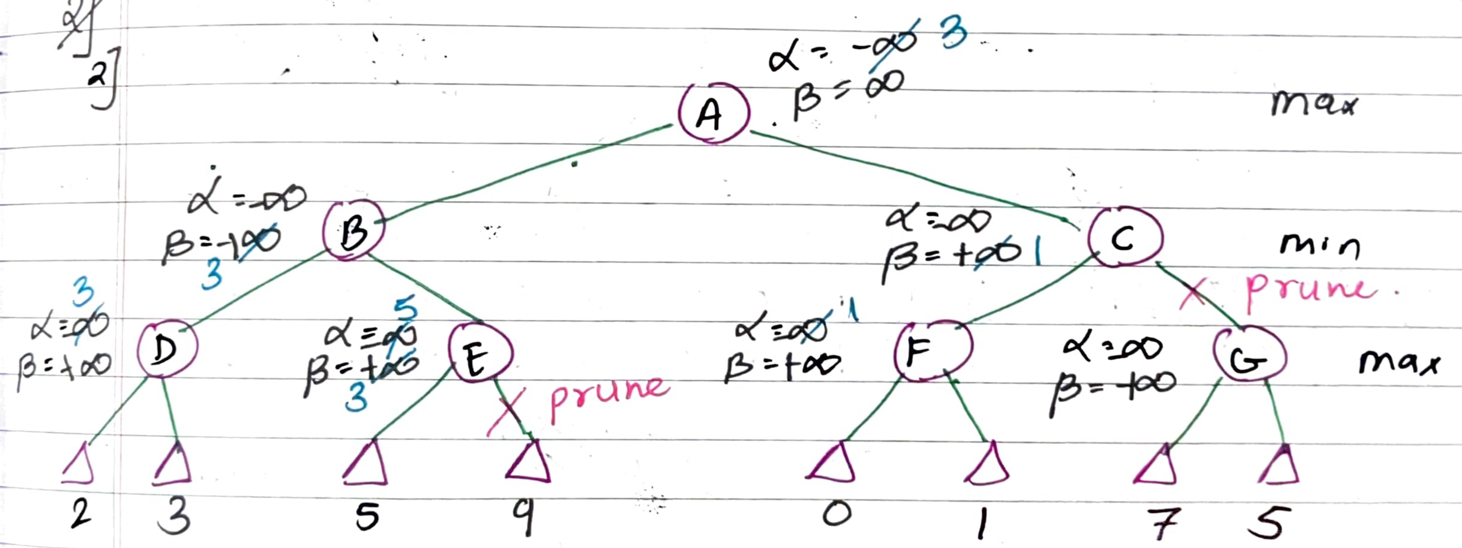
5] Start from the leaf node ~~and~~ of leftmost tree, check its left child compare acc. to the level and update with respective value then compare the right child & update.

6] If $\alpha \geq \beta$, then prune the path.

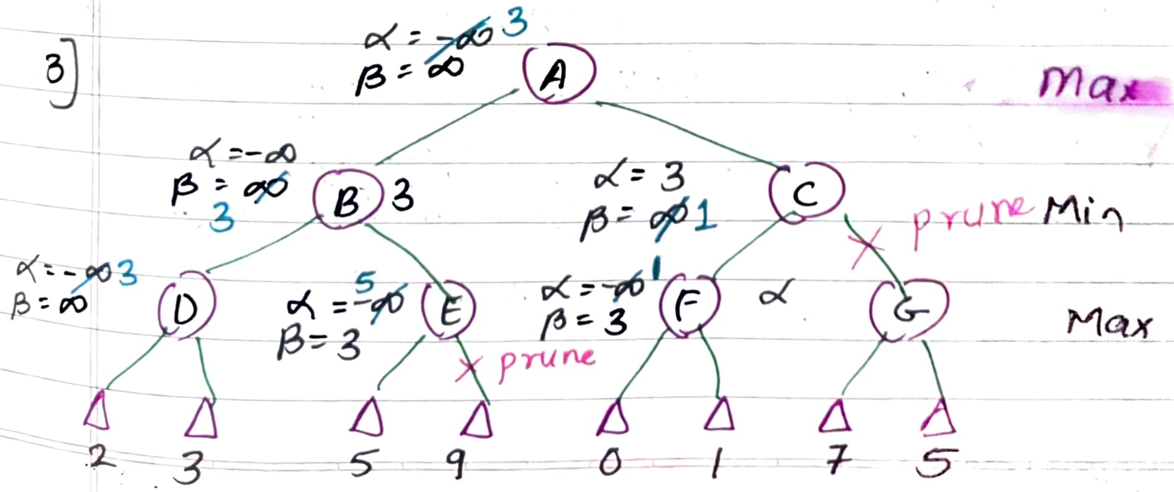
7] Time complexity = $O(b^{d/2})$.

