Prakhar Pande
DAA Tutorial 5
Sec-D, 2016906, 28

PAGE ____
DATE ____

tutorial - 5

Sol	BFS	DFS
=)	BFS, stands for Bredth First Search	DFS, stands for Depth First Search.
=)	BFS uses queue to find the shortest path	DFS uses Stack to find the shortest path.
=)	BFS is better when target is closer to source.	DFS is more suitable for decision tree. As with one decision we need to transfer further to argument the dec
=)	~~Slower~~ BFS is slower than DFS	DFS is faster than BFS.
=)	TC of BFS = O(V+E) where V is vertices	TC of DFS is also O(V+E) where V is vertices of

Application of DFS:-

Q3

If we perform DFS on unweighted graph then it will create minimum spanning tree for all pair shortest path tree.

=) We can detect cycles in a graph using DFS. If we get one back-edge during BFS, then there must be one cycle. Q4

Applications of BFS

=) Like DFS, BFS may also used for detecting cycles in a graph.

=> Finding shortest path & minimal spanning tree in unweighted graph.

=) In building the index by search engines crawlers.

Q2 BFS uses queue data structure for finding shortest path.

DFS uses stack data structure.

DFS algorithm traverses a graph in a depthward motion & uses a stack to remember to get the next vertex to start a search, when a dead end occurs in any iteration.

Q3 Sparse Graphs : A graph in which number of edges is much less than the possible number of edges.

Dense Graph : is one in which the number of edges is close to the maximal number of edges.

Q4 The existence of a cycle in directed & undirected graph finds an edge that points to an ancestors of current vertex. All the back edges which DFS skips are part of cycles.

Detect Cycle in directed Graph
DFS can be used to detect a cycle in a graph DFS is for a connected graph produce a tree.

For a disconnected graph, the DFS form as output to detect cycles. check for a cycle in individual trees.

Detect cycle Undirected Graph:
Run a DFS from every unvisited node. DFS can be used to detect a cycle in a graph. There is a

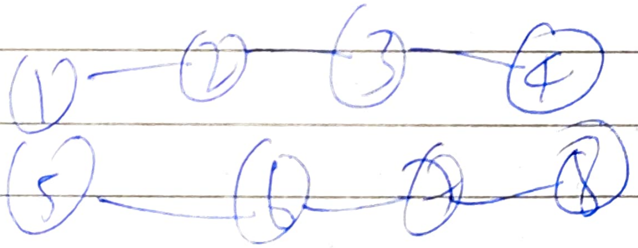
cycle in a graph only if there is a back edge present in graph.

Q.7 : Disjoint set Data Structure

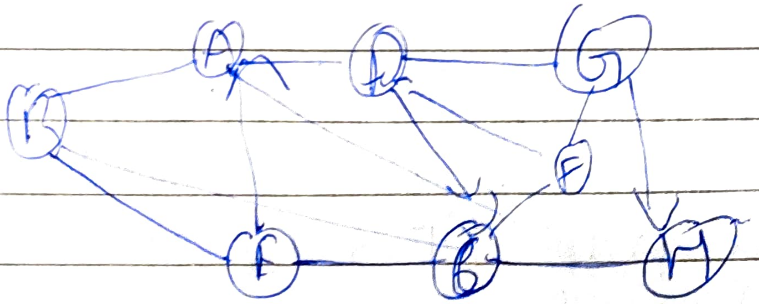
It allows to find out whether the two elements are in same set or not efficiently.

Ex: $S_1 = \{1, 2, 3, 4\}$

$S_2 = \{5, 6, 7, 8\}$



Q6

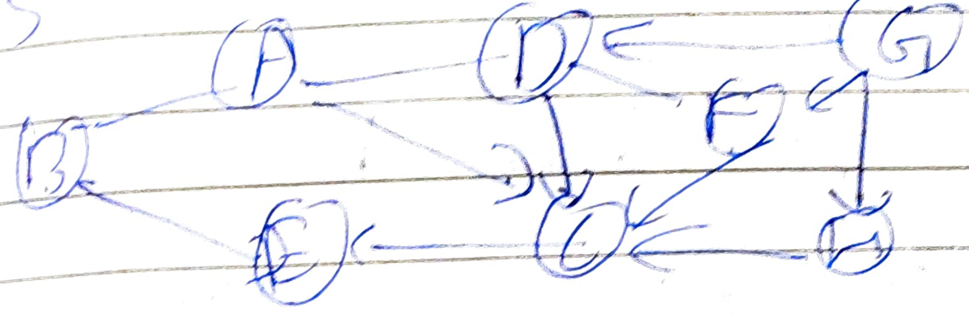


Node: B E C A D E

Parent : B B E A D

Path : B → E → A → D → F

DFS

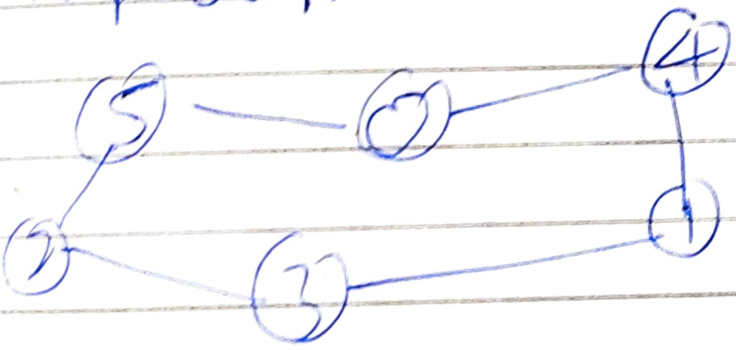


Node Proces: B B C E A D F

Stack: B CE EE AE DEFE

Path: B -> C -> E -> A -> D -> F E

Sol 8 Topological Sort



Adjacency List 0 ->

1 -> 1 ->

2 -> 3

3 -> 1

4 -> 0, 1

5 -> 2, 0

Visita

0	1	2	3	4
false	false	false	false	false

Stack.

S-1 List EmpThy. No recursion.

Stack [0]

S-2 Topo Sort (1) visited[1] = Tru

Stack [0 1]

S-3 Topological Sort(2), visited [2]=true

Stack [0|1|1|3|2 ...

S-5 Topo sort (5), visit[5] = true.
Stack [0|1|1|3|2|4|5

S-6 Print all the elements.