AMERICAN INTERNATIONAL UNIVERSITY – BANGLADESH

ID	Name
19-40895-2	LINKON,ANKON SARKER

Section: [M]

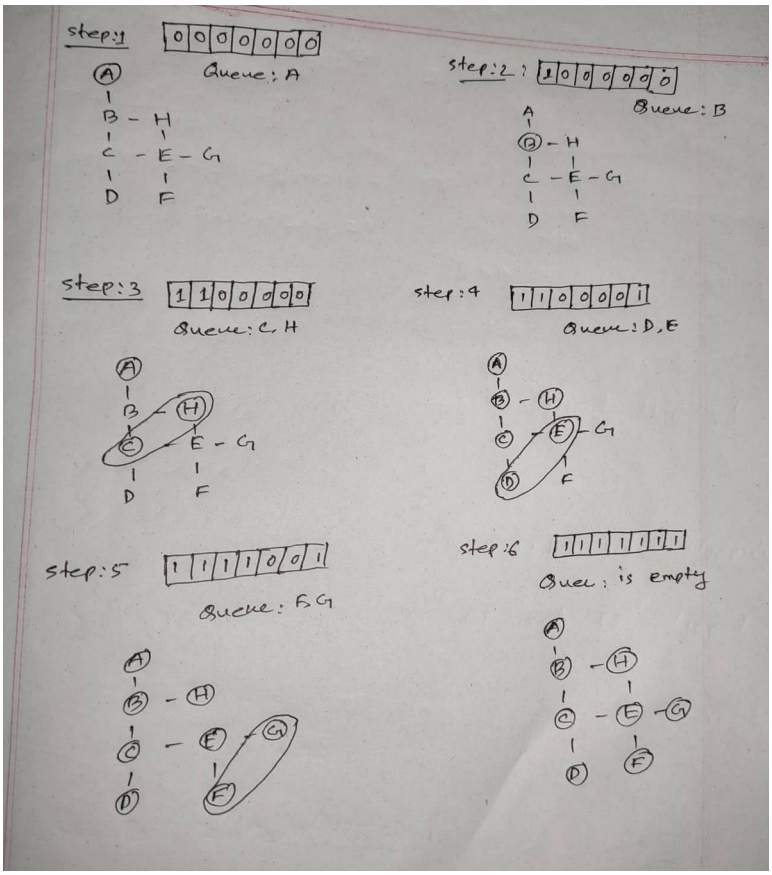
Course Title: Artificial Intelligence & Expert System

Midterm Assignment

Q1. Describe and differentiate BFS, DFS, and UCS with an example. Show necessary simulations.

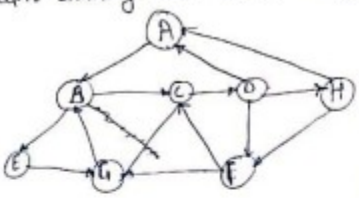
Answer:

BFS: Breadth first search is aa algorithm for searching a tree data structure for a node that satisfies a given property. It starts at the tree root and explores all nodes at the present depth prior to moving on to the nodes at the next depth level. Extra memory, usually a queue, is needed to keep track of the child nodes that were encountered but not yet explored.



DFS: Depth-first search (DFS) is an algorithm for traversing or searching tree or graph data structures. The algorithm starts at the root node (selecting some arbitrary node as the root node in the case of a graph) and explores as far as possible along each branch before backtracking.

Example :-
Consider, graph G along adjacency lists, Calculate the order to print all the nodes of graph starting from A by using (DFS).



Adjacency L
A : B
B : C, E
C : D, G
D : A, H
E : G
F : C
G : B, F

Solution:-
Push A onto the Stack
① Stack : A
Pop the top element of the stack A & print it. Push all the neighbours of A onto stack that are in ready state.
② print A
Stack : B
Pop the top element of the stack i.e B & push all the neighbours of B into stack.
③ print B
Stack : C, E
Same do all the steps
④ print E
Stack : G, C
⑤ print G
Stack : C
⑥ print C
Stack : D
⑦ print D
Stack : E, H
⑧ print H
Stack : F
⑨ print F
Stack ← stack now becomes empty.

UCS:

(Uniform-Cost Search) expands the node n with the lowest path cost g(n). This is done by storing the frontier as a priority queue ordered by g.Uniform-cost search does not care about the number of steps a path has, but only about their total cost. Therefore, it will get stuck in an infinite loop if there is a path with an infinite sequence.

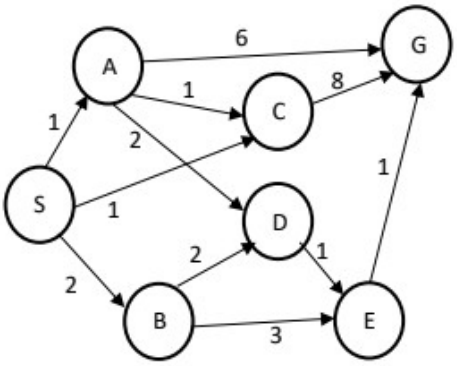
Differentiation:

S.NO	BFS	DFS	UCS
1	BFS is used to find vertex closer to source.	DFS is used to find vertex that are far away from source.	UCS is used to find solutions with less cost from source to destination.

