Artificial Intelligence

Daud Khan Khalil
School of Computer Sciences
INU Peshawar

Continued…

A* Tree Search

- A* Tree Search, or simply known as ***A* Search***, combines the strengths of uniform-cost search and greedy search.
- In this search, the heuristic is the summation of the cost in UCS, denoted by g(x), and the cost in greedy search, denoted by h(x).
- The summed cost is denoted by f(x).

Continued…

Heuristic

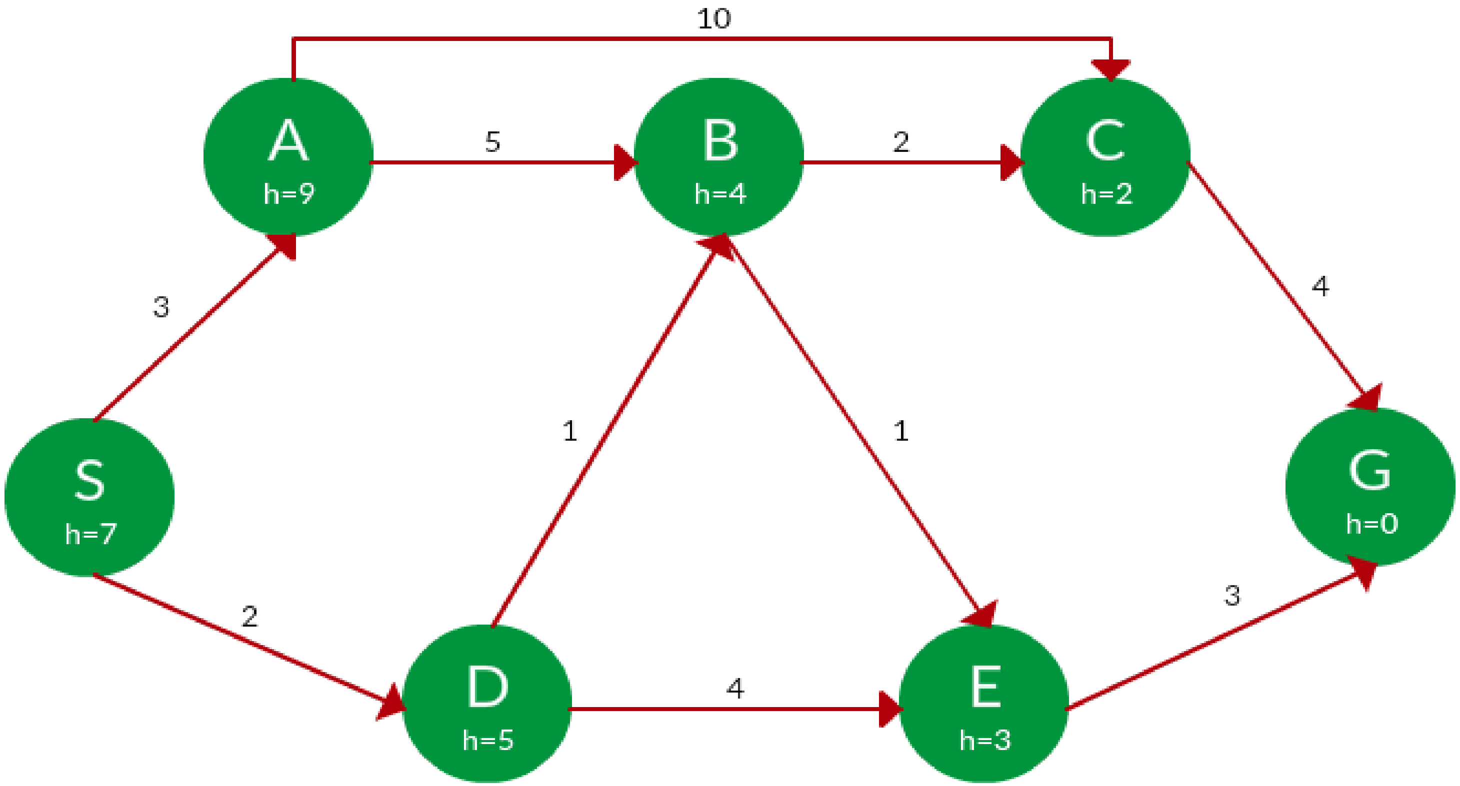
- The following points should be noted w.r.t heuristic
- $f(x) = g(x) + h(x)$
- Here, ***h(x)*** is called the **forward cost**, and is an estimate of the distance of the current node from the goal node.
- And, ***g(x)*** is called the **backward cost**, and is the cumulative cost of a node from the root node.

Strategy

- Choose the node with lowest f(x) value.

