A* ALGORITHM

Page No.:
Date / /

* A* Algorithm is an informed search algorithm.
* It is one of the best and popular techniques used for path finding and graph traversals.
* A lot of games and web based maps use this algorithm for finding the shortest path efficiently.
* It is essentially a Best First Search Algorithm

WORKING

* It maintains a tree of paths originating at the start node.
* It extends those paths one edge at a time.
* It continues until its terminating criterion is satisfied.

A* Algorithm extends the path that minimizes the following function:

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

Here,

- 'n' is the last node on the path
- g(n) is the cost of the path from start node to node 'n'
- h(n) is the heuristic function that estimates cost of the cheapest path from node 'n' to the goal node.

Page No. :
Date : / /

ALGORITHM

* The implementation of A* Algorithm involves maintaining two lists - OPEN & CLOSED
* OPEN contains those nodes that have been evaluated by the heuristic function but have not been expanded into successors yet.
* CLOSED contains those nodes that have already been visited.

Step 1:

- Define a list OPEN
- Initially, OPEN consists solely of a single node the start nodes.

Step 2:

If the list is empty, return failure and exit.

Step 3:

- Remove node 'n' with the smallest value of f(n) from OPEN and move it to list CLOSED.
- If node n is a goal state, return success and exit.

Step 4:

Expand node n.

Page No.:
Date : / /

Step 5:

- If any successor to n is the goal node, return success and the solution by tracing the path from goal node to S.
- Otherwise, go to step 6.

Step 6:

For each successor node,

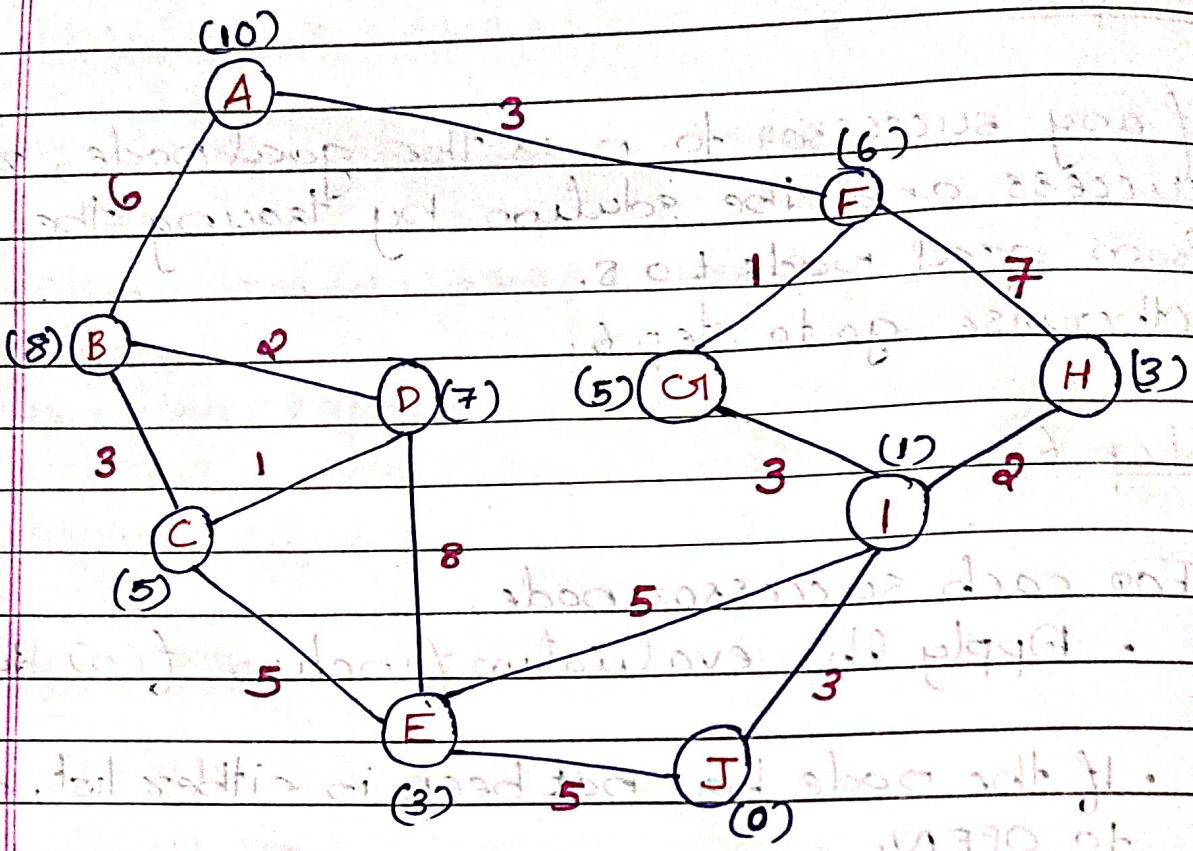
- Apply the evaluation function f(n) to the node.
- If the node has not been in either list, add it to OPEN.

