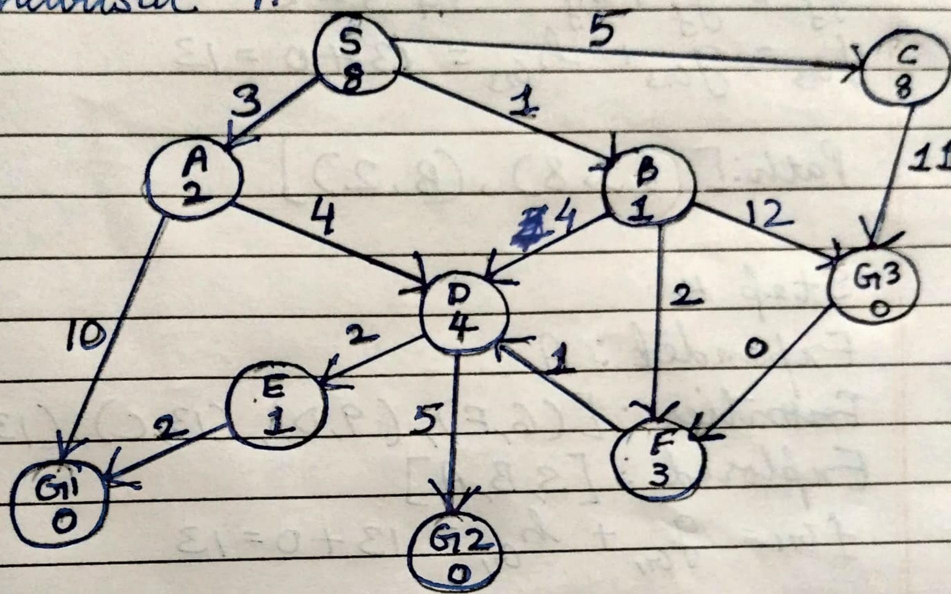
PAGE NO.
DATE / /

AI - Assignment 1
Theory

Aditya Aggarwal
2022028

Answer 1

PART-A : heuristic A*



Step 1
Expanded : ~~[/]~~ None
Frontier : [8, S]
Explored : [ ]
$g = 0$, $h = 8$ $\Rightarrow f(n) = 8$
Path : ~~[(S, 8)]~~ [ ]

Step 2
Expanded : S
Frontier : [(2, B), (5, A), (13, C)]
Explored : [S]

$f_B = g_B + h_B = 1+1 = 2$

$f_A = g_A + h_A = 3+2 = 5$

$f_c = g_c + h_c = 5+8 = 13$

Path : [(S, 8) ~~, (2, B)~~]

PAGE NO
DATE / /

Step 3
Expanded: B
Frontier : [(5,A), (6,F), (9,D), (13,C), (13,G3)]
Explored: [S,B]
$f_D = g_D + h_D = 5+4 = 9$
$f_f = g_f + h_f = 3+3 = 6$
$f_{G3} = g_{G3} + h_{G3} = 13+0 = 13$

Path: [(S,8), (B,2)]

Step 4
Expanded : A
Frontier : [(6,F), (9,D), (13,C), (13,G1), (13,G3)]
Explored : [S,B,A]
$f_{G1} = g_{G1} + h_{G1} = 13+0 = 13$

Path: [(S,8), (A,5)]

Step 5
Expanded : F
Frontier : [(8,D), (9,D), (13,C), (13,G1), (13,G3)]
Explored : [S,B,A,F]
$f_D = g_D + h_D = 4+4 = 8$

Path : [(S,8), (B,2), (F,6)]

Step 6
Expanded: D
Frontier : [(7,E), (9,D), (13,C), (13,G1), (13,G3)]
Explored: [S,B,A,F,D]
$\cancel{f_D = g_D + h_D = 8+9 = 17}$
$f_{G2} = g_{G2} + h_{G2} = 9+0 = 9$
$f_E = g_E + h_E = 6+1 = 7$

Path : [(S, 8) , (B, 2) , (F, 6) , (D, 8)]

Step 7
Expanded : E
Frontier : [(8, G1), (9, D), (13, C), (13, G1), (13, G3)]
Explored : [S, B, A, F, D, E]