~~2]~~

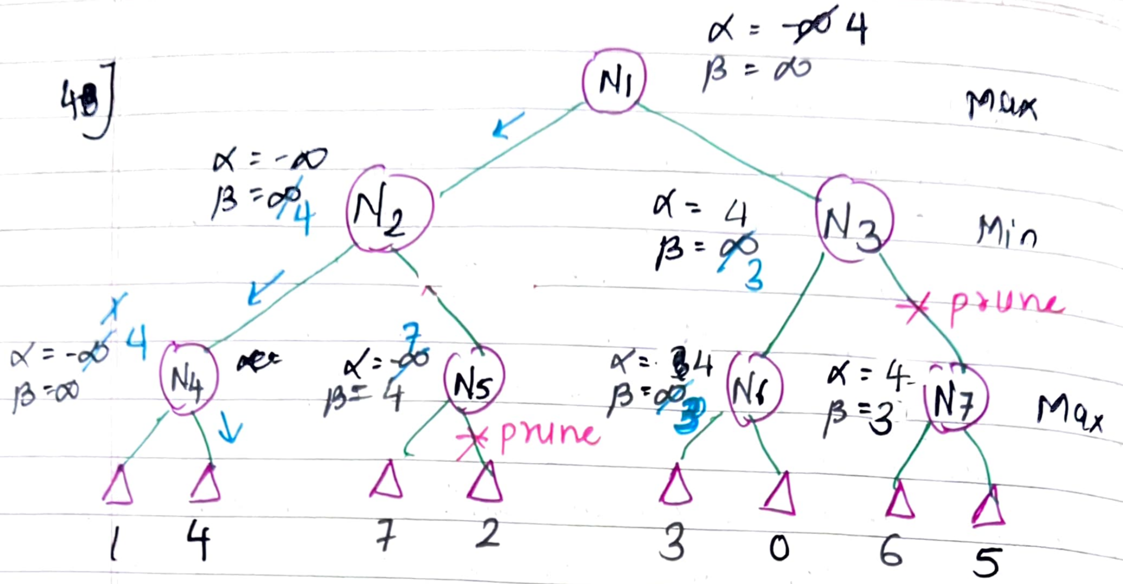
2]



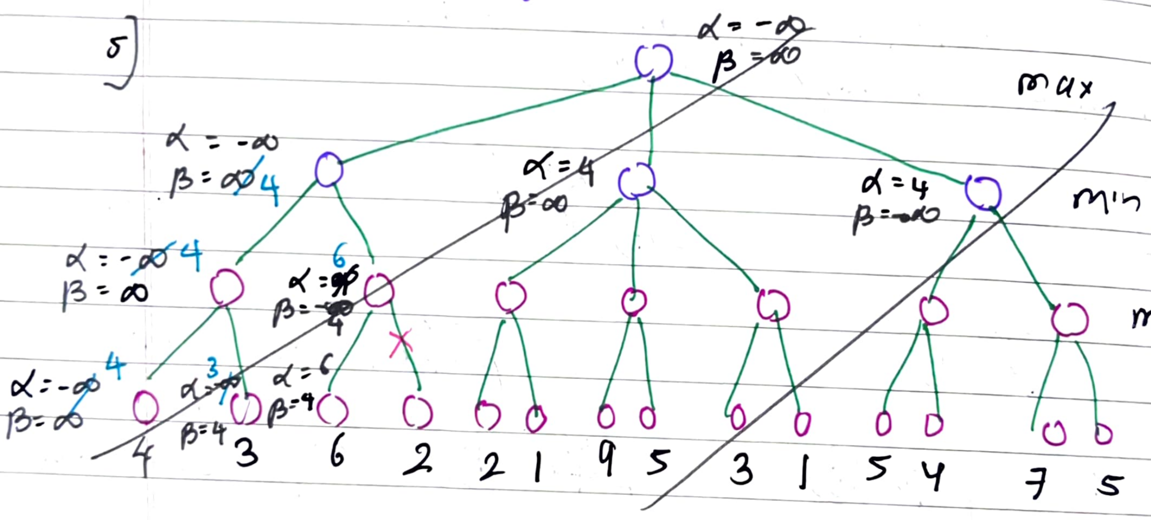
3]