Step 7:

Go back to step 2.

Page No. :
Date / /

EXAMPLE



The numbers written on edges represent the distance between the nodes
The number written on nodes represent the heuristic value. Find the most cost-effective path to reach from start state A to final state J using A* Algorithm.

Solution :

Step 1 :

- Start with node A.
- Node B & node F can be reached from A

A* Algorithm calculates f(B) and f(F)

$f(B) = 6+8 = 14$ $[g(n)+h(n)]$
$f(F) = 3+6 = 9$

since $f(F) < f(B)$. So it is decided to go to node F.

Path $A \rightarrow F$

Step 2:

Node G1 and H can be reached from node F.

A* Algorithm calculates $f(G1)$ and $f(H)$

- $f(G1) = (3+1)+5 = 9$ $(g(F) + g(G1) + h(G1))$
- $f(H) = (3+7)+3 = 13$

since $f(G1) < f(H)$. So it decided to go to node G1

Path $A \rightarrow F \rightarrow G1$

Step 3:

Node I can be reached from node G1.

$f(I) = (3+1+3)+1 = 8$

It decides to go to node I

Path- $A \rightarrow F \rightarrow G1 \rightarrow I$

Step 4:

Node E, Node H & J can be reached from node I.

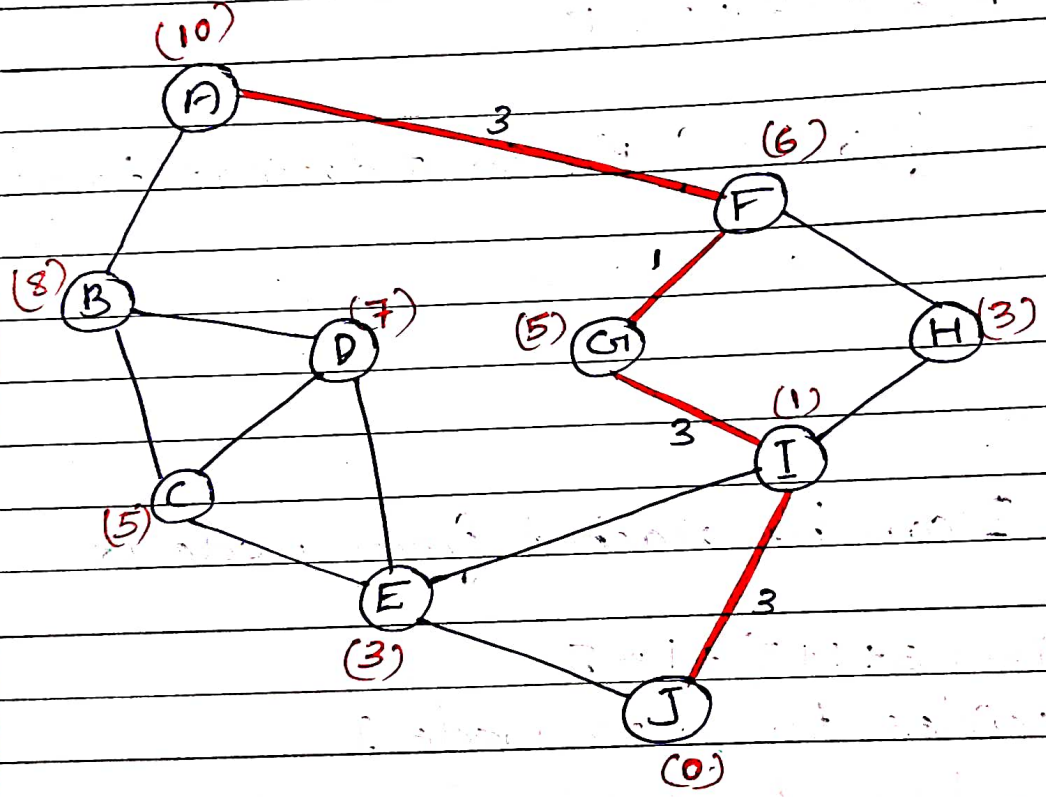
A* Algorithm calculates $f(E)$, $f(H)$ & $f(J)$

- $f(E) = (3+1+3+5) + 3 = 15$
- $f(H) = (3+1+3+0) + 3 = 10$
- $f(J) = (3+1+3+3) + 0 = 10$

Since $f(J)$ is least, so it decided to go to node J.

Path:- A → F → G1 → I → J

This is the required shortest path from node A to J



A* Algorithm is one of the best path finding algorithms. But it does not produce the shortest path always. This is because it heavily depends on heuristics.