$\cancel{f_E = g_E + h_E = 6+1=7}$
Path : [(S, 8), (B, 2), (F, 6), (D, 8), (E, 6)]

Step 8
Expanded : G1
Frontier : [(9, D), (13, C), (13, G1), (13, G3)]
Explored : [S, B, A, F, D, E, G1]
Path : [(S, 8), (B, 2), (F, 6), (D, 8), (E, 6), (G1, 8)]

NOTE : The algorithm would not terminate after this but the optimal solution is found at step 8.

final Path : S → B → F → D → E → G1

S ↓ start_node; G1 ↓ Goal_node

$\boxed{\text{Final cost : 8 units}}$

Explanation of algorithm : The algorithm follows a greedy approach by considering the node with minimum f value and expanding to neighbor nodes ~~##~~, storing f values in a priority queue. The value of $f(n) = g(n) + h(n)$ where $g(n)$ is actual path cost till n and $h(n)$ is the heuristic function (making the algo an informed algo).

b.) Uniform Cost Search

Step 1
Expanded : None
Frontier : [(0, S)]
Explored : [ ]
$f_S = 0$
Path : [ ]

Step 2
Expanded : S
Frontier : [(1, B), (3, A), (5, C)]
Explored : [S]
$f_B = 1$, $f_A = 3$, $f_C = 5$

Path : [(S, 0)]

Step 3
Expanded : B
Frontier : [(3, A), (3, F), (5, C), (5, D), (13, G_3)]
Explored : [S, B]
$f_F = 3$, $f_D = 5$, $f_{G_3} = 13$

Path : [(S, 0), (B, 1)]

Step 4
Expanded : A
Frontier : [(3, F), (5, C), (5, D), (13, G_1), (13, G_3)]
Explored : [S, B, A]
$f_{G_1} = 13$
Path : [(S, 0), ~~(B, 1)~~, (A, 3)]

PAGE NO.
DATE / /

Step 5
Expanded: F
Frontier: [(4, D), (5, C), (5, D), (13, G_1), (13, G_2)]
Explored: [S, B, A, F]
$f_D = 4$
Path: [(S, 0), (B, 1), (F, 3)]

Step 6
Expanded: D
Frontier: [(5, C), (5, D), (6, E), (9, G_2), (13, G_1), (13, G_3)]
Explored: [S, B, A, F, D]
$f_{G_2} = 9$, $f_E = 6$
Path: [(S, 0), (B, 1), (F, 3), (D, 4)]

Step 7
Expanded: C (chosen alphabetically)
Frontier: [(5, D), (6, E), (9, G_2) (13, G_1), (13, G_3)]
Explored: [S, B, A, F, D, C]
Path: [(S, 0), (C, 5)]

Step 8
Expanded: D
Frontier: [(6, E), (9, G_2), (10, G_2), (13, G_1), (13, G_3)]
Explored: [S, B, A, F, D, C, D]
$f_{G_2} = 10$
Path: We have better value for D already

Step 9
Expanded: E
Frontier: [(8, G_1), (9, G_2), (10, G_2), (13, G_1), (13, G_3)]
Explored: [S, B, A, F, D, C, D, E]

PAGE NO.
DATE / /

$f_{G_1} = 8$

Path: [(S,0) , (B,1), (F,3) , (D,4), (E,6)]

Step 10
Expanded: G_1
Frontier: [(9, G_2), (10, G_2), (13, G_1), (13, G_3)]
Explored: [S, B, A, F, D, C, D, E, G_1]
Path: [(S,0), (B,1), (F,3) (D,4), (E,6), (G,8)]

final path : S → B → F → D → E → G
final cost = 8 units

Explanation: The algo works in a similar fasion as the last one with one key difference: this algorithm was uninformed, hence it did not have a heuristic function.

$$f(n) = g(n)$$

↳ actual path length.