2	**Breadth-first search** expands the shallowest nodes first; it is complete, optimal for unit step costs, but has exponential space complexity.	**Depth-first search** expands the deepest unexpanded node first. It is neither complete nor optimal, but has linear space complexity.	**Uniform-cost search** expands the node with lowest path cost, g(n), and is optimal for general step costs.

Q2.

Consider the following search problem, represented as a graph. The start state is S and the only goal state is G. The heuristic values are given in the table below.



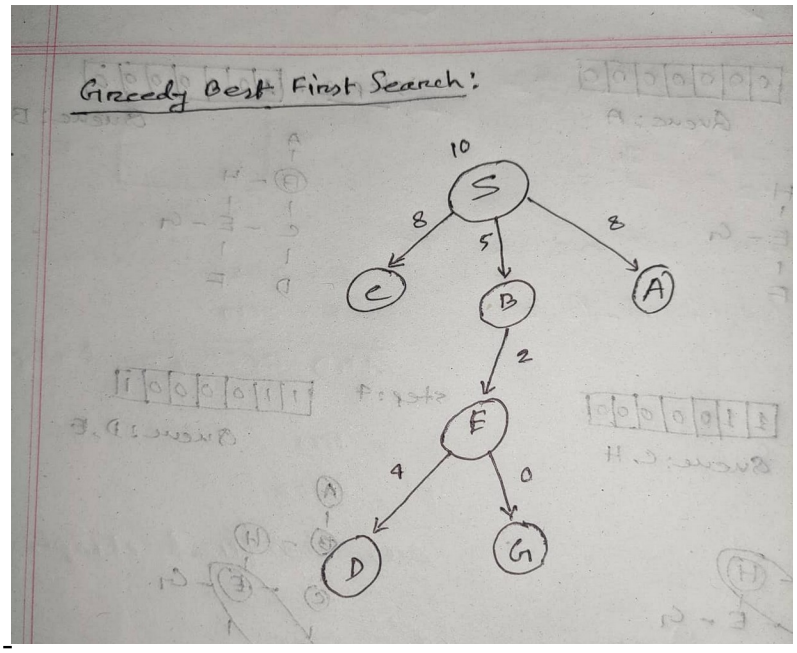
S	A	B	C	D	E	G
10	8	5	5	4	2	0

Answer:

Here, Start State= S

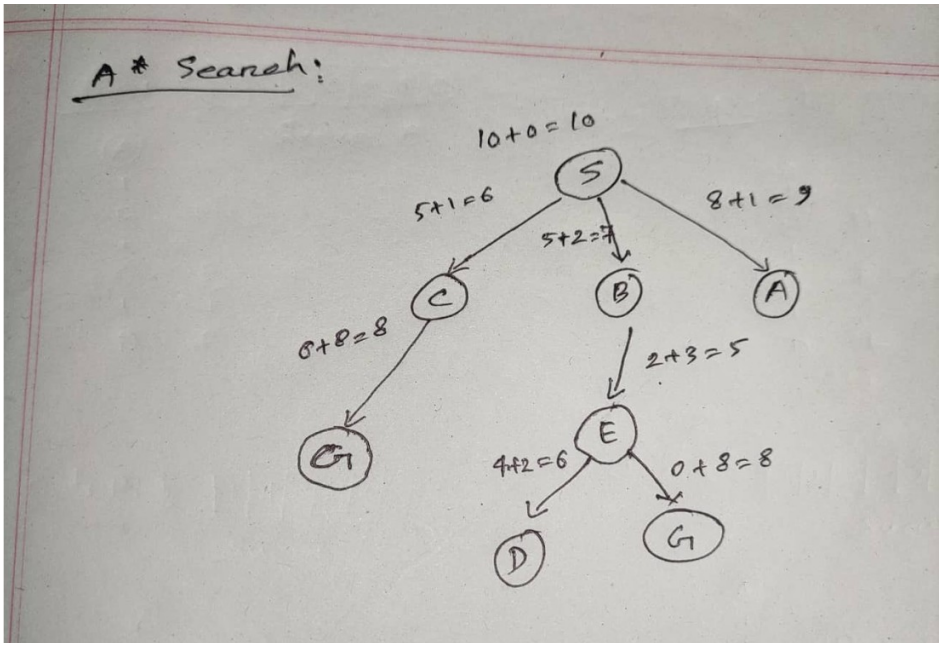
Goal State= G

Greedy Best First Search:



Output: S, B, E, G

A* Search:



Output: S, B, C, E, G

In figure 01 two nodes(S,B,E) are explored to reach the goal state ‘G’.

In figure 02 A* search three nodes (S,B,C,E) are explored to reach the goal state ‘G. So in this graph greedy best first search is faster

In the above figure the greedy best first search chose S -B-E-G but A* search algorithm chose S-B-C-E-G .here the cost of the path given by best first search is lower than the cost of the path of A* search.So greedy Best first search is faster as the cost of path and it is lower than the cost of the path of A* search.

Greedy Best First only uses heuristic function to evaluate a node but A* search calculates both path cost & heuristic value.

Optimality of Greedy Best First and A* search with respect to the above search problem:

Here, the tree-search version of A∗ is optimal because h(n) is admissible, while the graph-search version is optimal if h(n) is consistent.But greedy best first search, the main concern is in finding path with minimum cost. So the discovered path may not be optimal.