* A search strategy is defined by picking the order of node expansion.

* Uninformed / Informed Search:

→ Uninformed search algorithms looked through search space for all possible solutions of the problem without having any additional knowledge about search space.

→ Informed search algorithm contains an array of knowledge such as how far we are form the goal, path cost, how to reach to goal node, etc.

* Heuristics Search: এখানে একটা heuristic function use করা হয় to determine the least cost of each node.

→ Heuristic is a function which is used in Informed search, and it finds the most promising path.

→ Heuristic function estimates how close a state is to the goal state.

→ Goal state এর heuristic value always 0.

→ $g(n)$ calculate start to n node. $h(n)$ calculate n node to goal node

→ একটা node এর heuristic value বের করা জন্য সাধারণত Manhattan distance / Euclidean distance use করা হয়।

$$\text{Manhattan distance} = |x_2 - x_1| + |y_2 - y_1|$$

$$\text{Euclidean distance} = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

($x_{start} - x_{destination}$)

* Admissibility of the heuristic function is given as: $0 <= h(n) <= h^*(n)$.

Here, $h(n)$ = heuristic cost, and $h^*(n)$ = estimated cost.

* Greedy Best-first search:

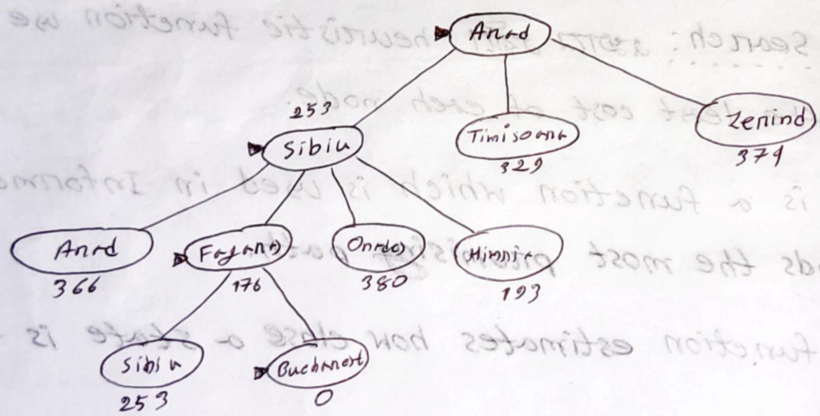
→ এখানে প্রত্যেকটি node এর যে evaluation function আছে সেটি অনুসারে এর heuristic function, $h(n)$ এর based করে calculate করা হয়।

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

→ It uses the heuristic function and search and totally ignores the path cost.

→ এখানে যে node এর heuristic value সবচেয়ে কম সে বেশি priority পাবে।

→ The greedy best. first algorithm is implemented by the priority queue.



→ এটি optimal solution দেয় না। কারণ এক্ষেত্রে

$$h(n) = 140 + 99 + 211 = 450 \text{ km}$$

$$h^*(n) = 140 + 80 + 97 + 101 = 418 \text{ km}$$

so, Admissibility : $0 \le h(n) \le h^*(n)$ [not true].

→ এই Greedy problem solve করার জন্য যে algorithm এসেছে তা হলো A^* search. যেটি আমাদের optimal solution provide করে।

Note Google maps, puzzle game using A* search.

A* search:

→ It combines the strengths of **UCS** and greedy best-first search, by which it solve the problem efficiently.

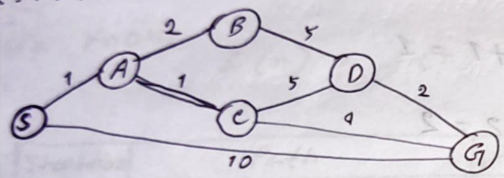
$$f(n) = g(n) + h(n)$$

f(n): estimated cost of the cheapest solution

g(n): path cost

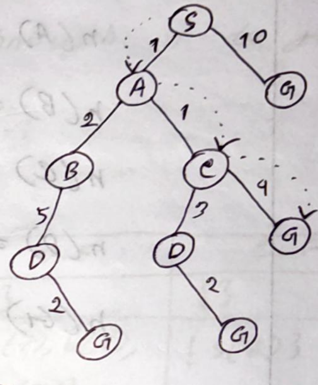
h(n): heuristic cost

Example:



state	h(n)
S	5
A	3
B	4
C	2
D	6
G	0

⇒



→ Optimal এর জন্য দুইটি property maintain করতে হবে:

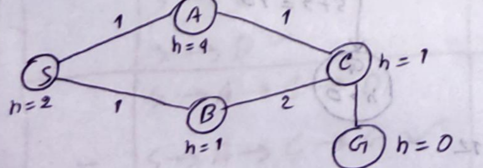
① Admissible: h(n) should be an admissible heuristic for A* tree search. An admissible heuristic is optimistic in nature.