PAGE NO.
DATE / /

c) Iterative deepening A*

Step 1
Expanded: None
Frontier: [(8, S)]
Explored: [ ]
Threshold: 8

Step 2
Expanded: S
Frontier: [(5, A), (2, B), (13, C)]
Explored: [S]
Threshold: 8

Step 3
Expanded: A
Frontier: [(13, G1), (2, B), (13, C)]
Explored: [S, A]
Threshold: 8

Step 4
Expanded: None
Frontier: [(2, B), (13, C)]
Explored: [S, A]
Threshold: 8

Step 5
Expanded: B
Frontier: [(9, D), (6, F), (13, G3), (13, C)]
Explored: [S, A, B]
Threshold: 8

PAGE NO.
DATE / /

Step 6
Expanded: None
Frontier: No Expansion, $Value_D$ > threshold=8
Explored: [S, A, B]
Threshold: 8

Step 7
Expanded: F
Frontier: [(8, D), (13, G3), (13, C)]
Explored: [S, A, B, F]
Threshold: 8

Step 8
Expanded: D
Frontier: [(7, E), (9, G2), (13, G3), (13, C)]
Explored: [S, A, B, F, D]
Threshold: 8

Step 9
Expanded: E
Frontier: [(8, G1), (9, G2), (13, G3), (13, C)]
Explored: [S, A, B, F, D, E]
Threshold: 8

Step 10
Expanded: G1
Frontier: [(9, G2), (13, G3), (13, C)]
Explored: [S, A, B, F, D, E, G1]
Threshold: 8

PAGE NO.
DATE / /

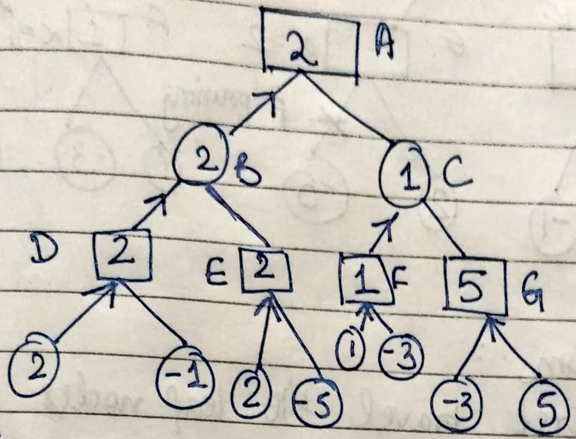
Final Path : S → A → B → F → D → E → G1
Cost = 8

NOTE : The given table is shown only for depth = 6 from depth = 0 → 5, the algorithm will not return a path. IDA ~~is~~ combined the concept of depth from IDS and heuristic function from A* algorithm.

Answer 2

PART A

Using Min max algorithm the final decisions are as follows :



Explanation :