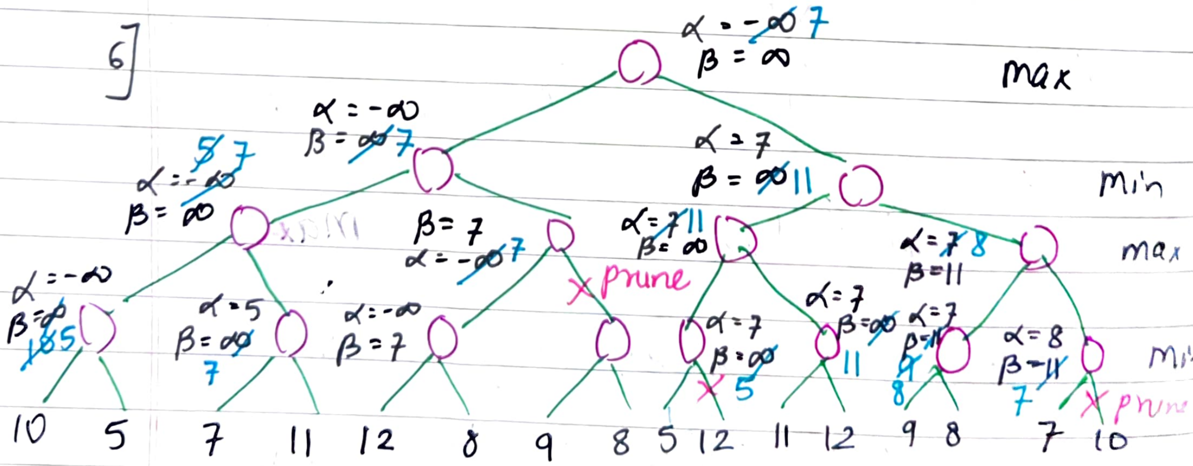
4B]

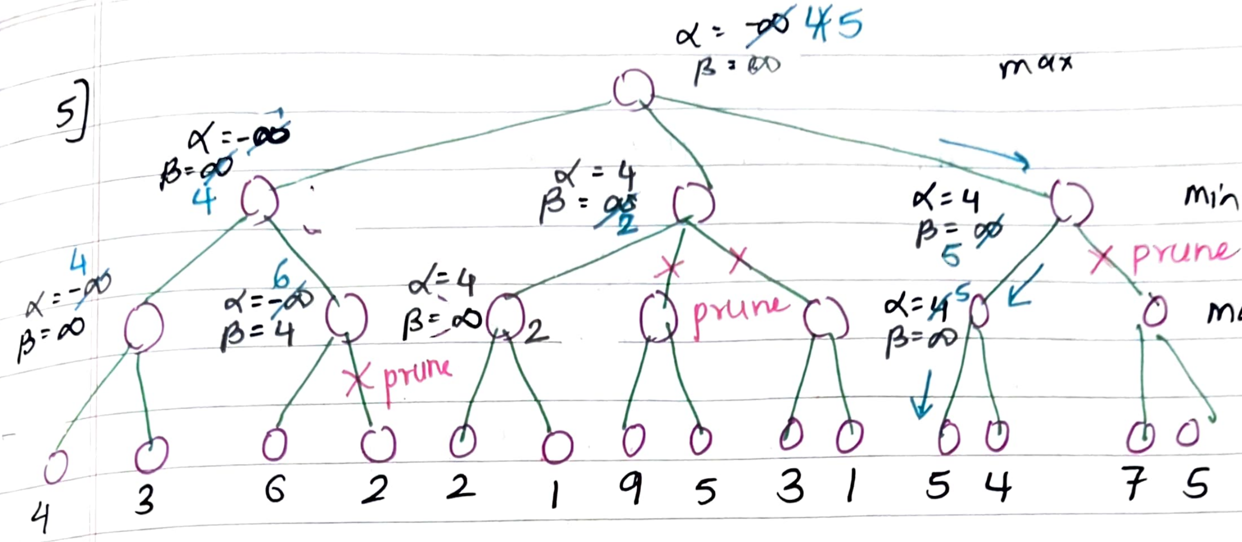

N1
N2
N3
N4
N5
N6
N7
Max
Min
Max
prune
prune
1 4 7 2 3 0 6 5

5]


max
min
max
4 3 6 2 2 1 9 5 3 1 5 4 7 5

6]


max
min
max
min
prune
prune
10 5 7 11 12 8 9 8 5 12 11 12 9 8 7 10

5] 

→ Hill Climbing Algorithm

1] Evaluate the initial state.

2] Loop until a solution is found or there are no options left.

(i) Select and apply a new operation.