Example

- Find the path to reach from S to G using A* search



Solution

- Starting from S, the algorithm computes g(x) + h(x) for all nodes in the fringe at each step, choosing the node with the lowest sum.
- The entire working is shown in the table ***(next slide)***
- Note that in the fourth set of iteration, we get two paths with equal summed cost f(x), so we expand them both in the next set.
- The path with lower cost on further expansion is the chosen path.

PATH	H(X)	G(X)	F(X)
S	7	0	7
S -> A	9	3	12
S -> D ✓	5	2	7
S -> D -> B ✓	4	2 + 1 = 3	7
S -> D -> E	3	2 + 4 = 6	9
S -> D -> B -> C ✓	2	3 + 2 = 5	7
S -> D -> B -> E ✓	3	3 + 1 = 4	7
S -> D -> B -> C -> G	0	5 + 4 = 9	9
S -> D -> B -> E -> G ✓	0	4 + 3 = 7	7

Path: S -> D -> B -> E -> G

Continued…

Path: S -> D -> B -> E -> G

Cost: 7

Continued…

A* Graph Search

- A* tree search works well, except that it takes time re-exploring the branches it has already explored.
- In other words, if the same node has expanded twice in different branches of the search tree.
- A* search might explore both of those branches, thus wasting time
- A* Graph Search, or simply Graph Search, removes this limitation by adding this rule: **do not expand the same node more than once.**

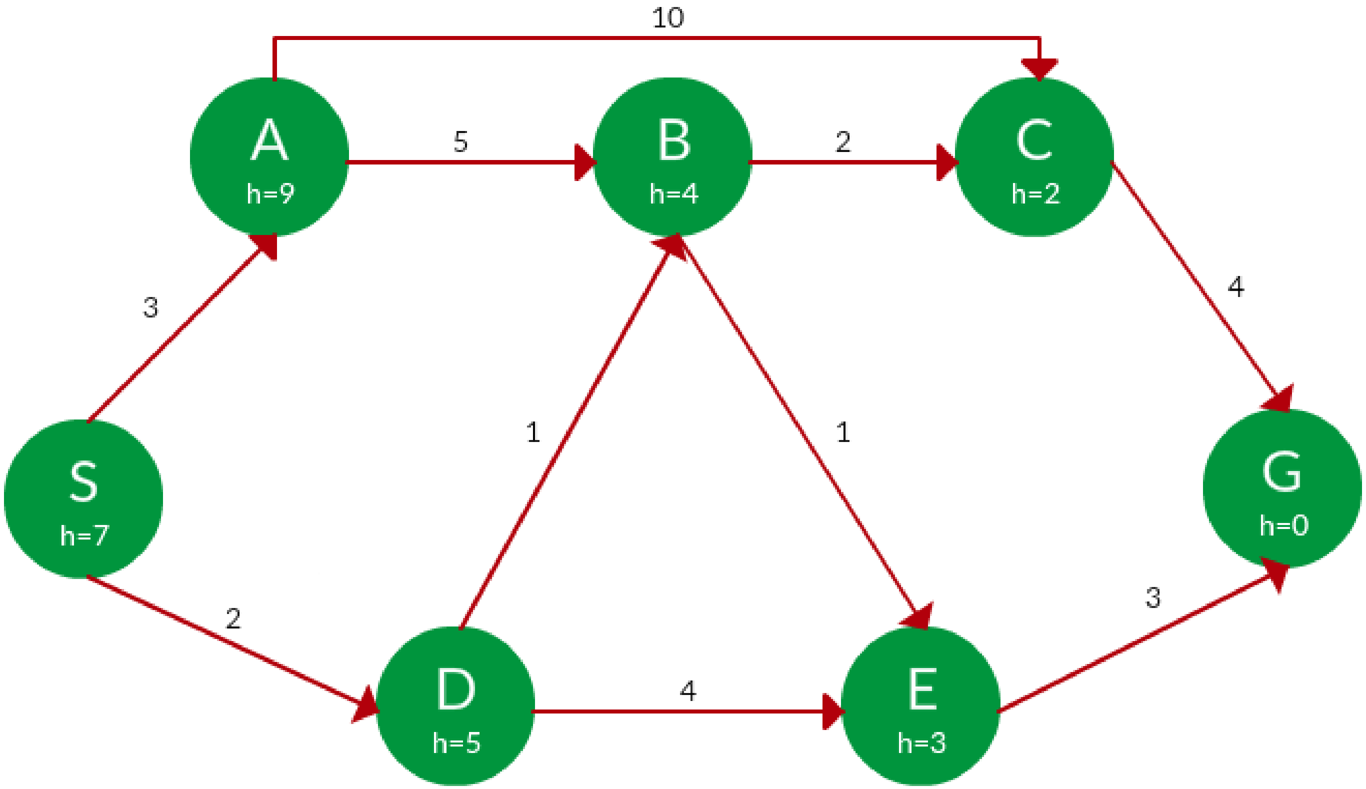
Continued…

Heuristic

- Graph search is optimal only when the forward cost between two successive nodes A and B, given by h(A) - h (B) , is less than or equal to the backward cost between those two nodes g(A -> B).
- This property of graph search heuristic is called **consistency**.
- Consistency: $h(A) - h(B) \leq g(A \rightarrow B)$