D = max (2, -1) = 2

E = max (2, -5) = 2

F = max (1, -3) = 1

G = max (-3, 5) = 5

B = ~~max~~ min (D, E) = 2

C = min (F, G) = 1

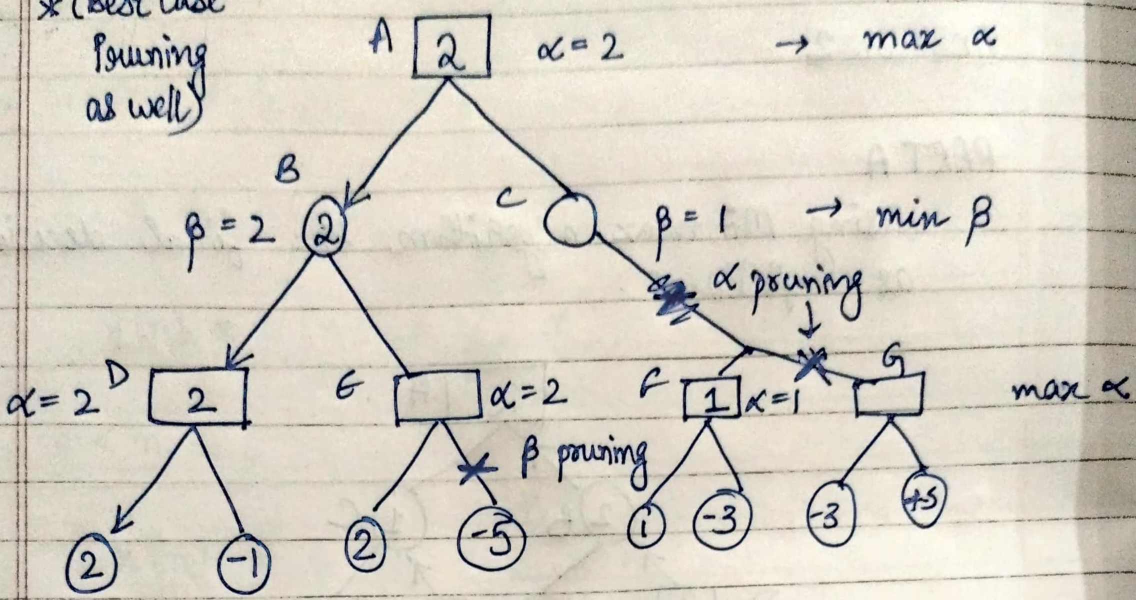
A = max (B, C) = 2

The alpha - beta pruning method prunes on the principle $\alpha \geq \beta$

Date

Page No.

* (Best Case Pruning as well)



Explanation :

→ We first travel till leaf nodes for D

→ max for D is 2, we set α = 2

→ Backtrack to B and set β = 2

→ Travel to E. Set α = left child of E = 2.

* Since $\alpha_E \geq \beta_B$, prune (β) right subtree of E

→ Backtrack to A, set α = 2

→ Travel to F. Set α = max (Children(F)) = 1 for F

→ Backtrack to C, set β = 1 for C

* $\alpha_A \geq \beta_C$, prune (α) right subtree(C)

Total nodes pruned = 4

PART B

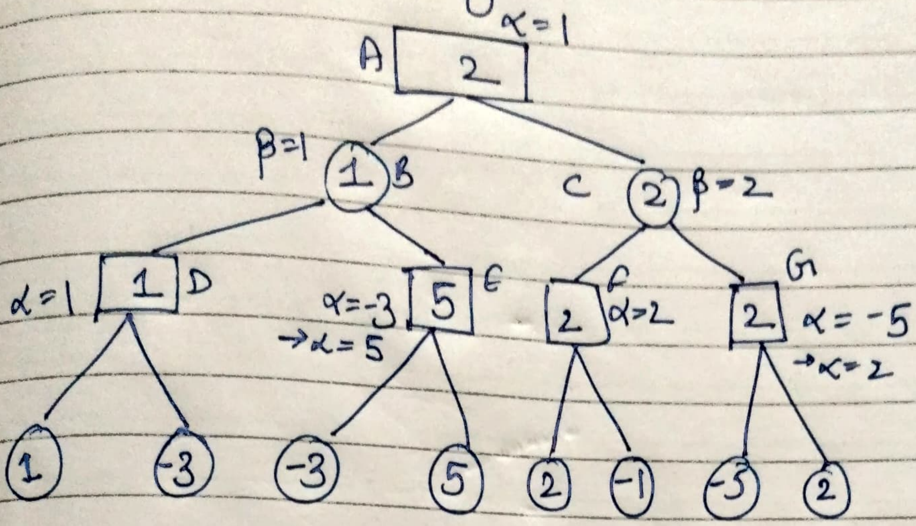
Best Case Pruning :

The above tree itself is the best case of pruning for the given leaf values.

There are 2 prunings, which is the max. amount possible for depth = 3 and no. of pruned nodes $\approx \sqrt{n}$. Hence, the above tree is the best pruning case

Worst Case Pruning

By intuitive hit & trial, we arrive at a tree with zero possible pruning:



The tree prevents α pruning by ensuring $\alpha_A < \beta_C$ and prevents β pruning by $\beta_B > \alpha_E$

Thus, no pruning occurs → worst case tree.

PART C

Without pruning:

i^{th} level in tree has b^i nodes

max level = depth

$\Rightarrow$ nodes visited $= \sum_{i=0}^{d} b^i \approx O(b^d)$

However, after best-case pruning, no. of nodes visited approximately reduce to $\sqrt{b}$

Thus, best case complexity $\approx O((\sqrt{b})^d)$

$$\boxed{T.C = O(b^{d/2})}$$