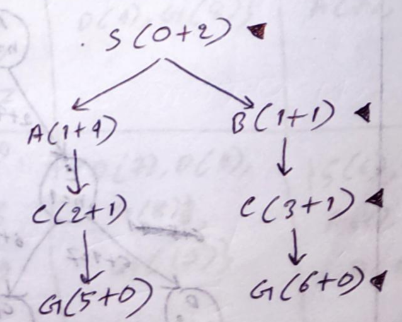
② Consistency: second required condition is consistency for only A* graph-search.

→ Time complexity: $O(b^d)$.

→ Space complexity: $O(b^d)$.

*





→ Admissibility is not enough to maintain completeness and optimality under A* graph search.

* Consider the last 2 digits of your ID is 30 and the calculation formula of $h(n)$ for all nodes are as follows:

$\therefore$ The heuristic values of S, A, B, C, D, and G will be as follows -

$h(S) = (30 \% 4) + 4 = 2+4 = 6$

$h(A) = (30 \% 7) + 3 = 2+3 = 5$

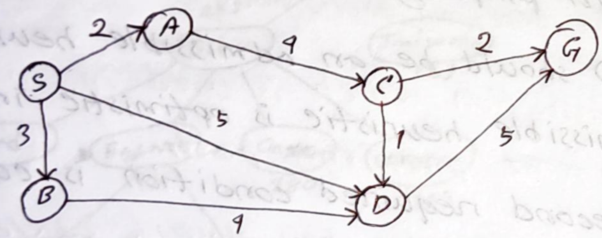
$h(B) = (30 \% 5) + 2 = 0+2 = 2$

$h(C) = (30 \% 3) + 1 = 0+1 = 1$

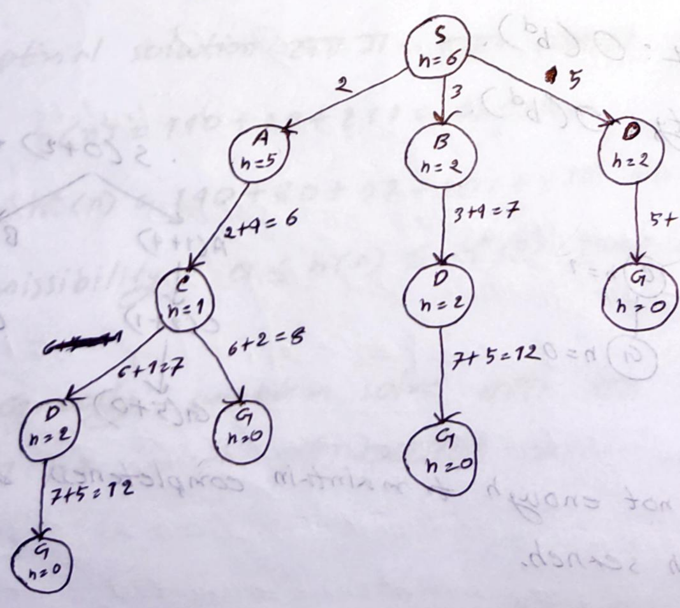
$h(D) = (30 \% 6) + 2 = 0+2 = 2$

$h(G) = 0$

The given ~~state~~ space graph:



The corresponding search tree:



Now, I will apply A* search algorithm on the search tree:

Here,

O_F = open fringe

C_F = close fringe

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

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

$f(n)$ = Estimated total cost of path through n to goal.