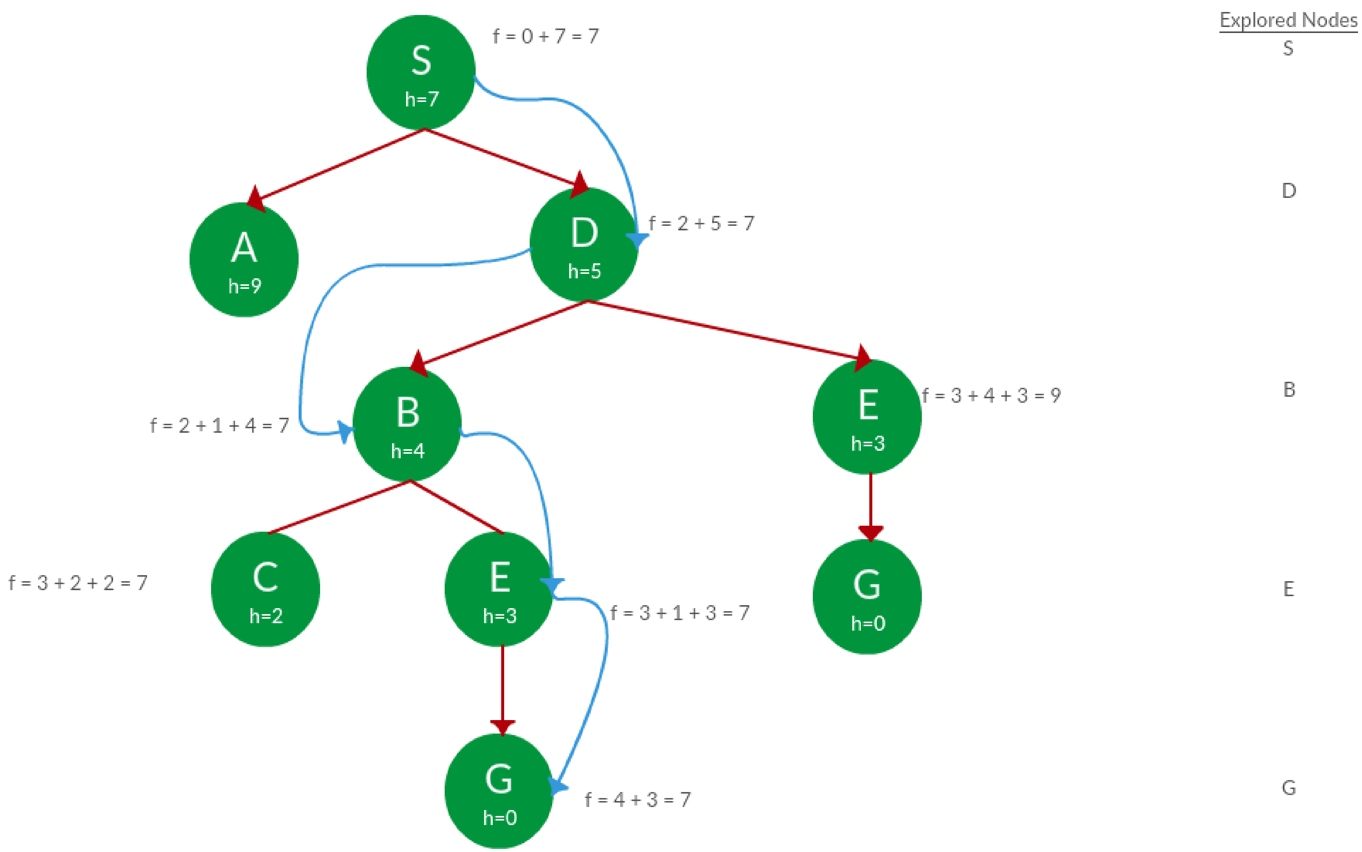
Example

- Use graph search to find path from S to G in the following graph.



Solution

- We solve this question pretty much the same way we solved last question, but in this case, we keep a track of nodes explored so that we don't re-explore them.



Continued…

Path: S -> D -> B -> C -> E -> G

Cost: 7