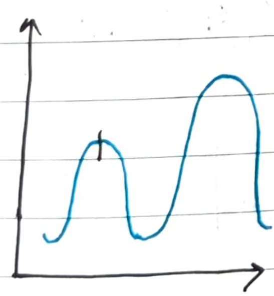
(ii) Evaluate the new state.

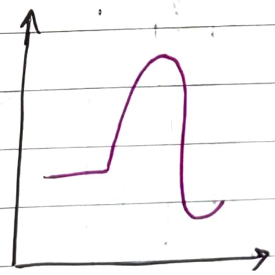
(iii) If goal is reached then quit.

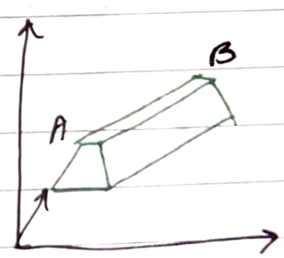
(iv) If new state is better than the current state then it becomes the current state.

3] Limitations

(i) Local Maxima (ii) Plateau (iii) Ridge







[Does not change direction]

→ 8 tiles/puzzle problem:

Using A* method (Heuristic)

1] Initial state:

2	8	3
1	6	4
7		5

Final state:

1	2	3
8		4
7	6	5

2	8	3
1	6	4
7		5

g = 0
h = 4
f = 4.

Step 1:

2	8	3
1	6	4
	7	5

g = 1
h = 5
f = 6

2	8	3
1		4
7	6	5

g = 1
h = 3
f = 4

2	8	3
1	6	4
7	5	

g = 1
h = 5
f = 6

Step 2:

2	8	3
	1	4
7	6	5

g = 2
h = 3
f = 5

2		3
1	8	4
7	6	5

g = 2
h = 3
f = 5

2	8	3
1	4	
7	6	5

g = 2
h = 4
f = 6

2	8	3
1	6	4
7		5

g = 2
h = 4
f = 6

(don't draw this)

Step 3:

2	8	3
	1	4
7	6	5

2		3
1	8	4
7	6	5

2	8	3
7	1	4
	6	5

g = 3
h = 4
f = 7

	8	3
2	1	4
7	6	5

g = 3
h = 3
f = 6

	2	3
1	8	4
7	6	5

g = 3
h = ~~3~~ 2
f = ~~6~~ 5

2	3	
1	8	4
7	6	5

g = 3
h = 4
f = 7

EcoNim Page No. Date / /

Step 4:

	8	3
2	1	4
7	6	5

8		3
2	1	4
7	6	5

~~g = 4~~
~~h = 3~~
~~f = 7~~

	2	3
1	8	4
7	6	5

1	2	3
	8	4
7	6	5

g = 4
h = 1
f = 5

1	2	3
7	8	4
	6	5

g = 5
h = 2
f = 7

1	2	3
8		4
7	6	5

g = 5
h = 0
f = 5.

(Final state).

2) Initial

2	8	3
1	6	4
7		5

Final

1	2	3
8		4
7	6	5

Using mahattan dist.

Step 1:

2	8	3
1	6	4
7		5

×

2	8	3
1	6	4
	7	5

g = 1
h = 1+1+2+0+0+0+1+1
= 6
f = 7

2	8	3
1		4
7	6	5

g = 1
h = 1+1+2+0+0+0+0+0 = 4
f = 5

×

2	8	3
1	6	4
7	5	

g = 1
h = 1+1+2+0+0+0+1+1
= 6
f = 7.

Step 2:

2	8	3
1		4
7	6	5

✓

2		3
1	8	4
7	6	5

$g = 2$

$h = 1+1+0+0+0+0+0+1$

$= 3$

$f = 5$

✗

2	8	3
	1	4
7	6	5

$g = 2$

$h = 1+2+2+0+0+0+0+0$

$= 5$

$f = 7$

✗

2	8	3
1	4	
7	6	5

$g = 2$

$h = 1+1+2+1+0+0+0+0$

$= 5$

$f = 7$

Step 3:

2		3
1	8	4
7	6	5

✓

	2	3
1	8	4
7	6	5

$g = 3$

$h = 1+1+0+0+0+0+0+0$

$= 2$

$f = 5$

✗

2	3	
1	8	4
7	6	5

$g = 3$

$h = 1+1+1+1+0+0+0+0$

$= 4$

$f = 7$

Step 4:

	2	3
1	8	4
7	6	5

↓

1	2	3
	8	4
7	6	5

$g = 4$

$h = 0+0+0+1+0+0+0+0+0$

$= 1$

$f = 5$

×

1	2	3
7	8	4
	6	5

$g = 5$

$h = 2$

$f = 7$

✓

1	2	3
8		4
7	6	5

final

$g = 5$

$h = 0$

$f = 5$