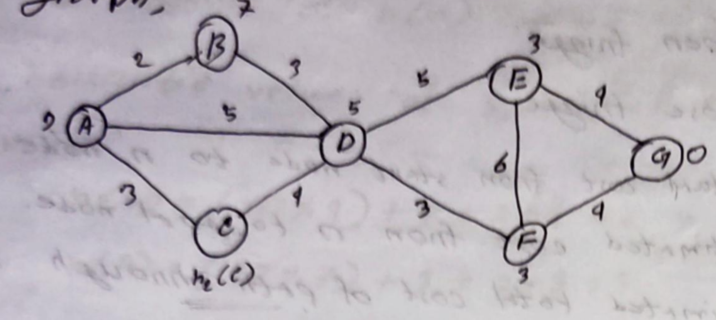
We Know,

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

Iteration	Path	g(n)	h(n)	f(n)	O_F	C_F
init	S	0	6	6	{ }	{ }
1	S→A S→B ✓ S→D	2 3 5	5 2 2	7 5 7	{A(7), B(5), D(7)}	{S(6)}
2	S→A ✓ S→B→D S→D	2 7 5	5 2 2	7 9 7	{A(7), D(7), D(9)}	{S(6), B(5)}
3	S→A→C ✓ S→B→D S→D	6 7 5	1 2 2	7 9 7	{~~A~~ D(7), D(9), C(7)}	{S(6), B(5), A(7)}
4	S→A→C→D S→A→C→G S→B→D S→D ✓	7 8 7 5	2 0 2 2	9 8 9 7	{D(7), D(9), D(9), G(8)}	{S(6), B(5), A(7), C(7)}
5	S→A→C→D S→A→C→G ✓ S→B→D S→D→G	7 8 7 10	2 0 2 0	9 8 9 10	{D(7), D(9), G(8), G(10)}	{S(6), B(5), A(7), C(7), D(9)}
6	So this iteration we find the goal node G in the close fringe. So, it will return the path S→A→C→G and will optimal cost 8.				{D(7), D(9), G(10)}	{S(6), B(5), A(7), C(7), D(9), G(8)}

4 No Qus Ans (b) ~~Fall-2020~~ Spring-2021

Given graph,



(i) In order for $h_2(C)$ to be admissible,

$$0 < h_2(C) \le \text{Optimal path cost from } C \text{ to } G$$

$$\Rightarrow 0 < h_2(C) \le cost(C \text{ to } D) + cost(D \text{ to } F) + cost(F \text{ to } G)$$

$$\Rightarrow 0 < h_2(C) \le 4 + 3 + 4$$

$$\Rightarrow 0 < h_2(C) \le 11$$

so, $h_2(C)$ is admissible for $0 < h_2(C) \le 11$.

(ii) In order for $h_2(C)$ to be consistent,

$$0 <= h_2(C) - h_2(A) <= cost(A \text{ to } C)$$

$$\Rightarrow 0 <= h_2(C) - 9 <= 3$$

$$\Rightarrow 0 <= h_2(C) <= 12$$

$$0 <= h_2(C) - h_2(D) <= cost(C \text{ to } D)$$

$$\Rightarrow 0 <= h_2(C) - 5 <= 4$$

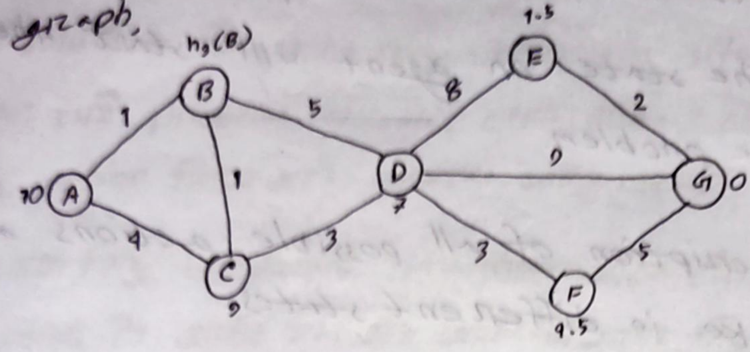
$$\Rightarrow 0 <= h_2(C) <= 9$$

so, $h_2(C)$ is consistent for $0 <= h_2(C) <= 9$.

CT-2:

2 No Qus Ans

(i) Given graph,



If $h_3(B)$ denotes the heuristic cost of node B then in order for the heuristic function h_3 to be admissible, it must fulfill the following condition

$0 \leq h_3(B) \leq$ Optimal total path cost from B to goal node.

If G is the goal node; Optimal total path cost of B is
cost(B to C) + cost(C to D) + cost(D to F) + cost(F to G)

$= 1+3+3+5 = 12$

So, for h_3 to be admissible, $h_3(B) \leq 12$

So, any value of $h_3(B)$ less than or equal to 12 and greater than or equal to 0 will make h_3 admissible. [however $h_3(B)$ cannot be 0 since B is not the goal node].

(ii) It is possible to formulate a given problem in artificial intelligence if an AI program is capable of solving or can be trained to solve it and if the problem fulfills th necessary criteria of problem formulation, such as having clearly defined states, initial state, goal state, etc.

The main components to formulate a problem are:

1. Initial state: The state an agent will start when solving the problem.

2. Actions: A description of all possible actions an agent can take in different states.

3. Transition Model: A description of what each action does.

4. Goal Test: A test to check if agent's current state is a goal, which indicates that the problem has been solved.

5. Path cost function: Assigns path cost to each of an agent's path. Agent will choose cost function that reflects its performance measure.

___ X ___